From Blockchains to Nash

Arka Rai Choudhuri

Pavel Hubáček

Chethan Kamath

Krzysztof Pietrzak

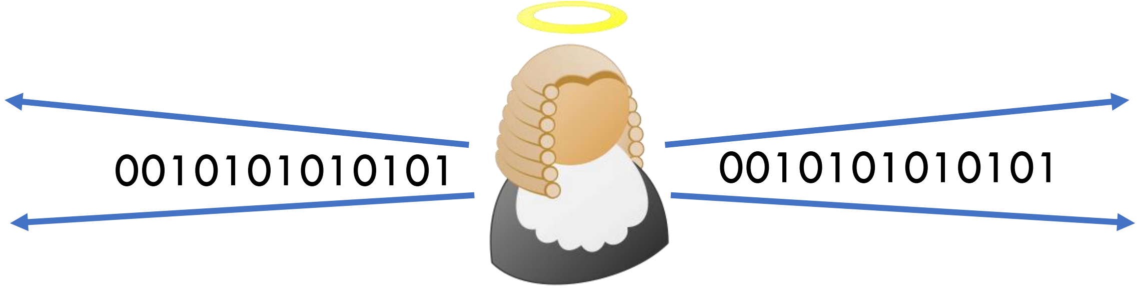
Alon Rosen

Guy Rothblum

Part 1: Blockchains

Verifiable Delay Function(s)

Verifiable Lottery via Randomness Beacon

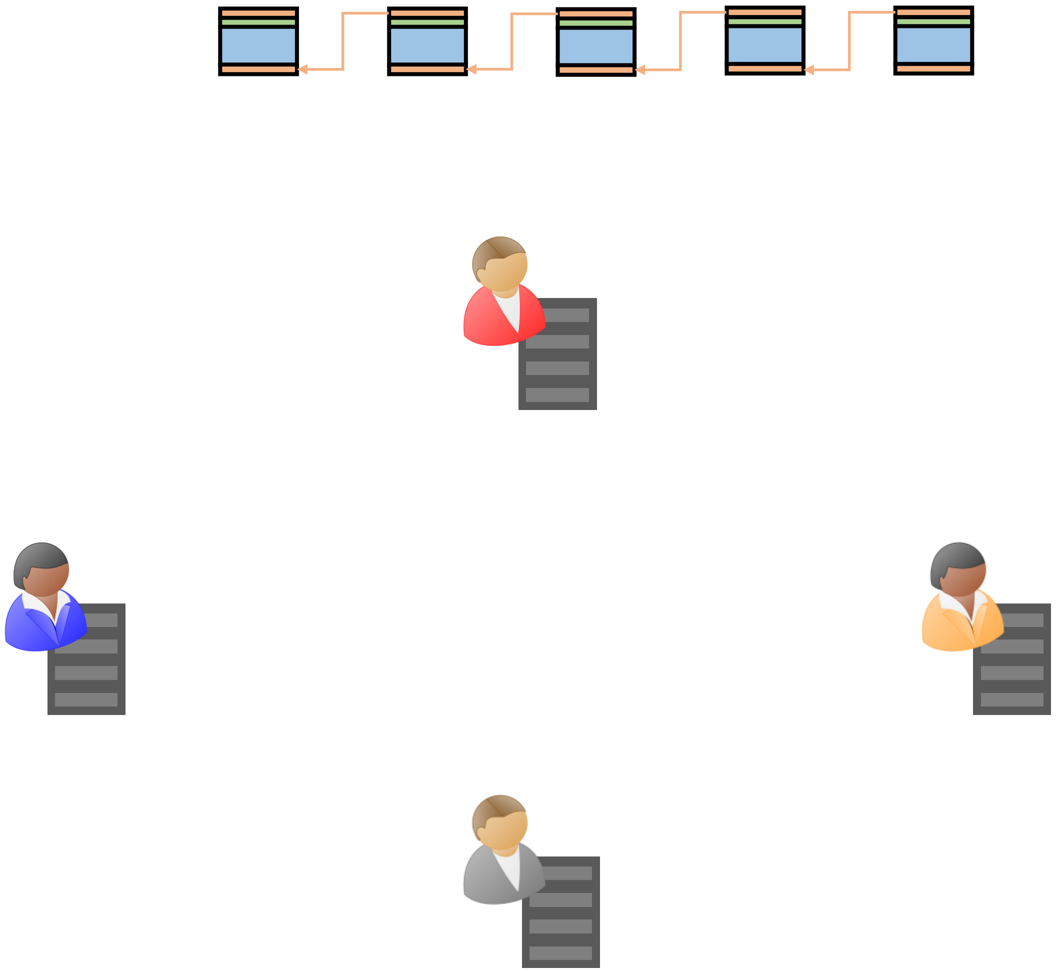


Randomness Beacon [Rabin'83]

Ideal service that periodically publishes random values that cannot be predicted or manipulated.

Randomness Beacon via Blockchains

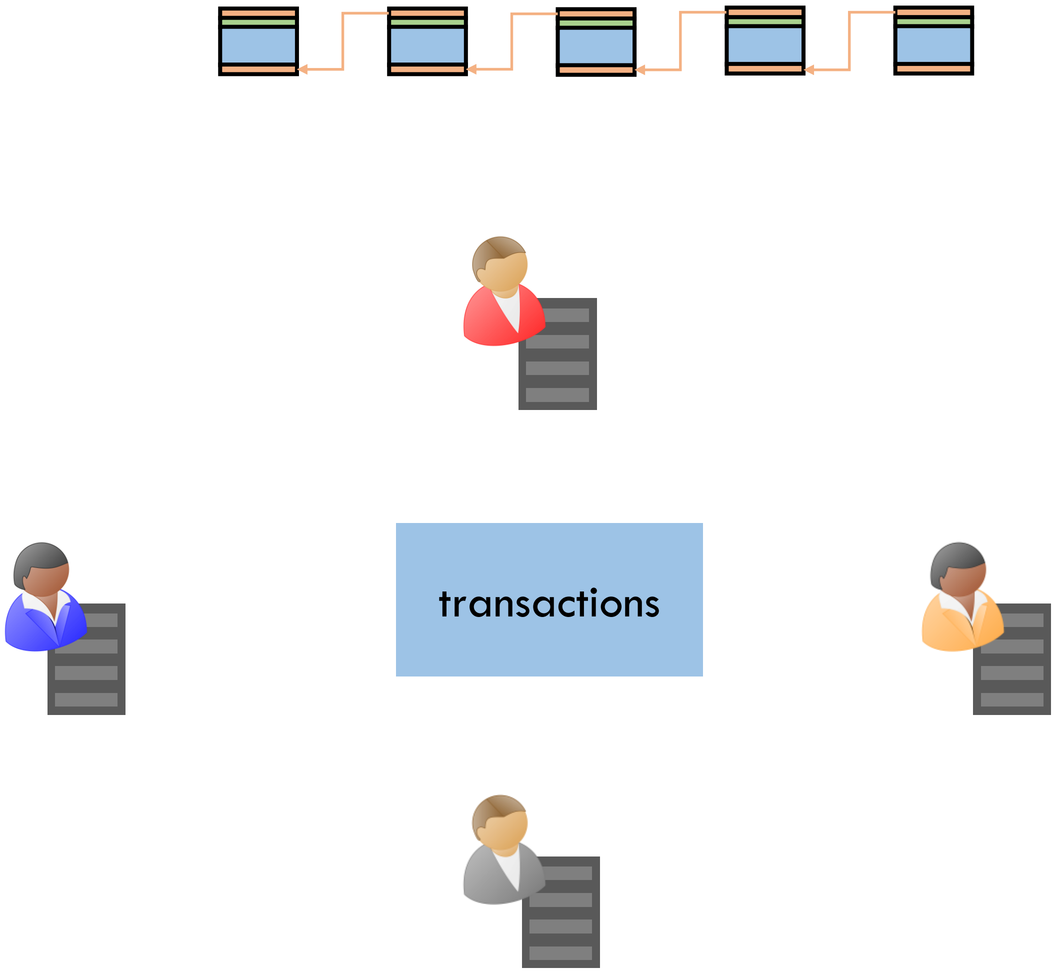
Randomness Beacon via Blockchains



Consider proof of work (PoW) blockchains.

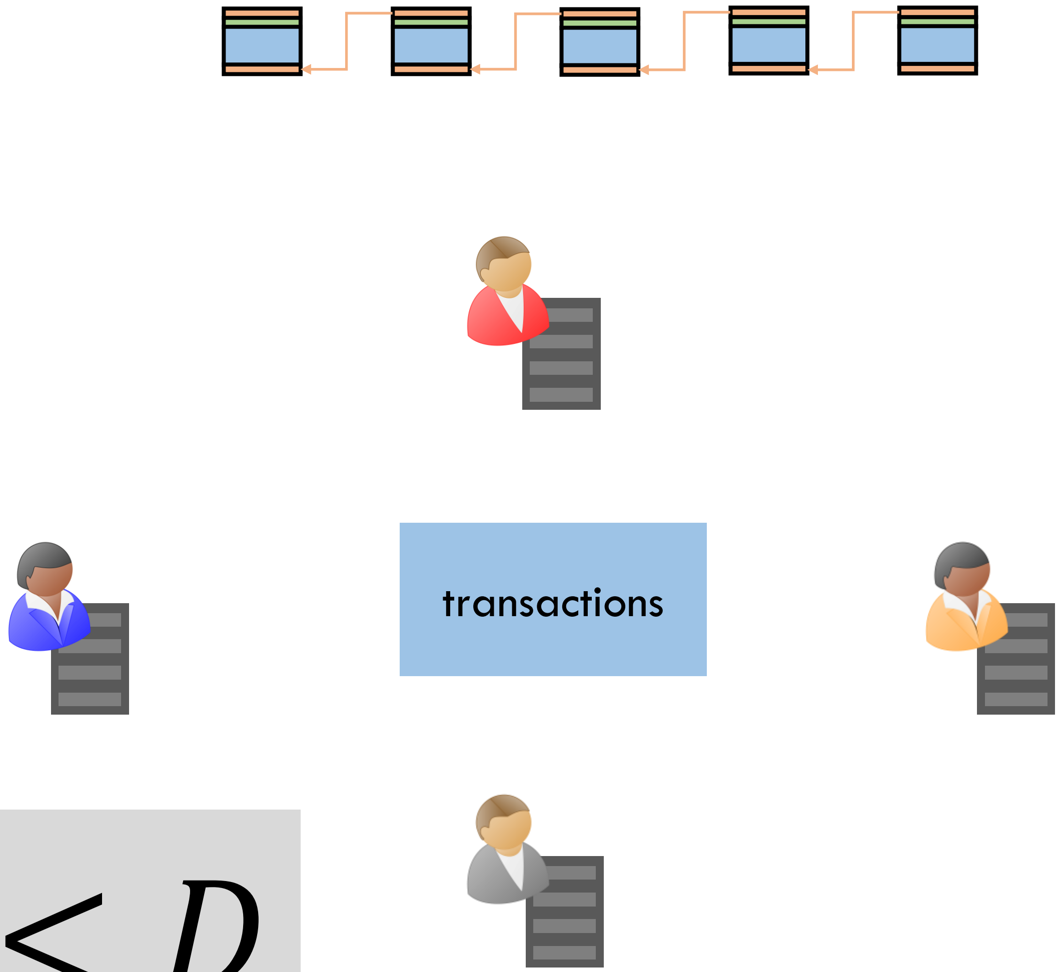
Randomness Beacon via Blockchains

Consider proof of work (PoW) blockchains.



Randomness Beacon via Blockchains

Consider proof of work (PoW) blockchains.

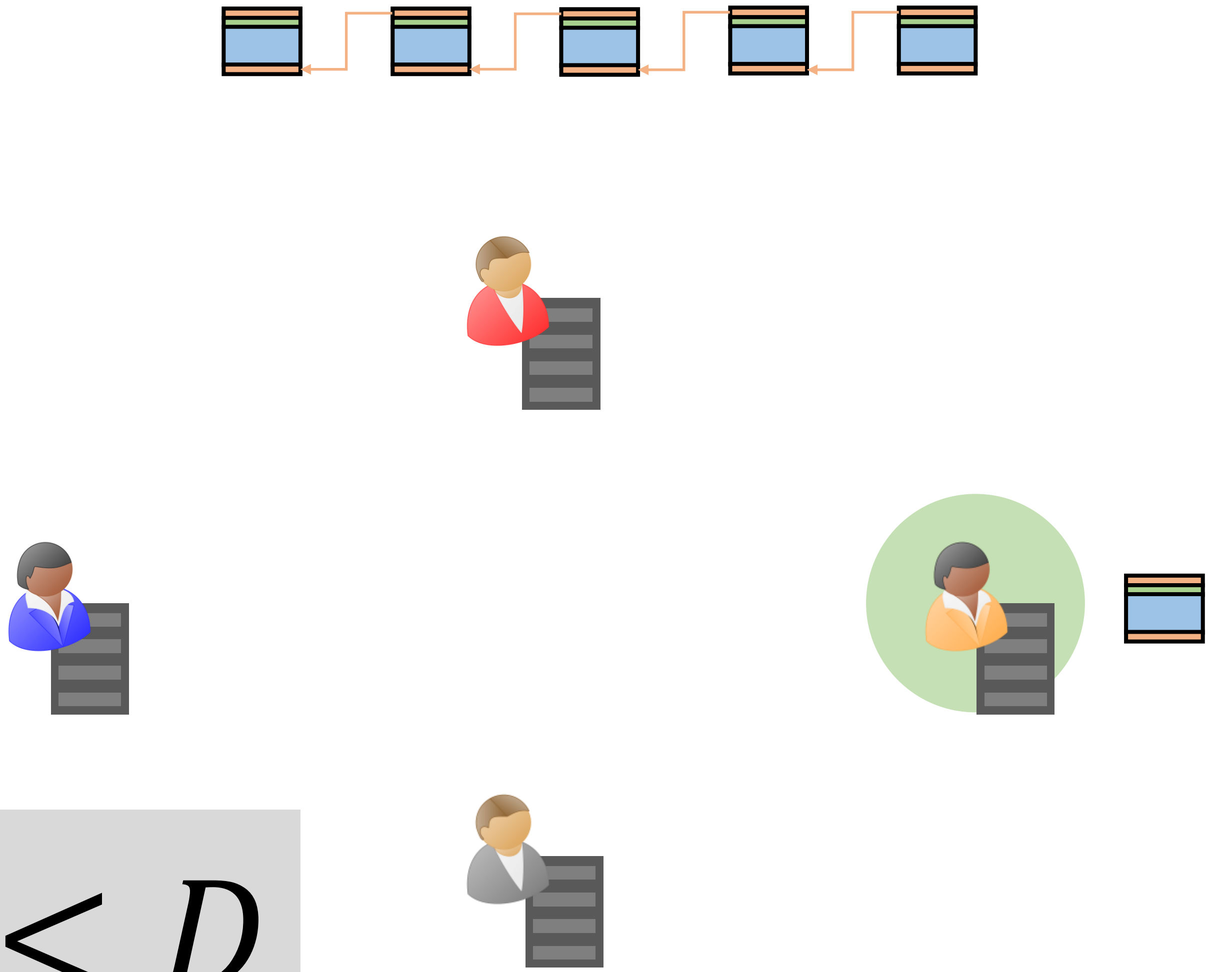


Find

such that $h(\quad) < D$

Randomness Beacon via Blockchains

Consider proof of work (PoW) blockchains.

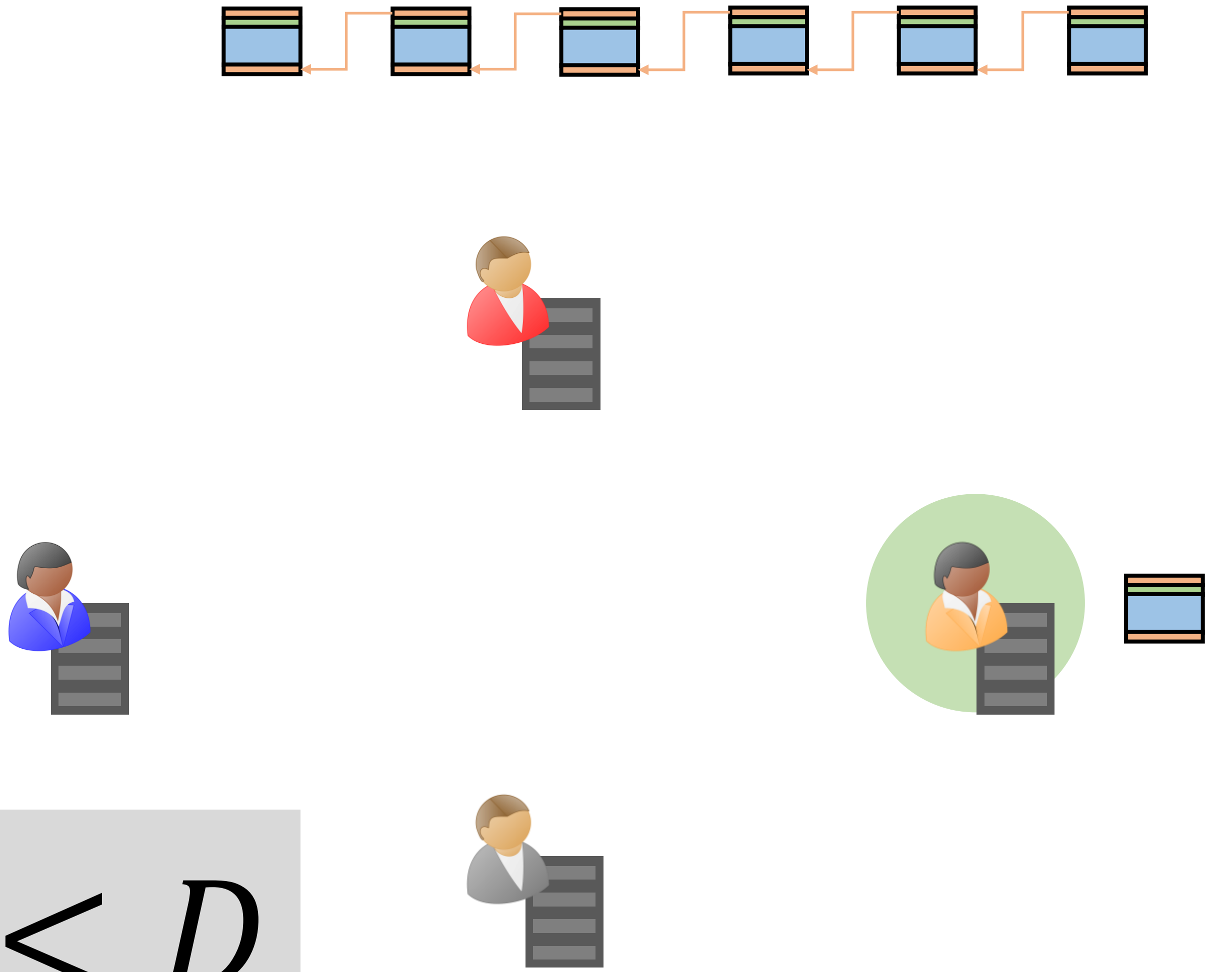


Find

such that $h\left(\quad\right) < D$

Randomness Beacon via Blockchains

Consider proof of work (PoW) blockchains.

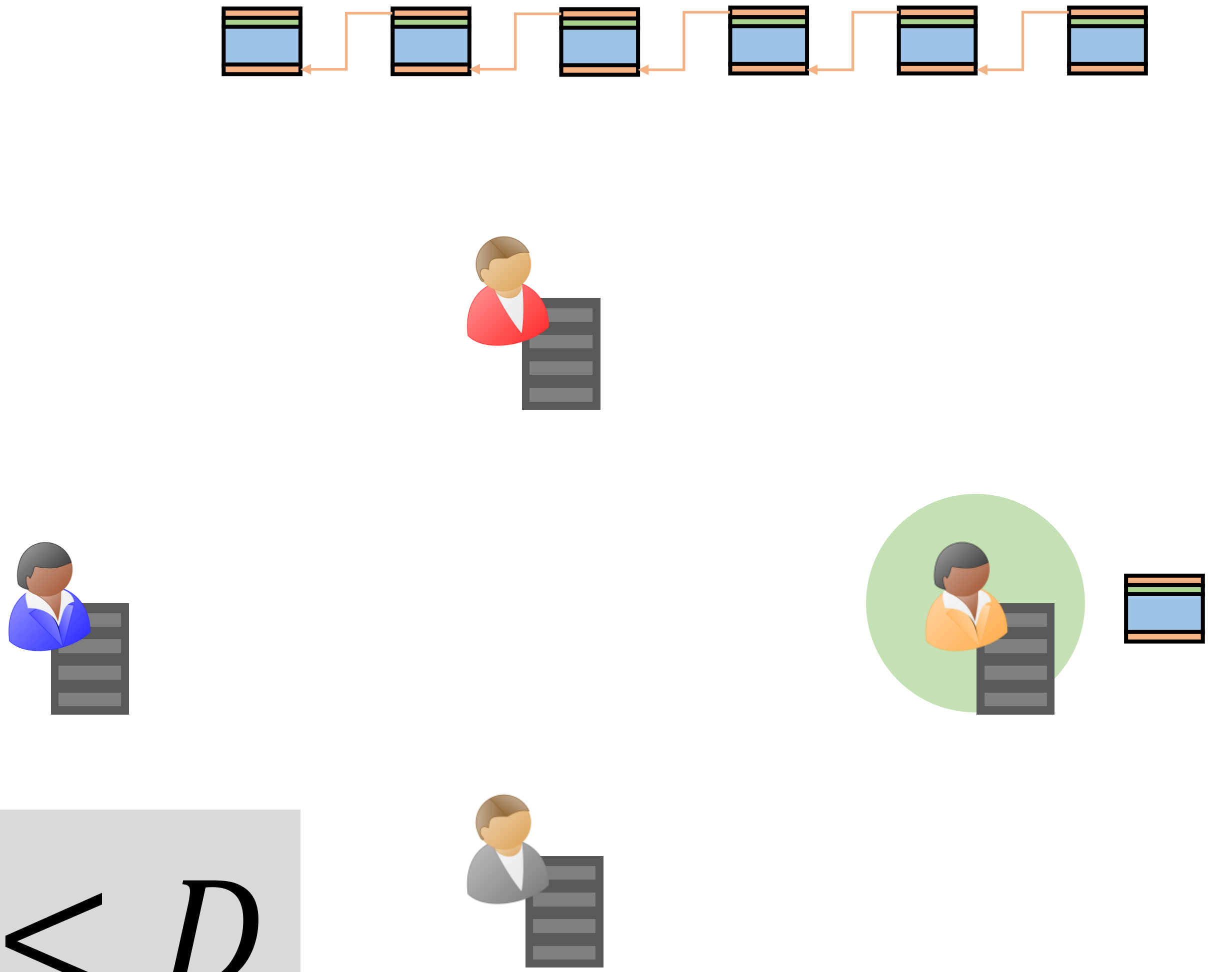


Find

such that $h(\quad) < D$

Randomness Beacon via Blockchains

Consider proof of work (PoW) blockchains.



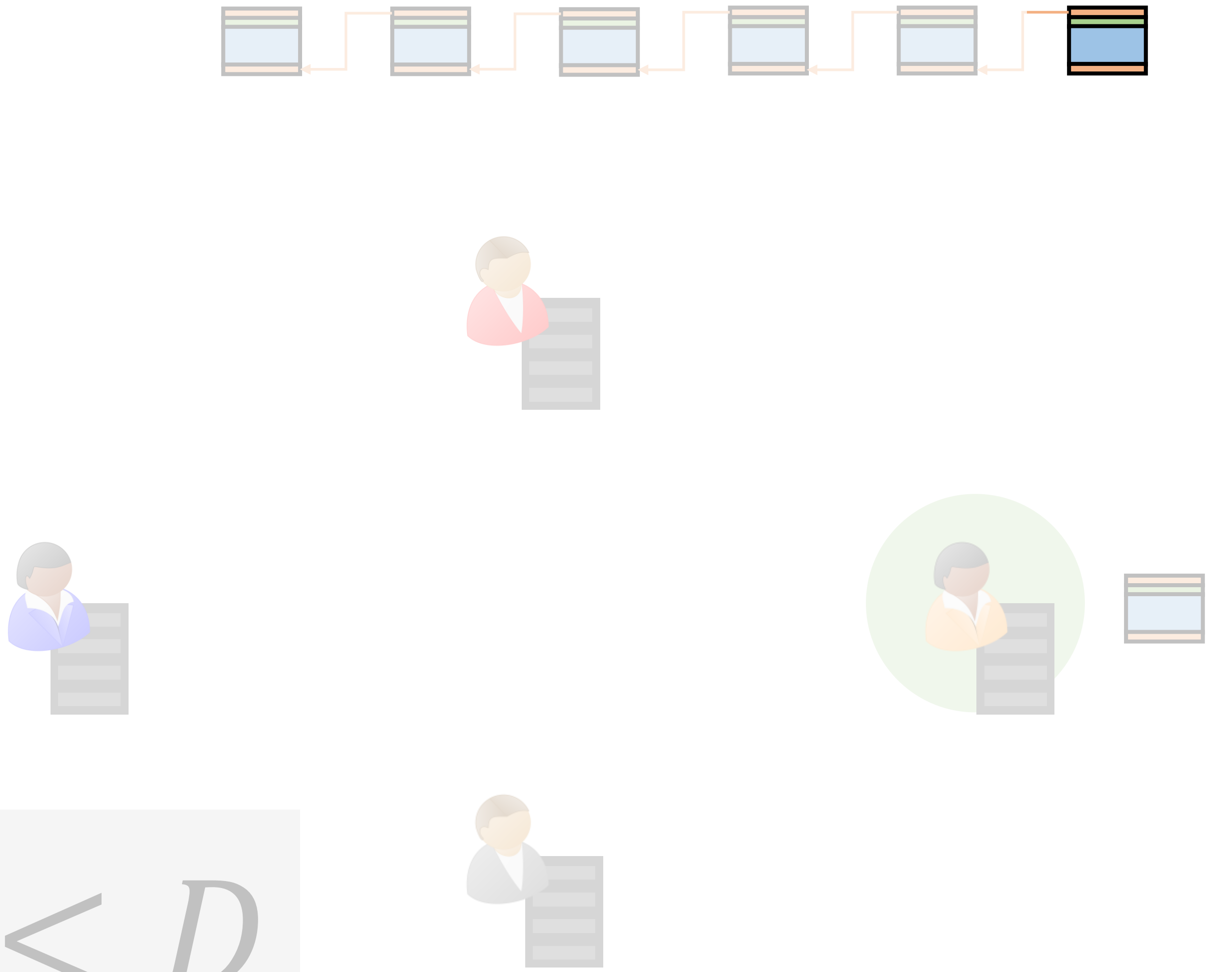
1) Blocks produced periodically.
2) Solution has some amount of unpredictability.

Find

such that $h(\quad) < D$

Randomness Beacon via Blockchains

Consider proof of work (PoW) blockchains.



1) Blocks produced periodically.
2) Solution has some amount of unpredictability.

Find

such that $h(\quad) < D$

Randomness Beacon via Blockchains

Extractor( )

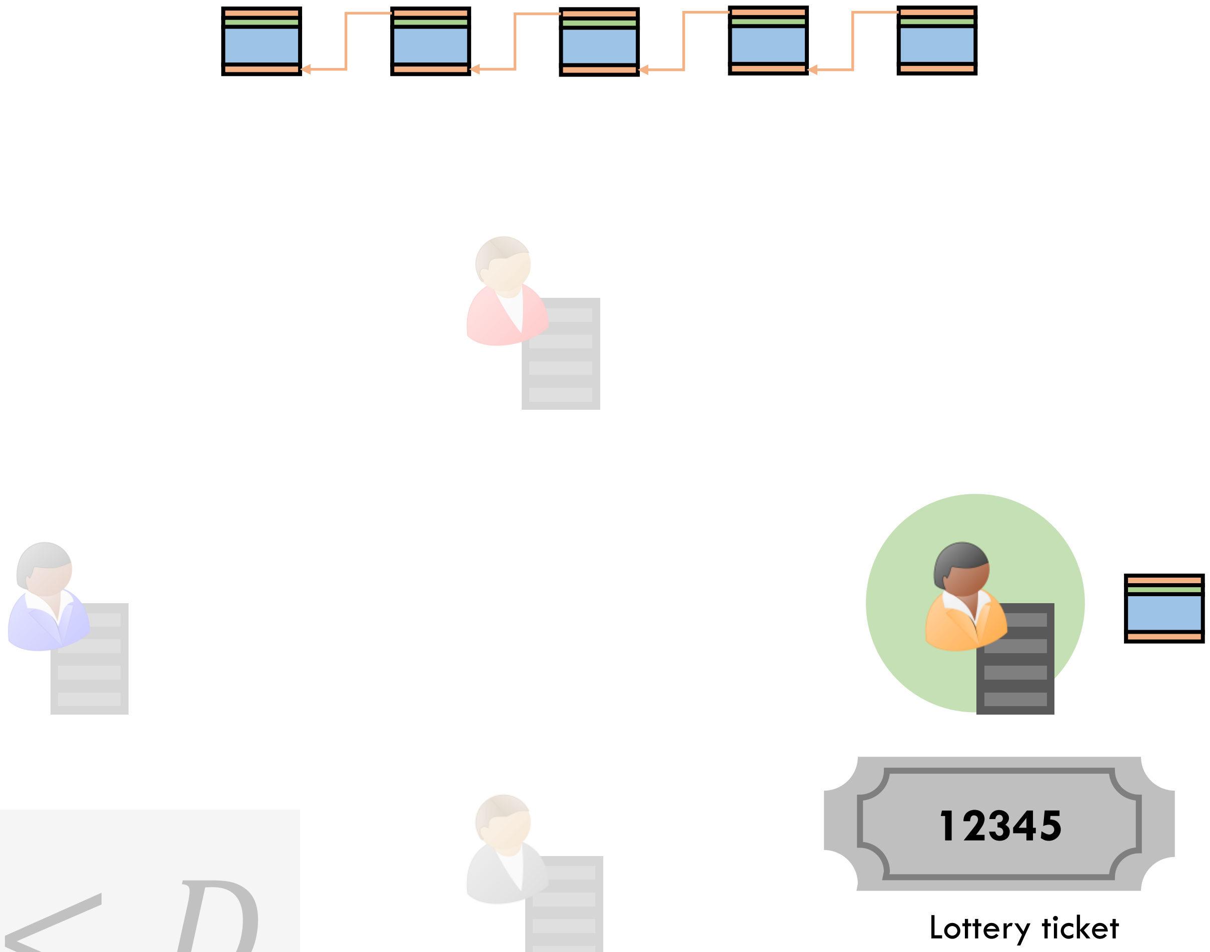
Consider proof of work (PoW) blockchains.

1) Blocks produced periodically.
2) Solution has some amount of unpredictability.

Find such that $h(\;) < D$

Randomness Beacon via Blockchains

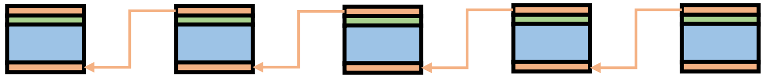
Consider proof of work (PoW) blockchains.



1) Blocks produced periodically.
2) Solution has some amount of unpredictability.

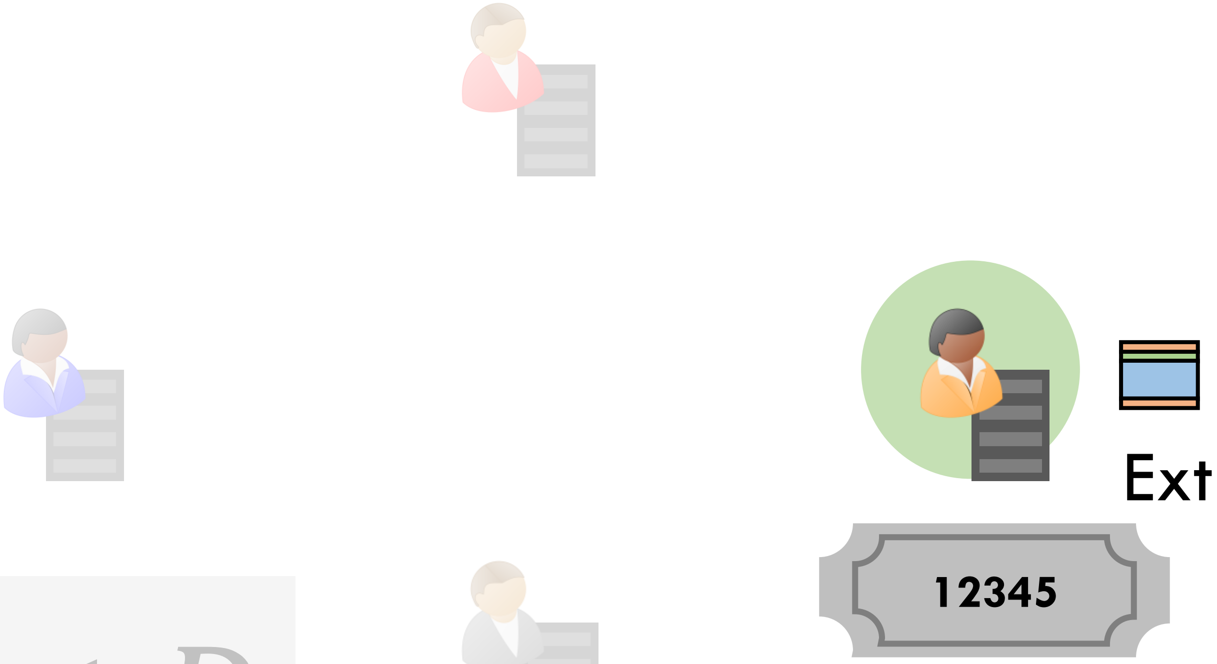
Find
such that $h\left(\quad\right) < D$

Randomness Beacon via Blockchains



Consider proof of work (PoW) blockchains.

1) Blocks produced periodically.
2) Solution has some amount of unpredictability.



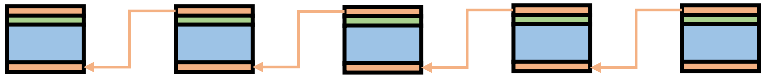
Lottery ticket

Extractor( ) $\stackrel{?}{=}$ 12345

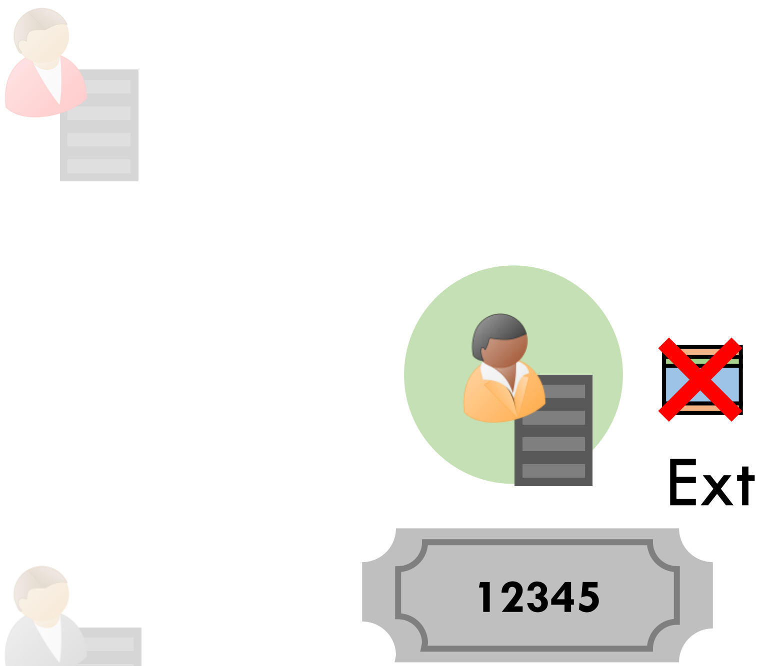
Find

such that $h(\ ) < D$

Randomness Beacon via Blockchains



Consider proof of work (PoW) blockchains.



Lottery ticket

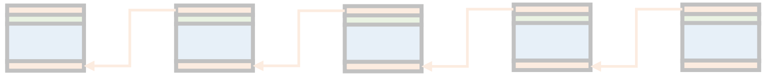
1) Blocks produced periodically.
2) Solution has some amount of unpredictability.

$$\text{Extractor}(\square) \overset{?}{=} 12345$$

Find such that

$$h(\square) < D$$

Randomness Beacon via Blockchains



Consider proof of work (PoW) blockchains.

Randomness Beacon [Rabin'83]

Ideal service that periodically publishes random values that cannot be predicted or manipulated.

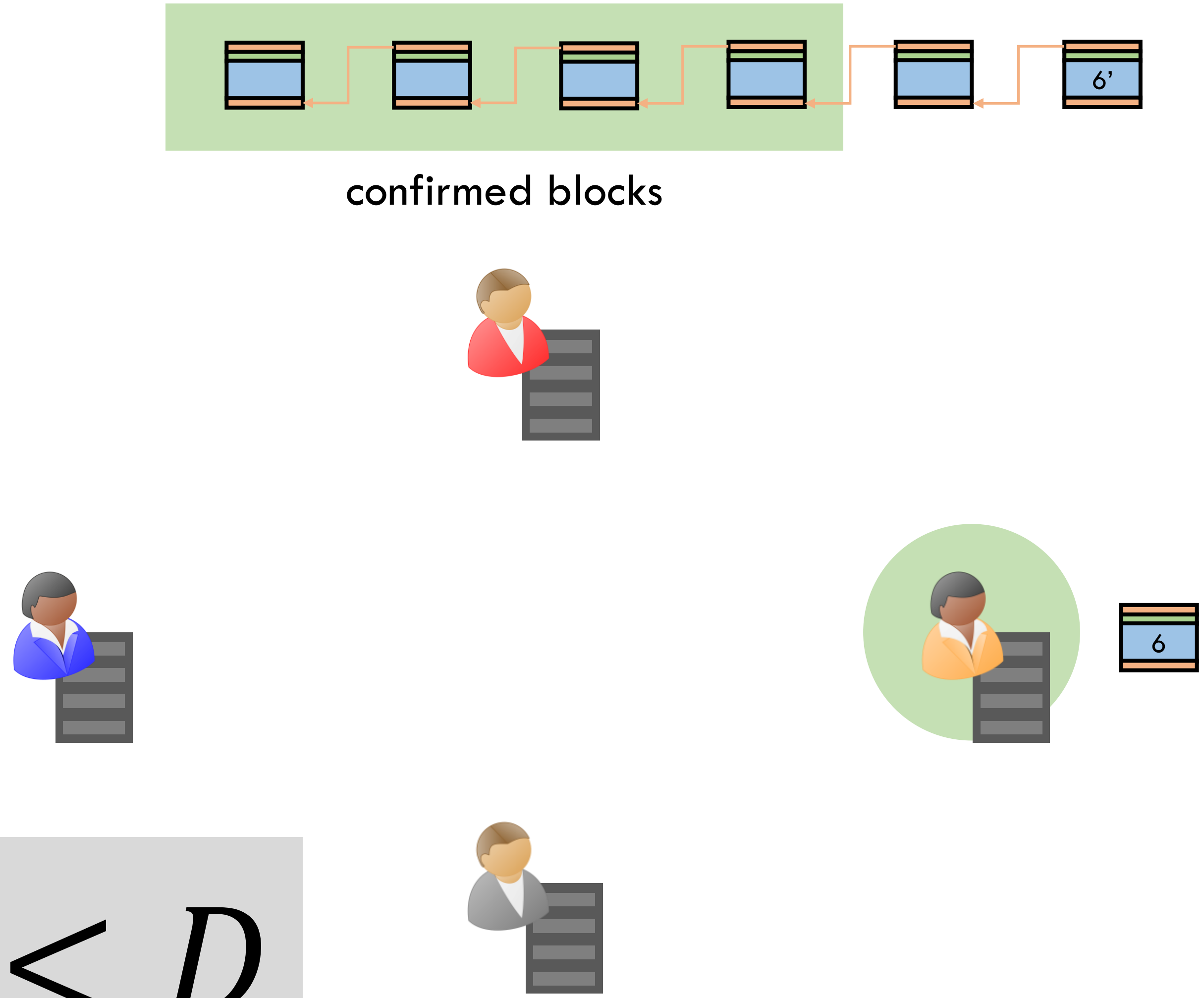
Find such that $h(\cdot) < D$

Extractor( ) $\stackrel{?}{=}$ 12345

12345

Lottery ticket

Randomness Beacon via Blockchains

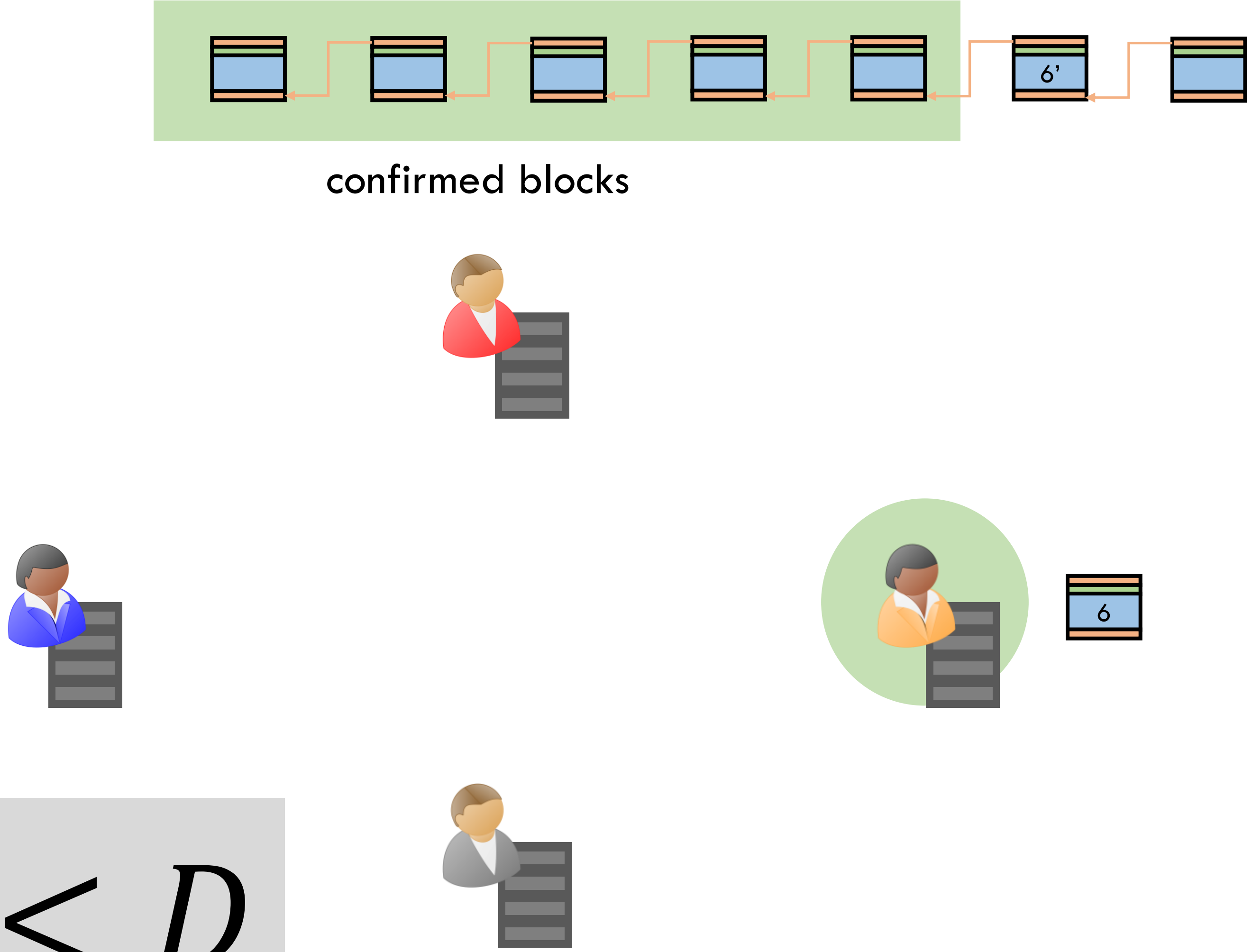


Consider proof of work (PoW) blockchains.

Find  such that

$$h\left(\square\right) < D$$

Randomness Beacon via Blockchains



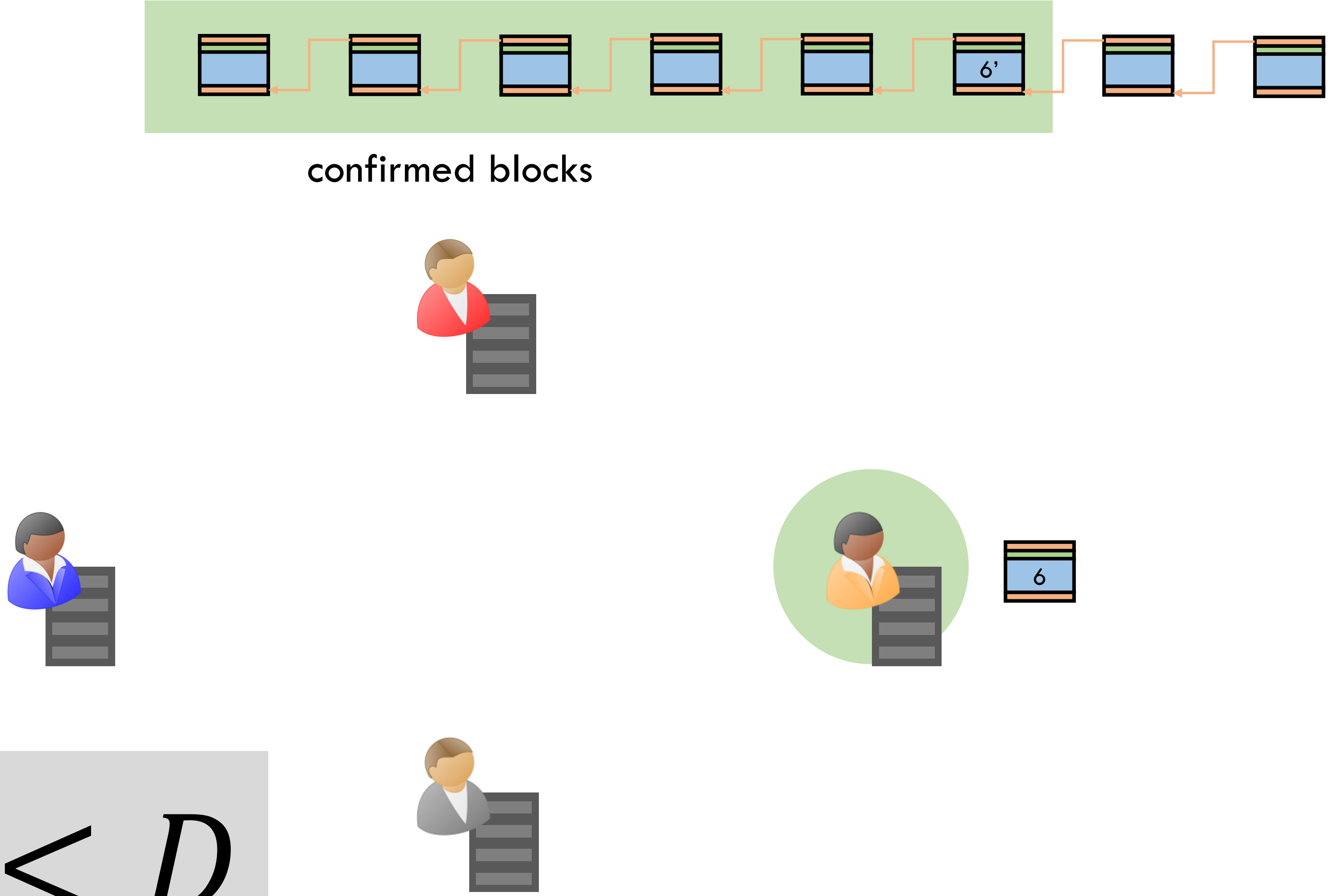
Consider proof of work (PoW) blockchains.

Find

such that $h(\;) < D$

Randomness Beacon via Blockchains

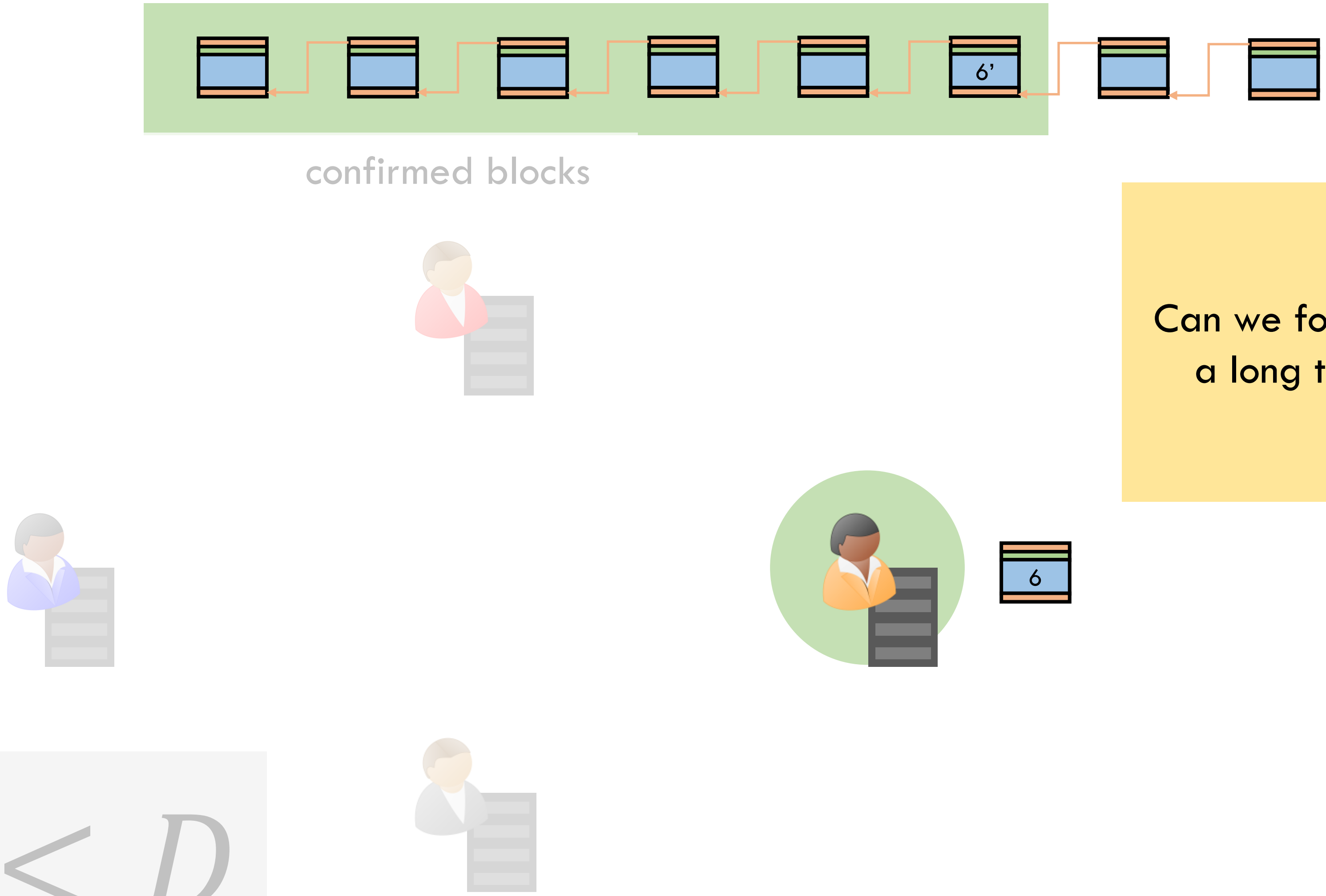
Consider proof of work (PoW) blockchains.



Find

such that $h(\quad) < D$

Randomness Beacon via Blockchains



Consider proof of work (PoW) blockchains.

Can we force Extract to take a long time to compute?

Find [nonce] such that $h(\text{block}) < D$

Cryptographically Slow Functions

$$f(x) = x^{2^T} \bmod N$$

$N = p \cdot q$ for primes p and q

x, T, N $\longrightarrow$  $\underbrace{x \longrightarrow x^2 \longrightarrow x^4 \longrightarrow x^8 \cdots \longrightarrow x^{2^T}}_{T \text{ squarings}} \pmod{N}$

Cryptographically Slow Functions

RSW Assumption [Rivest-Shamir-Wagner'96]

$=p{\cdot}q$ for primes p and q

Input: $N = p \cdot q$, $x \in \mathbb{Z}_N^*$, T

Goal: Find $x^{2^T} \bmod N$

$$\underbrace{x \longrightarrow x^2 \longrightarrow x^4 \longrightarrow x^8 \cdots \longrightarrow x^{2^T}}_{T \text{ squarings}} \pmod{N}$$

Any algorithm that computes $x^{2^T} \bmod N$ requires sequential time not much less than T.

Cryptographically Slow Functions

$$f(x) = x^{2^T} \bmod N$$

$N = p \cdot q$ for primes p and q

x, T, N $\longrightarrow$  $\underbrace{x \longrightarrow x^2 \longrightarrow x^4 \longrightarrow x^8 \cdots \longrightarrow x^{2^T}}_{T \text{ squarings}} \pmod{N}$

x, T, N, p, q $\longrightarrow$  $\phi(N) = (p-1)(q-1)$ $\quad \underbrace{z = 2^T \left(\bmod\ \phi(N)\right), \quad x^z \pmod{N}}_{2 \text{ exponentiations}}$

Cryptographically Slow Functions

$$f(x) = x^{2^T} \bmod N$$

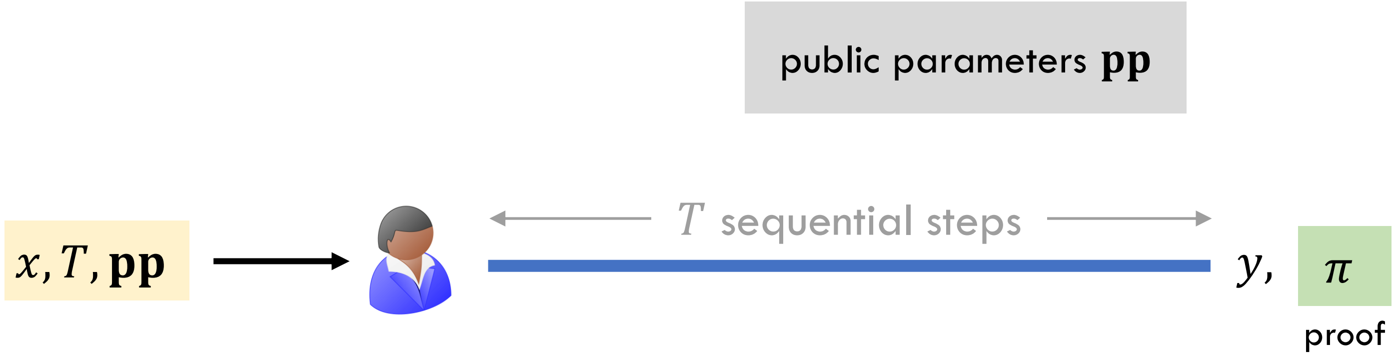
$N{=}p{\cdot}q$ for primes p and q

x, T, N →  $\underbrace{x \longrightarrow x^2 \longrightarrow x^4 \longrightarrow x^8 \cdots \longrightarrow x^{2^T}}_{T \text{ squarings}} \pmod N$

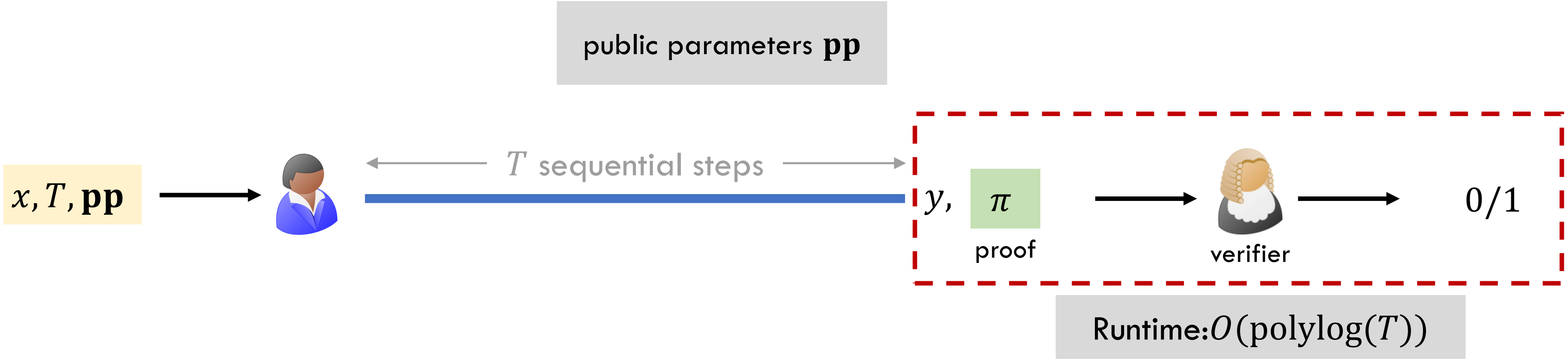
x, T, N, p, q →  $\phi(N) = (p-1)(q-1)$ $\underbrace{z = 2^T \left(\bmod \phi(N)\right),\ \ x^z \pmod N}_{2 \text{ exponentiations}}$

Efficient public verification?

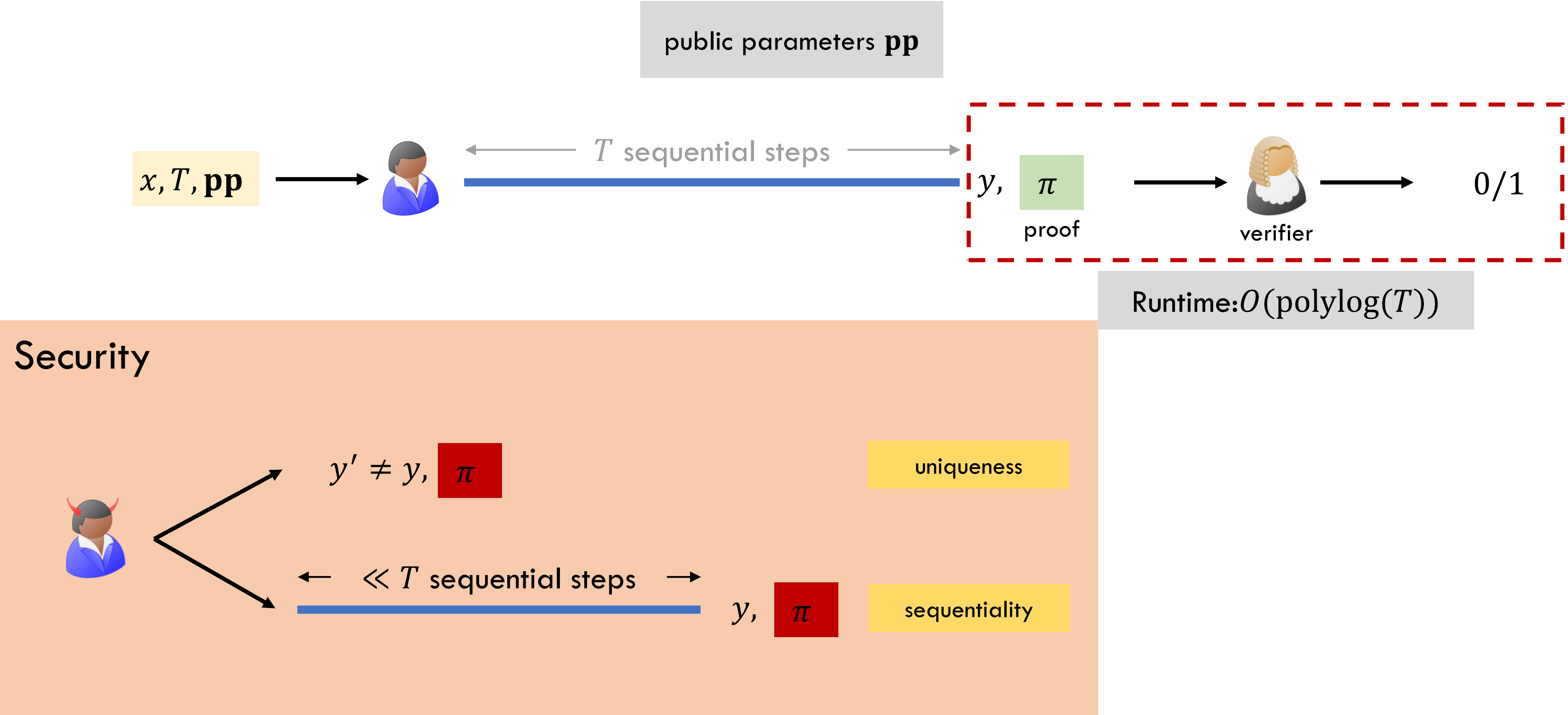
Verifiable Delay Functions [Boneh-Bonneau-Bünz-Fisch'18]



Verifiable Delay Functions [Boneh-Bonneau-Bünz-Fisch'18]



Verifiable Delay Functions [Boneh-Bonneau-Bünz-Fisch’18]



Randomness Beacon via Blockchains

Extractor( )

Consider proof of work (PoW) blockchains.

1) Blocks produced periodically.
2) Solution has some amount of unpredictability.

Find such that $h(\ ) < D$

Randomness Beacon via Blockchains

Extractor(VDF( ))

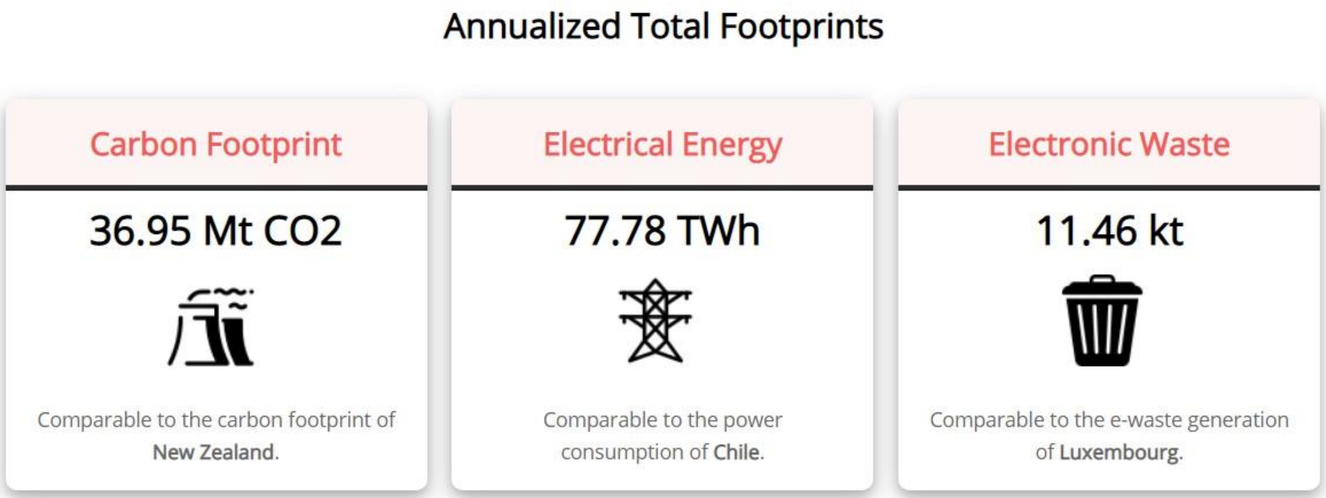
Consider proof of work (PoW) blockchains.

1) Blocks produced periodically.
2) Solution has some amount of unpredictability.

Find such that $h(\;) < D$

VDF Application: Resource Efficient Blockchains

bitcoin



https://digiconomist.net/bitcoin-energy-consumption

VDF Application: Resource Efficient Blockchains

bitcoin

Annualized Total Footprints

Carbon Footprint	Electrical Energy	Electronic Waste
36.95 Mt CO_2	77.78 TWh	11.46 kt
Comparable to the carbon footprint of **New Zealand**.	Comparable to the power consumption of **Chile**.	Comparable to the e-waste generation of **Luxembourg**.

https://digiconomist.net/bitcoin-energy-consumption

chia

https://chia.net/

Blockchain from proofs of space and VDFs.

VDF from Repeated Squaring

[Wesolowski'19,Pietrzak'19]

public parameters N

x, T, N $\longrightarrow$  $\longrightarrow$ $y = x^{2^T} \bmod N$

1

VDF from Repeated Squaring

[Wesolowski'19,Pietrzak'19]

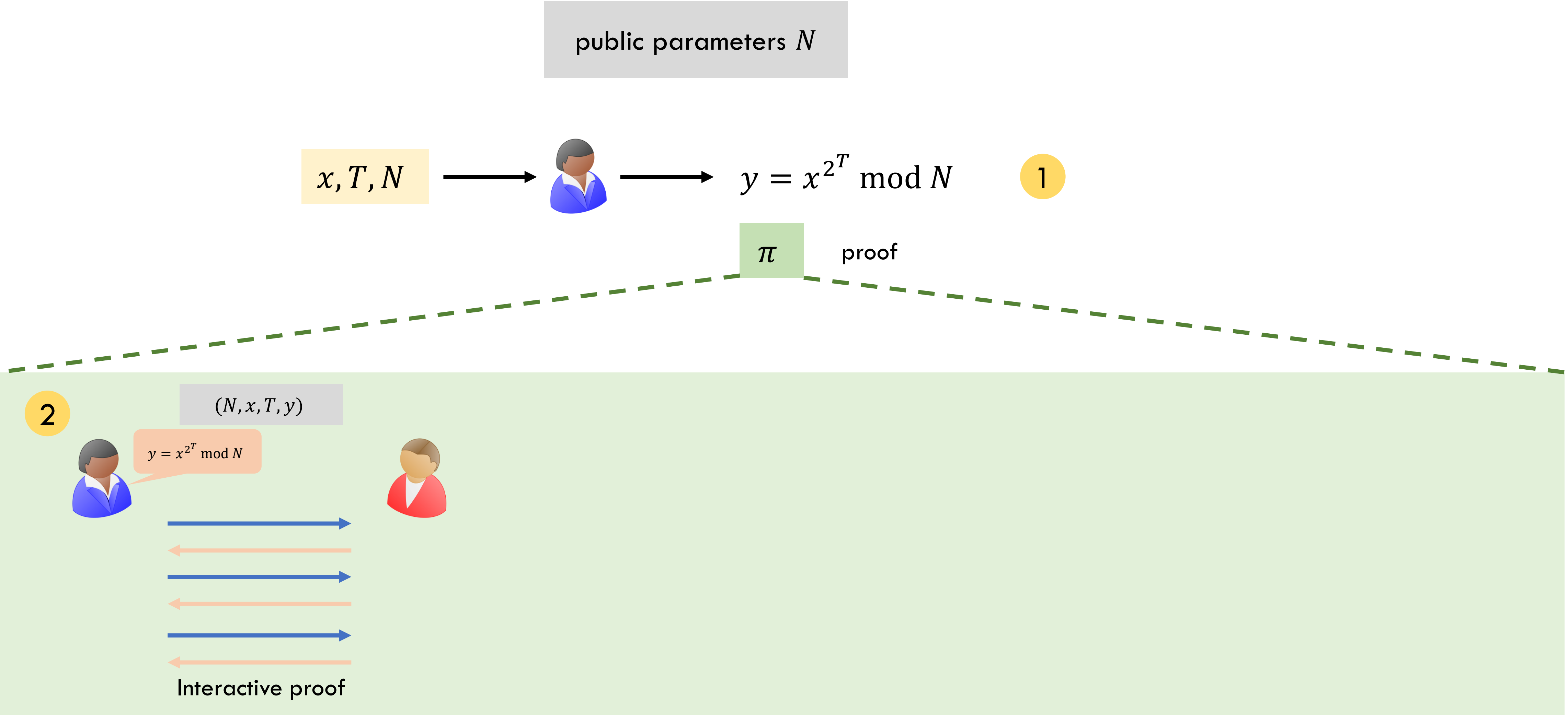
public parameters N

x, T, N →  → $y = x^{2^T} \bmod N$ 1

π proof

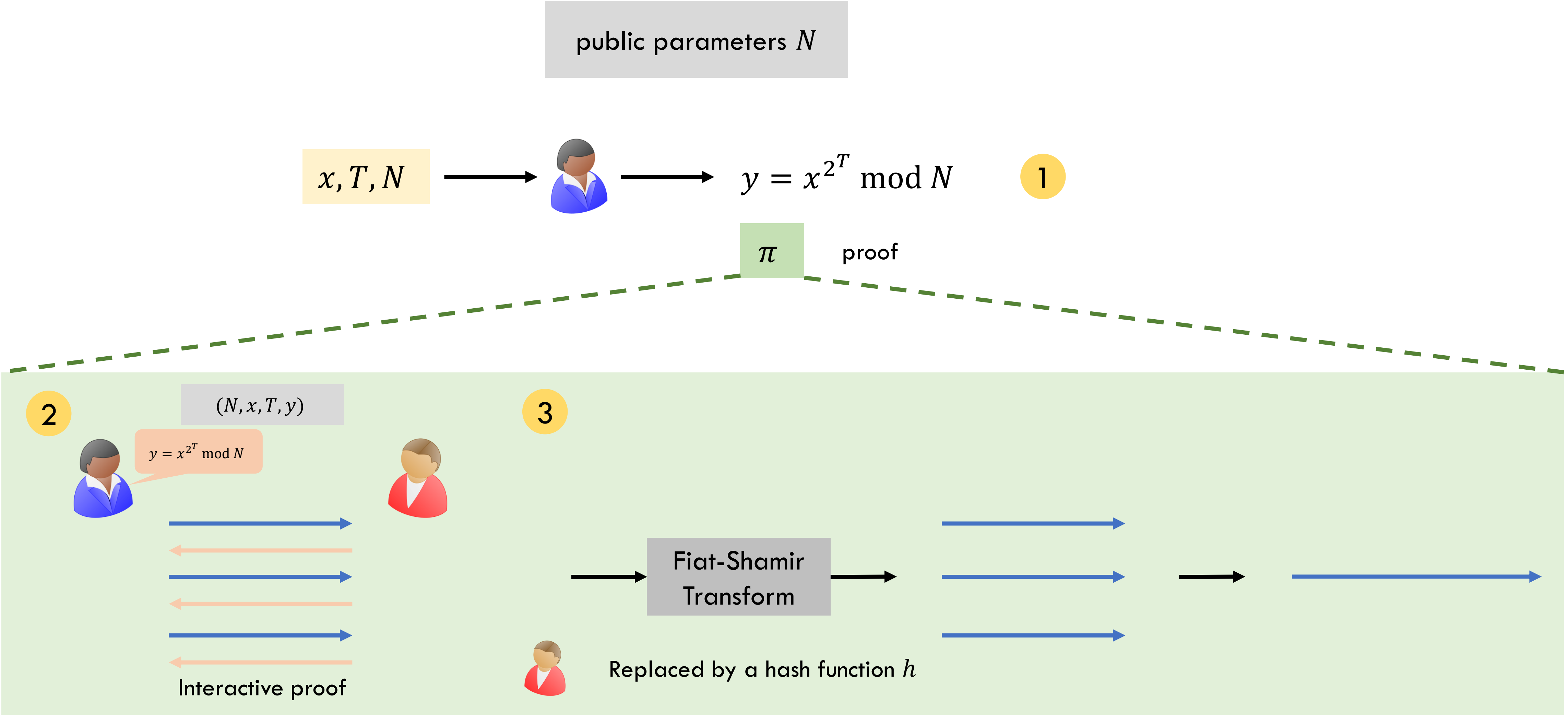
VDF from Repeated Squaring

[Wesolowski'19,Pietrzak'19]



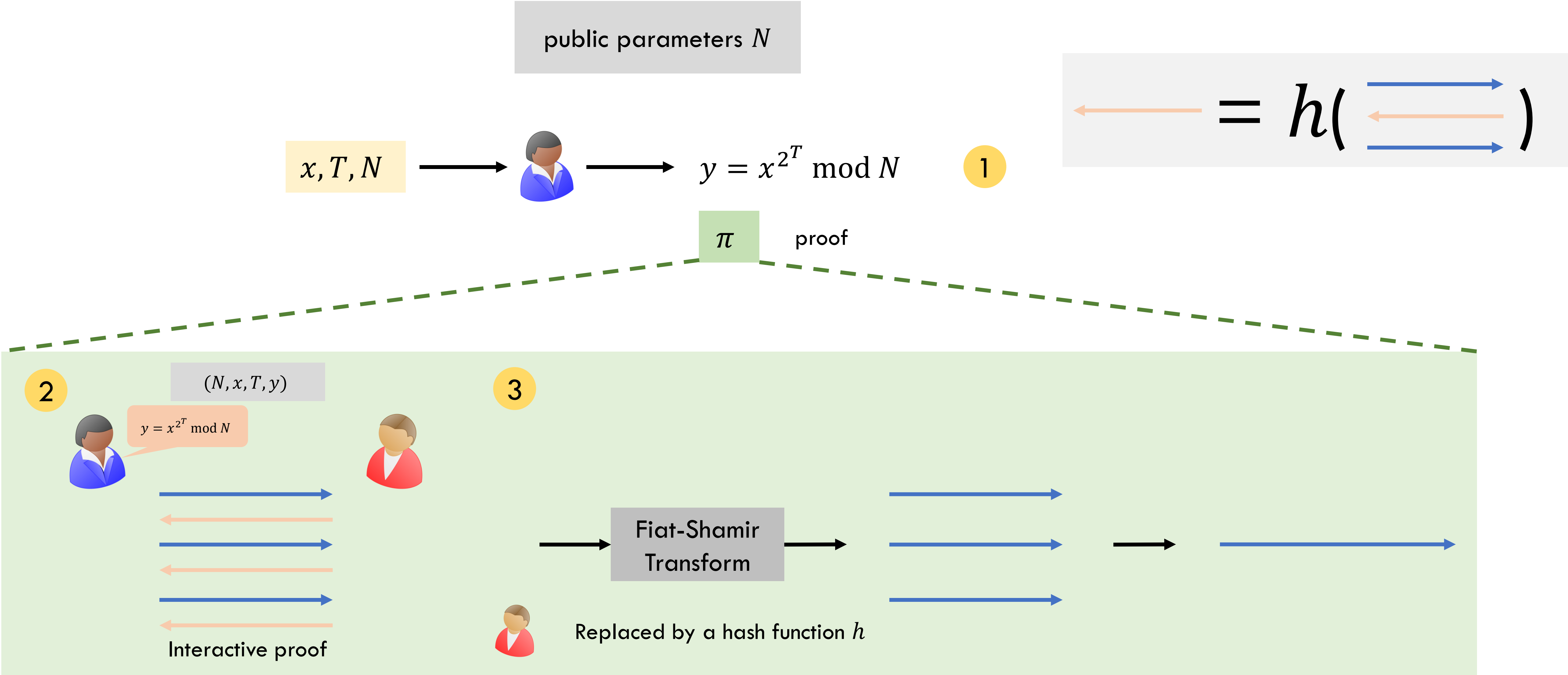
VDF from Repeated Squaring

[Wesolowski'19,Pietrzak'19]



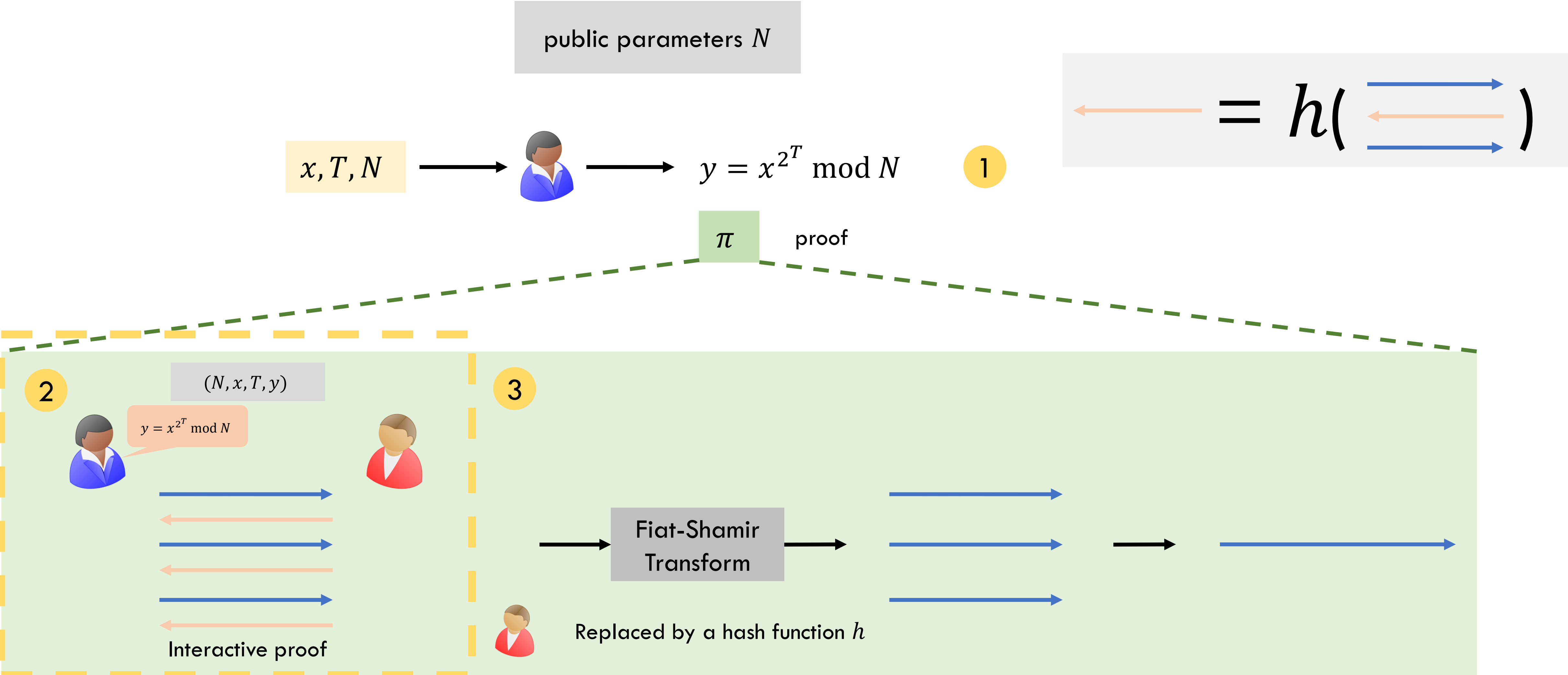
VDF from Repeated Squaring

[Wesolowski'19,Pietrzak'19]



VDF from Repeated Squaring

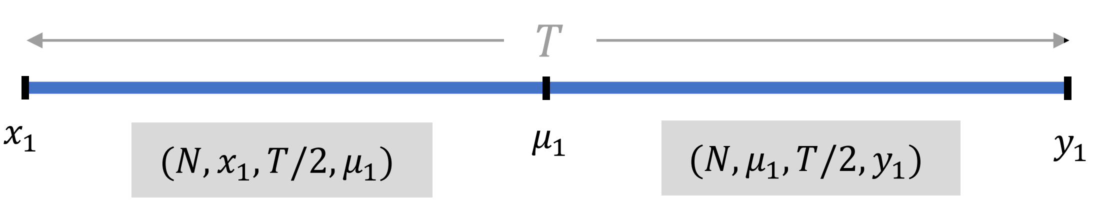
[Wesolowski'19,Pietrzak'19]



Pietrzak Proof System [Pietrzak'19]

$(N, x, T = 2^t, y)$

Pietrzak Proof System [Pietrzak'19]

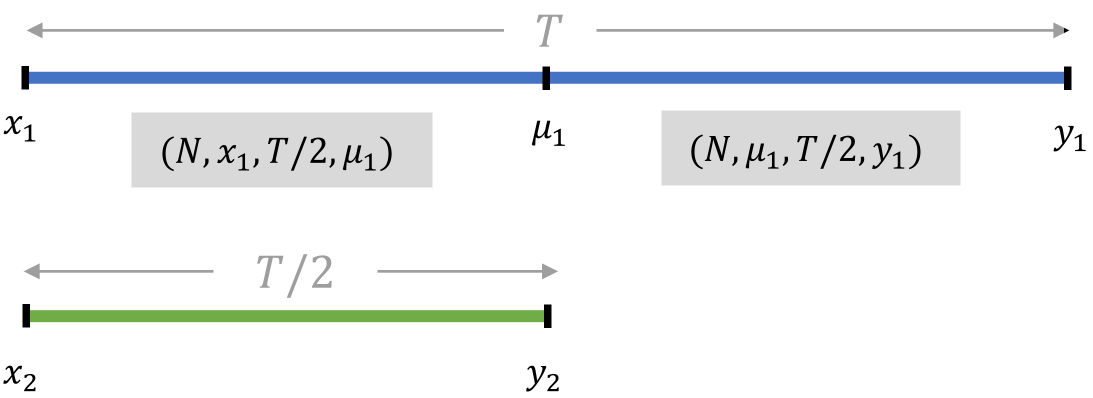


$(N, x, T = 2^t, y)$

$\mu_1 = x_1^{2^{\frac{T}{2}}}$

μ_1

Pietrzak Proof System [Pietrzak'19]



$(N, x, T = 2^t, y)$

$\mu_1 = x_1^{2^{\frac{T}{2}}}$

$\xrightarrow{\mu_1}$

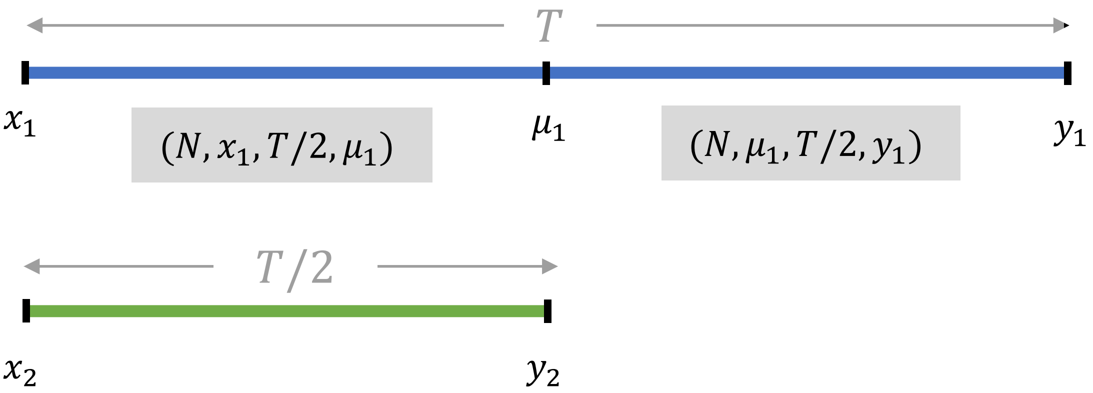
$r_1 \leftarrow \mathbb{Z}_{2^\lambda}$

$\xleftarrow{r_1}$

$x_2 = x_1^{r_1} \cdot \mu_1$
$y_2 = \mu_1^{r_1} \cdot y_1$

$x_2 = x_1^{r_1} \cdot \mu_1$
$y_2 = \mu_1^{r_1} \cdot y_1$

Pietrzak Proof System [Pietrzak'19]



$$x_2 = x_1^{r_1} \cdot \mu_1 = x_1^{r_1 + 2^{\frac{T}{2}}}$$

$$y_2 = \mu_1^{r_1} \cdot y_1 = x_1^{r_1 2^{\frac{T}{2}} + 2^T}$$

$(N, x, T = 2^t, y)$

$\mu_1 = x_1^{2^{\frac{T}{2}}}$ $\xrightarrow{\mu_1}$

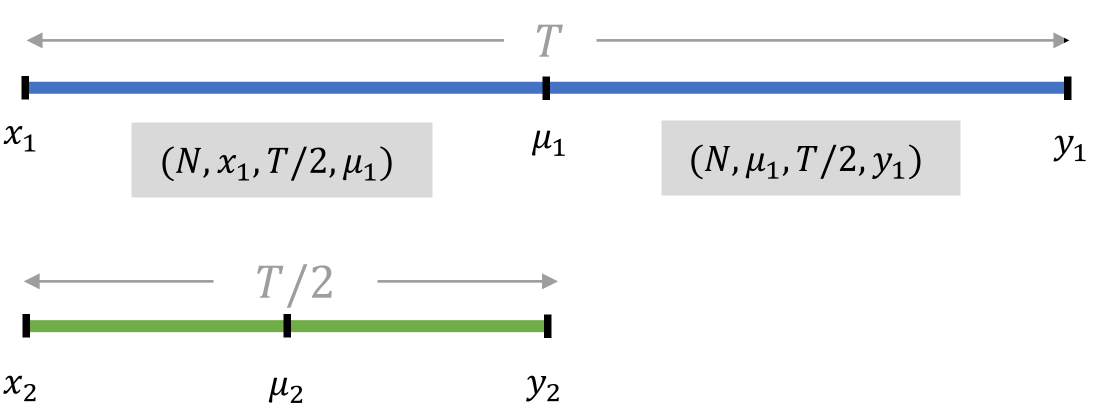
$r_1 \leftarrow \mathbb{Z}_{2^\lambda}$

$\xleftarrow{r_1}$

$x_2 = x_1^{r_1} \cdot \mu_1$
$y_2 = \mu_1^{r_1} \cdot y_1$

$x_2 = x_1^{r_1} \cdot \mu_1$
$y_2 = \mu_1^{r_1} \cdot y_1$

Pietrzak Proof System [Pietrzak'19]



$(N, x, T = 2^t, y)$

$\mu_1 = x_1^{2^{\frac{T}{2}}}$

$\xrightarrow{\mu_1}$

$r_1 \leftarrow \mathbb{Z}_{2^\lambda}$

$\xleftarrow{r_1}$

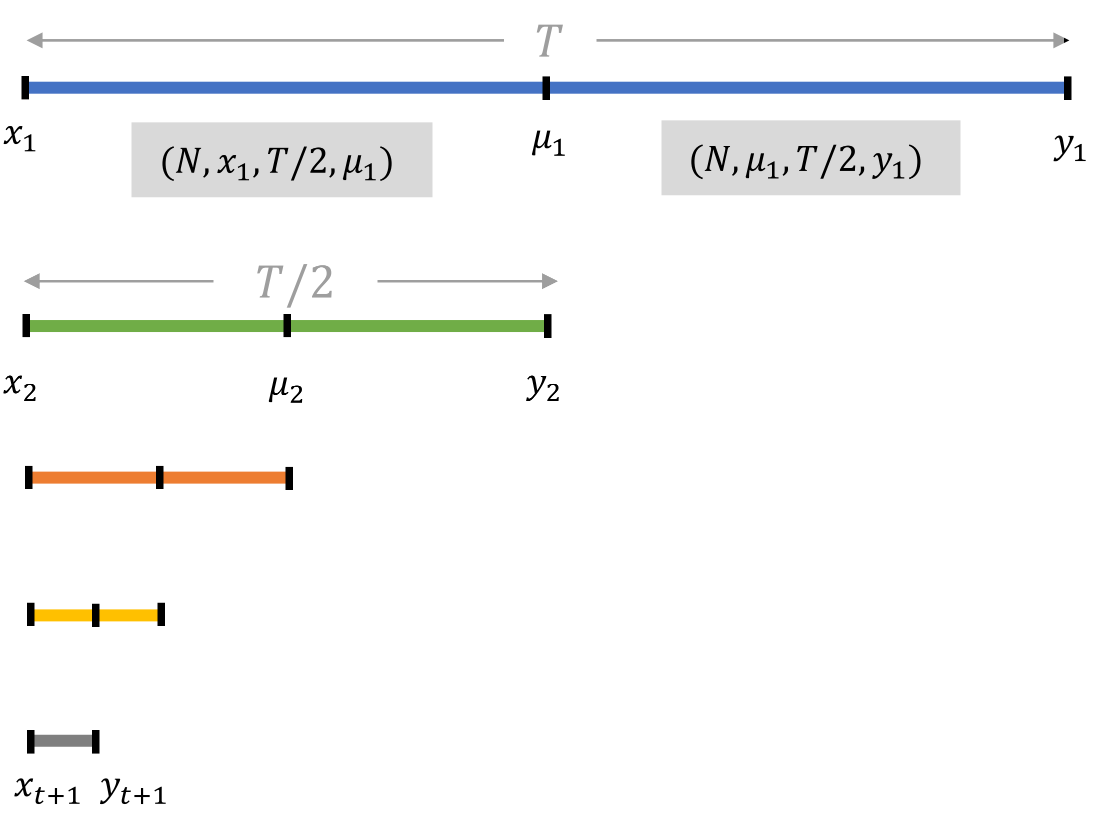
$x_2 = x_1^{r_1} \cdot \mu_1$
$y_2 = \mu_1^{r_1} \cdot y_1$

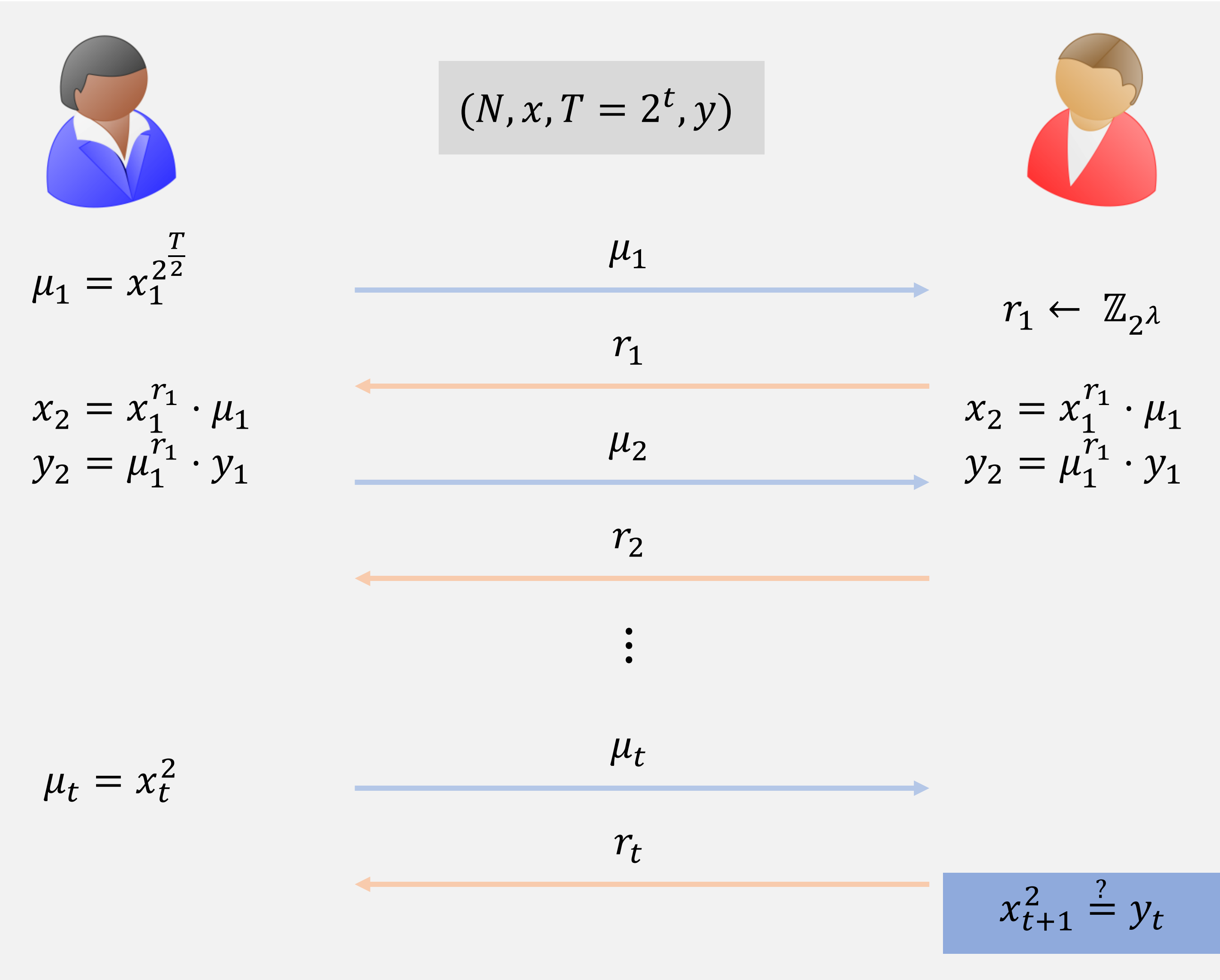
$x_2 = x_1^{r_1} \cdot \mu_1$
$y_2 = \mu_1^{r_1} \cdot y_1$

$\xrightarrow{\mu_2}$

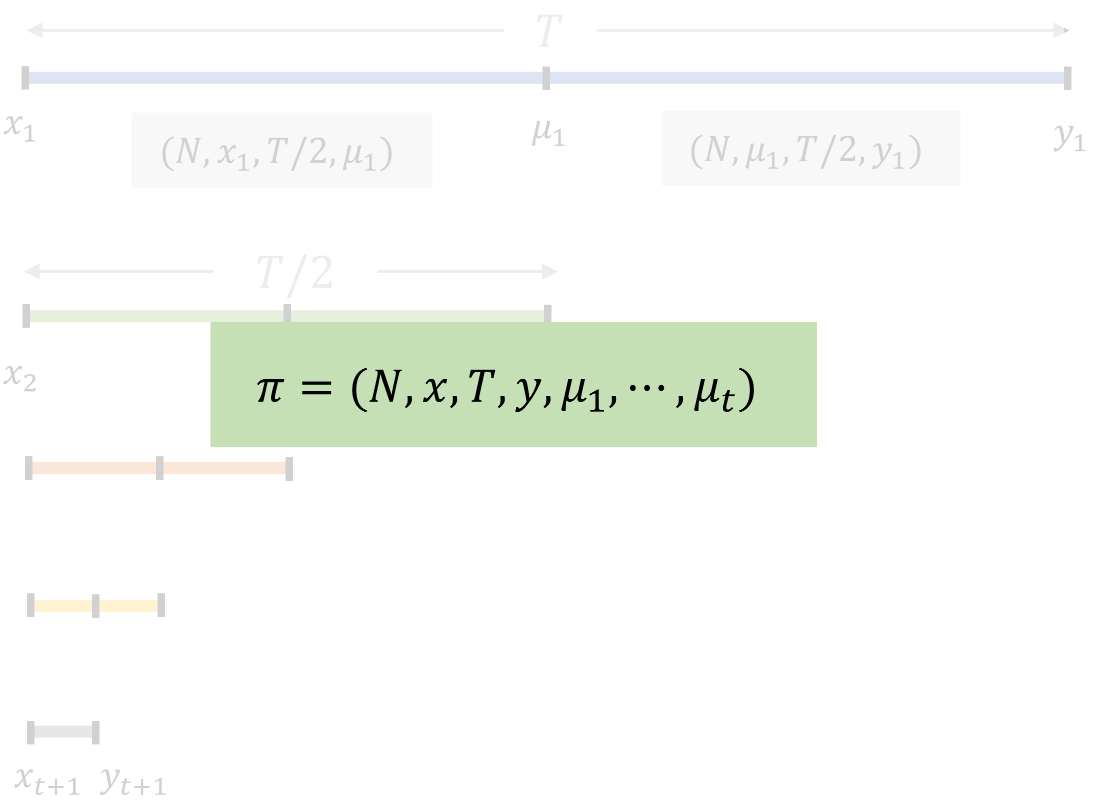
$\xleftarrow{r_2}$

Pietrzak Proof System [Pietrzak'19]





Pietrzak Proof System [Pietrzak'19]



$$\pi = (N, x, T, y, \mu_1, \cdots, \mu_t)$$

$(N, x, T = 2^t, y)$

$\mu_1 = x_1^{2^{\frac{T}{2}}}$ $\xrightarrow{\mu_1}$ $r_1 \leftarrow \mathbb{Z}_{2^\lambda}$

$\xleftarrow{r_1}$

$x_2 = x_1^{r_1} \cdot \mu_1$
$y_2 = \mu_1^{r_1} \cdot y_1$ $\xrightarrow{\mu_2}$ $x_2 = x_1^{r_1} \cdot \mu_1$
$y_2 = \mu_1^{r_1} \cdot y_1$

$\xleftarrow{r_2}$

$\vdots$

$\mu_t = x_t^2$ $\xrightarrow{\mu_t}$

$\xleftarrow{r_t}$

$x_{t+1}^2 \stackrel{?}{=} y_t$

Costs of the Pietrzak VDF

Time to compute y + proof

$O(T)$

Size of the proof

$\pi = (N, x, T, y, \mu_1, \cdots, \mu_t)$

$O(\text{polylog}(T))$

Time to verify Proof

$O(\log(T))$ exponentiations - $O(\text{polylog}(T))$

Continuous VDF (cVDF)

[Ephraim-Freitag-Komargodski-Pass’20]

Continuous VDF (cVDF)

[Ephraim-Freitag-Komargodski-Pass'20]

$5 Million prize to compute y

Continuous VDF (cVDF)

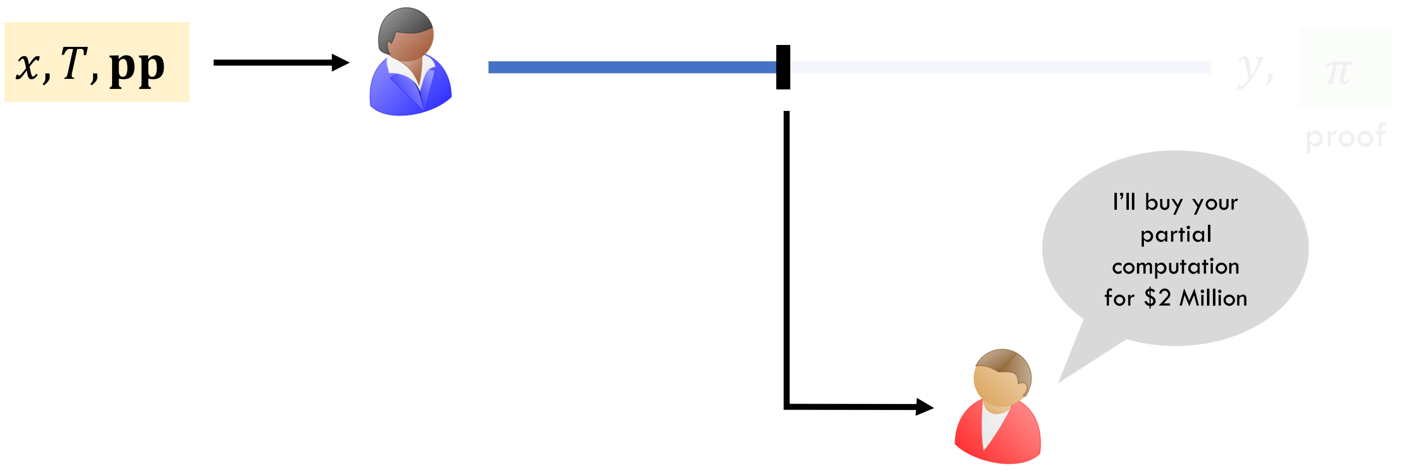
[Ephraim-Freitag-Komargodski-Pass’20]

$5 Million prize to compute y

Continuous VDF (cVDF)

[Ephraim-Freitag-Komargodski-Pass'20]

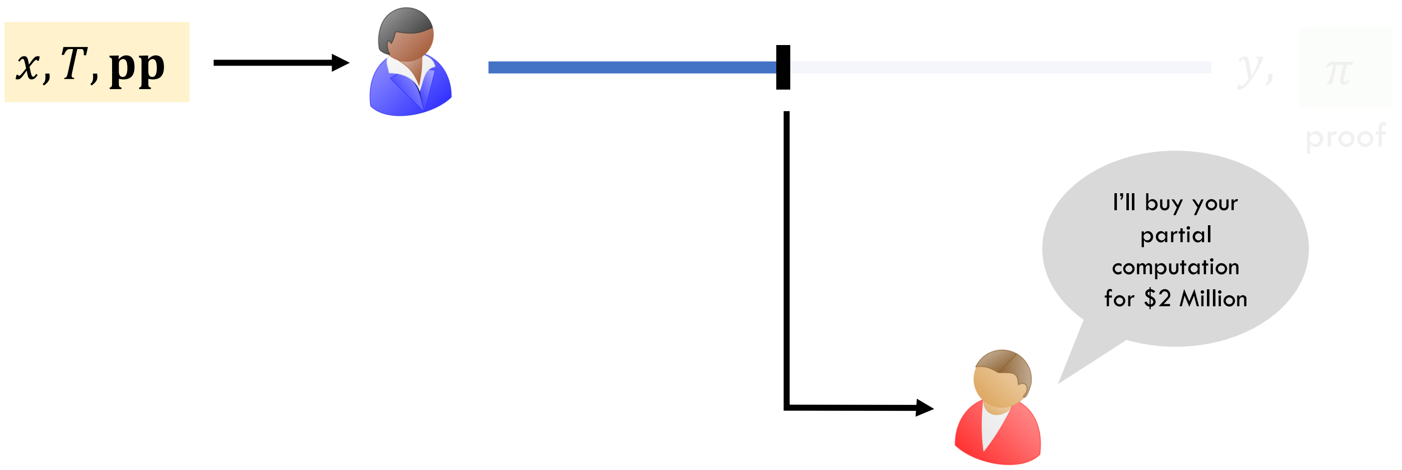
$5 Million prize to compute y



Continuous VDF (cVDF)

[Ephraim-Freitag-Komargodski-Pass'20]

$5 Million prize to compute y



How does Alice transfer a state that Bob is able to verify?

Continuously Verifiable Squaring

$x \xrightarrow{1} x^{2^1}$ $x^{2^1} \xrightarrow{1} x^{2^2}$ $x^{2^2} \xrightarrow{1} x^{2^3}$ $x^{2^2} \xrightarrow{1} x^{2^4}$ $x^{2^4} \xrightarrow{1} x^{2^5}$ $x^{2^5} \xrightarrow{1} x^{2^6}$ $x^{2^6} \xrightarrow{1} x^{2^7}$ $x^{2^7} \xrightarrow{1} x^{2^8}$

Continuously Verifiable Squaring

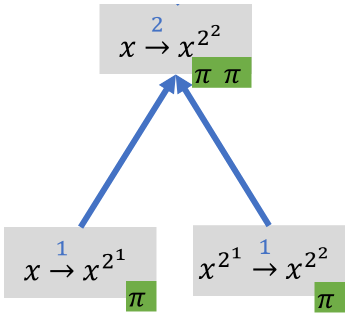
$x \xrightarrow{1} x^{2^1}$ π $\quad$ $x^{2^1} \xrightarrow{1} x^{2^2}$ π $\quad$ $x^{2^2} \xrightarrow{1} x^{2^3}$ π $\quad$ $x^{2^2} \xrightarrow{1} x^{2^4}$ π $\quad$ $x^{2^4} \xrightarrow{1} x^{2^5}$ π $\quad$ $x^{2^5} \xrightarrow{1} x^{2^6}$ π $\quad$ $x^{2^6} \xrightarrow{1} x^{2^7}$ π $\quad$ $x^{2^7} \xrightarrow{1} x^{2^8}$ π

Continuously Verifiable Squaring

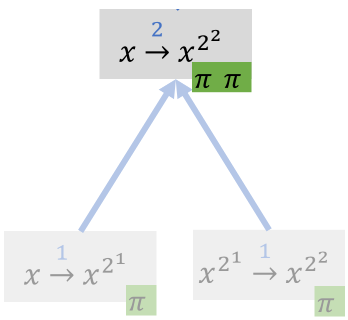
$x \xrightarrow{1} x^{2^1}$ π

$x^{2^1} \xrightarrow{1} x^{2^2}$ π

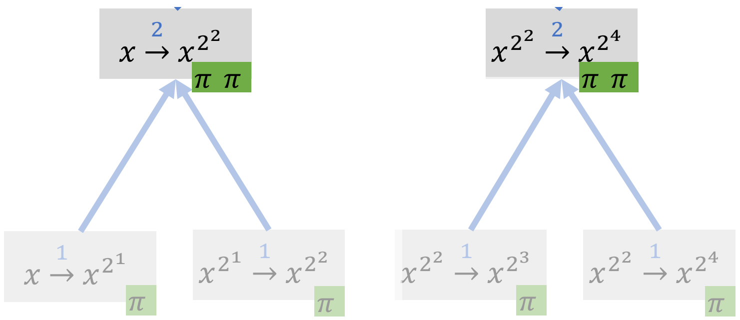
Continuously Verifiable Squaring



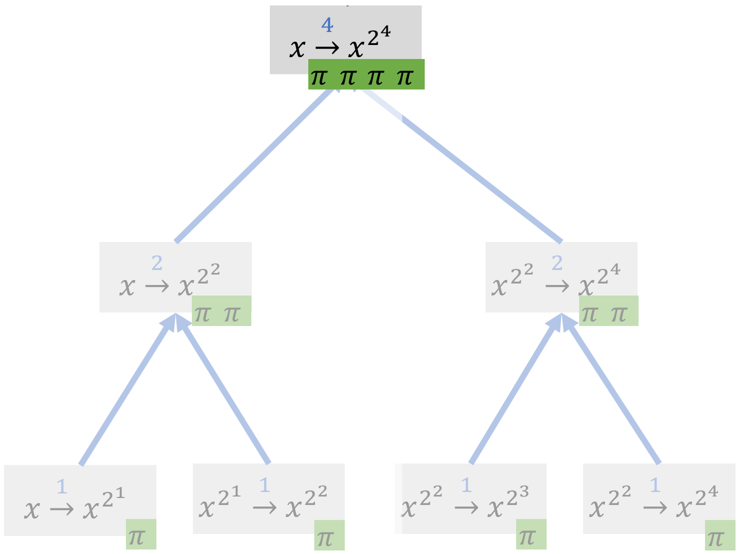
Continuously Verifiable Squaring



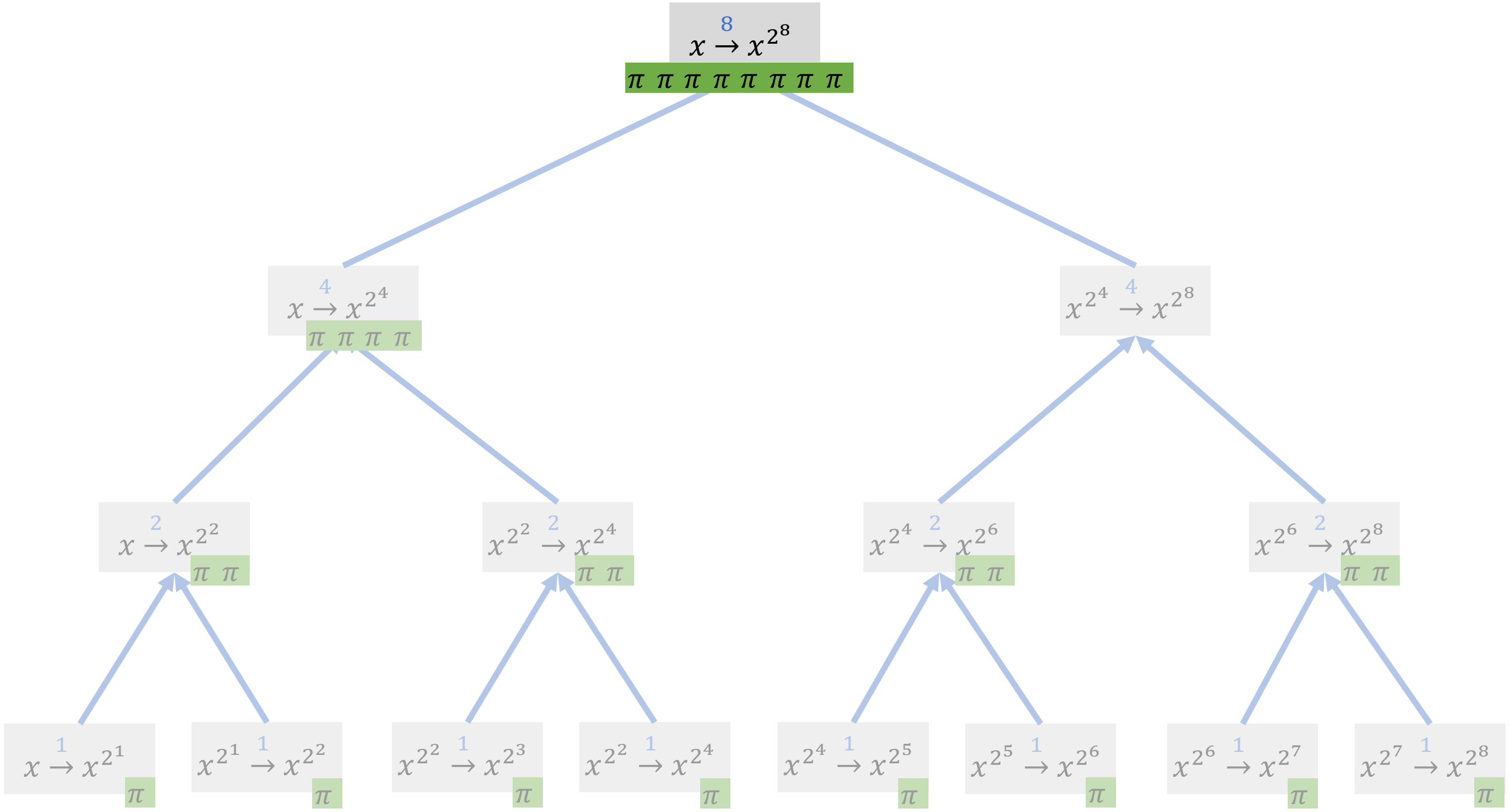
Continuously Verifiable Squaring



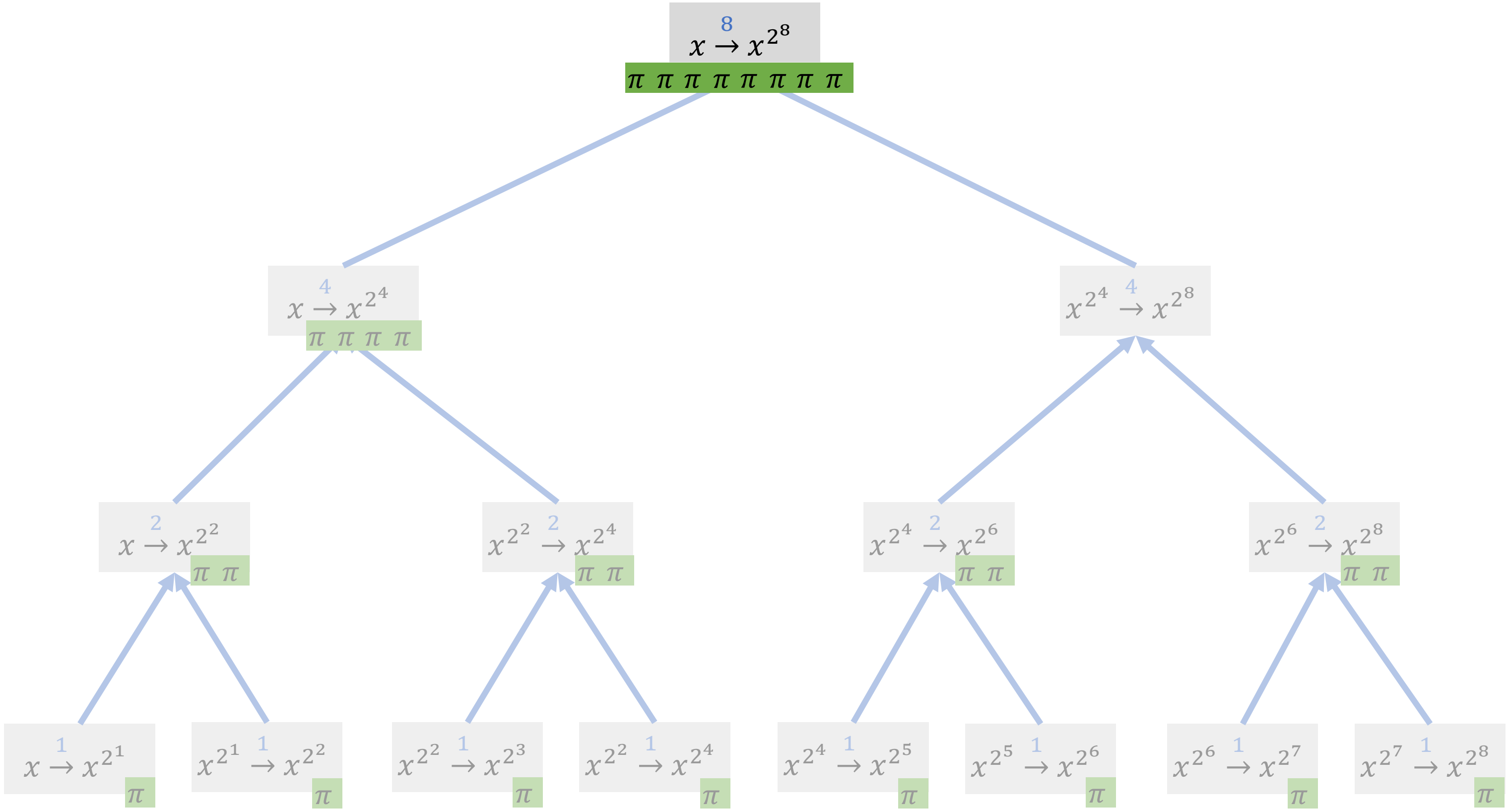
Continuously Verifiable Squaring



Continuously Verifiable Squaring



Continuously Verifiable Squaring



For T squarings, number of proofs is $O(T)$!

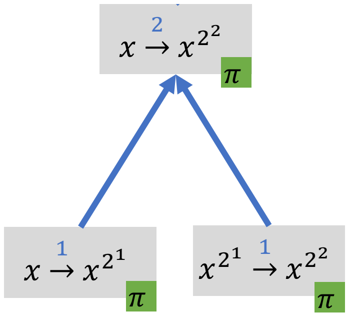
Merge Proofs [Valiant'06]

$x \xrightarrow{1} x^{2^1}$ π

$x^{2^1} \xrightarrow{1} x^{2^2}$ π

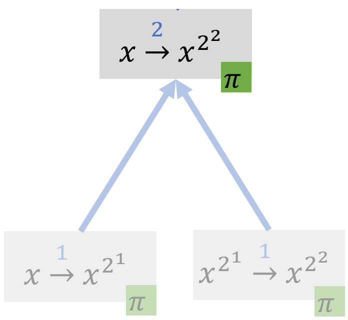
Merge Proofs [Valiant'06]

Merge proofs into a single proof in poly(λ) time



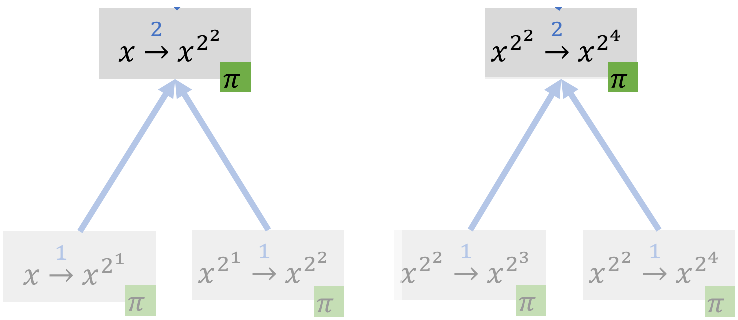
Merge Proofs [Valiant'06]

Merge proofs into a single proof in poly(λ) time



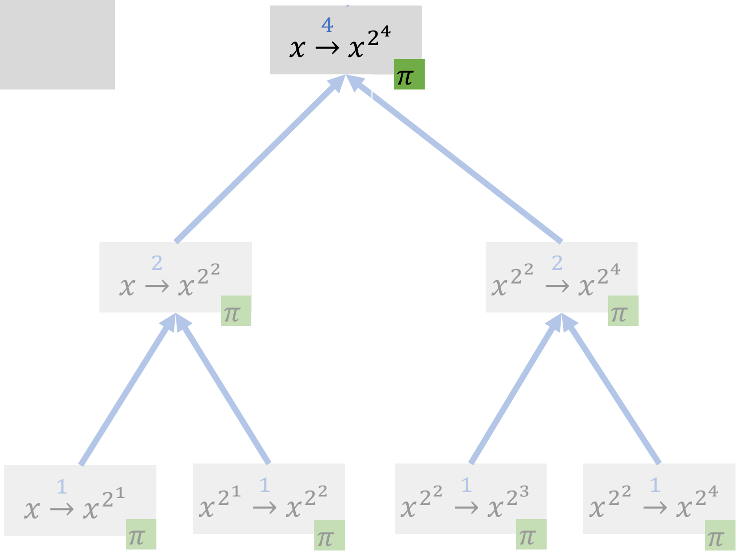
Merge Proofs [Valiant’06]

Merge proofs into a single proof in poly(λ) time



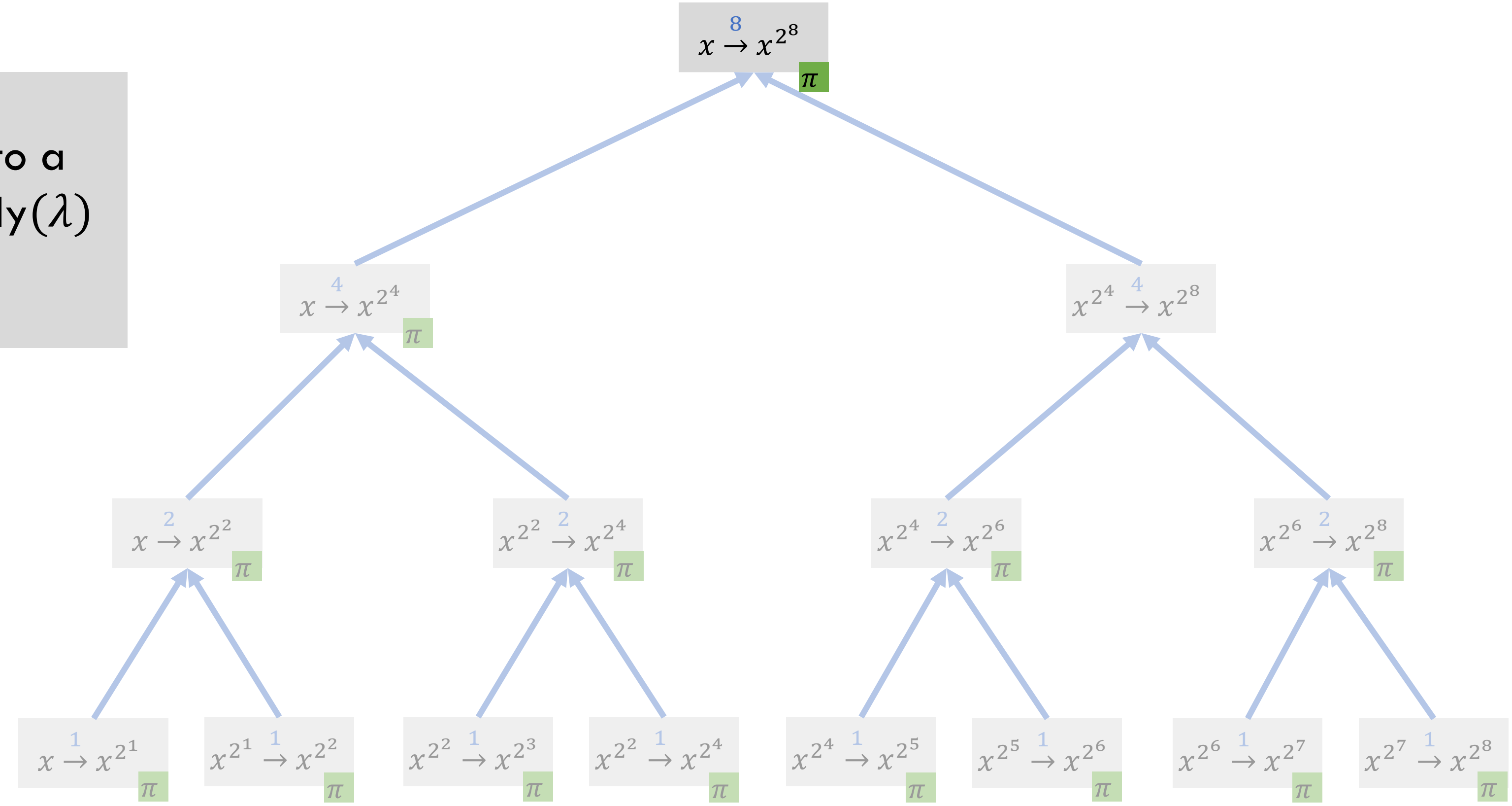
Merge Proofs [Valiant'06]

Merge proofs into a single proof in poly(λ) time



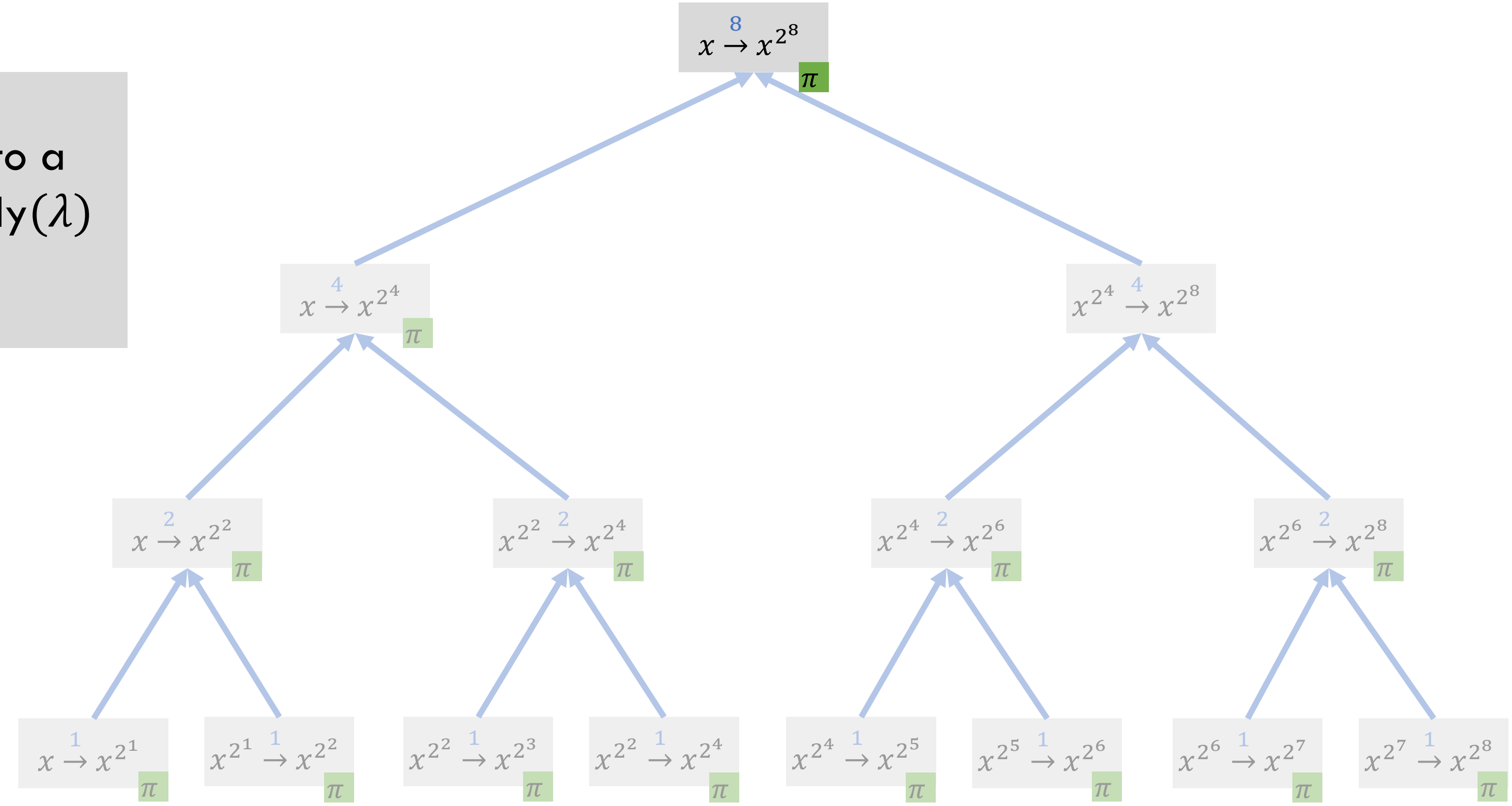
Merge Proofs [Valiant'06]

Merge proofs into a single proof in poly(λ) time



Merge Proofs [Valiant'06]

Merge proofs into a single proof in poly(λ) time

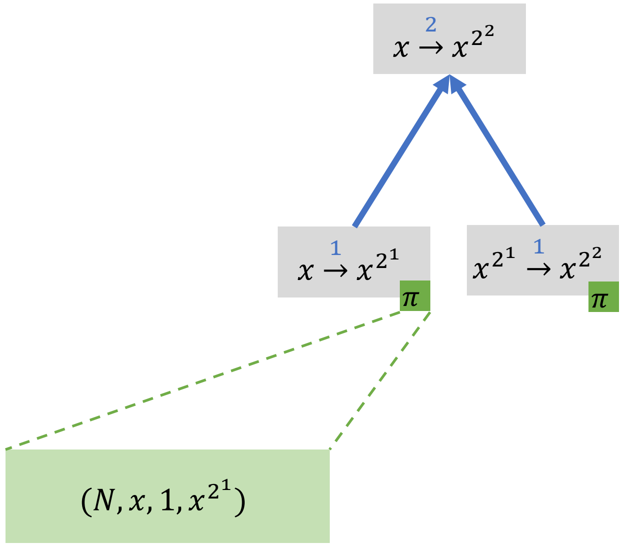


For T squarings, number of proofs is $O(1)$!

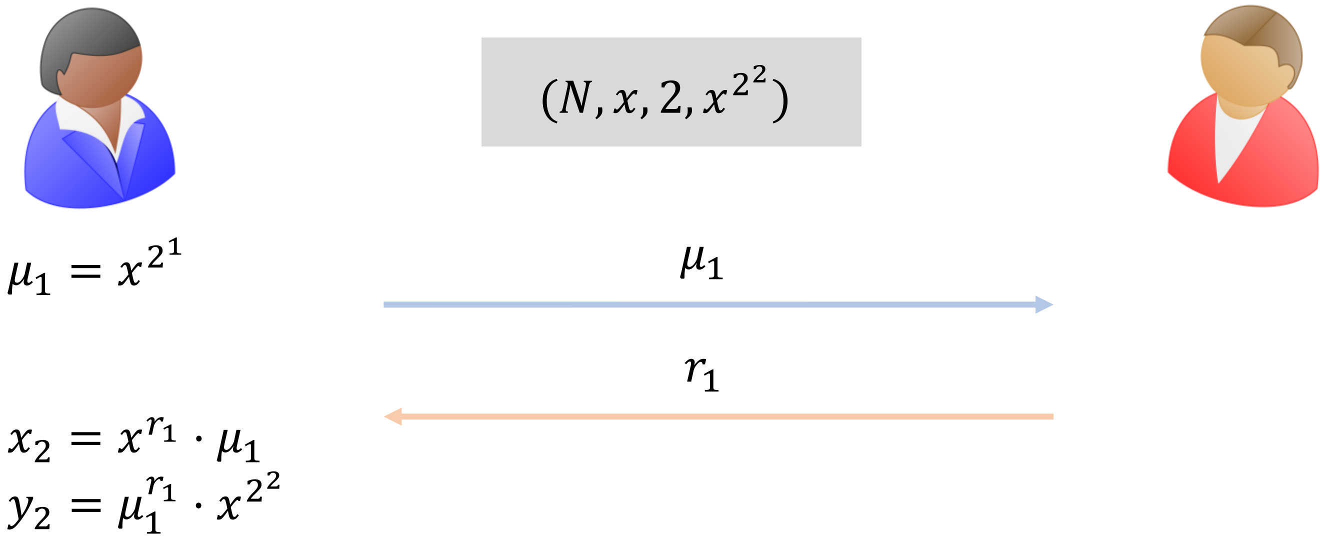
Non-standard assumptions.

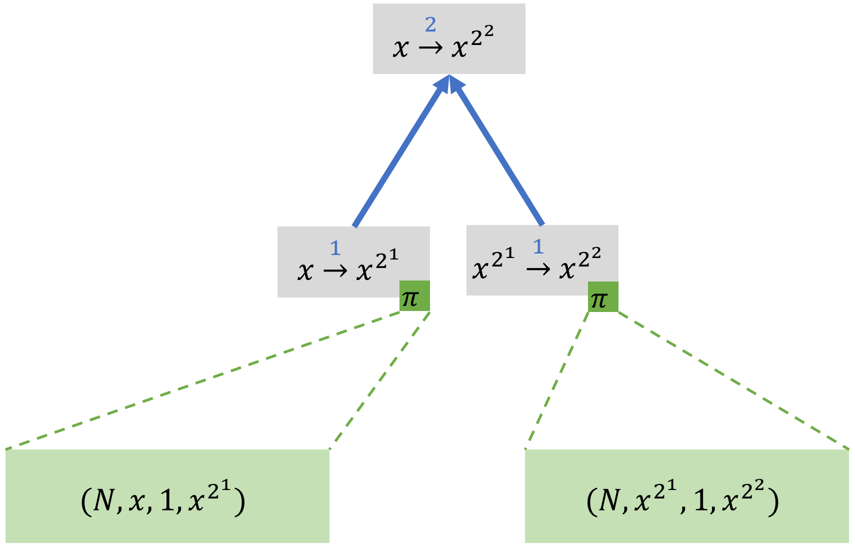
Incremental Merge

[C-Hubáček-Kamth-Pietrzak-Rosen-Rothblum'19]



Incremental Merge





$$\pi = (N, x, 2, x^{2^2}, \mu_1)$$

Incremental Merge

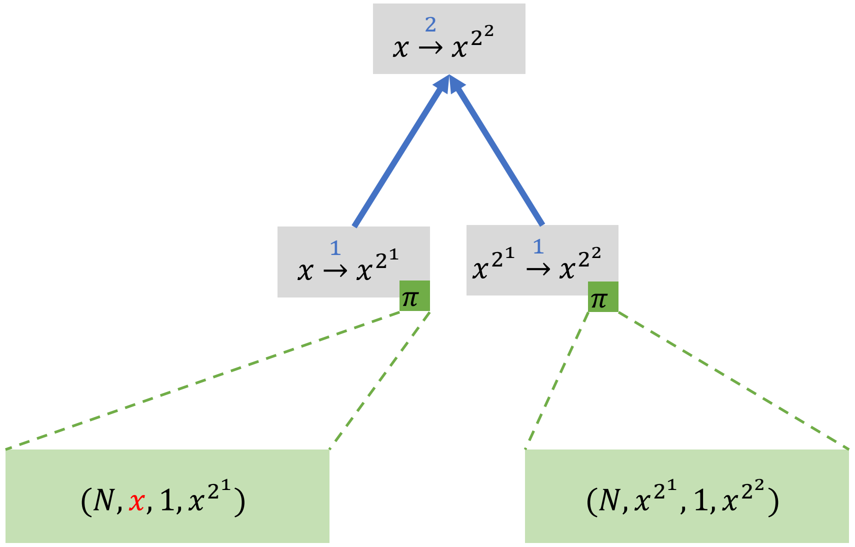
$$(N, x, 2, x^{2^2})$$

$\mu_1 = x^{2^1}$

$\xrightarrow{\mu_1}$

$\xleftarrow{r_1}$

$$x_2 = x^{r_1} \cdot \mu_1$$
$$y_2 = \mu_1^{r_1} \cdot x^{2^2}$$



$$\pi = (N, x, 2, x^{2^2}, \mu_1)$$

Incremental Merge

$(N, x, 2, x^{2^2})$

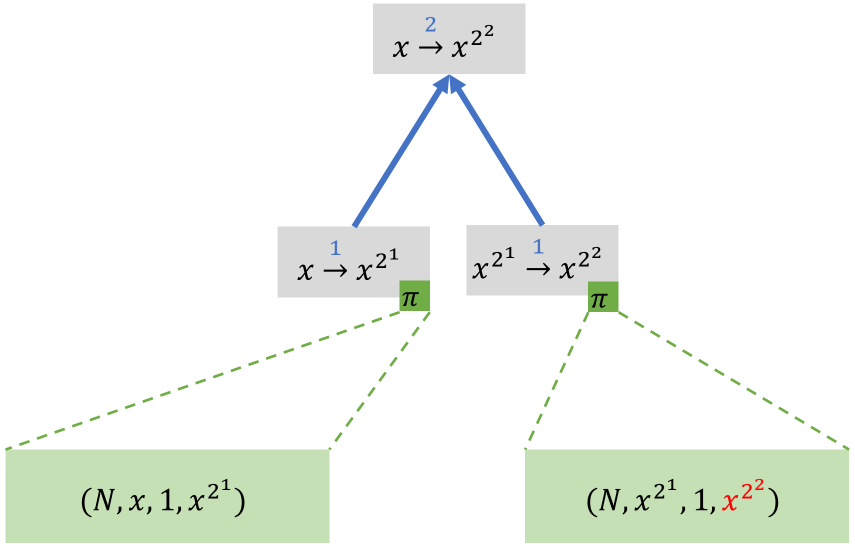
$\mu_1 = x^{2^1}$

$\xrightarrow{\mu_1}$

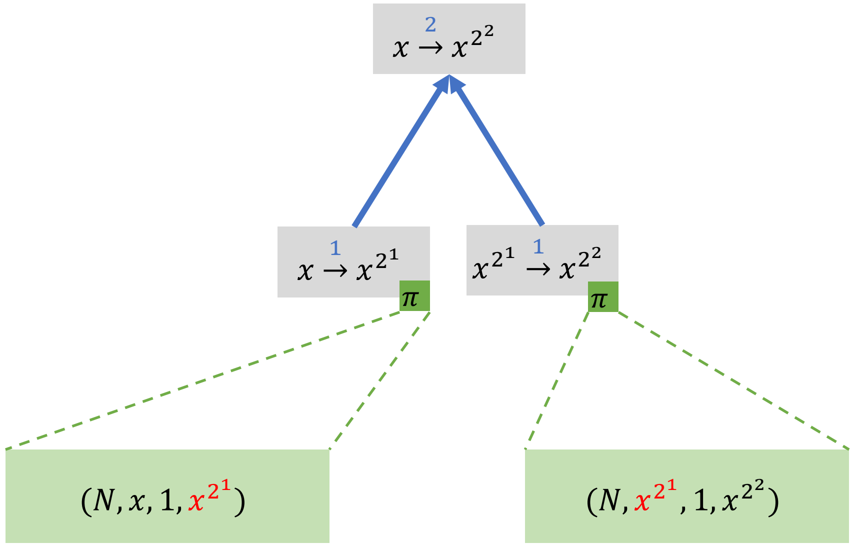
$\xleftarrow{r_1}$

$x_2 = x^{r_1} \cdot \mu_1$
$y_2 = \mu_1^{r_1} \cdot x^{2^2}$

$\pi = (N, x, 2, x^{2^2}, \mu_1)$



Incremental Merge



$(N, x, 2, x^{2^2})$

$\mu_1 = x^{2^1}$

$\xrightarrow{\mu_1}$

$\xleftarrow{r_1}$

$x_2 = x^{r_1} \cdot \mu_1$

$y_2 = \mu_1^{r_1} \cdot x^{2^2}$

$\pi = (N, x, 2, x^{2^2}, \mu_1)$

Incremental Merge

$(N, x, 2, x^{2^2})$

$\mu_1 = x^{2^1}$

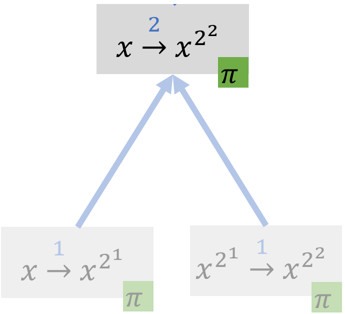
$\xrightarrow{\mu_1}$

$\xleftarrow{r_1}$

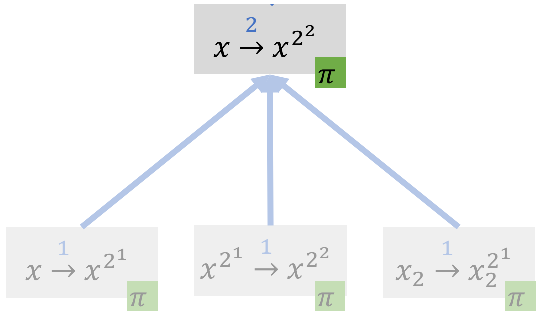
$x_2 = x^{r_1} \cdot \mu_1$

$y_2 = \mu_1^{r_1} \cdot x^{2^2}$

$\pi = (N, x, 2, x^{2^2}, \mu_1)$



Incremental Merge



$(N, x, 2, x^{2^2})$

$\mu_1 = x^{2^1}$

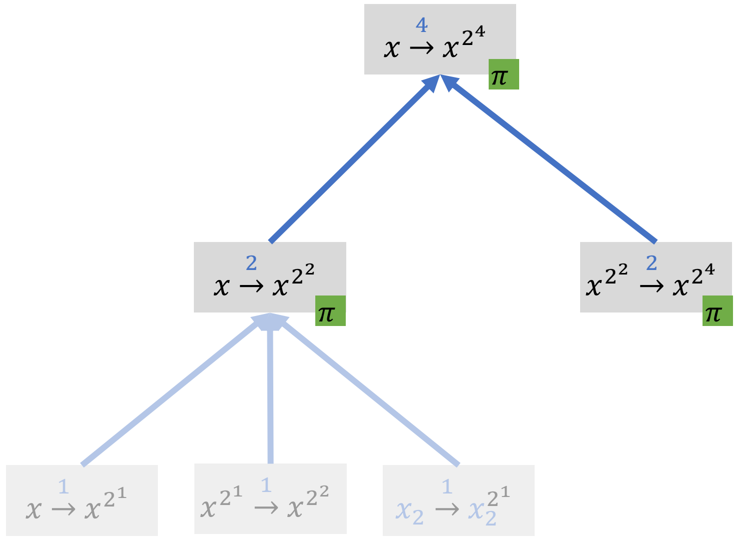
$\xrightarrow{\mu_1}$

$\xleftarrow{r_1}$

$x_2 = x^{r_1} \cdot \mu_1$

$y_2 = \mu_1^{r_1} \cdot x^{2^2}$

Incremental Merge



$(N, x, 4, x^{2^4})$

$\mu_1 = x^{2^2}$ $\quad \xrightarrow{\mu_1}$

$\xleftarrow{r_1}$

$x_2 = x^{r_1} \cdot \mu_1$

$y_2 = \mu_1^{r_1} \cdot x^{2^4}$

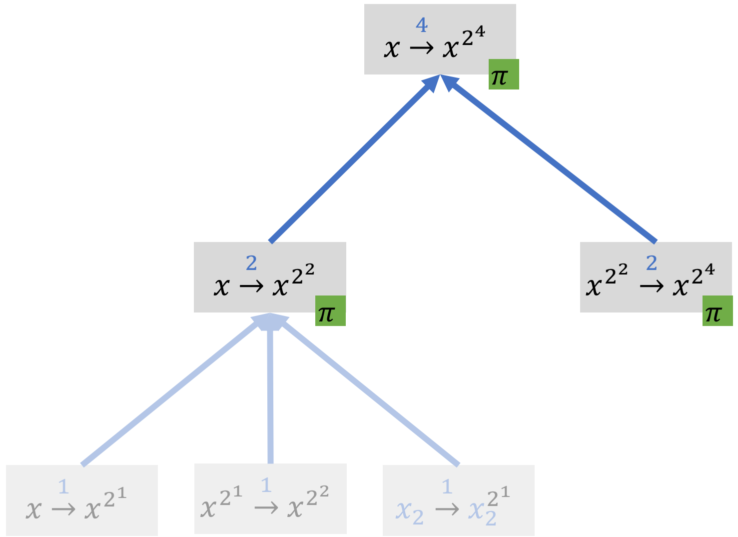
$\mu_2 = x_2^{2^1}$ $\quad \xrightarrow{\mu_2}$

$\xleftarrow{r_2}$

$x_3 = x_2^{r_2} \cdot \mu_2$

$y_3 = \mu_2^{r_2} \cdot y_2$

Incremental Merge



$(N, x, 4, x^{2^2})$

$\mu_1 = x^{2^2}$

μ_1

r_1

$x_2 = x^{r_1} \cdot \mu_1$

$y_2 = \mu_1^{r_1} \cdot x^{2^4}$

$(N, x_2, 2, y_2)$

$\mu_2 = x_2^{2^1}$

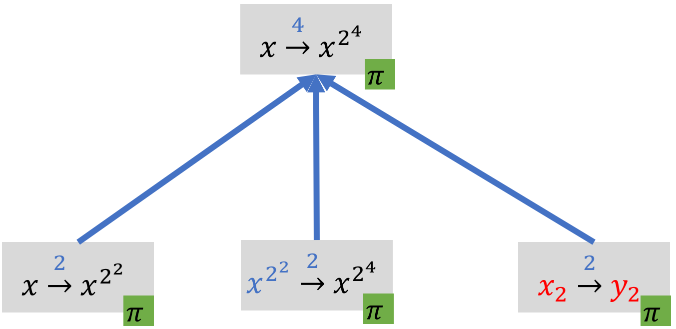
μ_2

r_2

$x_3 = x_2^{r_2} \cdot \mu_2$

$y_3 = \mu_2^{r_2} \cdot y_2$

Incremental Merge



$(N, x, 4, x^{2^2})$

$\mu_1 = x^{2^2}$ $\xrightarrow{\mu_1}$

r_1

$x_2 = x^{r_1} \cdot \mu_1$

$y_2 = \mu_1^{r_1} \cdot x^{2^4}$

$(N, x_2, 2, y_2)$

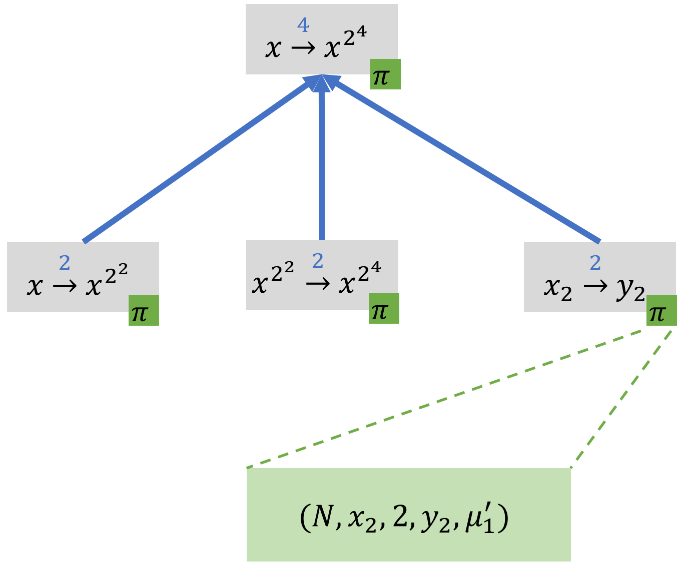
$\mu_2 = x_2^{2^1}$ $\xrightarrow{\mu_2}$

$\xleftarrow{r_2}$

$x_3 = x_2^{r_2} \cdot \mu_2$

$y_3 = \mu_2^{r_2} \cdot y_2$

Incremental Merge



$(N, x, 4, x^{2^2})$

$\mu_1 = x^{2^2}$ $\xrightarrow{\mu_1}$

$\xleftarrow{r_1}$

$x_2 = x^{r_1} \cdot \mu_1$

$y_2 = \mu_1^{r_1} \cdot x^{2^4}$

$\mu_2 = x_2^{2^1}$ $\xrightarrow{\mu_2}$

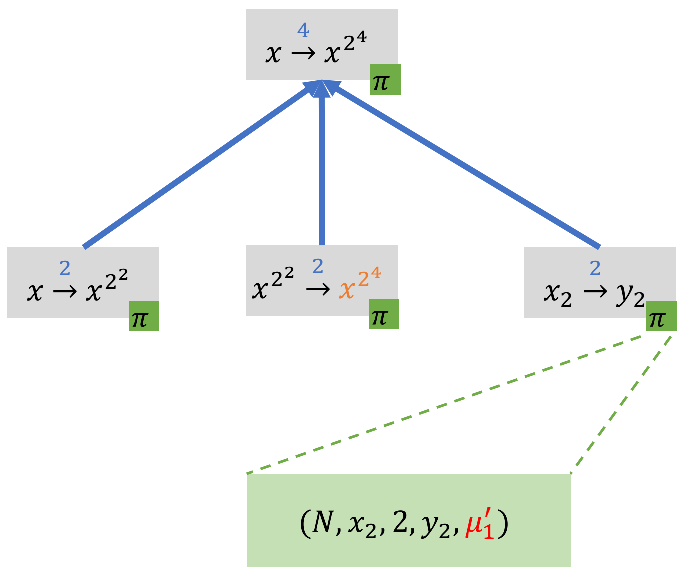
$\xleftarrow{r_2}$

$x_3 = x_2^{r_2} \cdot \mu_2$

$y_3 = \mu_2^{r_2} \cdot y_2$

$\pi = (N, x, 4, x^{2^4}, \mu_1, \mu_2)$

Incremental Merge



$(N, x, 4, x^{2^2})$

$\mu_1 = x^{2^2}$ $\quad\xrightarrow{\mu_1}$

$\xleftarrow{r_1}$

$x_2 = x^{r_1} \cdot \mu_1$

$y_2 = \mu_1^{r_1} \cdot x^{2^4}$

$\mu_2 = x_2^{2^1}$ $\quad\xrightarrow{\mu_2}$

$\xleftarrow{r_2}$

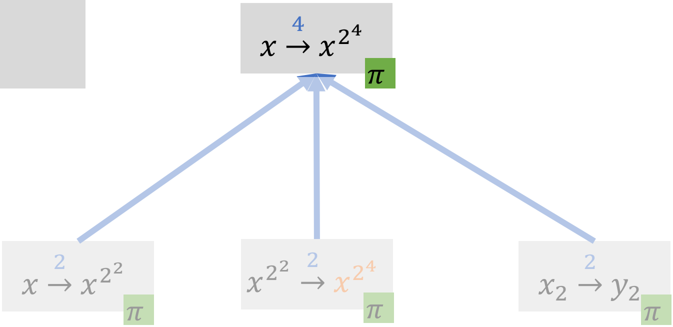
$x_3 = x_2^{r_2} \cdot \mu_2$

$y_3 = \mu_2^{r_2} \cdot y_2$

$\pi = (N, x, 4, x^{2^4}, \mu_1, \mu_2)$

Incremental Merge

Merge two proof for $T/2$ squarings in $O(T/2)$ time.



$(N, x, 4, x^{2^2})$

$\mu_1 = x^{2^2}$

μ_1

r_1

$x_2 = x^{r_1} \cdot \mu_1$

$y_2 = \mu_1^{r_1} \cdot x^{2^4}$

$\mu_2 = x_2^{2^1}$

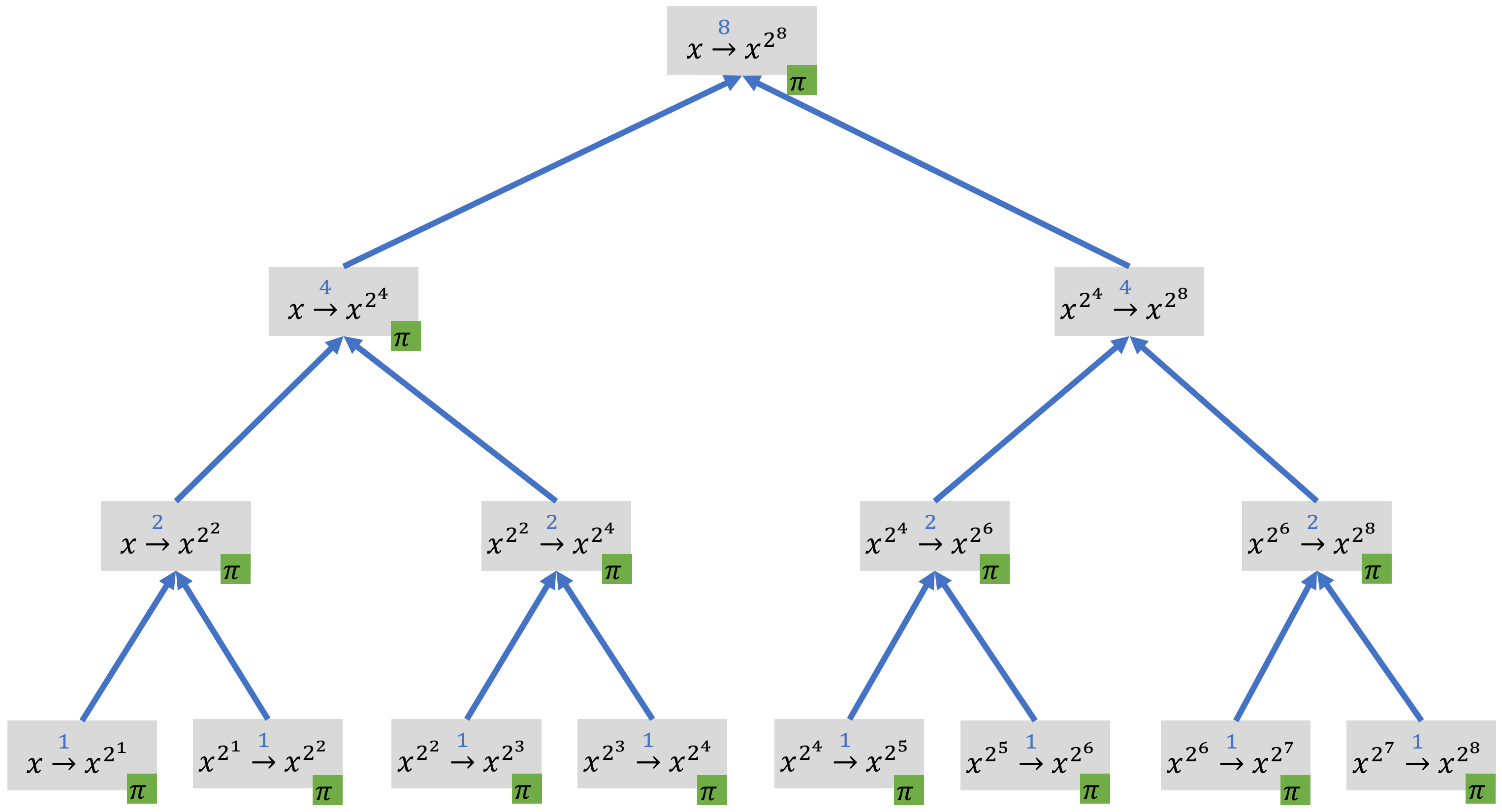
μ_2

r_2

$x_3 = x_2^{r_2} \cdot \mu_2$

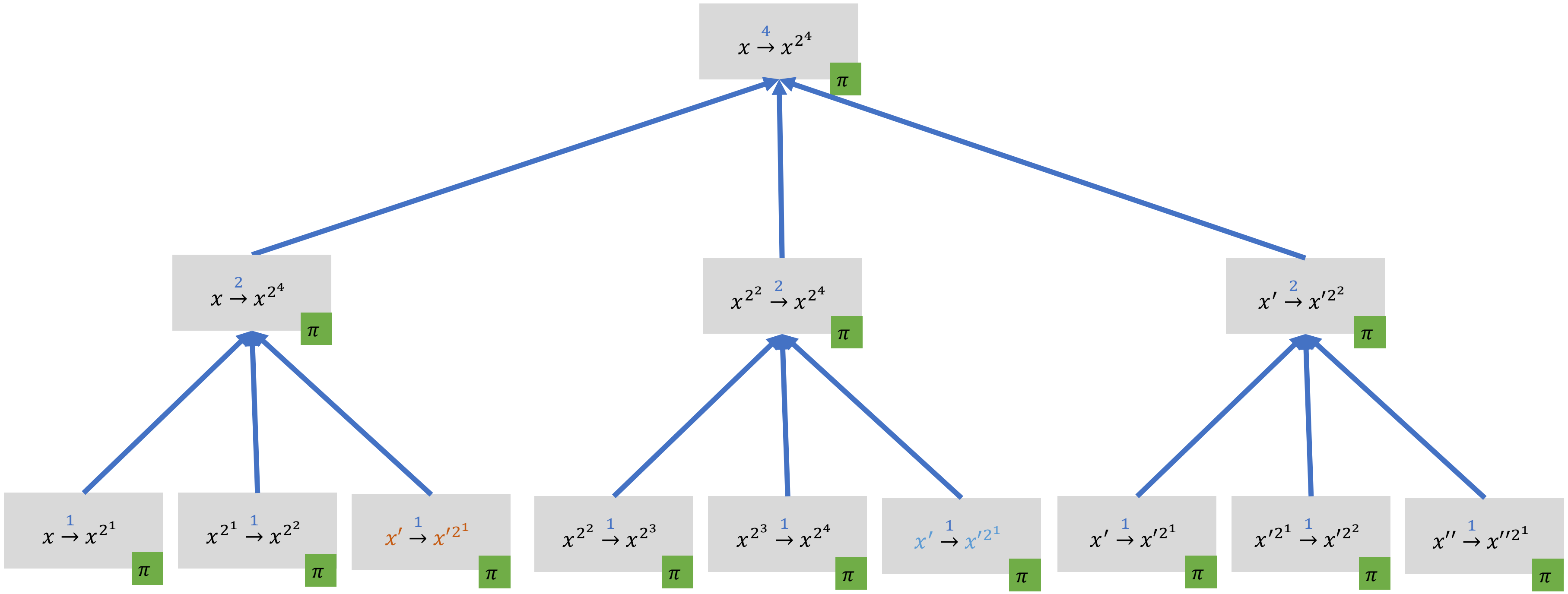
$y_3 = \mu_2^{r_2} \cdot y_2$

$\pi = (N, x, 4, x^{2^4}, \mu_1, \mu_2)$

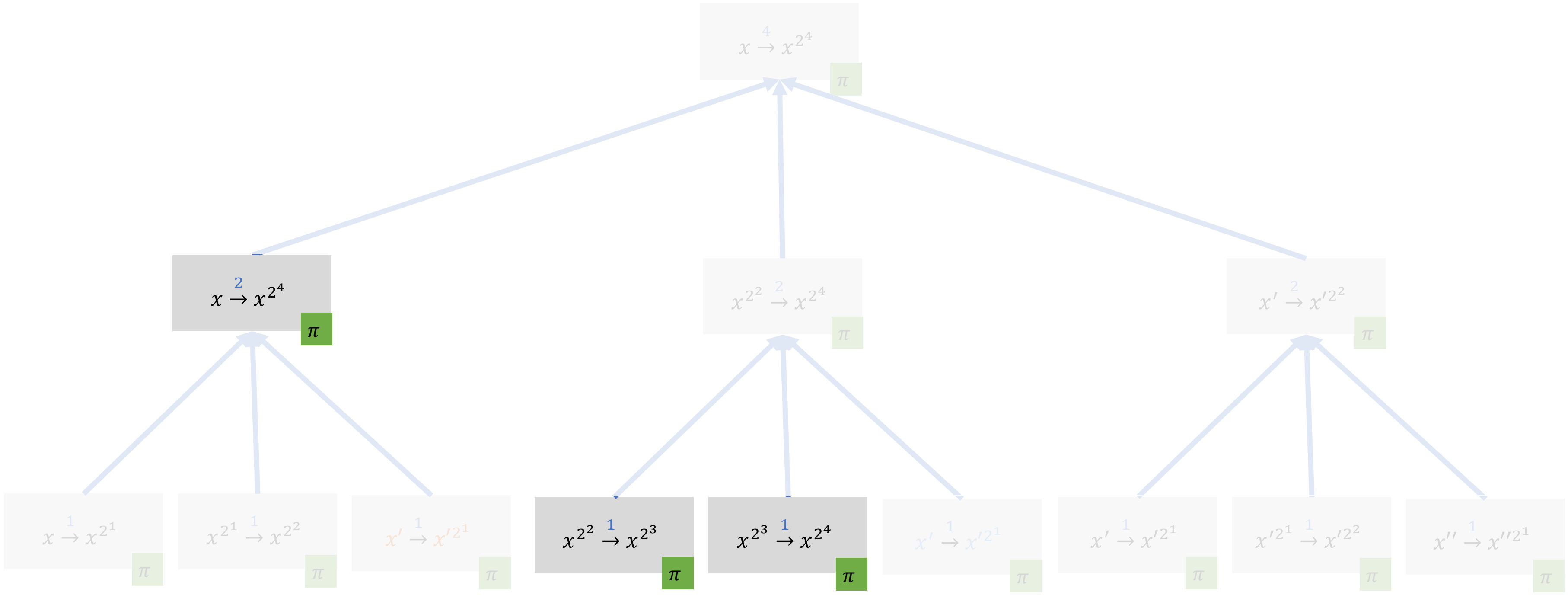


$x \xrightarrow{8} x^{2^8}$ π
$x \xrightarrow{4} x^{2^4}$ π
$x^{2^4} \xrightarrow{4} x^{2^8}$
$x \xrightarrow{2} x^{2^2}$ π
$x^{2^2} \xrightarrow{2} x^{2^4}$ π
$x^{2^4} \xrightarrow{2} x^{2^6}$ π
$x^{2^6} \xrightarrow{2} x^{2^8}$ π
$x \xrightarrow{1} x^{2^1}$ π
$x^{2^1} \xrightarrow{1} x^{2^2}$ π
$x^{2^2} \xrightarrow{1} x^{2^3}$ π
$x^{2^3} \xrightarrow{1} x^{2^4}$ π
$x^{2^4} \xrightarrow{1} x^{2^5}$ π
$x^{2^5} \xrightarrow{1} x^{2^6}$ π
$x^{2^6} \xrightarrow{1} x^{2^7}$ π
$x^{2^7} \xrightarrow{1} x^{2^8}$ π

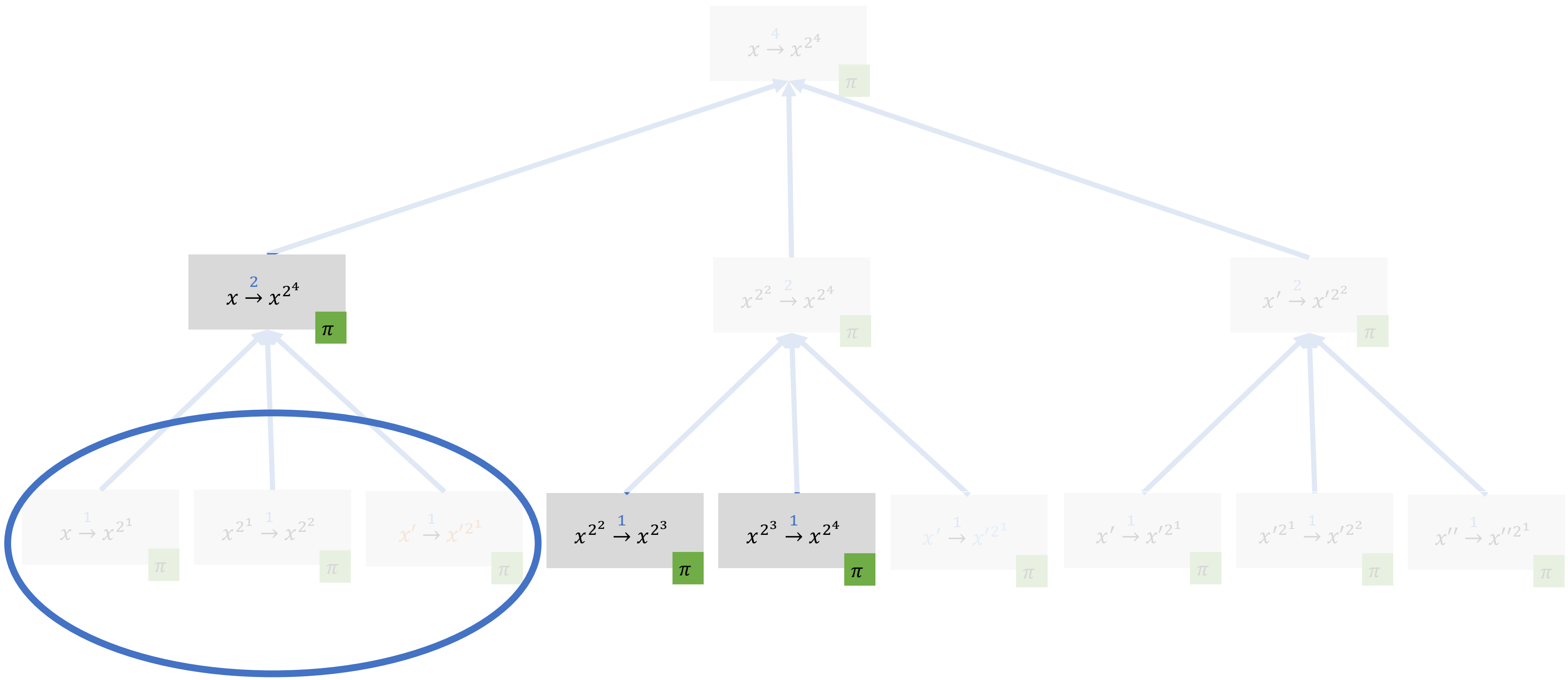
Internal State



Internal State

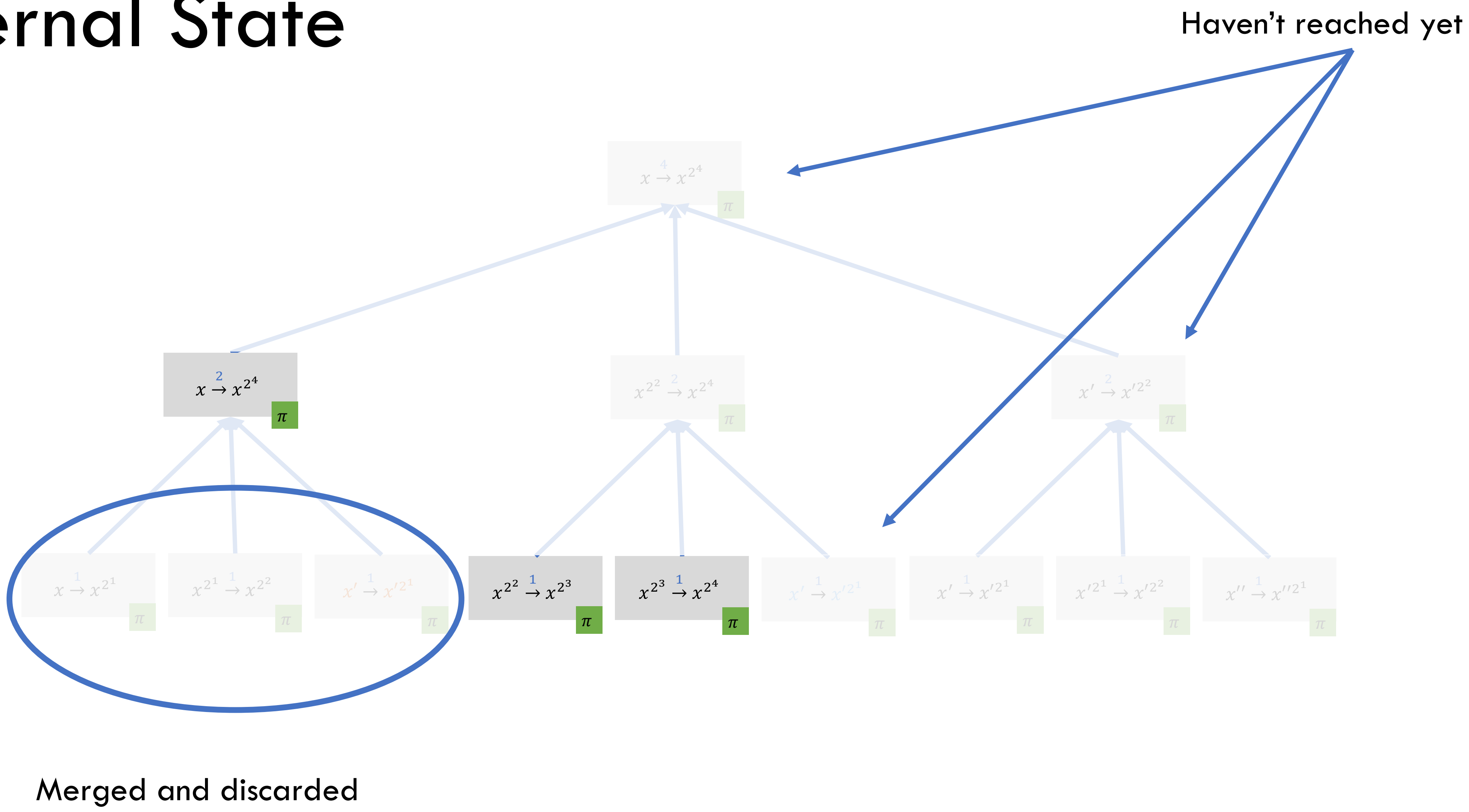


Internal State

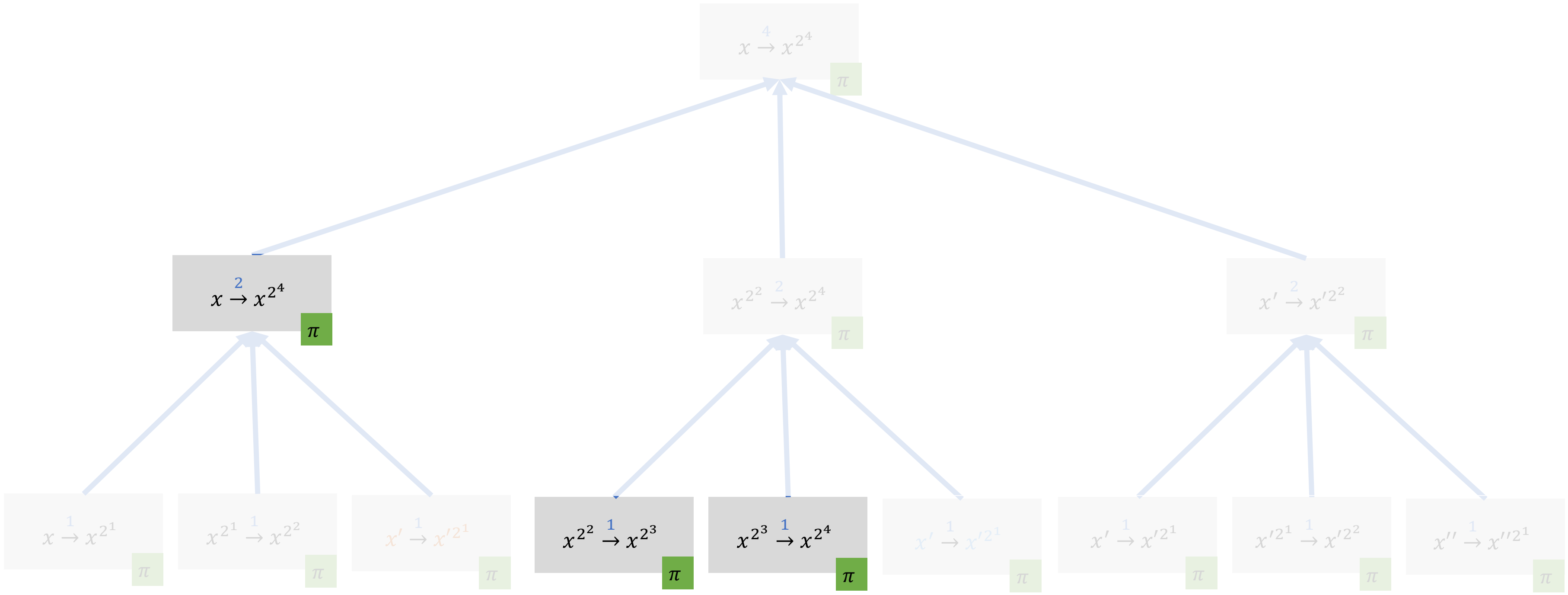


Merged and discarded

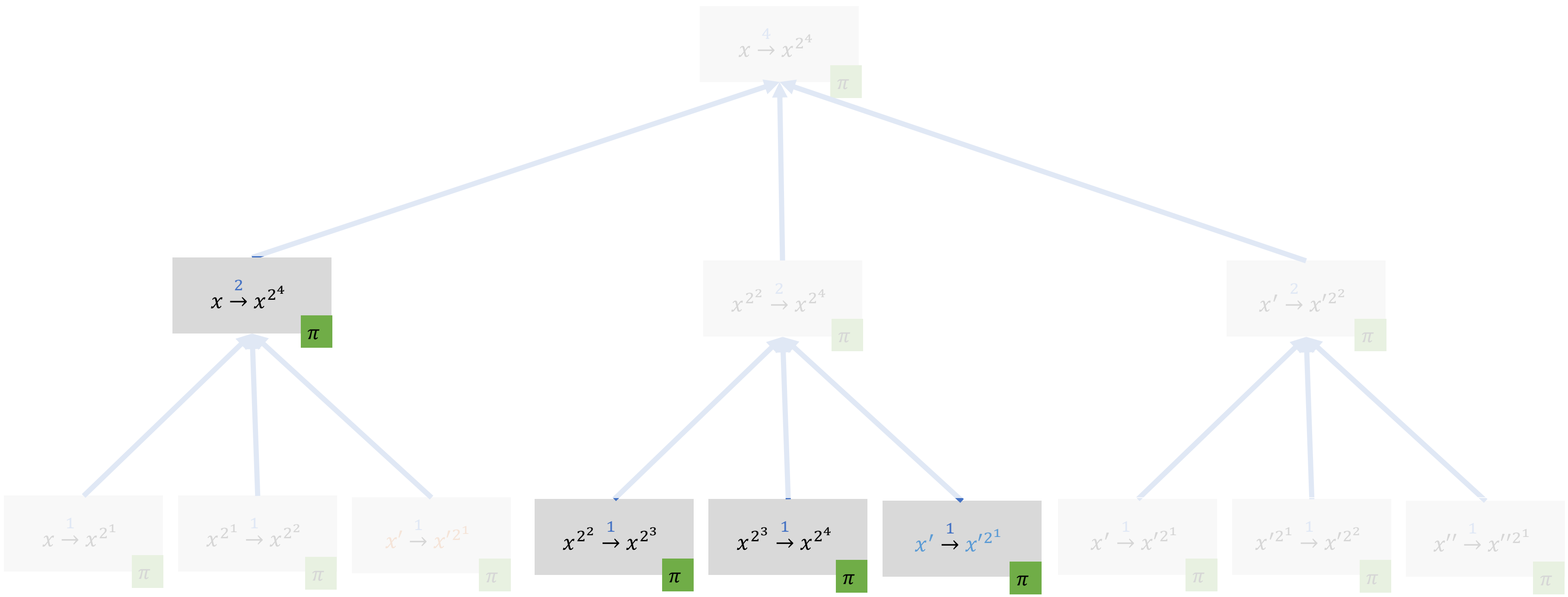
Internal State



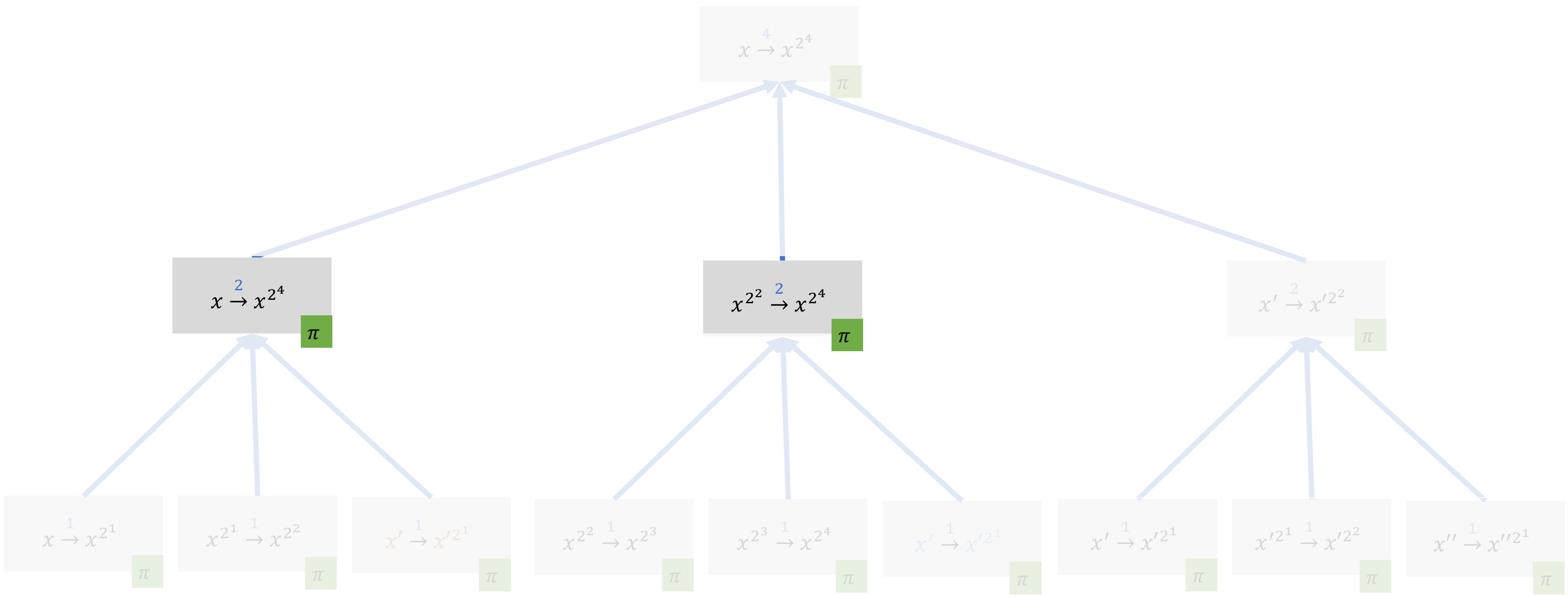
Internal State



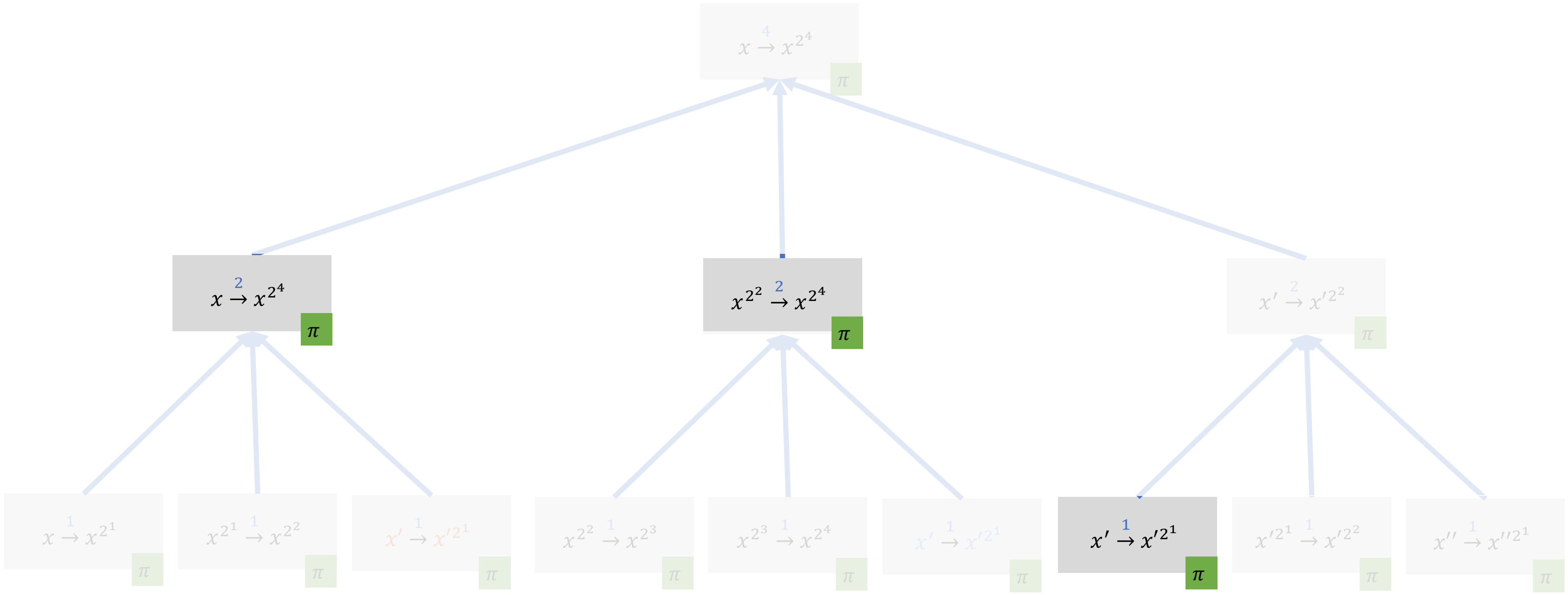
Internal State



Internal State

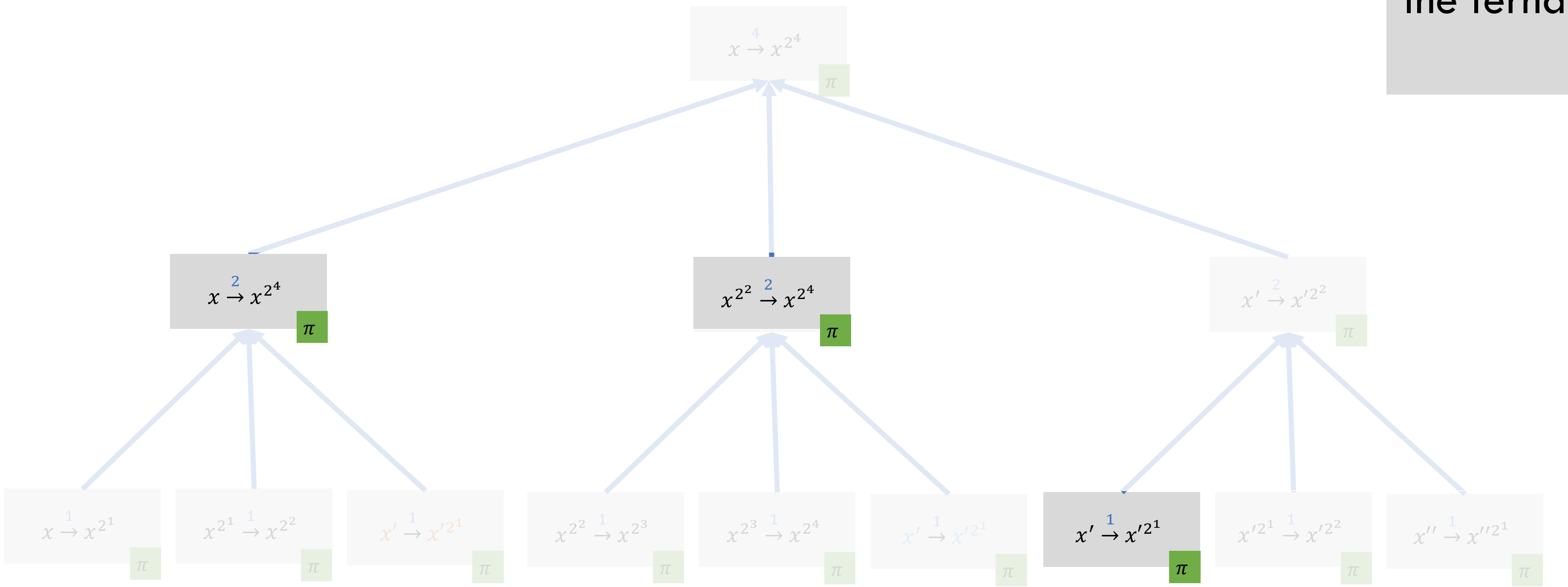


Internal State



Internal State

Depth first traversal of the ternary tree



Verifying i-th state:

1. Determine which nodes are active in i-th step of depth first traversal.
2. Verify proofs in each active node.

Putting it together: cVDF

Compute $x \xrightarrow{T} x^{2^T}$ in a continuous verifiable manner

Compute root of ternary tree

Prover cost

$$P(T) = 3P(T/2)$$

Proof size

$$O(\text{polylog}(T))$$

Part 2: Nash Equilibrium

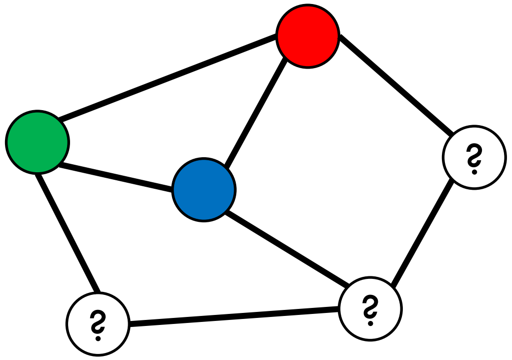
Sink of Verifiable Line (SVL)

Game Theory and Nash Equilibrium

	Left	Right
Left	1 \ -1	-1\ 1
Right	-1 \ 1	1 \ -1

[Nash'51]: A (mixed) equilibrium always exists

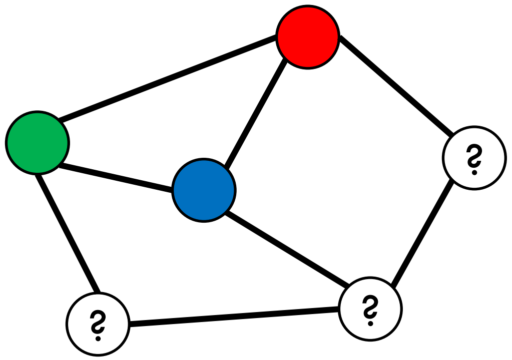
How hard is finding a Nash Equilibrium?



Reduction?

	Left	Right
Left	1 \ -1	-1\ 1
Right	-1 \ 1	1 \ -1

How hard is finding a Nash Equilibrium?



	Left	Right
Left	1 \ -1	-1\ 1
Right	-1 \ 1	1 \ -1

Complexity of Computing a Nash Equilibrium

Computing Nash unlikely to be FNP-hard

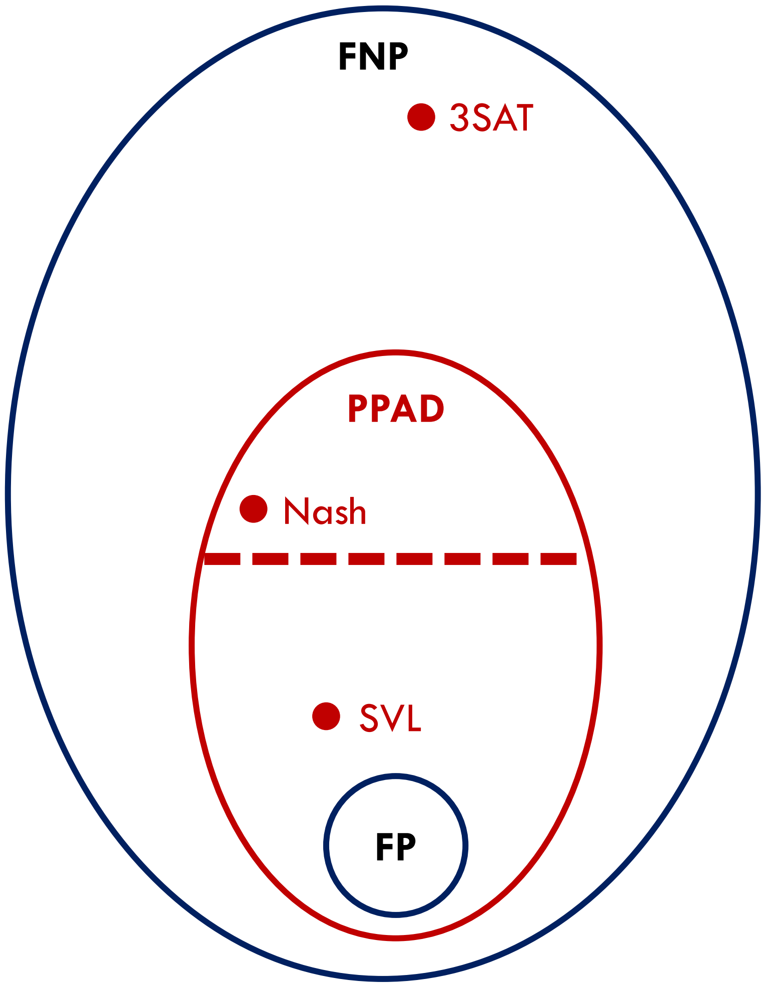
Look to Cryptography for hardness

Known from strong assumptions of **obfuscation**

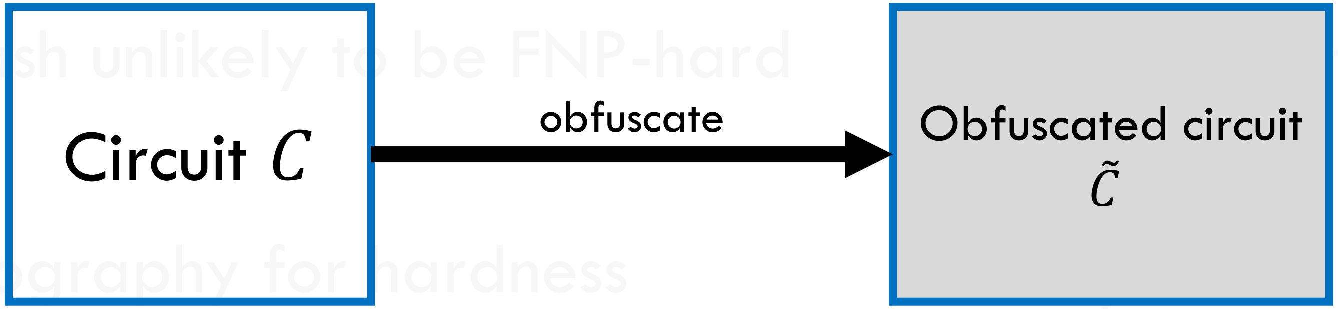
[Bitansky-Paneth-Rosen'15]

[Garg-Pandey-Srinivasan'16]

[Komorgodski-Segev'17]



Complexity of Computing a Nash Equilibrium



Can only observe input-output behavior

Known from strong assumptions of **obfuscation**

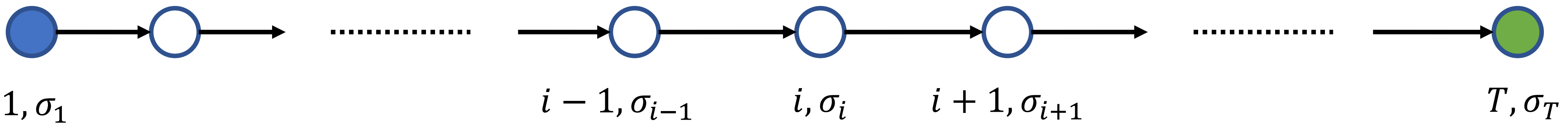
[Bitansky-Paneth-Rosen'15]

[Garg-Pandey-Srinivasan'16]

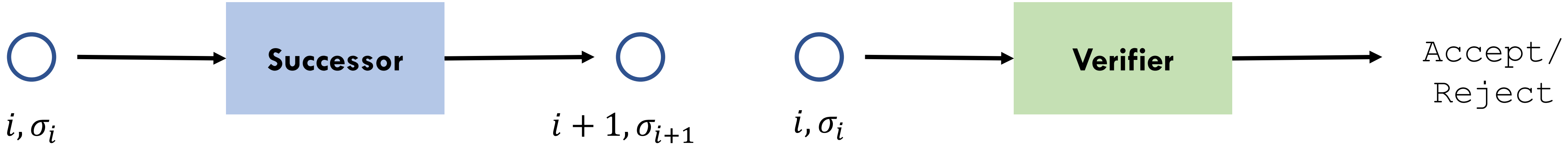
[Komorgodski-Segev'17]

Sink of Verifiable Line (SVL)

[Abbott-Kane-Valiant'04,Bitansky-Paneth Rosen'15]

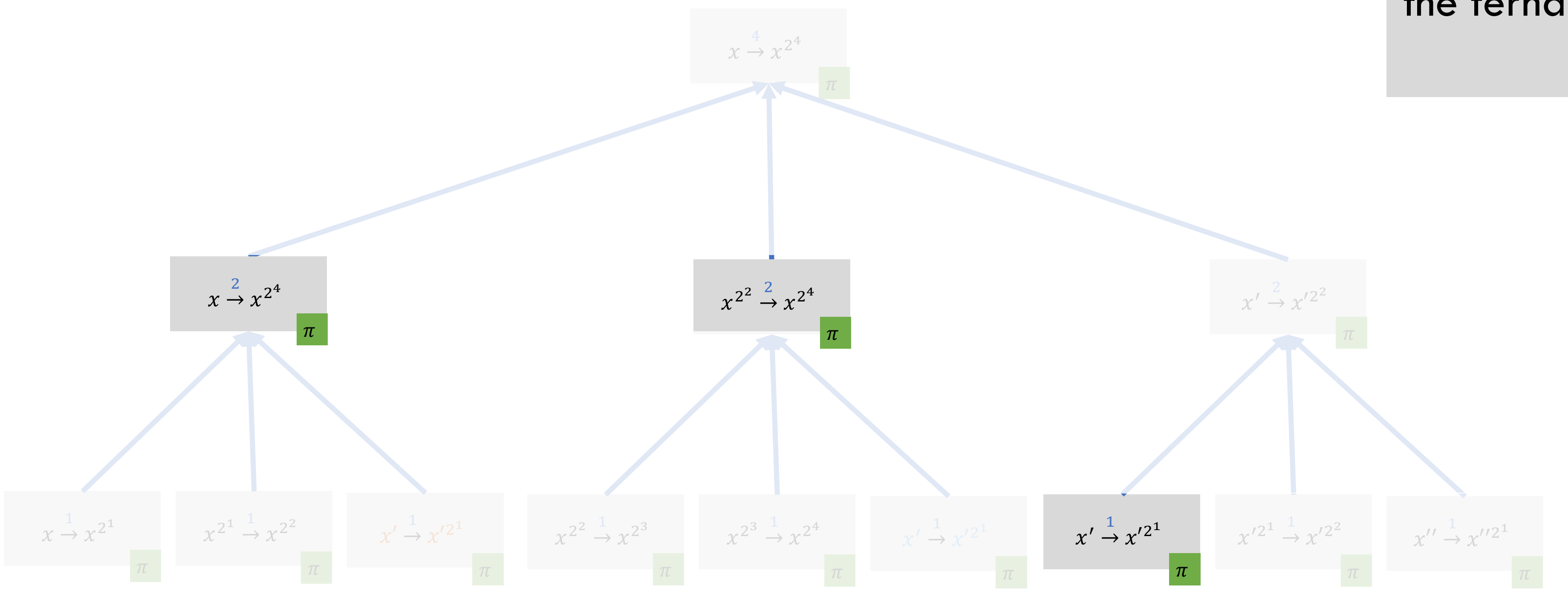


Goal: Find (T, σ_T) that verifies for an exponential T



Internal State of cVDF

Depth first traversal of the ternary tree



Verifying i-th state:

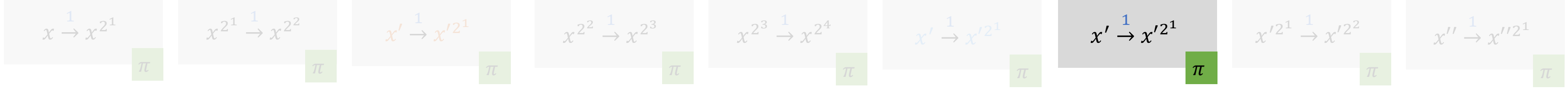
1. Determine which nodes are active in i-th step of depth first traversal.
2. Verify proofs in each active node.

Internal State of cVDF

Depth first traversal of the ternary tree

1. Set T to be exponential in λ.
2. Verification time $\mathrm{polylog}(T) = \mathrm{poly}(\lambda)$.
3. Hard for $\mathrm{poly}(\lambda)$ time adversary to compute $x^{2^T} \bmod N$.

Hard instance of SVL from hardness of repeated squaring (and Fiat-Shamir heuristic).



Verifying i-th state:

1. Determine which nodes are active in i-th step of depth first traversal.
2. Verify proofs in each active node.

Open Problems

VDFs from different assumptions?

VDFs from supersingular isogenies

Instantiating the Fiat-Shamir Heuristic in VDFs?

[Lombardi-Vaikuntanathan'19]

Other applications of continuous VDFs?

Hardness of Nash based on Factoring or other assumptions?

Thank you. Questions?

Arka Rai Choudhuri

achoud@cs.jhu.edu