Arithmetic sketching

Dan Boneh
Stanford

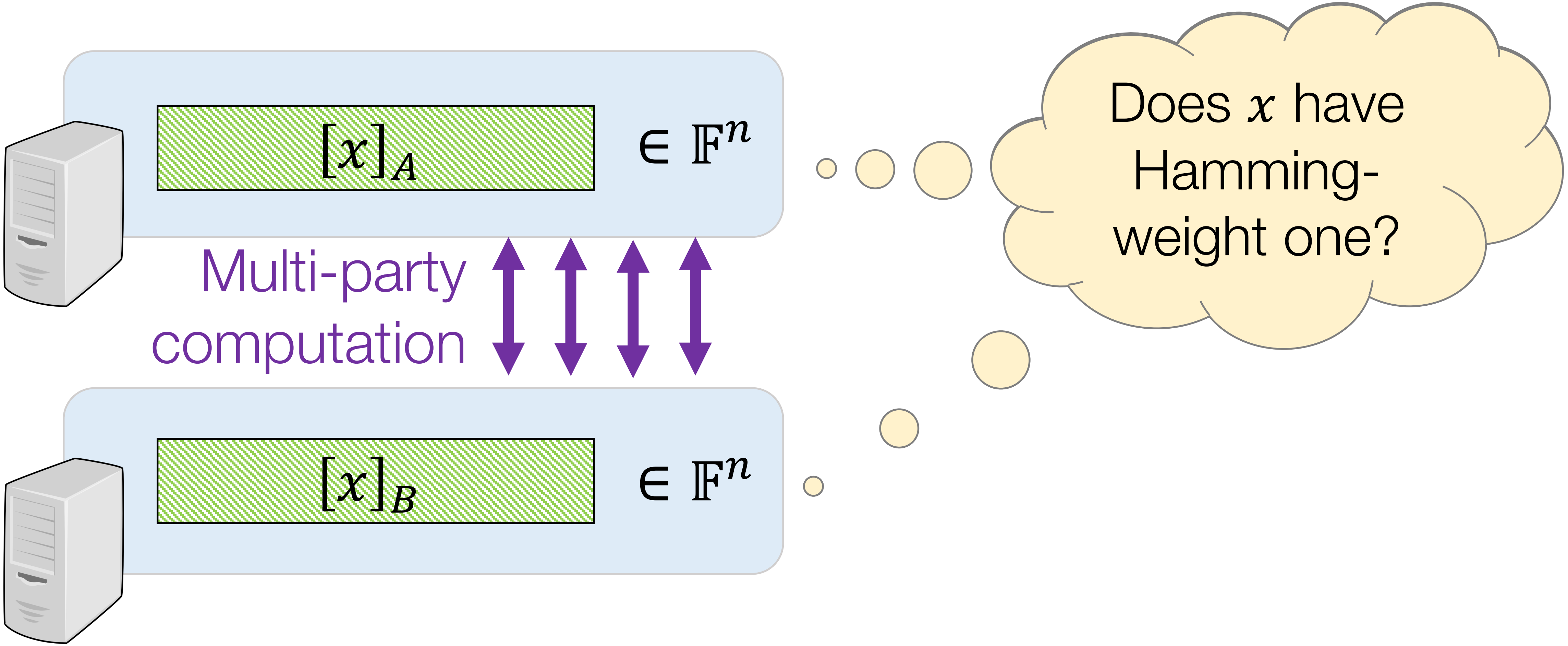
Elette Boyle
Reichman University
and NTT

Henry Corrigan-Gibbs
MIT

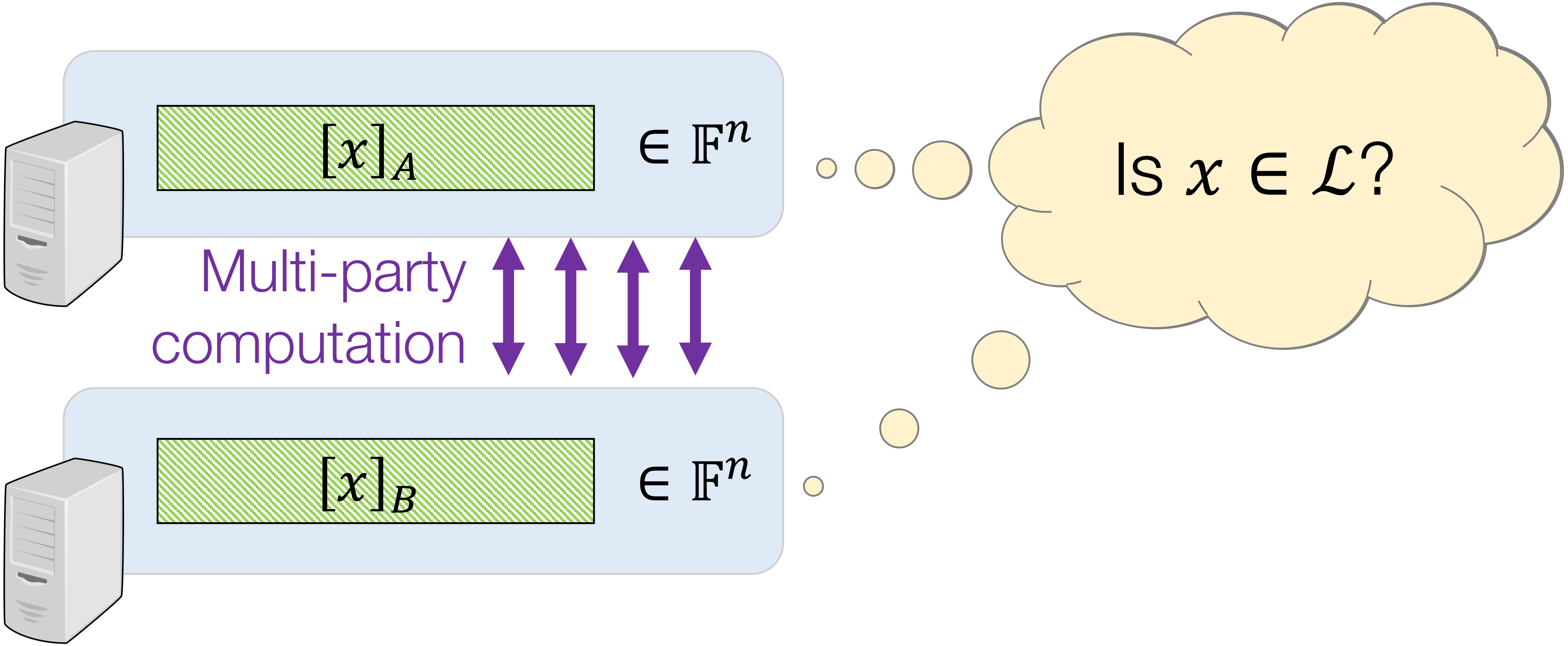
Niv Gilboa
Ben-Gurion University

Yuval Ishai
Technion

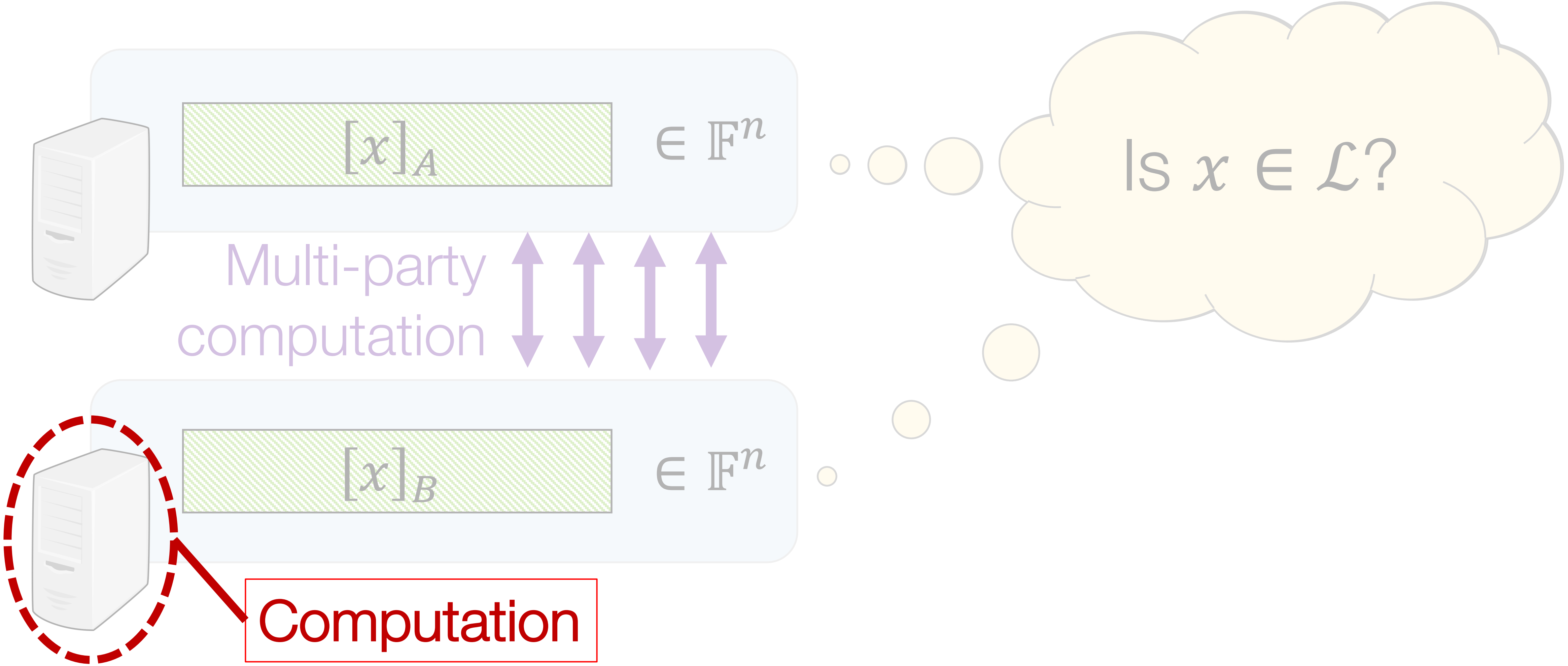
Common task: Test property of secret-shared vector



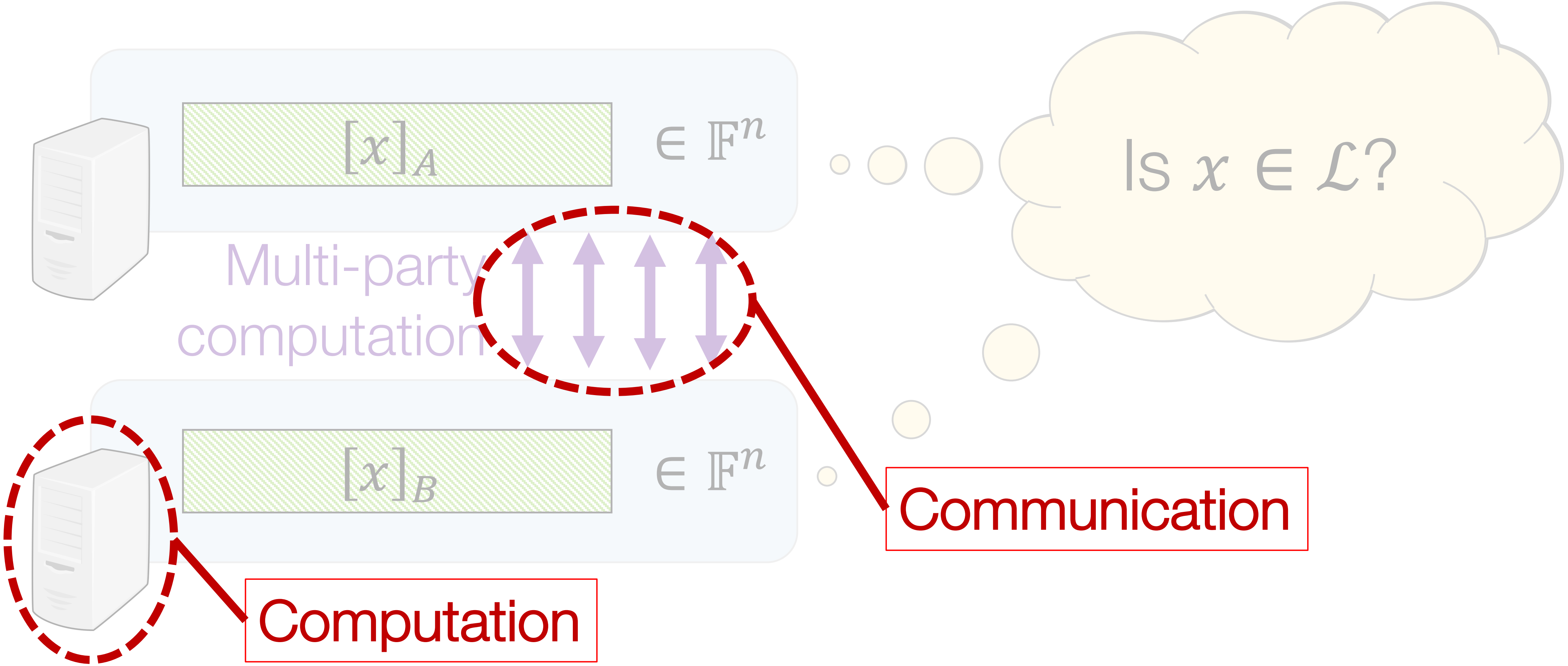
Common task: Test property of secret-shared vector



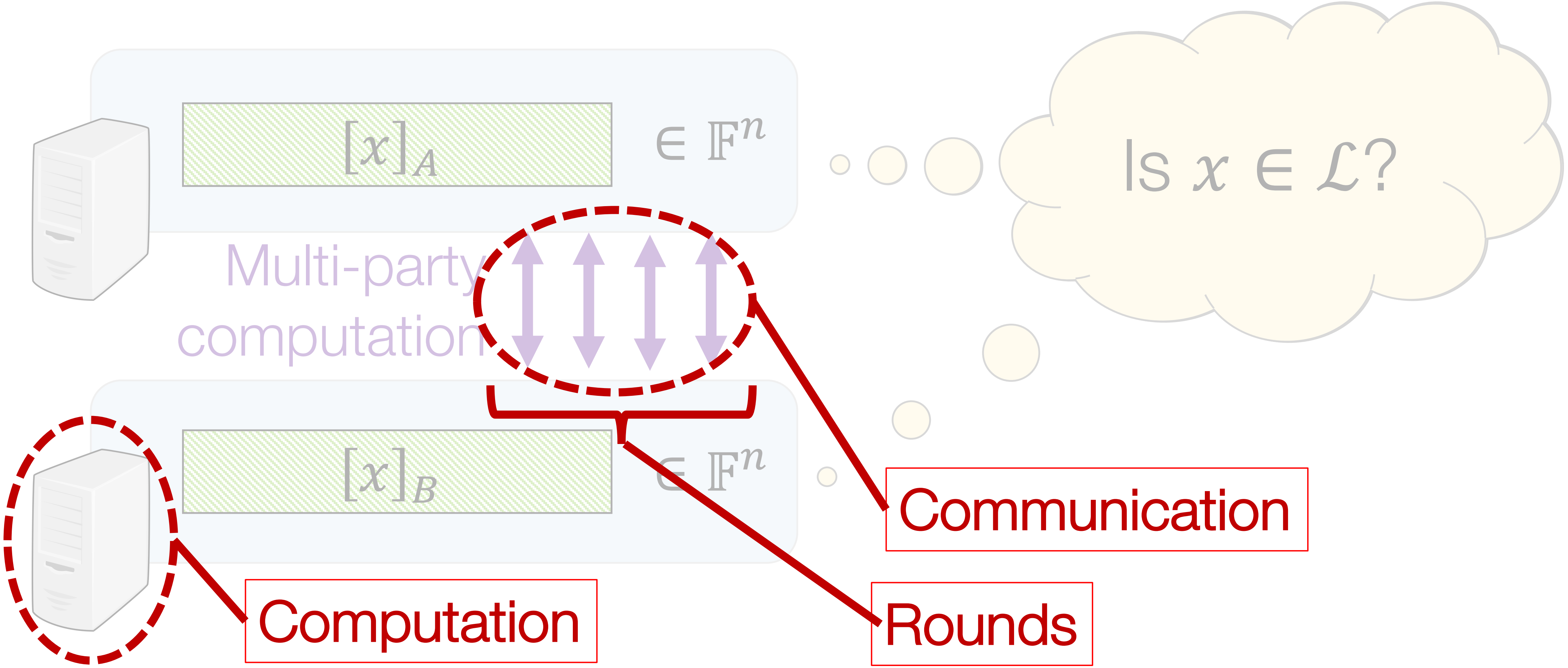
Common task: Test property of secret-shared vector



Common task: Test property of secret-shared vector



Common task: Test property of secret-shared vector



Simple languages are common in applications

Application	Language
PIR writing [OS97], …	Hamming weight one
Private messaging [CBM15], [APY20]	Hamming weight one
Private ads [GLM16], [TNBNB10], …	Hamming weight one, payload in $\{0,1\}$
Private analytics [PBB09], [CB17], …	Hamming weight one, payload in $\{-1,0,1\}$
“ “	Hamming weight $\leq w$, payload in $\{0,1\}$
Verifiable multi-point DPF [CP22]	Hamming weight $= w$
Malicious-secure OT [DLOSS18]	Hamming weight $= w$
E-voting [G05], …	“L_1 norm” $\leq w$
⋮	⋮

State of the art

Many clever special-purpose protocols [CBM15], [GLM16], [DLOSS18], …

- Including defns and protocols influencing our approach [BGI16]

Limitations

– Unclear how/whether special-purpose schemes generalize

– Many schemes require an auxiliary "proof" string

    - Not always feasible in secret-shared setting.

– Unclear optimality

This paper

1. Arithmetic sketching, a unifying abstraction

- The "information-theoretic" part of prior schemes
- Sketching scheme for $\mathcal{L}$ $\Rightarrow$ Protocol for testing shared vector in $\mathcal{L}$

2. New sketches for simple languages

$\Rightarrow$ New protocols for secret-shared, committed, encrypted vectors

3. Lower bounds proving optimality in certain cases

This paper

1. Arithmetic sketching, a unifying abstraction

- The "information-theoretic" part of prior schemes
- Sketching scheme for $\mathcal{L}$ $\Rightarrow$ Protocol for testing shared vector in $\mathcal{L}$

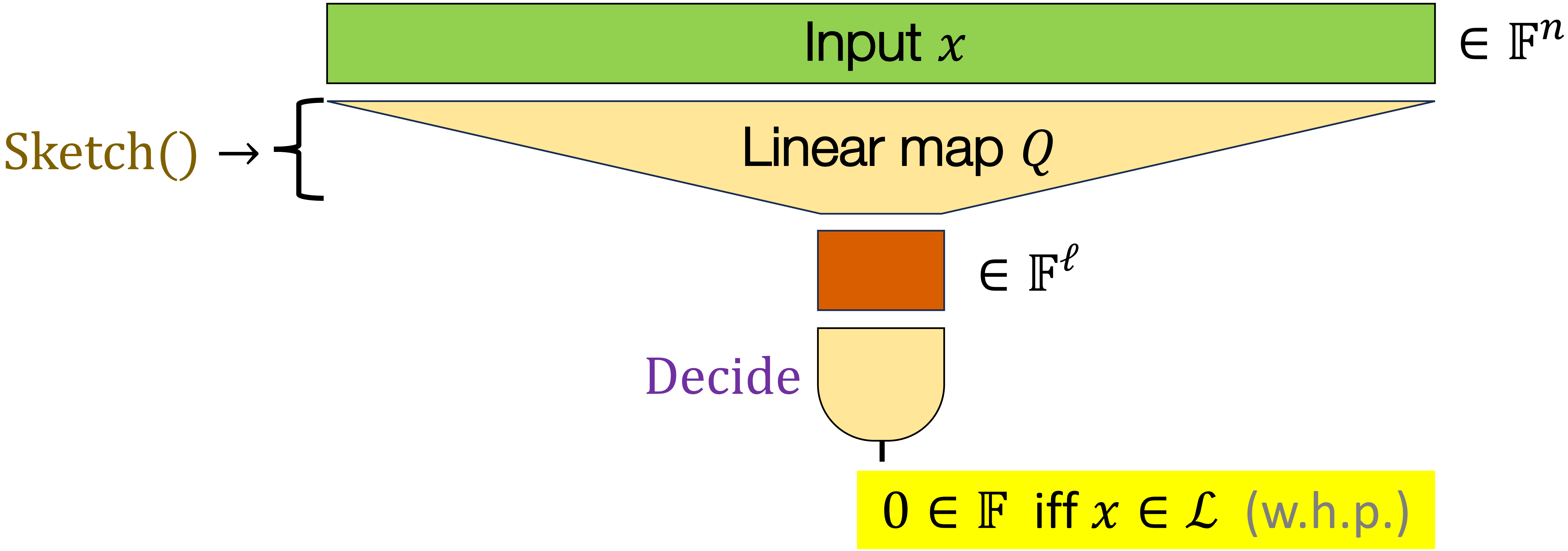
2. New sketches for simple languages

$\Rightarrow$ New protocols for secret-shared, committed, encrypted vectors

3. Lower bounds proving optimality in certain cases

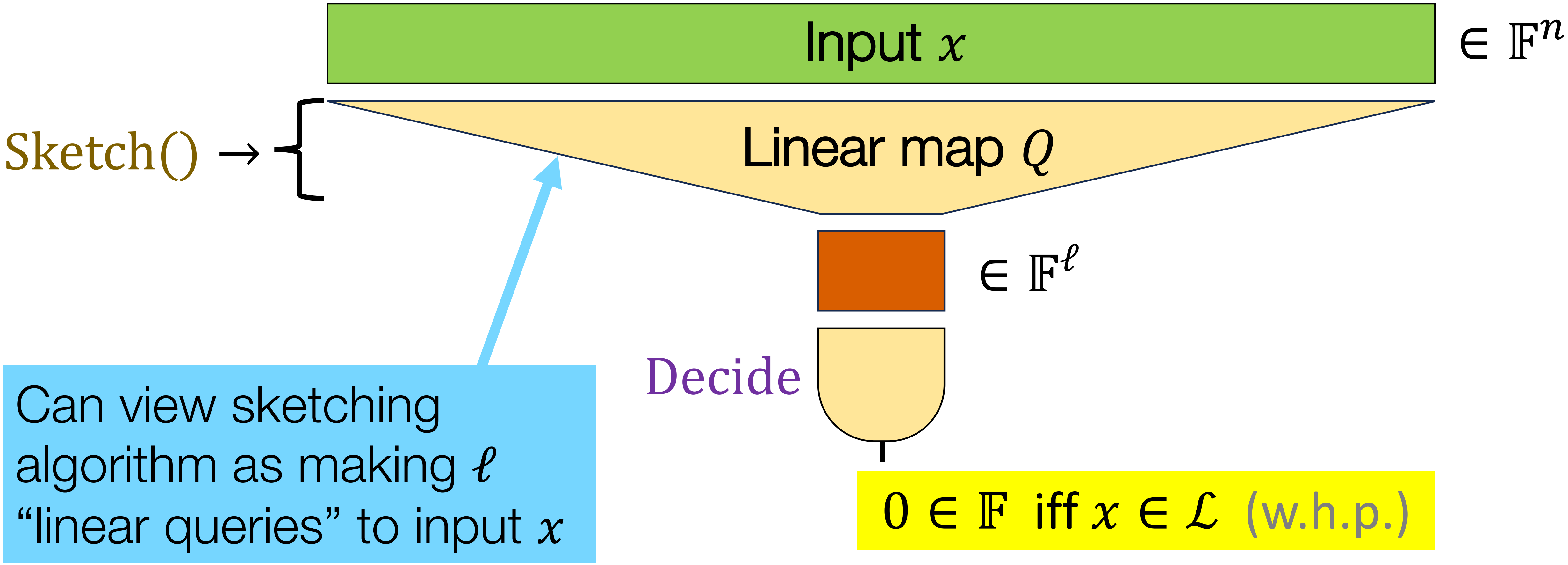
Definition: Arithmetic sketching

For finite field $\mathbb{F}$, language $\mathcal{L} \subseteq \mathbb{F}^n$



Definition: Arithmetic sketching

For finite field $\mathbb{F}$, language $\mathcal{L} \subseteq \mathbb{F}^n$



Definition: Arithmetic sketching

For finite field $\mathbb{F}$, language $\mathcal{L} \subseteq \mathbb{F}^n$

$\text{Sketch}(\,) \to Q \in \mathbb{F}^{\ell \times n}$ — ℓ is the "sketch size"

$\text{Decide}(Q \cdot x) \in \mathbb{F}$ — output can also be a vector in $\mathbb{F}^m$

Must be: Arithmetic circuit with size independent of field $\mathbb{F}$ and input size n

"A fully linear PCP without the proof" [BCIOP12], [BBCGI19]

Completeness — If $x \in \mathcal{L}$: $\Pr[\,\text{Decide}(\text{Sketch}(\,) \cdot x) = 0\,] = 1$

Soundness — If $x \notin \mathcal{L}$: $\Pr[\,\text{Decide}(\text{Sketch}(\,) \cdot x) = 0\,] \leq \epsilon$

Definition: Arithmetic sketching

For finite field $\mathbb{F}$, language $\mathcal{L} \subseteq \mathbb{F}^n$

$\mathrm{Sketch}(\,) \to Q \in \mathbb{F}^{\ell \times n}$ ℓ is the "sketch size"

$\mathrm{Decide}(Q \cdot x) \in \mathbb{F}$ output can also be a vector in $\mathbb{F}^m$

Must be: Arithmetic circuit with size independent of field $\mathbb{F}$ and input size n

"A fully li ear

Without this requirement,
a random linear combination is a good
sketch for any sparse language

Completenes $] = 1$

Soundness   If $x \notin \mathcal{L}$: $\Pr[\,\mathrm{Decide}(\mathrm{Sketch}(\,) \cdot x) = 0\,] \leq \epsilon$

Definition: Arithmetic sketching

For finite field $\mathbb{F}$, language $\mathcal{L} \subseteq \mathbb{F}^n$

$\mathrm{Sketch}(\,) \to Q \in \mathbb{F}^{\ell \times n}$ — ℓ is the "sketch size"

$\mathrm{Decide}(Q \cdot x) \in \mathbb{F}$ — output can also be a vector in $\mathbb{F}^m$

Must be: Arithmetic circuit with size independent of field $\mathbb{F}$ and input size n

"A fully linear PCP without the proof" [BCIOP12], [BBCGI19]

Completeness — If $x \in \mathcal{L}$: $\Pr[\,\mathrm{Decide}(\mathrm{Sketch}(\,) \cdot x) = 0\,] = 1$

Soundness — If $x \notin \mathcal{L}$: $\Pr[\,\mathrm{Decide}(\mathrm{Sketch}(\,) \cdot x) = 0\,] \leq \epsilon$

Definition: Arithmetic sketching

For finite field $\mathbb{F}$, language $\mathcal{L} \subseteq \mathbb{F}^n$

$\mathrm{Sketch}(\,) \to Q \in \mathbb{F}^{\ell \times n}$ — ℓ is the "sketch size"

$\mathrm{Decide}(Q \cdot x) \in \mathbb{F}$ — output can also be a vector in $\mathbb{F}^m$

Must be: Arithmetic circuit with size independent of field $\mathbb{F}$ and input size n

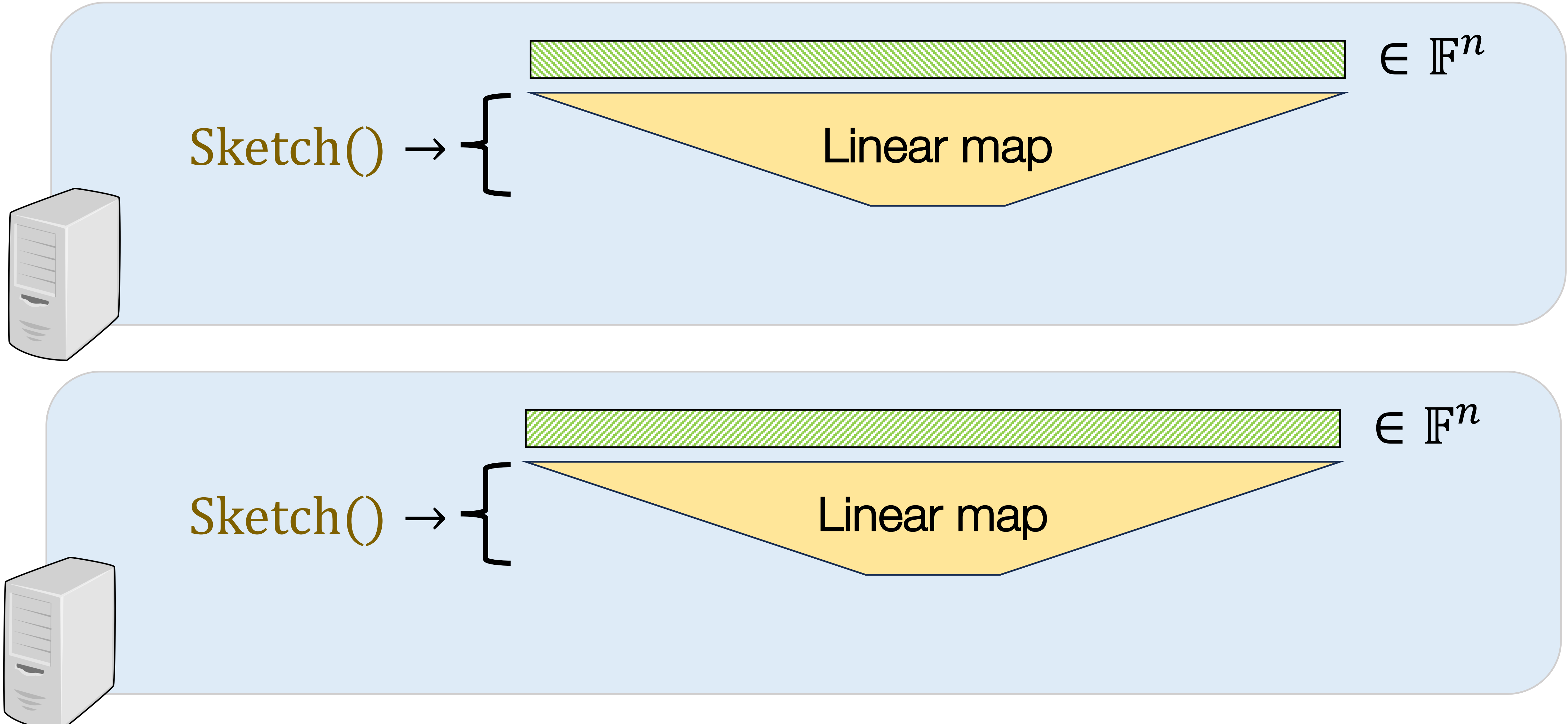
"A fully linear PCP without the proof" [BCIOP12], [BBCGI19]

<u>Completeness</u> If $x \in \mathcal{L}$: $\Pr[\,\mathrm{Decide}(\mathrm{Sketch}(\,) \cdot x) = 0\,] = 1$

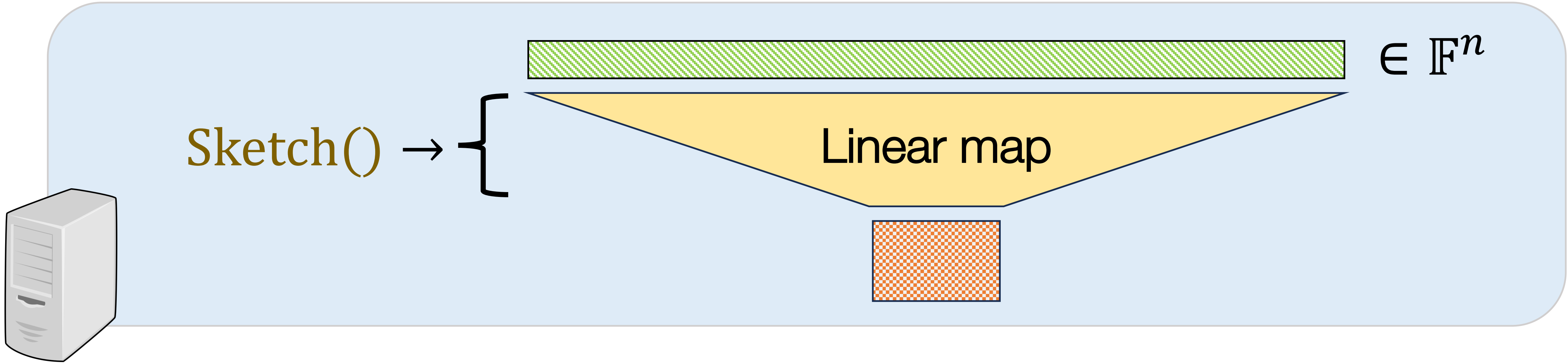
<u>Soundness</u> If $x \notin \mathcal{L}$: $\Pr[\,\mathrm{Decide}(\mathrm{Sketch}(\,) \cdot x) = 0\,] \leq \epsilon$

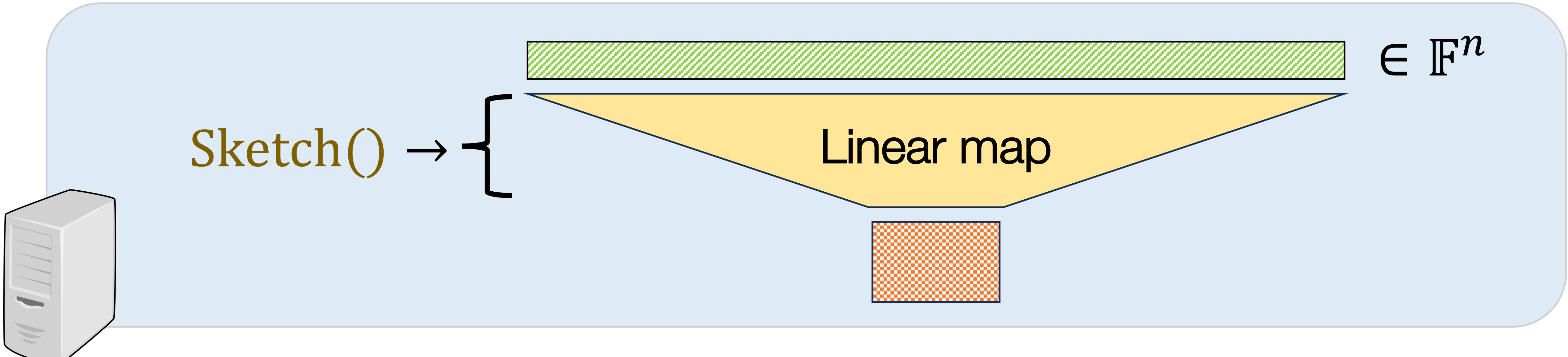
Application: Secret-shared data

Application: Secret-shared data

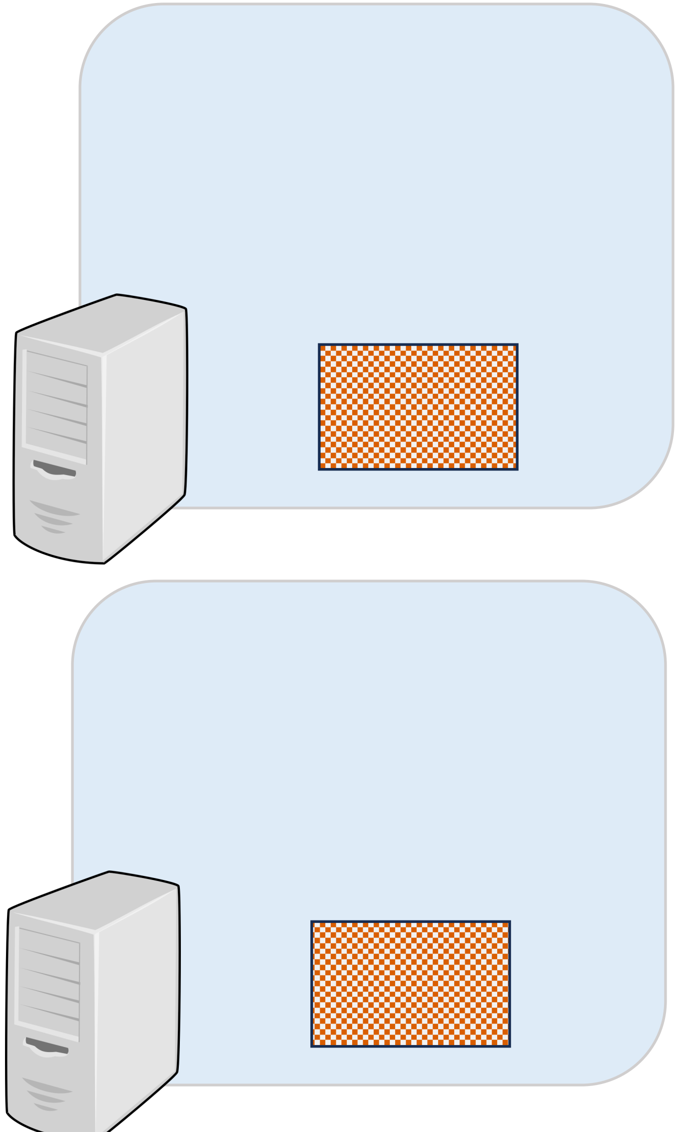


Application: Secret-shared data

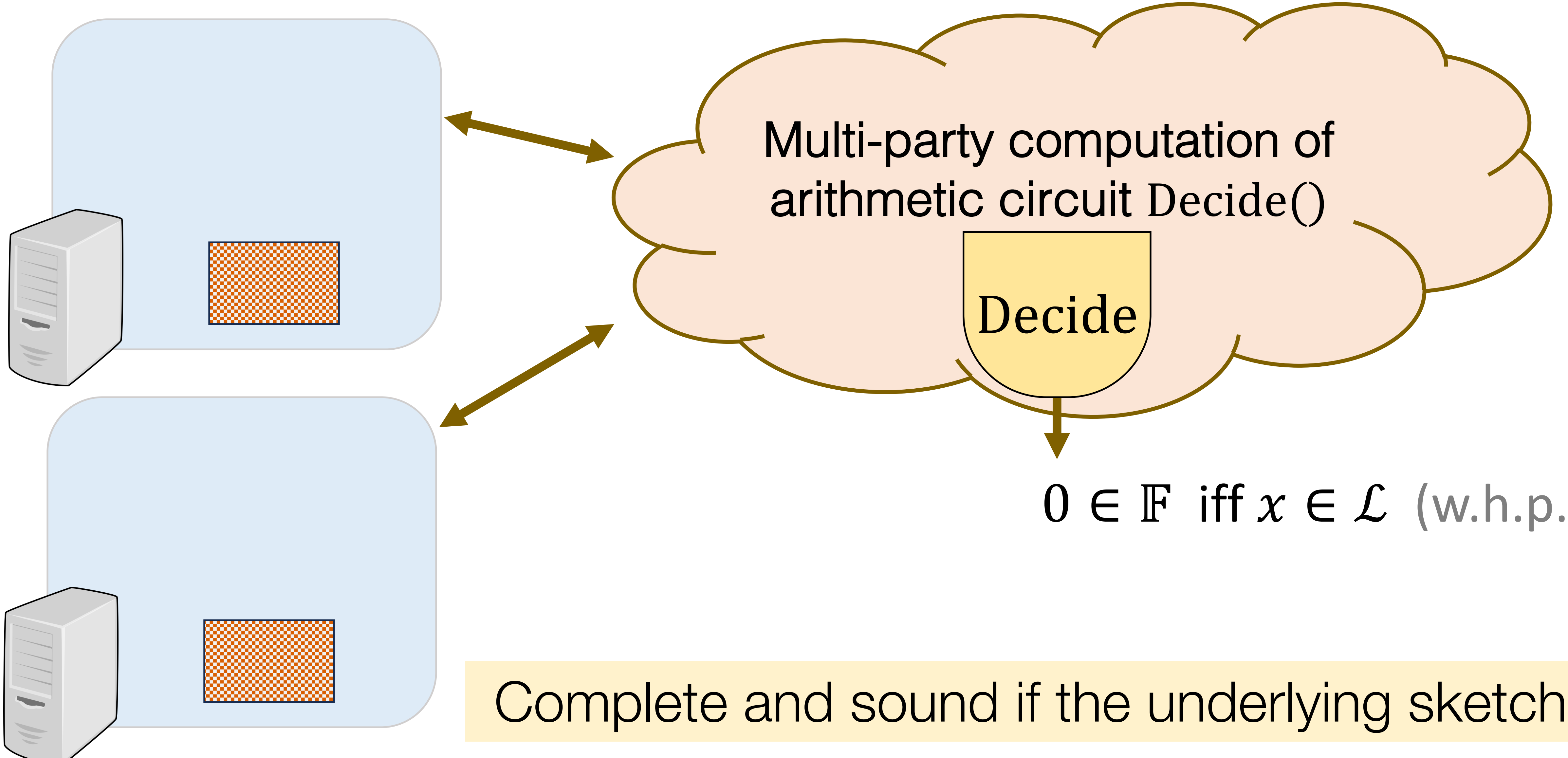




Application: Secret-shared data



Application: Secret-shared data



$0 \in \mathbb{F}$ iff $x \in \mathcal{L}$ (w.h.p.)

Complete and sound if the underlying sketch is

Zero knowledge comes for free

Standard zero knowledge: Privacy when input $x \in \mathcal{L}$.

For all $x \in \mathcal{L}$*, output of* $\mathbf{Decide}()$ *"leaks nothing" about* x.

→ Automatically provided when using MPC to compute $\mathbf{Decide}()$

Two-sided zero knowledge: Privacy for all inputs x.

For all x*, output of* $\mathbf{Decide}()$ *"leaks nothing" except whether* $x \in \mathcal{L}$.

→ Can achieve by randomizing output of decision circuit

Zero knowledge comes for free

Standard zero knowledge: Privacy when input $x \in \mathcal{L}$.

For all $x \in \mathcal{L}$, output of $\mathbf{Decide}()$ *"leaks nothing" about x.*

→ Automatically provided when using MPC to compute $\mathbf{Decide}()$

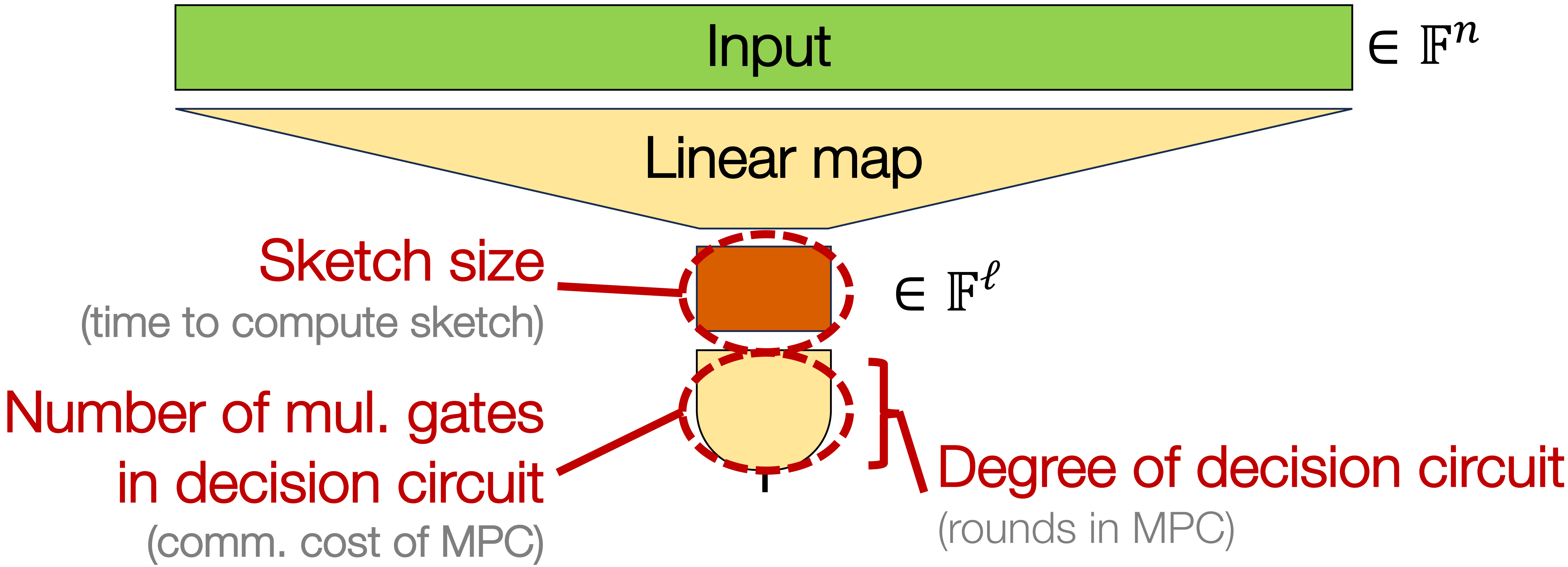
Two-sided zero knowledge: Privacy for all inputs x.

For all x, output of $\mathbf{Decide}()$ *"leaks nothing" except whether $x \in \mathcal{L}$.*

→ Can achieve by randomizing output of decision circuit

Complexity metrics

For finite field $\mathbb{F}$, language $\mathcal{L} \subseteq \mathbb{F}^n$



This paper

1. Arithmetic sketching, a unifying abstraction
- The "information-theoretic" part of prior schemes
- Sketching scheme for $\mathcal{L}$ $\Rightarrow$ Protocol for testing shared vector in $\mathcal{L}$

2. New sketches for simple languages

$\Rightarrow$ New protocols for secret-shared, committed, encrypted vectors

3. Lower bounds proving optimality in certain cases

This paper

1. Arithmetic sketching, a unifying abstraction

- The "information-theoretic" part of prior schemes
- Sketching scheme for $\mathcal{L}$ $\Rightarrow$ Protocol for testing shared vector in $\mathcal{L}$

2. New sketches for simple languages

$\Rightarrow$ New protocols for secret-shared, committed, encrypted vectors

3. Lower bounds proving optimality in certain cases

New constructions

Sketches for:

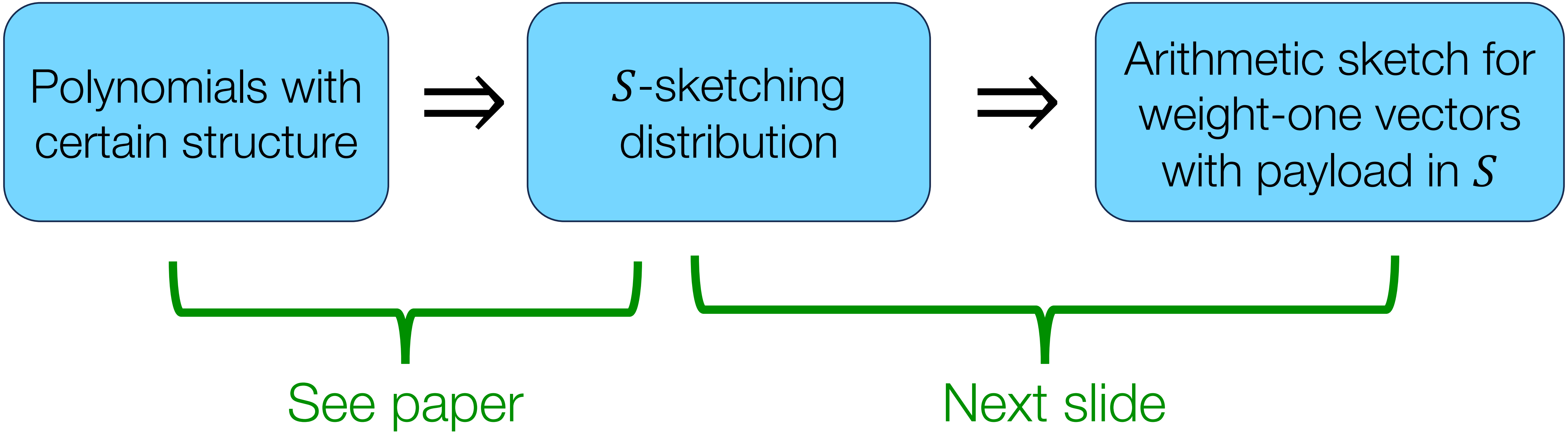
- weight-one vectors — unifies prior constructions + new ones too
- weight-w vectors — first constructions
- bounded "L_1-norm" — first constructions

General compiler: (see paper)

Arithmetic sketch for $\mathcal{L}$ $\Rightarrow$ Malicious-secure MPC testing vector in $\mathcal{L}$

General framework

Sketching for weight-one vectors



S-sketching distribution

Inspired by AMD codes [CDFPW18]

Two algorithms, defined with respect to set $S \subseteq \mathbb{F}$:

$$\mathsf{Sample}() \to \begin{bmatrix} \, \\ \, \\ \, \end{bmatrix} \in \mathbb{F}^{\ell} \qquad \mathsf{Verify}\left(\begin{bmatrix} \, \\ \, \\ \, \end{bmatrix}\right) \in \mathbb{F} \; \Big\} \text{ Arithmetic circuit}$$

Completeness

For all $\beta \in S$ ...

$$\mathsf{Verify}\left(\beta \cdot \begin{bmatrix} \, \\ \, \\ \, \end{bmatrix}\right) = 0$$

Soundness

For all $\beta' \in \mathbb{F} \setminus S$ ...

$$\mathsf{Verify}\left(\beta' \cdot \begin{bmatrix} \, \\ \, \\ \, \end{bmatrix}\right) \neq 0$$

w.h.p.

Manipulation detection

For non-zero $\gamma \in \mathbb{F}$, $\Delta \in \mathbb{F}^{\ell}$

$$\mathsf{Verify}\left(\gamma \cdot \begin{bmatrix} \, \\ \, \\ \, \end{bmatrix} + \begin{bmatrix} \, \\ \Delta \\ \, \end{bmatrix}\right) \neq 0$$

w.h.p.

S-sketching distribution

Inspired by AMD codes [CDFPW18]

Two algorithms, defined with respect to set $S \subseteq \mathbb{F}$:

$$\mathrm{Sample}() \to \begin{bmatrix} \\ \\ \end{bmatrix} \in \mathbb{F}^{\ell} \qquad \underbrace{\mathrm{Verify}\left(\begin{bmatrix} \\ \\ \end{bmatrix}\right) \in \mathbb{F}}_{\text{Arithmetic circuit}}$$

Completeness

For all $\beta \in S$ ...

$$\mathrm{Verify}\left(\beta \cdot \begin{bmatrix} \\ \\ \end{bmatrix}\right) = 0$$

Soundness

For all $\beta' \in \mathbb{F} \setminus S$ ...

$$\mathrm{Verify}\left(\beta' \cdot \begin{bmatrix} \\ \\ \end{bmatrix}\right) \neq 0$$

w.h.p.

Manipulation detection

For non-zero $\gamma \in \mathbb{F}$, $\Delta \in \mathbb{F}^{\ell}$

$$\mathrm{Verify}\left(\gamma \cdot \begin{bmatrix} \\ \\ \end{bmatrix} + \begin{bmatrix} \\ \Delta \\ \end{bmatrix}\right) \neq 0$$

w.h.p.

S-sketching distribution

Inspired by AMD codes [CDFPW18]

Two algorithms, defined with respect to set $S \subseteq \mathbb{F}$:

$$\mathsf{Sample}() \to \begin{bmatrix} \ \\ \ \\ \ \end{bmatrix} \in \mathbb{F}^{\ell} \qquad \mathsf{Verify}\left(\begin{bmatrix} \ \\ \ \\ \ \end{bmatrix}\right) \in \mathbb{F}$$

Arithmetic circuit

Completeness

For all $\beta \in S$ ...

$$\mathsf{Verify}\left(\beta \cdot \begin{bmatrix} \ \\ \ \\ \ \end{bmatrix}\right) = 0$$

Soundness

For all $\beta' \in \mathbb{F} \setminus S$ ...

$$\mathsf{Verify}\left(\beta' \cdot \begin{bmatrix} \ \\ \ \\ \ \end{bmatrix}\right) \neq 0$$

w.h.p.

Manipulation detection

For non-zero $\gamma \in \mathbb{F}$, $\Delta \in \mathbb{F}^{\ell}$

$$\mathsf{Verify}\left(\gamma \cdot \begin{bmatrix} \ \\ \ \\ \ \end{bmatrix} + \begin{bmatrix} \ \\ \Delta \\ \ \end{bmatrix}\right) \neq 0$$

w.h.p.

S-sketching distribution

Inspired by AMD codes [CDFPW18]

Two algorithms, defined with respect to set $S \subseteq \mathbb{F}$:

$\mathrm{Sample}() \rightarrow \begin{bmatrix} \square \\ \square \\ \square \end{bmatrix} \in \mathbb{F}^{\ell}$

$\left.\mathrm{Verify}\left(\begin{bmatrix} \square \\ \square \\ \square \end{bmatrix}\right) \in \mathbb{F} \right\}$ Arithmetic circuit

Completeness

For all $\beta \in S$ ...

$\mathrm{Verify}\left(\beta \cdot \begin{bmatrix} \square \\ \square \\ \square \end{bmatrix}\right) = 0$

Soundness

For all $\beta' \in \mathbb{F} \setminus S$ ...

$\mathrm{Verify}\left(\beta' \cdot \begin{bmatrix} \square \\ \square \\ \square \end{bmatrix}\right) \neq 0$ w.h.p.

Manipulation detection

For non-zero $\gamma \in \mathbb{F}$, $\Delta \in \mathbb{F}^{\ell}$

$\mathrm{Verify}\left(\gamma \cdot \begin{bmatrix} \square \\ \square \\ \square \end{bmatrix} + \begin{bmatrix} \square \\ \Delta \\ \square \end{bmatrix}\right) \neq 0$ w.h.p.

Example: Construction of S-sketching distribution, for $S = \{-1,0,1\}$

Sample():

- Choose random $r \leftarrow_R \mathbb{F}$
- Output $(r, r^3) \in \mathbb{F}^3$

Verify(s_1, s_2):

- Output $s_1^3 - s_2 \in \mathbb{F}$

Given: S-sketching distribution $(\mathsf{Sample}, \mathsf{Verify})$

Want to construct: Arithmetic sketch $(\mathsf{Sketch}, \mathsf{Decide})$

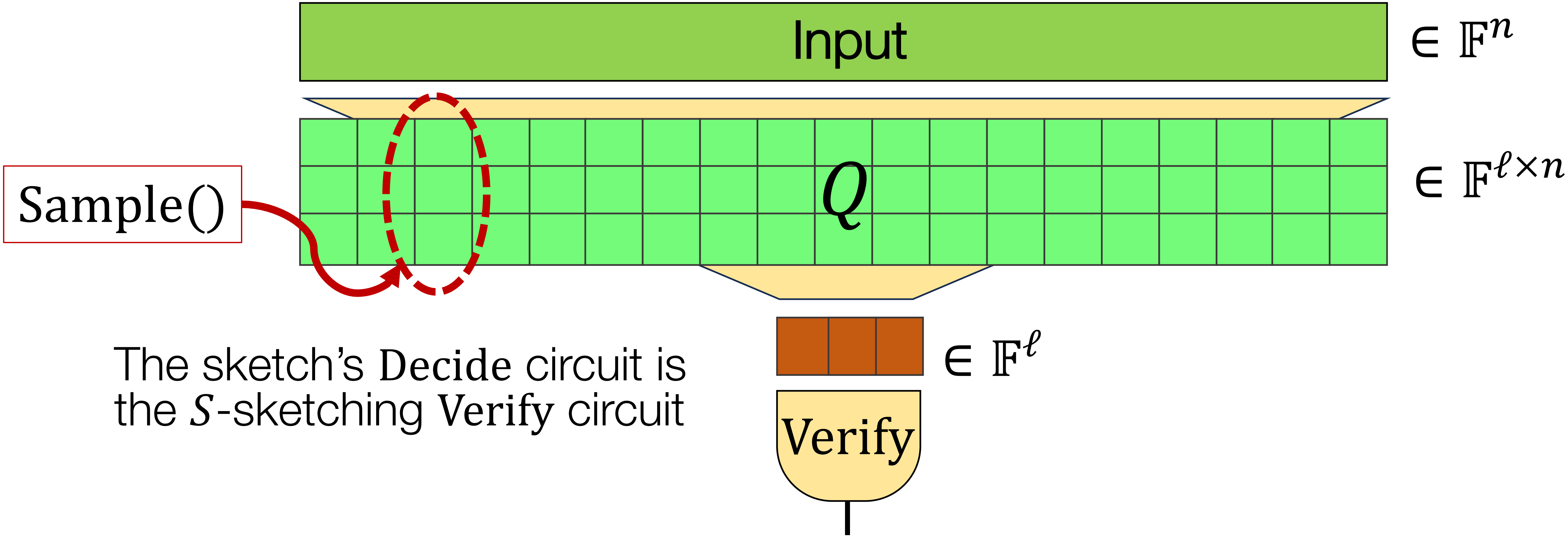
$\mathsf{Sketch}()$: Run $\mathsf{Sample}()$ algorithm n times (each output is in $\mathbb{F}^{\ell}$)

Use the samples as the $\ell \times n$ query matrix $Q \in \mathbb{F}^{\ell \times n}$

$\mathsf{Decide}(a \in \mathbb{F}^{\ell})$: Output $\mathsf{Verify}(a)$

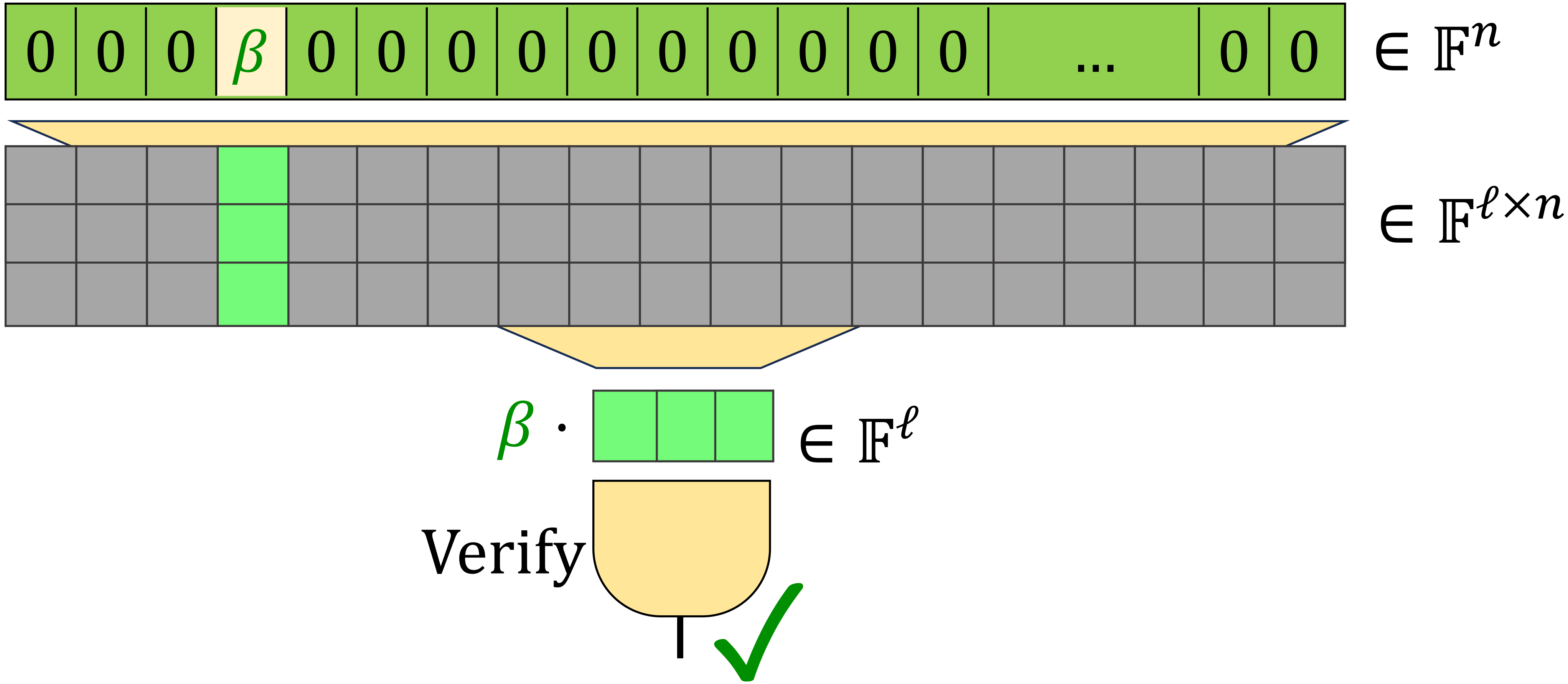
Construction: Sketching for weight-one, payload in S from S-sketching distribution

The sketching matrix is n samples from the S-sketching distribution

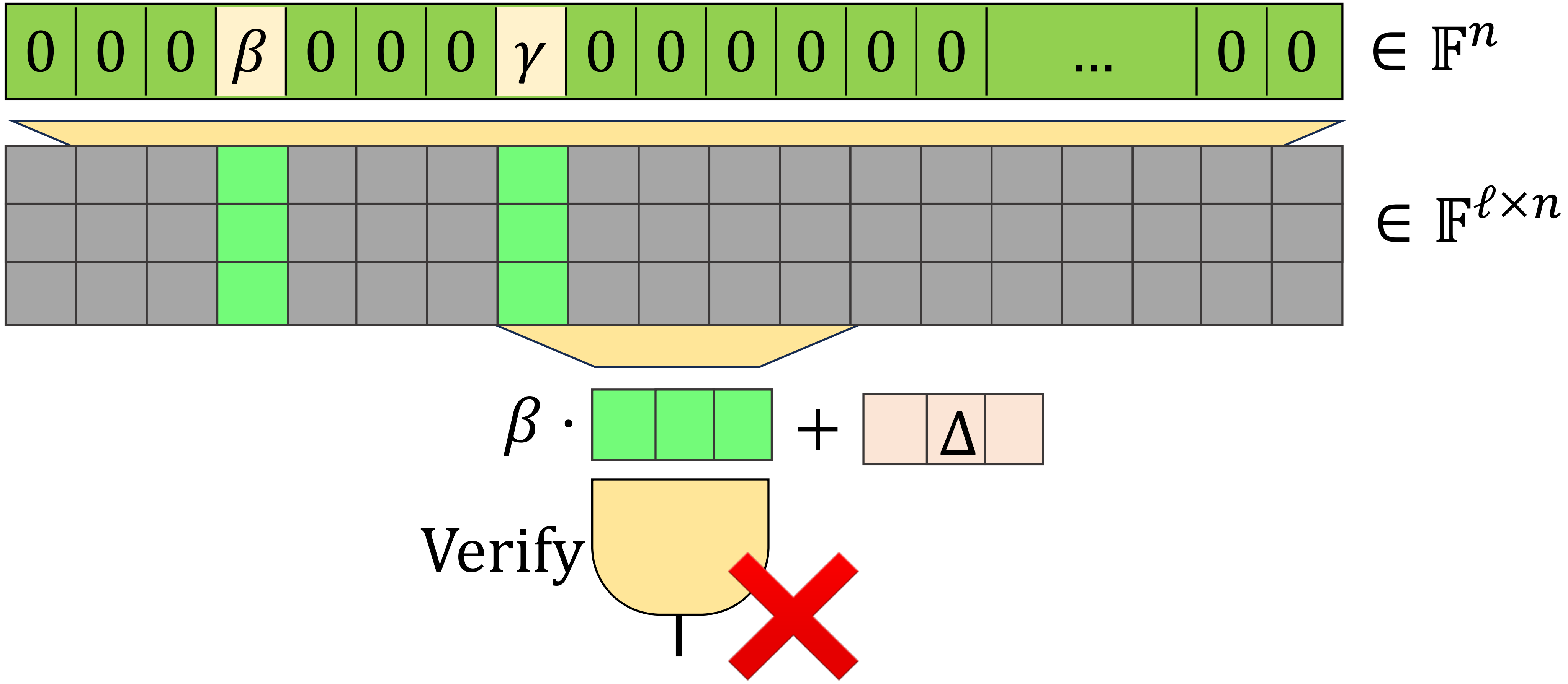


The sketch's Decide circuit is the S-sketching Verify circuit

Completeness: Weight one, payload $\in S$



Soundness: Weight ≥ 1 or payload $\notin S$



Results: Sketching for weight-one

Captures existing ad-hoc schemes [BGI16]

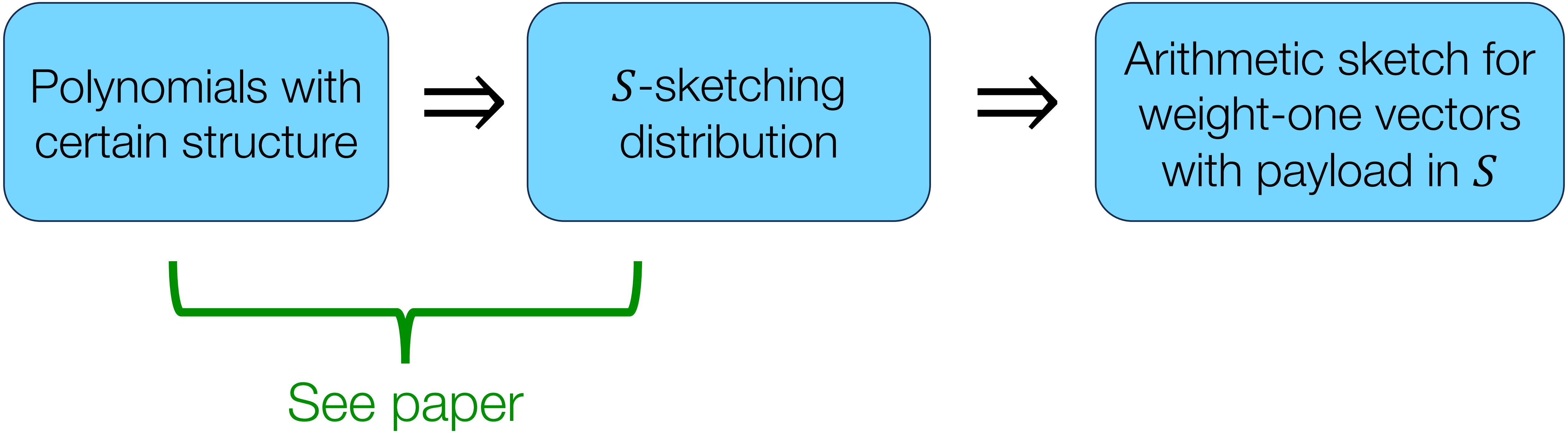
- $S = \{0,1\}$
- $S = \{1\}$

New constructions when $\mathrm{char}(\mathbb{F}) > 2$:

- $S = \mathbb{F}$ PIR writing, messaging
- $S = \{-1,0,1\}$ Upvoting/downvoting in private aggregation

General framework

Sketching for weight-one vectors



Sketching for weight-w vectors

1. View input $\boldsymbol{x} \in \mathbb{F}^n$ as coefficients of a polynomial $\boldsymbol{p}$ of degree $\le \boldsymbol{n-1}$
   - With one linear query, can evaluate $p(r)$ for any $r \in \mathbb{F}$

$x =$	x_0	x_1	x_2	x_3	x_4	…	x_{n-2}	x_{n-1}	$\in \mathbb{F}^n$
	1	r	r^2	r^3	r^4	…	r^{n-2}	r^{n-1}	

2. Apply existing polynomial-sparsity test [BT88, GJR10]
   - Tests whether p has w non-zero coefficients using $2w+1$ evaluations of p
   - Decision routine computes determinant

Sketching for “L_1-norm” $\leq w$

1. Make $w+1$ linear queries of the form:

$x =$	x_1	x_2	x_3	x_4	x_5	…	x_{n-1}	x_n	$\in \mathbb{F}^n$
Query i:	$r_1{}^i$	$r_2{}^i$	$r_3{}^i$	$r_4{}^i$	$r_5{}^i$	…	$r_{n-1}{}^i$	$r_n{}^i$	

2. Query answers give power sums of some values $\boldsymbol{z_1}, \dots, \boldsymbol{z_m} \in \mathbb{F}$:

$$(z_1 + \cdots + z_m), \quad (z_1^2 + \cdots + z_m^2), \quad \dots, \quad (z_1^{w+1} + \cdots + z_m^{w+1})$$

where $\boldsymbol{m}$ is the $\boldsymbol{L_1}$-norm of $\boldsymbol{x}$

3. Use Newton relations to test whether $\boldsymbol{m} \leq \boldsymbol{w}$

New constructions

Sketches for:

- weight-one vectors — unifies prior constructions + new ones too
- weight-w vectors — first constructions
- bounded L_1-norm — first constructions

General compiler: (see paper)

Arithmetic sketch for $\mathcal{L}$ $\Rightarrow$ Malicious-secure MPC testing vector in $\mathcal{L}$

This paper

1. Arithmetic sketching, a unifying abstraction

- The "information-theoretic" part of prior schemes
- Sketching scheme for $\mathcal{L}$ $\Rightarrow$ Protocol for testing shared vector in $\mathcal{L}$

2. New sketches for simple languages

$\Rightarrow$ New protocols for secret-shared, committed, encrypted vectors

3. Lower bounds proving optimality in certain cases

This paper

1. Arithmetic sketching, a unifying abstraction

- The “information-theoretic” part of prior schemes
- Sketching scheme for $\mathcal{L}$ $\Rightarrow$ Protocol for testing shared vector in $\mathcal{L}$

2. New sketches for simple languages

$\Rightarrow$ New protocols for secret-shared, committed, encrypted vectors

3. Lower bounds proving optimality in certain cases

Lower bounds (see paper)

From algebraic techniques

- No arithmetic sketch for weight-one vectors with sketch size ≤ 2

  ⇒ Our construction with sketch size 3 has optimal size

From communication complexity

- No arithmetic sketch for L_p norm $\leq w$ when $p > 1$
- No arithmetic sketch for “contains at least one value in S”
- No arithmetic sketch all zeros with contiguous run of ones

Arithmetic sketching

Decide $x \in \mathcal{L}$ by applying:

- a randomized linear map then
- a small arithmetic circuit.

+ Simple, useful tool

❑ Improved sketches?

❑ More sketchable languages?

❑ Approximate notions?

https://eprint.iacr.org/2023/1012