Homework 2 Due Thursday @ 11:59PM on Gradescope
Project Proposals due Tonight

Advanced Cryptography
CS 655

Week 7:

- Constructing Depth-Robust Graphs
- Sustained Space Complexity
- Bandwidth Hard-Functions

Spring 2023

Course Project Proposal

- Due Tonight by e-mail (jblocki@purdue.edu)

- Project Proposal: 2 Pages
  - Briefly the problem you plan to work on
  - Briefly summarize prior work on the problem and how your project is different
  - Identify several related papers that you plan to read as part of the project
  - Briefly describe your plan to attack the problem

A Few Project Ideas

- Pick a cryptographic scheme and try to find a tighter concrete security proof under idealized assumptions
  - Example: Tighter security analysis for Password Authenticated Key Exchange (PAKE) protocols such as CPACE in the generic group+random oracle model?
- Pick a cryptographic scheme/protocol and analyze the security with respect pre-processing attacks or provide a memory-tight reduction
  - **Example:** Memory-Tight Reduction for RSA-FDH under the One-More-RSA-Inversion problem?
  - **Example:** Security of PAKE protocols against pre-processing attacks?
  - **Example:** Security of AES-GCM vs pre-processing attacks?
- Pebbling Reduction for Salted iMHFs vs. Preprocessing Attackers
- Pebbling Reduction for Argon2 Round Function (in ideal permutation model)

A Few Project Ideas

- Implement a Cryptographic Protocol/Attack
  - **Example:** Implement Argon2 with different instantiations of round function
  - **Example:** Implement partitioning oracle attack on AES-GCM.
- Many other possibilities! Make sure your proposal is realistic.

- It is ok to try something and fail i.e., a final project report documenting your unsuccessful attempts to solve a problem is acceptable as long as the attempts are clearly described

Recap: iMHFs

- **Graph Pebbling Reduction [AS15]:** Complexity of iMHF $f_{G,H}$ is fully captured by pebbling cost of DAG G.

- **Informal Theorem [AS15]:** Any algorithm A evaluating $f_{G,H}$ in the parallel random oracle model has $CMC(A) \approx \lambda \times CC(G)$ where $H(x) \in \{0,1\}^{\lambda}$

- **Proof Sketch:** Use execution trace from A to extract a legal pebbling of G such that for all rounds i we have $|P_i| \approx |\sigma_i|/\lambda$

#pebbles at time i

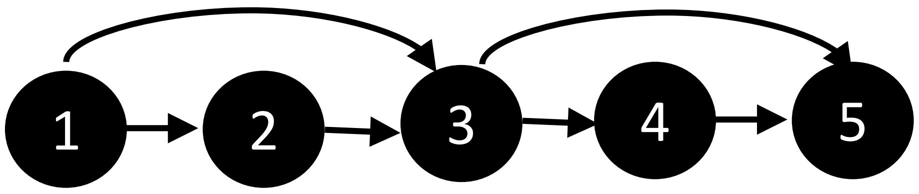
#bits in A's memory at time i

Recap: 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) $\leq \mathrm{d}$.

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

Example: (1,2)-reducible

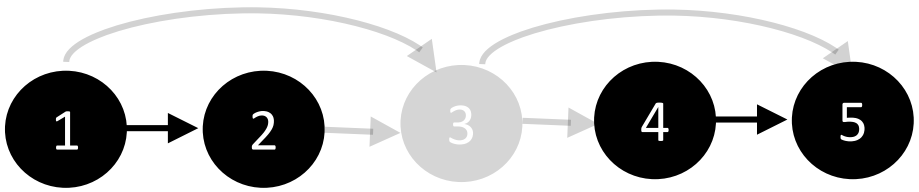


Recap: Depth Robustness

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) $\leq \mathrm{d}$.

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

Example: (1,2)-reducible



Recap: Depth-Robustness is Sufficient! [ABP17]

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

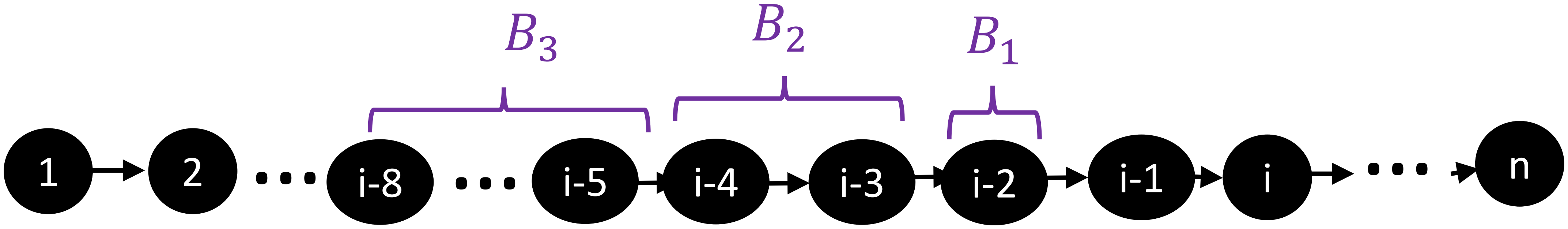
Implications: There exists a constant indegree graph G with

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

[AB16]: We cannot do better (in an asymptotic sense)

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

DRSample



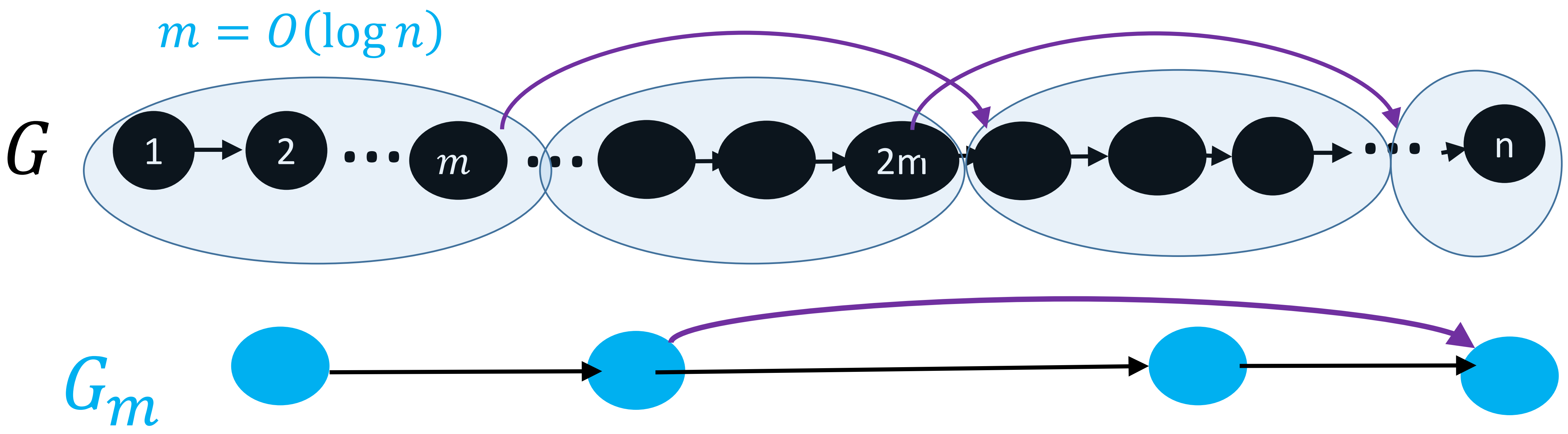
Indegree: $\delta = 2$

Key Modification to Argon2i: New distribution for r(i)

Buckets:$B_1, \ldots, B_{\log i}$

$$B_j = \left[i - 2^j, i - 2^{j-1} - 1\right]$$

DR-Sample: Meta-Graph



Each meta-node u corresponds to m nodes $u_1, \ldots, u_m$.
Let $F_u = \{u_1, \ldots, u_{m/3}\}$ and $L_u = \{u_{m-\frac{m}{3}+1}, \ldots, u_m\}$ denote the first (resp. last third) of these nodes

G_m has edge (u,v) if and only if G has some edge (x,y) with x ∈ L_u and y ∈ L_u

Recap: DRSample Analysis

- Let G be the DRSample graph. Define Meta-Graph G_m with $\mathrm{m} = \Omega(\log N)$ and $\mathrm{N}' = \Omega\left(\frac{N}{m}\right)$

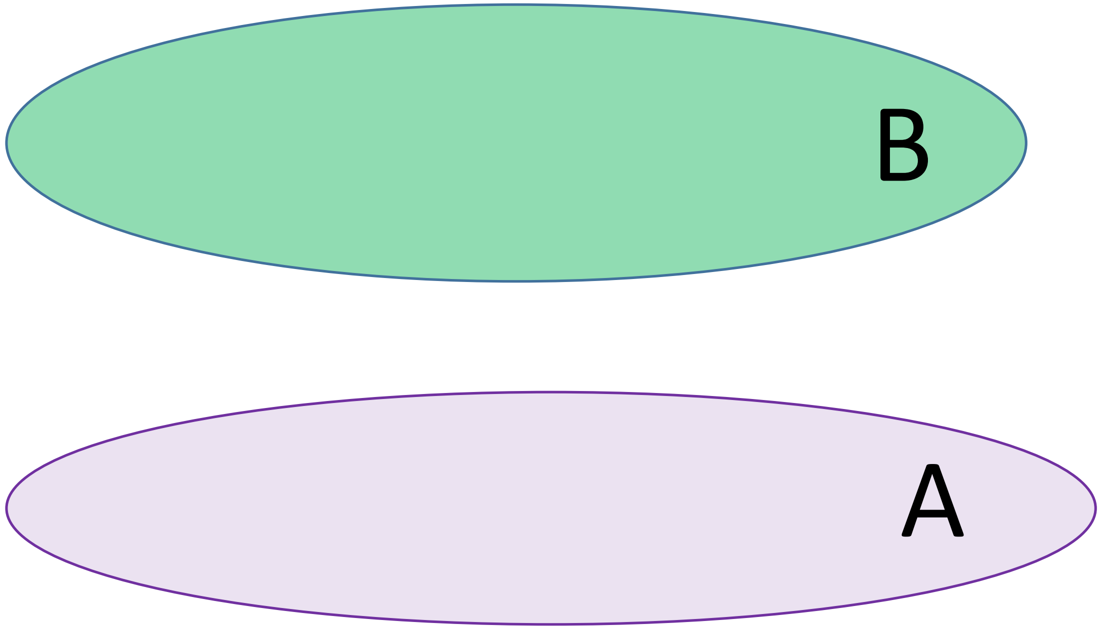
Last Class: We assumed that G_m was a δ −local expander and proved that any δ −local expander with $\mathrm{N}' = \Omega\left(\frac{N}{m}\right)$ nodes is $\left(\Omega(N'), \Omega(N')\right)$ depth-robust

- Meta-Graph G_m is $\left(\Omega\left(\frac{N}{m}\right), \Omega\left(\frac{N}{m}\right)\right)$ −depth-robust with $\mathrm{m} = \Omega(\log N)$

➔DRSample G is $\left(\Omega\left(\frac{N}{m}\right), \Omega(N)\right)$ −depth-robust

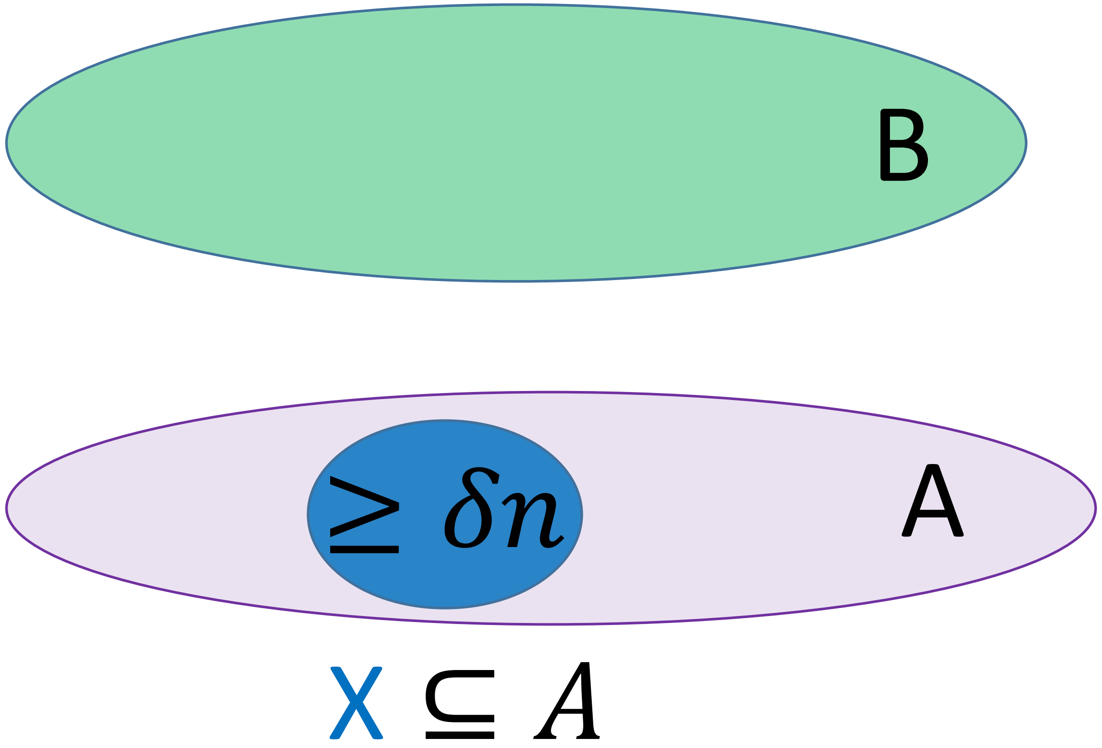
TODO: Prove that G_m is a δ −local expander * (*almost)

δ –bipartite expander



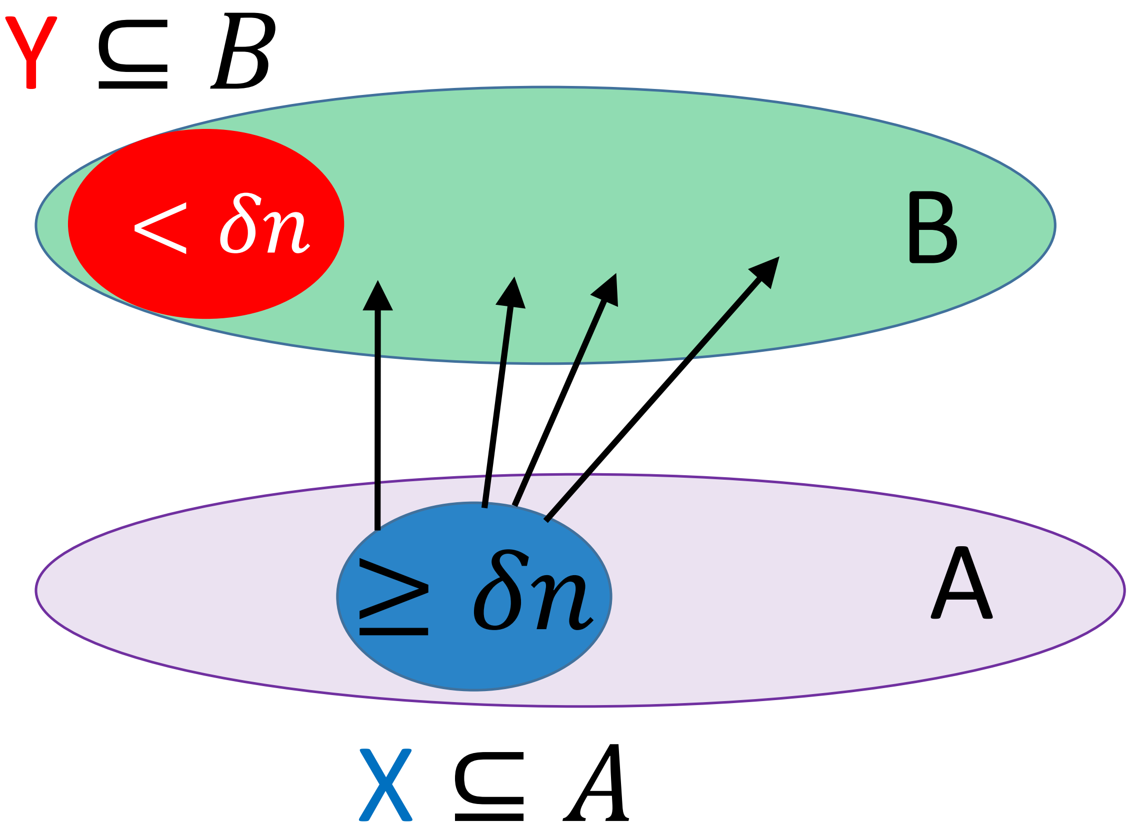
$$|A| = |B| = n$$

δ −bipartite expander



$$|A| = |B| = n$$

δ −bipartite expander

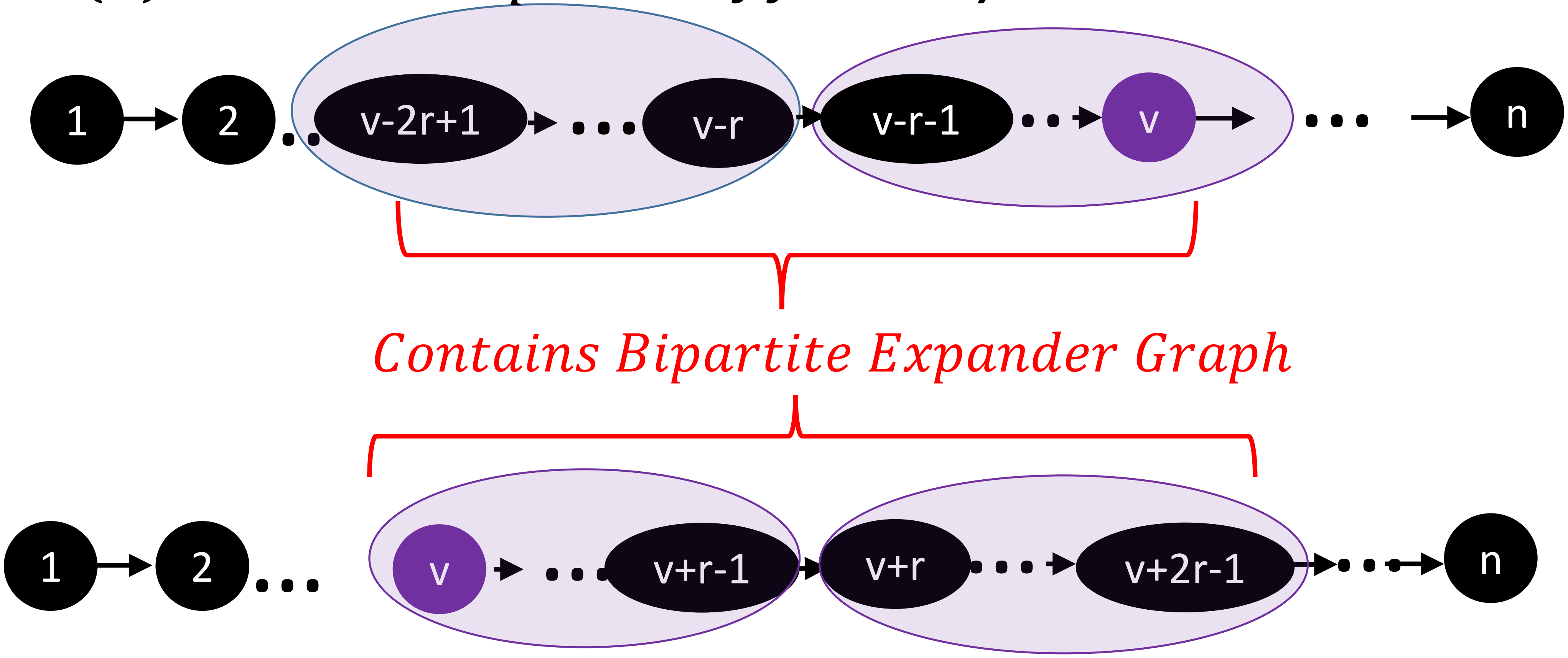


$$|A| = |B| = n$$

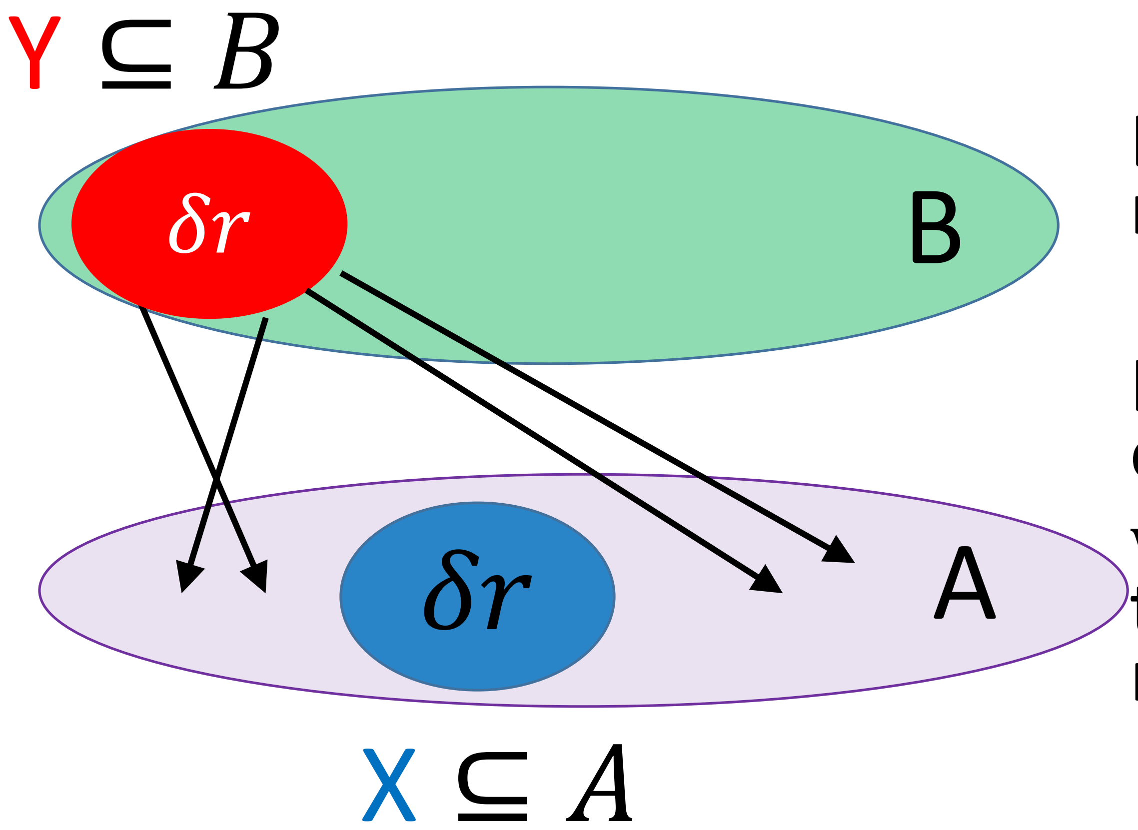
Y $\subseteq B$
(unreachable from X)

(δ)-local expander around v

We have $(\delta) - \textit{local exapnsion if for every}\ \mathrm{r}$



Not δ −bipartite expander?

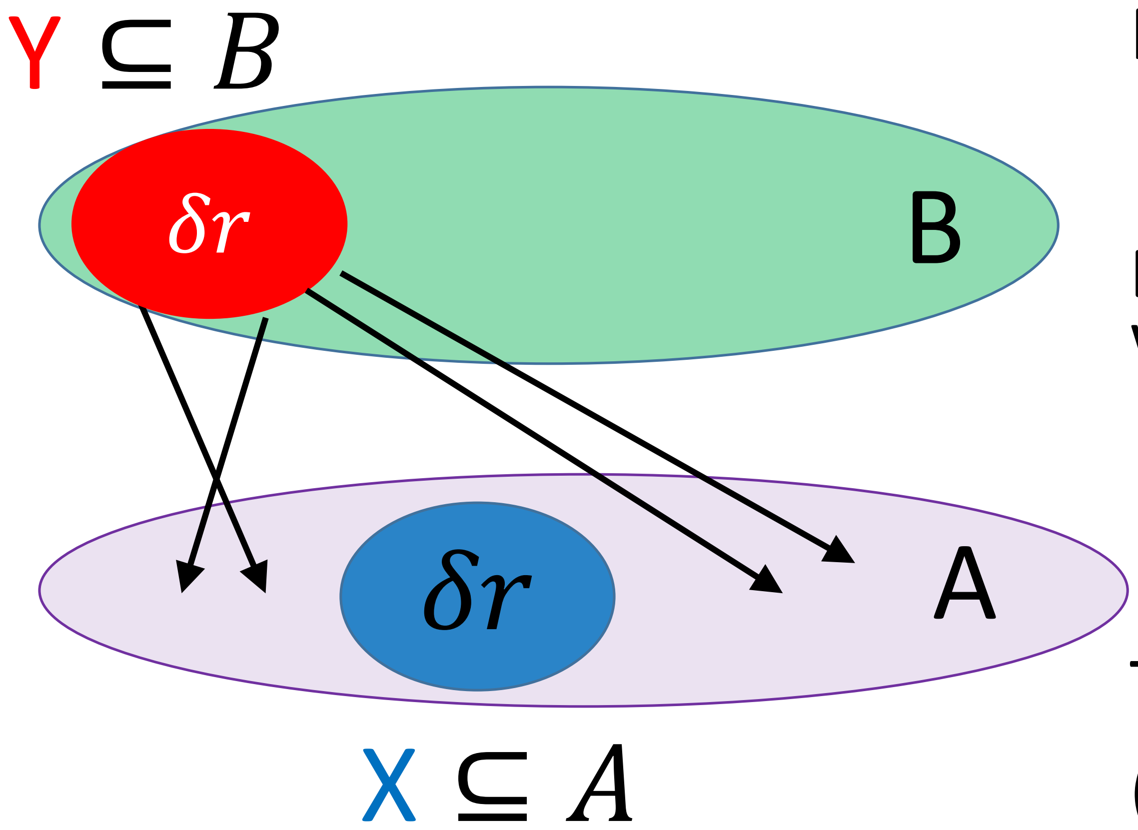


$|A| = |B| = r$

Let A,B be a set of 2r consecutive nodes in meta-graph.

If not δ −bipartite then there exists Y $\subseteq B$ and X $\subseteq A$ with size |Y|=δr and |X|=δr such that none of the edges from any meta-node in Y hit any node in X

Not δ −bipartite expander?



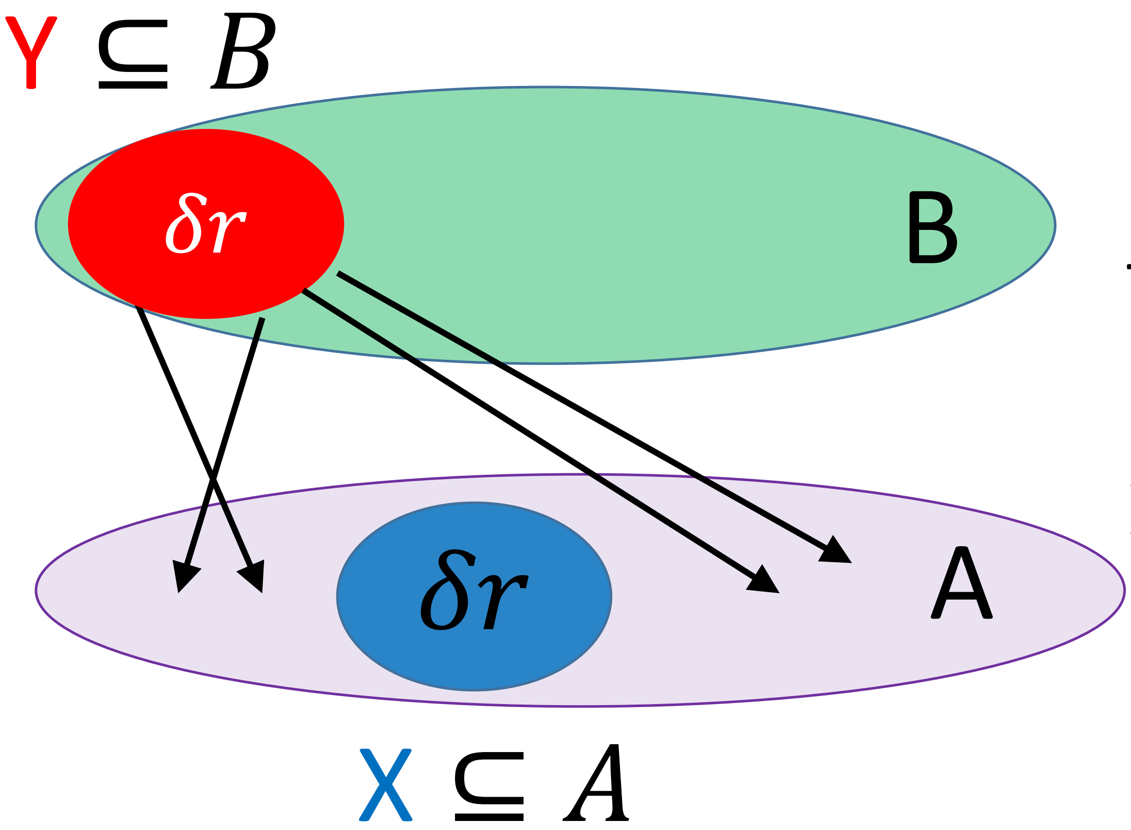
Fix some subsets |Y|=δr and |X|=δr

Each individual edge from Y hits X with probability

$$\approx \frac{\delta}{3\log n}$$

There are $\frac{m}{3} \times \delta r$ edges (all picked independently)

Not δ −bipartite expander?



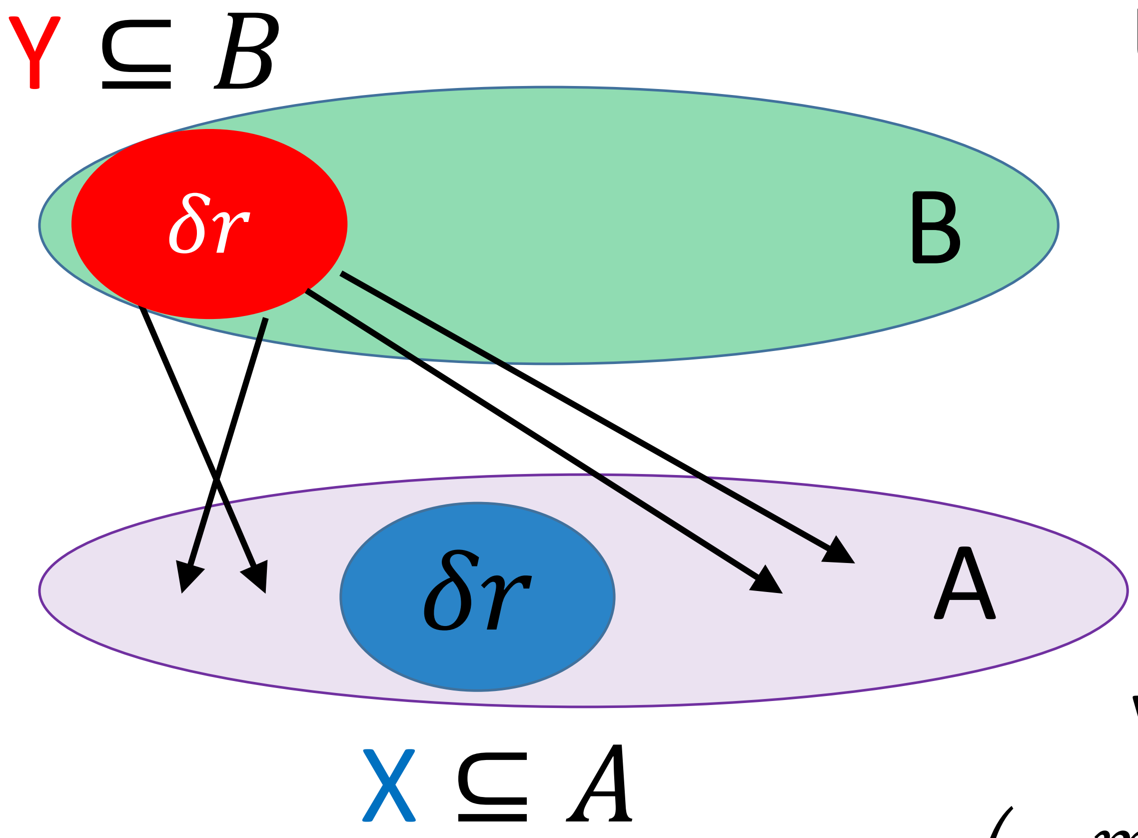
Each individual edge from Y hits X with probability $\approx \frac{\delta}{3 \log n}$

There are $\frac{m}{3} \times \delta r$ edges (independent)

$$\Pr[\text{Y Misses } X] \le \left(1 - \frac{\delta}{3 \log n}\right)^{\frac{m}{3} \times \delta r}$$

$$\le e^{-r\delta^2\left(\frac{m}{9 \log N}\right)}$$

Not δ −bipartite expander?



Union Bound:

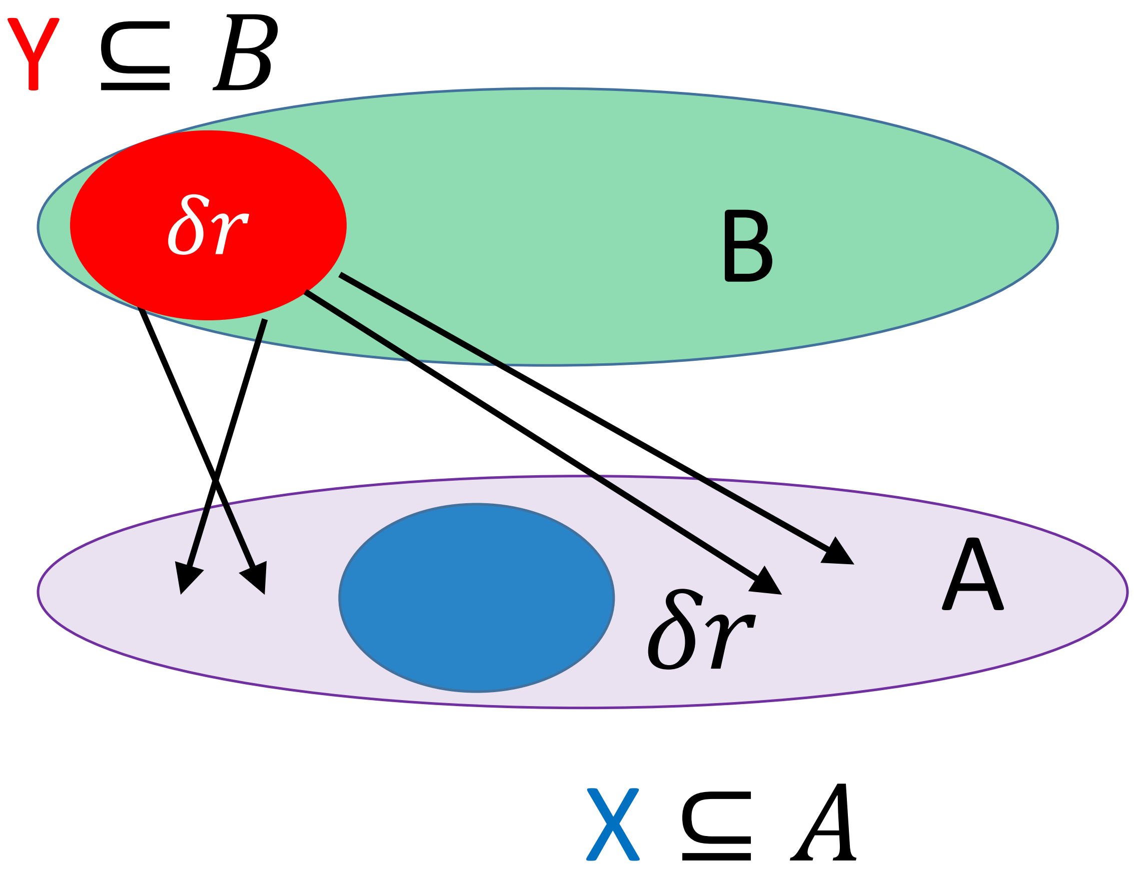
$$\Pr[\exists X, \mathrm{Y} \text{ s.t. } \mathrm{Y} \text{ Misses } X] \le e^{-r\delta^2\left(\frac{m}{9\log N}\right)} \binom{r}{\delta r}^2$$

$$\le \exp(z)$$

With

$$z = -r\delta^2\left(\frac{m}{9\log N}\right) + 2\delta r \ln\left(\frac{1}{\delta}\right) + 2r\ln\left(\frac{1}{1-\delta}\right)$$

Not δ −bipartite expander?



Union Bound:

$$\Pr[\exists X, Y \text{ s.t. } Y \text{ Misses } X]$$
$$\leq \exp(z)$$
$$\leq \exp(-2r)$$

Pick $m \geq$
$\left(18\delta^{-1}\ln\left(\frac{1}{\delta}\right) + 18\delta^{-2}\left(1 + \ln\left(\frac{1}{1-\delta}\right)\right)\right) \log N$

$$z = -r\delta^2\left(\frac{m}{9\log N}\right) + 2\delta r \ln\left(\frac{1}{\delta}\right) + 2r\ln\left(\frac{1}{1-\delta}\right)$$

Second Union Bound?

- Fixing any A=[u,…,u+r-1] and B=[u+r,…,u+2r-1] we say that A,B are connected with bipartite expander with probability at least $1 - \exp(-2r)$

- **Ideal:** Want to show that $G_m \; is \; a \; \delta$ −local expander i.e.., this holds for all u and all r
  - Union bound over all meta-nodes u and all r?
  - We can union bound over all $r \geq \log N$ and all u since

$$\sum_{u} \sum_{r \geq \log N} \exp(-2r) \leq N \sum_{r \geq \log N} \exp(-2r) \ll \frac{2}{N}$$

Second Union Bound?

- **Fix: Let $\boldsymbol{B_u}$ be the event that for some $r < \log N$ we do not have an expander between** A=[u,…,u+r-1] and B=[u+r,…,u+2r-1]

- **Key Idea:** Use concentration bounds to argue that $\sum_u \boldsymbol{B_u} \leq \varepsilon N$ with high probability (for some suitably small ε)

➔For at least $\mathrm{N} - \varepsilon N$ meta-nodes u we do have local expansion around u.

➔This is sufficient to argue that meta-graph is depth-robust.

Second Union Bound?

- **Fix: Let $\boldsymbol{B_u}$ be the event that for some $r < \log N$ we do not have an expander between** A=[u,…,u+r-1] and B=[u+r,…,u+2r-1]
- **Key Idea:** Use concentration bounds to argue that $\sum_u \boldsymbol{B_u} \leq \varepsilon N$ with high probability (for some suitably small ε)

Problem? $\boldsymbol{B_u}$ and $\boldsymbol{B_{u+1}}$ are not independent!

But, $\boldsymbol{B_u}$ and $\boldsymbol{B_v}$ are independent if $\boldsymbol{u - v \geq log\ N}$

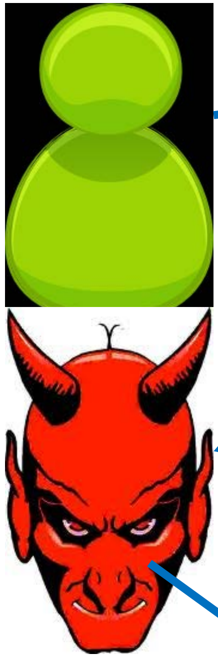
Solution: Partition random variables into $\boldsymbol{log\ N}$ buckets such that random variables in each bucket are independent. Apply concentration bounds to each bucket.

Sustained Space Complexity

Joël Alwen (IST Austria/Wickr)
Jeremiah Blocki (Purdue)
Krzysztof Pietrzak (IST Austria)

Institute of Science and Technology

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.

LastPass ****

SONY

ebay

AdultFriendFinder

Ashley Madison
Life is short. Have an affair.

LinkedIn

Dropbox

Zappos.com
the web's most popular shoe store!

rockyou

YAHOO!

Adobe

GAWKER

livingsocial

Offline Attacks: A Common Problem

- Password breaches at major companies have affected ~~millions~~ **billions**

TECH

Yahoo Triples Estimate of Breached Accounts to 3 Billion

Company disclosed late last year that 2013 hack exposed private information of over 1 billion users

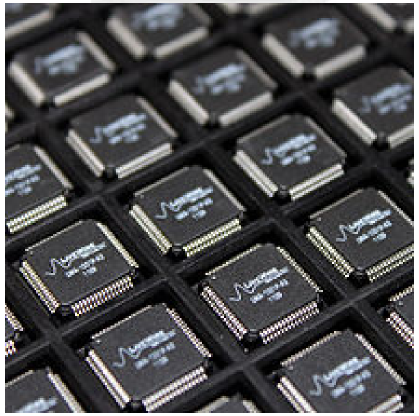
By Robert McMillan and Ryan Knutson

Updated Oct. 3, 2017 9:23 p.m. ET

A massive data breach at Yahoo in 2013 was far more extensive than previously disclosed, affecting all of its 3 billion user accounts, new parent company Verizon Communications Inc. said on Tuesday.

The figure, which Verizon said was based on new information, is three times the 1 billion accounts Yahoo said were affected when it first disclosed the breach in December 2016. The new disclosure, four months after Verizon completed its acquisition of Yahoo, shows that executives are still coming to grips with the extent of the...

Goal: Moderately Expensive Hash Function



Fast on PC and
Expensive on ASIC?

password
hashing
competition

(2013-2015)

https://password-hashing.net/

password
hashing
competition

(2013-2015)

We recommend that you use Argon2…

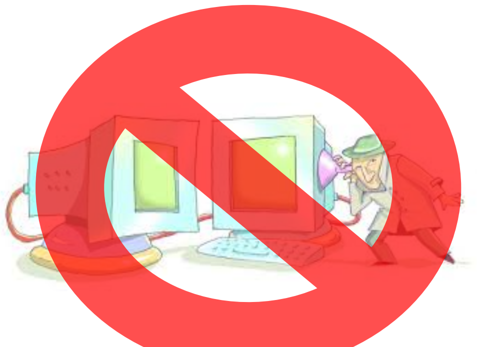
https://password-hashing.net/

password
hashing
competition

(2013-2015)

We recommend that you use Argon2…

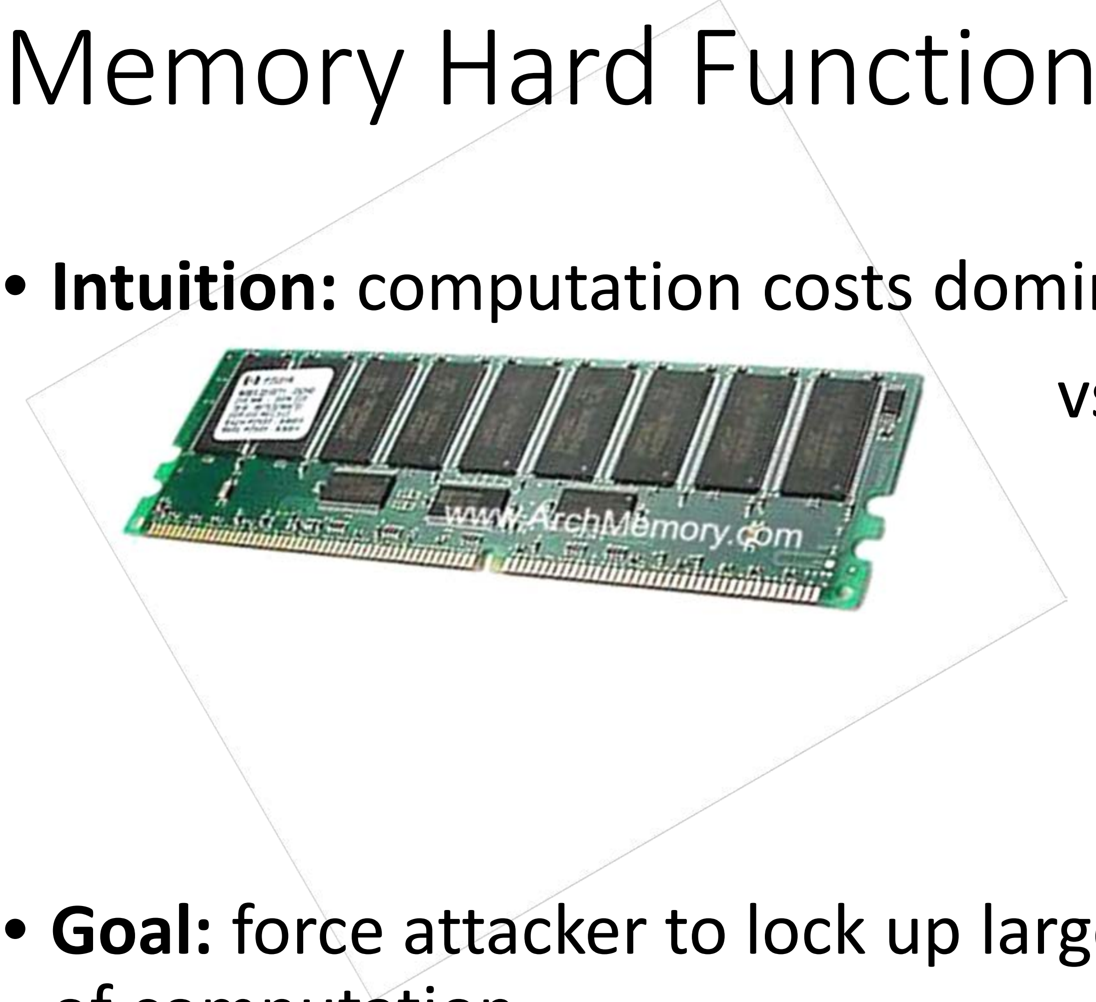
There are two main versions of Argon2, **Argon2i** and Argon2d. **Argon2i** is the safest against side-channel attacks



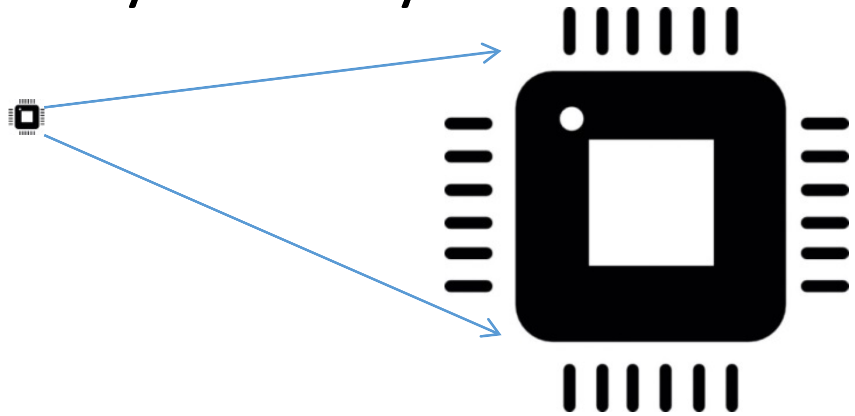
https://password-hashing.net/

Memory Hard Function (MHF)

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



vs.

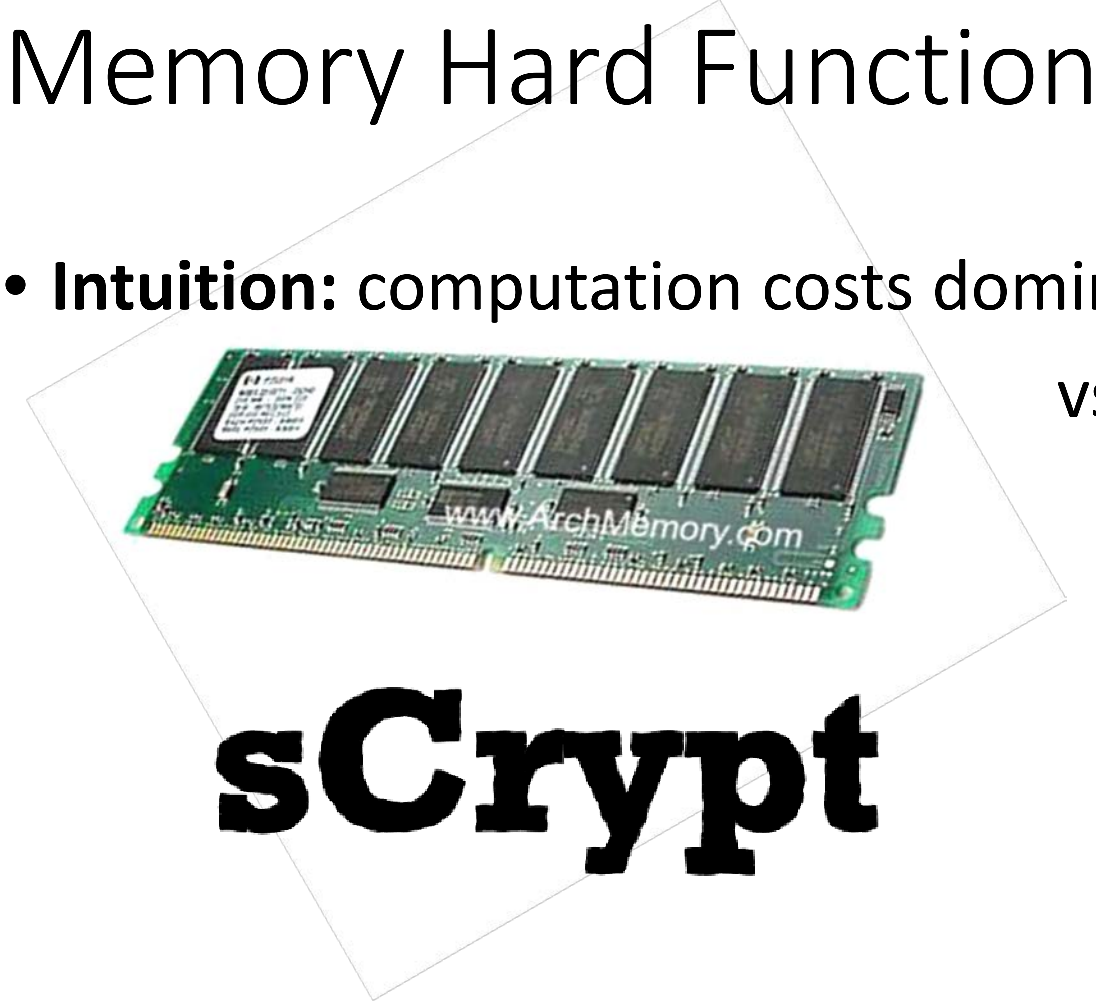


- **Goal:** force attacker to lock up large amounts of memory for duration of computation

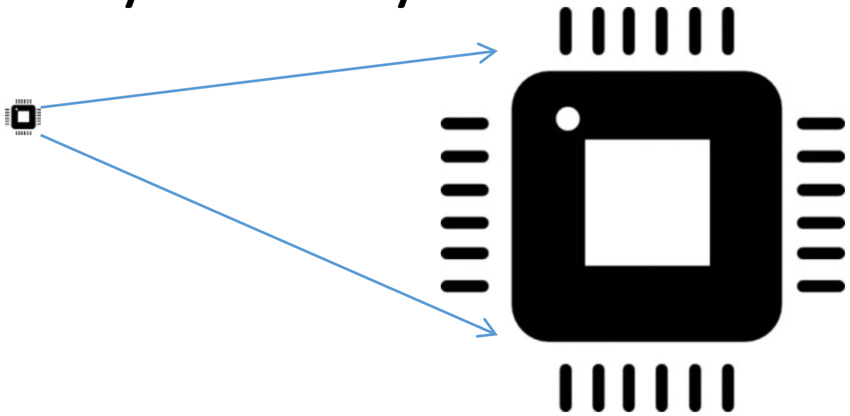
  →Expensive even on customized hardware

Memory Hard Function (MHF)

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



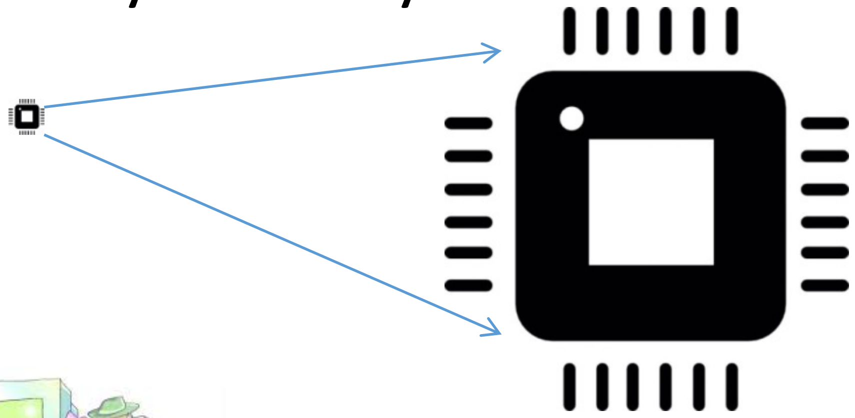
vs.



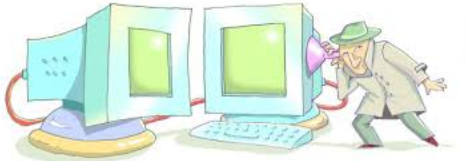
Memory Hard Function (MHF)

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

vs.



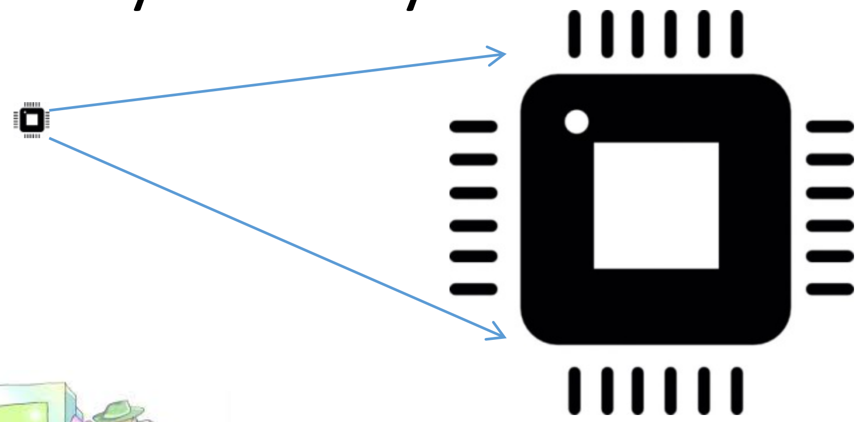
sCrypt



Memory Hard Function (MHF)

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

vs.



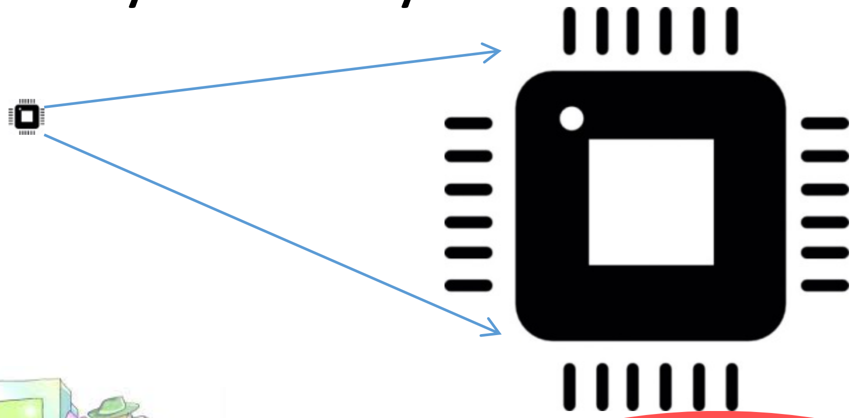
sCrypt

- 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

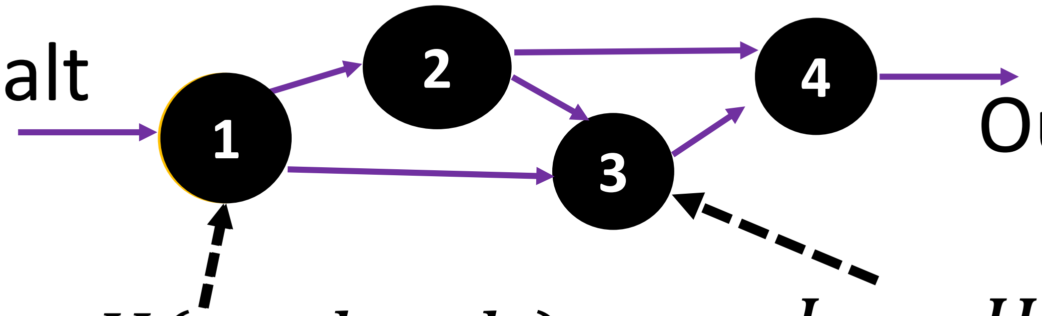
- Data Independent Memory Hard Function (iMHF)
  - Memory access pattern should not depend on input

Data-Independent Memory Hard Function (iMHF)

iMHF $f_{G,H}$ defined by

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

Input: pwd, salt

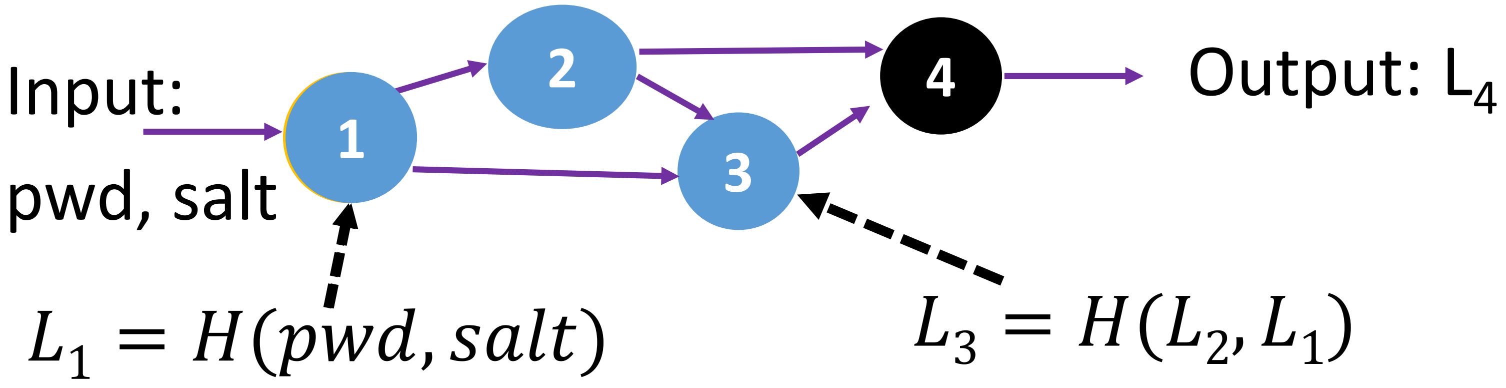


Output: $f_{G,H}$(pwd,salt)= L_4

$L_1 = H(pwd, salt)$

$L_3 = H(L_2, L_1)$

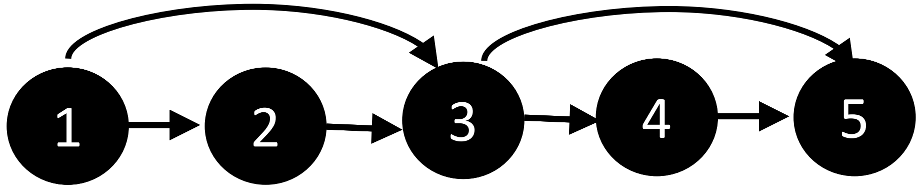
Evaluating an iMHF (pebbling)



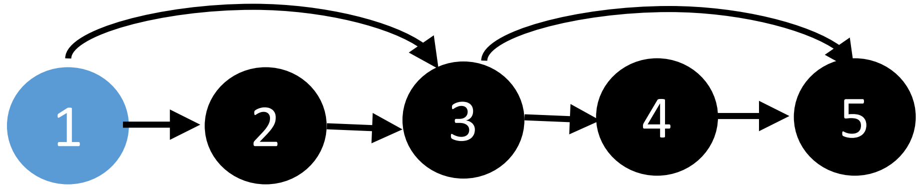
Pebbling Rules : $\vec{P}$=$P_1,\ldots,P_t \subset V$ s.t.

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

Evaluating an iMHF (pebbling)

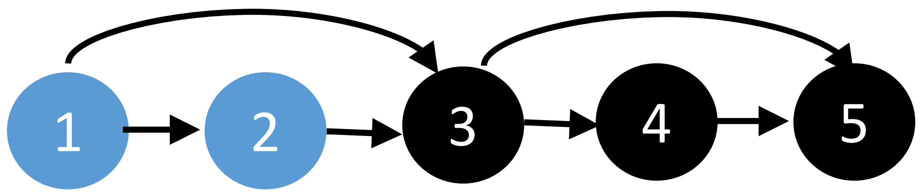


Evaluating an iMHF (pebbling)



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

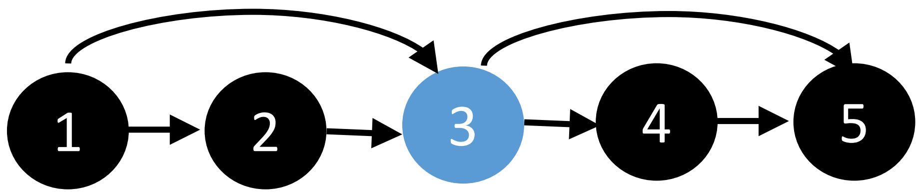
Pebbling Example



$P_1 = \{1\}$

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

Pebbling Example

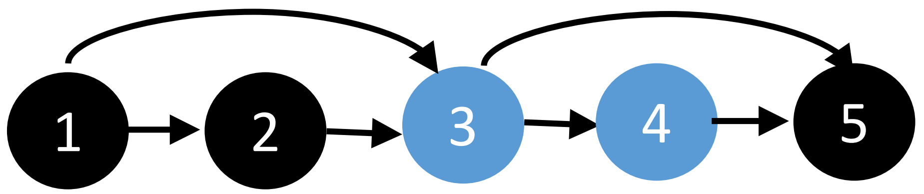


$P_1 = \{1\}$

$P_2 = \{1,2\}$

$P_3 = \{3\}$

Pebbling Example



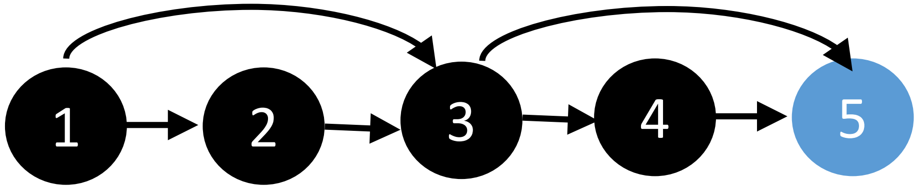
$P_1 = \{1\}$

$P_2 = \{1,2\}$

$P_3 = \{3\}$

$P_4 = \{3,4\}$

Pebbling Example



$P_1 = \{1\}$

$P_2 = \{1,2\}$

$P_3 = \{3\}$

$P_4 = \{3,4\}$

$P_5 = \{5\}$

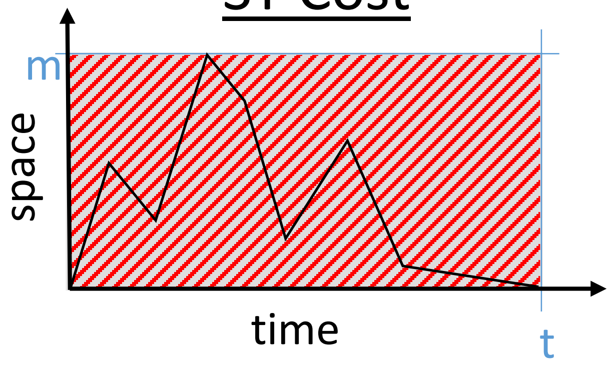
Measuring Cost: Attempt 1

- Space × Time (ST)-Complexity

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

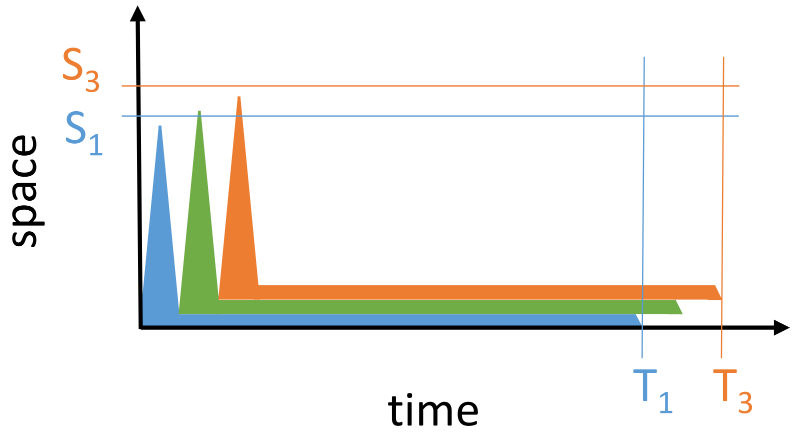
- Rich Theory
  - Space-time tradeoffs
  - But not appropriate for password hashing

<u>ST Cost</u>



Amortization and Parallelism

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



$$ST_1 = S_1 \times T_1 \approx S_3 \times T_3 = ST_3$$

cost of computing f once

cost of computing f three times

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

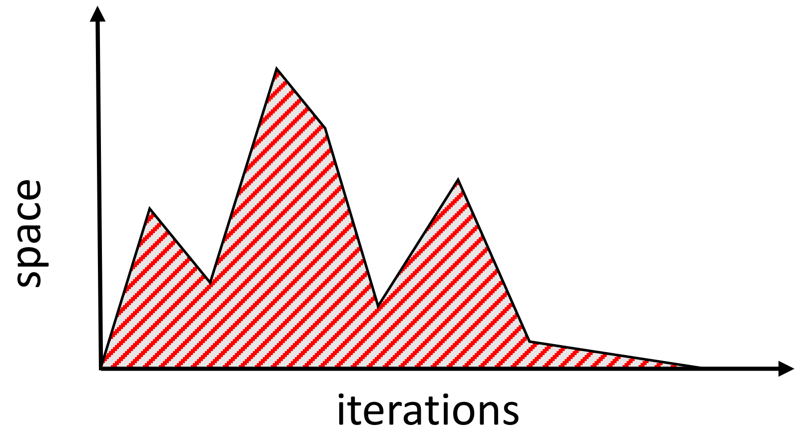
Measuring Pebbling Costs [AS15]

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

Approximates Amortized Area x Time Complexity of iMHF

Memory Used at Step i

Cumulative Memory Cost



Measuring Pebbling Costs [AS15]

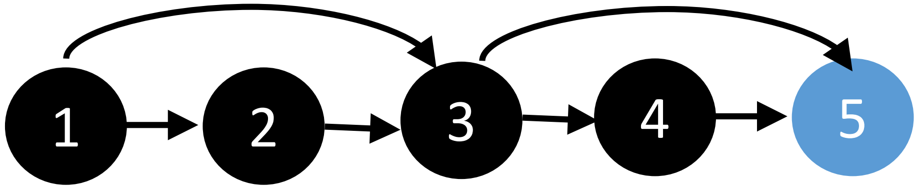
$$\mathrm{CC}(G) = \min_{\vec{P}} \sum_{i=1}^{t_{\vec{P}}} |P_i|$$

Memory Used at Step i

[AS15] Costs scale linearly with #password guesses

$$\mathrm{CC}(\underbrace{G, \ldots, G}_{m \text{ times}}) = \mathrm{m} \times \mathrm{CC}(G)$$

Pebbling Example: Cumulative Cost



$P_1 = \{1\}$

$P_2 = \{1,2\}$

$P_3 = \{3\}$

$P_4 = \{3,4\}$

$P_5 = \{5\}$

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

$$= 1 + 2 + 1 + 2 + 1$$

$$= 7$$

Lessons from SCRYPT

SCRYPT [Per09]

- **CON:** Data-Dependent→ Side-Channel Concerns
- **PRO:** Proven to have high CC [ACPRT17]
  - CC(SCRYPT) = $\Omega(n^2)$
  - Contrast: *any* iMHF has CC at most $O\left(\frac{n^2 \log\log n}{\log n}\right)$

- Maximally Memory Hard → Egalitarian?

What Happened?

- CC(SCRYPT) = $\Omega(n^2)$ the function can be computed with low memory
- Each strategy below is easily feasible
  - Evaluate with $\mathrm{O}(n)$ memory in $O(n)$ time
  - Evaluate with $\mathrm{O}(\sqrt{n})$ memory in $O(n\sqrt{n})$ time
  - Evaluate with $\mathrm{O}(1)$ memory in $O(n^2)$ time
- SCRYPT ASIC miners opt for low memory + high computation options
- **Goal:** Ensure that low memory options are infeasible

SilverFish

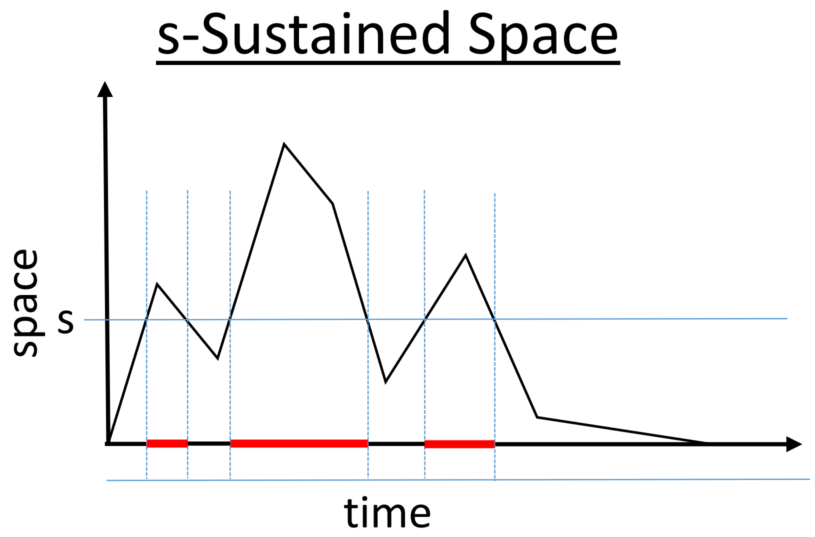
25MH Scrypt Miner

25MH/s @ 440w

Sustained Space

- Using memory is more costly than doing computation (at least for ASICs).
- **Idea:** Only charge for computational steps where a lot of memory is being used.
- **Definition:** s-Sustained Space

"Time spent above memory threshold s"



Intuition: trade-offs are free.

FACT:
$\mathrm{CC} > st$

Wanted: A Moderately Hard Function

- Desiderata:
  - Cost for honest & adversary roughly same:

Honest Computational Model

- Sequential Computation
- Single Evaluation
- Cost measured in ST Complexity

Adversarial Model

- Parallel Computation
- Amortization across many evaluations
- Cost measured in s-SS (for some large s)

Main Theorem

- For any n∈ℕ we give a function f_n and prove that in the parallel Random Oracle Model (PROM):

Honest	Adversarial
• Sequential Algorithm $\mathcal{E}$	• ∀ parallel algs. $\mathcal{A}$
• Time($\mathcal{E}(f_n)$) = n	• s-SS($\mathcal{A}(f_n)$) = Ω(n) per eval.
• ST($\mathcal{E}(f_n)$) = n^2	for s = Ω(n/log(n))

- Bonus: f_n is an iMHF.

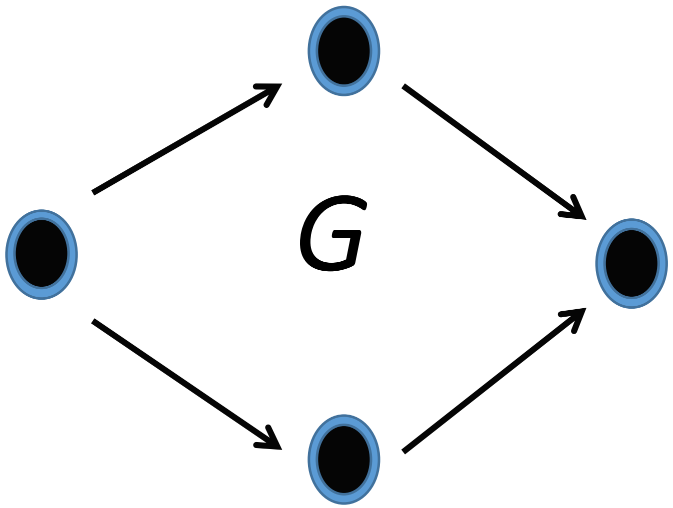
  ⇒$\mathcal{E}$ runs in constant time and has data-independent memory access pattern

The Parallel Black Pebbling Game

Parallel Black Pebbling Game: Same as Black Pebbling, except can touch many pebbles per iteration.

Goal: Place a pebble on the sink.
Rule 1: A node can be pebbled only if all parents contain a pebble.
Rule 2: A pebble can always be removed.

s-SS analogue: Count number of steps when at least s pebbles on graph.

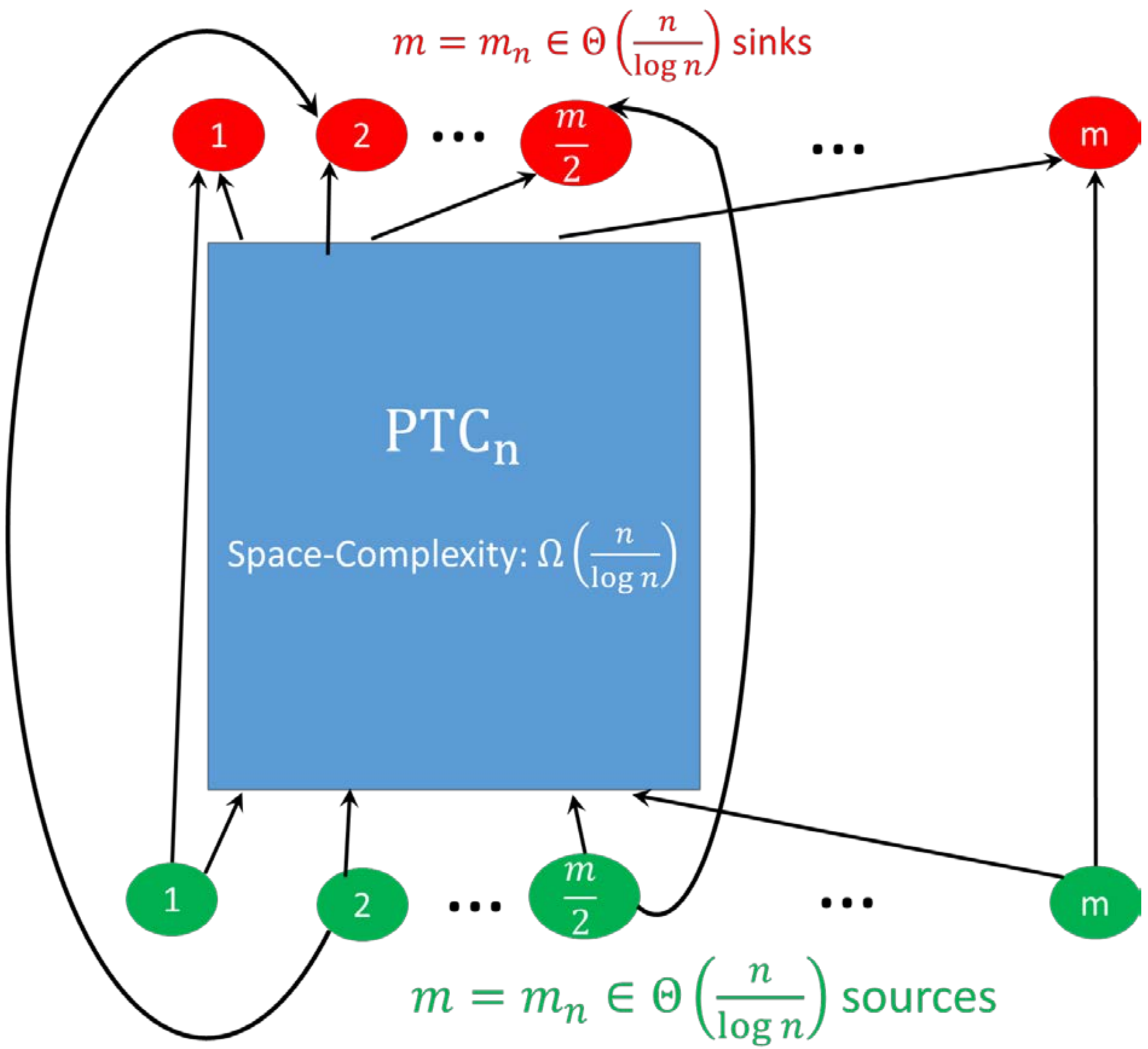


s-SS Complexity
1-SS = 3
2-SS = 1

Want *G* with…

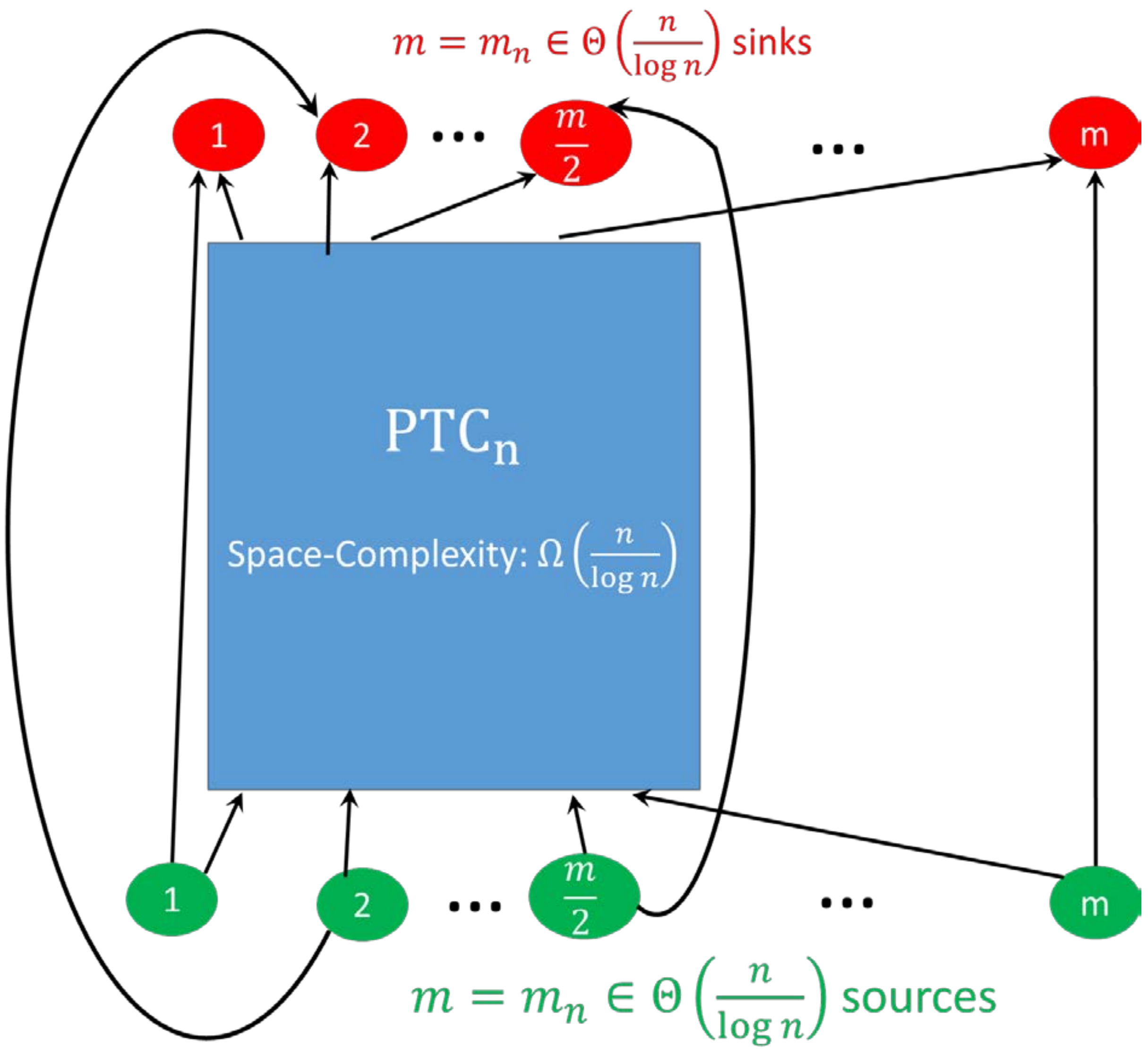
1. Size(G) = n
2. In-degree(G) = 2
3. $\frac{n}{\log(n)}$-SS(G) = Ω(n)

Technical Ingredient #1 [PTC77]



- [PTC77] Built a constant indegree DAG G with n nodes and proves that any sequential pebbling has at least one step in which there are at least $\Omega(n/\log n)$ pebbles on the graph.
- [Hopcroft77] Any constant indegree graph DAG G can be pebbled with space at most $O(n/\log n)$

Technical Ingredient #1 [PTC77]



- [PTC77] Built a constant indegree DAG G space complexity $\Omega(n/\log n)$
- Recursive Construction
  - $\mathrm{PTC_{2n}}$ contains 2 internal copies of $\mathrm{PTC_n}$
- Stronger Lemma used for Induction!
  - For any *sequential* pebbling $P_1, \ldots, P_t$

  We can find an interval $[i, j] \subseteq [t]$ such that both

  1. $|P_k| \geq c_1 m$ for each $\mathrm{k} \in [i, j]$
  2. At least $c_2 m$ source nodes are (re)pebbled during the interval

Technical Ingredient #1 [PTC77]

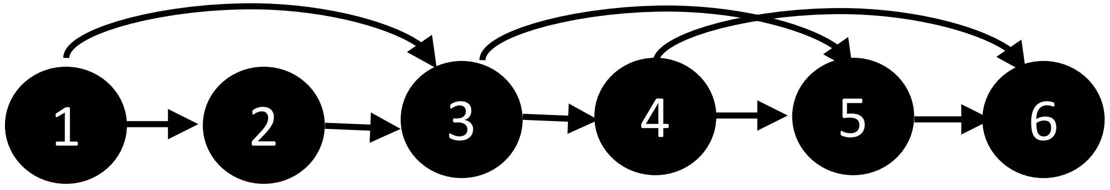
- [PTC77] Built a constant indegree DAG G space complexity $\Omega(n/\log n)$
- Recursive Construction
  - PTC_{2n} contains 2 internal copies of PTC_n
- Stronger Lemma used for Induction!
  - For any ***parallel (~~sequential~~)*** pebbling $P_t, \ldots, P_t$
  - Can find an interval $[i, j] \subseteq [t]$ such that
    1. $|P_k| \geq c_1 n/\log n$ for each $\mathrm{k} \in [i, j]$ (lots of pebbles on the graph)
    2. At least $c_2 n/\log n$ source nodes are (re)pebbled during the interval
  - Implication ($s = c_1 n/\log n$): s-SS(P) $\geq j + 1 - i$
  - **Sequential Pebbling:** $j + 1 - i \geq c_2 n/\log n$ (by (2) above)
  - **Parallel Pebbling:** Could (re)pebble all $c_2 n/\log n$ in one step!

Depth Robustness [ABP17]

Definition: A DAG G=(V,E) is (e,d)-depth-robust *for all* $S \subseteq V$ s.t. $|S| \leq e$ we have $\mathrm{depth}(G - S) > \mathrm{d}.$

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

Example: (e=2,d=2)-reducible

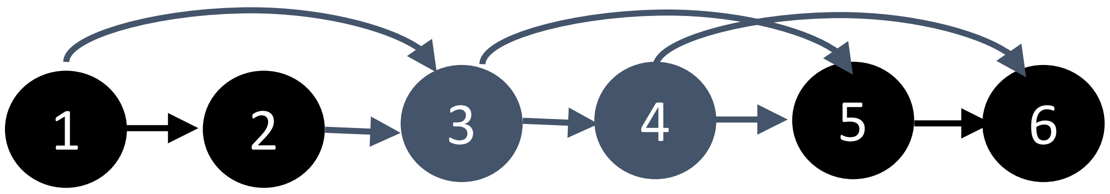


Block Depth Robustness [ABP17]

Definition: A DAG G=(V,E) is (e,d)-depth-robust *for all* $S \subseteq V$ s.t. $|S| \leq e$ we have $\mathrm{depth}(G - S) > \mathrm{d}$.

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

Example: (e=2,d=2)-reducible



Technical Ingredient #2

Definition: A DAG G_n^ε is ε-extremely depth robust if it is (e,d)-depth-robust *for all* $e + d \leq (1 - \varepsilon)n$.

- [EGS75] n node G with log(n) in-degree and (Ω(n), Ω(n))-depth-robust
  - Problem: Constants too small e.g., $e = 10^{-4}n$ and $d = 10^{-2}n$
  - Problem: in-degree too high.
- [MMV13] ε-extremely depth robust DAG G_n^ε with $\log^2(n)$ in-degree and (e,d)-DR for any e+d < n(1-ε).
  - Problem: in-degree too high.

Technical Ingredient #2

Definition: A DAG G_n^{ε} is ε-extremely depth robust if it is (e,d)-depth-robust *for all* $e + d \leq (1 - \varepsilon)n$.

- [MMV13] ε-extremely depth robust DAG G_n^{ε} with indegree $O(\log^2 n \ \text{polylog}(\log n))$.
  - Problem: in-degree too high.
- **[NEW]** ε-extremely depth robust DAG D_n^{ε} with indegree O(log (n))
  - Construction: similar to [EGS75]
  - Many technical details to work out (see paper)

Technical Ingredient #2

Definition: A DAG G_n^ε is ε-extremely depth robust if it is (e,d)-depth-robust *for all* $e + d \leq (1-\varepsilon)n$.

- **[NEW]** ε-extremely depth robust DAG D_n^ε with indegree O(log (n))
  - Construction: similar to [EGS75]
  - Many technical details to work out (see paper)

Useful Observation: *Any* subgraph of $D_n^\varepsilon[S]$ of size $|S| > \varepsilon n$ must contain a path of length $|S| - \varepsilon n$

Proof: Otherwise DAG D_n^ε is not (e, d)-depth robust for $\mathrm{d} = |S| - \varepsilon n$ and $e = |V \setminus S| = n - |S|$. Contradiction, D_n^ε is ε-extremely depth robust and

$$e + d = n - \varepsilon n \leq (1-\varepsilon)n.$$

Technical Ingredient #2

Definition: A DAG G_n^ε is ε-extremely depth robust if it is (e,d)-depth-robust *for all* $e + d \leq (1 - \varepsilon)n$.

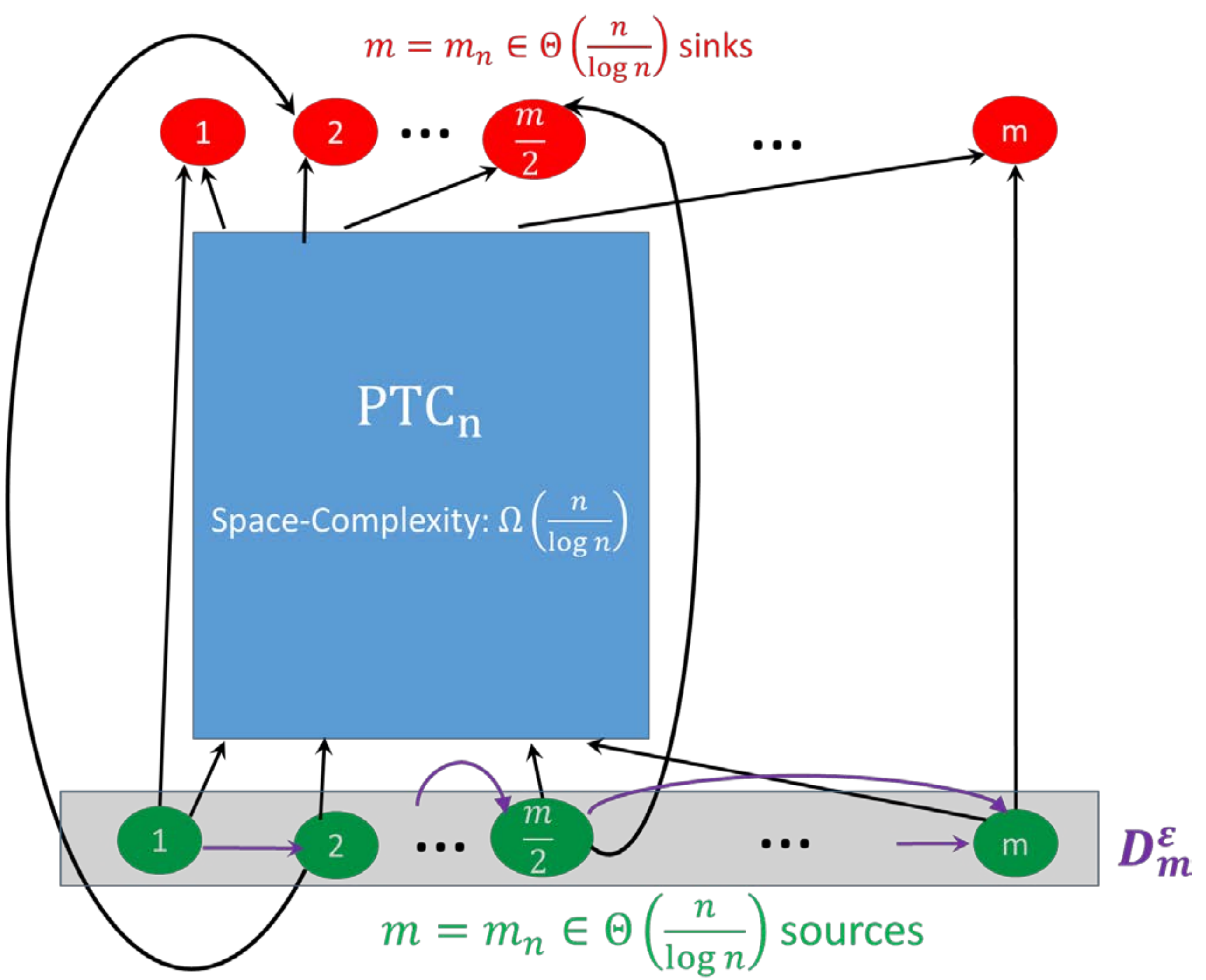
Lemma: If legal (parallel) pebbling $P_1,\ldots,P_t$ of D_n^ε has at least one pebbling round j with space $s = |P_j| > 2\varepsilon n$ then there are at least $t = \frac{|P_j|}{2} - \varepsilon n$ *distinct* time rounds k with space $|P_k| \geq \frac{|P_j|}{2}$

Proof: Let i<j be last round before round j such that $|P_i| < \frac{|P_j|}{2}$ and let $S = P_j \setminus P_i$. Any node in S is (re)pebbled during the interval [i,j].

→ (observation) the subgraph $J_n^\varepsilon[S]$ contains a path of length $t \geq \frac{|P_j|}{2} - \varepsilon n$

→ at least t pebbling rounds to reach configuration P_j from P_i

PTC Overlay (Attempt 1)



Lemma [PTC77] In any pebbling of $\mathrm{PTC_n}$ we can find an interval $[i, j] \subseteq [t]$ such that

1. $|P_k| \geq c_1 n / \log n$ for each $k \in [i, j]$ (lots of pebbles on the graph)
2. At least $c_2 n / \log n$ source nodes are (re)pebbled during the interval

[NEW] Now requires $\Omega(m)$ rounds since D_n^ε is ε-extremely depth robust

PTC Overlay (Attempt 1)

Lemma [PTC77] In any pebbling of PTC_n we can find an interval $[i, j] \subseteq [t]$ such that

1. $|P_k| \geq c_1 n/\log n$ for each $\mathrm{k} \in [i, j]$ (lots of pebbles on the graph)
2. At least $c_2 n/\log n$ source nodes are (re)pebbled during the interval

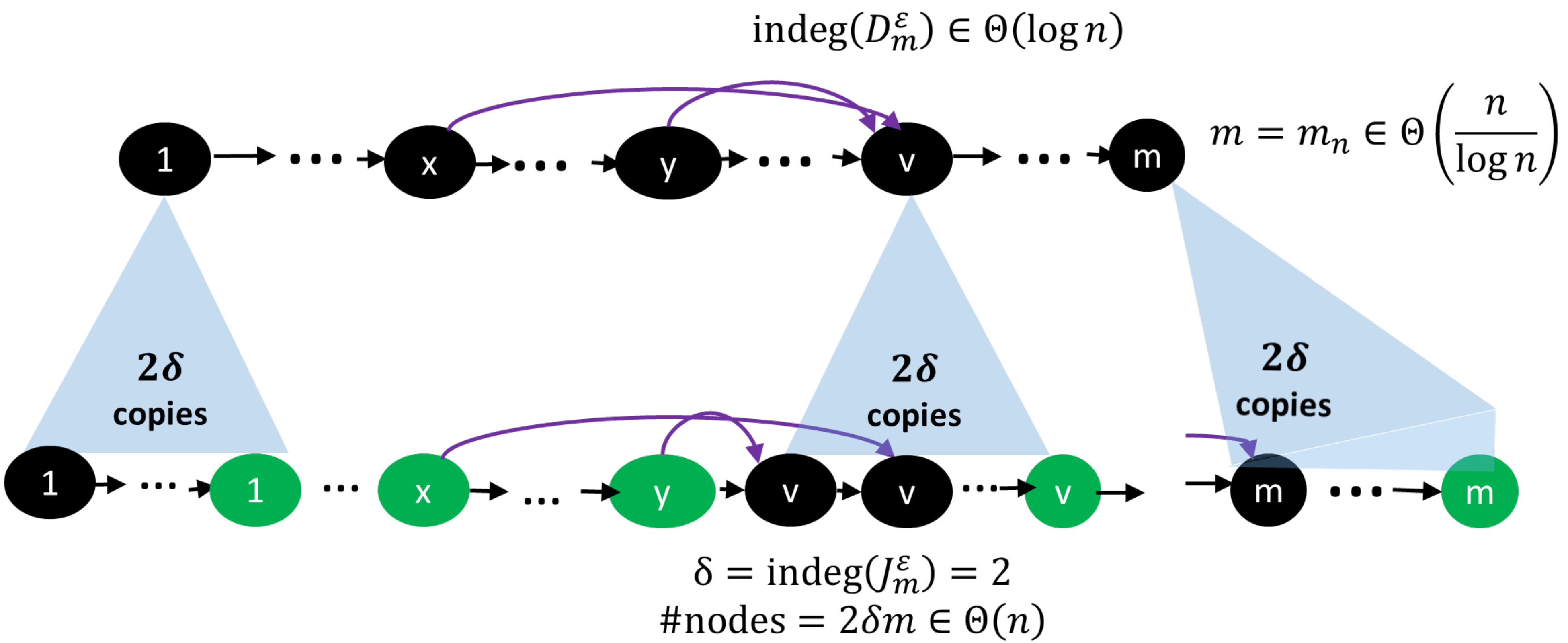
[NEW] Overlay requires $\Omega(m)$ rounds since D_m^ε is ε-extremely depth robust

Problems:

- Requires $\mathrm{s} = \Omega(n/\log n)$ pebbles for $\mathrm{t} = \Omega(n/\log n)$ rounds
  - I promised $\mathrm{s} = \Omega(n/\log n)$ pebbles for $\mathrm{t} = \Omega(n)$ rounds)
- Indegree still too high i.e., $\mathrm{indeg}(D_m^\varepsilon) = O(\log n)$
  - I promised constant indegree O(1)

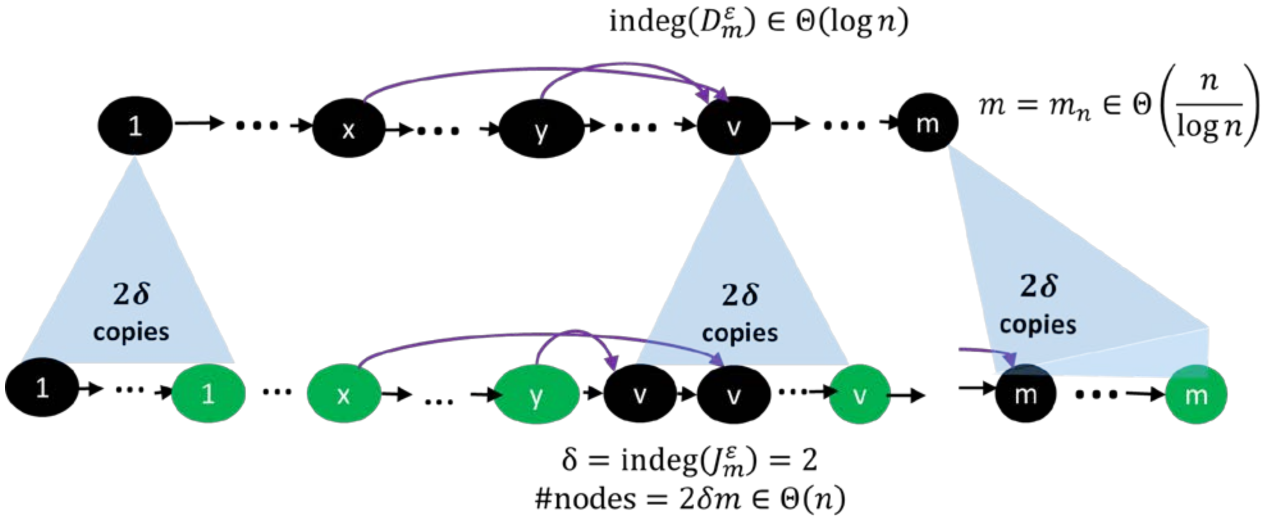
Technical Ingredient #3

- Indegree Reduction [ABP17] deals with both problems simultaneously!



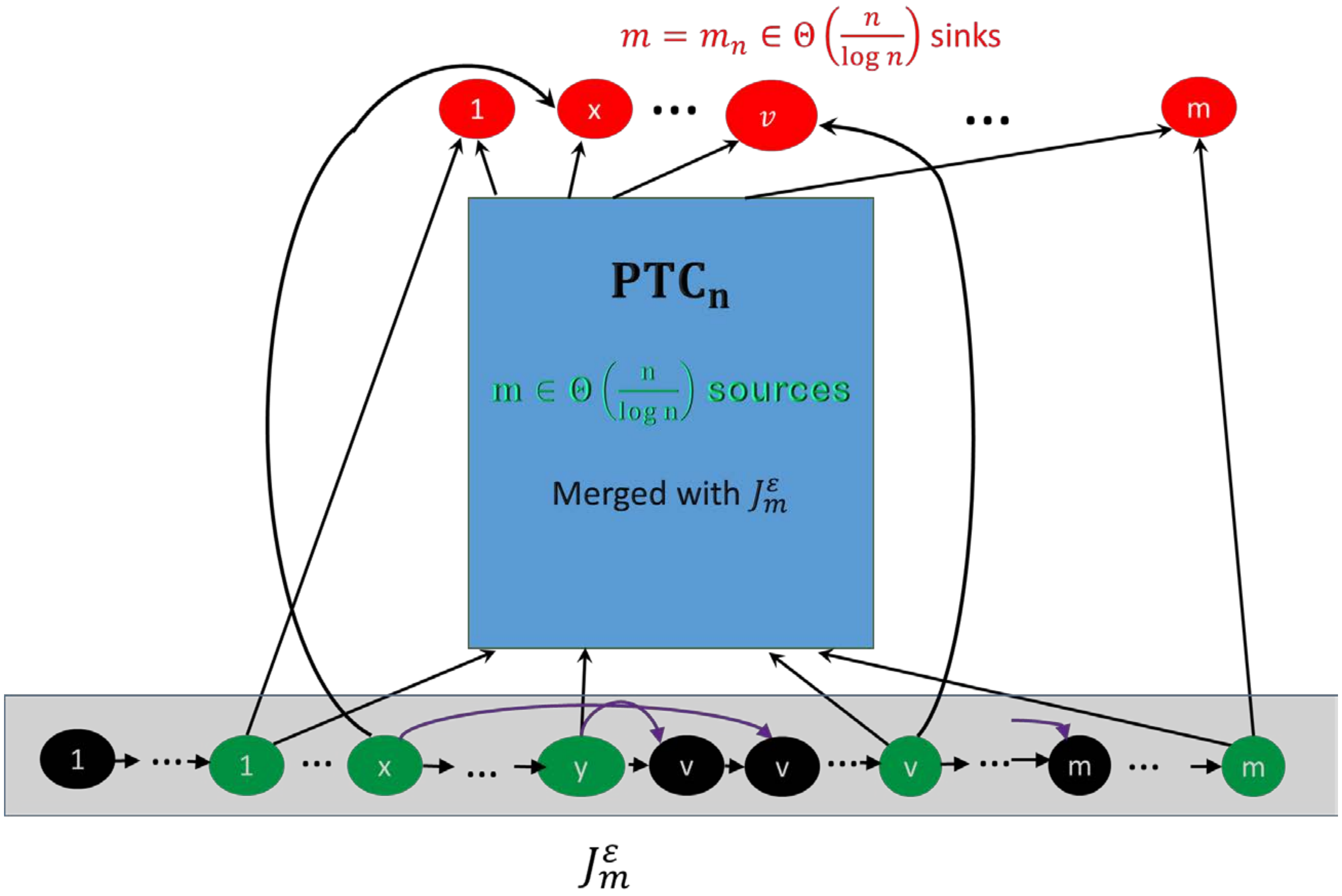
Technical Ingredient #3

- Indegree Reduction [ABP17] deals with both problems simultaneously!



Lemma [ABP17]: If D_m^ε is (e, d)-depth robust then J_m^ε is $(e, d\delta)$-depth robust. Furthermore, $\text{indeg}(\mathrm{J}_\mathrm{m}^\varepsilon) = 2$ and $\mathrm{J}_\mathrm{m}^\varepsilon$ has $2dm = O(n)$ nodes.

The Final Construction



Theorem: Any (parallel) pebbling requires

$s = \Omega(n/\log n)$ pebbles for $t = \Omega(n)$ rounds

Technical Details in paper

Consequences of new Depth-Robust Graphs

- Logic: "Parallel Black-White Pebbling"
  - Application: CNF formulas with very memory costly refutation resolution proofs.
- MHFS: Applications: Optimal CC for any graph of size n even though only O(log(n)) in-degree
  - $CC(D_m^{\varepsilon}) \geq \frac{(1-\eta)n^2}{2}$
  - Exact Constants!
  - Complete DAG: $CC\left(K_n\right) \leq \frac{n^2}{2}$ (prior result is almost optimal!)
- **Coding Theory:** better locally detectable error detection codes [BGGZ19]
- Improved Proof-of-Sequential work (temporarily. See "Simple Proofs of Sequential Work" for construction without depth-robust graphs).

A Few Open Questions

- Practical Construction of iMHF with high sustained space complexity?
- Analyze/improve constant factors in bounds
  - Computer Aided Analysis?
- Stronger Results for dMHFs? Hybrid Modes like Argon2id?
- Find constant indegree DAG with parallel space-time complexity $\mathrm{ST}^{||}(\mathrm{G}) = \Omega(n^2)$ or show that no such DAG exists
  - **Note:** [AB16] pebbling shows that $CC(\mathrm{G}) = O\left(\frac{n^2 \log\log n}{\log n}\right)$, but the pebbling attack P still has $\mathrm{ST}^{||}(P) = \Omega(n^2)$

A Few Open Questions

- Practical Construction of iMHF with high sustained space complexity?
  - See upcoming crypto 2019 paper
  - Data-Independent Memory Hard Functions: New Attacks and Stronger Constructions (with Ben Harsha and Siteng Kang and Seunghoon Lee and Lu Xing and Samson Zhou).

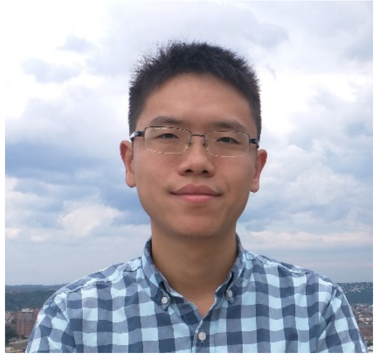
Theorem: Any pebbling of (practical) DAG G either has

1. Cumulative Cost $\omega(n^2)$, or
2. At least $\mathrm{s} = \Omega(n/\log n)$ pebbles for $\mathrm{t} = \Omega(n)$ rounds

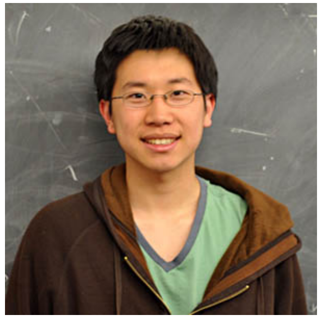
Announcements & Reminders

- Homework 2 Due Tonight (2/23/2023)
- Midterm Next Week
  - Informal Poll: Take Home vs. In-Class
- Course Presentation (Signup Sheet will be Announced Soon)

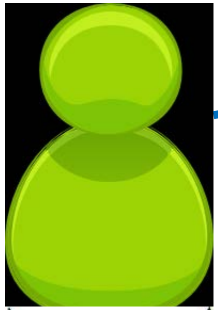
Bandwidth Hard Functions: Reductions and Lower Bounds



Jeremiah Blocki (Purdue)
Ling Ren (MIT)
Samson Zhou (Purdue)



Offline Attacks



jblocki, 123456

PlayStation™

Username	Salt	Hash
jblocki	89d978034a3f6	85e23cfe0021 f584e3db87aa 72630a9a234 5c062

HASH(12345689d978034a3f6)=85e23cfe0021f584e3db87aa72630a9a2345c062

+

Offline Attacks: A Common Problem

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

Key Stretching

Hash Function	Cost: C
H	
H^{τ}	

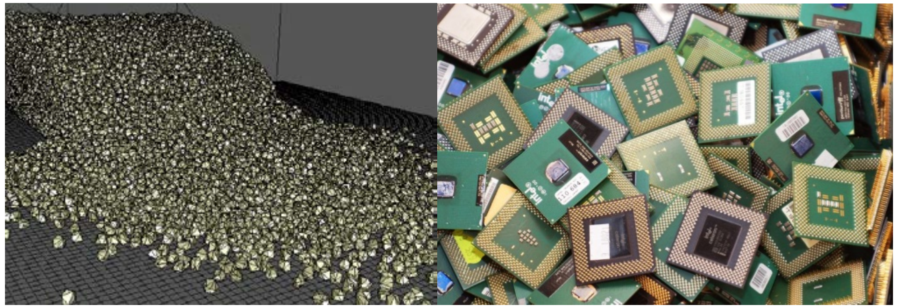
Hash Iteration

BCRYPT PBKDF2

Memory Hard Functions

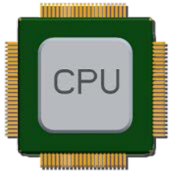
sCrypt

What is the ASIC Advantage?



Advertised Capacity: >200,000x faster than

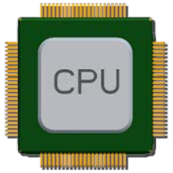
4.73 Th/s



$$ per eval():   capital   +   electricity

\# of lifetime eval()

What is the ASIC Advantage?



$$ per eval(): amortized capital + electricity

Reducing ASIC Advantage

Memory-hard functions [Percival'09 (scrypt)]:

"A natural way to reduce the advantage provided by an attacker's ability to construct highly parallel circuits is to **increase the size of the circuit."**

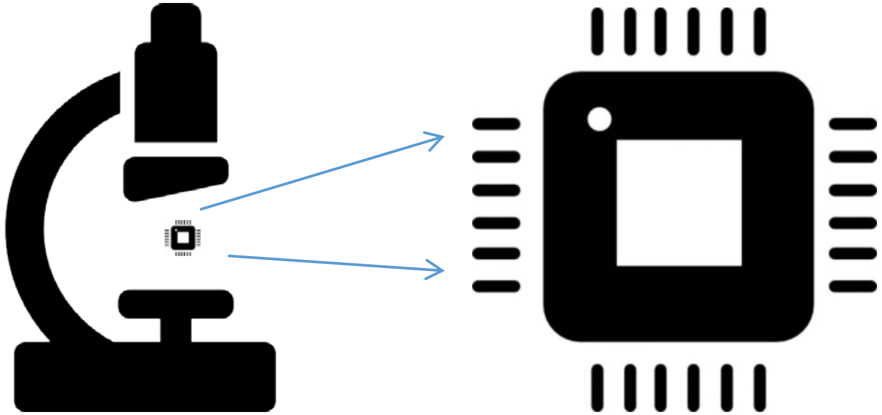
Size of the circuit:

- **dominated by memory**
- **Reasonable approximation of amortized capital costs**

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

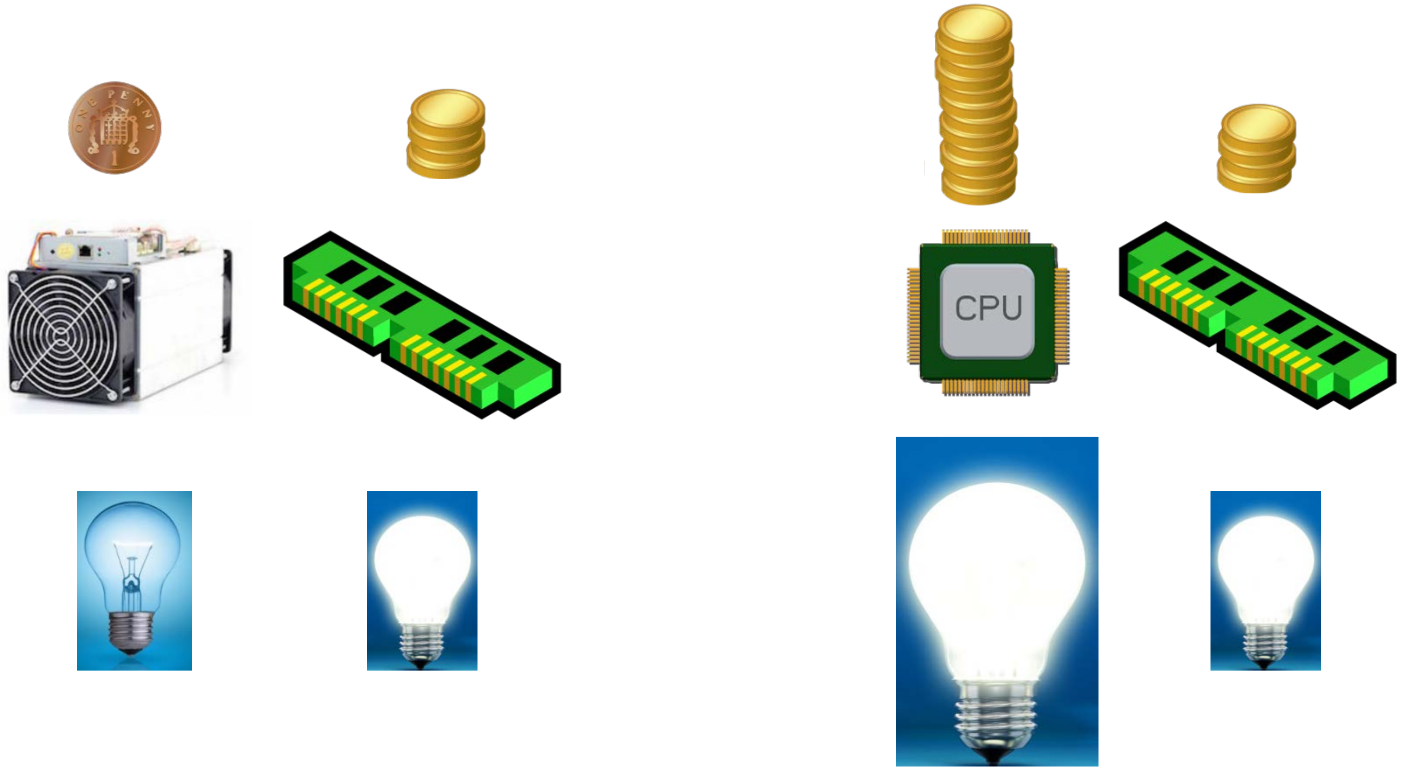
Lot’s of Work on Memory Hard Functions

- [Percival’09 (scrypt)]
- Password Hashing Competition
  - Argon2 (winner), Catena, Lyra2, yescrypt…
  - Data-Independent (iMHF) vs Data-Dependent (dMHF)
    - iMHF: harder to construct, but resistant to side-channel attacks like cache-timing
- [Boneh et al.’ 16 (Balloon Hash)]
- [Alwen & Serbinenko’ 15]
  - Definitional issue with ST-complexity (amortization of costs)
  - Cumulative Memory Complexity (stronger requirement to address amortization)
- [Alwen & Blocki’ 16, 17]
  - Argon2i, Balloon Hash and other iMHFs have low cumulative memory complexity

Lot’s of Work on Memory Hard Functions

- [Alwen & Blocki’ 16, 17]
  - Argon2i, Balloon Hash and other data-independent memory hard functions have low cumulative memory complexity (cmc)
- [ABP17]
  - Theoretical construction of iMHFs with asymptotically optimal cumulative memory complexity
- [ABH17]
  - First practical construction of iMHFs with asymptotically optimal cumulative memory complexity
- [ABP18] Sustained Space Complexity

Reducing ASIC Advantage



(memory-hard) (bandwidth-hard)

$$ per eval(): amortized capital + electricity

How to Define Bandwidth Hardness?

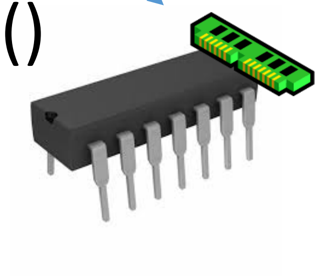
Energy Cost

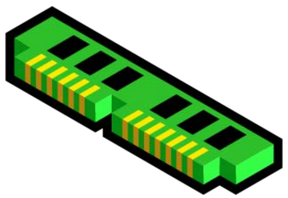
- Graph labeling, compute H:$\{0,1\}^{2w} \rightarrow \{0,1\}^{w}$ in a DAG
- Give the adversary a cache
- Energy Cost

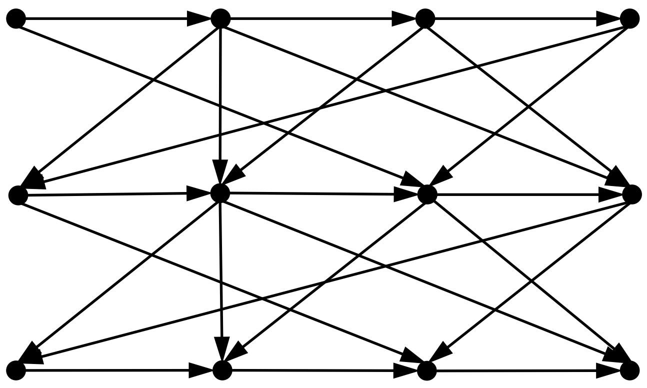
$$\mathrm{ecost}\left(f_{G,H}, mw\right) = C_b \times \frac{(\#\text{bits transfered to/from cache})}{w} + C_r \times (\#\text{evals H})$$

Bit in cache

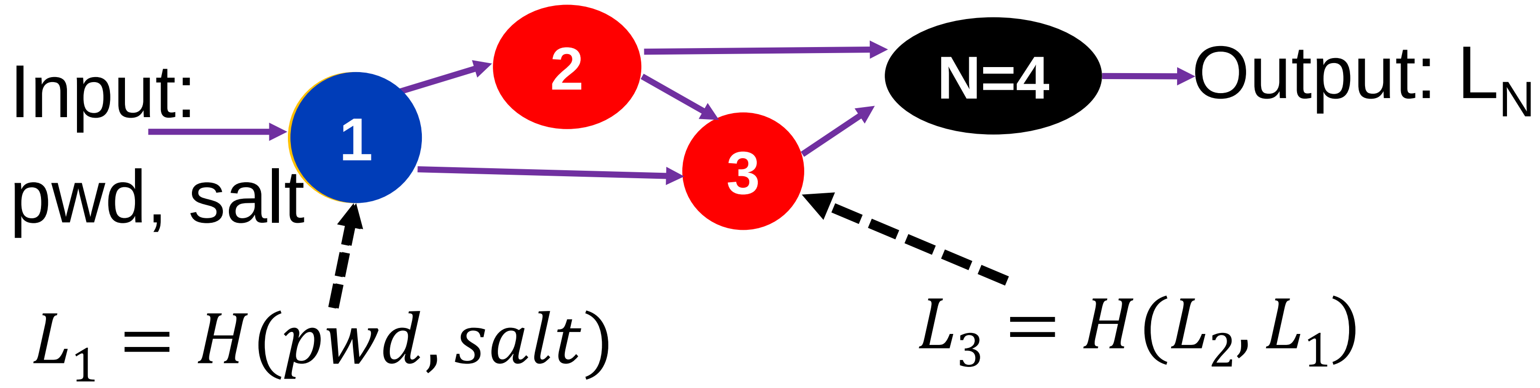
H()







Evaluating an iMHF (red-blue pebbling)



Pebbling: $\vec{P}$=(B_1,R_1)…, (B_t,R_t) where

- Set of labels stored in memory at round i: B_i
- Set of labels stored in cache at round i: R_i (Cache-Size: $|R_i| \leq m$)

Goal: place red pebble on last node (N) in in G

Evaluating an iMHF (red-blue pebbling)

Pebbling: $\vec{P} = (B_0 = \emptyset, R_0 = \emptyset), (B_1, R_1), \ldots, (B_t, R_t)$

- B_i set of labels stored in memory at time i
- R_i set of labels stored in cache at time i. (Cache-Size: $|R_i| \leq m$)

Legal Pebbling Moves between Rounds:

- [Blue Move] Change the color of a pebble (cache-miss: store/load value from memory)
- [Red Move] Place new red pebble on node v if $\text{parents}(v) \subset R_i$
- [Discard Pebble] May discard pebble(s) at any time.

Red-Blue Pebbling Cost [RD17]

$$\mathrm{rbpeb}(P) = \mathrm{C_b} \times (\#\text{Blue Moves in P}) + \mathrm{C_r} \times (\#\text{Red Moves in } P)$$

$$\mathrm{rbpeb}(G,m) = \min_{P \in \mathcal{RB}(G,m)} \mathrm{rbpeb}(P)$$

Set of all legal red-blue pebblings of DAG G with cache-size m.

Red-Blue Pebbling Cost Inequity [RD17]

Honest Party (CPU):

$$\mathrm{rbpeb}(P) = \mathrm{C_b} \times (\#\mathrm{Blue\ Moves\ in\ P}) + \mathrm{C_r} \times (\#\mathrm{Red\ Moves\ in\ } P)$$

Attacker (ASIC):

$$\mathrm{rbpeb}'(P') = \mathrm{C_b'} \times (\#\mathrm{Blue\ Moves\ in\ P'}) + \mathrm{C_r'} \times (\#\mathrm{Red\ Moves\ in\ } P')$$

Attacker gets to play with potentially advantageous constants

$$1\mathrm{nJ} \approx \mathrm{C_b'} \approx \mathrm{C_b} \approx \mathrm{C_r} \approx 10^{-3} \times \mathrm{C_r'} \approx 1\mathrm{pJ} \quad (\mathrm{C_r'} \ll \mathrm{C_r})$$

Red-Blue Pebbling Cost Inequity [RD17]

Honest Party (CPU):

$$\mathrm{rbpeb}(P) = \mathrm{C_b} \times (\#\text{Blue Moves in P}) + \mathrm{C_r} \times (\#\mathrm{R}$$

How can I make sure that the function is energy intensive for the attacker as well?

Attacker (ASIC):

$$\mathrm{rbpeb}'(P') = \mathrm{C_b'} \times (\#\text{Blue Moves in P}') + \mathrm{C_r'} \times (\#\text{Red} \quad)$$

Attacker gets to play with potentially advantageous cons

$\mathrm{C_r'} \ll \mathrm{C_r}$ $\qquad\qquad$ $\mathrm{C_b'} = \Theta(\mathrm{C_b})$

[Abadi et al.'05]
[Dwork et al.'03]

A Natural Approach

- An iMHF $f_{G,H}$ is memory-bound if:
  - Computable with at most *B* cache misses (resp. blue moves)
  - Not computable with < *cB* cache misses (resp. blue moves) even using a cache of size *M*

(definition for dMHFs is similar, but does not involve pebbling)

Problem: Hard to construct; must rule out *all* space-time tradeoffs

Theorem[Hopcroft'77]: If G has constant indegree then there is a black pebbling which never requires more than S = O(N/log(N)) pebbles.

Corollary: If M= O(N/log(N)) we need 0 blue-moves

Bandwidth-hard functions [RD17]

- **Observation:** computation is not free (even for attacker)!
  - Allows for slight relaxation of goal
- **Definition:** An iMHF $f_{G,H}$ is bandwidth hard against attacker with cache-size m if

Best Red-Blue Pebbling for Honest Party

$$\frac{\mathrm{rbpeb}(G,m)}{\mathrm{rbpeb}'(G,m)} = \Theta(1)$$

Best Red-Blue Pebbling for ASIC attacker

Sufficient Condition: $\mathrm{rbpeb}(G,m) = \Omega(N \times \mathrm{C_b})$

Prior State of Affairs (Bandwidth-Hardness)

Prior Results [RD17]:

- Proved that DAGs for several key iMHFs satisfy

$$\mathrm{rbpeb}(G, m) = \Omega(N \times \mathrm{C_b}) \quad (1)$$

  - Catena-BRG
  - Balloon Hash

- Proved that dMHF scrypt is bandwidth-hard *

*vs restricted class of attackers

Key Open Questions:

Pebbling Reduction? Is it true that *any* algorithm A computing $f_{G,H}$ in the random oracle model can be described as a red-blue pebbling strategy?

(**Thm:** [AS15] holds for black pebblings)

Does equation (1) hold for

- Argon2i? (PHC Winner)
- DRSample? (Maximal CMC [ABH17])
- aATSample? (Maximal CMC [ABH17])

Pebbling Reduction [BRZ18]

Pebbling Reduction: *Any* algorithm A computing $f_{G,H}$ in the random oracle model can be described as a red-blue pebbling strategy with comparable cost.

$$\text{ecost}\left(f_{G,H}, m \times w\right) \geq \Omega\left(\text{rbpeb}(G, 8m)\right)$$

Arguably a reasonable upper bound on cache-size
Typical $N = 2^{20}$ (1KB Blocks) = (1GB RAM)
$N = 2^{40/3}$ (1KB Blocks) = (10MB cache)

- Argon2i: $\text{ecost}\left(G, \tilde{O}\left(N^{2/3}\right)\right) = \Omega(N \times C_b)$
- DRSample: $\text{ecost}\left(G, O\left(N^{1-\varepsilon}\right)\right) = \Omega(N \times C_b)$
- aATSample: $\text{ecost}\left(G, \tilde{O}(N)\right) = \Omega(N \times C_b)$

Tolerates Larger Cache-Size

Additional Results [BRZ18]

Computational Complexity: NP-Hard to find $\mathrm{ecost}(\mathrm{G})$.

(Open Question: Approximate $\mathrm{ecost}(\mathrm{G})$?)

Tight Relationship between parallel and sequential pebblings:

$$\mathrm{rbpeb}(G, 2m) \leq \mathrm{rbpeb}^{\parallel}(G, m)$$

(this relationship does not hold for black pebblings!)

Generic Connection Between Memory Hardness and Bandwidth Hardness: *Any MHF f(.) with high cumulative memory complexity must have reasonably high energy cost.*

Additional Results [BRZ18]

Generic Connection Between Memory Hardness and Bandwidth Hardness: *Any MHF f(.) with high cumulative memory complexity must have reasonably high energy cost.*

$$\mathrm{ecost}(f, mw) \geq \Omega\left(\min_{\mathrm{t}}\left(t\, \mathrm{C_r} + \mathrm{C_b}\left(\frac{\mathrm{cmc(f)}}{tw} - m\right)\right)\right)$$

Theorem [ABPRT17]: $\mathrm{cmc(scrypt)} = \Omega(\mathrm{N}^2)$

Corollary: $\mathrm{ecost(scrypt)} = \Omega\left(\mathrm{N}\sqrt{\mathrm{C_r} \times \mathrm{C_b}}\right)$

(first unconditional lower bound on energy cost of scrypt)

Bonus: More Contributions

$$\mathrm{ecost}(f, mw) \ \geq \Omega\left(\min_{\mathrm{t}}\left(t\,\mathrm{C_r} + \mathrm{C_b}\left(\frac{\mathrm{cmc(f)}}{tw} - m\right)\right)\right)$$

Theorem [ABPRT17]: $\mathrm{cmc(scrypt)} = \Omega(\mathrm{N}^2)$

Corollary: $\mathrm{ecost(scrypt)} = \Omega\left(\mathrm{N}\sqrt{\mathrm{C_r} \times \mathrm{C_b}}\right)$

(first unconditional lower bound on energy cost of scrypt)

Comparison: [RD17] lower bound is slightly stronger $\Omega(\mathrm{N} \times \mathrm{C_b})$ for restricted adversary class.

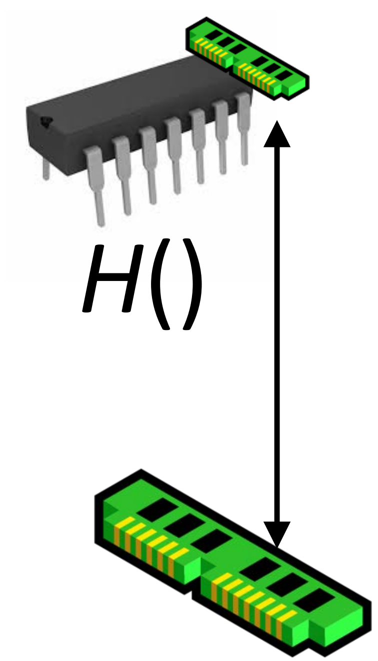
Recent: Unconditional proof that $\mathrm{ecost(scrypt)} = \Omega(\mathrm{N} \times \mathrm{C_b})$

Pebbling Reduction

- **Goal:** Compute $f_{G,H}$

minimize

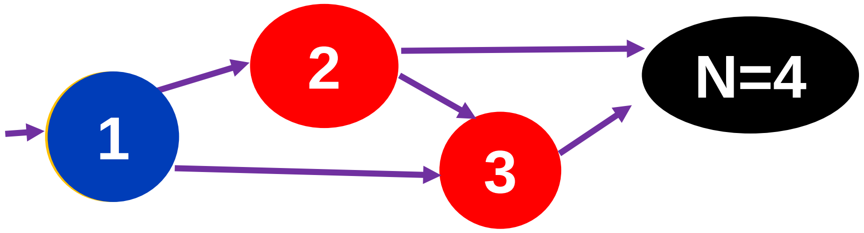
$\mathrm{ecost}\left(f_{G,H}, mw\right)$



- **Goal:** Pebble G

minimize

$\mathrm{rbpeb}(G, m)$

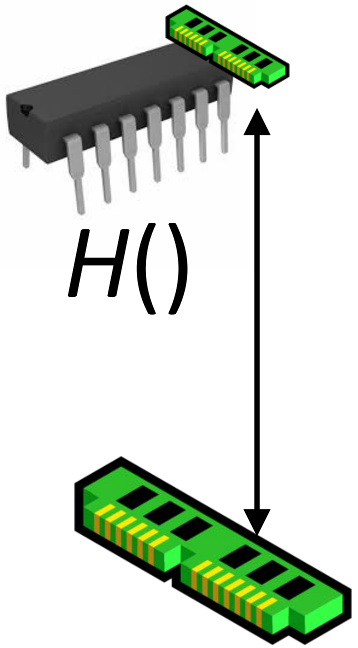


Pebbling Reduction

- **Goal:** Compute $f_{G,H}$

minimize

$$\mathrm{ecost}\left(f_{G,H}, mw\right)$$

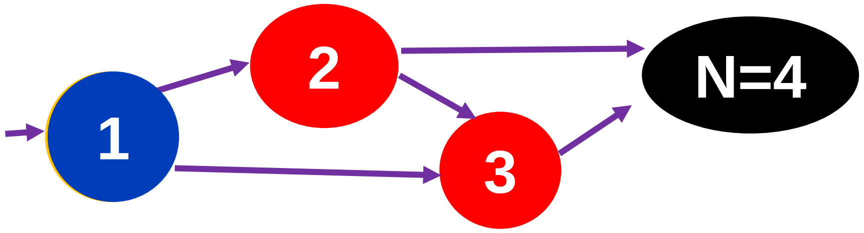


Easy Direction

- **Goal:** Pebble G

minimize

$$\mathrm{rbpeb}(G, m)$$

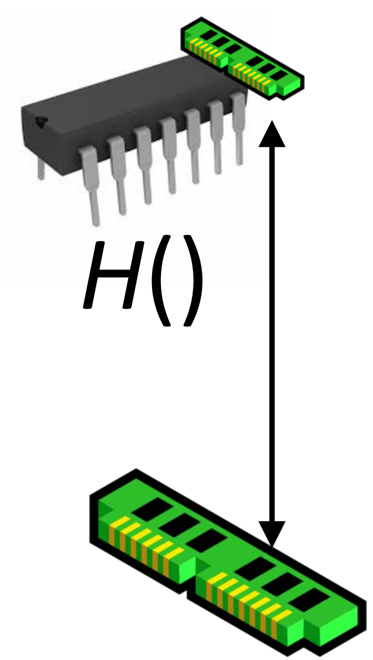


Pebbling Reduction

- **Goal:** Compute $f_{G,H}$

minimize

$$\mathrm{ecost}\left(f_{G,H}, mw\right)$$

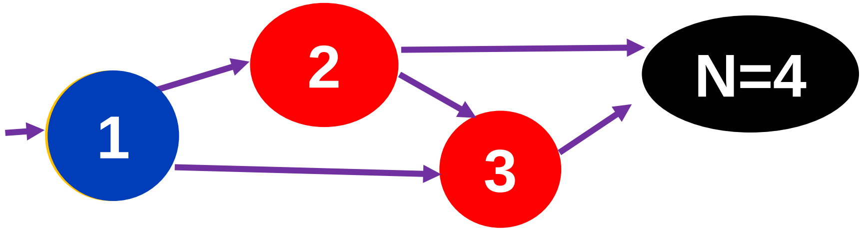


Extractor Argument
(can't compress
labels from RO)

- **Goal:** Pebble G

minimize

$$\mathrm{rbpeb}(G, O(m))$$



Pebbling Reduction

- **Prior** pebbling reduction implies that total number of pebbles on graph (red or blue) is proportional to overall state size (cache+RAM)
- **Challenge:** Ex-post facto pebbling only gives us black pebbling $P_1,\ldots,P_t$.
  - Which pebbles should we color blue/red in each round?
  - We cannot directly see what labels are transferred to/from cache (the labels might be stored in encrypted form!)
- **Recall:** In ex-post facto pebbling P_i denotes labels that appear "out of the blue" in our simulation i.e., the next time these labels appear will be as the input to a random oracle query.

Pebbling Reduction

- **Prior** pebbling reduction implies that total number of pebbles on graph (red or blue) is proportional to overall state size (cache+RAM)

- **Challenge:** Ex-post facto pebbling only gives us black pebbling $P_1,\ldots,P_t$.

- **Recall:** In ex-post facto pebbling P_i denotes labels that appear "out of the blue" in our simulation i.e., the next time these labels appear will be as the input to a random oracle query.

- **Intuition:** We expect that at least $|P_i|$-m of the labels in P_i will have to be transferred from cache in the future at cost $(|P_i|-m)C_b$.

Pebbling Reduction

- **Challenge:** Ex-post facto pebbling only gives us black pebbling $P_1,\ldots,P_t$.
- **Recall:** In ex-post facto pebbling P_i denotes labels that appear "out of the blue" in our simulation i.e., the next time these labels appear will be as the input to a random oracle query.
- **Intuition:** We expect that at least $|P_i|$-m of the labels in P_i will have to be transferred from cache in the future incurring cost $(|P_i|-m)C_b$.
- **Suppose Not:** If fewer than $(|P_i|-m)C_b/2$ **bits are transferred to/from cache after round i then extractor hint would include**
  - **Cache State at round i (mw) bits**
  - **Bits transferered between cache/memory ($(|P_i|-m)C_b/2$ bits)**
  - **Additional information to extract labels ($\ll(|P_i|-m)\, C_b/2$ bits)**
  - **➔ Contradiction! We would extract a random** $|P_i|$w-bit string with a much shorter hint

Pebbling Reduction

- **Key Definition:** QueryFirst(t_1,t_2)
  - Data-labels L_v that appear "out of the blue" as input to RO query before output during rounds [t_1, t_2]
  - Dependent on execution trace of attacker.
- **Partition time into intervals** [t_1, t_2], [1+t_2, t_3]… s.t

  4m > |QueryFirst(t_i,t_{i+1})| > 3m

- **Claim 1:** Attacker must transfer at least mw bits to/from cache during each interval [1+t_i,t_{i+1}]
- **Claim 2:** Can find legal red-blue pebbling in which
  1. The number of blue moves during each interval [1+t_i,t_{i+1}] is at most 4m
  2. We never use more than 8m red pebbles.

Pebbling Reduction

Claim 1: Attacker must transfer at least mw bits to/from cache during each interval $[1+t_i, t_{i+1}]$

Proof Sketch: Suppose not then we could use an extractor to extract 3m labels with a hint of size

$$(|\mathrm{h}| - 2mw) \ll 3\mathrm{mw}$$

The odds of this happening are negligible!

- **Extractor Hint:**
  - **State** $\boldsymbol{\sigma_{1+t_i}}$ **of PROM attacker cache A at time** $1+t_i$
    - **ignore memory** $\left(\boldsymbol{\xi_{1+t_i}}\right)$
    - **At most mw bits**
  - **List of messages passed to/from cache during interval** $[1+t_i, t_{i+1}]$
    - **At most mw bits**
  - *List of labels in* QueryFirst(t_i,t_{i+1}) to extract (plus information to recognize relevant queries)
    - $O\big(m\ (\log(n+q))\big) \ll mw$

Extractor for Pebbling Reduction

- Given state of cache $\boldsymbol{\sigma_{1+t_i}}$ and list of messages passed to/from memory we can simulate the attacker.
- When the attacker submits the ith random oracle query
  - Check hint to see the i^{th} query x is of interest
  - Otherwise forward query to random oracle and forward the response to the attacker

- Label appears "out of the blue"
- Making the query "ruins" label L_v we want to extract
  - $L_v = H(v, L_{v-1}, L_{r(V)})$
  - How to identify such a query?
    - Rely on hint.
  - How to continue simulation without making the RO query?
    - L_v previously appeared out of the blue.
    - Thus, extractor can simply send the response L_v

Bandwidth Hardness of Candidate iMHFs

- **Key Pebbling Lemma:** Lower bounds $\mathrm{rbpeb}(G, m, T, B, R)$ cost to pebble target nodes $T \subseteq [N]$ starting from configuration with
  - Blue Pebbles on $B \subseteq [N] \backslash \mathrm{T}$
  - Red Pebbles on $R \subseteq [N] \backslash \mathrm{T}$
- Let $B' \subseteq B$ be blue moves that are eventually converted to red pebbles.

$$\mathrm{rbpeb}(G, m, T, B, R) \geq C_b |B'|$$

$$\mathrm{rbpeb}(G, m, T, B, R) \geq C_r |ancestors_{G - R \cup B'}(T)|$$

- **Intuition:** If there is a path from v to T which avoids the set $R \cup B'$ then node v must be pebbled at some point at cost C_r.

Bandwidth Hardness of Candidate iMHFs

- Key Lemma (central to all proofs)
- Lower bounds $\mathrm{rbpeb}(G, m, T, B, R)$ cost to pebble target nodes $T \subseteq [N]$ starting from configuration with
  - Blue Pebbles on $B \subseteq [N]\backslash \mathrm{T}$
  - Red Pebbles on $R \subseteq [N]\backslash \mathrm{T}$
- Let $B' \subseteq B$ be blue moves that are eventually converted to red pebbles.

Lemma: $\forall T, B, R \subseteq [N]\backslash \mathrm{T}$

$$\mathrm{rbpeb}(G, m, T, B, R) \geq \min_{B' \subseteq B} \left(C_r \left|ancestors_{G-R\cup B'}(T)\right| + C_b |B'|\right)$$

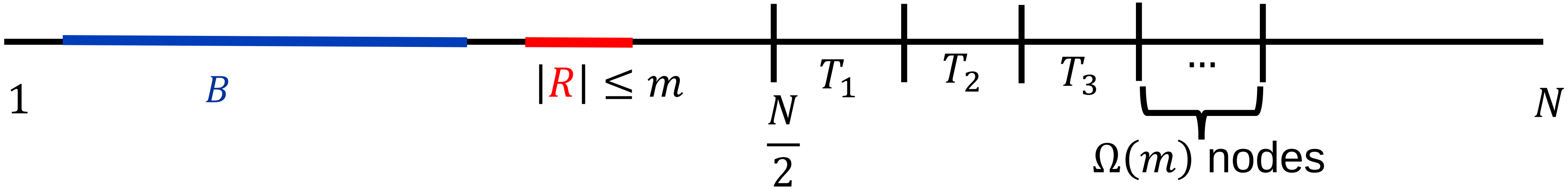
Bandwidth Hardness of Candidate iMHFs

Lemma: $\forall T, B, R \subseteq [N] \backslash \mathrm{T}$

$$\mathrm{rbpeb}(G, m, T, B, R) \geq \min_{B' \subseteq B} \left(C_r \left| ancestors_{G-R \cup B'}(T) \right| + C_b |B'| \right)$$

Partition the nodes $[N] \backslash \left[\frac{N}{2}\right]$ into $\Omega\left(\frac{N}{m}\right)$ intervals $T_1, T_2, \ldots,$ each containing $\Omega(m)$ nodes.

$$\mathrm{rbpeb}(G, m) \geq \sum_{i \geq 1} \min_{\substack{B, R \subseteq [N] \backslash T_i \\ s.t. |R| \leq m}} \left(\mathrm{rbpeb}(G, m, T_i, B, R) \right)$$



Bandwidth Hardness of Candidate iMHFs

Lemma: $\forall T, B, R \subseteq [N] \setminus T$

$$\mathrm{rbpeb}(G, m, T, B, R) \geq \mathrm{mi}$$

We lower bound this quantity for three iMHF candidates Argon2i, DRSample and aATSample

Partition the nodes $[N] \setminus \left[\frac{N}{2}\right]$ in … aining $\Omega(m)$ nodes.

$$\mathrm{rbpeb}(G, m) \geq \sum_{i \geq 1} \min_{\substack{B, R \subseteq [N] \setminus T_i \\ s.t. |R| \leq m}} \left(\mathrm{rbpeb}(G, m, T_i, B, R)\right)$$

$$\geq \sum_{i \geq 1} \min_{\substack{B, R \subseteq [N] \setminus T_i \\ s.t. |R| \leq m}} \left(\min_{B' \subseteq B} \left(C_r |ancestors_{G - R \cup B'}(T_i)| + C_b |B'|\right)\right)$$

Bandwidth Hardness of Candidate iMHFs

Theorem: $[N] \backslash \left[\frac{N}{2}\right]$ into $\Omega\left(\frac{N}{m}\right)$ intervals $T_1, T_2, \ldots,$ each containing $\Omega(m)$ nodes then $\mathrm{rbpeb}(G, m) \geq$

$$\sum_{i \geq 1} \min_{\substack{B, R \subseteq [N] \backslash T_i \\ s.t. |R| \leq m}} \left(\min_{B' \subseteq B} (C_r |ancestors_{G - R \cup B'}(T_i)| + C_b |B'|) \right)$$

Argon2i if $m = O\left(N^{\frac{2}{3} - \varepsilon}\right)$ then for each interval T_i of $\Omega(m)$ nodes we have

$$\min_{\substack{B, R \subseteq [N] \backslash T_i \\ s.t. |R| \leq m}} \left(\min_{B' \subseteq B} (C_r |ancestors_{G - R \cup B'}(T)| + C_b |B'|) \right) = \Omega\left(\min\left\{ N\, C_r, N^{\frac{2}{3}} C_b \right\} \right)$$

Bandwidth Hardness of Candidate iMHFs

Theorem: $[N] \backslash \left[\frac{N}{2}\right]$ into $\Omega\left(\frac{N}{m}\right)$ inter... T ...ng $\Omega(m)$ nodes then $\mathrm{rbpeb}(G, m) \geq$

$$\sum_{i \geq 1} \min_{\substack{B,R \subseteq [N] \backslash T_i \\ s.t. |R| \leq m}} \left(\min_{B' \subseteq} \ldots \right)$$

Amortized: $\Omega(1)$ blue moves per node in interval (best possible)

Argon2i if $m = O\left(N^{\frac{2}{3}-\varepsilon}\right)$ then for each interval T_i of $\Omega(m)$ nodes we have

$$\min_{\substack{B,R \subseteq [N] \backslash T_i \\ s.t. |R| \leq m}} \left(\min_{B' \subseteq B} \left(C_r |ancestors_{G-R \cup B'}(T)| + C_b |B'| \right) \right) = \Omega\left(\min\left\{ N\, C_r, N^{\frac{2}{3}} C_b \right\} \right)$$

Bandwidth Hardness of Candidate iMHFs

Theorem: $[N]\setminus\left[\frac{N}{2}\right]$ into $\Omega\left(\frac{N}{m}\right)$ intervals T_1, T … g $\Omega(m)$ nodes then $\mathrm{rbpeb}(G, m) \geq$

$$\sum_{i\geq 1} \min_{\substack{B,R\subseteq[N]\setminus T_i \\ s.t.|R|\leq m}} \left(\min_{B' \subseteq B} \right.$$

> Amortized: $\Omega\left(N^{\frac{1}{3}}\right)$ red moves per node in interval (expensive even if $C_r \ll C_b$

Argon2i if $m = O\left(N^{\frac{2}{3}-\varepsilon}\right)$ then for each interval T_i of $\Omega(m)$ nodes we have

$$\min_{\substack{B,R\subseteq[N]\setminus T_i \\ s.t.|R|\leq m}} \left(\min_{B' \subseteq B}\left(C_r\left|ancestors_{G-R\cup B'}(T_i)\right| + C_b|B'|\right)\right) = \Omega\left(\min\left\{N\, C_r, N^{\frac{2}{3}} C_b\right\}\right)$$

Bandwidth Hardness of Candidate iMHFs

Argon2i

$$\min_{\substack{B,R\subseteq[N]\setminus T_i \\ s.t. |R|\le m}} \left(\min_{B' \subseteq B} \left(C_r \left| ancestors_{G-R\cup B'}(T_i) \right| + C_b |B'| \right) \right) = \Omega\left(\min\left\{ N\, C_r, N^{\frac{2}{3}} C_b \right\} \right)$$

We must pay this cost $\Omega\left(\frac{N}{m}\right)$ times for each interval T_i

$$\mathrm{rbpeb}\left(G, N^{\frac{2}{3}-\varepsilon}\right) = \Omega\left(\min\left\{ N^{\frac{4}{3}} C_r, N C_b \right\} \right)$$

Bandwidth Hardness of Candidate iMHFs

Lemma: $\forall T, B, R \subseteq [N]\backslash \mathrm{T}$

$$\mathrm{rbpeb}(G, m, T, B, R) \geq \min_{B' \subseteq B} (C_r |\mathrm{ancestors}_{\mathrm{G-R\cup B'}}(\mathrm{T})| + C_b |B'|)$$

DRSample: For any constant $\rho < 1$ if $\mathrm{m} = O(N^{\rho})$

$$\min_{\substack{B,R \subseteq [N]\backslash T_i \\ s.t. |R| \leq m}} \left(\min_{B' \subseteq B} (C_r |ancestors_{G-R\cup B'}(T_i)| + C_b |B'|) \right)$$

$$= \Omega\left(\min\left\{N^{\frac{1}{2}+\frac{\rho}{2}} C_r, N^{\rho} C_b\right\}\right)$$

Bandwidth Hardness of Candidate iMHFs

Lemma: $\forall T, B, R \subseteq [N]\backslash T$
$\mathrm{rbpeb}(G, m, T, B, R \ldots |)$

Amortized ($\rho = 0.8$): $\Omega(N^{0.1})$
red moves per node in interval
(expensive even if $C_r \ll C_b$

DRSample: For any constant ρ

$$\min_{\substack{B,R \subseteq [N]\backslash T_i \\ s.t. |R| \leq m}} \left(\min_{B' \subseteq B} \left(C_r \left| \mathrm{ancestors}_{G-R\cup B'}(T_i) \right| + C_b |B'| \right) \right)$$

$$= \Omega\left(\min\left\{ N^{\frac{1}{2}+\frac{\rho}{2}} C_r, N^{\rho} C_b \right\} \right)$$

Bandwidth Hardness of Candidate iMHFs

Lemma: $\forall T, B, R \subseteq [N]\backslash \mathrm{T}$
$\mathrm{rbpeb}(G, m, T, B, R) \ldots |)$

Amortized: $\Omega(1)$ blue moves per node in interval (best possible)

DRSample: For any constant ρ

$$\min_{\substack{B,R \subseteq [N]\backslash T_i \\ s.t. |R| \leq m}} \left(\min_{B' \subseteq B} \left(C_r |ancestors_{G-R \cup B'}(T_i)| + C_b |B'| \right) \right)$$

$$= \Omega\left(\min\left\{ N^{\frac{1}{2}+\frac{\rho}{2}} C_r, N^{\rho} C_b \right\} \right)$$

Bandwidth Hardness of Candidate iMHFs

DRSample: For any constant $\rho < 1$ if $\mathrm{m} = O(N^\rho)$

$$\min_{\substack{B,R \subseteq [N]\backslash T_i \\ s.t. |R| \le m}} \left(\min_{B\prime \subseteq B} (C_r |ancestors_{G-R \cup B'}(T_i)| + C_b |B'|) \right)$$

$$= \Omega\left(\min\left\{ N^{\frac{1}{2}+\frac{\rho}{2}} C_r, N^\rho C_b \right\} \right)$$

If $\mathrm{m} = O(N^\rho)$ must pay this cost $\Omega\left(\frac{N}{N^\rho}\right)$ times

$$\mathrm{rbpeb}(G, N^\rho) = \Omega\left(\min\left\{ N^{\frac{3}{2}-\frac{\rho}{2}} C_r, N C_b \right\} \right)$$

Comparison between Argon2i and DRSample

- **Argon2i** is maximally bandwidth hard if attacker's cache size is $m = o\left(N^{2/3}\right)$ ☺
  - Arguably a reasonable assumption in practice
- Argon2i is not maximally memory hard ☹
  - But it does beat out other entrants in the Password Hashing Competition ☺

- **DRSample** is both maximally memory hard and maximally bandwidth hard ☺
  - Even if attackers cache size is $m = O(N^{1-\varepsilon})$
- **aATSample** is also maximally memory hard and maximally bandwidth hard ☺
  - Even if attackers cache size is $m = O\left(\frac{N}{\log N}\right)$

Scrypt is maximally memory-hard

Joël Alwen
IST Austria

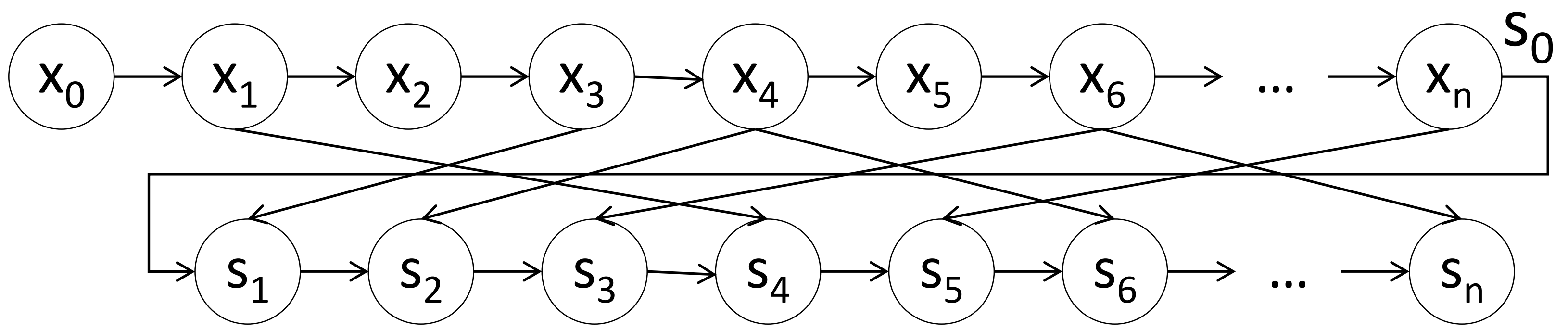
Binyi Chen
UCSB

Krzysztof Pietrzak
IST Austria

Leonid Reyzin
Boston U.
(work done at IST Austria)

Stefano Tessaro
UCSB

[Percival 2009]: scrypt



H: $\{0,1\}^* \rightarrow \{0,1\}^w$ random oracle

Input: x_0

Repeat n times: $x_i = H(x_{i-1})$

$s_0 = x_n$

Repeat n times: $s_i = H(s_{i-1} \oplus x_j)$ for $\mathbf{j = s_{i-1} \bmod n}$

Output: s_n

Data-Dependent Memory Access
➔ Pebbling Attacks Don't apply

scrypt in the wild

- Used in several cryptocurrencies, most notably Litecoin (a top-4 cryptocurrency by market cap)
- Idea behind password-hashing winner Argon2d
- Attempts to standardize within IETF (RFC 7914)

Memory-Hard Functions

Goal: Find moderately hard F for which
special-purpose hardware, parallelism, and amortization do not help.

Proposal [Percival 2009]: make a function that needs a lot of memory

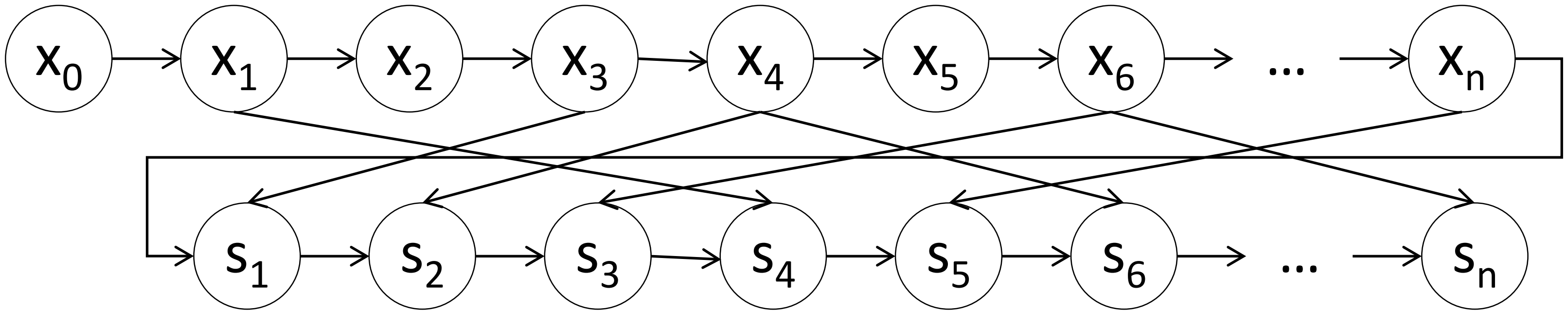
(memory is always general, unlike computation)

Make sure parallelism cannot help
(force evaluation to cost the same)

Complexity measure: memory $\times$ time

What's the best we can hope for?

H: $\{0,1\}^* \to \{0,1\}^w$ random oracle



Upper bound on cc(scrypt):

The naïve algorithm stores every x_i value.

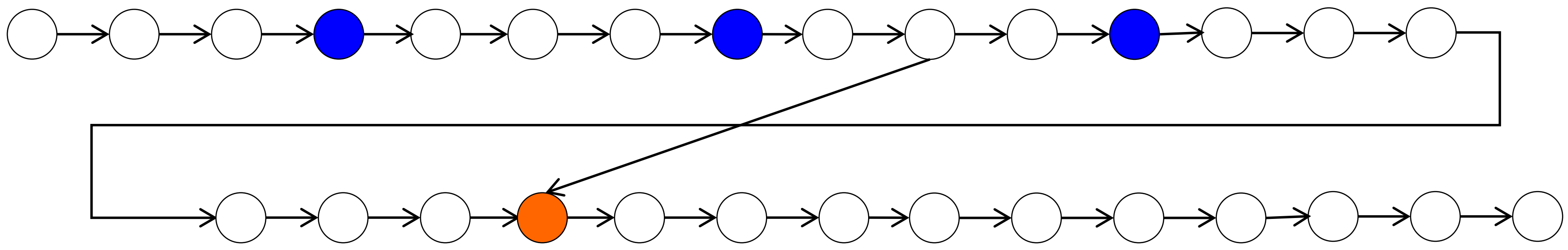
Time: 2n. Memory: $\leq$n. Total: $\leq 2n^2$ (in w-bit units).

Note: any function that has an n-step sequential algorithm has cc $\leq n^2/2$ (because memory $\leq$ time)

No function so far has been proven to have cc of n^2

(several candidates were proposed during password-hashing competition 2013-15; some have been broken)

Data-Independent Memory Hard Functions



Observation: any function whose memory access pattern
is independent of the input
can be represented as a fixed graph

Sequential algorithm of time n $\Rightarrow$ n nodes

Term: iMHF (Data-independent Memory Hard Function)

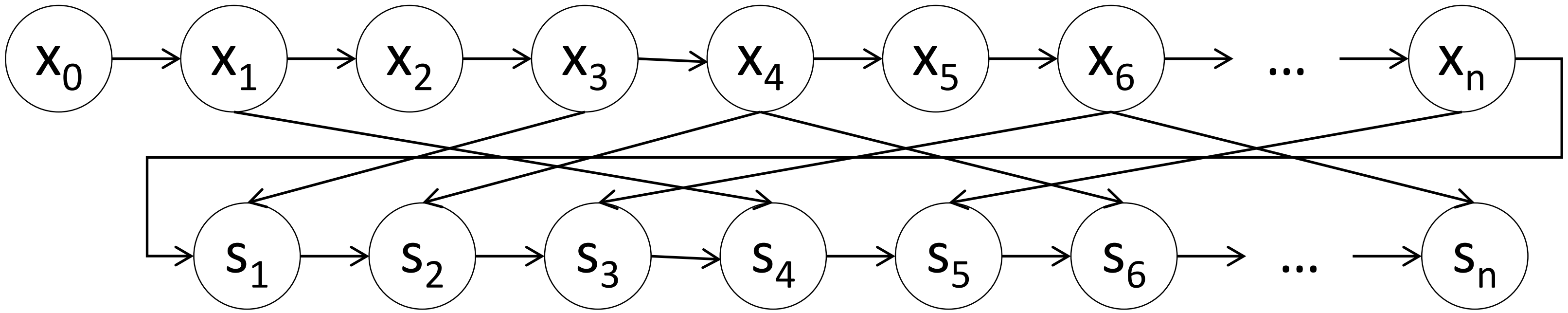
[Alwen-Blocki 16]: for any iMHF, cc $\leq n^2 \log \log n / \log n$

scrypt is a very simple dMHF

Q: can scrypt beat this iMHF bound?

Our Result

H: $\{0,1\}^* \rightarrow \{0,1\}^w$ random oracle



<u>Theorem</u>: in the parallel RO model, cc(scrypt) = Ω (n^2)

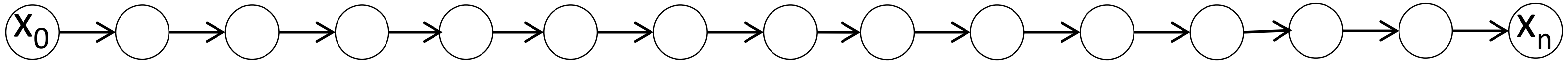
The first ever construction works!

Talk Outline

- ✓ Memory-hard functions for password hashing
- ✓ Design of scrypt
- ✓ How to measure cost: cumulative complexity (cc)
- ✓ Main Result: cc(scrypt) is highest possible n^2 in parallel RO model

Before proving: can we simplify scrypt?

How quickly can you play this game?



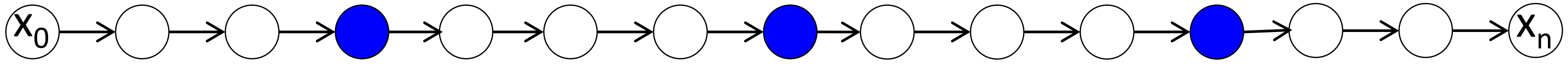
You have x_0 and whatever storage you want

I give you uniform challenge c from 1 to n

You return x_c

If you store nothing but x_0: n/2 H-queries per challenge

How quickly can you play this game?



You have x_0 and whatever storage you want

I give you uniform challenge c from 1 to n

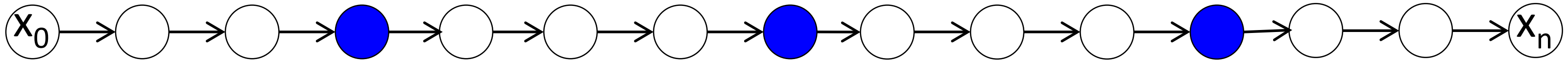
You return x_c

If you store nothing but x_0: n/2 H-queries per challenge

If you store p hash values: n/(2p) H-queries per challenge

If you store something other than hash values?

Result for the scrypt one-shot game



You have x_0 and whatever storage you want

I give you uniform challenge i from 1 to n

You return x_i

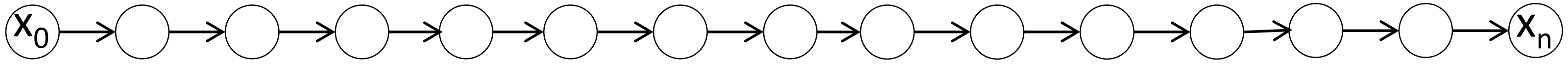
Prior result 1: if you store p <u>labels</u>, expected time $\geq n/(2p)$

Prior result 2 [Alwen Chen Kamath Kolmogorov Pietrzak Tessaro '16]:
same if you store <u>"entangled" labels</u>
(such as XOR or more general linear functions)

but not portions of labels, XORs of portions, etc.

Our result: same for arbitrary storage of pw bits!
(where w is label length = output length of H)

Claim: time $\geq n/(2p)$ if storage pw

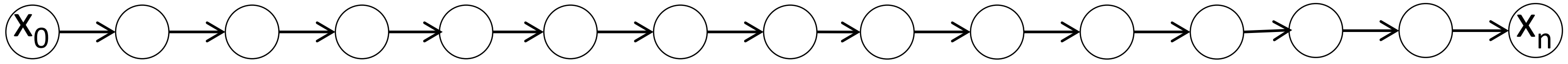


Basic idea of the argument (inspired by [Alwen-Serbinenko]):

if A is too fast, then
we can extract many labels from A's storage w/o querying H

but can't extract more than p labels b/c RO not compressible

Extracting labels from A's memory



Imagine: run A on every possible challenge and record queries

…

c=23

$x_5 \to x_6$, $x_{14} \to x_{15}$

$x_6 \to x_7$, $x_{15} \to x_{16}$

$x_{22} \to x_{23}$

c=24

$x_5 \to x_6$, $x_{14} \to x_{15}$

$x_6 \to x_7$, $x_{15} \to x_{16}$

$x_{22} \to x_{23}$

$x_{23} \to x_{24}$

c=25

$x_{21} \to x_{22}$, $x_{12} \to x_{13}$

$x_{13} \to x_{14}$

$x_7 \to x_8$

$x_{24} \to x_{25}$

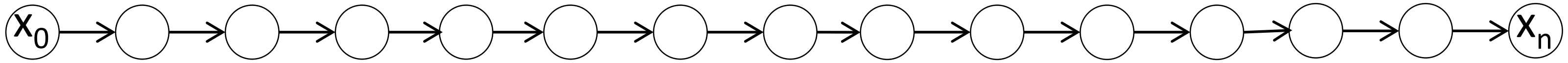
c=26

$x_{30} \to x_{31}$, $x_5 \to x_6$

x_{26}

…

Extracting labels from A's memory



Mark blue any label whose <u>earliest</u> appearance is not from H

…

c=23

- $x_5 \rightarrow x_6$, $x_{14} \rightarrow x_{15}$
- $x_6 \rightarrow x_7$, $x_{15} \rightarrow x_{16}$
- $x_{22} \rightarrow x_{23}$

c=24

- $x_5 \rightarrow x_6$, $x_{14} \rightarrow x_{15}$
- $x_6 \rightarrow x_7$, $x_{15} \rightarrow x_{16}$
- $x_{22} \rightarrow x_{23}$
- $x_{23} \rightarrow x_{24}$

c=25

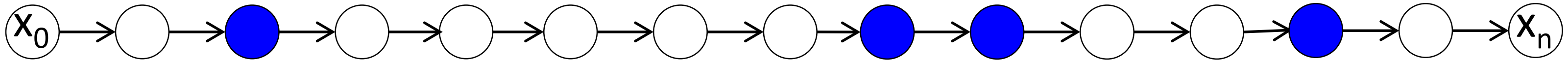
- $x_{21} \rightarrow x_{22}$, $x_{12} \rightarrow x_{13}$
- $x_{13} \rightarrow x_{14}$
- $x_7 \rightarrow x_8$
- $x_{24} \rightarrow x_{25}$

c=26

- $x_{30} \rightarrow x_{31}$, $x_5 \rightarrow x_6$
- x_{26}

…

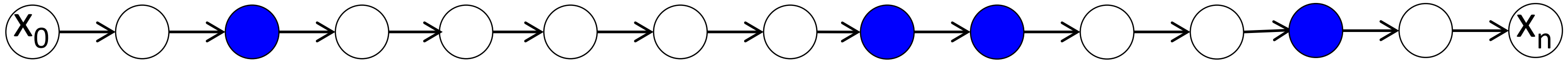
Extracting labels from A's memory



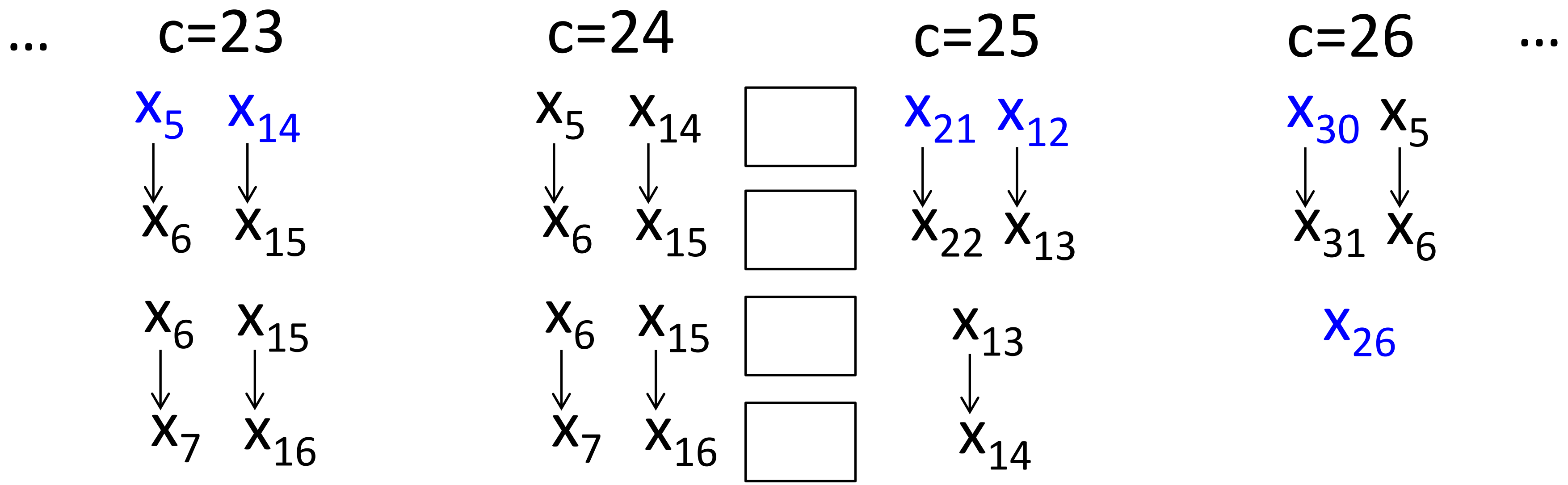
Mark blue any label whose <u>earliest</u> appearance is not from H

…	c=23	c=24	c=25	c=26	…
	$x_5 \to x_6$ $x_{14} \to x_{15}$	$x_5 \to x_6$ $x_{14} \to x_{15}$	$x_{21} \to x_{22}$ $x_{12} \to x_{13}$	$x_{30} \to x_{31}$ $x_5 \to x_6$	
	$x_6 \to x_7$ $x_{15} \to x_{16}$	$x_6 \to x_7$ $x_{15} \to x_{16}$	$x_{13} \to x_{14}$	x_{26}	
	$x_{22} \to x_{23}$	$x_{22} \to x_{23}$	$x_7 \to x_8$		
		$x_{23} \to x_{24}$	$x_{24} \to x_{25}$		

Extracting labels from A's memory



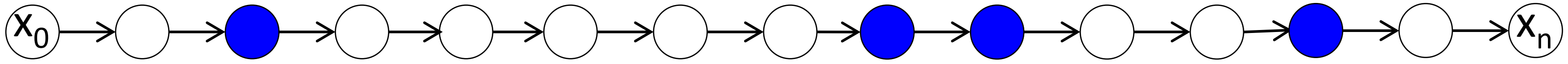
Mark blue any label whose earliest appearance is not from H



Lemma 1: all blue labels can be extracted from memory of A without querying H

Proof: Make a predictor for H that runs A in parallel on all challenges, one step at a time, predicting blue values by querying H only when needed

Extracting labels from A's memory

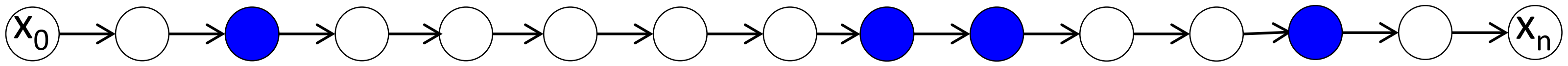


Mark blue any label whose <u>earliest</u> appearance is not from H

…	c=23	c=24	c=25	c=26	…
	$x_5 \to x_6$, $x_{14} \to x_{15}$	$x_5 \to x_6$, $x_{14} \to x_{15}$	$x_{21} \to x_{22}$, $x_{12} \to x_{13}$	$x_{30} \to x_{31}$, $x_5 \to x_6$	
	$x_6 \to x_7$, $x_{15} \to x_{16}$	$x_6 \to x_7$, $x_{15} \to x_{16}$	$x_{13} \to x_{14}$	x_{26}	

Lemma 1: all blue labels can be extracted from memory of A without querying H (so |blue set| ≤ |memory|/w)

Extracting labels from A's memory

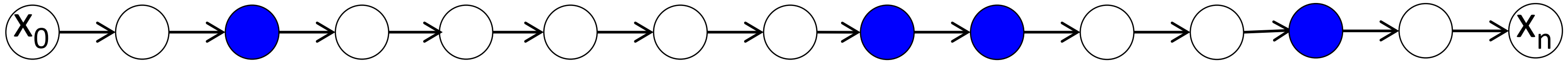


Mark blue any label whose earliest appearance is not from H

…	c=23	c=24	c=25	c=26	…
	$x_5 \to x_6$ $x_{14} \to x_{15}$	$x_5 \to x_6$ $x_{14} \to x_{15}$	$x_{21} \to x_{22}$ $x_{12} \to x_{13}$	$x_{30} \to x_{31}$ $x_5 \to x_6$	
	$x_6 \to x_7$ $x_{15} \to x_{16}$	$x_6 \to x_7$ $x_{15} \to x_{16}$	$x_{13} \to x_{14}$	x_{26}	

Lemma 1: all blue labels can be extracted from memory of A without querying H (so |blue set| $\leq$ pw/w)

Extracting labels from A's memory

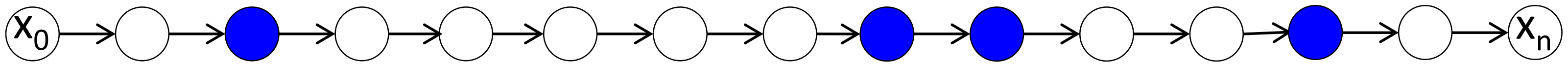


Mark blue any label whose <u>earliest</u> appearance is not from H

…	c=23	c=24	c=25	c=26	…
	$x_5 \to x_6$, $x_{14} \to x_{15}$	$x_5 \to x_6$, $x_{14} \to x_{15}$	$x_{21} \to x_{22}$, $x_{12} \to x_{13}$	$x_{30} \to x_{31}$, $x_5 \to x_6$	
	$x_6 \to x_7$, $x_{15} \to x_{16}$	$x_6 \to x_7$, $x_{15} \to x_{16}$	$x_{13} \to x_{14}$	x_{26}	

Lemma 1: all blue labels can be extracted from memory of A without querying H (so $|\text{blue set}| \le p$)

memory pw $\Rightarrow$ time $\geq n/(2p)$



Mark blue any label whose <u>earliest</u> appearance is not from H

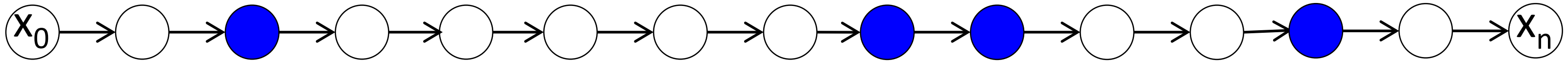
…	c=23	c=24	c=25	c=26	…
	$x_5 \to x_6$ $x_{14} \to x_{15}$	$x_5 \to x_6$ $x_{14} \to x_{15}$	$x_{21} \to x_{22}$ $x_{12} \to x_{13}$	$x_{30} \to x_{31}$ $x_5 \to x_6$	
	$x_6 \to x_7$ $x_{15} \to x_{16}$	$x_6 \to x_7$ $x_{15} \to x_{16}$	$x_{13} \to x_{14}$	x_{26}	

Lemma 1: all blue labels can be extracted from memory of A without querying H (so $|$blue set$| \leq p$)

Lemma 2: Time to answer c $\geq$ distance from nearest blue

Proof: induction

memory pw $\Rightarrow$ time $\geq n/(2p)$



Mark blue any label whose <u>earliest</u> appearance is not from H

…	c=23	c=24	c=25	c=26	…
	$x_5 \to x_6$, $x_{14} \to x_{15}$	$x_5 \to x_6$, $x_{14} \to x_{15}$	$x_{21} \to x_{22}$, $x_{12} \to x_{13}$	$x_{30} \to x_{31}$, $x_5 \to x_6$	
	$x_6 \to x_7$, $x_{15} \to x_{16}$	$x_6 \to x_7$, $x_{15} \to x_{16}$	$x_{13} \to x_{14}$	x_{26}	

Lemma 1: all blue labels can be extracted from memory of A without querying H (so |blue set| $\leq p$)

Lemma 2: Time to answer c $\geq$ distance from nearest blue

<u>Conclusion</u>: storage pw $\Rightarrow$ time $\geq n/(2p)$

Talk Outline

✓Memory-hard functions for password hashing

✓Design of scrypt

✓How to measure cost: cumulative complexity (cc)

✓ scrypt: very simple dMHF (and iMHF won't work)

Main Result: cc(scrypt) is highest possible n^2
in parallel RO model

Proof in two parts

✓1. memory vs. time to answer one random challenge
2. cumulative complexity of n challenges

How to go from this…

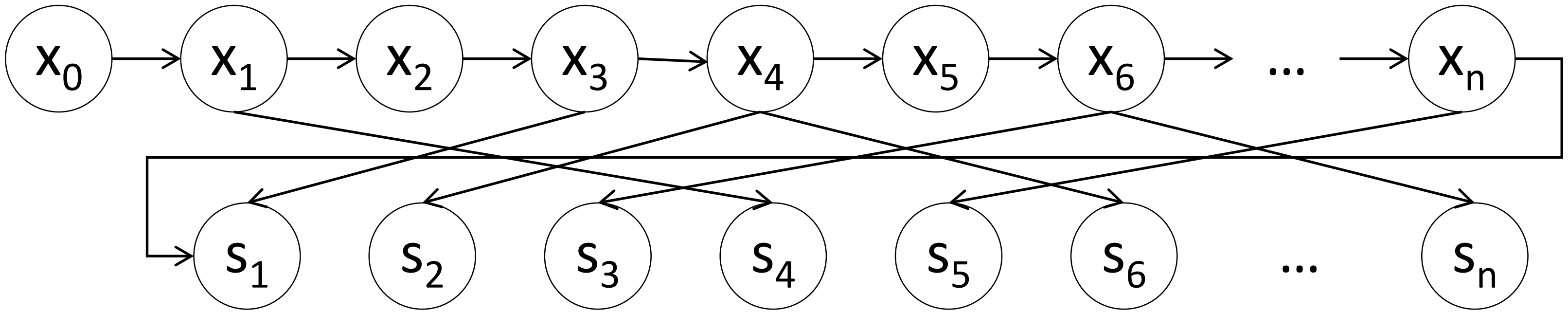
$H: \{0,1\}^* \rightarrow \{0,1\}^w$ random oracle

$$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow x_3 \rightarrow x_4 \rightarrow x_5 \rightarrow x_6 \rightarrow \ \dots \ \rightarrow x_n$$

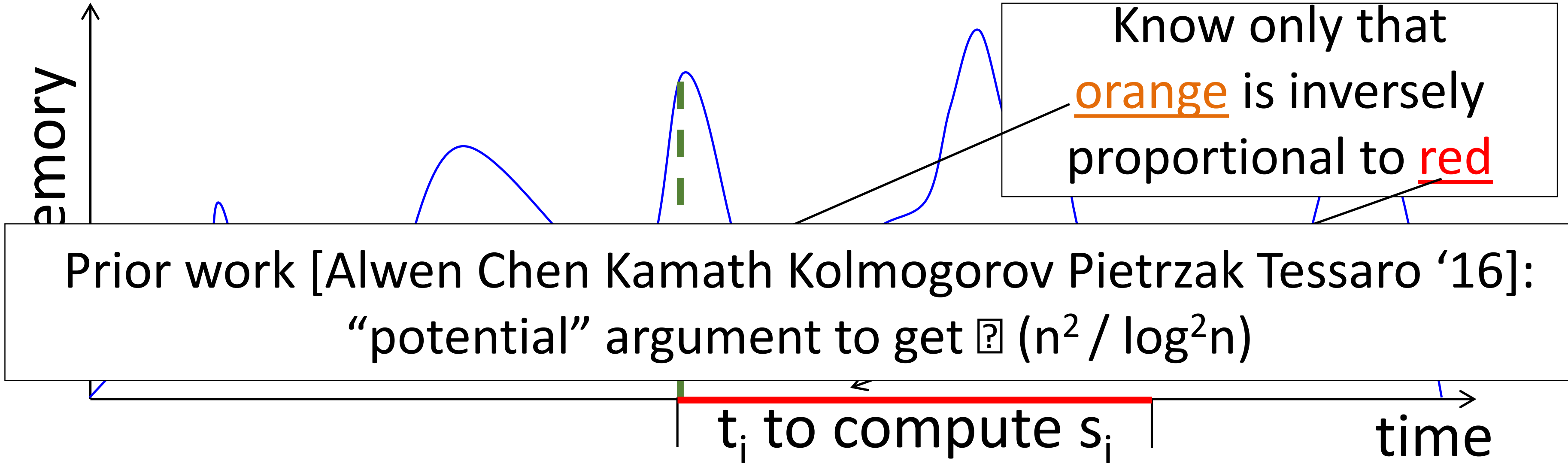
Single random challenge: memory $\geq \dfrac{nw}{2} \bullet \dfrac{1}{\text{time}}$

… to cc(n challenges)

H: $\{0,1\}^* \rightarrow \{0,1\}^w$ random oracle



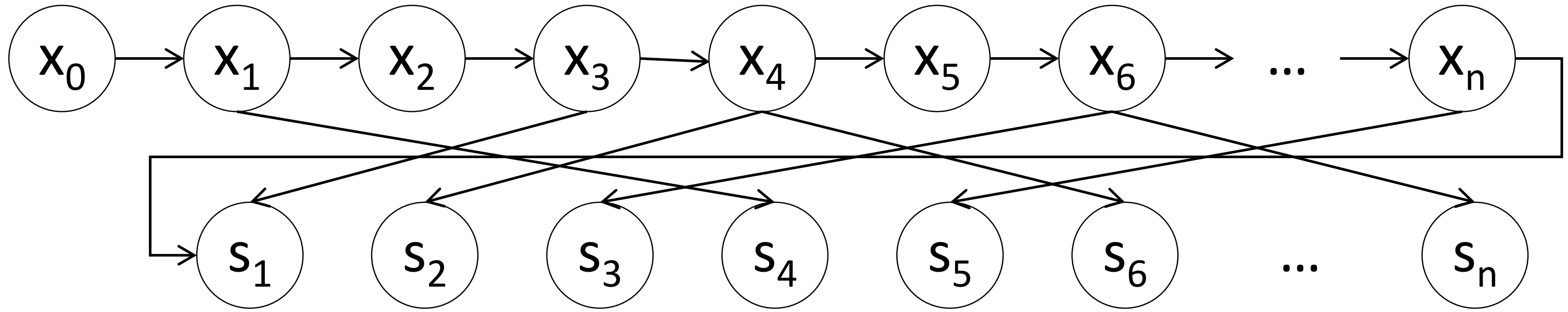
Single random challenge: memory $\geq \dfrac{nw}{2} \bullet \dfrac{1}{\text{time}}$



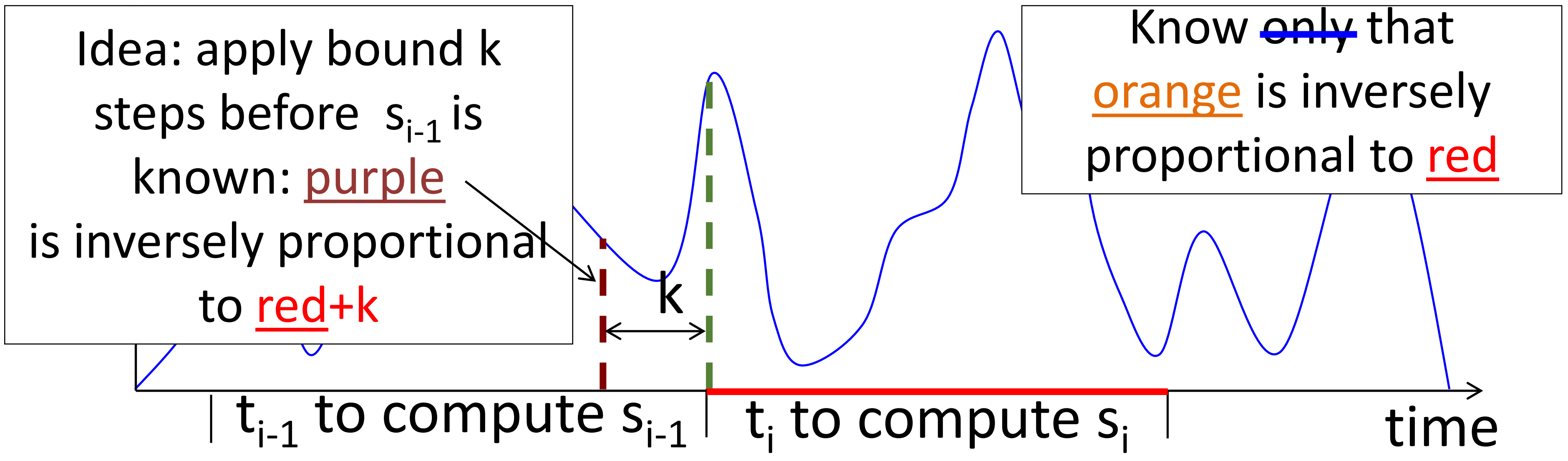
Prior work [Alwen Chen Kamath Kolmogorov Pietrzak Tessaro '16]:
"potential" argument to get $\Omega(n^2 / \log^2 n)$

… to cc(n challenges)

H: $\{0,1\}^* \to \{0,1\}^w$ random oracle

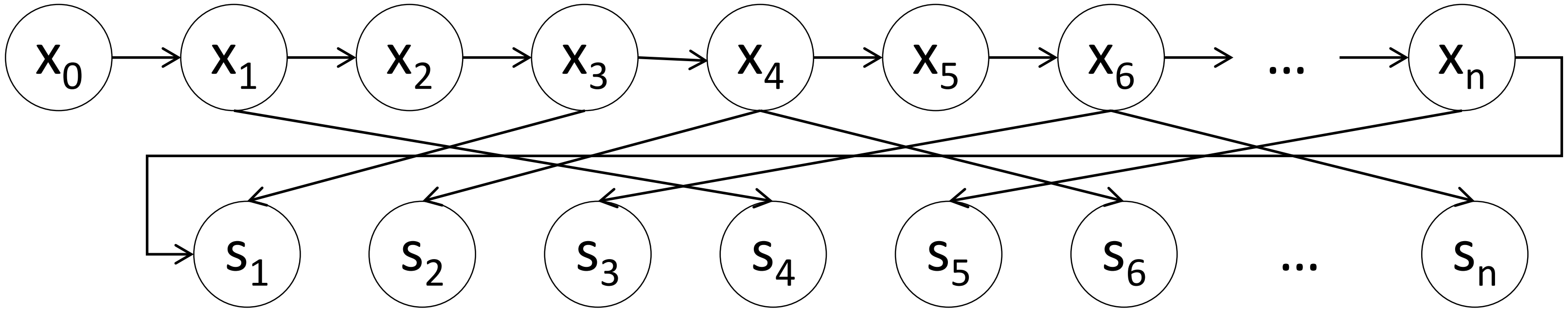


Single random challenge: memory $\geq \frac{nw}{2} \bullet \frac{1}{\text{time}}$

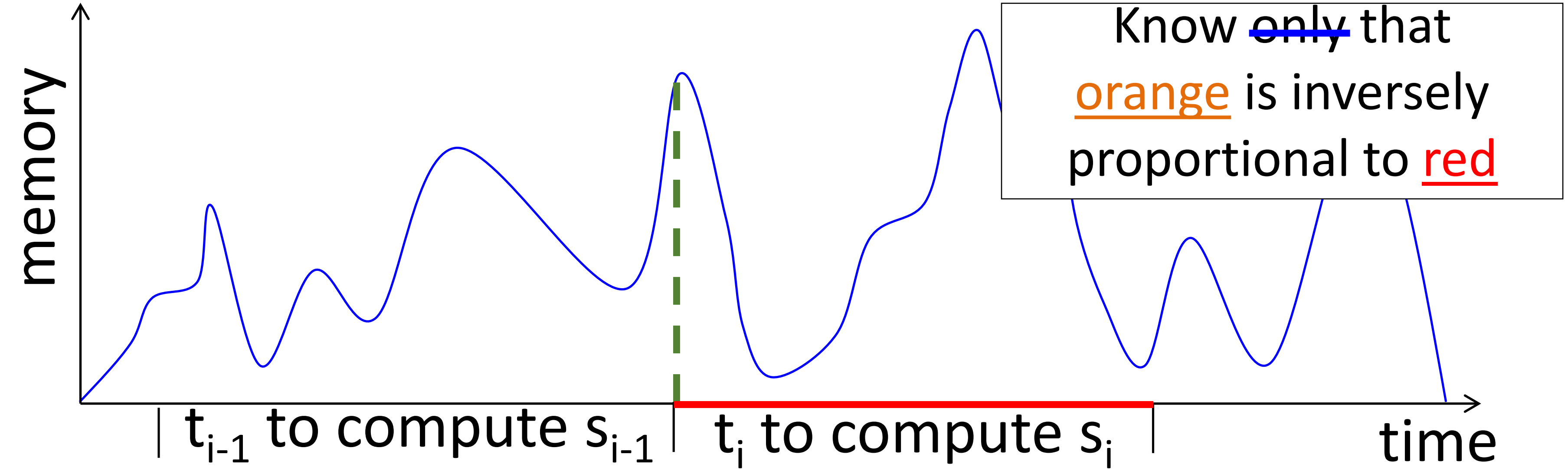


… to cc(n challenges)

$H: \{0,1\}^* \to \{0,1\}^w$ random oracle

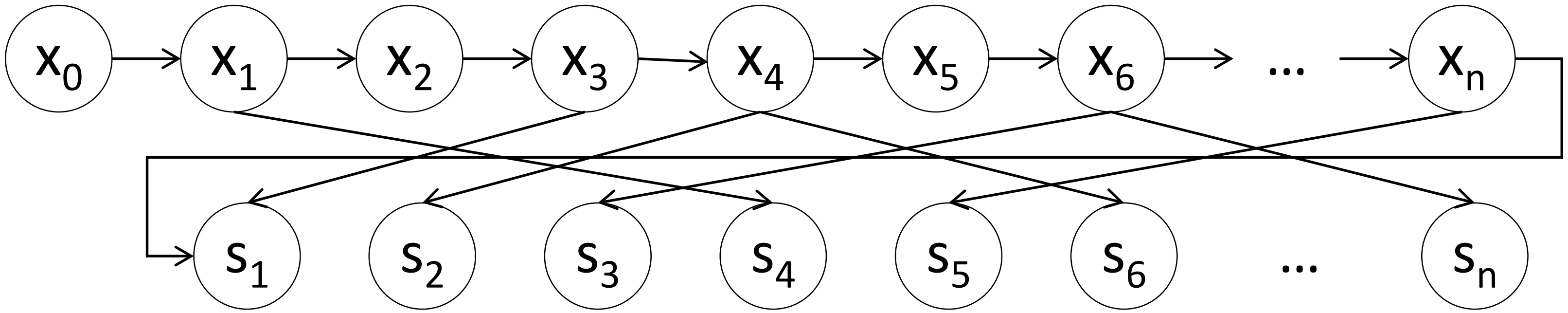


Single random challenge: memory $\geq \frac{nw}{2} \bullet \frac{1}{\text{time}}$

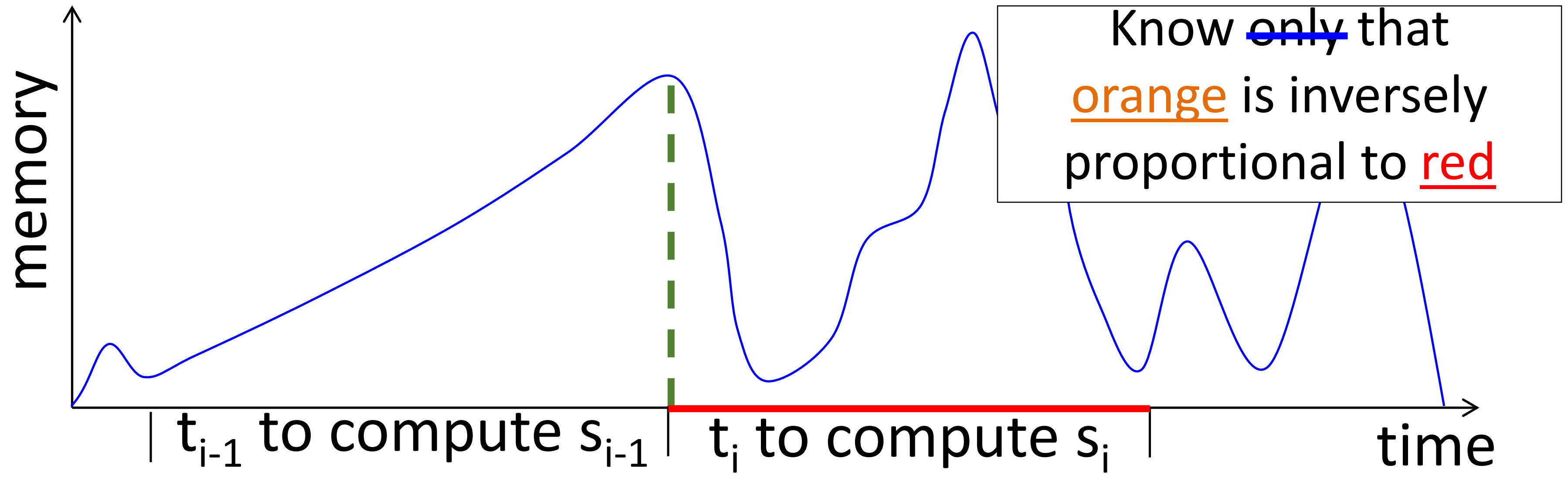


… to cc(n challenges)

H: $\{0,1\}^* \rightarrow \{0,1\}^w$ random oracle



Single random challenge: memory $\geq \frac{nw}{2} \bullet \frac{1}{\text{time}}$

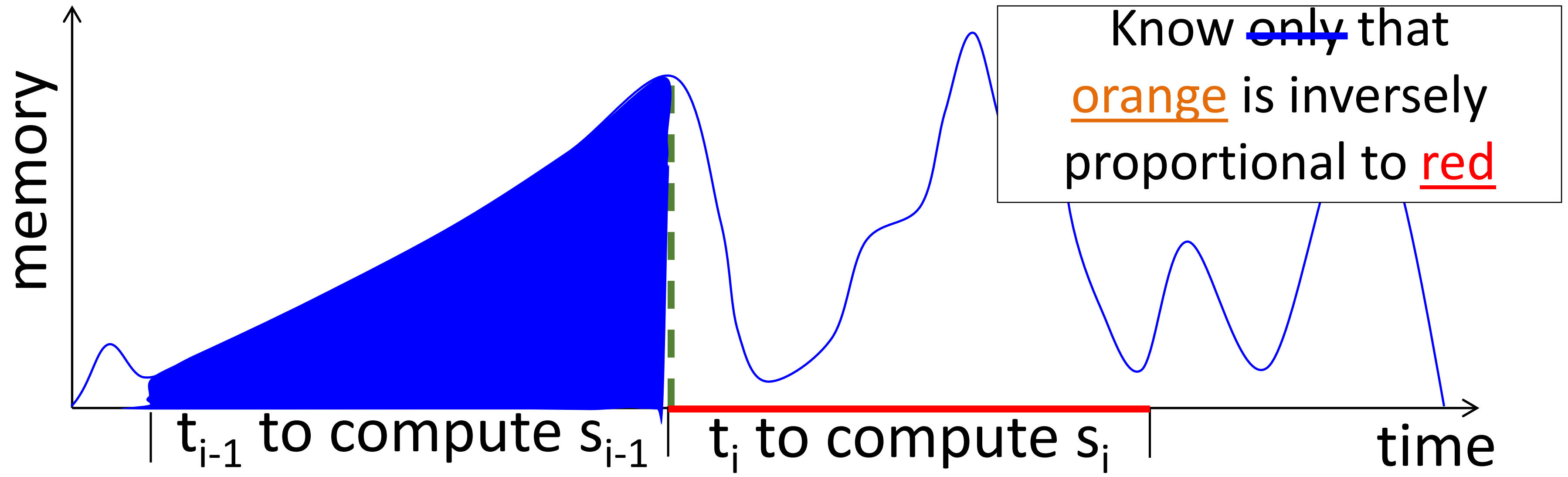


… to cc(n challenges)

Adding up memory used during previous challenge:

$$\frac{nw}{2}\left(\frac{1}{t_i} + \frac{1}{t_i+1} + \cdots + \frac{1}{t_i + t_{i-1}}\right) \geq \frac{nw}{2}\,(\ln\,(t_i+t_{i-1}) - \ln\, t_i)$$

Single random challenge: memory $\geq \frac{nw}{2} \bullet \frac{1}{\text{time}}$



… to cc(n challenges)

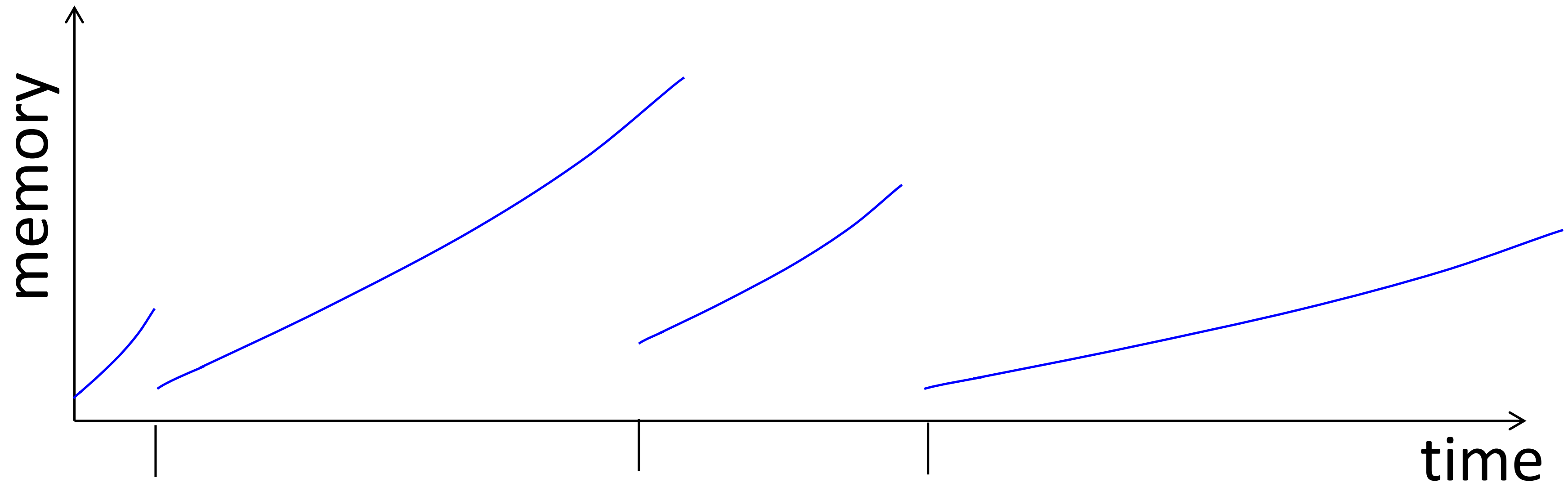
Adding up memory used during previous challenge:

$$\frac{nw}{2}\left(\frac{1}{t_i} + \frac{1}{t_i+1} + \cdots + \frac{1}{t_i + t_{i-1}}\right) \geq \frac{nw}{2}\,(\ln\,(t_i+t_{i-1}) - \ln\, t_i)$$

Adding up over all challenges i from 1 to n:

$$\tfrac{1}{2}nw\,(\,\ln\,(t_1+t_2) - \ln\, t_2 + \ln\,(t_2+t_3) - \ln\, t_3 + \ldots + \ln\,(t_{n-1}+ t_n) - \ln\, t_n)$$

$$\geq \tfrac{1}{2}nw\,(n \ln 2) \geq \Omega(n^2 w)$$



Talk Outline

✓ Memory-hard functions for password hashing

✓ Design of scrypt

✓ How to measure cost: cumulative complexity (cc)

✓ scrypt: very simple dMHF (and iMHF won’t work)

Main Result: cc(scrypt) is highest possible n^2
in parallel RO model

Proof in two parts

✓ 1. memory vs. time to answer one random challenge

✓ 2. cumulative complexity of n challenges

Thanks for Listening