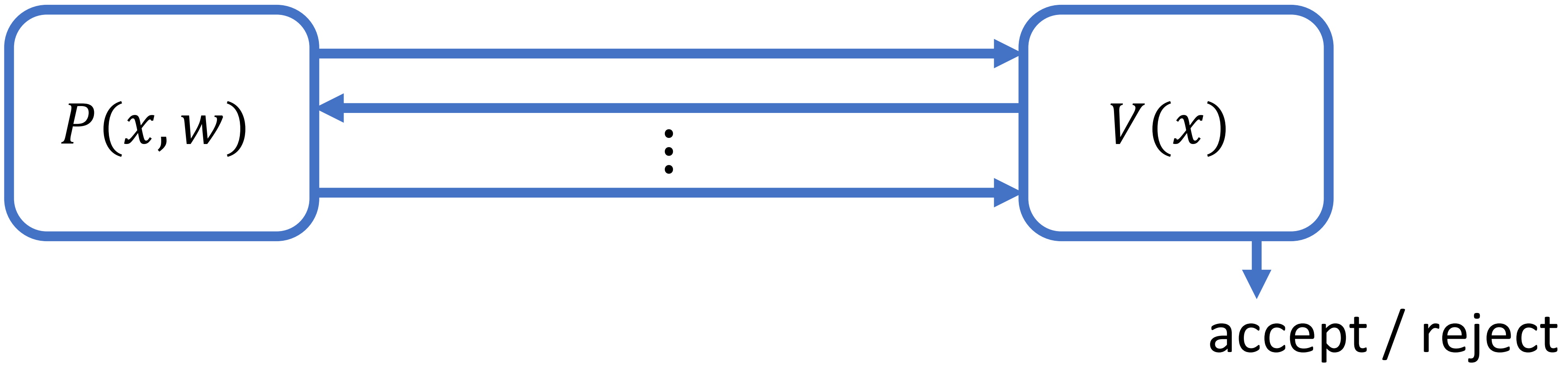
Quasi-Optimal SNARGs via Linear Multi-Prover Interactive Proofs

Dan Boneh, Yuval Ishai, Amit Sahai, and David J. Wu

Interactive Arguments for NP

$$\mathcal{L}_C = \{x : C(x,w) = 1 \text{ for some } w\}$$



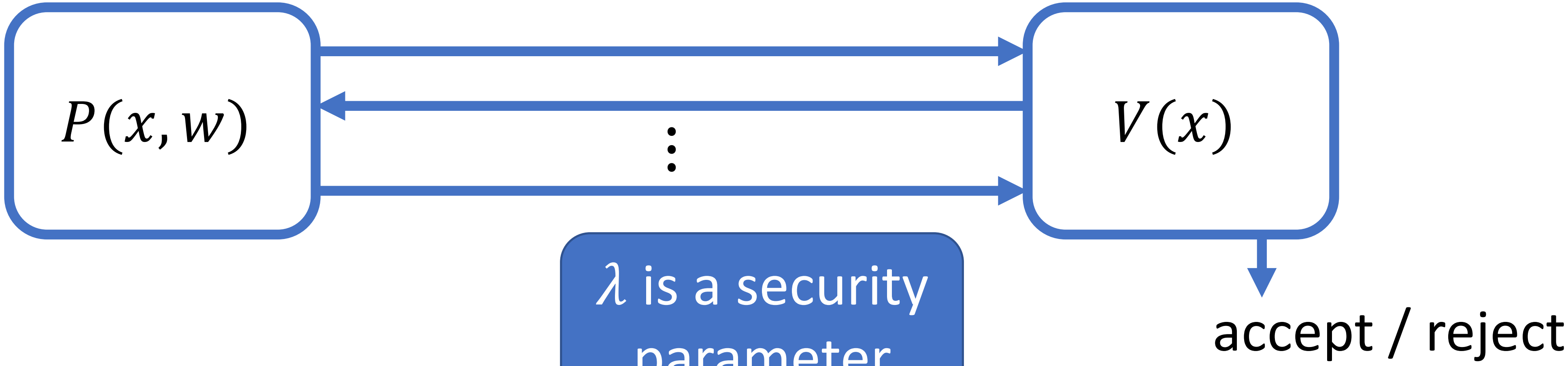
Completeness: $C(x,w) = 1 \Longrightarrow \Pr[\langle P(x,w), V(x)\rangle = 1] = 1$

Soundness: for all provers $P^\star$ of size 2^λ:

$$x \notin \mathcal{L}_C \Longrightarrow \Pr[\langle P^\star(x), V(x)\rangle = 1] \leq 2^{-\lambda}$$

Succinct Arguments

$$\mathcal{L}_C = \{x : C(x, w) = 1 \text{ for some } w\}$$



λ is a security parameter

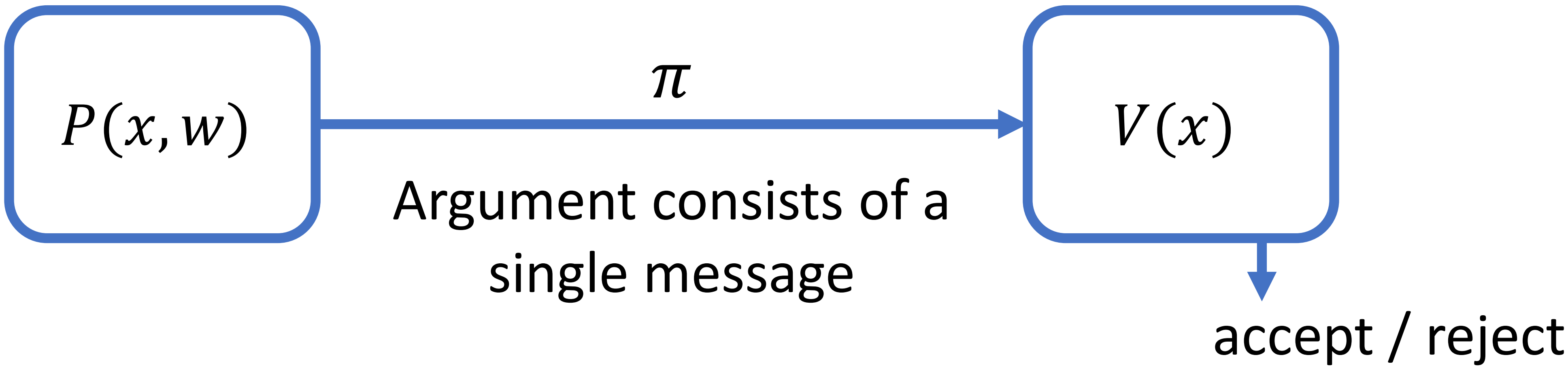
Argument system is *succinct* if:

- Prover communication is $\text{poly}(\lambda + \log|C|)$
- V can be implemented by a circuit of size $\text{poly}(\lambda + |x| + \log|C|)$

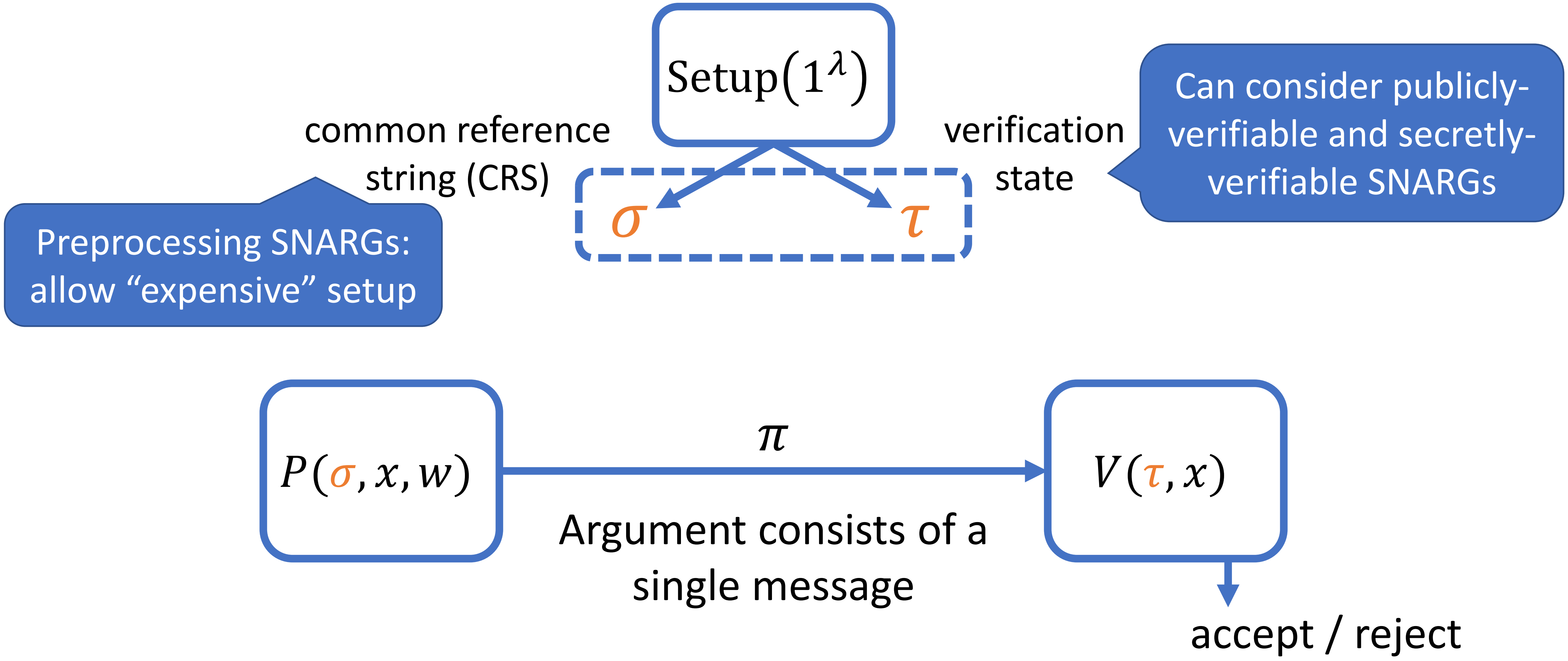
Verifier complexity significantly smaller than classic NP verifier

Succinct Non-Interactive Arguments (SNARGs)

Instantiation: “CS proofs” in the random oracle model [Mic94]



Succinct Non-Interactive Arguments (SNARGs)



Complexity Metrics for SNARGs

Soundness: for all provers $P^\star$ of size 2^λ:

$$x \notin \mathcal{L}_C \Longrightarrow \Pr[\langle P^\star(x), V(x)\rangle = 1] \leq 2^{-\lambda}$$

How short can the proofs be?

$$|\pi| = \Omega(\lambda)$$

Even in the designated-verifier setting

[See paper for details]

How much work is needed to generate the proof?

$$|P| = \Omega(|C|)$$

Quasi-Optimal SNARGs

Soundness: for all provers $P^\star$ of size 2^λ:

$$x \notin \mathcal{L}_C \Longrightarrow \Pr[\langle P^\star(x), V(x) \rangle = 1] \leq 2^{-\lambda}$$

A SNARG (for Boolean circuit satisfiability) is quasi-optimal if it satisfies the following properties:

- Quasi-optimal succinctness:

$$|\pi| = \lambda \cdot \mathrm{polylog}(\lambda, |C|) = \tilde{O}(\lambda)$$

- Quasi-optimal prover complexity:

$$|P| = \tilde{O}(|C|) + \mathrm{poly}(\lambda, \log|C|)$$

Quasi-Optimal SNARGs

Construction	Prover Complexity	Proof Size	Assumption
CS Proofs [Mic94]	$\tilde{O}(\lvert C \rvert)$	$\tilde{O}(\lambda^2)$	Random Oracle
Groth [Gro16]	$\tilde{O}(\lambda \lvert C \rvert)$	$\tilde{O}(\lambda)$	Generic Group
Groth [Gro10]	$\tilde{O}(\lambda \lvert C \rvert^2 + \lvert C \rvert \lambda^2)$	$\tilde{O}(\lambda)$	Knowledge of Exponent
GGPR [GGPR12]	$\tilde{O}(\lambda \lvert C \rvert)$	$\tilde{O}(\lambda)$	Knowledge of Exponent
BCIOP (Pairing) [BCIOP13]	$\tilde{O}(\lambda \lvert C \rvert)$	$\tilde{O}(\lambda)$	Linear-Only Encryption
BISW (LWE/RLWE) [BISW17]	$\tilde{O}(\lambda \lvert C \rvert)$	$\tilde{O}(\lambda)$	Linear-Only Vector Encryption

For simplicity, we ignore low order terms $\text{poly}(\lambda, \log|C|)$

Construction	Prover Complexity	Proof Size	Assumption
CS Proofs [Mic94]	$\tilde{O}(\lvert C \rvert)$	$\tilde{O}(\lambda^2)$	Random Oracle
Groth [Gro16]	$\tilde{O}(\lambda \lvert C \rvert)$	$\tilde{O}(\lambda)$	Generic Group
Groth [Gro10]	$\tilde{O}(\lambda \lvert C \rvert^2 + \lvert C \rvert \lambda^2)$	$\tilde{O}(\lambda)$	Knowledge of Exponent
GGPR [GGPR12]	$\tilde{O}(\lambda \lvert C \rvert)$	$\tilde{O}(\lambda)$	Knowledge of Exponent
BCIOP (Pairing) [BCIOP13]	$\tilde{O}(\lambda \lvert C \rvert)$	$\tilde{O}(\lambda)$	Linear-Only Encryption
BISW (LWE/RLWE) [BISW17]	$\tilde{O}(\lambda \lvert C \rvert)$	$\tilde{O}(\lambda)$	Linear-Only Vector Encryption

For simplicity, we ignore low order terms $\mathrm{poly}(\lambda, \log|C|)$

Construction	Prover Complexity	Proof Size	Assumption
CS Proofs [Mic94]	$\tilde{O}(\|C\|)$	$\tilde{O}(\lambda^2)$	Random Oracle
Groth [Gro16]	$\tilde{O}(\lambda\|C\|)$	$\tilde{O}(\lambda)$	Generic Group
Groth [Gro10]	$\tilde{O}(\lambda\|C\|^2 + \|C\|\lambda^2)$	$\tilde{O}(\lambda)$	Knowledge of Exponent
GGPR [GGPR12]	$\tilde{O}(\lambda\|C\|)$	$\tilde{O}(\lambda)$	
BCIOP (Pairing) [BCIOP13]	$\tilde{O}(\lambda\|C\|)$	$\tilde{O}(\lambda)$	Linear-Only Encryption
BISW (LWE/RLWE) [BISW17]	$\tilde{O}(\lambda\|C\|)$	$\tilde{O}(\lambda)$	Linear-Only Vector Encryption
This work	$\tilde{O}(\|C\|)$	$\tilde{O}(\lambda)$	Linear-Only Vector Encryption

This Work

New framework for building SNARGs (following [BCIOP13, BISW17])

Step 1 (information-theoretic):

- Linear multi-prover interactive proofs (linear MIPs)
- This work: first construction of a quasi-optimal linear MIP

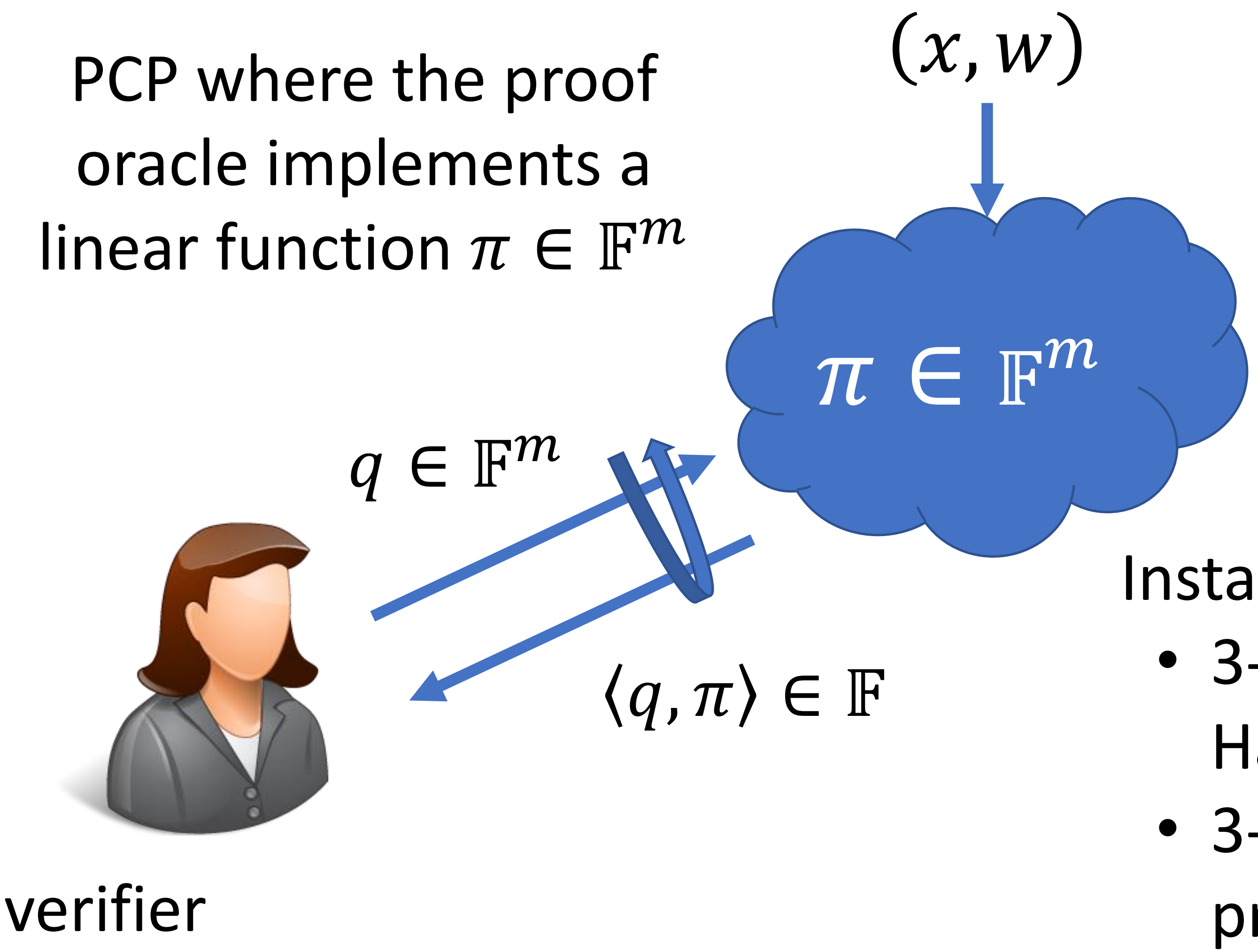
Step 2 (cryptographic):

- Linear-only vector encryption to simulate linear MIP model
- This work: linear MIP $\Longrightarrow$ preprocessing SNARG

Results yield the first quasi-optimal SNARG (from linear-only vector encryption over rings)

Linear PCPs [IKO07]

PCP where the proof oracle implements a linear function $\pi \in \mathbb{F}^m$



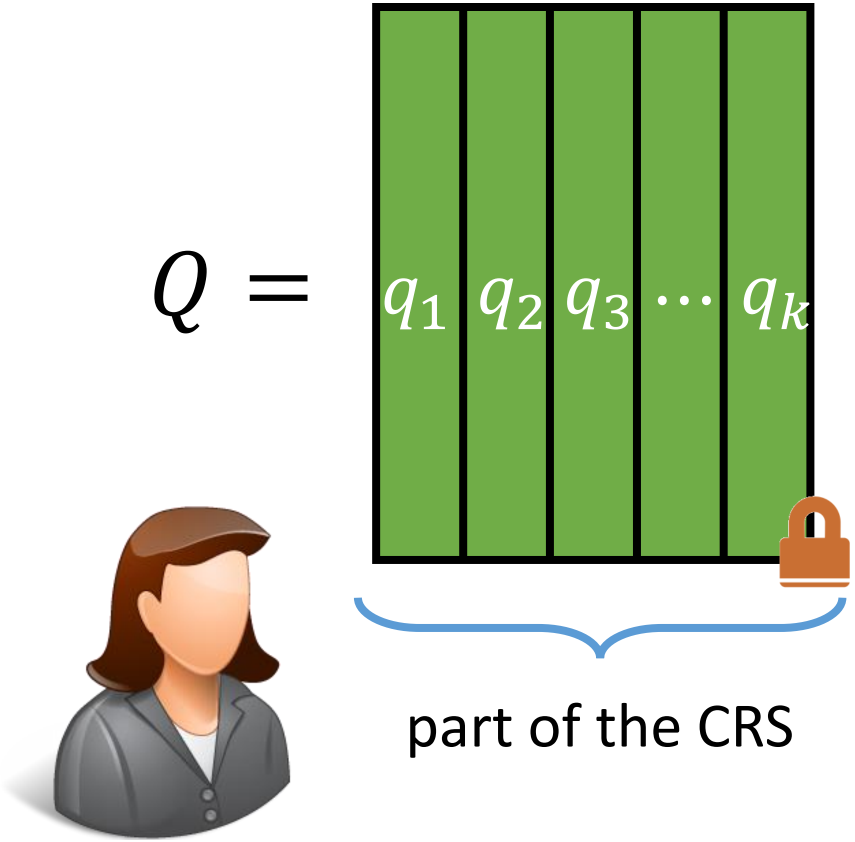
In these instantiations, verifier is oblivious (queries independent of statement)

Instantiations:

- 3-query LPCP based on the Walsh-Hadamard code: $m = O(|C|^2)$ [ALMSS92]
- 3-query LPCP based on quadratic span programs: $m = O(|C|)$ [GGPR13]

From Linear PCPs to SNARGs [BCIOP13]

Verifier encrypts its queries using a linear-only encryption scheme



CPs to SNARGs [BCIOP13]

Encryption scheme that only supports linear homomorphism

Verifier encrypts its queries using a linear-only encryption scheme

$$Q = \begin{bmatrix} q_1 & q_2 & q_3 & \cdots & q_k \end{bmatrix}$$

part of the CRS

From Linear PCPs to SNARGs [BCIOP13]

Verifier encrypts its queries using a <u>linear-only</u> encryption scheme

$$Q = \begin{bmatrix} q_1 & q_2 & q_3 & \cdots & q_k \end{bmatrix}$$

part of the CRS

Prover constructs linear PCP π from (x, w)

$$(x, w) \downarrow \pi \in \mathbb{F}^m$$

Prover homomorphically computes responses to linear PCP queries

$\langle \pi, q_1 \rangle$	$\langle \pi, q_2 \rangle$	$\cdots$	$\langle \pi, q_k \rangle$

SNARG proof

From Linear PCPs to SNARGs [BCIOP13]

Evaluating inner product requires $O(km)$ homomorphic operations on ciphertexts: prover complexity $O(\lambda) \cdot O(km) = O(\lambda|C|)$

Prover constructs linear PCP π from (x, w)

$$(x, w) \downarrow \pi \in \mathbb{F}^m$$

$$Q = \begin{bmatrix} q_1 & q_2 & q_3 & \cdots & q_k \end{bmatrix}$$

Proof consists of a <u>constant</u> number of ciphertexts: total length $O(\lambda)$ bits

Prover homomorphically computes responses to linear PCP queries

$\langle \pi, q_1 \rangle$	$\langle \pi, q_2 \rangle$	$\cdots$	$\langle \pi, q_k \rangle$

SNARG proof

From Linear PCPs to SNARGs [BCIOP13]

Evaluating inner product requires $O(km)$ homomorphic operations on ciphertexts: prover complexity $O(\lambda) \cdot O(km) = O(\lambda|C|)$

$Q =$ | q_1 | q_2 | q_3 | $\cdots$ | q_k |

Proof consists of a <u>constant</u> number of ciphertexts: total length $O(\lambda)$ bits

Prover constructs linear PCP π from (x, w)

(x, w)

We pay $O(\lambda)$ for each homomorphic operation. Can we reduce this?

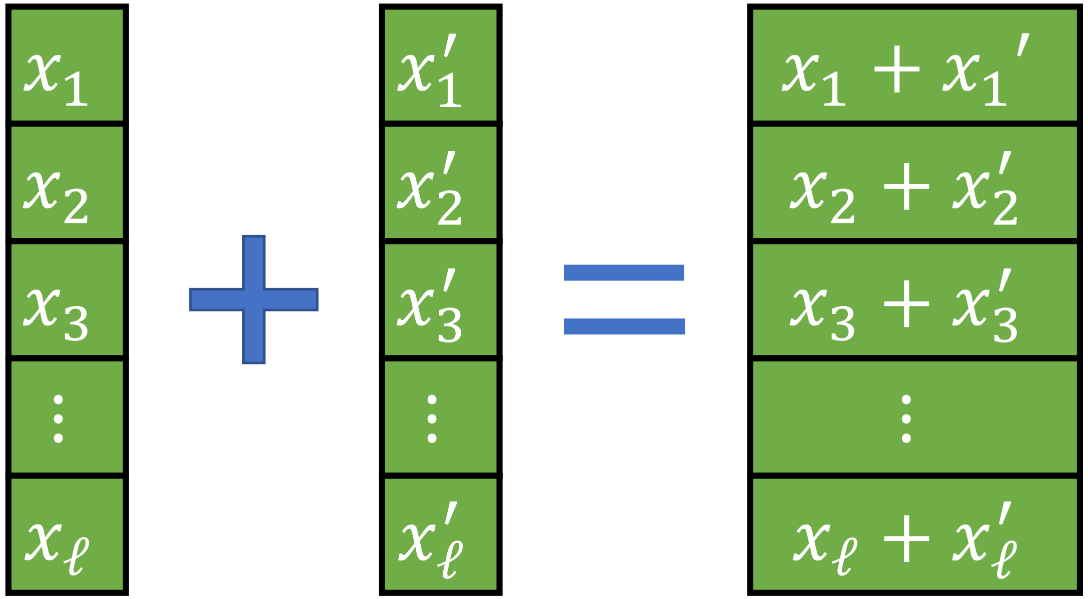
Prove … es
responses … eries

$\langle \pi, q_1 \rangle$	$\langle \pi, q_2 \rangle$	$\cdots$	$\langle \pi, q_k \rangle$

SNARG proof

Linear-Only Encryption over Rings

Consider encryption scheme over a polynomial ring $R_p = \mathbb{Z}_p[x]/\Phi_\ell(x) \cong \mathbb{F}_p^\ell$



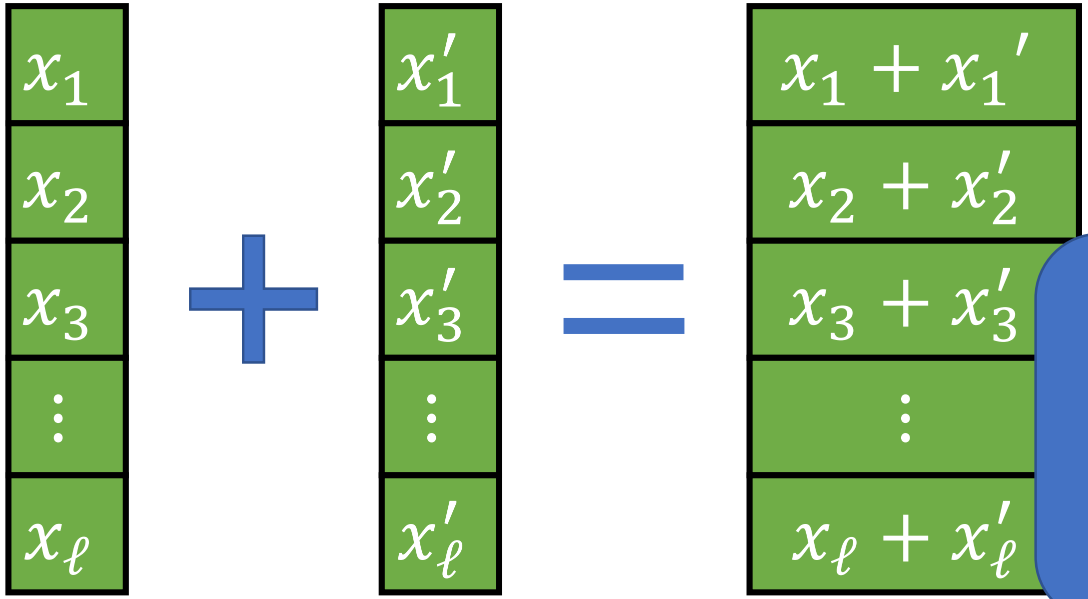
Homomorphic operations correspond to <u>component-wise</u> additions and scalar multiplications

Plaintext space can be viewed as a vector of field elements

Using RLWE-based encryption schemes, can encrypt $\ell = \tilde{O}(\lambda)$ field elements ($p = \text{poly}(\lambda)$) with ciphertexts of size $\tilde{O}(\lambda)$

Linear-Only Encryption over Rings

Consider encryption scheme over a polynomial ring $R_p = \mathbb{Z}_p[x]/\Phi_\ell(x) \cong \mathbb{F}_p^\ell$



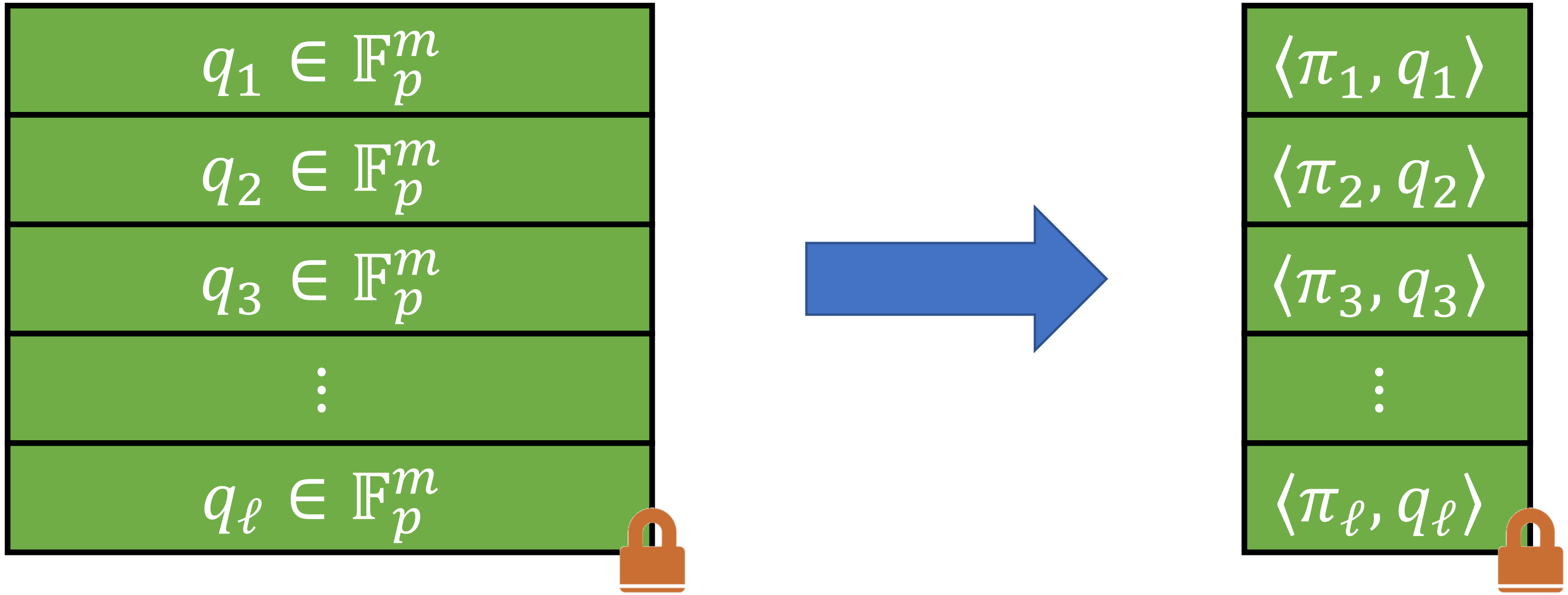
Homomorphic operations

Amortized cost of homomorphic operation on a single field element is $\mathrm{polylog}(\lambda)$

Plaintext space can be viewed as a vector of field elements

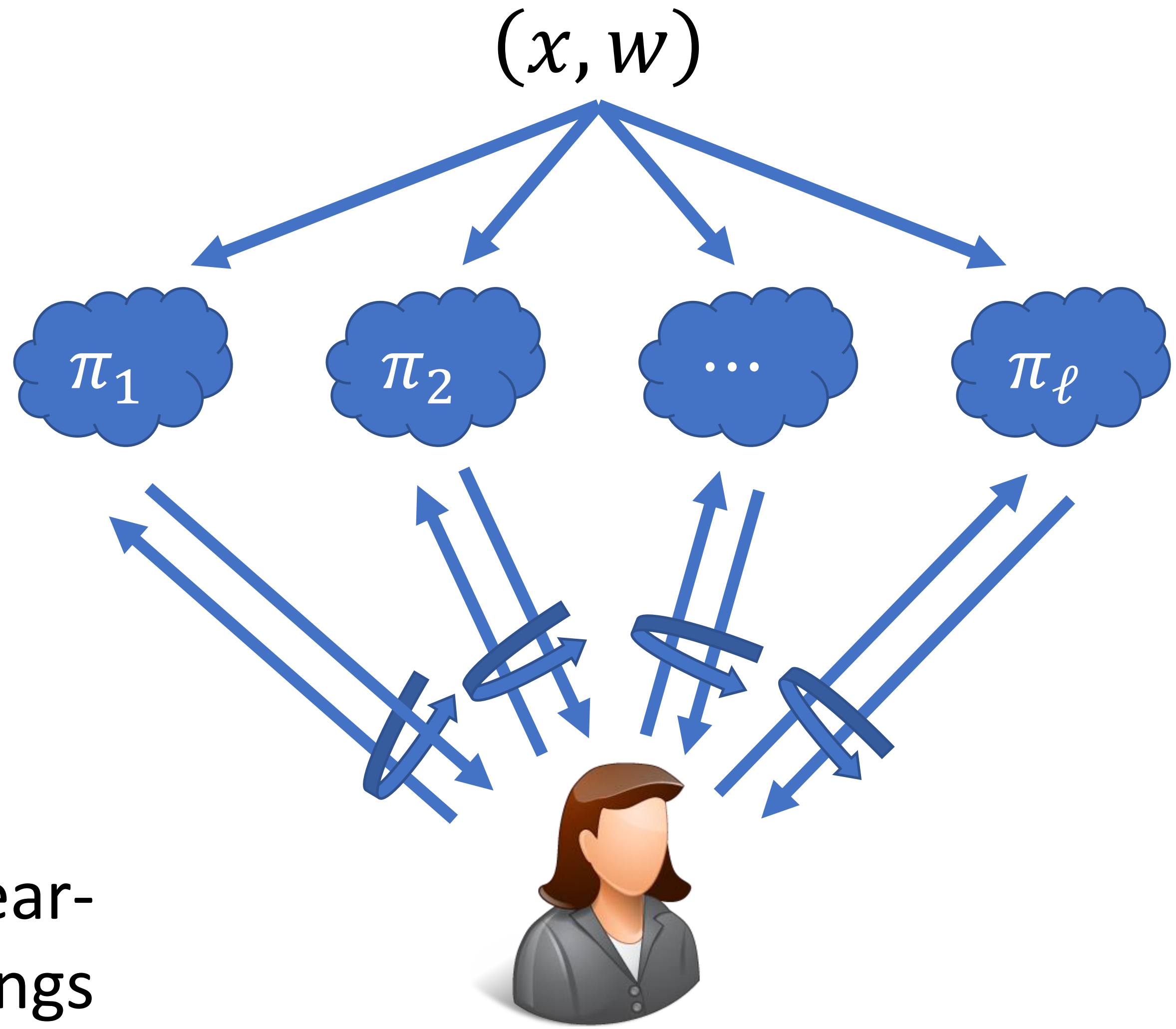
Using RLWE-based encryption schemes, can encrypt $\ell = \tilde{O}(\lambda)$ field elements ($p = \mathrm{poly}(\lambda)$) with ciphertexts of size $\tilde{O}(\lambda)$

Linear-Only Encryption over Rings



Given encrypted set of query vectors, prover can homomorphically apply <u>independent</u> linear functions to each slot

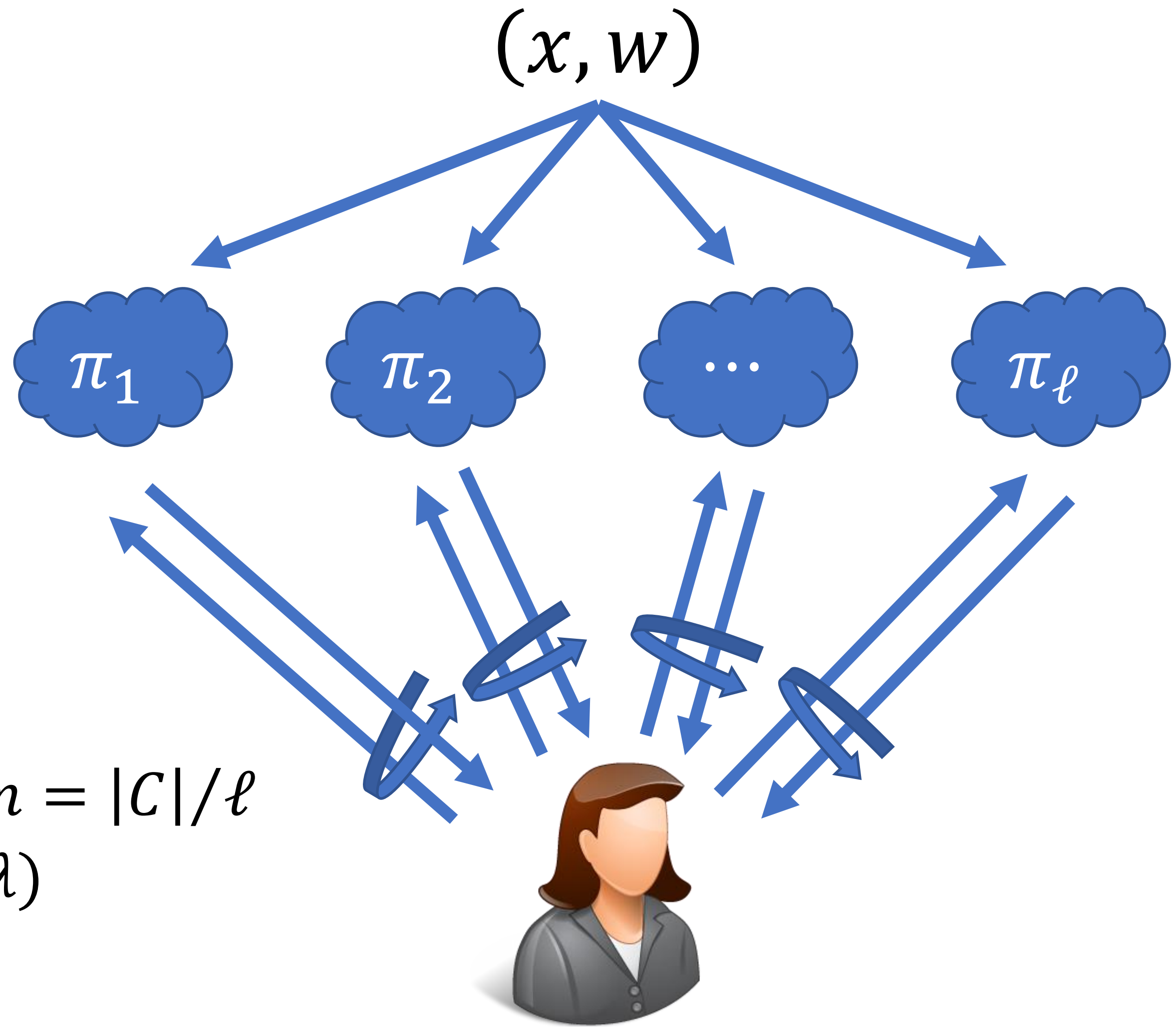
Linear Multi-Prover Interactive Proofs (MIPs)



Verifier has oracle access to <u>multiple</u> linear proof oracles

Can convert linear MIP to preprocessing SNARG using linear-only (vector) encryption over rings

Linear Multi-Prover Interactive Proofs (MIPs)



Suppose

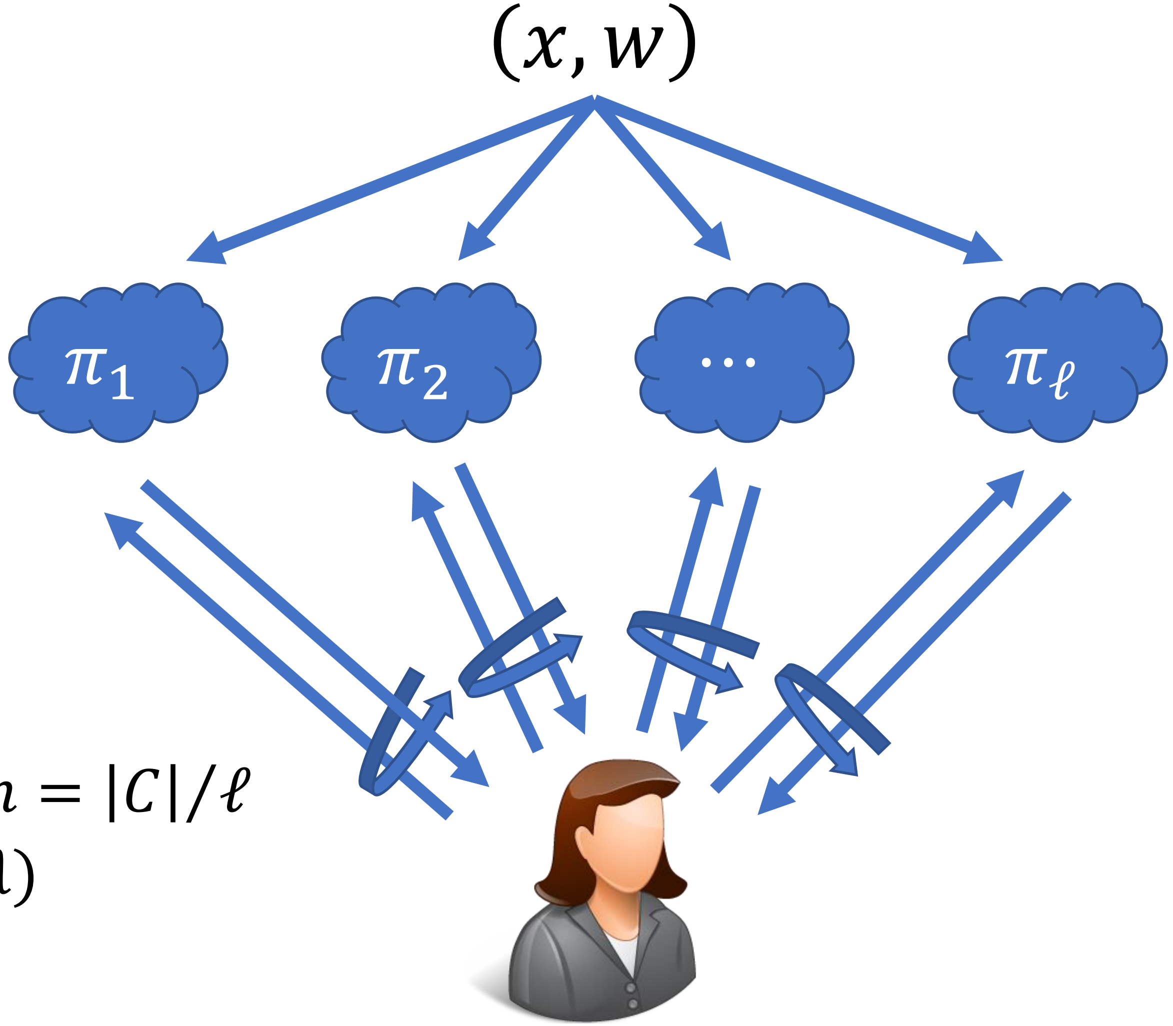
- Number of provers $\ell = \tilde{O}(\lambda)$
- Proofs $\pi_1, \ldots, \pi_\ell \in \mathbb{F}_p^m$ where $m = |C|/\ell$
- Number of queries is $\text{polylog}(\lambda)$

Then, linear MIP is quasi-optimal

Linear Multi-Prover Interactive Proofs (MIPs)

Prover complexity:
$\tilde{O}(\ell m) = \tilde{O}(|C|)$

Proof size:
$O\big(\ell \cdot \text{polylog}(\lambda)\big) = \tilde{O}(\lambda)$



Suppose

- Number of provers $\ell = \tilde{O}(\lambda)$
- Proofs $\pi_1, \dots, \pi_\ell \in \mathbb{F}_p^m$ where $m = |C|/\ell$
- Number of queries is $\text{polylog}(\lambda)$

Then, linear MIP is quasi-optimal

Linear Multi-Prover Interactive Proofs (MIPs)

Goal: Construct quasi-optimal linear MIP (with soundness $2^{-\lambda}$) and following properties:

- Number of provers is $\tilde{O}(\lambda)$
- Each proof has length $\tilde{O}(|C|/\lambda)$
- Proofs are over a polynomial-size field: $p = \mathrm{poly}(\lambda)$
- Query complexity is $\mathrm{polylog}(\lambda)$

More provers, shorter (individual) proofs

Linear Multi-Prover Interactive Proofs (MIPs)

Goal: Construct quasi-optimal linear MIP (with soundness $2^{-\lambda}$) and following properties:

- Number of provers is $\tilde{O}(\lambda)$
- Each proof has length $\tilde{O}(|C|/\lambda)$
- Proofs are over a polynomial-size field: $p = \text{poly}(\lambda)$
- Query complexity is $\text{polylog}(\lambda)$

Linear PCPs used in [BCIOP13] require a field of size $2^{\Omega(\lambda)}$

Can we use existing linear PCPs?

Linear Multi-Prover Interactive Proofs (MIPs)

Goal: Construct quasi-optimal linear MIP (with soundness $2^{-\lambda}$) and following properties:

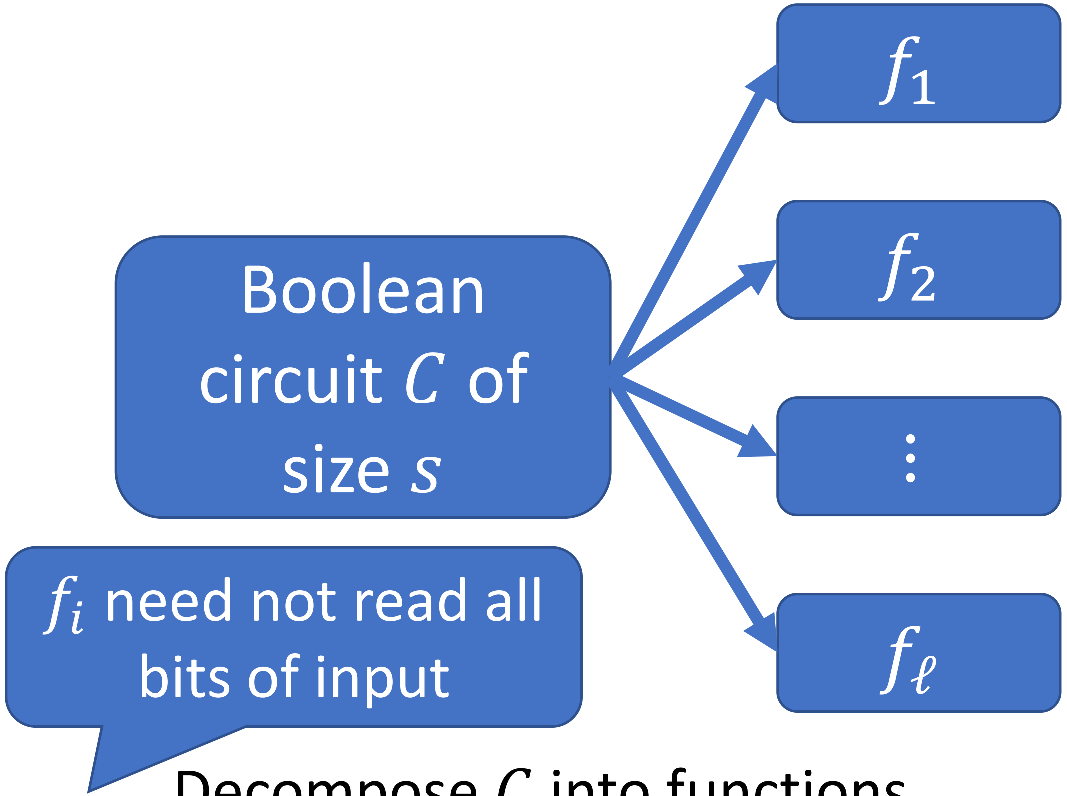
- Number of provers is $\tilde{O}(\lambda)$
- Each proof has length $\tilde{O}(|C|/\lambda)$
- Proofs are over a polynomial-size field: $p = \text{poly}(\lambda)$
- Query complexity is $\text{polylog}(\lambda)$

Linear PCPs used in [BISW17] have query complexity $\Omega(\lambda)$

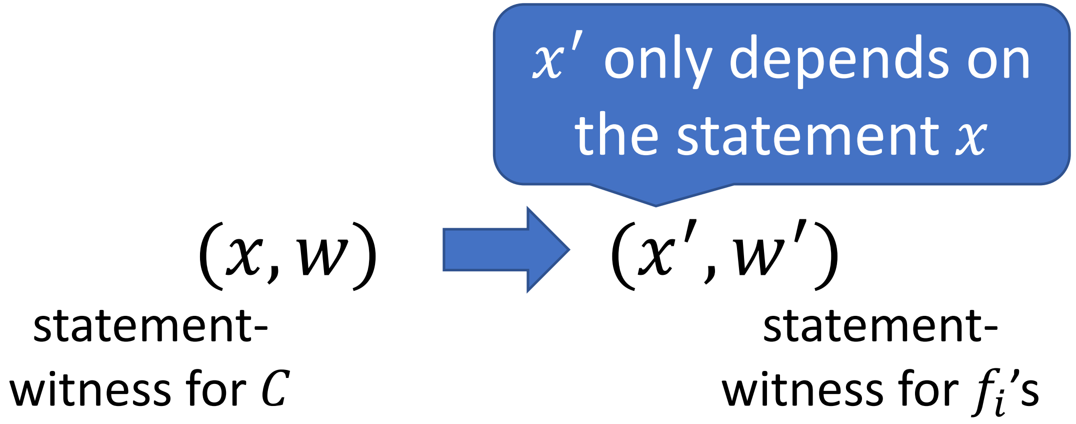
Can we use existing linear PCPs?

This work: Construction of a quasi-optimal linear MIP for Boolean circuit satisfiability

Quasi-Optimal Linear MIP



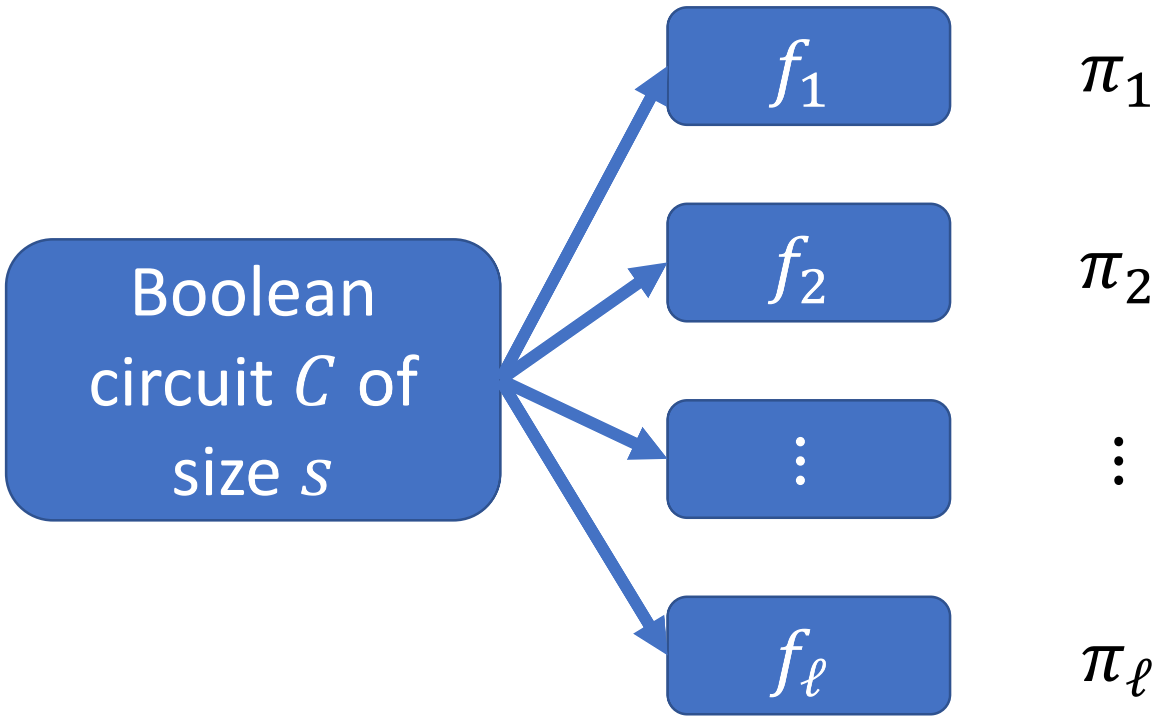
Decompose C into functions $f_1, \ldots, f_\ell$, where each function can be computed by a circuit of size s/ℓ



- **Completeness:** If $C(x, w) = 1$, then $f_i(x', w') = 1$ for all i
- **Robustness:** If $x \notin \mathcal{L}$, then for all w', at most $2/3$ of $f_i(x', w') = 1$
- **Efficiency:** (x', w') can be computed by a circuit of size $\tilde{O}(s)$

Quasi-Optimal Linear MIP

$(x, w) \Rightarrow (x', w')$

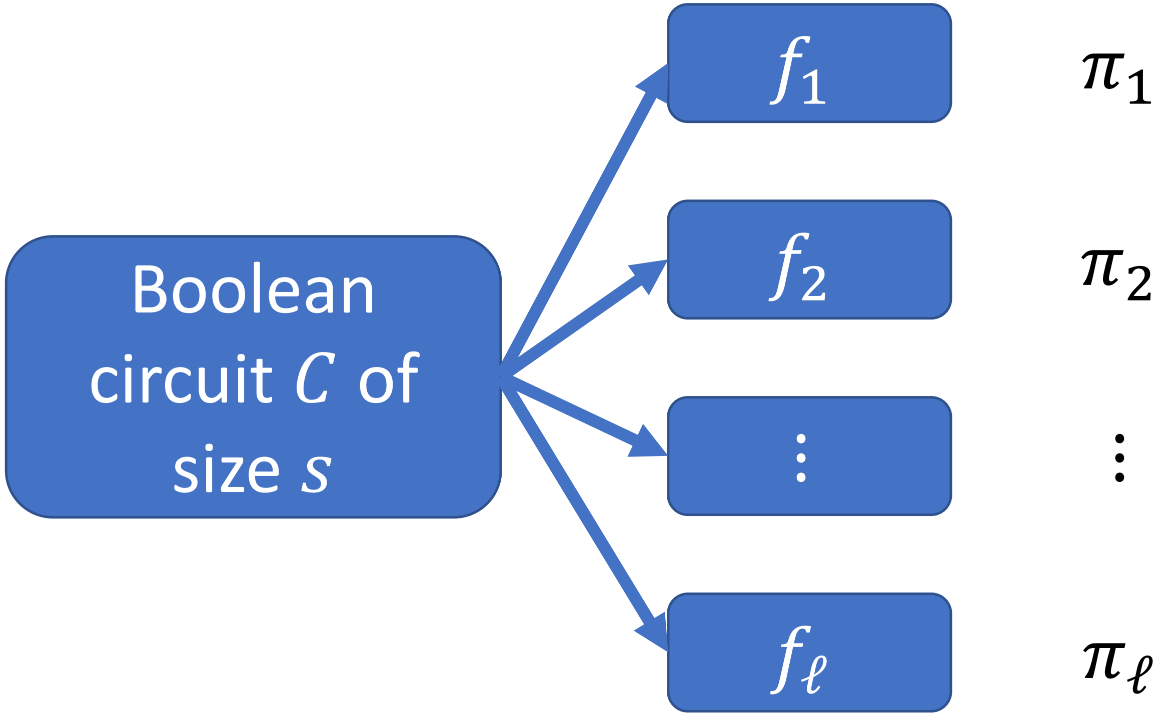


Using constant-query linear PCP based on QSPs [GGPR13], $\pi_i \in \mathbb{F}_p^m$ where $m = O(|C|/\ell)$ and provides soundness $1/\text{poly}(\lambda)$

π_i: linear PCP that $f_i(x', \cdot)$ is satisfiable
(instantiated over $\mathbb{F}_p$ where $p = \text{poly}(\lambda)$)

Quasi-Optimal Linear MIP

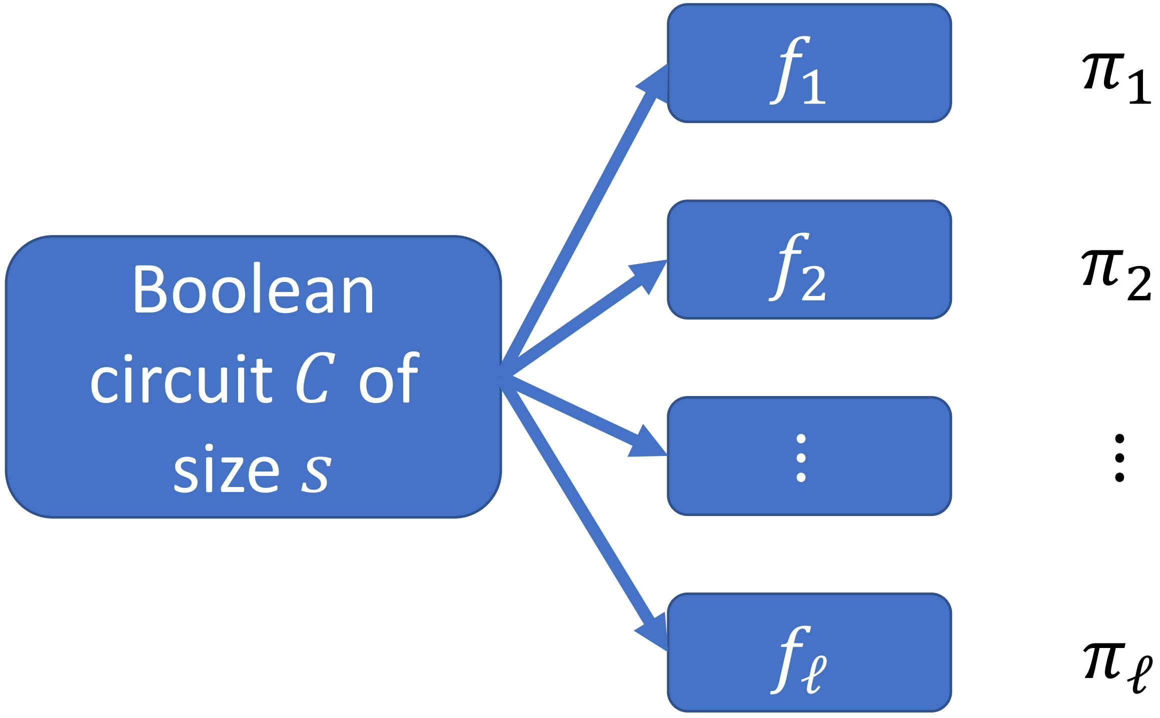
$(x, w) \Rightarrow (x', w')$



Verifier invokes linear PCP verifier for each instance

π_i: linear PCP that $f_i(x', \cdot)$ is satisfiable
(instantiated over $\mathbb{F}_p$ where $p = \text{poly}(\lambda)$)

Quasi-Optimal Linear MIP



- **Completeness:** Follows by completeness of decomposition and linear PCPs
- **Soundness:** Each linear PCP provides $1/\mathrm{poly}(\lambda)$ soundness and for false statement, at least $1/3$ of the statements are false, so if $\ell = \Omega(\lambda)$, verifier accepts with probability $2^{-\Omega(\lambda)}$

π_i: linear PCP that $f_i(x', \cdot)$ is satisfiable
(instantiated over $\mathbb{F}_p$ where $p = \mathrm{poly}(\lambda)$)

Quasi-Optimal Linear MIP

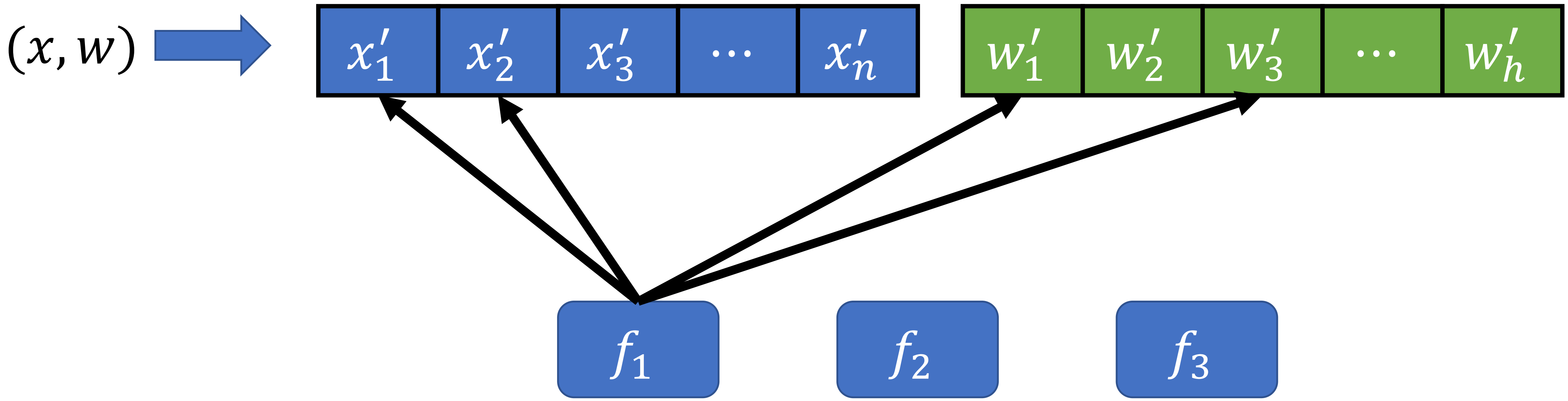
Robustness: If $x \notin \mathcal{L}$, then for all w', at most $2/3$ of $f_i(x', w') = 1$

For false x, no single w' can simultaneously satisfy $f_i(x', \cdot)$; however, all of the $f_i(x', \cdot)$ could individually be satisfiable

- **Completeness:** Follows by completeness of decomposition and linear PCPs
- **Soundness:** Each linear PCP provides $1/\text{poly}(\lambda)$ soundness and for false statement, at least $1/3$ of the statements are false, so if $\ell = \Omega(\lambda)$, verifier accepts with probability $2^{-\Omega(\lambda)}$

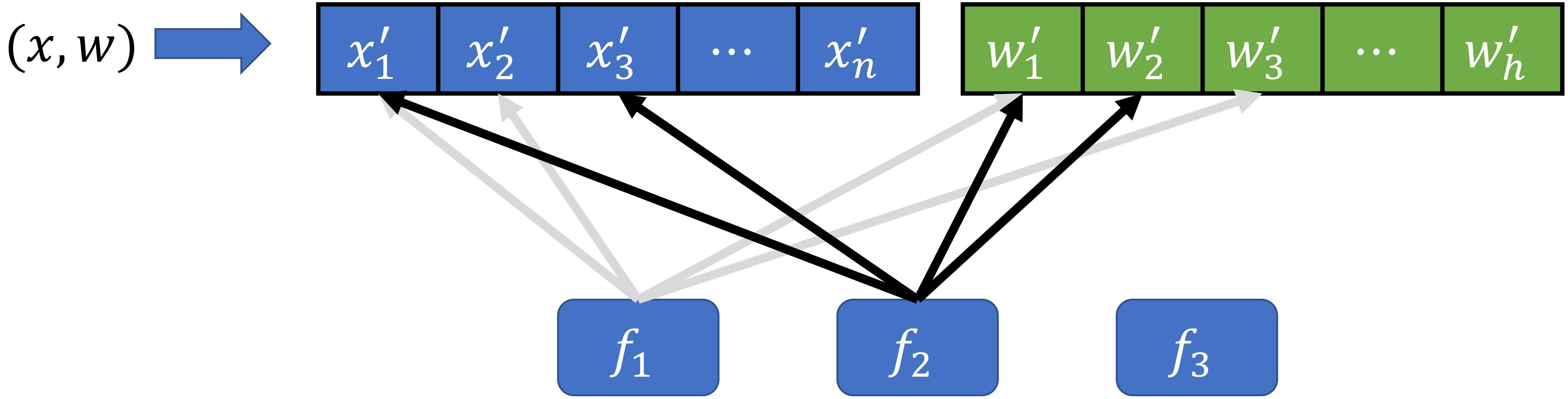
Problematic however if prover uses different (x', w') to construct proofs for different f_i's

Consistency Checking



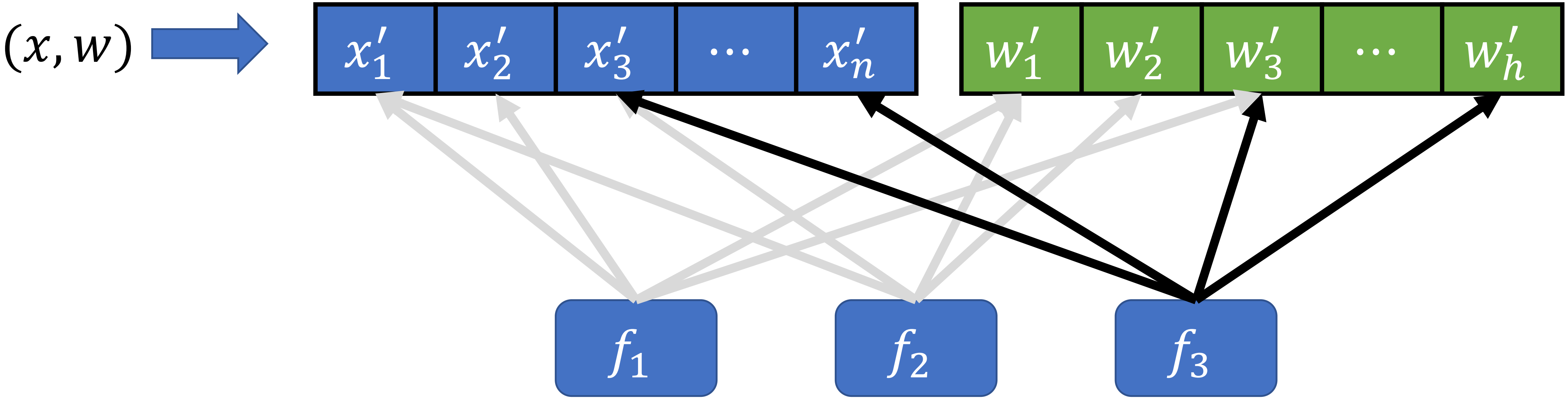
Each constraint function f_i reads a few bits of the common statement-witness (x', w')

Consistency Checking



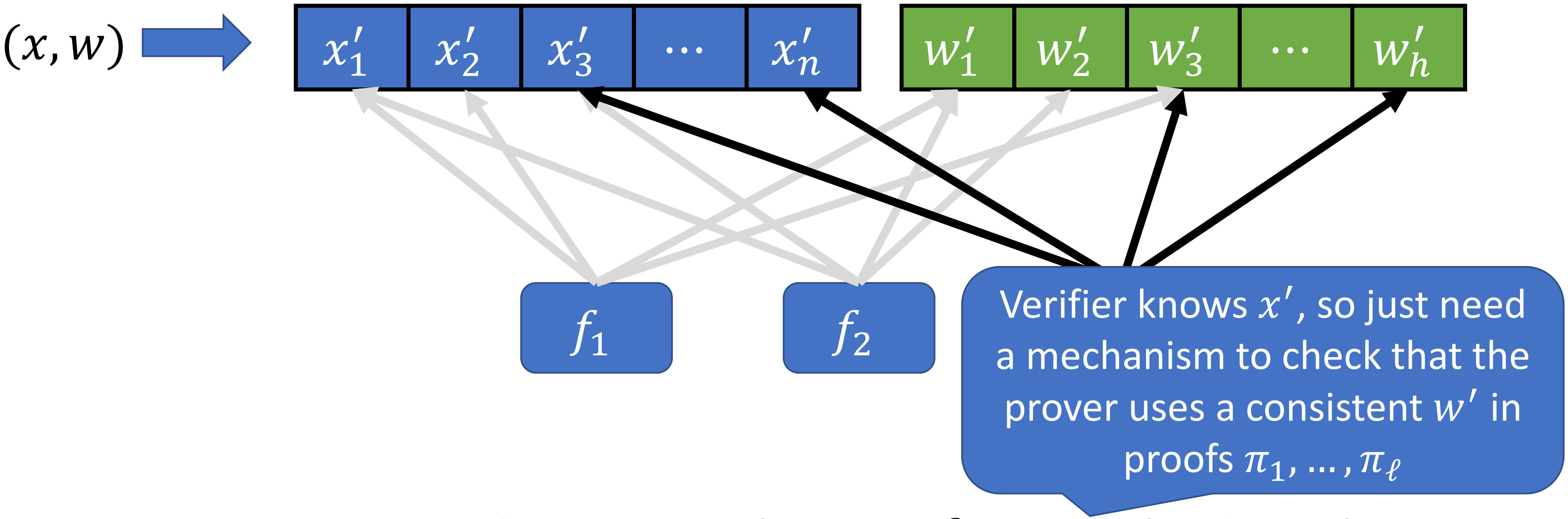
Each constraint function f_i reads a few bits of the common statement-witness (x', w')

Consistency Checking



Each constraint function f_i reads a few bits of the common statement-witness (x', w')

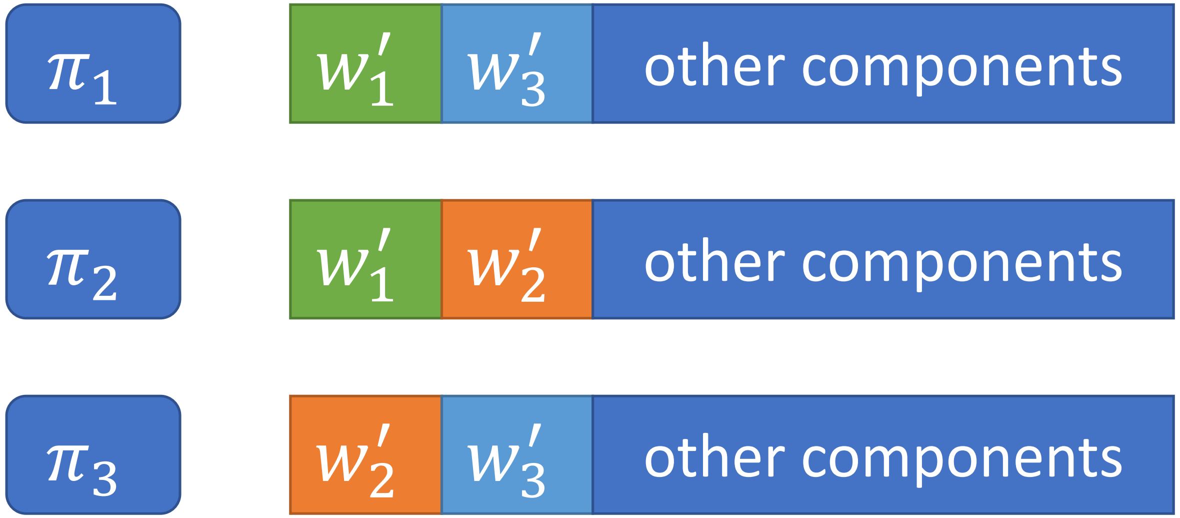
Consistency Checking



Each constraint function f_i reads a few bits of the common statement-witness (x', w')

Consistency Checking

Require that linear PCPs are systematic: linear PCP π contains a copy of the witness:



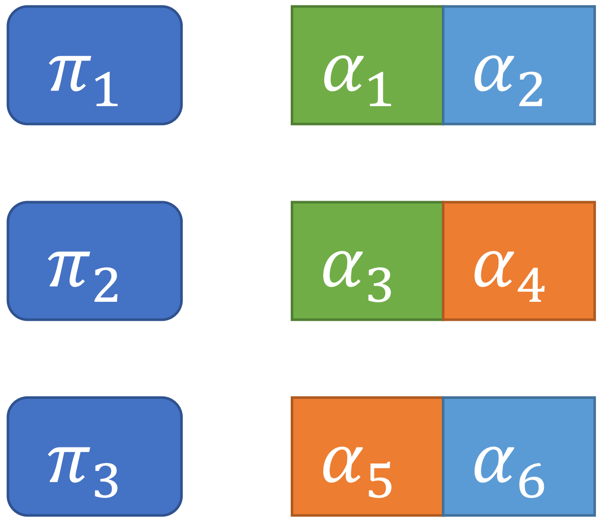
Goal: check that assignments to w' are consistent via linear queries to π_i

First few components of proof correspond to witness associated with the statement

Each proof induces an assignment to a few bits of the common witness w'

Consistency Checking

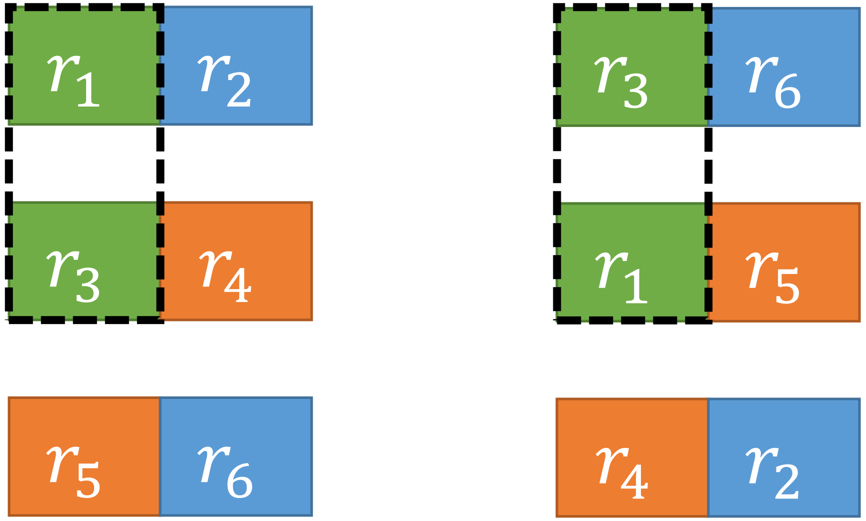
Global consistency check [Gro09]



Proof components

Consistent if $\alpha_1 = \alpha_3$, $\alpha_2 = \alpha_6$, and $\alpha_4 = \alpha_5$

Permute queries according to replication pattern



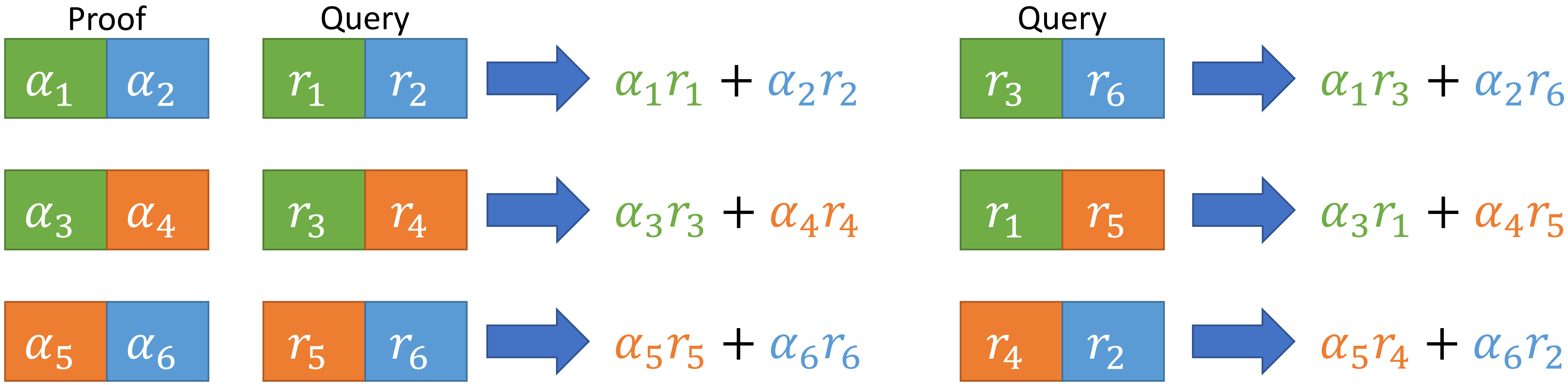
Random components

Permuted components

Verifier's queries (remaining components padded with 0s)

Consistency Checking

Global consistency check [Gro09]



If $\alpha_1 = \alpha_3$, then $\alpha_1 r_1 + \alpha_3 r_3 = \alpha_1 r_3 + \alpha_3 r_1$

If $\alpha_2 = \alpha_6$, then $\alpha_2 r_2 + \alpha_6 r_6 = \alpha_2 r_6 + \alpha_2 r_2$

If $\alpha_4 = \alpha_5$, then $\alpha_4 r_4 + \alpha_5 r_5 = \alpha_4 r_5 + \alpha_5 r_4$

Consistency Checking

Global consistency check [Gro09]

Proof	Query		Query	
α_1 α_2	r_1 r_2	$\Rightarrow \alpha_1 r_1 + \alpha_2 r_2$	r_3 r_6	$\Rightarrow \alpha_1 r_3 + \alpha_2 r_6$
α_3 α_4	r_3 r_4	$\Rightarrow \alpha_3 r_3 + \alpha_4 r_4$	r_1 r_5	$\Rightarrow \alpha_3 r_1 + \alpha_4 r_5$
α_5 α_6	r_5 r_6	$\Rightarrow \alpha_5 r_5 + \alpha_6 r_6$	r_4 r_2	$\Rightarrow \alpha_5 r_4 + \alpha_6 r_2$

If assignment is consistent, then sum of responses to first query equals sum of responses to second query

Consistency Checking

Global consistency check [Gro09]

Proof		Query			Query		
α_1	α_2	r_1	r_2	$\Rightarrow \alpha_1 r_1 + \alpha_2 r_2$	r_3	r_6	$\Rightarrow \alpha_1 r_3 + \alpha_2 r_6$
α_3	α_4	r_3	r_4	$\Rightarrow \alpha_3 r_3 + \alpha_4 r_4$	r_1	r_5	$\Rightarrow \alpha_3 r_1 + \alpha_4 r_5$
α_5	α_6	r_5	r_6	$\Rightarrow \alpha_5 r_5 +$			$+ \alpha_6 r_2$

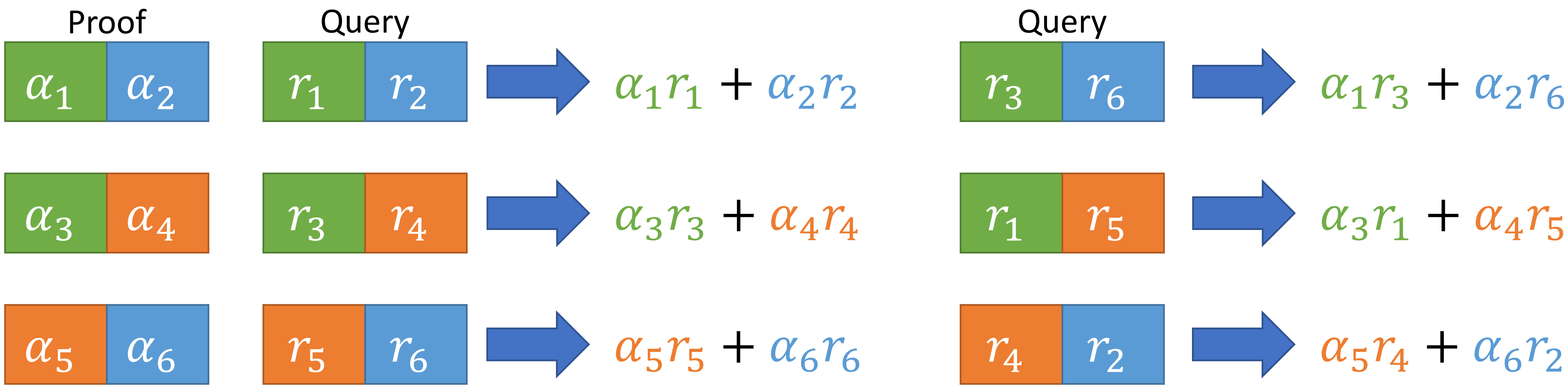
By the Schwartz-Zippel lemma, with probability $1/|\mathbb{F}|$ over choice of r_1, r_3, this holds

If $\alpha_1 \neq \alpha_3$, then $\alpha_1 r_1 + \alpha_3 r_3 \neq \alpha_1 r_3 + \alpha_3 r_1$

unless $r_1(\alpha_1 - \alpha_3) + r_3(\alpha_1 - \alpha_3) = 0$

Consistency Checking

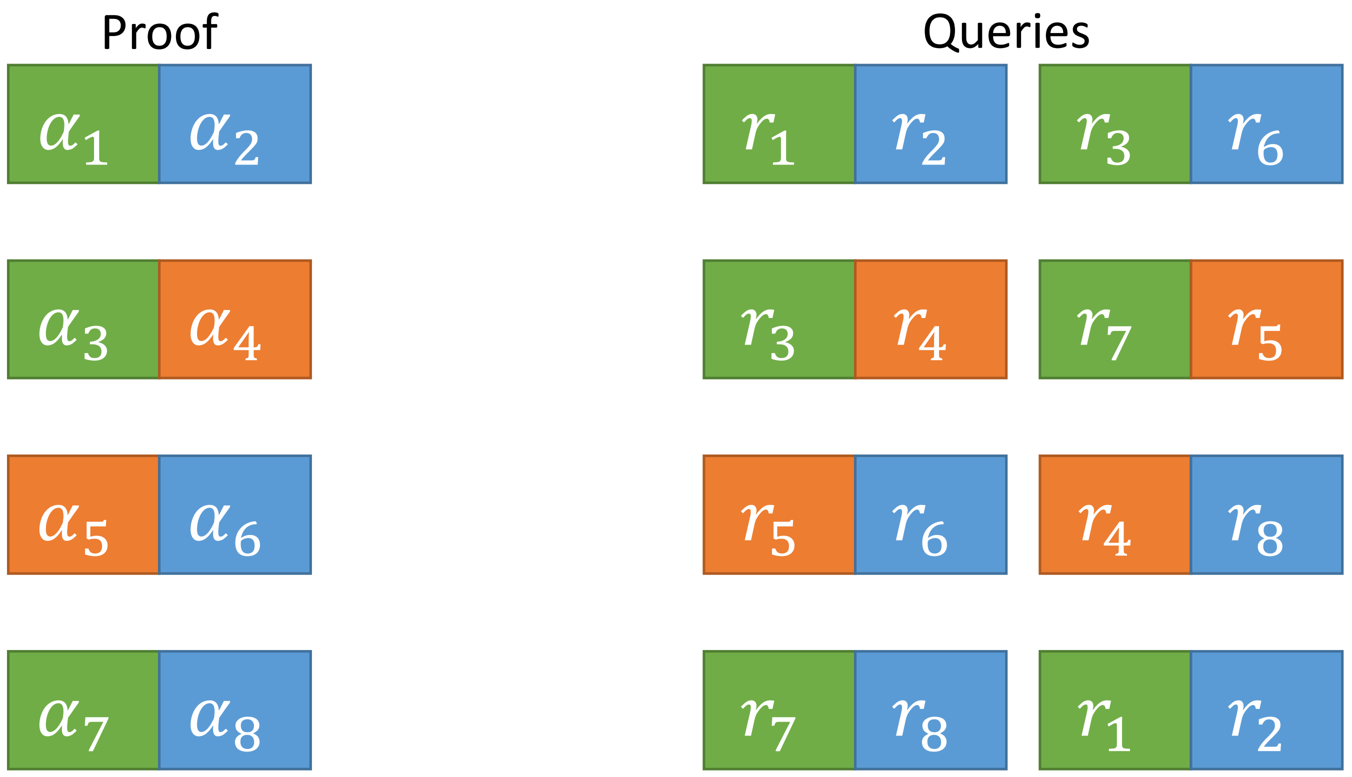
Global consistency check [Gro09]



We can detect inconsistent assignment with probability $1/|\mathbb{F}|$ — not sufficient for a quasi-optimal linear MIP with soundness $2^{-\lambda}$ since $|\mathbb{F}| = \text{poly}(\lambda)$

Consistency Checking

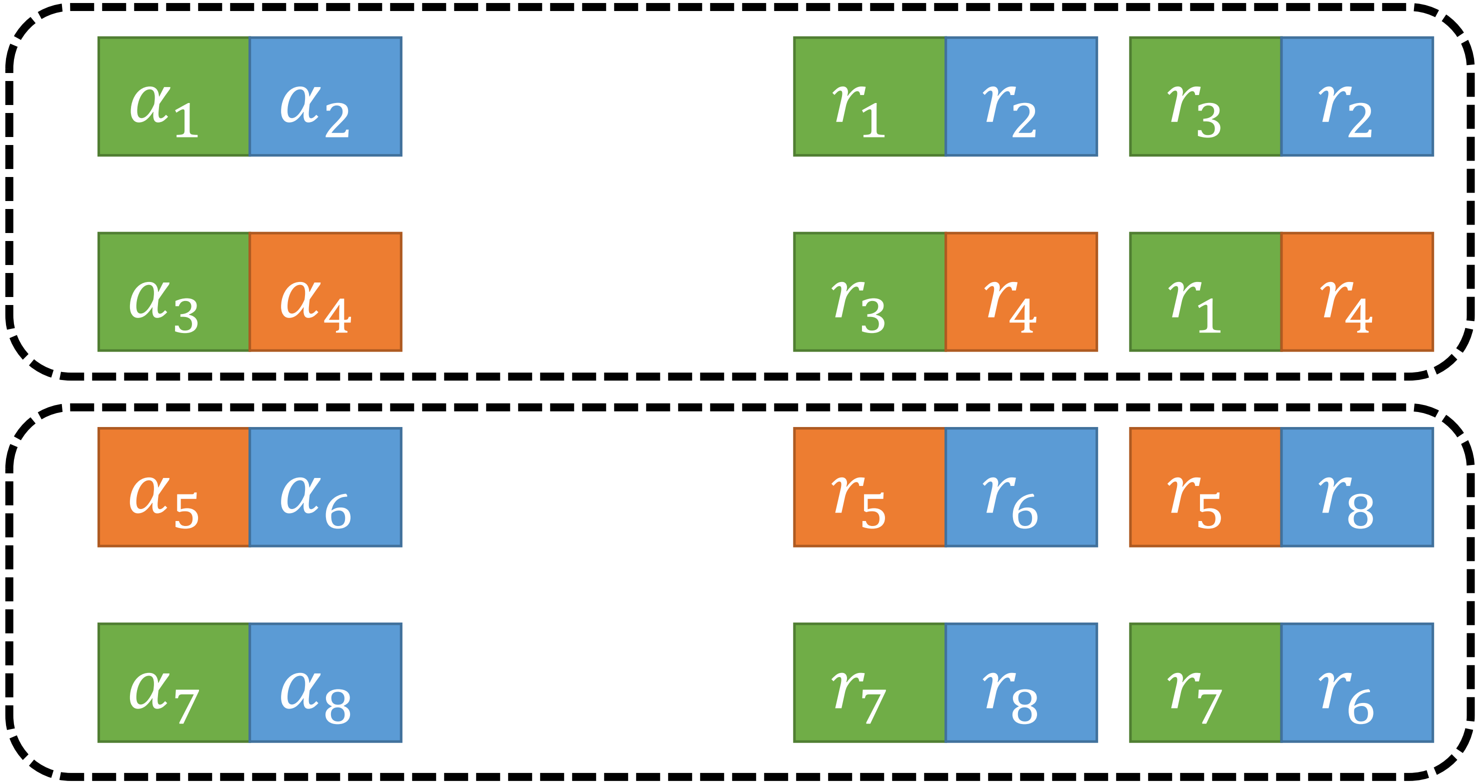
Key idea: pairwise consistency checks



Global consistency check: permute queries according to global replication pattern

Consistency Checking

Key idea: pairwise consistency checks



Local consistency check: permute queries according to local replication pattern

Only check consistency of variables that are replicated within the same block

If there are inconsistencies in the assignments in $\Omega(\lambda)$ blocks, then verifier rejects with probability $1 - 2^{-\Omega(\lambda)}$

Consistency Checking

Robustness: If $x \notin \mathcal{L}$, then for all w', at most $2/3$ of $f_i(x', w') = 1$

For a statement $x \notin \mathcal{L}$, robustness implies the following:

- There are $\Omega(\lambda)$ proofs π_i with respect to a consistent witness w'

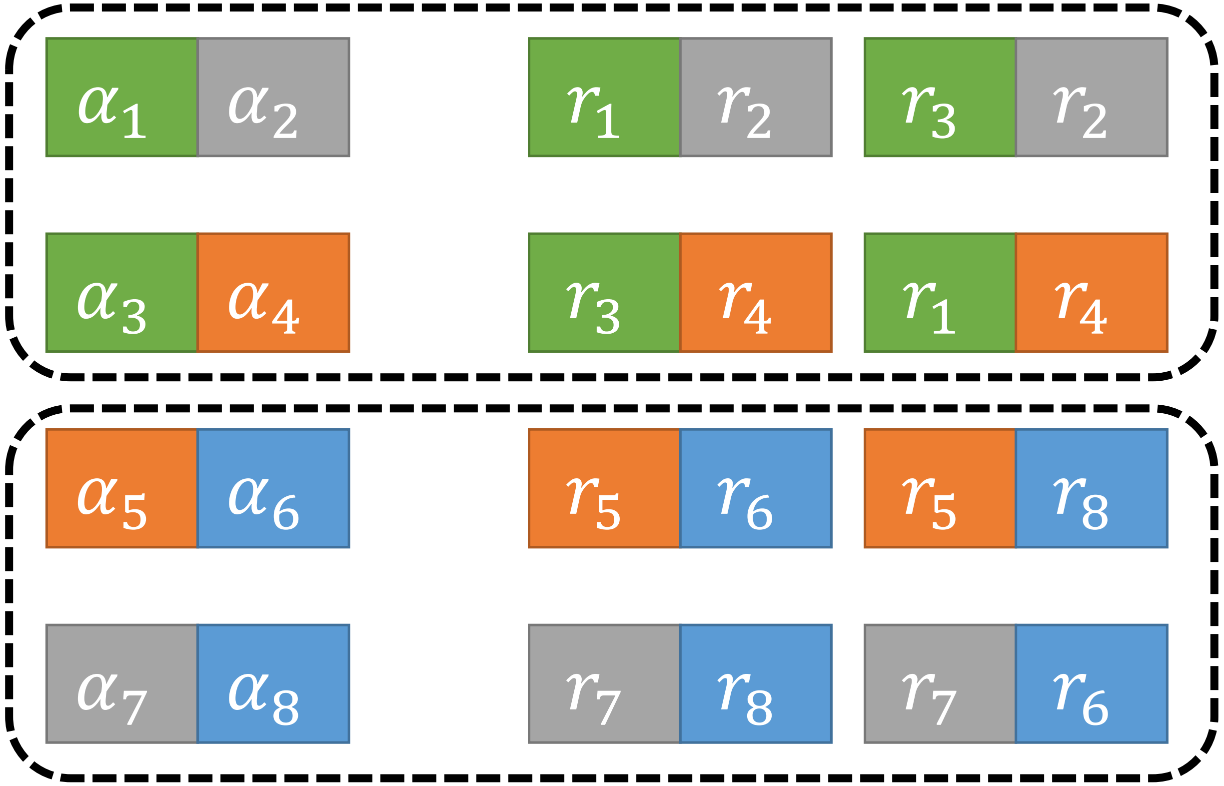
$\Omega(\lambda)$ linear PCP instances will fail to verify

- There are $\Omega(\lambda)$ disjoint pairs of rows containing an inconsistent assignment to some common variable in w'

Hope: pairwise consistency check rejects with probability $1 - 2^{-\Omega(\lambda)}$

Consistency Checking

Key idea: pairwise consistency checks



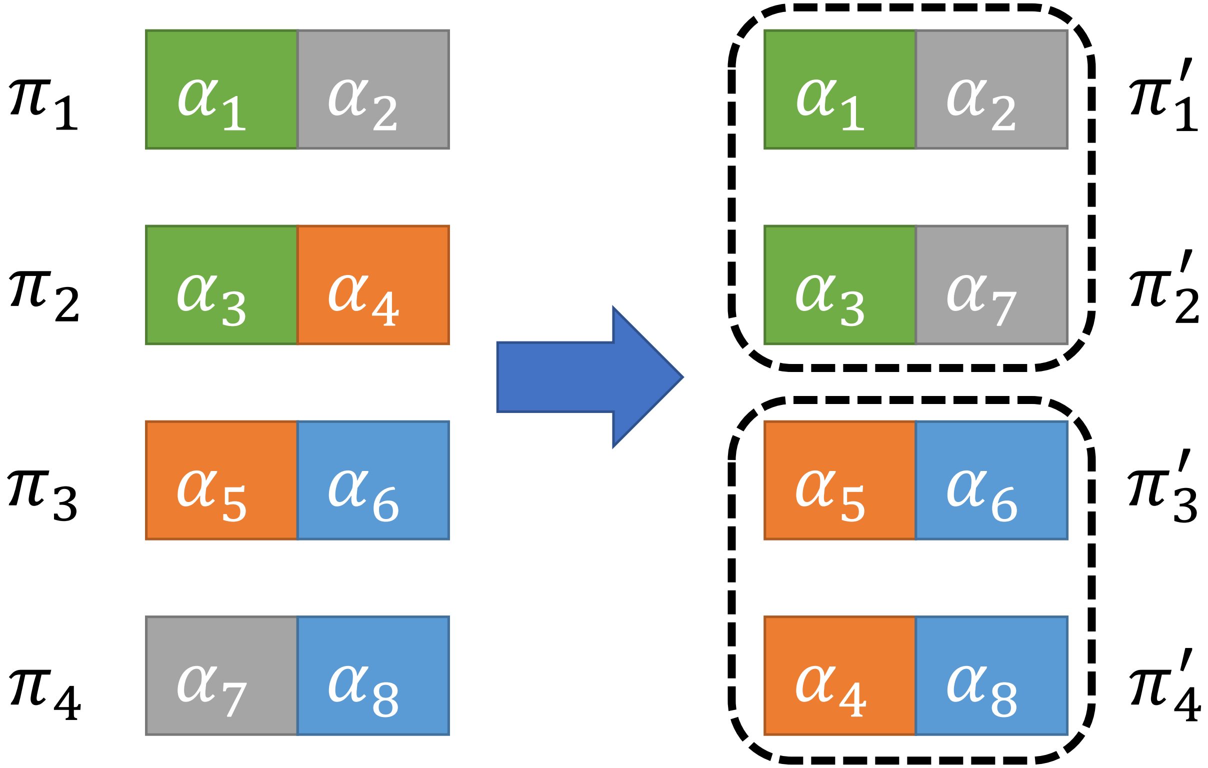
- For a false statement where all of the linear PCPs verify, there must be $\Omega(\lambda)$ disjoint pairs of rows containing an inconsistent assignment
- **Question:** Which pairs to check in the pairwise inconsistency check?

This partitioning checks whether $\alpha_1 = \alpha_3$ and $\alpha_6 = \alpha_8$ but not $\alpha_2 = \alpha_7$ or $\alpha_4 = \alpha_5$

Consistency Checking

Key idea: permute the entries in the proofs so that repeated variable appear in adjacent rows

Proofs used to verify linear PCP instances



Permuted proofs used for consistency checks

Replicated variables appear in adjacent rows (easily checked by pairwise consistency check)

Linear MIP consists original proofs π_i and the permuted proofs π_i'

Consistency Checking

Key idea: permute the entries in the proofs so that repeated variable appear in adjacent rows

Verifier also needs to check that $\{\pi_i\}_{i \in [\ell]}$ and $\{\pi_i'\}_{i \in [\ell]}$ are correctly permuted (in the linear MIP model)

- Use a Beneš network to "route" the different permutations – incurs $\log \lambda$ overhead
- Rely on randomization since prover can introduce $o(\lambda)$ inconsistencies without being detected

[See paper for details]

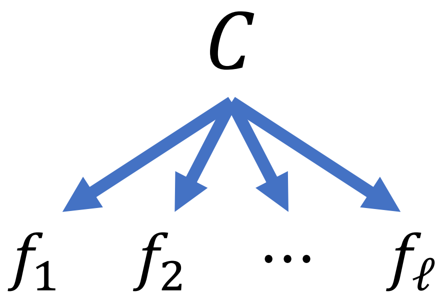
Linear MIP consists original proofs π_i and the permuted proofs π_i'

Permuted proofs used for consistency checks

Replicated variables appear in adjacent rows (easily checked by pairwise consistency check)

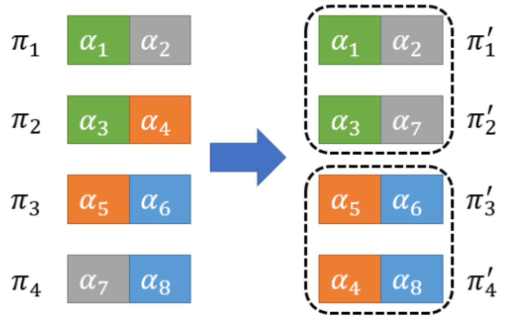
Quasi-Optimal Linear MIPs

Robust Decomposition



- Checking satisfiability of C corresponds to checking satisfiability of $f_1, \ldots, f_\ell$ (each of which can be checked by a circuit of size $|C|/\ell$)
- For a false statement, no single witness can simultaneously satisfy more than constant fraction of f_i

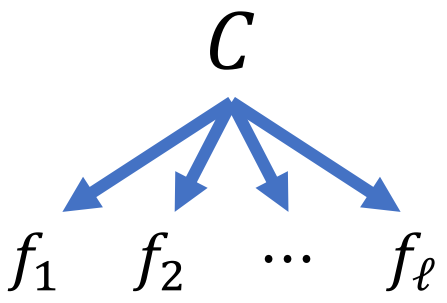
Consistency Check



- Check that consistent witness is used to prove satisfiability of each f_i
- Pairwise consistency checks used for soundness amplification

Quasi-Optimal Linear MIPs

Robust Decomposition

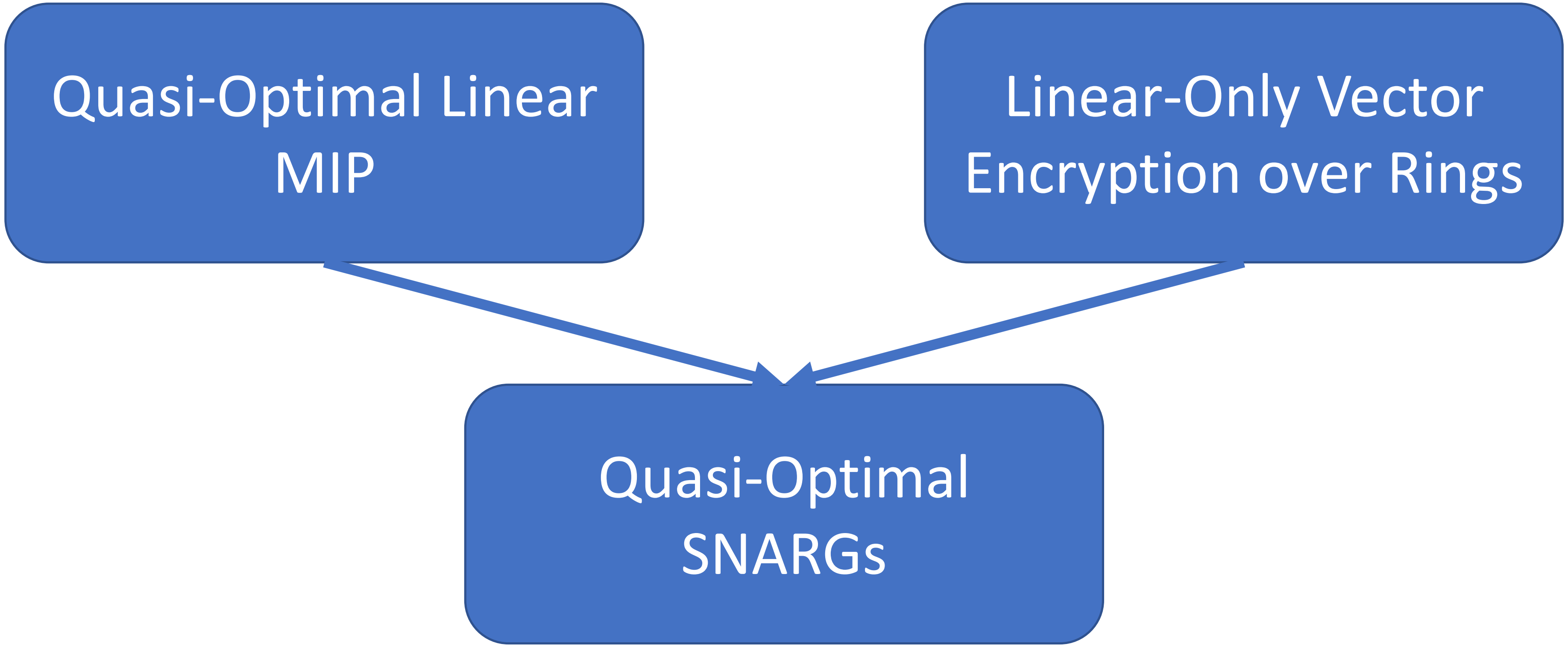


- Checking satisfiability of C corresponds to checking satisfiability of $f_1, \ldots, f_\ell$ (each of which can be checked by a circuit of size $|C|/\ell$)
- For a false statement, no single witness can simultaneously satisfy more than constant fraction of f_i

Robust decomposition can be instantiated by combining "MPC-in-the-head" paradigm [IKOS07] with information-theoretic MPC protocol with polylogarithmic overhead [DIK10]

[See paper for details]

Quasi-Optimal SNARGs



This work: first construction of a quasi-optimal SNARG from concrete cryptographic assumptions

Even Shorter SNARGs

Theorem. For general NP languages, achieving soundness error $2^{-\rho}$ against polynomially-bounded provers requires proofs of size $\Omega(\rho)$.

Can we build a (designated-verifier) 1-bit SNARG with soundness $1/2 + \text{negl}(\lambda)$?

- Natural primitive for building optimally succinct SNARGs

More generally, we can view this as an optimally laconic interactive argument (i.e., an interactive argument system where the prover communicates a single bit)

1-Bit SNARGs and Laconic Arguments

Can we build a 1-bit SNARG with soundness $1/2 + \text{negl}(\lambda)$?

Possible from indistinguishability obfuscation (iO)

Proof is a 1-bit MAC on the statement

$\text{Prove}_{C,k}(x, w)$:

- if $C(x, w) = 1$, outputs $\text{PRF}(k, x)$
- else, output $\perp$

CRS is obfuscation of this program

Secret verification state is PRF key k

1-Bit SNARGs and Laconic Arguments

Can we build a 1-bit SNARG with soundness $1/2 + \mathrm{negl}(\lambda)$?

In the interactive setting, can build optimally-laconic arguments for NP from witness encryption for NP

1-Bit SNARGs and Laconic Arguments

Can we build

Messages are encrypted with respect to a statement x, and semantic security holds with respect to a randomly chosen NO instance x

Optimally-laconic arguments for NP $\Longrightarrow$ variant of witness encryption for cryptographically-hard languages

Languages where YES instances are computationally indistinguishable from NO instances

Weaker than the usual notion of witness encryption, but suffices to build PKE with fast key-generation

1-Bit SNARGs and Laconic Arguments

Can we build a 1-bit SNARG with soundness $1/2 + \mathrm{negl}(\lambda)$?

Optimally-laconic arguments for NP $\Longrightarrow$ variant of witness encryption for cryptographically-hard languages

Since $\mathcal{L}$ is cryptographically-hard, there is exactly 1 accepting proof

Statement x

Soundness says that b should be hard to guess – can be used to blind a message

$\mathrm{Setup}(1^\lambda, x)$

$b \in \{0,1\}$

1-Bit SNARGs and Laconic Arguments

Can we build a <u>1-bit SNARG</u> with soundness $1/2 + \mathrm{negl}(\lambda)$?

Optimally-laconic arguments for NP $\Longrightarrow$ variant of witness encryption for cryptographically-hard languages

- Can be viewed as a soundness-to-secrecy transformation (dual of secrecy-to-soundness [AIK10])
- Optimally-laconic arguments is a powerful primitive
- Conceptually similar to recent work [BDRV17] showing connections between laconic zero-knowledge arguments and PKE

In the case of 1-bit SNARGs, <u>soundness</u> alone suffices for our variant of witness encryption

1-Bit SNARGs and Laconic Arguments

Can we build a 1-bit SNARG with soundness $1/2 + \mathrm{negl}(\lambda)$?

Optimally-laconic arguments for NP $\Longrightarrow$ variant of witness encryption for cryptographically-hard languages

- Can be viewed as a soundness-to-secrecy transformation (dual of secrecy-to-soundness [AIK10])
- Optimally-laconic arguments is a powerful primitive
- Conceptually similar to recent work [BDRV17] showing connections between laconic zero-knowledge arguments and PKE
- **Open problem:** Can we construct optimally-laconic arguments from standard assumptions?

Conclusions

A SNARG is quasi-optimal if it satisfies the following properties:

- Quasi-optimal succinctness: $|\pi| = \tilde{O}(\lambda)$
- Quasi-optimal prover complexity: $|P| = \tilde{O}(|C|) + \mathrm{poly}(\lambda, \log|C|)$

New framework for building quasi-optimal SNARG by combining quasi-optimal linear MIP with linear-only vector encryption

- Construction of a quasi-optimal linear MIP possible by combining robust decomposition and consistency check

Introduced new notion of a 1-bit SNARG (and optimally laconic argument) – has connections to witness encryption

Open Problems

Quasi-optimal SNARGs with additional properties:

- Publicly-verifiable / multi-theorem (in designated verifier setting)
- Zero-knowledge

New constructions of 1-bit SNARGs and optimally laconic arguments

Thank you!