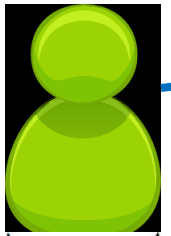
Sustained Space and Cumulative Complexity Trade-offs for Data-Dependent Memory-Hard Functions

Jeremiah Blocki

Blake Holman

Motivation: Password Storage



bholman, 123456

Username	Salt	Hash
bholman	89d978034a3f6	85e23cfe0021f584e3db87aa72630a9a2345c062

SHA1(12345689d978034a3f6)=85e23cfe0021f584e3db87aa72630a9a2345c062

Offline Attacks: A Common Problem

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

TECH

Yahoo Triples Estimate of Breached Accounts to 3 Billion

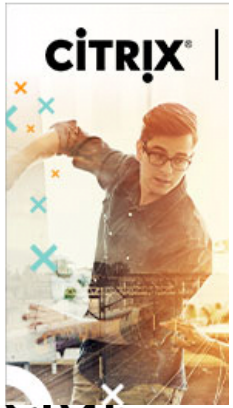
Company disclosed late last year that 2013 hack exposed private information of over 1 billion users

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...



A Dangerous Problem

Can we increase guessing costs for the attacker?

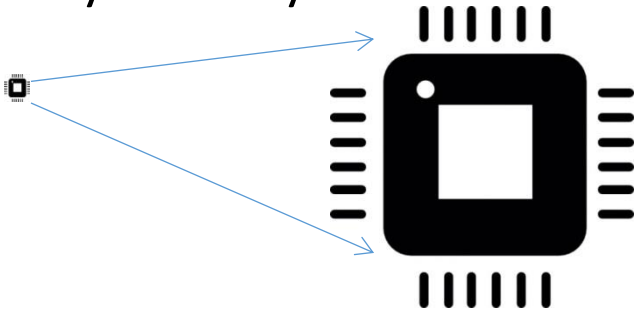
Goal: Moderately Expensive Hash Function

Fast on PC and
Expensive on ASIC?

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

Types of Memory-Hard Functions

- Data-**Independent** Memory-Hard Functions (iMHFs)
  - Memory access pattern during evaluation is **independent** of the input

- Data-**Dependent** Memory-Hard Functions (dMHFs)
  - Memory access pattern during evaluation may **depend** on the input
  - No restrictions as with iMHFs

Data-Independent vs. Data-Dependent MHFs

- Data-Independent:
  - Side-channel resistant
  - Weaker memory hardness

- Data-Dependent:
  - Possible side-channel leakage
  - Simple constructions
  - **Provably better memory hardness**

Why Data-Dependent MHFs?

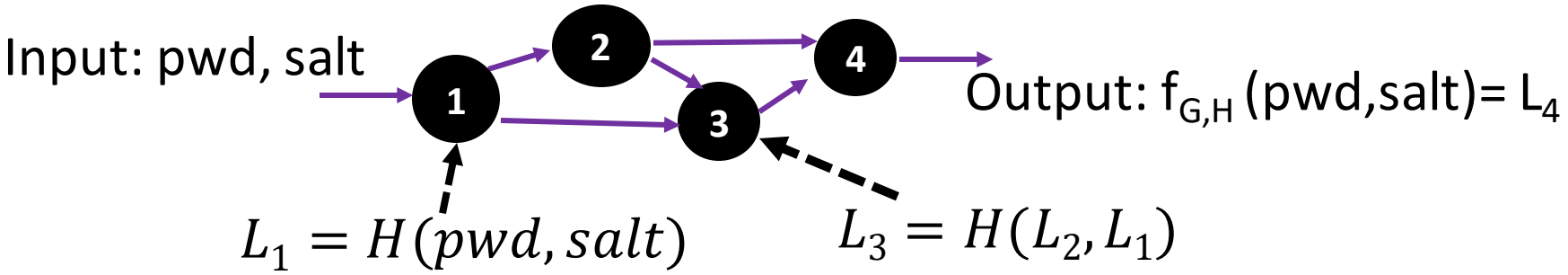
- Uses where leakage isn't a concern
  - Securing blockchains
  - Litecoin/Dogecoin

- Combine with iMHFs
  - Data-independent phases give baseline memory hardness

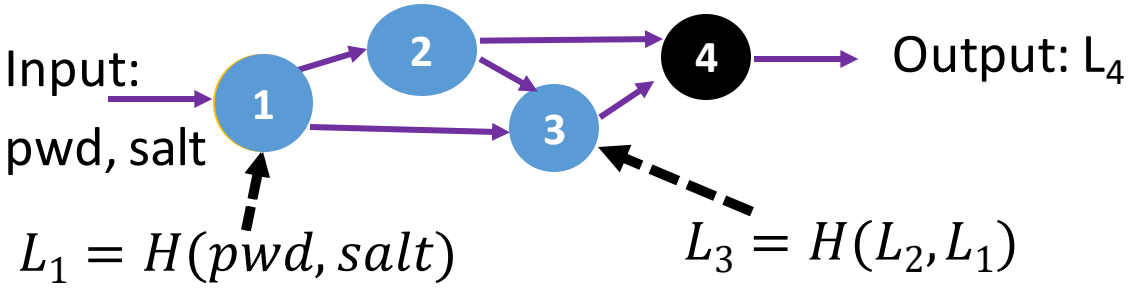
Evaluating iMHF ($f_{G,H}$)

Defined by

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



Evaluating an iMHF (pebbling)



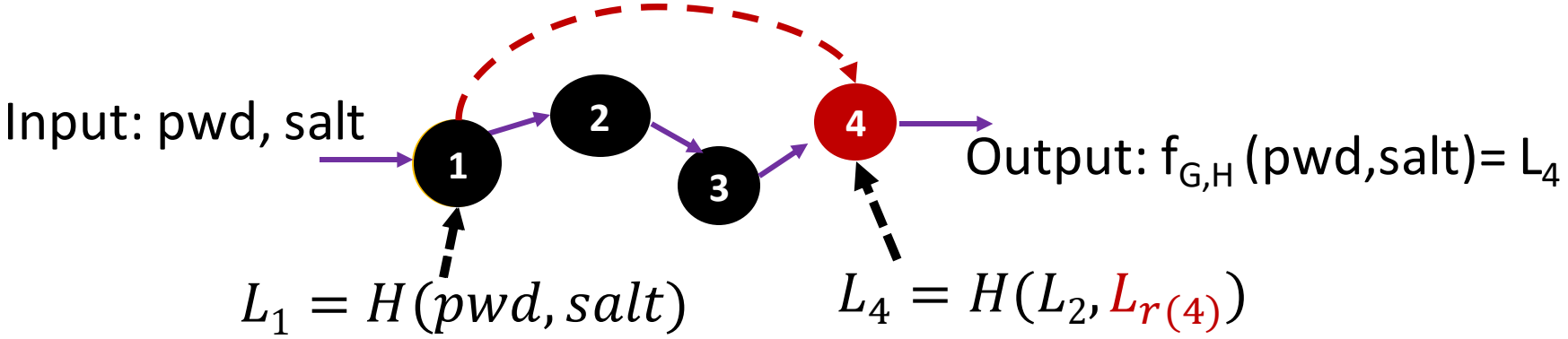
Pebbling Rules : $\vec{P}$=P_1,…,P_t⊂ V s.t.

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

Evaluating ~~iMHF ($f_{G,H}$)~~ dMHF($f_{G,H}$)

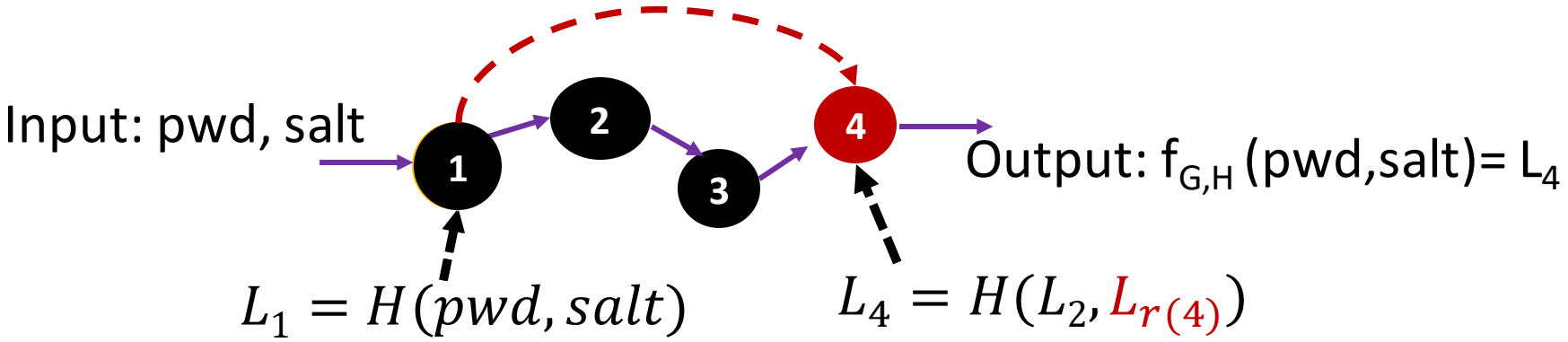
Defined by

- H:$\{0,1\}^{2k} \rightarrow \{0,1\}^{k}$ (Random Oracle)
- DAG Distribution $\mathbb{G}$ (Dynamic Pebbling Graph)
- DAG $G \sim \mathbb{G}$
  - Random edges appear



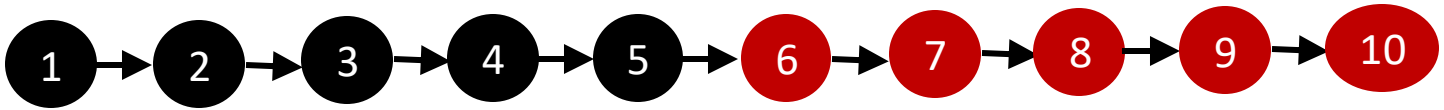
Data-Dependent Memory Hard Functions

- Each random edge $(r(i), i)$ is chosen using random label of its predecessor $i - 1$.
- Example: $r(4) = L_3 \bmod 2 = H(L_2) \bmod 2$
  - Upon pebbling node 3, we discover the edge $(1,4)$



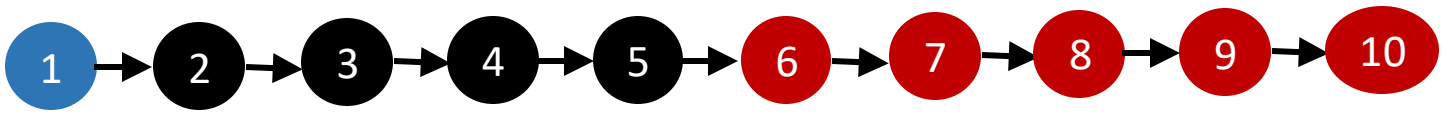
Case Study: Scrypt

- First half line graph
- Second half has dynamic edges coming from the first half
- N=5
- Each node $i \geq N/2$ has a random edge $r(i)$ from the first half
- Here, $L_i = H(L_{i-1}, L_{L_{i-1} \bmod N/2})$



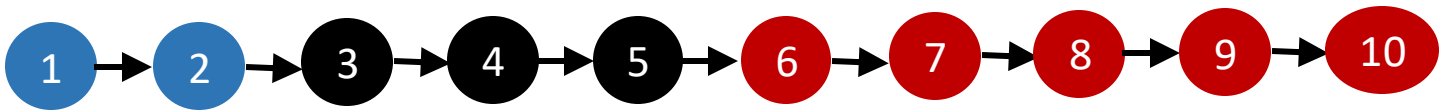
Scrypt: Naïve Pebbling Strategy

- Idea: pebble as many nodes as possible
  - Never take any pebbles of the graph “just in case”



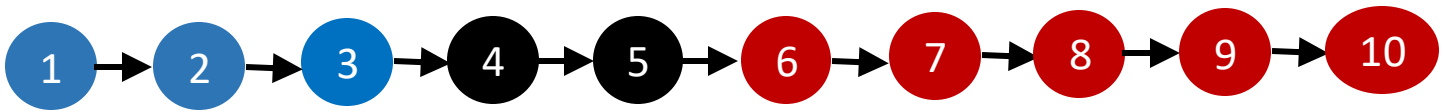
Scrypt: Naïve Pebbling Strategy

- Idea: pebble as many nodes as possible
  - Never take any pebbles of the graph “just in case”



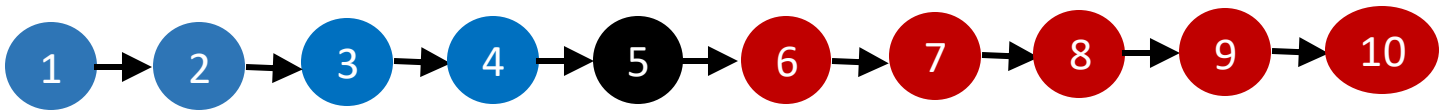
Scrypt: Naïve Pebbling Strategy

- Idea: pebble as many nodes as possible
  - Never take any pebbles of the graph “just in case”



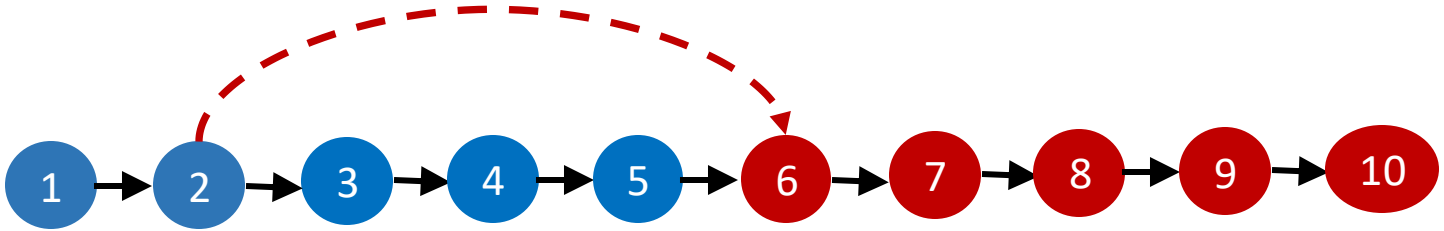
Scrypt: Naïve Pebbling Strategy

- Idea: pebble as many nodes as possible
  - Never take any pebbles of the graph “just in case”



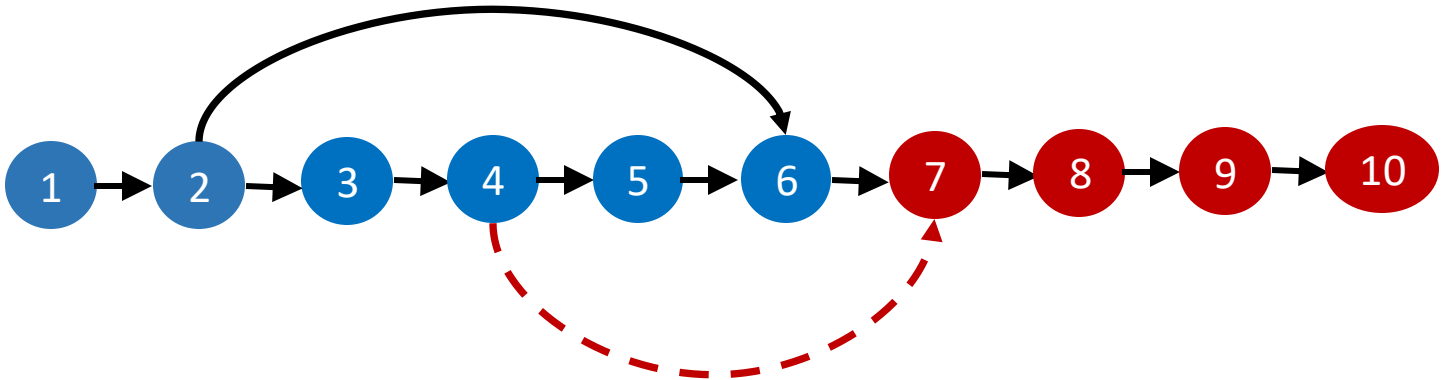
Scrypt: Naïve Pebbling Strategy

- Idea: pebble as many nodes as possible
  - Never take any pebbles of the graph “just in case”



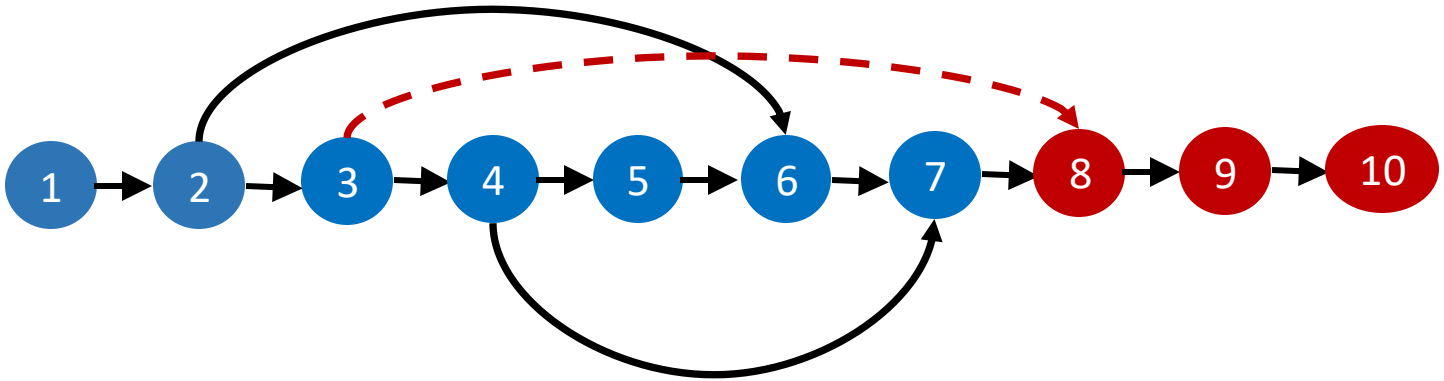
Scrypt: Naïve Pebbling Strategy

- Idea: pebble as many nodes as possible
  - Never take any pebbles of the graph “just in case”



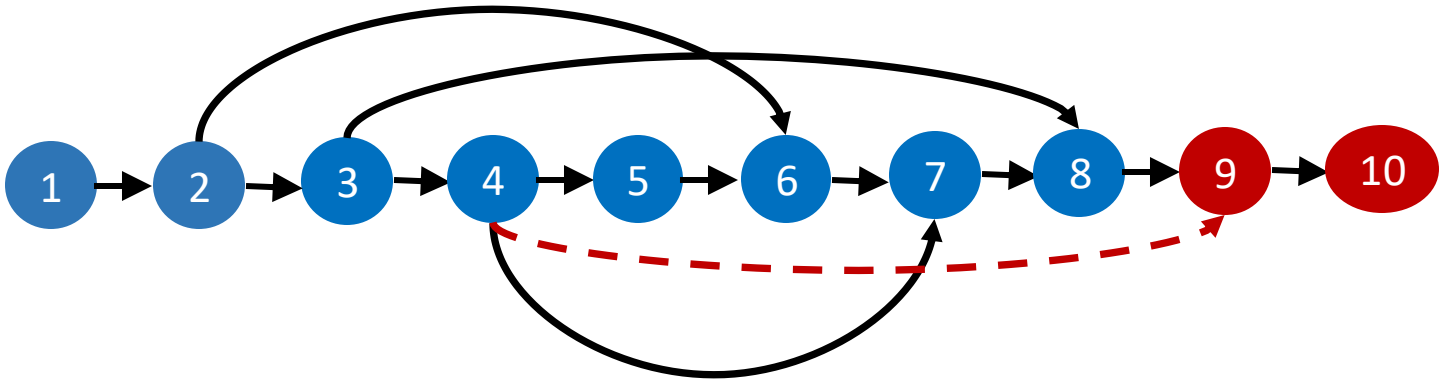
Scrypt: Naïve Pebbling Strategy

- Idea: pebble as many nodes as possible
  - Never take any pebbles of the graph “just in case”



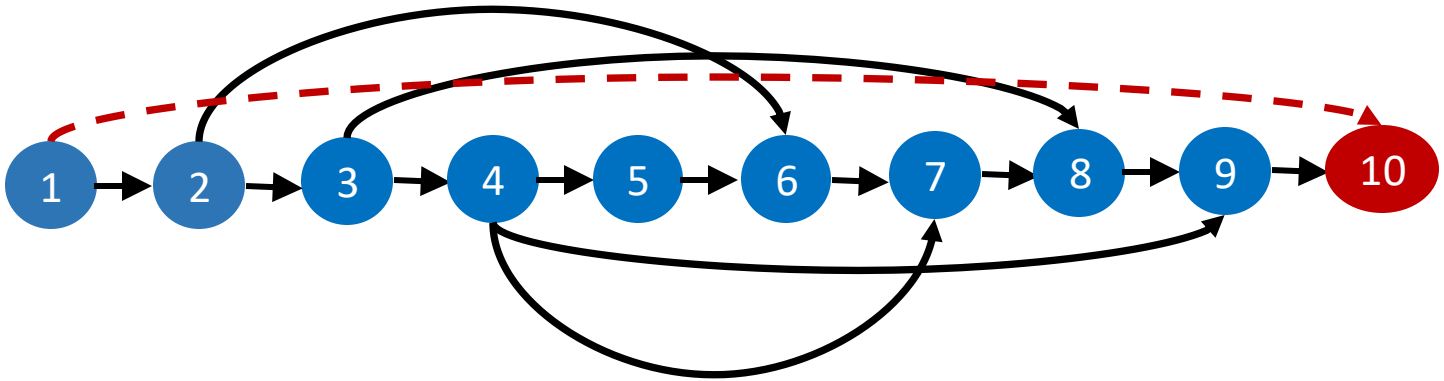
Scrypt: Naïve Pebbling Strategy

- Idea: pebble as many nodes as possible
  - Never take any pebbles of the graph “just in case”



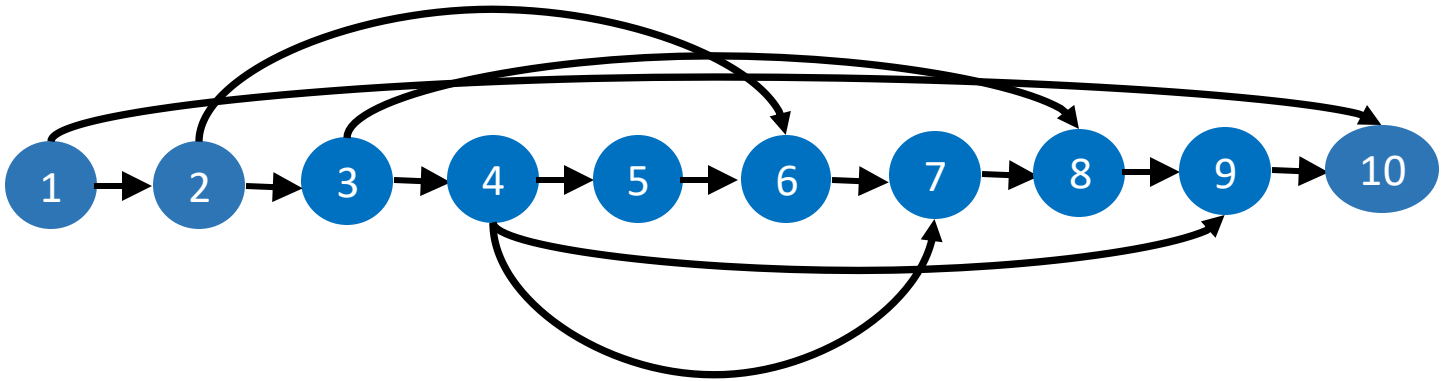
Scrypt: Naïve Pebbling Strategy

- Idea: pebble as many nodes as possible
  - Never take any pebbles of the graph “just in case”



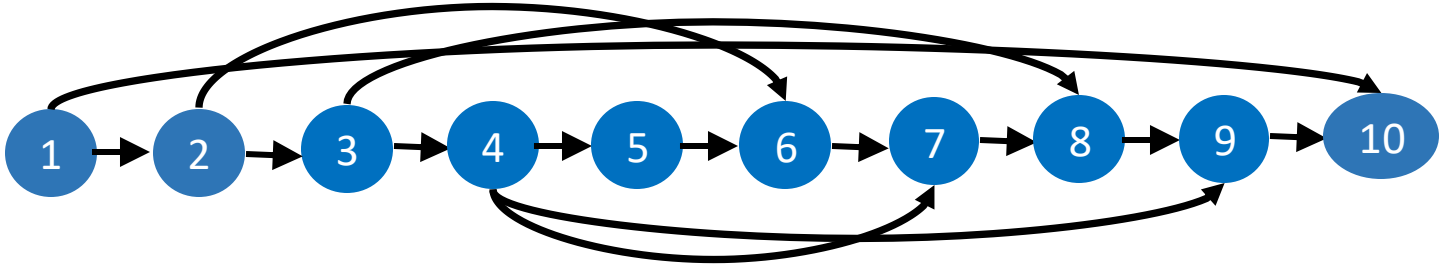
Scrypt: Naïve Pebbling Strategy

- Question: How memory hard is Scrypt?



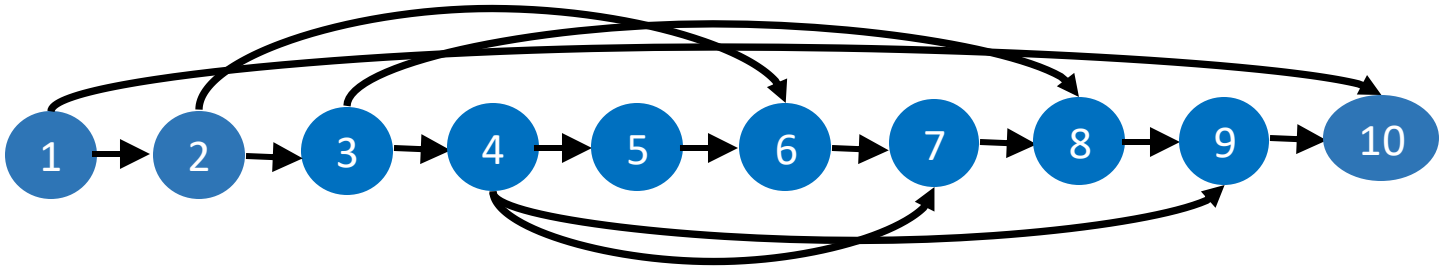
Measuring Cost Attempt 1: Cumulative Complexity

- Idea: we want to capture how the memory is held-up during the computation
- $cc(P) = \sum|P_i|$
- $cc(G) = \min_{P \in \mathcal{P}}\{cc(P)\}$
- Strategy S is an algorithm which outputs a pebbling $S(G) = P$ for $G \sim \mathbb{G}$
- $cc(\mathbb{G}) = \min_{S \in \mathcal{S}}\{\mathbb{E}[cc(S(G))]\}$
- Naïve pebbling strategy:
  - Cumulative Complexity $O(N^2)$ is always possible



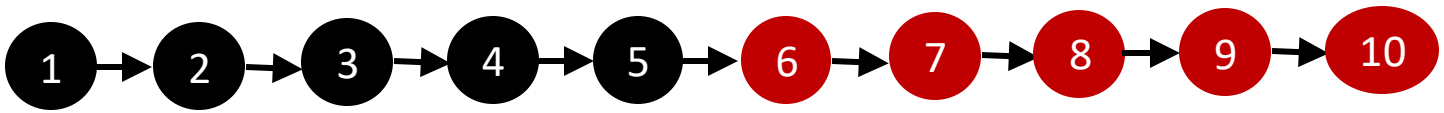
Cumulative Complexity (CC): iMHFs vs. dMHFs

- Static pebbling graphs (iMHFs) have weaker CC guarantees than dynamic pebbling graphs (dMHFs)
  - Any static DAG has cumulative complexity $O\left(\frac{N^2 \log\log N}{N}\right)$ [ABP17]
  - It's easy to construct a dynamic pebbling graph (dMHF) with cumulative complexity $\Omega(N^2)$



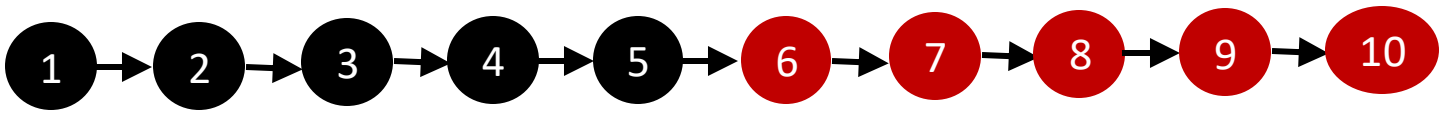
Does Cumulative Complexity Capture Memory Hardness?

- Scrypt has maximal cumulative complexity $\Omega(N^2)$! [ACP+17]
- Unfortunately, cumulative complexity doesn't always imply high memory usage



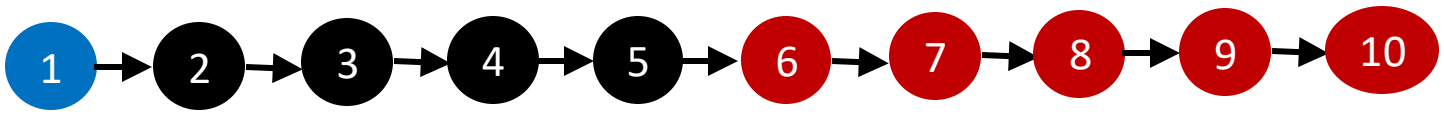
Scrypt: Low Memory Strategy

- Idea: only keep a minimal amount of pebbles



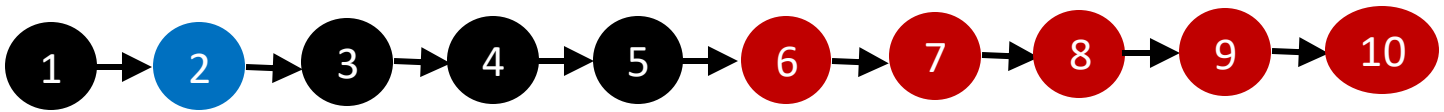
Scrypt: Low Memory Strategy

- Idea: only keep a minimal amount of pebbles



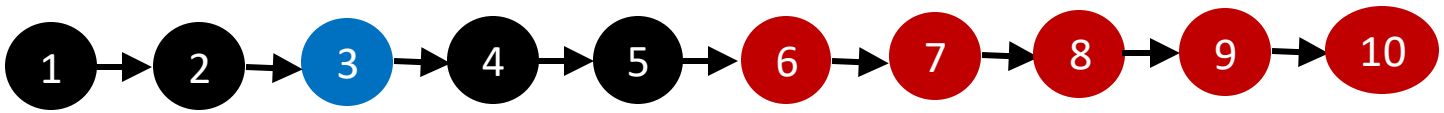
Scrypt: Low Memory Strategy

- Idea: only keep a minimal amount of pebbles



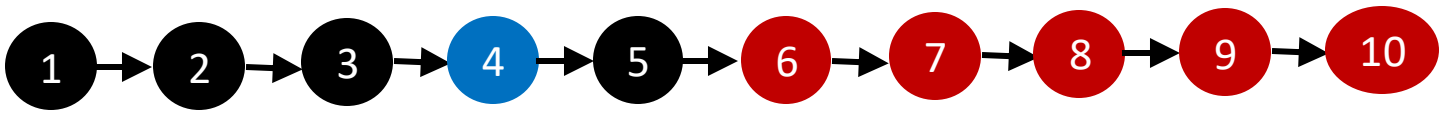
Scrypt: Low Memory Strategy

- Idea: only keep a minimal amount of pebbles



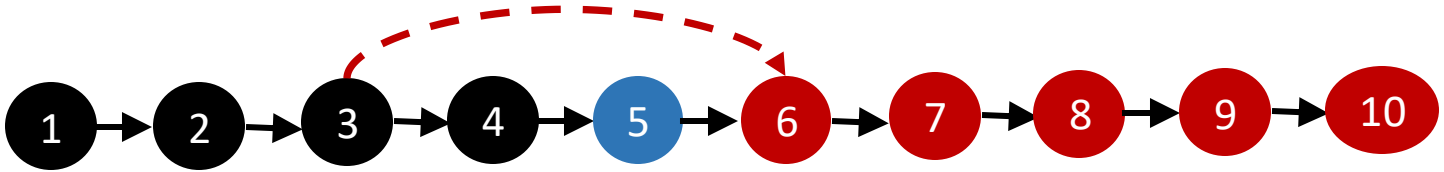
Scrypt: Low Memory Strategy

- Idea: only keep a minimal amount of pebbles



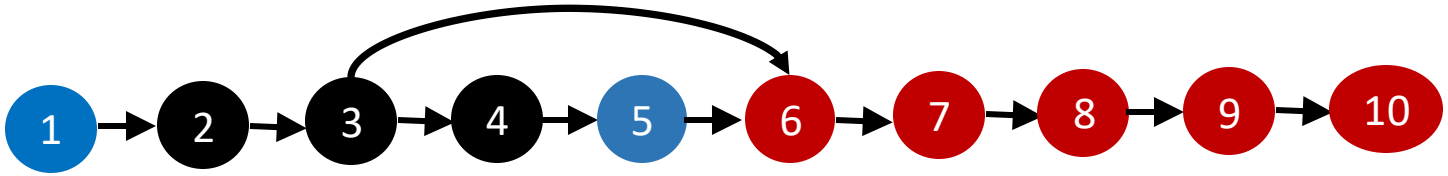
Scrypt: Low Memory Strategy

- Idea: only keep a minimal amount of pebbles



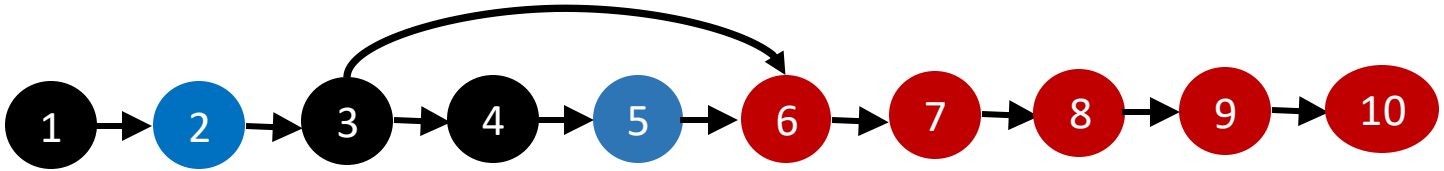
Scrypt: Low Memory Strategy

- Idea: only keep a minimal amount of pebbles



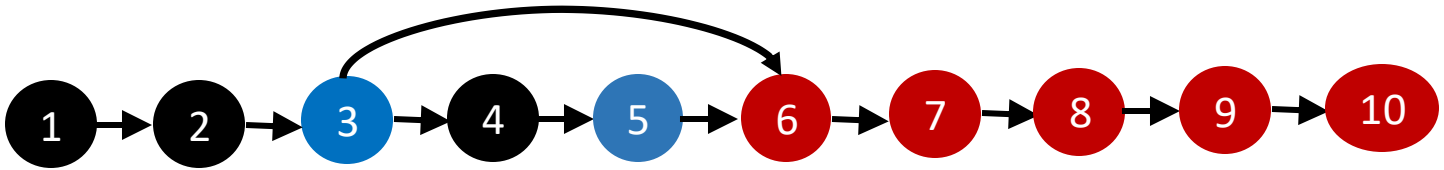
Scrypt: Low Memory Strategy

- Idea: only keep a minimal amount of pebbles



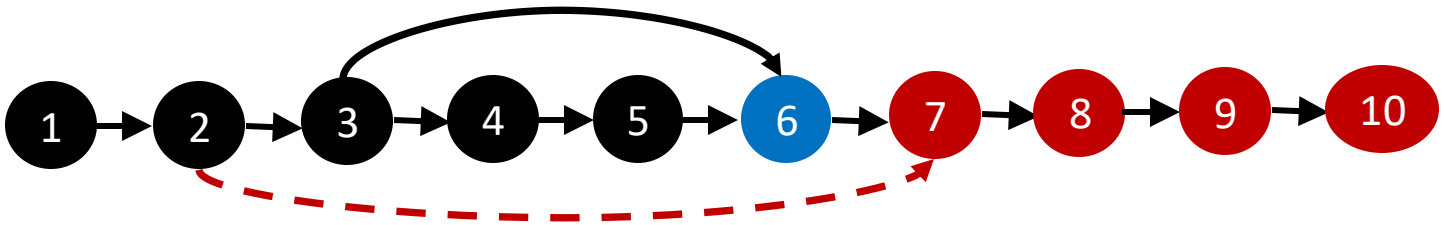
Scrypt: Low Memory Strategy

- Idea: only keep a minimal amount of pebbles



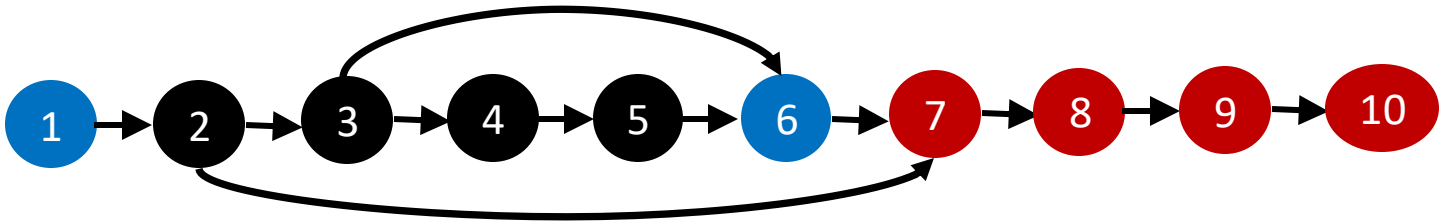
Scrypt: Low Memory Strategy

- Idea: only keep a minimal amount of pebbles



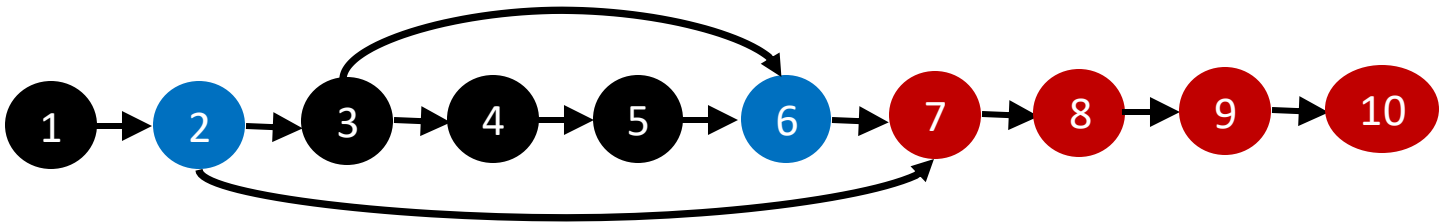
Scrypt: Low Memory Strategy

- Idea: only keep a minimal amount of pebbles



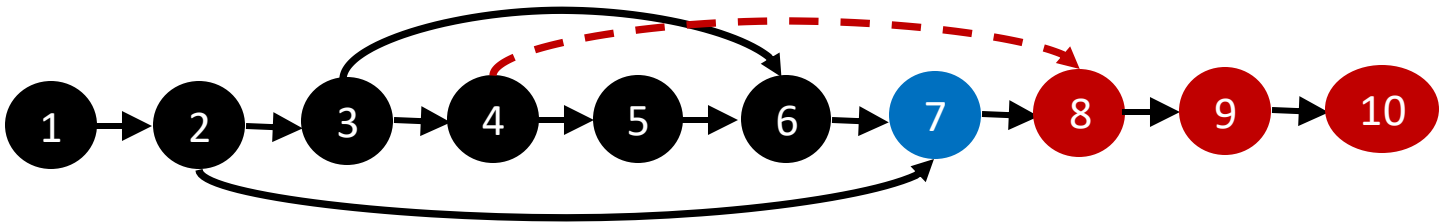
Scrypt: Low Memory Strategy

- Idea: only keep a minimal amount of pebbles



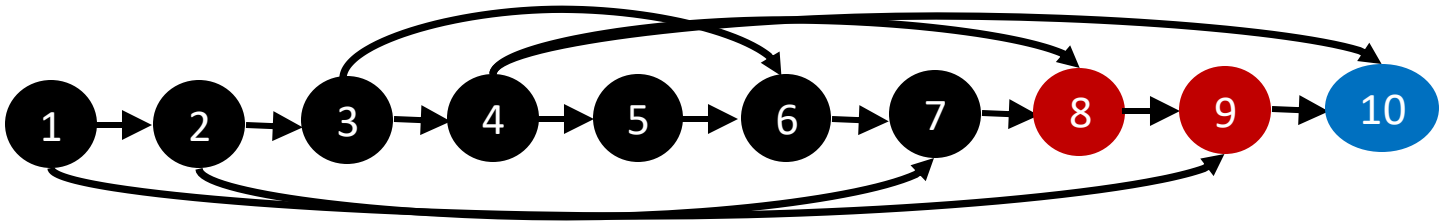
Scrypt: Low Memory Strategy

- Idea: only keep a minimal amount of pebbles



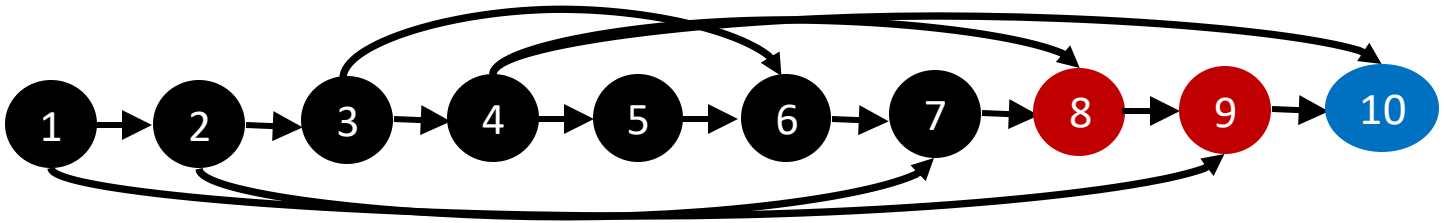
Scrypt: Low Memory Strategy

- Idea: only keep a minimal amount of pebbles



Scrypt: Low Memory Strategy

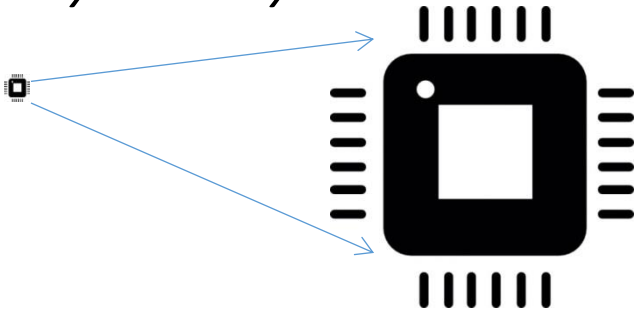
- Idea: only keep a minimal amount of pebbles
- CC is still $O(N^2)$ but we used $O(1)$ space!



Recall: Memory Hard Function (MHF)

- *Intuition: computation costs dominated by memory costs*

vs.

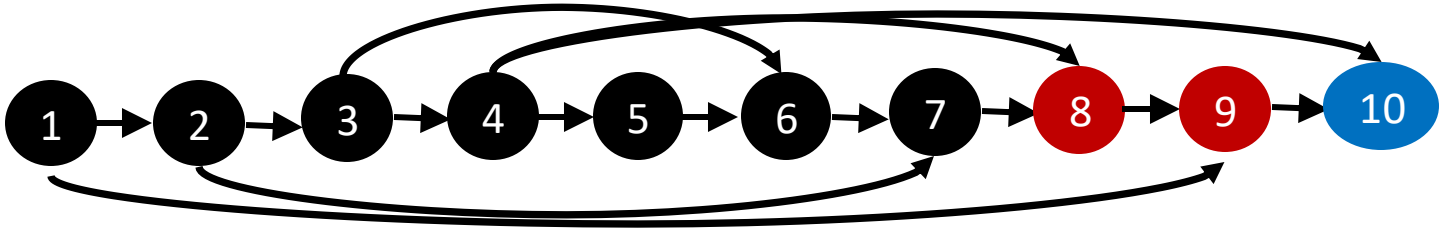


- ***Goal:*** *force attacker to lock up large amounts of memory for duration of computation*

  →Expensive even on customized hardware

Memory Cost Attempt 2: Sustained Space Complexity (SSC)

- s-Sustained Space: the number of rounds in which at least s pebbles on the graph [ABP17]
- s-SSC$(P) = \sum|\{i: |P_i| \geq s\}|$
- Stronger than CC
  - If a pebbling with S-SSC of T has CC at least ST
- Scrypt has s-Sustained Space of 0 for any $s > 2$
- Is there a family of pebbling graphs which requires $\Omega(N)$ pebbles for $\Omega(N)$ steps?



Sustained Space: iMHFs vs. dMHFs

- No, every pebbling graph (dynamic or otherwise) has a strategy that sustains $\mathrm{O}\left(N/\log N\right)$ for some number (maybe exponentially many) steps [HPV77]
- Idea: let's try to construct graphs where low memory attackers have huge CC

This work: CC/SSC Trade-Offs

- Goal: For some $0 < p, \ell < 1$, construct a (dynamic) pebbling graph such that any strategy either sustains at least $\boldsymbol{pN}$ pebbles for at least $\boldsymbol{\ell N}$ steps, or has CC $\boldsymbol{\Omega(N^3)}$
- Why do we need dMHFs and dynamic pebbling graphs?
  - For any $0 < c < 1$ there's a $0 < c' < 1$ such that any static graph on N nodes can be pebbled using at most cN pebbles for $c'N$ steps [LenTar79]
  - We hope that dynamic edges can leave a low-memory attacker unprepared to answer the challenge

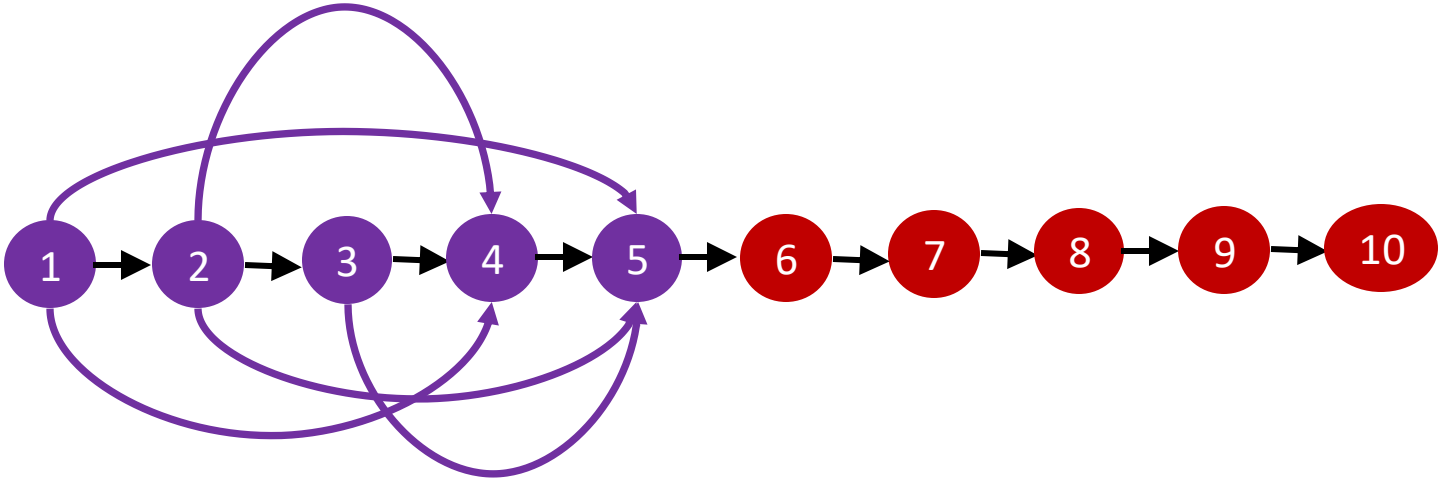
CC/SSC Trade-offs for Dynamic Pebbling Graphs

- We examine four dynamic pebbling graph families
- Practicality: graphs having constant indegree is necessary for their MHFs to be usable in practice

Dynamic Graph	Indegree	Sustained Space (over $\Omega(N)$ steps)	Cumulative Cost
Scrypt [Per09]	2	2	$O(N^2)$
Hybrid EGS [EGS75]	$O(\log N)$	$\Omega(N)$	$\Omega(N^3)$
Argon2id [BDK2015]	2	$\Omega(N^{1-\epsilon})$	$\Omega(N^{2+2\epsilon})$
Hybrid DRSample [ABH17]	3	$\Omega\left(N/\log N\right)$	$\Omega\left(N^3/\log N\right)$
Our Construction	**2**	$\boldsymbol{\Omega(N)}$	$\boldsymbol{\Omega(N^{3-\epsilon})}$

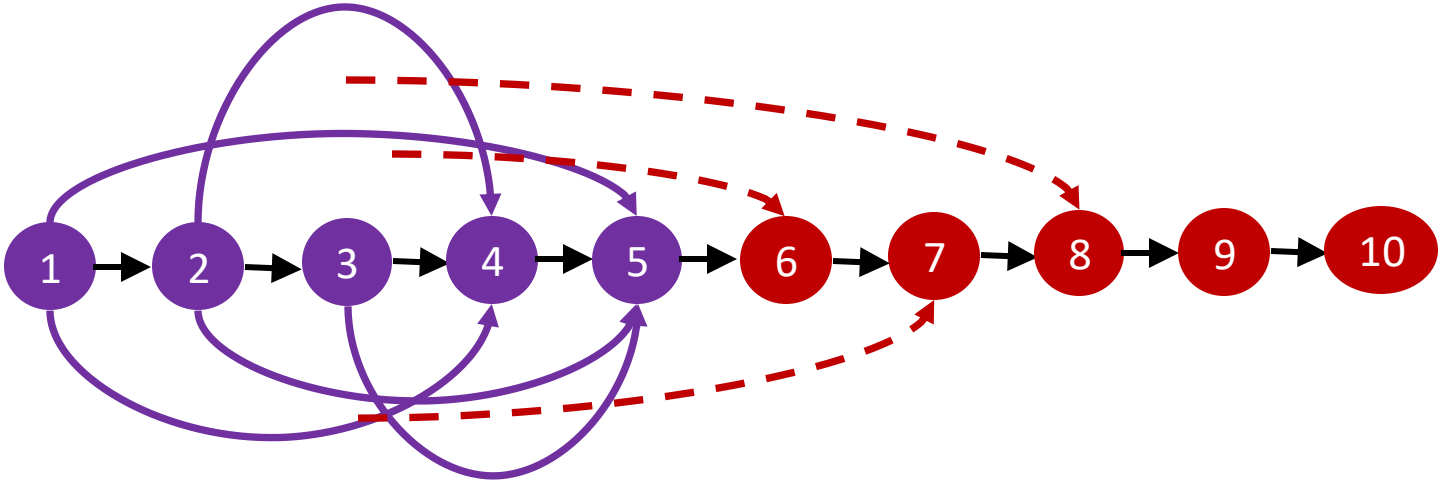
This work: CC/SSC Trade-Offs

- General construction: have dynamic edges coming from the first half of the graph which is
  - Highly connect and has high cumulative complexity



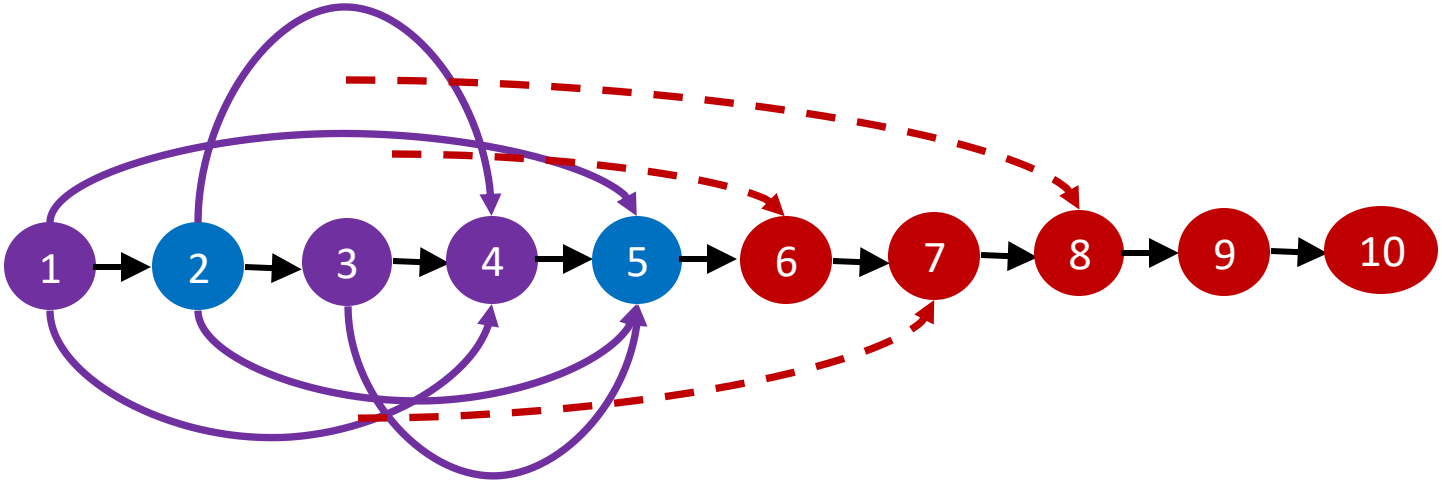
This work: CC/SSC Trade-Offs

- General construction: have dynamic edges coming from the first half of the graph which is
  - Highly connect and has high cumulative complexity



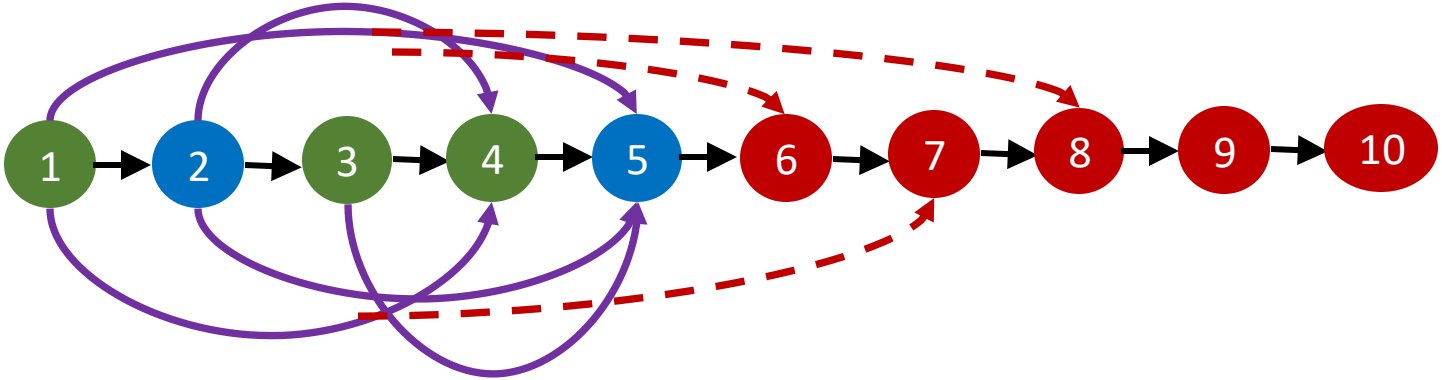
This work: CC/SSC Trade-Offs

- General construction: have dynamic edges coming from the first half of the graph which is
  - Highly connect and has high cumulative complexity
- If a strategy has few pebbles on the graph



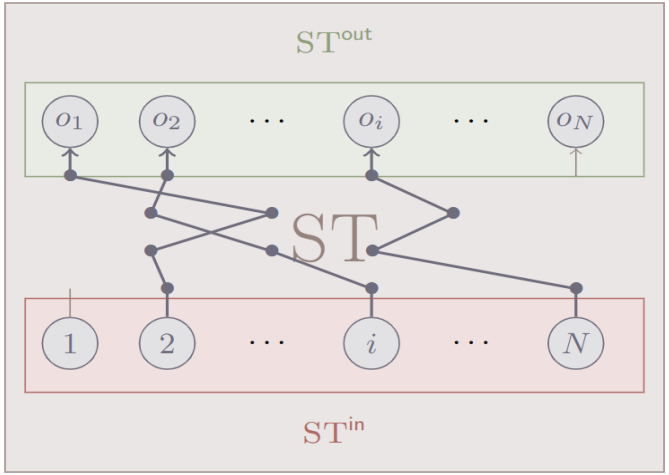
This work: CC/SSC Trade-Offs

- General construction: have dynamic edges coming from the first half of the graph which is
  - Highly connect and has high cumulative complexity
- If a strategy has few pebbles on the graph
- It will have to pebbles a large proportion of the first half, incurring high CC



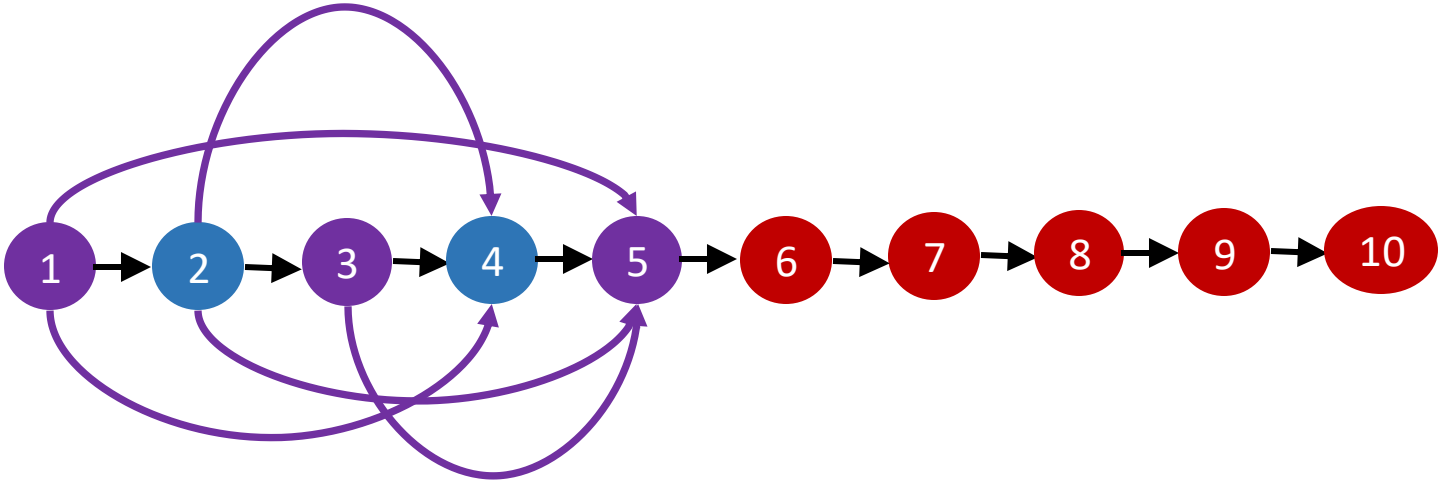
Ingredient 1: ST-Robust Graphs [BC21]

- Maximally ST-Robust Graph ST
  - N inputs, N outputs
  - For any k and $D \subseteq V$ of size k, there are paths from $n - k$ inputs each to $n - k$ outputs in $ST - D$
- Even after removing many nodes, there are still paths from many inputs to many outputs



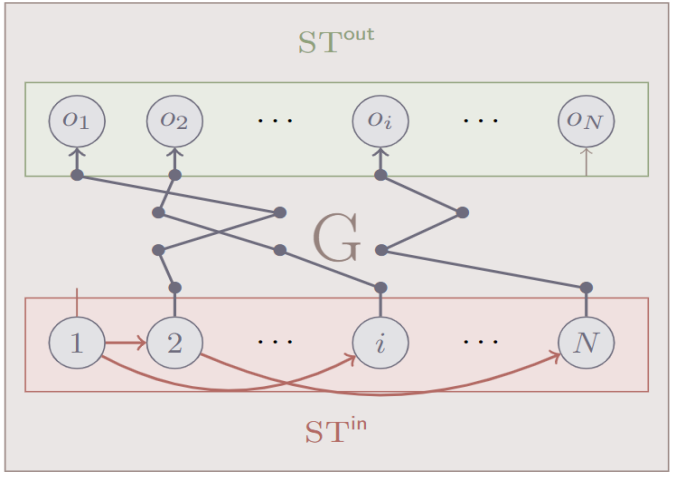
Ingredient 2: Depth-robustness

- A graph $G = (V, E)$ is $(\boldsymbol{e}, \boldsymbol{d})$***-depth robust*** if for any $S \subseteq V$ with $|S|$, $G - S$ has depth at least d.
- If G is (e, d)-depth robust, then $cc(G) \geq ed$ [ABP17]



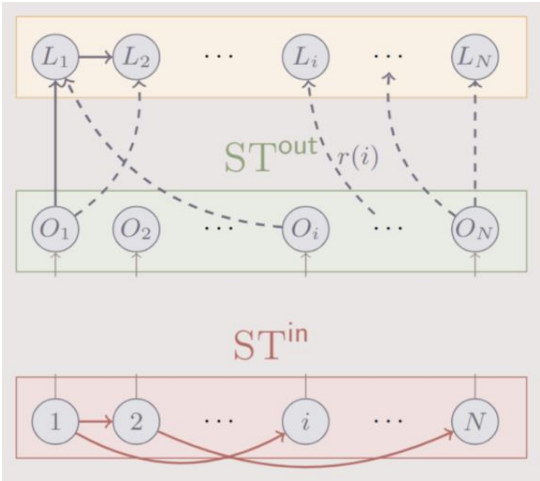
Ingredient 2: Depth-Robust Inputs

- Overlay an $(\Omega(N), d)$-depth robust graph on the inputs to the ST-robust graph
- Pebbling many inputs takes CC $\Omega(Nd)$
- How can we make a low-memory attacker repebble the inputs many times?



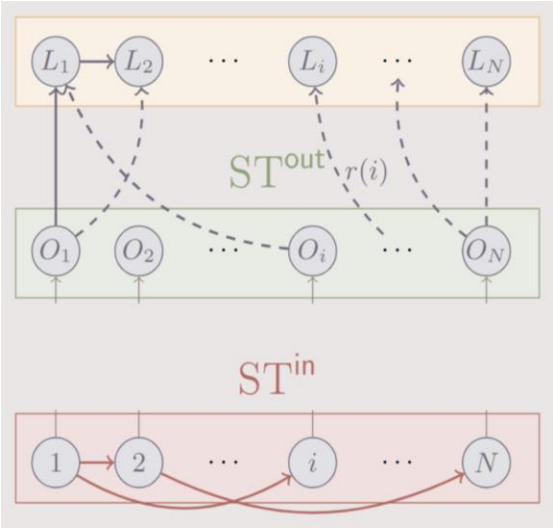
Ingredient 3: A Line Graph with Dynamic Edges

- A line graph on N nodes
- Random edges from each node of the line graph to each node in the outputs
- If an attacker doesn't have many pebbles on the graph, then they'll likely not have a pebble on $r(i)$
- There are likely many paths to $r(i)$ from many inputs
- Each time this happens it costs CC $\Omega(Nd)$
- If the strategy is low memory, then this happens $\Omega(N)$ times, resulting in CC $\Omega(N^2 d)$



Our Construction: Instantiations

- These dynamic pebbling graphs are constructed with the parameterized depth-robust graph over the inputs
- We use the Grates construction, which is $(\Omega(N), \Omega(N^{1-\epsilon}))$-depth robust [Sch83]
- Result: Any pebbling strategy either sustains pN pebbles for ℓN steps, or has CC $\Omega(N^{3-\epsilon})$



Conclusion and Open Problems

- Showed that some current/practical MHFs have high SSC/CC tradeoffs
- Theoretical construction with almost maximal trade-offs
- Open Problem: pebbling reduction for dMHFs?
- Is it possible to have a constant indegree graph where any strategy must sustain $\Omega(N)$ pebbles for $\Omega(N)$ or have CC $\Omega(N^3)$