On Graphs of Incremental Proofs of Sequential Work

Hamza Abusalah

PKC'25, Røros, Norway

Outline

1. Proofs of Sequential Work (PoSWs): Definition and an Example Construction

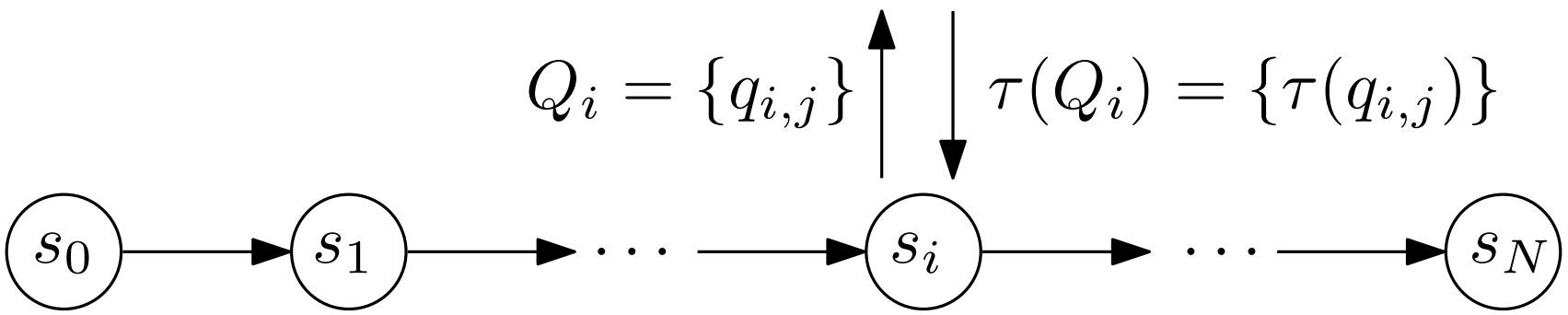
2. Incremental PoSWs (iPoSW): Motivation and a Definition

3. Main Results: Generalizing iPoSWs and an Impossibility Result

4. An Open Problem

Parallel Random Oracle Model

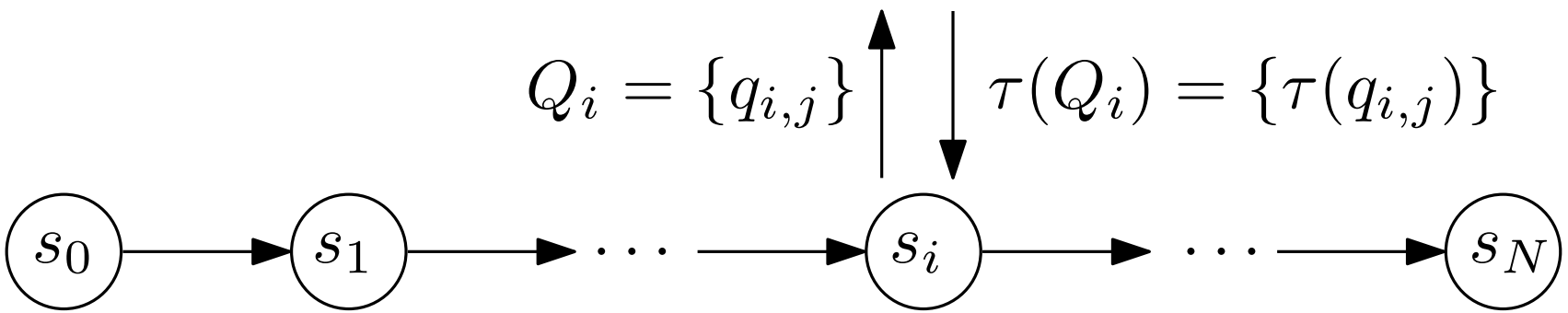
Random oracle: $\tau : \{0,1\}^* \to \{0,1\}^\lambda$



Unbounded $\mathsf{A}^{\tau(\cdot)}$ with initial state s_0

Parallel Random Oracle Model

Random oracle: $\tau : \{0,1\}^* \to \{0,1\}^\lambda$

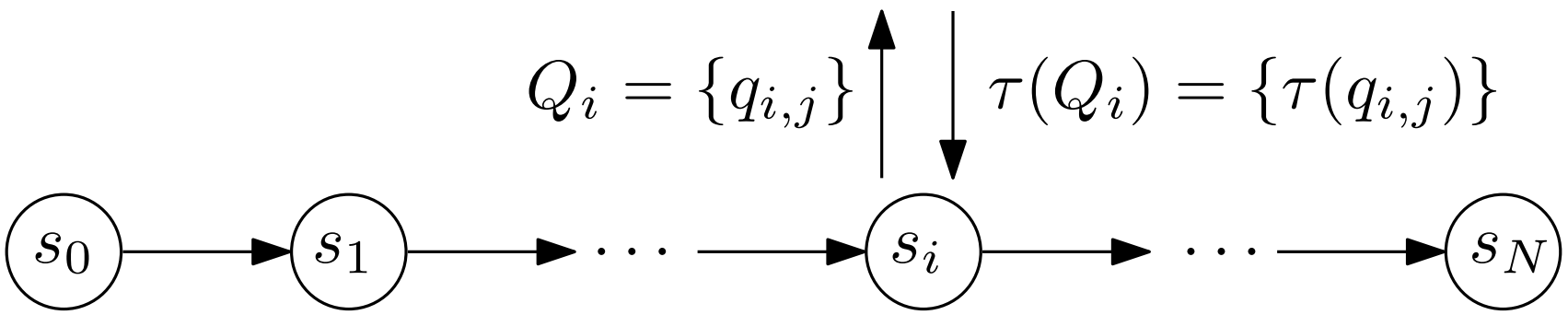


Unbounded $\mathsf{A}^{\tau(\cdot)}$ with initial state s_0

$\mathsf{Time}(\mathsf{A}^{\tau(\cdot)}) := N$

Parallel Random Oracle Model

Random oracle: $\tau : \{0,1\}^* \to \{0,1\}^\lambda$



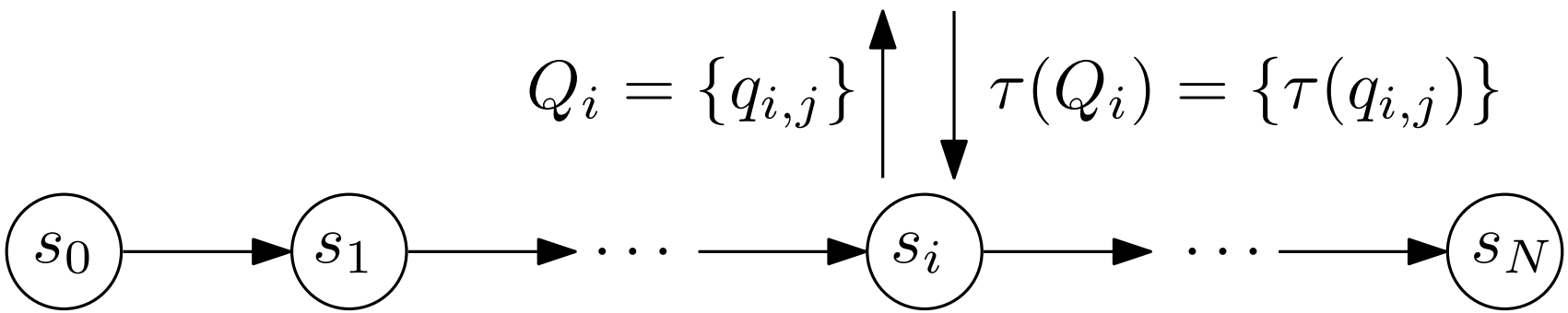
Unbounded $\mathsf{A}^{\tau(\cdot)}$ with initial state s_0

$\mathsf{Time}(\mathsf{A}^{\tau(\cdot)}) := N$

$\mathsf{Space}(\mathsf{A}^{\tau(\cdot)}) := \max\{|s_0|, \ldots, |s_N|\}$

Parallel Random Oracle Model

Random oracle: $\tau : \{0,1\}^* \to \{0,1\}^\lambda$



Unbounded $\mathsf{A}^{\tau(\cdot)}$ with initial state s_0

$\mathsf{Time}(\mathsf{A}^{\tau(\cdot)}) := N$

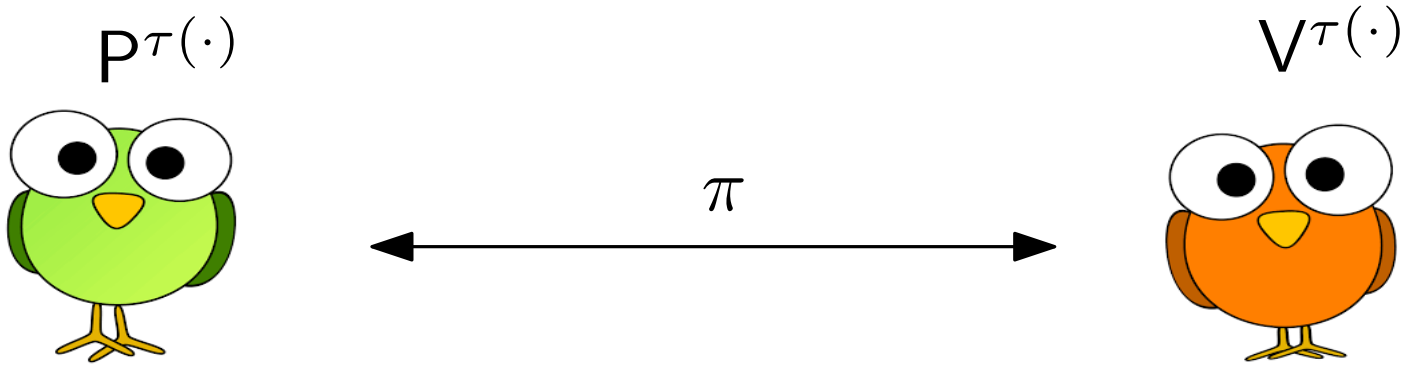
$\mathsf{Space}(\mathsf{A}^{\tau(\cdot)}) := \max\{|s_0|, \ldots, |s_N|\}$

Total number of queries $q := |Q_0| + \cdots + |Q_N|$

PoSWs

[Mahmoody-Moran-Vadhan'13]

Parameter: N

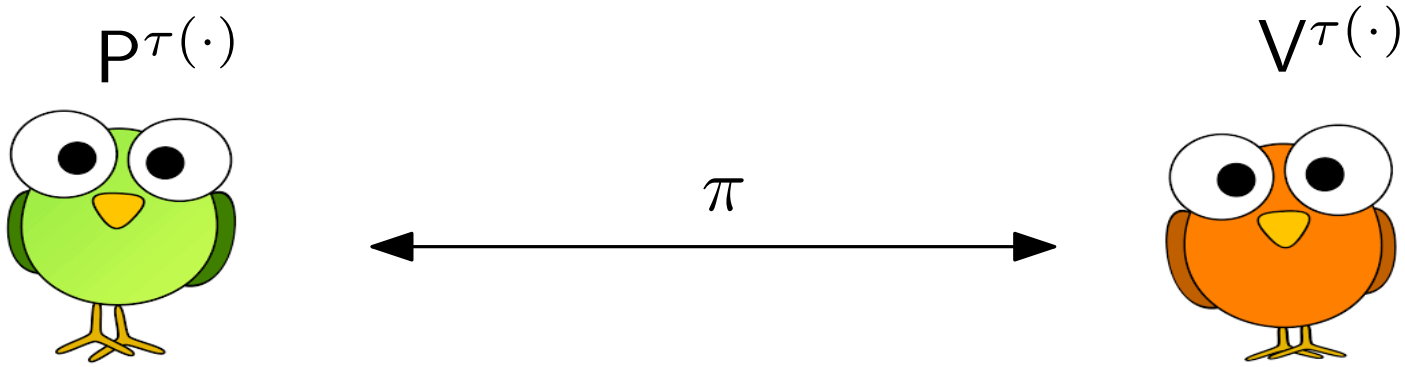


Completeness: Honest P always makes V accept

PoSWs

[Mahmoody-Moran-Vadhan'13]

Parameter: N



Completeness: Honest P always makes V accept

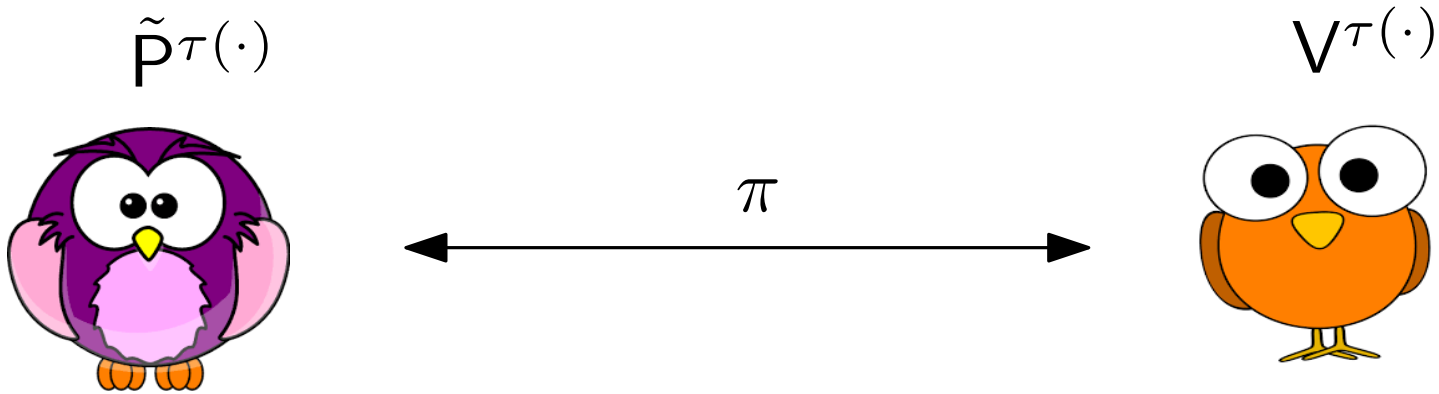
Efficiency:

$\mathsf{Time}(\mathsf{P}) = N, \quad |\pi| \in \mathsf{poly}(\log N, \lambda), \quad \mathsf{Time}(\mathsf{V}) \in \mathsf{polylog}(N)$

PoSWs

[Mahmoody-Moran-Vadhan'13]

Parameter: N



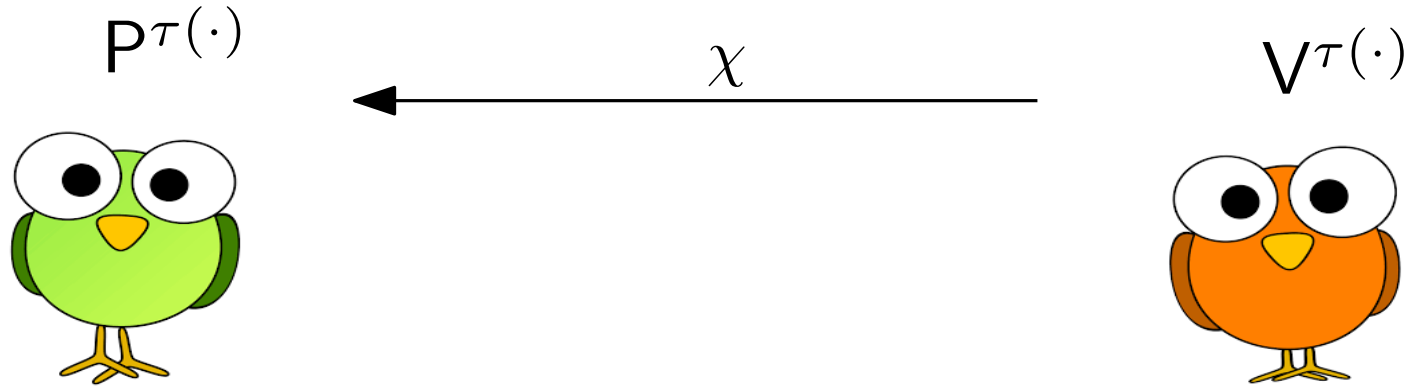
Completeness: Honest P always makes V accept

Efficiency:
$\mathsf{Time}(\mathsf{P}) = N, \quad |\pi| \in \mathsf{poly}(\log N, \lambda), \quad \mathsf{Time}(\mathsf{V}) \in \mathsf{polylog}(N)$

(α, q, ϵ)-**Soundness**: $\tilde{\mathsf{P}}$ making at most q queries and running in $\mathsf{Time}(\tilde{\mathsf{P}}) \leq \alpha \cdot N$ makes V accept with prob. $\leq \epsilon(\lambda)$

Graph-Labeling PoSWs

Parameters: G on $\{0, \ldots, N\}$ vertices

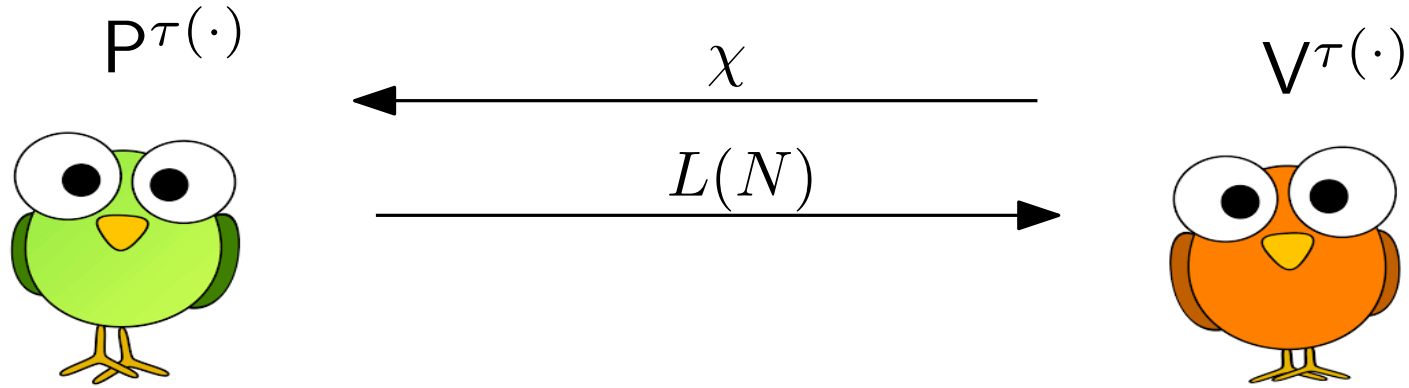


Random oracle $\tau : \{0,1\}^* \to \{0,1\}^\lambda$ with $\tau := \tau(\chi, \cdot)$

$$L(i) := \begin{cases} \tau(i) & \text{if parents}(i) = \emptyset, \\ \tau\big(i, L(\text{parents}(i))\big) & \text{otherwise.} \end{cases}$$

Graph-Labeling PoSWs

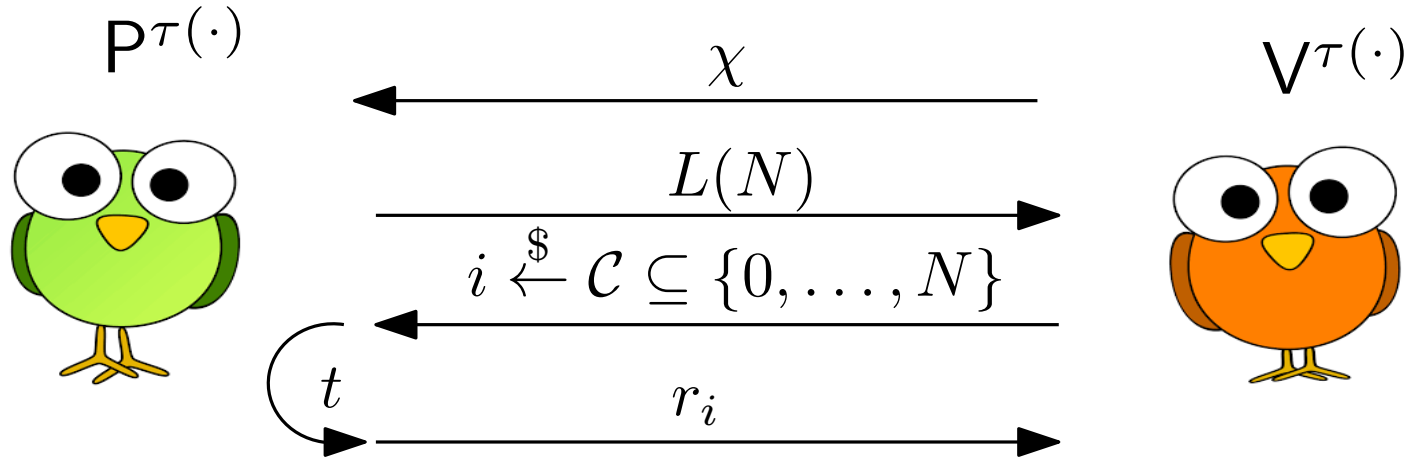
Parameters: G on $\{0, \ldots, N\}$ vertices



Random oracle $\tau : \{0,1\}^* \to \{0,1\}^\lambda$ with $\tau := \tau(\chi, \cdot)$

$$L(i) := \begin{cases} \tau(i) & \text{if parents}(i) = \emptyset, \\ \tau\big(i, L(\text{parents}(i))\big) & \text{otherwise.} \end{cases}$$

Graph-Labeling PoSWs

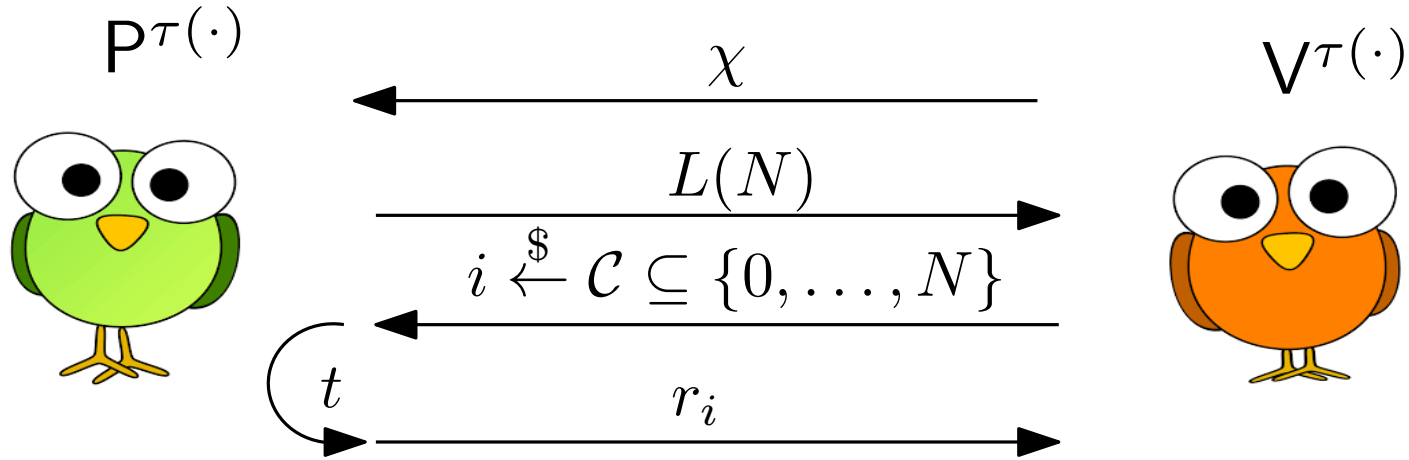


Random oracle $\tau : \{0,1\}^* \to \{0,1\}^\lambda$ with $\tau := \tau(\chi, \cdot)$

$$L(i) := \begin{cases} \tau(i) & \text{if parents}(i) = \emptyset, \\ \tau\big(i, L(\text{parents}(i))\big) & \text{otherwise.} \end{cases}$$

Graph-Labeling PoSWs

Parameters: G on $\{0, \ldots, N\}$ vertices, t



(P's messages are graph labels)

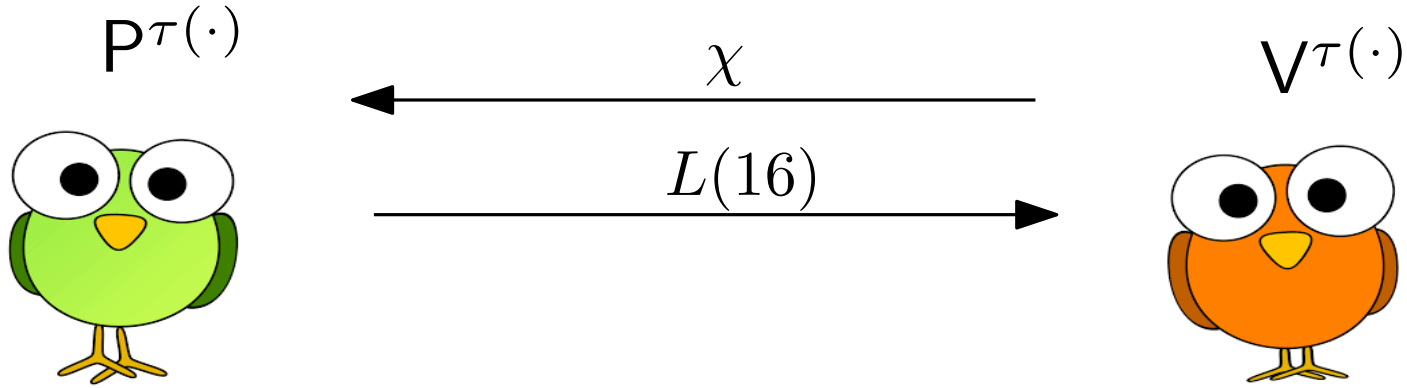
Random oracle $\tau : \{0,1\}^* \to \{0,1\}^\lambda$ with $\tau := \tau(\chi, \cdot)$

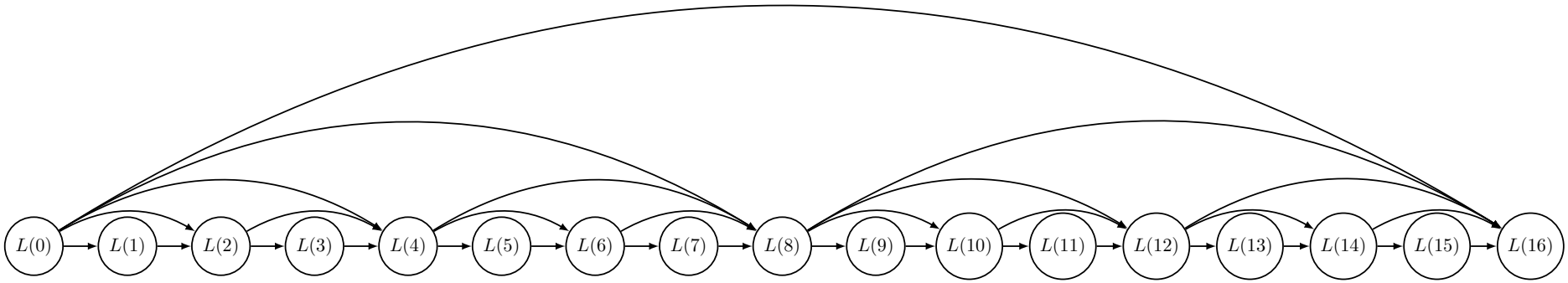
$$L(i) := \begin{cases} \tau(i) & \text{if parents}(i) = \emptyset, \\ \tau\big(i, L(\text{parents}(i))\big) & \text{otherwise.} \end{cases}$$

The Skiplist PoSW

[Abusalah-Fuchsbauer-Gaži-Klein'22]

Parameters: G, t

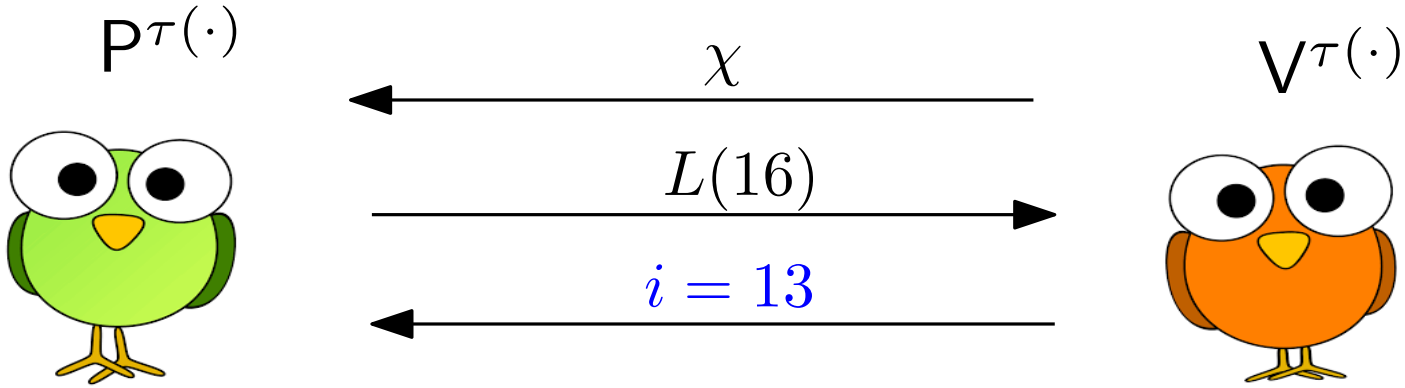


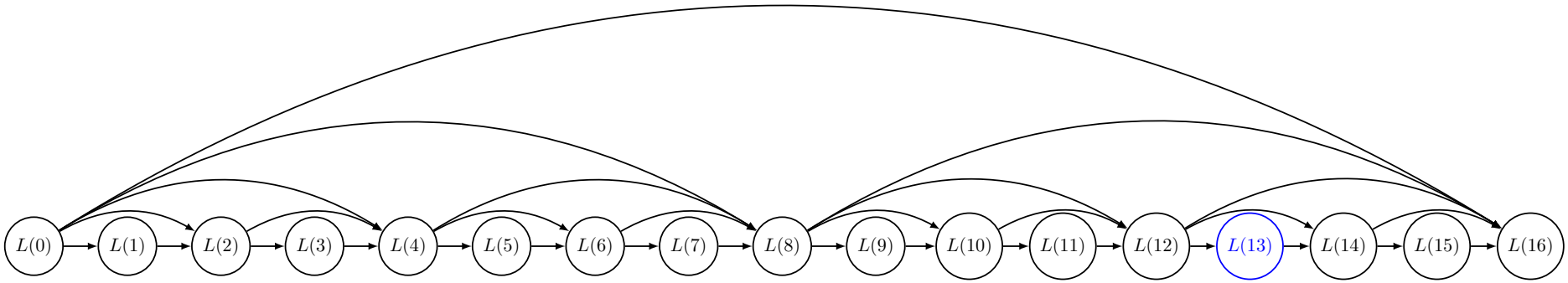


The Skiplist PoSW

[Abusalah-Fuchsbauer-Gaži-Klein'22]

Parameters: G, t

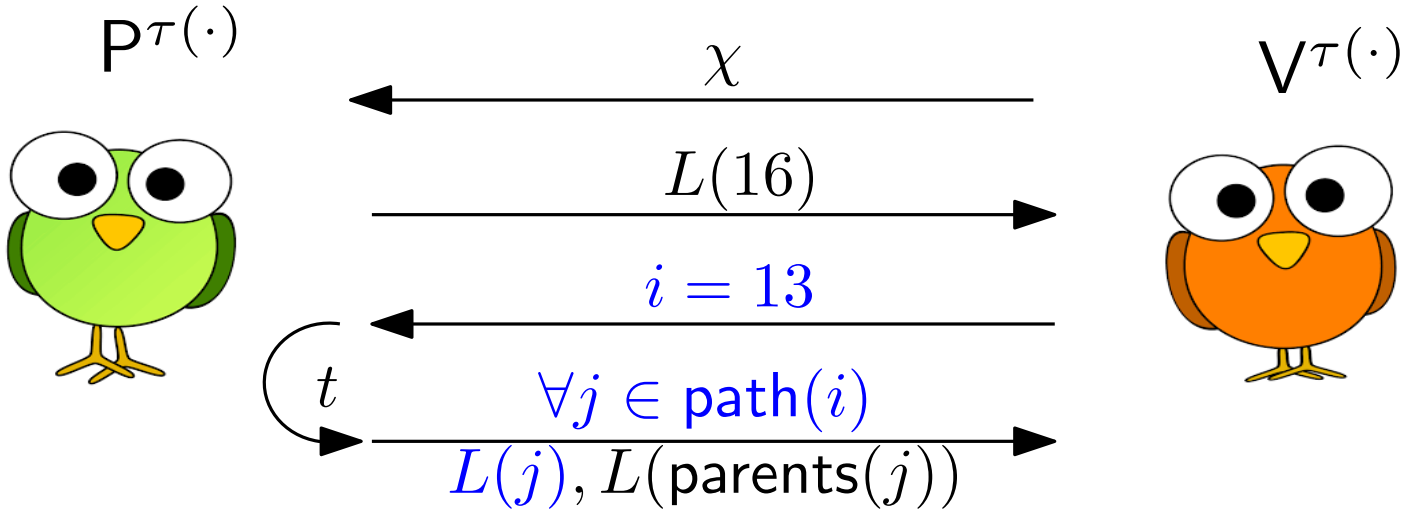


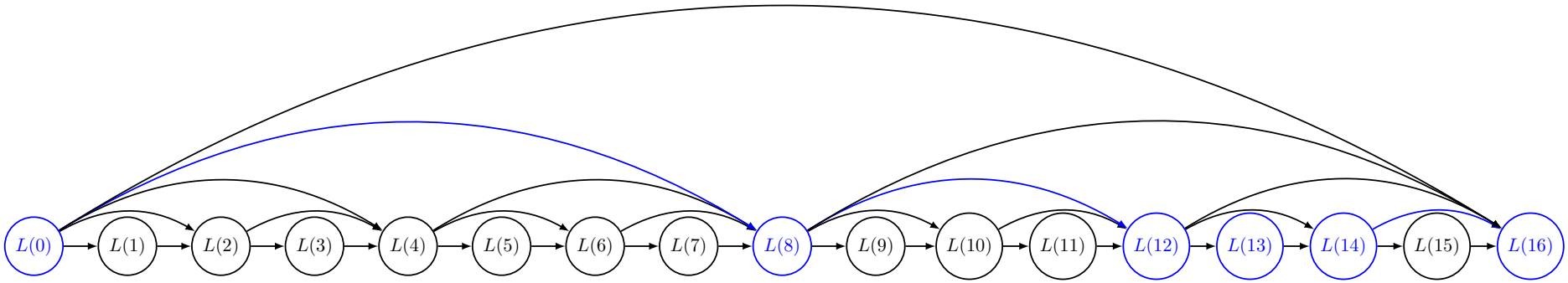


The Skiplist PoSW

[Abusalah-Fuchsbauer-Gaži-Klein'22]

Parameters: G, t

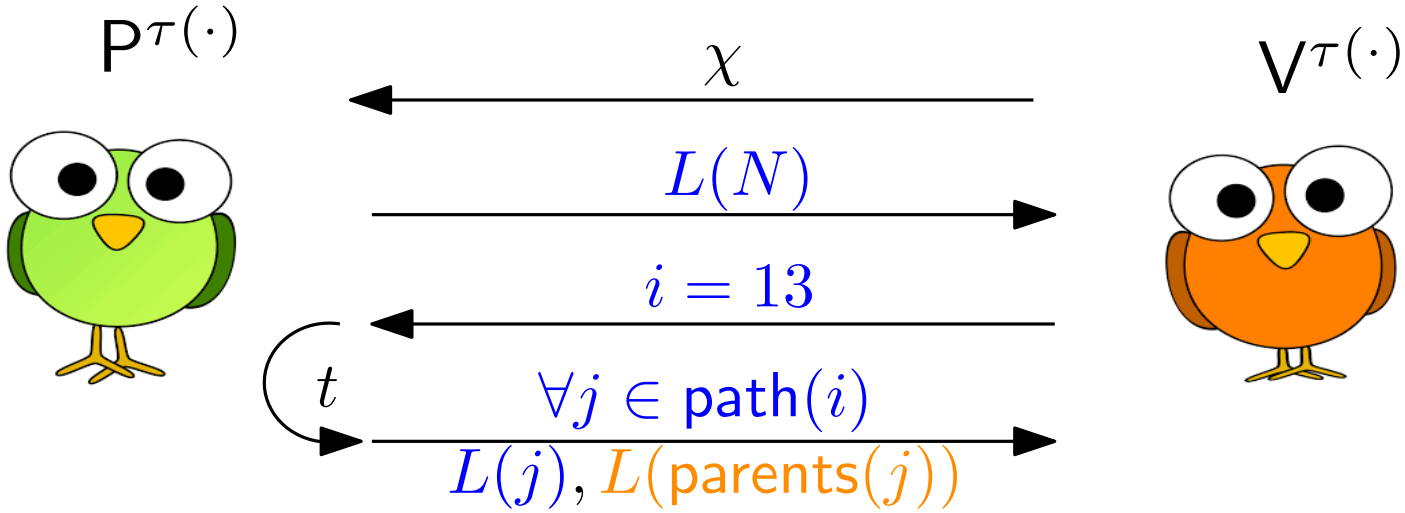


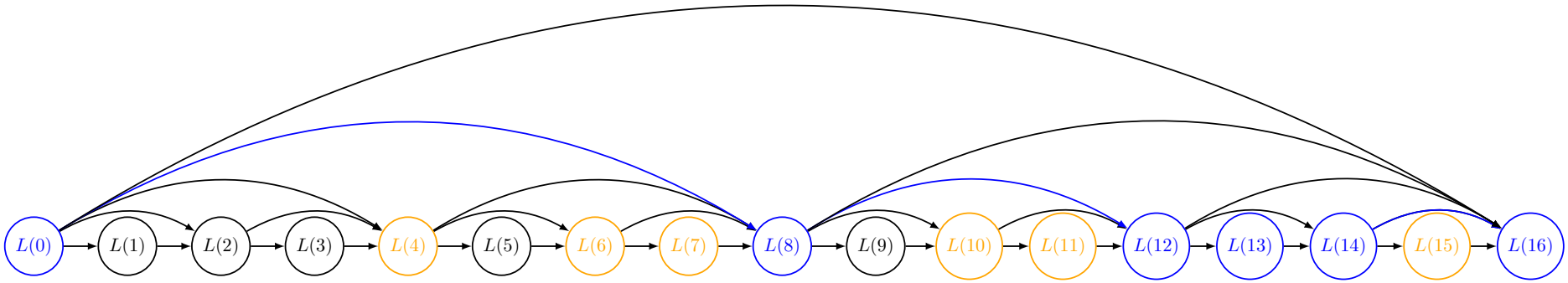


The Skiplist PoSW

[Abusalah-Fuchsbauer-Gaži-Klein'22]

Parameters: G, t

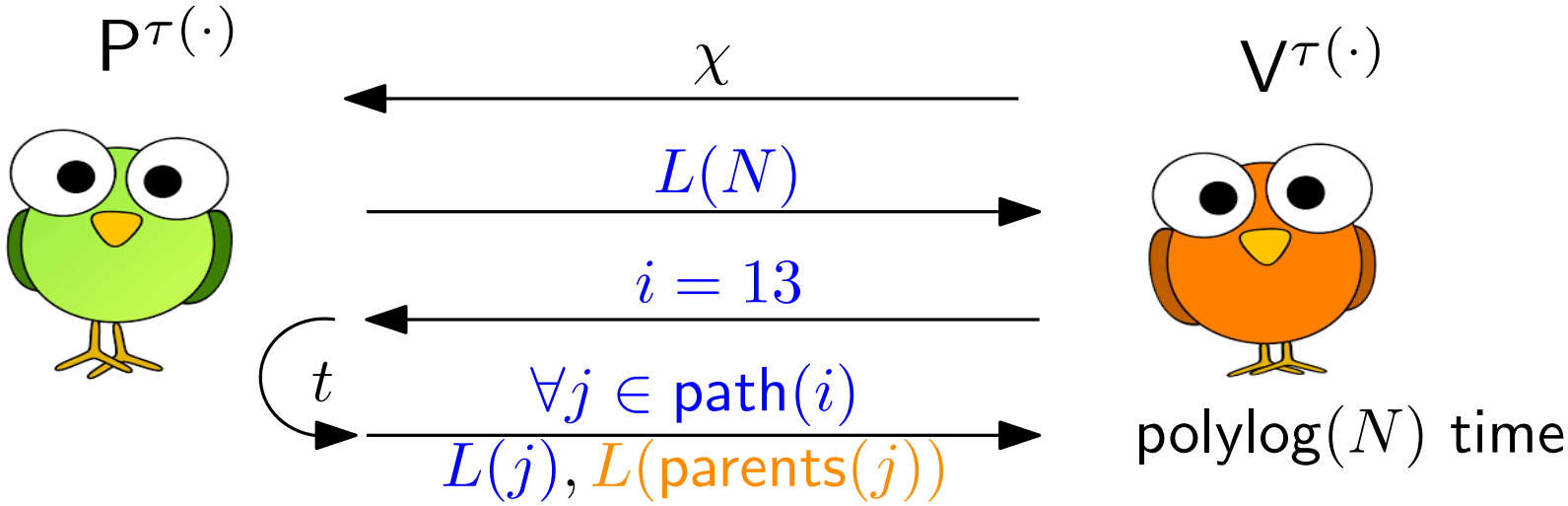




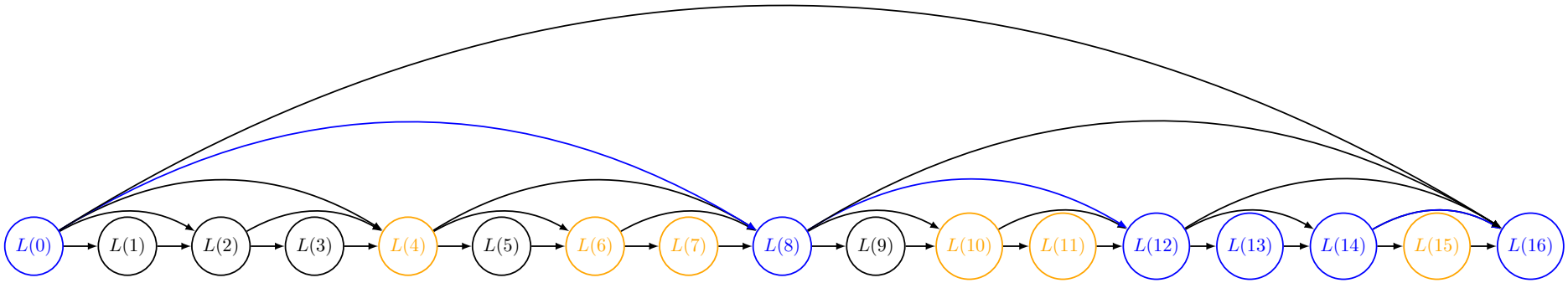
The Skiplist PoSW

[Abusalah-Fuchsbauer-Gaži-Klein'22]

Parameters: G, t



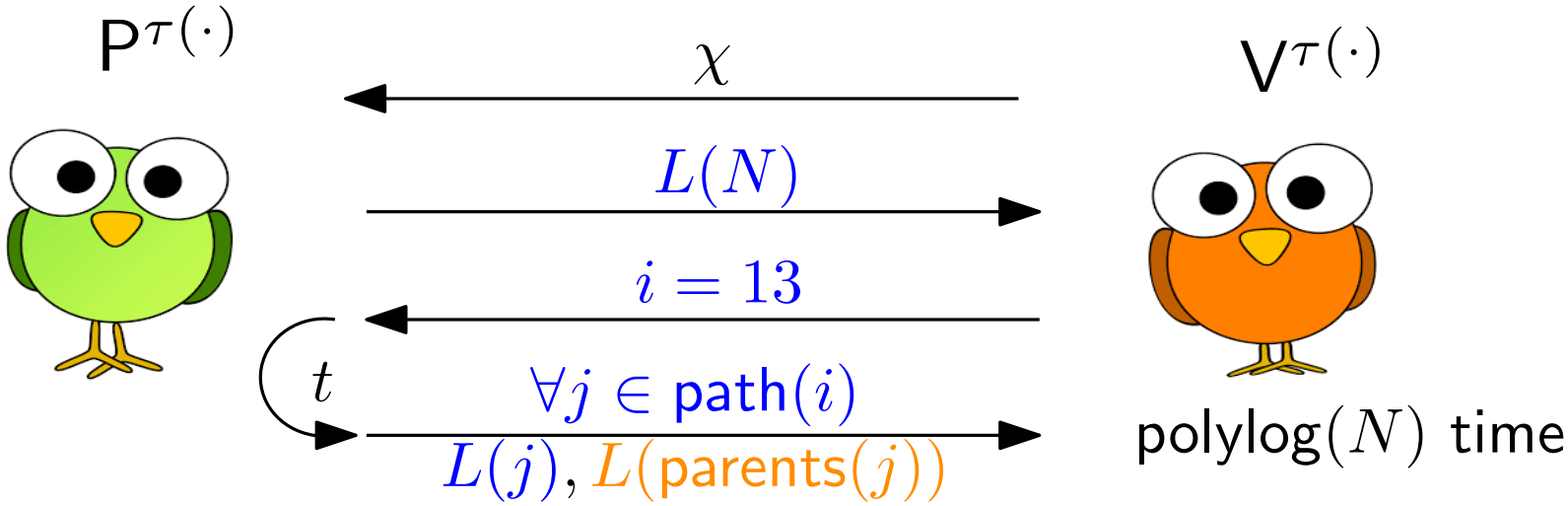
Completeness and efficiency: satisfied



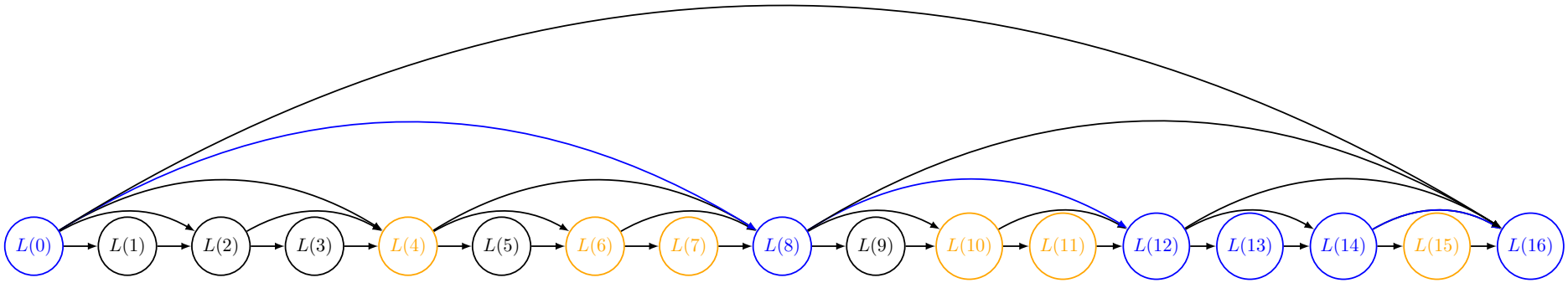
The Skiplist PoSW

[Abusalah-Fuchsbauer-Gaži-Klein'22]

Parameters: G, t



(α, q, ϵ)-Soundness: $\epsilon \leq \alpha^t + 3 \cdot q^2/2^\lambda$



Space Efficiency of Known PoSWs

For all schemes: P has two extreme strategies to answer its challenges:

1. stores all labels $L(0), \ldots, L(N)$

2. stores nothing and recomputes $L(0), \ldots, L(N)$

Space Efficiency of Known PoSWs

For all schemes: P has two extreme strategies to answer its challenges:

1. stores all labels $L(0), \ldots, L(N)$

   Time(P) $= N$ Space(P) $= N \cdot \lambda$

2. stores nothing and recomputes $L(0), \ldots, L(N)$

   Time(P) $= 2N$ Space(P) depends but hopefully small

Space Efficiency of Known PoSWs

For all schemes: P has two extreme strategies to answer its challenges:

1. stores all labels $L(0), \ldots, L(N)$

   Time(P) $= N$ Space(P) $= N \cdot \lambda$

2. stores nothing and recomputes $L(0), \ldots, L(N)$

   Time(P) $= 2N$ Space(P) depends but hopefully small

For some schemes: [CP'18, AFGK'22] P has more strategies

3. stores S labels and additionlly recomputes N/S labels

Space Efficiency of Known PoSWs

For all schemes: P has two extreme strategies to answer its challenges:

1. stores all labels $L(0), \ldots, L(N)$

   $\mathsf{Time}(\mathsf{P}) = N$ $\quad$ $\mathsf{Space}(\mathsf{P}) = N \cdot \lambda$

2. stores nothing and recomputes $L(0), \ldots, L(N)$

   $\mathsf{Time}(\mathsf{P}) = 2N$ $\quad$ $\mathsf{Space}(\mathsf{P})$ depends but hopefully small

For some schemes: [CP'18, AFGK'22] P has more strategies

3. stores S labels and additionlly recomputes N/S labels

   $\mathsf{Time}(\mathsf{P}) = N + N/S$ $\quad$ $\mathsf{Space}(\mathsf{P}) = S \cdot \lambda$

   $\mathsf{Time}(\mathsf{P}) = N + \sqrt{N}$ $\quad$ $\mathsf{Space}(\mathsf{P}) = \sqrt{N} \cdot \lambda$

Space Efficiency of Known PoSWs

For all schemes: P has two extreme strategies to answer its challenges:

1. stores all labels $L(0), \ldots, L(N)$

   $\mathsf{Time}(\mathsf{P}) = N$ $\quad$ $\mathsf{Space}(\mathsf{P}) = N \cdot \lambda$

2. stores nothing and recomputes $L(0), \ldots, L(N)$

   $\mathsf{Time}(\mathsf{P}) = 2N$ $\quad$ $\mathsf{Space}(\mathsf{P})$ depends but hopefully small

For some schemes: [CP'18, AFGK'22] P has more strategies

3. stores S labels and additionlly recomputes N/S labels

   $\mathsf{Time}(\mathsf{P}) = N + N/S$ $\quad$ $\mathsf{Space}(\mathsf{P}) = S \cdot \lambda$

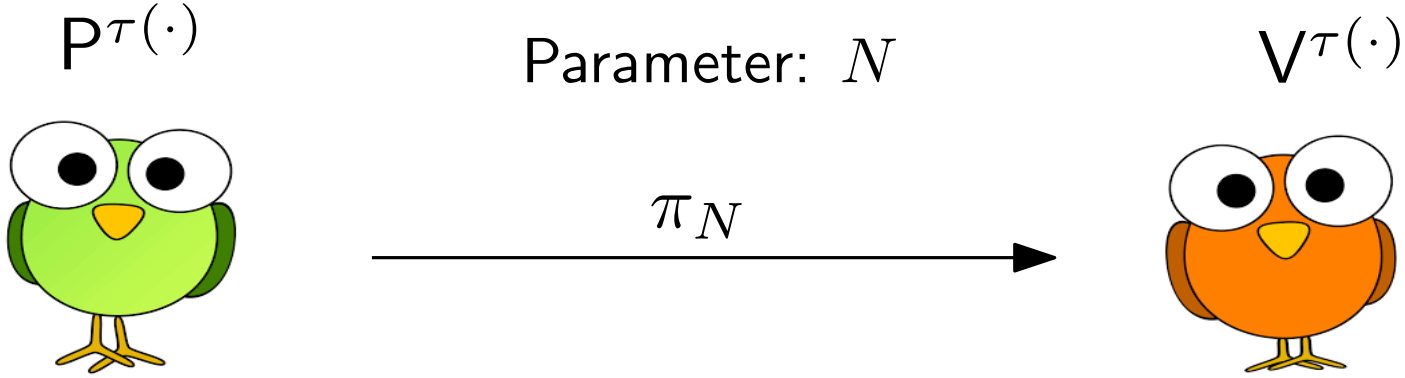
   $\mathsf{Time}(\mathsf{P}) = N + \sqrt{N}$ $\quad$ $\mathsf{Space}(\mathsf{P}) = \sqrt{N} \cdot \lambda$

Can we get the best of both worlds?

$\mathsf{Time}(\mathsf{P}) = N$ $\quad$ $\mathsf{Space}(\mathsf{P}) \in \mathsf{poly}(\log N, \lambda)$

iPoSW

[Döttling-Lai-Malavolta'19][this work]

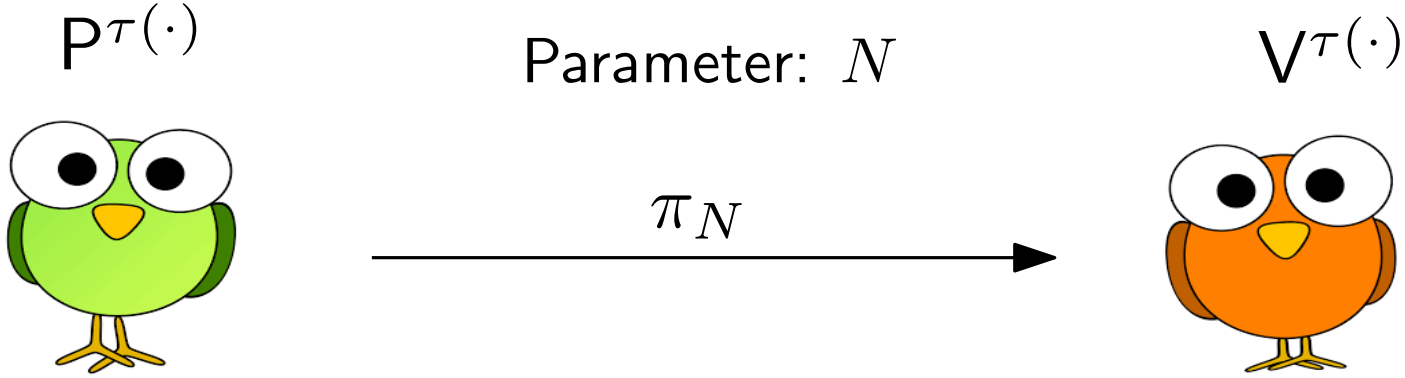


An **iPoSW** is a **non-interactive** proof system $(\mathsf{P}, \mathsf{V}, \mathsf{Inc})$ where

- (P, V) is a space-efficient and tight PoSW

iPoSW

[Döttling-Lai-Malavolta'19][this work]



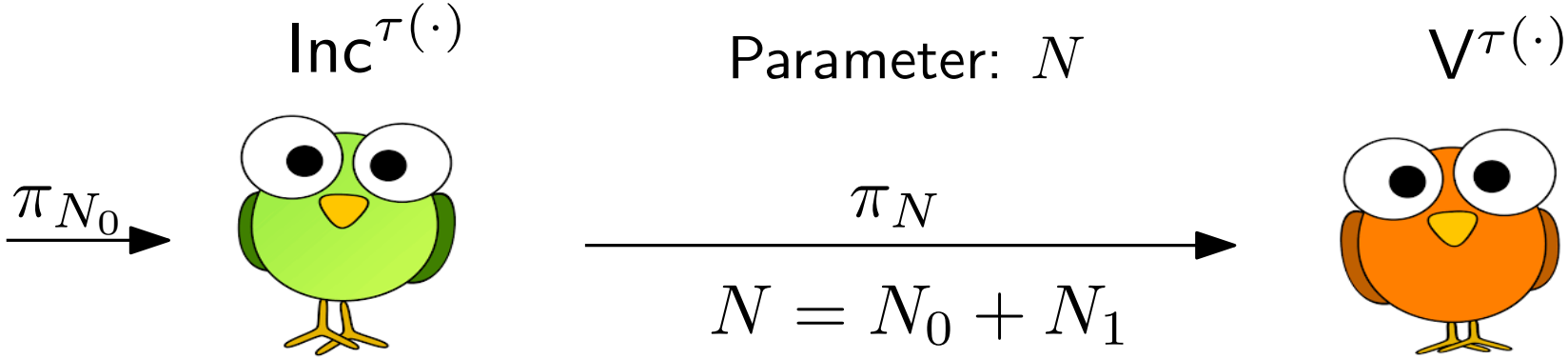
An **iPoSW** is a **non-interactive** proof system $(\mathsf{P}, \mathsf{V}, \mathsf{Inc})$ where

- (P, V) is a space-efficient and tight PoSW

$$\mathsf{Time}(\mathsf{P}) = N \qquad \mathsf{Space}(\mathsf{P}) \in \mathsf{poly}(\log N, \lambda)$$
$$|\pi| \in \mathsf{poly}(\log N, \lambda) \qquad \mathsf{Time}(\mathsf{V}) \in \mathsf{polylog}(N)$$

iPoSW

[Döttling-Lai-Malavolta'19][this work]



An **iPoSW** is a **non-interactive** proof system $(\mathsf{P}, \mathsf{V}, \mathsf{Inc})$ where

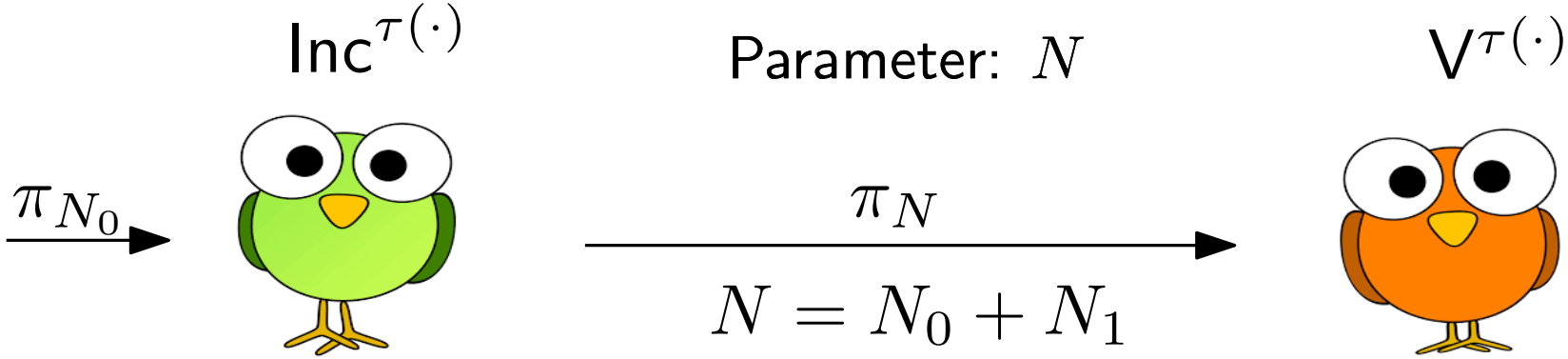
- (P, V) is a space-efficient and tight PoSW

$$\mathsf{Time}(\mathsf{P}) = N \qquad \mathsf{Space}(\mathsf{P}) \in \mathsf{poly}(\log N, \lambda)$$
$$|\pi| \in \mathsf{poly}(\log N, \lambda) \qquad \mathsf{Time}(\mathsf{V}) \in \mathsf{polylog}(N)$$

- Inc: given an accepting π_{N_0}, it produces an accepting π_N with

iPoSW

[Döttling-Lai-Malavolta'19][this work]



An **iPoSW** is a **non-interactive** proof system $(\mathsf{P}, \mathsf{V}, \mathsf{Inc})$ where

- (P, V) is a space-efficient and tight PoSW

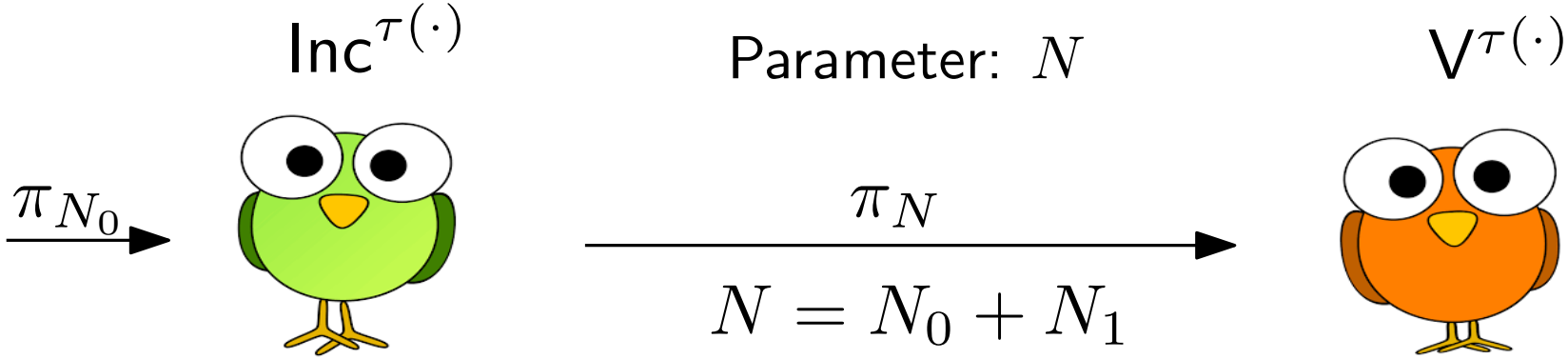
$$\mathsf{Time}(\mathsf{P}) = N \qquad \mathsf{Space}(\mathsf{P}) \in \mathsf{poly}(\log N, \lambda)$$
$$|\pi| \in \mathsf{poly}(\log N, \lambda) \qquad \mathsf{Time}(\mathsf{V}) \in \mathsf{polylog}(N)$$

- Inc: given an accepting π_{N_0}, it produces an accepting π_N with

$$\mathsf{Time}(\mathsf{Inc}) = N_1 \qquad \mathsf{Space}(\mathsf{P}) \in \mathsf{poly}(\log N, \lambda)$$
$$|\pi| \in \mathsf{poly}(\log N, \lambda) \qquad \mathsf{Time}(\mathsf{V}) \in \mathsf{polylog}(N)$$

iPoSW

[Döttling-Lai-Malavolta'19][this work]



An **iPoSW** is a **non-interactive** proof system $(\mathsf{P}, \mathsf{V}, \mathsf{Inc})$ where

- (P, V) is a space-efficient and tight PoSW

$$\mathsf{Time}(\mathsf{P}) = N \qquad \mathsf{Space}(\mathsf{P}) \in \mathsf{poly}(\log N, \lambda)$$
$$|\pi| \in \mathsf{poly}(\log N, \lambda) \qquad \mathsf{Time}(\mathsf{V}) \in \mathsf{polylog}(N)$$

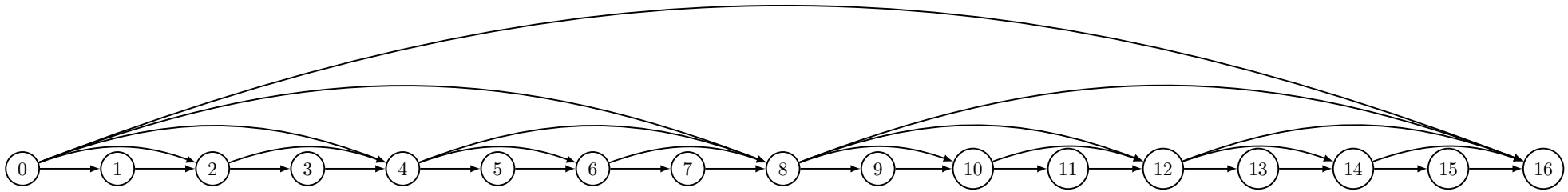
- Inc: given an accepting π_{N_0}, it produces an accepting π_N with

$$\mathsf{Time}(\mathsf{Inc}) = N_1 \qquad \mathsf{Space}(\mathsf{P}) \in \mathsf{poly}(\log N, \lambda)$$
$$|\pi| \in \mathsf{poly}(\log N, \lambda) \qquad \mathsf{Time}(\mathsf{V}) \in \mathsf{polylog}(N)$$

$$\longrightarrow \underset{N_0}{\pi_0} \longrightarrow \underset{+\ N_1}{\pi_1} \longrightarrow \underset{+\ N_2}{\pi_2} \longrightarrow \underset{+\ \cdots}{\cdots}$$

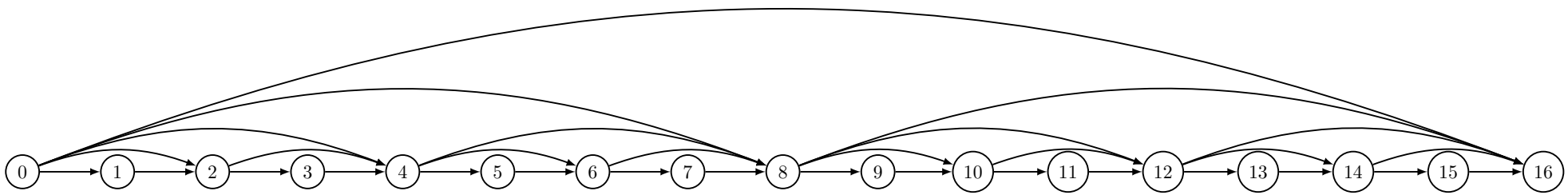
Graphs of Known PoSWs

1. [Abusalah-Kamath-Klein-Walter-Pietrzak'19] constructed **reversible PoSWs** (a precursor to VDFs)

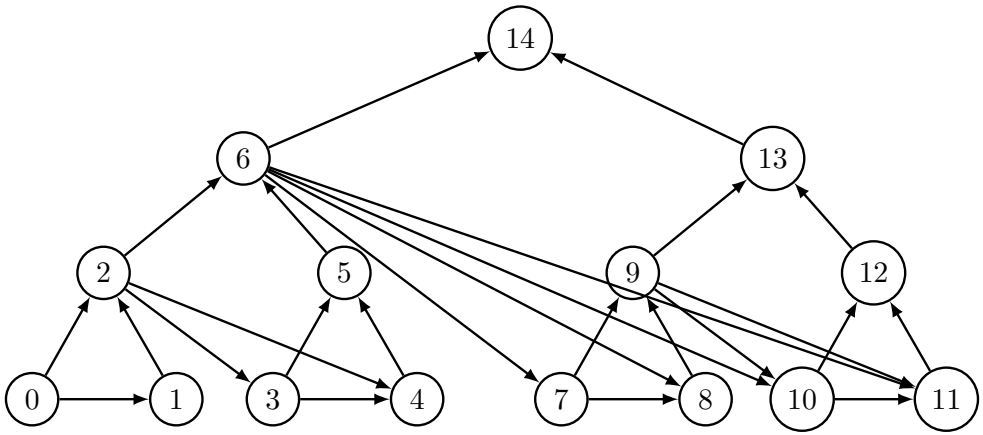


Graphs of Known PoSWs

1. [Abusalah-Kamath-Klein-Walter-Pietrzak'19] constructed **reversible PoSWs** (a precursor to VDFs)

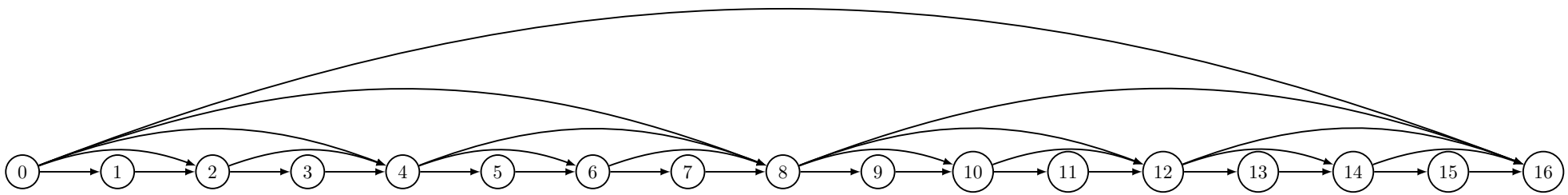


2. [Cohen-Pietrzak'18]: Merkle-tree like graphs

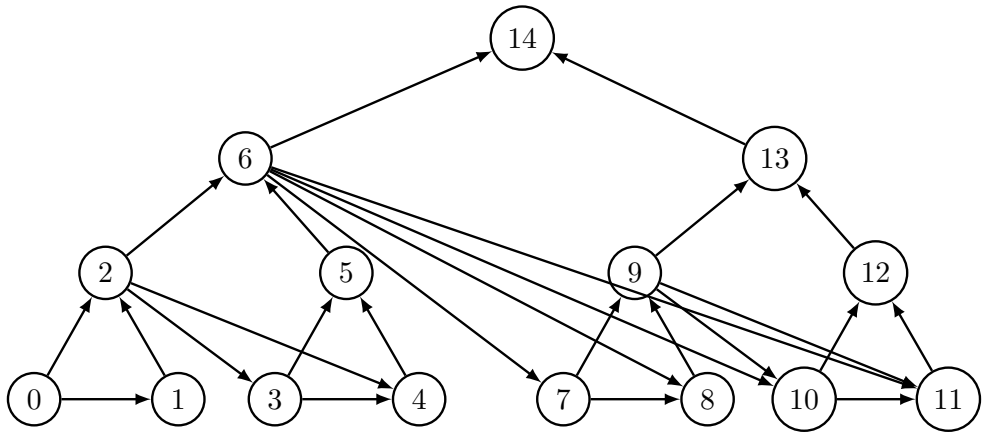


Graphs of Known PoSWs

1. [Abusalah-Kamath-Klein-Walter-Pietrzak'19] constructed **reversible PoSWs** (a precursor to VDFs)



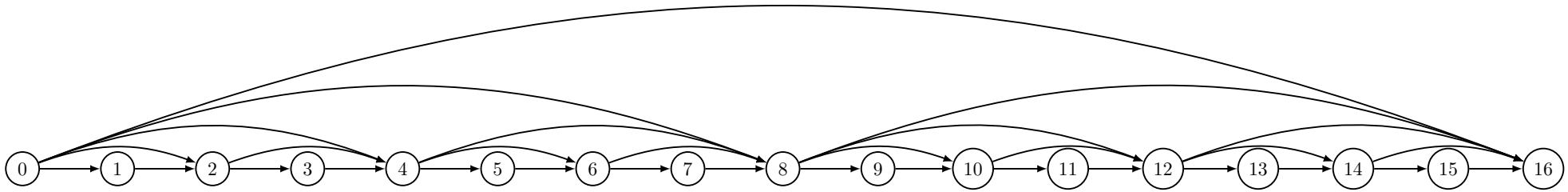
2. [Cohen-Pietrzak'18]: Merkle-tree like graphs



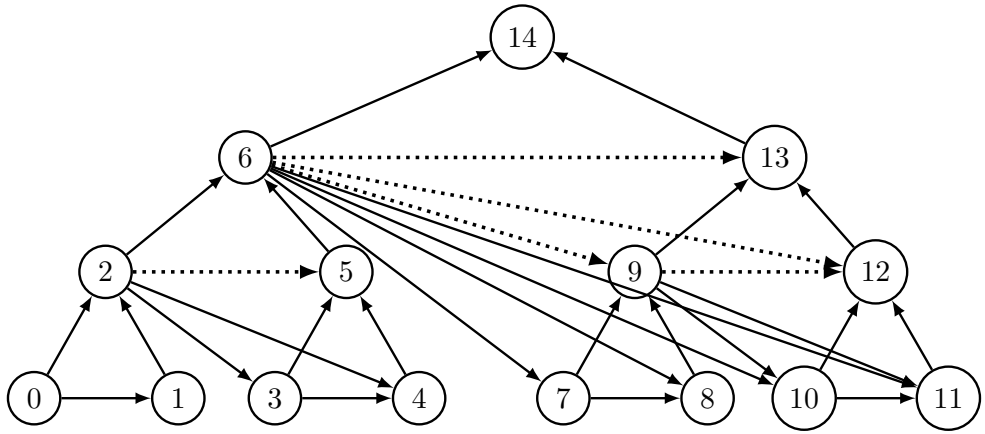
3. [Mahmoody-Moran-Vadhan'13] gave the first PoSW from "highly" **depth-robust graphs** with a Merkle-tree on top

Graphs of Known PoSWs

1. [Abusalah-Kamath-Klein-Walter-Pietrzak'19] constructed **reversible PoSWs** (a precursor to VDFs)
   [Abusalah-Fuchsbauer-Gaži-Klein'22] adapted it to blockchain applications



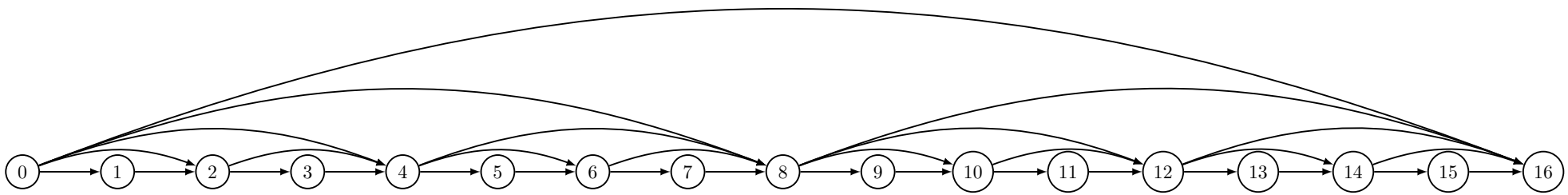
2. [Cohen-Pietrzak'18]: Merkle-tree like graphs
   [Abusalah-Fuchsbauer-Gaži-Klein'22] adapted it to blockchain applications



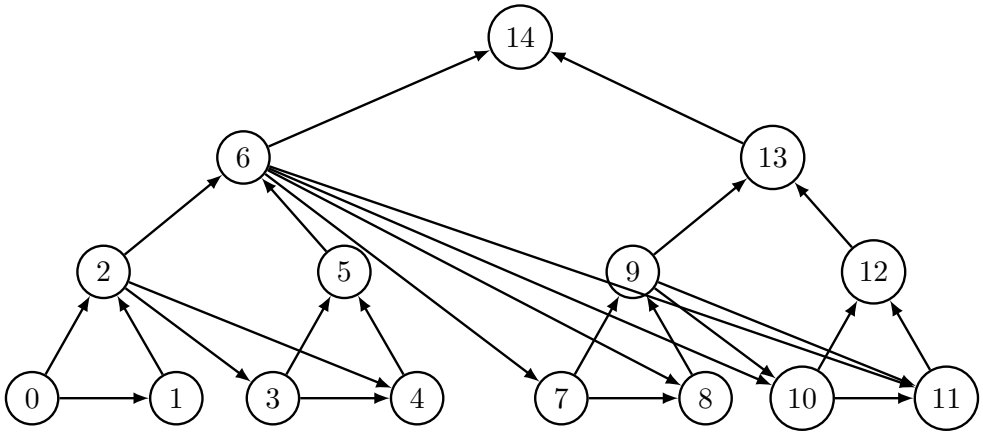
3. [Mahmoody-Moran-Vadhan'13] gave the first PoSW from "highly" **depth-robust graphs** with a Merkle-tree on top

Graphs of Known **iPoSWs**

1. [Abusalah-Cini'23] by transforming the skiplist PoSW

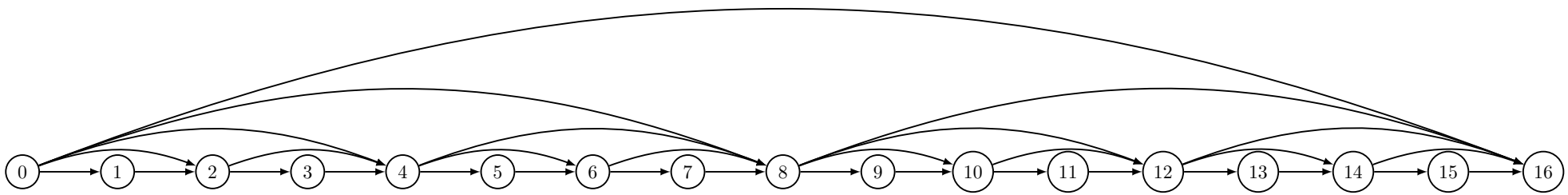


2. [Döttling-Lai-Malavolta'19] by transforming [CP'18]

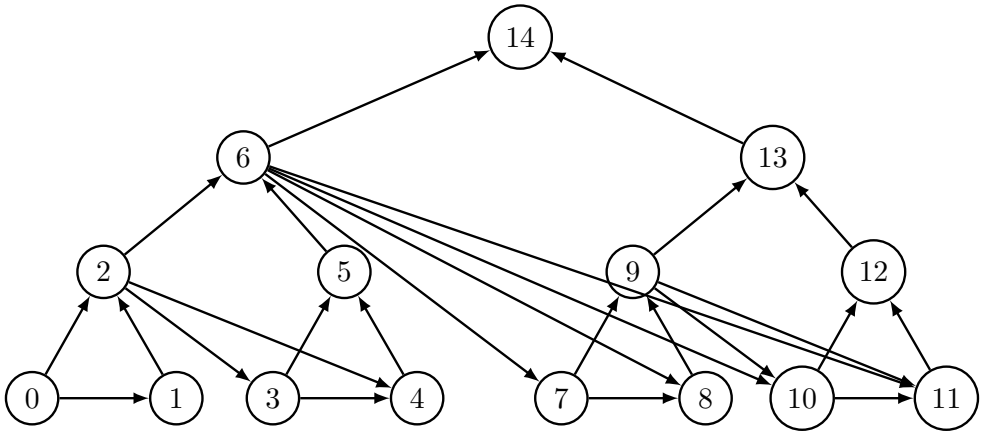


Graphs of Known **iPoSWs**

1. [Abusalah-Cini'23] by transforming the skiplist PoSW



2. [Döttling-Lai-Malavolta'19] by transforming [CP'18]



(Both schemes use the **on-the-fly/incremental sampling** of [DLM'19])

Open Problems

1. Can we transform [MMV'13] into an iPoSW?

2. Can we generalize [DLM19] and [AC'23]?

Solutions

1. Can we transform [MMV'13] into an iPoSW? **No**

2. Can we generalize [DLM19] and [AC'23]?

Solutions

1. Can we transform [MMV'13] into an iPoSW? **No**

   1. Define **incremental graphs** and show that (graph-labeling) iPoSWs have incremental graphs (reling on efficiency of Inc)

2. Can we generalize [DLM19] and [AC'23]?

Solutions

1. Can we transform [MMV'13] into an iPoSW? **No**
   1. Define **incremental graphs** and show that (graph-labeling) iPoSWs have incremental graphs (reling on efficiency of Inc)
   2. Show that depth-robust graphs are not incremental
      (a) Show incrementable graphs have small space pebbling complexity
      (b) DRGs have high space pebbling complexity [Alwen-Blocki-Pietrzak'17]

2. Can we generalize [DLM19] and [AC'23]?

Solutions

1. Can we transform [MMV'13] into an iPoSW? **No**
   1. Define **incremental graphs** and show that (graph-labeling) iPoSWs have incremental graphs (reling on efficiency of Inc)
   2. Show that depth-robust graphs are not incremental
      (a) Show incrementable graphs have small space pebbling complexity
      (b) DRGs have high space pebbling complexity [Alwen-Blocki-Pietrzak'17]
   3. Conclude [MMV'13] can't be made into an iPoSW; it uses DRGs
2. Can we generalize [DLM19] and [AC'23]?

Solutions

1. Can we transform [MMV'13] into an iPoSW? **No**
   1. Define **incremental graphs** and show that (graph-labeling) iPoSWs have incremental graphs (reling on efficiency of Inc)
   2. Show that depth-robust graphs are not incremental
      (a) Show incrementable graphs have small space pebbling complexity
      (b) DRGs have high space pebbling complexity [Alwen-Blocki-Pietrzak'17]
   3. Conclude [MMV'13] can't be made into an iPoSW; it uses DRGs
2. Can we generalize [DLM19] and [AC'23]? **Yes**

Solutions

1. Can we transform [MMV'13] into an iPoSW? **No**

   1. Define **incremental graphs** and show that (graph-labeling) iPoSWs have incremental graphs (reling on efficiency of Inc)

   2. Show that depth-robust graphs are not incremental
      (a) Show incrementable graphs have small space pebbling complexity
      (b) DRGs have high space pebbling complexity [Alwen-Blocki-Pietrzak'17]

   3. Conclude [MMV'13] can't be made into an iPoSW; it uses DRGs

2. Can we generalize [DLM19] and [AC'23]? **Yes**

   Any PoSW with underlying **incremental and dynamic graphs** can be transformed into an iPoSW.

Solutions

1. Can we transform [MMV'13] into an iPoSW? **No**

   1. Define **incremental graphs** and show that (graph-labeling) iPoSWs have incremental graphs (reling on efficiency of Inc)

   2. Show that depth-robust graphs are not incremental
      (a) Show incrementable graphs have small space pebbling complexity
      (b) DRGs have high space pebbling complexity [Alwen-Blocki-Pietrzak'17]

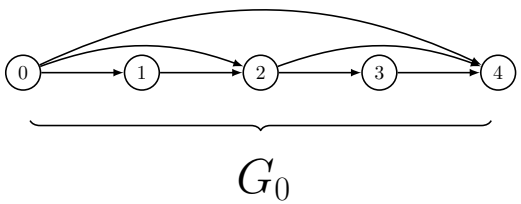
   3. Conclude [MMV'13] can't be made into an iPoSW; it uses DRGs

2. Can we generalize [DLM19] and [AC'23]? **Yes**

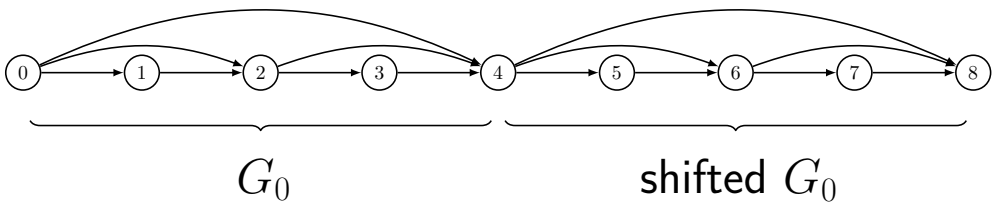
   Any PoSW with underlying **incremental and dynamic graphs** can be transformed into an iPoSW.

   + a unified transformation
   + corollary: modified CP [AFGK'22] is also incremental

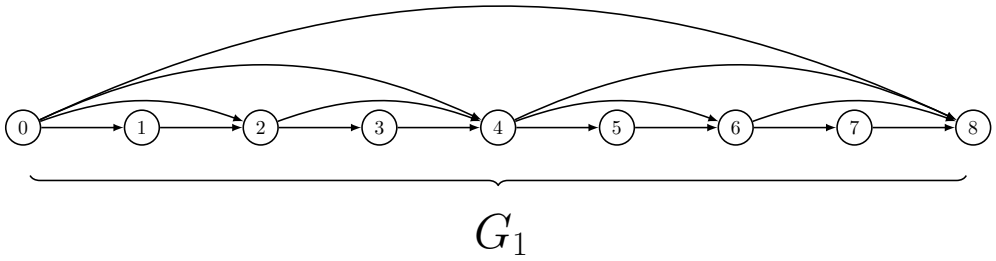
Dynamic Graphs: An Example



Dynamic Graphs: An Example

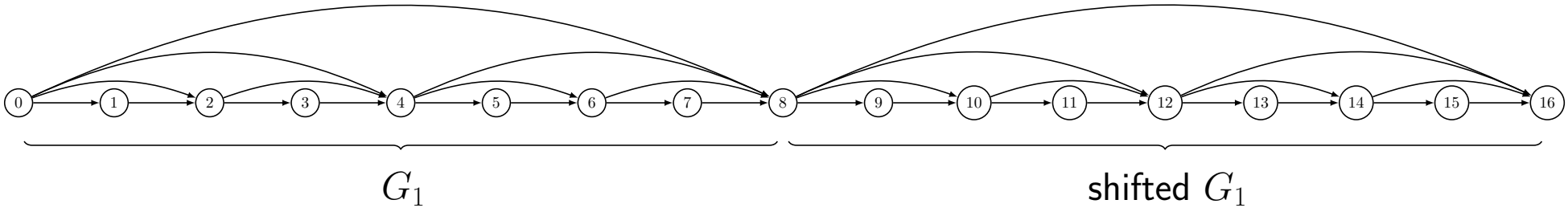


Dynamic Graphs: An Example

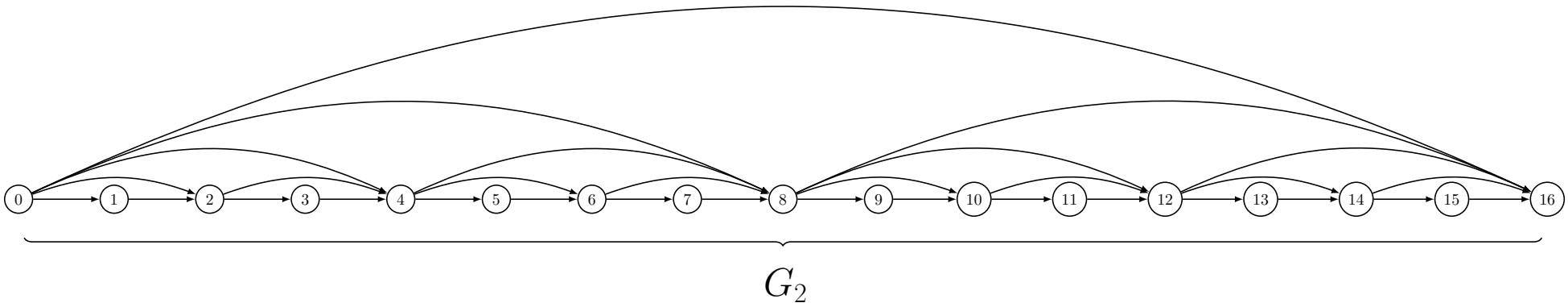


G_1

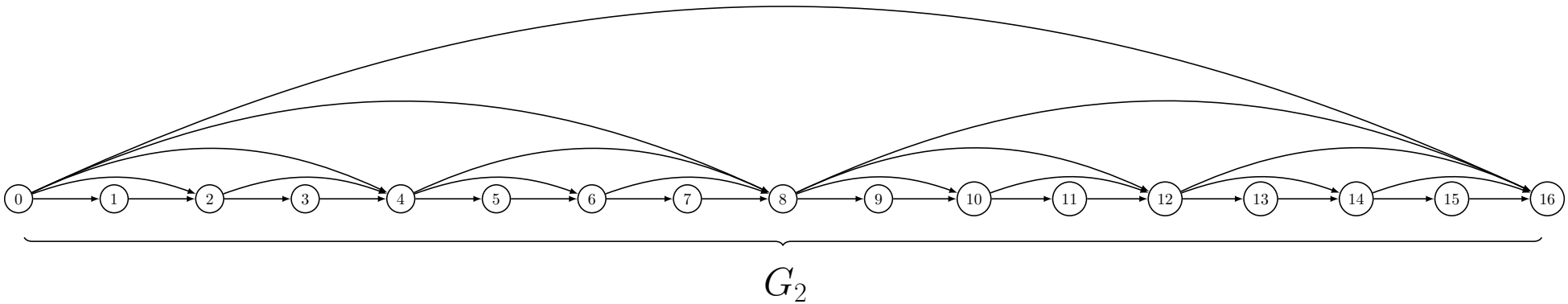
Dynamic Graphs: An Example



Dynamic Graphs: An Example

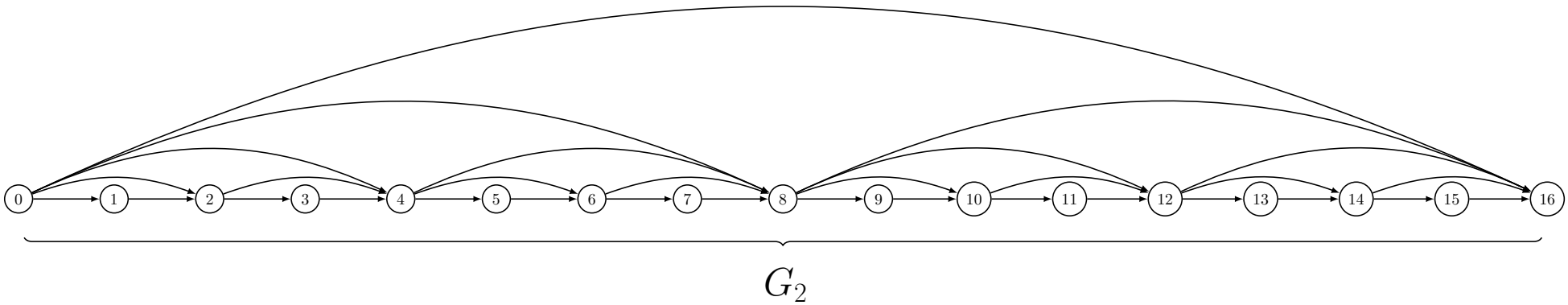


Dynamic Graphs: An Example



Easy to see that all known PoSW graphs [MMV'13, CP'18, AKKWP'19, AFGK'22] are dynamic – with a minor generalization to this example

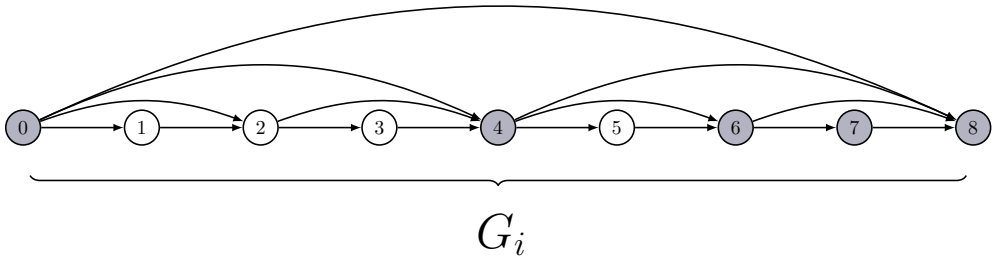
Dynamic Graphs: An Example



Easy to see that all known PoSW graphs [MMV'13, CP'18, AKKWP'19, AFGK'22] are dynamic – with a minor generalization to this example

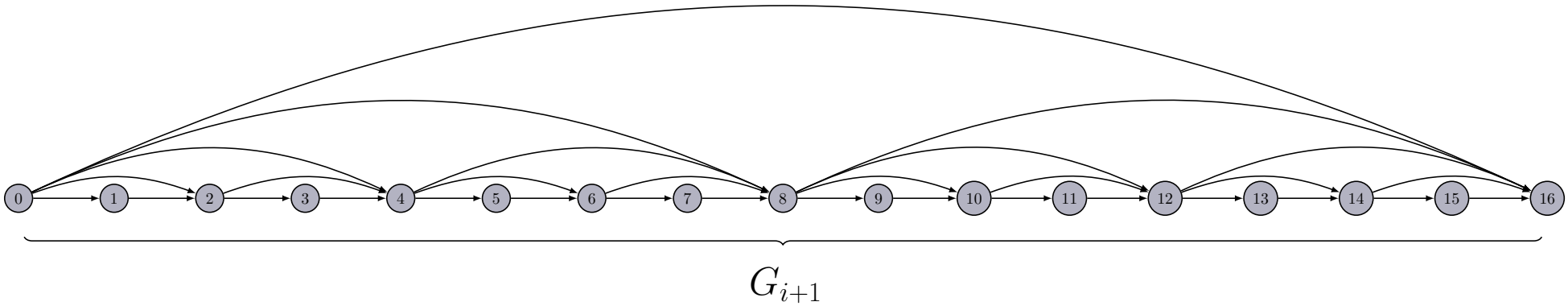
Dynamic graphs are friendly to the incremental sampling technique [DLM'19] used by all iPoSWs [DLM'19, AC'23]

Incremental Graphs



Definition of Incremental Graphs:
Given pebbles on $T \subseteq V(G_i)$

Incremental Graphs



Definition of Incremental Graphs:
Given pebbles on $T \subseteq V(G_i)$, G_{i+1} can be pebbled in time and space complexities

$$|V(G_{i+1}) \setminus V(G_i)| \quad \text{and} \quad \text{polylog}(|V(G_{i+1})|)$$

To Conclude

1. We generalized iPoSWs relying on dynamic and incremental graphs of PoSWs

To Conclude

1. We generalized iPoSWs relying on dynamic and incremental graphs of PoSWs

2. We proved that [MMV'13] is not incremental by showing incremental graphs are necessary and depth-robust graphs are not incremental

To Conclude

1. We generalized iPoSWs relying on dynamic and incremental graphs of PoSWs

2. We proved that [MMV'13] is not incremental by showing incremental graphs are necessary and depth-robust graphs are not incremental

An open problem:

What are the necessary and sufficient graph conditions that allow (standalone) PoSWs?