Proof-Carrying Data From Arithmetized Random Oracles

Thank you for many of the slides!

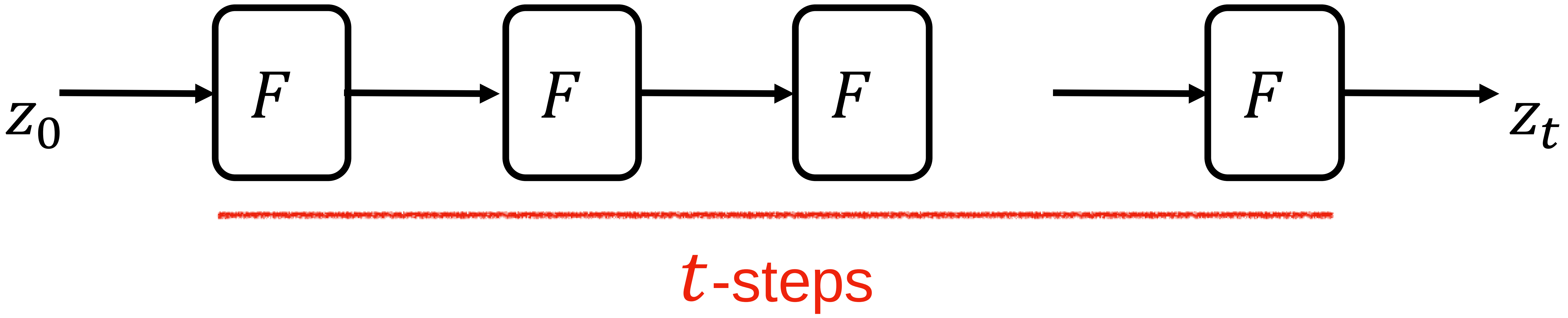
Jack O'Connor
Joint work with Megan Chen, Alessandro Chiesa, Tom Gur and Nicholas Spooner
Eurocrypt 2023

Our setting: “streaming” verification of t-step NP computations

Our setting: “streaming” verification of t-step NP computations

Goal: Prove correctness of a t-step non-deterministic computation:

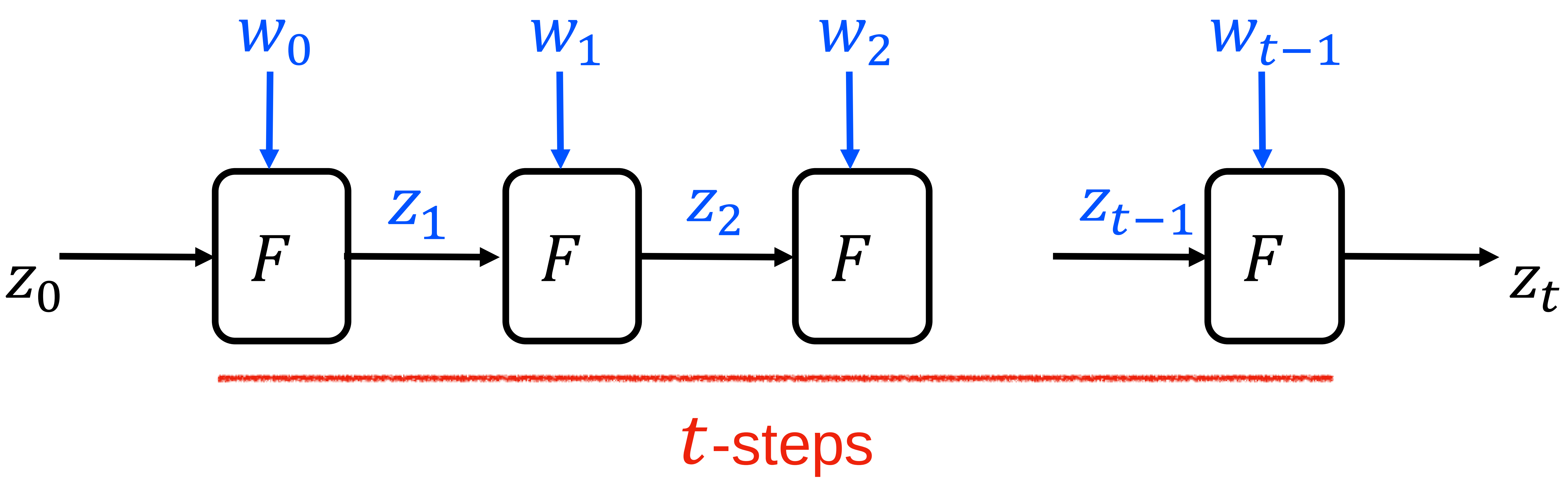
Given F, z_0, z_t,



Our setting: "streaming" verification of t-step NP computations

Goal: Prove correctness of a t-step non-deterministic computation:

Given F, z_0, z_t, check that $\exists z_1, \ldots, z_{t-1}, w_0, \ldots, w_{t-1}: \forall i \in [t], F(z_i, w_i) = z_{i+1}$

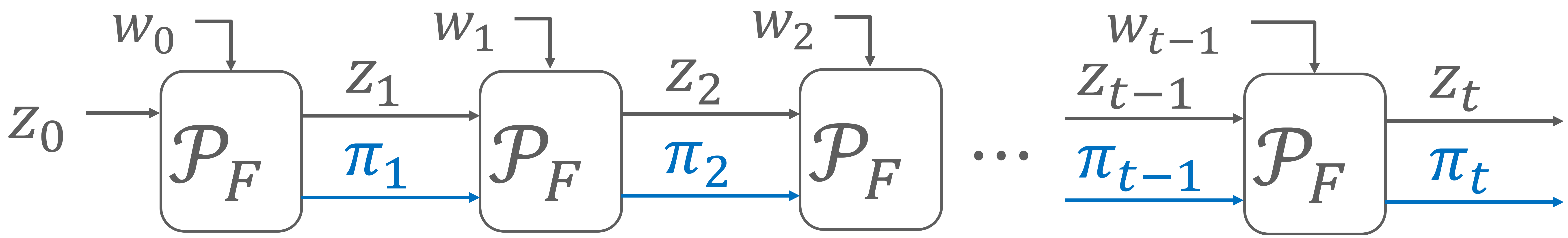


Our setting: “streaming” verification of t-step NP computations

Goal: Prove correctness of a t-step non-deterministic computation:

Given F, z_0, z_t, check that $\exists z_1, \dots, z_{t-1}, w_0, \dots, w_{t-1}: \forall i \in [t], F(z_i, w_i) = z_{i+1}$

Incrementally verifiable computation (IVC) [Valiant08]

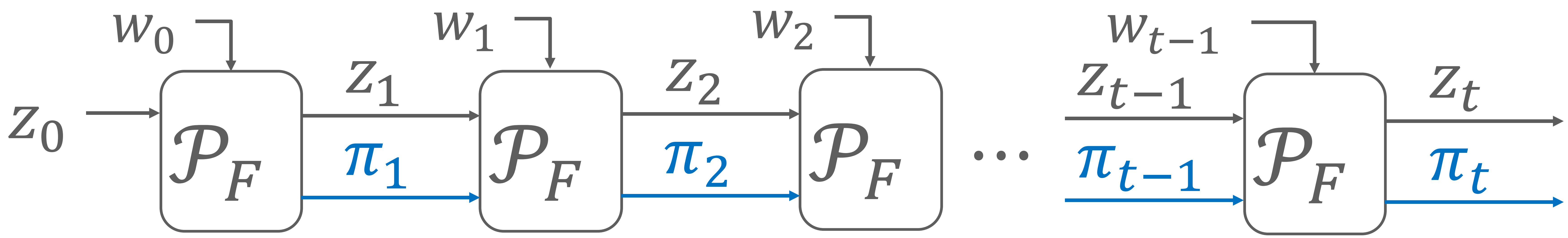


Our setting: "streaming" verification of t-step NP computations

Goal: Prove correctness of a t-step non-deterministic computation:

Given F, z_0, z_t, check that $\exists z_1, \ldots, z_{t-1}, w_0, \ldots, w_{t-1}: \forall i \in [t], F(z_i, w_i) = z_{i+1}$

Incrementally verifiable computation (IVC) [Valiant08]



Proof-carrying Data (PCD) [CT10, BCCT13] generalizes path graph to DAG.

IVC from SNARK

SNARK = Succinct Non-interactive ARgument of Knowledge

IVC from SNARK

SNARK = Succinct Non-interactive ARgument of Knowledge

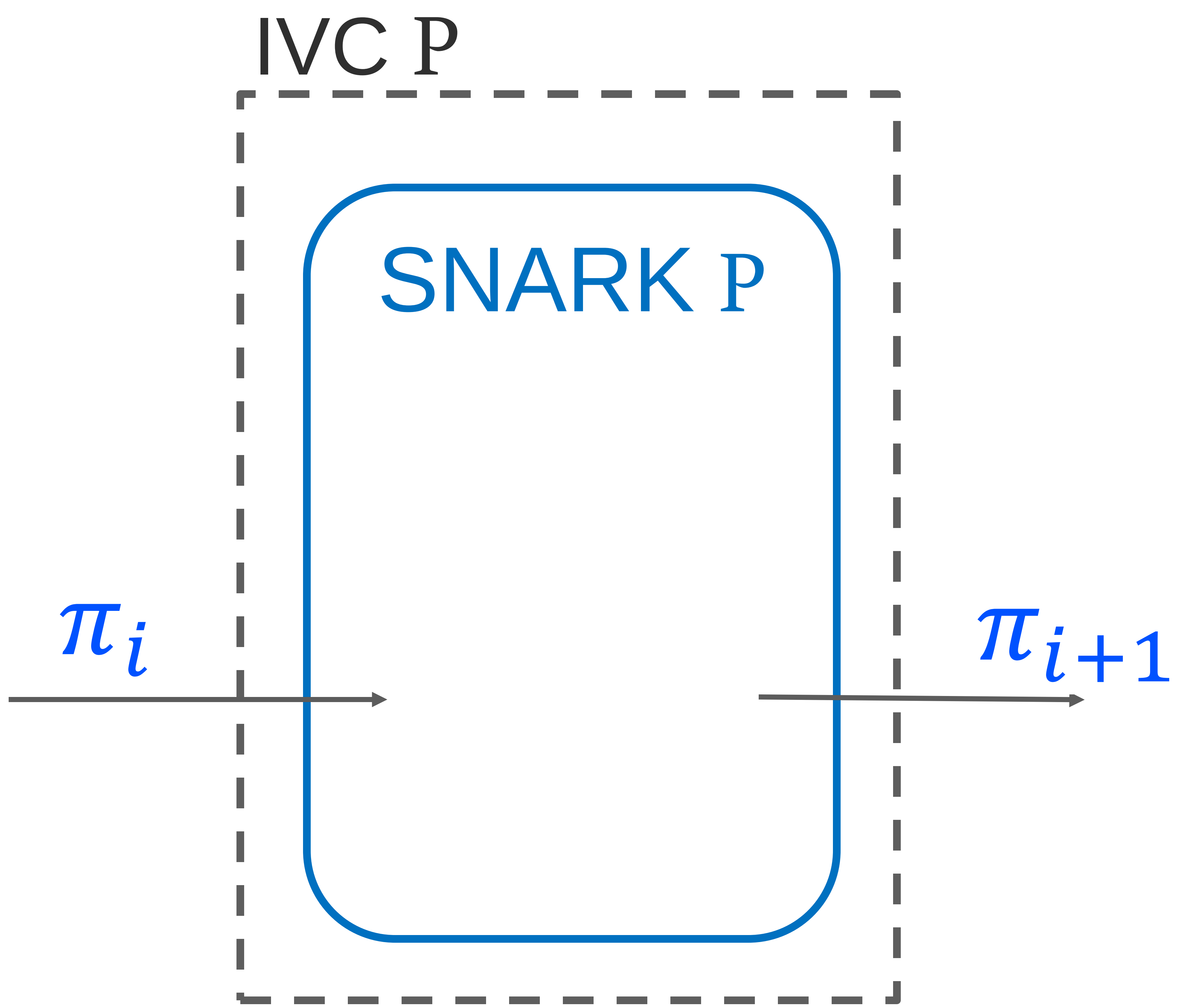
IVC P

IVC from SNARK

SNARK = Succinct Non-interactive ARgument of Knowledge

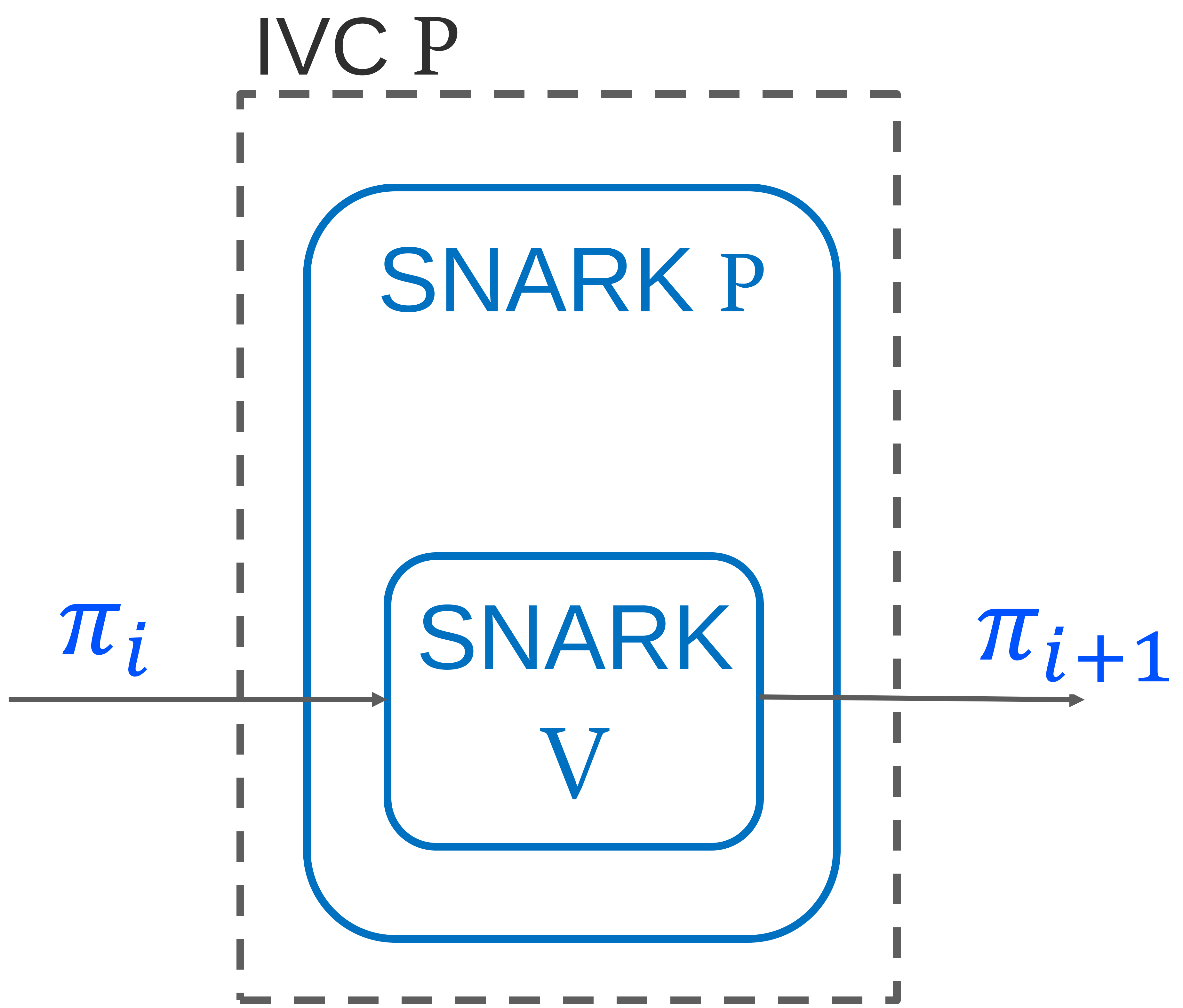
IVC from SNARK

SNARK = Succinct Non-interactive ARgument of Knowledge



IVC from SNARK

SNARK = Succinct Non-interactive ARgument of Knowledge

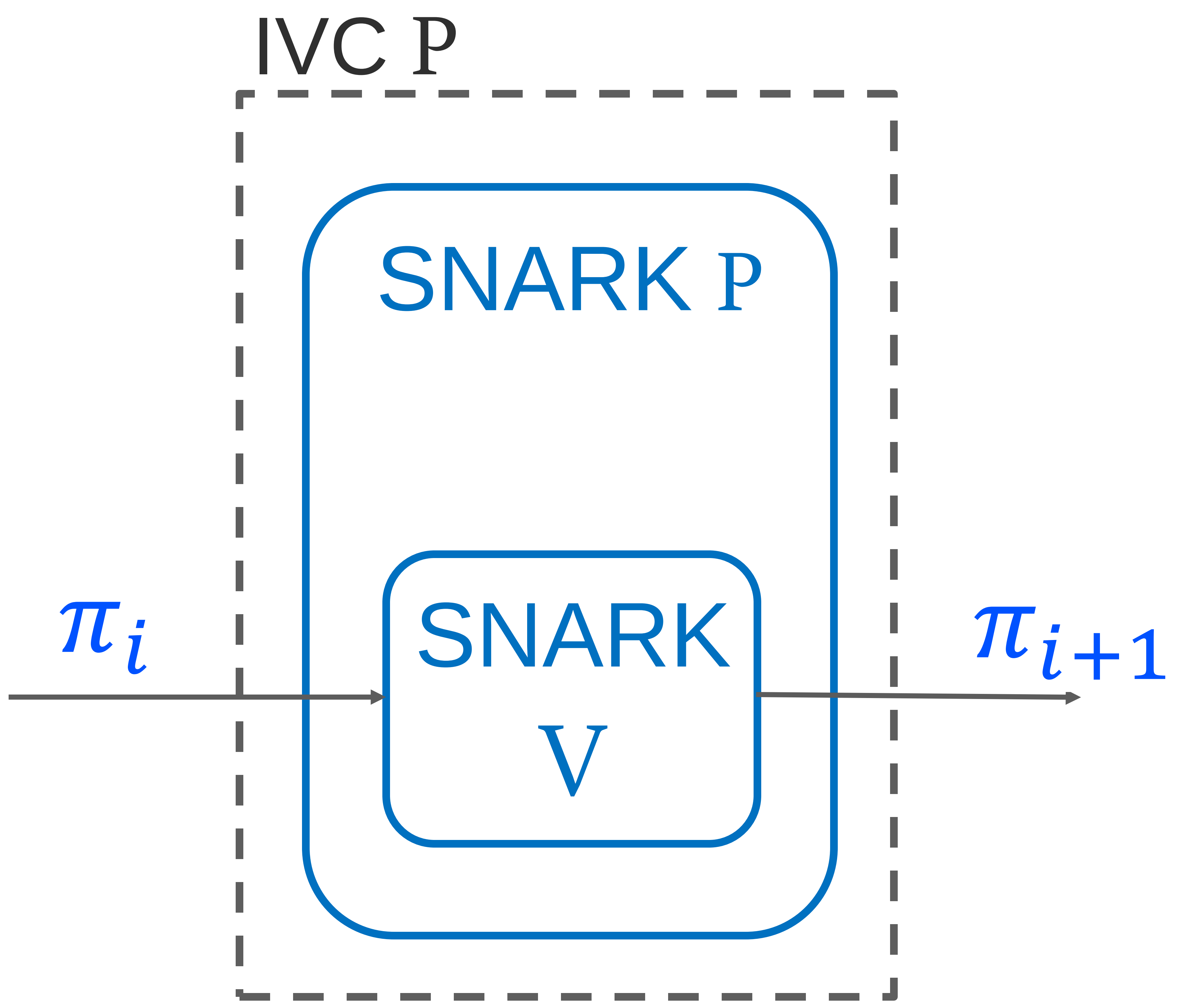


Prior works: SNARKs

SNARK = Succinct Non-interactive ARgument of Knowledge

Approach 1: CRS + knowledge (extraction) assumptions

[Groth10; GennaroGPR13; BitanskyCIOP13; Ben-SassonCTV14; BitanskyCCGLRT14; Groth16; GrothKMMM18]



Prior works: SNARKs

SNARK = Succinct Non-interactive ARgument of Knowledge

Approach 1: CRS + knowledge (extraction) assumptions

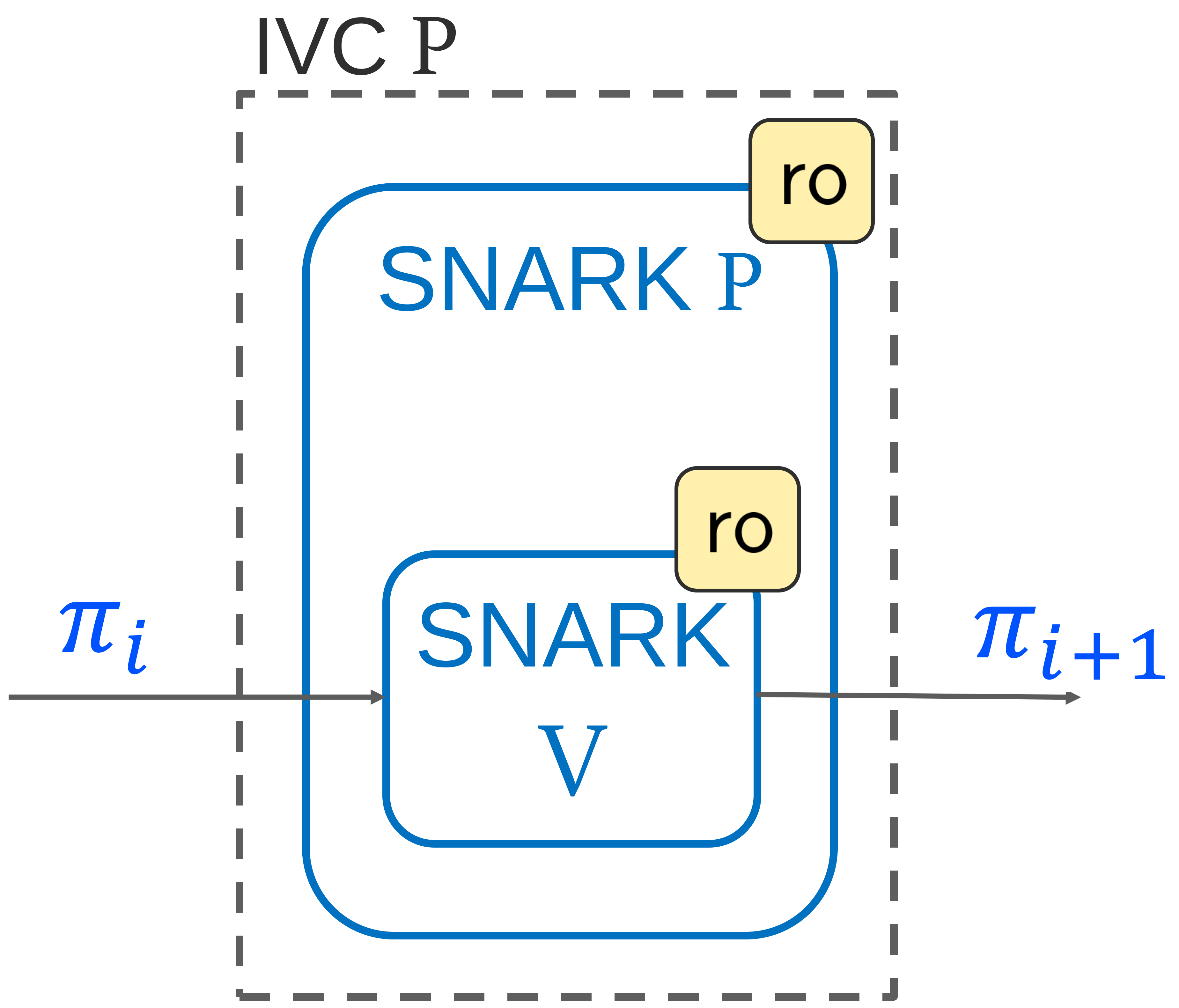
[Groth10; GennaroGPR13; BitanskyCIOP13; Ben-SassonCTV14; BitanskyCCGLRT14; Groth16; GrothKMMM18]

Approach 2: SNARKs in ROM

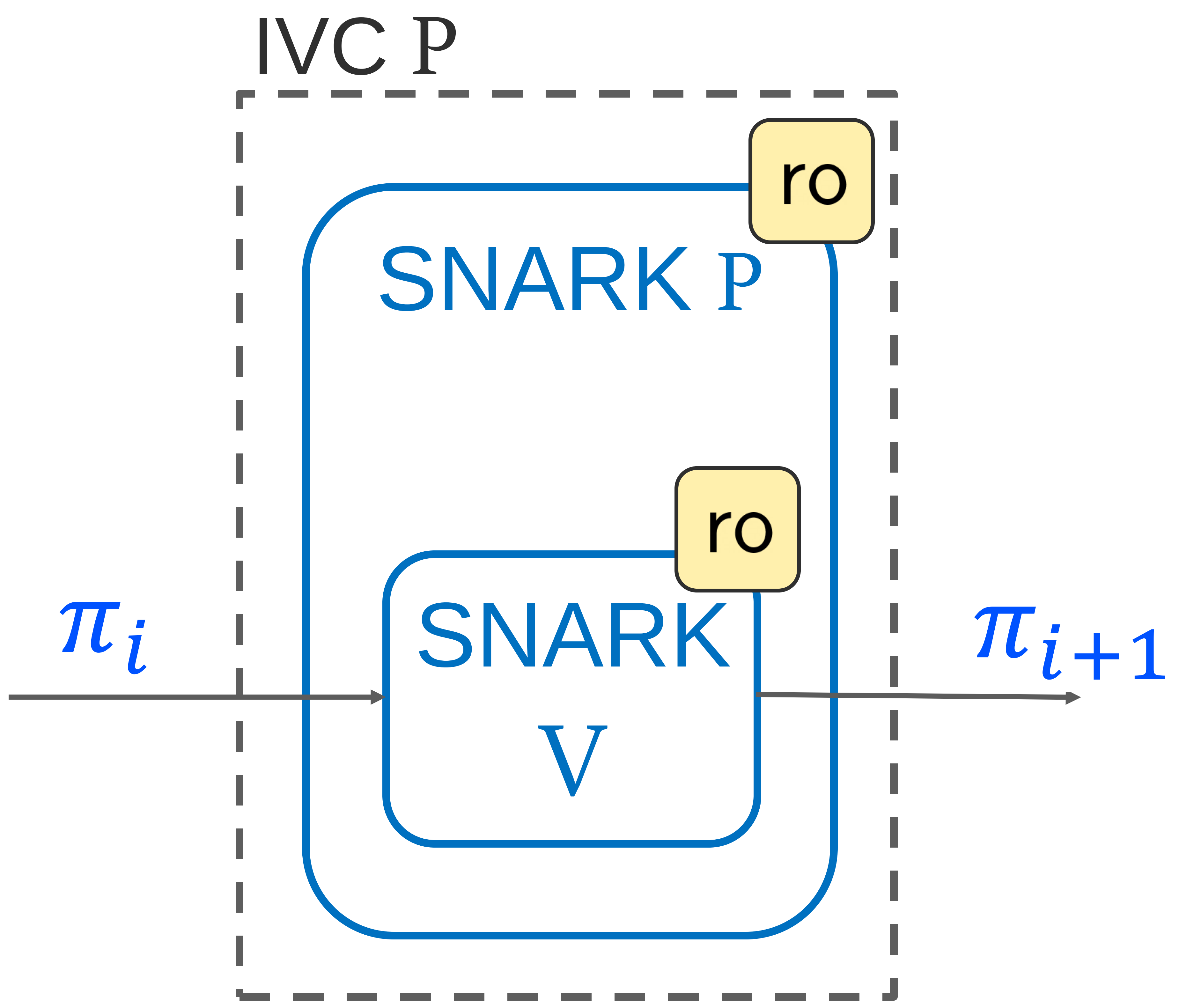
[Micali00; Ben-SassonCS16; ChiesaOS20; ChiesaHMMVW20]

Benefits:

- Transparent / universal setup
- Efficiency improvements

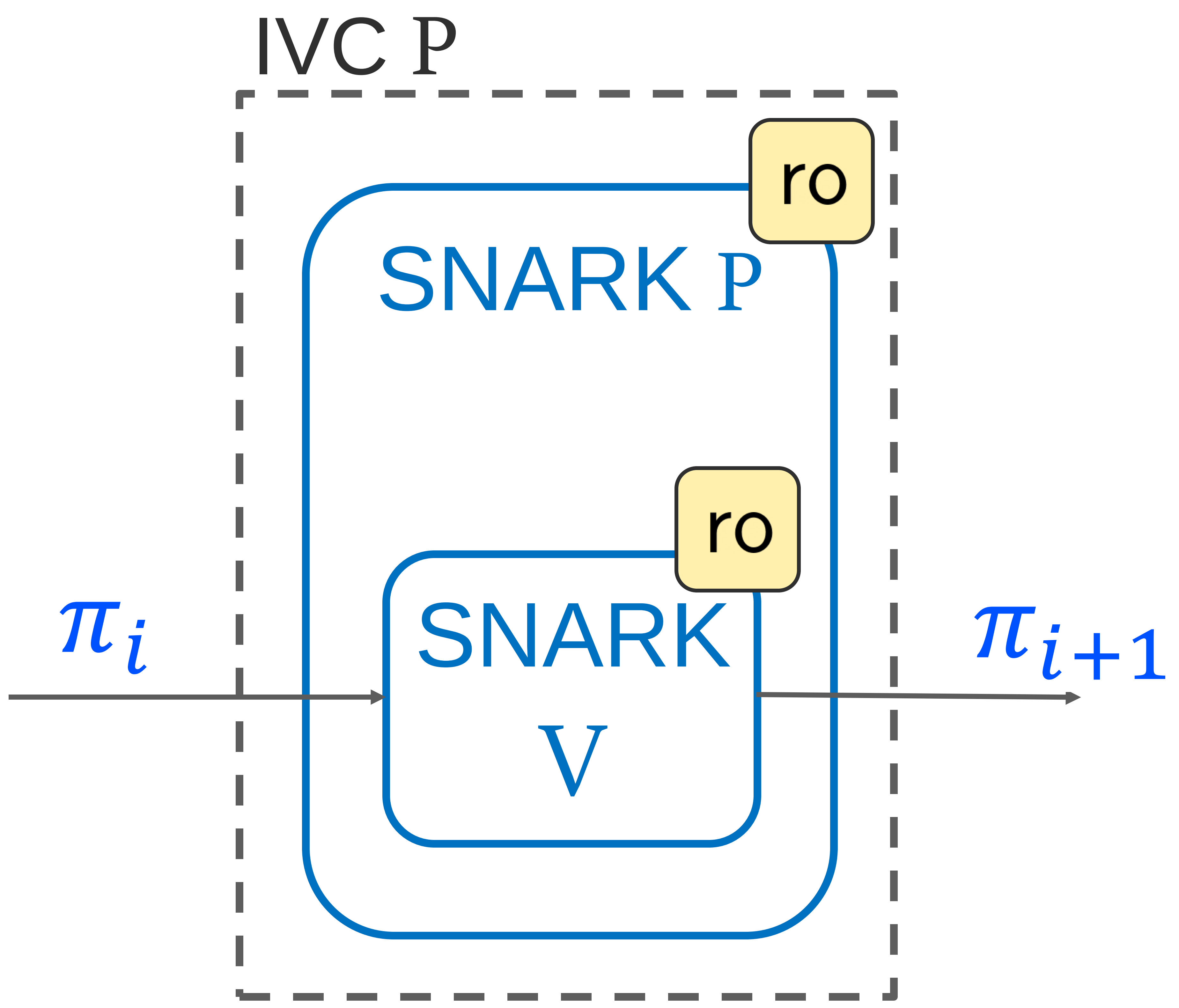


Issues with IVC from SNARKs in the ROM



Issues with IVC from SNARKs in the ROM

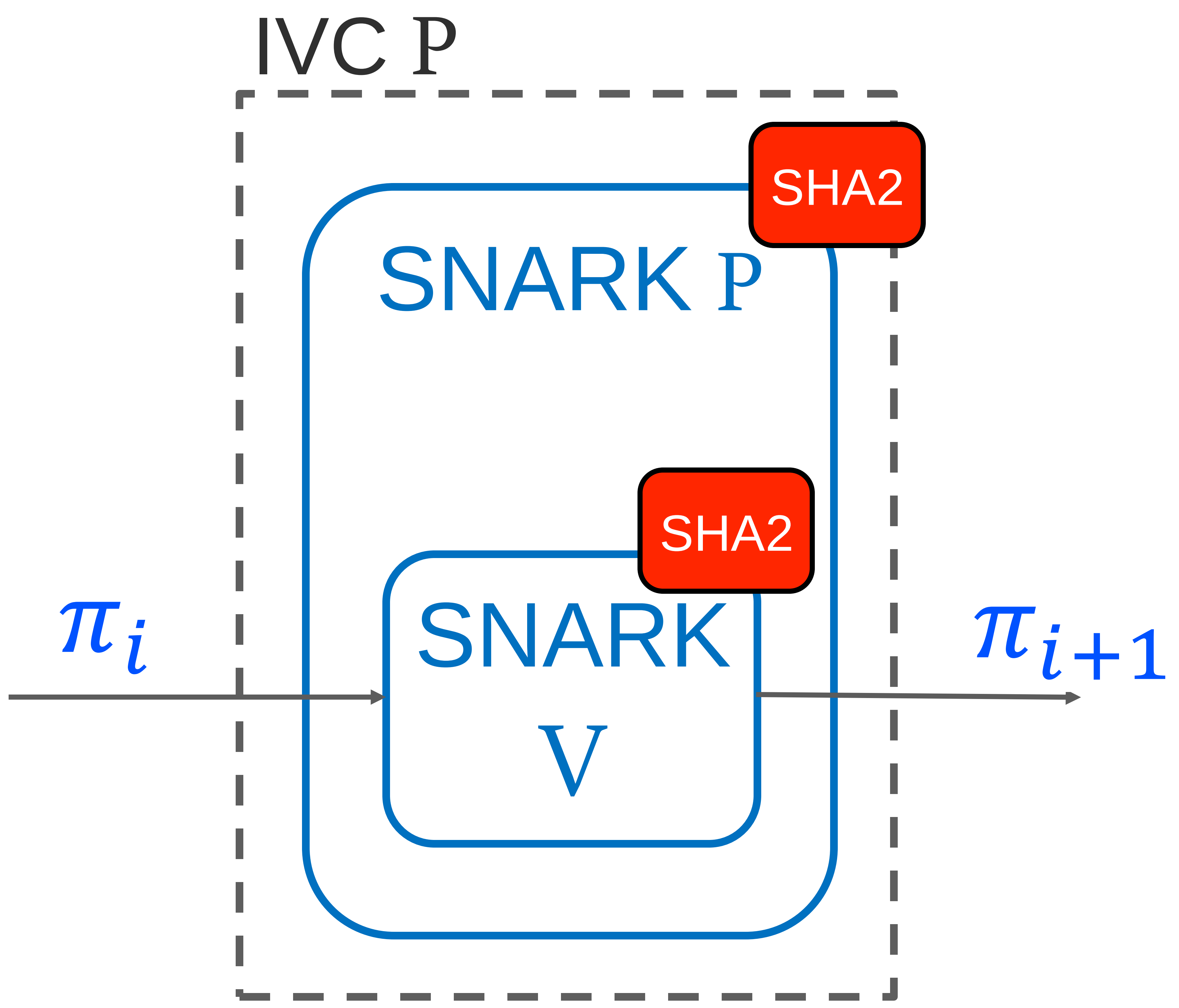
- SNARK verifier makes oracle queries, but SNARK is for non-oracle computations.



Issues with IVC from SNARKs in the ROM

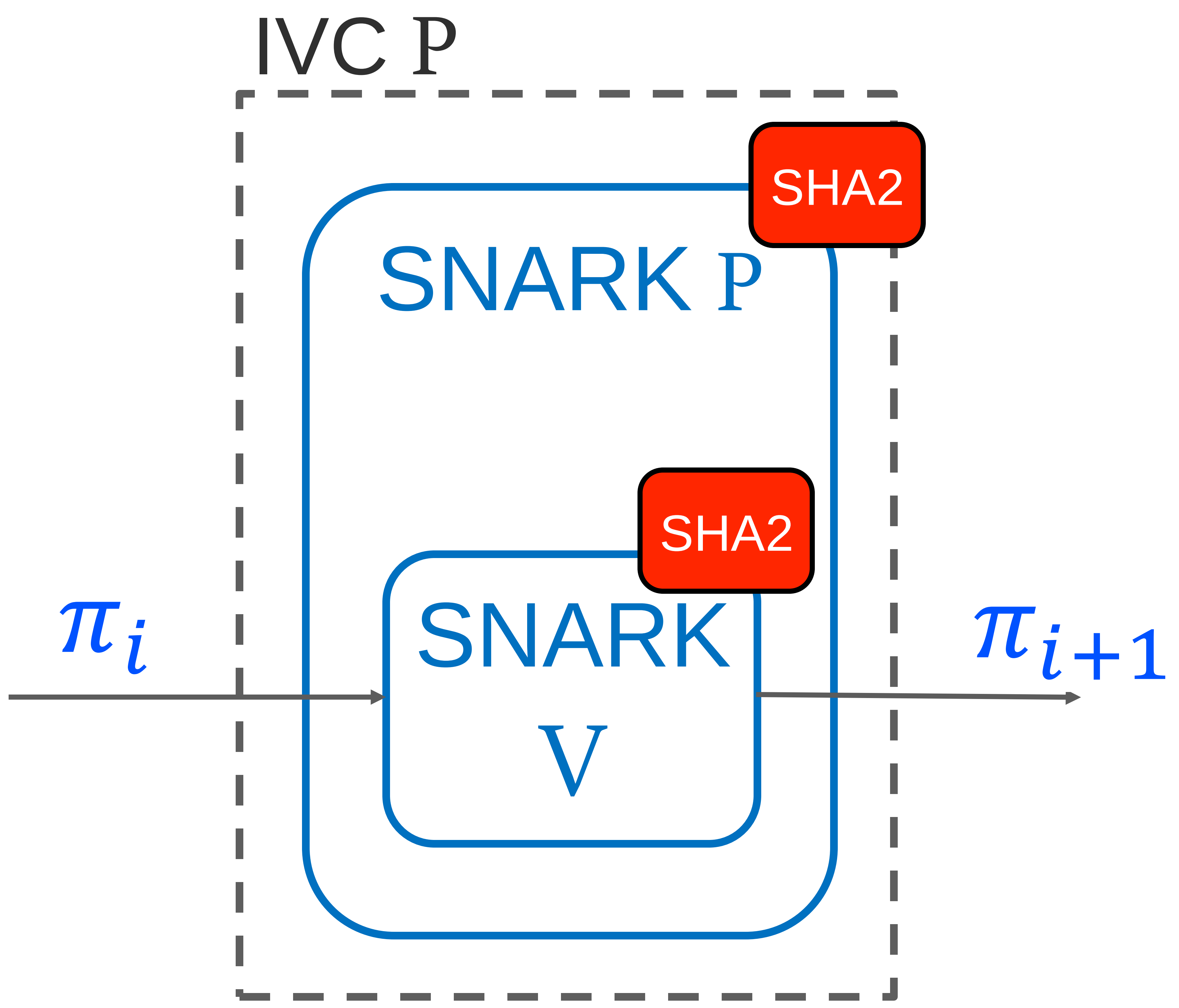
- SNARK verifier makes oracle queries, but SNARK is for non-oracle computations.

- [COS20;,…] Heuristically instantiate ro.



Issues with heuristic RO instantiation

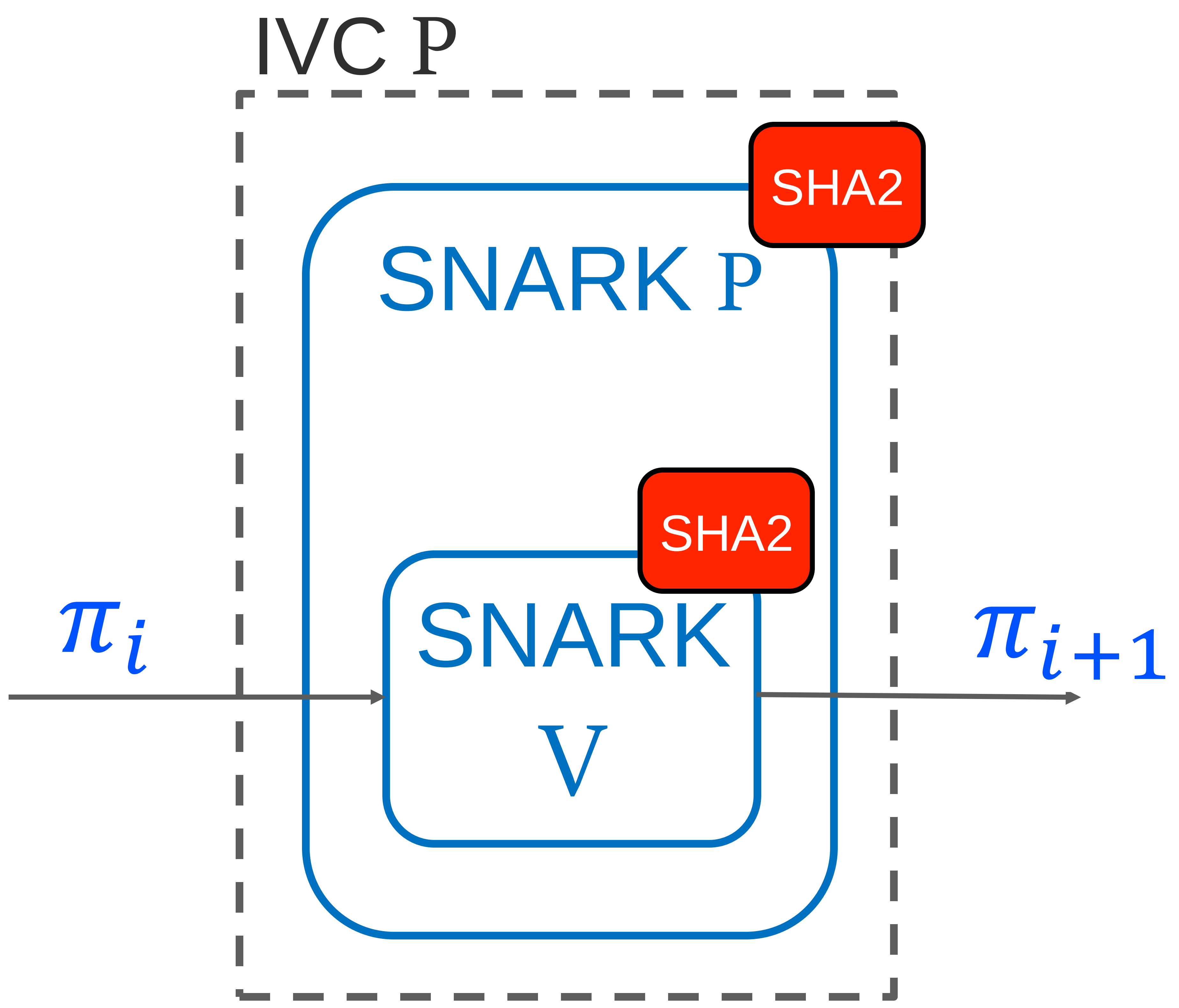
Theoretical:



Issues with heuristic RO instantiation

Theoretical:

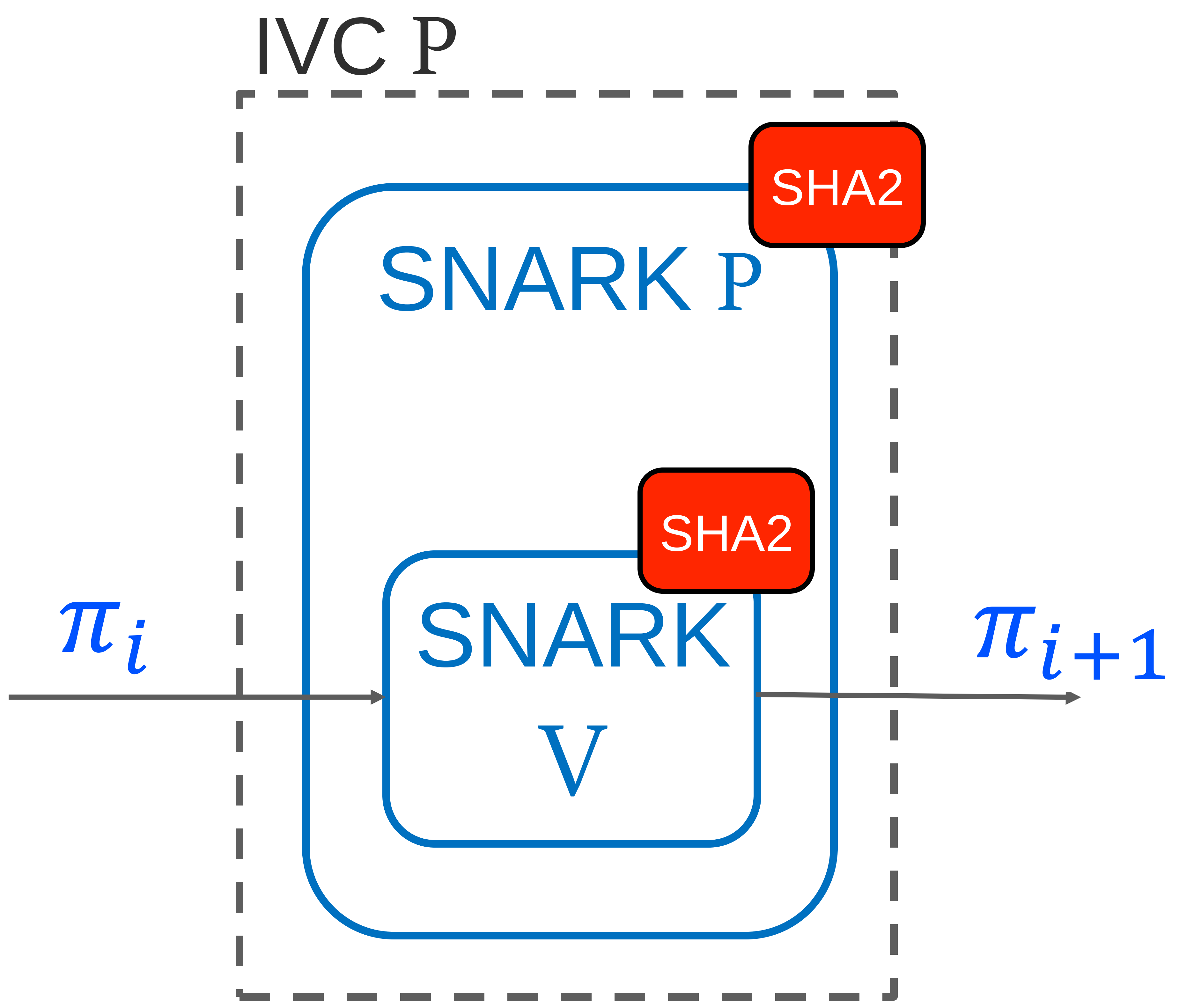
- Requires non-blackbox use of oracle; this breaks the RO abstraction.



Issues with heuristic RO instantiation

Theoretical:

- Requires non-blackbox use of oracle; this breaks the RO abstraction.
- Security flaws may be in the heuristic step [GoldwasserK03; CanettiGH04].

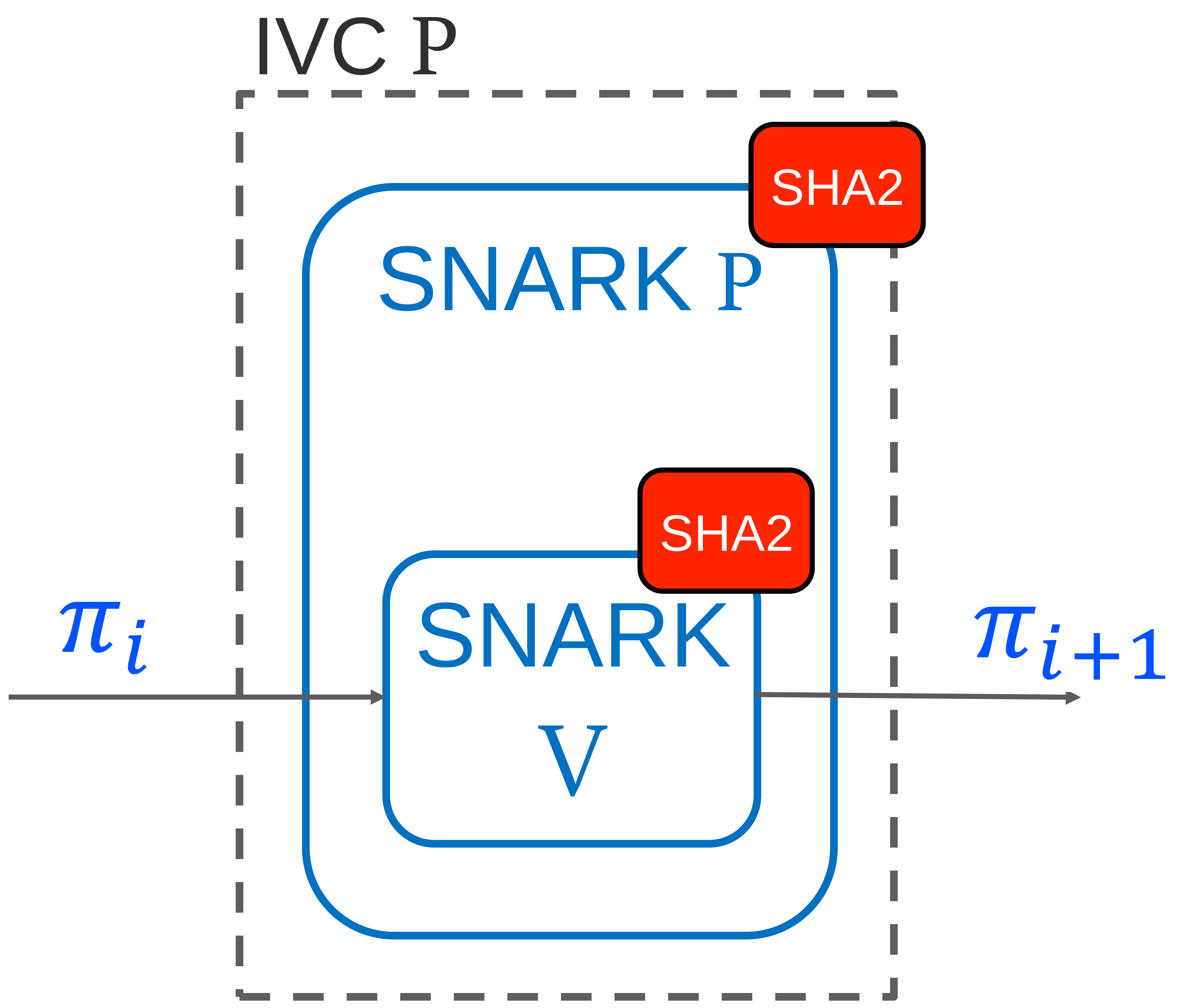


Issues with heuristic RO instantiation

Theoretical:

- Requires non-blackbox use of oracle; this breaks the RO abstraction.
- Security flaws may be in the heuristic step [GoldwasserK03; CanettiGH04].

Practical:



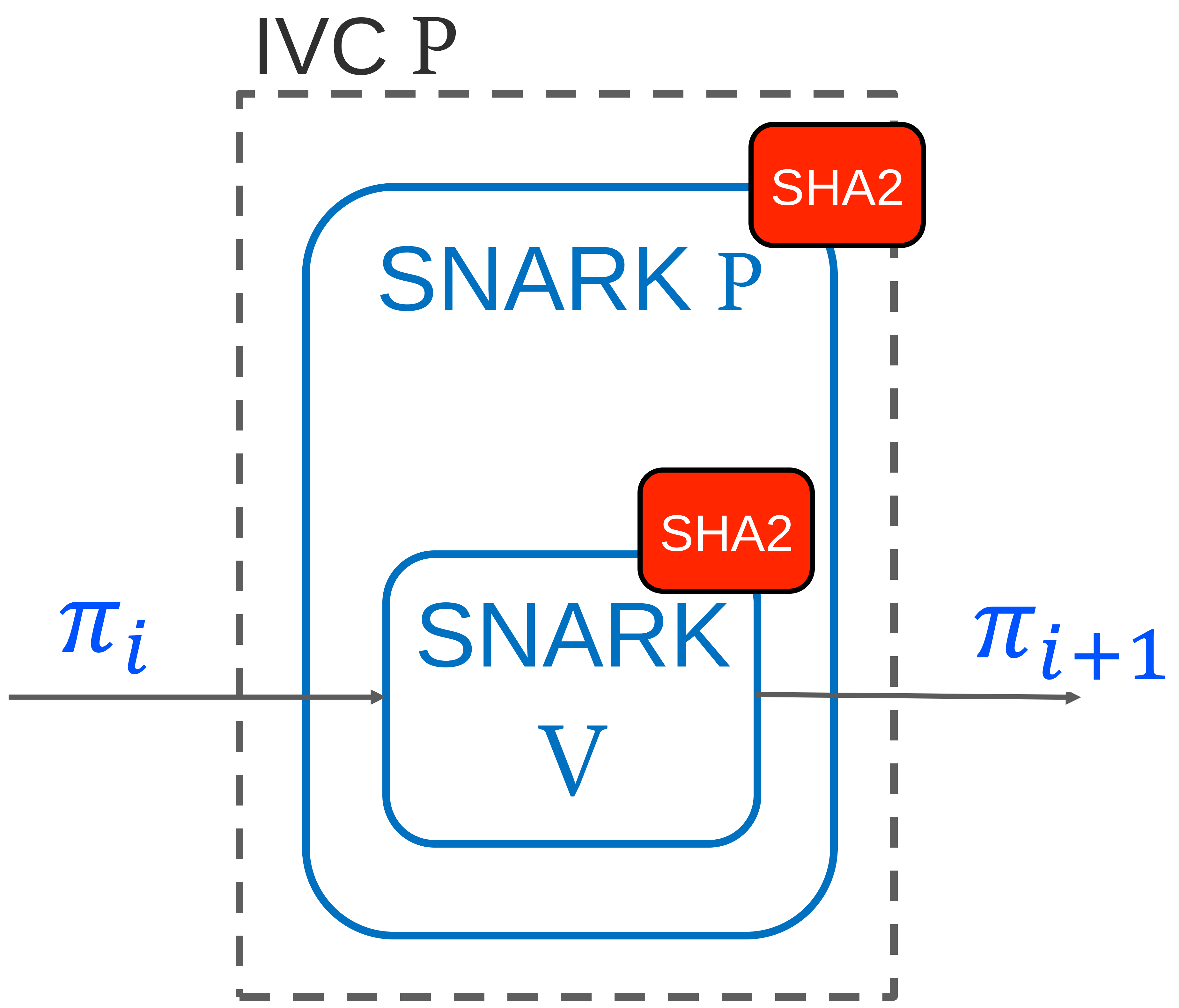
Issues with heuristic RO instantiation

Theoretical:

- Requires non-blackbox use of oracle; this breaks the RO abstraction.
- Security flaws may be in the heuristic step [GoldwasserK03; CanettiGH04].

Practical:

- No flexibility: Oracle must be instantiated as a circuit: can't use MPC, hardware token.



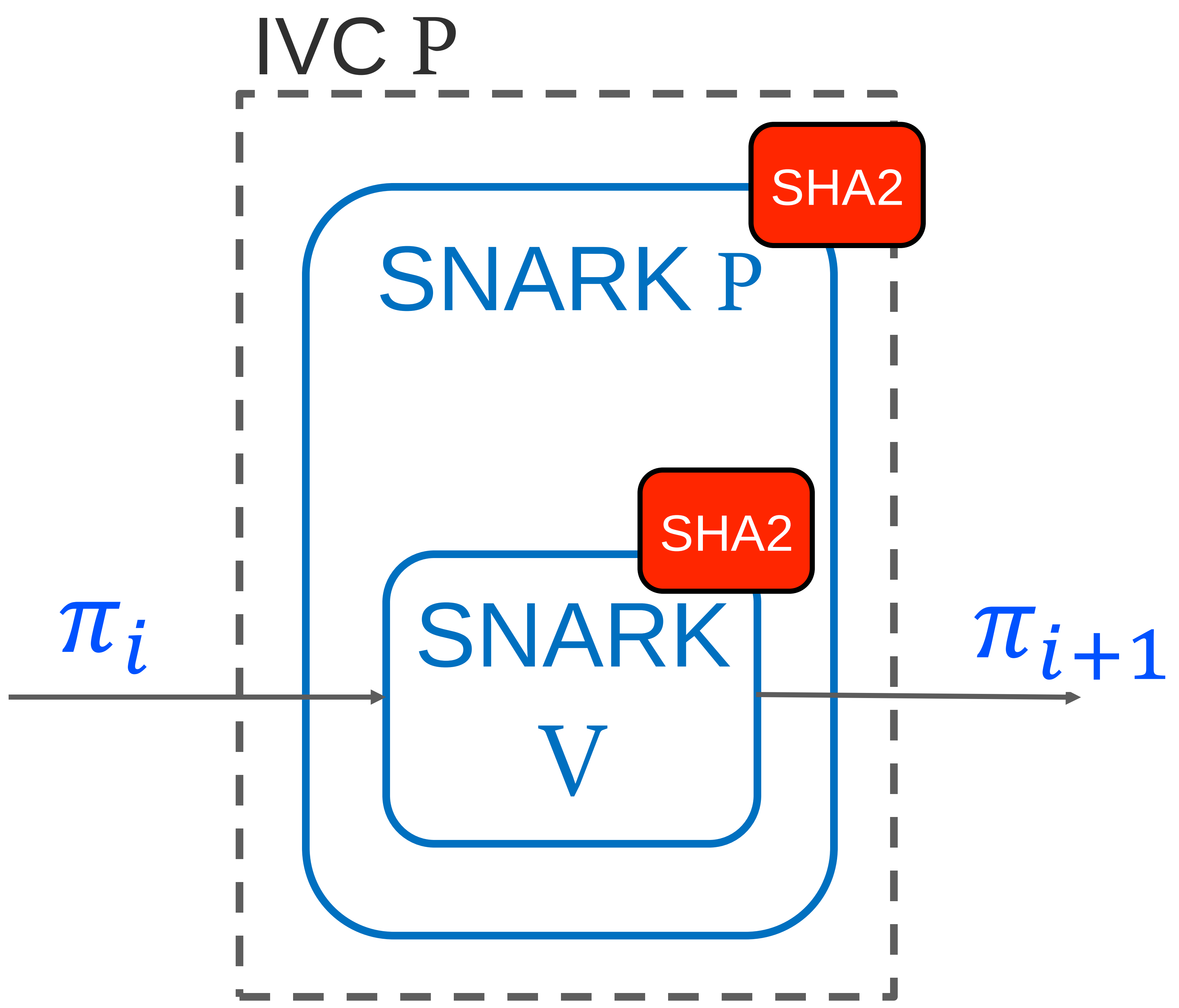
Issues with heuristic RO instantiation

Theoretical:

- Requires non-blackbox use of oracle; this breaks the RO abstraction.
- Security flaws may be in the heuristic step [GoldwasserK03; CanettiGH04].

Practical:

- No flexibility: Oracle must be instantiated as a circuit: can't use MPC, hardware token.
- Inefficient: SNARKs about SHA2, BLAKE are expensive!



IVC from additional oracle structure

IVC from additional oracle structure

It is unknown whether we can obtain IVC in the ROM.

IVC from additional oracle structure

It is unknown whether we can obtain IVC in the ROM.

IVC from additional oracle structure

It is unknown whether we can obtain IVC in the ROM.

Signed Random Oracle Model [CT10]

Random oracle signs responses using a signature scheme.

IVC from additional oracle structure

It is unknown whether we can obtain IVC in the ROM.

Signed Random Oracle Model [CT10]

Random oracle signs responses using a signature scheme.

We obtain IVC in the signed random oracle model.

IVC from additional oracle structure

It is unknown whether we can obtain IVC in the ROM.

Signed Random Oracle Model [CT10]

Random oracle signs responses using a signature scheme.

We obtain IVC in the signed random oracle model.

IVC from additional oracle structure

It is unknown whether we can obtain IVC in the ROM.

Signed Random Oracle Model [CT10]

Random oracle signs responses using a signature scheme.

We obtain IVC in the signed random oracle model.

Low-Degree Random Oracle Model [CCS22]

Random oracle is extended to a low-degree multivariate polynomial.

IVC from additional oracle structure

It is unknown whether we can obtain IVC in the ROM.

Signed Random Oracle Model [CT10]

Random oracle signs responses using a signature scheme.

We obtain IVC in the signed random oracle model.

Low-Degree Random Oracle Model [CCS22]

Random oracle is extended to a low-degree multivariate polynomial.

We obtain IVC in the low-degree random oracle model.

IVC from additional oracle structure

It is unknown whether we can obtain IVC in the ROM.

Signed Random Oracle Model [CT10]

Random oracle signs responses using a signature scheme.

We obtain IVC in the signed random oracle model.

Low-Degree Random Oracle Model [CCS22]

Random oracle is extended to a low-degree multivariate polynomial.

We obtain IVC in the low-degree random oracle model.

IVC from additional oracle structure

It is unknown whether we can obtain IVC in the ROM.

Signed Random Oracle Model [CT10]

Random oracle signs responses using a signature scheme.

We obtain IVC in the signed random oracle model.

Low-Degree Random Oracle Model [CCS22]

Random oracle is extended to a low-degree multivariate polynomial.

We obtain IVC in the low-degree random oracle model.

We don't know of any software-only instantiation of the model.

IVC from additional oracle structure

It is unknown whether we can obtain IVC in the ROM.

Signed Random Oracle Model [CT10]

Random oracle signs responses using a signature scheme.

We obtain IVC in the signed random oracle model.

Low-Degree Random Oracle Model [CCS22]

Random oracle is extended to a low-degree multivariate polynomial.

We obtain IVC in the low-degree random oracle model.

We don't know of any software-only instantiation of the model.

The question

Is there an oracle model for which:

1. There exists a PCD scheme in this model under standard assumptions; and

2. The oracle can be heuristically instantiated in software?

Our results

Yes!

We propose a new oracle model, the arithmetized random oracle model (AROM), which has a plausible heuristic instantiation.

Our results

Yes!

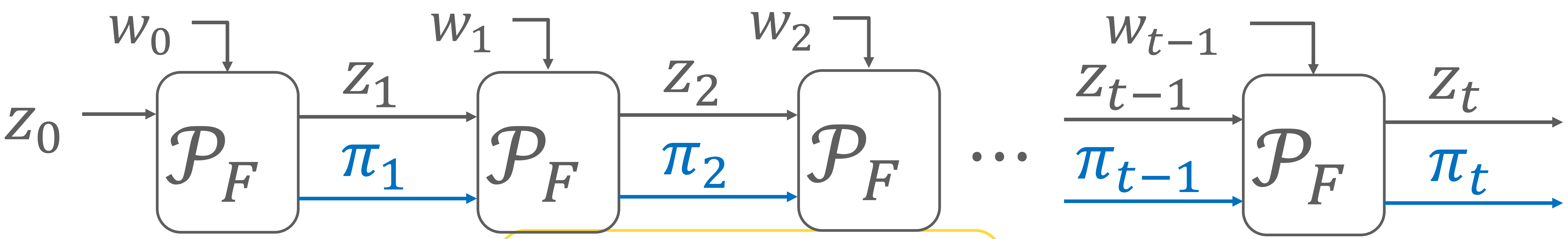
We propose a new oracle model, the arithmetized random oracle model (AROM), which has a plausible heuristic instantiation.

Theorem: There exists PCD/IVC in the AROM, assuming the existence of collision-resistant hash functions in the standard model.

Our results

Yes!

We propose a new oracle model, the arithmetized random oracle model (AROM), which has a plausible heuristic instantiation.

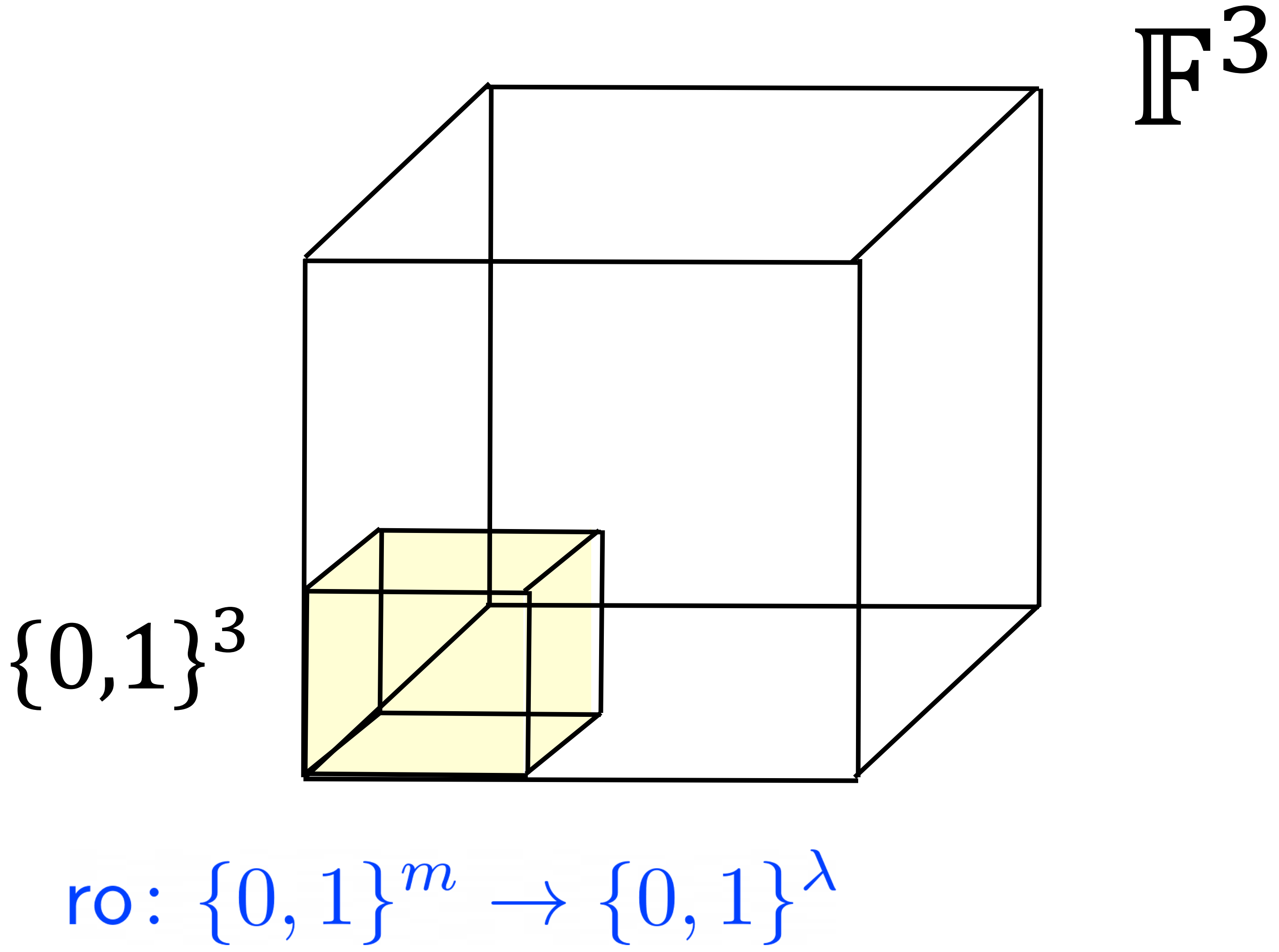


Secure to arbitrary depth t

Theorem: There exists PCD/IVC in the AROM, assuming the existence of collision-resistant hash functions in the standard model.

Low-degree random oracle [CCS22]

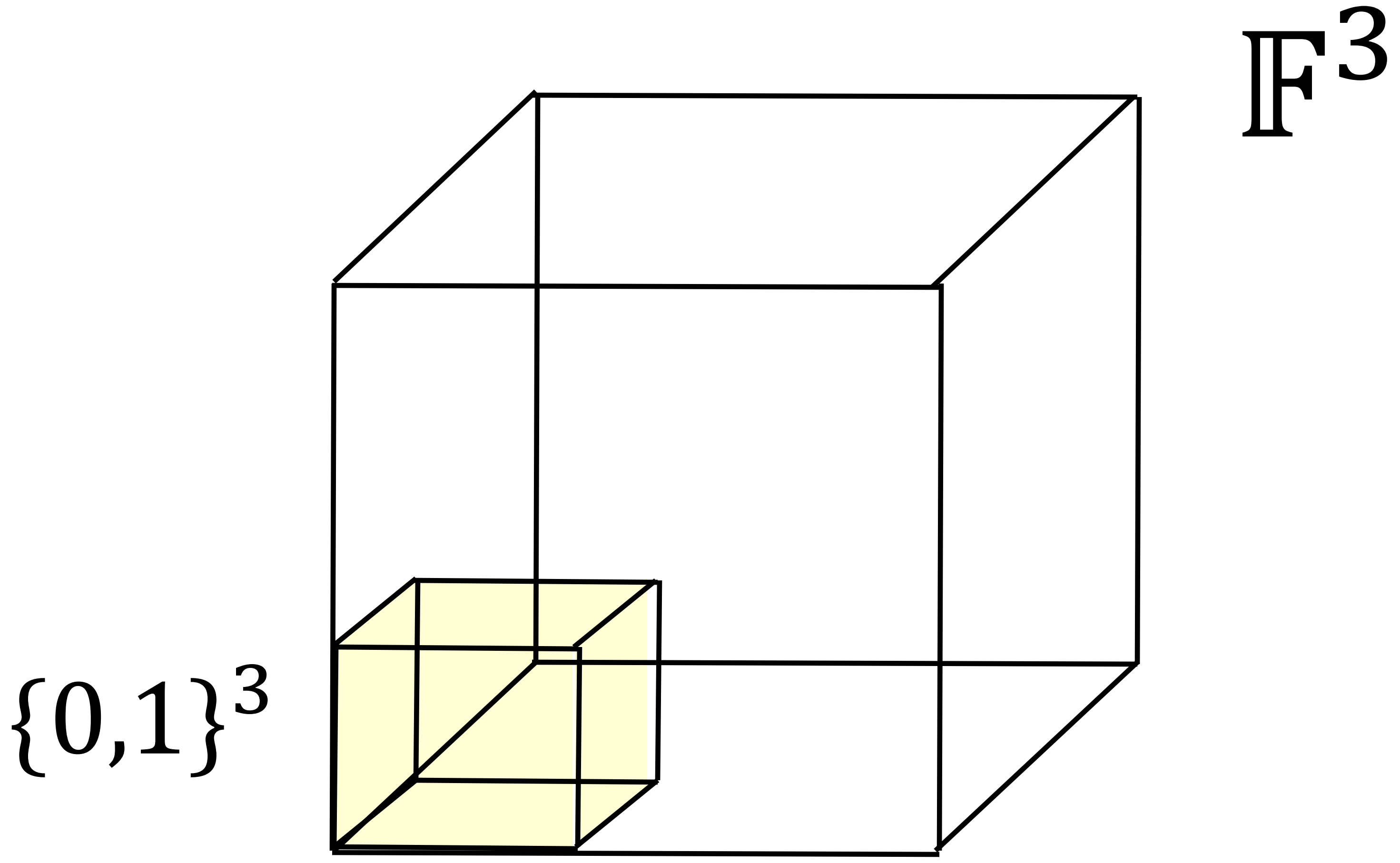
Low-degree random oracle [CCS22]



Random $\hat{\rho} \in \mathbb{F}^{\leq d}[X_1, \ldots, X_m]$ such that:

- Points in Boolean hypercube agree with the random oracle.
- Is low-degree (e.g. $d = O(1)$).
- Can query ANY point in $\mathbb{F}^m$.

Low-degree random oracle [CCS22]

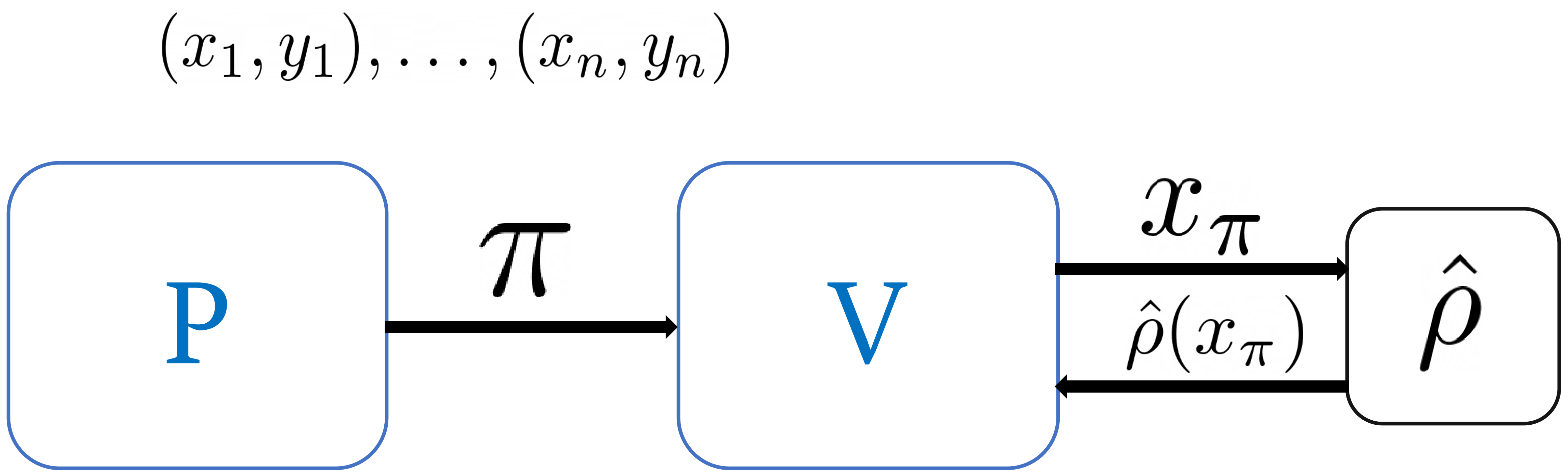


ro: $\{0,1\}^m \to \{0,1\}^\lambda$

Random $\hat{\rho} \in \mathbb{F}^{\leq d}[X_1, \ldots, X_m]$ such that:

- Points in Boolean hypercube agree with the random oracle.
- Is low-degree (e.g. $d = O(1)$).
- Can query ANY point in $\mathbb{F}^m$.

- [CCS22] give a non-interactive **query-reduction** protocol in the LDRO.
- This allows V to verify n queries to $\hat{\rho}$ with a single query to $\hat{\rho}$!



Arithmetization of hash functions

- Let's try to instantiate the low-degree random oracle...

Arithmetization of hash functions

- Let's try to instantiate the low-degree random oracle...

$$H$$

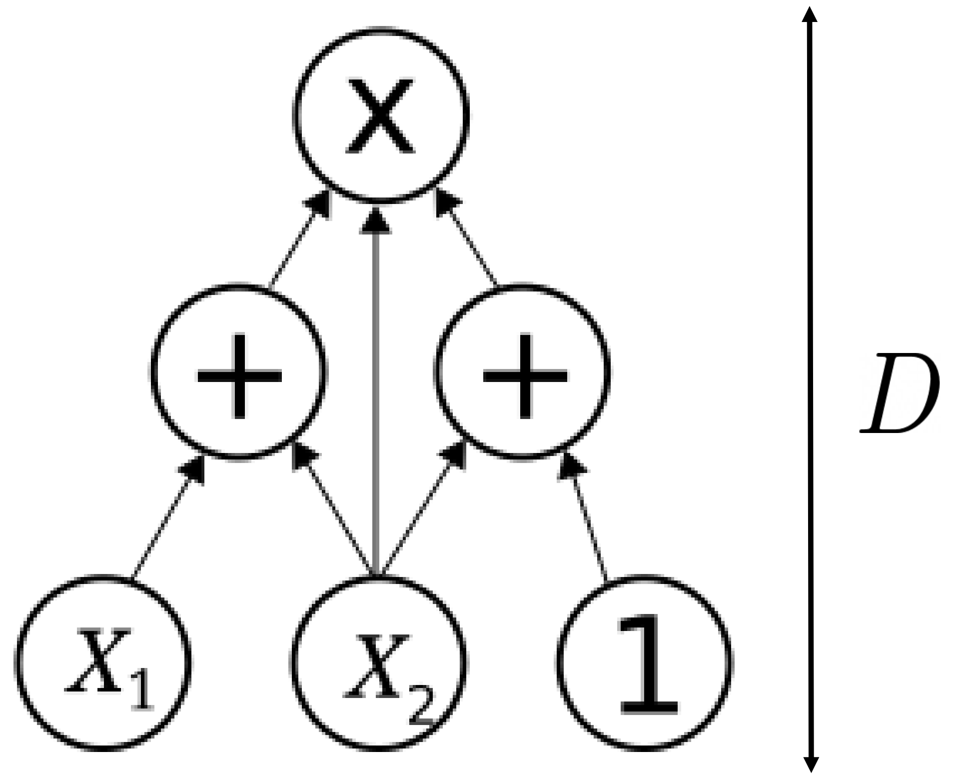
Hash function

Arithmetization of hash functions

- Let's try to instantiate the low-degree random oracle...

H

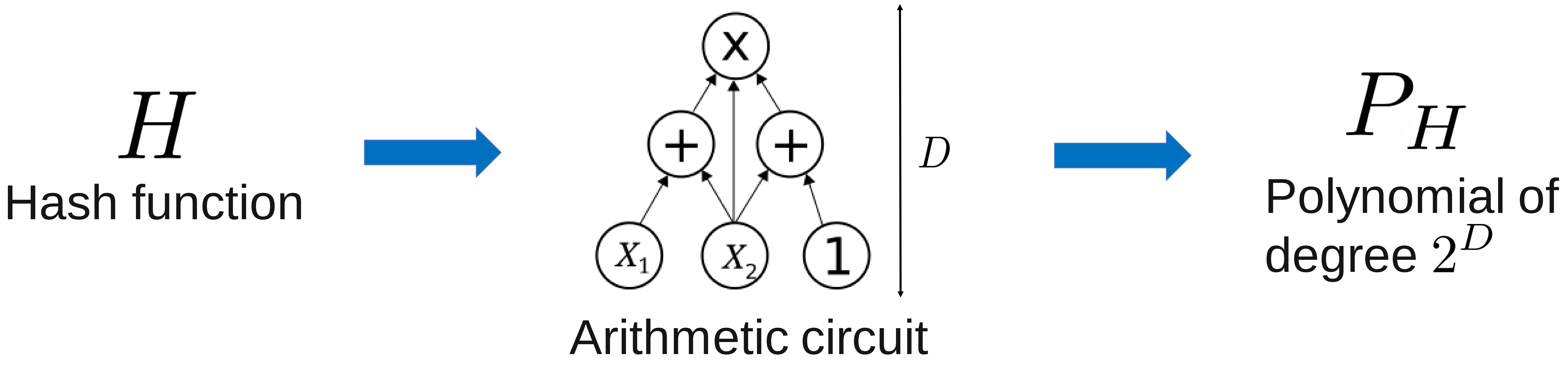
Hash function



Arithmetic circuit

Arithmetization of hash functions

- Let's try to instantiate the low-degree random oracle...

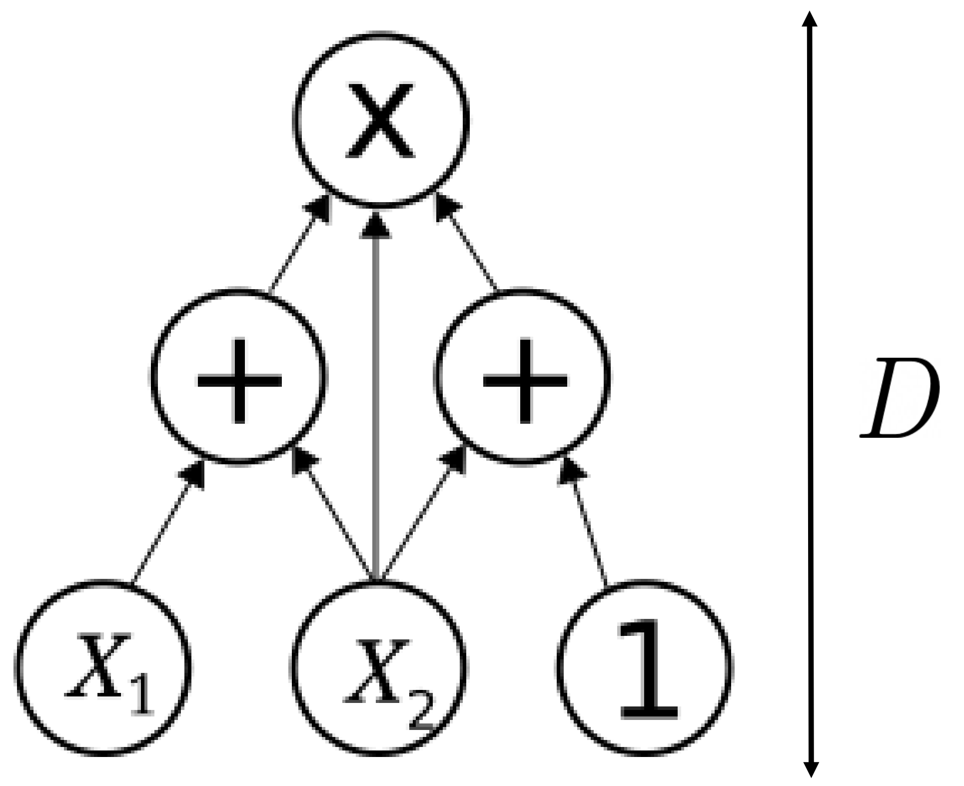


Arithmetization of hash functions

- Let's try to instantiate the low-degree random oracle...

H

Hash function



Arithmetic circuit

P_H

Polynomial of degree 2^D

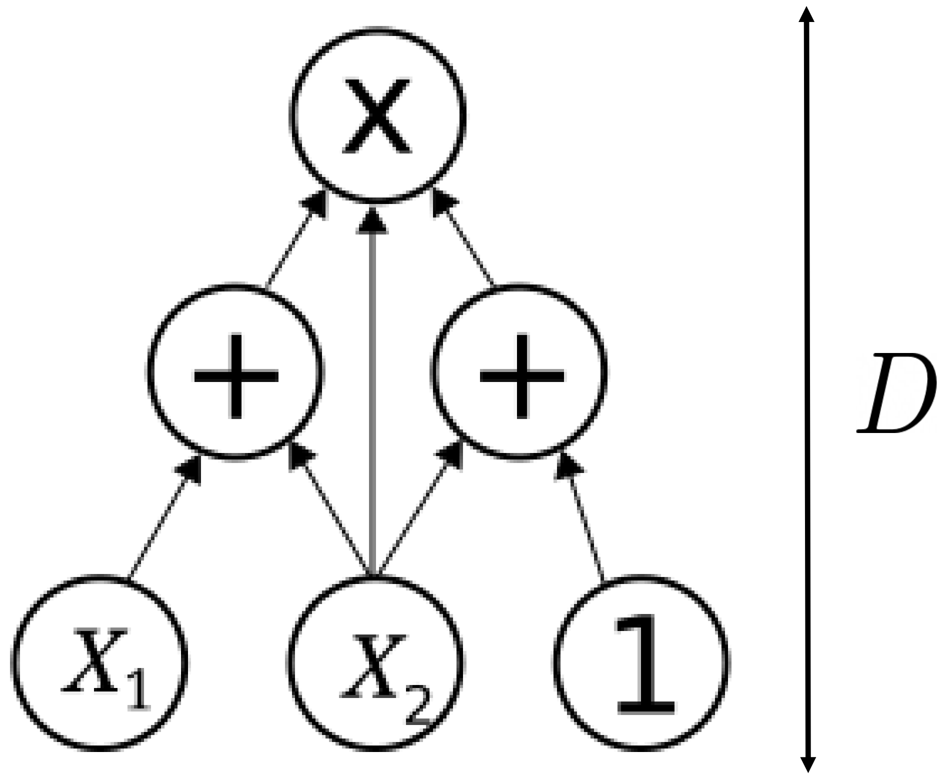
- The IVC verifier for the low-degree random oracle runs in time at least $O(2^D)$.

Arithmetization of hash functions

- Let's try to instantiate the low-degree random oracle...

H

Hash function



Arithmetic circuit

P_H

Polynomial of degree 2^D

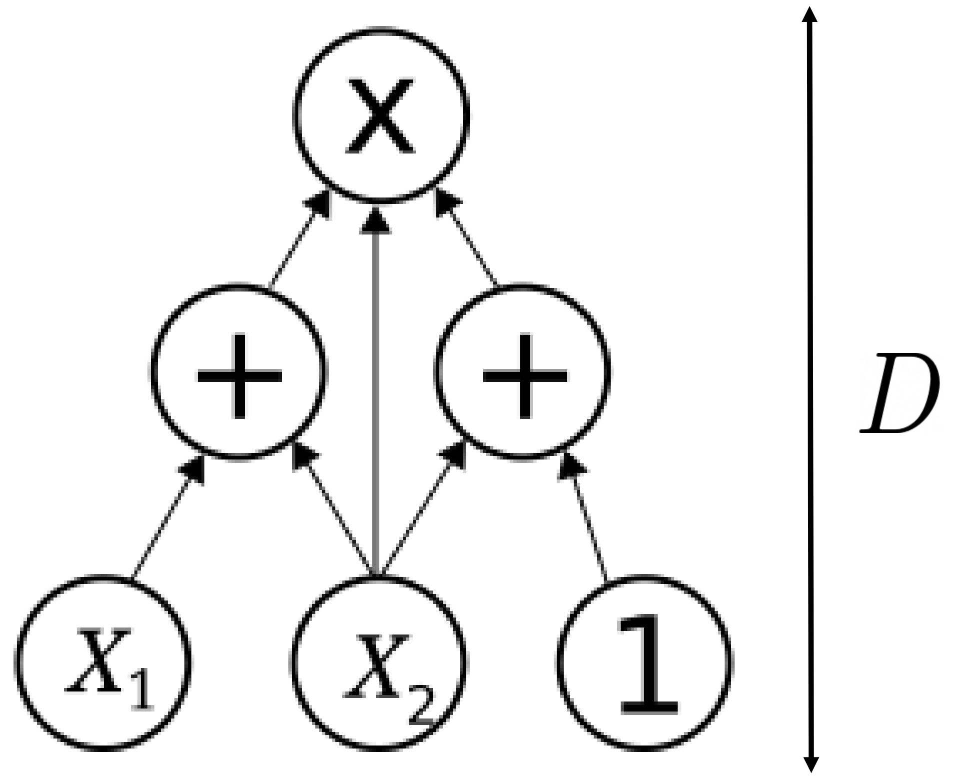
- The IVC verifier for the low-degree random oracle runs in time at least $O(2^D)$.
- For widely used hash functions, $D > 25$.

Arithmetization of hash functions

- Let's try to instantiate the low-degree random oracle...

H

Hash function



Arithmetic circuit

P_H

Polynomial of degree 2^D

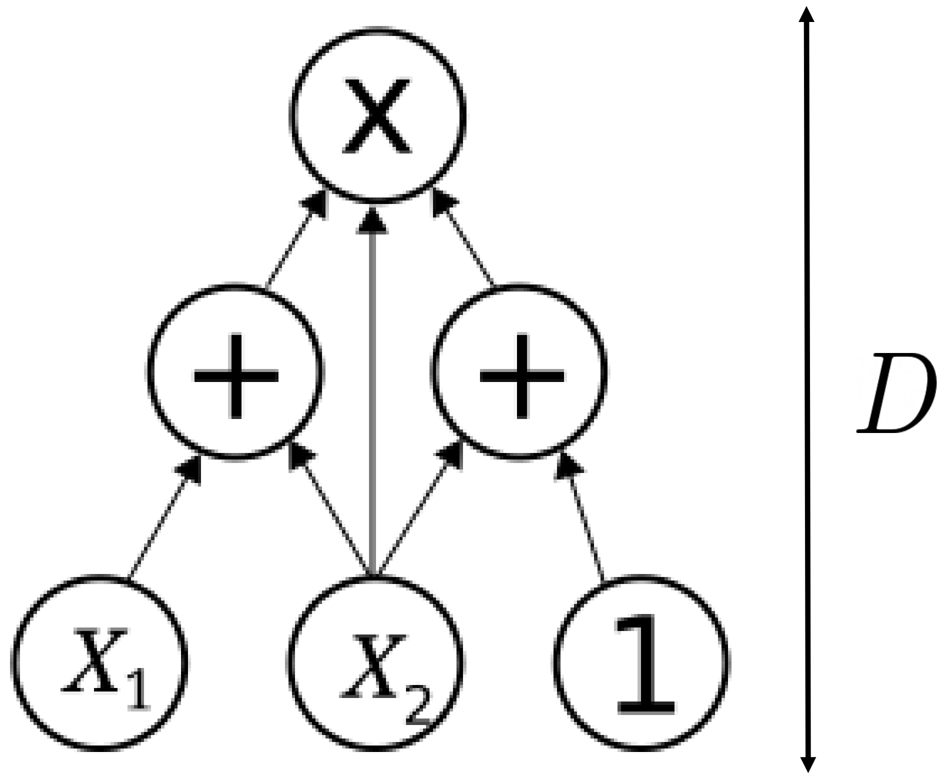
- The IVC verifier for the low-degree random oracle runs in time at least $O(2^D)$.
- For widely used hash functions, $D > 25$.

Arithmetization of hash functions

- Let's try to instantiate the low-degree random oracle...

H

Hash function →



Arithmetic circuit

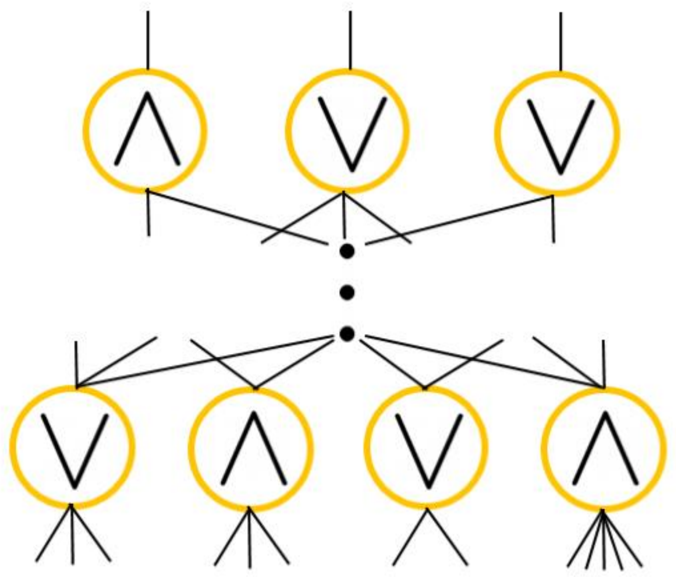
→ P_H

Polynomial of degree 2^D

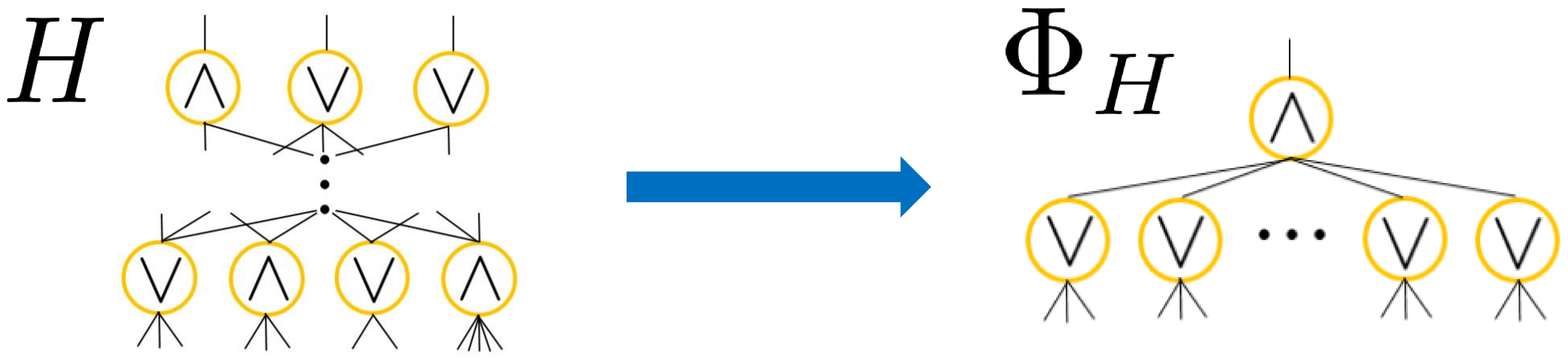
- The IVC verifier for the low-degree random oracle runs in time at least $O(2^D)$.
- For widely used hash functions, $D > 25$.
- We need to reduce the depth of this circuit.

Degree reduction

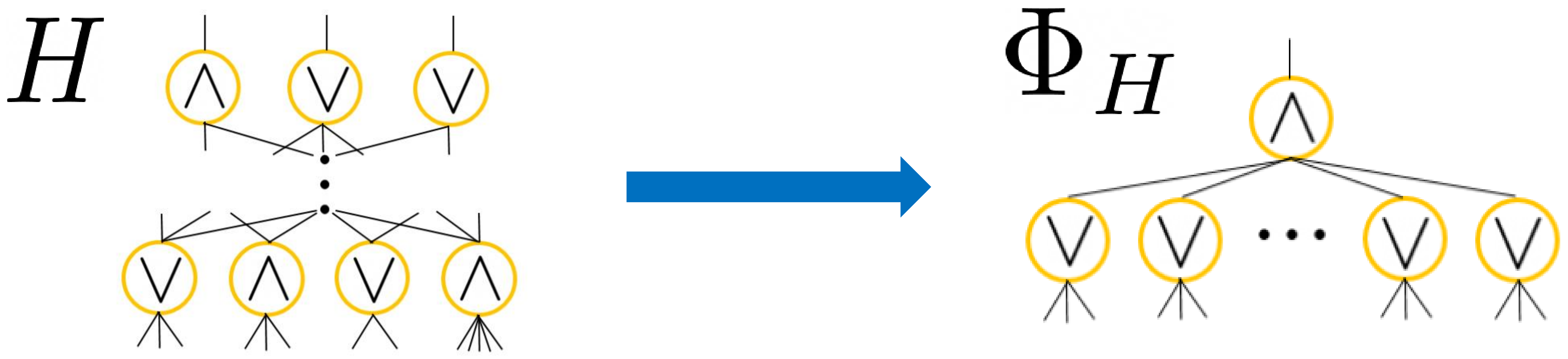
H



Degree reduction



Degree reduction

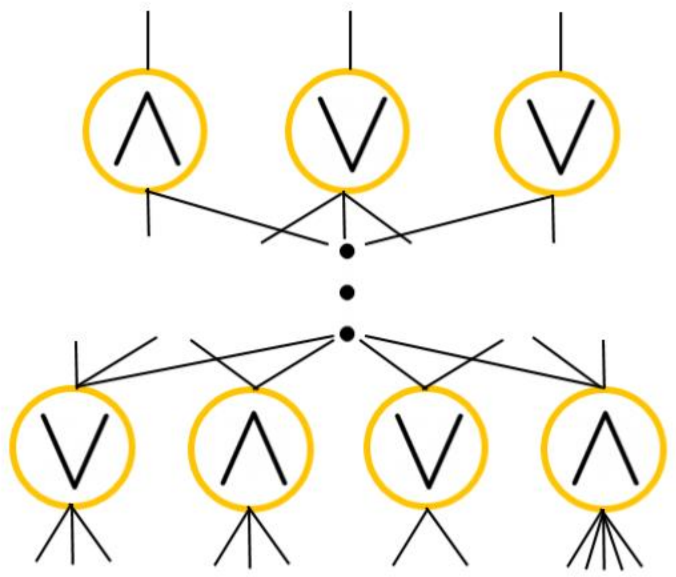


$$\Phi_H(x, y, z) = \begin{cases} 1 & \text{if } H(x) = y \text{ and } W_H(x) = z \\ 0 & \text{otherwise} \end{cases}$$

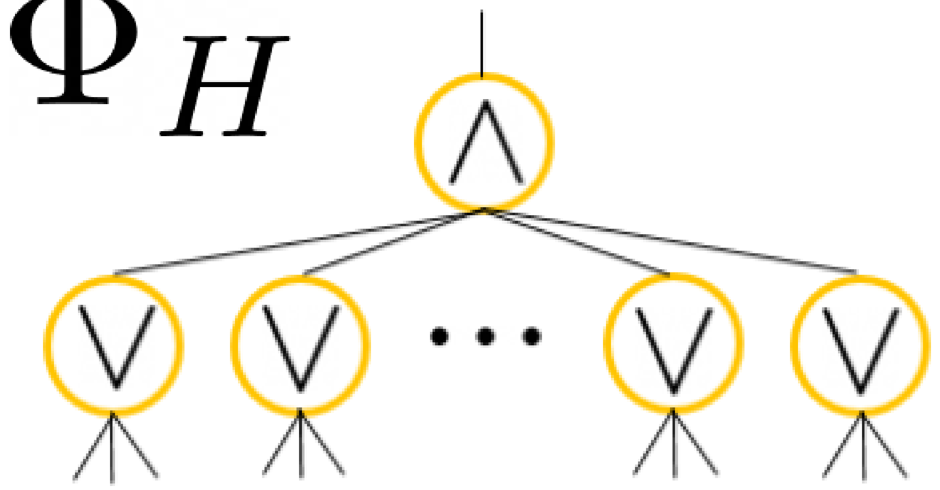
Efficiently computable

Degree reduction

H



Φ_H



P_H

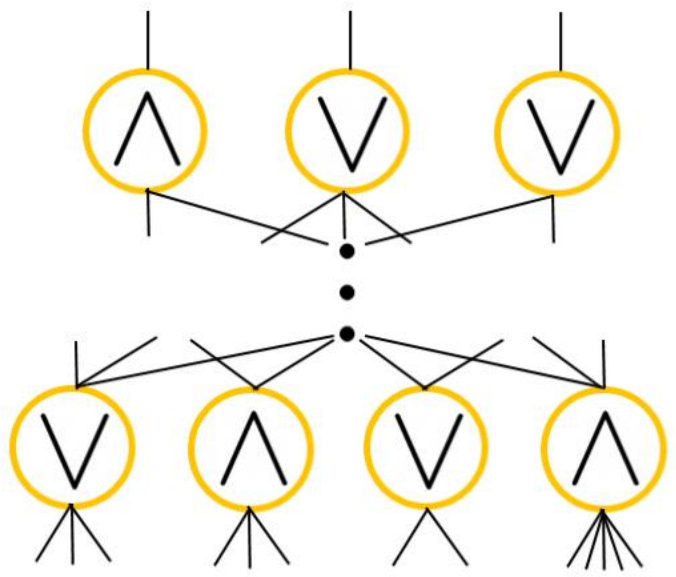
Polynomial of degree $O(|H|)$

$$\Phi_H(x, y, z) = \begin{cases} 1 & \text{if } H(x) = y \text{ and } W_H(x) = z \\ 0 & \text{otherwise} \end{cases}$$

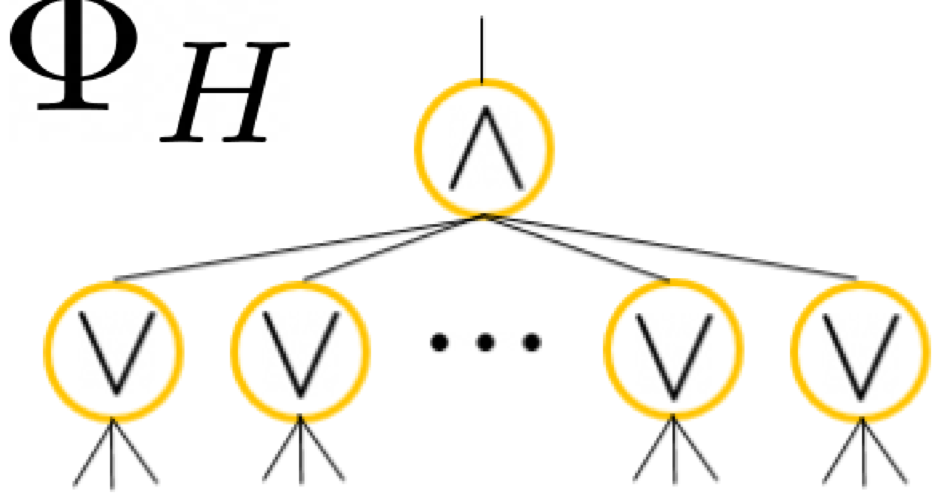
Efficiently computable

Degree reduction

H



Φ_H



P_H

Polynomial of degree $O(|H|)$

$$\Phi_H(x, y, z) = \begin{cases} 1 & \text{if } H(x) = y \text{ and } W_H(x) = z \\ 0 & \text{otherwise} \end{cases}$$

Efficiently computable

- This is <u>not</u> a low-degree extension of H, so we can't instantiate the LDRO this way.

Arithmetized Random Oracle Model

Hash function H

Witness function W_H

Verification polynomial

$$P_H(x, y, z) = 1$$

$$\iff$$

$$H(x) = y \wedge W_H(x) = z$$

Arithmetized Random Oracle Model

Hash function H

Random oracle
ro: $\{0,1\}^m \rightarrow \{0,1\}^\lambda$

Witness function W_H

Verification polynomial

$$P_H(x, y, z) = 1$$
$$\Longleftrightarrow$$
$$H(x) = y \wedge W_H(x) = z$$

Arithmetized Random Oracle Model

Hash function H → **Random oracle** ro: $\{0,1\}^m \to \{0,1\}^\lambda$

Witness function W_H → **Witness oracle** wo: $\{0,1\}^m \to \{0,1\}^w$

Verification polynomial

$$P_H(x,y,z) = 1$$
$$\iff$$
$$H(x) = y \wedge W_H(x) = z$$

Arithmetized Random Oracle Model

Hash function H → **Random oracle** ro: $\{0,1\}^m \to \{0,1\}^\lambda$

Witness function W_H → **Witness oracle** wo: $\{0,1\}^m \to \{0,1\}^w$

Verification polynomial

$$P_H(x,y,z) = 1$$
$$\iff$$
$$H(x) = y \wedge W_H(x) = z$$

→

Extended verification oracle $\hat{\text{vo}}\colon \mathbb{F}^m \to \mathbb{F}$

low-degree extension of

$$\text{vo}(x,y,z) = 1$$
$$\iff$$
$$\text{ro}(x) = y \wedge \text{wo}(x) = z$$

Modelling Challenges

Modelling Challenges

1. **How do we define** WO **?**

Modelling Challenges

1. **How do we define WO?**

WO is an adversarially chosen PPT-computable function.

Modelling Challenges

1. **How do we define wo?**

wo is an adversarially chosen PPT-computable function.

2. **How do we define v̂o?**

Modelling Challenges

1. **How do we define** wo **?**

wo is an adversarially chosen PPT-computable function.

2. **How do we define** $\hat{\text{vo}}$**?**

- If $d = 1$, an efficient adversary can invert ro [JKRS09].

Modelling Challenges

1. **How do we define wo?**

wo is an adversarially chosen PPT-computable function.

2. **How do we define $\hat{\mathsf{vo}}$?**

- If $d = 1$, an efficient adversary can invert ro [JKRS09].
- If $\hat{\mathsf{vo}}$ is adversarially chosen, an efficient adversary can invert ro.

Modelling Challenges

1. How do we define wo?

wo is an adversarially chosen PPT-computable function.

2. How do we define $\hat{vo}$?

- If $d = 1$, an efficient adversary can invert ro [JKRS09].
- If $\hat{vo}$ is adversarially chosen, an efficient adversary can invert ro.

$\hat{vo}$ is a uniformly random low-degree extension of vo with $d \geq 2$.

Modelling Challenges

1. **How do we define wo?**

wo is an adversarially chosen PPT-computable function.

2. **How do we define $\hat{\text{vo}}$?**

- If $d = 1$, an efficient adversary can invert ro [JKRS09].
- If $\hat{\text{vo}}$ is adversarially chosen, an efficient adversary can invert ro.

$\hat{\text{vo}}$ is a uniformly random low-degree extension of vo with $d \geq 2$.

- Weakening this choice is a question for future work.

The question

Is there an oracle model for which:

1. There exists a PCD scheme in this model under standard assumptions; and

2. The oracle can be heuristically instantiated in software?

The question

Is there an oracle model for which:

1. There exists a PCD scheme in this model under standard assumptions; and

2. The oracle can be heuristically instantiated in software? ✓

Overview of construction

Overview of construction

PCD in the AROM.

Overview of construction

Security in ROM => Security in AROM.

Via the combinatorial nullstellensatz [Alo99] and results from algebraic query complexity [AW09].

PCD in the AROM.

Overview of construction

By constructing an accumulation scheme [BCMS20] for the arithmetized random oracle.

Security in ROM => Security in AROM.

+

Query reduction in the AROM.

=

PCD in the AROM.

Via the combinatorial nullstellensatz [Alo99] and results from algebraic query complexity [AW09].

The question

Is there an oracle model for which:

1. There exists a PCD scheme in this model under standard assumptions; and

2. The oracle can be heuristically instantiated in software? ✓

The question

Is there an oracle model for which:

1. There exists a PCD scheme in this model under standard assumptions; and

2. The oracle can be heuristically instantiated in software?

Thanks!

Emulation of the ARO

Lemma: There exists a probabilistic algorithm $\mathcal{M}$ such that for every security parameter $\lambda \in \mathbb{N}$, query bound $t \in \mathbb{N}$, and t-query adversary:

$$\left| \Pr_{(\mathsf{ro},\mathsf{wo},\hat{\mathsf{vo}}) \leftarrow \mathcal{O}(\lambda)} \left[\mathcal{A}^{(\mathsf{ro},\mathsf{wo},\hat{\mathsf{vo}})} = 1 \right] - \Pr_{(\mathsf{ro},\mathsf{wo},\hat{\mathsf{vo}}) \leftarrow \mathcal{O}(\lambda)} \left[\mathcal{A}^{\mathcal{M}^{(\mathsf{ro},\mathsf{wo})}} = 1 \right] \right| \leq \frac{t}{2^\lambda}$$