

Non-Interactive Batch Arguments and more



Arka Rai Choudhuri

University of California, Berkeley



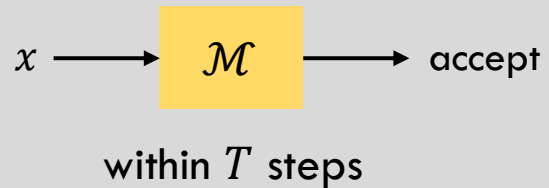
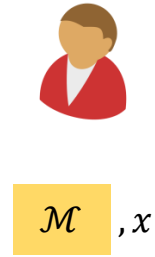
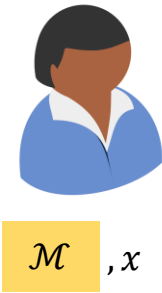
Zhengzhong Jin

Johns Hopkins University

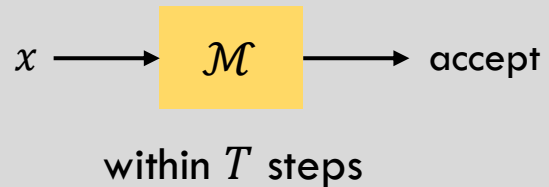
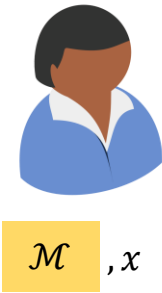
Abhishek Jain

Johns Hopkins University

Succinct Non-Interactive Arguments (SNARGs)



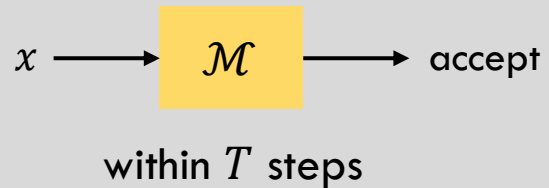
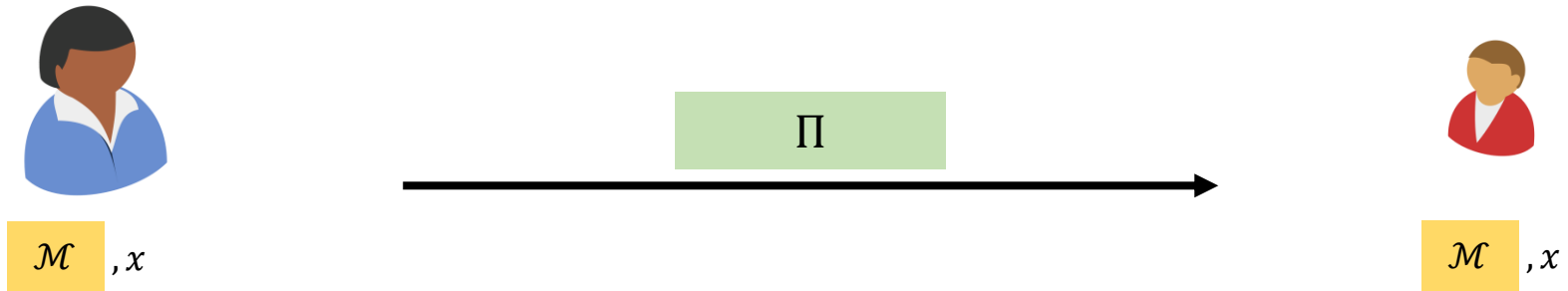
Succinct Non-Interactive Arguments (SNARGs)



wants to delegate computation to



Succinct Non-Interactive Arguments (SNARGs)



Succinct Non-Interactive Arguments (SNARGs)

Common Reference String (CRS)



\mathcal{M}, x

Π



\mathcal{M}, x

$x \longrightarrow \mathcal{M} \longrightarrow \text{accept}$

within T steps

Succinct Non-Interactive Arguments (SNARGs)

Common Reference String (CRS)



\mathcal{M}, x

Π



\mathcal{M}, x

Π is publicly verifiable

$x \longrightarrow \mathcal{M} \longrightarrow \text{accept}$

within T steps

Succinct Non-Interactive Arguments (SNARGs)

Common Reference String (CRS)



\mathcal{M}, x

$\leftarrow \text{polylog}(T) \rightarrow$

Π



\mathcal{M}, x

Verifier **running time**:
 $\text{polylog}(T)$

Π is **publicly verifiable**

$x \rightarrow \mathcal{M} \rightarrow \text{accept}$

within T steps

Succinct Non-Interactive Arguments (SNARGs)

Common Reference String (CRS)



\mathcal{M}, x

$\leftarrow \text{polylog}(T) \rightarrow$

Π



\mathcal{M}, x

Verifier running time:
 $\text{polylog}(T)$

Π is publicly verifiable

$x \rightarrow \mathcal{M} \rightarrow \text{accept}$

within T steps

No PPT  can produce accepting Π if

$x \rightarrow \mathcal{M} \xrightarrow{\text{X}} \text{accept}$

within T steps

Succinct Non-Interactive Arguments (SNARGs)

Common Reference String (CRS)



\mathcal{M}, x

$\leftarrow \text{polylog}(T) \rightarrow$

Π



\mathcal{M}, x

Verifier running time:
 $\text{polylog}(T)$

Π is publicly verifiable

$x \rightarrow \mathcal{M} \rightarrow \text{accept}$

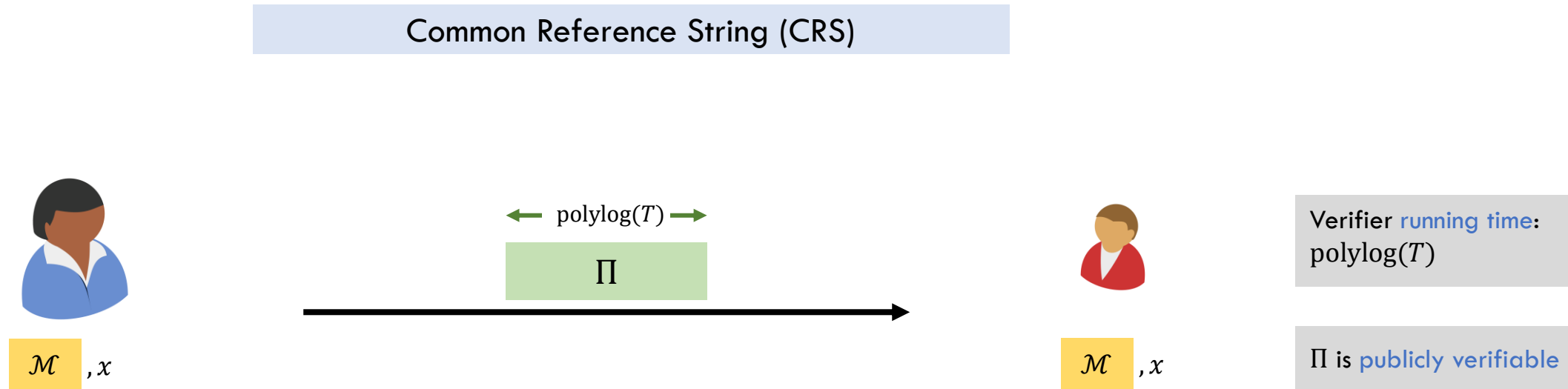
within T steps

No PPT  can produce accepting x, Π if

$x \rightarrow \mathcal{M} \xrightarrow{\text{X}} \text{accept}$

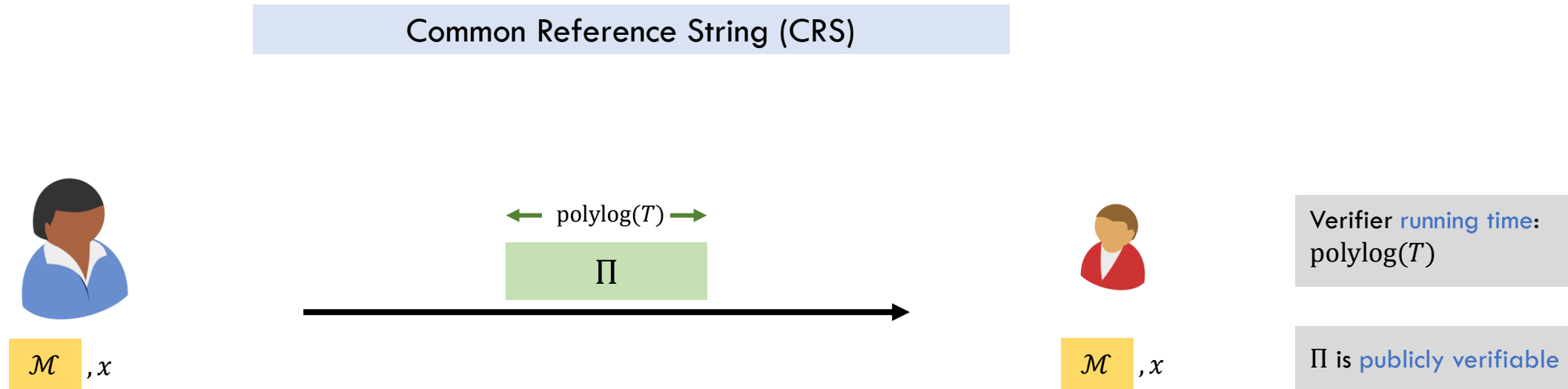
within T steps

Succinct Non-Interactive Arguments (SNARGs)



What kind of computation can we hope to **delegate** based on **standard assumptions**?

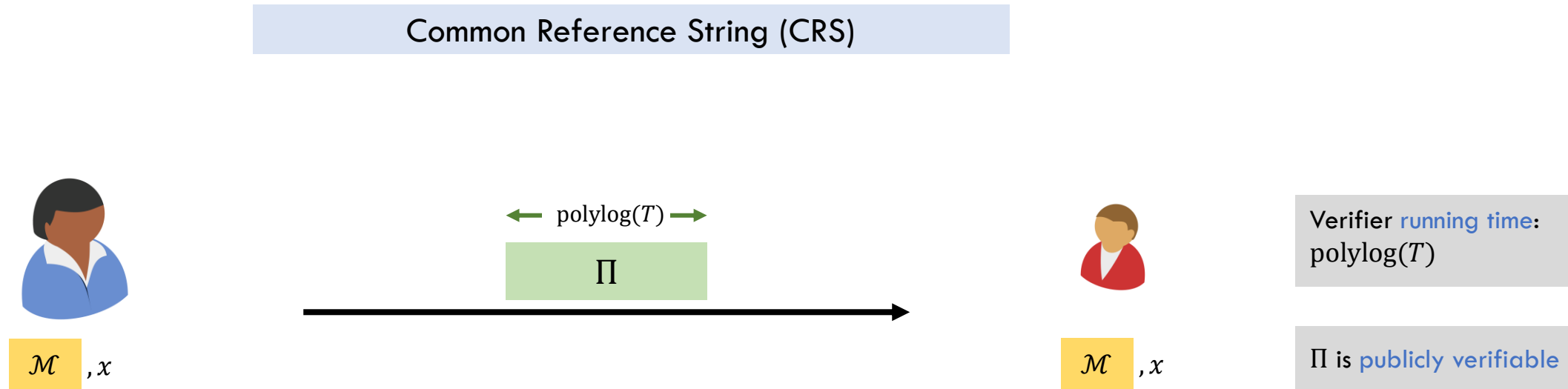
Succinct Non-Interactive Arguments (SNARGs)



What kind of computation can we hope to **delegate** based on **standard assumptions**?

- Nondeterministic polynomial-time computation (NP)? **Unlikely!** [Gentry-Wichs'11]

Succinct Non-Interactive Arguments (SNARGs)



What kind of computation can we hope to **delegate** based on **standard assumptions**?

- Nondeterministic polynomial-time computation (NP)? **Unlikely!** [Gentry-Wichs'11]
- Deterministic polynomial-time computation (P)?

Prior Works

Prior Works

Non-falsifiable assumptions/ Random oracle model

[Micali'94, Groth'10, Lipmaa'12, Damgård-Faust-Hazay'12, Gennaro-Gentry-Parno-Raykova'13, Bitansky-Chiesa-Ishai-Ostrovsky-Paneth'13, Bitansky-Canetti-Chiesa-Tromer'13, Bitansky-Canetti-Chiesa-Goldwasser-Lin-Rubinstein-Tromer'17]

Some works can
delegate NP

Prior Works

Non-falsifiable assumptions/ Random oracle model

[Micali'94, Groth'10, Lipmaa'12, Damgård-Faust-Hazay'12, Gennaro-Gentry-Parno-Raykova'13, Bitansky-Chiesa-Ishai-Ostrovsky-Paneth'13, Bitansky-Canetti-Chiesa-Tromer'13, Bitansky-Canetti-Chiesa-Goldwasser-Lin-Rubinstein-Tromer'17]

Some works can
delegate NP

“Less standard” assumptions

[Canetti-Holmgren-Jain-Vaikuntanathan'15, Koppula-Lewko-Waters'15, Bitansky-Garg-Lin-Pass-Telang'15, Canetti-Holmgren'16, Ananth-Chen-Chung-Lin-Lin'16, Chen-Chow-Chung-Lai-Lin-Zhou'16, Paneth-Rothblum'17, Canetti-Chen-Holmgren-Lombardi-Rothblum-Rothblum-Wichs'19, Kalai-Paneth-Yang'19]

Delegation for P

Prior Works

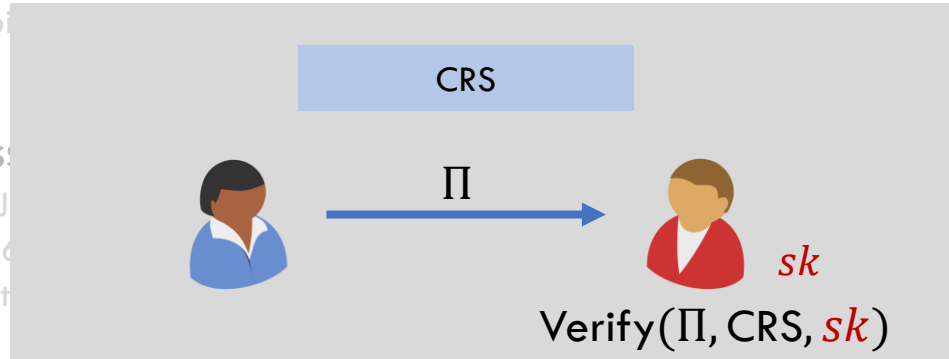
Non-falsifiable assumptions/ Random oracle model

[Micali'94, Groth'10, Lipmaa'12, Damgård-Faust-Hazay'12, Gennaro-Gentry-Parno-Raykova'13, Bitansky-Chiesa-Ishai-Ostrovsky-Paneth'13, Bitansky-Canetti-Chiesa-Tromer'13, Bitansky-Canetti-Chiesa-Goldwasser-Lin-Rubinfeld'14, Gennaro-Halevi-Parno-Raykova'14, Gennaro-Halevi-Parno-Raykova'15, Gennaro-Halevi-Parno-Raykova'16, Gennaro-Halevi-Parno-Raykova'17, Gennaro-Halevi-Parno-Raykova'18, Gennaro-Halevi-Parno-Raykova'19, Gennaro-Halevi-Parno-Raykova'20, Gennaro-Halevi-Parno-Raykova'21, Gennaro-Halevi-Parno-Raykova'22, Gennaro-Halevi-Parno-Raykova'23, Gennaro-Halevi-Parno-Raykova'24, Gennaro-Halevi-Parno-Raykova'25, Gennaro-Halevi-Parno-Raykova'26, Gennaro-Halevi-Parno-Raykova'27, Gennaro-Halevi-Parno-Raykova'28, Gennaro-Halevi-Parno-Raykova'29, Gennaro-Halevi-Parno-Raykova'30, Gennaro-Halevi-Parno-Raykova'31, Gennaro-Halevi-Parno-Raykova'32, Gennaro-Halevi-Parno-Raykova'33, Gennaro-Halevi-Parno-Raykova'34, Gennaro-Halevi-Parno-Raykova'35, Gennaro-Halevi-Parno-Raykova'36, Gennaro-Halevi-Parno-Raykova'37, Gennaro-Halevi-Parno-Raykova'38, Gennaro-Halevi-Parno-Raykova'39, Gennaro-Halevi-Parno-Raykova'40, Gennaro-Halevi-Parno-Raykova'41, Gennaro-Halevi-Parno-Raykova'42, Gennaro-Halevi-Parno-Raykova'43, Gennaro-Halevi-Parno-Raykova'44, Gennaro-Halevi-Parno-Raykova'45, Gennaro-Halevi-Parno-Raykova'46, Gennaro-Halevi-Parno-Raykova'47, Gennaro-Halevi-Parno-Raykova'48, Gennaro-Halevi-Parno-Raykova'49, Gennaro-Halevi-Parno-Raykova'50, Gennaro-Halevi-Parno-Raykova'51, Gennaro-Halevi-Parno-Raykova'52, Gennaro-Halevi-Parno-Raykova'53, Gennaro-Halevi-Parno-Raykova'54, Gennaro-Halevi-Parno-Raykova'55, Gennaro-Halevi-Parno-Raykova'56, Gennaro-Halevi-Parno-Raykova'57, Gennaro-Halevi-Parno-Raykova'58, Gennaro-Halevi-Parno-Raykova'59, Gennaro-Halevi-Parno-Raykova'60, Gennaro-Halevi-Parno-Raykova'61, Gennaro-Halevi-Parno-Raykova'62, Gennaro-Halevi-Parno-Raykova'63, Gennaro-Halevi-Parno-Raykova'64, Gennaro-Halevi-Parno-Raykova'65, Gennaro-Halevi-Parno-Raykova'66, Gennaro-Halevi-Parno-Raykova'67, Gennaro-Halevi-Parno-Raykova'68, Gennaro-Halevi-Parno-Raykova'69, Gennaro-Halevi-Parno-Raykova'70, Gennaro-Halevi-Parno-Raykova'71, Gennaro-Halevi-Parno-Raykova'72, Gennaro-Halevi-Parno-Raykova'73, Gennaro-Halevi-Parno-Raykova'74, Gennaro-Halevi-Parno-Raykova'75, Gennaro-Halevi-Parno-Raykova'76, Gennaro-Halevi-Parno-Raykova'77, Gennaro-Halevi-Parno-Raykova'78, Gennaro-Halevi-Parno-Raykova'79, Gennaro-Halevi-Parno-Raykova'80, Gennaro-Halevi-Parno-Raykova'81, Gennaro-Halevi-Parno-Raykova'82, Gennaro-Halevi-Parno-Raykova'83, Gennaro-Halevi-Parno-Raykova'84, Gennaro-Halevi-Parno-Raykova'85, Gennaro-Halevi-Parno-Raykova'86, Gennaro-Halevi-Parno-Raykova'87, Gennaro-Halevi-Parno-Raykova'88, Gennaro-Halevi-Parno-Raykova'89, Gennaro-Halevi-Parno-Raykova'90, Gennaro-Halevi-Parno-Raykova'91, Gennaro-Halevi-Parno-Raykova'92, Gennaro-Halevi-Parno-Raykova'93, Gennaro-Halevi-Parno-Raykova'94, Gennaro-Halevi-Parno-Raykova'95, Gennaro-Halevi-Parno-Raykova'96, Gennaro-Halevi-Parno-Raykova'97, Gennaro-Halevi-Parno-Raykova'98, Gennaro-Halevi-Parno-Raykova'99, Gennaro-Halevi-Parno-Raykova'100]

Some works can delegate NP

“Less standard” assumptions

[Canetti-Holmgren-Juels'01, Canetti-Holmgren'02, Canetti-Holmgren'03, Canetti-Holmgren'04, Canetti-Holmgren'05, Canetti-Holmgren'06, Canetti-Holmgren'07, Canetti-Holmgren'08, Canetti-Holmgren'09, Canetti-Holmgren'10, Canetti-Holmgren'11, Canetti-Holmgren'12, Canetti-Holmgren'13, Canetti-Holmgren'14, Canetti-Holmgren'15, Canetti-Holmgren'16, Canetti-Holmgren'17, Canetti-Holmgren'18, Canetti-Holmgren'19, Canetti-Holmgren'20, Canetti-Holmgren'21, Canetti-Holmgren'22, Canetti-Holmgren'23, Canetti-Holmgren'24, Canetti-Holmgren'25, Canetti-Holmgren'26, Canetti-Holmgren'27, Canetti-Holmgren'28, Canetti-Holmgren'29, Canetti-Holmgren'30, Canetti-Holmgren'31, Canetti-Holmgren'32, Canetti-Holmgren'33, Canetti-Holmgren'34, Canetti-Holmgren'35, Canetti-Holmgren'36, Canetti-Holmgren'37, Canetti-Holmgren'38, Canetti-Holmgren'39, Canetti-Holmgren'40, Canetti-Holmgren'41, Canetti-Holmgren'42, Canetti-Holmgren'43, Canetti-Holmgren'44, Canetti-Holmgren'45, Canetti-Holmgren'46, Canetti-Holmgren'47, Canetti-Holmgren'48, Canetti-Holmgren'49, Canetti-Holmgren'50, Canetti-Holmgren'51, Canetti-Holmgren'52, Canetti-Holmgren'53, Canetti-Holmgren'54, Canetti-Holmgren'55, Canetti-Holmgren'56, Canetti-Holmgren'57, Canetti-Holmgren'58, Canetti-Holmgren'59, Canetti-Holmgren'60, Canetti-Holmgren'61, Canetti-Holmgren'62, Canetti-Holmgren'63, Canetti-Holmgren'64, Canetti-Holmgren'65, Canetti-Holmgren'66, Canetti-Holmgren'67, Canetti-Holmgren'68, Canetti-Holmgren'69, Canetti-Holmgren'70, Canetti-Holmgren'71, Canetti-Holmgren'72, Canetti-Holmgren'73, Canetti-Holmgren'74, Canetti-Holmgren'75, Canetti-Holmgren'76, Canetti-Holmgren'77, Canetti-Holmgren'78, Canetti-Holmgren'79, Canetti-Holmgren'80, Canetti-Holmgren'81, Canetti-Holmgren'82, Canetti-Holmgren'83, Canetti-Holmgren'84, Canetti-Holmgren'85, Canetti-Holmgren'86, Canetti-Holmgren'87, Canetti-Holmgren'88, Canetti-Holmgren'89, Canetti-Holmgren'90, Canetti-Holmgren'91, Canetti-Holmgren'92, Canetti-Holmgren'93, Canetti-Holmgren'94, Canetti-Holmgren'95, Canetti-Holmgren'96, Canetti-Holmgren'97, Canetti-Holmgren'98, Canetti-Holmgren'99, Canetti-Holmgren'100]



Delegation for P

Designated Verifier (standard assumptions)

Prior Works

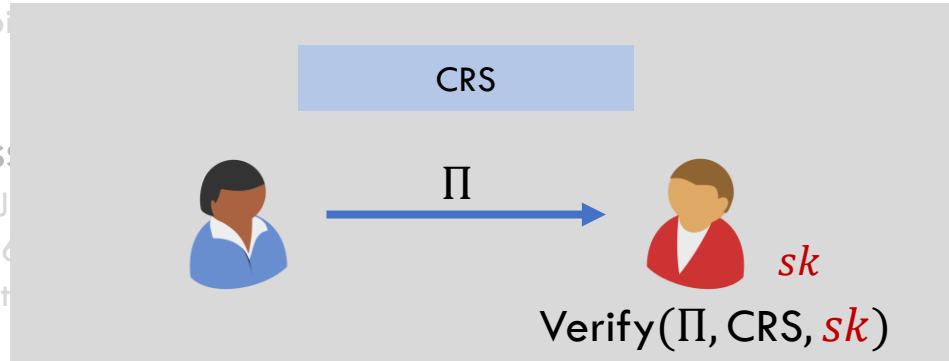
Non-falsifiable assumptions/ Random oracle model

[Micali'94, Groth'10, Lipmaa'12, Damgård-Faust-Hazay'12, Gennaro-Gentry-Parno-Raykova'13, Bitansky-Chiesa-Ishai-Ostrovsky-Paneth'13, Bitansky-Canetti-Chiesa-Tromer'13, Bitansky-Canetti-Chiesa-Goldwasser-Lin-Rubinfeld'14, Canetti-Holmgren-Juels'14, Canetti-Holmgren'16, Rothblum'17, Canetti-Holmgren'18, Canetti-Holmgren'19]

Some works can delegate NP

“Less standard” assumptions

[Canetti-Holmgren-Juels'14, Canetti-Holmgren'16, Rothblum'17, Canetti-Holmgren'18, Canetti-Holmgren'19, Canetti-Holmgren'20, Canetti-Holmgren'21, Canetti-Holmgren'22, Canetti-Holmgren'23, Canetti-Holmgren'24, Canetti-Holmgren'25, Canetti-Holmgren'26, Canetti-Holmgren'27, Canetti-Holmgren'28, Canetti-Holmgren'29, Canetti-Holmgren'30, Canetti-Holmgren'31, Canetti-Holmgren'32, Canetti-Holmgren'33, Canetti-Holmgren'34, Canetti-Holmgren'35, Canetti-Holmgren'36, Canetti-Holmgren'37, Canetti-Holmgren'38, Canetti-Holmgren'39, Canetti-Holmgren'40, Canetti-Holmgren'41, Canetti-Holmgren'42, Canetti-Holmgren'43, Canetti-Holmgren'44, Canetti-Holmgren'45, Canetti-Holmgren'46, Canetti-Holmgren'47, Canetti-Holmgren'48, Canetti-Holmgren'49, Canetti-Holmgren'50, Canetti-Holmgren'51, Canetti-Holmgren'52, Canetti-Holmgren'53, Canetti-Holmgren'54, Canetti-Holmgren'55, Canetti-Holmgren'56, Canetti-Holmgren'57, Canetti-Holmgren'58, Canetti-Holmgren'59, Canetti-Holmgren'60, Canetti-Holmgren'61, Canetti-Holmgren'62, Canetti-Holmgren'63, Canetti-Holmgren'64, Canetti-Holmgren'65, Canetti-Holmgren'66, Canetti-Holmgren'67, Canetti-Holmgren'68, Canetti-Holmgren'69, Canetti-Holmgren'70, Canetti-Holmgren'71, Canetti-Holmgren'72, Canetti-Holmgren'73, Canetti-Holmgren'74, Canetti-Holmgren'75, Canetti-Holmgren'76, Canetti-Holmgren'77, Canetti-Holmgren'78, Canetti-Holmgren'79, Canetti-Holmgren'80, Canetti-Holmgren'81, Canetti-Holmgren'82, Canetti-Holmgren'83, Canetti-Holmgren'84, Canetti-Holmgren'85, Canetti-Holmgren'86, Canetti-Holmgren'87, Canetti-Holmgren'88, Canetti-Holmgren'89, Canetti-Holmgren'90, Canetti-Holmgren'91, Canetti-Holmgren'92, Canetti-Holmgren'93, Canetti-Holmgren'94, Canetti-Holmgren'95, Canetti-Holmgren'96, Canetti-Holmgren'97, Canetti-Holmgren'98, Canetti-Holmgren'99, Canetti-Holmgren'100]



Delegation for P

Designated Verifier (standard assumptions)

[Kalai-Raz-Rothblum'13, Kalai-Raz-Rothblum'14, Kalai-Paneth'16, Brakerski-Holmgren-Kalai'17, Badrinarayanan-Kalai-Khurana-Sahai-Wichs'18, Holmgren-Rothblum'18, Brakerski-Kalai'20]

Delegation for P

Do there exists **SNARGs** for **P** based on
standard assumptions?

Previously best known: [Jawale-Kalai-Khurana-Zhang'21] for **depth
bounded computation** based on **sub-exponential hardness of LWE**.

Builds on [Canetti-Chen-Holmgren-Lombardi-Rothblum-Rothblum-Wichs'19]

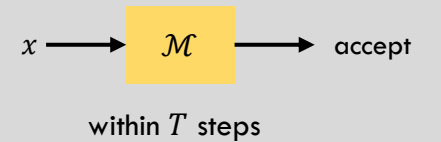
Our Result

Theorem

Assuming **LWE** there exists a SNARG for P where

$$|\text{CRS}|, |\Pi|, |\text{decommit}| = \text{polylog}(T)$$

LWE – Learning with Errors



Our Result

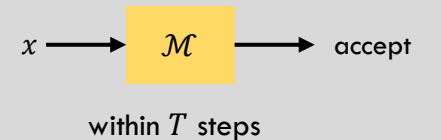
Theorem

Assuming **LWE** there exists a SNARG for P where

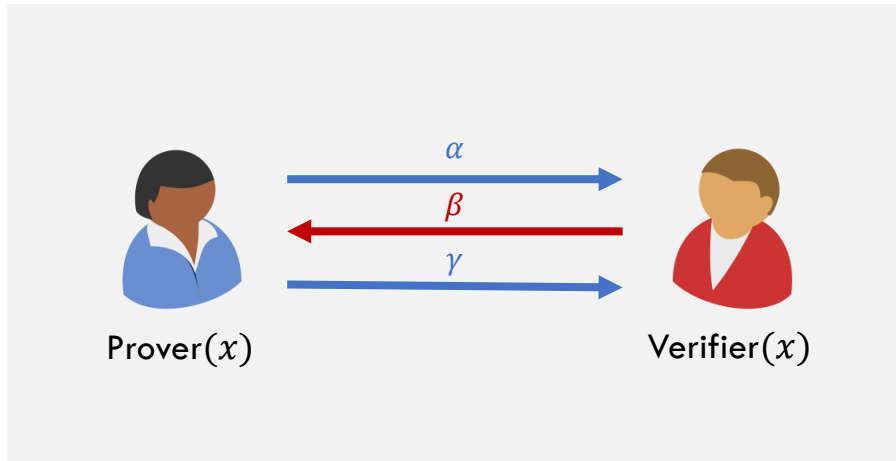
$$|\text{CRS}|, |\Pi|, |\text{👤}| = \text{polylog}(T)$$

LWE – Learning with Errors

Tool: Fiat-Shamir (FS) Methodology

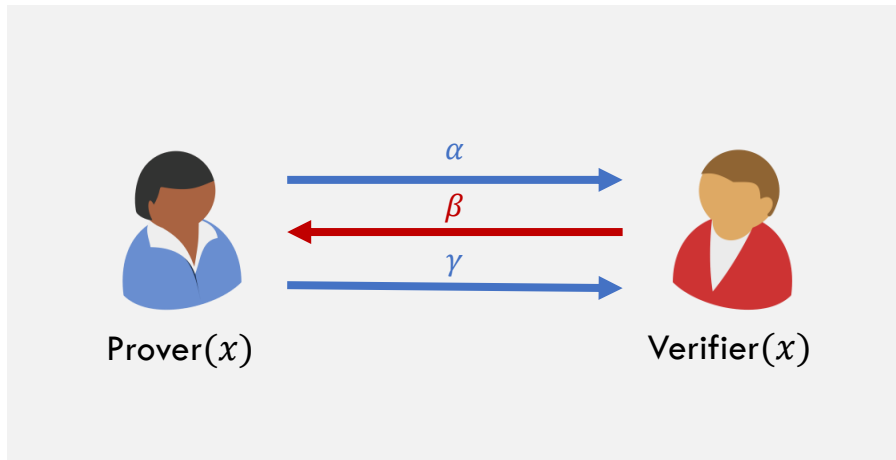


Fiat-Shamir (FS) Methodology



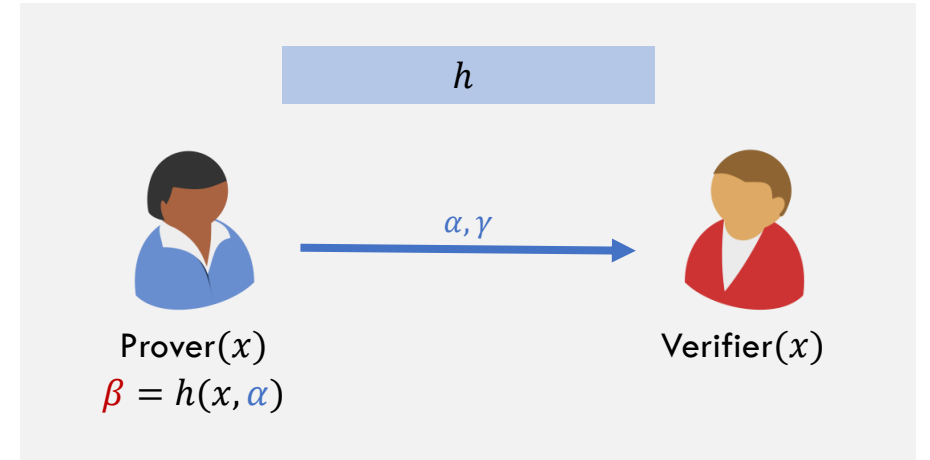
β is a random string

Fiat-Shamir (FS) Methodology

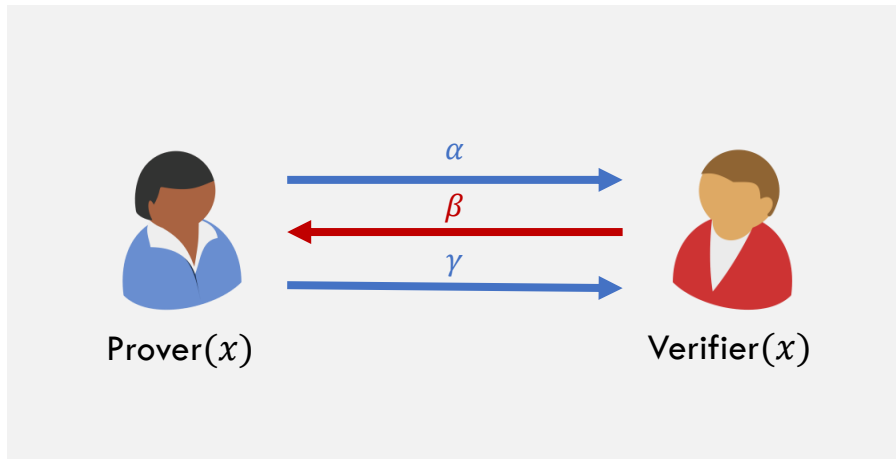


β is a random string

→
[Fiat-Shamir'86]

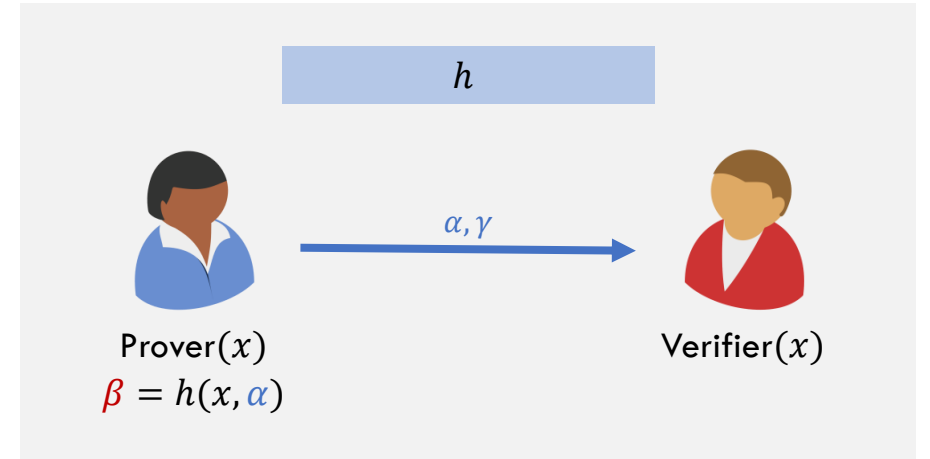


Fiat-Shamir (FS) Methodology



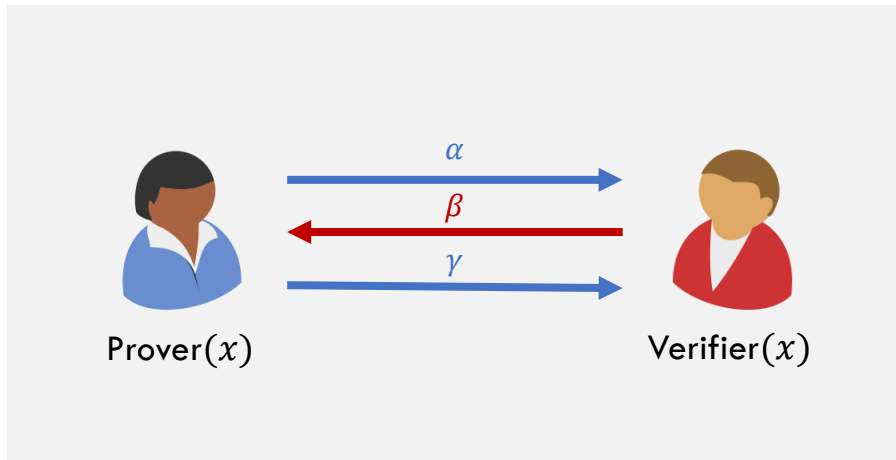
β is a random string

[Fiat-Shamir'86]



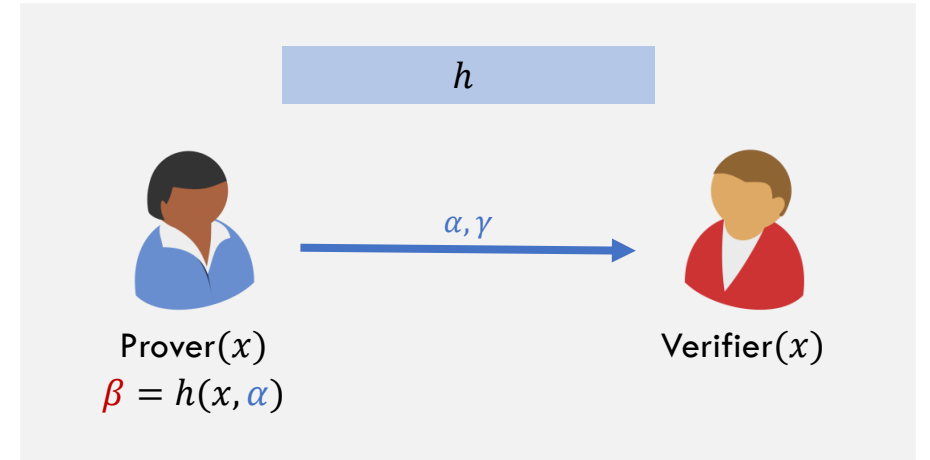
$$\forall x \notin \mathcal{L}$$
$$\text{BAD}_{x,\alpha} = \{\beta \mid \exists \gamma \text{ s.t. Verifier accepts } (\alpha, \beta, \gamma)\}$$

Fiat-Shamir (FS) Methodology



β is a random string

[Fiat-Shamir'86]

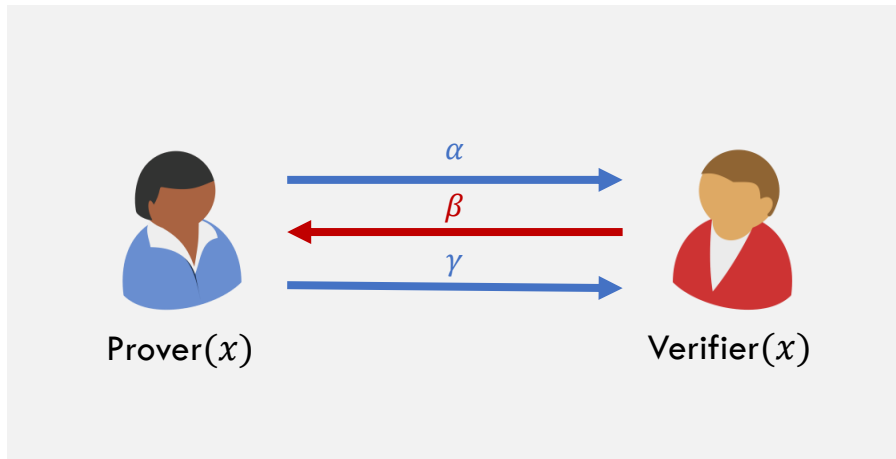


$$\forall x \notin \mathcal{L}$$
$$\text{BAD}_{x,\alpha} = \{\beta \mid \exists \gamma \text{ s.t. Verifier accepts } (\alpha, \beta, \gamma)\}$$

If $x \notin \mathcal{L}$, no PPT  can find α such that

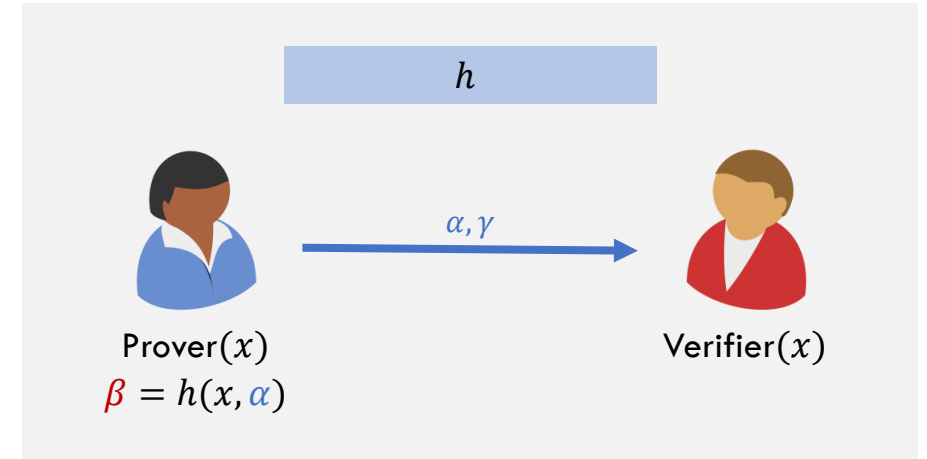
$$h(x, \alpha) \in \text{BAD}_{x,\alpha}$$

Correlation Intractability [Canetti-Goldreich-Halevi'98]



β is a random string

[Fiat-Shamir'86]



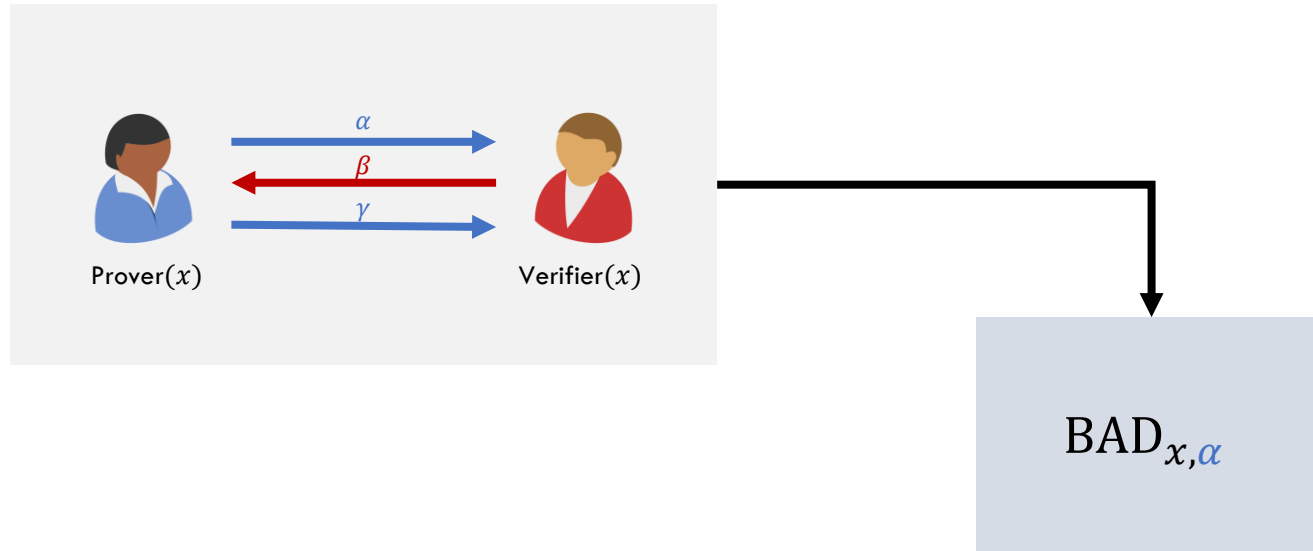
$$\forall x \notin \mathcal{L}$$
$$\text{BAD}_{x,\alpha} = \{\beta \mid \exists \gamma \text{ s.t. Verifier accepts } (\alpha, \beta, \gamma)\}$$

If $x \notin \mathcal{L}$, no PPT  can find α such that

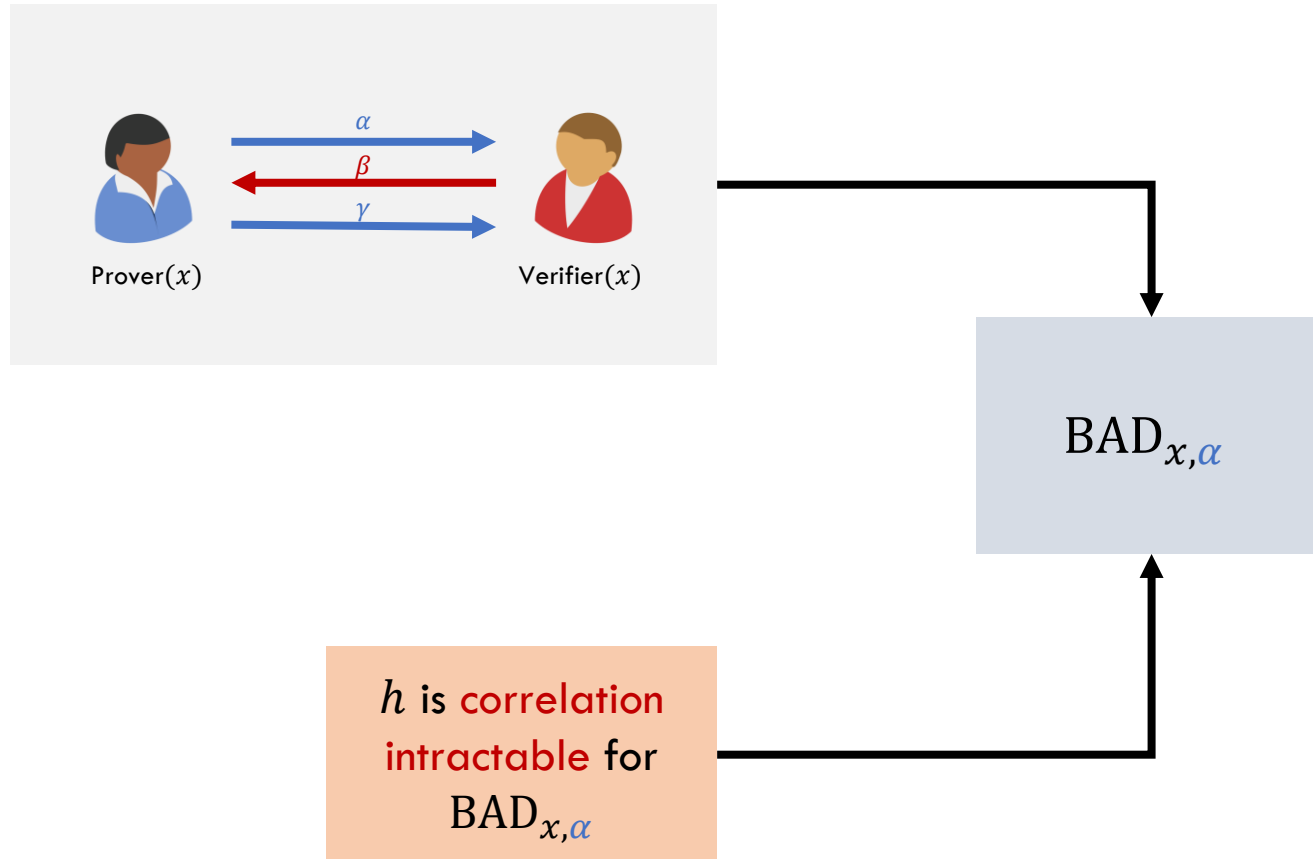
$$h(x, \alpha) \in \text{BAD}_{x,\alpha}$$

h is **correlation intractable (CI)** for $\text{BAD}_{x,\alpha}$

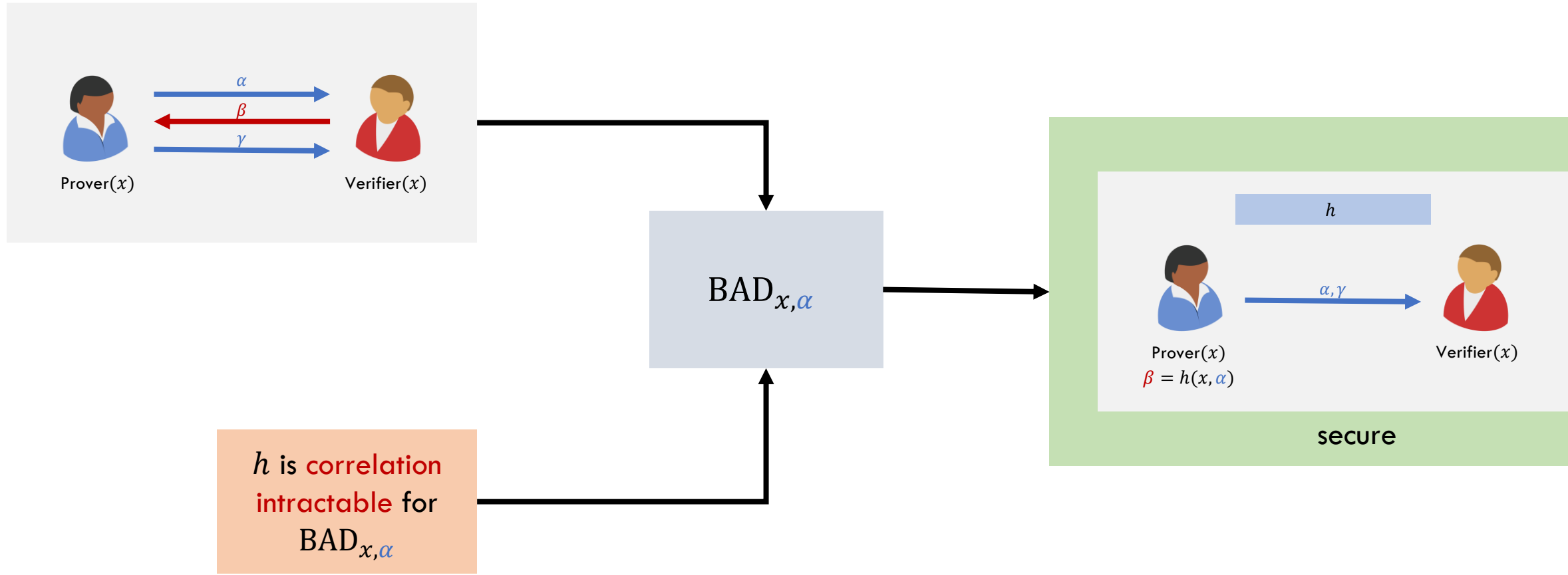
Instantiating the FS Transform



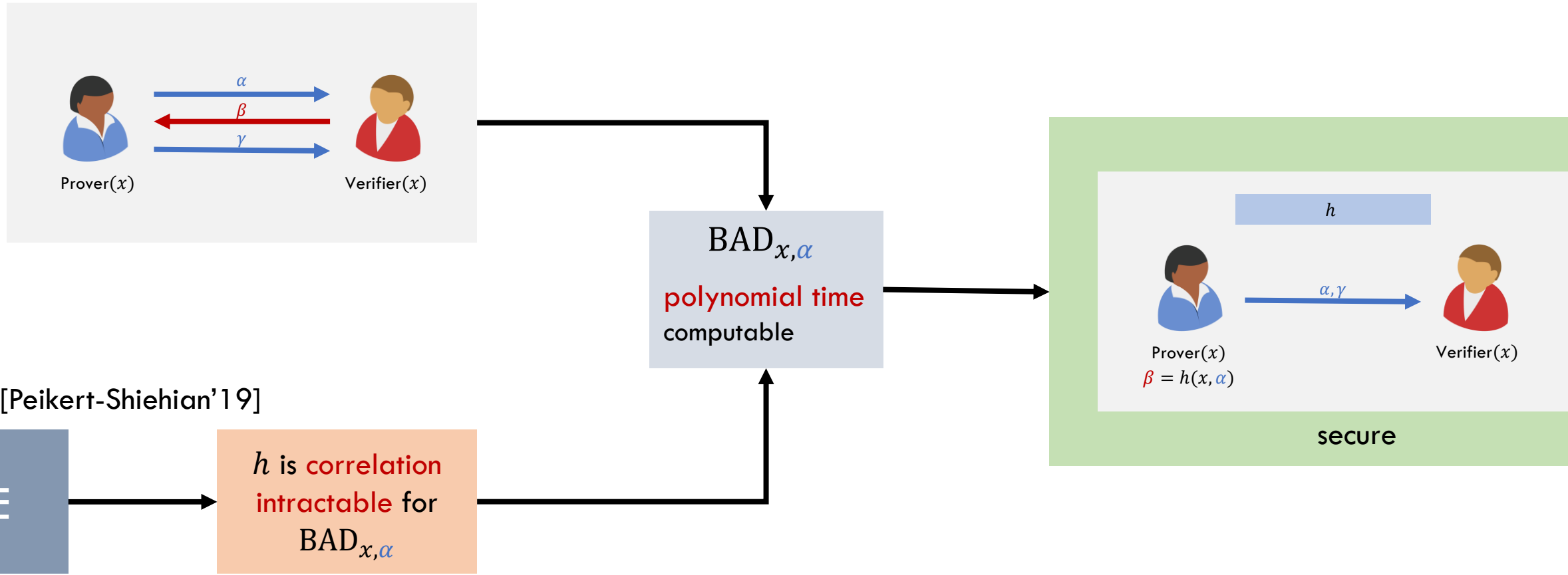
Instantiating the FS Transform



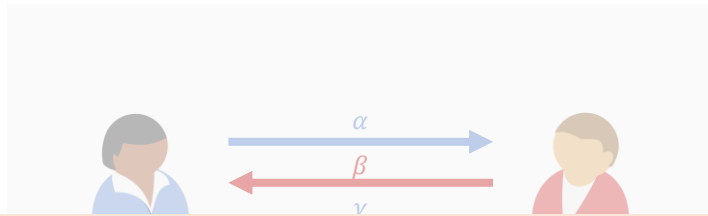
Instantiating the FS Transform



Instantiating the FS Transform



Instantiating the FS Transform



FS methodology is secure for certain protocols under a variety of assumptions (via **correlation intractable hash functions**)

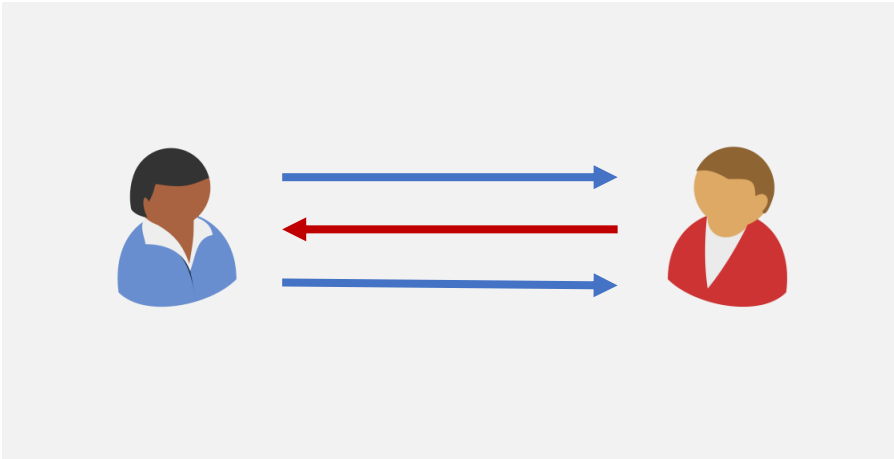
[Kalai-Rothblum-Rothblum'17, Canetti-Chen-Reyzin-Rothblum'18, Holmgren-Lombardi'18, Canetti-Chen-Holmgren-Lombardi-Rothblum-Rothblum-Wichs'19, Peikert-Sheihian'19, Brakerski-Koppula-Mour'20, Couteau-Katsumata-Ursu'20, Jain-Jin'21, Jawale-Kalai-Khurana-Zhang'21, Holmgren-Lombardi-Rothblum'21]

LWE

h is **correlation intractable** for $\text{BAD}_{x,\alpha}$

secure

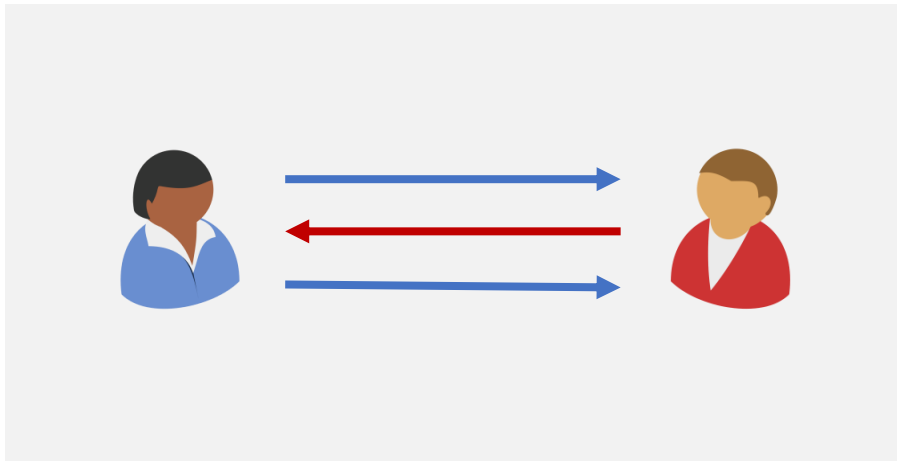
Fiat-Shamir (FS) Methodology



[Kilian'92]

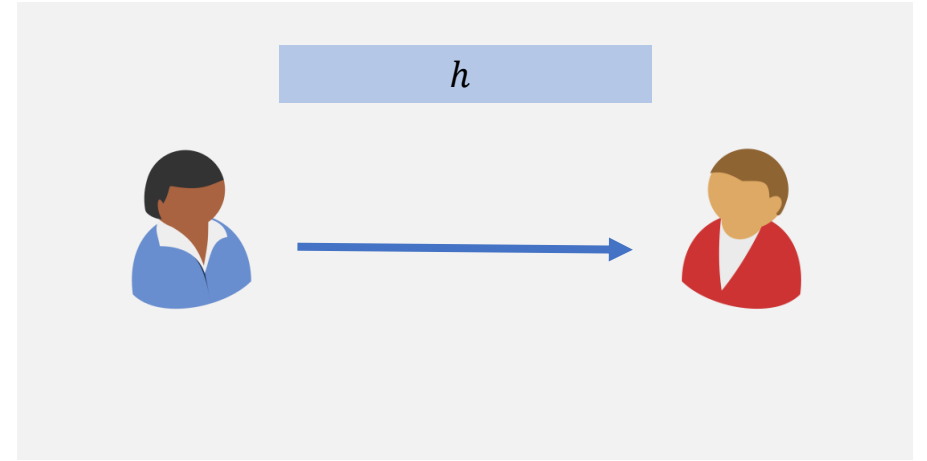
Succinct interactive arguments for all polynomial time computation.

Fiat-Shamir (FS) Methodology



[Kilian'92]

Instantiate with
CIH

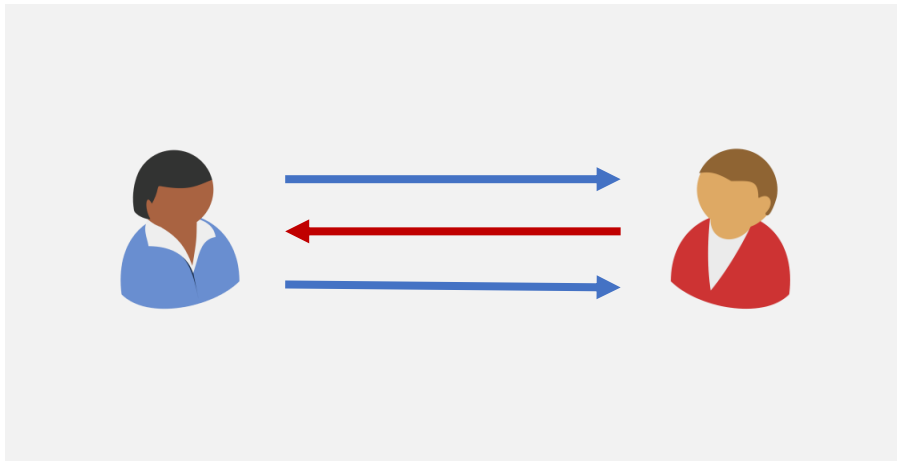


Succinct interactive arguments for all
polynomial time computation.

[Bartusek-Bronfman-Holmgren-Ma-Rothblum'19]

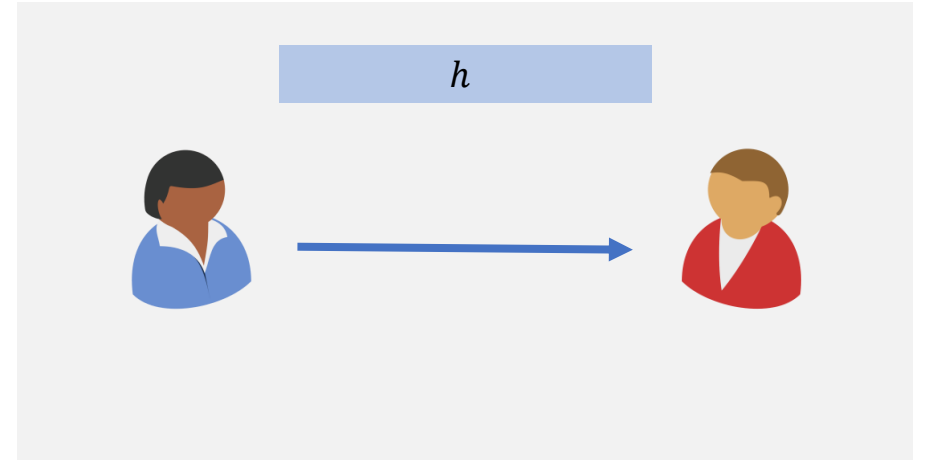
Instantiating hash function for Fiat-Shamir transformation of Kilian's
protocol is hard.

Fiat-Shamir (FS) Methodology



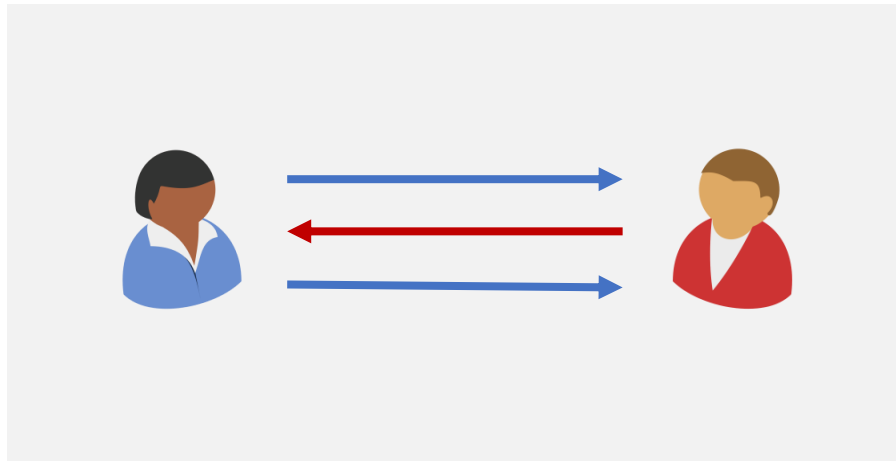
[Kilian'92]

Instantiate with
CIH



Succinct interactive **arguments** for all
polynomial time computation.

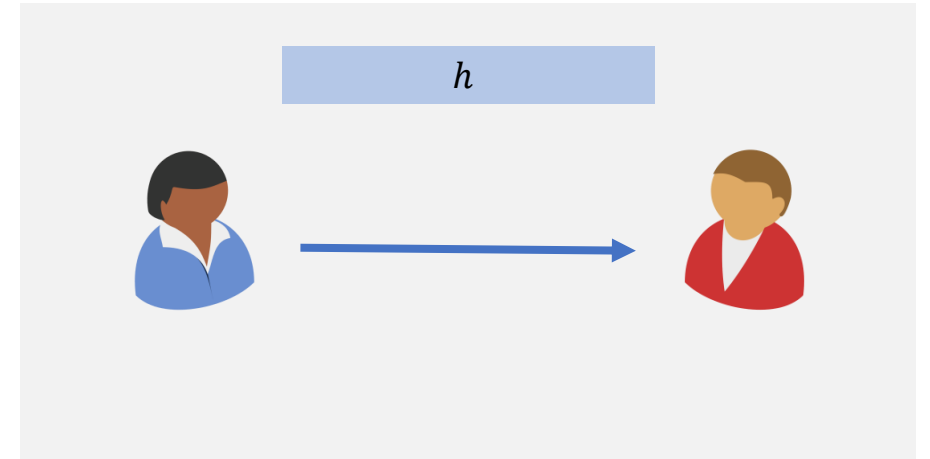
Fiat-Shamir (FS) Methodology



[Kilian'92]

Succinct interactive **arguments** for all polynomial time computation.

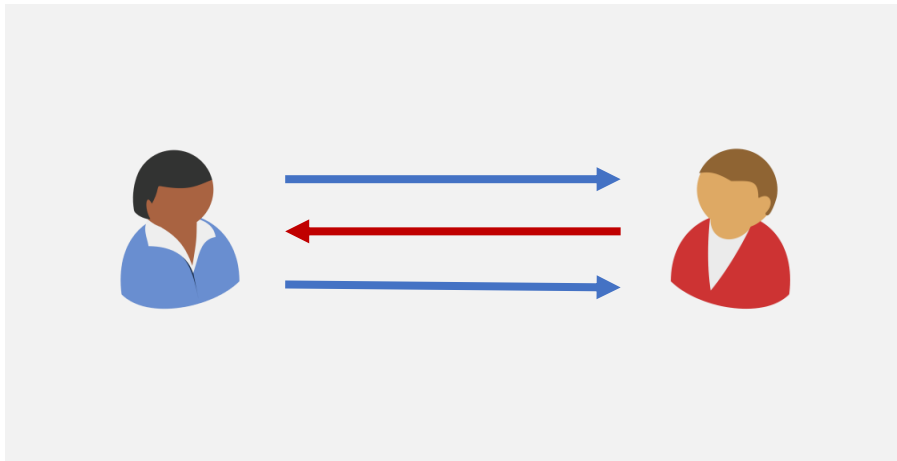
Instantiate with
CIH



Known instantiations of CI Hash for Fiat-Shamir transform are for **proofs**.

[Canetti-Sarkar-Wang'20] instantiate Fiat-Shamir transform for specific Sigma protocol that is an argument.

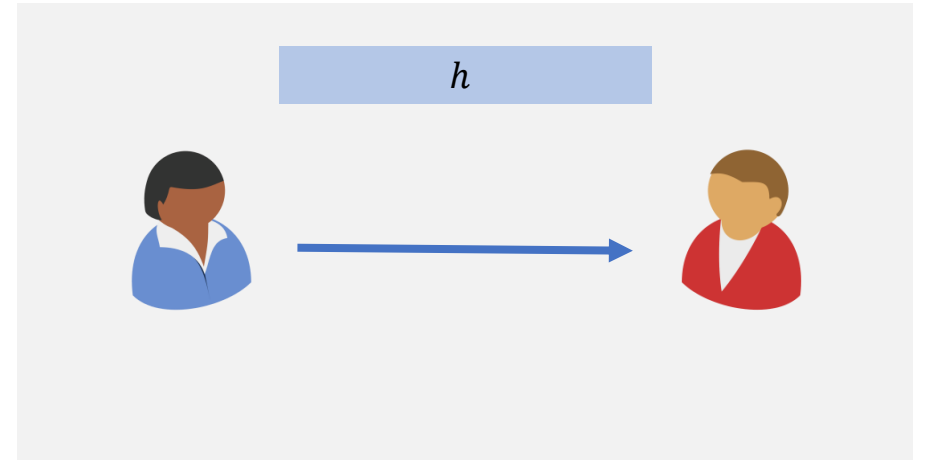
Fiat-Shamir (FS) Methodology



[Goldwasser-Kalai-Rothblum'08]

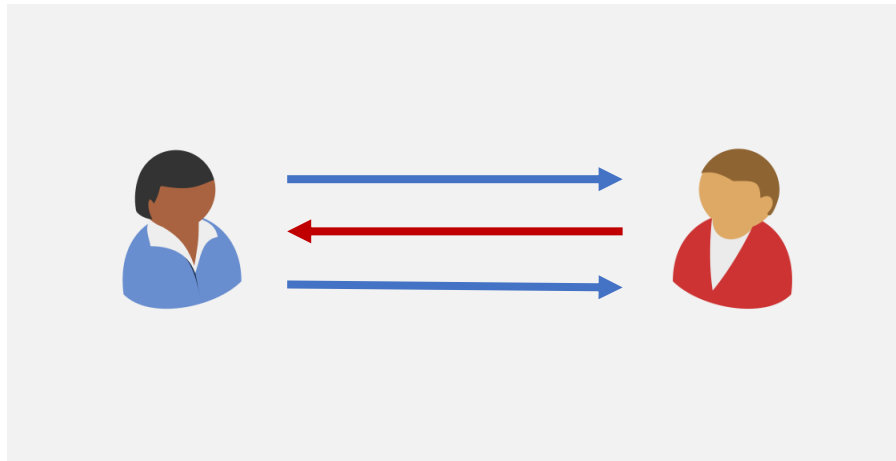
Succinct interactive **proof** for **depth**
bounded computation.

→
Instantiate with
CIH



[Canetti-Chen-Holmgren-Lombardi-Rothblum-Rothblum-Wichs'19,
Jawale-Kalai-Khurana-Zhang'21]

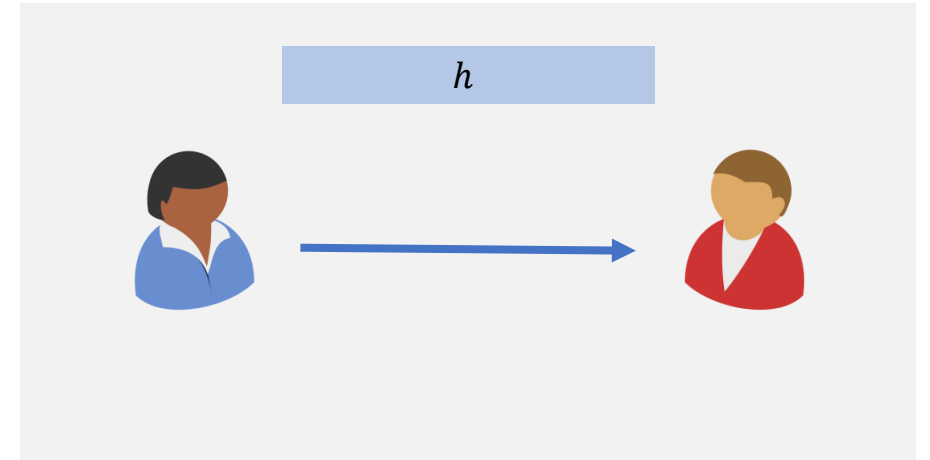
Fiat-Shamir (FS) Methodology



[Goldwasser-Kalai-Rothblum'08]

Succinct interactive proof for depth bounded computation.

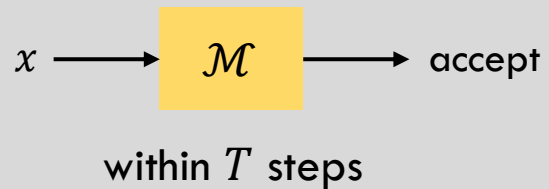
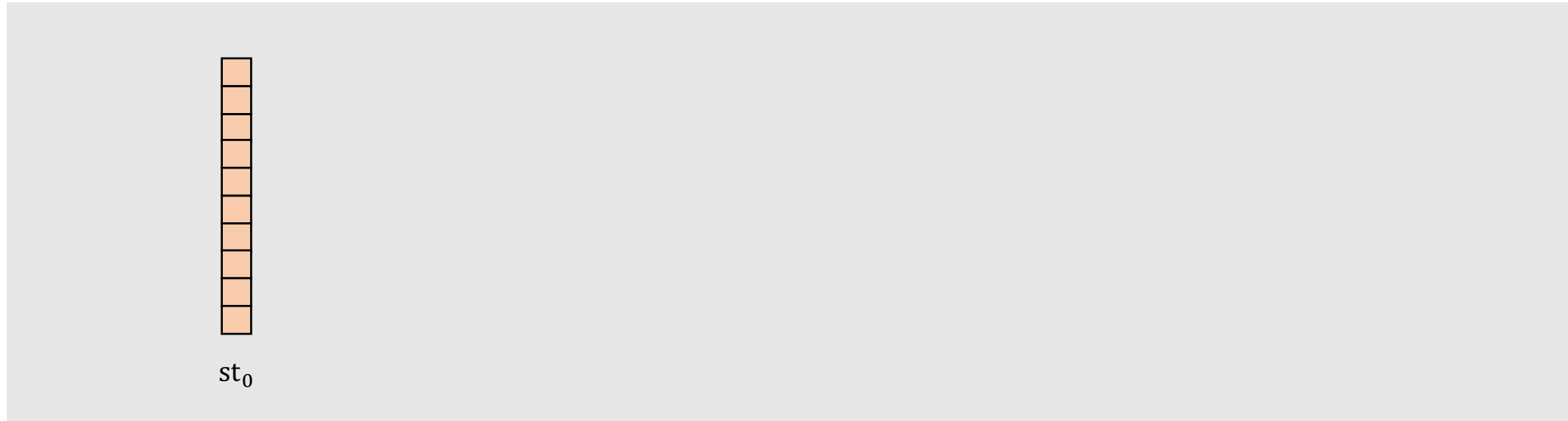
Instantiate with
CIH



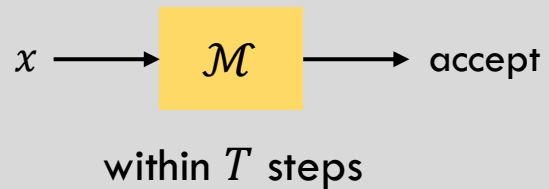
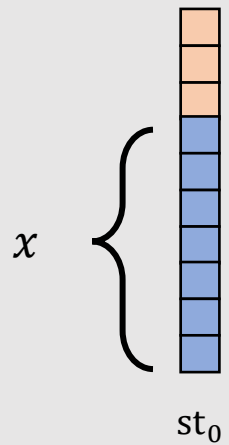
[Canetti-Chen-Holmgren-Lombardi-Rothblum-Rothblum-Wichs'19,
Jawale-Kalai-Khurana-Zhang'21]

Interactive proofs for all polynomial time computation unlikely to exist.

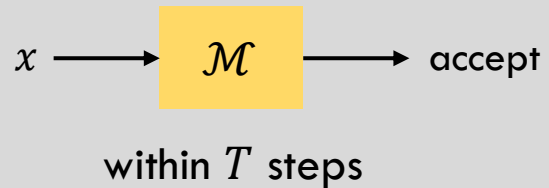
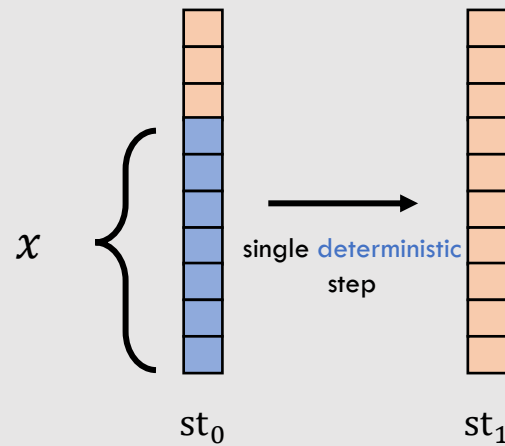
Delegation via **Batching** [Reingold-Rothblum-Rothblum'16]



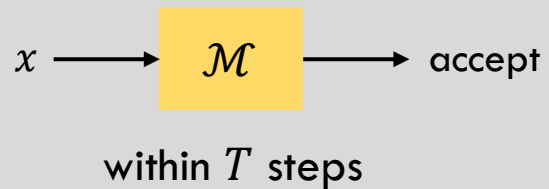
Delegation via **Batching** [Reingold-Rothblum-Rothblum'16]



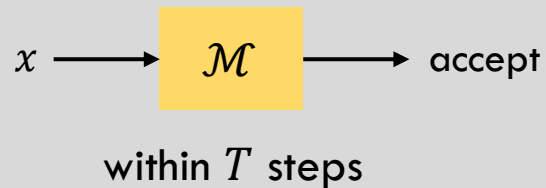
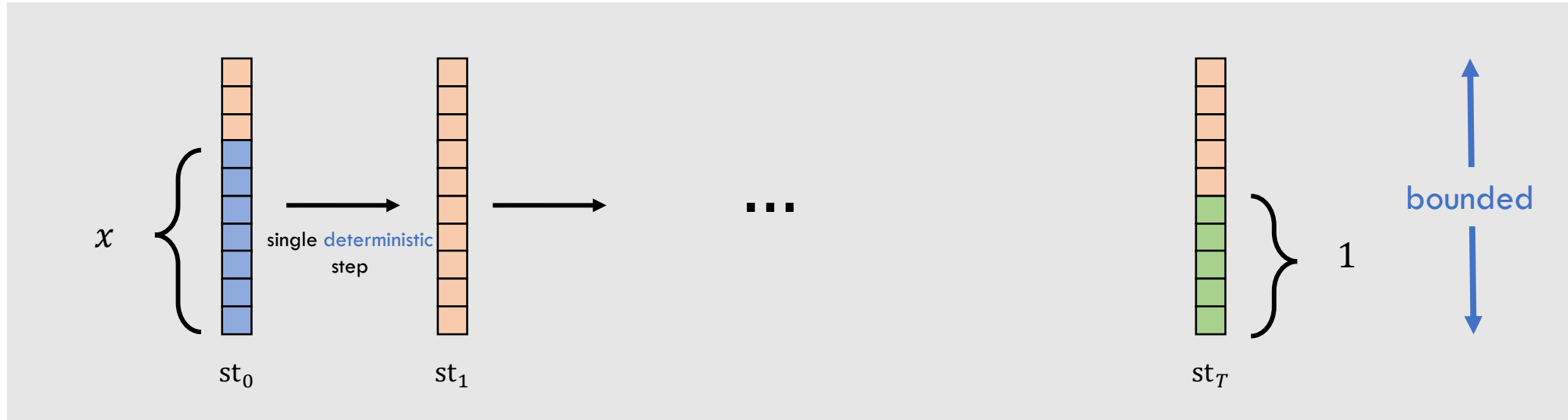
Delegation via **Batching** [Reingold-Rothblum-Rothblum'16]



Delegation via **Batching** [Reingold-Rothblum-Rothblum'16]

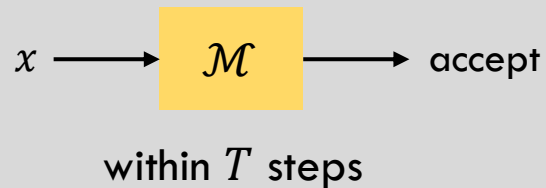
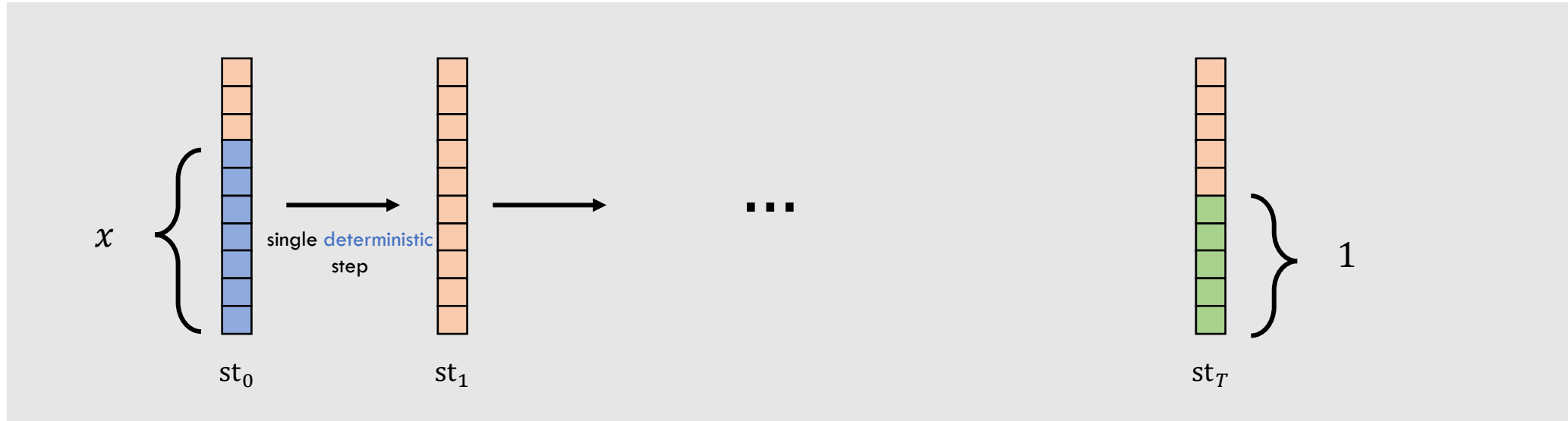


Delegation via **Batching** [Reingold-Rothblum-Rothblum'16]



This talk: **Bounded** space computation

Delegation via **Batching** [Reingold-Rothblum-Rothblum'16]



Prove for every $i \in [0, \dots, T - 1]$
 $st_i \rightarrow st_{i+1}$
is the correct transition.

SNARGs for Batch NP

CRS



C, x_1, \dots, x_k

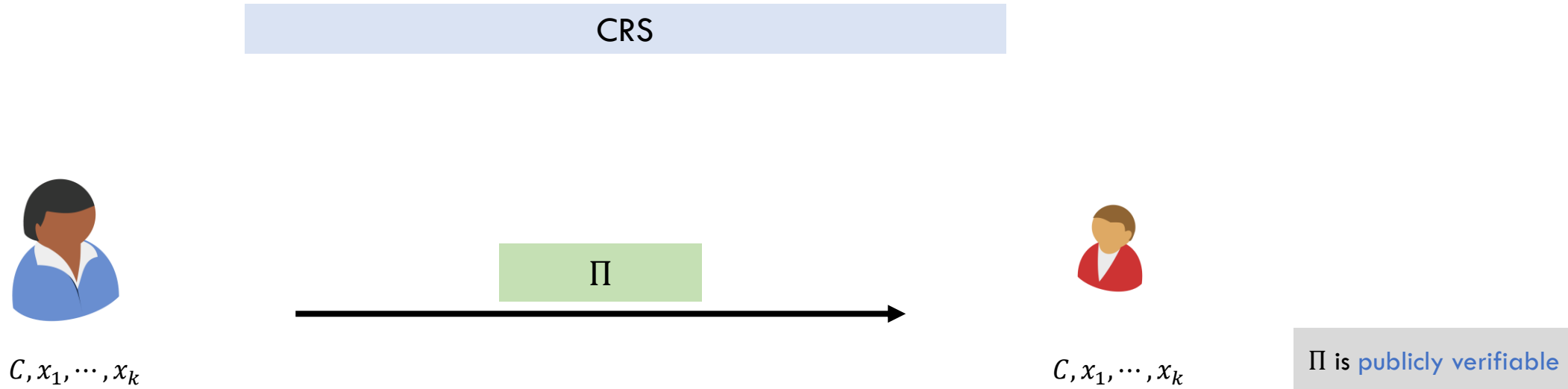


C, x_1, \dots, x_k

$\text{SAT} = \{(C, x) \mid \exists w \text{ s.t. } C(x, w) = 1\}$

$\forall i \in [k], (C, x_i) \in \text{SAT}$

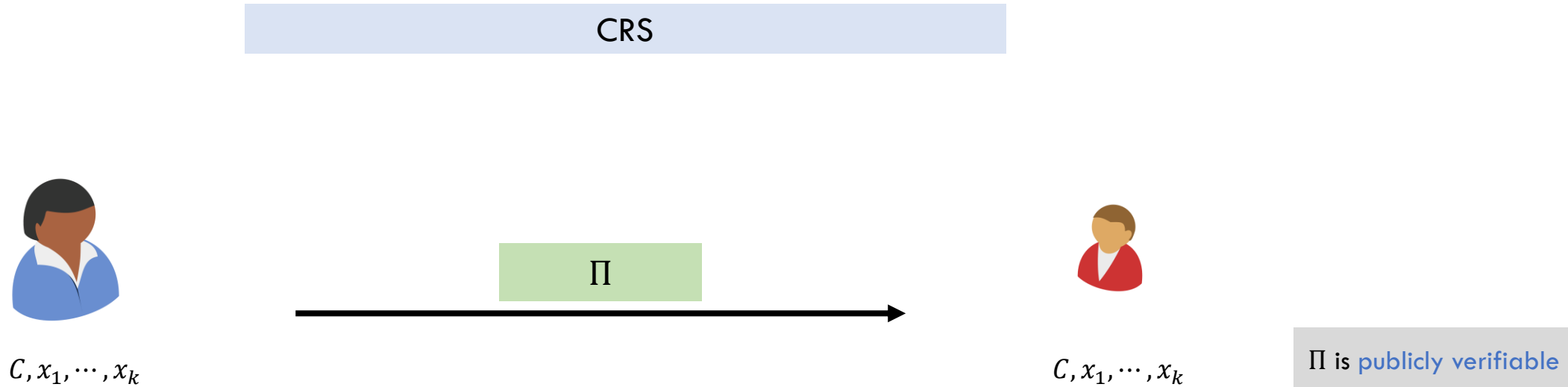
SNARGs for Batch NP



$$\text{SAT} = \{(C, x) \mid \exists w \text{ s.t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

SNARGs for Batch NP



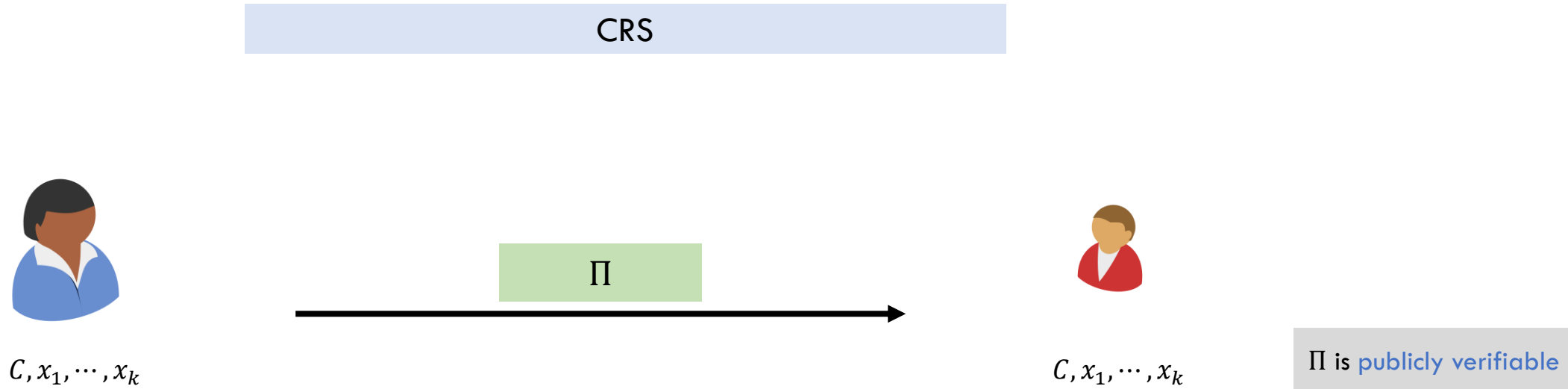
$$\text{SAT} = \{(C, x) \mid \exists w \text{ s.t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

No PPT  can produce **accepting** Π if

$$\exists i^* \in [k], (C, x_{i^*}) \not\in \text{SAT}$$

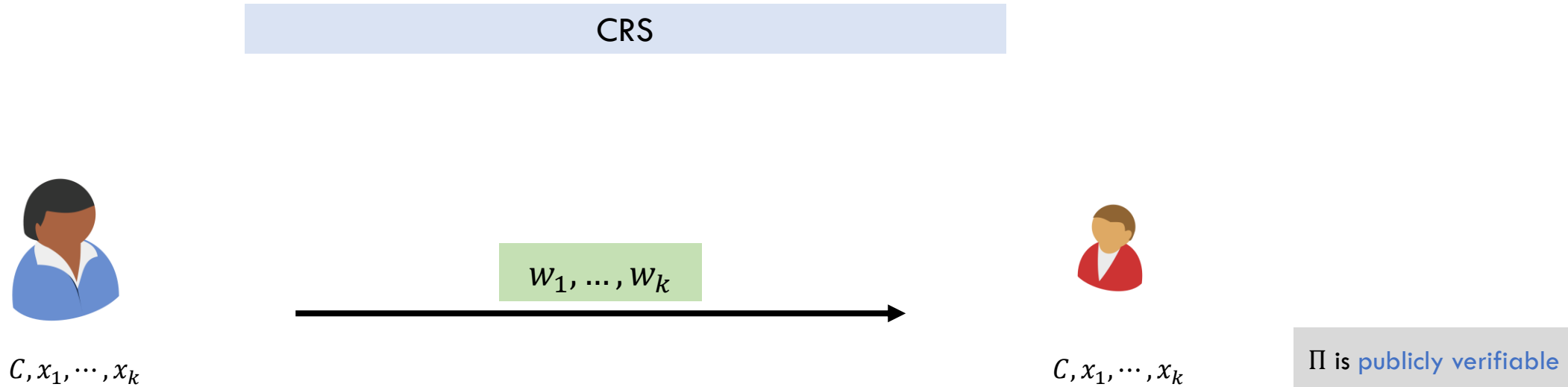
SNARGs for Batch NP



$$\text{SAT} = \{(C, x) \mid \exists w \text{ s.t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

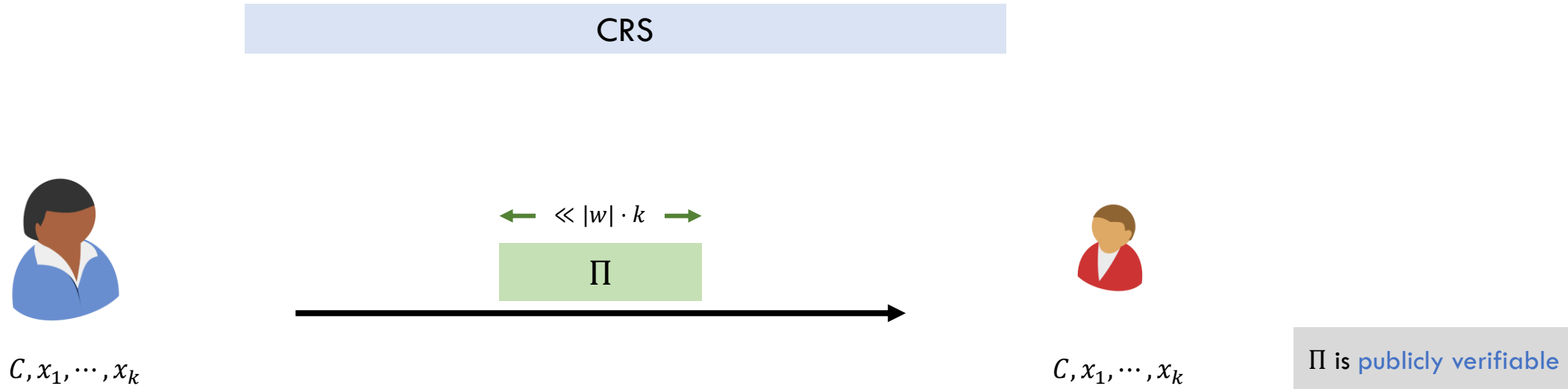
SNARGs for Batch NP



$$\text{SAT} = \{(C, x) \mid \exists w \text{ s.t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

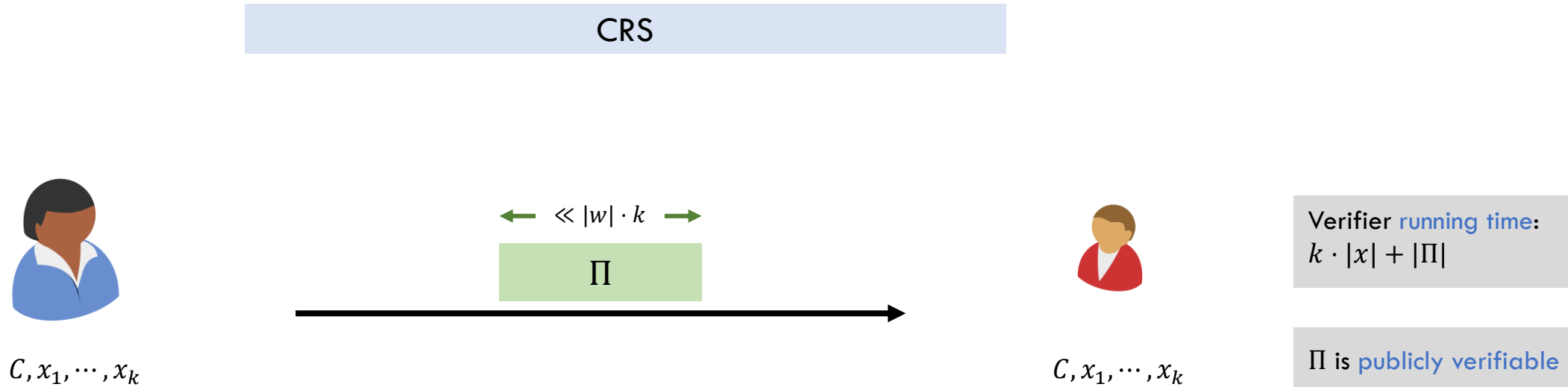
SNARGs for Batch NP



$$\text{SAT} = \{(C, x) \mid \exists w \text{ s.t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

SNARGs for Batch NP



$$\text{SAT} = \{(C, x) \mid \exists w \text{ s.t. } C(x, w) = 1\}$$

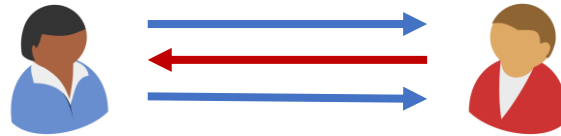
$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Prior Works

Prior Works

Interactive batch proofs for UP

[Reingold-Rothblum-Rothblum'16, Reingold-Rothblum-Rothblum'18, Rothblum-Rothblum'20]



UP – each statement has a **unique witness**.

Prior Works

Interactive batch proofs for UP

[Reingold-Rothblum-Rothblum'16, Reingold-Rothblum-Rothblum'18, Rothblum-Rothblum'20]

Succinct Non-interactive Arguments (SNARGs) for NP

[Micali'94, Damgård-Faust-Hazay'12, Bitansky-Canetti-Chiesa-Tromer'13, Bitansky-Canetti-Chiesa-Goldwasser-Lin-Rubinfeld-Tromer'16]

SNARGs

$$|\Pi| \ll |w|$$

$$\text{SAT} = \{(C, x) \mid \exists w \text{ s.t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Prior Works

Interactive batch proofs for UP

[Reingold-Rothblum-Rothblum'16, Reingold-Rothblum-Rothblum'18, Rothblum-Rothblum'20]

Succinct Non-interactive Arguments (SNARGs) for NP

[Micali'94, Damgård-Faust-Hazay'12, Bitansky-Canetti-Chiesa-Tromer'13, Bitansky-Canetti-Chiesa-Goldwasser-Lin-Rubinfeld-Tromer'16]

SNARGs

$$|\Pi| \ll |w|$$

$$\text{SAT}^{\otimes k} = \{(C, x_1, \dots, x_k) \mid \forall i \in [k], (C, x_i) \in \text{SAT}\}$$

$$\text{SAT} = \{(C, x) \mid \exists w \text{ s.t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Prior Works

Interactive batch proofs for UP

[Reingold-Rothblum-Rothblum'16, Reingold-Rothblum-Rothblum'18, Rothblum-Rothblum'20]

SNARGs for NP from **Non-falsifiable assumptions**/ **Random oracle model**

[Micali'94, Damgård-Faust-Hazay'12, Bitansky-Canetti-Chiesa-Tromer'13, Bitansky-Canetti-Chiesa-Goldwasser-Lin-Rubinfeld-Tromer'16]

SNARGs

$$|\Pi| \ll |w|$$

$$\text{SAT}^{\otimes k} = \{(C, x_1, \dots, x_k) \mid \forall i \in [k], (C, x_i) \in \text{SAT}\}$$

$$\text{SAT} = \{(C, x) \mid \exists w \text{ s.t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Prior Works

Interactive batch proofs for UP

[Reingold-Rothblum-Rothblum'16, Reingold-Rothblum-Rothblum'18, Rothblum-Rothblum'20]

SNARGs for NP from Non-falsifiable assumptions/ Random oracle model

[Micali'94, Damgård-Faust-Hazay'12, Bitansky-Canetti-Chiesa-Tromer'13, Bitansky-Canetti-Chiesa-Goldwasser-Lin-Rubinfeld-Tromer'16]

Designated Verifier SNARGs for Batch NP

[Brakerski-Holmgren-Kalai'17, Brakerski-Kalai'20]

Prior Works

Interactive batch proofs for UP

[Reingold-Rothblum-Rothblum'16, Reingold-Rothblum-Rothblum'18, Rothblum-Rothblum'20]

SNARGs for NP from Non-falsifiable assumptions/ Random oracle model

[Micali'94, Damgård-Faust-Hazay'12, Bitansky-Canetti-Chiesa-Tromer'13, Bitansky-Canetti-Chiesa-Goldwasser-Lin-Rubinstein-Tromer'16]

Designated Verifier from **standard assumptions**

[Brakerski-Holmgren-Kalai'17, Brakerski-Kalai'20]

Prior Works

Interactive batch proofs for UP

[Reingold-Rothblum-Rothblum'16, Reingold-Rothblum-Rothblum'18, Rothblum-Rothblum'20]

SNARGs for NP from Non-falsifiable assumptions/ Random oracle model

[Micali'94, Damgård-Faust-Hazay'12, Bitansky-Canetti-Chiesa-Tromer'13, Bitansky-Canetti-Chiesa-Goldwasser-Lin-Rubinfeld-Tromer'16]

Designated Verifier from standard assumptions

[Brakerski-Holmgren-Kalai'17, Brakerski-Kalai'20]

SNARGs for Batch NP from new **non-standard assumption**

[Kalai-Paneth-Yang'19]

Falsifiable assumption
on groups with bilinear
maps.

Do there exists SNARGs for Batch NP based on standard assumptions?

Our Result

Theorem

There exists SNARGs for Batch NP

Assuming QR + sub-exp DDH

$$|\Pi| = \tilde{O}(|C| + \sqrt{k|C|})$$

[C-Jain-Jin'21a]

QR – Quadratic residuosity, LWE – Learning with Error, DDH – Decisional Diffie-Hellman

$$\text{SAT} = \{(C, x) \mid \exists w \text{ s.t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Our Result

Theorem

There exists SNARGs for Batch NP

Assuming **LWE**

$$|\Pi| = \text{poly}(\log k, |C|)$$

[C-Jain-Jin'21b]

Assuming **QR + sub-exp DDH**

$$|\Pi| = \tilde{O}(|C| + \sqrt{k|C|})$$

[C-Jain-Jin'21a]

QR – **Q**uadratic **r**esiduosity, LWE – **L**earning **w**ith **E**rror, DDH – **D**ecisional **D**iffie-**H**ellman

$$\text{SAT} = \{(C, x) \mid \exists w \text{ s.t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Results Overview

QR + sub-exp
DDH



SNARGs for Batch NP

$$|\Pi| = \tilde{O}(|C| + \sqrt{k|C|})$$

LWE



SNARGs for Batch NP

$$|\Pi| = \text{poly}(\log k, |C|)$$

Results Overview

QR + sub-exp
DDH

SNARGs for Batch NP

$$|\Pi| = \tilde{O}(|C| + \sqrt{k|C|})$$

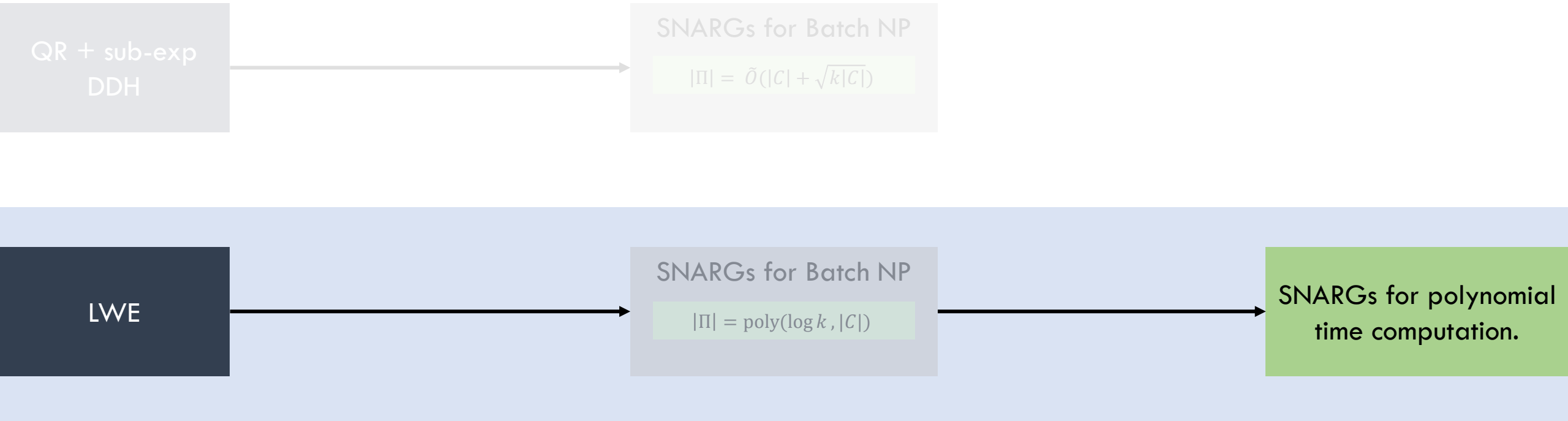
LWE

SNARGs for Batch NP

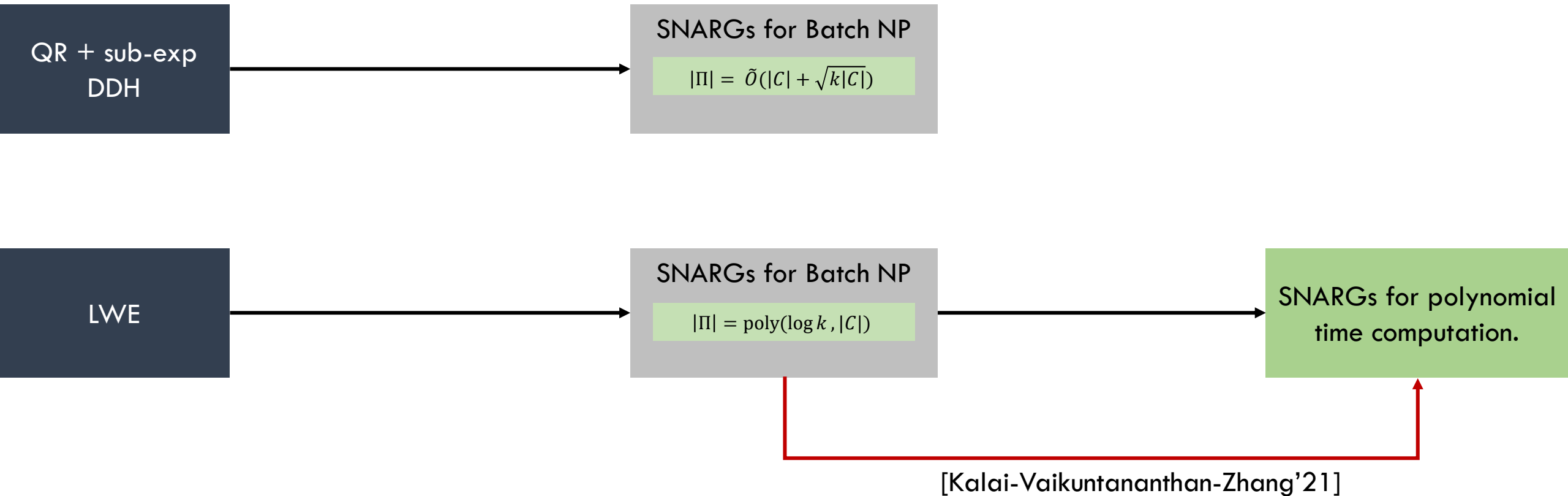
$$|\Pi| = \text{poly}(\log k, |C|)$$

SNARGs for polynomial
time computation.

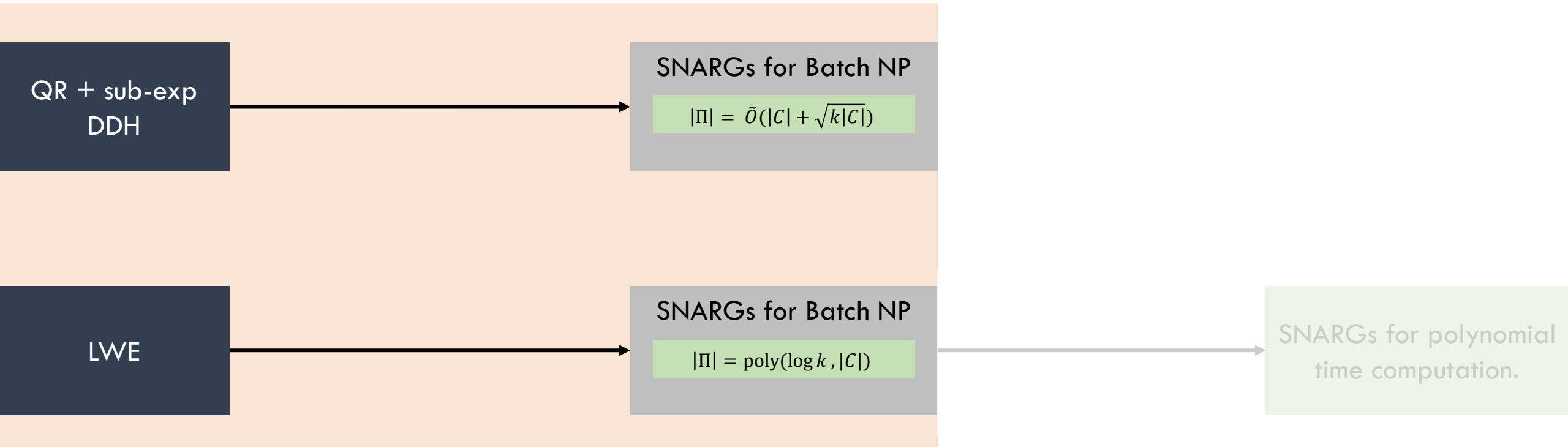
Results Overview



Results Overview

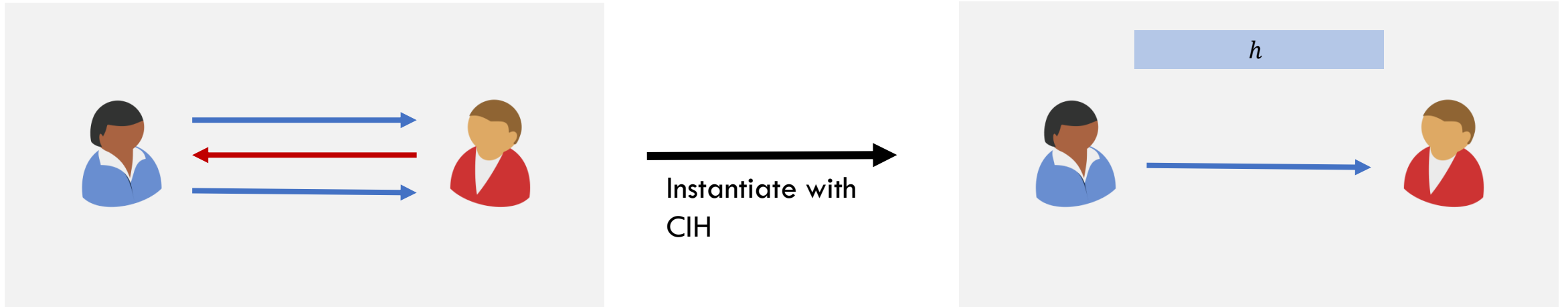


Results Overview



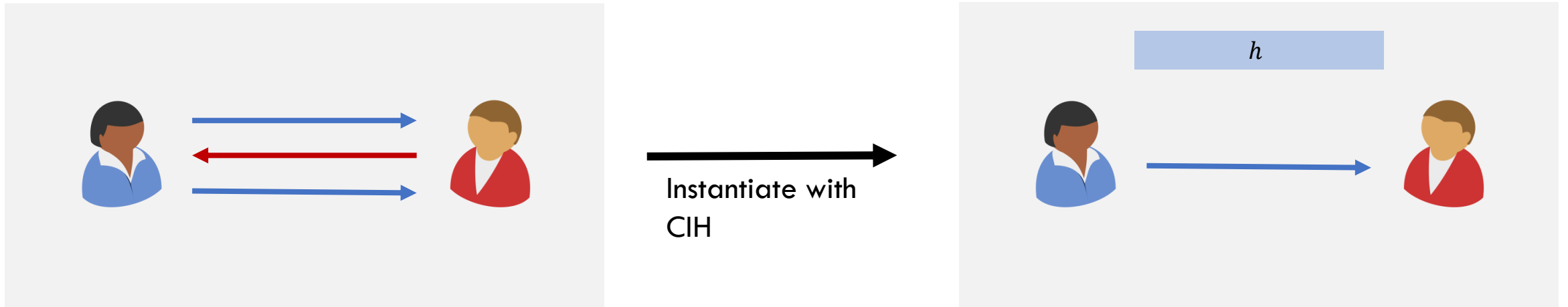
SNARGs for Batch NP

Fiat-Shamir (FS) Methodology



No known interactive proof for batch NP

Fiat-Shamir (FS) Methodology



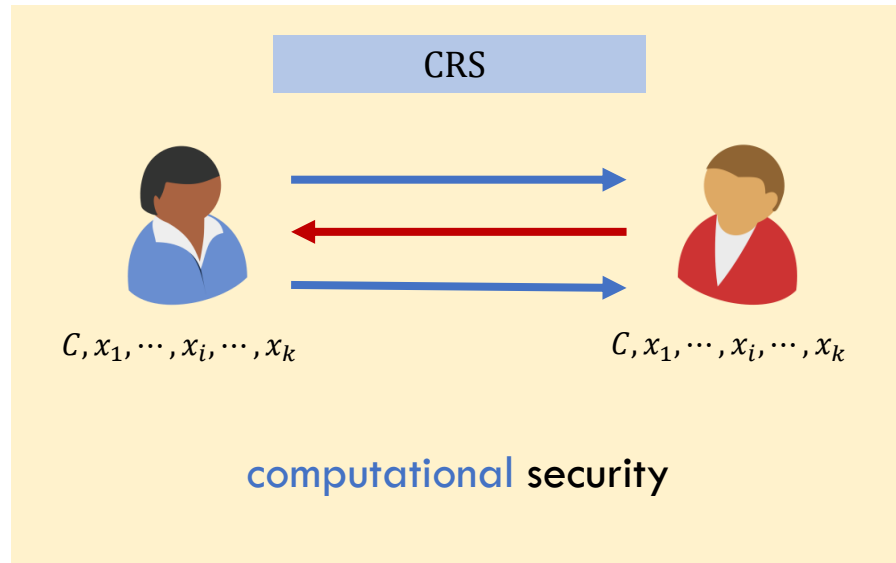
A Different Starting Point for Fiat-Shamir Methodology

No known interactive proof for batch NP

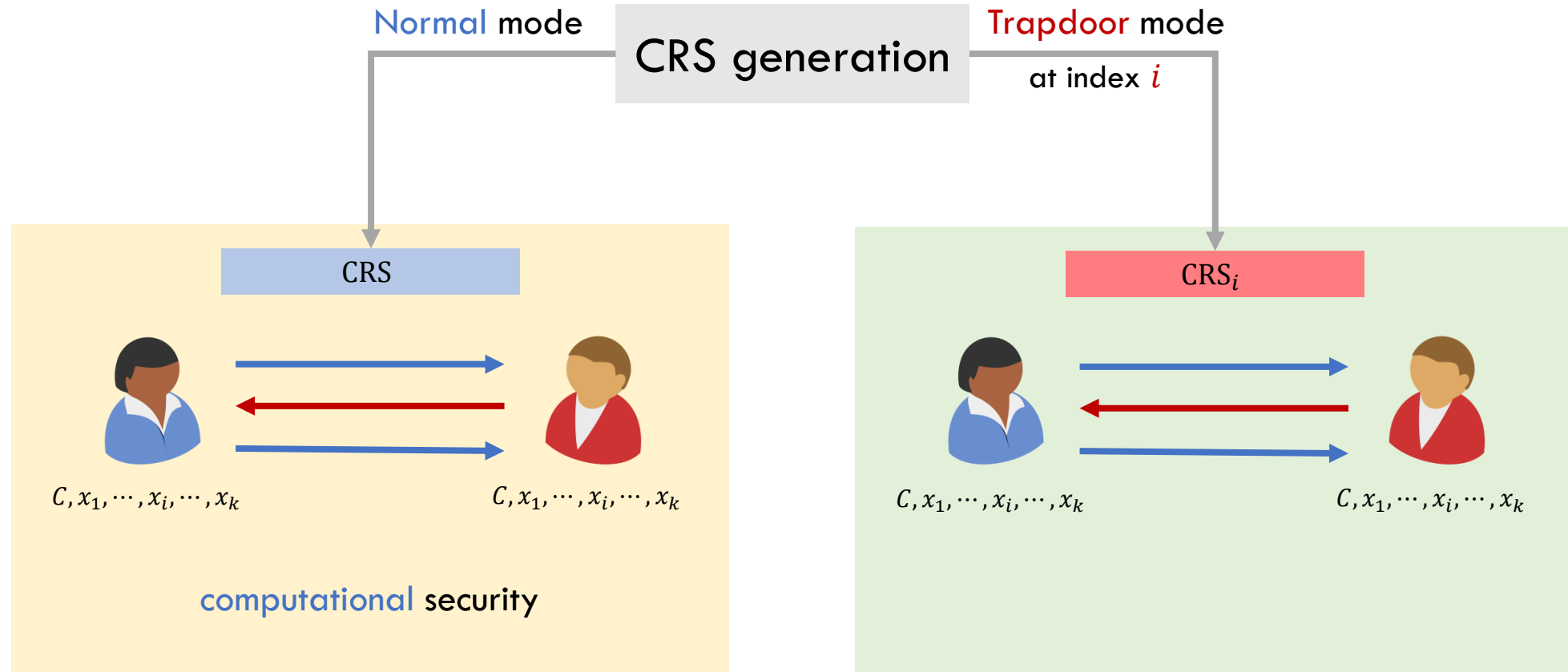
Dual-Mode Interactive Batch Arguments

Dual-Mode

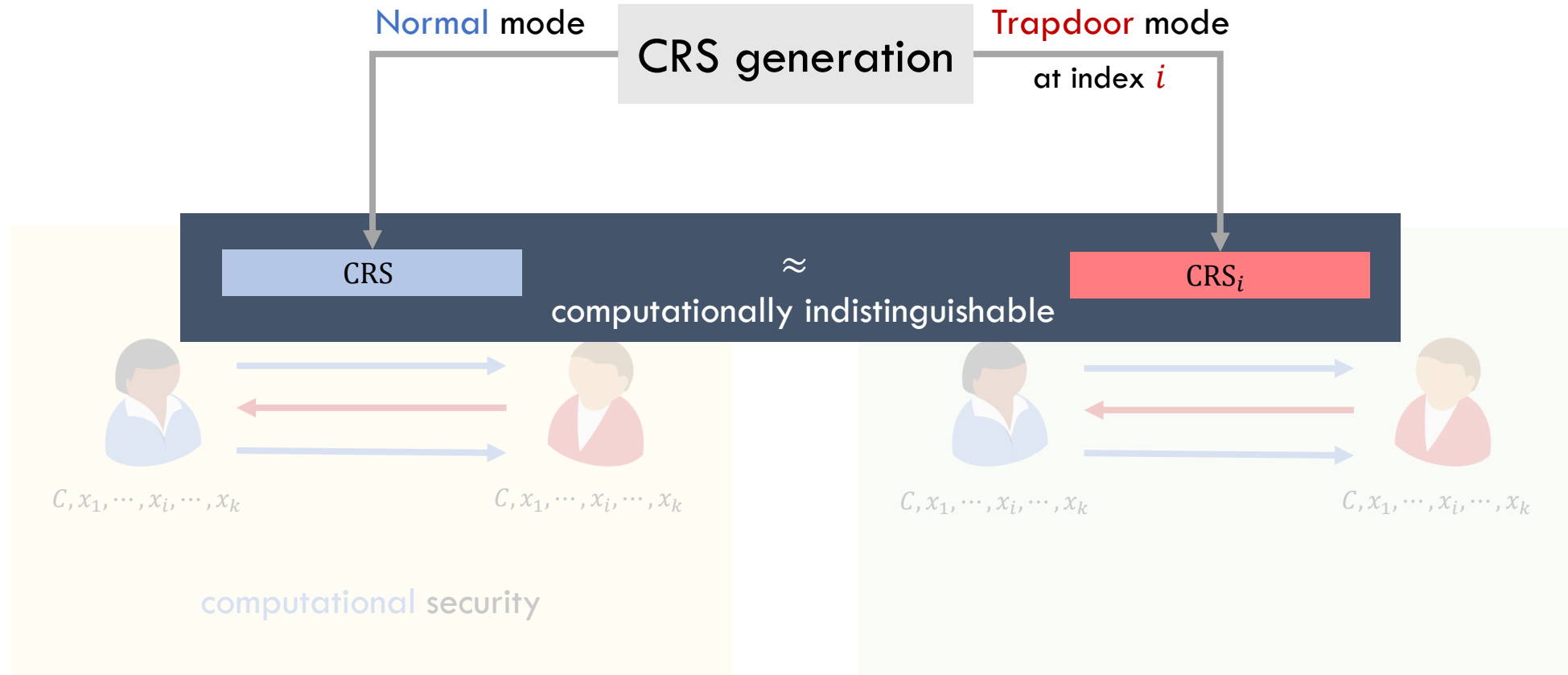
Dual-Mode Interactive Batch Arguments



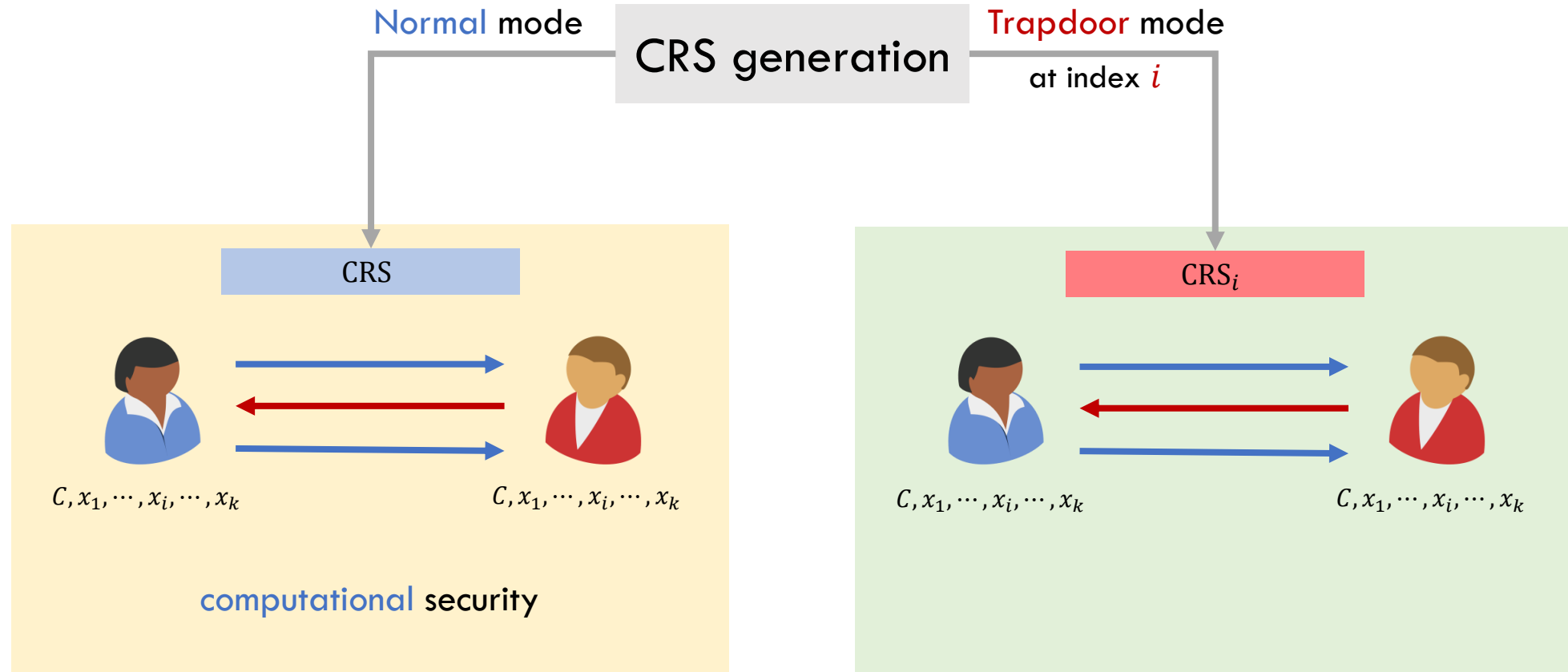
Dual-Mode Interactive Batch Arguments



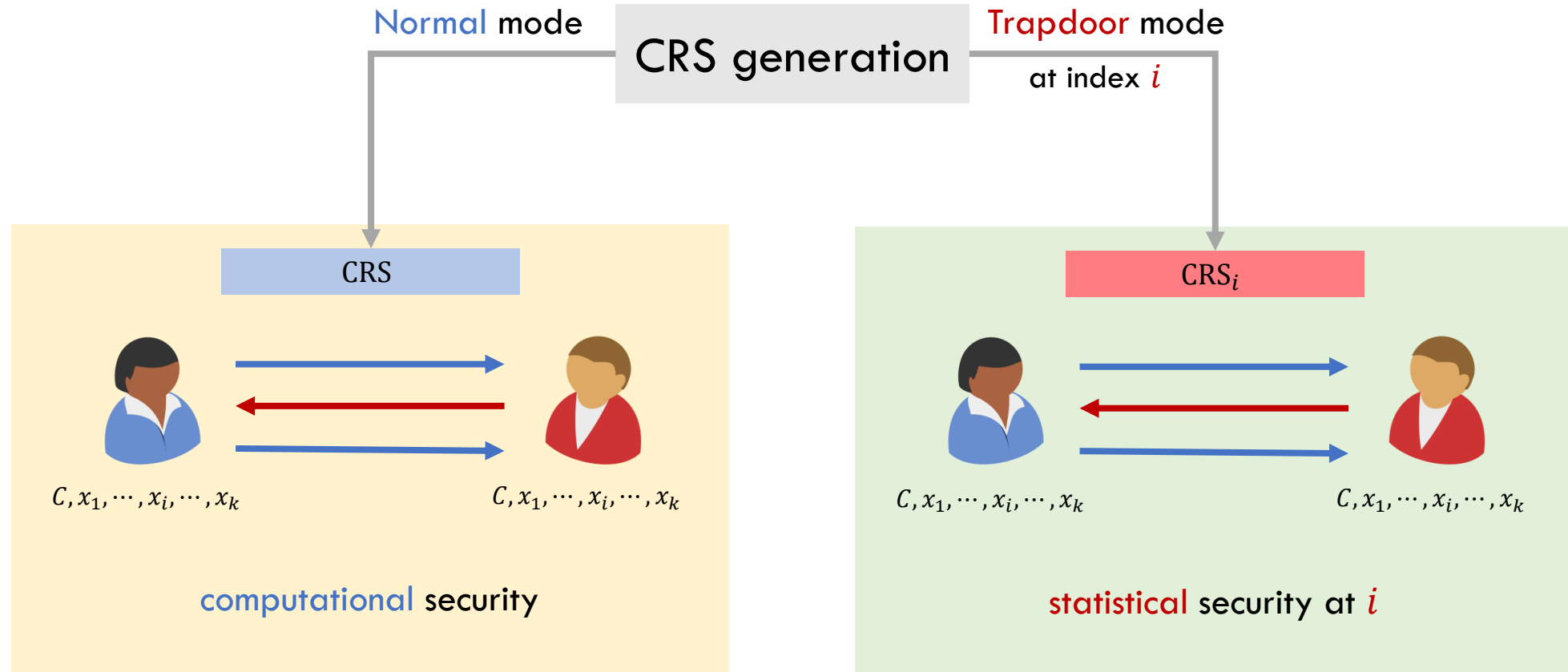
Dual-Mode Interactive Batch Arguments



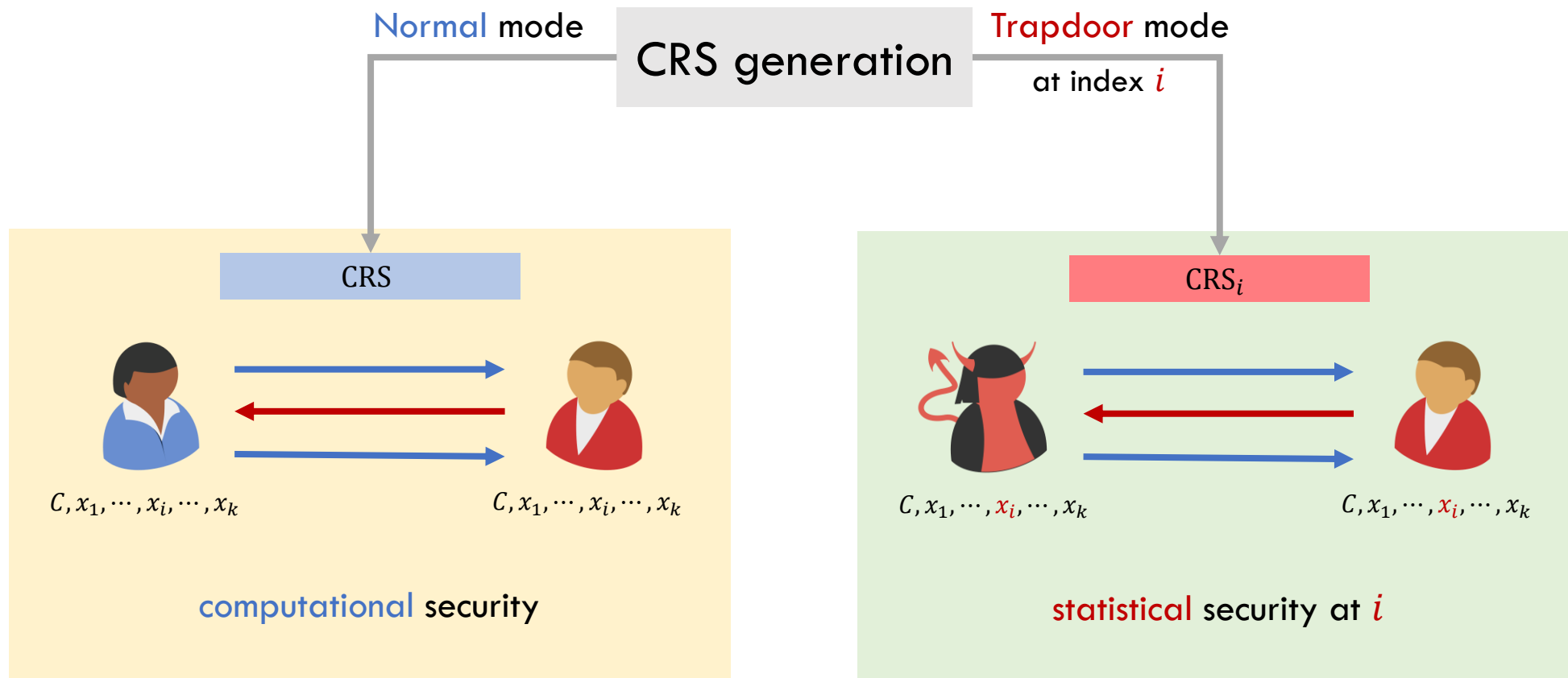
Dual-Mode Interactive Batch Arguments



Dual-Mode Interactive Batch Arguments

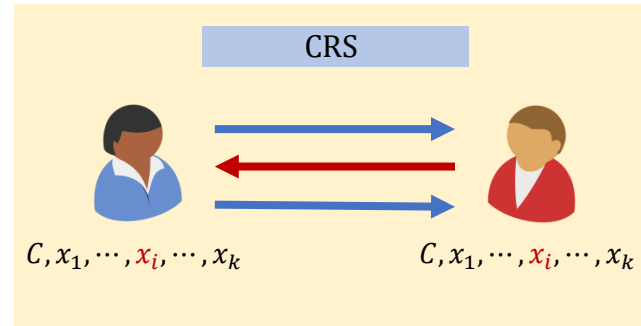


Dual-Mode Interactive Batch Arguments

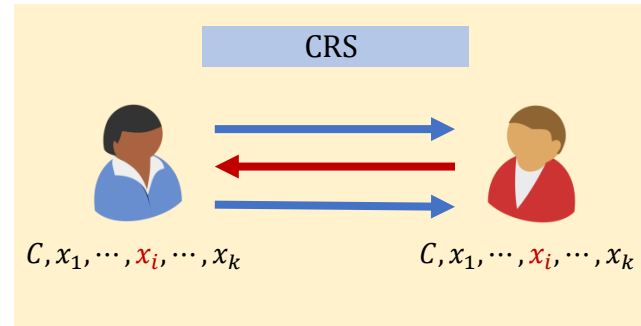


Even **unbounded**  cannot make  accept if $(C, x_i) \notin \text{SAT}$

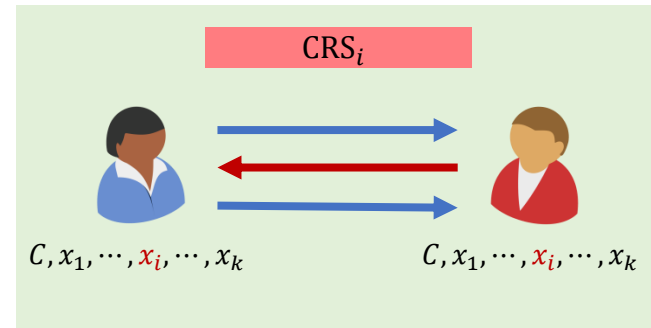
Security Intuition



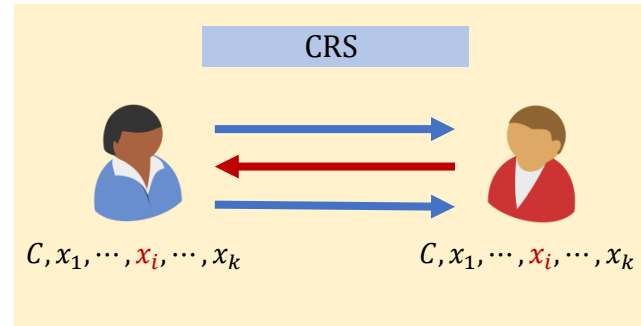
Security Intuition



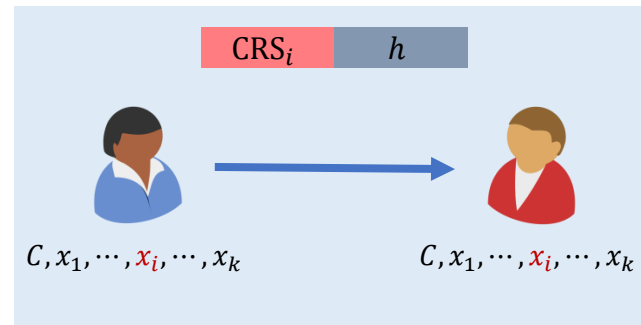
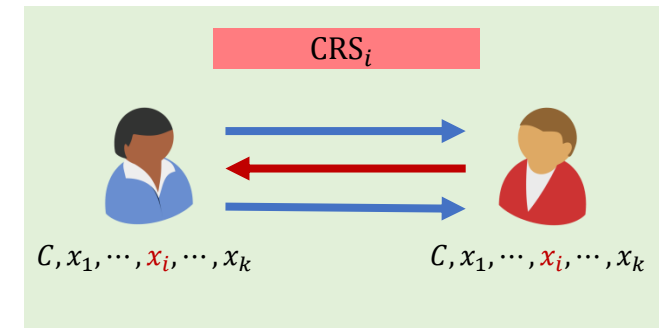
Switch to trapdoor mode at i



Security Intuition

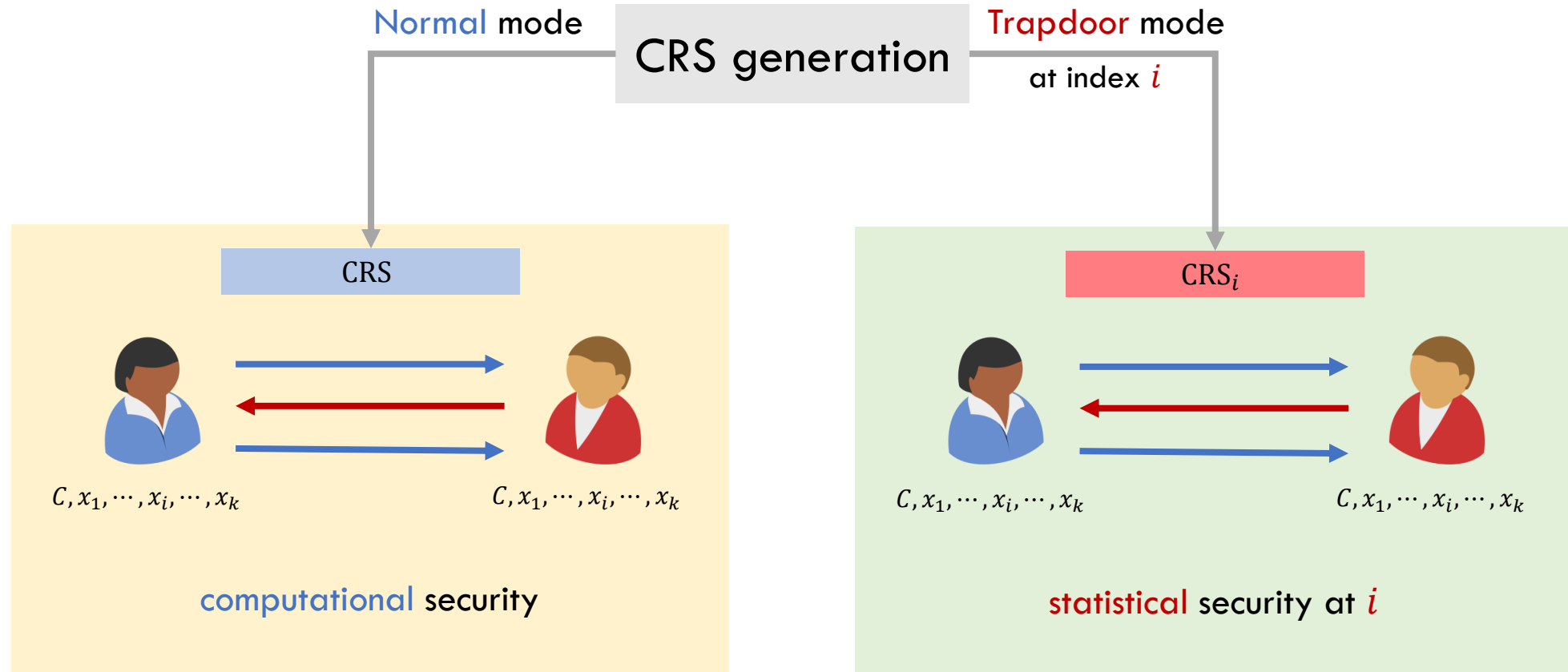


Switch to trapdoor mode at i



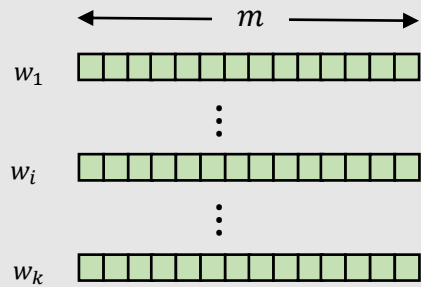
Rely on FS transformation

Dual-Mode Interactive Batch Arguments



Dual Mode Batch Argument

Protocol Template

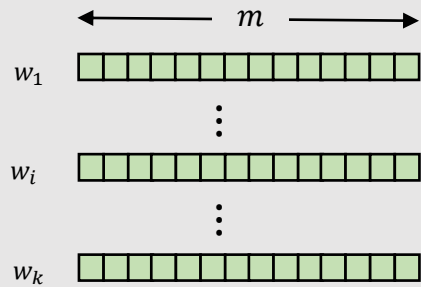


$$\text{SAT} = \{(C, x) \mid \exists w \text{ s. t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Dual Mode Batch Argument

Protocol Template



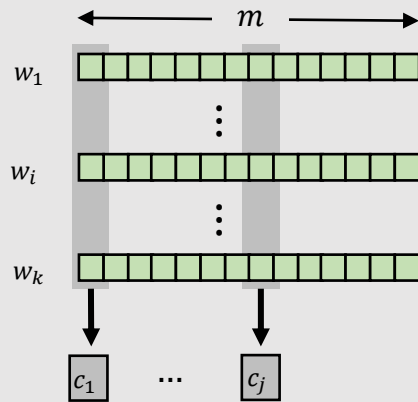
commitment key K

$$\text{SAT} = \{(C, x) \mid \exists w \text{ s. t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Dual Mode Batch Argument

Protocol Template

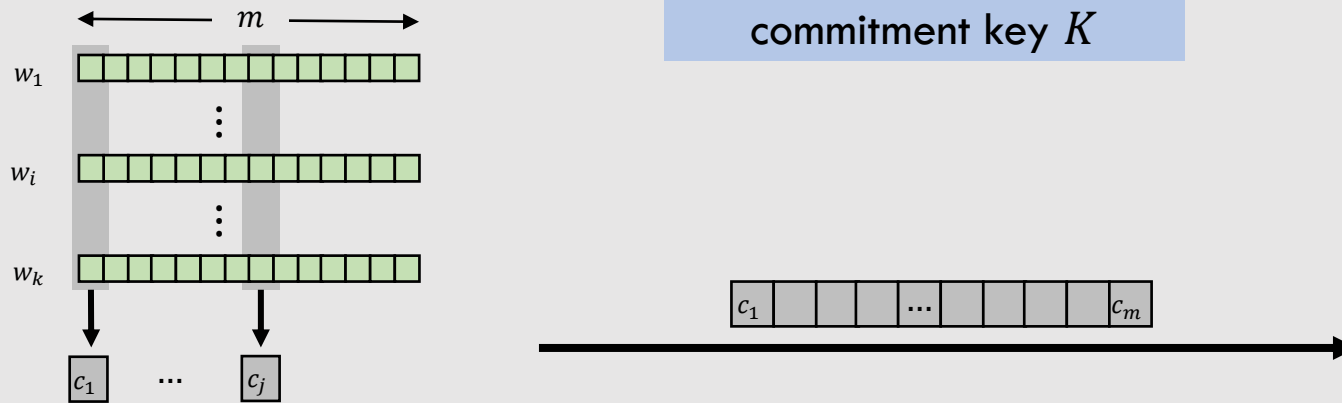


$$\text{SAT} = \{(C, x) \mid \exists w \text{ s. t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Dual Mode Batch Argument

Protocol Template

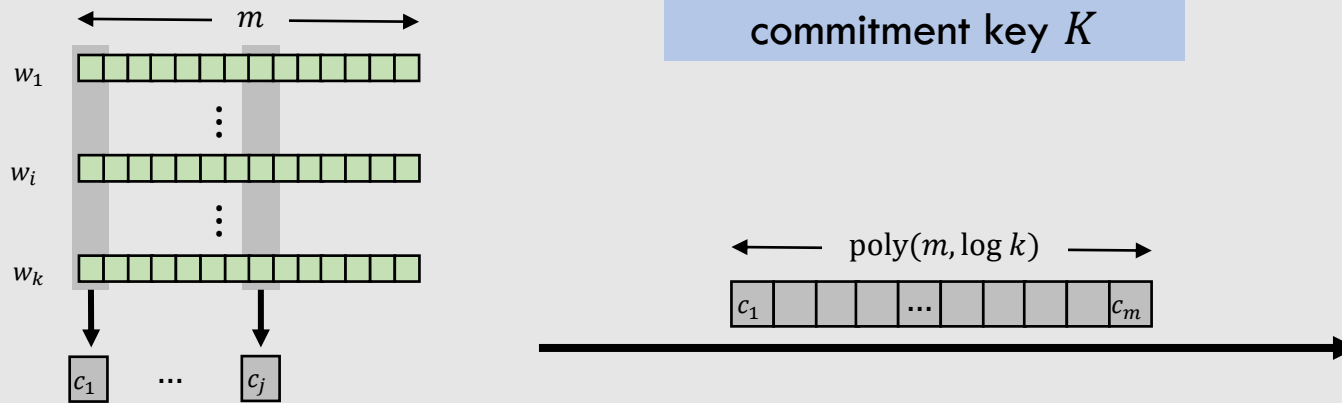


$$\text{SAT} = \{(C, x) \mid \exists w \text{ s. t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Dual Mode Batch Argument

Protocol Template

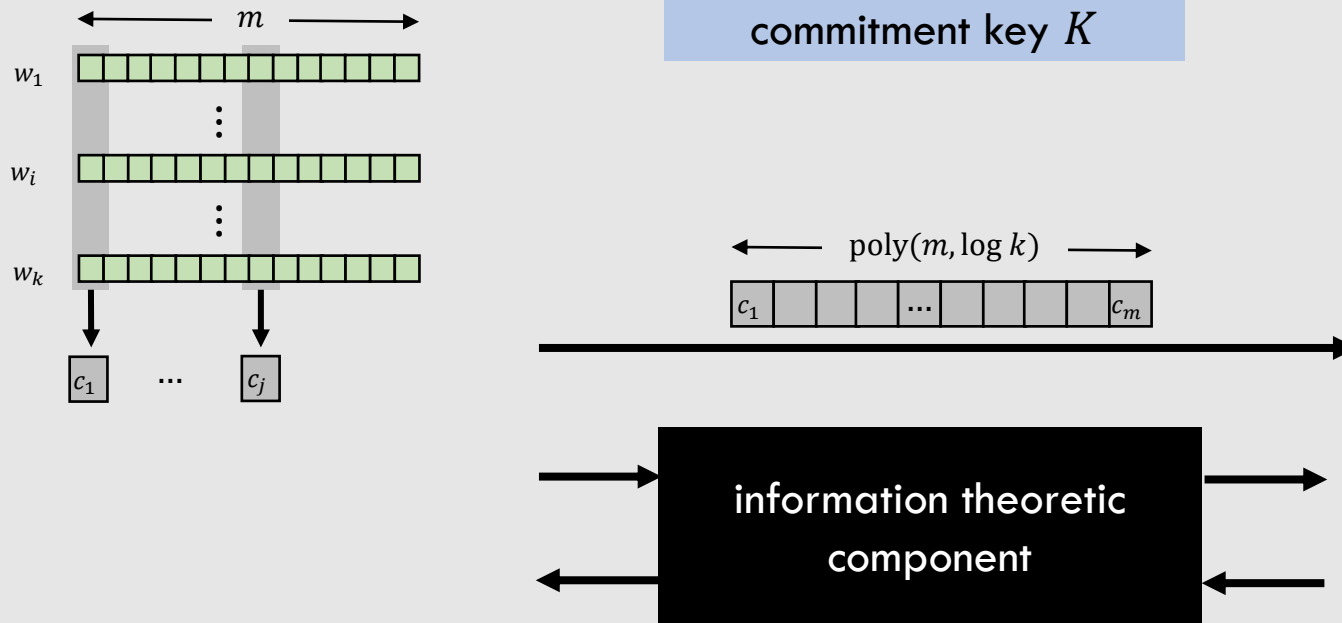


$$\text{SAT} = \{(C, x) \mid \exists w \text{ s. t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Dual Mode Batch Argument

Protocol Template

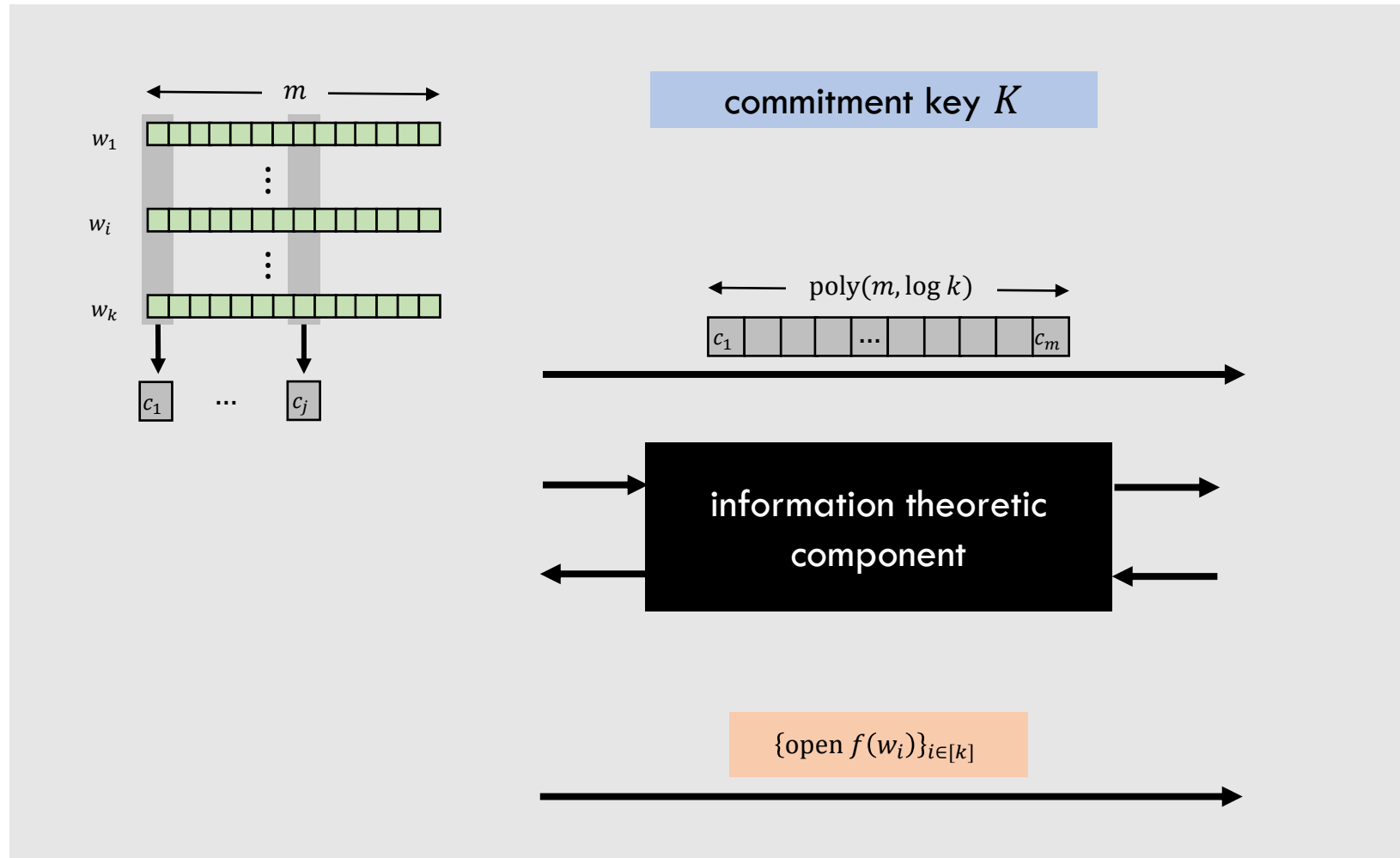


$$\text{SAT} = \{(C, x) \mid \exists w \text{ s. t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Dual Mode Batch Argument

Protocol Template



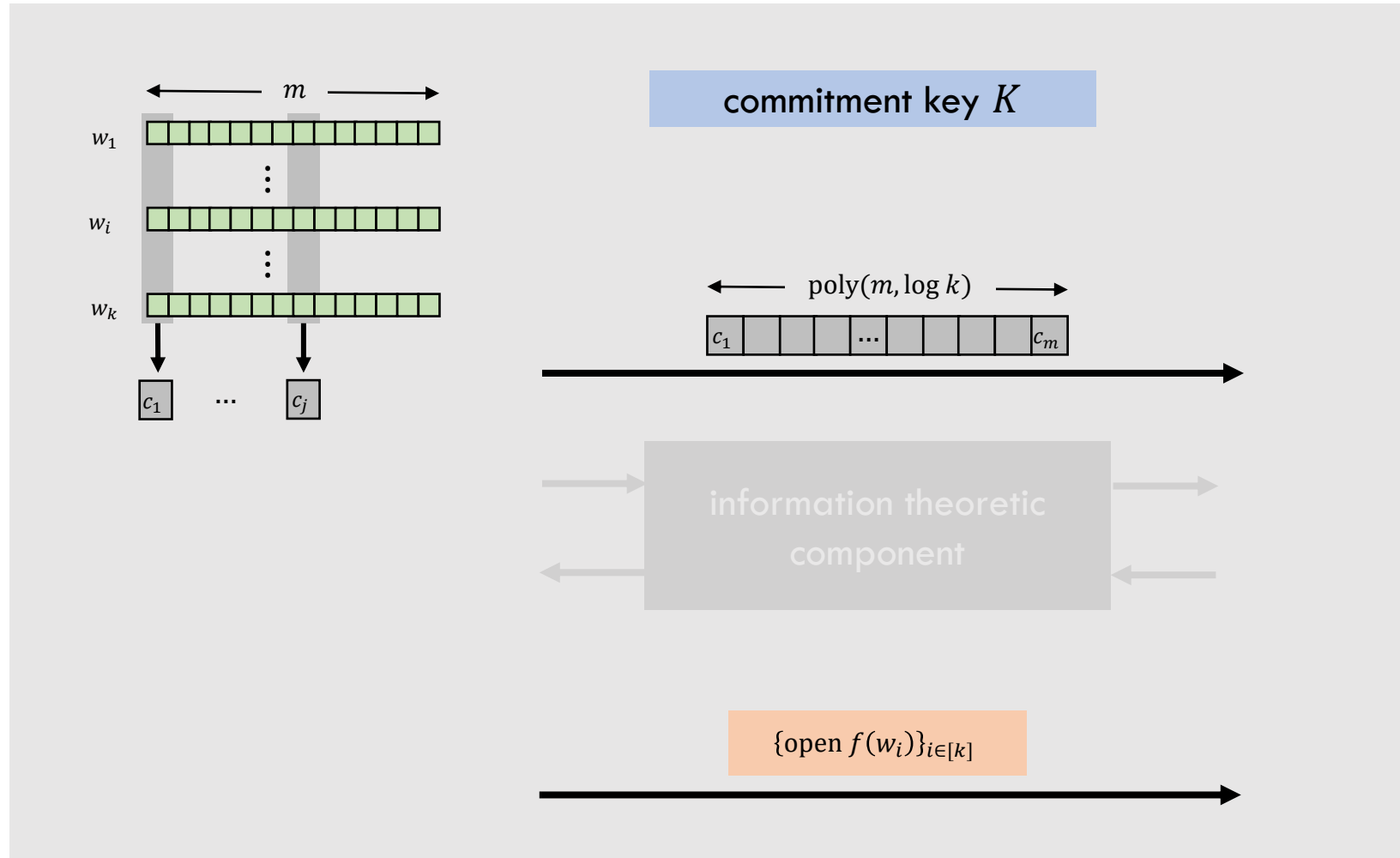
$$\text{SAT} = \{(C, x) \mid \exists w \text{ s. t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

f determined by the information theoretic component.

Dual Mode Batch Argument

Protocol Template

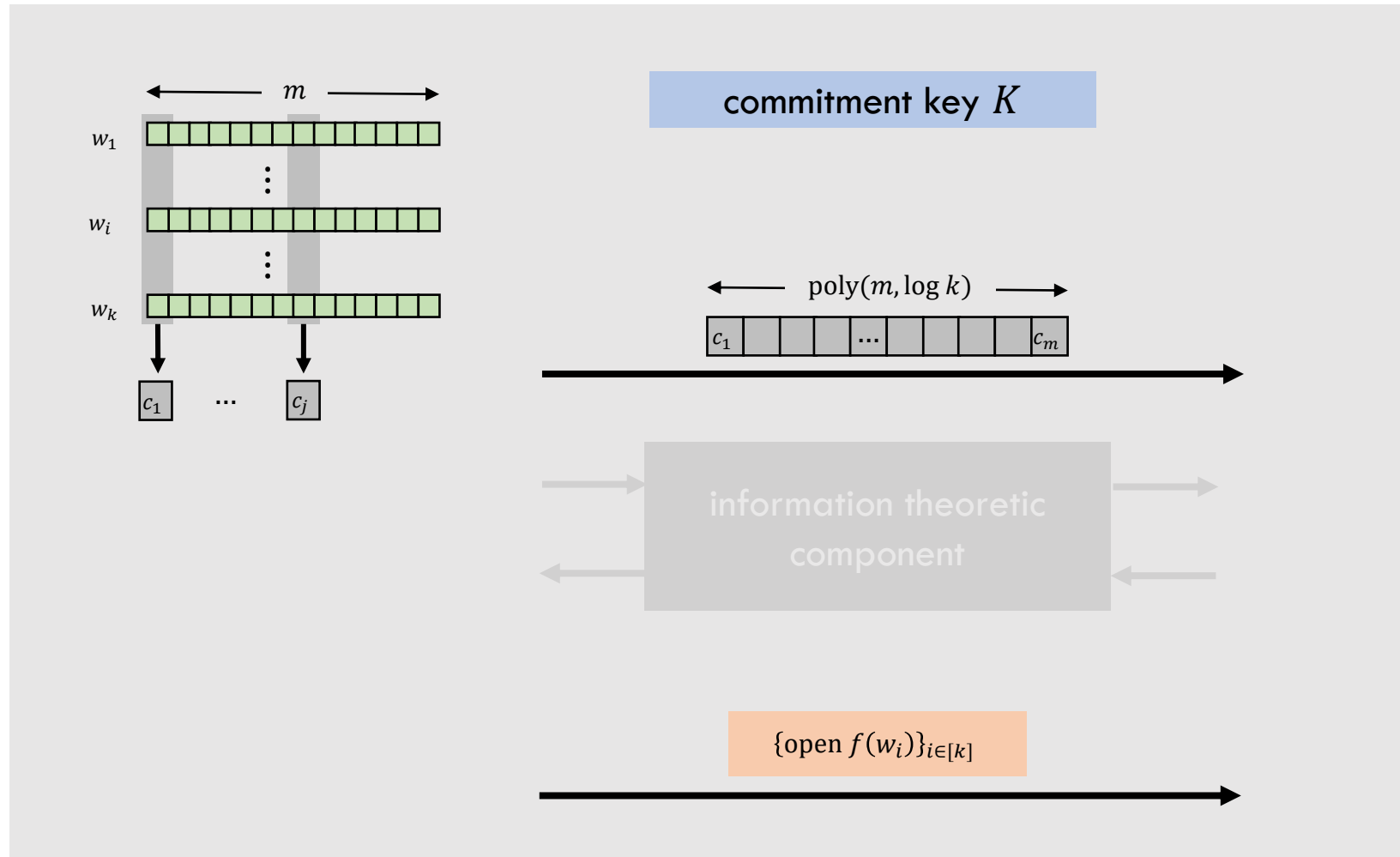


$$\text{SAT} = \{(C, x) \mid \exists w \text{ s. t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Dual Mode Batch Argument

Protocol Template



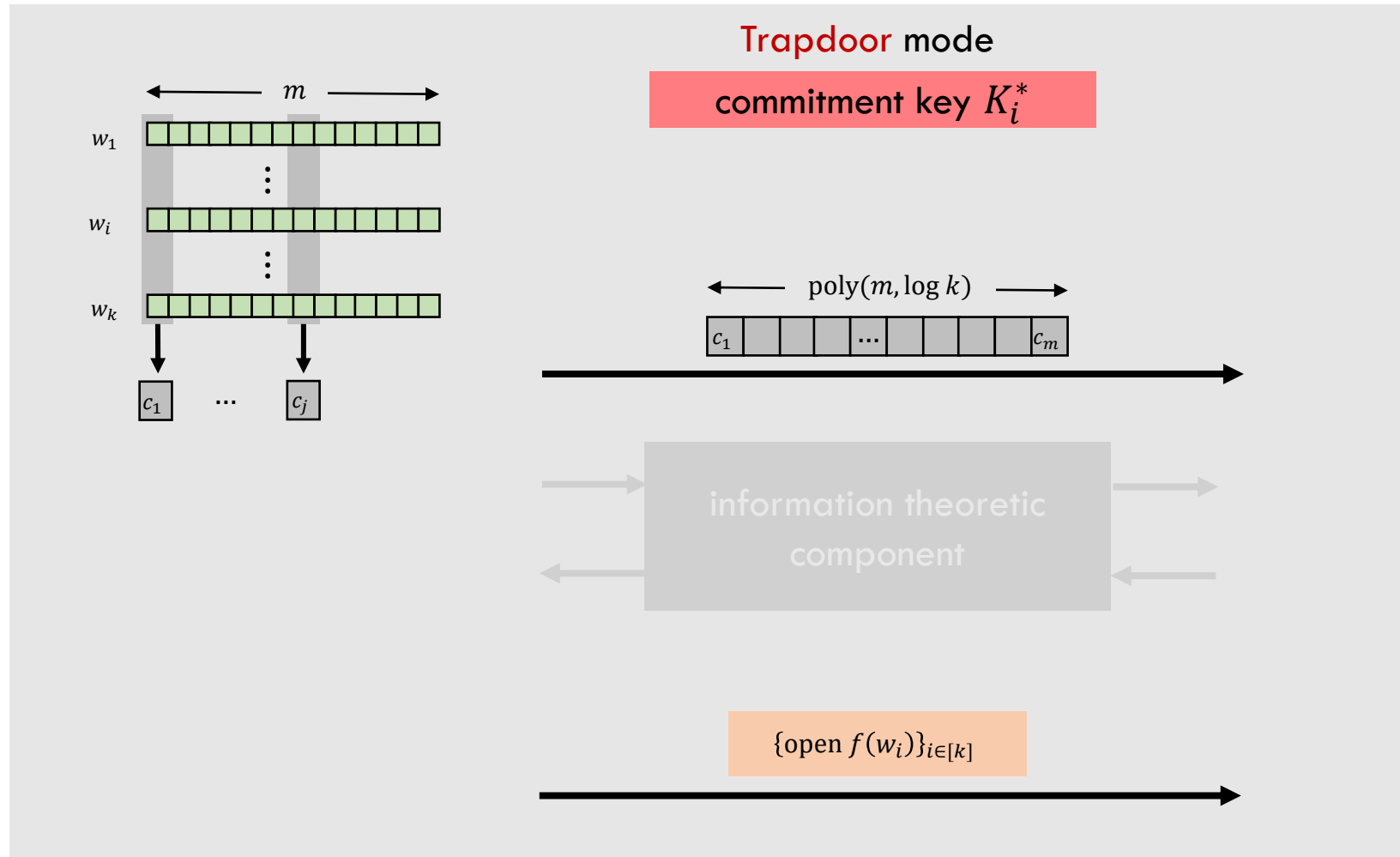
$$\text{SAT} = \{(C, x) \mid \exists w \text{ s. t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Somewhere Statistically
Binding (SSB) Commitment
Scheme [Hubáček - Wichs'15]

Dual Mode Batch Argument

Protocol Template



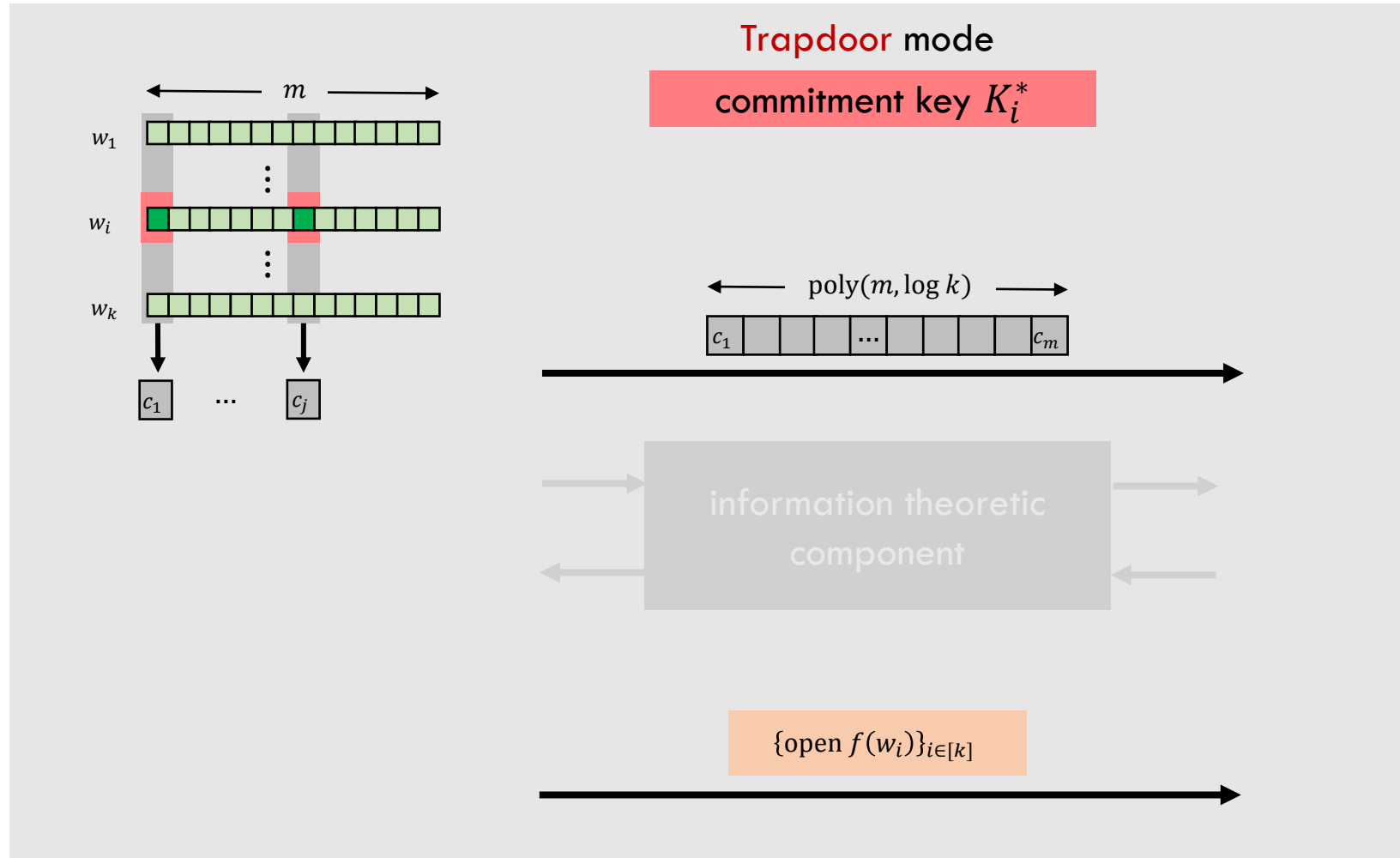
$$\text{SAT} = \{(C, x) \mid \exists w \text{ s. t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Somewhere Statistically
Binding (SSB) Commitment
Scheme

Dual Mode Batch Argument

Protocol Template



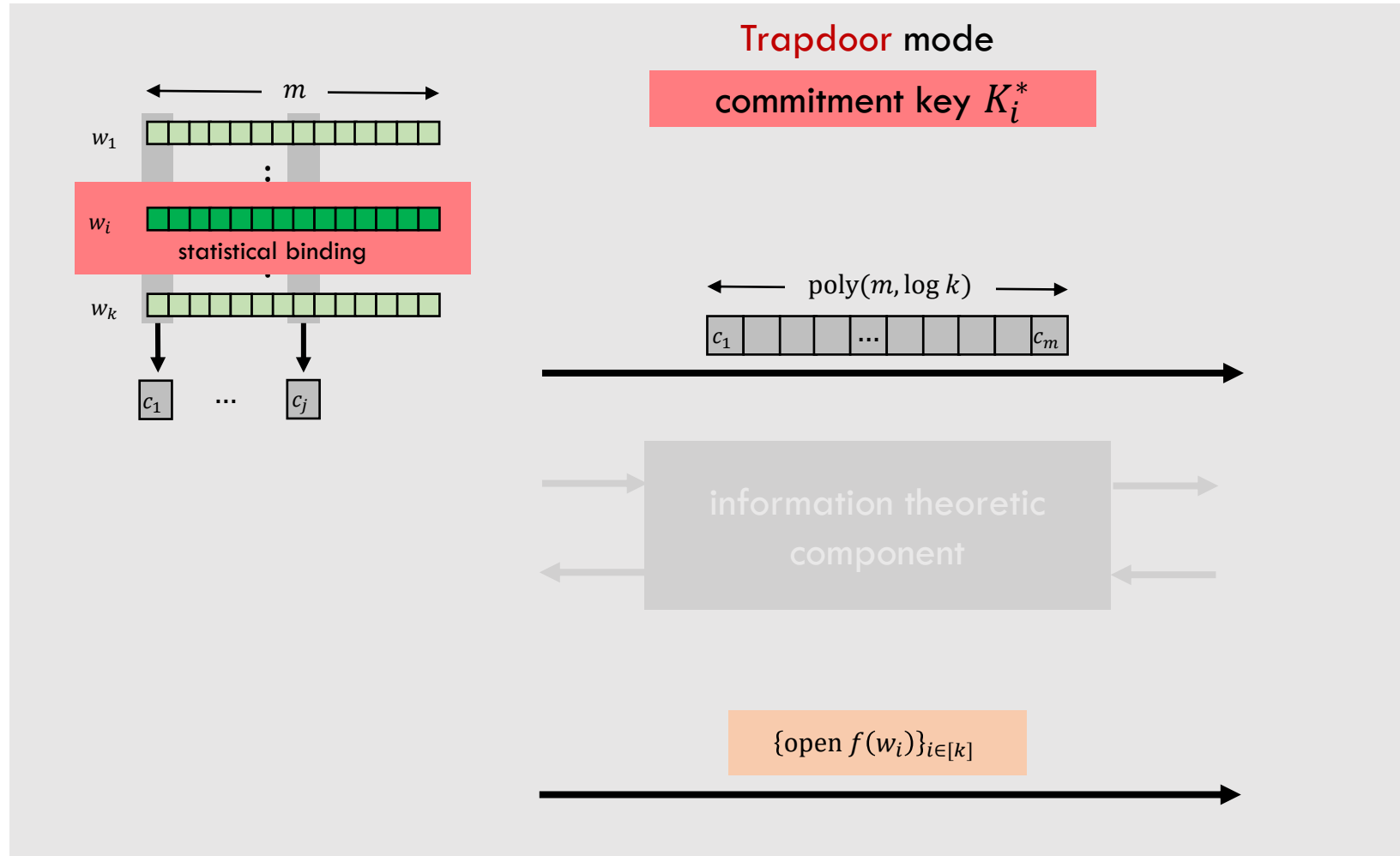
$$\text{SAT} = \{(C, x) \mid \exists w \text{ s. t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Somewhere Statistically
Binding (SSB) Commitment
Scheme

Dual Mode Batch Argument

Protocol Template



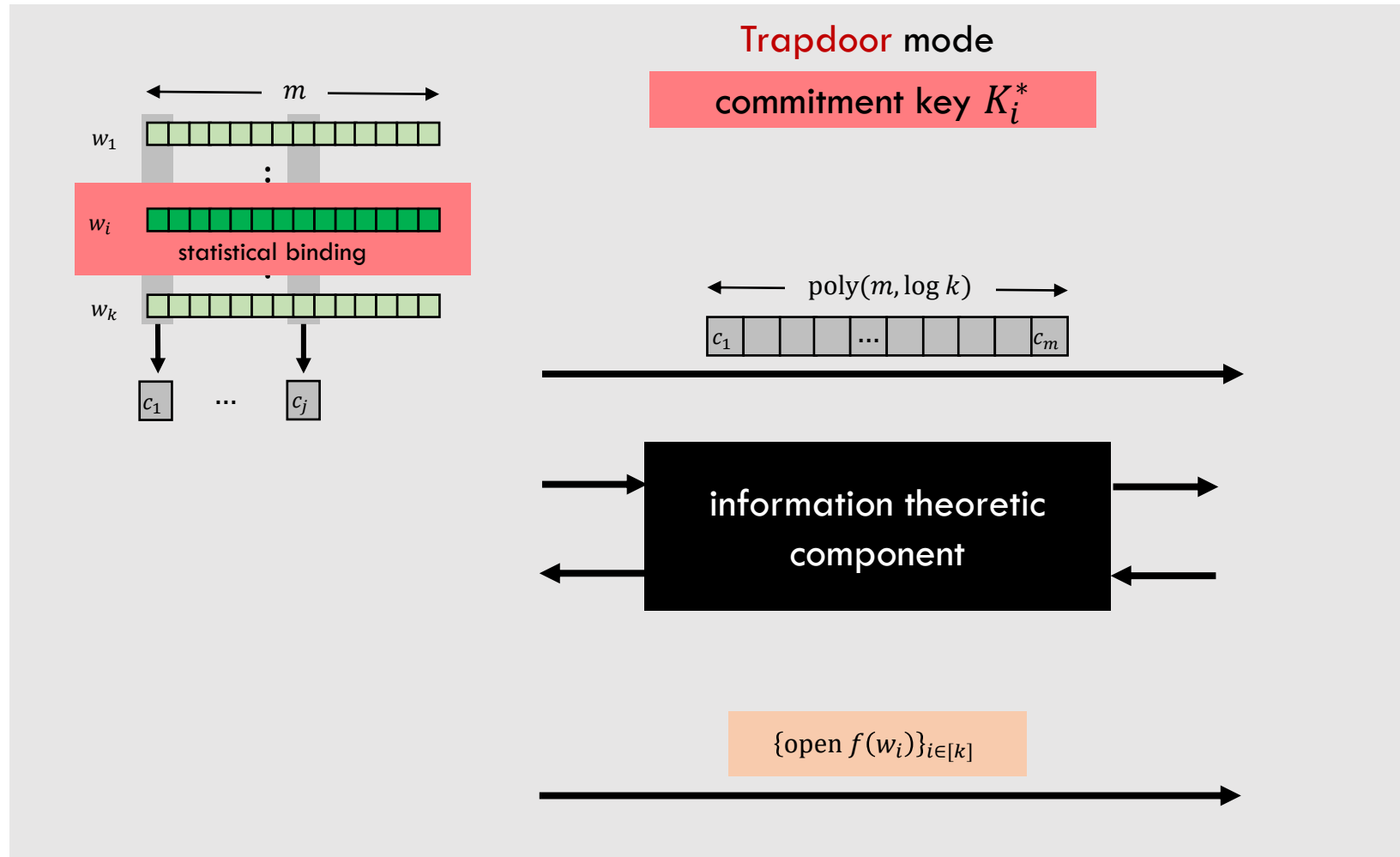
$$\text{SAT} = \{(C, x) \mid \exists w \text{ s. t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Somewhere Statistically
Binding (SSB) Commitment
Scheme

Dual Mode Batch Argument

Protocol Template



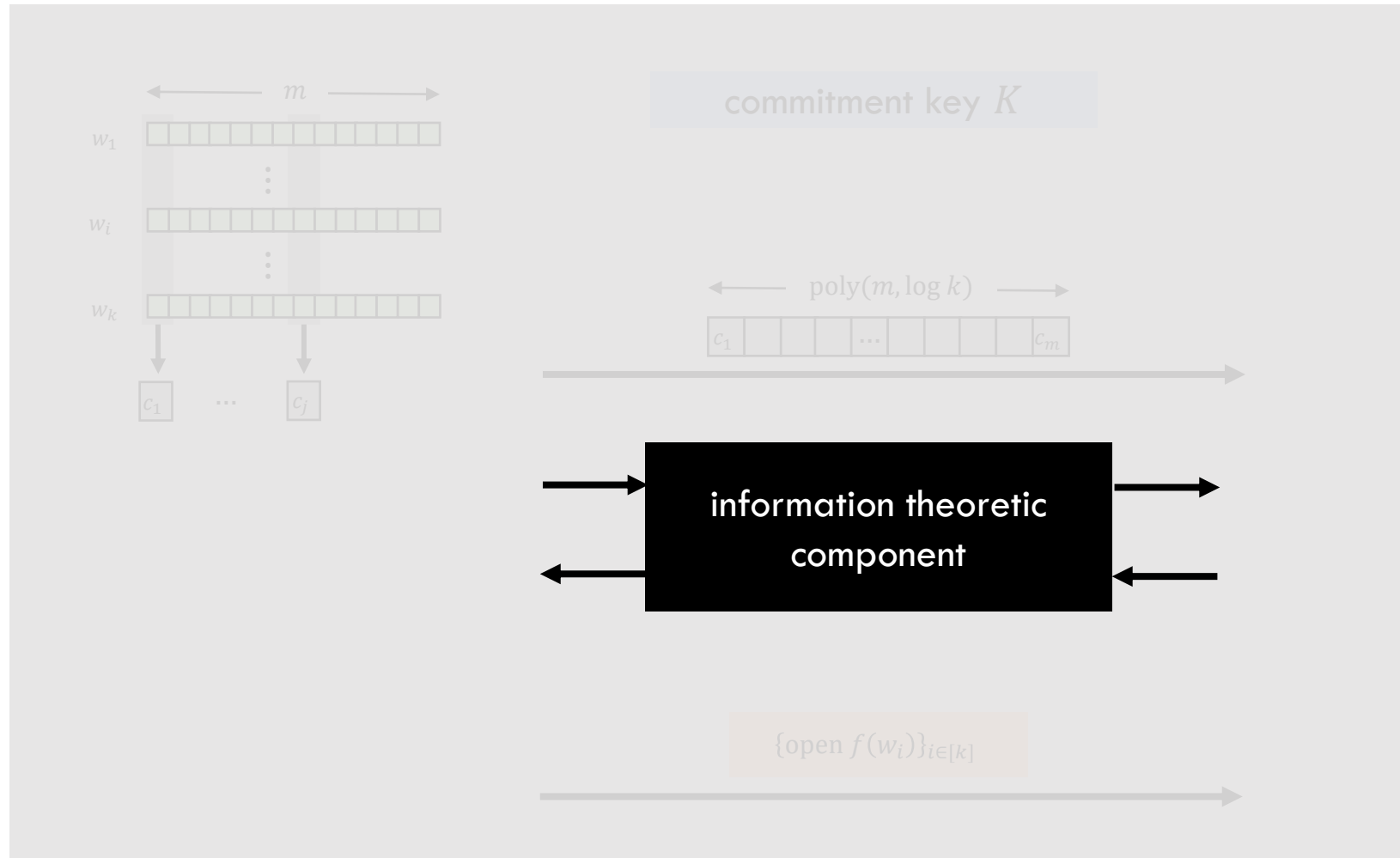
$$\text{SAT} = \{(C, x) \mid \exists w \text{ s. t. } C(x, w) = 1\}$$

$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Somewhere Statistically
Binding (SSB) Commitment
Scheme

Dual Mode Batch Argument

Protocol Template



$$\text{SAT} = \{(C, x) \mid \exists w \text{ s. t. } C(x, w) = 1\}$$

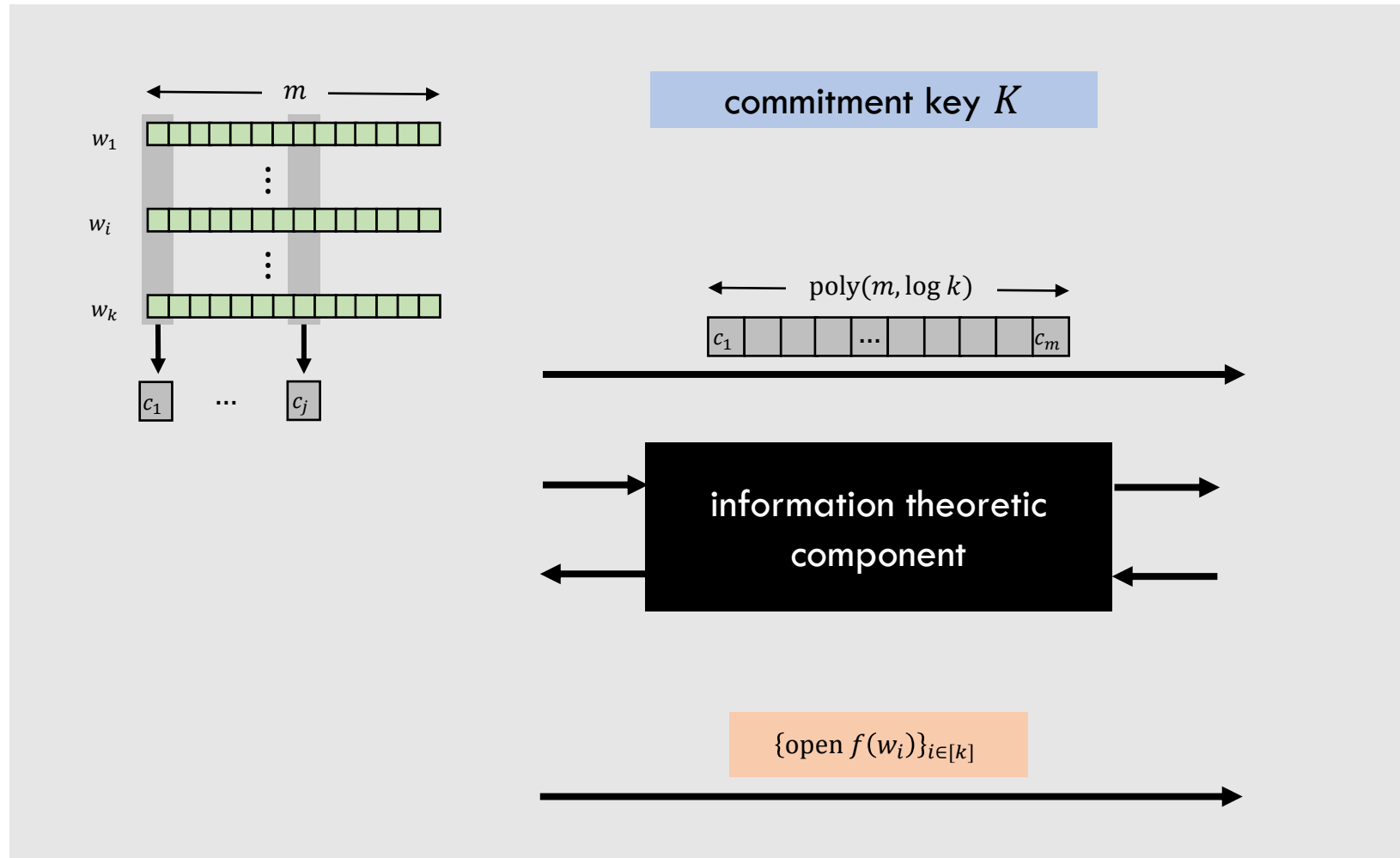
$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Somewhere Statistically
Binding (SSB) Commitment
Scheme

Needs to be Fiat-Shamir
friendly.
Based on LWE/sub-exp DDH

Dual Mode Batch Argument

Protocol Template



$$\text{SAT} = \{(C, x) \mid \exists w \text{ s. t. } C(x, w) = 1\}$$

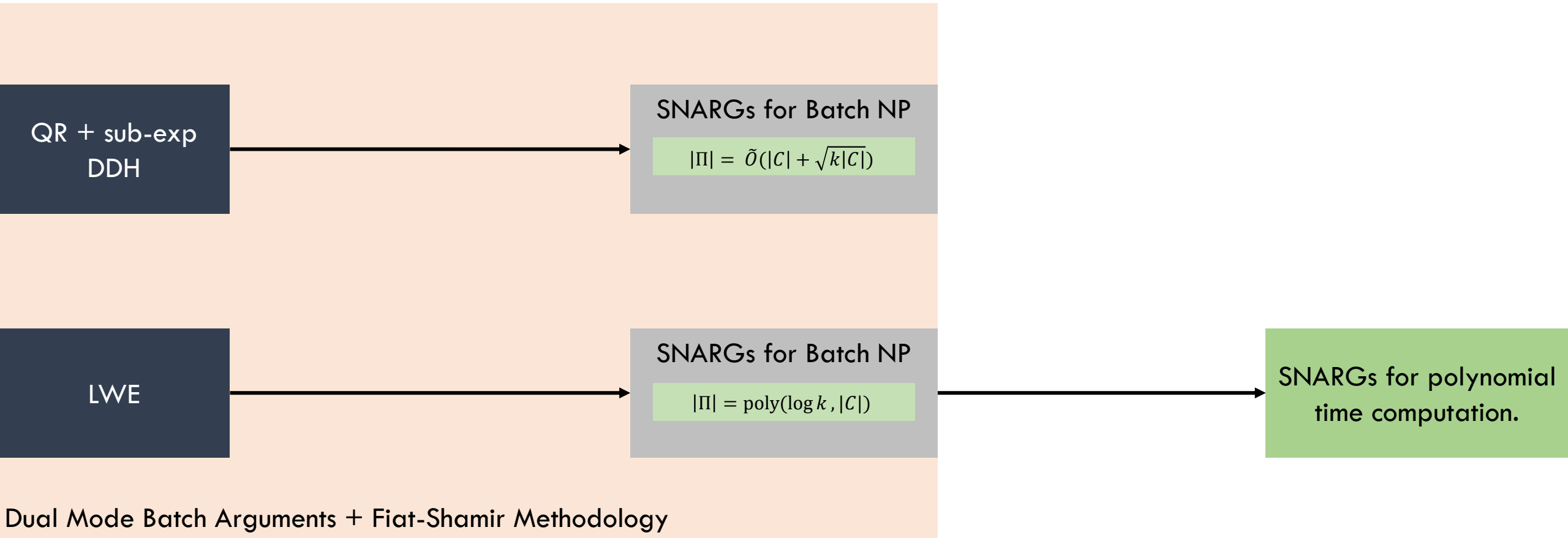
$$\forall i \in [k], (C, x_i) \in \text{SAT}$$

Somewhere Statistically Binding (SSB) Commitment Scheme

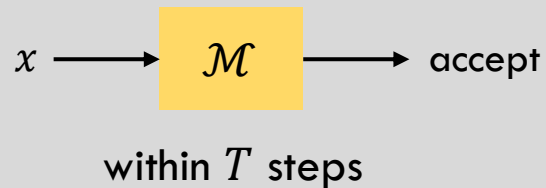
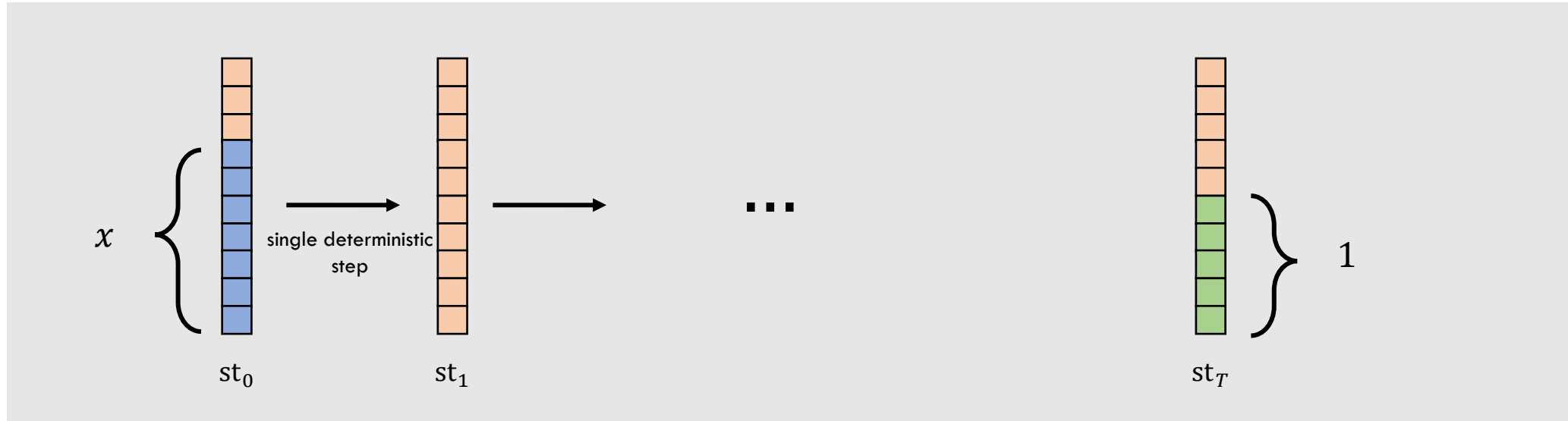
Needs to be Fiat-Shamir friendly.

Based on LWE/sub-exp DDH

Results Overview

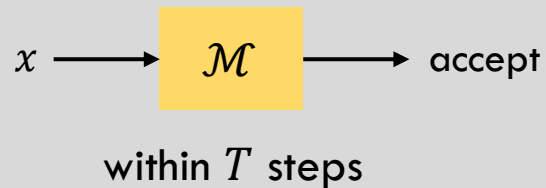
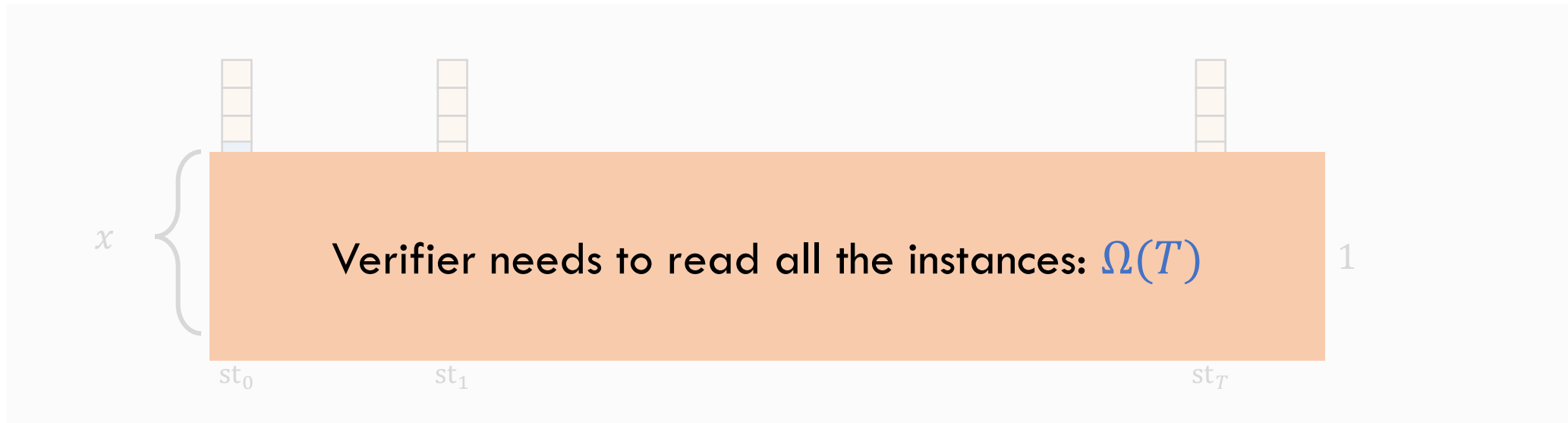


Delegation via **Batching** [Reingold-Rothblum-Rothblum'16]



Prove for every $i \in [0, \dots, T - 1]$
 $st_i \rightarrow st_{i+1}$
is the correct transition.

Delegation via **Batching** [Reingold-Rothblum-Rothblum'16]



Prove for every $i \in [0, \dots, T - 1]$
 $st_i \rightarrow st_{i+1}$
is the correct transition.

SNARGs for Batch Index

$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$



C, k

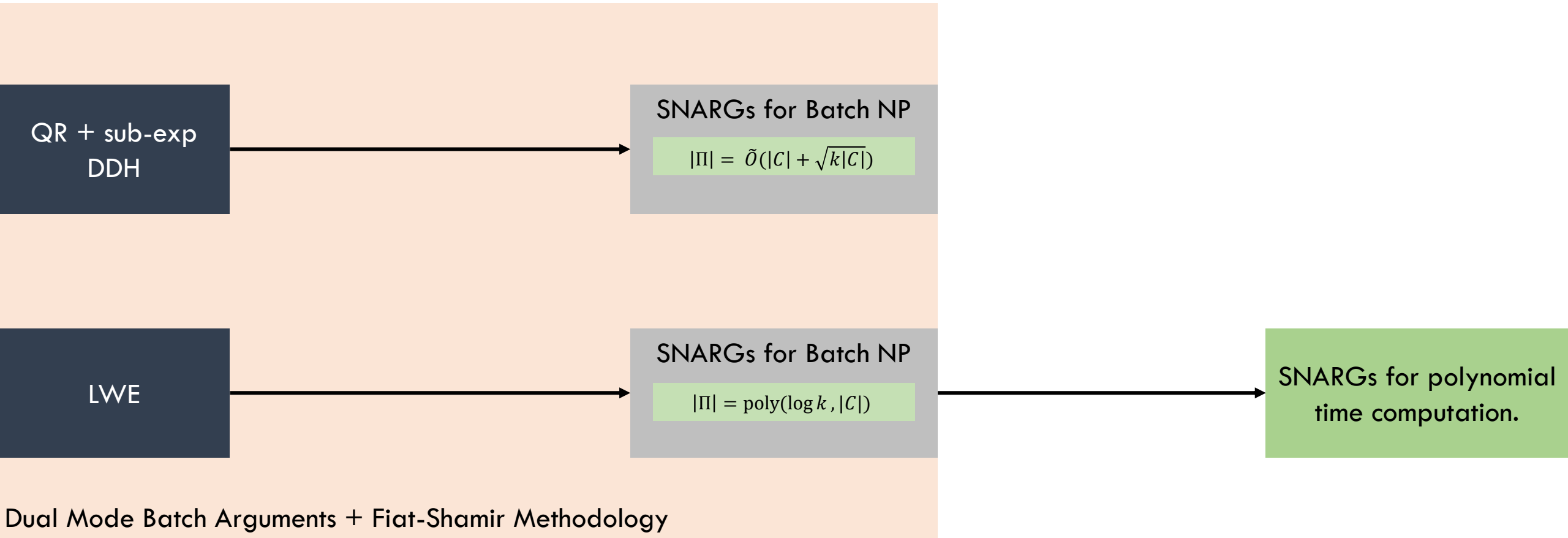
Π



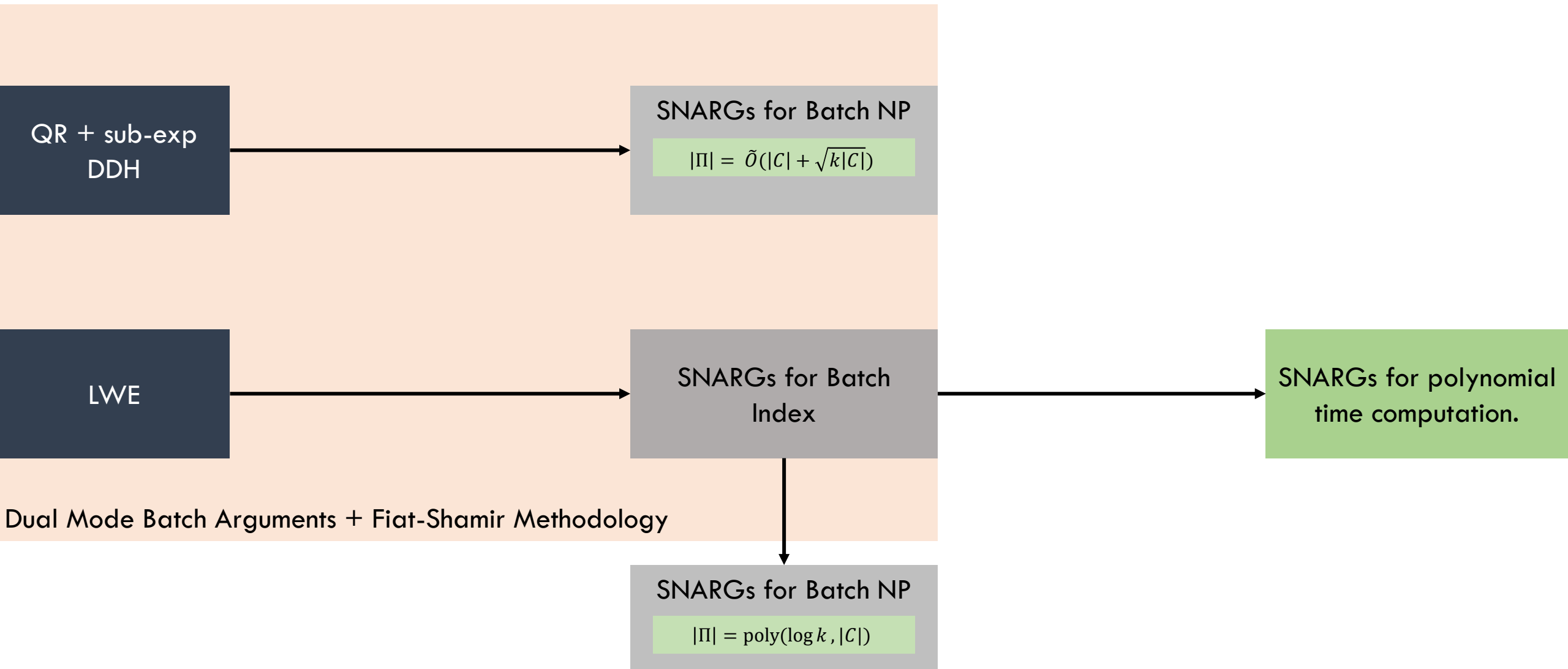
C, k

Verifier **running time**:
 $\text{poly}(\log k, |C|)$

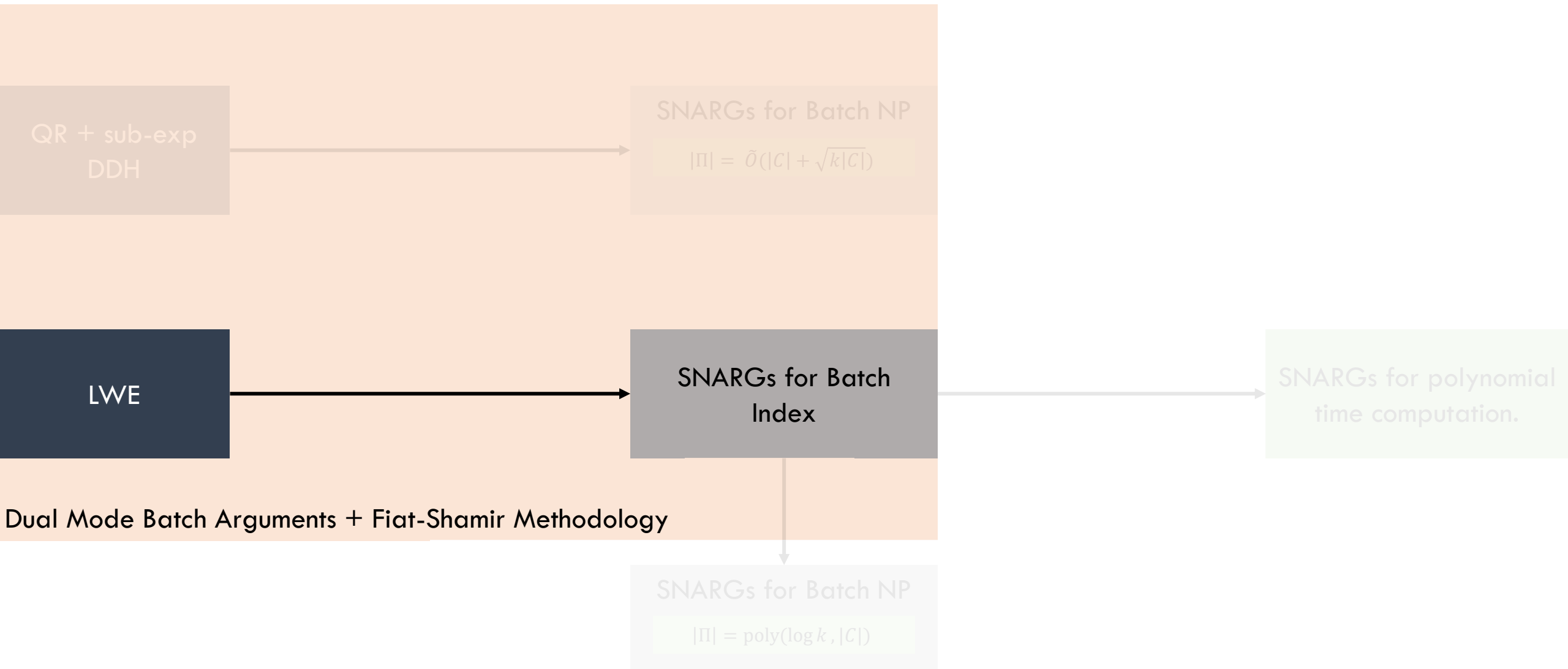
Results Overview



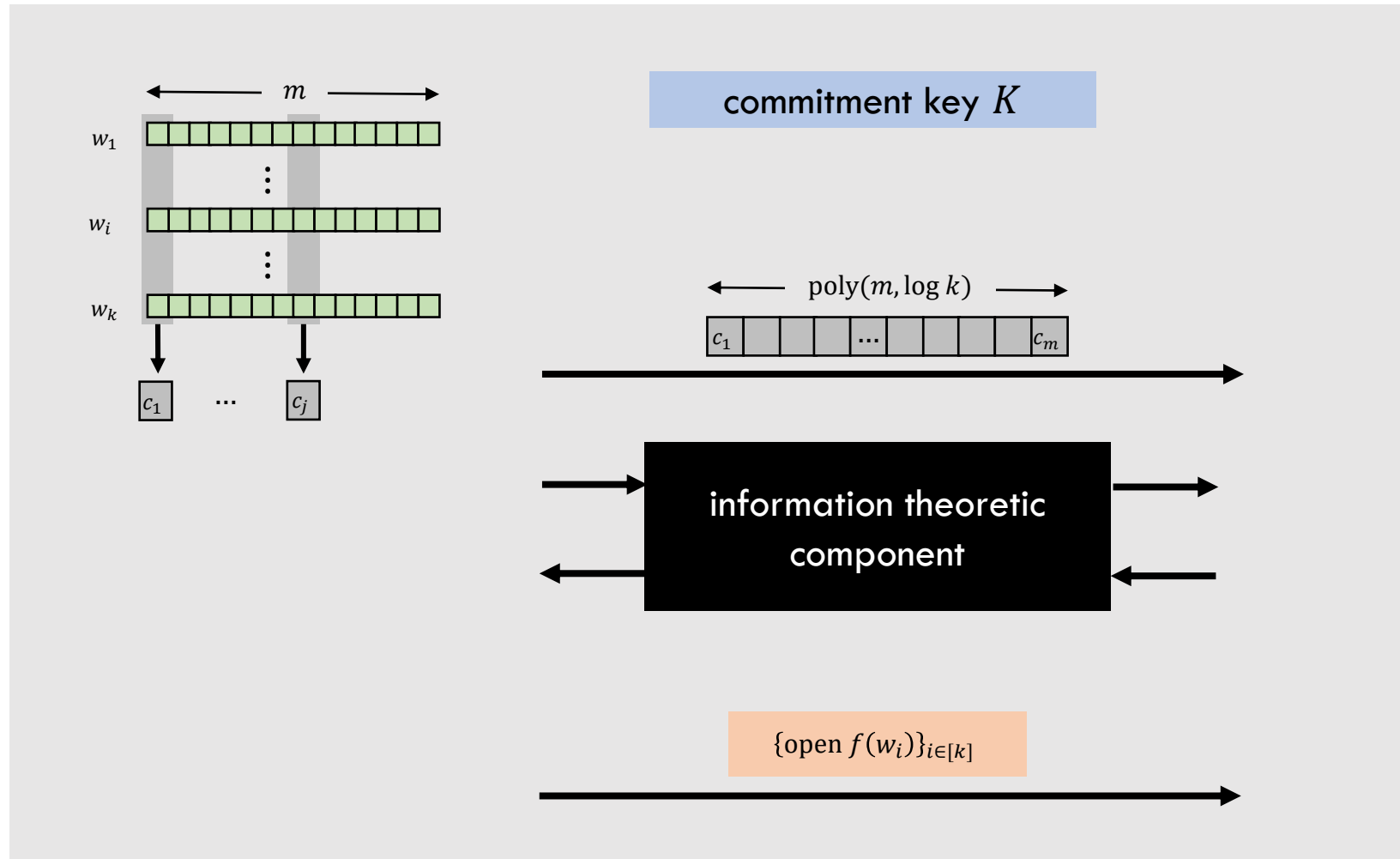
Results Overview



Results Overview



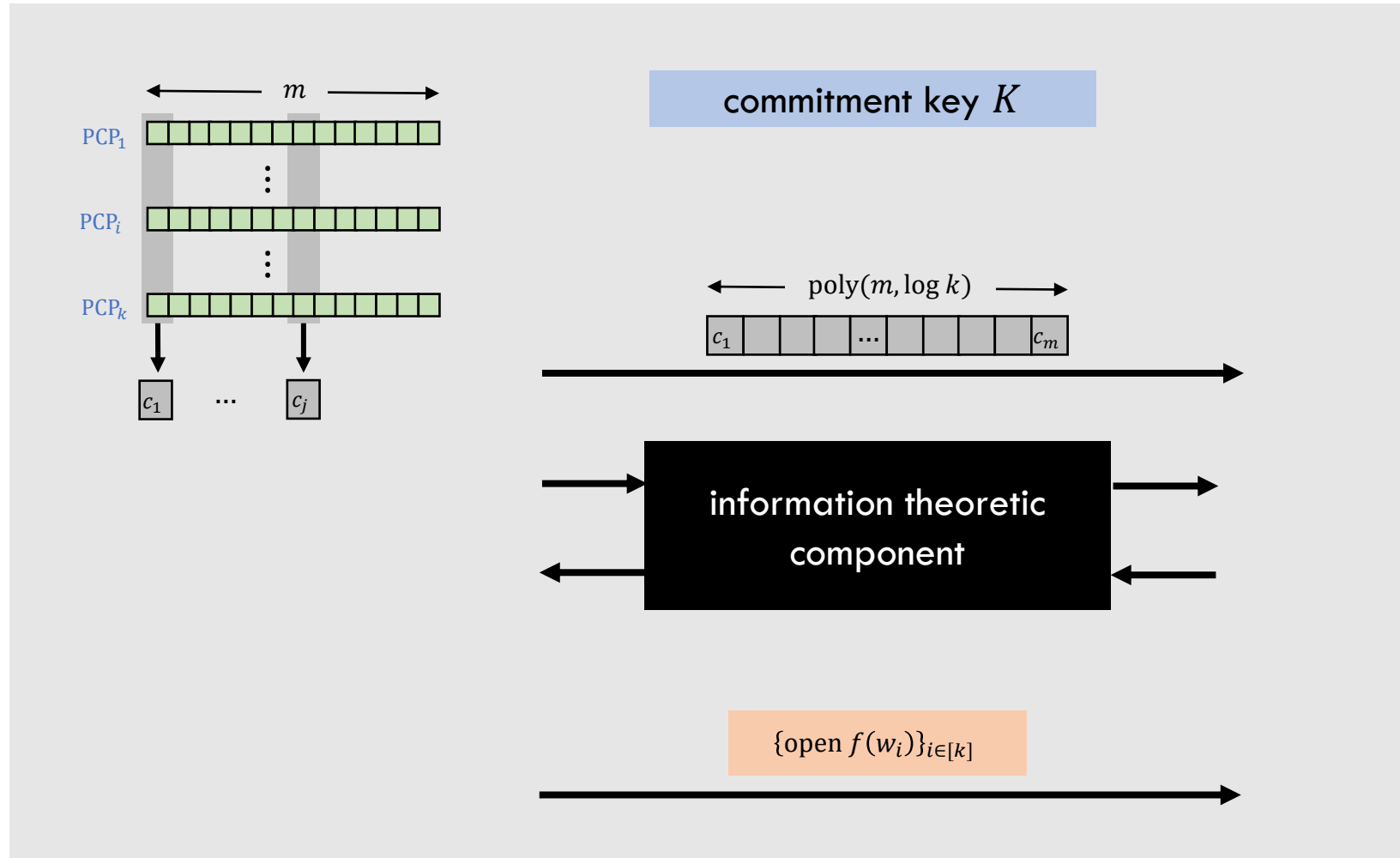
Dual Mode Argument for Batch Index



$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

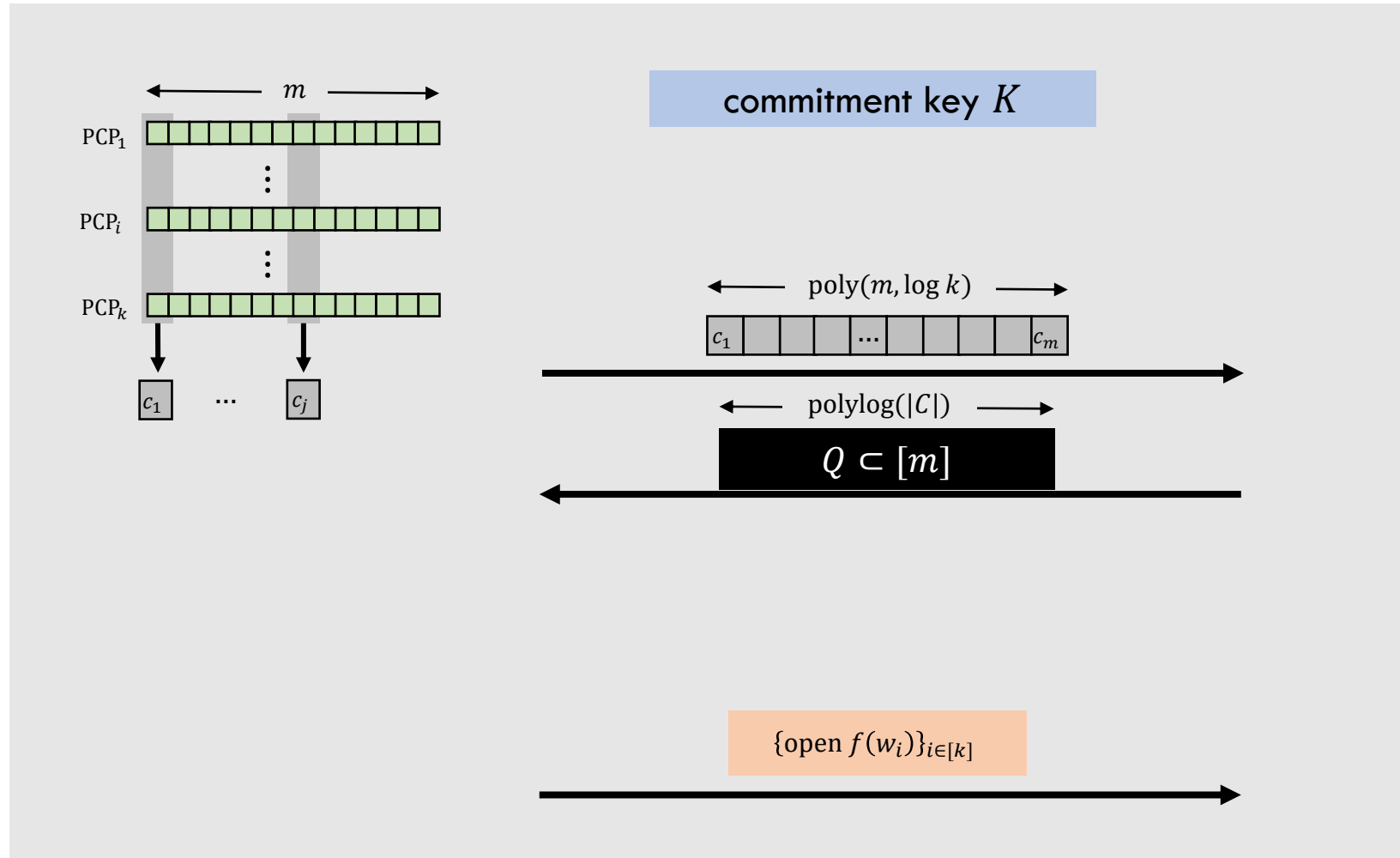
Dual Mode Argument for Batch Index



$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

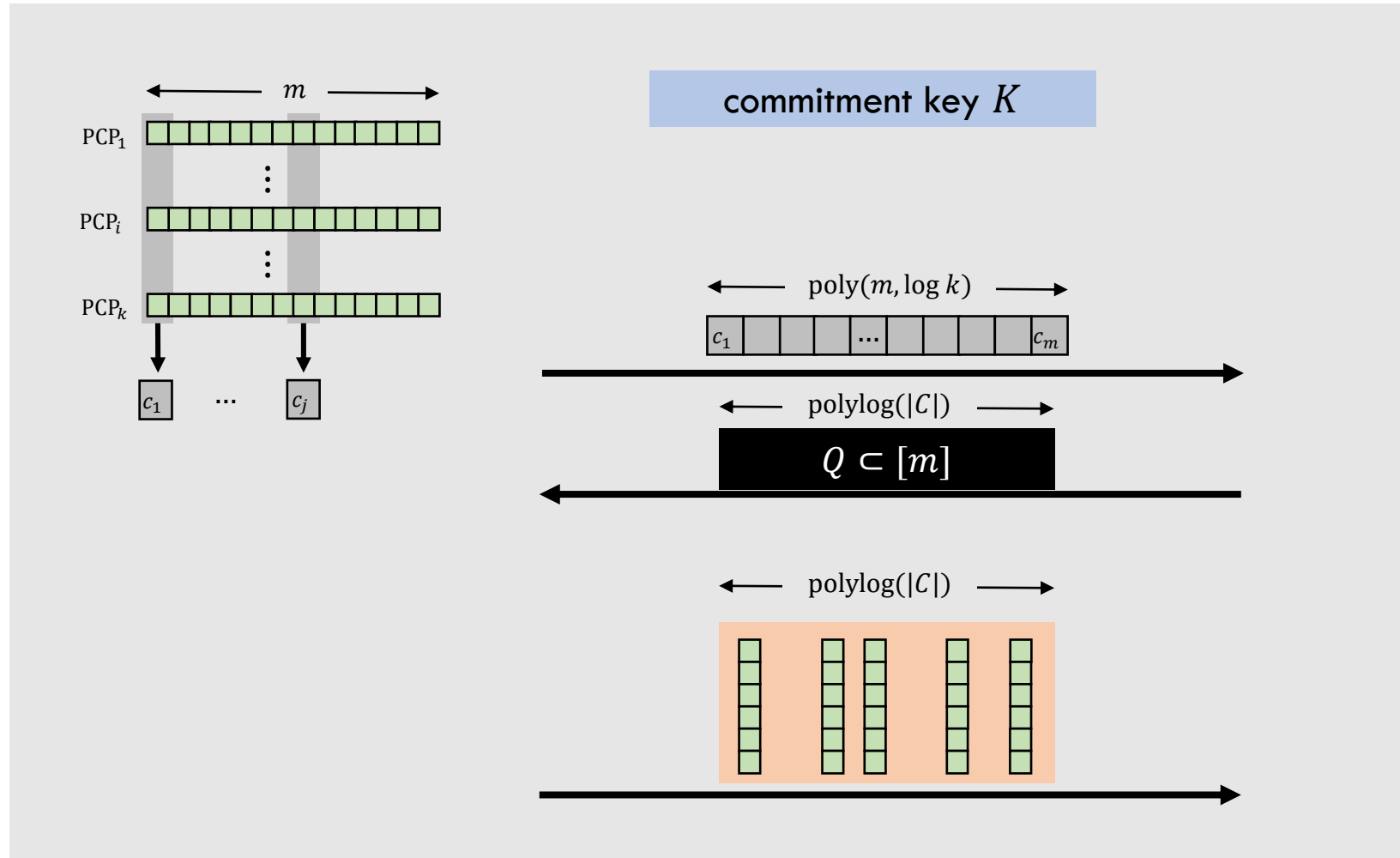
Dual Mode Argument for Batch Index



$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

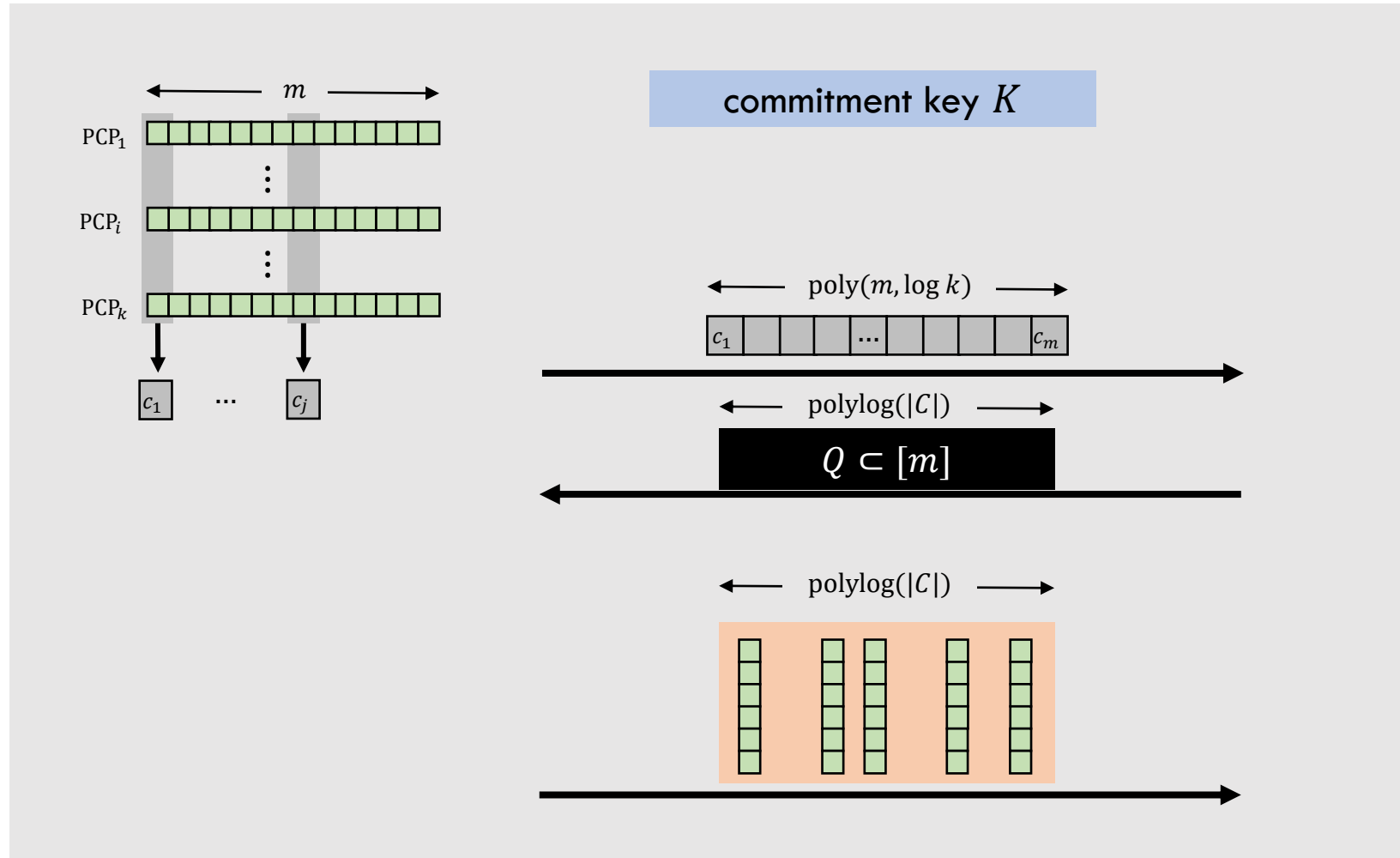
Dual Mode Argument for Batch Index



$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

Dual Mode Argument for Batch Index



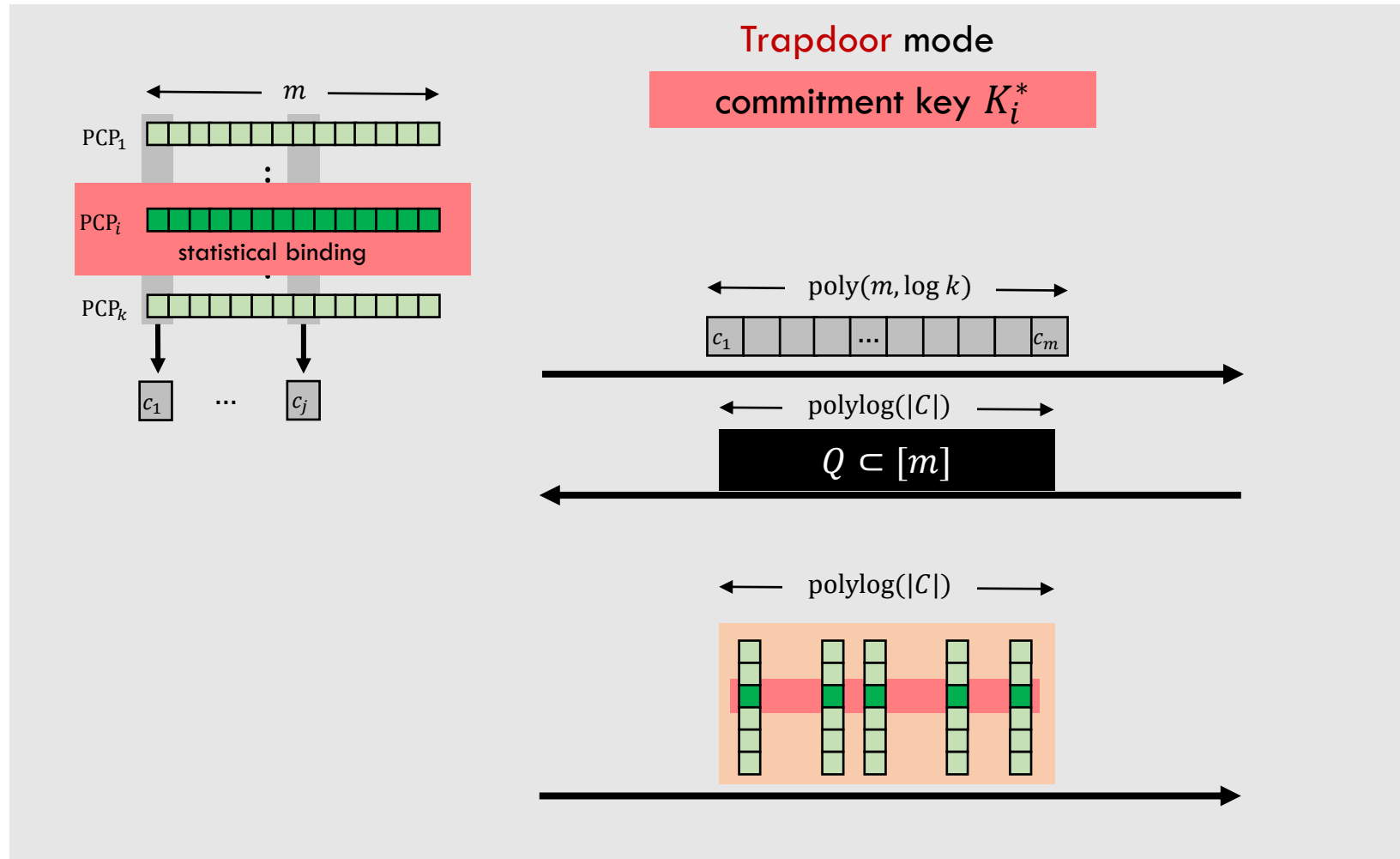
$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

Verify:

1. Commitment openings are valid.
2. PCP responses verify on Q

Dual Mode Argument for Batch Index



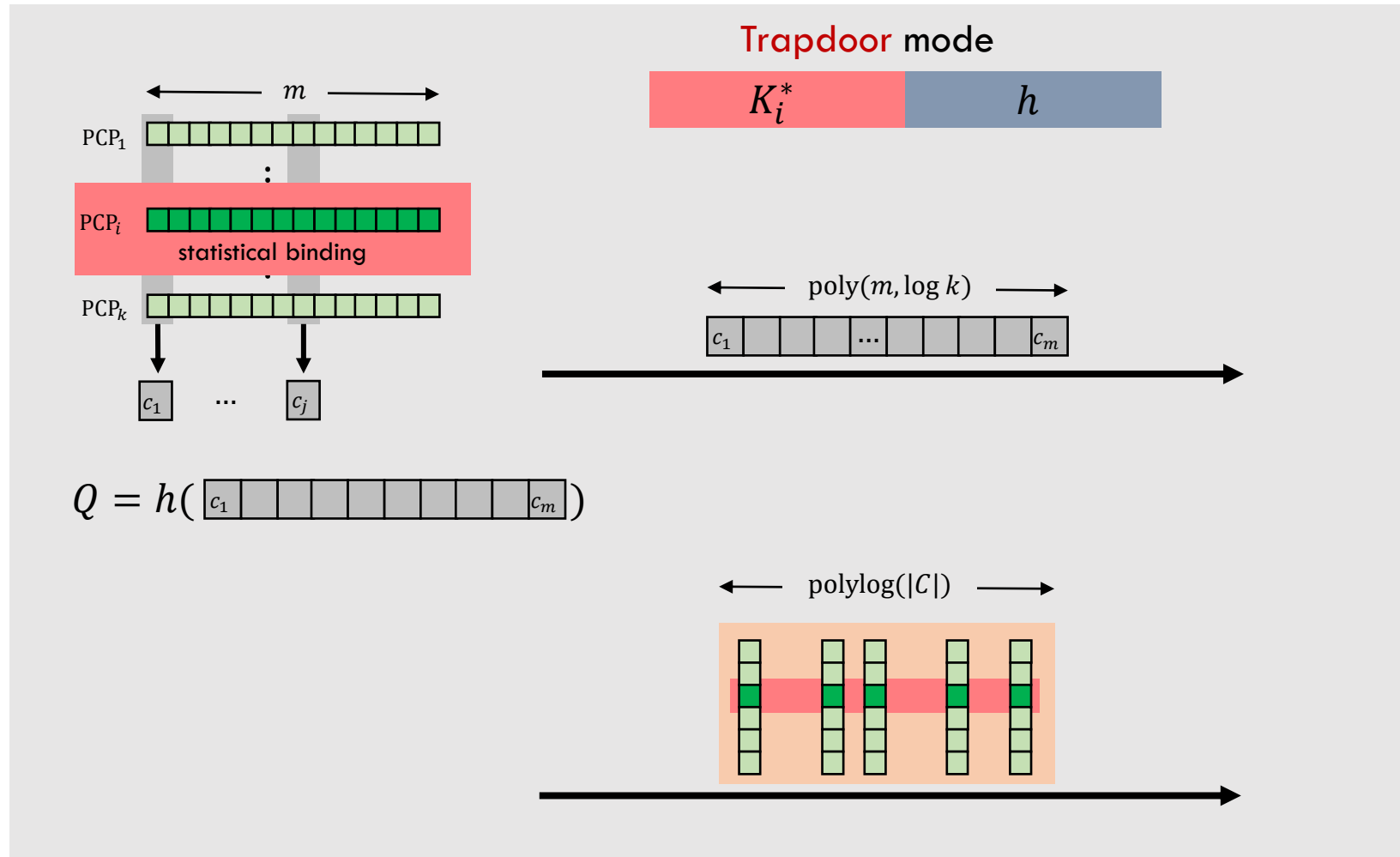
$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

Verify:

1. Commitment openings are valid.
2. PCP responses verify on Q

Dual Mode Argument for Batch Index



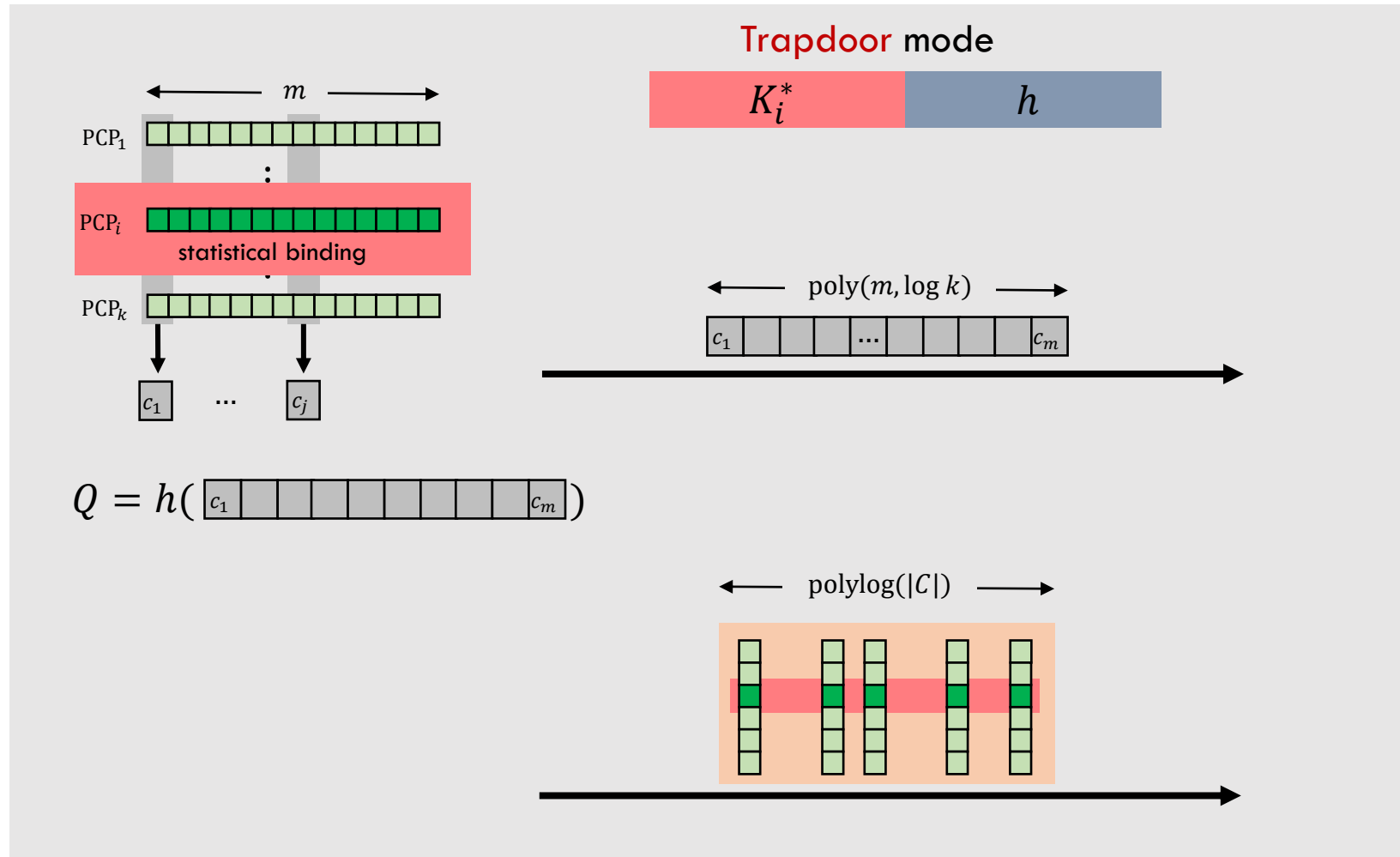
$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

Verify:

1. Commitment openings are valid.
2. PCP responses verify on Q

Dual Mode Argument for Batch Index



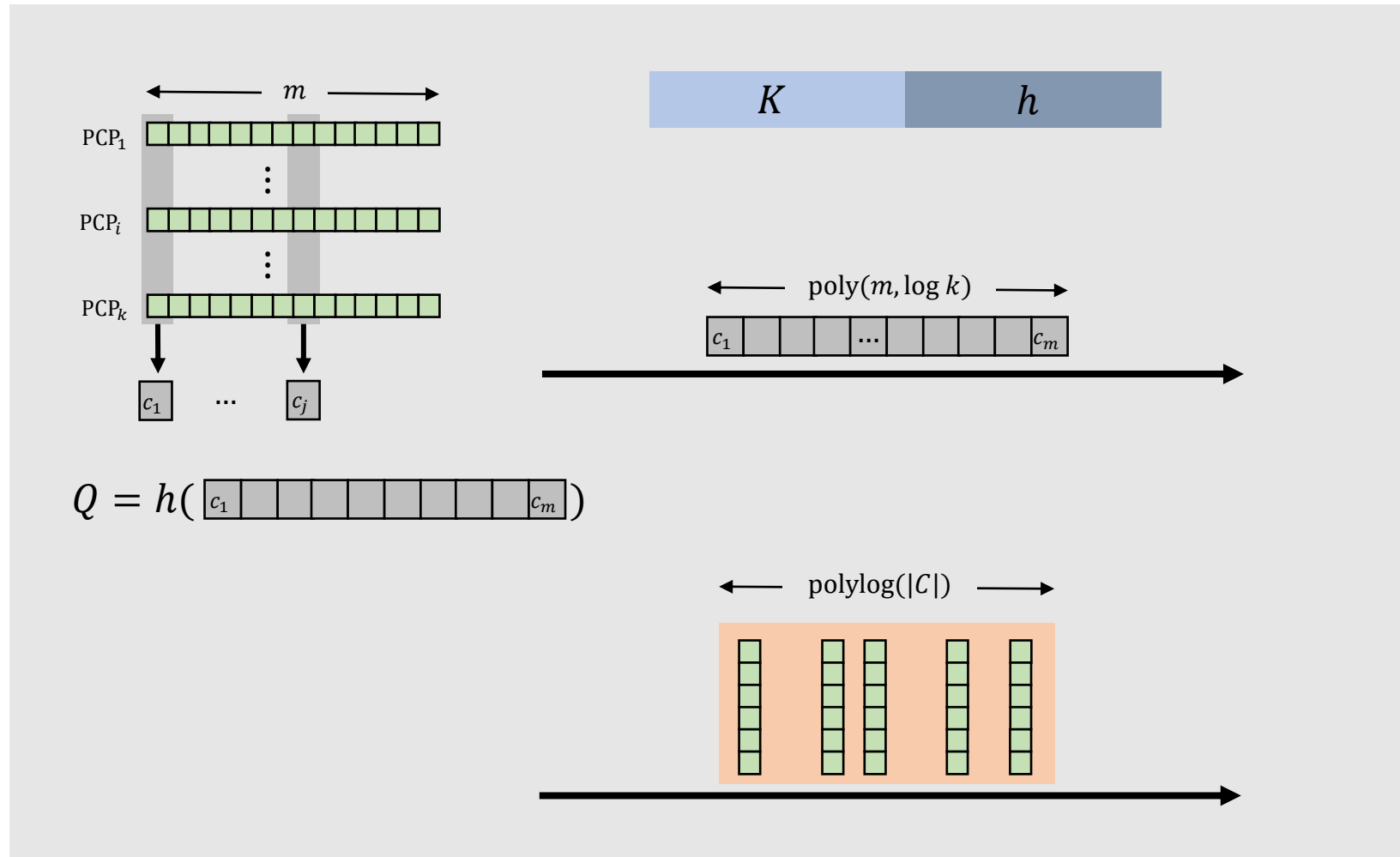
$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

[Holmgren-Lombardi-Rothblum'21]
Assuming **LWE**, the transformation is sound.

- Verify:
1. Commitment openings are valid.
 2. PCP responses verify on Q

SNARG for Batch Index



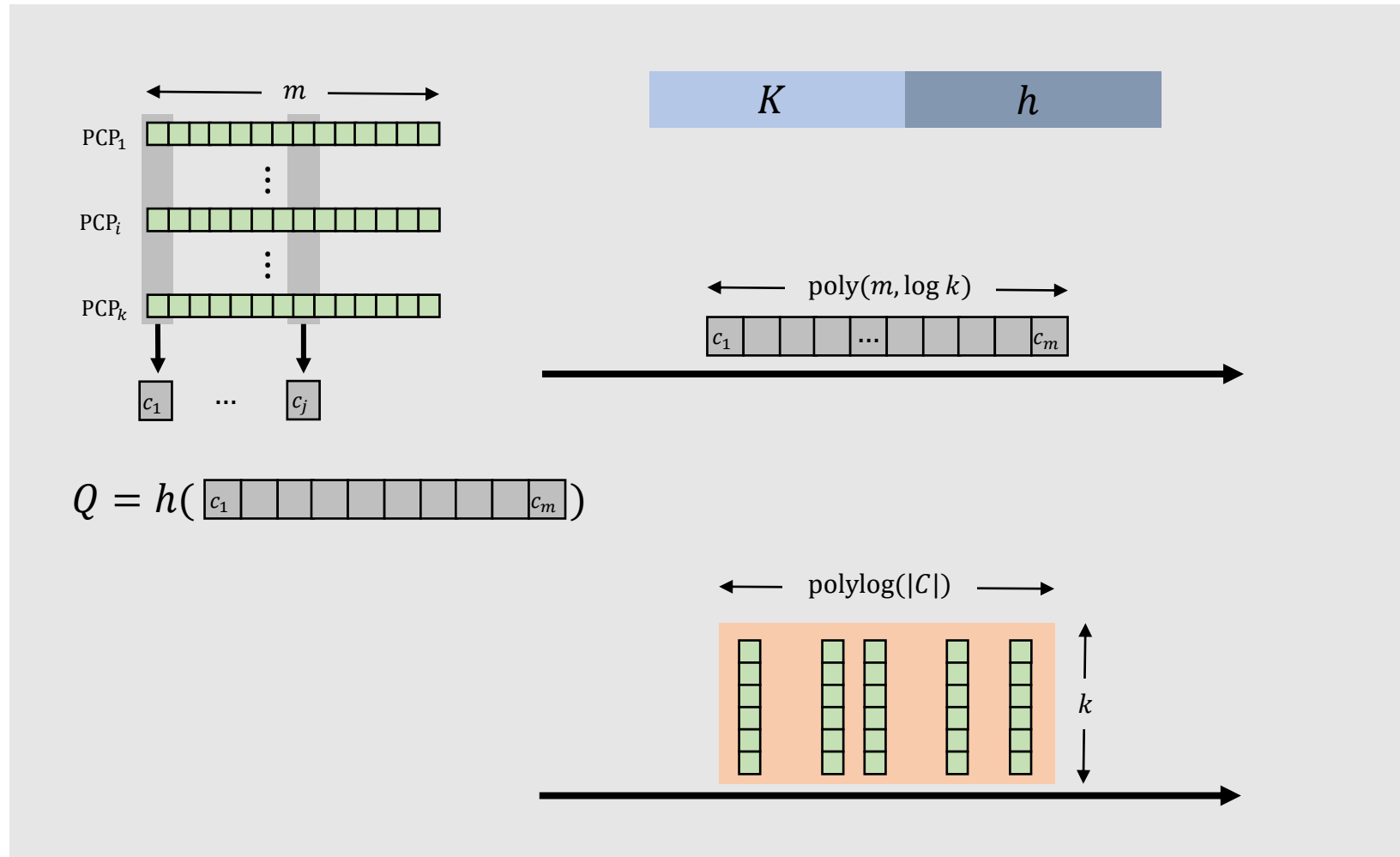
$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

Verify:

1. Commitment openings are valid.
2. PCP responses verify on Q

SNARG for Batch Index



$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

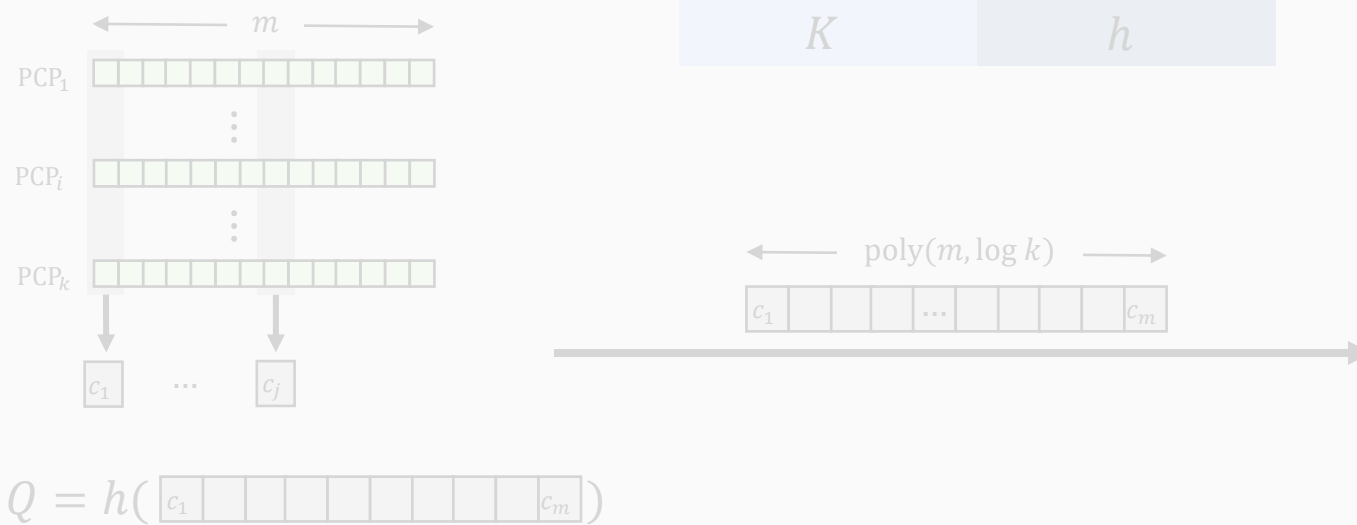
$$\forall i \in [k], i \in L_C$$

$$|\text{Verifier}| = \Omega(k)$$

Verify:

1. Commitment openings are valid.
2. PCP responses verify on Q

SNARG for Batch Index



Send **proof of verification**

$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

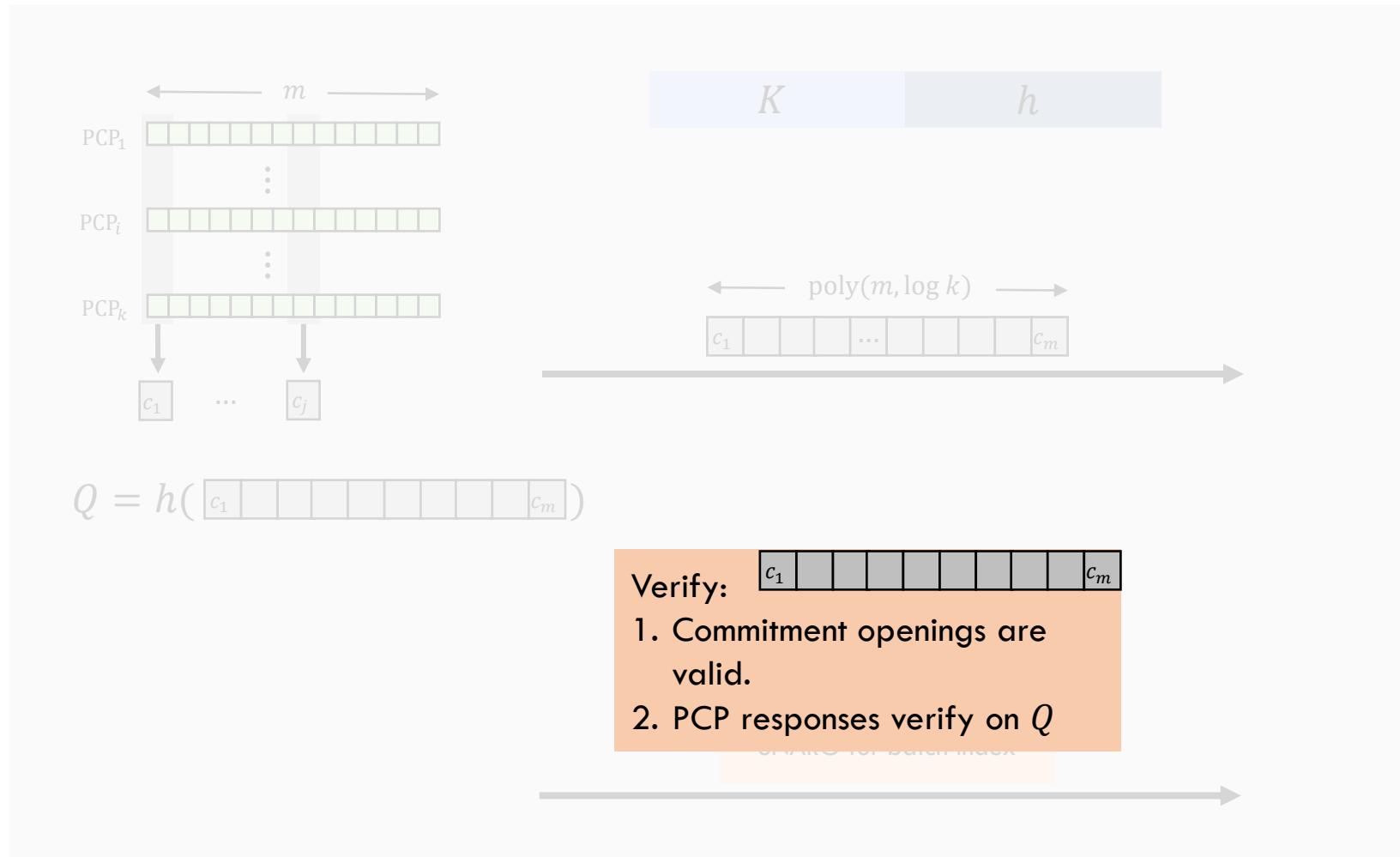
$$\forall i \in [k], i \in L_C$$

$$|\text{Verifier}| = \Omega(k)$$

Verify:

1. Commitment openings are valid.
2. PCP responses verify on Q

SNARG for Batch Index

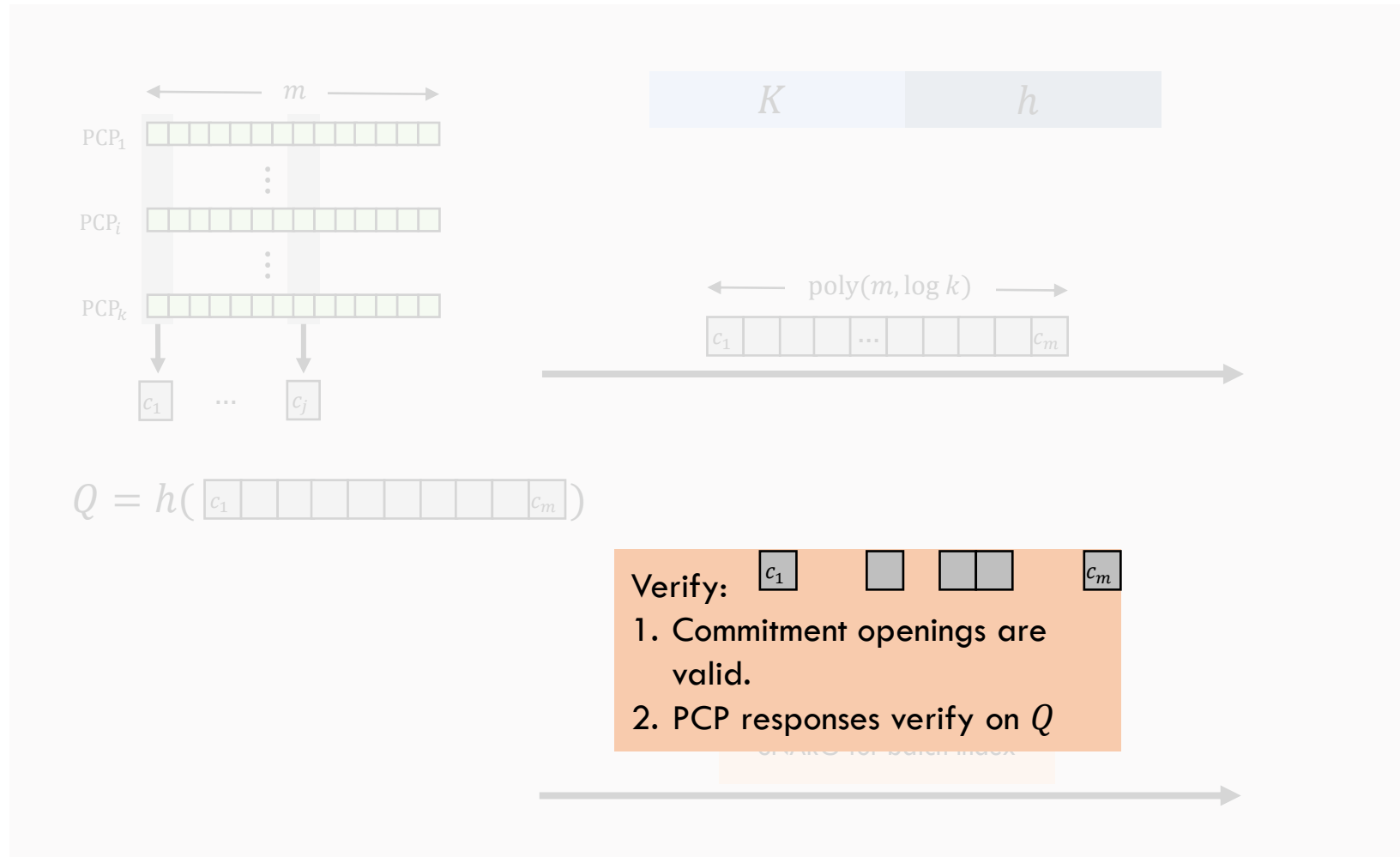


$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

$$|\text{Verifier}| = \Omega(k)$$

SNARG for Batch Index

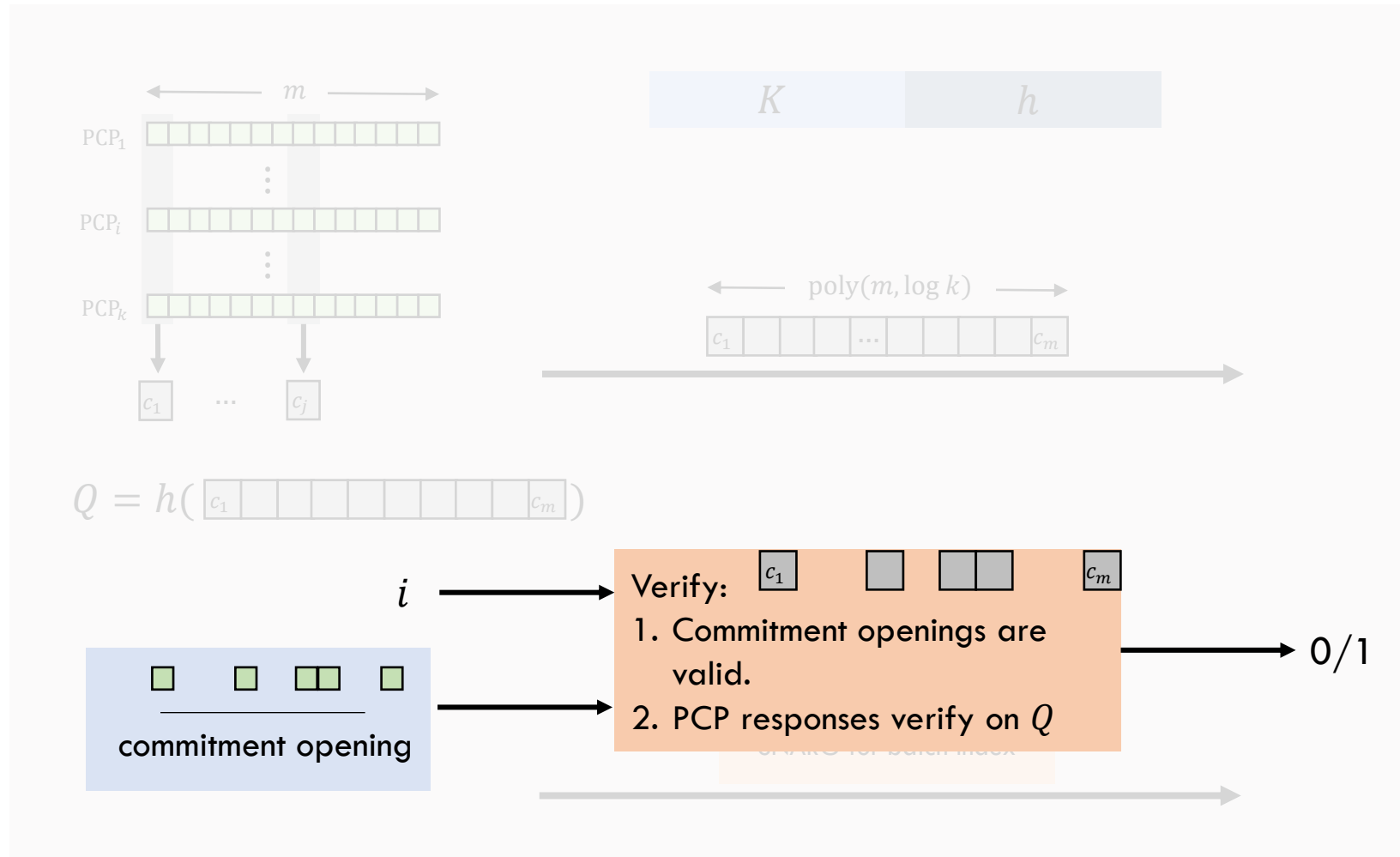


$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

$$|\text{Verifier}| = \Omega(k)$$

SNARG for Batch Index

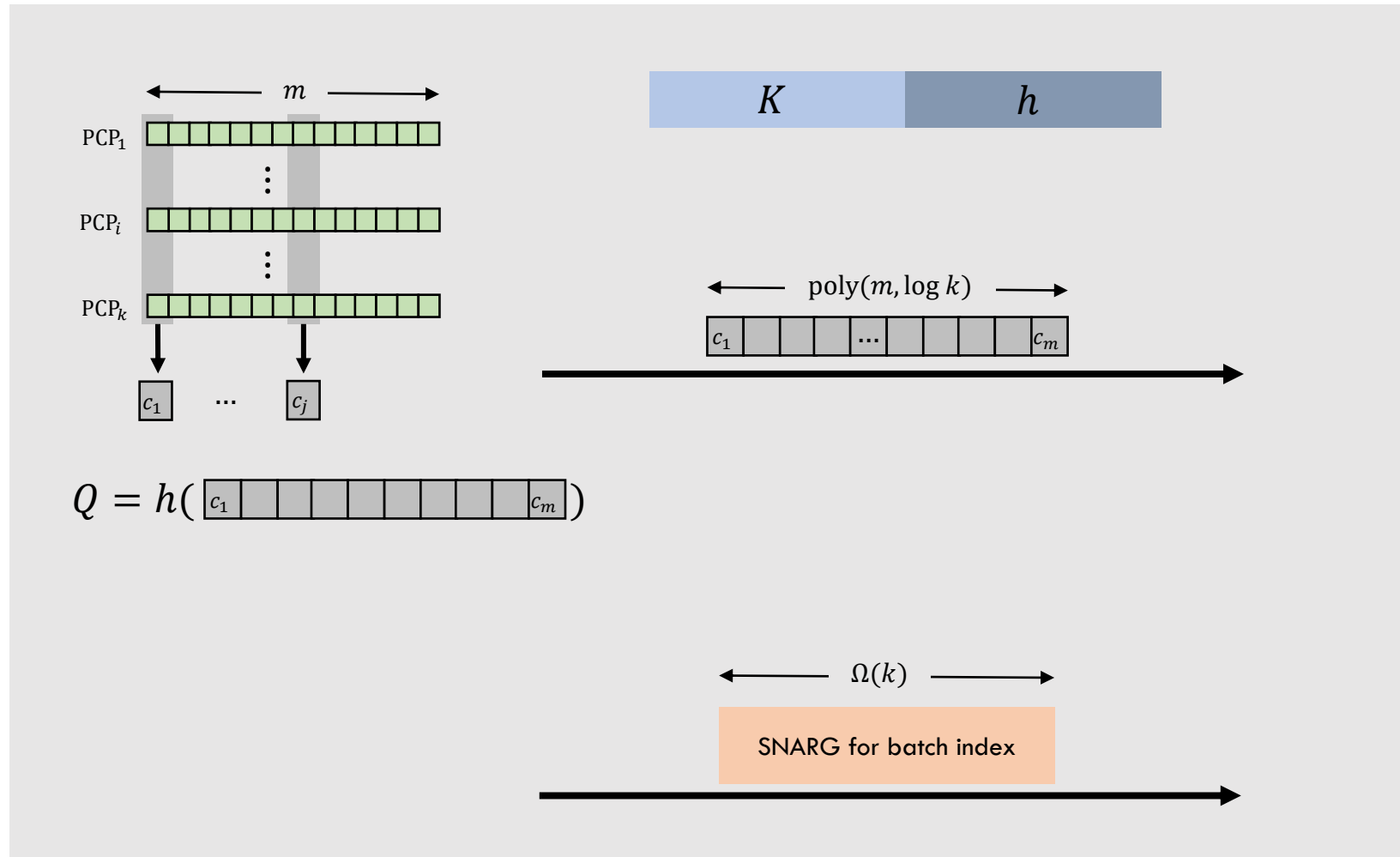


$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

$$|\text{person icon}| = \Omega(k)$$

SNARG for Batch Index



$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

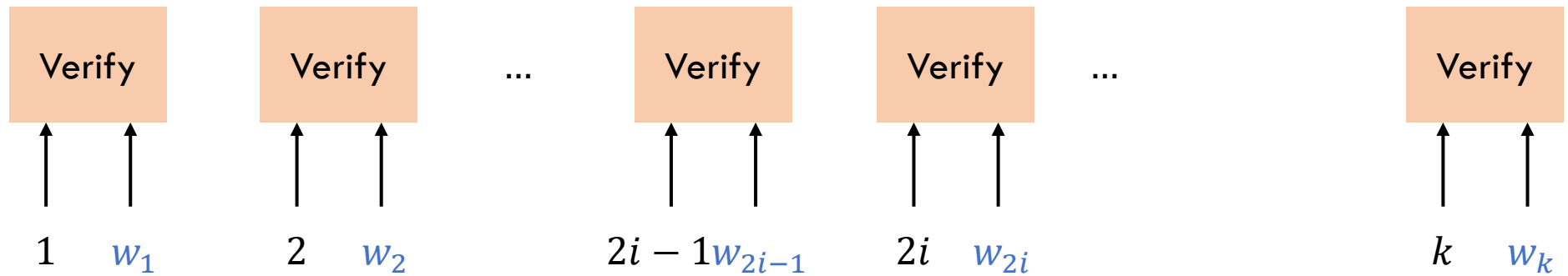
$$|\text{Verifier}| = \Omega(k)$$

Verify:

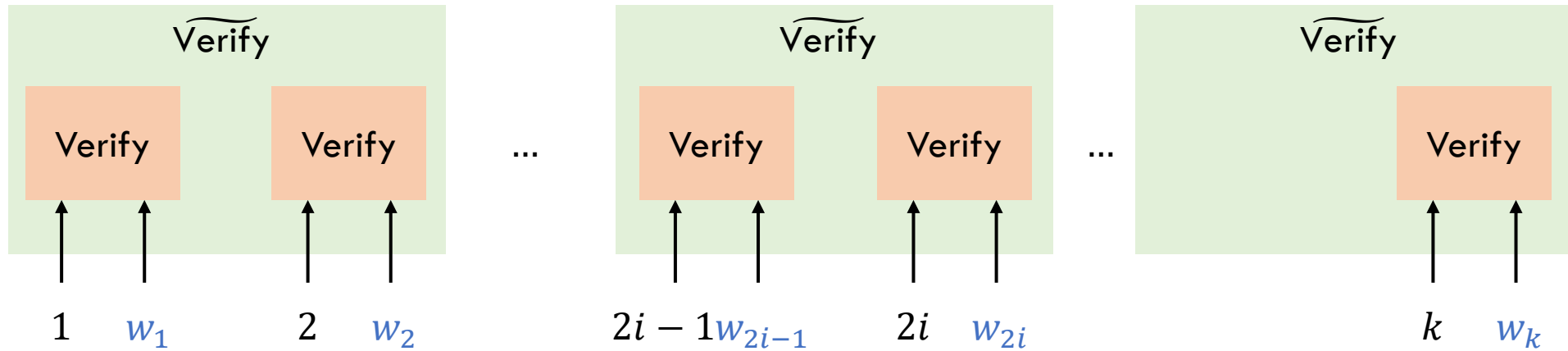


1. Commitment openings are valid.
2. PCP responses verify on Q

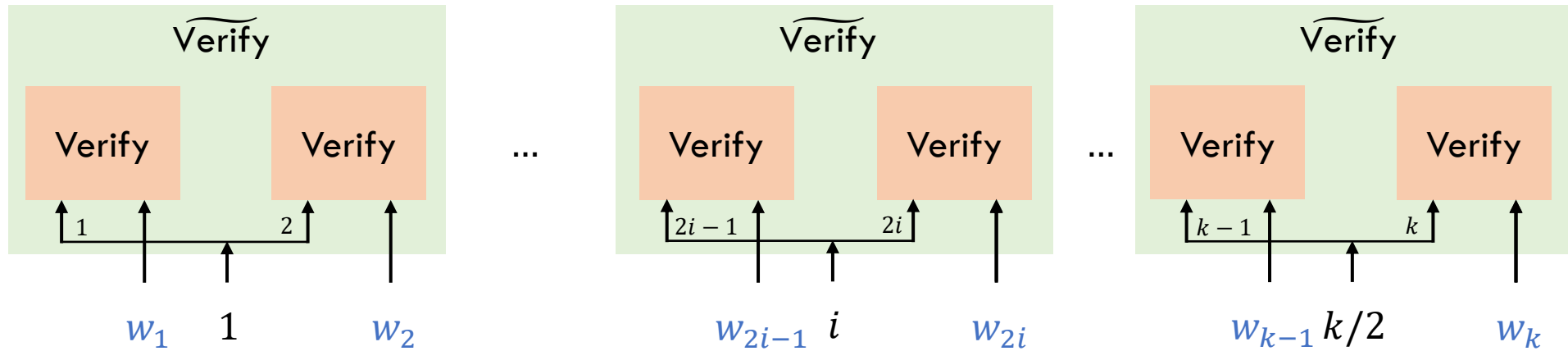
Grouping Instances for Recursion



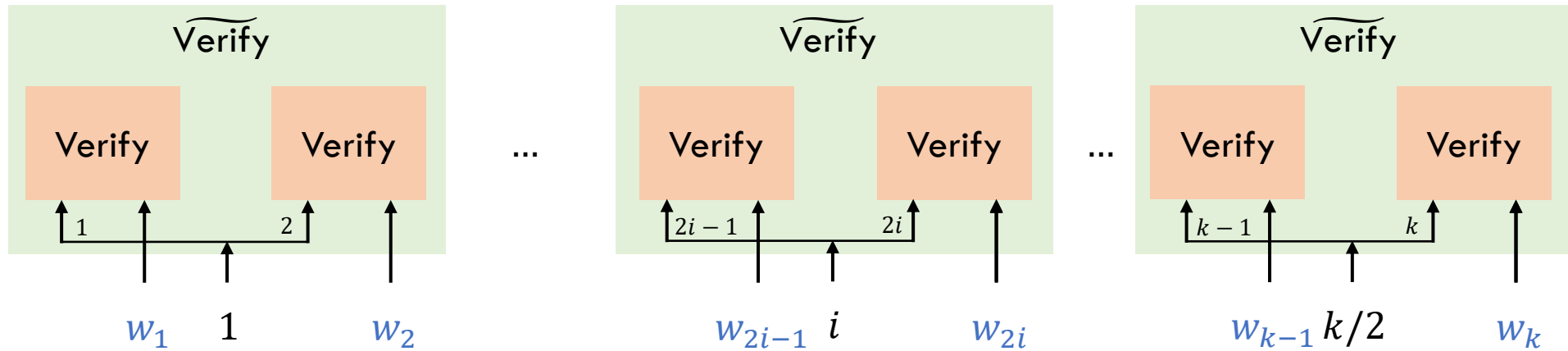
Grouping Instances for Recursion



Grouping Instances for Recursion



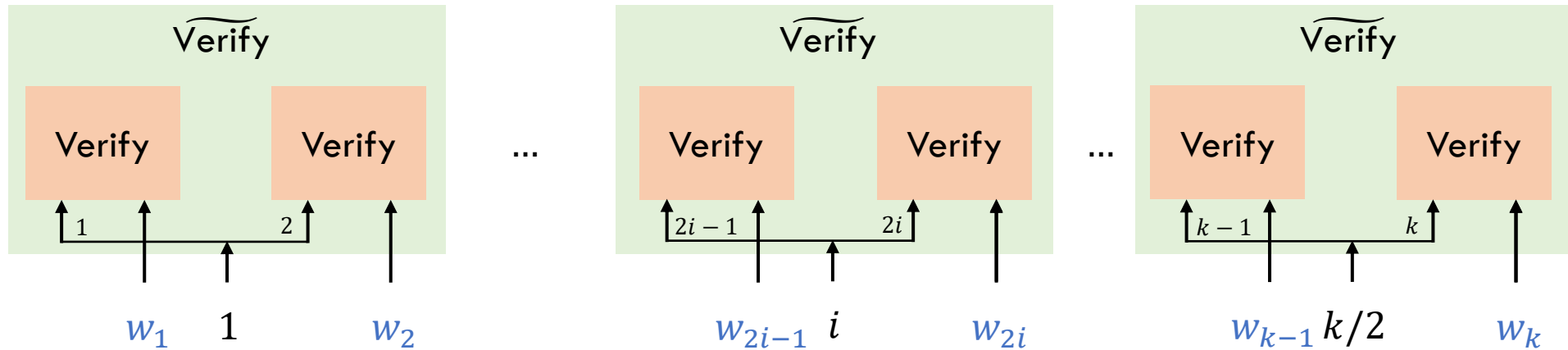
Grouping Instances for Recursion



$k/2$ instances for circuit $\widetilde{\text{Verify}}$

Grouping Instances for Recursion

$$|\widetilde{\text{Verify}}| \geq 2|\text{Verify}|$$



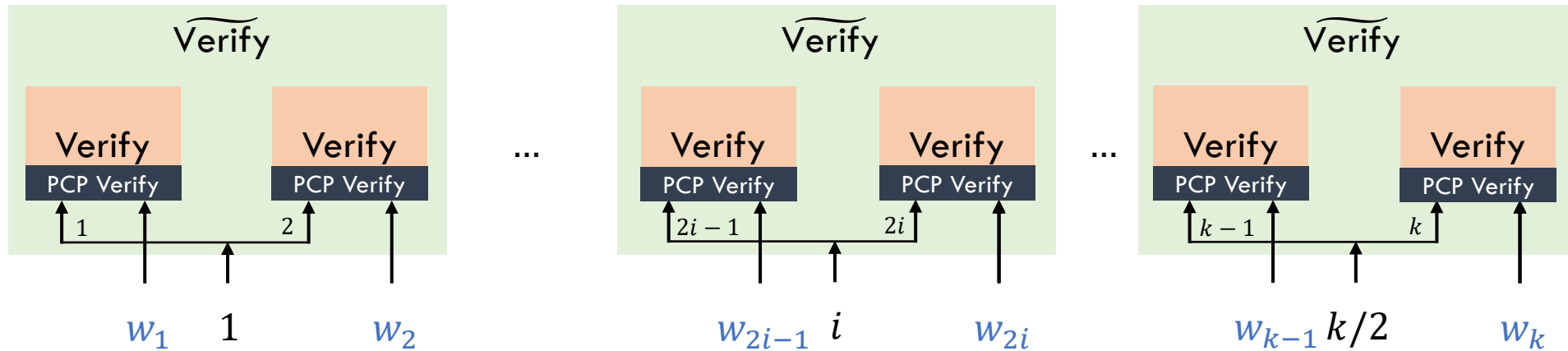
$k/2$ instances for circuit $\widetilde{\text{Verify}}$

Grouping Instances for Recursion

$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

$$|\widetilde{\text{Verify}}| \geq 2|\text{Verify}|$$



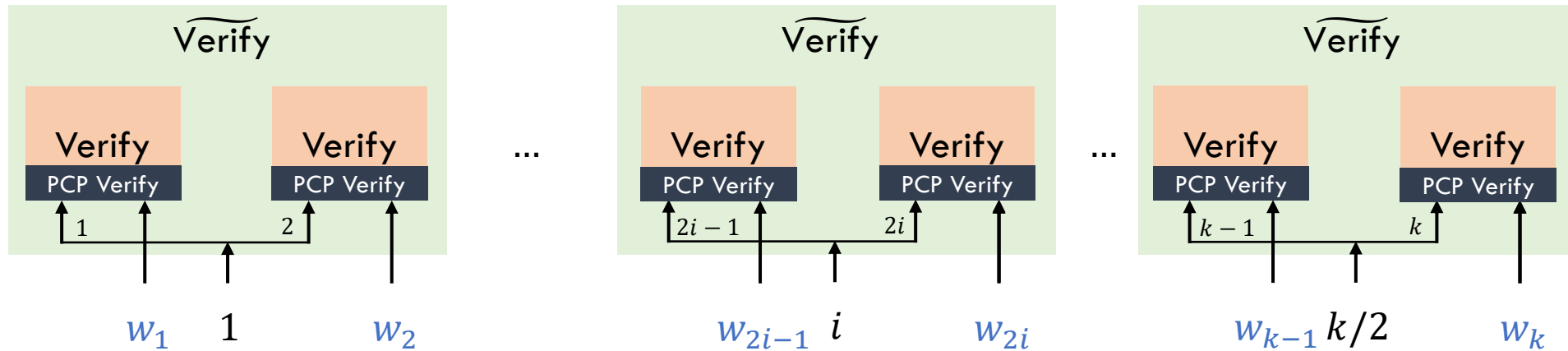
$k/2$ instances for circuit $\widetilde{\text{Verify}}$

Grouping Instances for Recursion

$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

$$|\widetilde{\text{Verify}}| \geq 2|\text{Verify}| \geq 2|C|$$



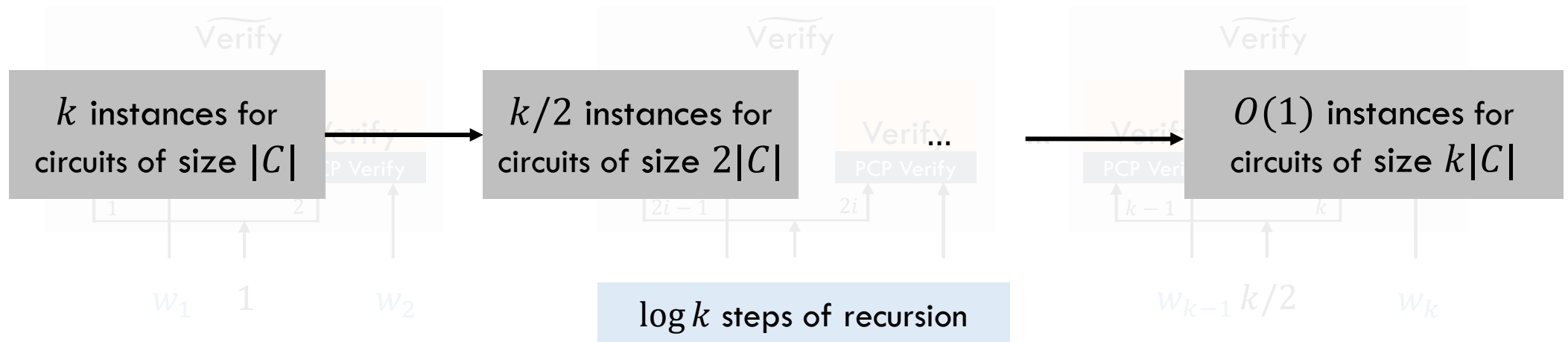
$k/2$ instances for circuit $\widetilde{\text{Verify}}$

Grouping Instances for Recursion

$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

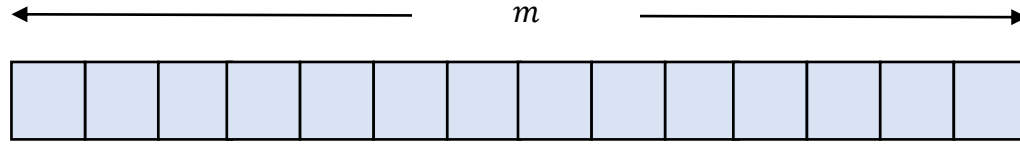
$$|\widetilde{\text{Verify}}| \geq 2|\text{Verify}| \geq 2|C|$$



Tool: PCP with Fast Online Verification

C, x, w

PCP

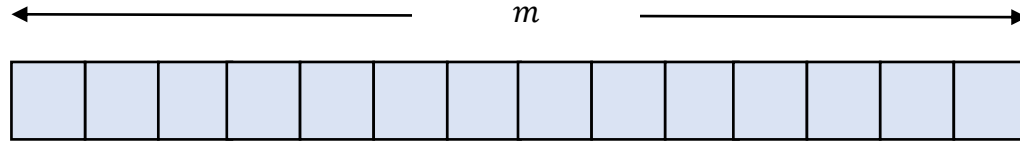


C, x

Tool: PCP with Fast Online Verification

C, x, w

PCP



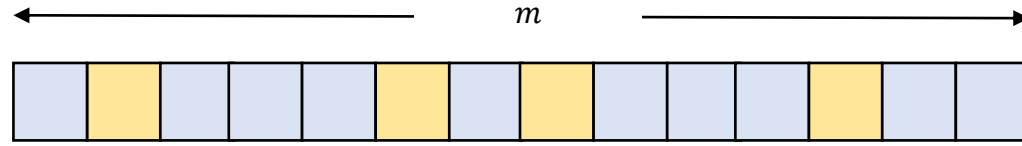
C, x

$Q \leftarrow [m]$

Tool: PCP with Fast Online Verification

C, x, w

PCP

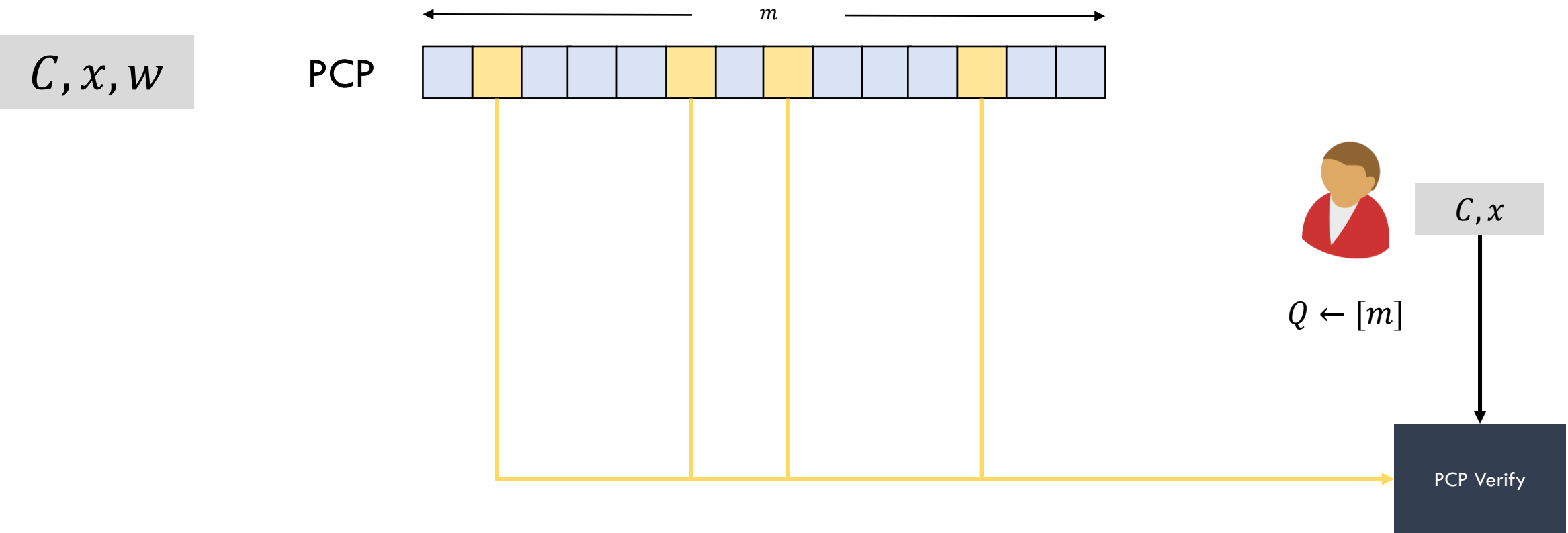


C, x

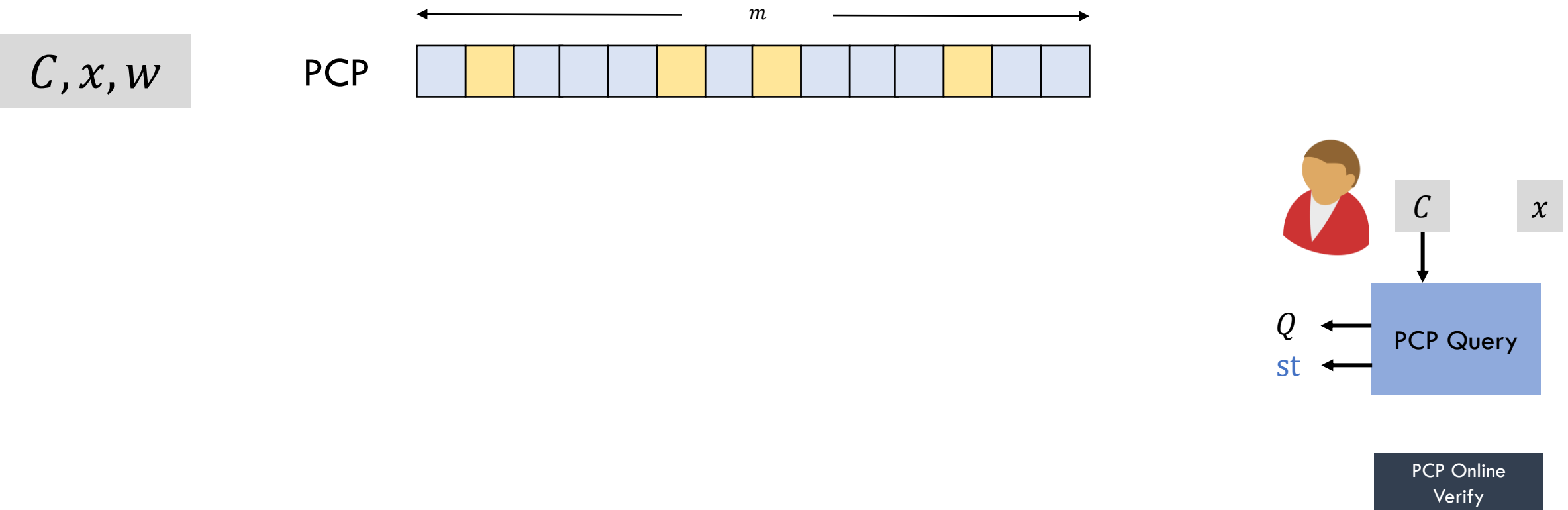
$Q \leftarrow [m]$

PCP Verify

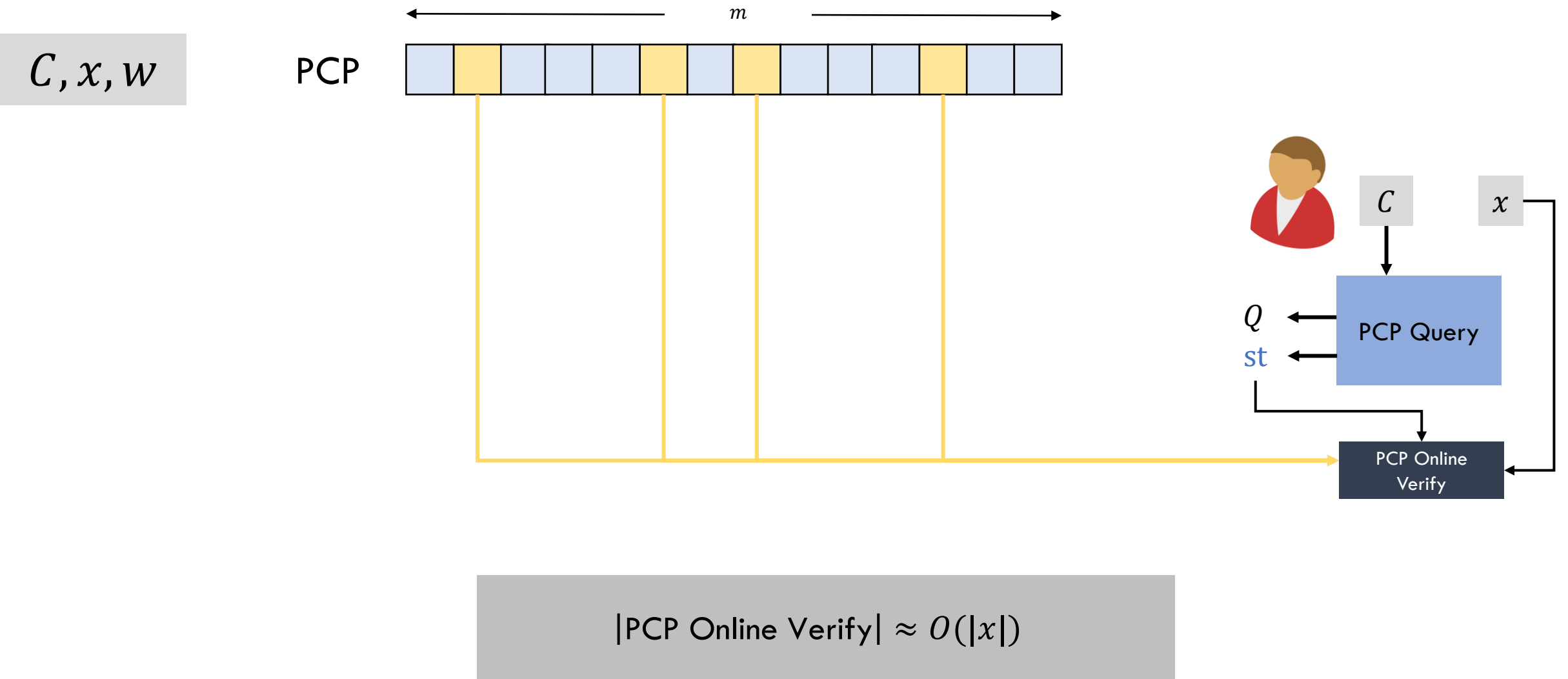
Tool: PCP with Fast Online Verification



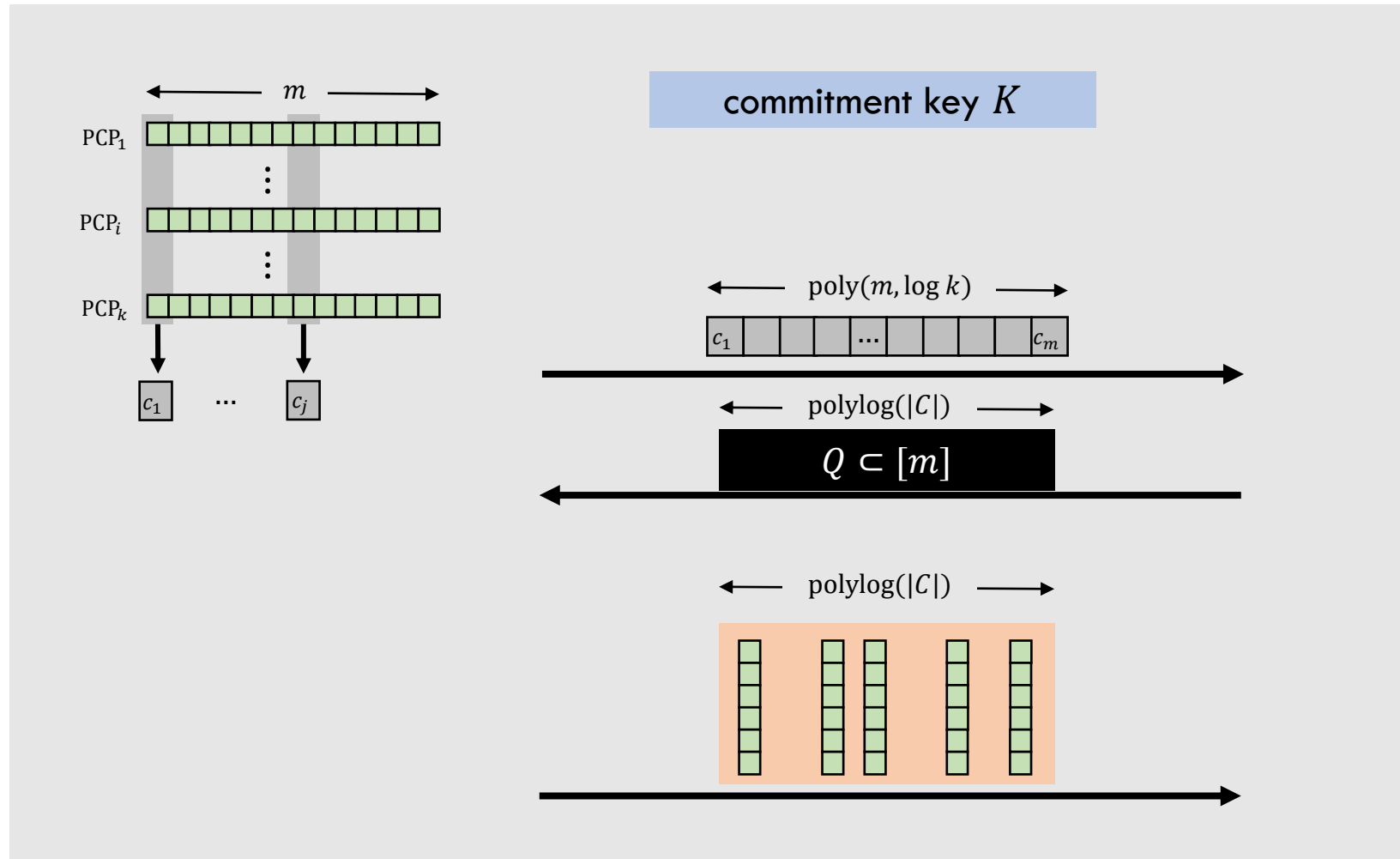
Tool: PCP with Fast Online Verification



Tool: PCP with Fast Online Verification



Dual Mode Argument for Batch Index



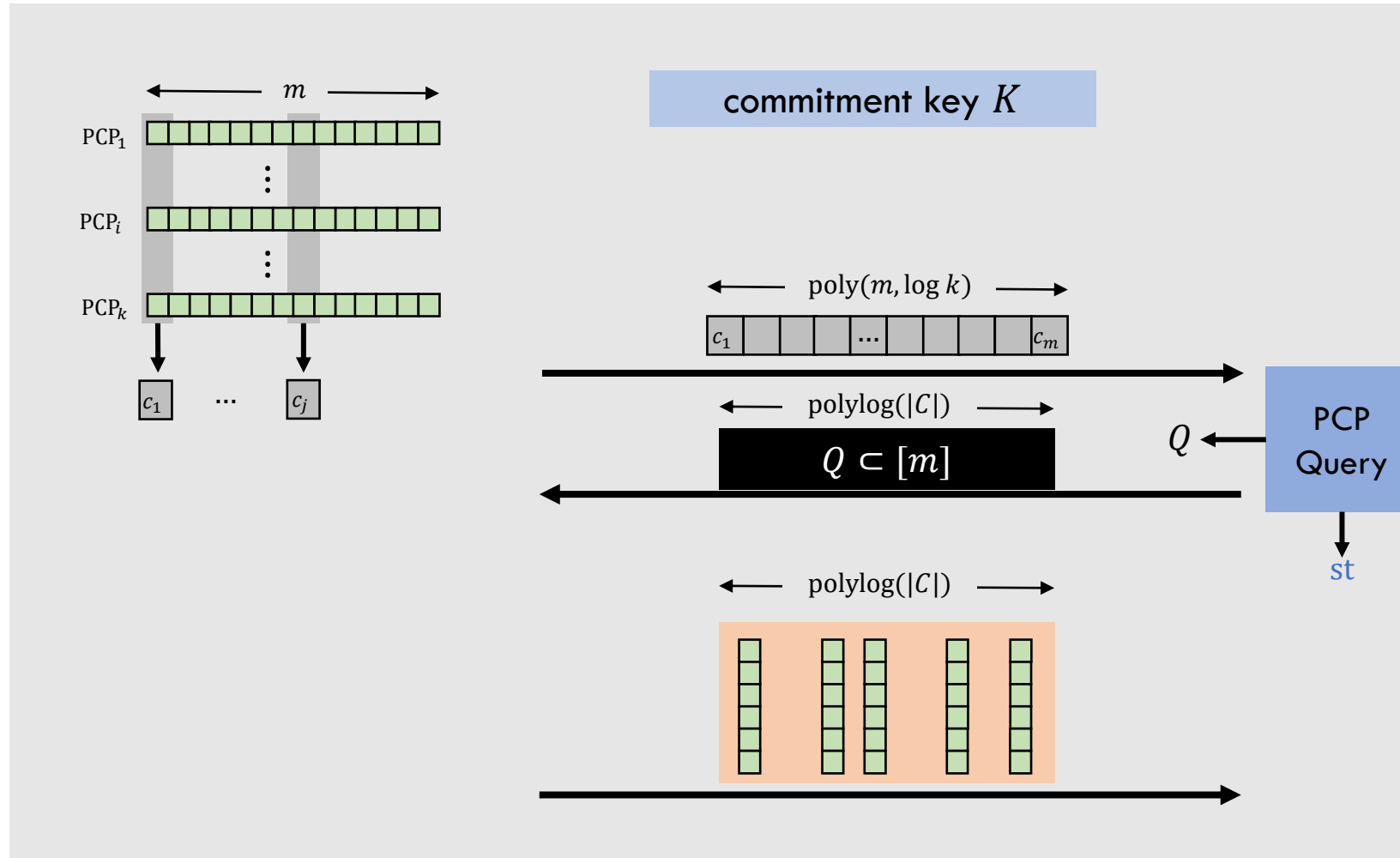
$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

Verify:

1. Commitment openings are valid.
2. PCP responses verify on Q

Dual Mode Argument for Batch Index



$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

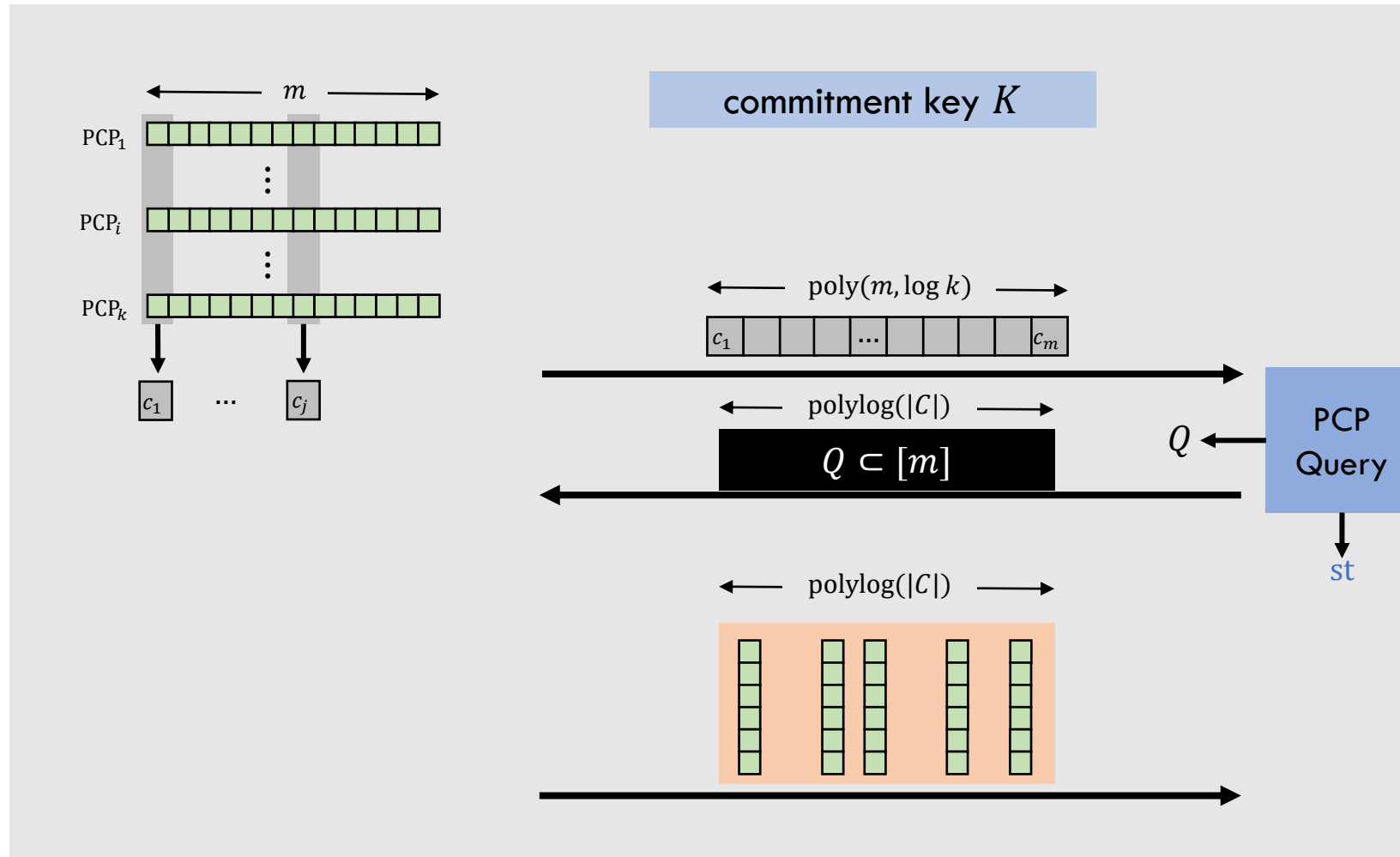
$$\forall i \in [k], i \in L_C$$

Verify:

1. Commitment openings are valid.
2. PCP responses verify on Q, st

PCP  Verify

Dual Mode Argument for Batch Index



$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

Verify:

PCP  Verify

1. Commitment openings are valid.
2. PCP responses verify on Q, st

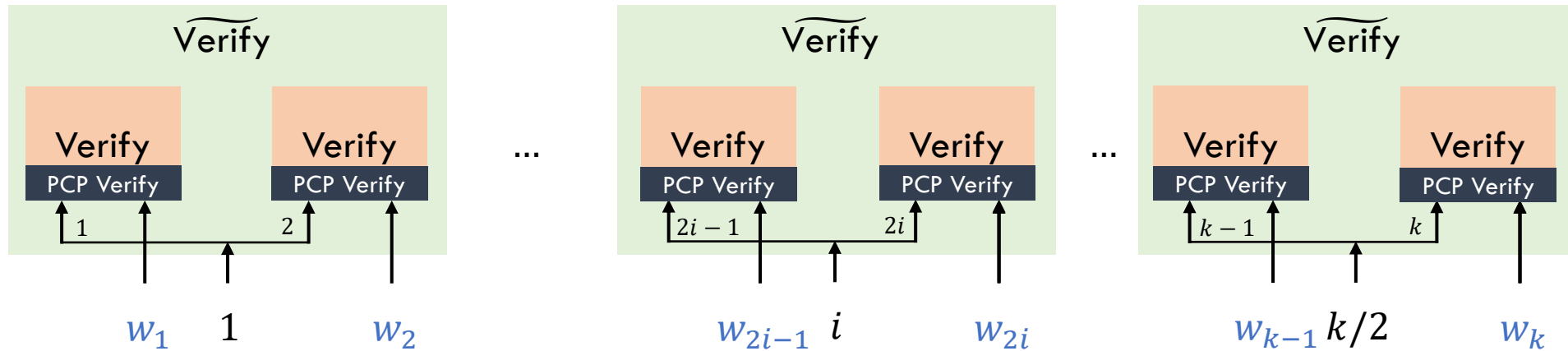
$$|\text{Verify}| \approx \text{polylog}(k, |C|)$$

Grouping Instances for Recursion

$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

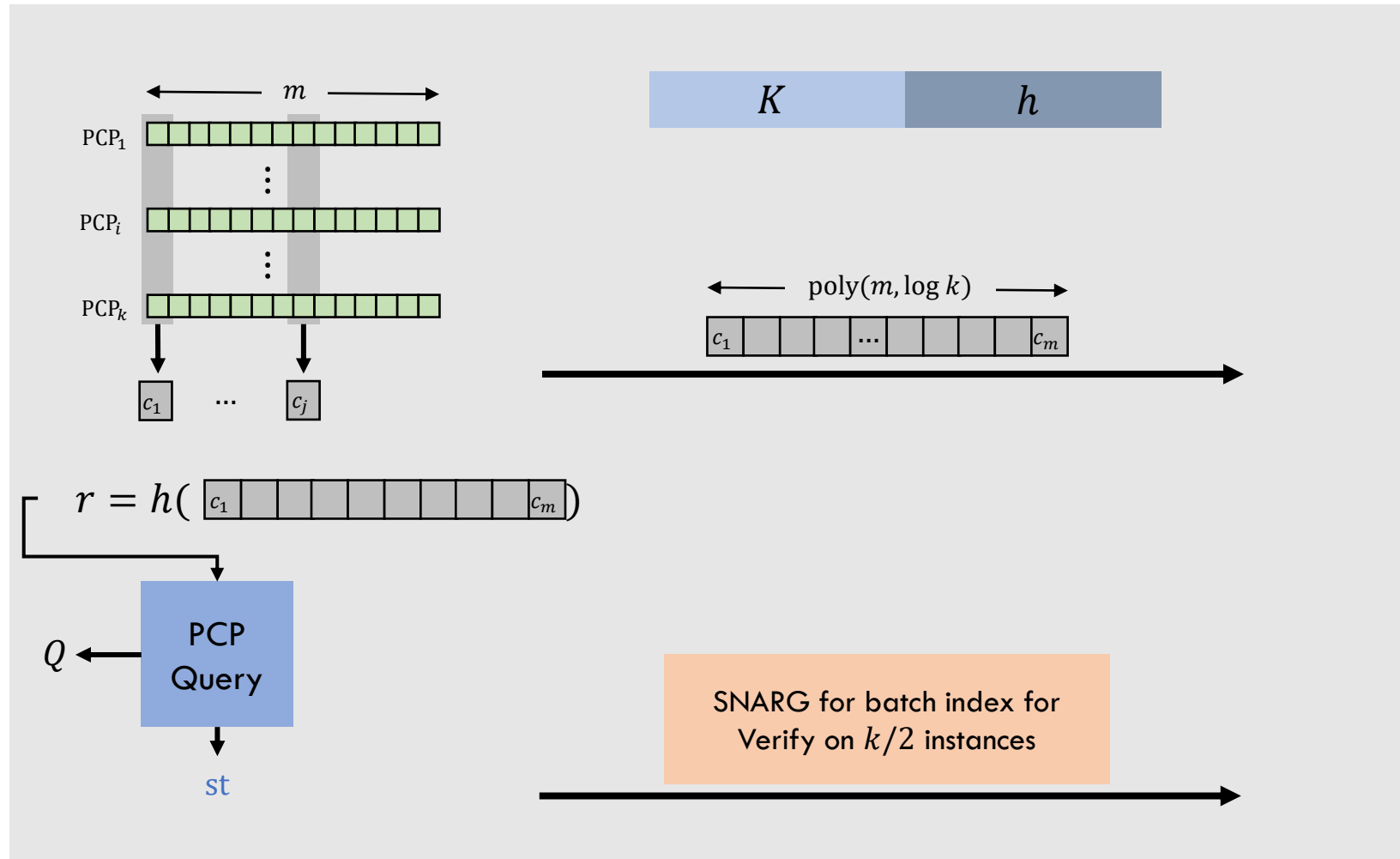
$$\forall i \in [k], i \in L_C$$

$$|\widetilde{\text{Verify}}| \geq 2|\text{Verify}| \geq 2 \text{polylog}(k, |C|)$$



$k/2$ instances for circuit $\widetilde{\text{Verify}}$

SNARG for Batch Index



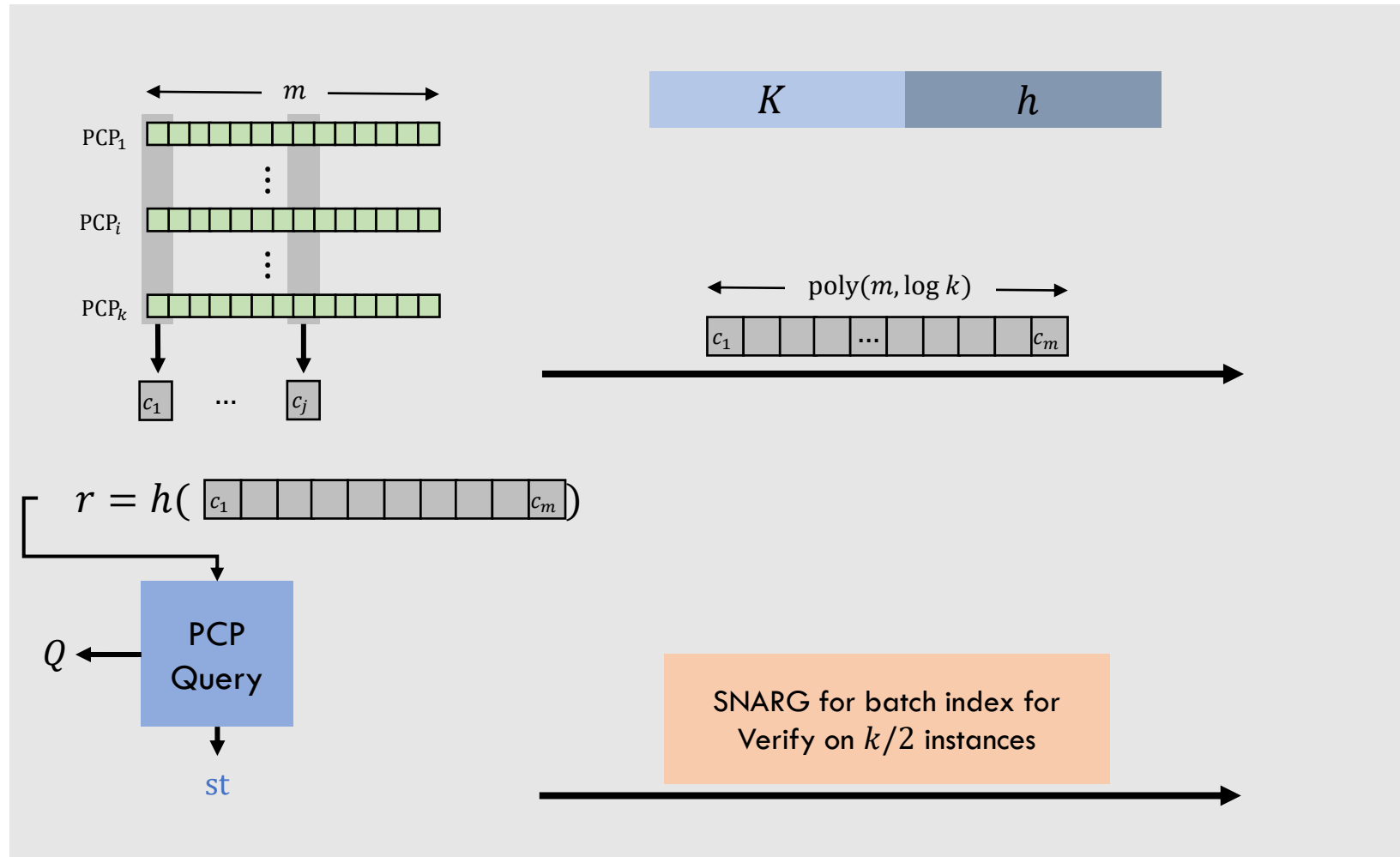
$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

$$\forall i \in [k], i \in L_C$$

Verify: c_1 \square \square c_m

1. Commitment openings are valid.
2. PCP responses verify on Q, st

SNARG for Batch Index



$$L_C = \{i \mid \exists w \text{ s.t. } C(i, w) = 1\}$$

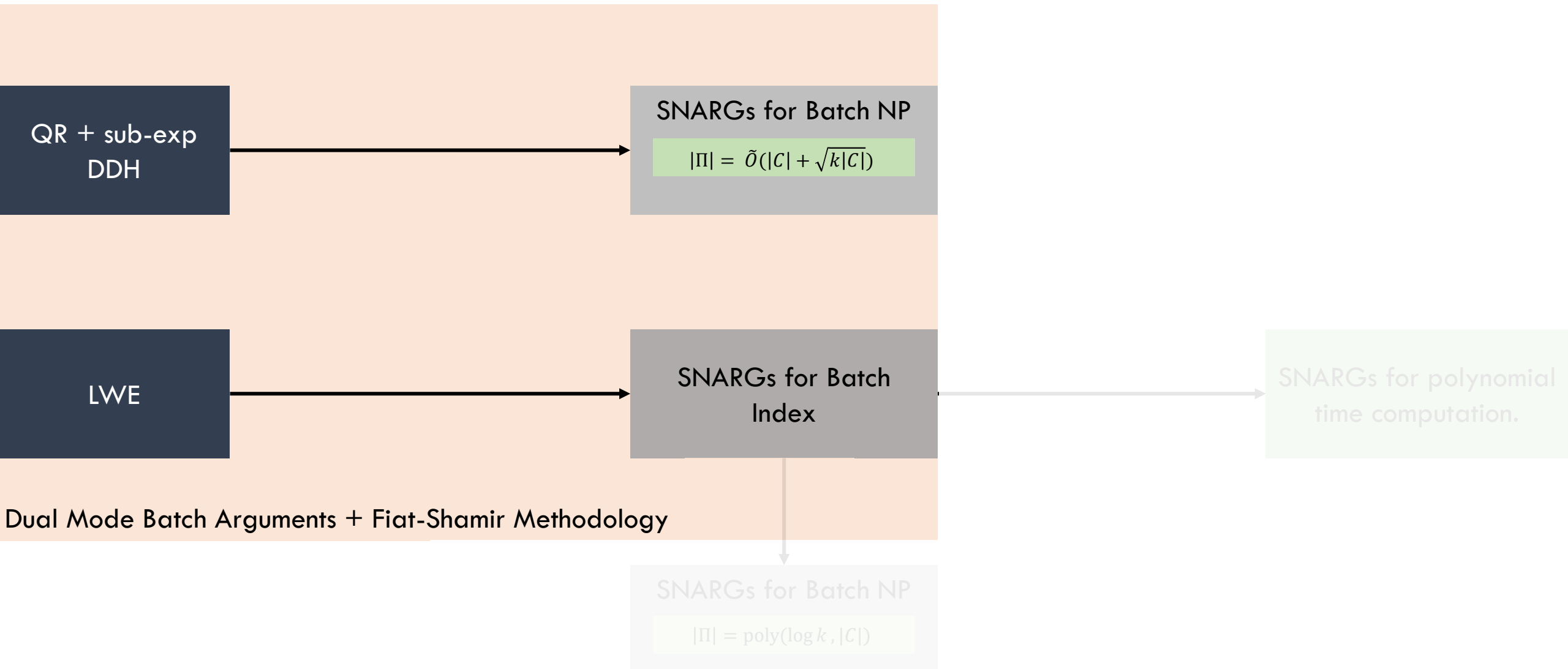
$$\forall i \in [k], i \in L_C$$

Recurse $\log k$ times

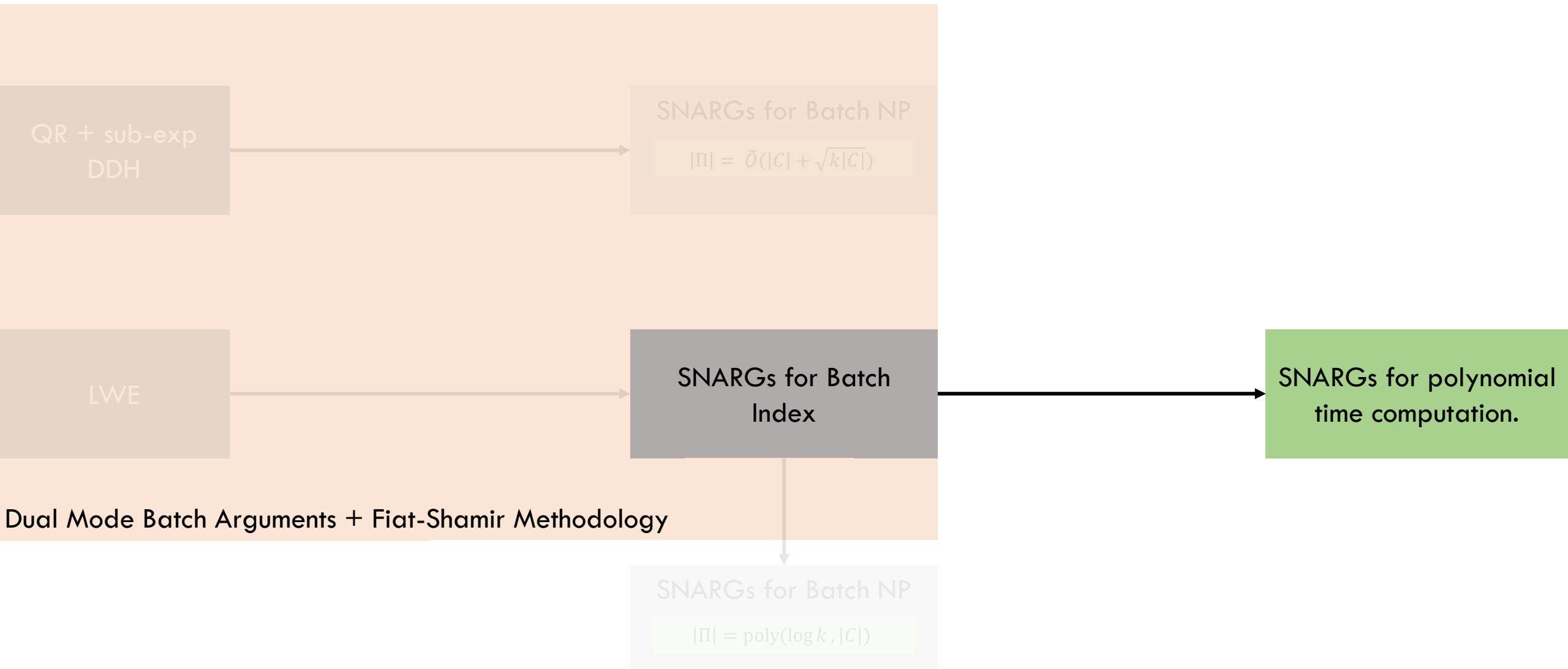
Verify: $[c_1, \dots, c_m]$

1. Commitment openings are valid.
2. PCP responses verify on Q, st

Results Overview

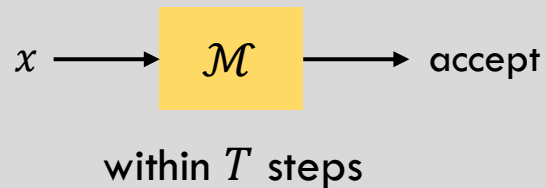
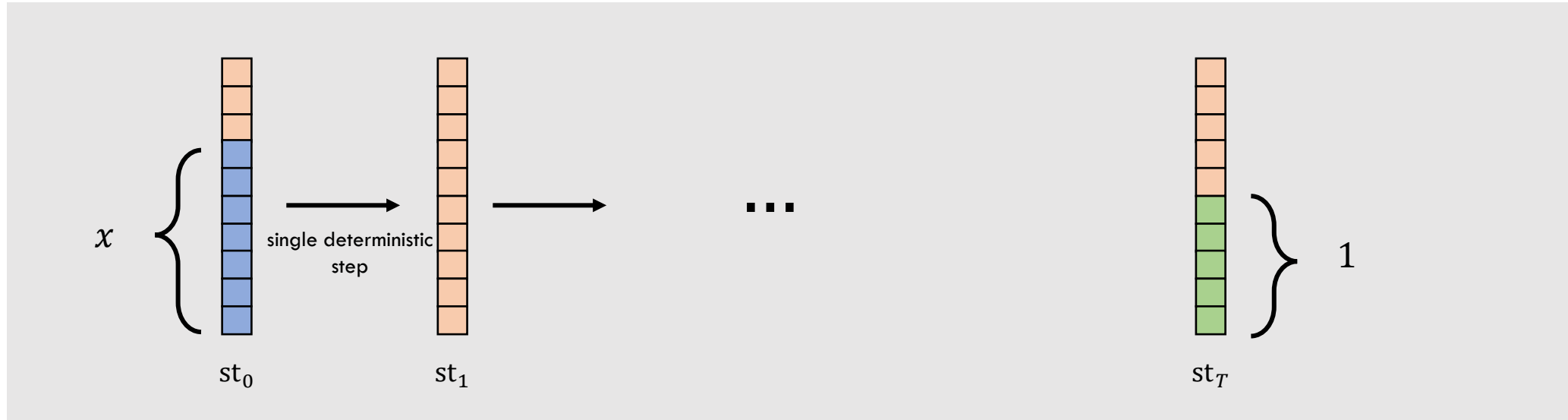


Results Overview



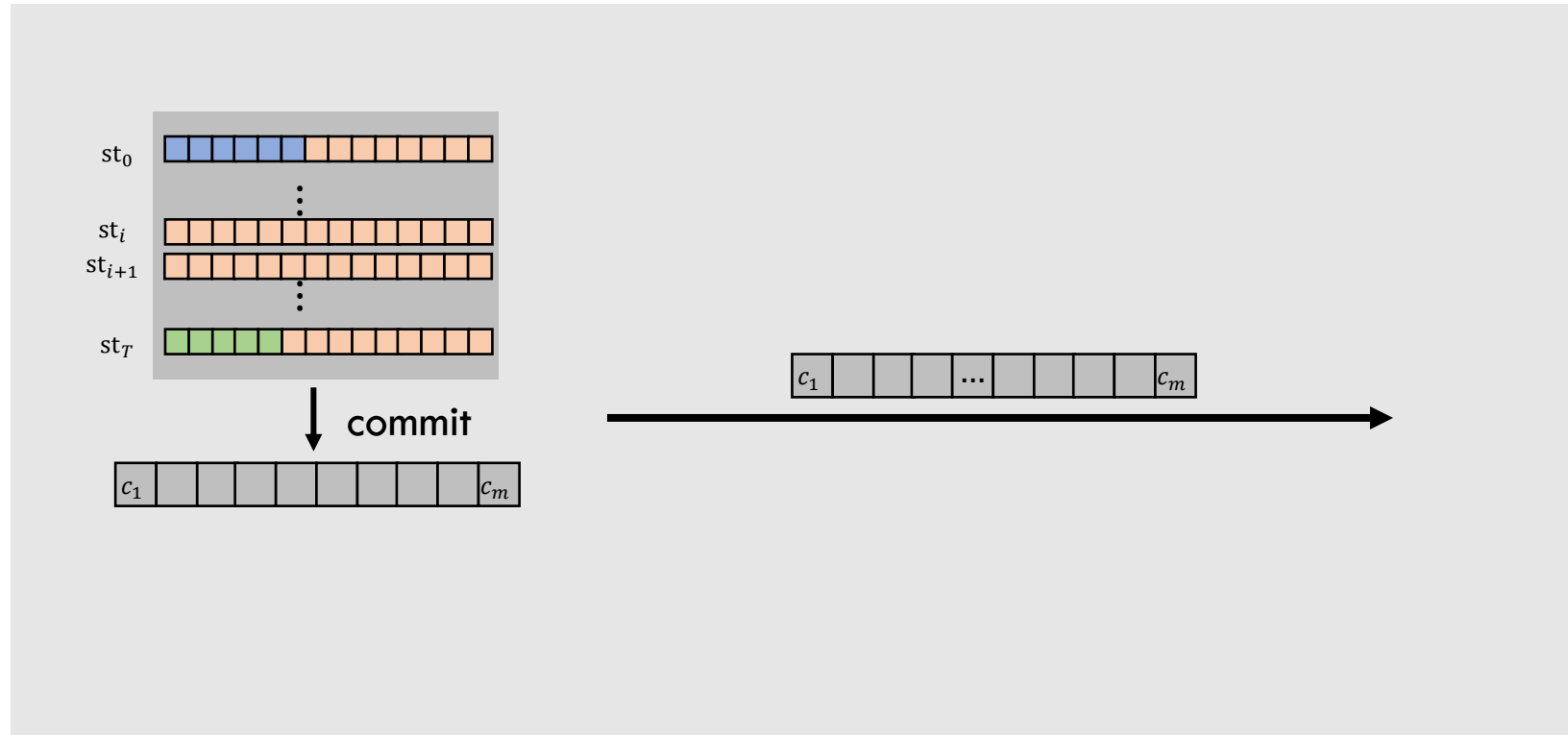
SNARGs for Batch Index \rightarrow SNARGs for P

Delegation via **Batching** [Reingold-Rothblum-Rothblum'16]

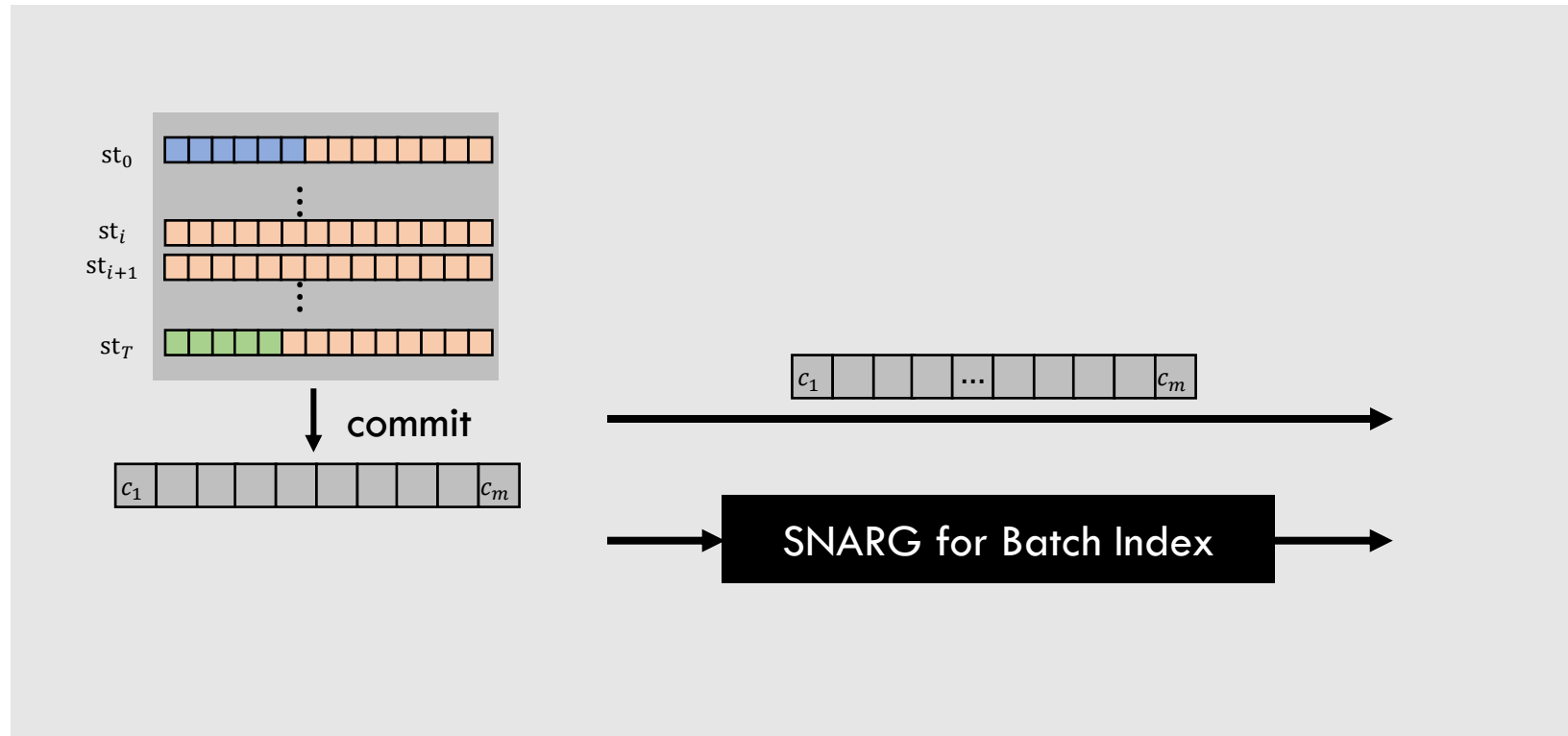


Prove for every $i \in [0, \dots, T - 1]$
 $st_i \rightarrow st_{i+1}$
is the correct transition.

SNARGs for Polynomial-time Computation



SNARGs for Polynomial-time Computation

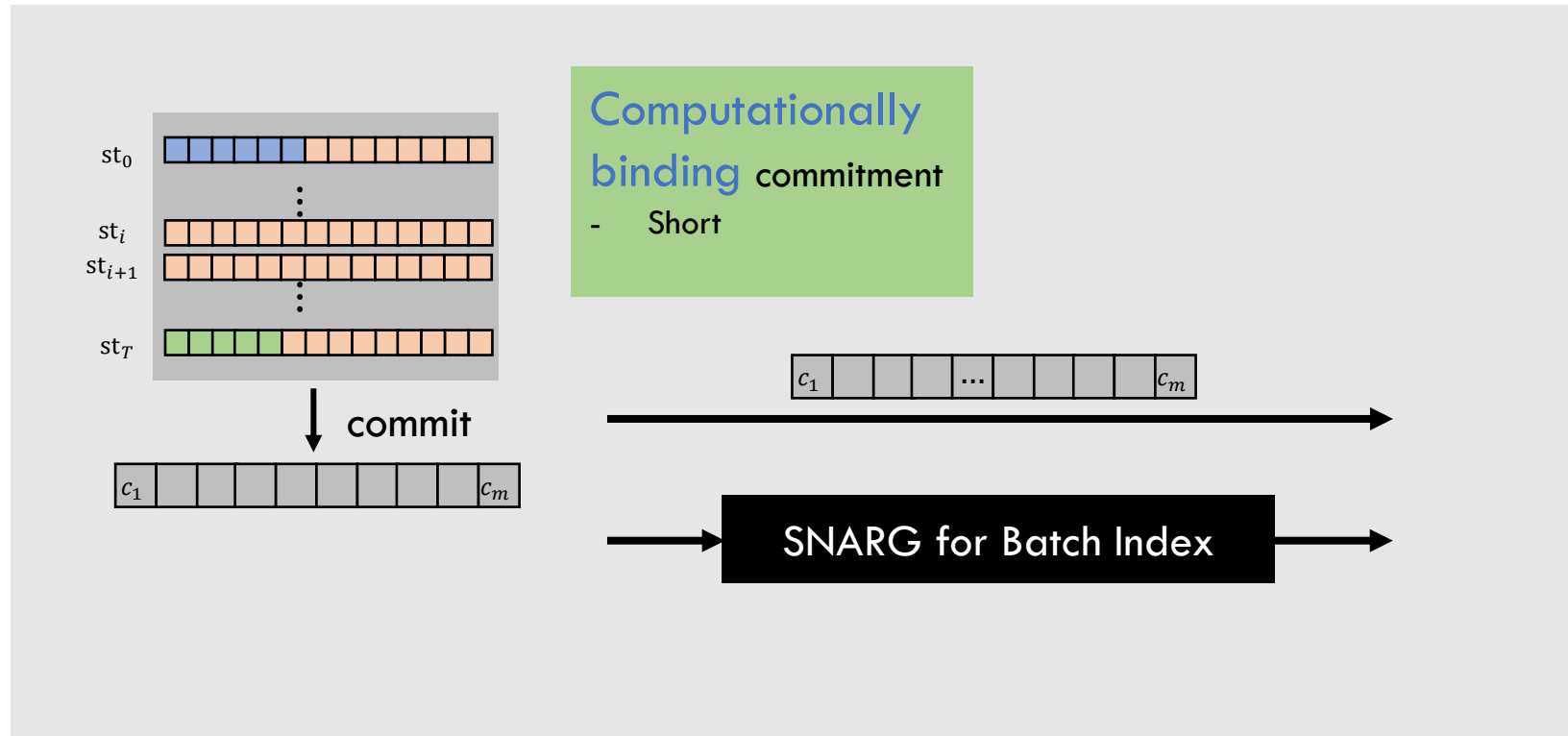


SNARG for Batch Index

For every $i \in [0, \dots, T - 1]$

1. Commitment contains st_i and st_{i+1}
2. Valid transition $st_i \rightarrow st_{i+1}$

SNARGs for Polynomial-time Computation

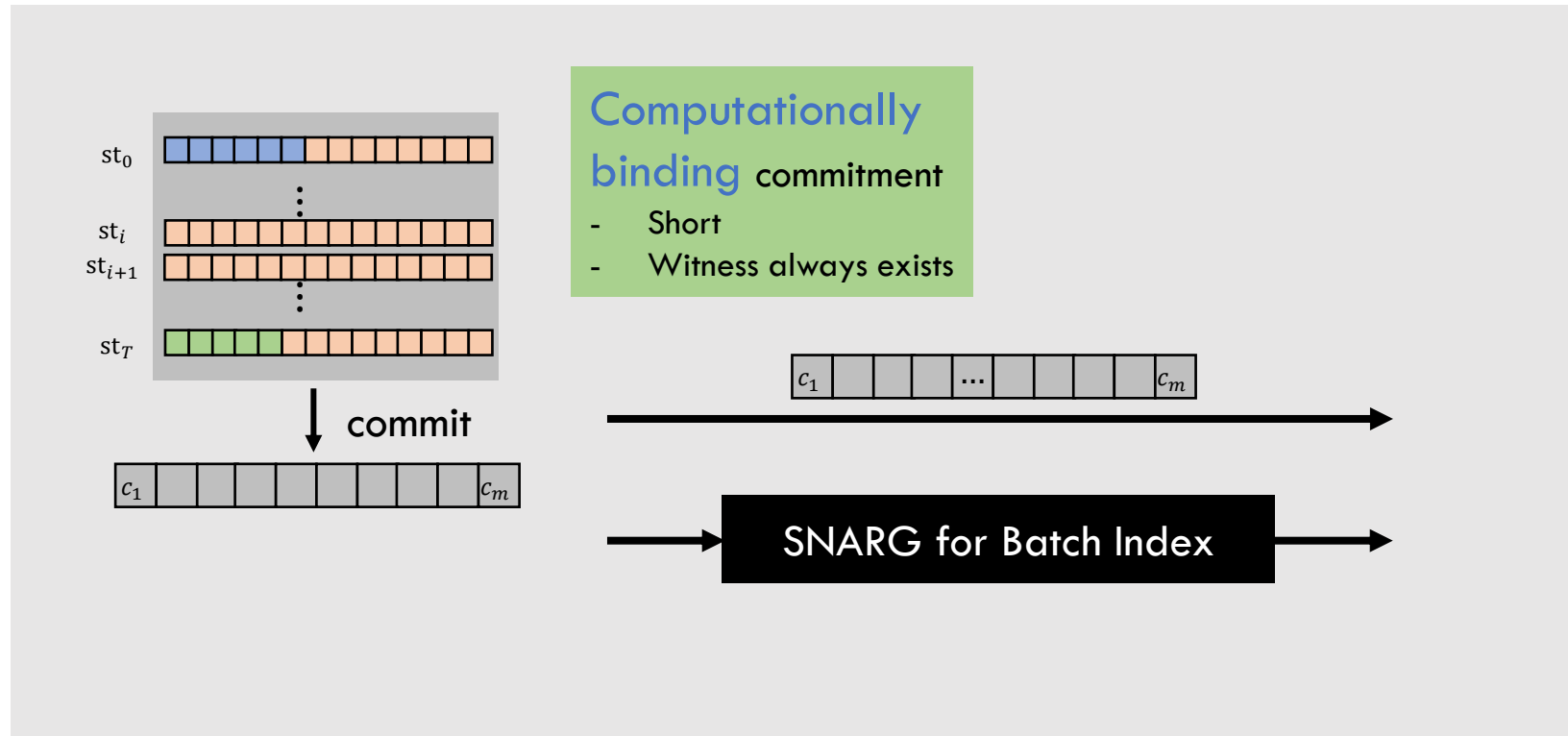


SNARG for Batch Index

For every $i \in [0, \dots, T - 1]$

1. Commitment contains st_i and st_{i+1}
2. Valid transition $st_i \rightarrow st_{i+1}$

SNARGs for Polynomial-time Computation

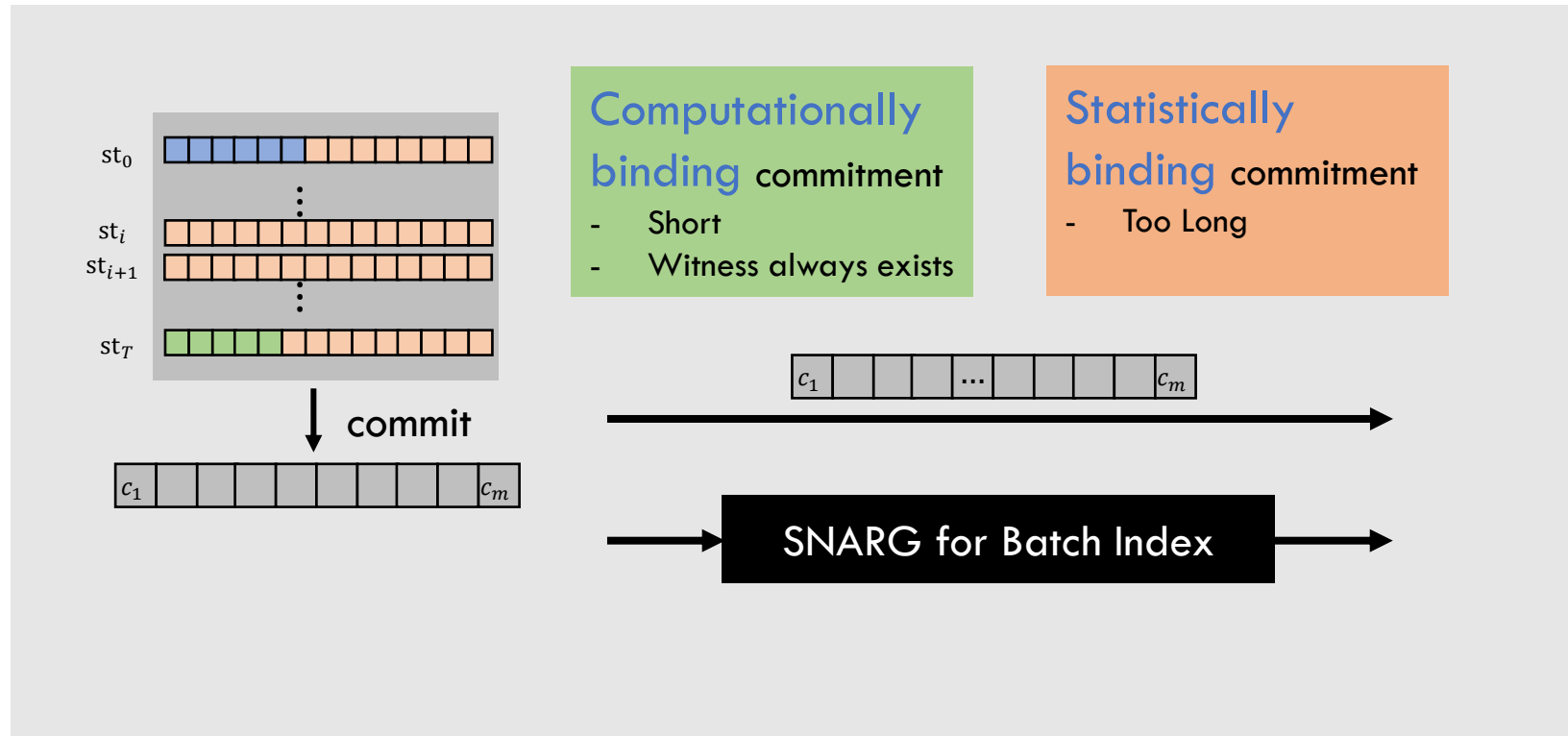


SNARG for Batch Index

For every $i \in [0, \dots, T - 1]$

1. Commitment **opening** to st_i and st_{i+1}
2. Valid transition $st_i \rightarrow st_{i+1}$

SNARGs for Polynomial-time Computation

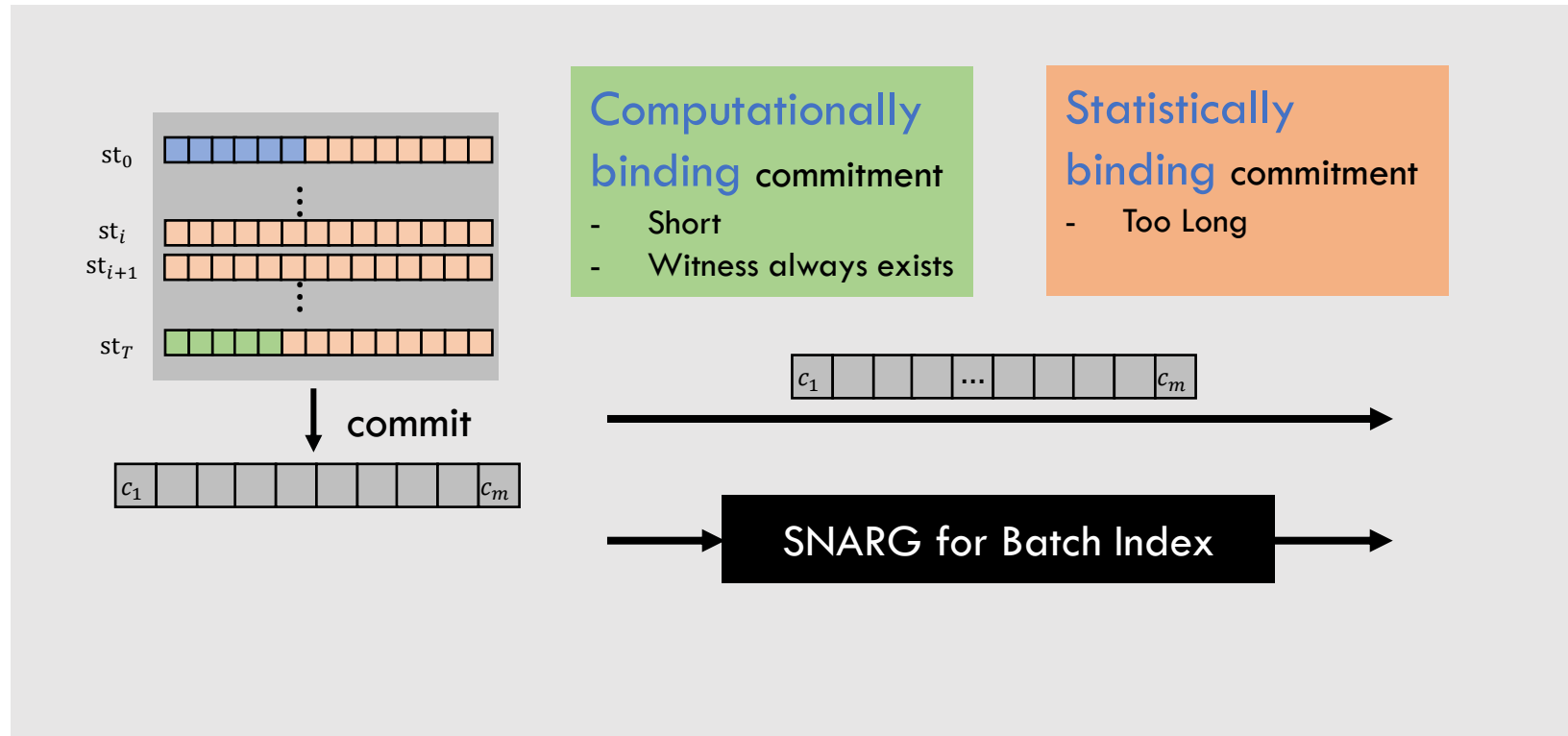


SNARG for Batch Index

For every $i \in [0, \dots, T - 1]$

1. Commitment **opening to** st_i and st_{i+1}
2. Valid transition $st_i \rightarrow st_{i+1}$

SNARGs for Polynomial-time Computation



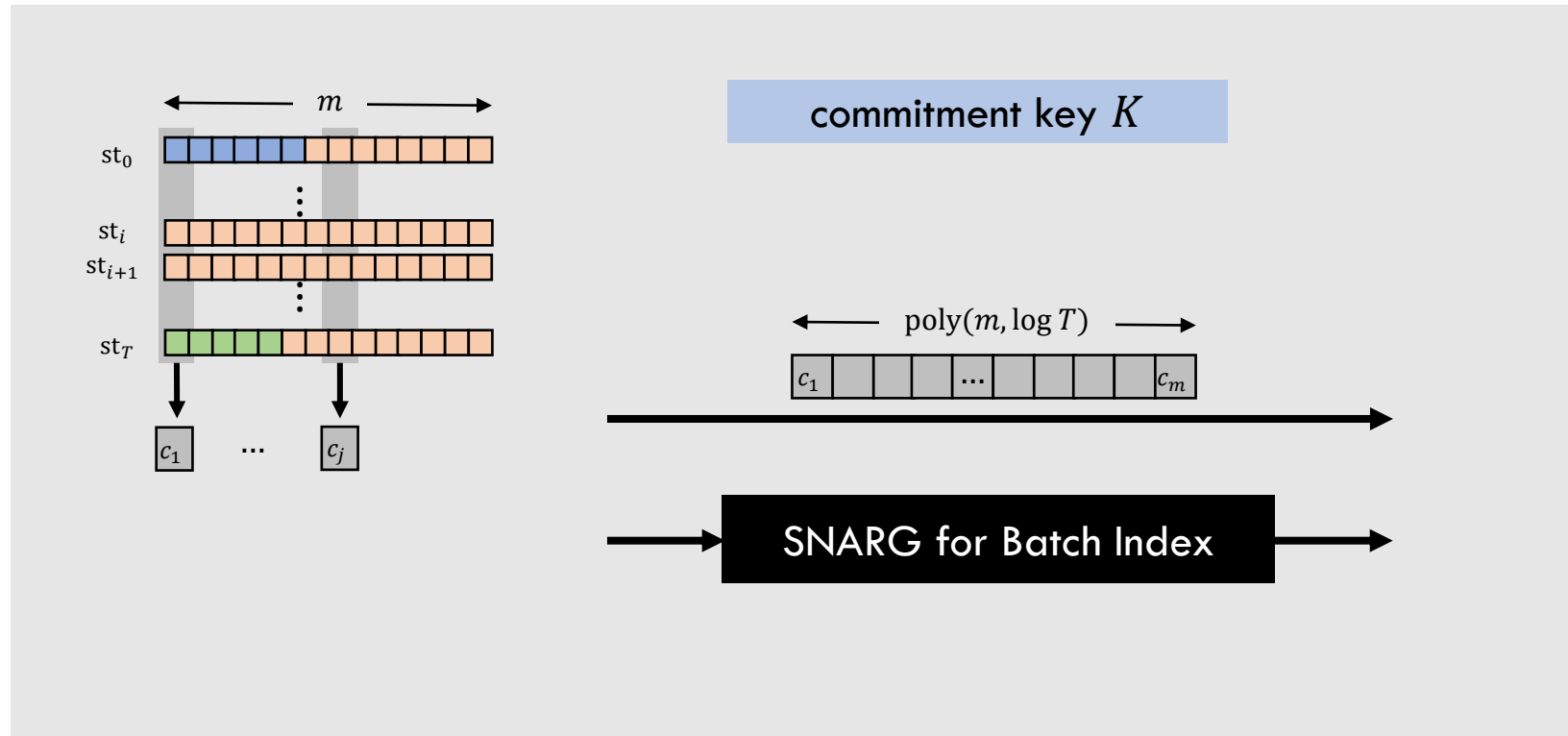
Use SSB Commitments

SNARG for Batch Index

For every $i \in [0, \dots, T - 1]$

1. Commitment **opening to** st_i and st_{i+1}
2. Valid transition $st_i \rightarrow st_{i+1}$

SNARGs for Polynomial-time Computation

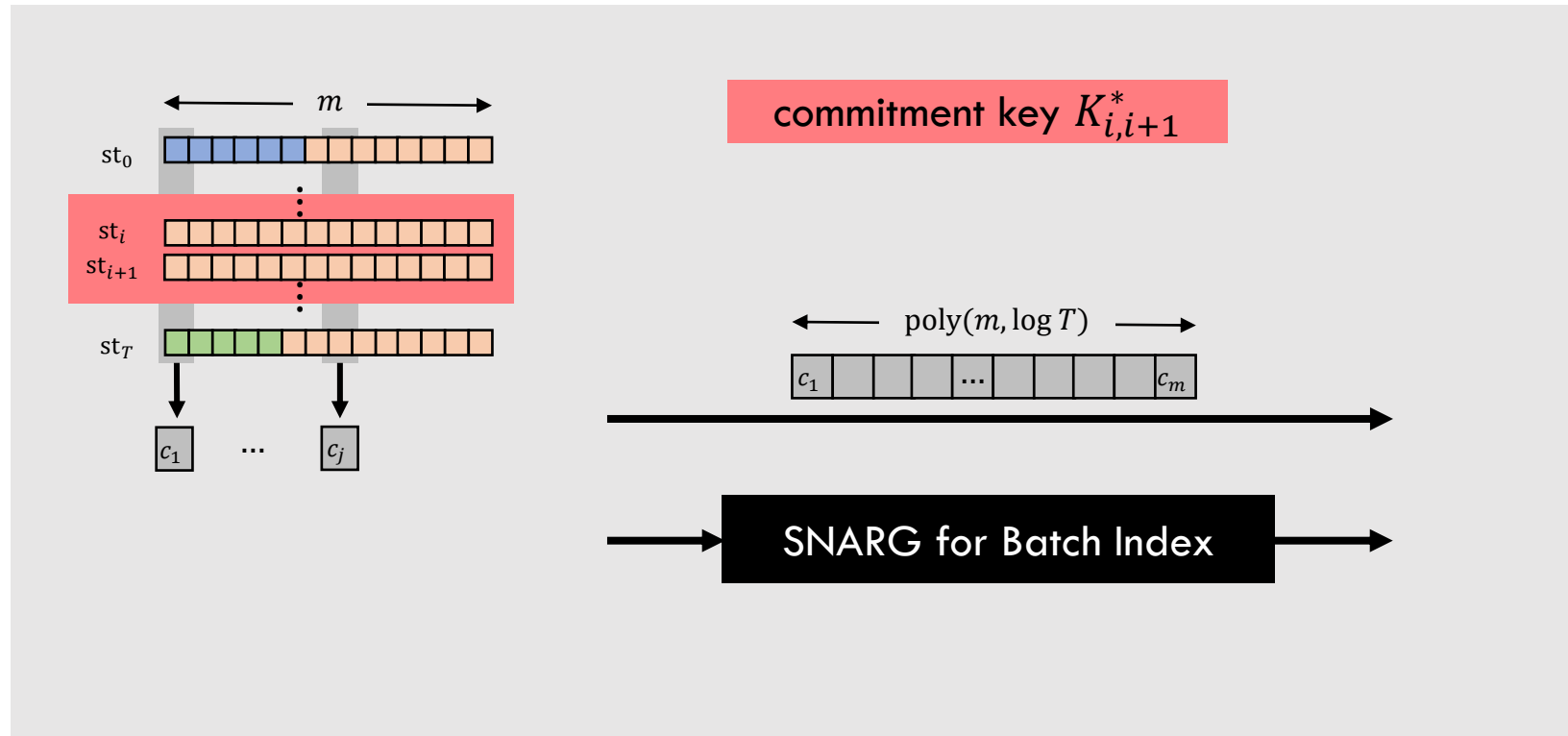


SNARG for Batch Index

For every $i \in [0, \dots, T - 1]$

1. Commitment **opening** to st_i and st_{i+1}
2. Valid transition $st_i \rightarrow st_{i+1}$

SNARGs for Polynomial-time Computation

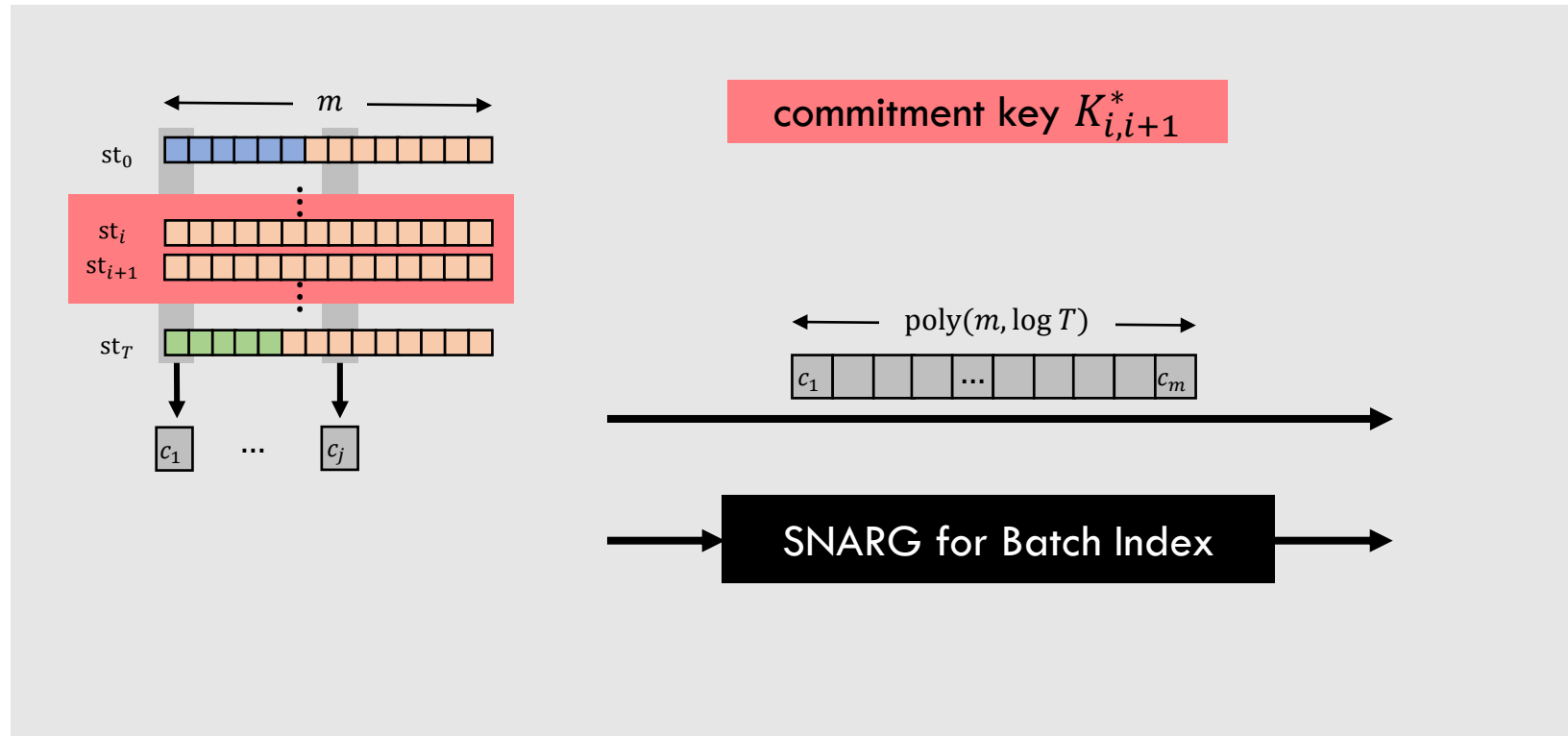


SNARG for Batch Index

For every $i \in [0, \dots, T - 1]$

1. Commitment **opening** to st_i and st_{i+1}
2. Valid transition $st_i \rightarrow st_{i+1}$

SNARGs for Polynomial-time Computation



Local Soundness

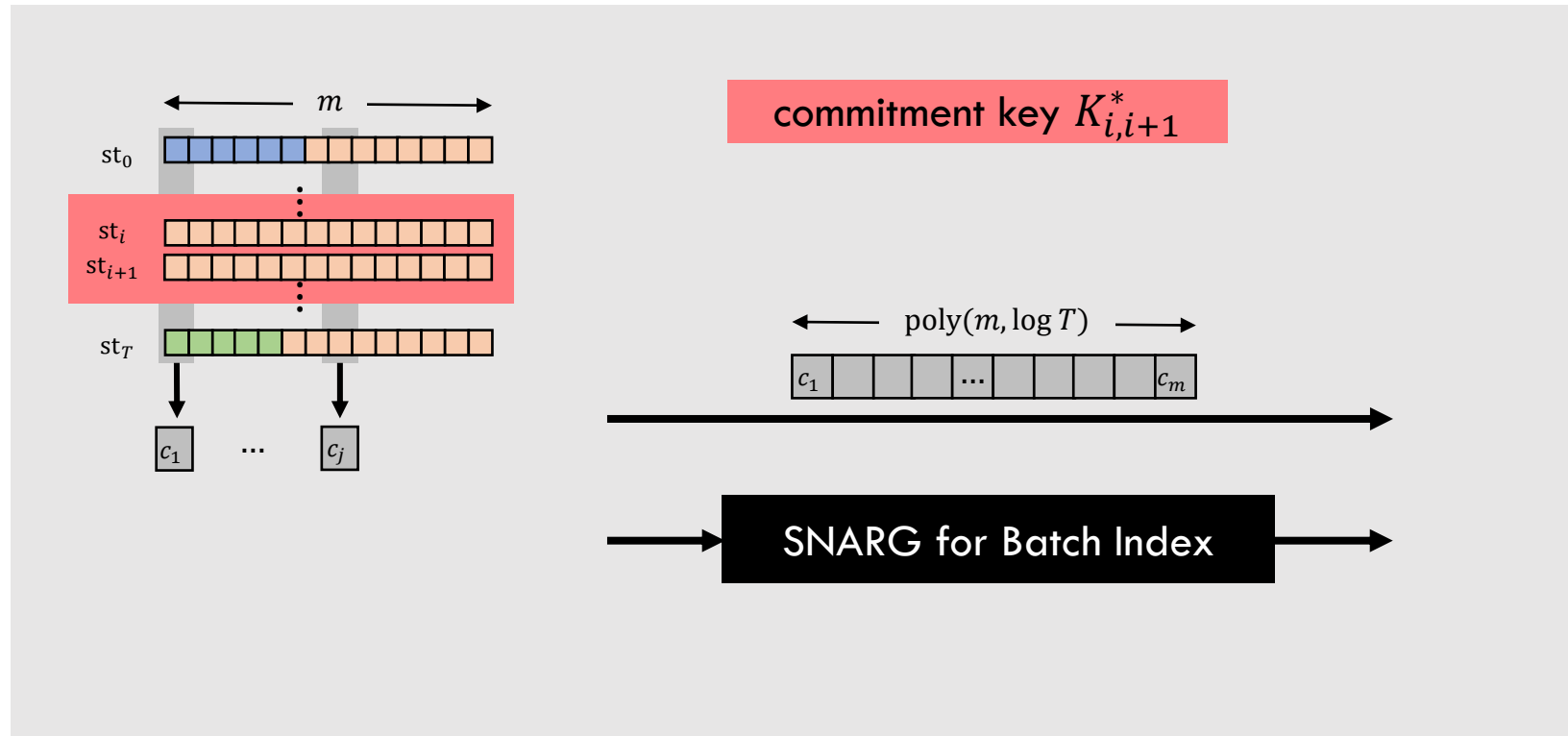
i -th state transition correct

SNARG for Batch Index

For every $i \in [0, \dots, T - 1]$

1. Commitment **opening** to st_i and st_{i+1}
2. Valid transition $st_i \rightarrow st_{i+1}$

SNARGs for Polynomial-time Computation



Local Soundness
 i -th state transition correct



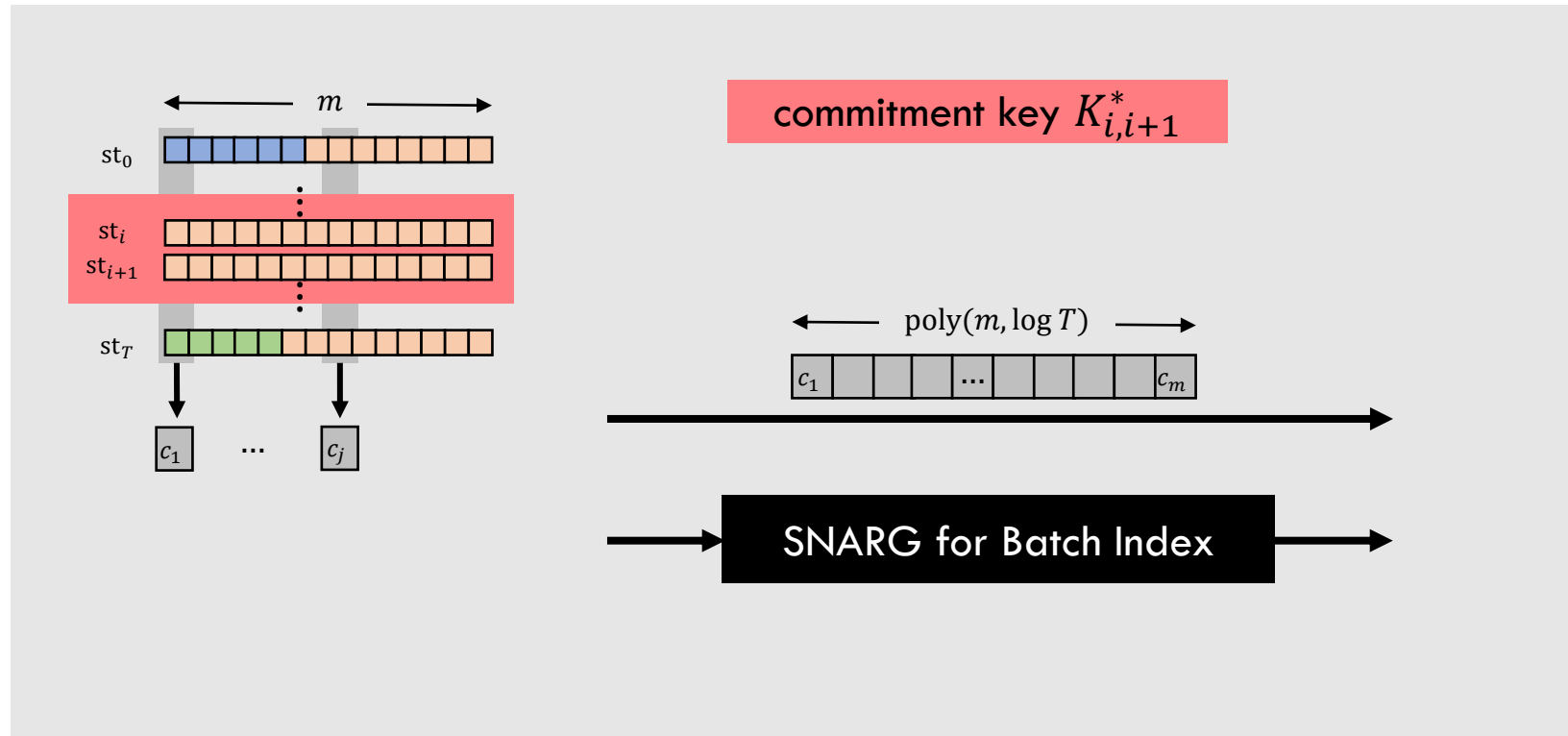
Global Soundness
Local soundness at **all** i

SNARG for Batch Index

For every $i \in [0, \dots, T - 1]$

1. Commitment **opening** to st_i and st_{i+1}
2. Valid transition $st_i \rightarrow st_{i+1}$

SNARGs for Polynomial-time Computation



No-Signaling SSB Commitment Scheme [González-Zacharakis'21]

Local Soundness
 i -th state transition correct



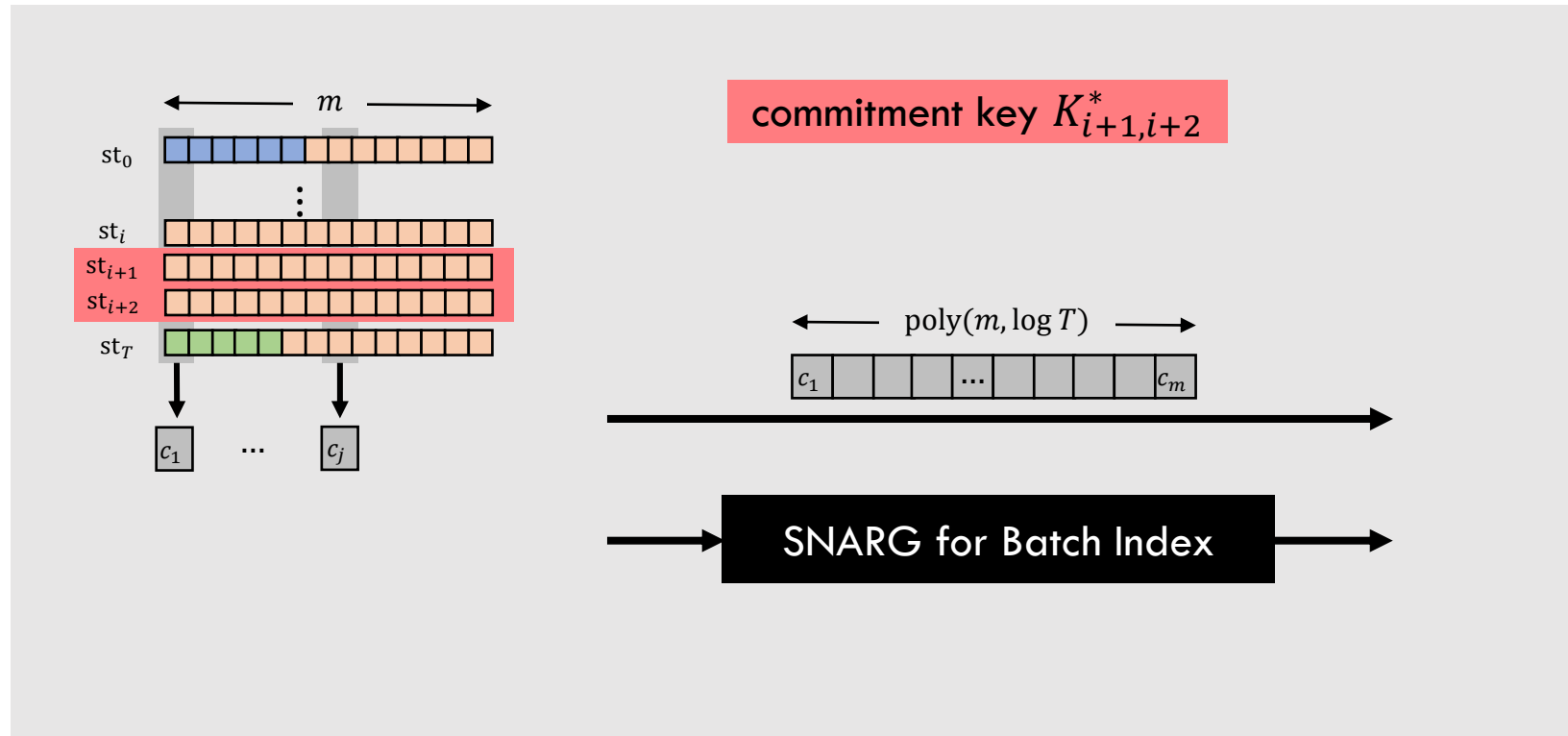
Global Soundness
Local soundness at **all** i

SNARG for Batch Index

For every $i \in [0, \dots, T - 1]$

1. Commitment **opening to** st_i and st_{i+1}
2. Valid transition $st_i \rightarrow st_{i+1}$

SNARGs for Polynomial-time Computation



No-Signaling SSB Commitment Scheme [González-Zacharakis'21]

Local Soundness
 i -th state transition correct

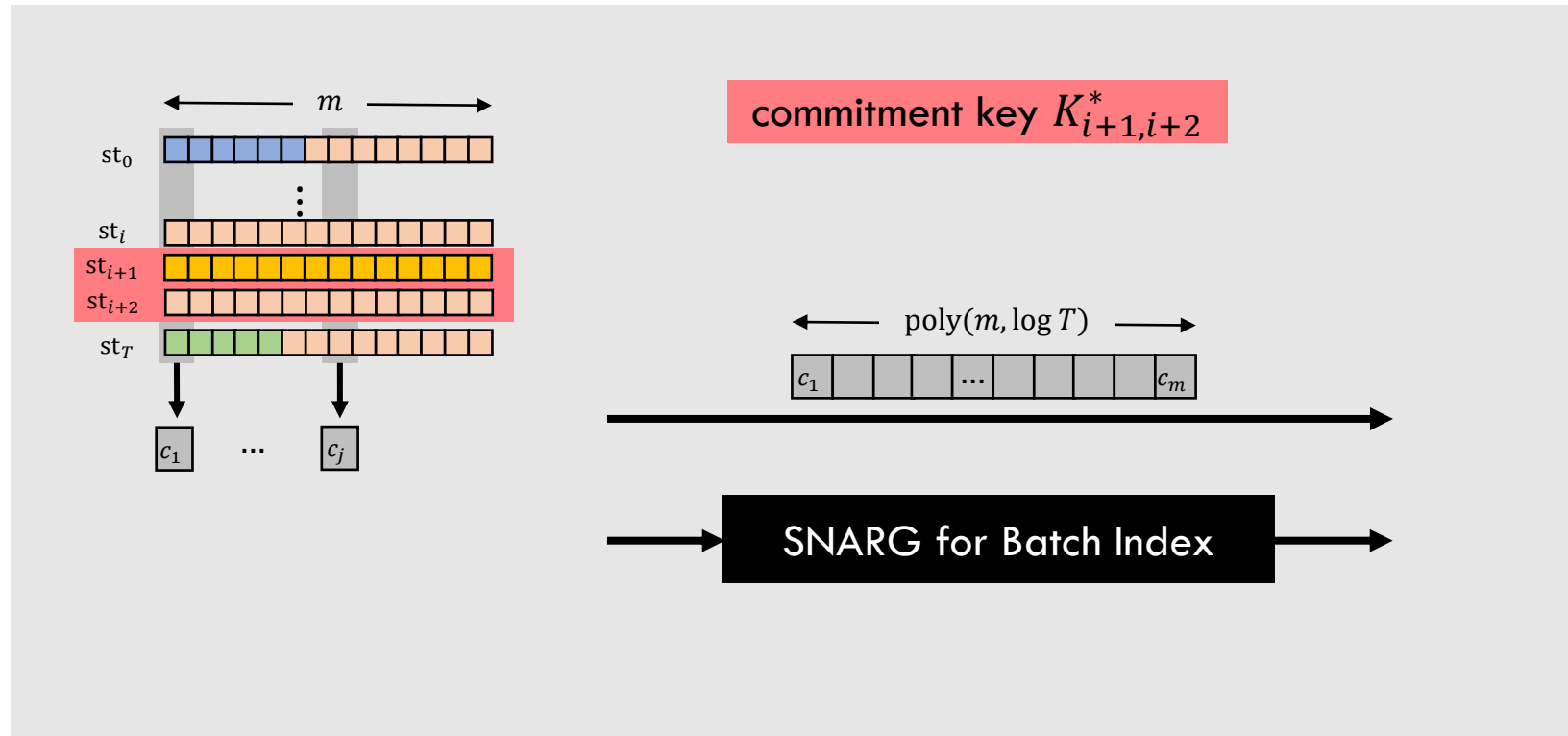
Global Soundness
Local soundness at **all** i

SNARG for Batch Index

For every $i \in [0, \dots, T - 1]$

1. Commitment **opening to** st_i and st_{i+1}
2. Valid transition $st_i \rightarrow st_{i+1}$

SNARGs for Polynomial-time Computation



No-Signaling SSB Commitment Scheme [González-Zacharakis'21]

Local Soundness
 i -th state transition correct



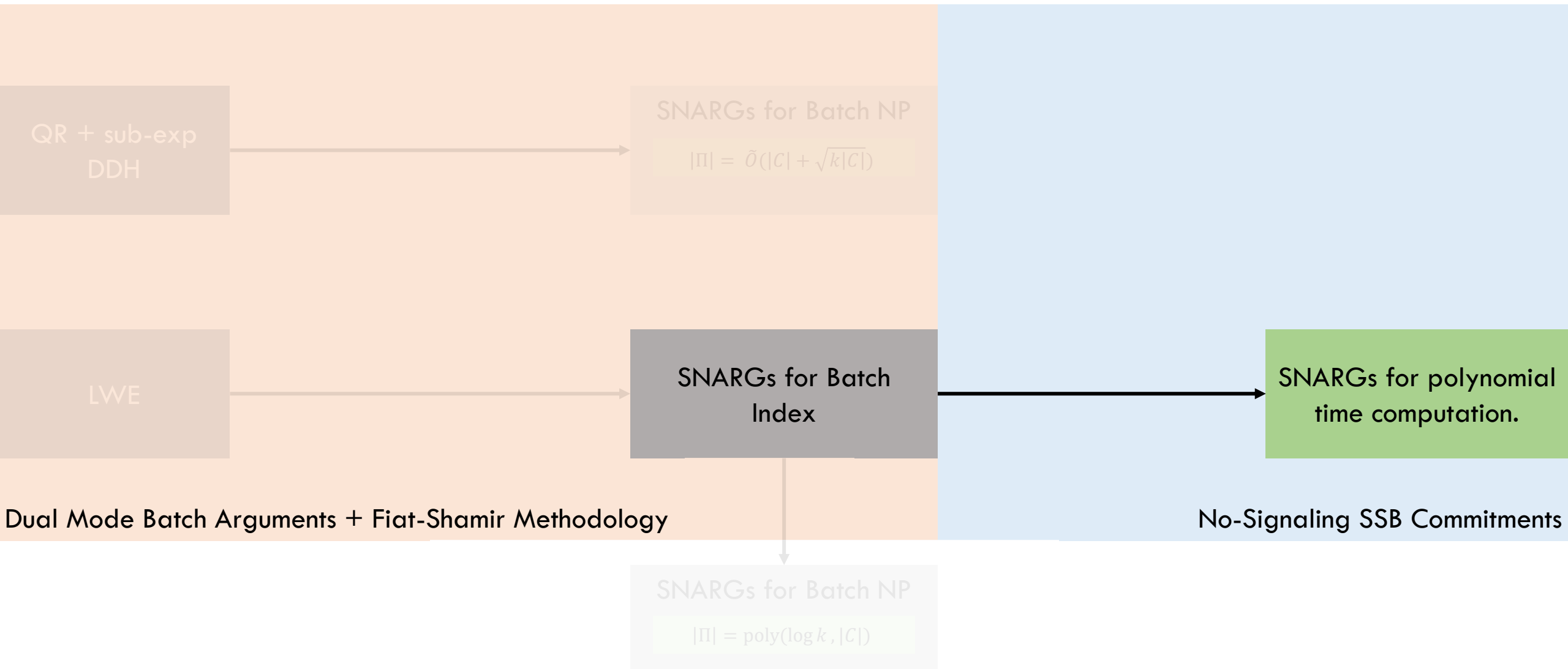
Global Soundness
Local soundness at **all** i

SNARG for Batch Index

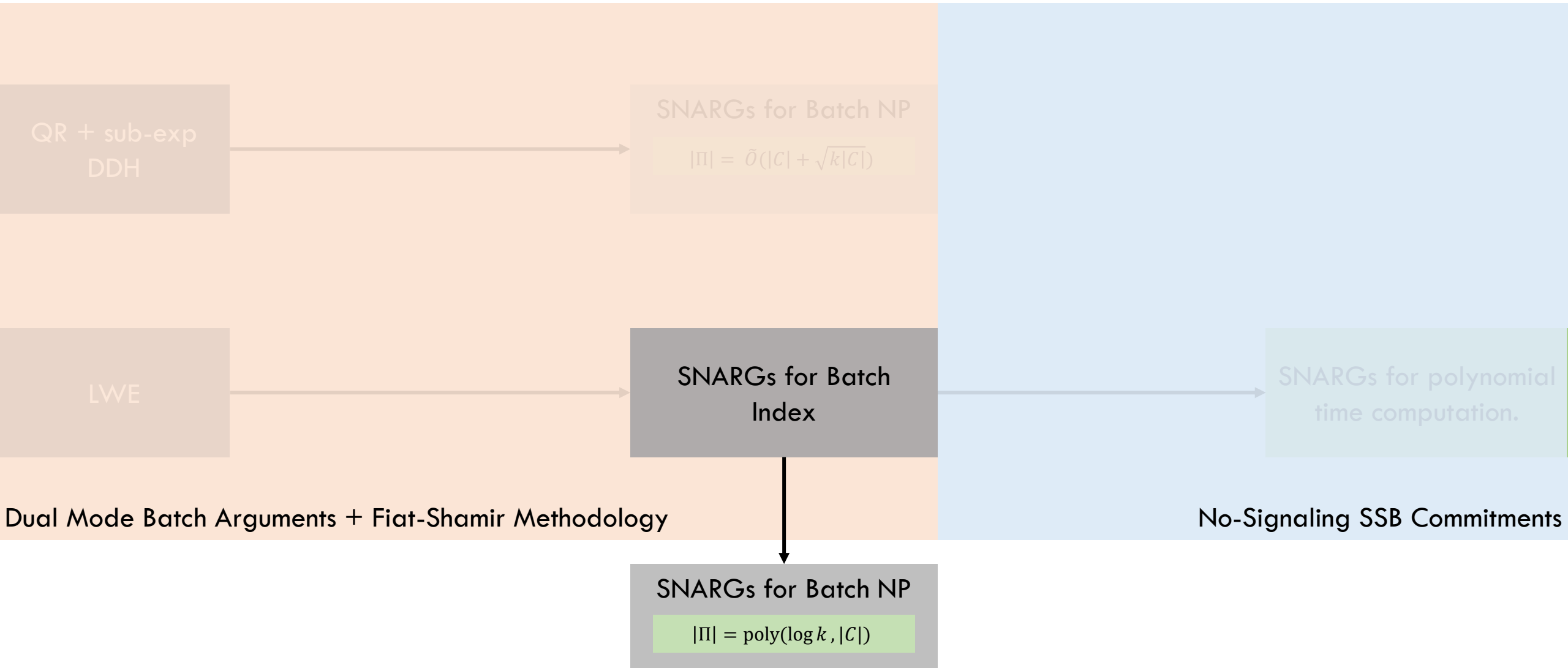
For every $i \in [0, \dots, T - 1]$

1. Commitment **opening to** st_i and st_{i+1}
2. Valid transition $st_i \rightarrow st_{i+1}$

Results Overview

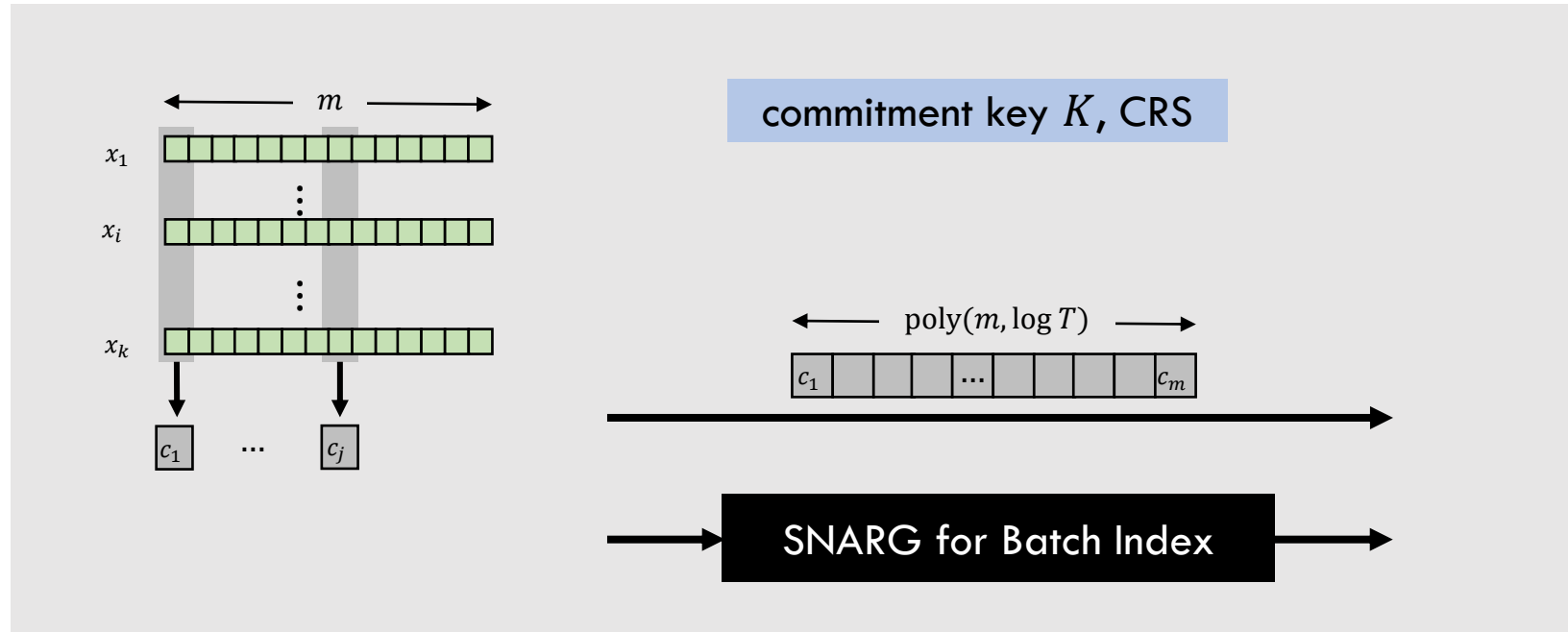


Results Overview



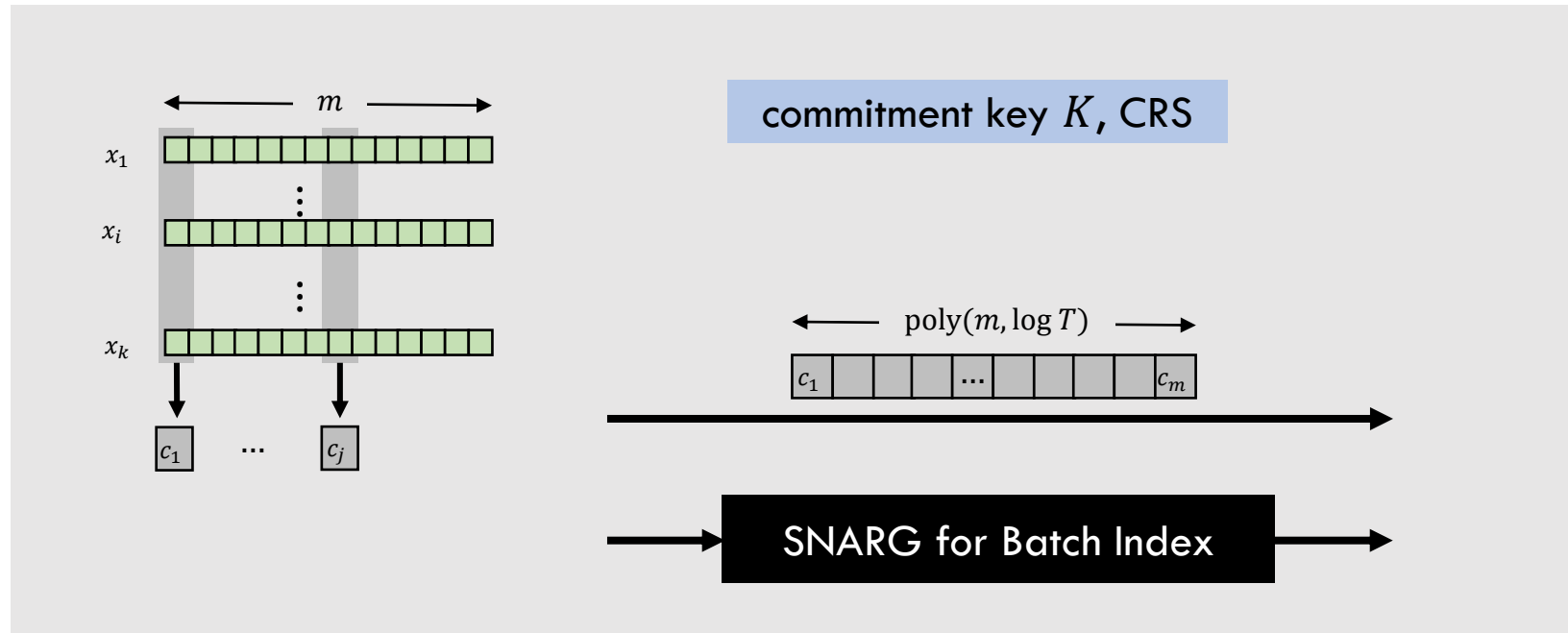
SNARGs for Batch Index \rightarrow SNARGs for Batch NP

SNARGs for Batch NP

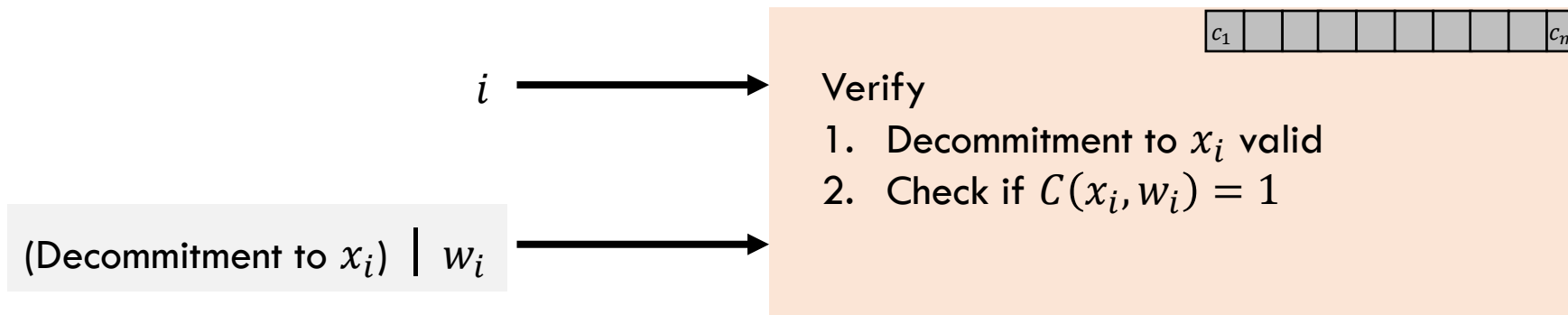


Use fixed randomness for the commitment (say 0)

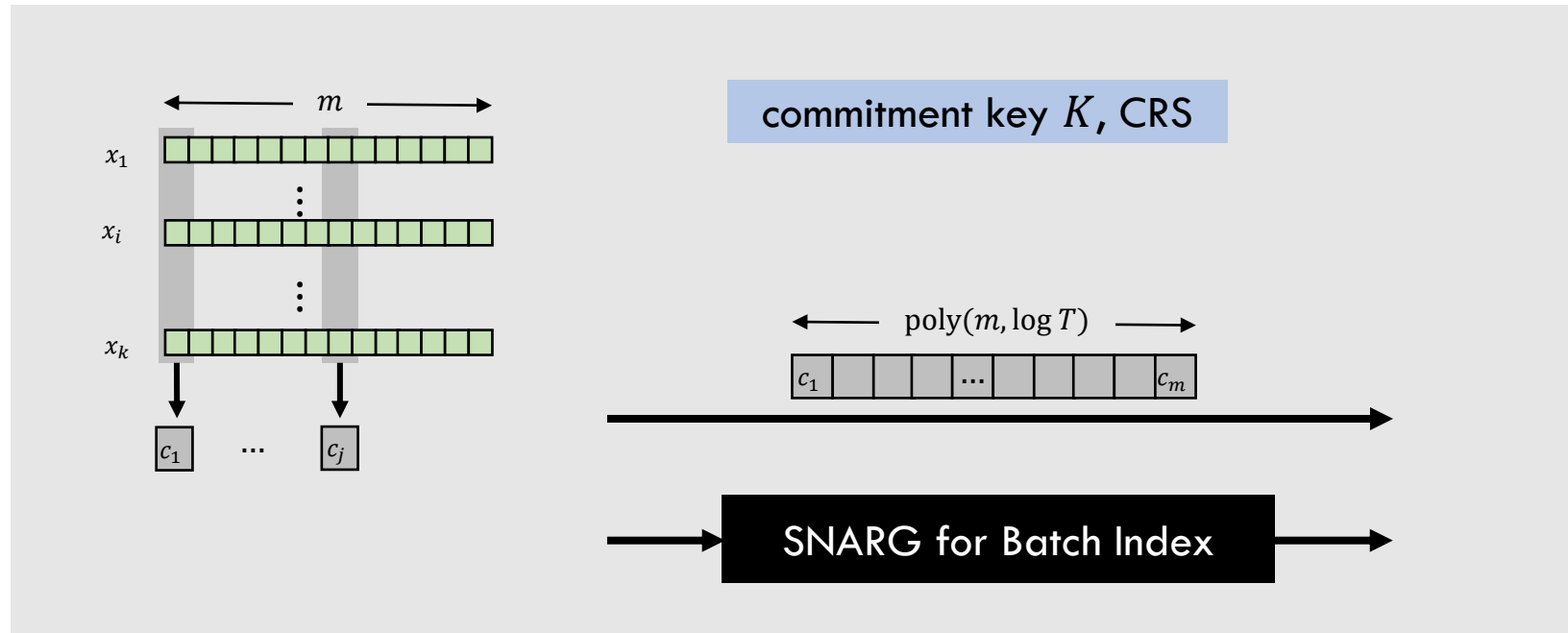
SNARGs for Batch NP



Use fixed randomness for the commitment (say 0)

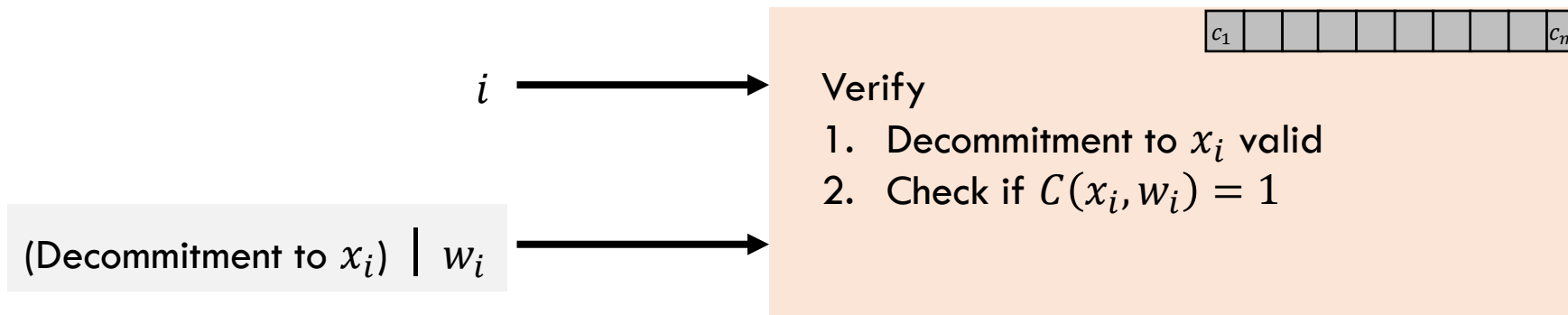


SNARGs for Batch NP

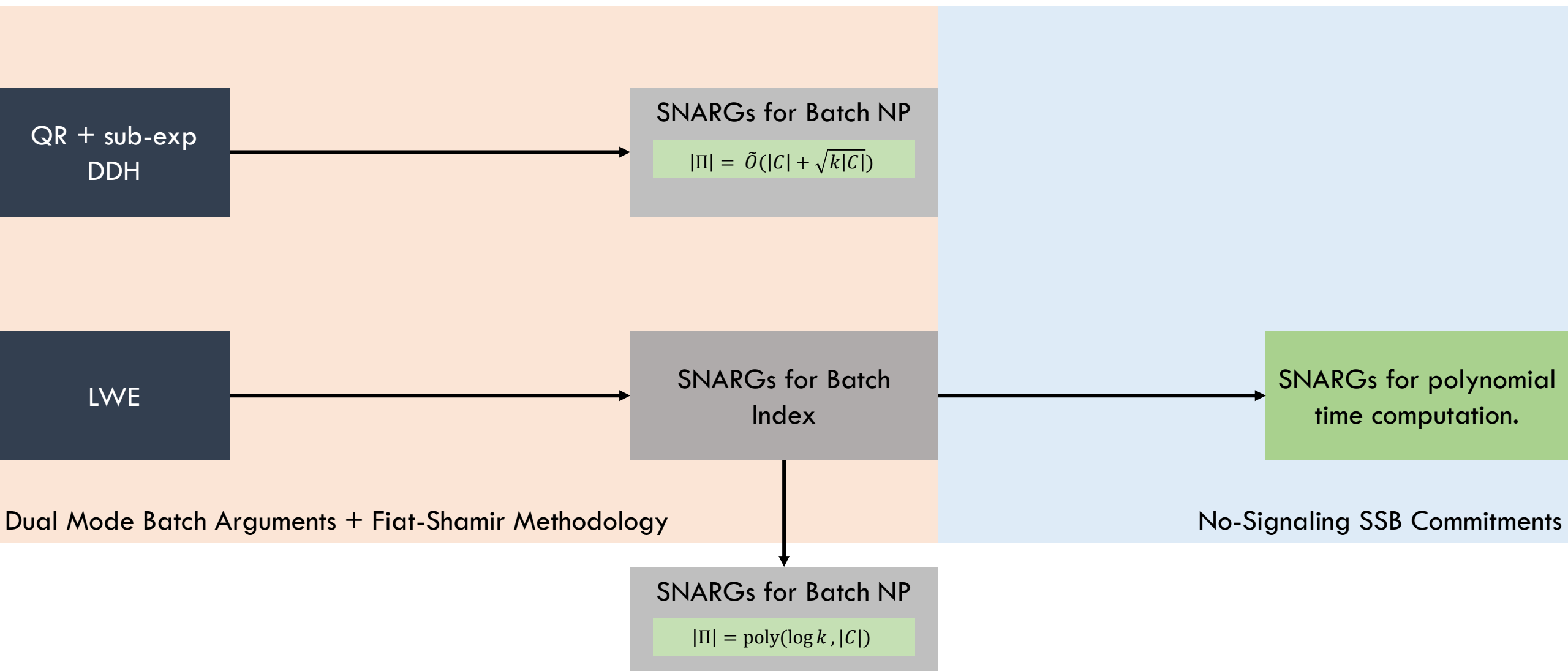


Use fixed randomness for the commitment (say 0)

Require SSB Commitments to rely on SNARG soundness.



Recap



Open Questions

Achieving succinct delegation from DDH?

Incrementally verifiable computation from LWE?

Establishing hardness of complexity classes such as PPAD, PLS?

Thank you. Questions?

Arka Rai Choudhuri

arkarc@berkeley.edu

ia.cr/2021/807, ia.cr/2021/808