Security Bounds for Proof-Carrying Data from Straightline Extractors

Alessandro Chiesa, Ziyi Guan, Shahar Samocha, Eylon Yogev

TL;DR

What is *proof-carrying data* (PCD)?
- Recursive compositions of SNARKs.
- It’s useful for efficiently verifying distributed computations.

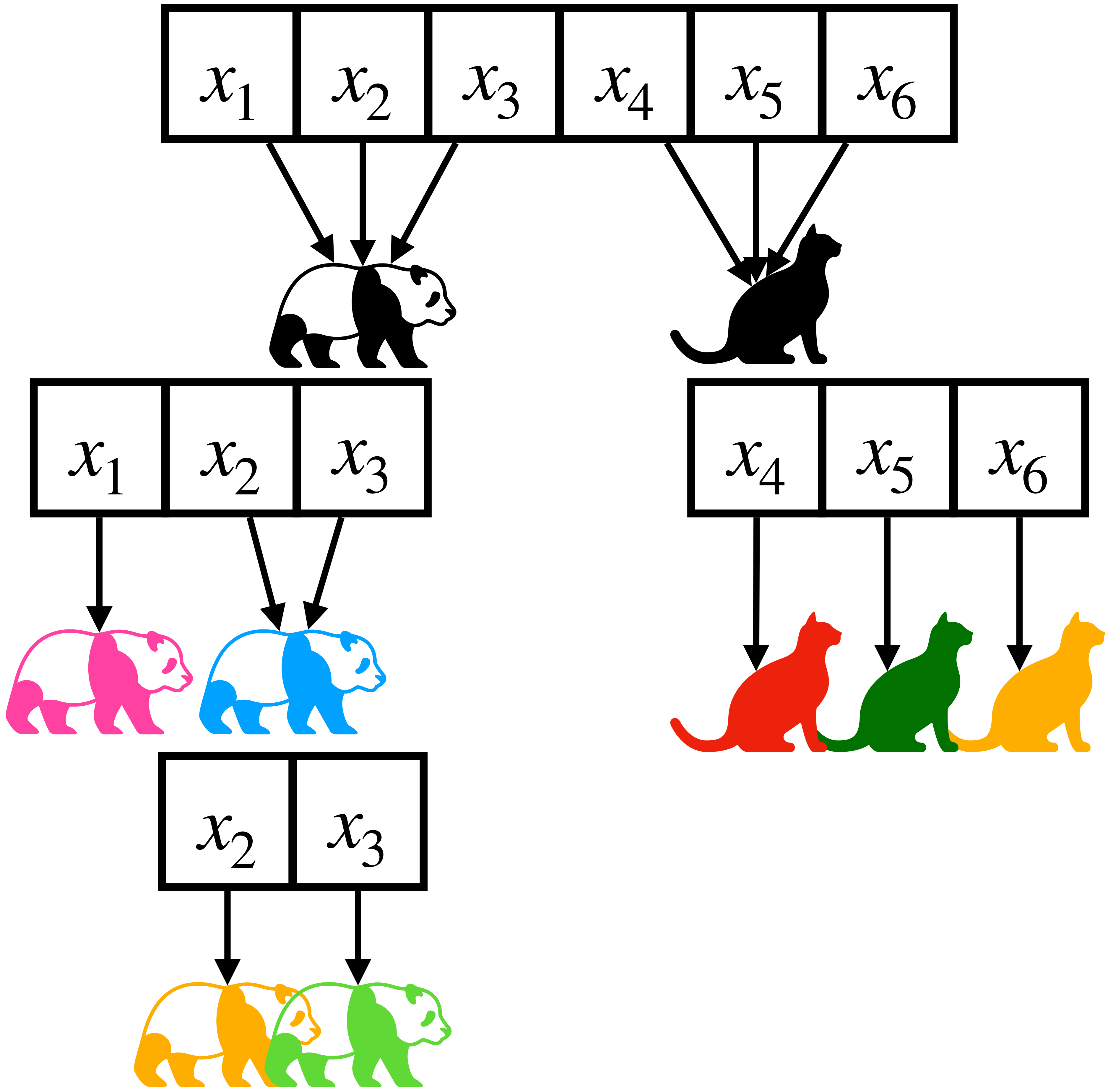
Problem:
- PCD is deployed under the assumption "security of PCD" = "security of underlying SNARK".
- BUT existing security analyses show a huge gap in security ("PCD is far less secure than underlying SNARK").

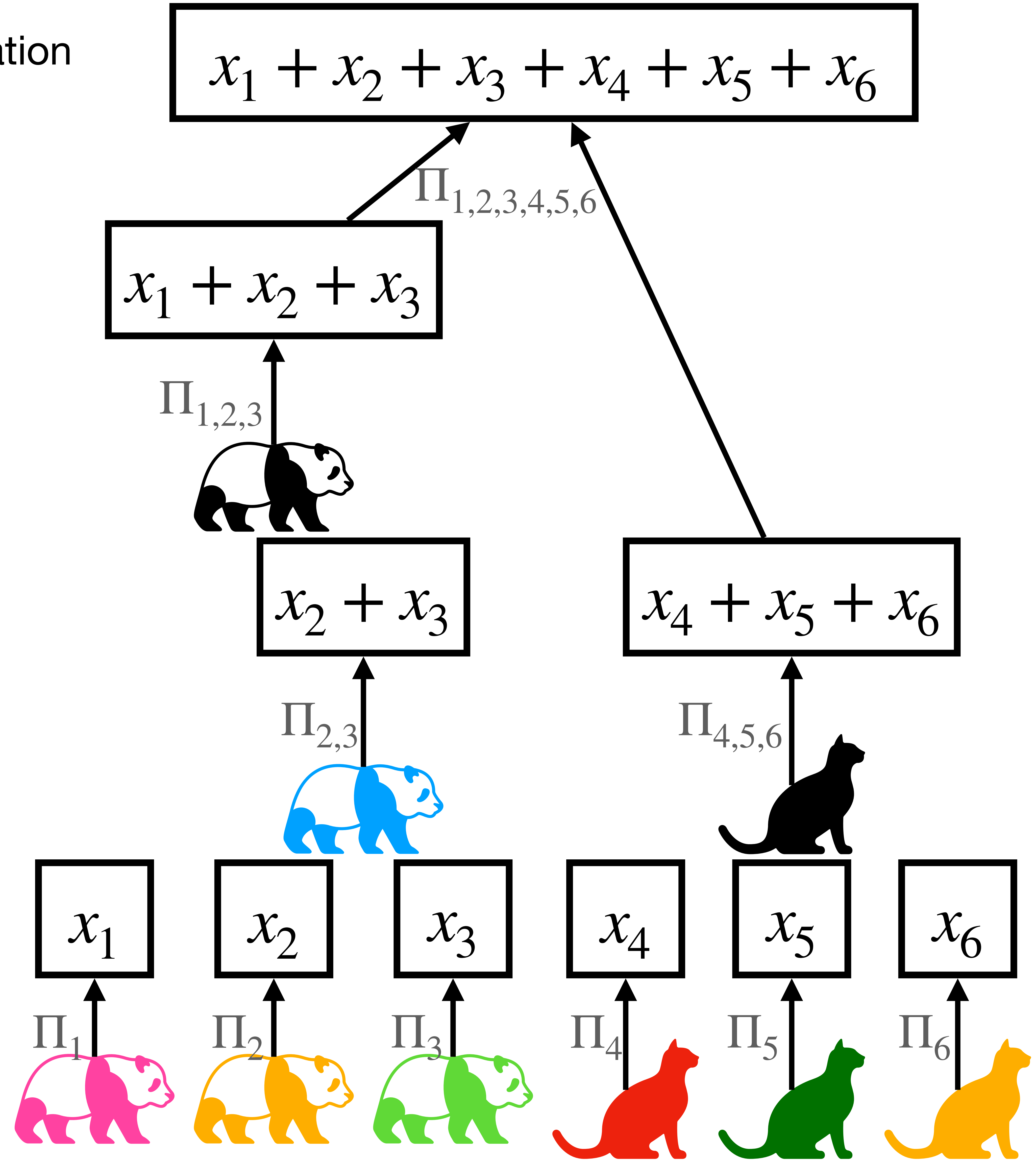
This work:
- We propose an **idealized PCD** that models hash-based PCD in practice.
- We prove that this idealized PCD is **as secure as its underlying SNARK**.

What is proof-carrying data (PCD)? [1/2]

Proof-carrying data (PCD)
- Enables mutually distrustful parties to perform a distributed computation
- The correctness of each step can be **verified efficiently**

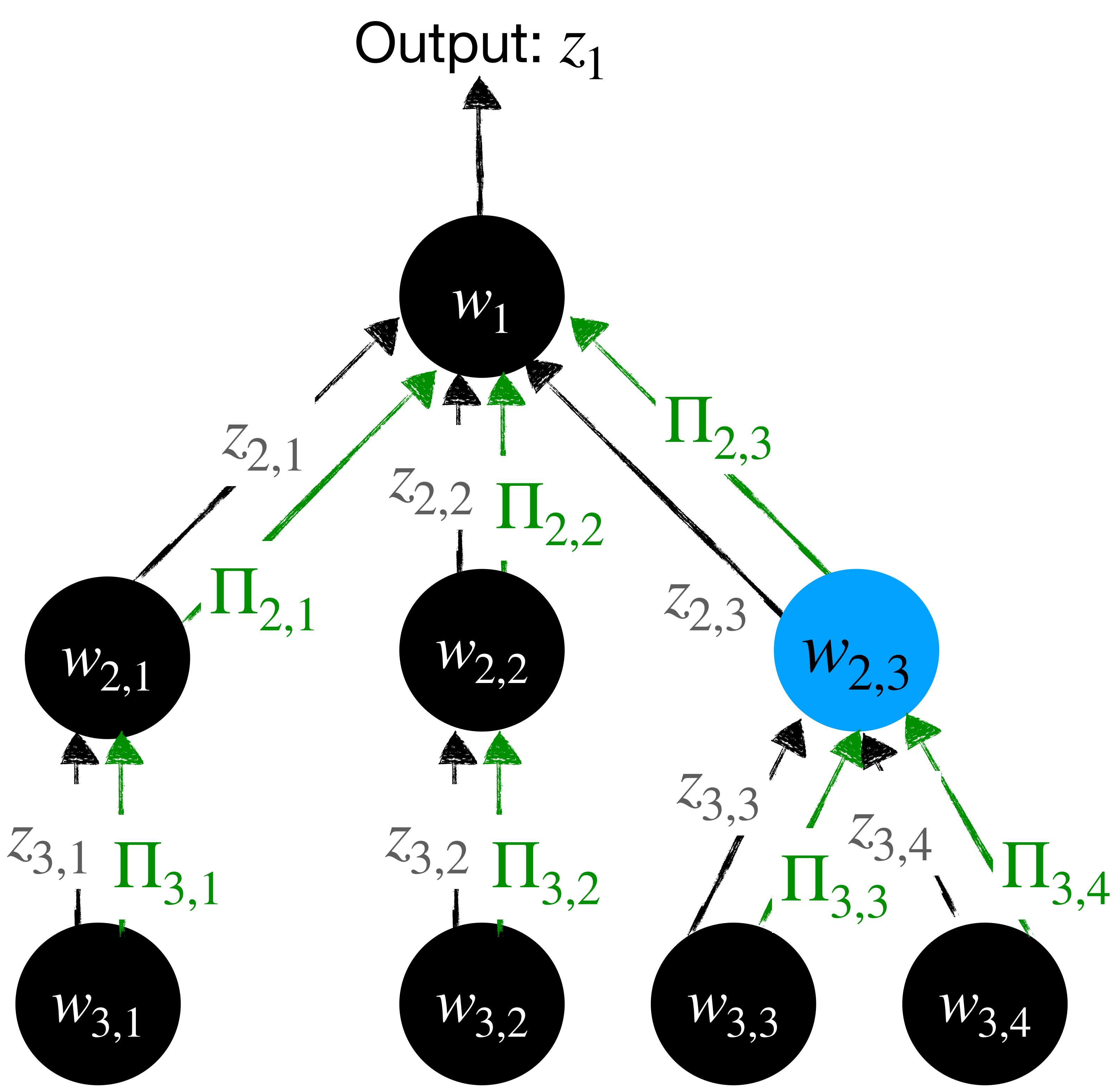
E.g. A simple distributed computation: summing six numbers





What is proof-carrying data (PCD)? [2/2]

Proof-carrying data (PCD)
- Enables mutually distrustful parties to perform a distributed computation
- The correctness of each step can be **verified efficiently**

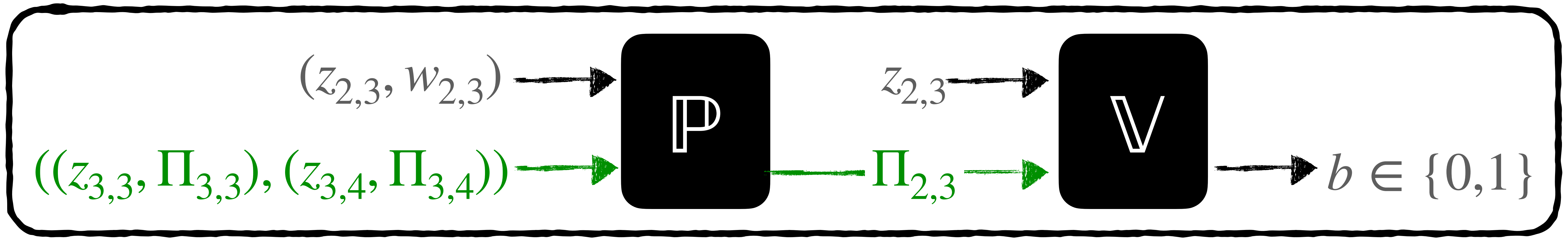


PCD transcript T for a distributed computation
with size $\mathrm{N} = 8$ and depth $\mathrm{D} = 3$

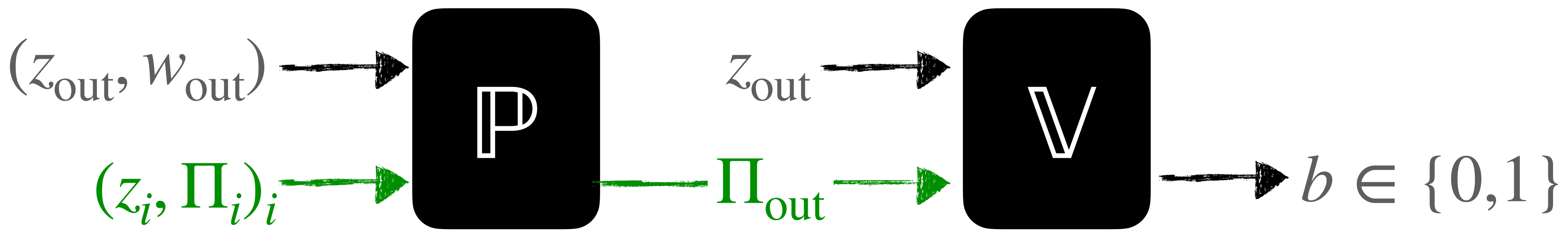
Correctness of transcript T is determined by compliance predicate ϕ
- Node $(2,3)$ is correct if $\phi(z_{2,3}, w_{2,3}, (z_{3,3}, z_{3,4})) = 1$.
- T is ϕ-compliant if all nodes are correct.

The proof string $\Pi_{2,3}$ attests that:
- node $(2,3)$ is correct, AND
- each child vertex of node $(2,3)$ has a valid proof string.

PCD prover $\mathbb{P}$ and PCD verifier $\mathbb{V}$



Security guarantee of PCD

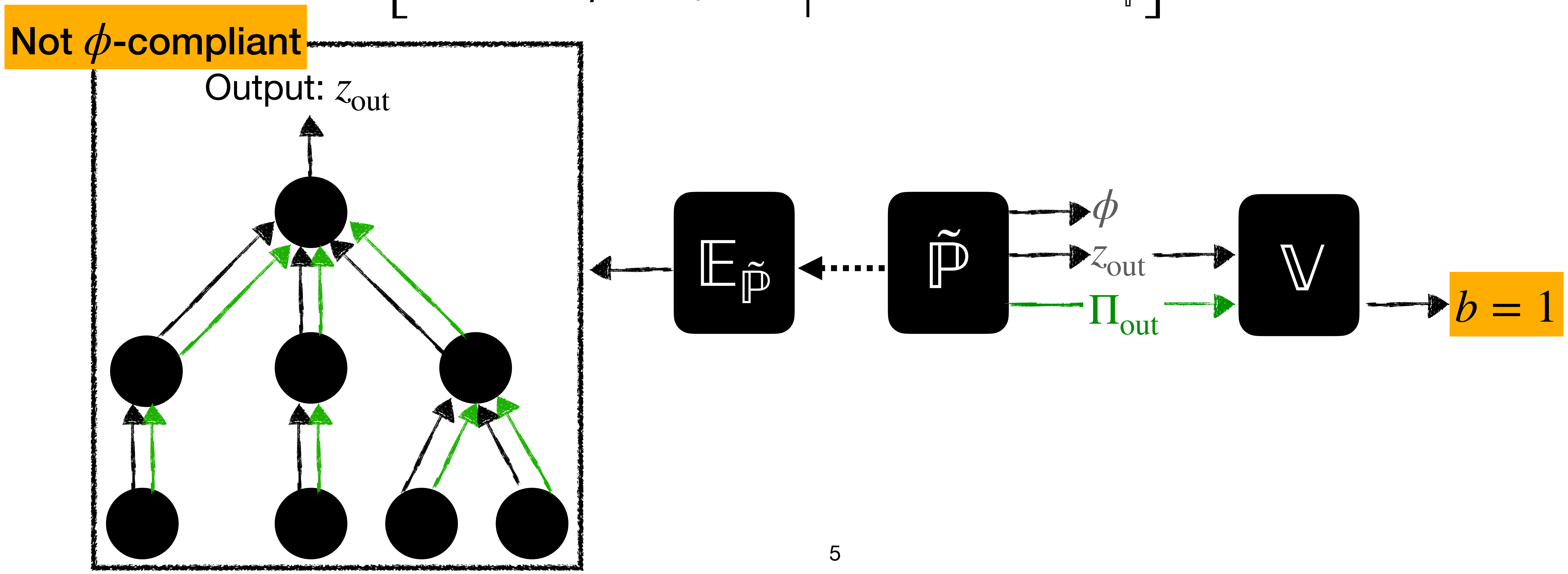


λ: security parameter
T: computation transcript
D: maximum transcript depth
N: maximum transcript size

Perfect completeness: $\mathbb{P}$ can convince $\mathbb{V}$ of correct computations.

Knowledge soundness: $\forall$ bounded $\tilde{\mathbb{P}}$, $\exists$ an efficient extractor $\mathbb{E}_{\tilde{\mathbb{P}}}$ such that

$$\Pr\left[\begin{array}{l} \mathbb{V}(z_{\text{out}}, \Pi_{\text{out}}) = 1 \\ \wedge\, T \text{ is not } \phi\text{-compliant} \end{array} \,\middle|\, \begin{array}{r} (\phi, z_{\text{out}}, \Pi_{\text{out}}) \leftarrow \tilde{\mathbb{P}} \\ T \leftarrow \mathbb{E}_{\tilde{\mathbb{P}}} \end{array}\right] \leq \kappa(\lambda, \mathrm{D}, \mathrm{N}).$$

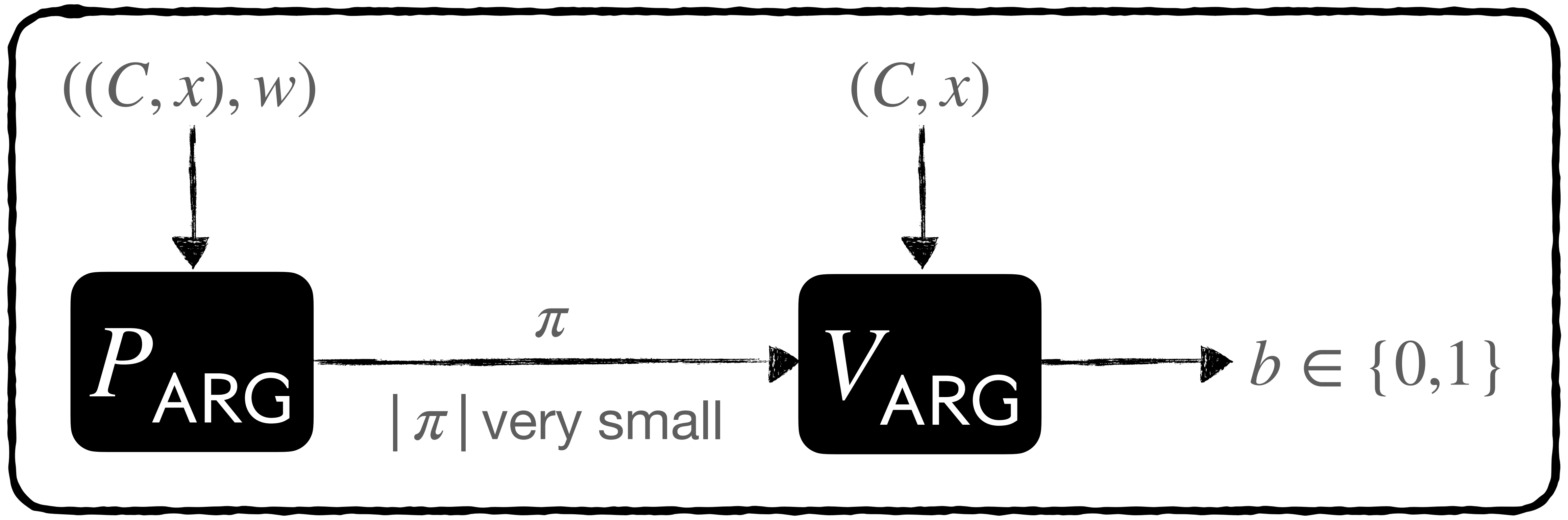


Review: SNARK

PCD can be constructed from a SNARK (e.g., for CSAT).

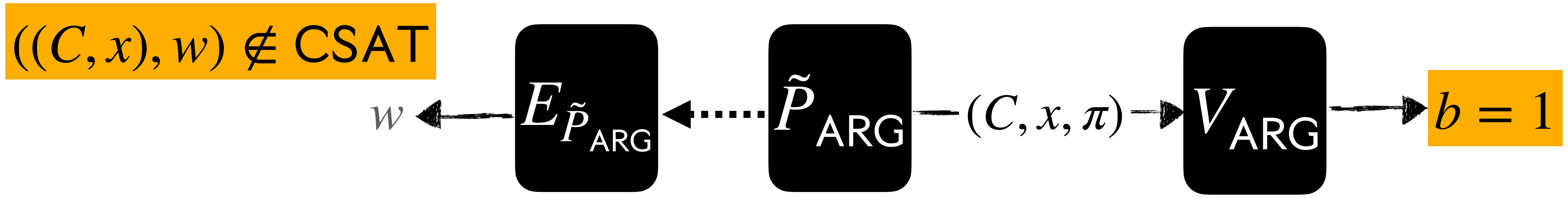
$$\mathsf{CSAT} := \{((C,x),w) : C(x,w) = 1\}$$

$\mathsf{ARG} = (P_{\mathsf{ARG}}, V_{\mathsf{ARG}})$

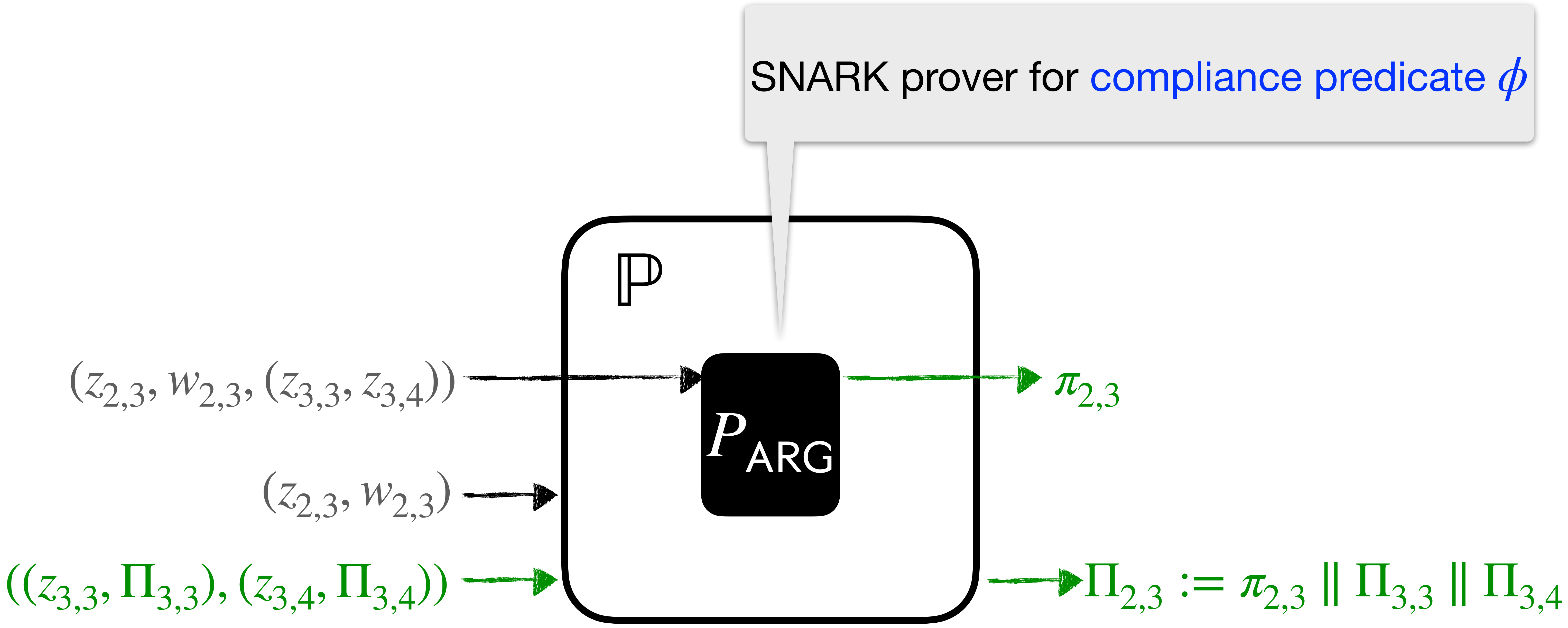


- Perfect completeness: P_{ARG} convinces V_{ARG} if $C(x,w) = 1$.
- Knowledge soundness: $\forall$ bounded $\tilde{P}_{\mathsf{ARG}}$, $\exists$ an efficient extractor $E_{\tilde{P}_{\mathsf{ARG}}}$ such that

$$\Pr\left[\begin{array}{c|r} ((C,x),w) \notin \mathsf{CSAT} & (C,x,\pi) \leftarrow \tilde{P}_{\mathsf{ARG}} \\ \wedge\ V_{\mathsf{ARG}}(C,x,\pi) = 1 & w \leftarrow E_{\tilde{P}_{\mathsf{ARG}}} \end{array}\right] \leq \kappa_{\mathsf{ARG}}(\lambda).$$



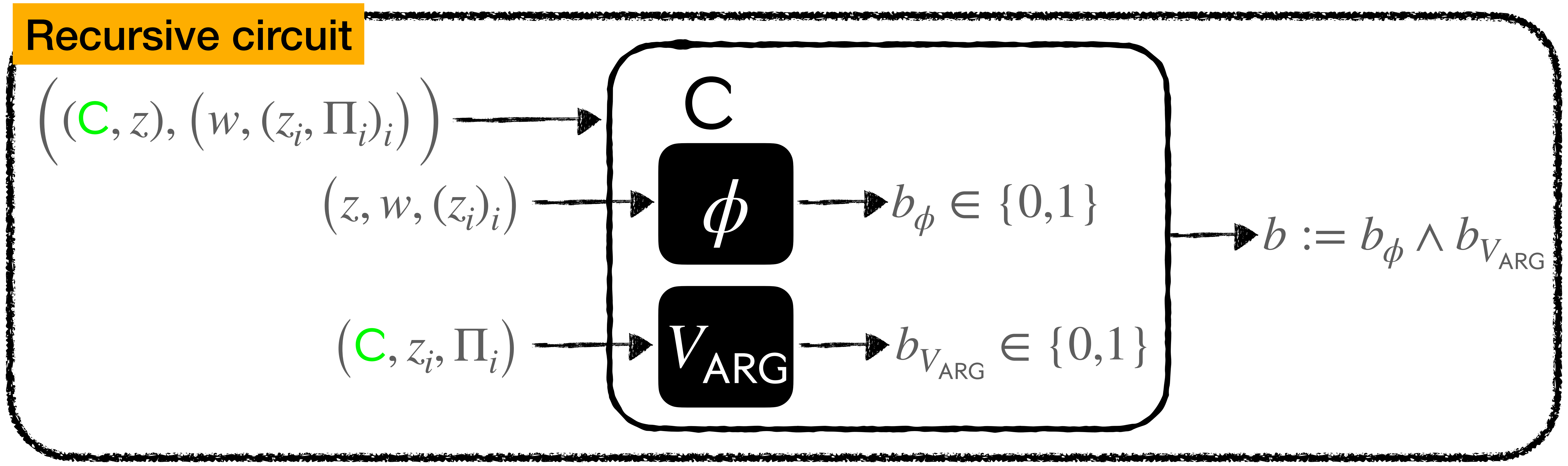
Naive approach: concatenate SNARK proofs

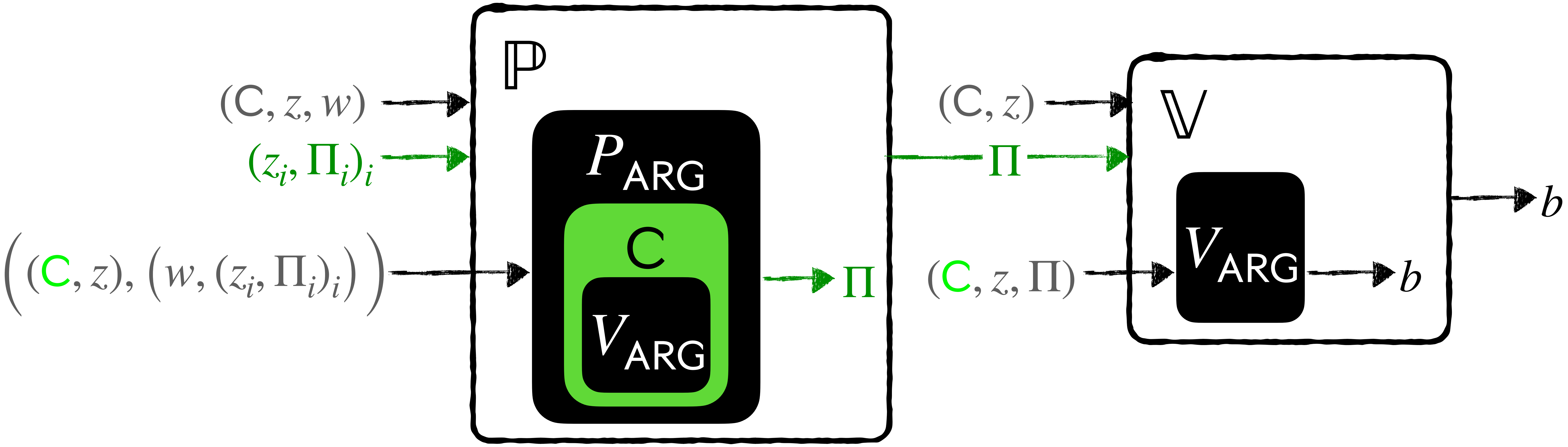


Issue: $\Pi_{2,3}$ is NOT succinct (linear in number of vertices)

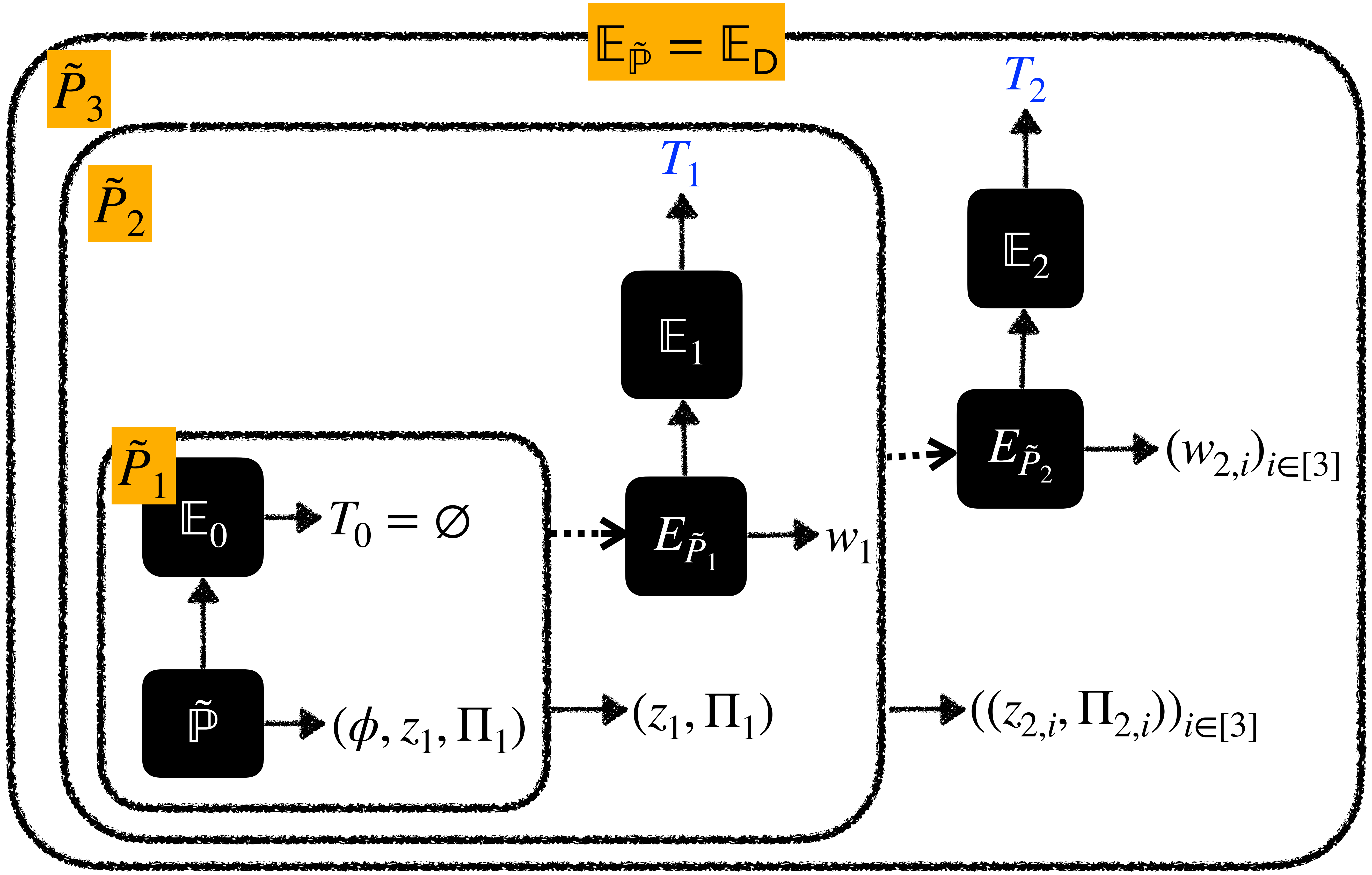
Working idea: Recursively compose the SNARK proofs

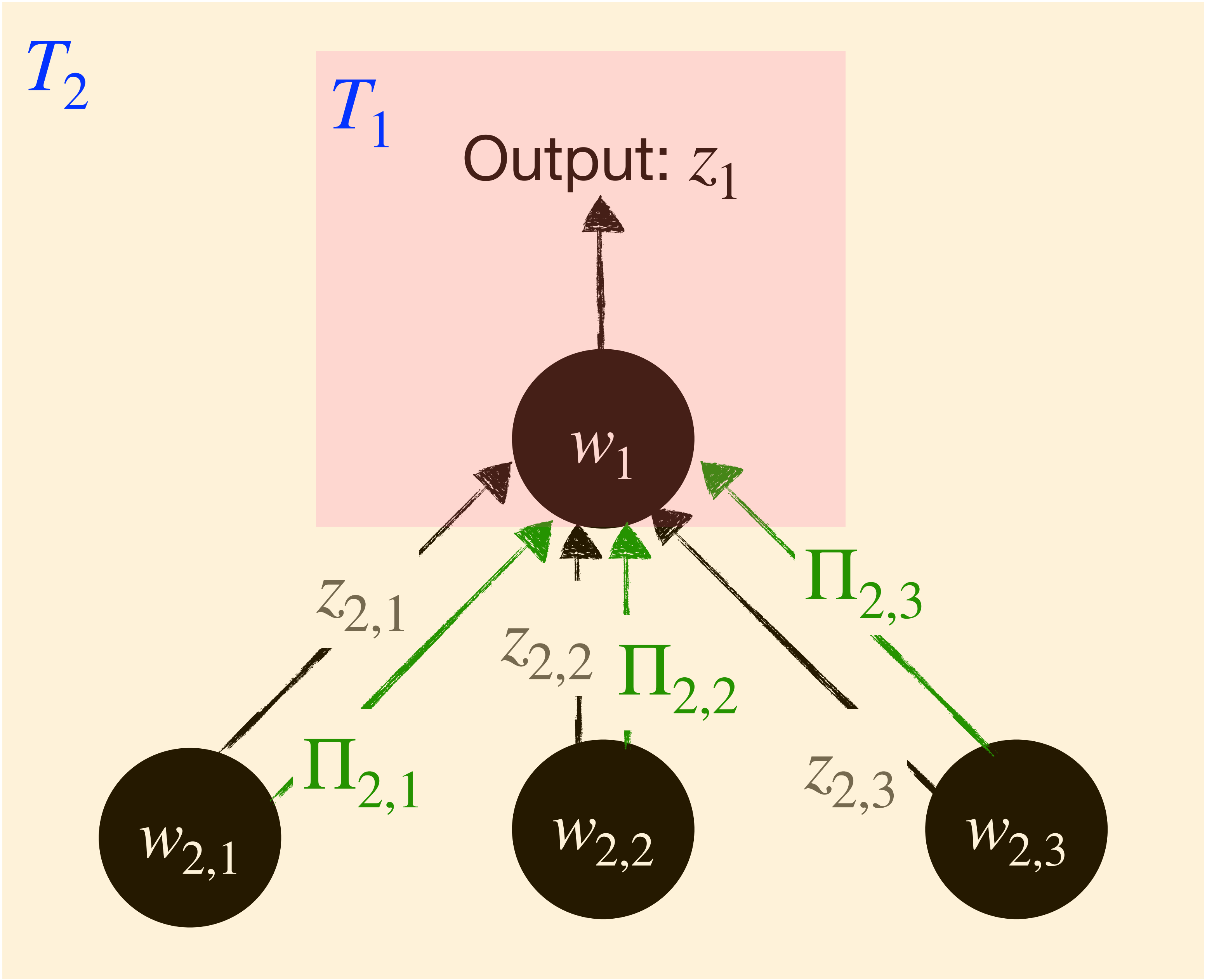
PCD formalizes the recursive proof composition of a SNARK:
- PCD prover and verifier invoke SNARK prover and verifier (for CSAT) for the recursive circuit C.





Canonical security analysis of PCD





Non-black-box knowledge soundness is problematic: size of extractor grows too quickly.

Size of extractor

- $|\tilde{P}_i| = |\mathbb{E}_{i-1}| + O(m^i) \Longrightarrow |E_{\tilde{P}_i}| = \mathrm{t}_E(|\tilde{P}_i|)$
- $|\mathbb{E}_i| \leq |E_{\tilde{P}_i}| + O(m^i)$
- $\mathrm{t}_E : n \mapsto n^c \Longrightarrow |\mathbb{E}_{\tilde{\mathbb{P}}}| = O\left(|\tilde{\mathbb{P}}|^{c^{\mathsf{D}}}\right)$

$\Longrightarrow |\mathbb{E}_{\tilde{\mathbb{P}}}|$ is polynomial only when D is constant.

Finding a better analysis remains a MAJOR open problem in this area.

Today: focus on PCD based on SNARKs with "strong" extraction.

Our result

Theorem. We prove a significantly improved security bound for PCD based on SNARKs with **straightline extraction**:

SNARK for CSAT with straightline extraction → (Recursive proof composition) → PCD with straightline extraction

$\kappa(\lambda, \mathsf{q}, \mathsf{D}, \mathsf{N}) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N})$

Prior works

SNARK for CSAT → (Recursive proof composition) → PCD

$\kappa(\lambda, \mathsf{q}, \mathsf{D}, \mathsf{N}) \leq \exp(\mathsf{D}) \cdot \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N})$

No security when D is larger than constant.

In practice, SNARKs have non-black-box knowledge soundness.
Straightline extraction only exists in idealized models.
How can we apply our theorem in practice then?

Applications

Application 1 [main].

- We propose a new idealization of hash-based PCD used in practice as a "PCD" in the ROM.
- We apply our theorem: $\kappa(\lambda, \mathsf{q}, \mathsf{D}, \mathsf{N}) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N}) = \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q})$.
- First justification for current choice of parameters of hash-based PCD in practice! [Polygon, Sharp]

Application 2.

- [CT10]: SNARK with straightline extraction in the SROM (*signed random oracle model*).
- Their bound: $\kappa(\lambda, \mathsf{q}, \mathsf{D}, \mathsf{N}) \leq \mathsf{N} \cdot \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N})$.
- Our bound: $\kappa(\lambda, \mathsf{q}, \mathsf{D}, \mathsf{N}) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N})$.

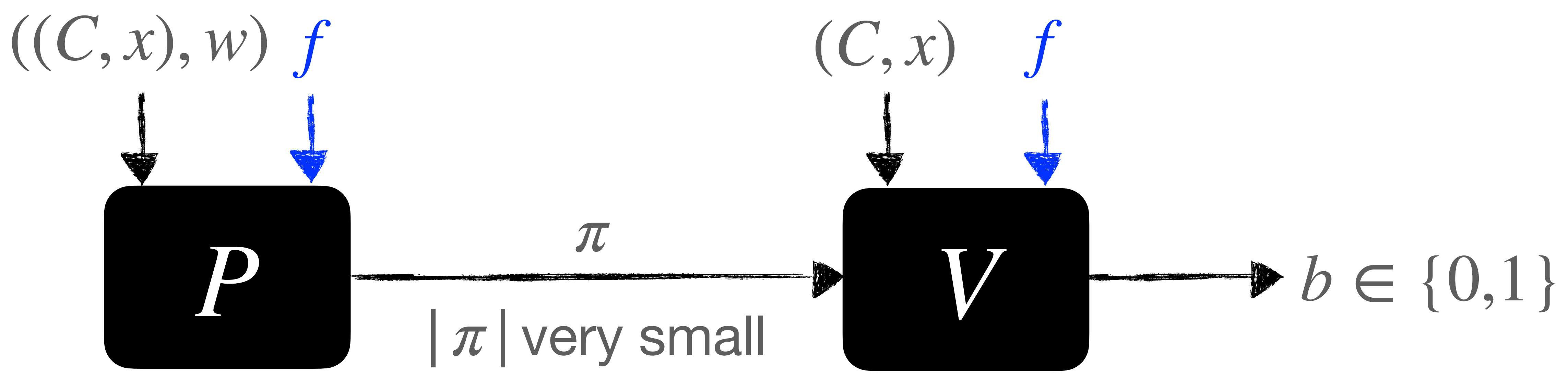
Application 3.

- [CCGOS23]: SNARK with straightline extraction in the AROM (*arithmetized random oracle model*).
- Their bound: $\kappa(\lambda, \mathsf{q}, \mathsf{D}, \mathsf{N}) \leq \mathsf{N} \cdot \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N})$.
- Our bound: $\kappa(\lambda, \mathsf{q}, \mathsf{D}, \mathsf{N}) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N})$.

Recursive proof composition with straightline extraction

SNARKs with straightline extraction

SNARKs in an oracle model (e.g. ROM):

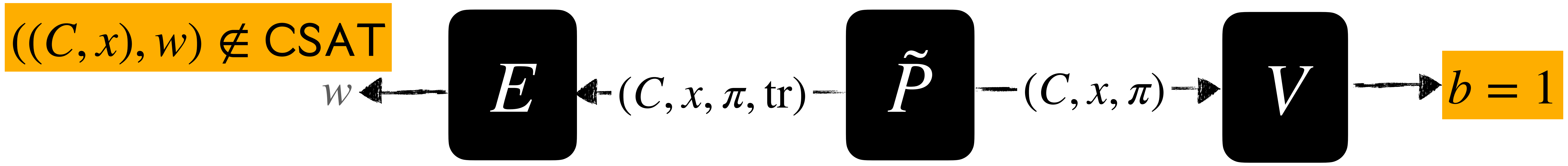


Straightline knowledge soundness: $\exists$ a deterministic extractor E such that $\forall$ bounded adversary $\tilde{P}$,

$$\Pr\left[\begin{array}{l|r} ((C,x),w) \notin \mathsf{CSAT} & f \leftarrow U(\lambda) \\ \wedge\, V^f(C,x,\pi)=1 & (C,x,\pi) \overset{\mathrm{tr}}{\leftarrow} \tilde{P}^f \\ & w \leftarrow E(C,x,\pi,\mathrm{tr}) \end{array}\right] \leq \kappa_{\mathsf{ARG}}(\lambda,\mathsf{q}).$$

λ: security parameter

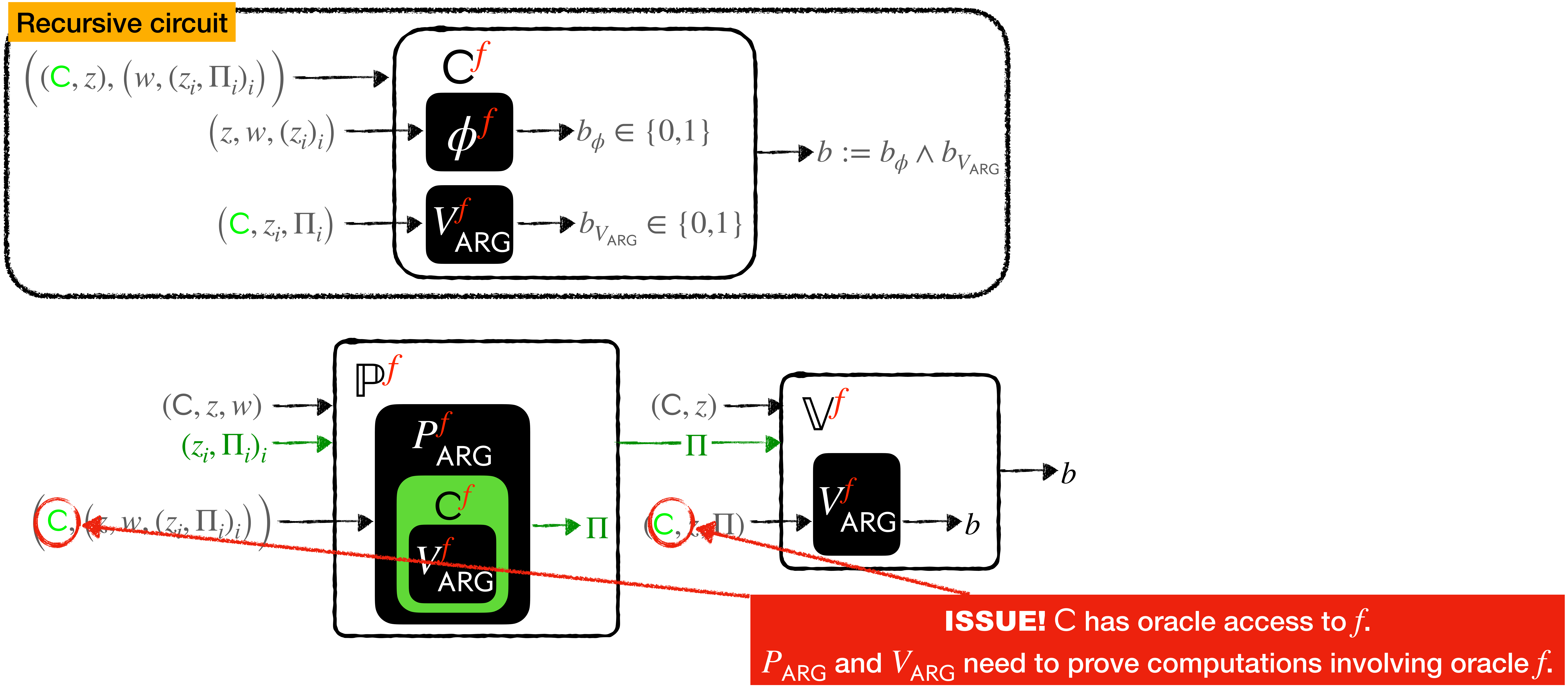
q: adversary query bound



Wonderful Fact: in the ROM (and other interesting oracle models) there are SNARKs of interest with straightline extraction!

(E.g., the Micali SNARK and BCS SNARK and related constructions.)

Can't we use the previous recursive composition?



Relativized SNARKs in an oracle model

We need SNARK in the oracle model that can prove/verify for oracle relations
- Relativized SNARK!

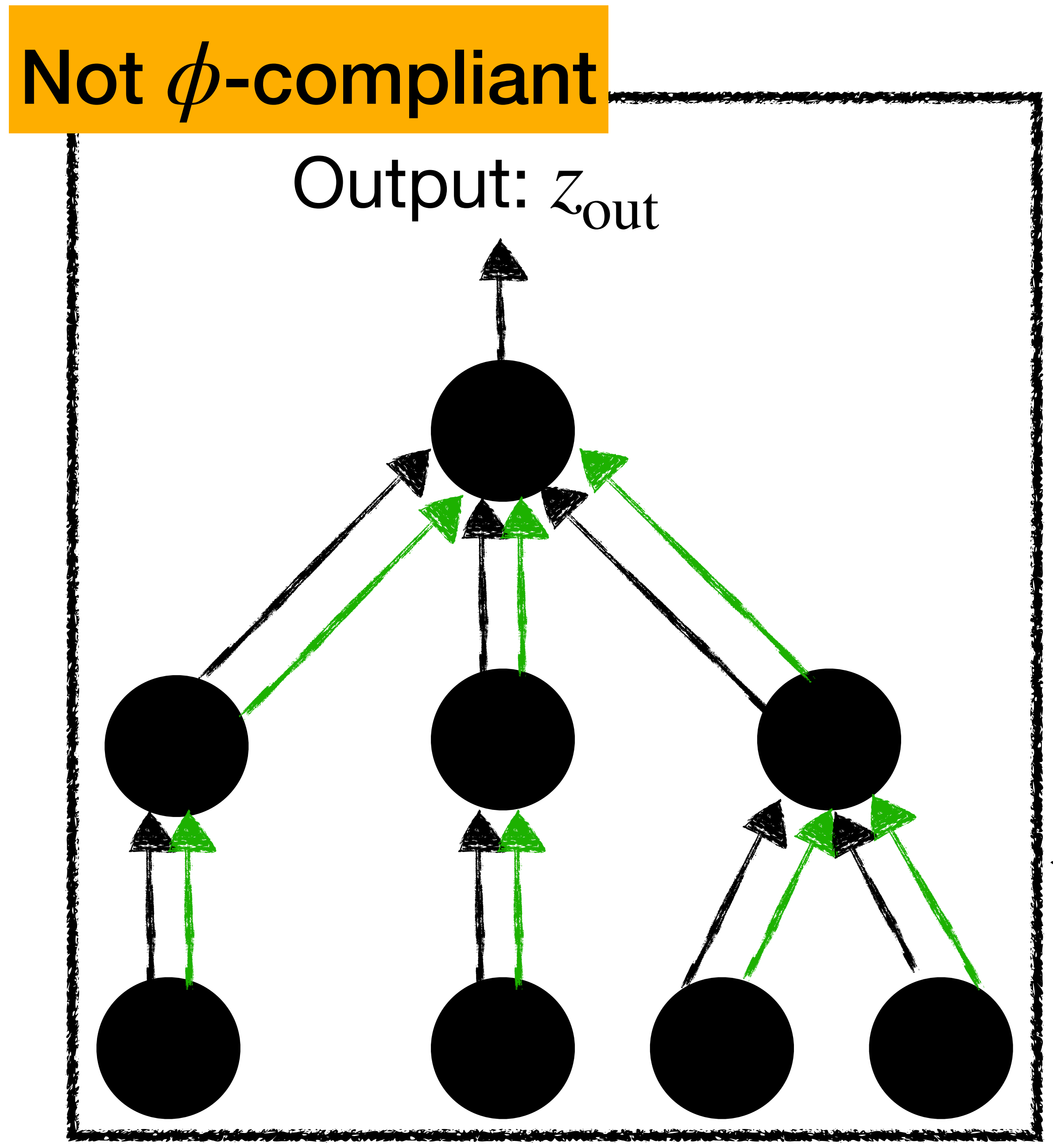
$\text{CSAT}^f := \{((C, x), w) : C^f(x, w) = 1\}$

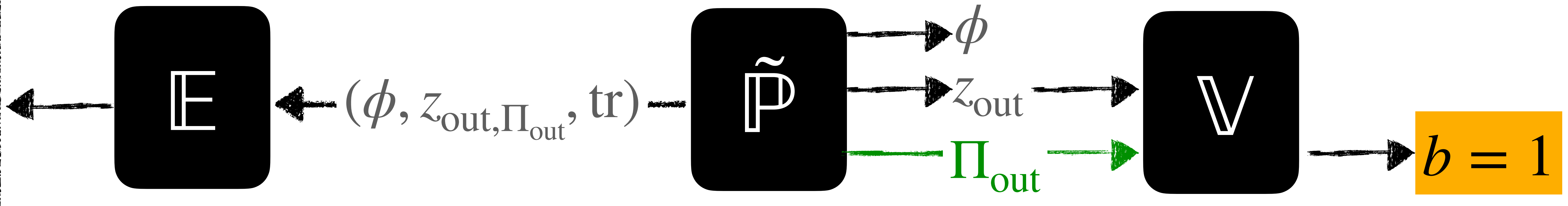
Relativized SNARK for CSAT^f —(Recursive proof composition)→ PCD with straightline extraction

PCD straightline knowledge soundness: $\exists$ a deterministic extractor $\mathbb{E}$ such that $\forall$ bounded adversary $\tilde{P}$,

$$\Pr\left[\begin{array}{l|r} \mathbb{V}^f(z_{\text{out}}, \Pi) = 1 & f \leftarrow U(\lambda) \\ \wedge\, T \text{ is not } \phi\text{-compatible} & (\phi, z_{\text{out}}, \Pi_{\text{out}}) \xleftarrow{\text{tr}} \tilde{\mathbb{P}}^f \\ & T \leftarrow \mathbb{E}(\phi, z_{\text{out}}, \Pi_{\text{out}}, \text{tr}) \end{array}\right] \leq \kappa(\lambda, \text{q}, \text{N}).$$

λ: security parameter
N: maximum transcript size
q: adversary query bound



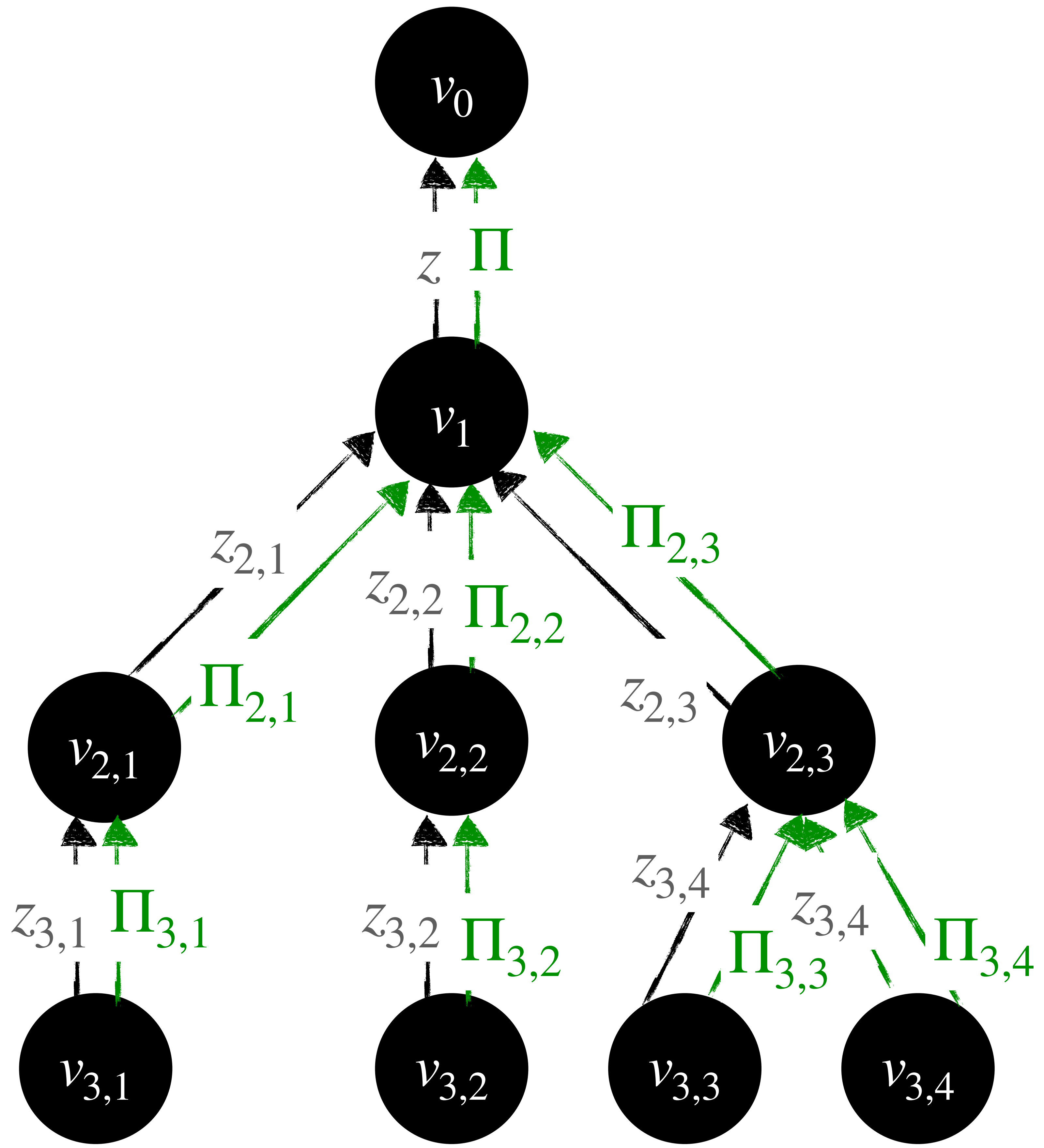


Concrete security of PCD with straightline extraction

Construction of the PCD extractor

In general, PCD extractor is constructed by repeatedly invoking SNARK extractor.

$(\phi, z, \Pi, \mathsf{tr}) \longrightarrow \mathbb{E} \longrightarrow T$



Extraction queue Q $\boxed{v_1}$

$(\mathsf{C}, z, \Pi, \mathsf{tr}) \longrightarrow E_{\mathsf{ARG}} \longrightarrow w_{v_1}$

Parse w_{v_1} as $(w_1, (z_{2,i}, \Pi_{(v_{2,i}, v_1)})_{i \in [3]})$

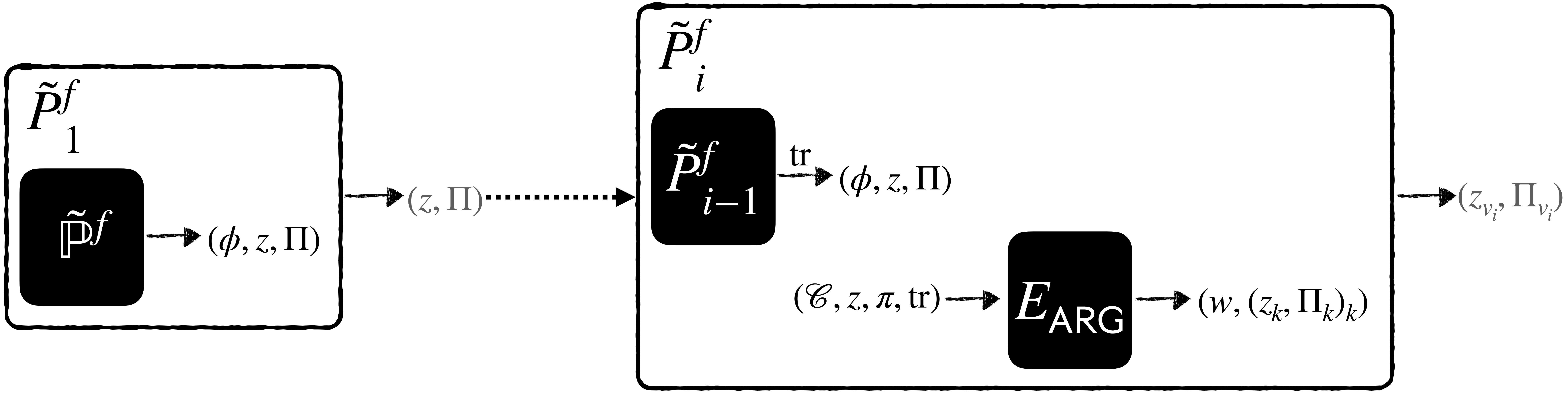
Extraction queue Q | $v_{2,1}$ | $v_{2,2}$ | $\cdots$ | $v_{2,m}$ |

$(\mathscr{C}, z_{2,1}, \Pi_{2,1}, \mathsf{tr}) \longrightarrow E_{\mathsf{ARG}} \longrightarrow w_{v_{2,1}}$

Security analysis in previous works

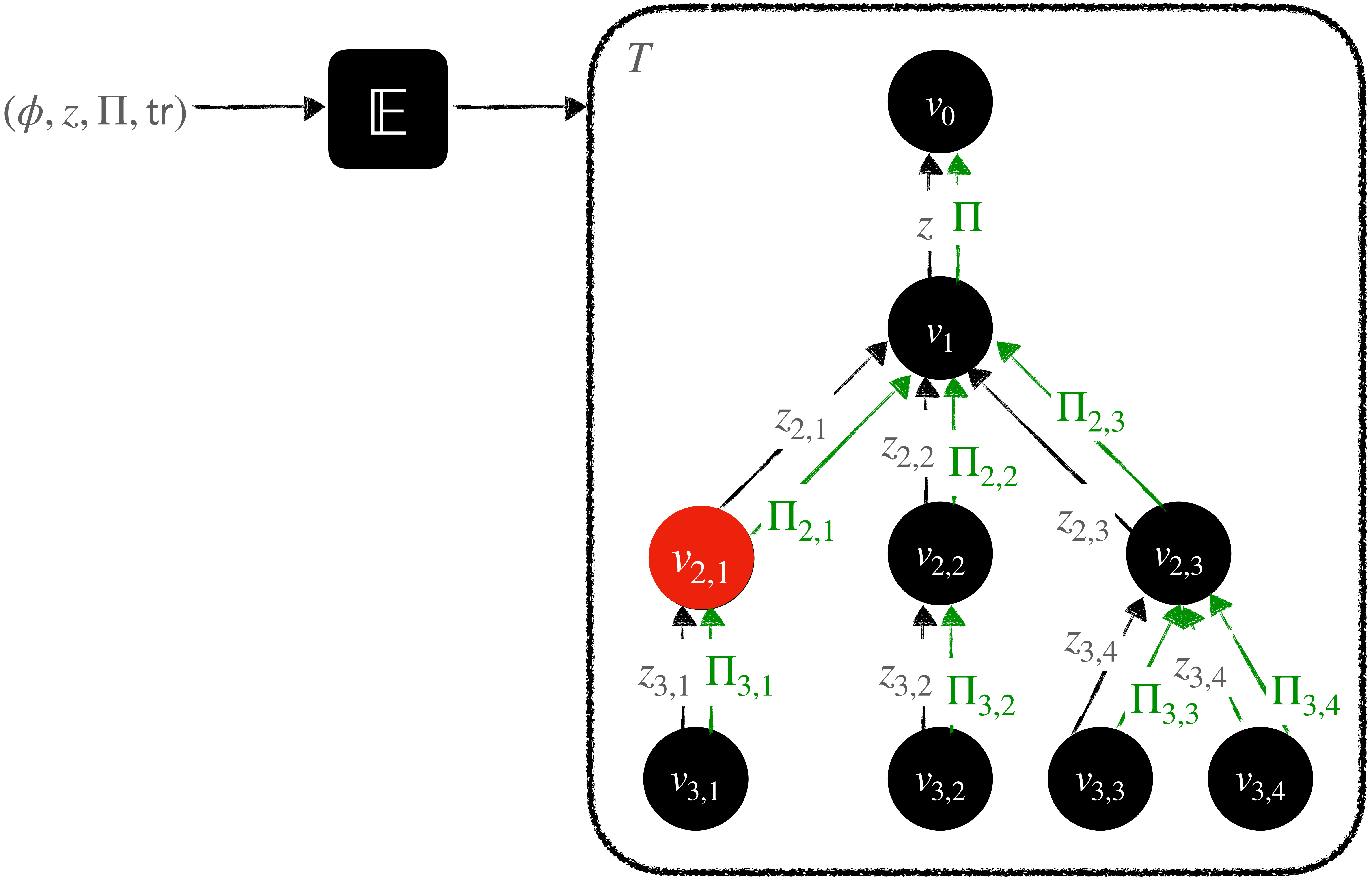
A natural analysis gives us this bound: $\kappa(\lambda, \mathsf{q}, \mathsf{N}) \leq \mathsf{N} \cdot \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N})$

- Each recursion pays the knowledge soundness error of the argument.
- The i-th extraction: invoking E_{ARG} for a corresponding argument prover $\tilde{P}_i$.



Warning: the actual construction of $\tilde{P}_i$ is more complicated. This is for intuitive explanation only.

Our security analysis [1/2]



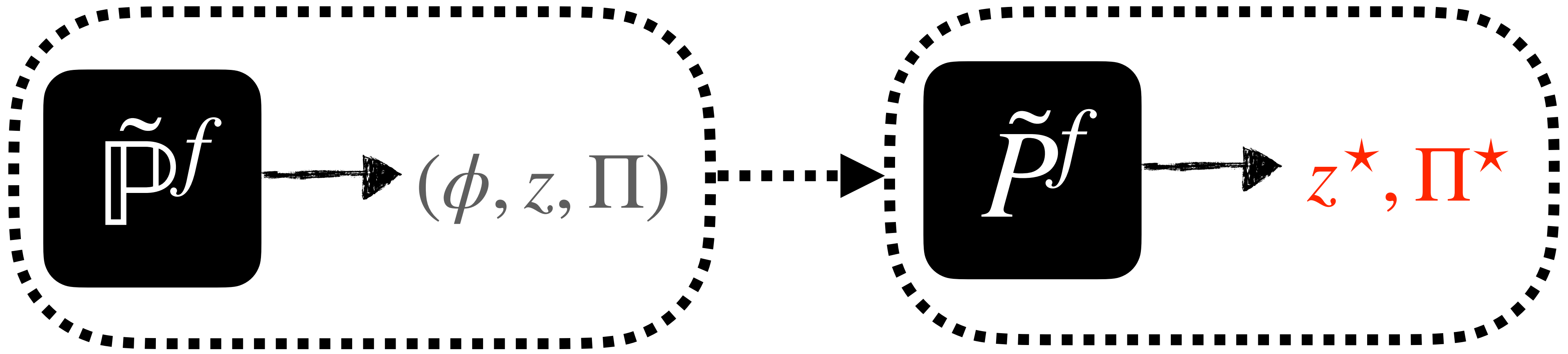
T not ϕ-compliant

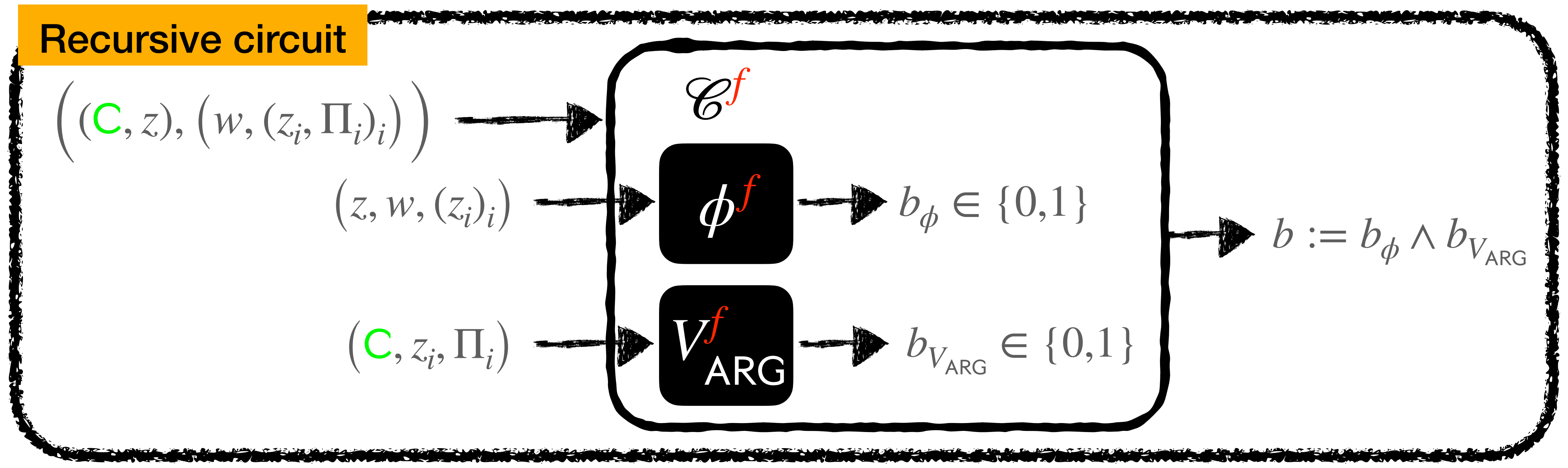
$\Longrightarrow$ There is one vertex in T that is not ϕ-compliant

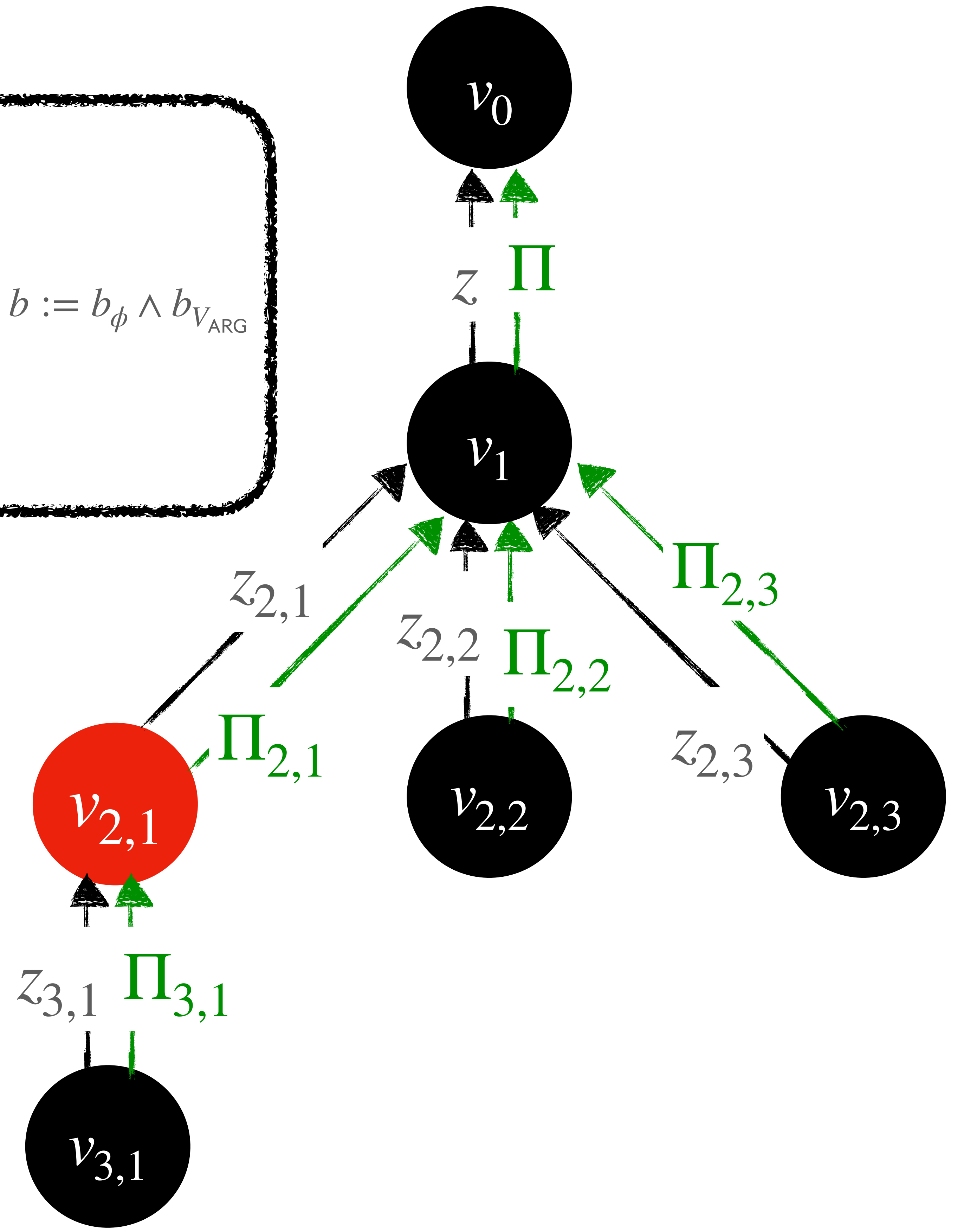
Find such vertex in one pass and output it

$\Longrightarrow \kappa(\lambda, \mathsf{q}, \mathsf{N}) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N})$.

Our security analysis [2/2]







Extraction queue Q | v_1 |

$(\mathscr{C}, z, \Pi, \mathsf{tr}) \rightarrow E_{\mathsf{ARG}} \rightarrow w_{v_1}$

Parse w_{v_1} as $(w_1, (z_{2,i}, \Pi_{(v_{2,i}, v_1)})_{i \in [3]})$

$\phi^f(z_1, w_1, (z_{2,i})_{i\in[3]}) \neq 1$ or
$\exists i \in [3]$ such that $V^f(z_{2,i}, \Pi_{(v_{2,i},v_1)}) \neq 1$?

Extraction queue Q | $v_{2,1}$ | $v_{2,2}$ | $\cdots$ | $v_{2,m}$ |

$(\mathscr{C}, z_{2,1}, \Pi_{2,1}, \mathsf{tr}) \rightarrow E_{\mathsf{ARG}} \rightarrow w_{v_{2,1}}$

Parse $w_{v_{2,1}}$ as $(w_1, (z_{3,1}, \Pi_{(v_{3,1}, v_{2,1})})$

$\phi^f(z_{2,1}, w_{2,1}, (z_{3,1})) \neq 1$ or
$V^f(z_{3,1}, \Pi_{(v_{3,1},v_{2,1})}) \neq 1$?

$(z^\star, \Pi^\star) := (z_{2,1}, \Pi_{2,1})$

$\mathsf{C}\left((\mathsf{C}, z^\star), \left(w_{2,1}, (z_{3,1}, \Pi_{3,1})\right)\right) \neq 1$

Yet $V^f(\mathsf{C}, z^\star, \Pi^\star) = 1$

Our theorem: $\kappa(\lambda, \mathsf{q}, \mathsf{D}, \mathsf{N}) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N})$

Application:
Set security for hash-based PCD

Warm-up: analyzing hash-based SNARKs

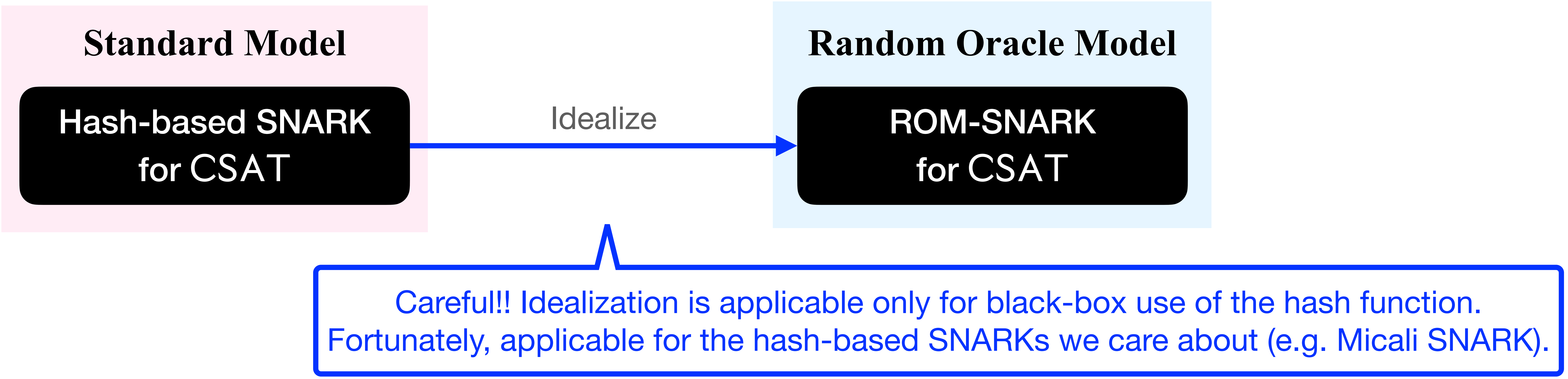
Three-step recipe:

Step 1. Model the hash function as "ideal": a random function.

- the hash-based SNARK is idealized as **a SNARK in the random oracle model (ROM-SNARK)**.

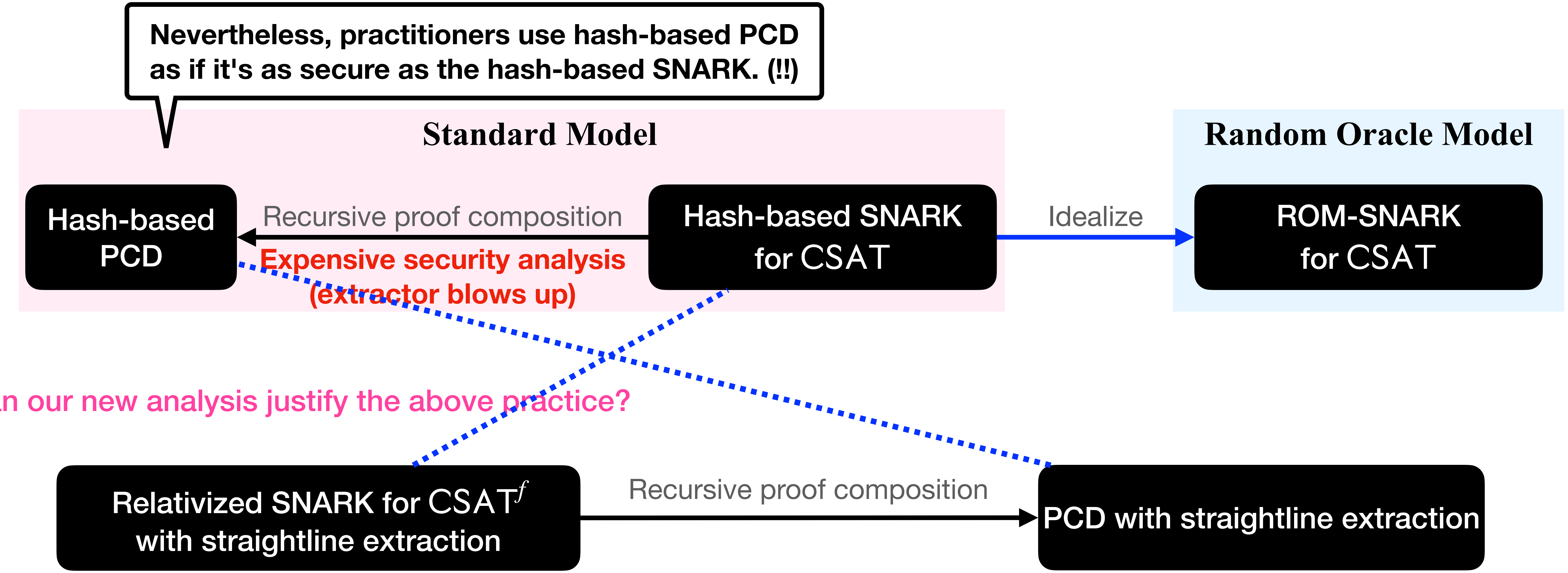
Step 2. Establish **concrete** security bounds for the ROM-SNARK.

Step 3. Set security parameters of the hash-based SNARK accordingly.



First attempt for idealization of hash-based PCD

PCDs are deployed based on various approaches. A popular approach is **hash-based PCD**.



Our theorem: $\kappa(\lambda, \mathsf{q}, \mathsf{D}, \mathsf{N}) \le \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N})$

Second attempt for idealization of hash-based PCD

Idealization is applicable only for black-box use of the hash function - not true in general.

What we hope to do

Hash-based PCD → (Idealize) → PCD in the ROM ← (Recursive proof composition) ← Relativized SNARK in the ROM with straightline extraction

Our theorem: $\kappa(\lambda, q, D, N) \leq \kappa_{ARG}(\lambda, q, N)$

Reality

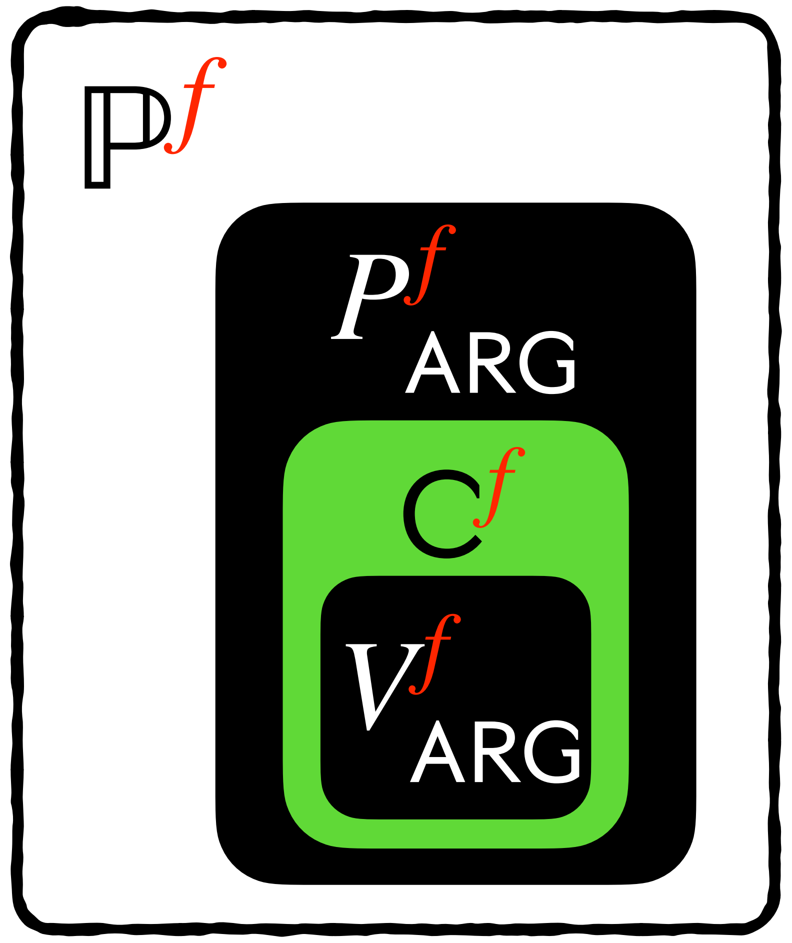
Hash-based PCD → (Idealize) → PCD in the ROM ← (Recursive proof composition) ← Relativized SNARK in the ROM with straightline extraction

Not believed to exist! [Val08, HN23] (PCD in the ROM)

Not believed to exist! [CL20] (Relativized SNARK in the ROM with straightline extraction)

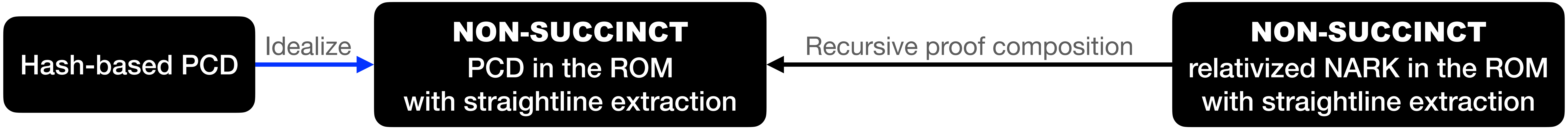
Can't apply — Our theorem: $\kappa(\lambda, q, D, N) \leq \kappa_{ARG}(\lambda, q, N)$ ☹

Our idealization for hash-based PCD



Issue: Hash-based PCD uses hash function in a non-black-box way.
Observation 1: PCD looks at hash function to check the correctness, it doesn't "destroy" the hash function.
Observation 2: C is an oracle circuit because V_{ARG} make oracle queries.
Solution: Forward all the queries of C by asking P_{ARG} to attach C's "query-answer trace" in the proof.

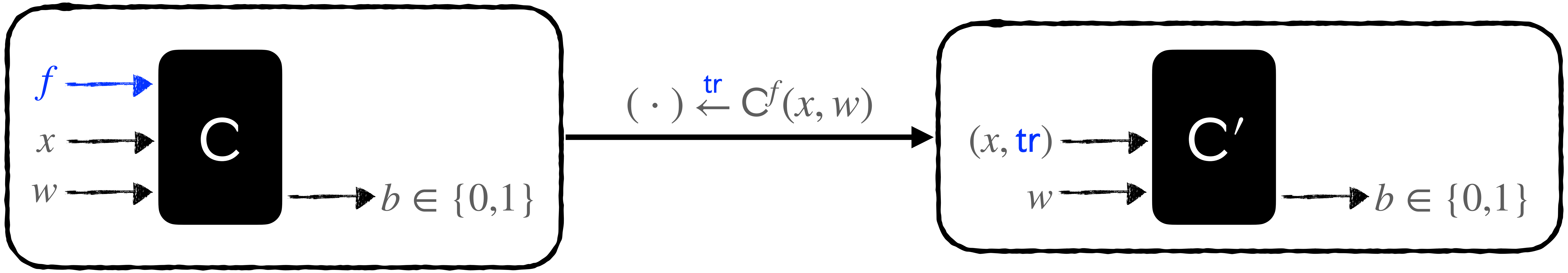
Forwarding the queries makes the proof non-succinct

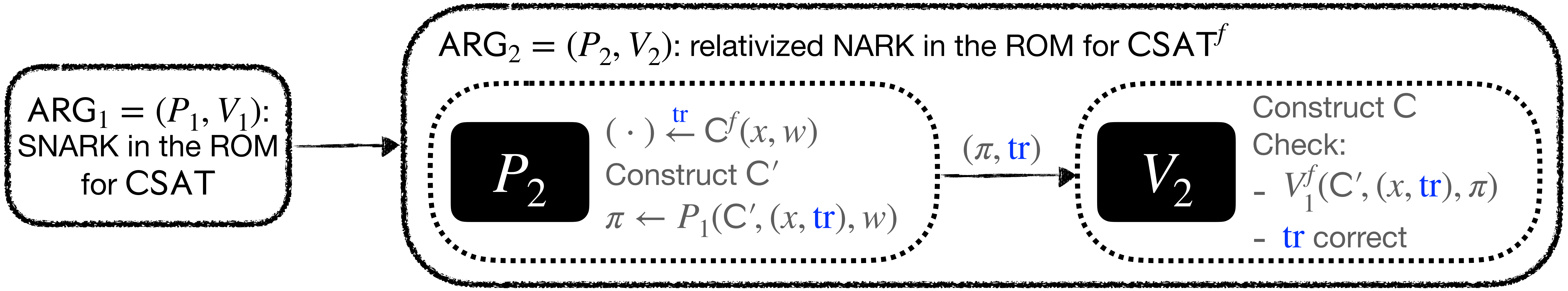


Our theorem: $\kappa(\lambda, \mathsf{q}, \mathsf{D}, \mathsf{N}) \leq \kappa_{\text{ARG}}(\lambda, \mathsf{q}, \mathsf{N}) = \kappa_{\text{ARG}}(\lambda, \mathsf{q})$

Last step: relativized ROM-NARK

Idea: Given an oracle circuit, remove its oracle gate by attaching its "query-answer trace" to instance.





TL;DR

What is *proof-carrying data* (PCD)?
- Recursive compositions of SNARKs.
- It's useful for efficiently verifying distributed computations.

Problem:
- PCD is deployed under the assumption "security of PCD" = "security of underlying SNARK".
- BUT existing security analyses show a huge gap in security ("PCD is far less secure than underlying SNARK").

This work:
- We propose **an idealized PCD** that models hash-based PCD in practice.
- We prove that this idealized PCD is **as secure as its underlying SNARK**.

Thank you!

https://eprint.iacr.org/2023/1646

Technical extension: Probabilistic straightline extraction

Probabilistic straightline extraction

Probabilistic straightline knowledge soundness for SNARKs:
$\exists$ a probabilistic extractor E such that $\forall$ bounded adversary $\tilde{P}$,

$$\Pr\left[\begin{array}{l|r} ((C,x),w) \notin \mathsf{CSAT}^f & f \leftarrow U(\lambda) \\ \wedge\, V^f(C,x,\pi) = 1 & (C,x,\pi) \overset{\mathrm{tr}}{\leftarrow} \tilde{P}^f \\ & w \leftarrow E(C,x,\pi,\mathrm{tr}) \end{array}\right] \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}).$$

λ: security parameter
q: adversary query bound

PCD probabilistic straightline knowledge soundness: $\exists$ a probabilistic extractor $\mathbb{E}$ such that $\forall$ bounded adversary $\tilde{P}$,

$$\Pr\left[\begin{array}{l|r} & f \leftarrow U(\lambda) \\ \mathbb{V}^f(z_{\mathrm{out}}, \Pi) = 1 & (\phi, z_{\mathrm{out}}, \Pi_{\mathrm{out}}) \overset{\mathrm{tr}}{\leftarrow} \tilde{\mathbb{P}}^f \\ \wedge\, T \text{ is not } \phi\text{-compatible} & T \leftarrow \mathbb{E}(\phi, z_{\mathrm{out}}, \Pi_{\mathrm{out}}, \mathrm{tr}) \end{array}\right] \leq \kappa(\lambda, \mathsf{q}, \mathsf{N}).$$

λ: security parameter
N: maximum transcript size
q: adversary query bound

Our security analysis

Theorem. We prove an improved security bound even for PCD based on SNARKs with **probabilistic straightline extraction**:

SNARK for CSAT with probabilistic straightline extraction → (Recursive proof composition) → **PCD with probabilistic straightline extraction**

$$\kappa(\lambda, \mathsf{q}, \mathsf{D}, \mathsf{N}) \leq \mathsf{N} \cdot \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N})$$

The multiplicative factor N is tight:

- With probabilistic straightline extraction, at each node, $\mathbb{E}$ pays for both the extraction error and the randomness error of E_{ARG}.
- If let ϵ be the randomness error of E_{ARG}, it's possible to show:

$$\kappa(\lambda, \mathsf{q}, \mathsf{D}, \mathsf{N}) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N}) + \mathsf{N} \cdot \epsilon.$$

Application:
Improved concrete security for black-box PCD constructions

PCD in the SROM

- Signed random oracle model (SROM):
  - On input x, samples a random answer y, generates a signature σ on (x, y), and outputs (y, σ).
  - Repeated inputs have the same answer.
- [CT10]: SNARK in the ROM $\rightarrow$ SNARK in the SROM (preserves straightline extraction)
  - The argument verifier doesn't need to query the oracle: verify σ is enough.
  - [CT10] gives a bound $\kappa(\lambda, \mathsf{q}, \mathsf{N}) \leq \mathsf{N} \cdot \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N})$.
  - Our analysis improves it to $\kappa(\lambda, \mathsf{q}, \mathsf{N}) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N})$.

PCD in the AROM

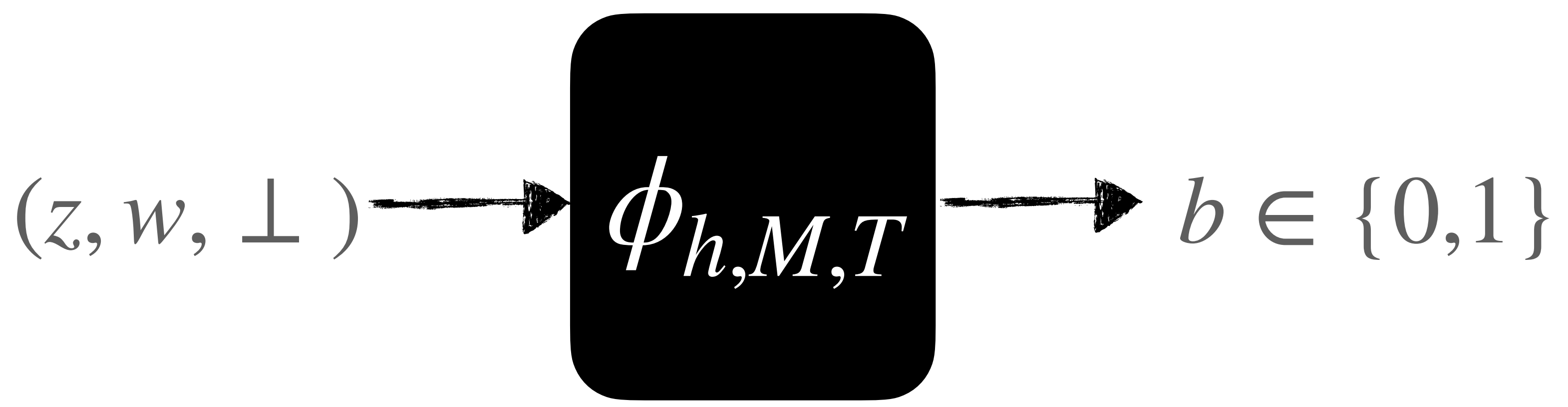
- Arithmetized random oracle model (AROM):
  - A random oracle: idealization of a concrete hash function h;
  - An arithmetization oracle: idealization of a low degree polynomial that encodes the circuit of h.
- [CCGOS22]: SNARK in the ROM → SNARK in the AROM (preserves straightline extraction)
  - Queries in the AROM can be accumulated.
  - [CCGOS22] gives a bound $\kappa(\lambda, \mathsf{q}, \mathsf{N}) \leq \mathsf{N} \cdot \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N})$.
  - Our analysis improves it to $\kappa(\lambda, \mathsf{q}, \mathsf{N}) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{N})$.

Example:
Real-world compliance predicate with unbounded transcript size

A real-world compliance predicate

- $h : \{0,1\}^* \to \{0,1\}^\lambda$, a collision resistant hash function.
- M: a universal Turing machine. On input a program P and an input x, $M(P, x)$ outputs $P(x)$.
- $T \in \mathbb{N}$ a maximum time bound.

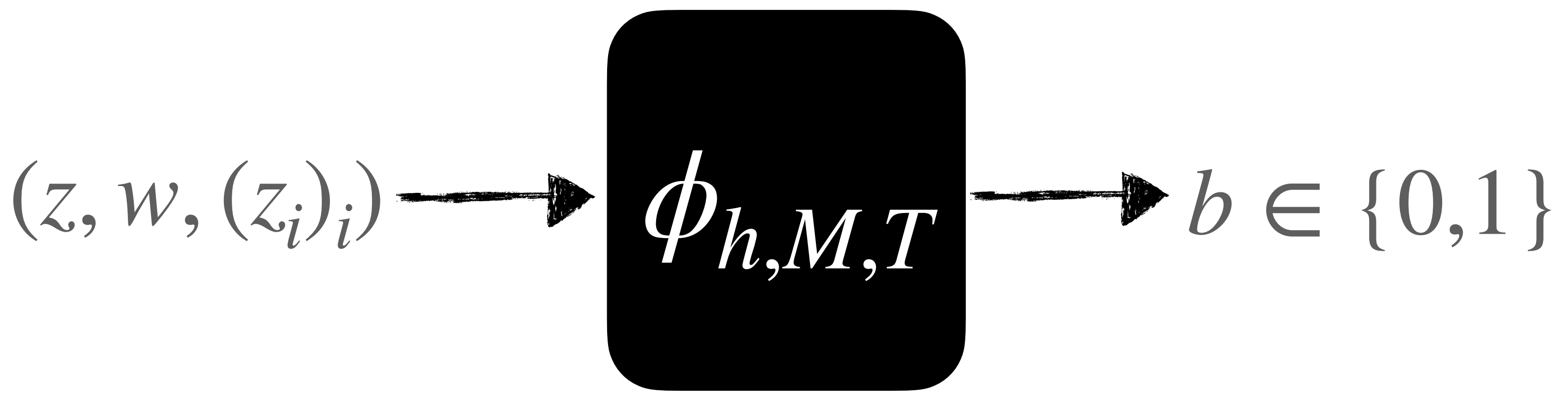
Base case.



Parse z as (y, t)
Parse w as (P, x)

$$b := \begin{pmatrix} t \leq T \wedge M(P, x) = y \\ \wedge\ M(P, x) \text{ runs in } t \text{ steps} \end{pmatrix}$$

Recursive case.



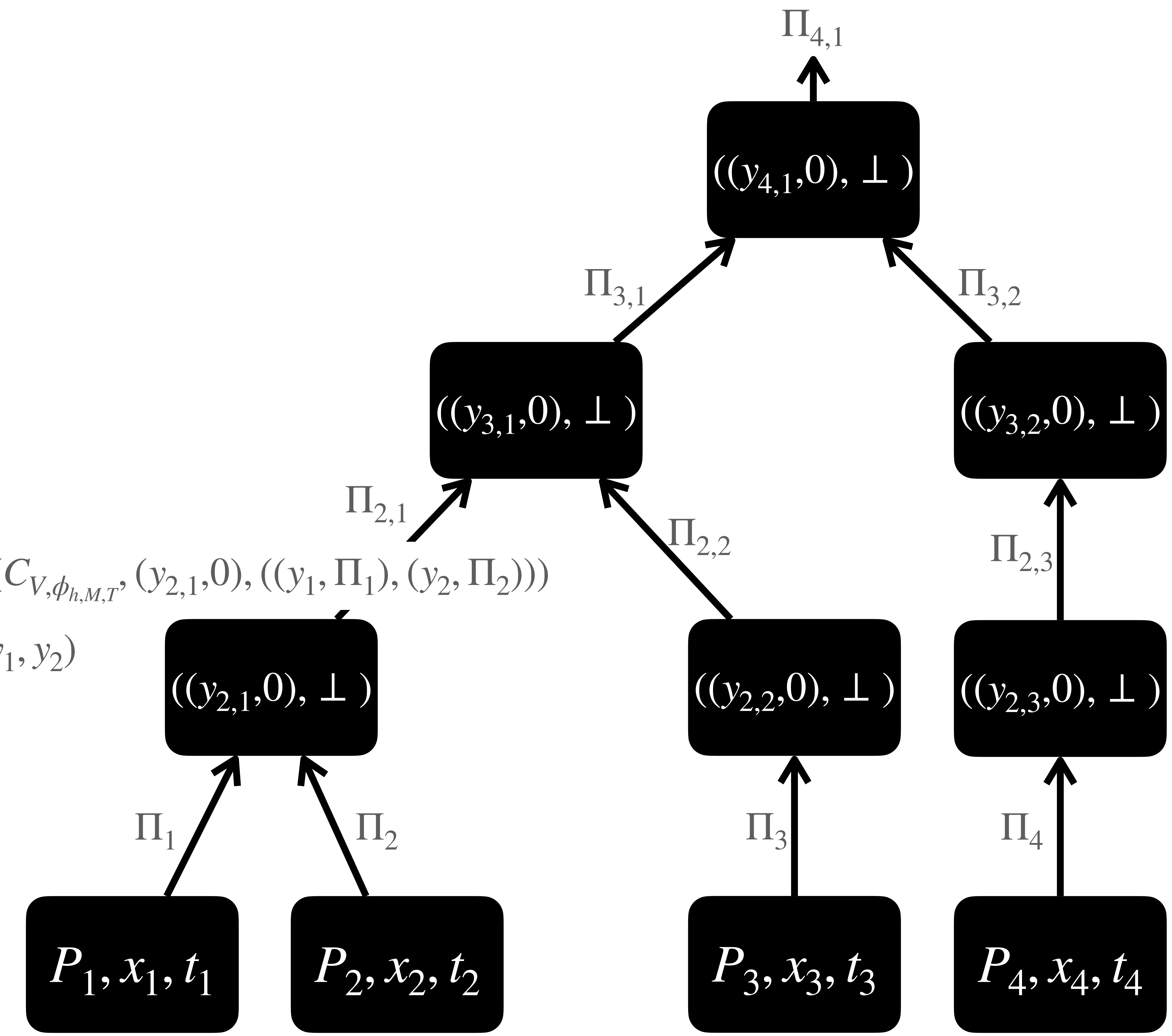
Parse z as (y, t)
Parse z_i as (y_i, t_i) for each i

$$b := \begin{pmatrix} t = 0 \wedge w = \bot \\ \wedge\ \forall i, t_i \leq T \wedge h((y_i)_i) = y \end{pmatrix}$$

No restriction on the size of the transcript!

- N can be arbitrarily large $\implies$ prior works can not guarantee security.
- Our result shows that security of the underlying SNARK is inherited by the PCD without loss.

Recursive STARKs



$\Pi_{2,1} \leftarrow \mathbb{P}(C_{V,\phi_{h,M,T}}, (y_{2,1},0), ((y_1, \Pi_1), (y_2, \Pi_2)))$

$y_{2,1} := h(y_1, y_2)$

$y_1 := M(P_1, x_1)$

$\Pi_1 \leftarrow \mathbb{P}(C_{V,\phi_{h,M,T}}, (y_1, t_1), (P_1, x_1), \perp)$

- Computation in Ethereum smart contract is expensive:
  - Each computation step is re-executed by every node.
- Layer 2 proof-based rollups: move computation off-chain.
  - User sends computation requests to an aggregator.
  - Aggregator produces a SNARK proof about batch of computations.
  - Ethereum smart contract verifiers the SNARK proof and update states.
- Aggregator: PCD prover.
- Ethereum smart contract: PCD verifier.