Lattice-Based Succinct Arguments for NP with Polylogarithmic-Time Verification

Jonathan Bootle

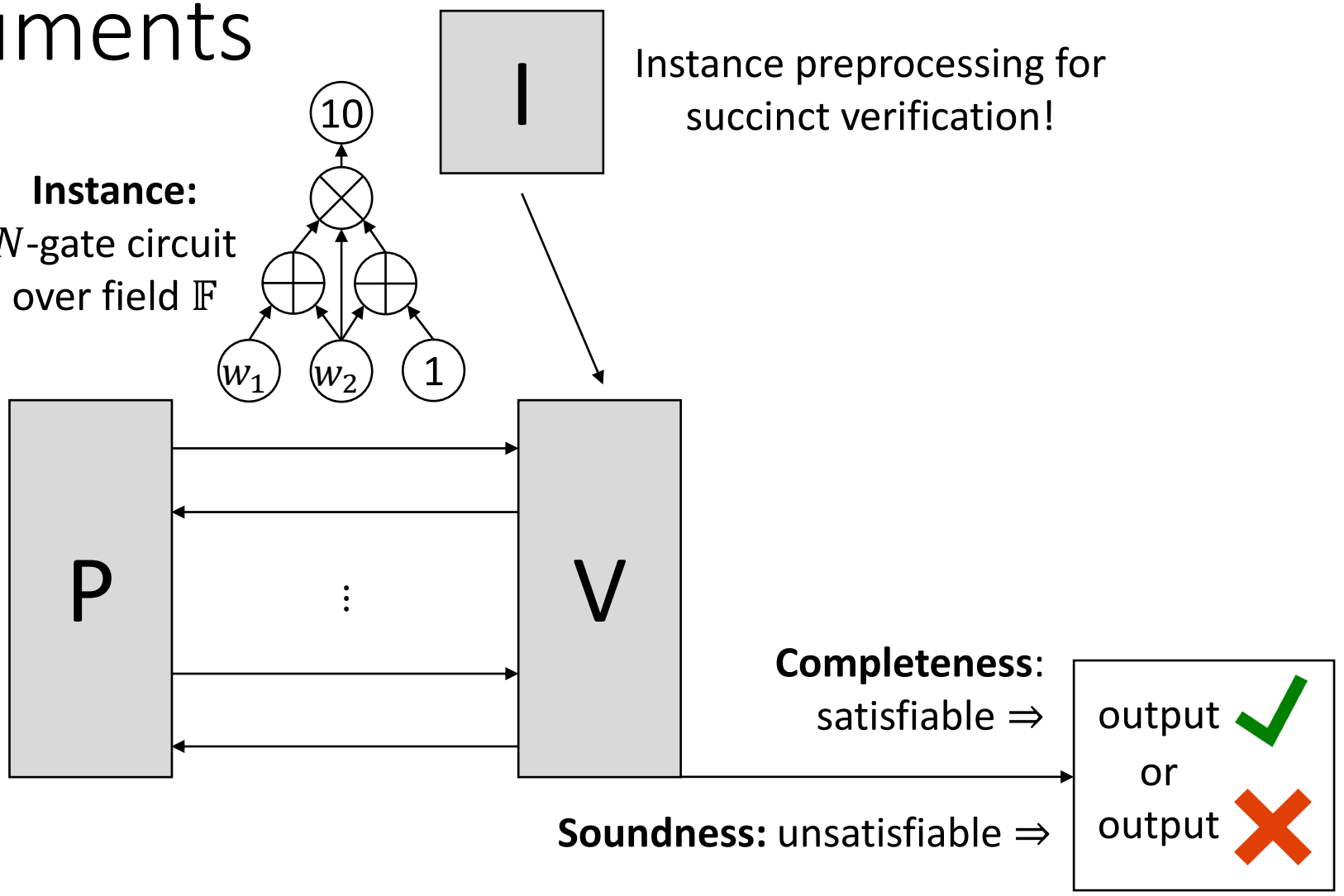
Based on joint work with

Alessandro Chiesa (EPFL) and Katerina Sotiraki (UC Berkeley)

Succinct arguments

Witness:

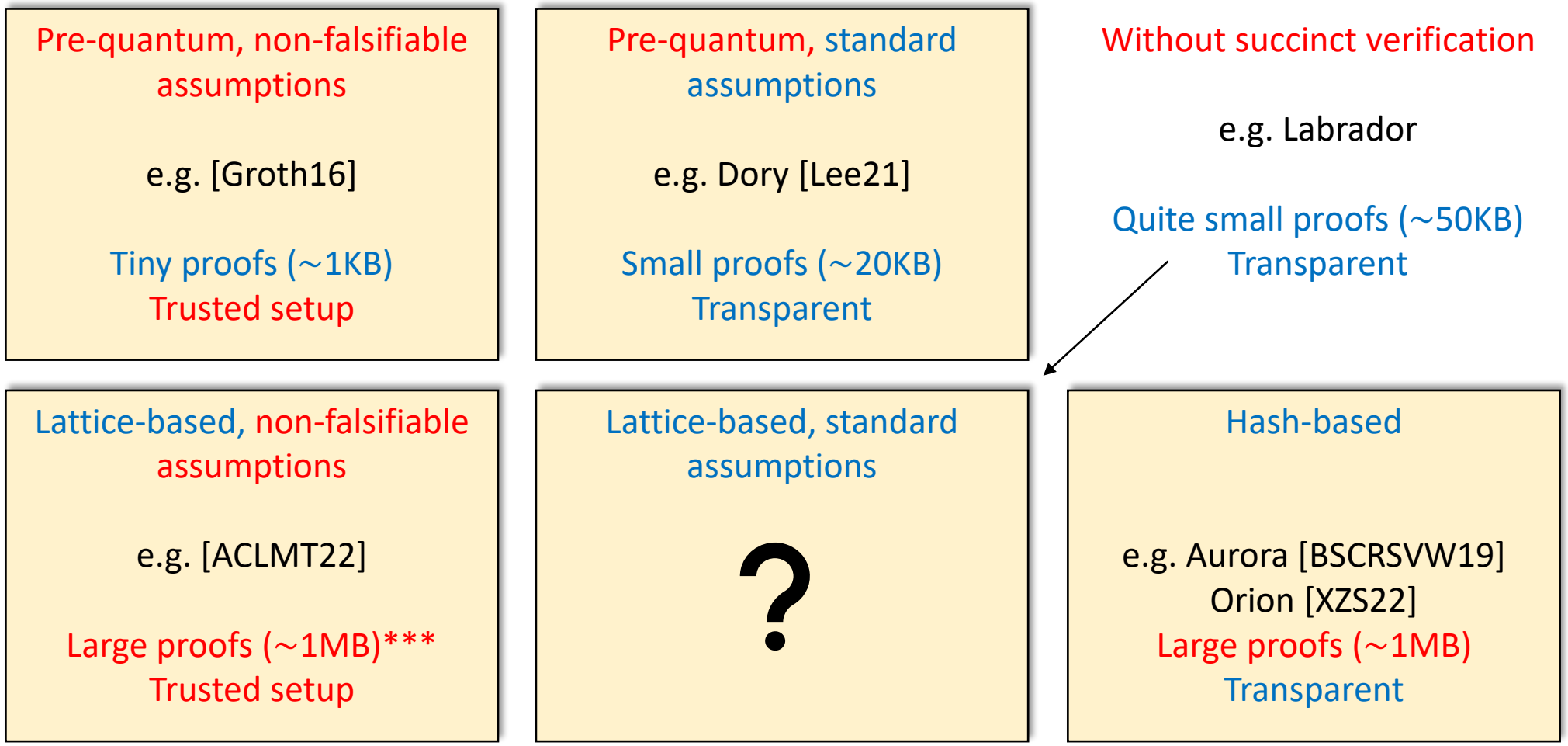
$w_1 = 4$
$w_2 = 1$



Succinctness: proof size ≪ instance size
verifier work ≪ checking $x \in L$ naively

No zero-knowledge today!

Existing succinct arguments



Question: can we construct transparent, succinct arguments directly from **standard** lattice assumptions?

Results

Main result

R1CS problem over a ring $\boldsymbol{R}$: given matrices $A, B, C \in R^{N \times N}$, does there exist $z \in R^n$ satisfying $Az \circ Bz = Cz$?

Bilinear module: a triple of modules (M_L, M_R, M_T) over the same ring with a bilinear map $e : M_L \times M_R \to M_T$.

Has enough structure for Pedersen and Schnorr

Main result

R1CS problem over a ring R: given matrices $A, B, C \in R^{N \times N}$, does there exist $z \in R^n$ satisfying $Az \circ Bz = Cz$?

k-level bilinear module: triples of modules $(M_L, M_R, M_T)_{i=1}^{k}$ over the same ring with a bilinear maps $e_i : M_{L,i} \times M_{R,i} \to M_{T,i}$.

Theorem 1: Let $\left(M_{L,i}, M_{R,i}, M_{T,i}, e_i\right)_{i=1}^{\log N}$ be a "secure", $\log N$-level bilinear module where $M_{L,1}$ is a ring. Let $I \subseteq M_{L,1}$ be a suitable ideal. There is a succinct argument of knowledge for R1CS with

R1CS Ring	Indexer time	Prover time	Verifier time	Proof size
M_L/I	$O(N)$ ops in $M_{T,\log N}$	$O(N)$ ops in $M_{T,\log N}$	$O\left(\log^2 N\right)$ ops in $M_{T,\log N}$	$O\left(\log^2 N\right)$ elems of $M_{T,\log N}$

Corollaries

Corollary 1: Let d be a power of 2, $p \ll q$ primes, $R_p := \mathbb{Z}_p[X]/\langle X^d + 1\rangle$ and similarly for R_q. Assuming SIS is hard over R_q, there is a succinct argument of knowledge for R1CS with

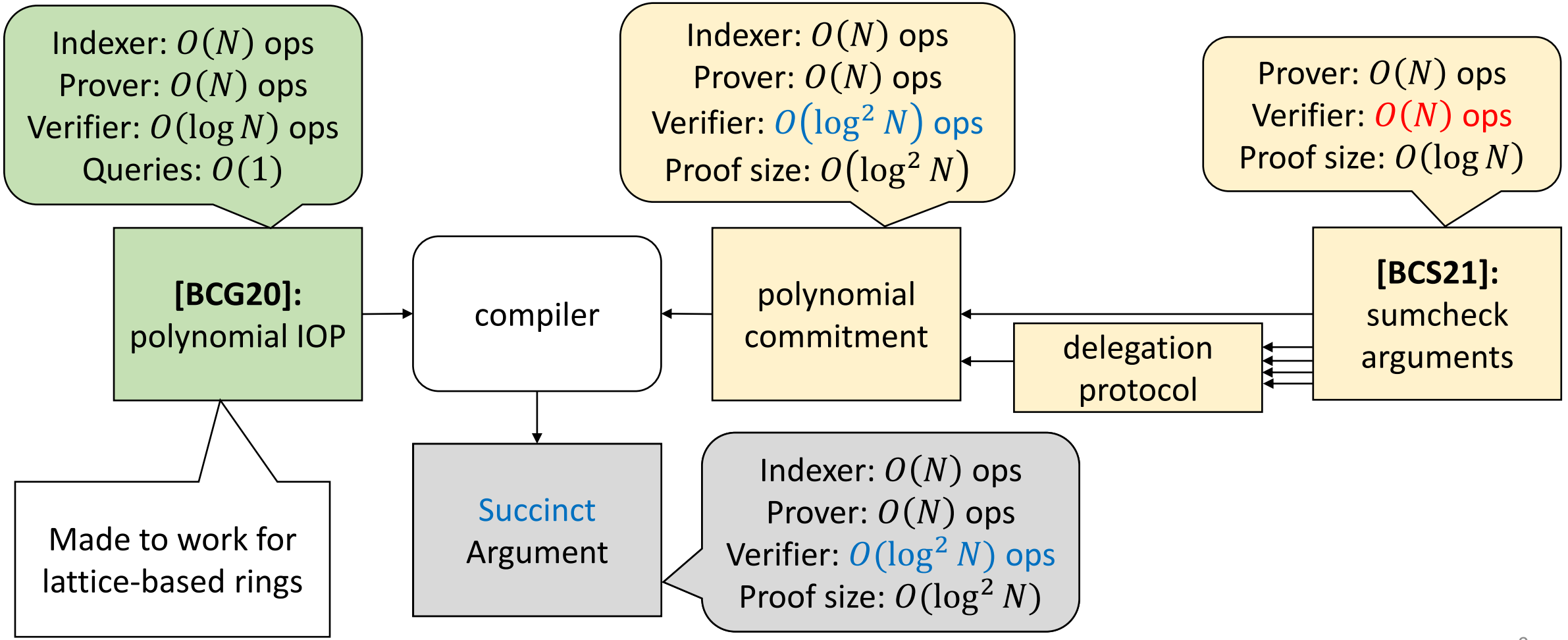
R1CS Ring	Indexer time	Prover time	Verifier time	Proof size
R_p	$O(N)$ ops in R_q	$O(N)$ ops in R_q	$O(\log^2 N)$ ops in R_q	$O(\log^2 N)$ elems of R_q

Corollary 2: Let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a bilinear group of prime order p. Assuming SXDH, there is a succinct argument of knowledge for R1CS with

Related to [Lee21], [Thaler]

R1CS Ring	Indexer time	Prover time	Verifier time	Proof size
$\mathbb{F}_p$	$O(N)$ ops in $\mathbb{G}_T$	$O(N)$ ops in $\mathbb{G}_T$	$O(\log^2 N)$ ops in $\mathbb{G}_T$	$O(\log^2 N)$ elems of $\mathbb{G}_T$

Our approach



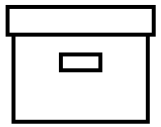
Polynomial commitments with efficient verification

Polynomial commitment schemes

Keygen → 

length N

Commit( , )

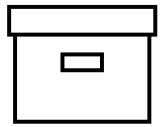
$=$ 

Binding: hard to find

Commit( , )

$=$ Commit( , )

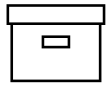
Eval:

Instance:  

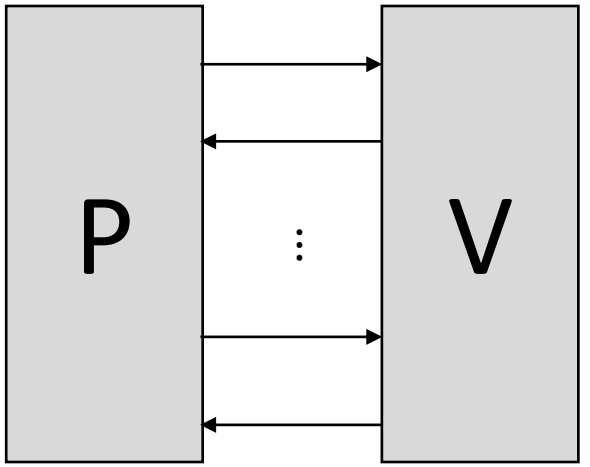
Inputs $z_1, \dots, z_\ell$, output v

Witness: p 

Commit( , )

$=$ 

p  $(z_1, \dots, z_\ell) = v$



$O(\log(N))$ rounds

$O(\log(N))$ ops

$O(N)$ ops to evaluate p  at $r_1, \dots, r_\ell$

$O(\log(N))$ communication

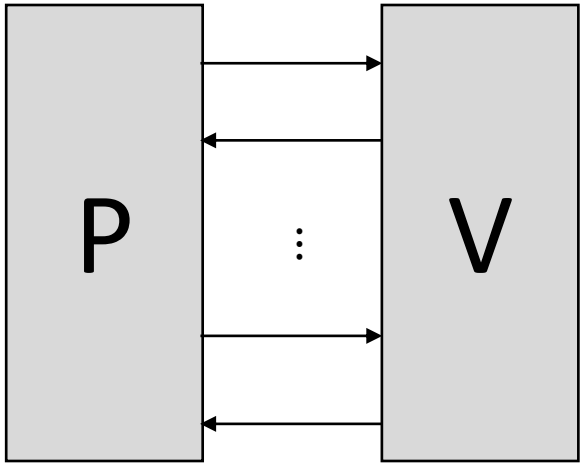
[BCS21]: sumcheck-friendly polynomial commitment schemes

Polynomial commitment schemes

Goal: delegate computation to the prover

Instance:

Inputs $z_1, \ldots, z_\ell$, output v



$O(\log(N))$ rounds

$O(\log(N))$ ops

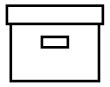
$O(N)$ ops to evaluate p at $r_1, \ldots, r_\ell$

$O(\log(N))$ communication

21]: sumcheck-friendly polynomial commitment schemes

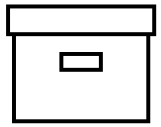
Succinct verification through delegation

Witness: p 

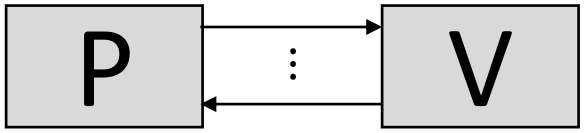
$\text{Commit}(p$  $,$  $) =$ 

p  $(z_1, \dots, z_\ell) = v$

Instance: 
length N



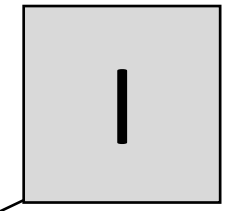
Inputs $z_1, \dots, z_\ell$, output v



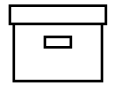
$O(\log(N))$ ops

$O(N)$ ops to evaluate p  at $r_1, \dots, r_\ell$

Succinct verification through delegation

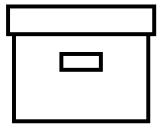


Witness: p

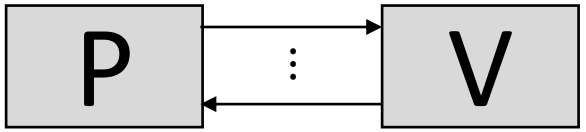
$\text{Commit}(p, \cdot) =$ 

$p(z_1, \ldots, z_\ell) = v$

Instance: 



Inputs $z_1, \ldots, z_\ell$, output v



$O(\log(N))$ ops

Delegate to the prover

$= \text{Commit}(p, \cdot)$

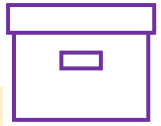
Instance compression for succinct verification!

$p(r_1, \ldots, r_\ell) = v'$

v'

New Witness: p

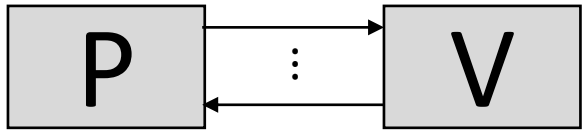
New Instance: 



$\text{Commit}(p, \cdot) =$ 

Inputs $r_1, \ldots, r_\ell$, output v'

$p(r_1, \ldots, r_\ell) = v'$

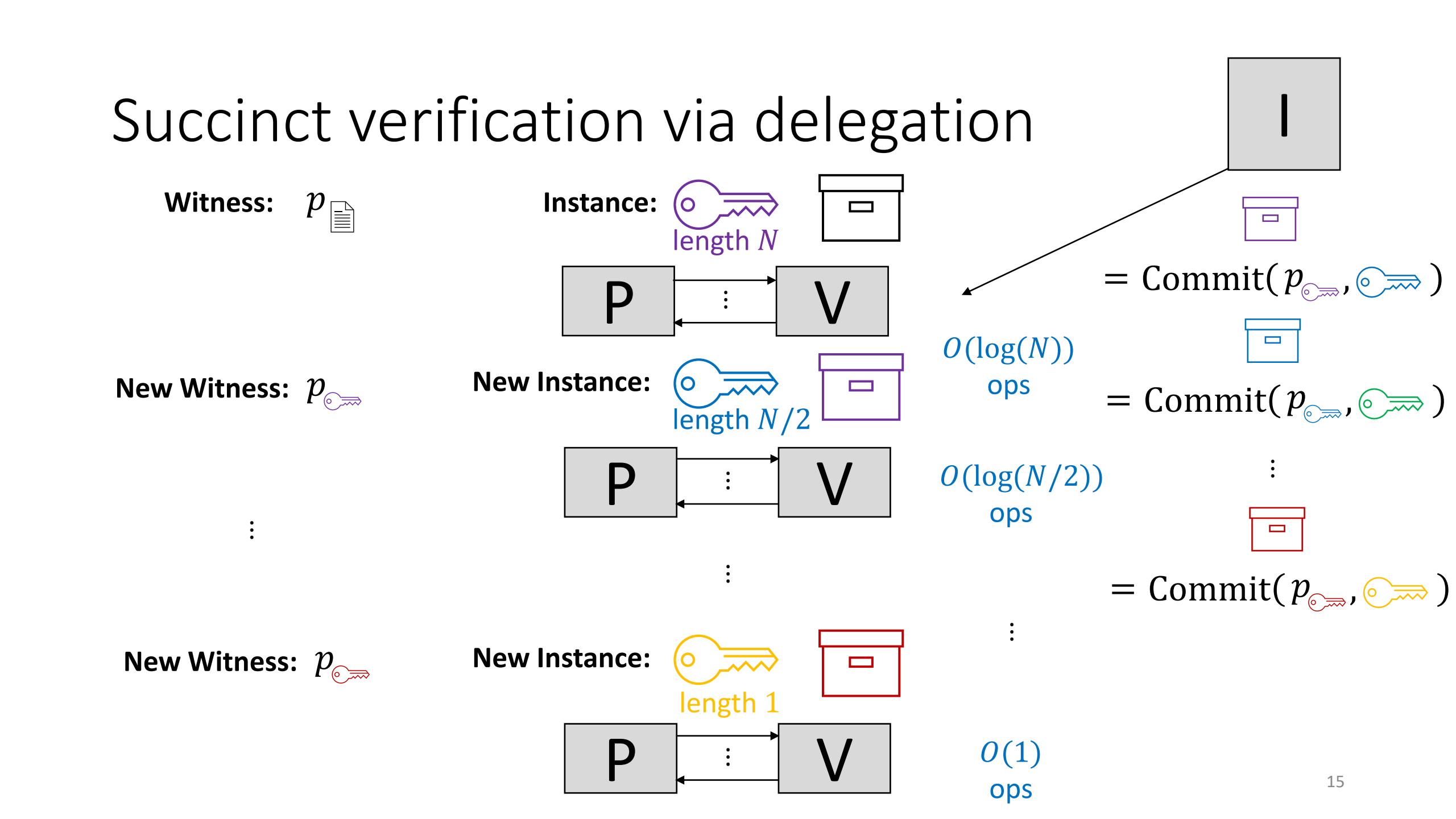


$O(\log(N/2))$ ops

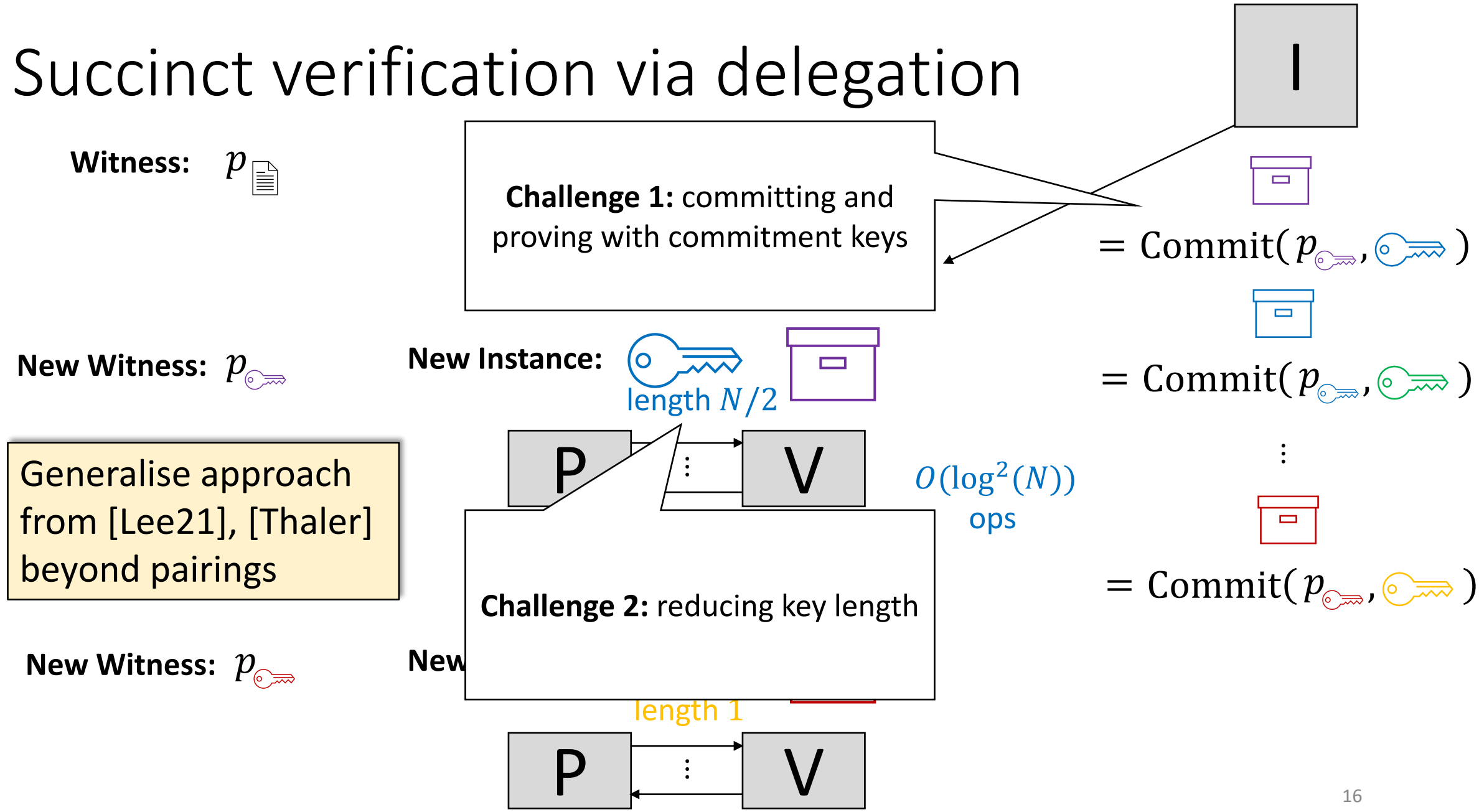
$O(N/2)$ ops to evaluate p

Recurse!

Succinct verification via delegation



Succinct verification via delegation



A closer look at [BCS21] polynomial commitments

Commitments over bilinear modules

- R-module : a vector space over a ring R

Bilinear module:

- R-modules M_L, M_R and M_T
- $e : M_L \times M_R \to M_T$ is R-bilinear

'Multiply' message and key elements using e

$$C = p_0 G_0 + \cdots + p_{N-1} G_{N-1} = \langle \underline{p}, \underline{G} \rangle$$

Add the pieces together

Hard to find small $\underline{p}$ such that $\langle \underline{p}, \underline{G} \rangle = 0$

Messages	Keys	Commitments	Assumption
small M_L	M_R	M_T	Bilinear Relation Assumption
$\mathbb{G}_1$	$\mathbb{G}_2$	$\mathbb{G}_T$	DPAIR (⇐ SXDH)
small R	R_q	R_q	Ring SIS

e is multiplication mod q,
$R = {}^{\mathbb{Z}[X]}/_{X^d+1}$

Polynomial commitments over bilinear modules

$\text{Keygen}(N)$:

$G_0, \dots, G_{N-1} \leftarrow M_R$

$\text{Commit}\left(p_{\underline{m}}(\underline{X}), \underline{G}\right) = \langle \underline{m}, \underline{G} \rangle \in M_T$

$\underline{m} = (m_0, \dots, m_{N-1}) \in M_L^N$

$p_{\underline{m}}(\underline{X}) = \sum_i X_1^{i_1} \cdots X_\ell^{i_\ell} \, m_i \in M_L[\underline{X}]$

multilinear

Eval:

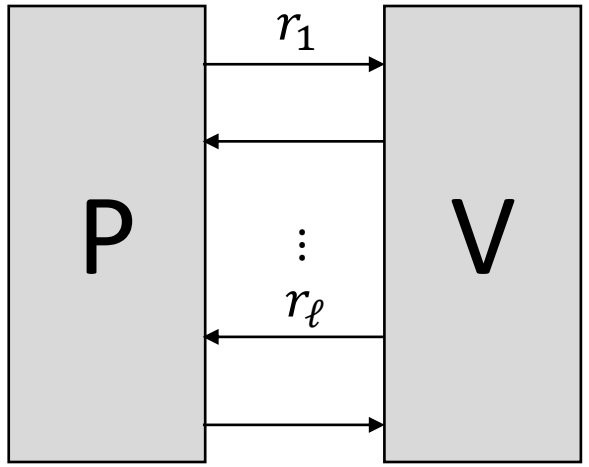
Instance: $G_0, \dots, G_{N-1} \in M_R \quad C \in M_T$

Inputs $z_1, \dots, z_\ell \in R$, output $v \in M_L$

Witness: $p_{\underline{m}}$

$\langle \underline{m}, \underline{G} \rangle = C$

$p_{\underline{m}}(z_1, \dots, z_\ell) = v$



$O(\log(N))$ rounds

$O(\log(N))$ ops

$O(N)$ ops to compute $p_{\underline{G}}(\underline{r})$

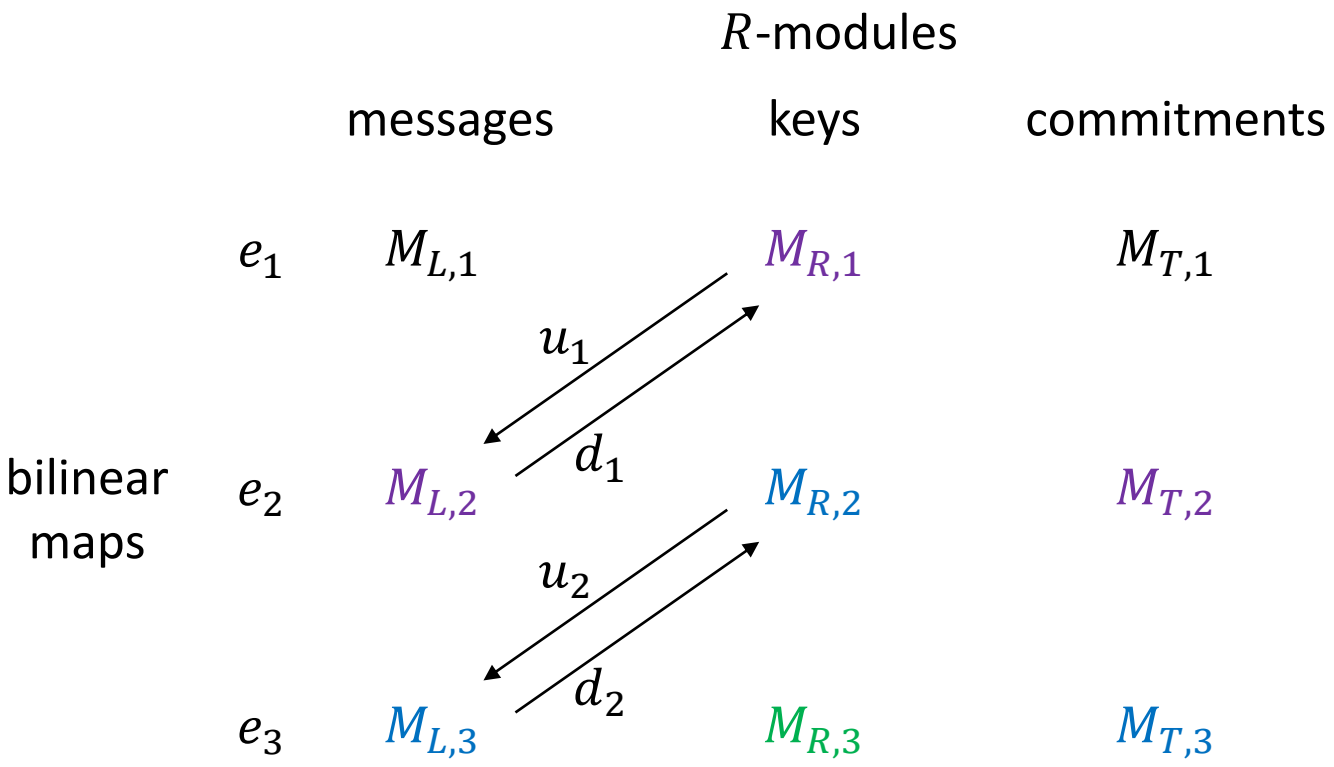
$O(\log(N))$ communication

[BCS21]: sumcheck-friendly polynomial commitment schemes

Think of lattice bulletproofs (Russell's talk)

Challenge 1: committing and proving with commitment keys

Levelled bilinear modules



Requirements:

$$`M_{R,1} \subseteq M_{L,2}'$$

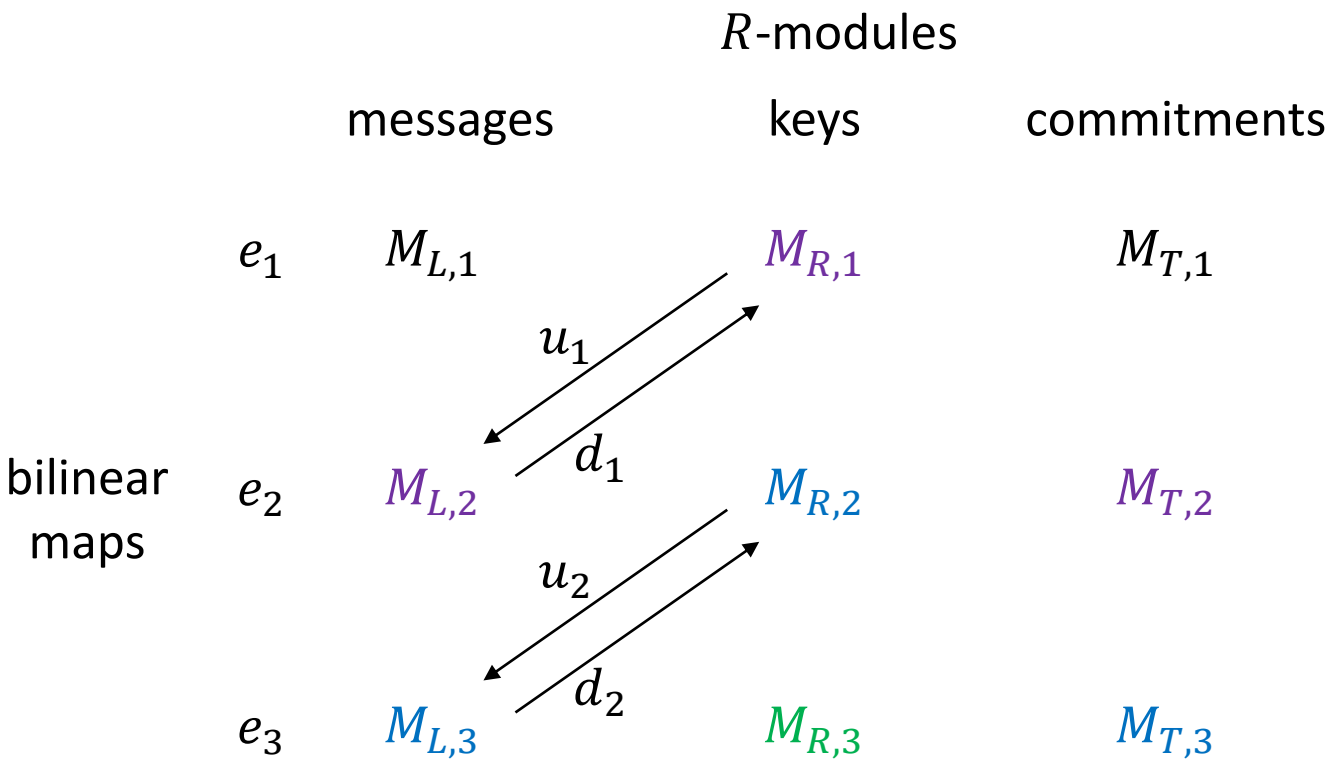
$$u_i(M_{R,i}) \subseteq M_{L,i+1}$$

$$p_{u_i(\underline{G})}(r_1, \dots, r_\ell) = u_i(v')$$

$$\Downarrow$$

$$p_{\underline{G}}(r_1, \dots, r_\ell) = v'$$

Levelled bilinear modules



Requirements:

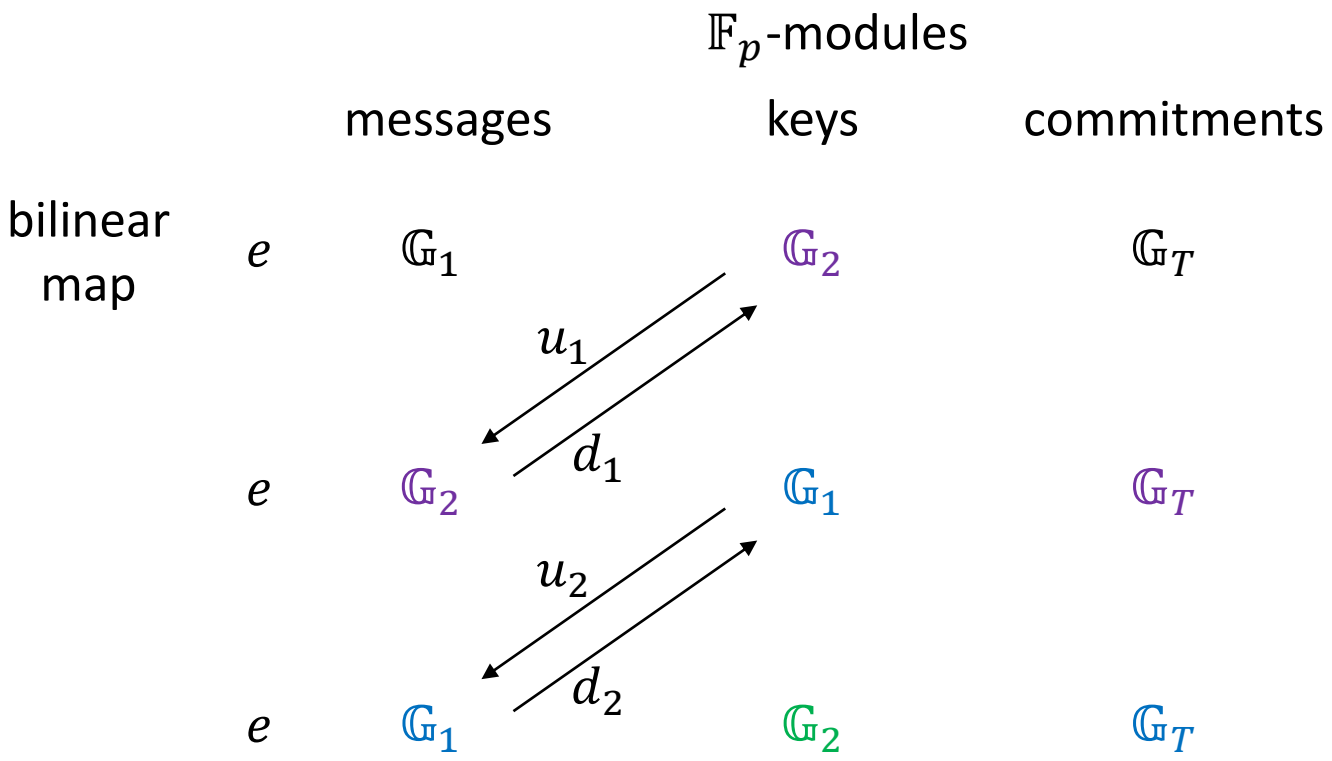
R-homomorphisms
$d_i : M_{L,i+1} \to M_{R,i}$

$u_i = d_i^{-1}$

Example: $d_i : \mathbb{Z} \to \mathbb{Z}_p$, u_i 'forgets' the mod p

Want bilinear relation assumption at each level

Pairing instantiation



$\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ groups of prime order p

$\text{keys}_{2i} \rightarrow \text{messages}_{2i+1}$

$\mathbb{G}_1 \qquad\qquad \mathbb{G}_2$

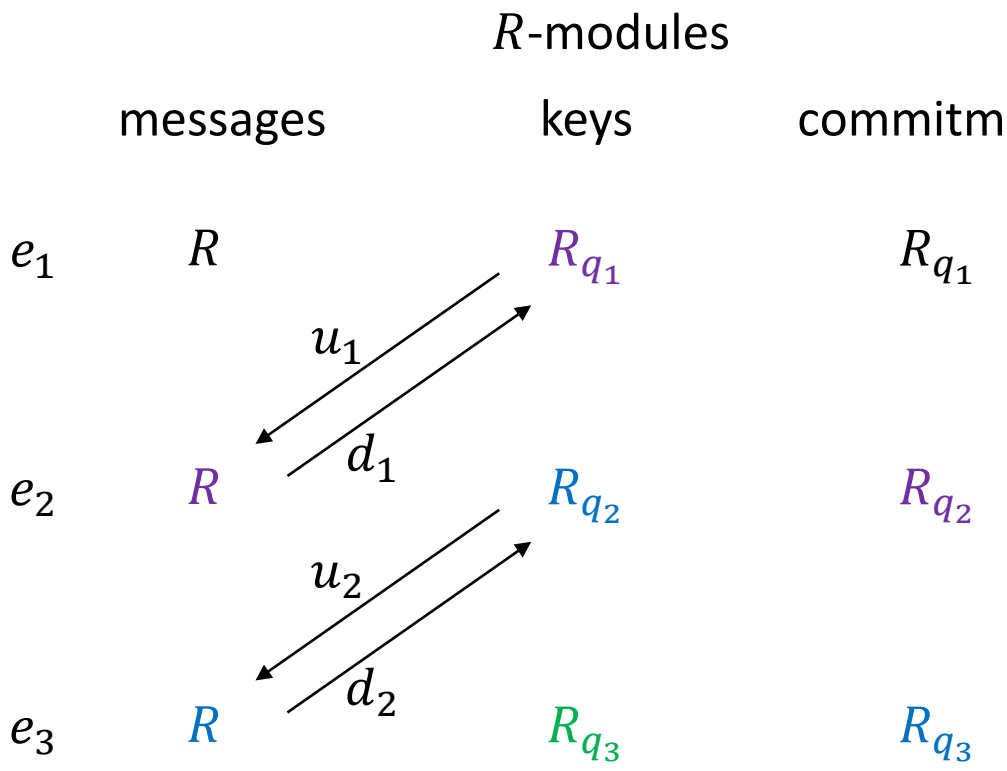
$\text{mesages}_{2i+2} \leftarrow \text{keys}_{2i+1}$

$u_i = d_i = \text{Id}$

SXDH ⇒ **both** bilinear relation assumptions

Lattice instantiation 1

e_1 is multiplication mod q_1, $R = {\mathbb{Z}[X]}/{X^d+1}$



d_i = reduce mod q_i

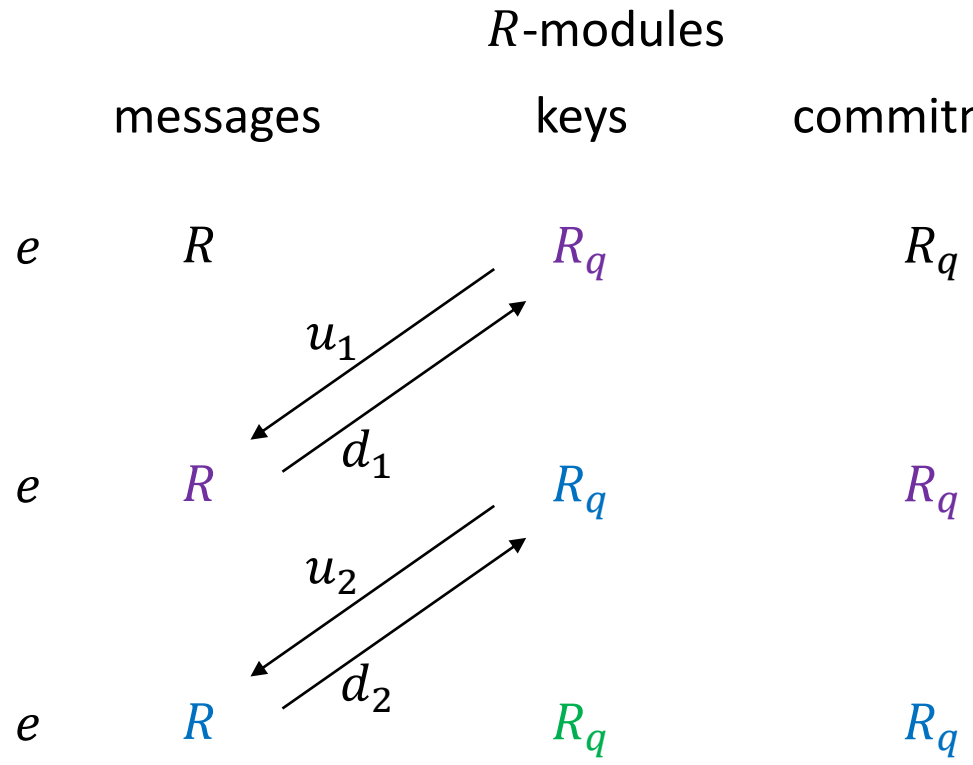
u_i = forget the mod q_i

$q_1 \ll q_2 \ll q_3 \ldots$

RSIS for $R_{q_i} \Rightarrow$ bilinear relation assumptions at level i

Lattice instantiation 2

e is multiplication mod q, $R = \mathbb{Z}[X]/_{X^d+1}$



u_i = forget the mod q and bit-decompose

d_i = form an integer and reduce mod q

R R_q

$\text{mesages}_{\text{i+1}} \leftarrow \text{keys}_i$

RSIS for $R_q \Rightarrow$ bilinear relation assumptions at every level

Extra technicalities

$$p_{u_i(\underline{G})}(r_1, \dots, r_\ell) = u_i(v')$$

$$\Downarrow \quad \text{Soundness}$$

$$p_{\underline{G}}(r_1, \dots, r_\ell) = v'$$

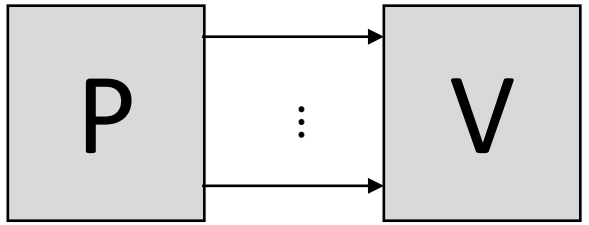
$$p_{u_i(\underline{G})}(r_1, \dots, r_\ell) = u_i(v')$$

$$\Uparrow \quad \text{Completeness}$$

$$p_{\underline{G}}(r_1, \dots, r_\ell) = v' \quad \text{Not true!}$$

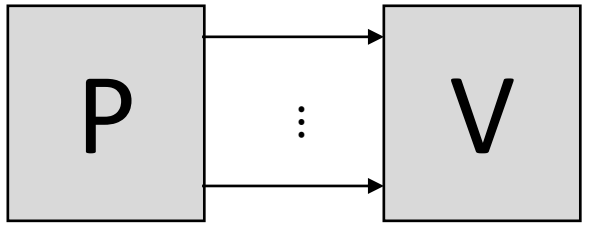
$$(1 \bmod 2) + (1 \bmod 2) \neq (2 \bmod 2)$$

Proof that $p_{\underline{m}}(z_1, \dots, z_\ell) = v$



$p_{\underline{G}}(r_1, \dots, r_\ell) = v'$ $\xrightarrow{v'}$

Proof that $p_{u_1(\underline{G})}(r_1, \dots, r_\ell) = u_1(v')$



Extra technicalities

$$p_{u_i(\underline{G})}(r_1, \dots, r_\ell) = u_i(v')$$

$$\Downarrow \quad \text{Soundness}$$

$$p_{\underline{G}}(r_1, \dots, r_\ell) = v'$$

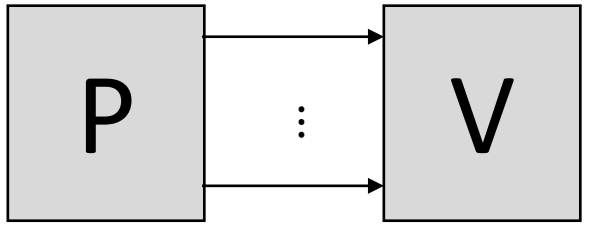
$$p_{u_i(\underline{G})}(r_1, \dots, r_\ell) = u_i(v') \; mod \; \ker d_i$$

$$\Uparrow \quad \text{Completeness}$$

$$p_{\underline{G}}(r_1, \dots, r_\ell) = v' \quad \text{Not true!}$$

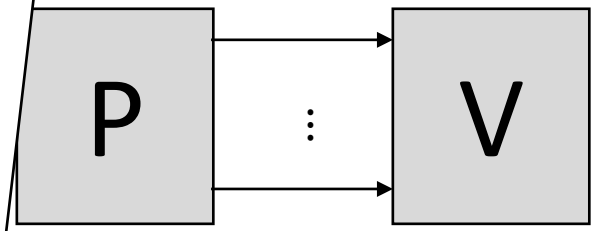
$$(1 \; mod \; 2) + (1 \; mod \; 2) \neq (2 \; mod \; 2)$$

Proof that $p_{\underline{m}}(z_1, \dots, z_\ell) = v$



$p_{\underline{G}}(r_1, \dots, r_\ell) = v'$ $\xrightarrow{v'}$

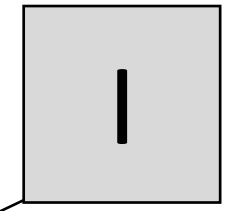
Proof that $p_{u_1(\underline{G})}(r_1, \dots, r_\ell) \equiv u_1(v')$



Lattice instantiation 2: key length increases
Additional compression step required

Challenge 2: reducing key length

Key splitting

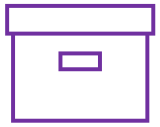


Witness: $p_{\text{🔑}}$

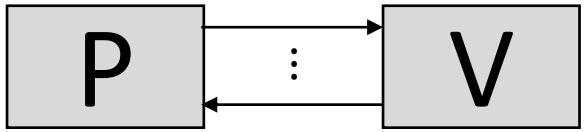
$\text{Commit}(p_{\text{🔑}}, \text{🔑}) =$ 

$p_{\text{🔑}}(r_1, \ldots, r_\ell) = v'$

Instance: 



Inputs $r_1, \ldots, r_\ell$, output v'



$= \text{Commit}(p_{\text{🔑}}, \text{🔑})$

$= \text{Commit}(p_{\text{🔑}}, \text{🔑})$

$\neq$

Key splitting

Witness: $p_{\underline{G}}(\underline{X})$

$$\text{Commit}\left(p_{\underline{G_L}}, \underline{G'}\right) = C_L$$

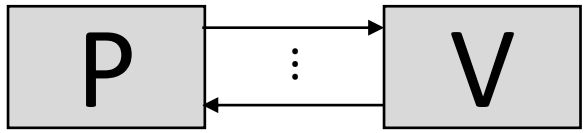
$$\text{Commit}\left(p_{\underline{G_R}}, \underline{G'}\right) = C_R$$

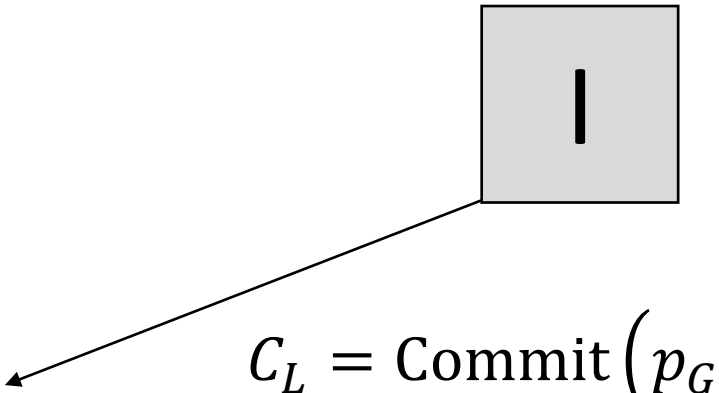
$$p_{\underline{G}}(r_1, \ldots, r_\ell) = v'$$

$$C_L, C_R \in M_{T,2}$$

Instance: $G'_0, \ldots, G'_{N/2-1} \in M_{R,2}$

Inputs $r_1, \ldots, r_\ell$, output v'





$$C_L = \text{Commit}\left(p_{\underline{G_L}}, \underline{G'}\right)$$

$$C_R = \text{Commit}\left(p_{\underline{G_R}}, \underline{G'}\right)$$

$$\underline{G} = \left(\underline{G_L}, \underline{G_R}\right) \in M_L^{N/2} \times M_L^{N/2}$$

$$p_{\underline{G}}(r_1, \ldots, r_\ell) = v'$$
$$= \sum_i r_1^{i_1} \cdots r_\ell^{i_\ell} \, G_i$$

$$p_{\underline{G}}(r_1, \ldots, r_\ell) = p_{\underline{G_L}}(r_2, \ldots, r_\ell) + r_1 \cdot p_{\underline{G_R}}(r_2, \ldots, r_\ell)$$
$$= p_{\underline{G_L} + r_1 \underline{G_R}}(r_2, \ldots, r_\ell)$$

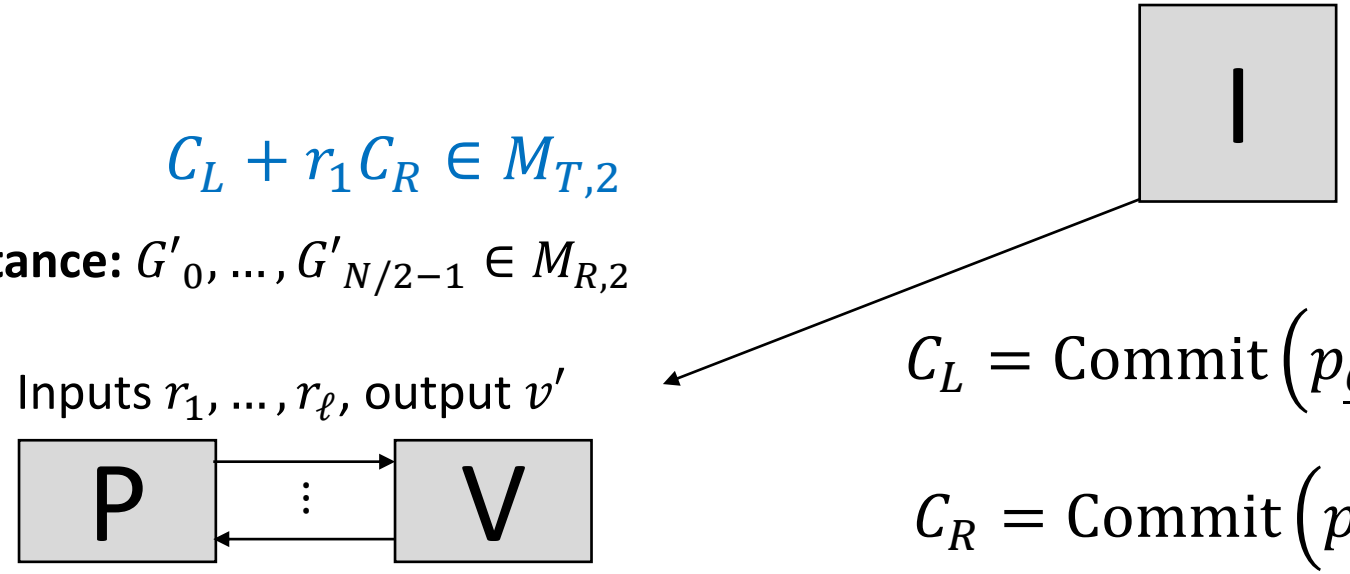
Ignore u map on this slide

Key splitting

$C_L + r_1 C_R \in M_{T,2}$

Witness: $p_{\underline{G_L}+r_1\underline{G_R}}(\underline{X})$

Instance: $G'_0, \ldots, G'_{N/2-1} \in M_{R,2}$



Inputs $r_1, \ldots, r_\ell$, output v'

$$\text{Commit}\left(p_{\underline{G_L}}, \underline{G'}\right) = C_L$$

$$\text{Commit}\left(p_{\underline{G_R}}, \underline{G'}\right) = C_R$$

$$p_{\underline{G_L}+r_1\underline{G_R}}(r_1, \ldots, r_\ell) = v'$$

$$C_L = \text{Commit}\left(p_{\underline{G_L}}, \underline{G'}\right)$$

$$C_R = \text{Commit}\left(p_{\underline{G_R}}, \underline{G'}\right)$$

$$\underline{G} = \left(\underline{G_L}, \underline{G_R}\right) \in M_L^{N/2} \times M_L^{N/2}$$

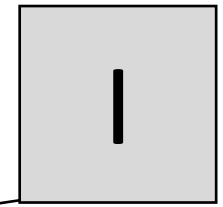
$$p_{\underline{G}}(r_1, \ldots, r_\ell) = v'$$
$$= \sum_i r_1^{i_1} \cdots r_\ell^{i_\ell} \, G_i$$

$$p_{\underline{G}}(r_1, \ldots, r_\ell) = p_{\underline{G_L}}(r_2, \ldots, r_\ell) + r_1 \cdot p_{\underline{G_R}}(r_2, \ldots, r_\ell)$$
$$= p_{\underline{G_L}+r_1\underline{G_R}}(r_2, \ldots, r_\ell)$$

Ignore u map on this slide

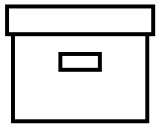
Putting everything together

Succinct verification via delegation

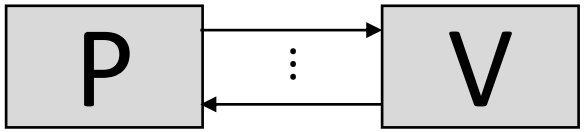


Witness: $p_{\underline{m}}$

Instance: i/o and



$= \mathrm{Commit}(p_{\underline{m}}, \text{key})$



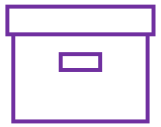
$O(\log(N))$

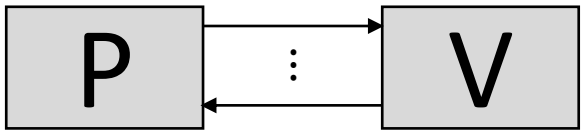
Split key, apply u_1, fix technicalities

$= \mathrm{Commit}(p_{u_1(\underline{G}_L + r_1 \underline{G}_R)}, \text{key})$

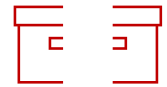
New Witness: $p_{u_1(\underline{G}_L + r_1 \underline{G}_R)}$

New Instance: i/o and





$O(\log(N/2))$

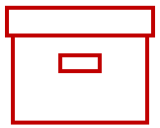


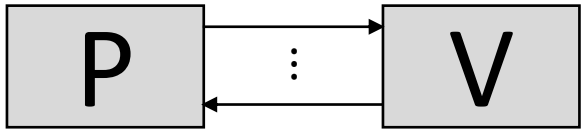
$= \mathrm{Commit}(p_{u_\ell(G_L^{(\ell-1)} + r^{(\ell)} G_R^{(\ell-1)})}, \text{key})$

New Witness:

$p_{u_\ell(G_L^{(\ell-1)} + r^{(\ell)} G_R^{(\ell-1)})}$

New Instance: i/o and





$O(1)$

$O(\log(N))$ preprocessed commitments

Conclusion and future work

Main result

R1CS problem over a ring R: given matrices $A, B, C \in R^{N \times N}$, does there exist $z \in R^n$ satisfying $Az \circ Bz = Cz$?

k-level bilinear module: triples of modules $(M_L, M_R, M_T)_{i=1}^{k}$ over the same ring with a bilinear maps $e_i : M_{L,i} \times M_{R,i} \to M_{T,i}$.

Theorem 1: Let $\left(M_{L,i}, M_{R,i}, M_{T,i}, e_i\right)_{i=1}^{\log N}$ be a "secure", $\log N$-level bilinear module where $M_{L,1}$ is a ring. Let $I \subseteq M_{L,1}$ be a suitable ideal. There is a succinct argument of knowledge for R1CS with

R1CS Ring	Indexer time	Prover time	Verifier time	Proof size
M_L/I	$O(N)$ ops in $M_{T,\log N}$	$O(N)$ ops in $M_{T,\log N}$	$O\left(\log^2 N\right)$ ops in $M_{T,\log N}$	$O\left(\log^2 N\right)$ elems of $M_{T,\log N}$

Corollaries

Corollary 1: Let d be a power of 2, $p \ll q$ primes, $R_p := \mathbb{Z}_p[X]/\langle X^d + 1\rangle$ and similarly for R_q. Assuming SIS is hard over R_q, there is a succinct argument of knowledge for R1CS with

R1CS Ring	Indexer time	Prover time	Verifier time	Proof size
R_p	$O(N)$ ops in R_q	$O(N)$ ops in R_q	$O(\log^2 N)$ ops in R_q	$O(\log^2 N)$ elems of R_q

Corollary 2: Let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a bilinear group of prime order p. Assuming SXDH, there is a succinct argument of knowledge for R1CS with

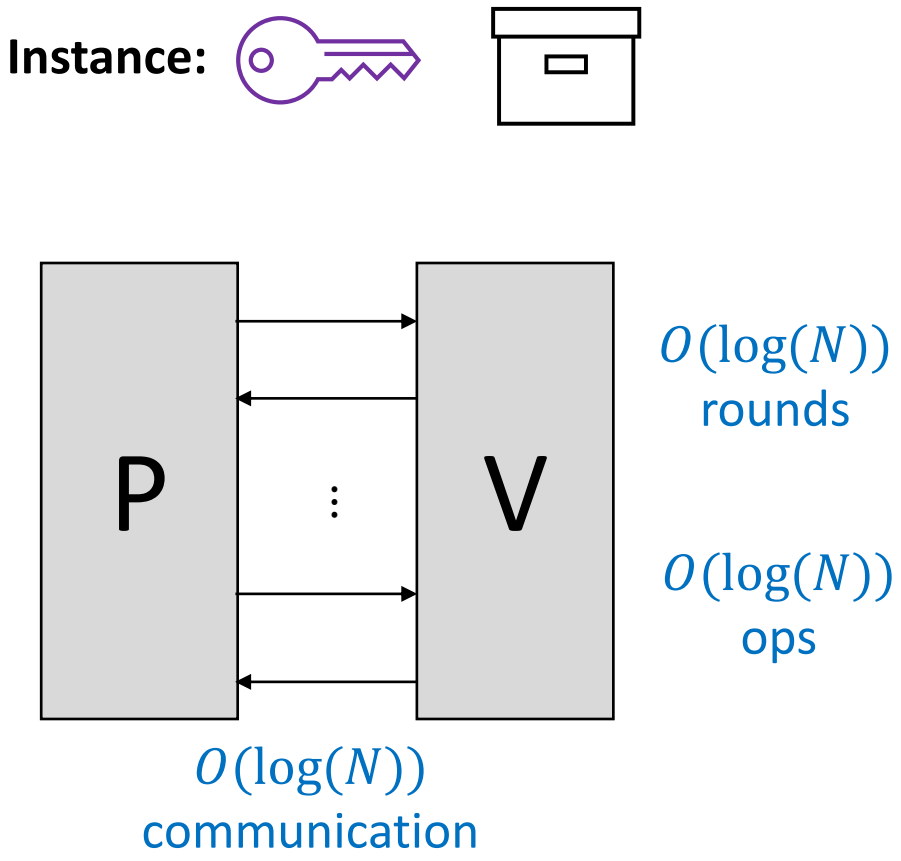
R1CS Ring	Indexer time	Prover time	Verifier time	Proof size
$\mathbb{F}_p$	$O(N)$ ops in $\mathbb{G}_T$	$O(N)$ ops in $\mathbb{G}_T$	$O(\log^2 N)$ ops in $\mathbb{G}_T$	$O(\log^2 N)$ elems of $\mathbb{G}_T$

Future work

- Replace [BCS20] with something more concretely efficient…

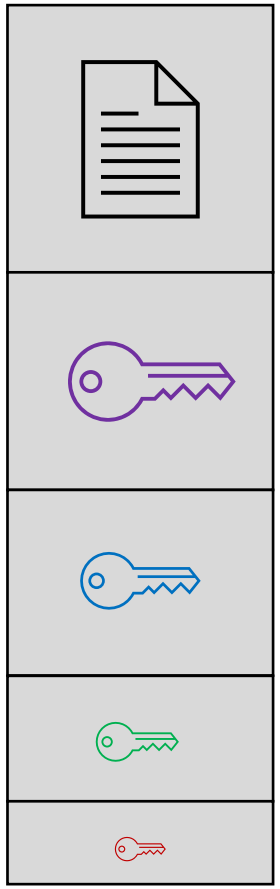
…Labrador?

- Low-memory prover algorithms



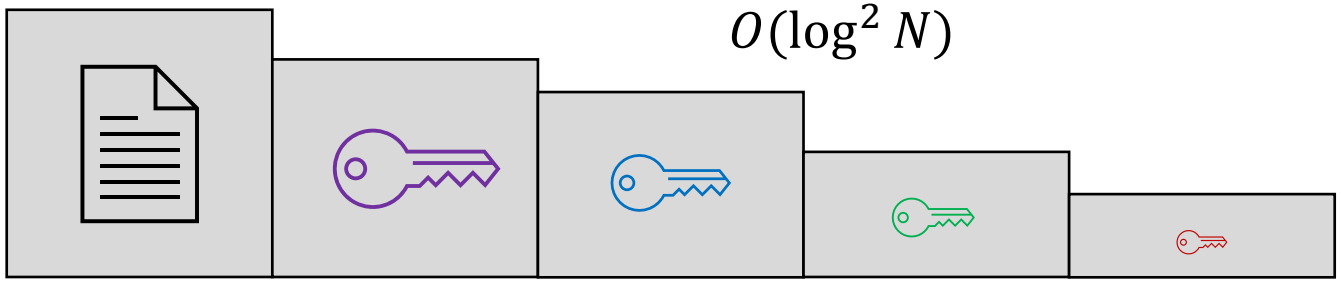
[BCS20]: sumcheck-friendly polynomial commitment schemes

Further optimization ideas



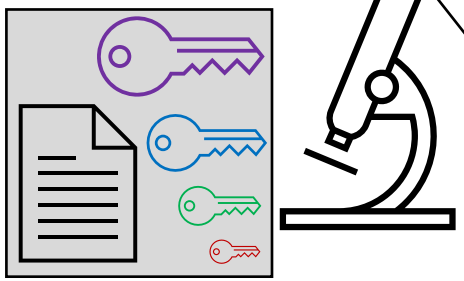
Pairings: [Thaler]
Lattices: this work

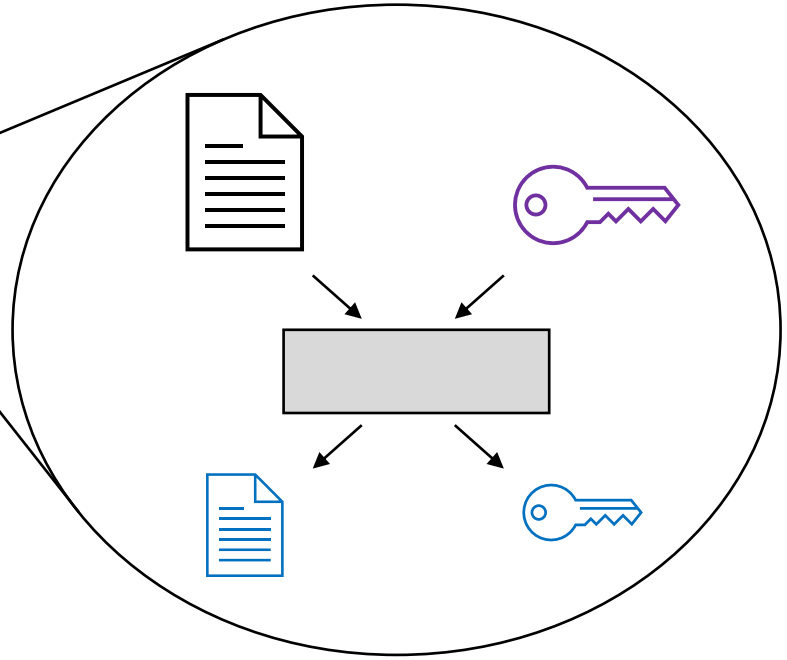
$O(\log^2 N)$



$O(\log^2 N)$

$O(\log N)$





Pairings: Dory [Lee21]
Lattices: ??

Further optimization ideas