Delegation with (nearly) optimal time/space overhead

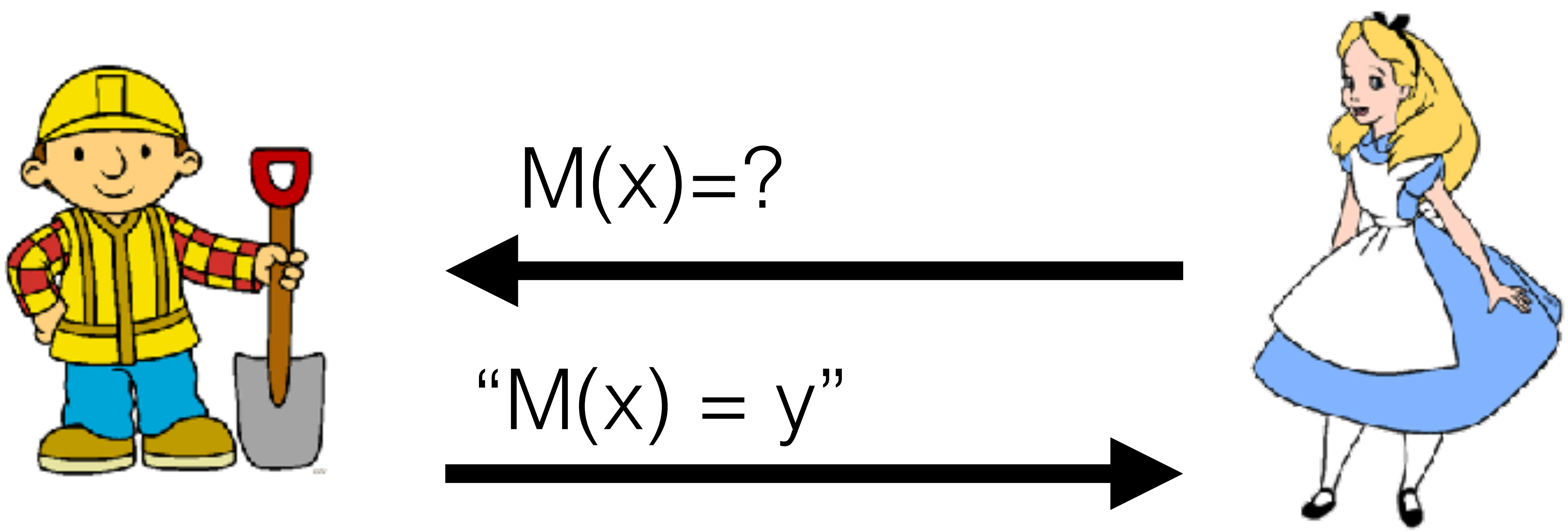
Justin Holmgren
MIT

Ron Rothblum
MIT

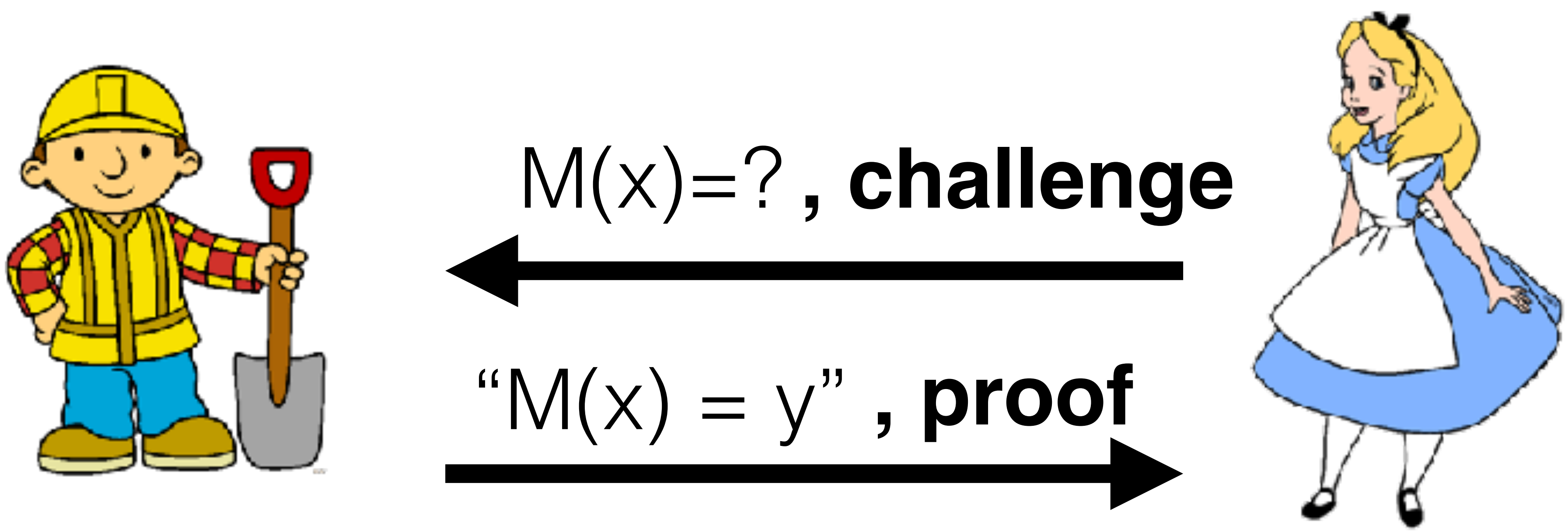
Verifiable Computation

Verifiable Computation

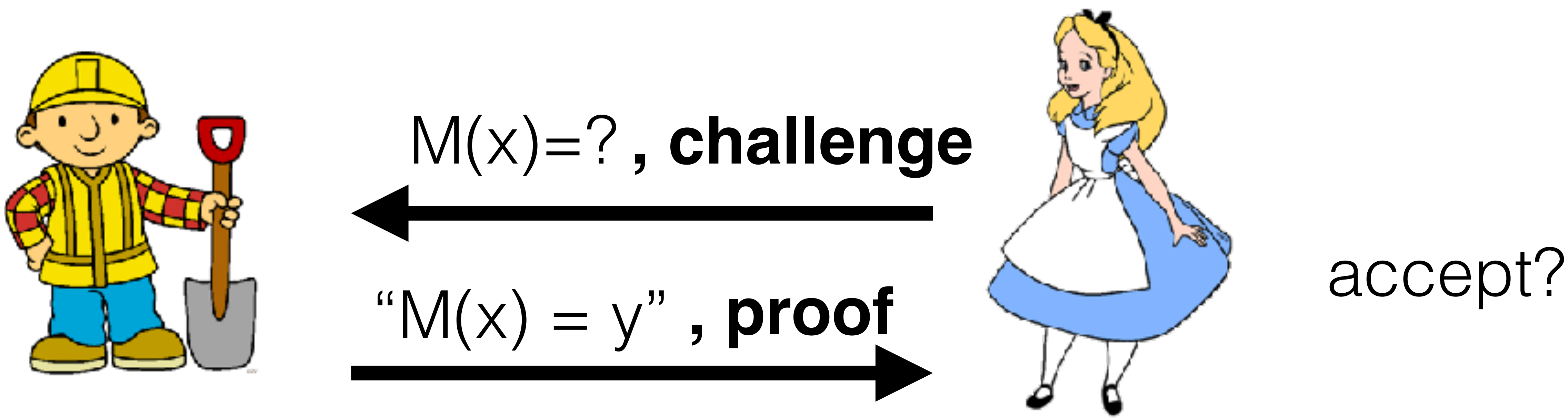
Verifiable Computation



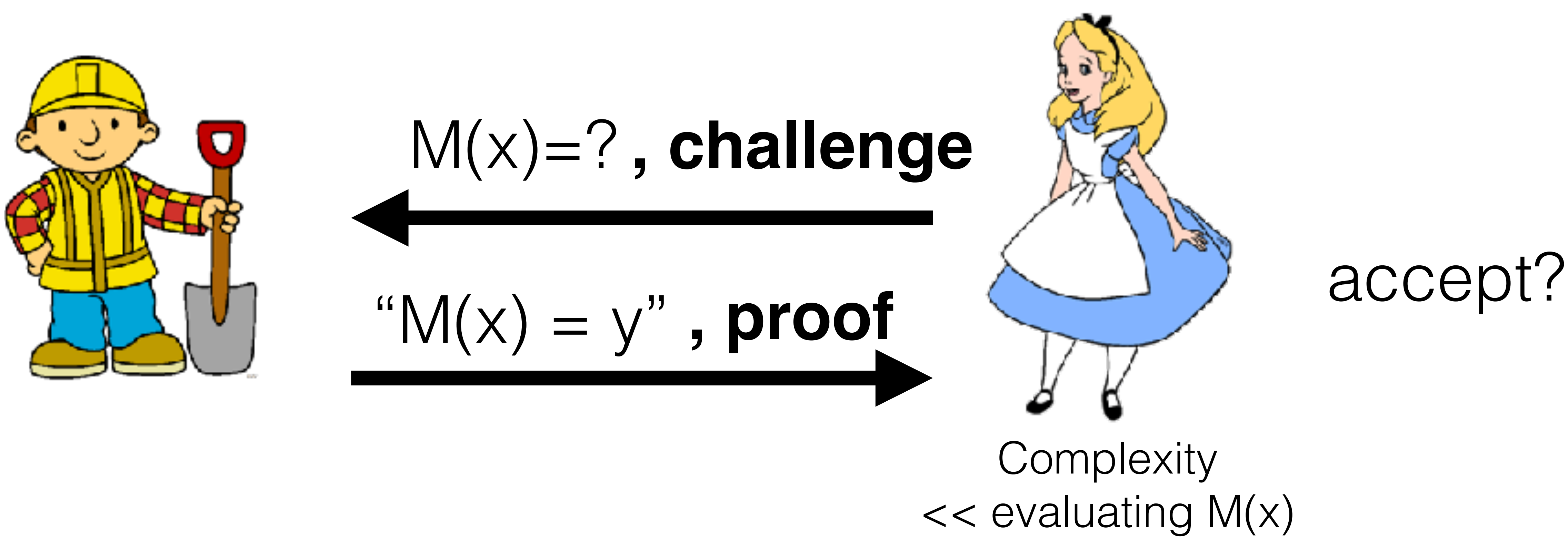
Verifiable Computation



Verifiable Computation



Verifiable Computation



Verifiable Computation

Complexity
~evaluating M(x)

M(x)=? **, challenge**

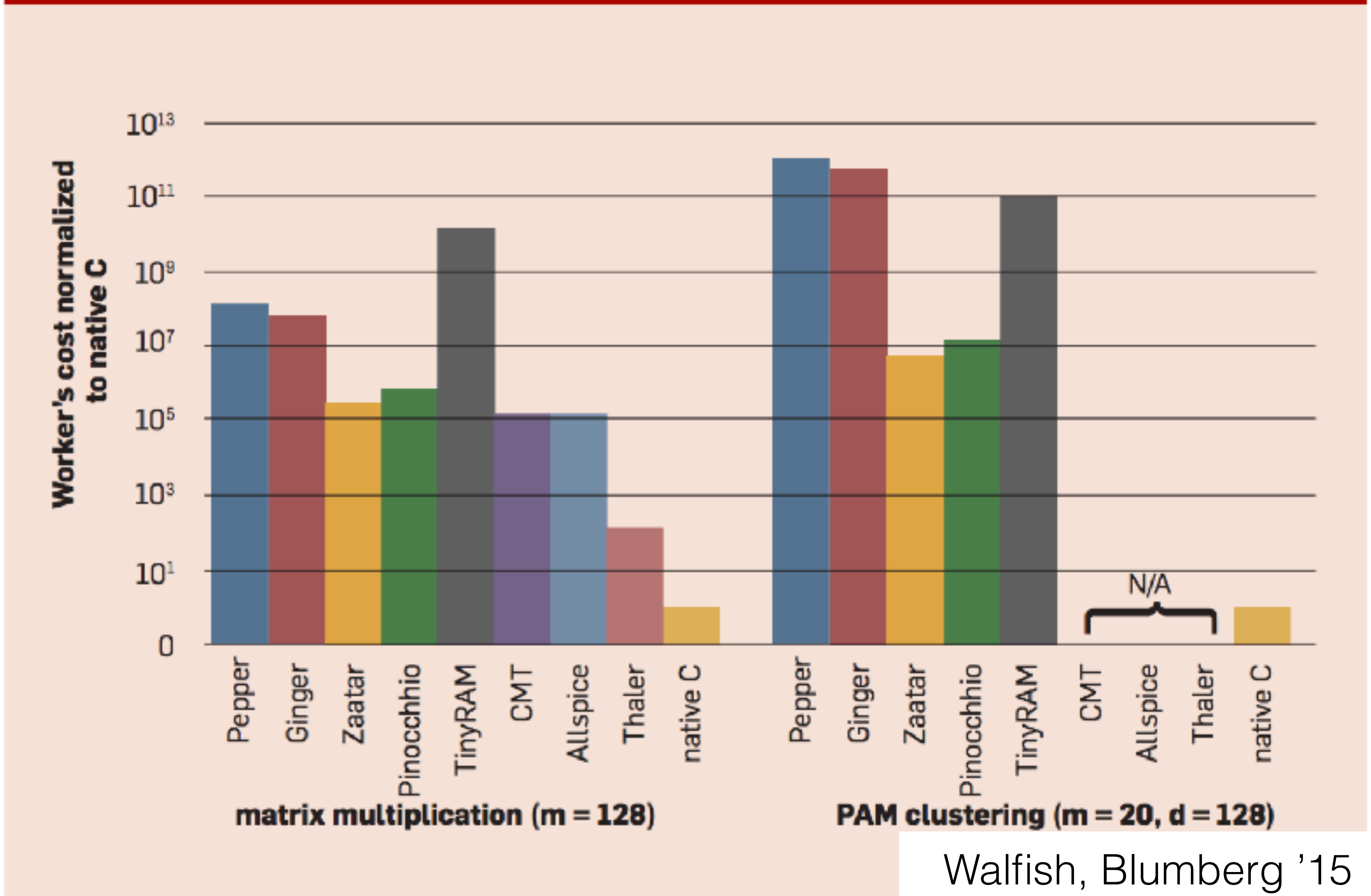
"M(x) = y" **, proof**

Complexity
<< evaluating M(x)

accept?

Verifiable Computation In Practice

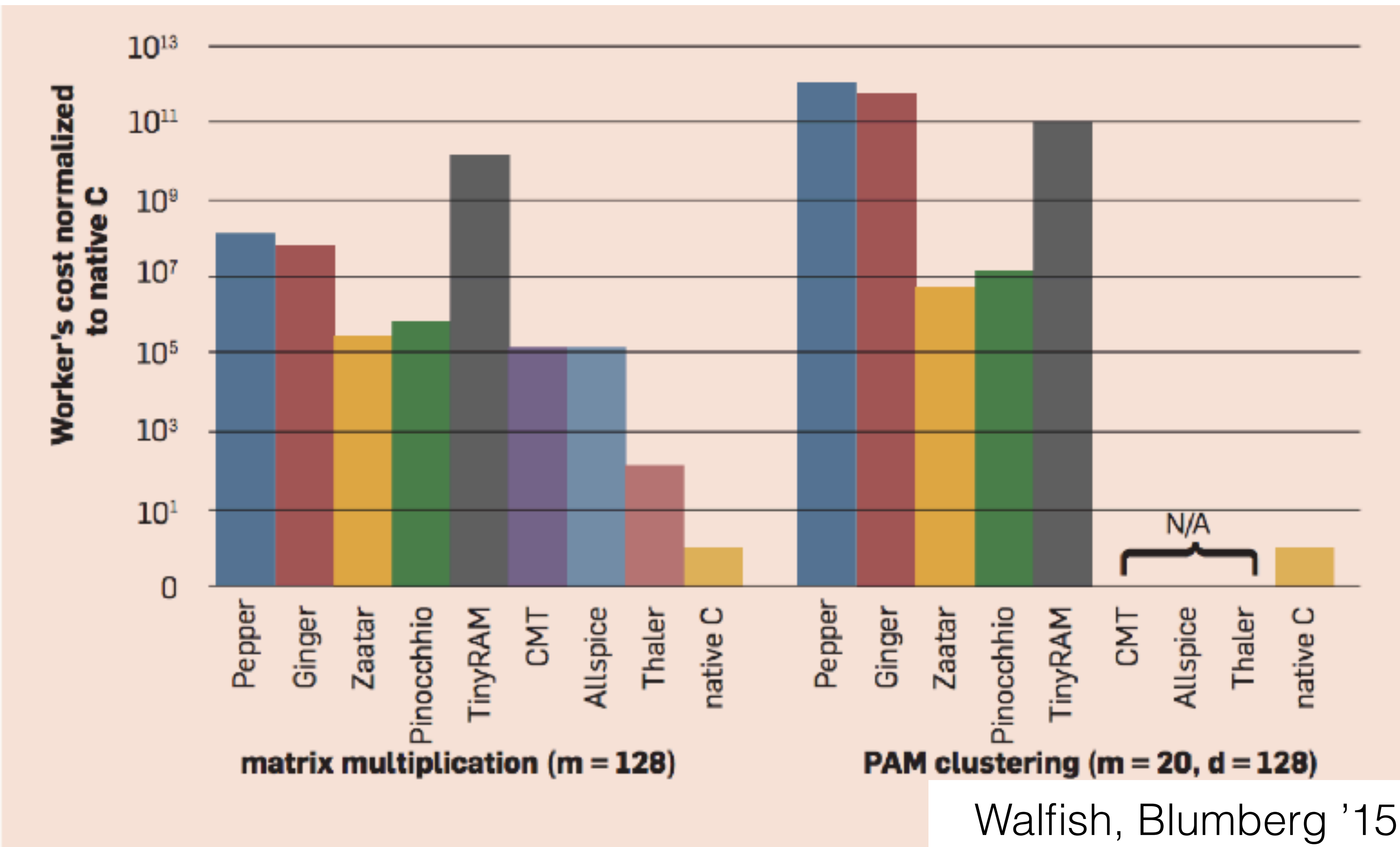
Figure 5. Prover overhead normalized to native execution cost for two computations. Prover overheads are generally enormous.



Walfish, Blumberg '15

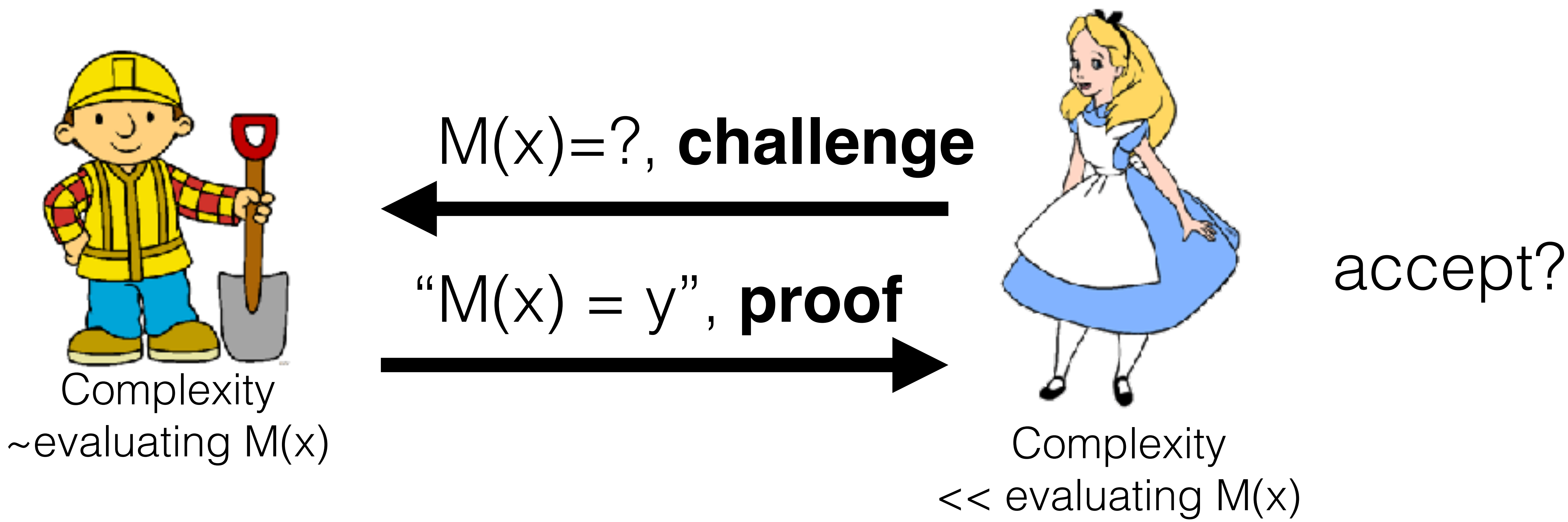
Verifiable Computation In Practice

"An additional bottleneck is memory: the prover must materialize a transcript of a computation's execution."



Walfish, Blumberg '15

Verifiable Computation



Our focus:

- Prover efficiency
- Computational assumptions

Prior Work

Prior Work

	Model	Assumptions	Prover Time	Prover Space
No-Signaling PCP [KRR14, KP15, B**H**K16]	RAM	PIR	poly(T)	poly(T)

Prior Work

	Model	Assumptions	Prover Time	Prover Space
No-Signaling PCP [KRR14, KP15, B**H**K16]	RAM	PIR	$T^{60}?$	$T^{60}?$

Prior Work

	Model	Assumptions	Prover Time	Prover Space
No-Signaling PCP [KRR14, KP15, B**H**K16]	RAM	PIR	$T^3?$	$T^3?$

Prior Work

	Model	Assumptions	Prover Time	Prover Space
No-Signaling PCP [KRR14, KP15, B**H**K16]	RAM	PIR	T^3?	T^3?
SNARKs [BC12, BCCT12, …]	RAM	Non-Falsifiable	$T \cdot \text{poly}(\kappa)$	$S \cdot \text{poly}(\kappa)$
Succinct Garbling [GHRW14, KLW15, C**H**15, CCCLLZ15]	RAM	Obfuscation	$T \cdot \text{poly}(\kappa)$	$S \cdot \text{poly}(\kappa)$

Prior Work

	Model	Assumptions	Prover Time	Prover Space
No-Signaling PCP [KRR14, KP15, B**H**K16]	RAM	PIR	T^3?	T^3?
SNARKs [BC12, BCCT12, …]	RAM	Non-Falsifiable	$T \cdot \text{poly}(\kappa)$	$S \cdot \text{poly}(\kappa)$
Succinct Garbling [GHRW14, KLW15, C**H**15, CCCLLZ15]	RAM	Obfuscation	$T \cdot \text{poly}(\kappa)$	$S \cdot \text{poly}(\kappa)$
[this work]	TM	"Slightly" Homomorphic Encryption	$T \cdot \text{poly}(\kappa)$	$S + \text{poly}(\kappa)$

Prior Work

	Model	Assumptions	Prover Time	Prover Space
No-Signaling PCP [KRR14, KP15, B**H**K16]	RAM	PIR	T^3?	T^3?
SNARKs [BC12, BCCT12, …]	RAM	Non-Falsifiable	$T \cdot \mathsf{poly}(\kappa)$	$S \cdot \mathsf{poly}(\kappa)$
Succinct Garbling [GHRW14, KLW15, C**H**15, CCCLLZ15]	RAM	Obfuscation	$T \cdot \mathsf{poly}(\kappa)$	$S \cdot \mathsf{poly}(\kappa)$
[this work]	TM	"Slightly" Homomorphic Encryption	$T \cdot \mathsf{poly}(\kappa)$	$S + \mathsf{poly}(\kappa)$

Extends to (cache-efficient) RAM

Prior Work

	Model	Assumptions	Prover Time	Prover Space
No-Signaling PCP [KRR14, KP15, B**H**K16]	RAM	PIR	$T^3?$	$T^3?$
SNARKs [BC12, BCCT12, …]	RAM	Non-Falsifiable	$T \cdot \text{poly}(\kappa)$	$S \cdot \text{poly}(\kappa)$
Succinct Garbling [GHRW14, KLW15, C**H**15, CCCLLZ15]	RAM	Obfuscation	$T \cdot \text{poly}(\kappa)$	$S \cdot \text{poly}(\kappa)$
[this work]	TM	"Slightly" Homomorphic Encryption	$T \cdot \text{poly}(\kappa)$	$S + \text{poly}(\kappa)$

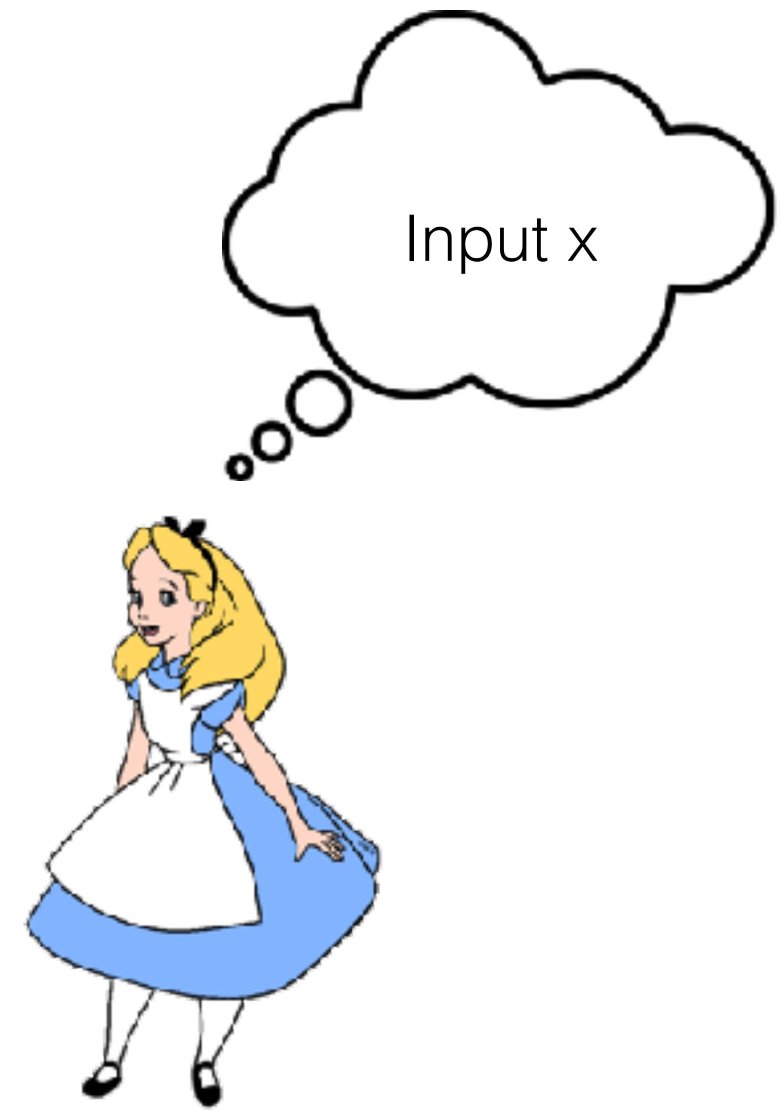
Extends to (cache-efficient) RAM

Probabilistically Checkable Proofs

Probabilistically Checkable Proofs

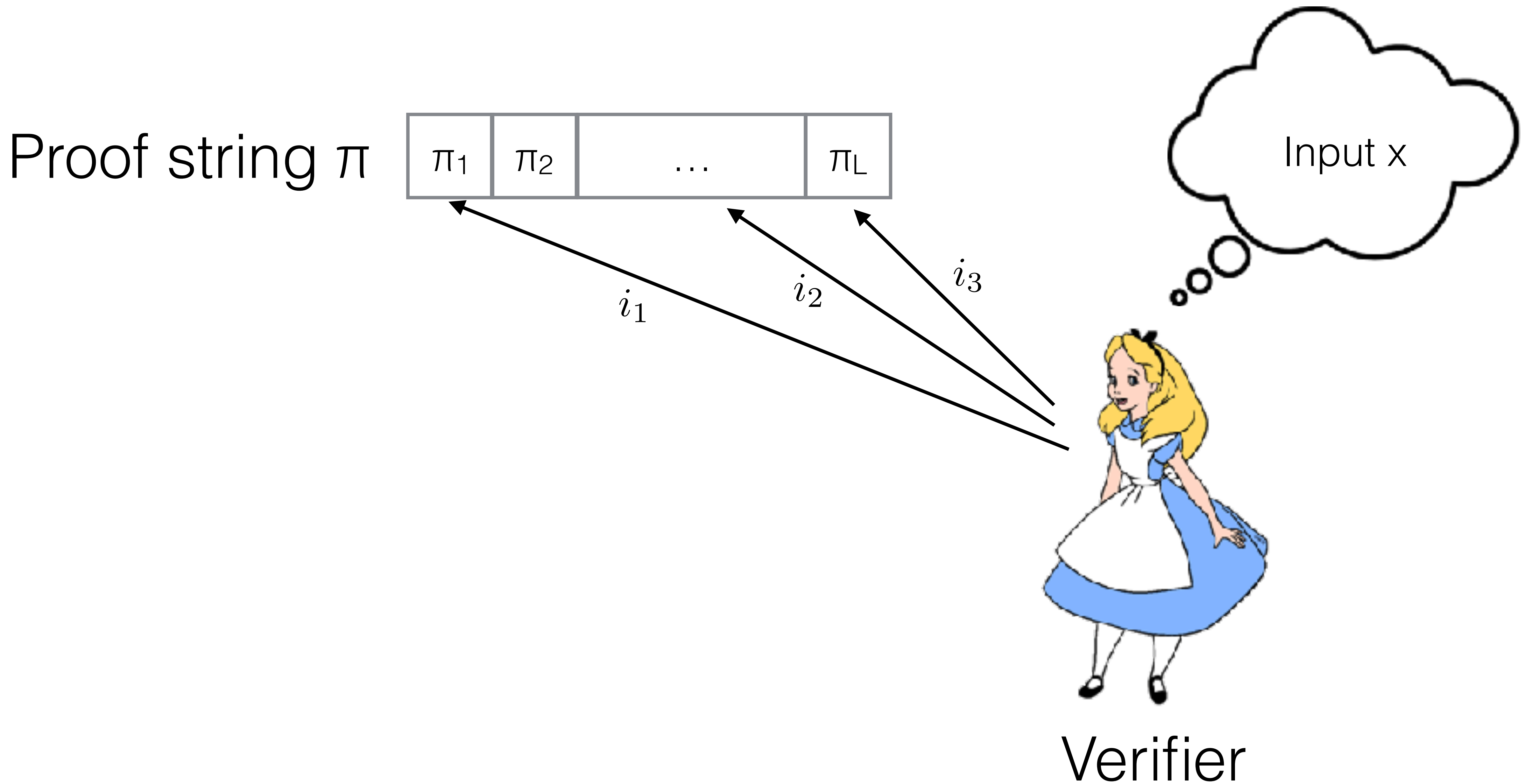
Proof string π

π_1	π_2	…	π_L



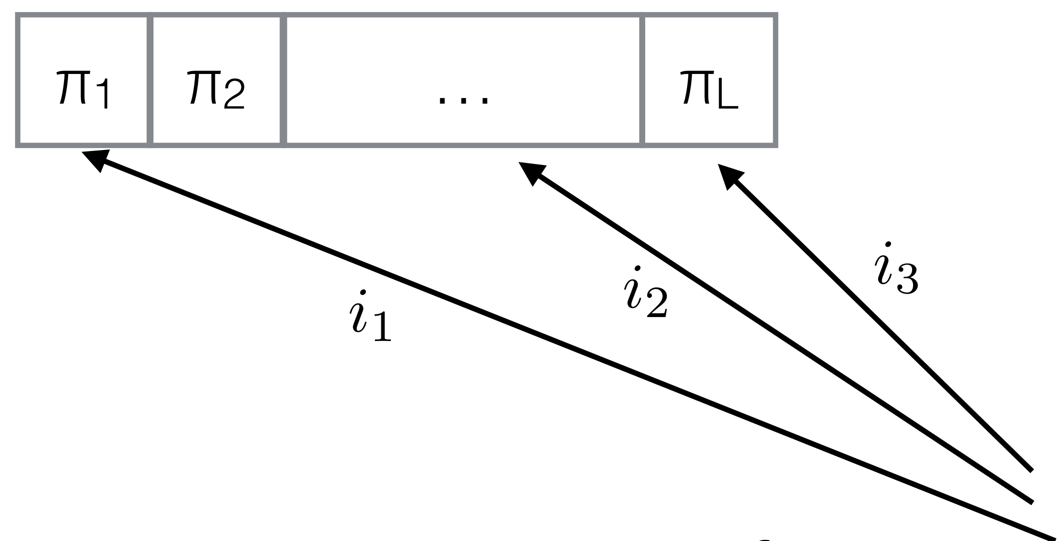
Verifier

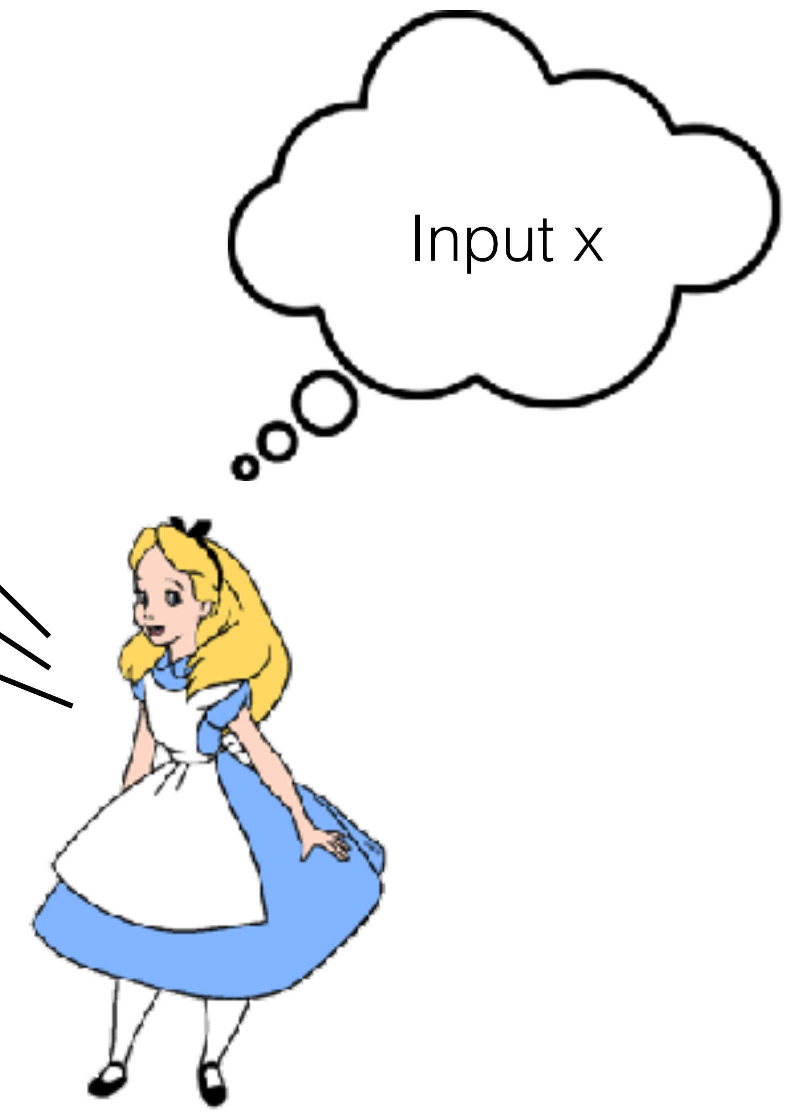
Probabilistically Checkable Proofs



Probabilistically Checkable Proofs

Proof string π



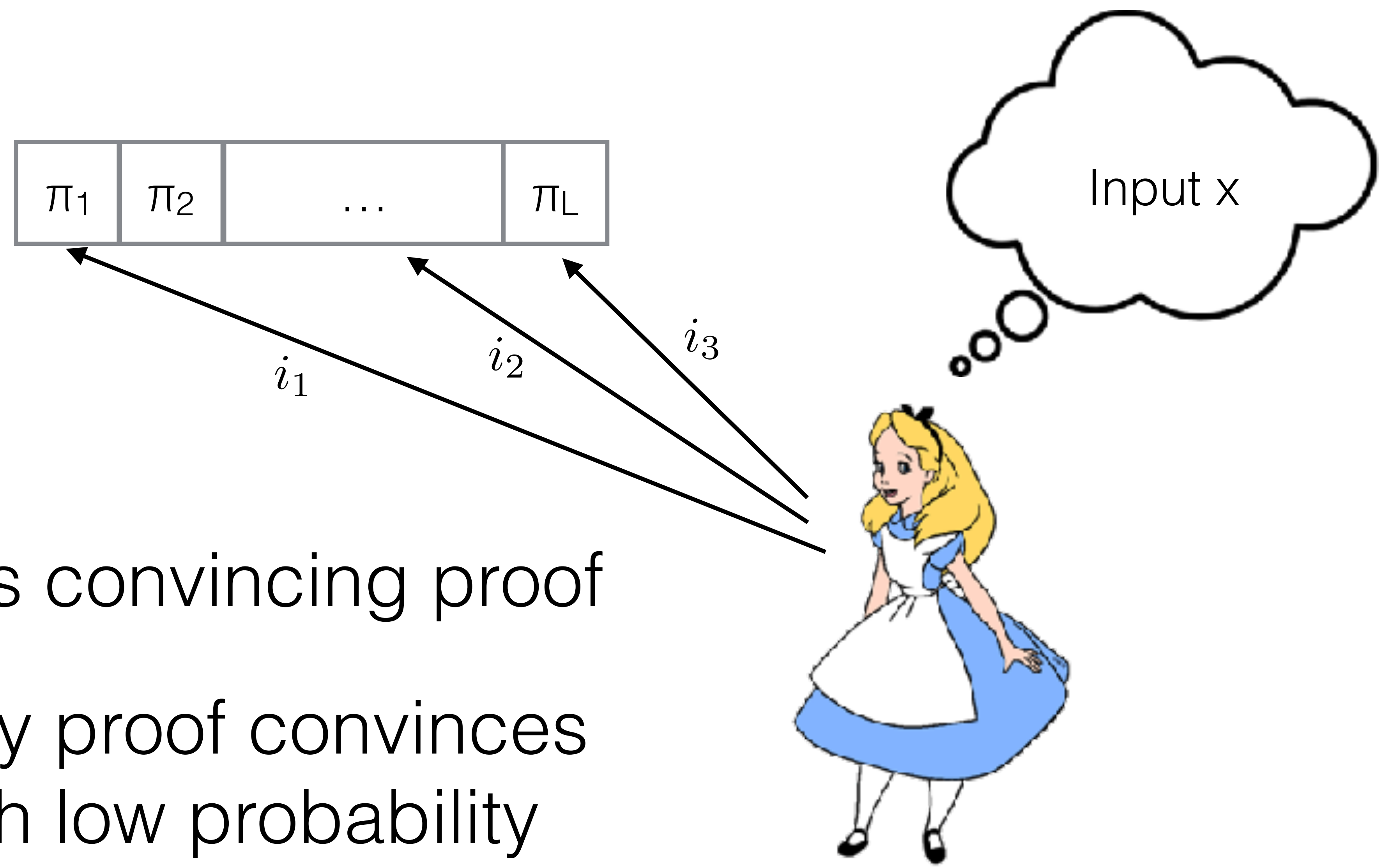


Verifier

$x \in \mathcal{L} \implies$ exists convincing proof

Probabilistically Checkable Proofs

Proof string π



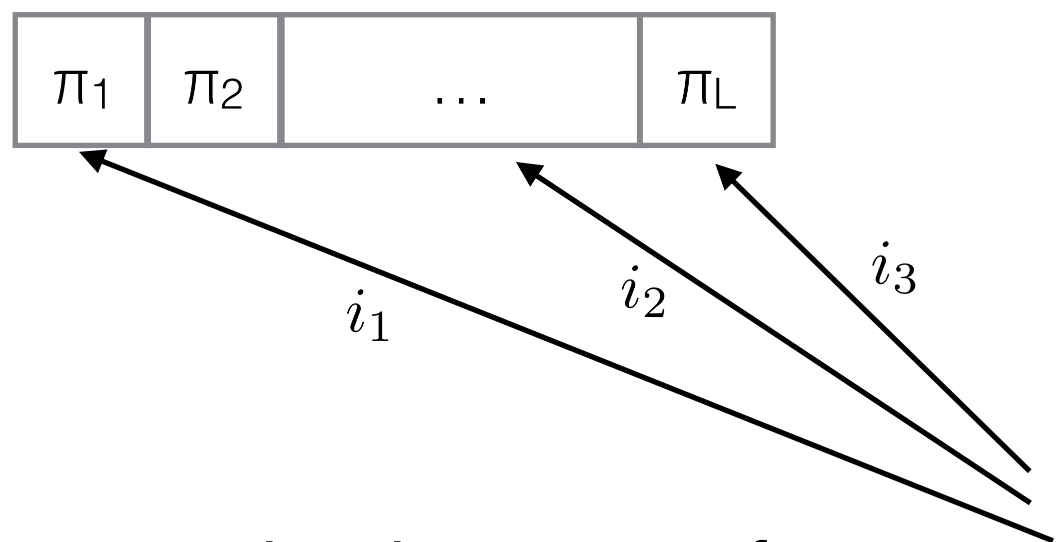
Verifier

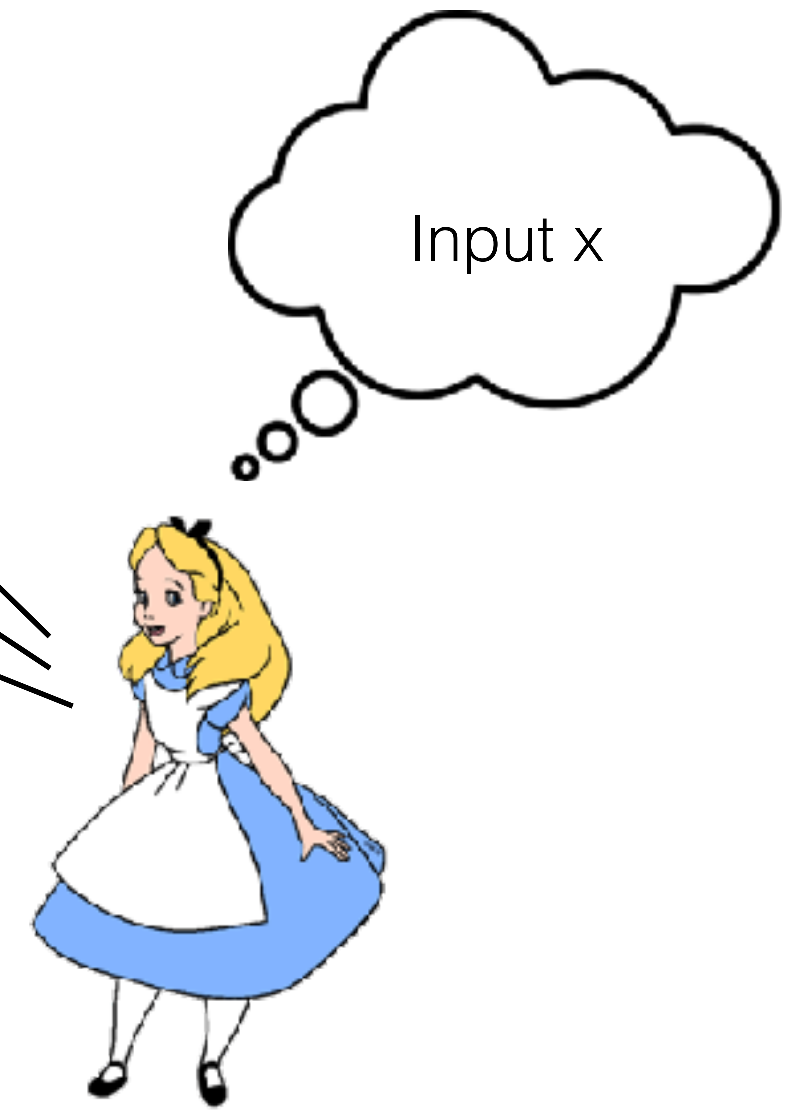
$x \in \mathcal{L} \implies$ exists convincing proof

$x \notin \mathcal{L} \implies$ every proof convinces with low probability

Probabilistically Checkable Proofs

Proof string π





Verifier

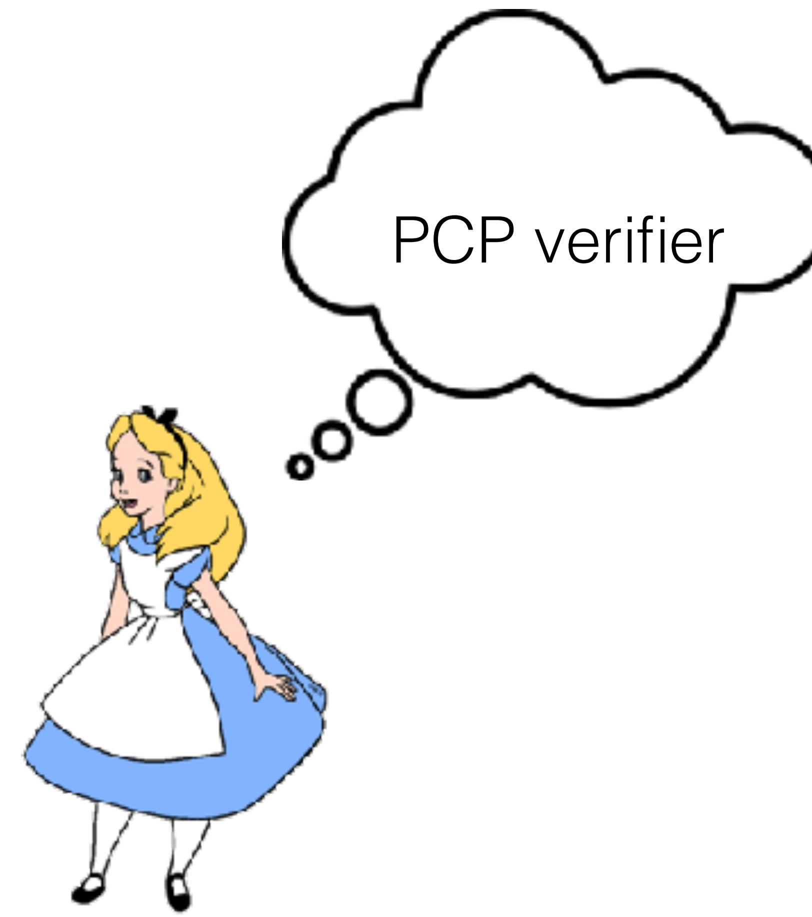
$x \in \mathcal{L} \implies$ exists convincing proof

$x \notin \mathcal{L} \implies$ every proof convinces with low probability

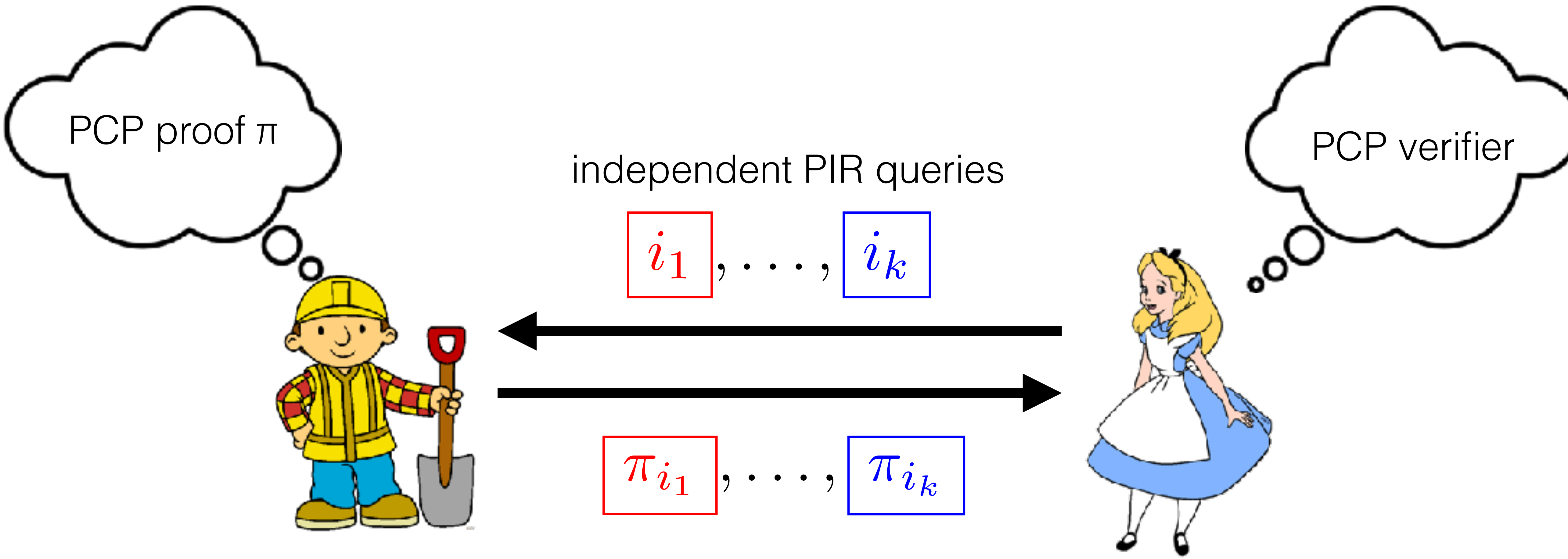
Not a standard-model delegation scheme

PCP-based Delegation

PCP-based Delegation

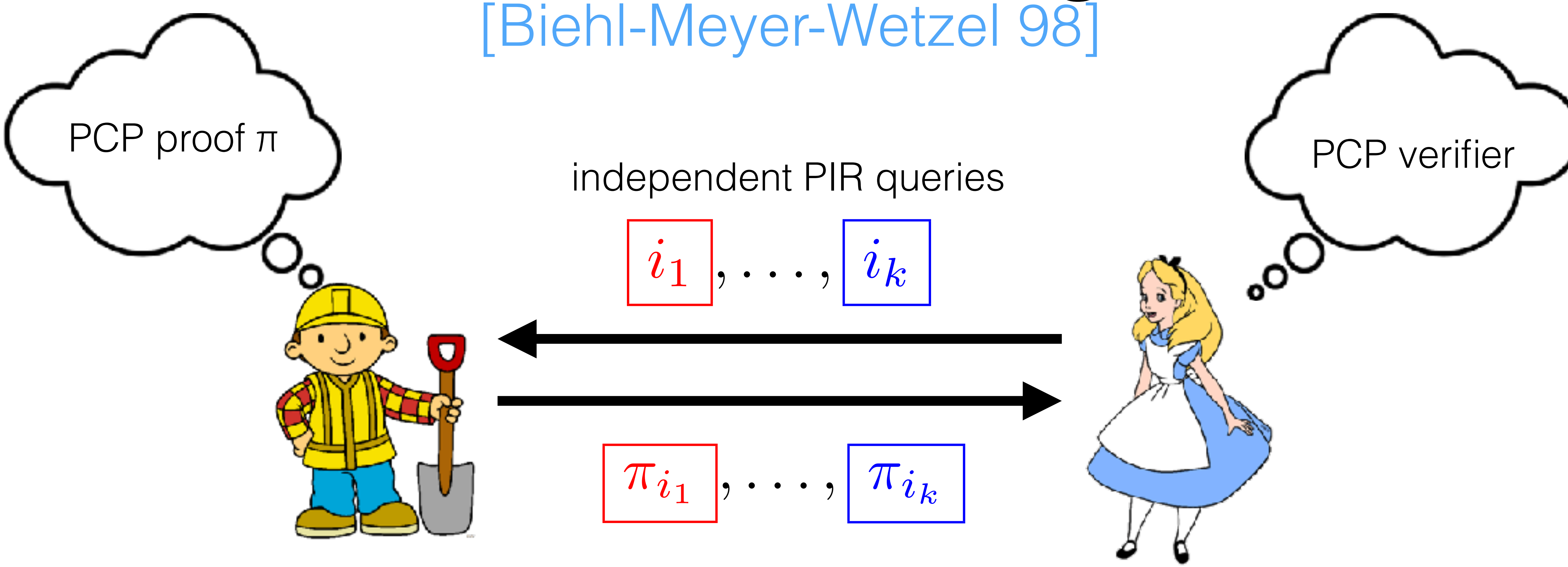


PCP-based Delegation



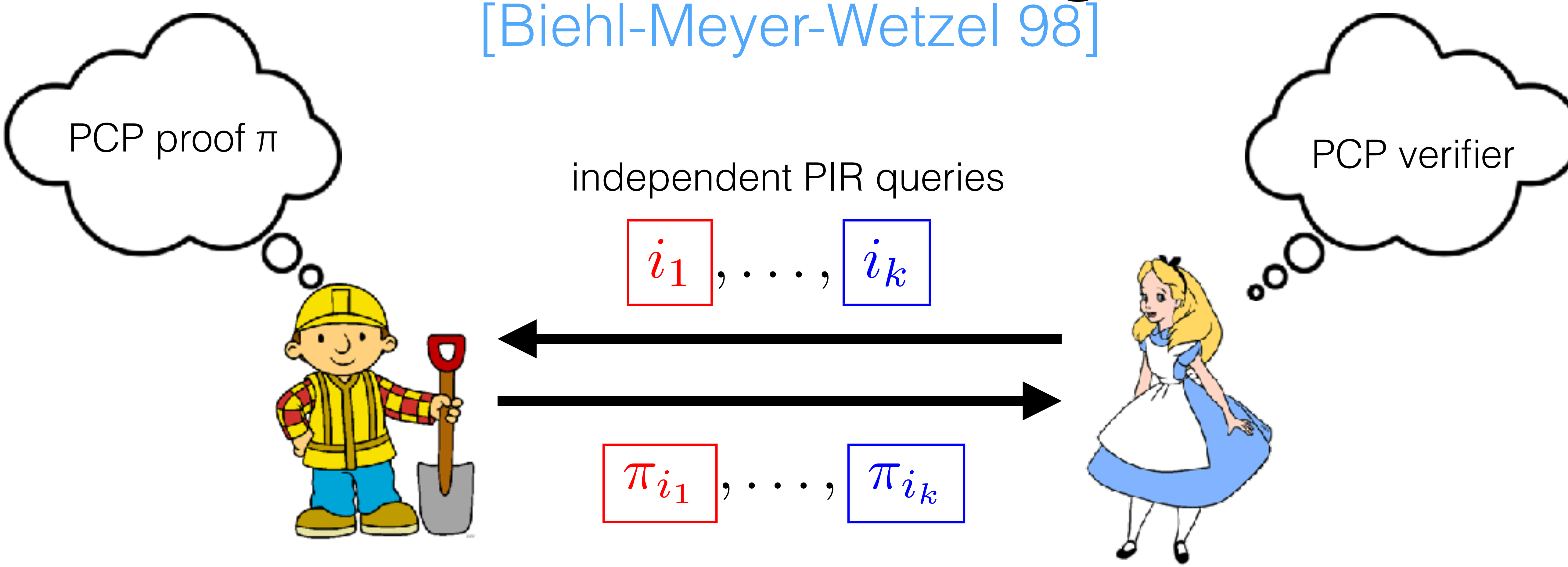
PCP-based Delegation

[Biehl-Meyer-Wetzel 98]



PCP-based Delegation

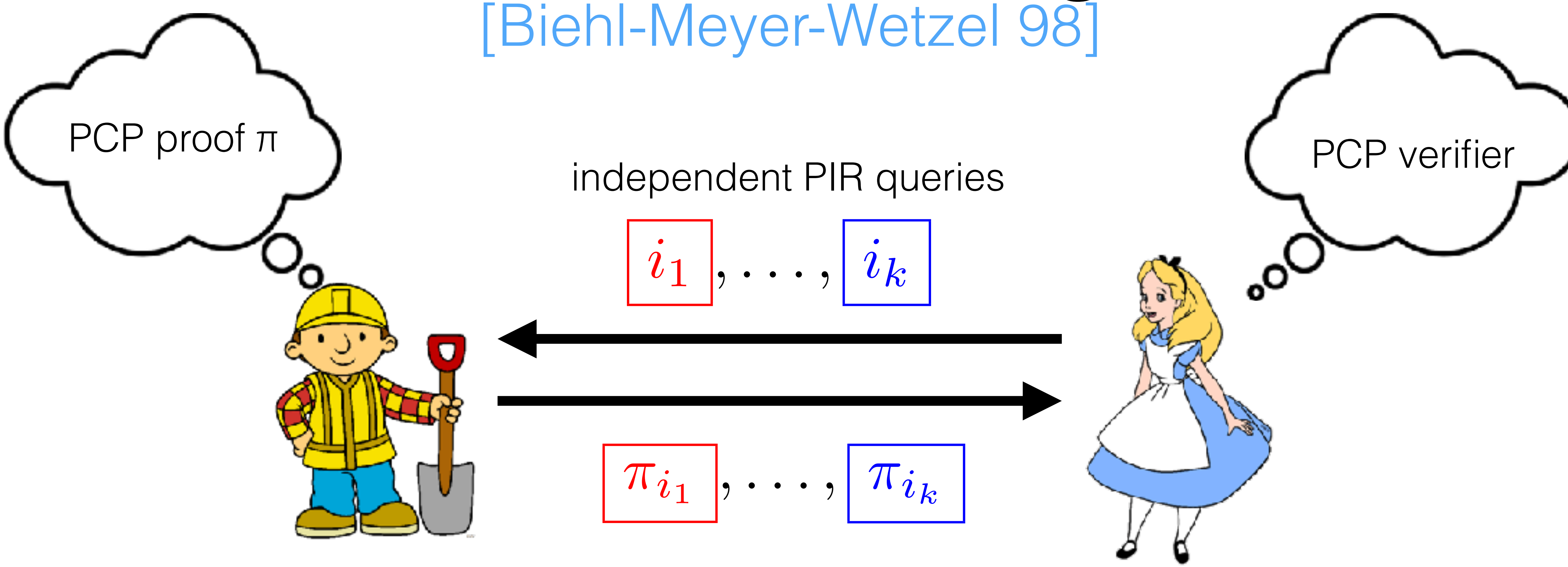
[Biehl-Meyer-Wetzel 98]



- Not sound in general
  [Dwork-Langberg-Naor-Nissim-Reingold 01]

PCP-based Delegation

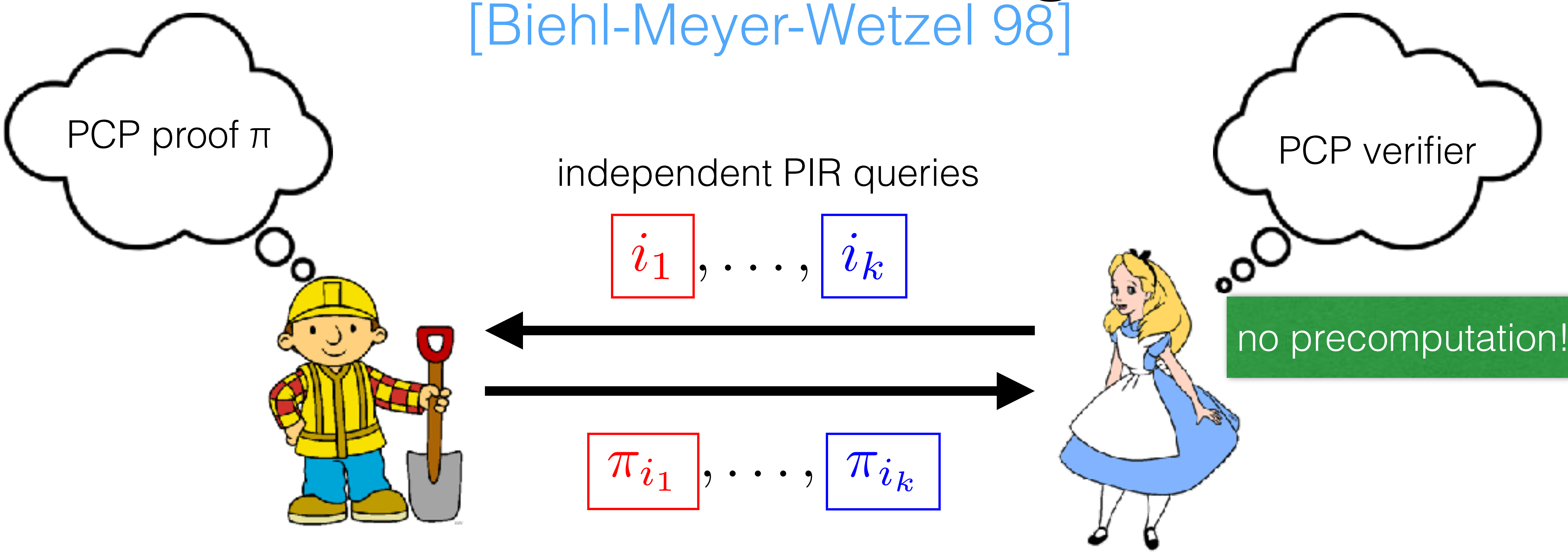
[Biehl-Meyer-Wetzel 98]



- Not sound in general [Dwork-Langberg-Naor-Nissim-Reingold 01]
- Sound if the PCP is ***no-signaling*** sound [Kalai-Raz-Rothblum 14]

PCP-based Delegation

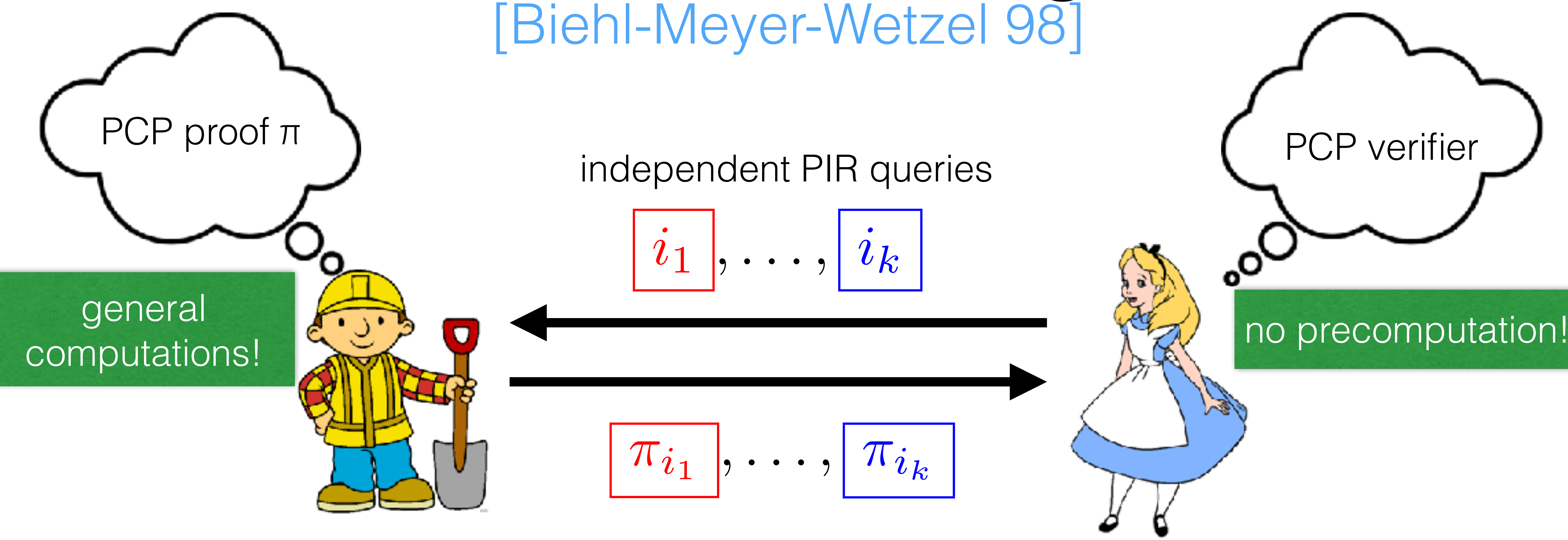
[Biehl-Meyer-Wetzel 98]



- Not sound in general [Dwork-Langberg-Naor-Nissim-Reingold 01]
- Sound if the PCP is ***no-signaling*** sound [Kalai-Raz-Rothblum 14]

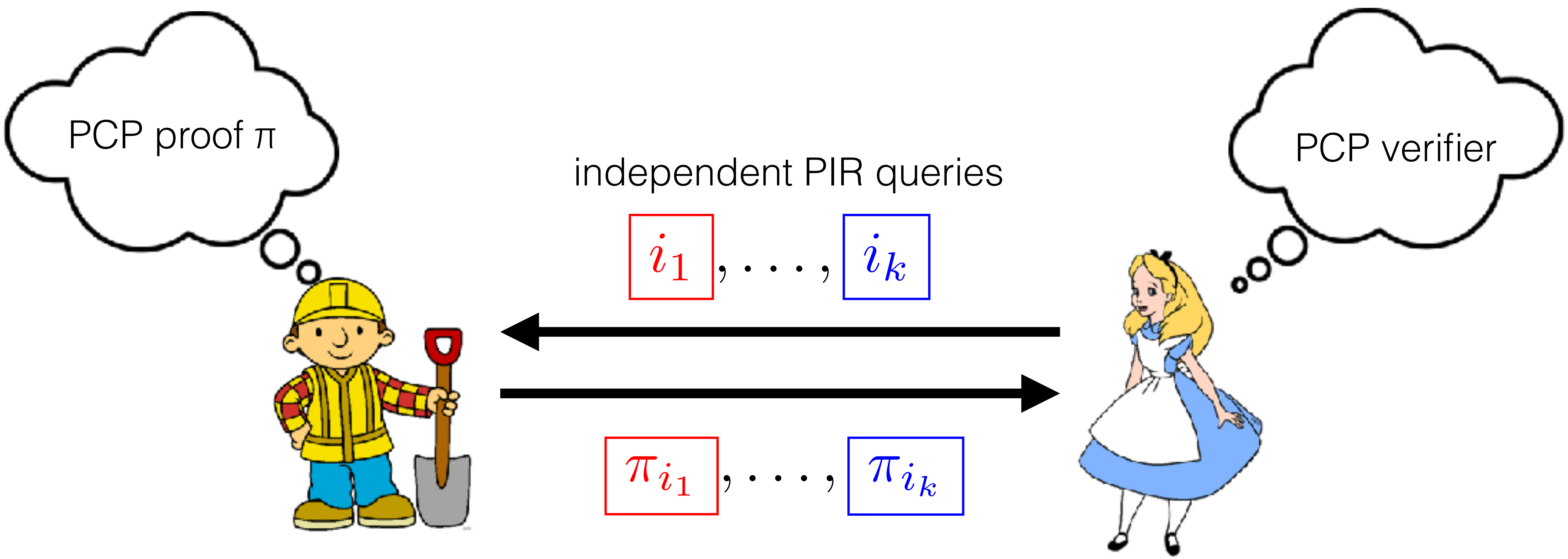
PCP-based Delegation

[Biehl-Meyer-Wetzel 98]

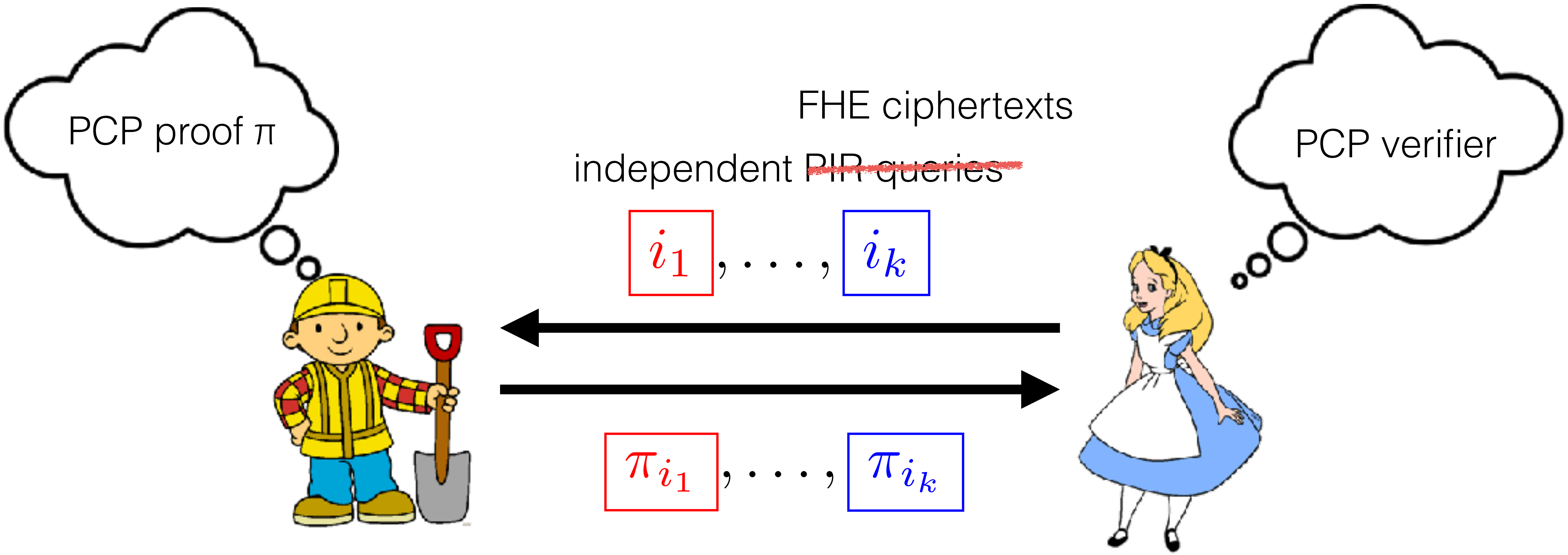


- Not sound in general [Dwork-Langberg-Naor-Nissim-Reingold 01]
- Sound if the PCP is ***no-signaling*** sound [Kalai-Raz-Rothblum 14]

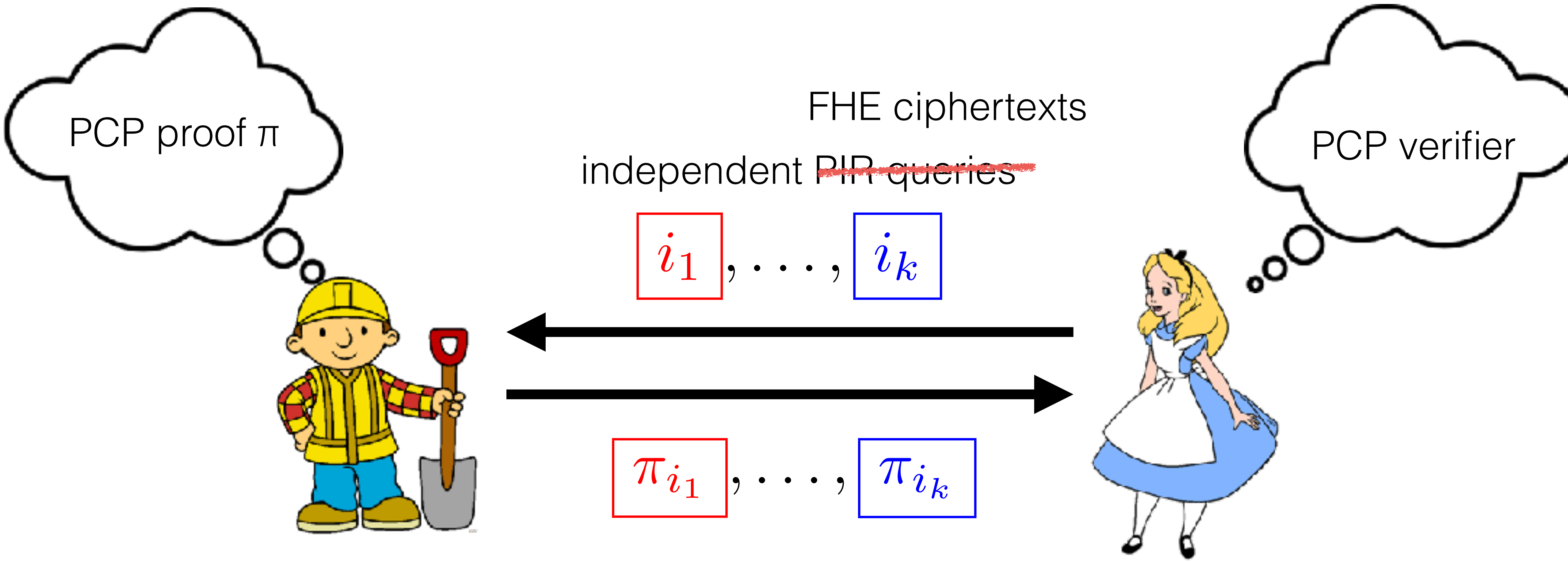
Observation 0



Observation 0

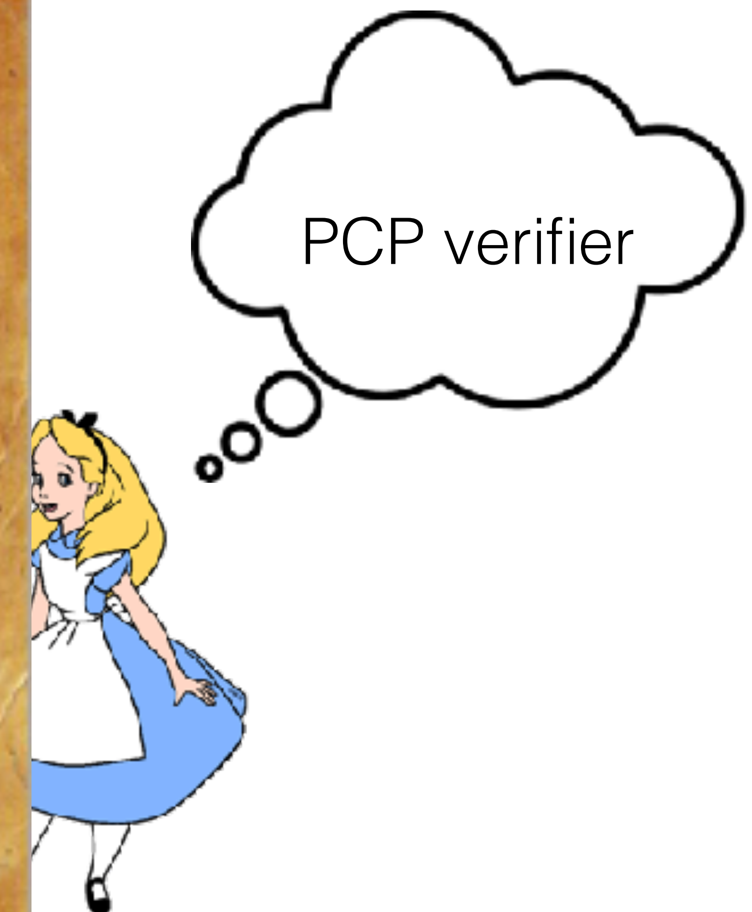


Observation 0



- If PIR = FHE, just need efficient “random-access” to PCP.

Observation 0



- If PIR = FI … dom-
  access” t…

Our Technical Contributions

Our Technical Contributions

1. Simpler and direct NS-PCP(essentially BFLS) for any language $\mathcal{L} \in \mathsf{TISP}(T, S)$

Our Technical Contributions

Remove major component of KRR, namely "augmented circuit"

1. Simpler and direct NS-PCP(essentially BFLS) for any language $\mathcal{L} \in \mathsf{TISP}(T, S)$

Our Technical Contributions

Remove major component of KRR, namely "augmented circuit"

1. Simpler and direct NS-PCP(essentially BFLS) for any language $\mathcal{L} \in \mathsf{TISP}(T, S)$

2. Super-efficient prover: Any symbol computable in time: $\tilde{O}(T)$ space: $S + \mathsf{polylog}(T)$

Our Technical Contributions

Remove major component of KRR, namely "augmented circuit"

1 Simpler and direct NS-PCP(essentially BFLS) for any language $\mathcal{L} \in \mathsf{TISP}(T, S)$

2 Super-efficient prover: Any symbol computable in time: $\tilde{O}(T)$ space: $S + \mathsf{polylog}(T)$

2' Limited efficiency loss under FHE

Our Technical Contributions

Remove major component of KRR, namely "augmented circuit"

1 Simpler and direct NS-PCP(essentially BFLS) for any language $\mathcal{L} \in \mathsf{TISP}(T, S)$

2 Super-efficient prover: Any symbol computable in

time: $\tilde{O}(T)$ space: $S + \mathsf{polylog}(T)$

2' Limited efficiency loss under FHE

time: $T \cdot \mathsf{poly}(\lambda)$

Our Technical Contributions

Remove major component of KRR, namely "augmented circuit"

1 Simpler and direct NS-PCP(essentially BFLS) for any language $\mathcal{L} \in \mathsf{TISP}(T, S)$

2 Super-efficient prover: Any symbol computable in time: $\tilde{O}(T)$ space: $S + \mathsf{polylog}(T)$

2' Limited efficiency loss under FHE time: $T \cdot \mathsf{poly}(\lambda)$ space: $S + \mathsf{poly}(\lambda)$

Our Technical Contributions

Remove major component of KRR, namely “augmented circuit”

1 Simpler and direct NS-PCP(essentially BFLS) for any language $\mathcal{L} \in \mathsf{TISP}(T, S)$

2 Super-efficient prover: Any

BFLS already known to be complexity-preserving? [BC12, BTVW14]

in

time: $\tilde{O}(T)$ space: S

2’ Limited efficiency loss under FHE

time: $T \cdot \mathsf{poly}(\lambda)$ space: $S + \mathsf{poly}(\lambda)$

Our Technical Contributions

Remove major component of KRR, namely “augmented circuit”

1 Simpler and direct NS-PCP(essentially BFLS) for any language $\mathcal{L} \in \mathsf{TISP}(T, S)$

for ***deterministic*** computations

2 Super-efficient prover: Any ... in

time: $\tilde{O}(T)$ space: S ...

BFLS already known to be complexity-preserving?
[BC12, BTVW14]

2’ Limited efficiency loss under FHE

time: $T \cdot \mathsf{poly}(\lambda)$ space: $S + \mathsf{poly}(\lambda)$

Our Technical Contributions

Remove major component of KRR, namely "augmented circuit"

1 Simpler and direct NS-PCP(essentially BFLS) for any language $\mathcal{L} \in \mathsf{TISP}(T, S)$

for ***deterministic*** computations

BFLS already known to be complexity-preserving? [BC12, BTVW14]

2 Super-efficient prover: Any ... in

time: $\tilde{O}(T)$ space: S ...

with ***non-deterministic*** computations

2' Limited efficiency loss under FHE

time: $T \cdot \mathsf{poly}(\lambda)$ space: $S + \mathsf{poly}(\lambda)$

Talk Outline

Talk Outline

NOT proving NS-soundness of BFLS for deterministic circuits

Talk Outline

NOT proving NS-soundness of BFLS for deterministic circuits

Part 1: Turing / RAM Machines ➡ (non-succinct) *deterministic* circuits

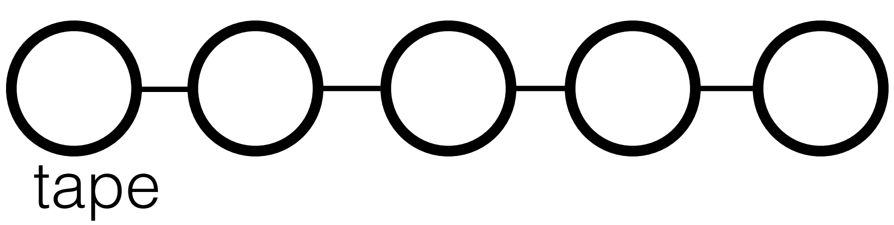
Talk Outline

NOT proving NS-soundness of BFLS for deterministic circuits

Part 1: Turing / RAM Machines ➡ (non-succinct) *deterministic* circuits

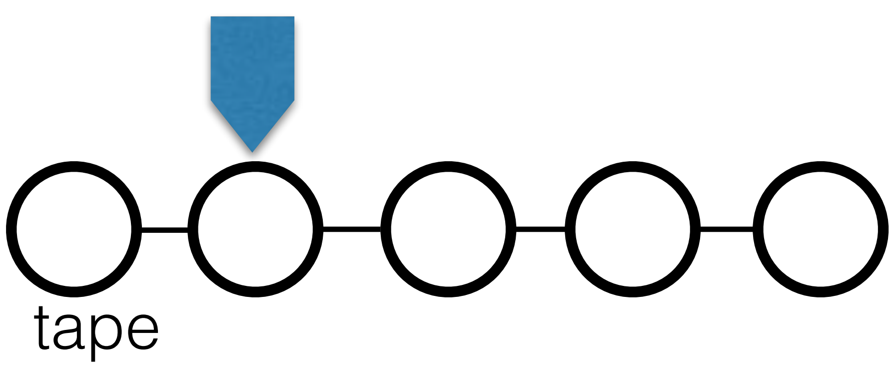
Part 2: (part of) BFLS prover efficiency despite non-succinctness.

Turing Machines as Circuits



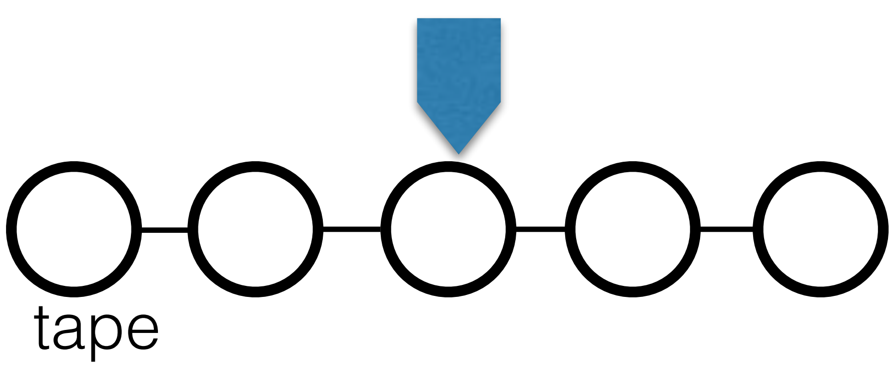
TM Configuration

Turing Machines as Circuits



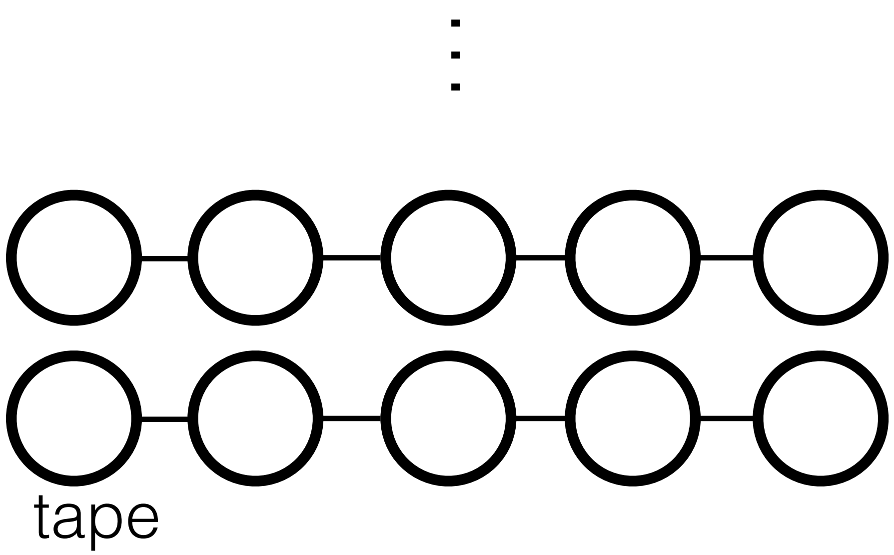
TM Configuration

Turing Machines as Circuits



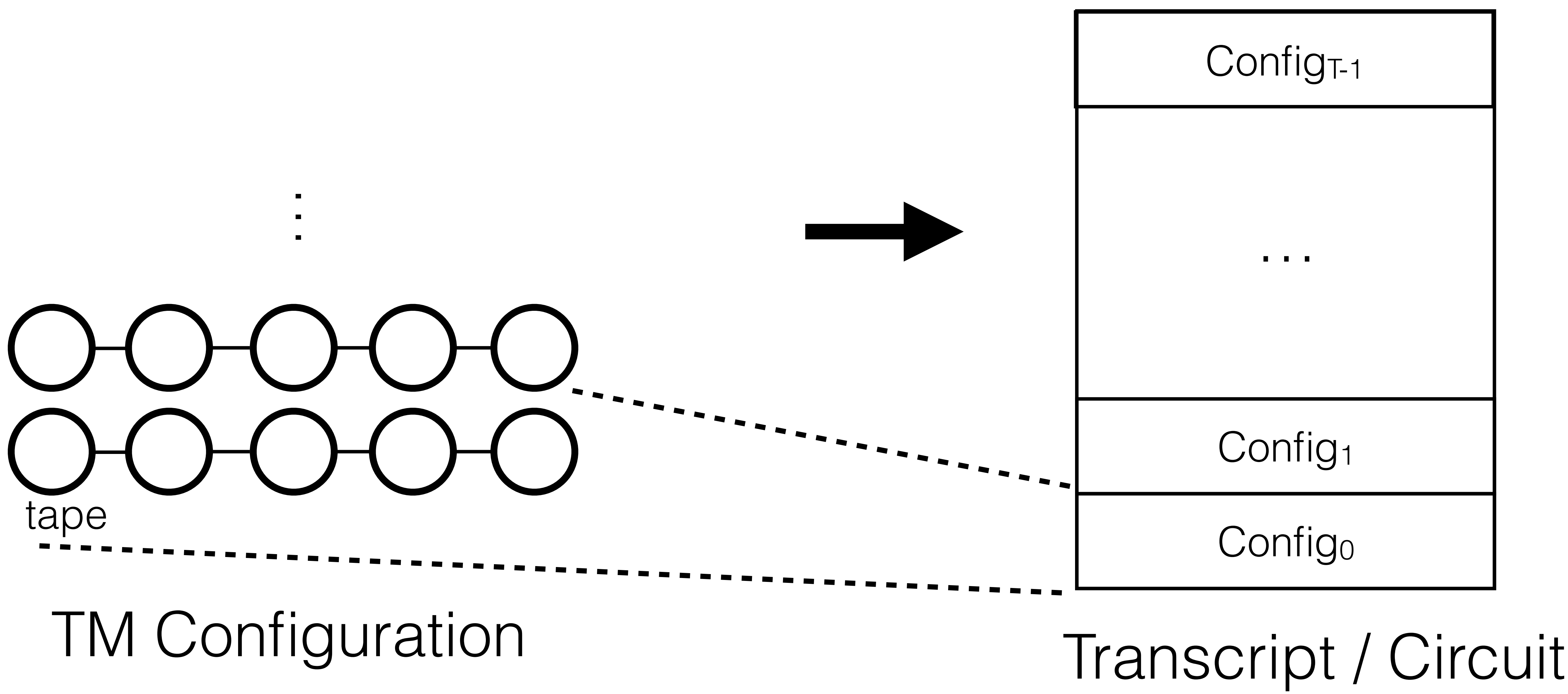
TM Configuration

Turing Machines as Circuits

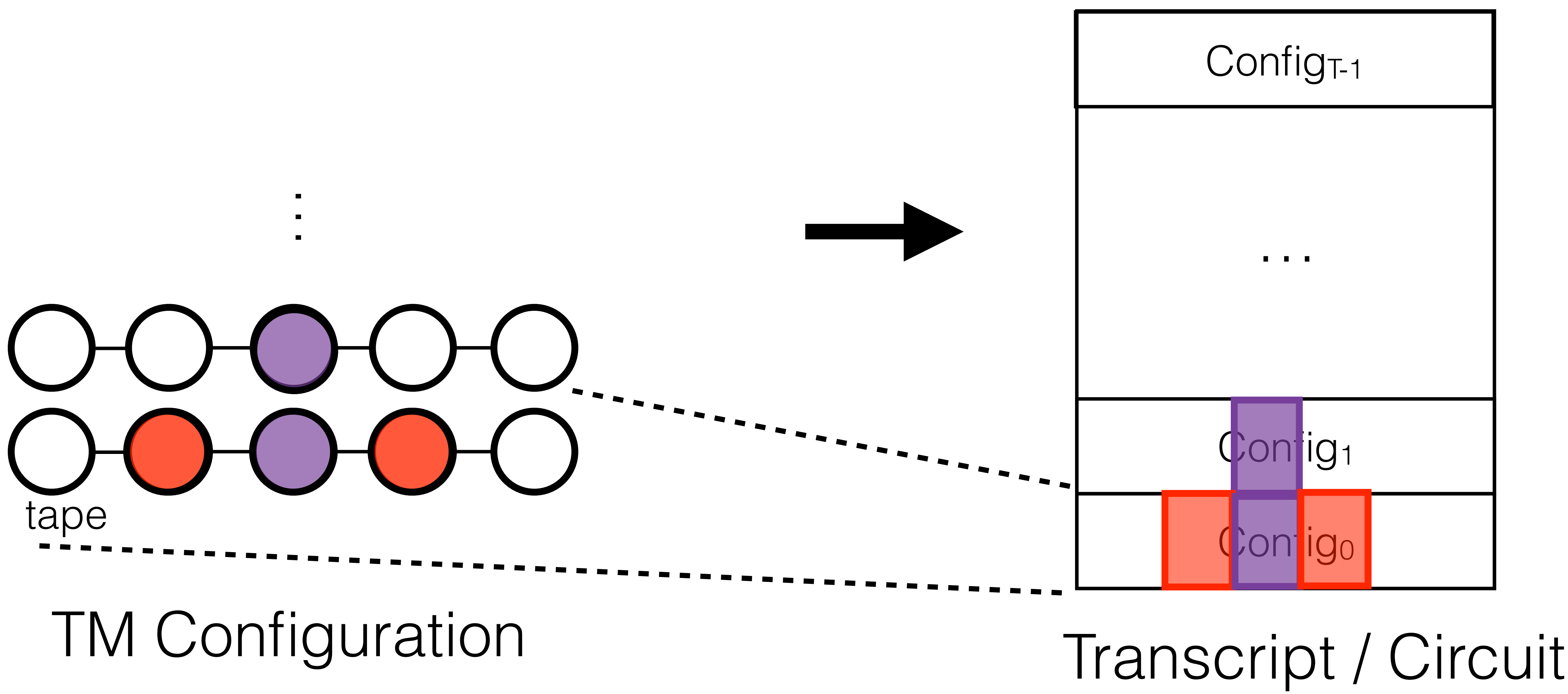


TM Configuration

Turing Machines as Circuits

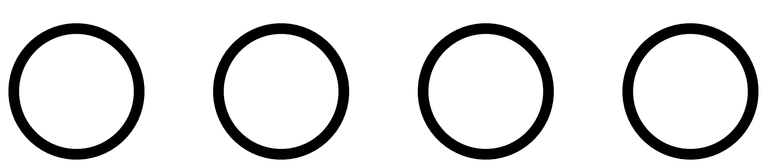


Turing Machines as Circuits



RAM Machines as Circuits

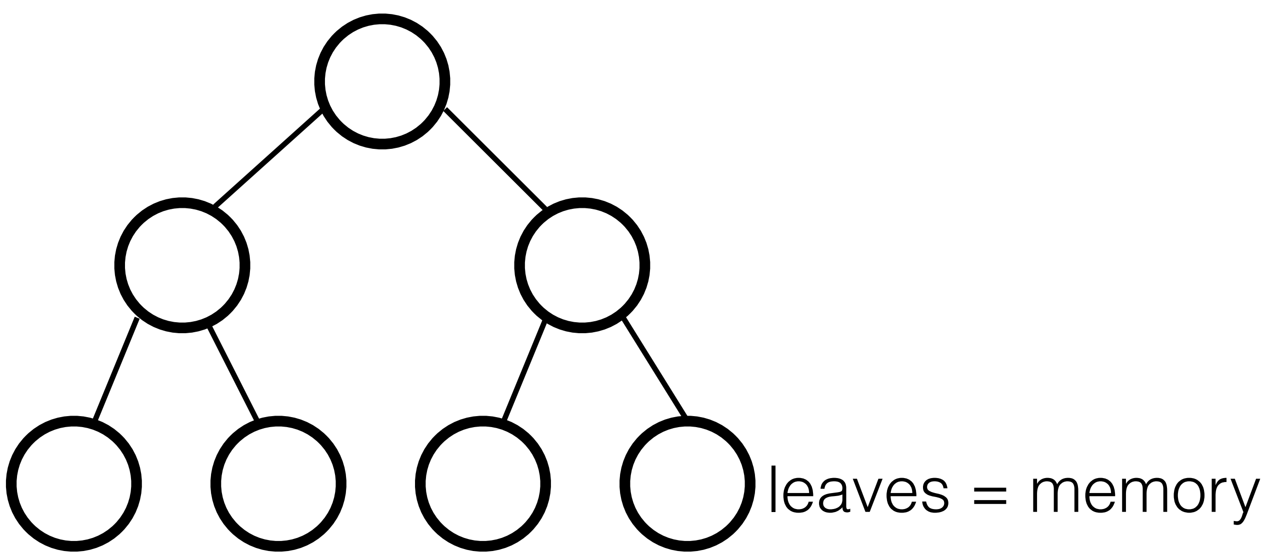
Configuration:



RAM Machines as Circuits

Configuration:

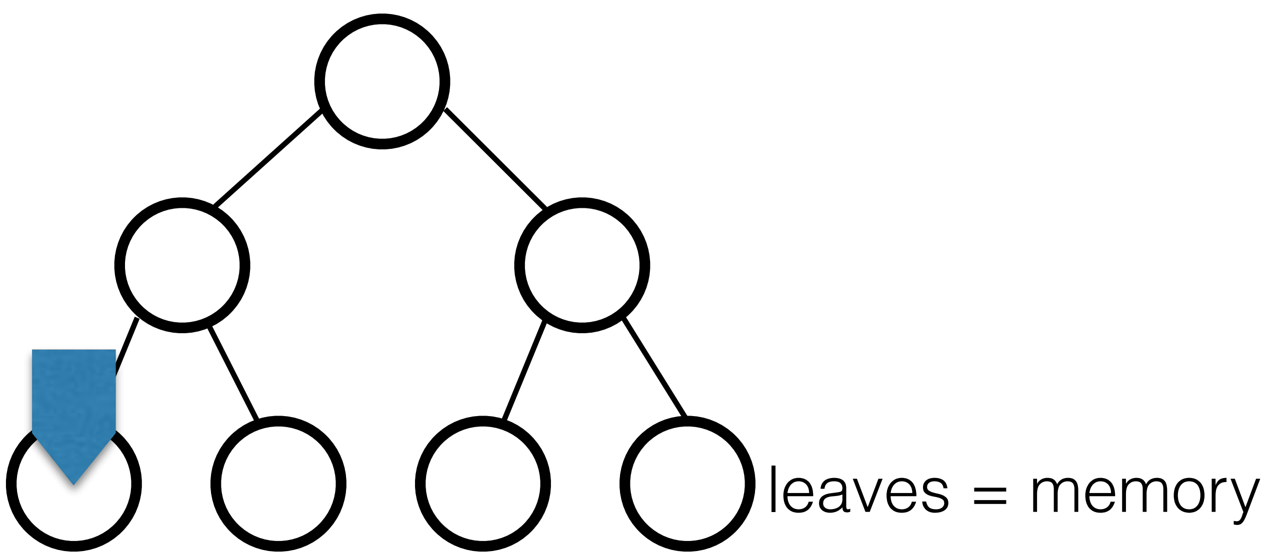
(diameter log S)



RAM Machines as Circuits

Configuration:

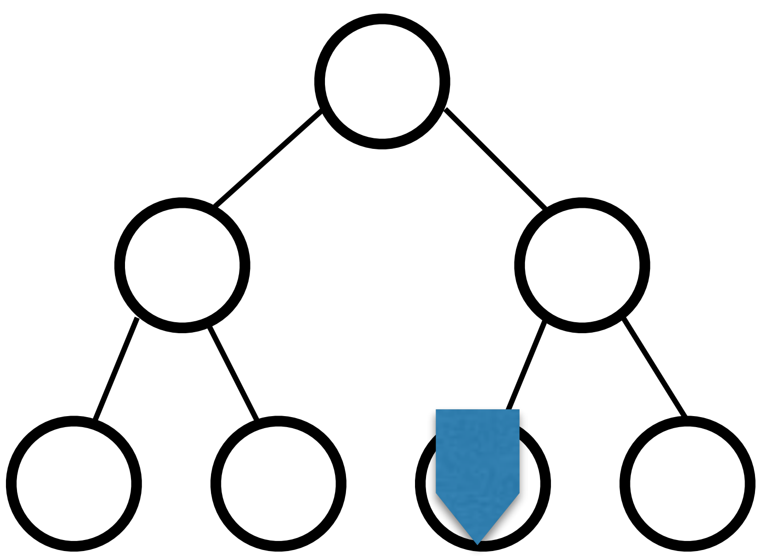
(diameter log S)



RAM Machines as Circuits

Configuration:

(diameter log S)

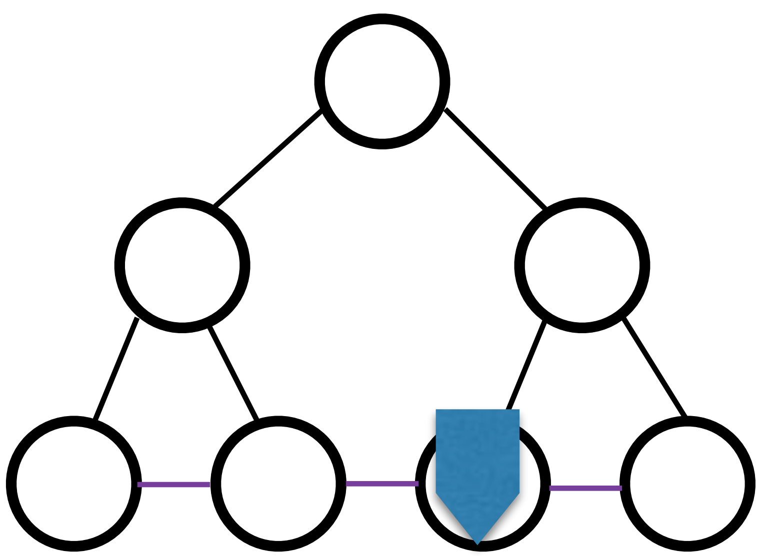


leaves = memory

RAM Machines as Circuits

Configuration:

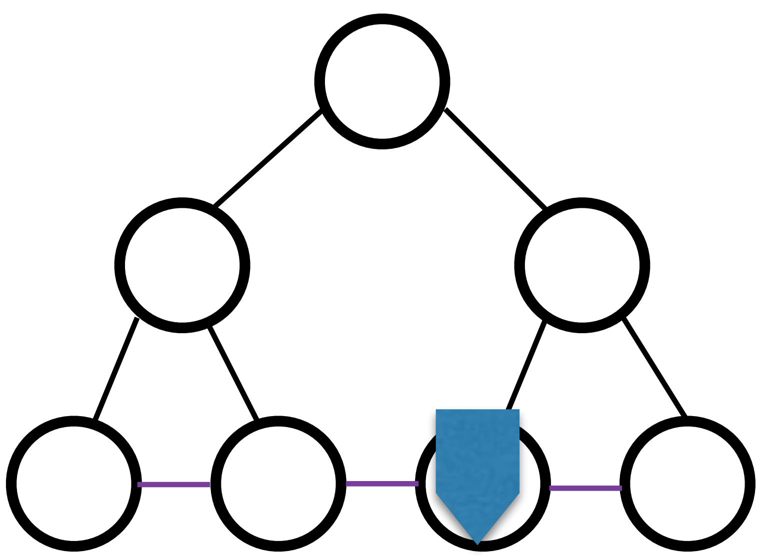
(diameter log S)



RAM Machines as Circuits

Configuration:

(diameter log S)



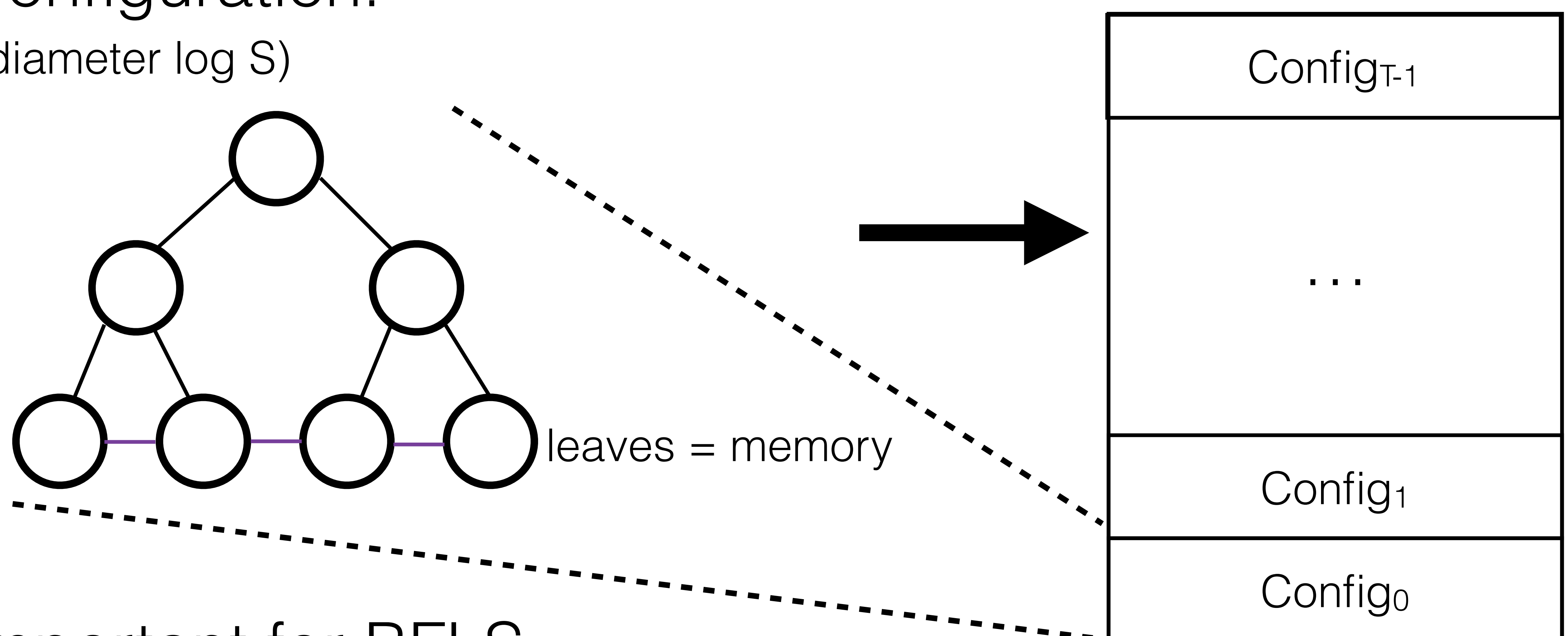
Important for BFLS:
Graph is “regular”!

RAM Machines as Circuits

Configuration:

(diameter log S)

Transcript / Circuit:



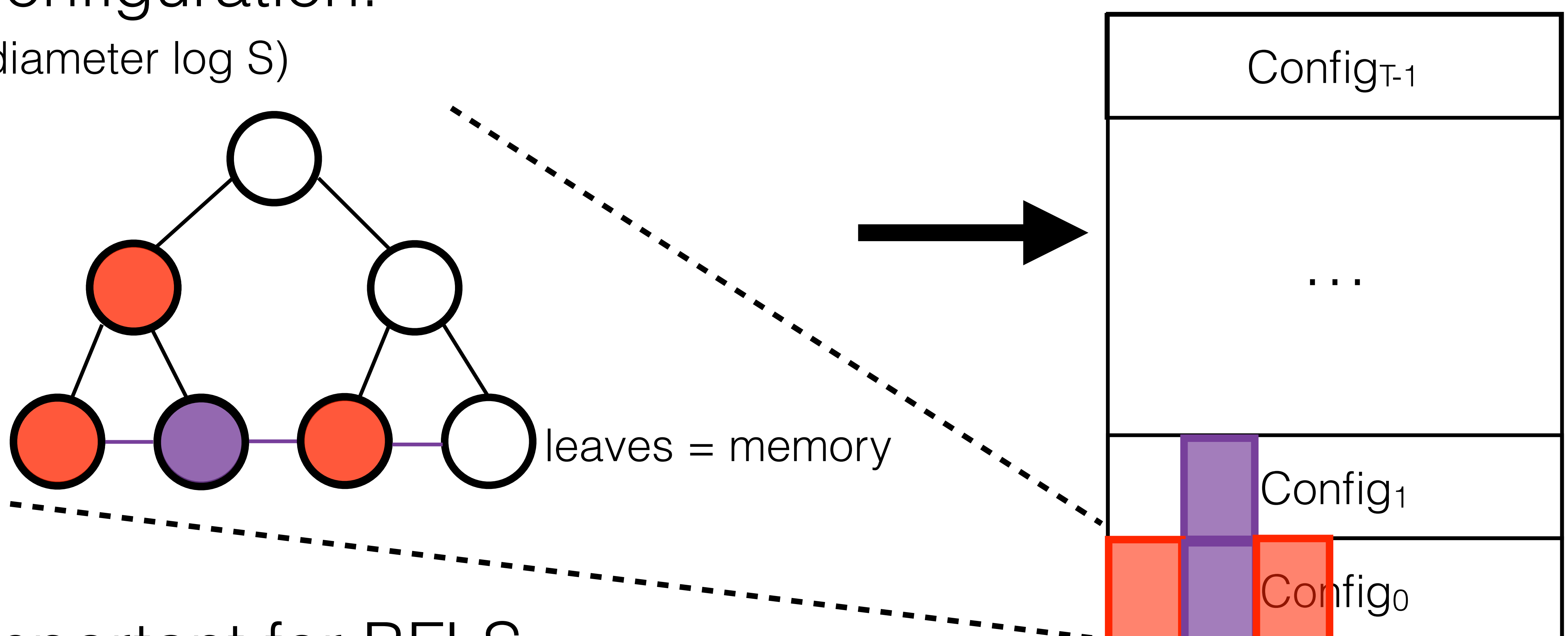
Important for BFLS:
Graph is “regular”!

RAM Machines as Circuits

Transcript / Circuit:

Configuration:

(diameter log S)



Important for BFLS:
Graph is "regular"!

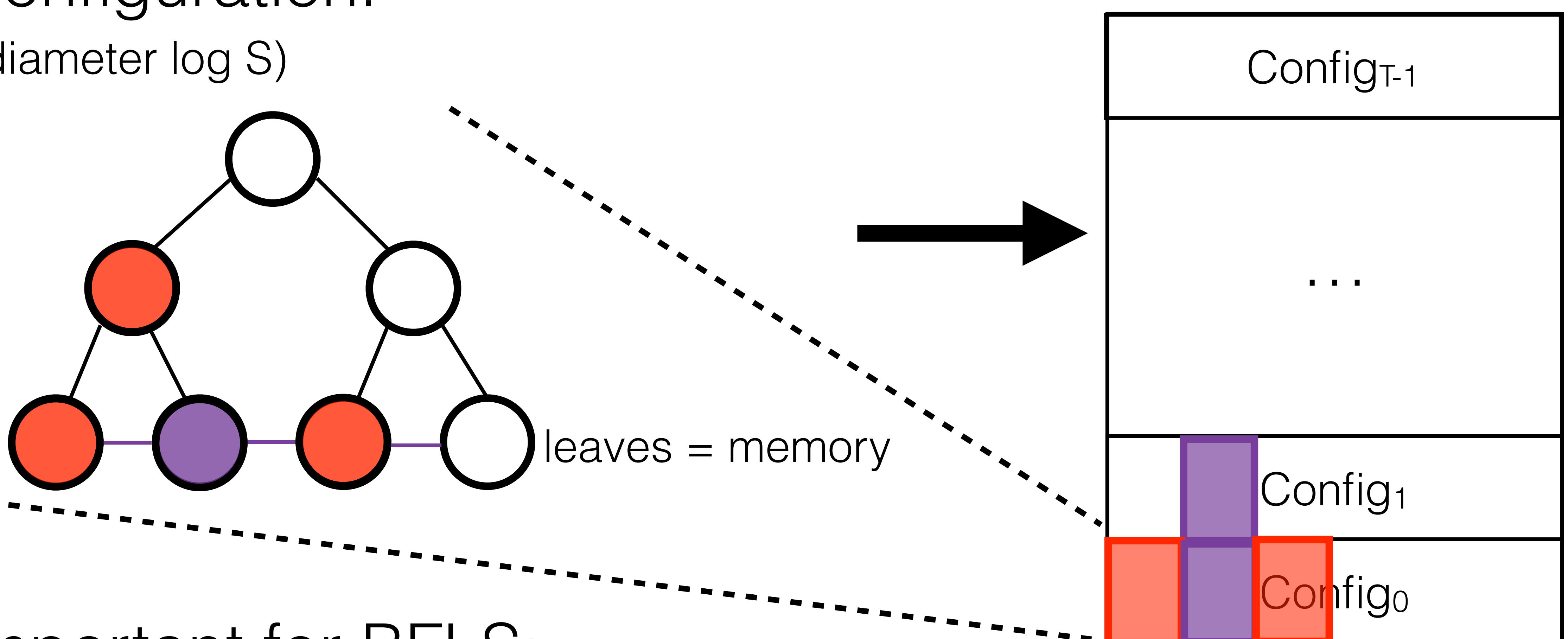
RAM Machines as Circuits

no routing networks!

Transcript / Circuit:

Configuration:

(diameter log S)



Important for BFLS:
Graph is “regular”!

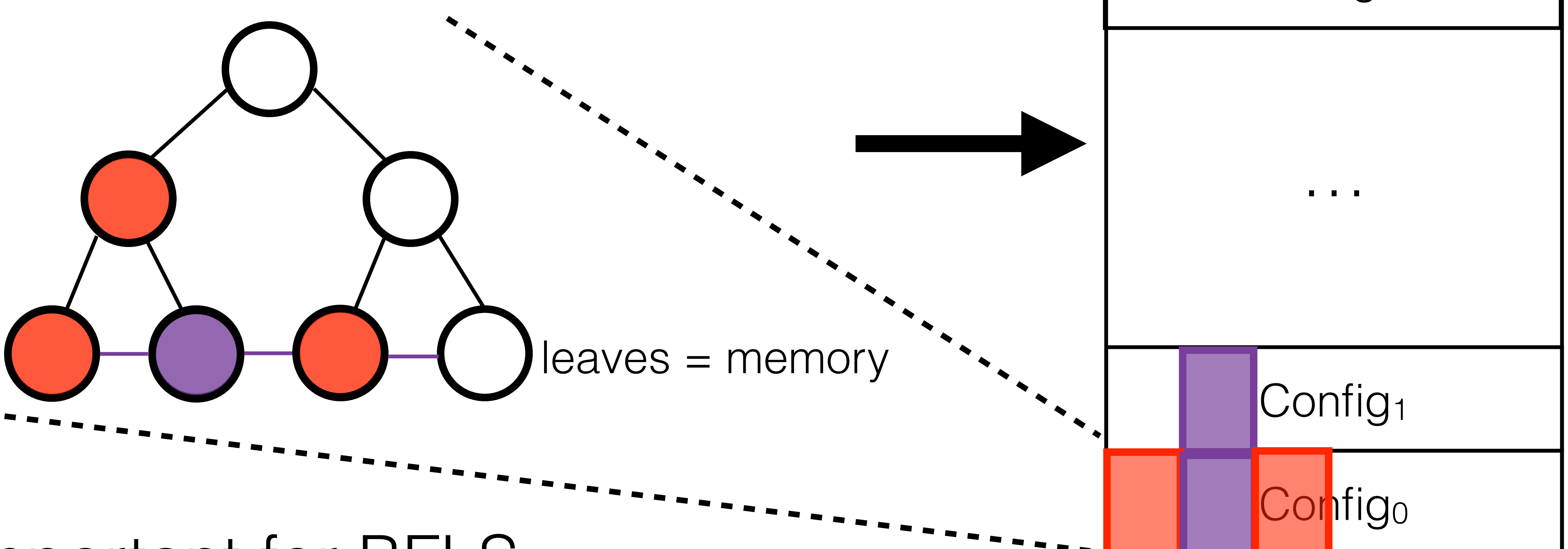
RAM Machines as Circuits

no Merkle trees!

no routing networks!

Transcript / Circuit:

Configuration:
(diameter log S)



Important for BFLS:
Graph is “regular”!

The PCP (BFLS) Part 1: Multilinear extension

The PCP (BFLS) Part 1: Multilinear extension

Let $f : \{0,1\}^m \to \mathbb{F}$
be any function.

The PCP (BFLS) Part 1: Multilinear extension

Let $f : \{0,1\}^m \to \mathbb{F}$
be any function.

0	0
0	1

The PCP (BFLS) Part 1: Multilinear extension

Let $f : \{0,1\}^m \to \mathbb{F}$ be any function.

0	0
0	1

→

multilinear $\hat{f} : \mathbb{F}^m \to \mathbb{F}$

0	-2	-4	-6
0	-1	-2	-3
0	**0**	0	0
0	**1**	2	3

The PCP (BFLS) Part 1: Multilinear extension

Let $f : \{0,1\}^m \to \mathbb{F}$ be any function.

0	0
0	1

multilinear $\hat{f} : \mathbb{F}^m \to \mathbb{F}$

0	-2	-4	-6
0	-1	-2	-3
0	**0**	0	0
0	**1**	2	3

$$\hat{f}(\mathbb{x}) = \sum_{\mathbf{x} \in \{0,1\}^m} f(\mathbf{x}) \cdot \hat{\mathbf{1}_{\mathbf{x}}}(\mathbb{x})$$

The PCP (BFLS) Part 1: Multilinear extension

Let $f : \{0,1\}^m \to \mathbb{F}$ be any function.

0	0
0	1

multilinear $\hat{f} : \mathbb{F}^m \to \mathbb{F}$

0	-2	-4	-6
0	-1	-2	-3
0	**0**	0	0
0	**1**	2	3

"funny x" $\in \mathbb{F}^m$

$$\hat{f}(\mathbb{x}) = \sum_{\mathbf{x}\in\{0,1\}^m} f(\mathbf{x}) \cdot \hat{\mathbf{1}_{\mathbf{x}}}(\mathbb{x})$$

The PCP (BFLS) Part 1: Multilinear extension

Let $f : \{0,1\}^m \to \mathbb{F}$ be any function.

0	0
0	1

multilinear $\hat{f} : \mathbb{F}^m \to \mathbb{F}$

0	-2	-4	-6
0	-1	-2	-3
0	**0**	0	0
0	**1**	2	3

"funny x" $\in \mathbb{F}^m$

$$\hat{f}(\mathbb{x}) = \sum_{\mathbf{x} \in \{0,1\}^m} f(\mathbf{x}) \cdot \hat{\mathbf{1}_{\mathbf{x}}}(\mathbb{x})$$

"bold x" $\in \{0,1\}^m$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x}) \cdot \hat{\mathbf{1}}_{\mathbf{y},\mathbf{x}}(\mathbb{y}, \mathbb{x})$$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \rightarrow \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$

Config$_0$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y},\mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y},\mathbf{x})$$

Config$_0$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$

3 Config$_0$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

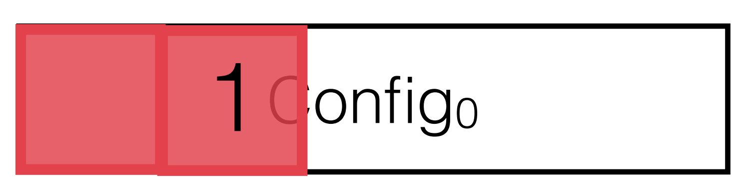
$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$

Config$_0$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$



Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y},\mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y},\mathbf{x})$$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \rightarrow \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \rightarrow \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y},\mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y},\mathbf{x})$$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \rightarrow \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y},\mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y},\mathbf{x})$$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y},\mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y},\mathbf{x})$$

Prover Efficiency

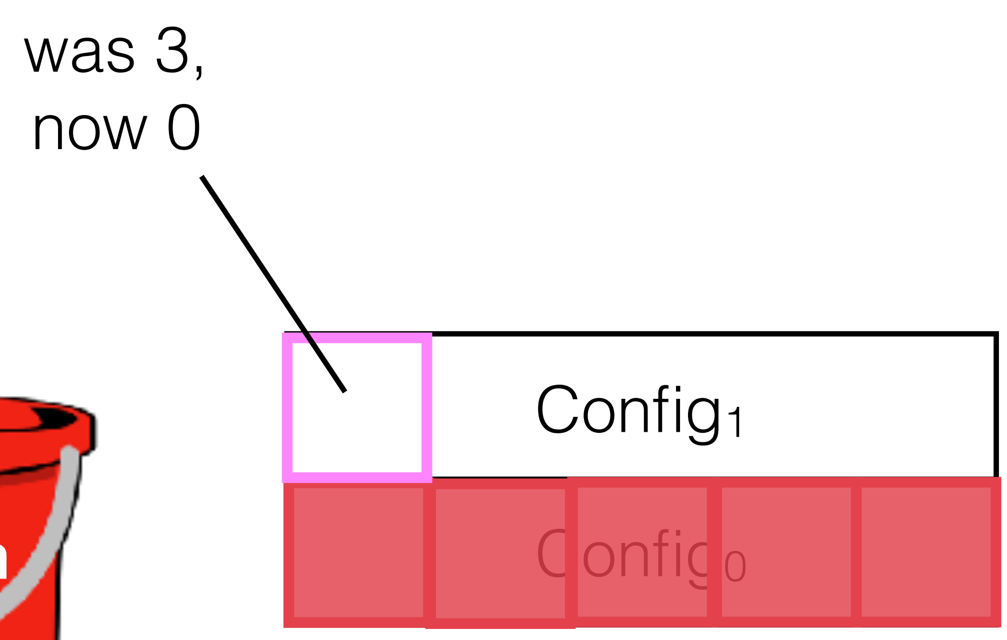
1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$

Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

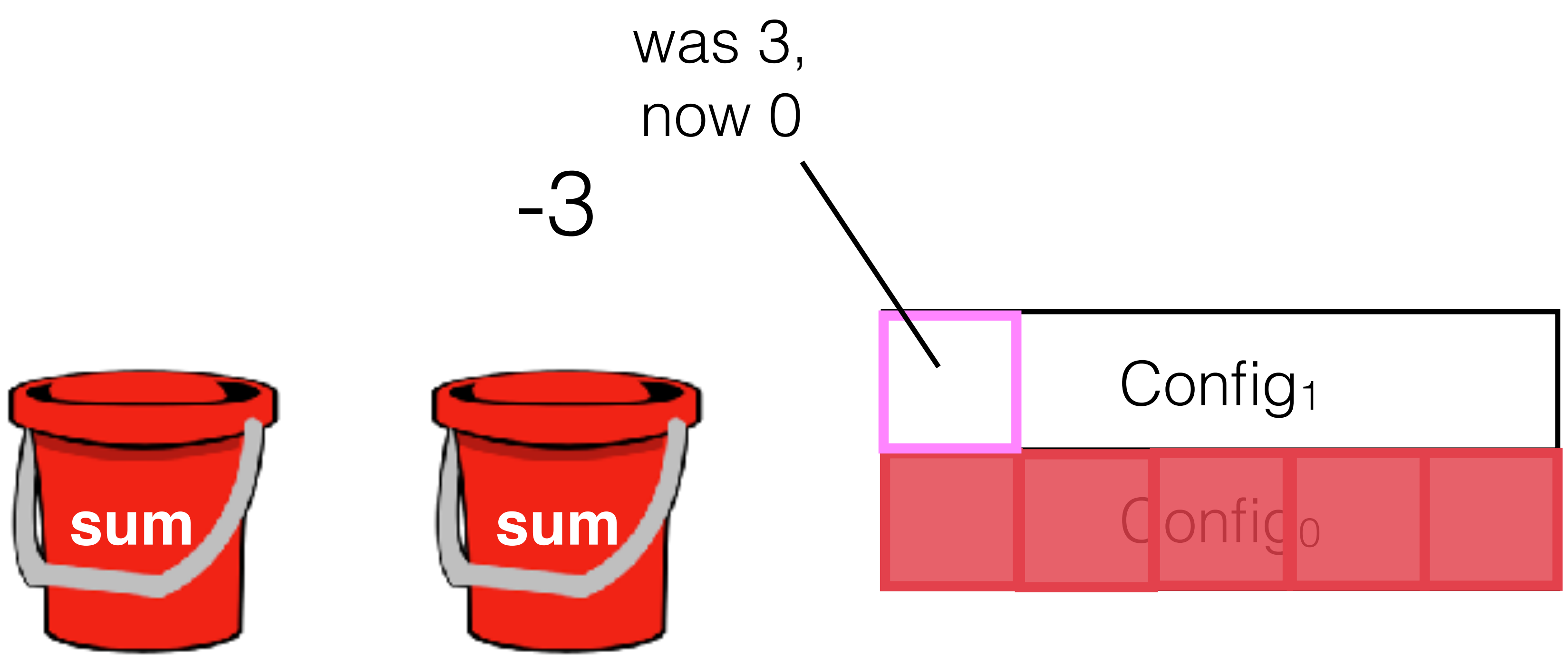
$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$



Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \rightarrow \{0,1\}$

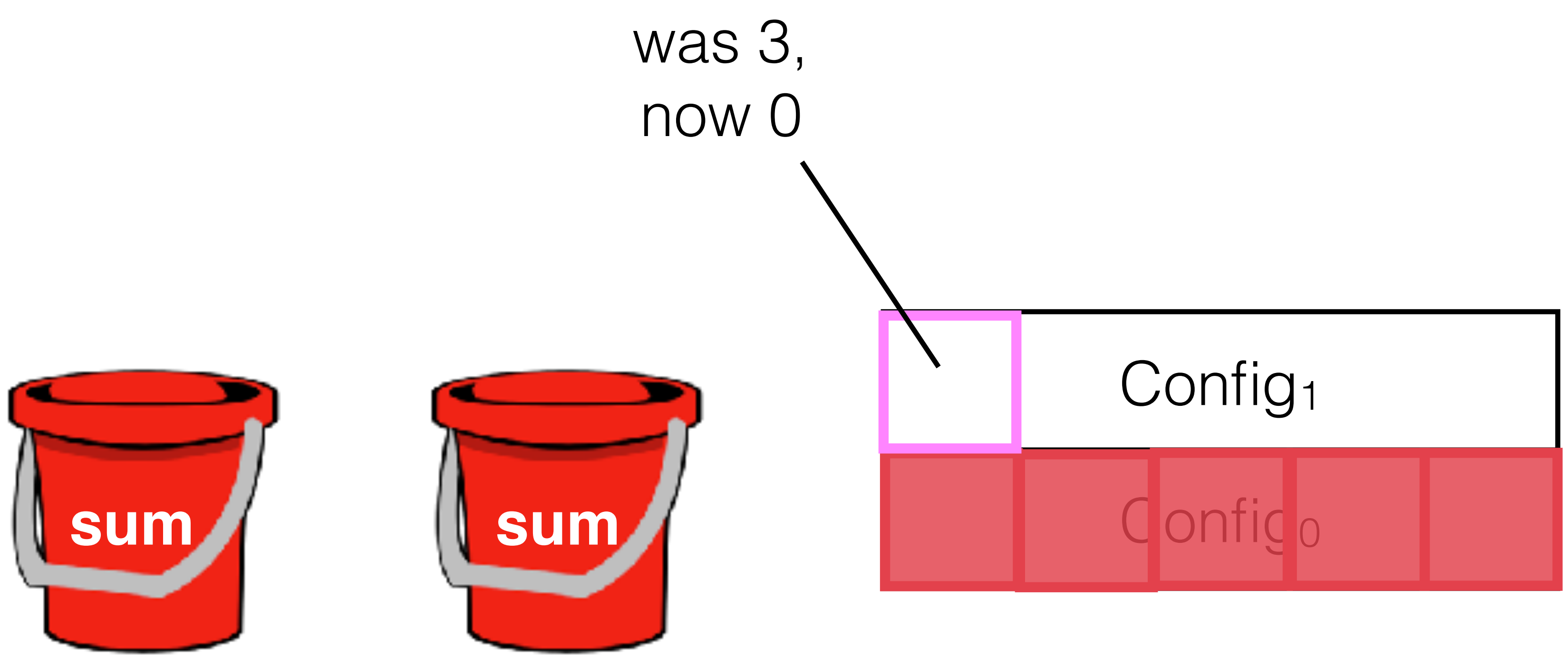
$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$



Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

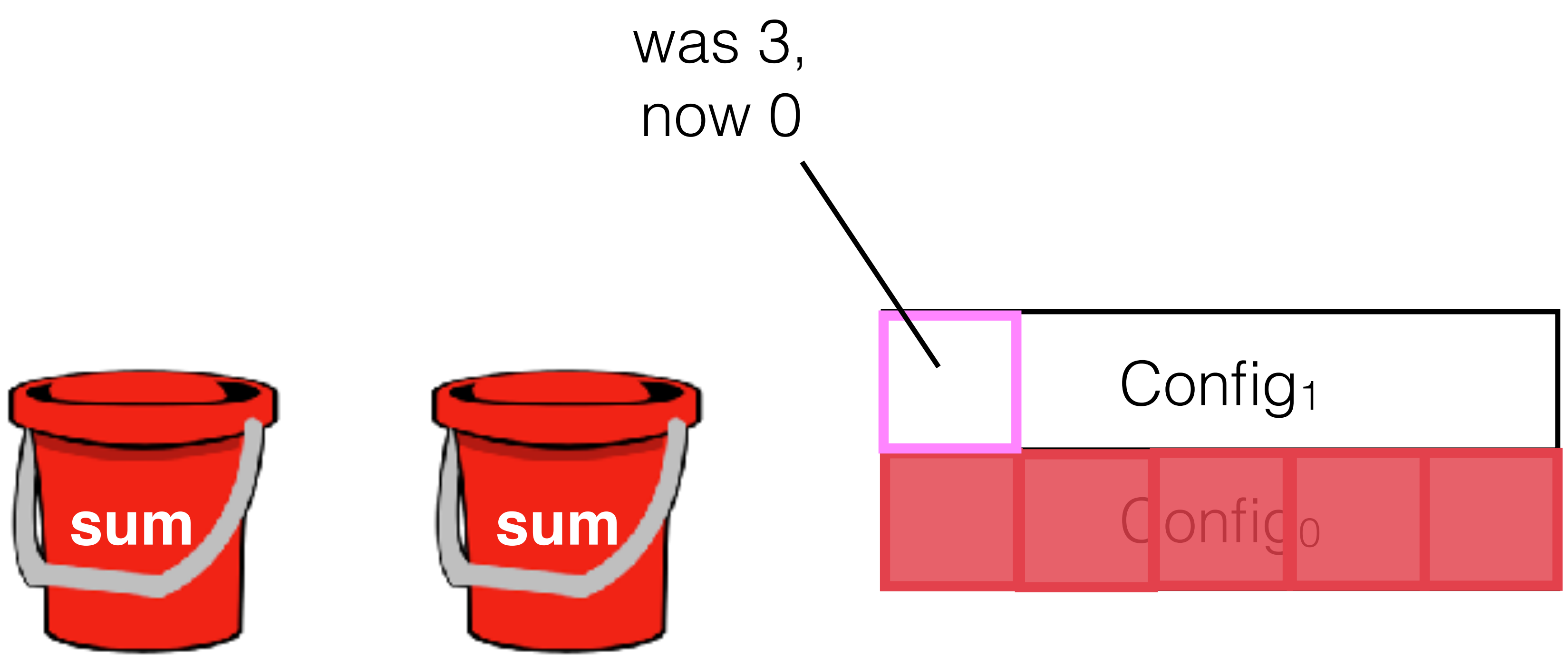
$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$



Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

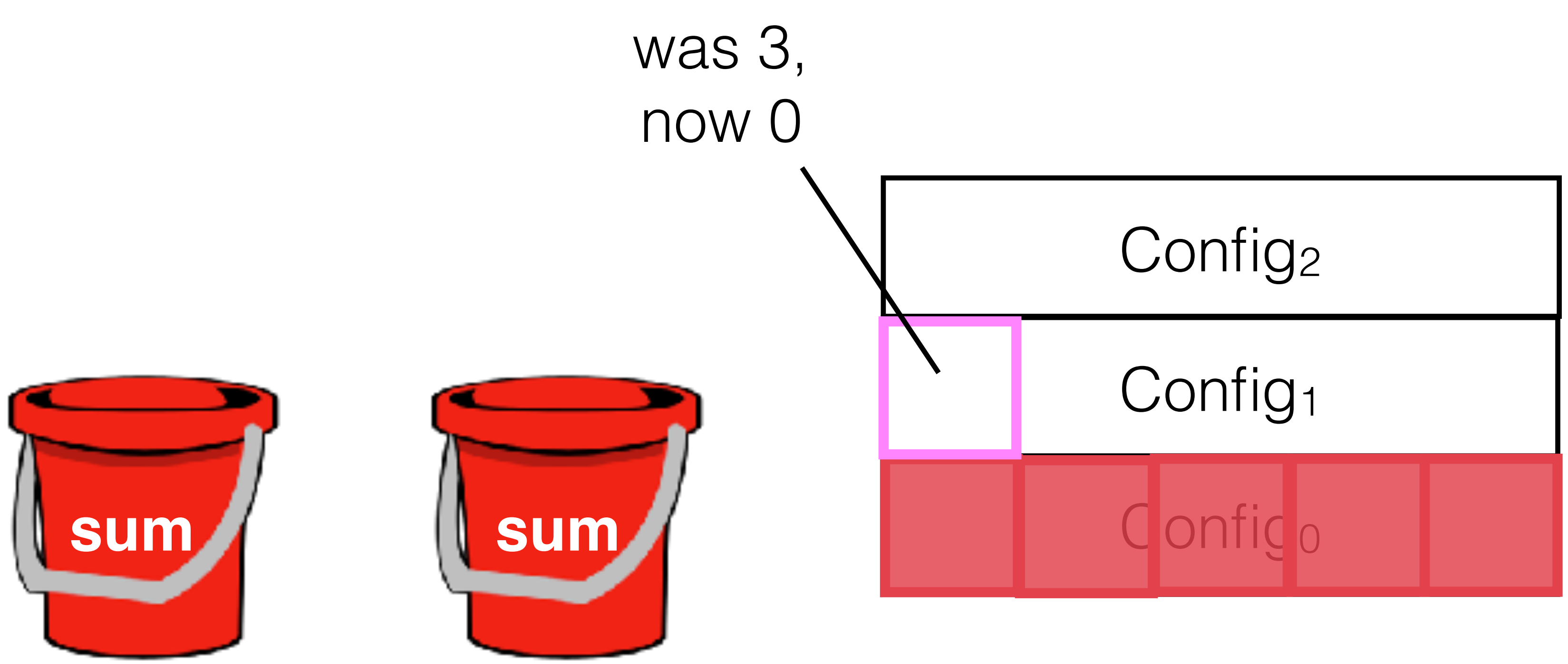
$$\hat{\mathcal{C}}(\mathbb{y},\mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y},\mathbf{x})$$



Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

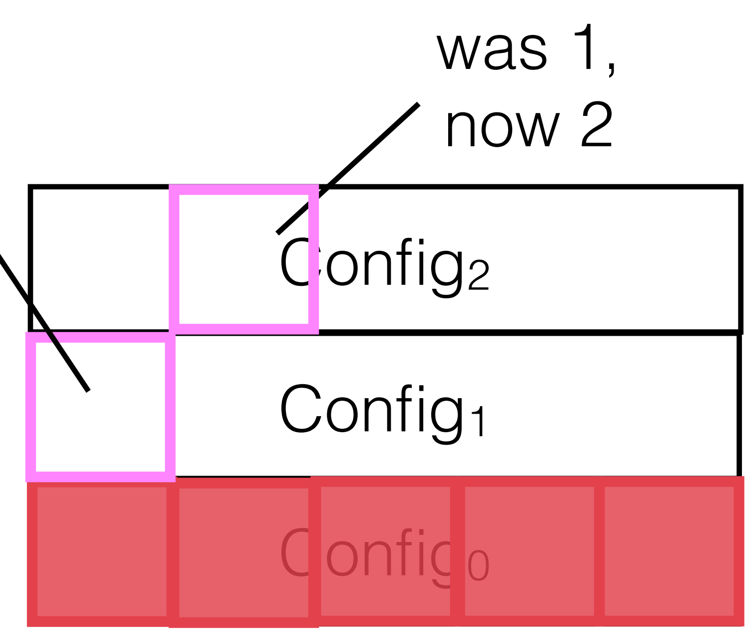
$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$



Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

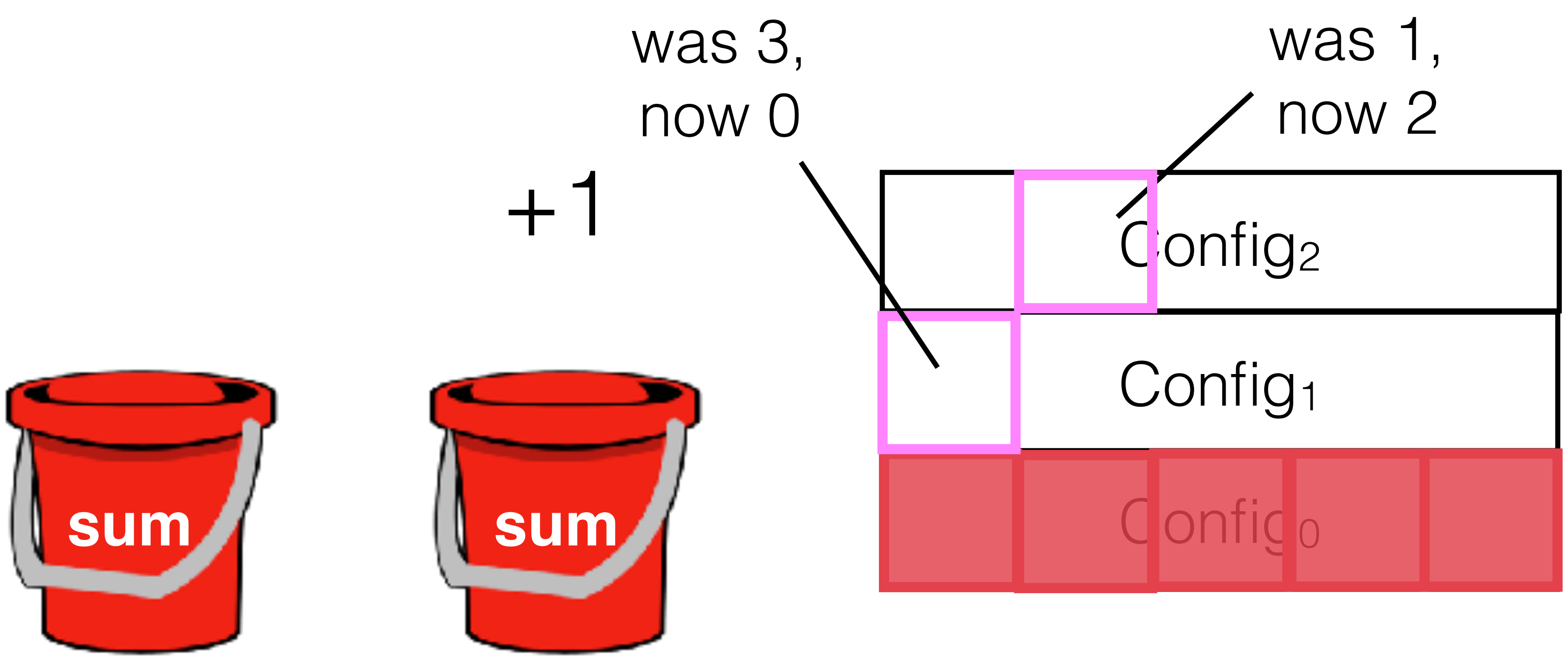
$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$



Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

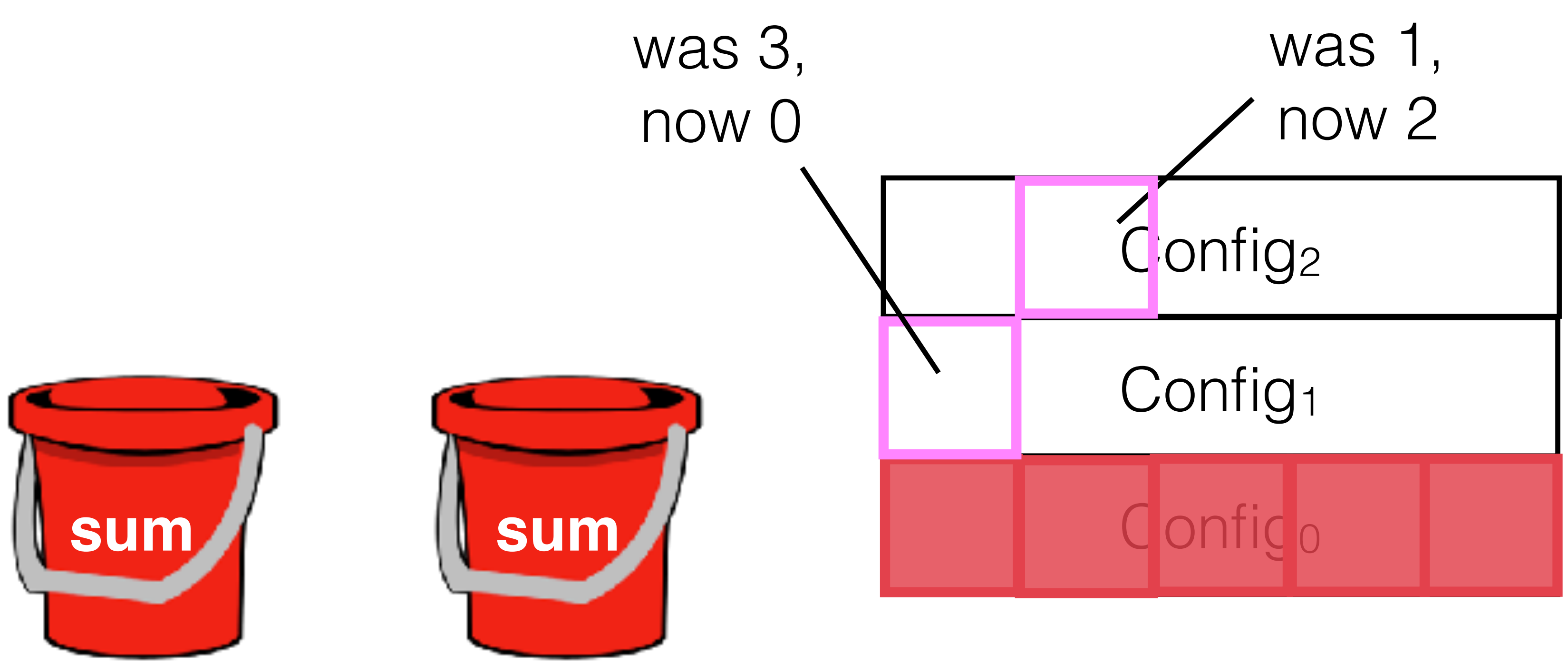
$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$



Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

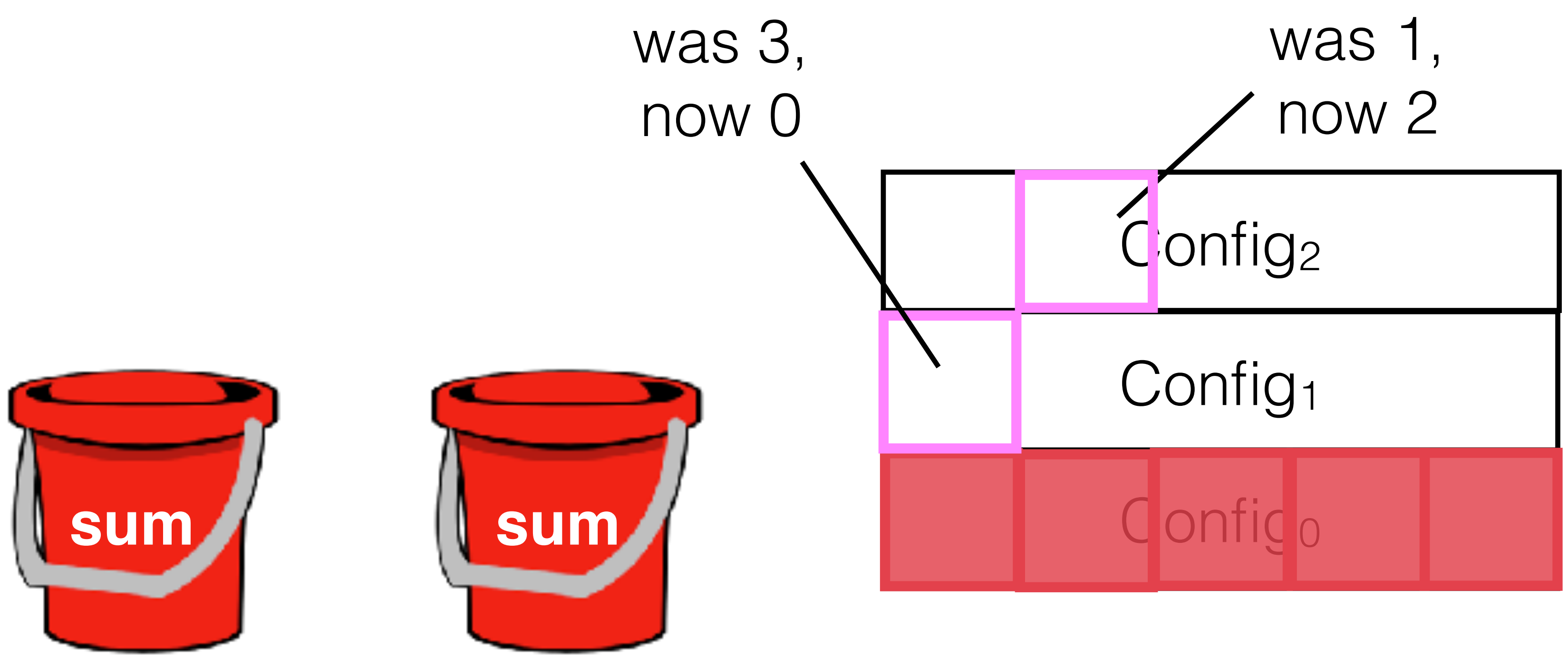
$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$



Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \rightarrow \{0,1\}$

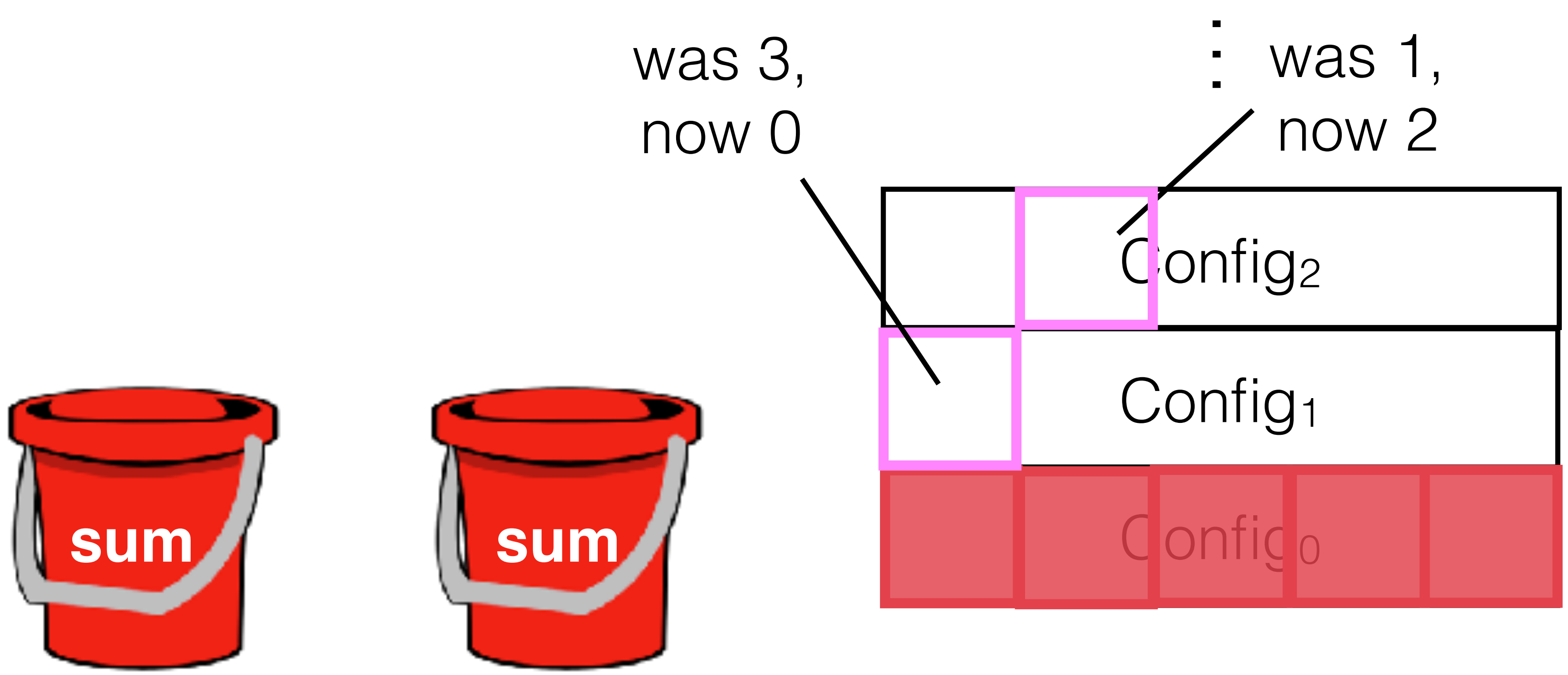
$$\hat{\mathcal{C}}(\mathbb{y},\mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y},\mathbf{x})$$



Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$

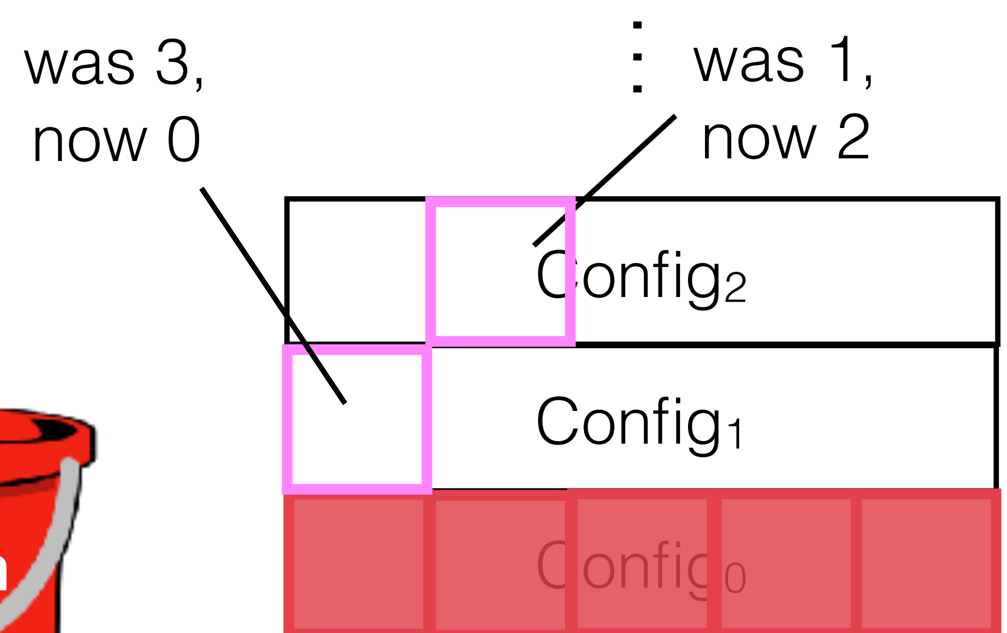


Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y},\mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y},\mathbf{x})$$

$$\sum_{\mathbf{x},\mathbf{y}} \mathcal{C}(\mathbf{x},\mathbf{y})$$

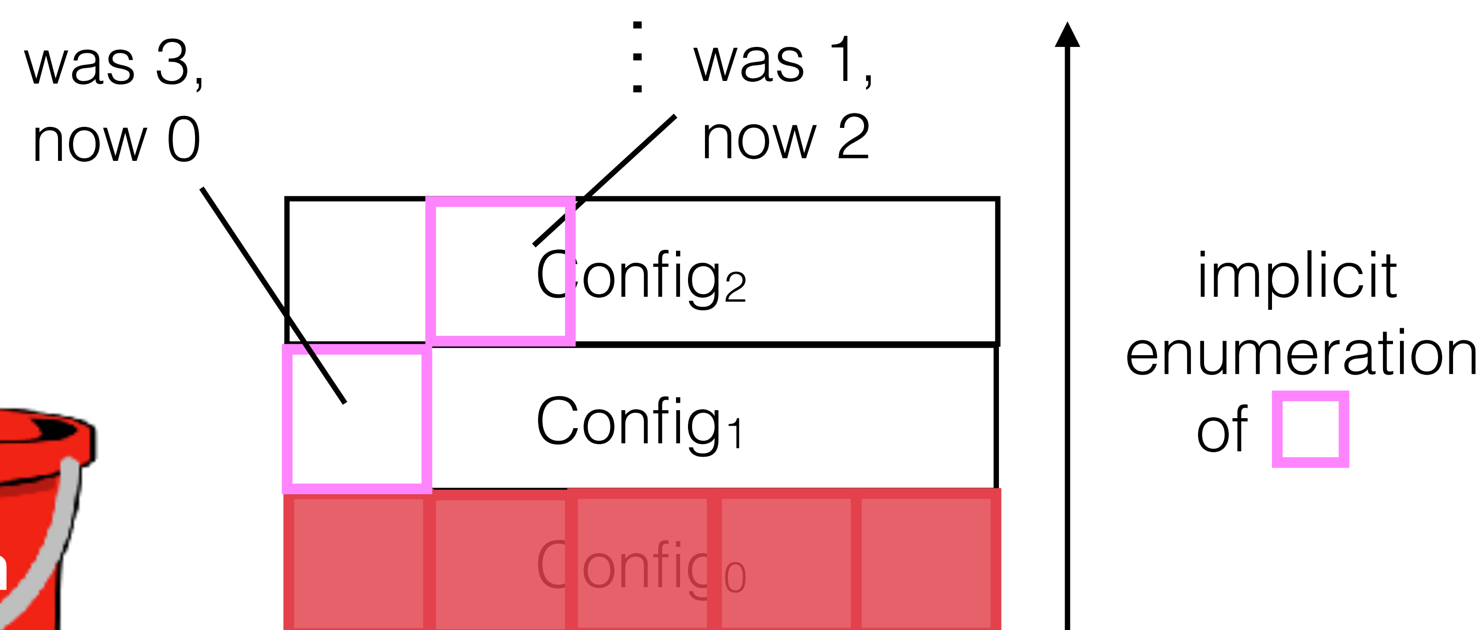


Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x})$$

$$\sum_{\mathbf{x},\mathbf{y}} \mathcal{C}(\mathbf{x}, \mathbf{y})$$

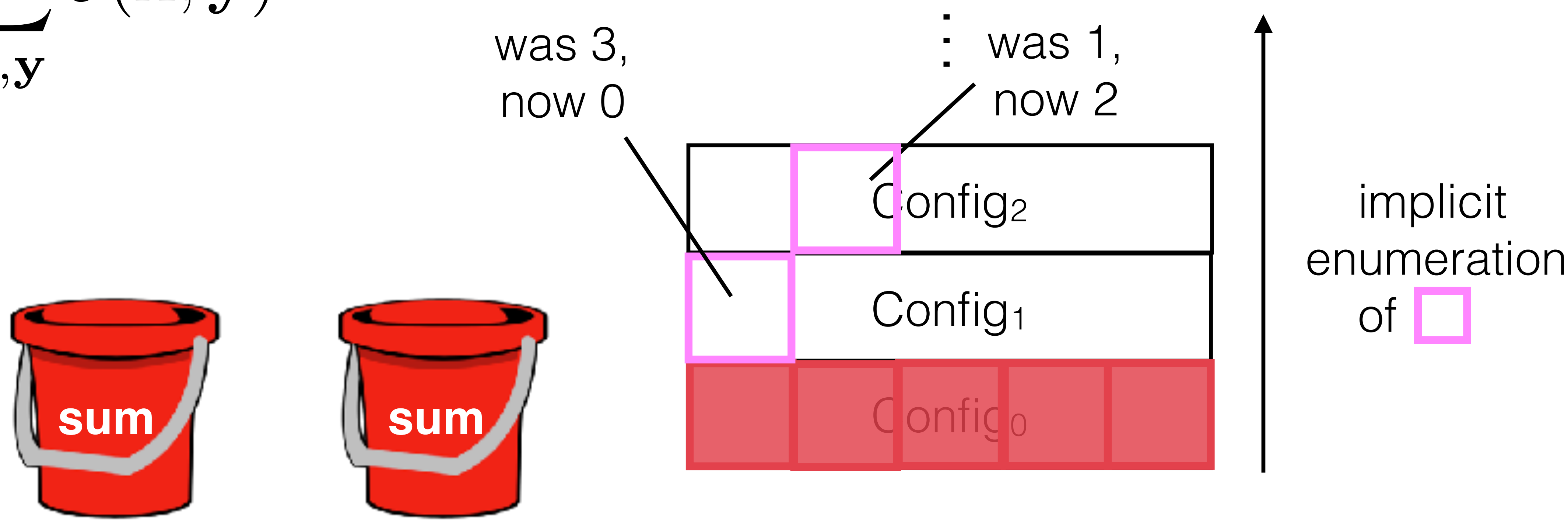


Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x}) \cdot \hat{\mathbf{1}}_{\mathbf{y},\mathbf{x}}(\mathbb{y}, \mathbb{x})$$

$$\sum_{\mathbf{x},\mathbf{y}} \mathcal{C}(\mathbf{x}, \mathbf{y})$$

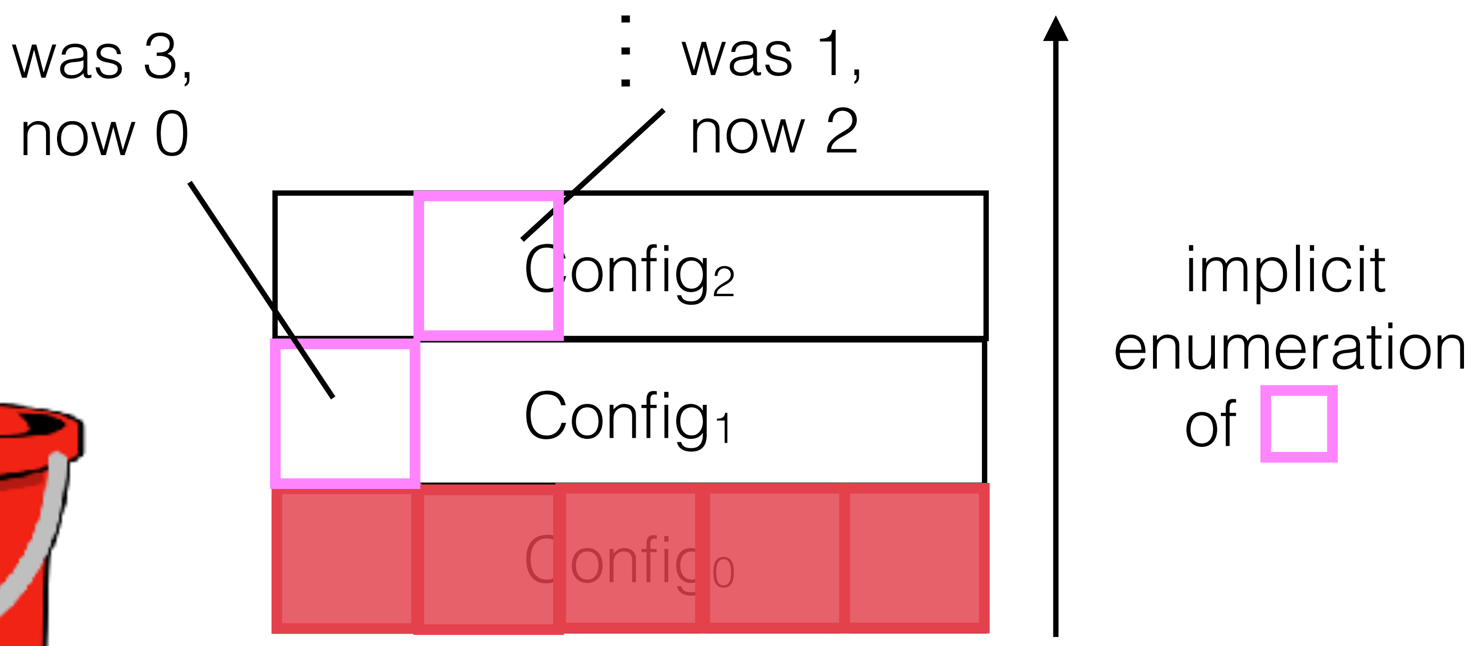


Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y},\mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y},\mathbf{x}) \cdot \hat{\mathbf{1}}_{\mathbf{y},\mathbf{x}}(\mathbb{y},\mathbb{x})$$

$$\sum_{\mathbf{x},\mathbf{y}} \mathcal{C}(\mathbf{x},\mathbf{y})$$



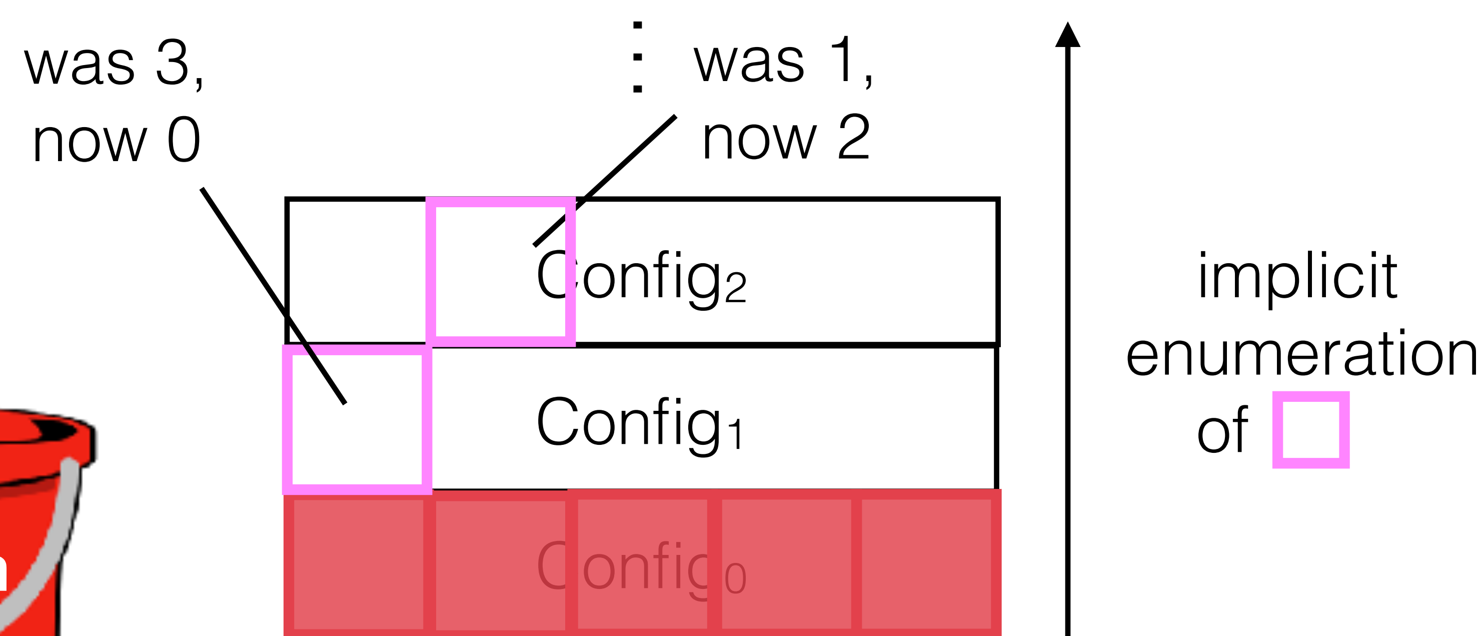
Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y},\mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y},\mathbf{x}) \cdot \hat{\mathbf{1}}_{\mathbf{y},\mathbf{x}}(\mathbb{y},\mathbb{x})$$

Coefficients structured;
all is still well

$$\sum_{\mathbf{x},\mathbf{y}} \mathcal{C}(\mathbf{x},\mathbf{y})$$



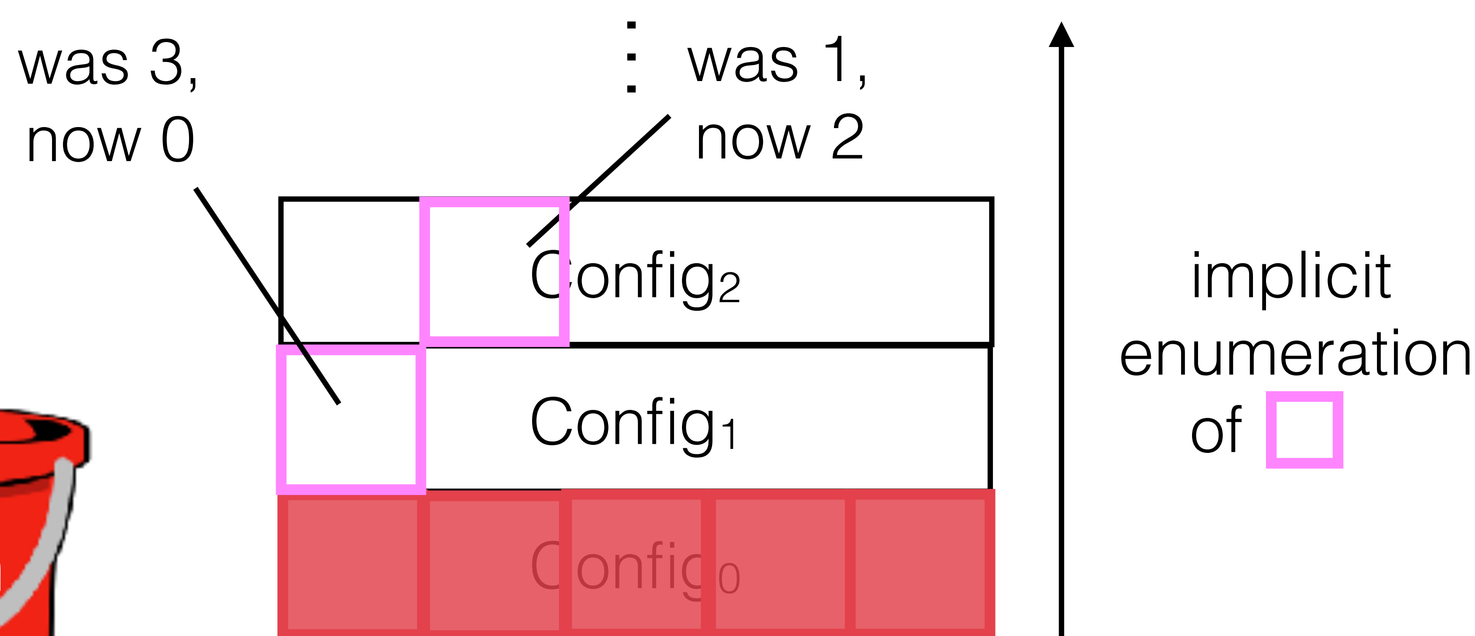
Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \to \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y},\mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y},\mathbf{x}) \cdot \hat{\mathbf{1}}_{\mathbf{y},\mathbf{x}}(\mathbb{y},\mathbb{x})$$

Coefficients structured;
all is still well

$$\sum_{\mathbf{x},\mathbf{y}} \mathcal{C}(\mathbf{x},\mathbf{y})$$



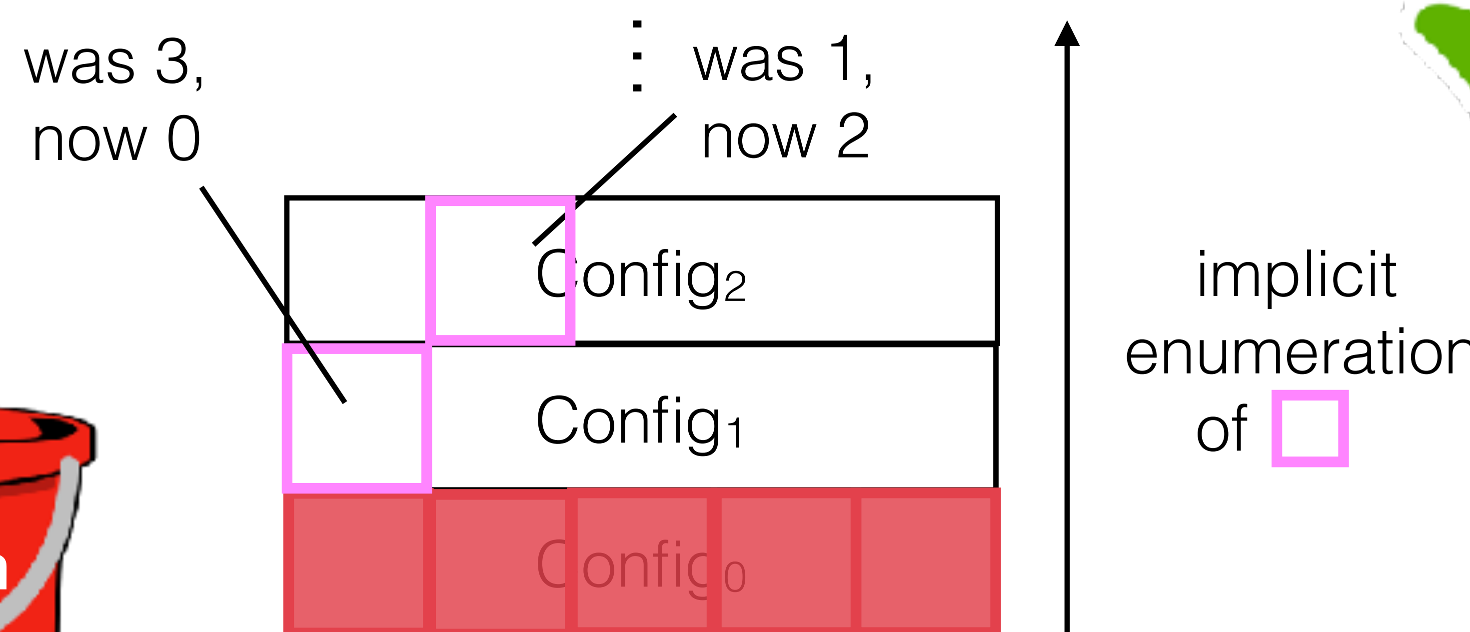
Prover Efficiency

1. Evaluating extension of transcript $\hat{\mathcal{C}} : \{0,1\}^{t+s} \rightarrow \{0,1\}$

$$\hat{\mathcal{C}}(\mathbb{y}, \mathbb{x}) = \sum_{\mathbf{y},\mathbf{x}} \mathcal{C}(\mathbf{y}, \mathbf{x}) \cdot \hat{\mathbf{1}}_{\mathbf{y},\mathbf{x}}(\mathbb{y}, \mathbb{x})$$

Coefficients structured;
all is still well

$$\sum_{\mathbf{x},\mathbf{y}} \mathcal{C}(\mathbf{x}, \mathbf{y})$$



Additional Challenges

Additional Challenges

- Other (sum-check) polynomials

Additional Challenges

- Other (sum-check) polynomials
- Getting rid of KRR's augmented circuit

Additional Challenges

- Other (sum-check) polynomials
- Getting rid of KRR's augmented circuit
- Prover efficiency under somewhat homomorphic encryption

Additional Challenges

- Other (sum-check) polynomials
- Getting rid of KRR’s augmented circuit
- Prover efficiency under somewhat homomorphic encryption
  - Low multiplicative degree,
    O(1) field operations per step

Additional Challenges

- Other (sum-check) polynomials
- Getting rid of KRR's augmented circuit
- Prover efficiency under somewhat homomorphic encryption
  - Low multiplicative degree,
    O(1) field operations per step
  - Space stays $S + \mathsf{poly}(\kappa)$, not $S \cdot \mathsf{poly}(\kappa)$

Summary

	Assumptions	Prover Time	Prover Space
No-Signaling PCPs [KRR, ...]	PIR	$\geq T^3 S^3$	$\geq T^3 S^3$
SNARKs [BC,BCCT, ...]	Non-Falsifiable	$T \cdot \text{poly}(\kappa)$	$S \cdot \text{poly}(\kappa)$
Succinct Garbling [GHRW, KLW, ...]	Obfuscation/ multilinear maps	$T \cdot \text{poly}(\kappa)$	$S \cdot \text{poly}(\kappa)$
[this work]	"Slightly" Homomorphic Encryption	$T \cdot \text{poly}(\kappa)$	$S + \text{poly}(\kappa)$

Open Questions

- How does this compare in practice? What are the remaining bottlenecks?
- Can PCP query complexity be reduced?
- Is there an FHE scheme which is extra efficient for our prover?
    - Efficiently evaluate low-degree arithmetic circuits (large fields)

Open Questions

- How does this compare in practice? What are the remaining bottlenecks?
- Can PCP query complexity be reduced?
- Is there an FHE scheme which is extra efficient for our prover?
  - Efficiently evaluate low-degree arithmetic circuits (large fields)

low "asymmetric" degree (GSW) even better