# Simple Proofs of Sequential Work

Bram Cohen

Krzysztof Pietrzak

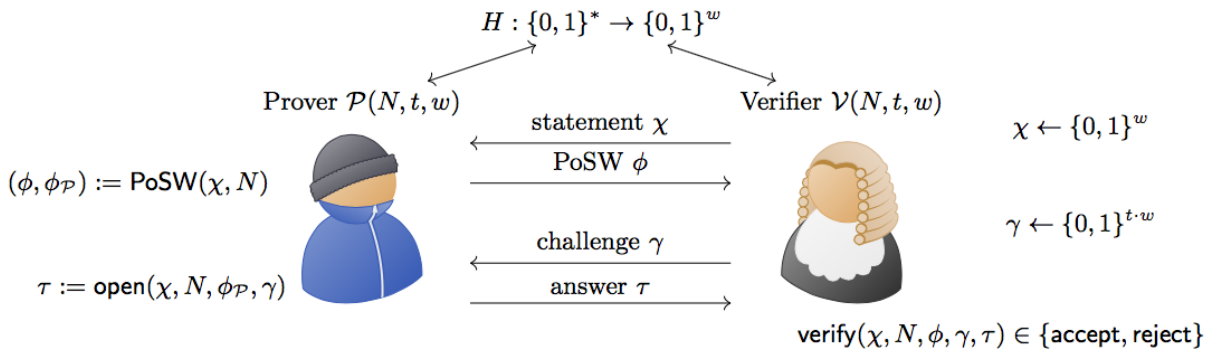

Eurocrypt 2018, Tel Aviv, May 1st 2018

# Outline

- What Proofs of Sequential Work
- How Sketch of Construction & Proof
- Why Sustainable Blockchains

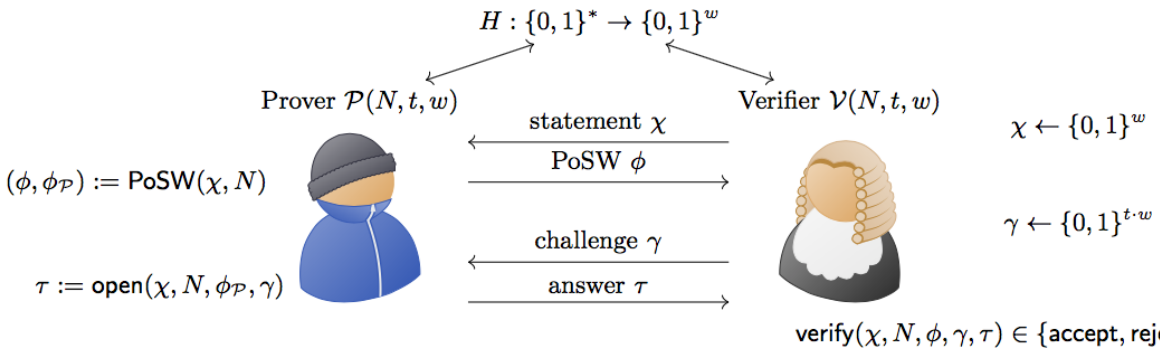
# Outline

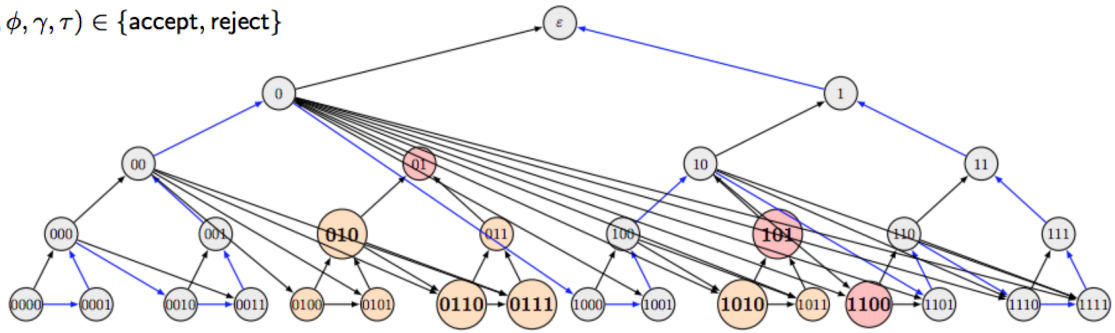
- What Proofs of Sequential Work
- How Sketch of Construction & Proof
- Why Sustainable Blockchains



# Outline

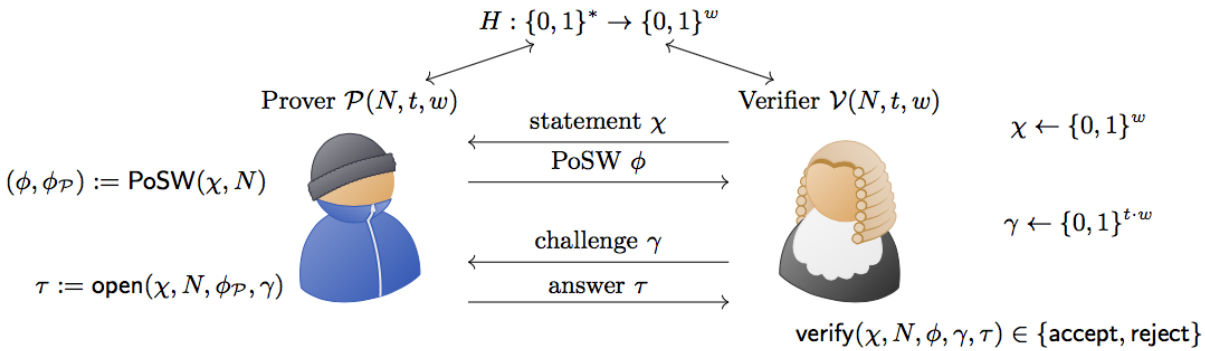
- What Proofs of Sequential Work
- How Sketch of Construction & Proof
- Why Sustainable Blockchains

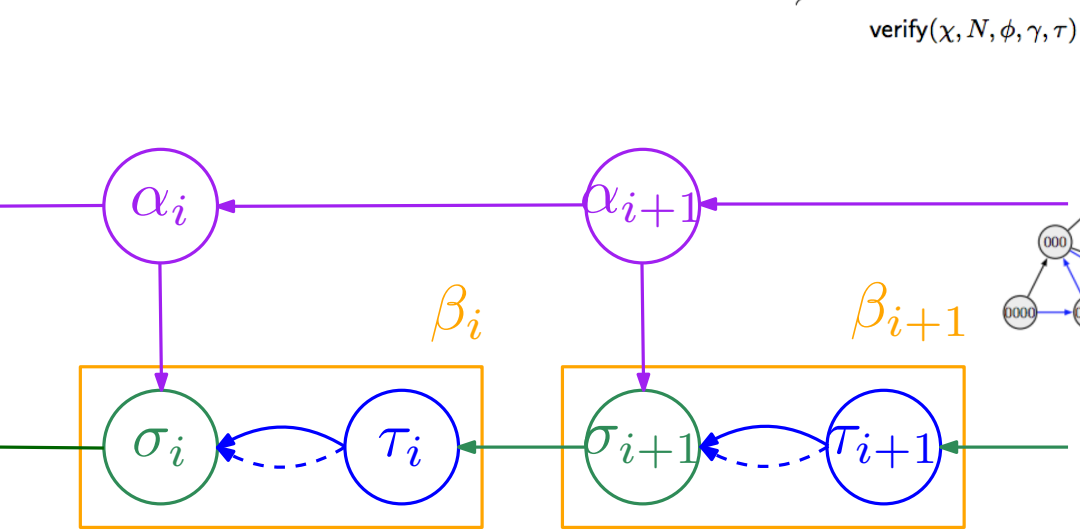


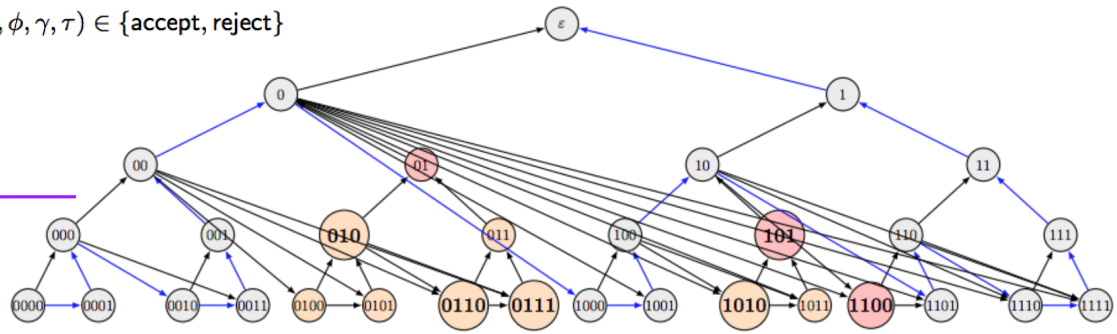



# Outline

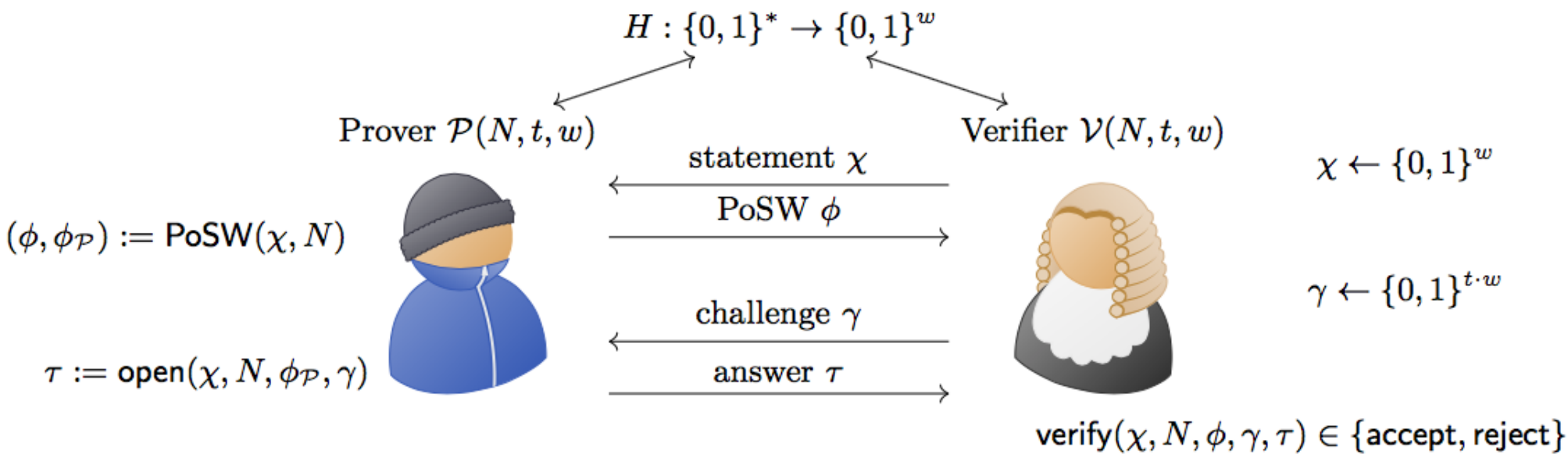
- What Proofs of Sequential Work
- How Sketch of Construction & Proof
- Why Sustainable Blockchains



# Proofs of Sequential Work



# Time-lock puzzles and timed-release Crypto

Ronald L. Rivest*, Adi Shamir**, and David A. Wagner***

Revised March 10, 1996

# Time-lock puzzles and timed-release Crypto

Ronald L. Rivest*, Adi Shamir**, and David A. Wagner***

Revised March 10, 1996

puzzle: $(N = p \cdot q, x, T)$ , solution: $x^{2^T} \mod N$

solution computed with two exponentiation given $p, q$:

$$e \leftarrow 2^T \mod \phi(N) \quad , \quad x^{2^T} = x^e \bmod N$$

conjectured to require $T$ sequential squarings given only $N$

$$x \rightarrow x^2 \rightarrow x^{2^2} \rightarrow \ldots x^{2^T} \mod N$$

# Time-lock puzzles and timed-release Crypto

Ronald L. Rivest*, Adi Shamir**, and David A. Wagner***

Revised March 10, 1996

puzzle: $(N = p \cdot q, x, T)$ , solution: $x^{2^T} \mod N$

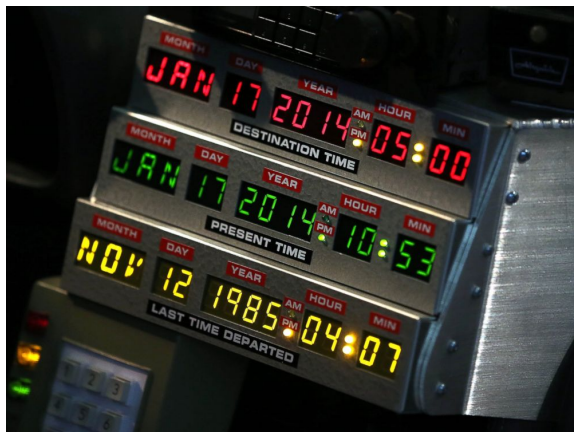
solution computed with two exponentiation given $p, q$:

$$e \leftarrow 2^T \mod \phi(N) \quad , \quad x^{2^T} = x^e \bmod N$$

conjectured to require $T$ sequential squarings given only $N$

$$x \rightarrow x^2 \rightarrow x^{2^2} \rightarrow \ldots x^{2^T} \mod N$$

sequential computation $\sim$
computation time $\Rightarrow$
"send message to the future"



# Publicly Verifiable Proofs of Sequential Work

Mohammad Mahmoody* Tal Moran† Salil Vadhan‡

February 18, 2013

# PoSW vs. Time-Lock Puzzles

Publicly Verifiable Proofs of Sequential Work

Mohammad Mahmoody* Tal Moran† Salil Vadhan‡

February 18, 2013

Time-lock puzzles and timed-release Crypto

Ronald L. Rivest*, Adi Shamir**, and David A. Wagner***

Revised March 10, 1996

## Functionality

- Prove that time has passed
  $\Rightarrow$ Non-interactive time-stamps
- Send message to the future

# PoSW vs. Time-Lock Puzzles

Publicly Verifiable Proofs of Sequential Work

Mohammad Mahmoody* Tal Moran† Salil Vadhan‡

February 18, 2013

Time-lock puzzles and timed-release Crypto

Ronald L. Rivest*, Adi Shamir**, and David A. Wagner***

Revised March 10, 1996

## Functionality

- Prove that time has passed
  $\Rightarrow$ Non-interactive time-stamps
- Send message to the future

## Assumption

- Random oracle model or "sequential" hash-function
- Non-standard algebraic assumption

# PoSW vs. Time-Lock Puzzles

Publicly Verifiable Proofs of Sequential Work

Mohammad Mahmoody* Tal Moran† Salil Vadhan‡

February 18, 2013

Time-lock puzzles and timed-release Crypto

Ronald L. Rivest*, Adi Shamir**, and David A. Wagner***

Revised March 10, 1996

## Functionality

- Prove that time has passed $\Rightarrow$ Non-interactive time-stamps
- Send message to the future

## Assumption

- Random oracle model or "sequential" hash-function
- Non-standard algebraic assumption

## Public vs. Private

- Public-coin $\Rightarrow$ Publicly verfiable
- Private-coin $\Rightarrow$ Designated verifier

# Proofs of Sequential Work

aka. Verifiable Delay Algorithm

Prover $\mathcal{P}$

statement $\chi$

Time $T \in \mathbb{N}$

Verifier $\mathcal{V}$

$\chi \leftarrow$ 

# Proofs of Sequential Work

aka. Verifiable Delay Algorithm

Prover $\mathcal{P}$

statement $\chi$

Time $T \in \mathbb{N}$

$\tau = \tau(\chi, T)$

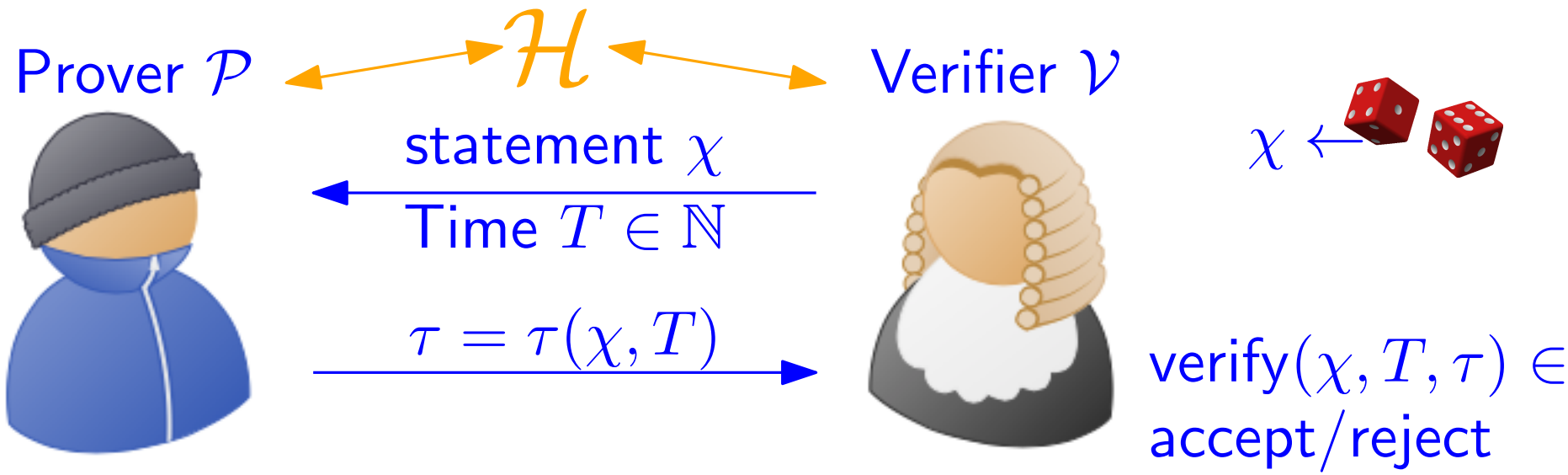
Verifier $\mathcal{V}$

$\chi \leftarrow$ 

verify$(\chi, T, \tau) \in$
accept/reject

# Proofs of Sequential Work

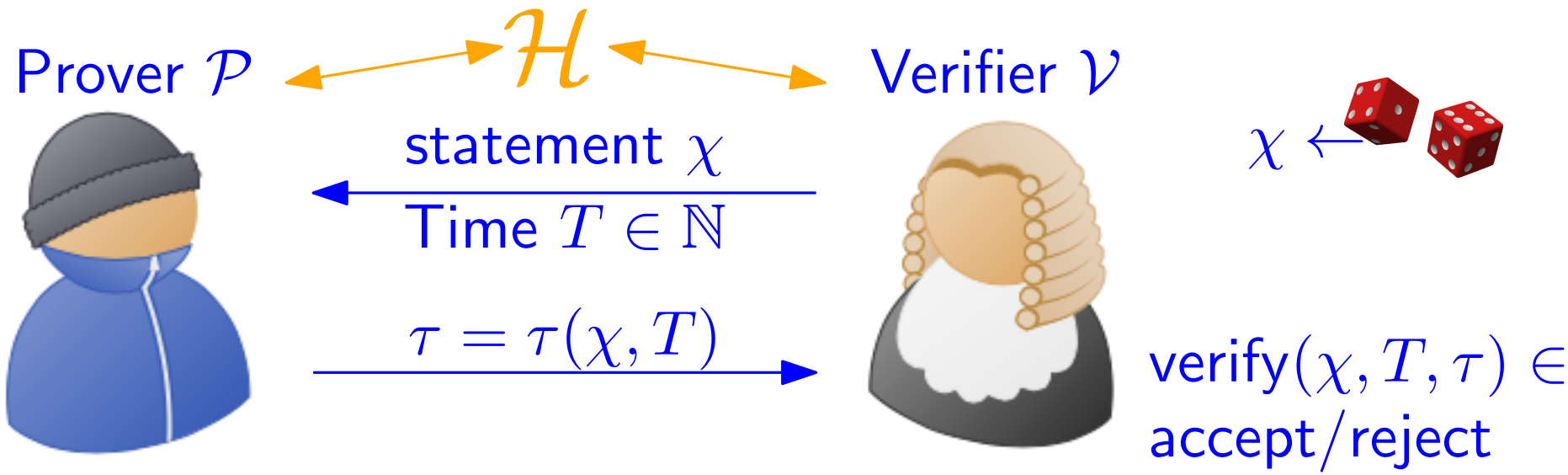
aka. Verifiable Delay Algorithm




Completeness and Soundness *in the random oracle model:*

# Proofs of Sequential Work

aka. Verifiable Delay Algorithm



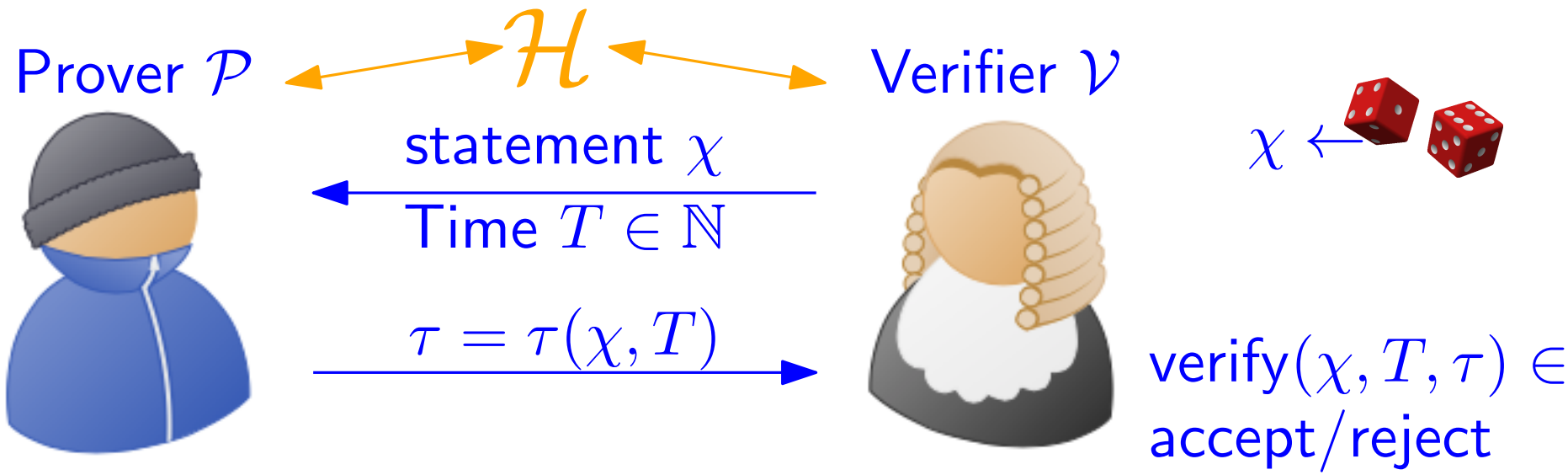

Completeness and Soundness *in the random oracle model:*

**Completeness:** $\tau(c, T)$ can be computed making $T$ queries to $\mathcal{H}$

**Soundness:** Computing any $\tau'$ s.t. verify$(\chi, T, \tau') =$accept for random $\chi$ requires almost $T$ *sequential* queries to $\mathcal{H}$

# Proofs of Sequential Work

aka. Verifiable Delay Algorithm




Completeness and Soundness *in the random oracle model:*

**Completeness:** $\tau(c, T)$ can be computed making $T$ queries to $\mathcal{H}$

**Soundness:** Computing any $\tau'$ s.t. verify$(\chi, T, \tau') =$accept for random $\chi$ requires almost $T$ *sequential* queries to $\mathcal{H}$

massive parallelism useless to generate valid proof faster $\Rightarrow$ prover must make almost $T$ sequential queries $\sim T$ time

# Three Problems of the [MMV'13] PoSW

1) **Space Complexity :** Prover needs massive (linear in T) space to compute proof.
2) **Poor/Unclear Parameters** due to usage of sophisticated combinatorial objects.
3) **Uniqueness :** Once an accepting proof is computed, many other valid proofs can be generated (not a problem for time-stamping, but for blockchains).

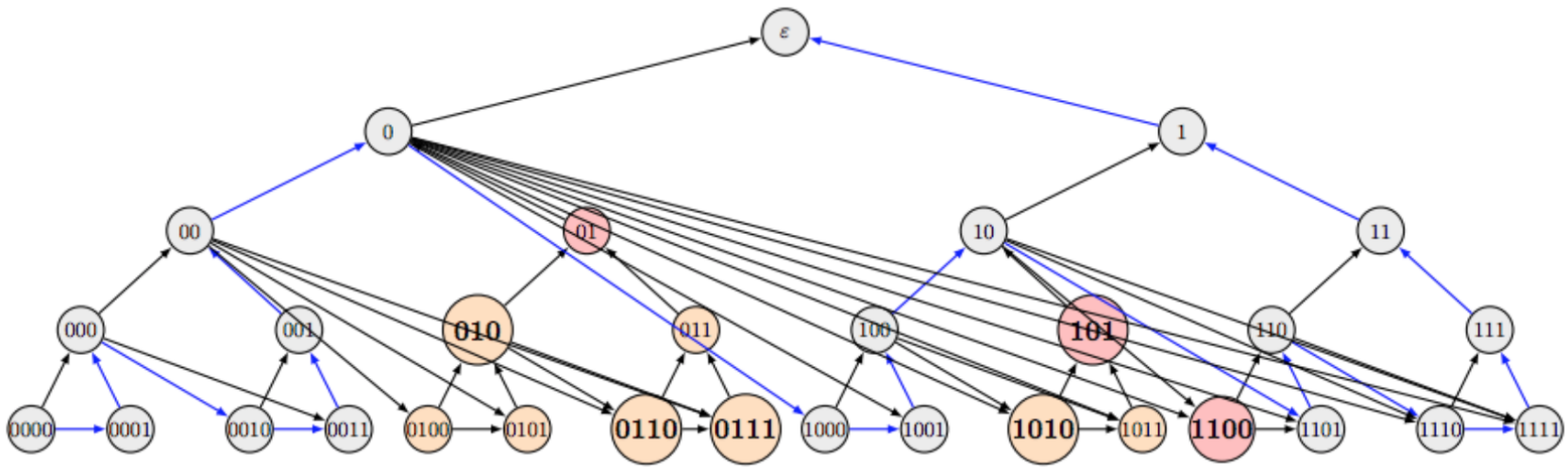
# Three Problems of the [MMV'13] PoSW

1) **Space Complexity :** Prover needs massive (linear in T) space to compute proof.
2) **Poor/Unclear Parameters** due to usage of sophisticated combinatorial objects.
3) **Uniqueness :** Once an accepting proof is computed, many other valid proofs can be generated (not a problem for time-stamping, but for blockchains).

# New Construction

1) Prover needs only $O(log(T))$ (not $O(T)$) space, e.g. for $T = 2^{42}$ ($\approx$ a day) that's $\approx 10KB$ vs. $\approx 1PB$.
2) Simple construction and proof with good concrete parameters.
3) Awesome open problem!

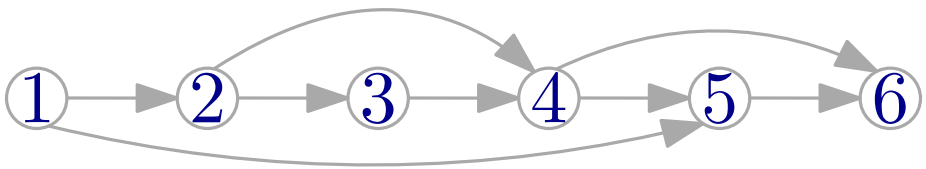
# Construction and Proof Sketch



# Three Basic Concepts

# Three Basic Concepts

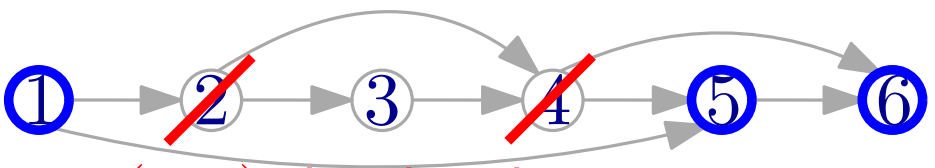
**Depth-Robust** Graphs (only [MMV'13])




DAG $G = (V, E)$ is $(e, d)$ **depth-robust** if after removing any $e$ nodes a path of length $d$ exists.

# Three Basic Concepts

## **Depth-Robust** Graphs (only [MMV'13])

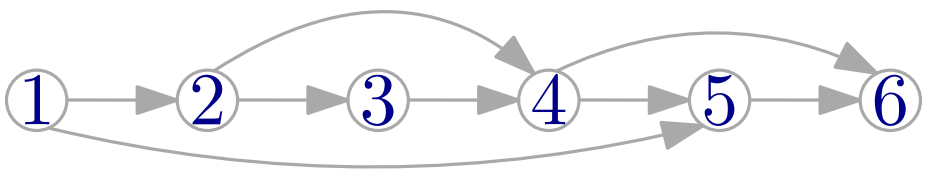



is $(2, 3)$ depth-robust

DAG $G = (V, E)$ is $(e, d)$ **depth-robust** if after removing any $e$ nodes a path of length $d$ exists.

# Three Basic Concepts

## **Depth-Robust** Graphs (only [MMV'13])



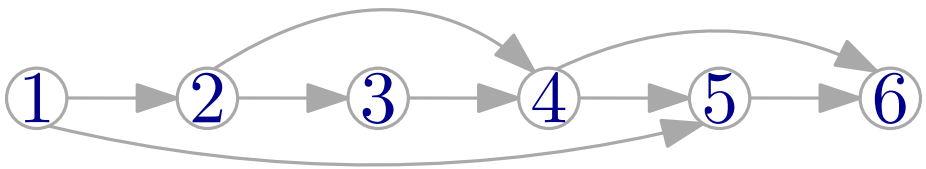

DAG $G = (V, E)$ is $(e, d)$ **depth-robust** if after removing any $e$ nodes a path of length $d$ exists.

## Graph **Labelling**

label $\ell_i = \mathcal{H}(\ell_{parents(i)})$, e.g. $\ell_4 = \mathcal{H}(\ell_3, \ell_4)$

# Three Basic Concepts

## **Depth-Robust** Graphs (only [MMV'13])

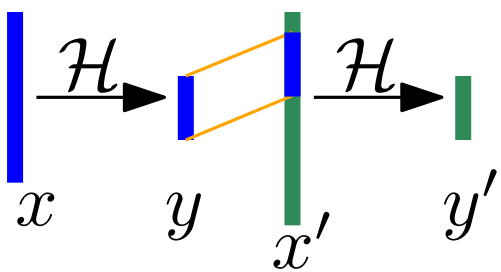



DAG $G = (V, E)$ is $(e, d)$ **depth-robust** if after removing any $e$ nodes a path of length $d$ exists.

## Graph **Labelling**

label $\ell_i = \mathcal{H}(\ell_{parents(i)})$, e.g. $\ell_4 = \mathcal{H}(\ell_3, \ell_4)$

## Random Oracles are **Sequential**




queries $y = \mathcal{H}(x), y' = \mathcal{H}(x')$ where $y \subseteq x' \Rightarrow$ query $x'$ was made after $x$

# The MMV'13 Construction

Prover $\mathcal{P}$ $\leftrightarrow$ $\mathcal{H}$ $\leftrightarrow$ Verifier $\mathcal{V}$

statement $\chi$

Time $T = 6$

$\chi \leftarrow$ 

# The MMV'13 Construction

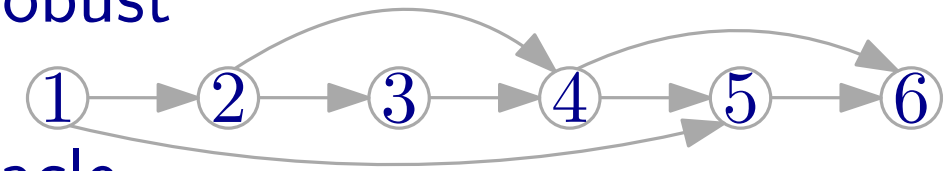
Prover $\mathcal{P}$ $\leftrightarrow$ $\mathcal{H}$ $\leftrightarrow$ Verifier $\mathcal{V}$

statement $\chi$

Time $T = 6$

$\chi \leftarrow$ 

- Protocol specifies depth-robust DAG $G$ on $T$ nodes
- Define "fresh" random oracle $\mathcal{H}_\chi(\cdot) \equiv \mathcal{H}(\chi \| \cdot)$



# The MMV'13 Construction

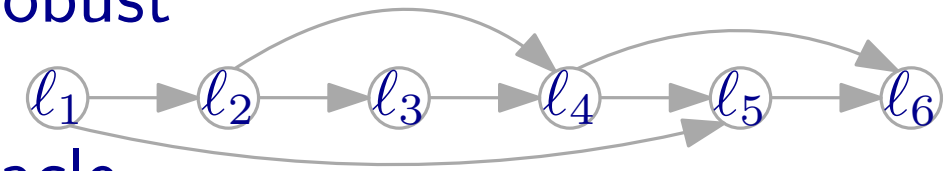
Prover $\mathcal{P}$ $\leftrightarrow$ $\mathcal{H}$ $\leftrightarrow$ Verifier $\mathcal{V}$

statement $\chi$

Time $T = 6$

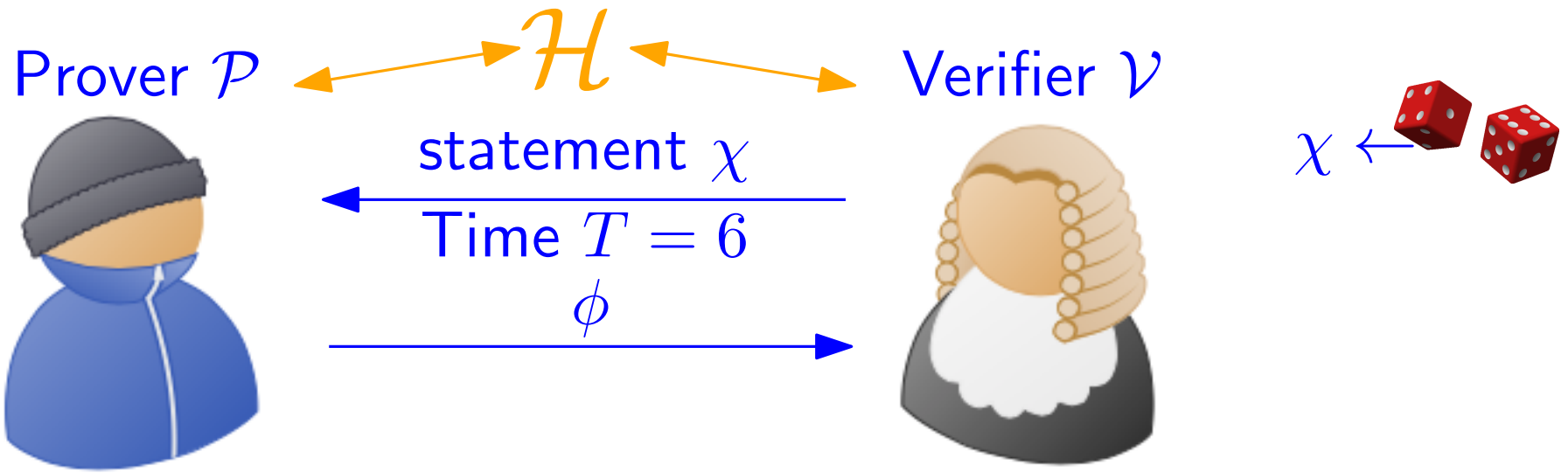
$\chi \leftarrow$ 

- Protocol specifies depth-robust DAG $G$ on $T$ nodes

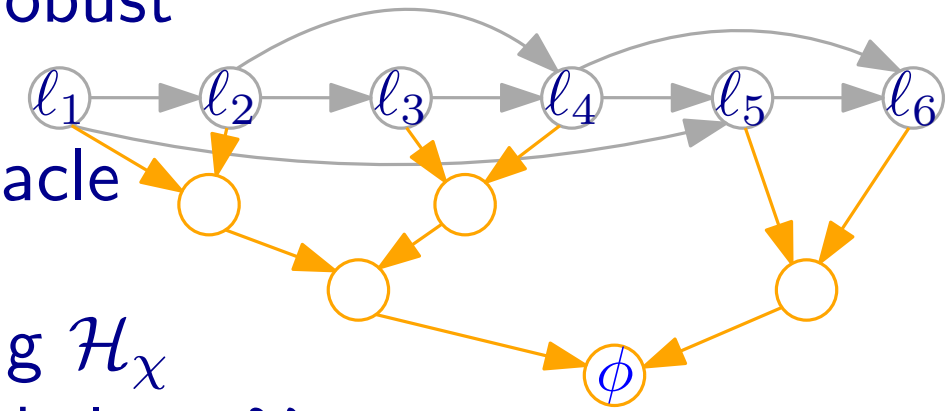



- Define "fresh" random oracle $\mathcal{H}_\chi(\cdot) \equiv \mathcal{H}(\chi \| \cdot)$
- Compute labels of $G$ using $\mathcal{H}_\chi$

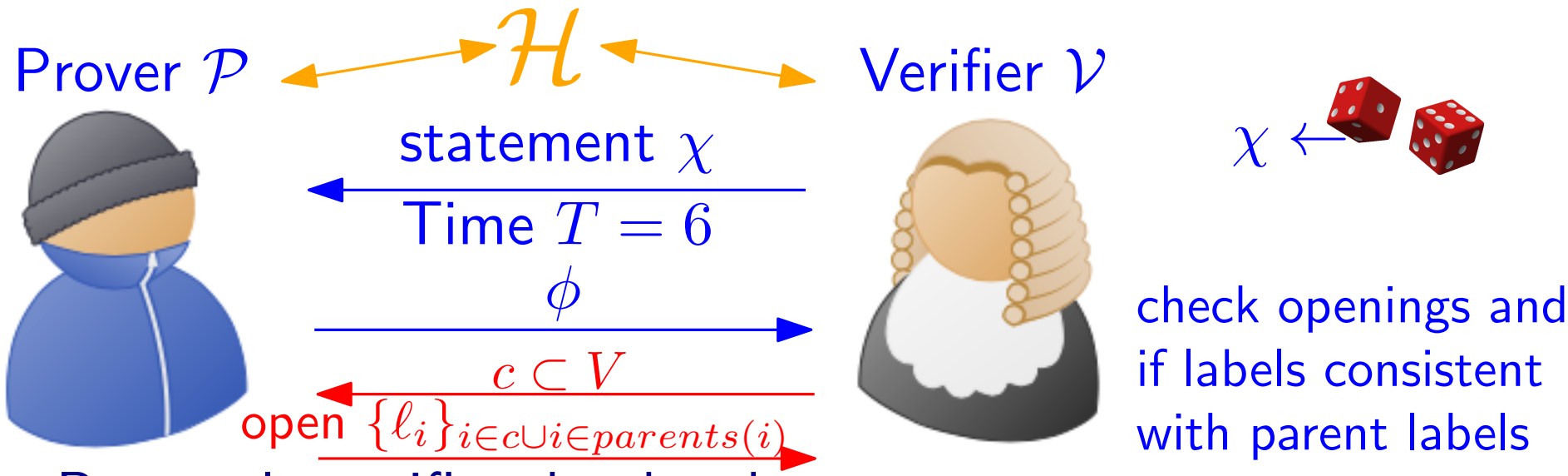
# The MMV'13 Construction




- Protocol specifies depth-robust DAG $G$ on $T$ nodes
- Define "fresh" random oracle $\mathcal{H}_\chi(\cdot) \equiv \mathcal{H}(\chi \| \cdot)$
- Compute labels of $G$ using $\mathcal{H}_\chi$
- Send commitment $\phi$ to labels to $\mathcal{V}$

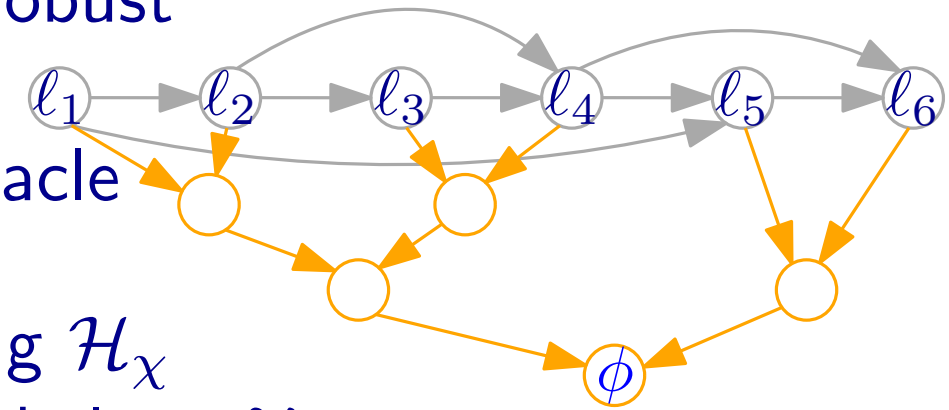


# The MMV'13 Construction

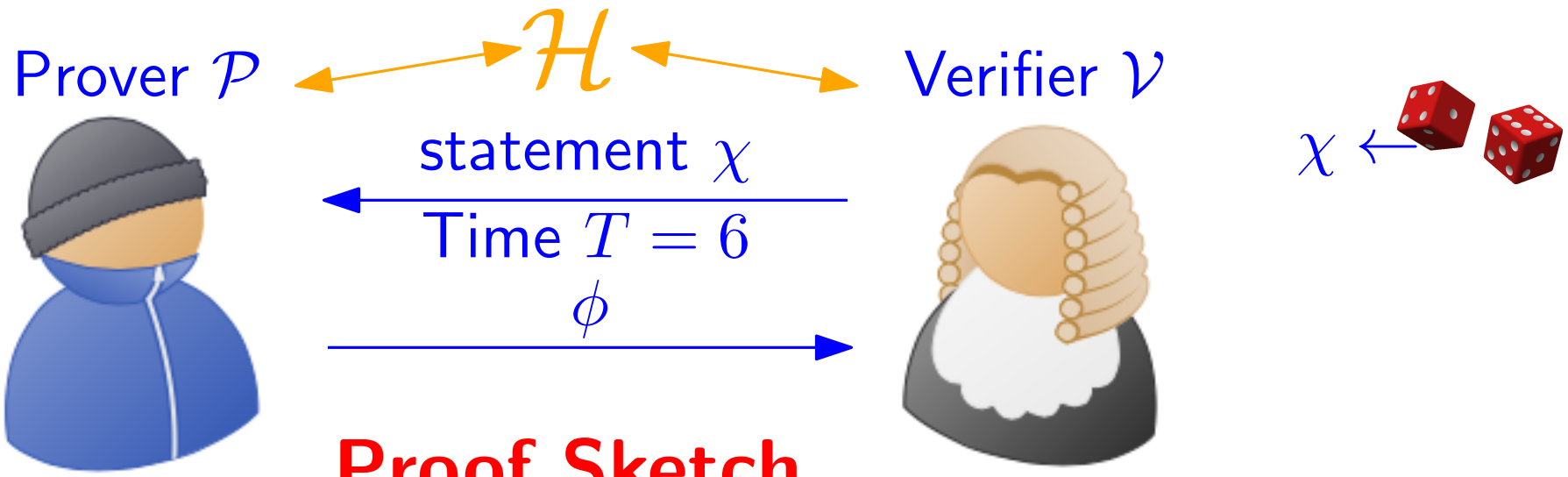



- Protocol specifies depth-robust DAG $G$ on $T$ nodes
- Define "fresh" random oracle $\mathcal{H}_\chi(\cdot) \equiv \mathcal{H}(\chi \| \cdot)$
- Compute labels of $G$ using $\mathcal{H}_\chi$
- Send commitment $\phi$ to labels to $\mathcal{V}$
- $\mathcal{V}$ challenged to open random subset of nodes and parents

(interaction can be removed using Fiat-Shamir)

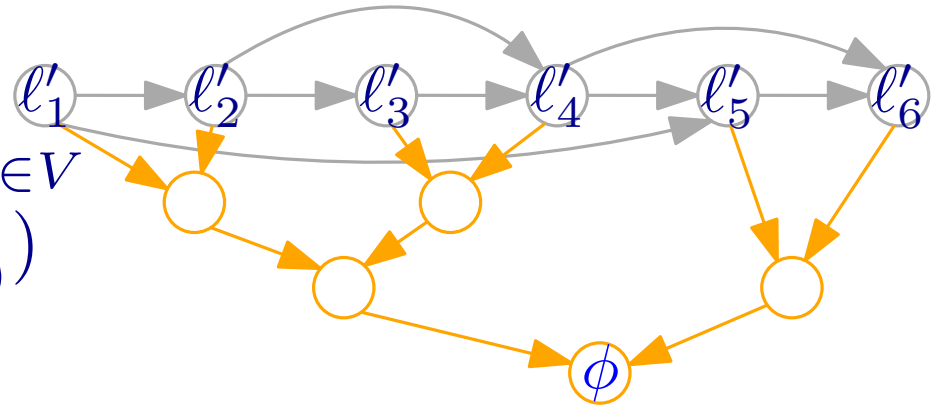


# The MMV'13 Construction




## Proof Sketch

- $G$ is $(e, d)$ depth-robust
- $\phi$ commits $\tilde{\mathcal{P}}$ to labels $\{\ell'_i\}_{i \in V}$
- $i$ is **bad** if $\ell'_i \neq \mathcal{H}(\ell'_{parents(i)})$



# The MMV'13 Construction

Prover $\mathcal{P}$ $\leftrightarrow$ $\mathcal{H}$ $\leftrightarrow$ Verifier $\mathcal{V}$

statement $\chi$

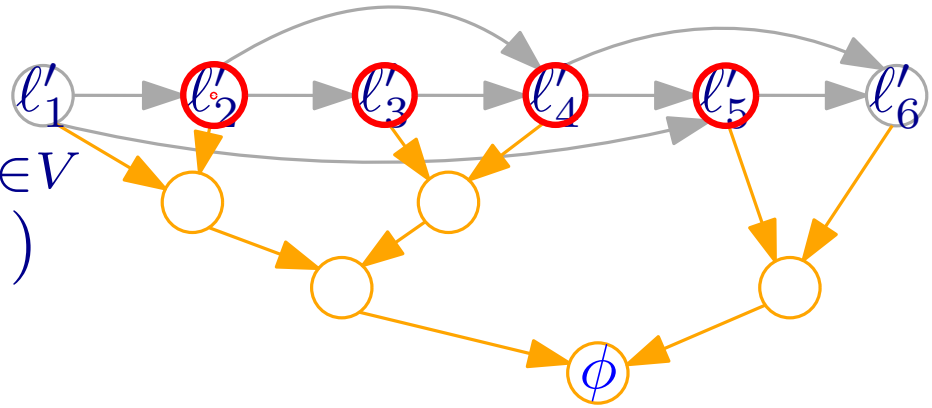
Time $T = 6$

$\phi$

$\chi \leftarrow$ 

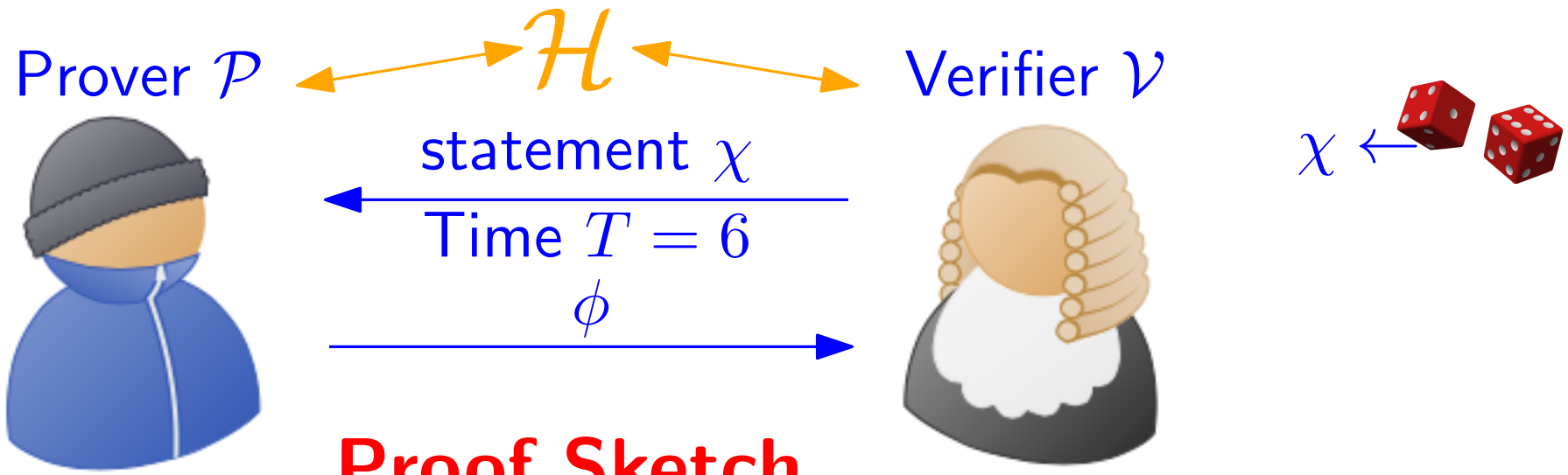
## Proof Sketch

- $G$ is $(e, d)$ depth-robust
- $\phi$ commits $\tilde{\mathcal{P}}$ to labels $\{\ell'_i\}_{i \in V}$
- $i$ is **bad** if $\ell'_i \neq \mathcal{H}(\ell'_{parents(i)})$



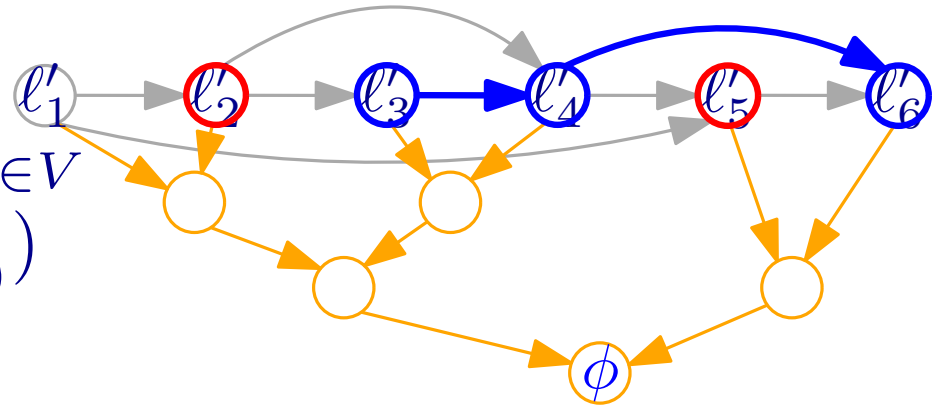

- Case 1: $\geq e$ bad nodes $\Rightarrow$ will fail opening phase whp.

# The MMV'13 Construction



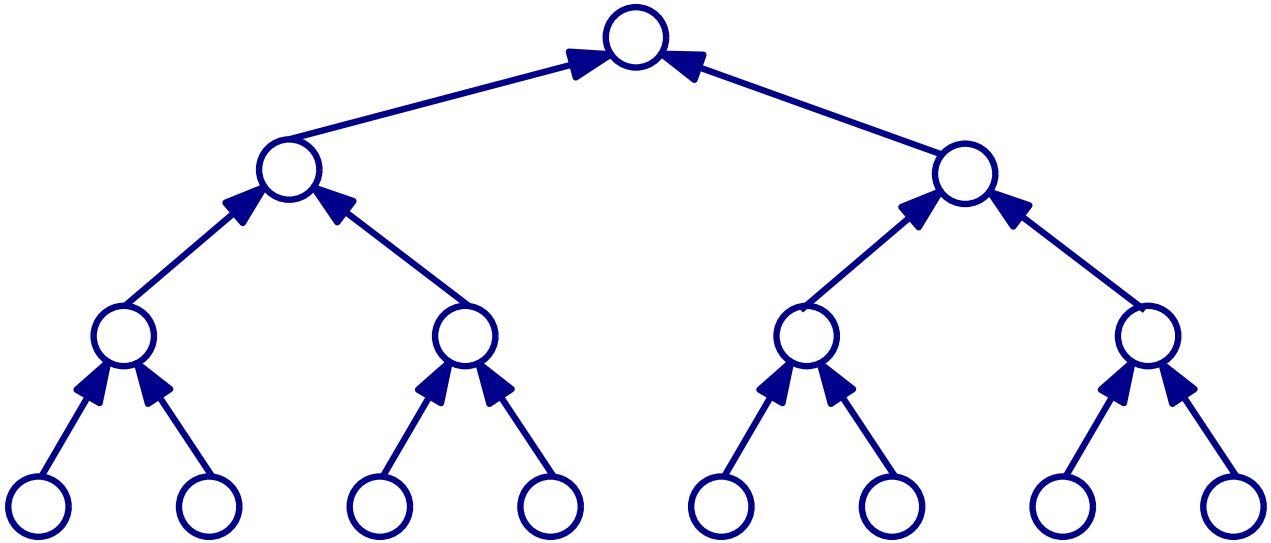

## Proof Sketch

- $G$ is $(e, d)$ depth-robust
- $\phi$ commits $\tilde{\mathcal{P}}$ to labels $\{\ell'_i\}_{i \in V}$
- $i$ is **bad** if $\ell'_i \neq \mathcal{H}(\ell'_{parents(i)})$



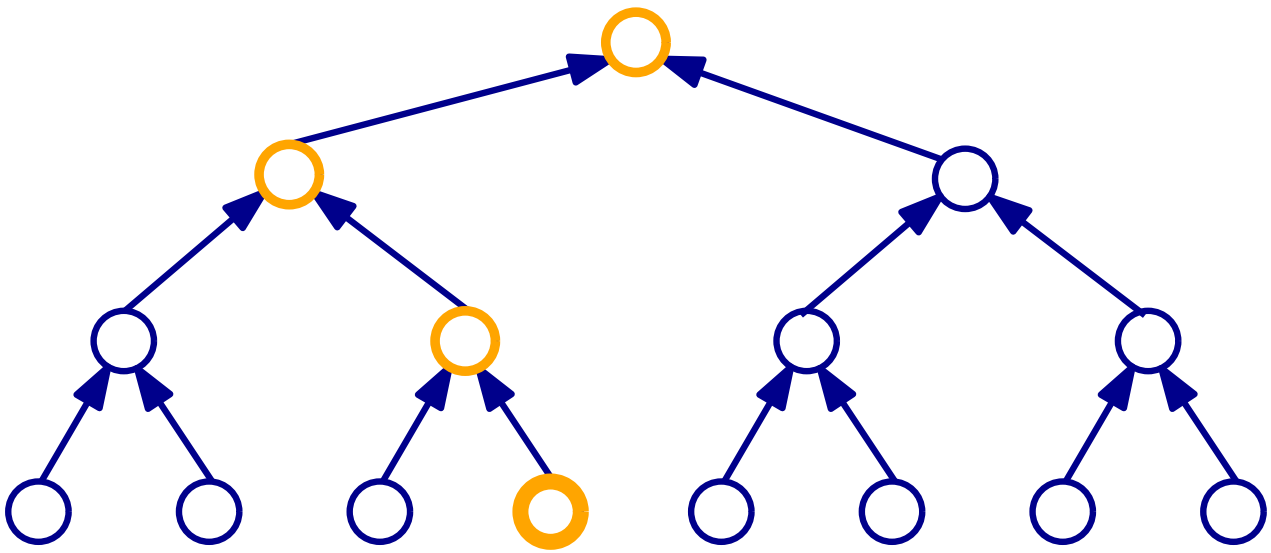

- Case 1: $\geq e$ bad nodes $\Rightarrow$ will fail opening phase whp.
- Case 2: Less than $e$ bad labels $\Rightarrow \exists$ path of good nodes (by $(e, d)$ depth-robustness) $\Rightarrow \tilde{P}$ made $d$ sequential queries (by sequantality of RO)

# The New Construction



$T = 15$

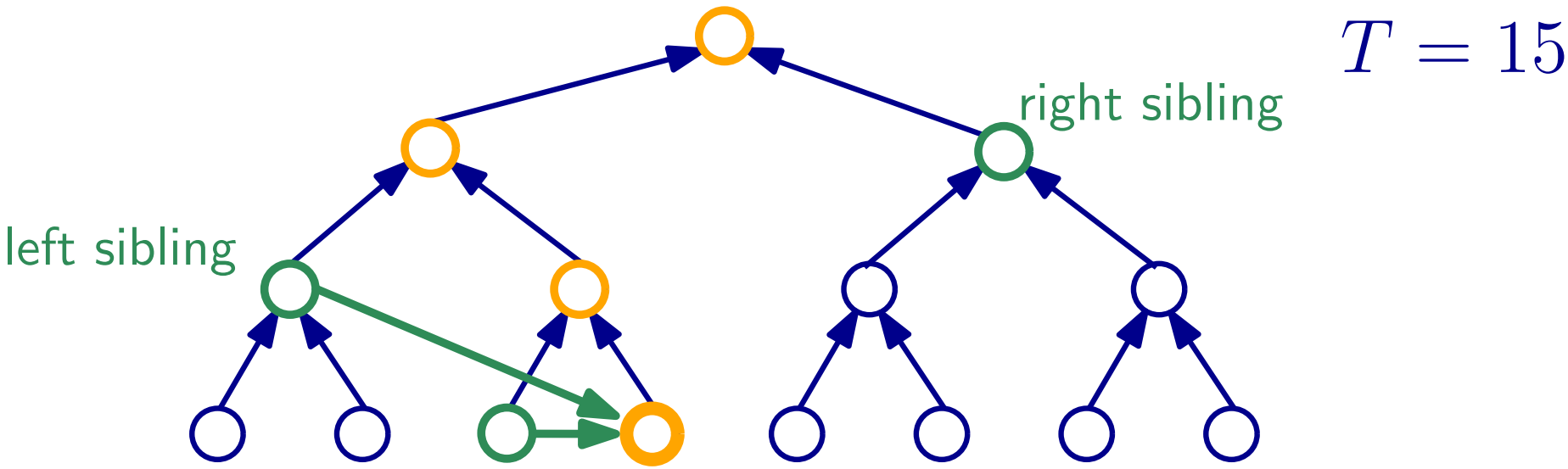
# The New Construction



$T = 15$

For every leaf $i$ add all edges $(j, i)$ where $j$ is left sibling of node on path $i \to root$

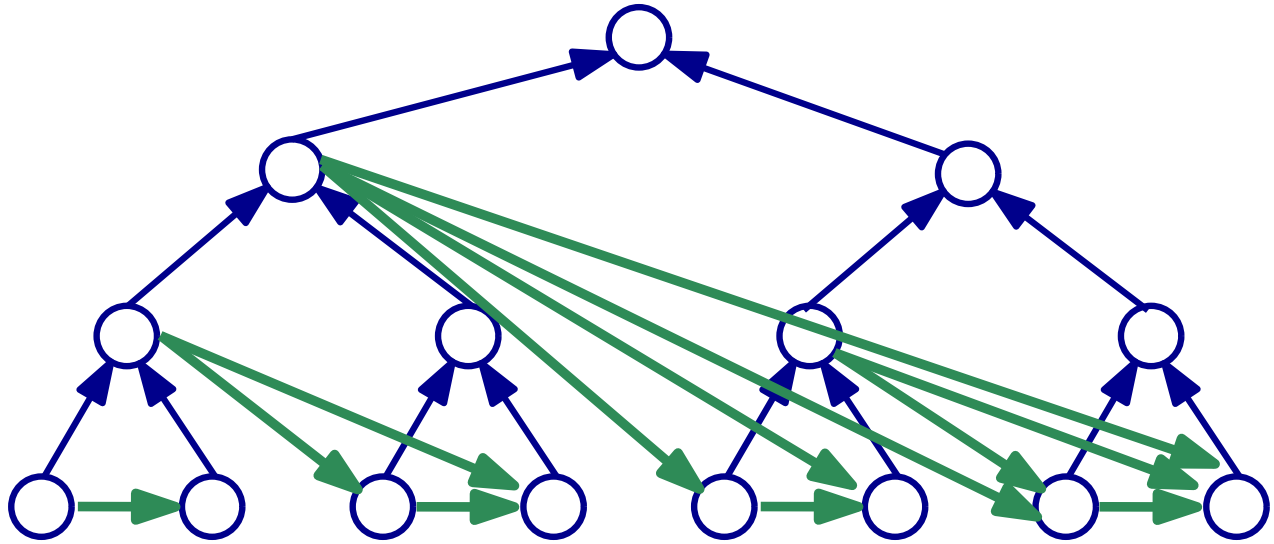
# The New Construction




For every leaf $i$ add all edges $(j, i)$ where $j$ is left sibling of node on path $i \to root$

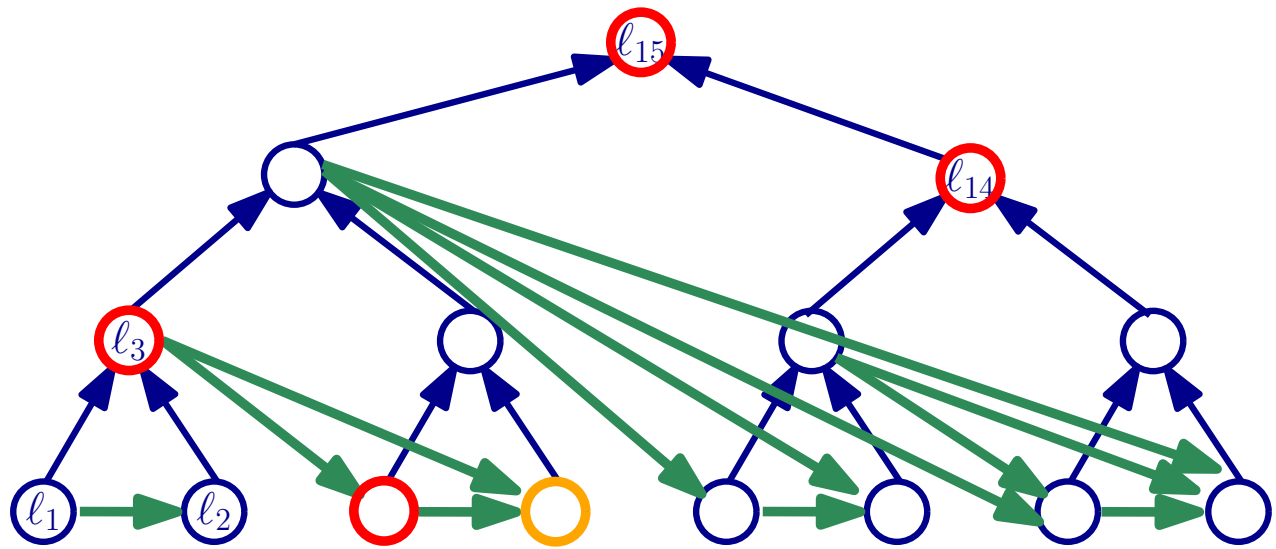
# The New Construction

$T = 15$



For every leaf $i$ add all edges $(j, i)$ where $j$ is left sibling of node on path $i \to root$

# The New Construction

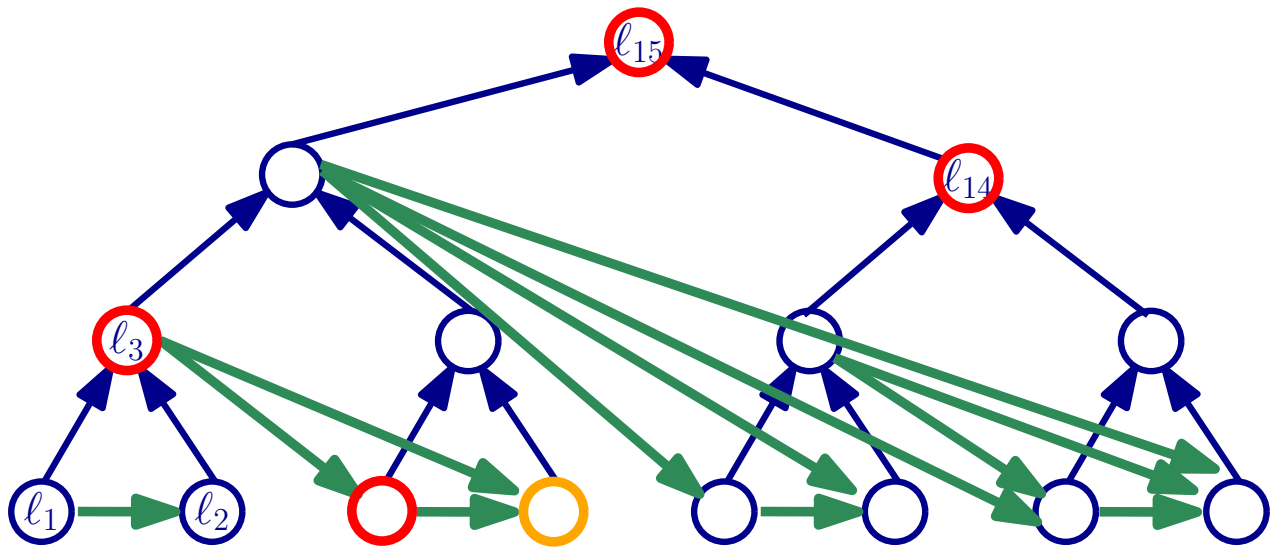



$T = 15$

For every leaf $i$ add all edges $(j, i)$ where $j$ is left sibling of node on path $i \to root$

- $\mathcal{P}$ computes labelling $\ell_i = \mathcal{H}(\ell_{parents(i)})$ and sends root label $\phi = \ell_T$ to $\mathcal{V}$. Can be done storing only $\log(T)$ labels.
- $\mathcal{V}$ challenges $\mathcal{P}$ to open a subset of leaves and checks consistency (blue and green edges!)

# The New Construction




$T = 15$

For every leaf $i$ add all edges $(j, i)$ where $j$ is left sibling of node on path $i \to root$

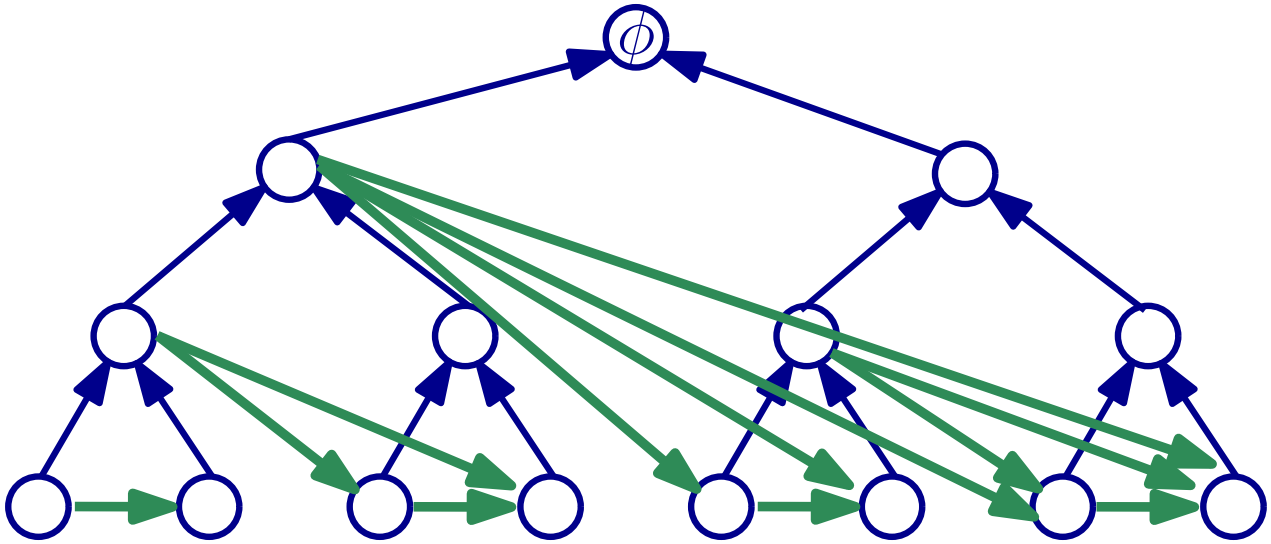
- $\mathcal{P}$ computes labelling $\ell_i = \mathcal{H}(\ell_{parents(i)})$ and sends root label $\phi = \ell_T$ to $\mathcal{V}$. Can be done storing only $\log(T)$ labels.
- $\mathcal{V}$ challenges $\mathcal{P}$ to open a subset of leaves and checks consistency (blue and green edges!)

**Optimally Efficient Accountable Time-Stamping**

PKC'00

Ahto Buldas*[1], Helger Lipmaa*[1], and Berry Schoenmakers**[2]

# The New Construction

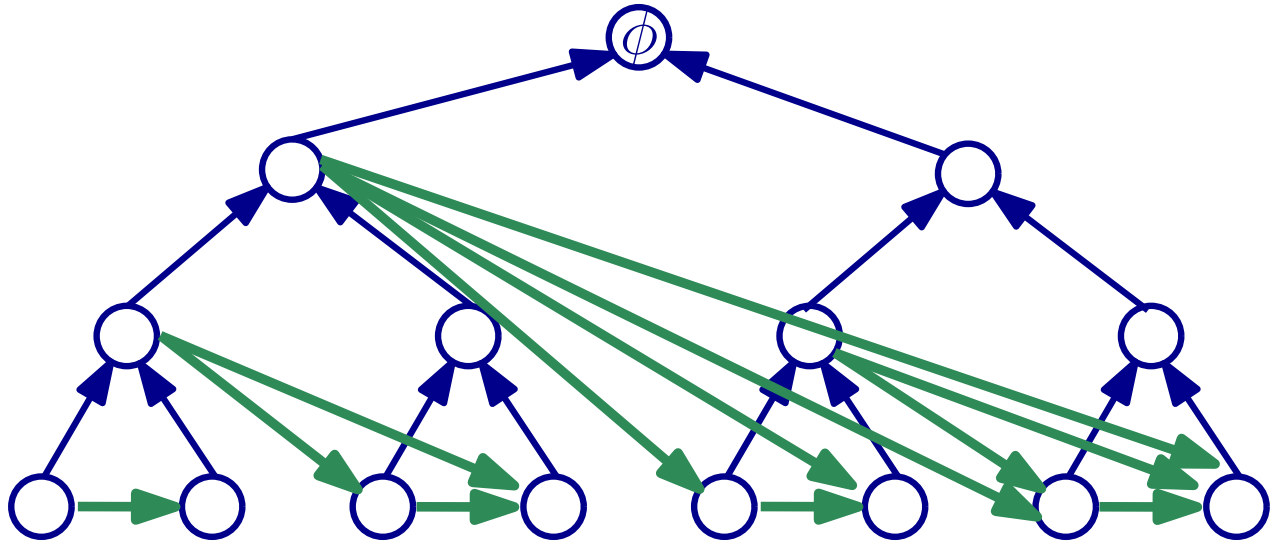



$T = 15$

**Proof Sketch**

# The New Construction

$T = 15$

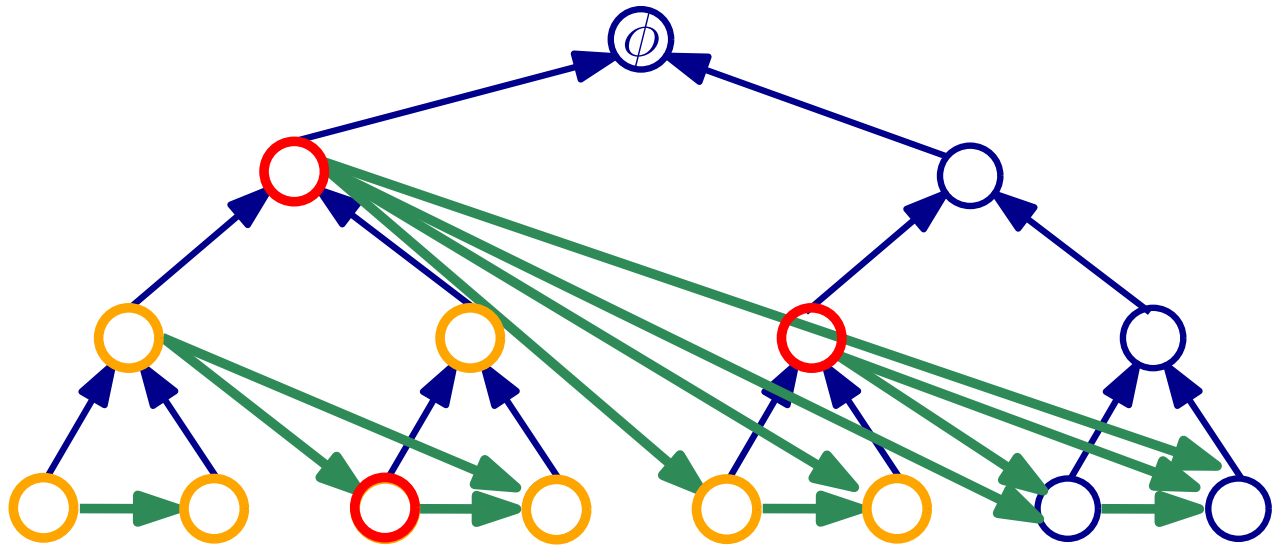



## Proof Sketch

- $\tilde{P}$ committed to labels $\ell'_i$ after sending $\phi = \ell_{15}$.
- $i$ is bad if $\ell'_i \neq \mathcal{H}(\ell'_{parents(i)})$.

# The New Construction

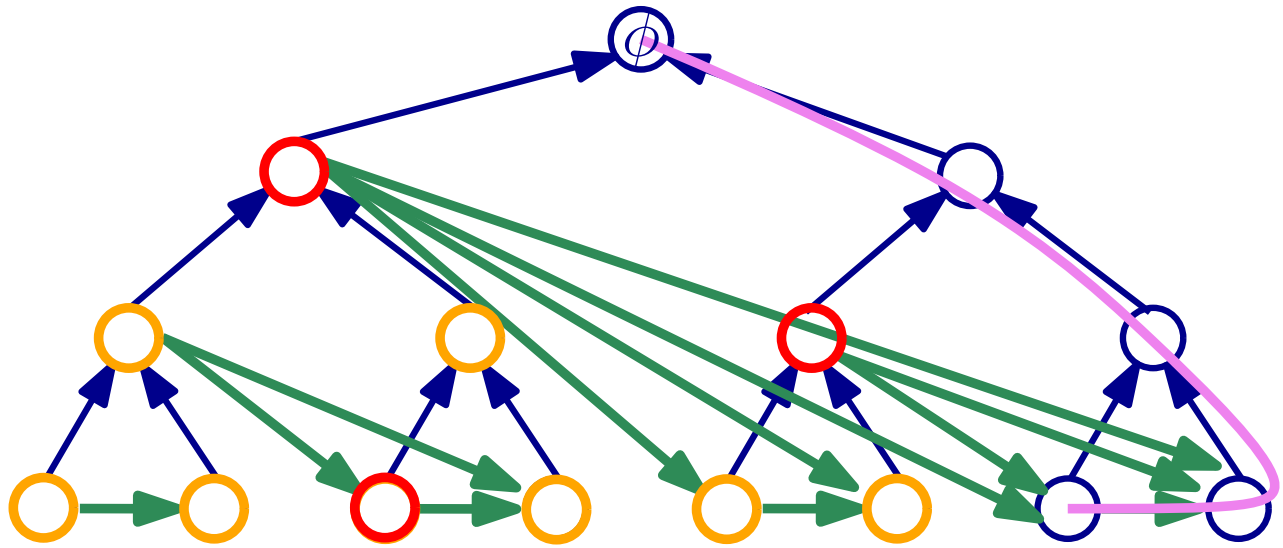
$T = 15$




## Proof Sketch

- $\tilde{P}$ committed to labels $\ell'_i$ after sending $\phi = \ell_{15}$.
- $i$ is bad if $\ell'_i \neq \mathcal{H}(\ell'_{parents(i)})$.
- Let $S \subset V$ denote the bad nodes and all nodes below.

# The New Construction

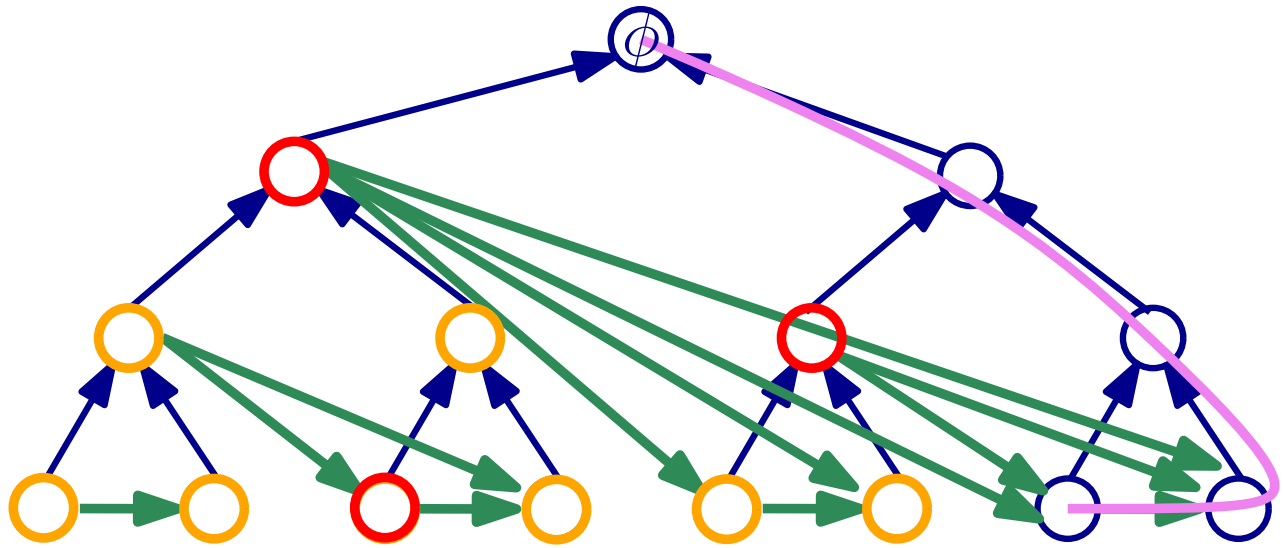



$T = 15$

## Proof Sketch

- $\tilde{P}$ committed to labels $\ell'_i$ after sending $\phi = \ell_{15}$.
- $i$ is bad if $\ell'_i \neq \mathcal{H}(\ell'_{parents(i)})$.
- Let $S \subset V$ denote the bad nodes and all nodes below.
- Claim 1: $\exists$ path going through $V - S$ (of length $T - |S|$).

# The New Construction

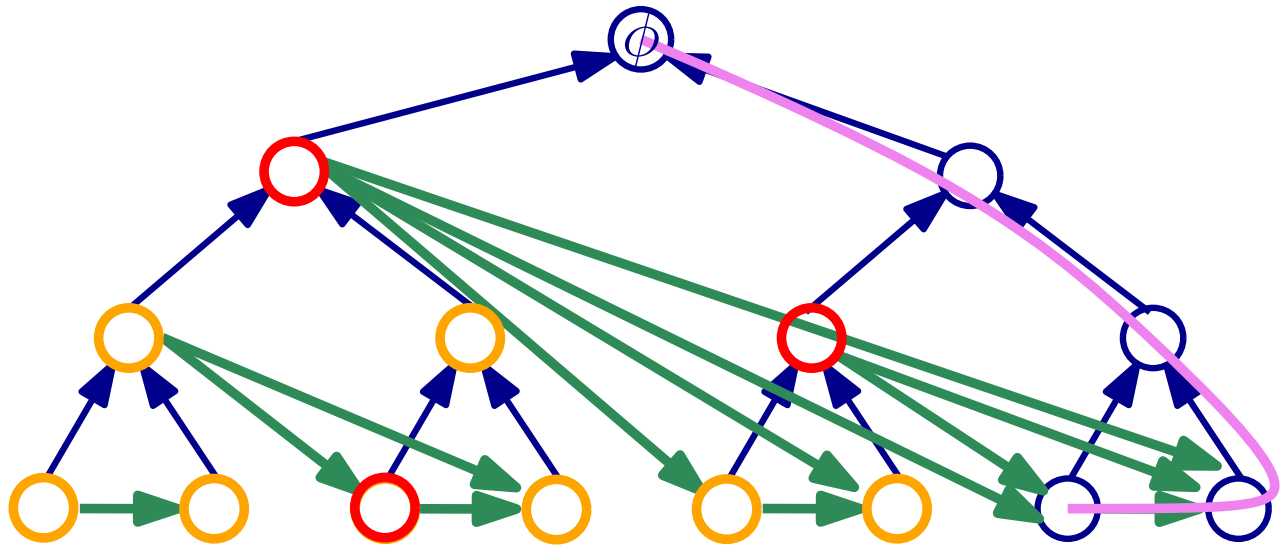


$T = 15$

## Proof Sketch

- $\tilde{P}$ committed to labels $\ell'_i$ after sending $\phi = \ell_{15}$.
- $i$ is bad if $\ell'_i \neq \mathcal{H}(\ell'_{parents(i)})$.
- Let $S \subset V$ denote the bad nodes and all nodes below.
- Claim 1: $\exists$ path going through $V - S$ (of length $T - |S|$).
- Claim 2: $\tilde{P}$ can't open $|S|/T$ fraction of leafs.

# The New Construction




$T = 15$

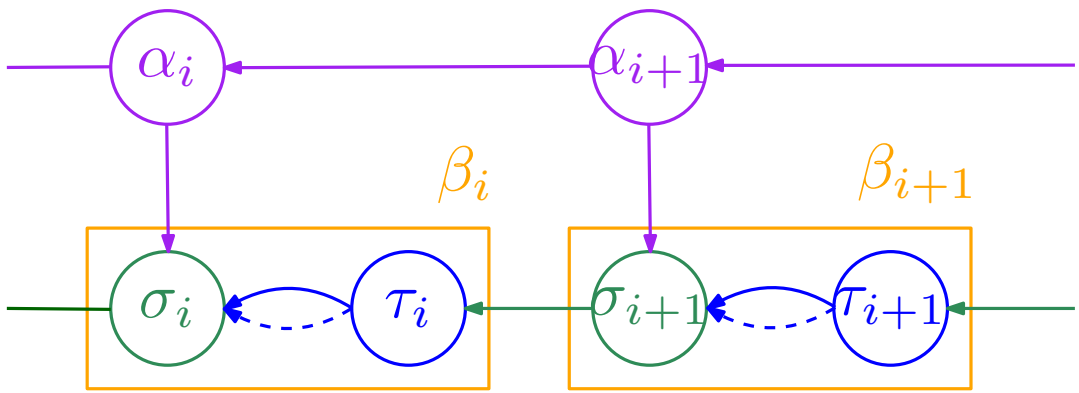
## Proof Sketch

- $\tilde{P}$ committed to labels $\ell'_i$ after sending $\phi = \ell_{15}$.
- $i$ is bad if $\ell'_i \neq \mathcal{H}(\ell'_{parents(i)})$.
- Let $S \subset V$ denote the bad nodes and all nodes below.
- Claim 1: $\exists$ path going through $V - S$ (of length $T - |S|$).
- Claim 2: $\tilde{P}$ can't open $|S|/T$ fraction of leafs.

Theorem: $\tilde{P}$ made only $T(1-\epsilon)$ sequential queries
$\Rightarrow$ will pass opening phase with prob. $\leq (1-\epsilon)^{\#\text{of challenges}}$

why we care

# Sustainable Blockchains



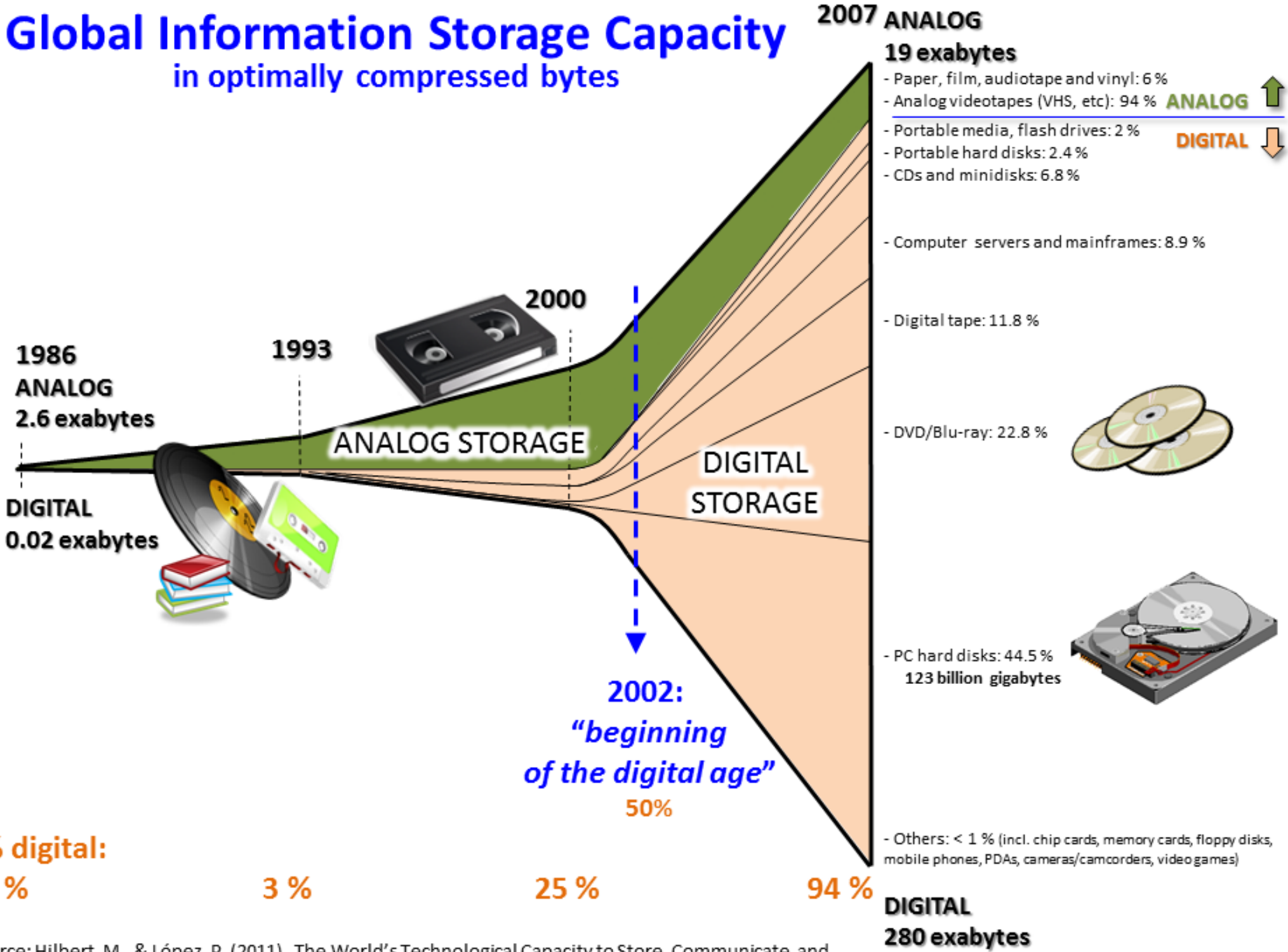
# Mining Bitcoin (Proofs of Work)

# Mining Bitcoin (Proofs of Work)

# Can we have a more "sustainable" Blockchain?

# Global Information Storage Capacity

## in optimally compressed bytes




Source: Hilbert, M., & López, P. (2011). The World's Technological Capacity to Store, Communicate, and Compute Information. *Science*, 332(6025), 60 –65. http://www.martinhilbert.net/WorldInfoCapacity.html

# Proofs of Work

computation as resource

prob. of solving PoW **first** $\sim$ fraction of hashing power

dynamics

proof of work hardness set so blocks appear $\approx$ every 10 minutes


# Proofs of Work

computation as resource

prob. of solving PoW **first** $\sim$ fraction of hashing power

dynamics

proof of work hardness set so blocks appear $\approx$ every 10 minutes


# Proofs of Space and Time

# Proofs of Work

computation as resource

prob. of solving PoW **first** $\sim$ fraction of hashing power

dynamics

proof of work hardness set so blocks appear $\approx$ every 10 minutes

# Proofs of Space and Time

space as resource

prob. of finding PoSpace **of best quality** $\sim$ fraction of dedicated space

# Proofs of Work

computation as resource

prob. of solving PoW **first** $\sim$ fraction of hashing power

dynamics

proof of work hardness set so blocks appear $\approx$ every 10 minutes

# Proofs of Space and Time

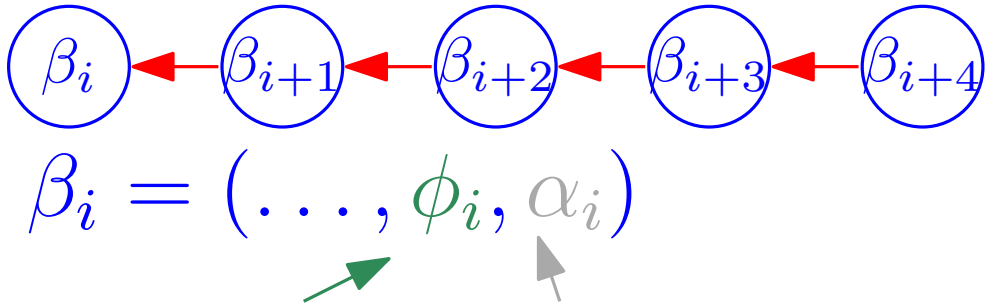
space as resource

prob. of finding PoSpace **of best quality** $\sim$ fraction of dedicated space

dynamics

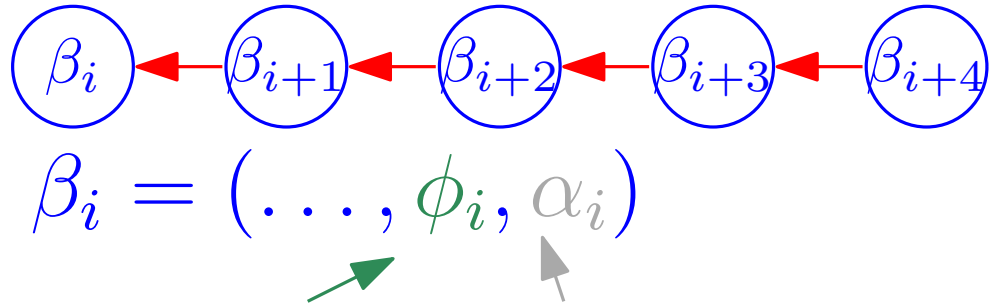
Run PoSW on top of PoSpace for $T \sim$ quality of PoSpace to "finalize" block

bitcoin



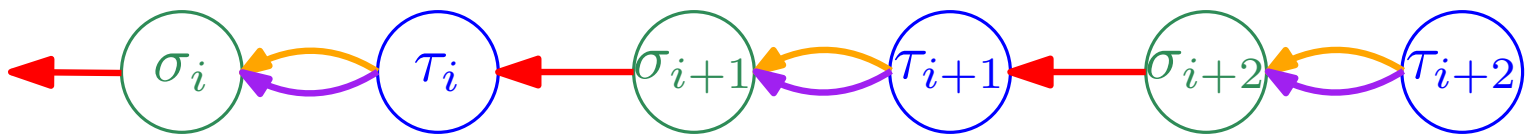

$\phi_i$ : proof of work on challenge $hash(\beta_{i-1})$ transactions

bitcoin




$\phi_i$ : proof of work on challenge $hash(\beta_{i-1})$ transactions

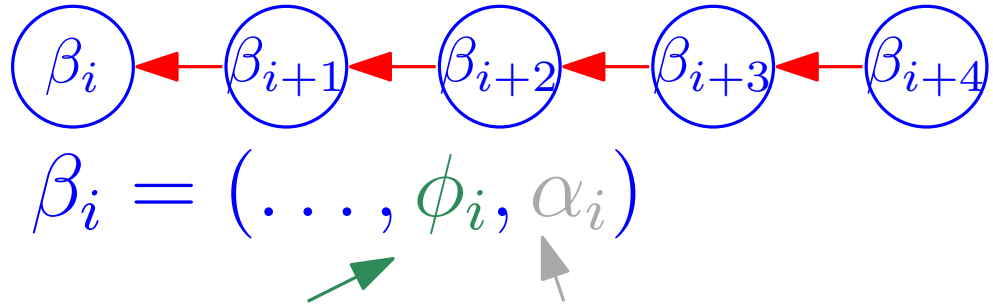
chia




$\sigma_i$ : proof of space on challenge $hash(\tau_{i-1})$

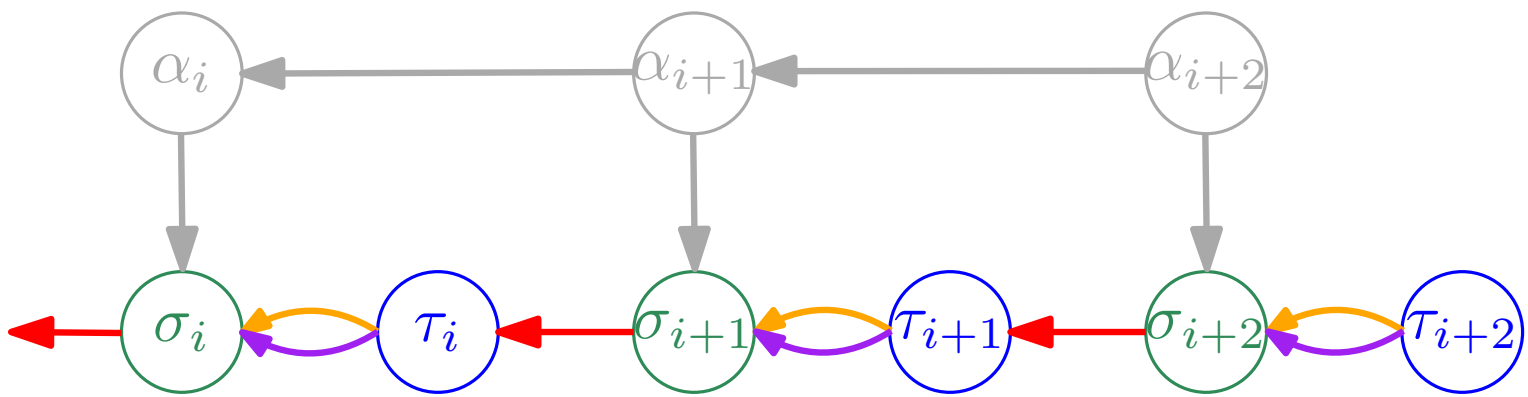
$\tau_i$ : proof of sequential work on challenge $hash(\sigma_{i-1})$
and time parameter $quality(\sigma_{i-1})$

bitcoin




$\phi_i$ : proof of work on challenge $hash(\beta_{i-1})$ transactions

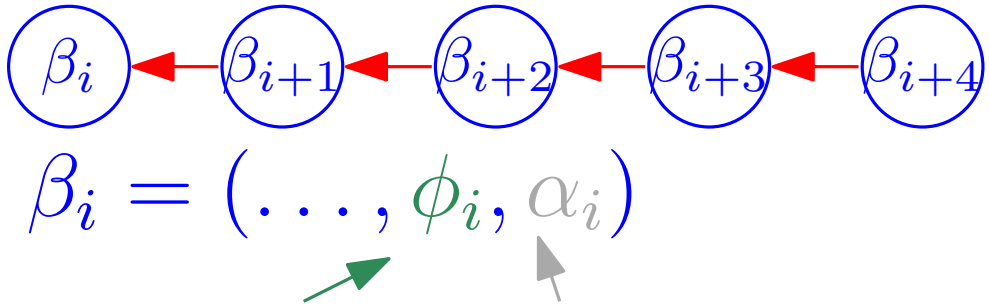
chia




$\sigma_i$ : proof of space on challenge $hash(\tau_{i-1})$

$\tau_i$ : proof of sequential work on challenge $hash(\sigma_{i-1})$
and time parameter $quality(\sigma_{i-1})$

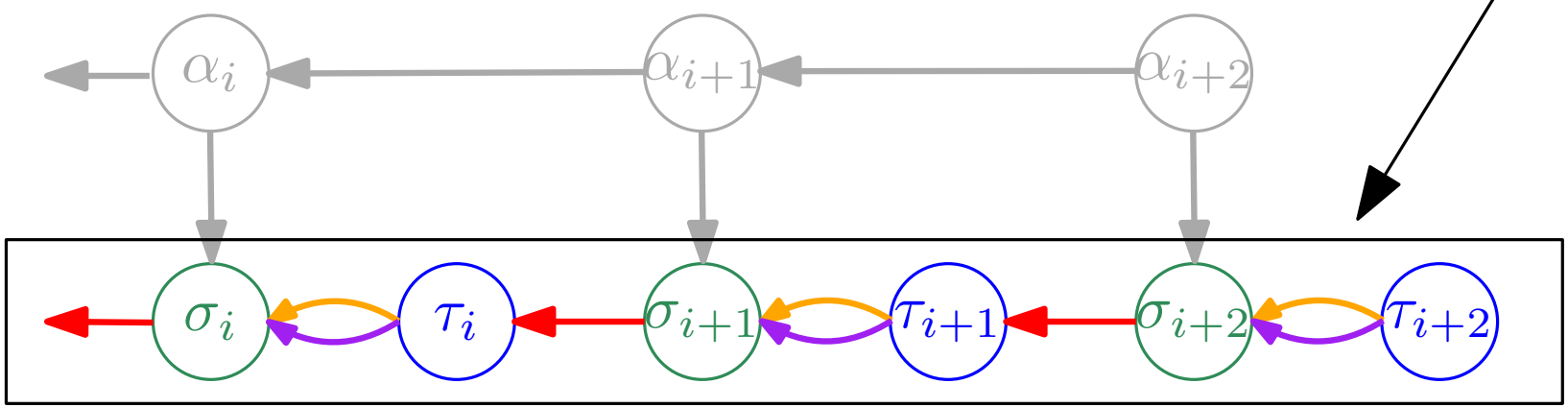
bitcoin




$\phi_i$ : proof of work on challenge $hash(\beta_{i-1})$ transactions

chia

NOTHING TO GRIND HERE!




$\sigma_i$ : proof of space on challenge $hash(\tau_{i-1})$

$\tau_i$ : proof of sequential work on challenge $hash(\sigma_{i-1})$
and time parameter $quality(\sigma_{i-1})$