Quantum Random Oracles 2/2: Extractability via Compressed Oracles

Warsaw IACR Summer School on Post-Quantum Cryptography 2024

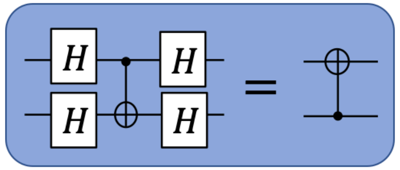
Christian Majenz
DTU Compute
Technical University of Denmark

Outline

- Another look at the compressed oracle
- Query complexity from compressed oracles
- Extractable commitments in the QROM
- Applications

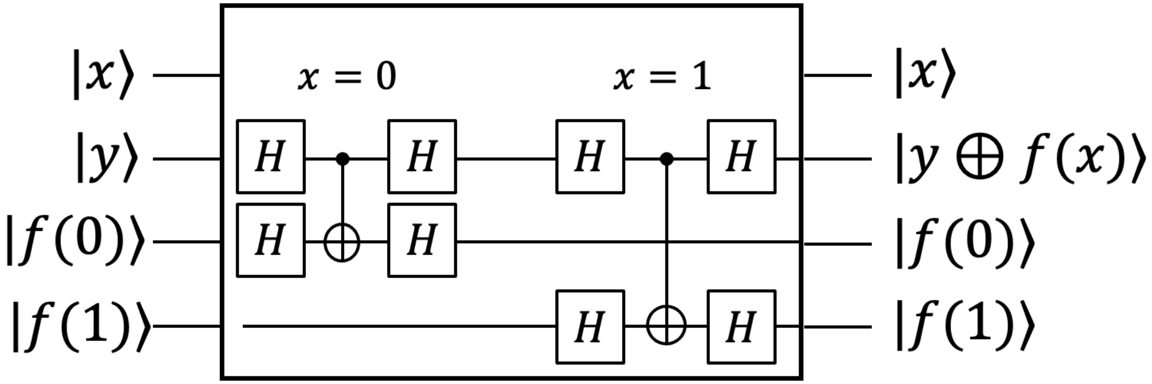
Another look at the compressed oracle

The compressed oracle

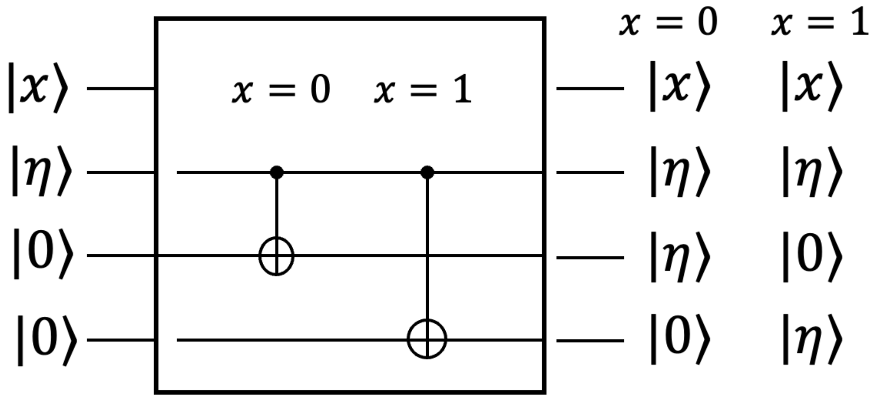


Change of Viewpoint: Fourier Oracle

Standard Oracle



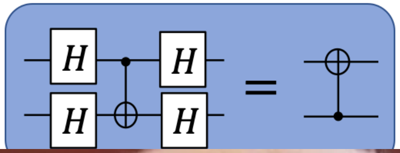
Fourier Oracle

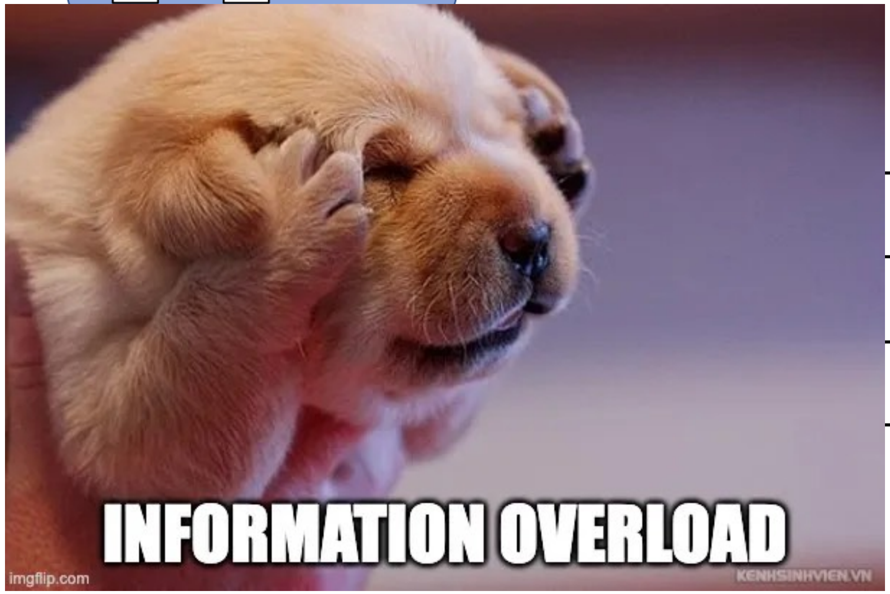


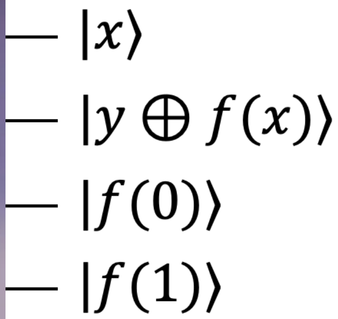
- By making a query, Eve entangles herself with the truth table in a very clean way, when observed in the Fourier basis!

The compressed oracle

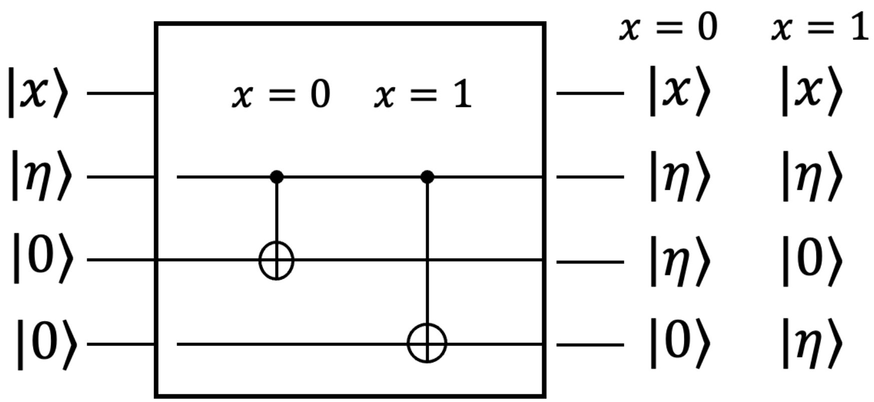
Change of Viewpoint: Fourier Oracle







Fourier Oracle



- By making a query, Eve entangles herself with the truth table in a very clean way, when observed in the Fourier basis!

Can we improve the compressed oracle?

Purpose of the Fourier representation:

Can we improve the compressed oracle?

Purpose of the Fourier representation:

1. Compression (sparse representation)

Can we improve the compressed oracle?

Purpose of the Fourier representation:

1. Compression (sparse representation)
2. Access information: was x queried or not?

Can we improve the compressed oracle?

Purpose of the Fourier representation:

1. Compression (sparse representation)
2. Access information: was x queried or not?
3. Actually want answer to “was x queried or not, and if yes: what the function value?”

Can we improve the compressed oracle?

Purpose of the Fourier representation:

1. Compression (sparse representation)
2. Access information: was x queried or not?
3. Actually want answer to “was x queried or not, and if yes: what the function value?”

Idea: make *minimal* basis change to achieve 1. & 2:

- map initial state to special symbol
- leave everything else as is

Can we improve the compressed oracle?

Purpose of the Fourier representation:

1. Compression (sparse representation)
2. Access information: was x queried or not?
3. Actually want answer to “was x queried or not, and if yes: what the function value?”

Idea: make *minimal* basis change to achieve 1. & 2:

- map initial state to special symbol
- leave everything else as is

First attempt: basis change operator V s.t.

- $V|+^n\rangle = |\perp\rangle$
- $V|x\rangle = |x\rangle$ for all $x \in \{0,1\}^n$

Can we improve the compressed oracle?

Purpose of the Fourier representation:

1. Compression (sparse representation)
2. Access information: was x queried or not?
3. Actually want answer to "was x queried or not, and if yes: what the function value?"

Idea: make *minimal* basis change to achieve 1. & 2:

- map initial state to special symbol
- leave everything else as is

First attempt: basis change operator V s.t.

- $V|+^n\rangle = |\perp\rangle$
- $V|x\rangle = |x\rangle$ for all $x \in \{0,1\}^n$

Too much to ask: $\langle +^n | x\rangle \neq 0$**!**

Can we improve the compressed oracle?

Purpose of the Fourier representation:

1. Compression (sparse representation)
2. Access information: was x queried or not?
3. Actually want answer to "was x queried or not, and if yes: what the function value?"

Idea: make *minimal* basis change to achieve 1. & 2:

- map initial state to special symbol
- leave everything else as is

First attempt: basis change operator V s.t.

- $V|$ [illegible]
- $V|x\rangle$ [illegible] $\{0,1\}^n$

[illegible] to ask: $\langle +$ [illegible]

Can we improve the compressed oracle?

Purpose of the Fourier representation:

1. Compression (sparse representation)
2. Access information: was x queried or not?
3. Actually want answer to “was x queried or not, and if yes: what the function value?”

Idea: make *minimal* basis change to achieve 1. & 2:

- map initial state to special symbol
- leave everything else as is

First attempt: basis change operator V s.t.

- $V|$ [illegible]
- $V|x\rangle$ [illegible] $\{0,1\}^n$

[illegible] to ask: $\langle +$ [illegible]

Second attempt: basis change operator V s.t.

- $V|+^n\rangle = |\bot\rangle$
- $(V|\bot\rangle = |+^n\rangle)$

Can we improve the compressed oracle?

Purpose of the Fourier representation:

1. Compression (sparse representation)
2. Access information: was x queried or not?
3. Actually want answer to “was x queried or not, and if yes: what the function value?”

Idea: make *minimal* basis change to achieve 1. & 2:

- map initial state to special symbol
- leave everything else as is

First attempt: basis change operator V s.t.

- $V|$ [illegible] $\rangle$
- $V|x\rangle$ [illegible] $\}^n$

[illegible] to ask: $\langle +$ [illegible]

Second attempt: basis change operator V s.t.

- $V| +^n \rangle = | \perp \rangle$
- $(V| \perp \rangle = | +^n \rangle)$
- $V|\phi\rangle = |\phi\rangle$ for all $|\phi\rangle$ with $\langle\phi| \perp \rangle = \langle\phi| +^n \rangle = 0$

Pre-compressed oracle

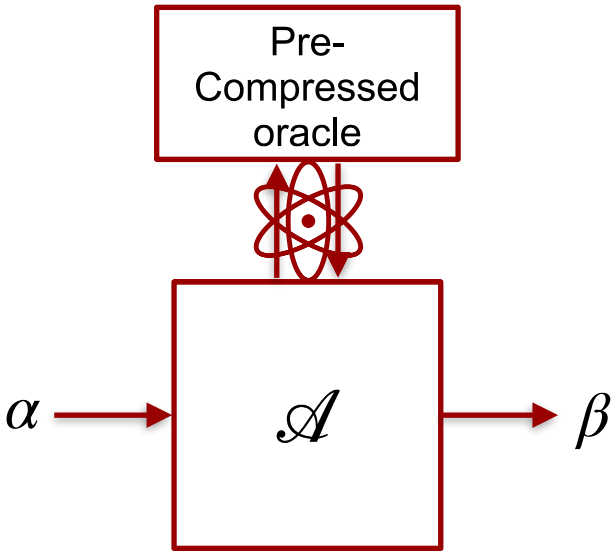
Properties of the pre-compressed oracle for random n-bit to n-bit function f

- Initial state: $\left(| \perp \rangle^{\otimes 2^n} \right)_D$, "database register" $D = D_{0\ldots000} D_{0\ldots001} D_{0\ldots010} \cdots D_{1\ldots1}$
- Compression: Sparse representation with default symbol $\perp$

Pre-compressed oracle

Properties of the pre-compressed oracle for random n-bit to n-bit function f

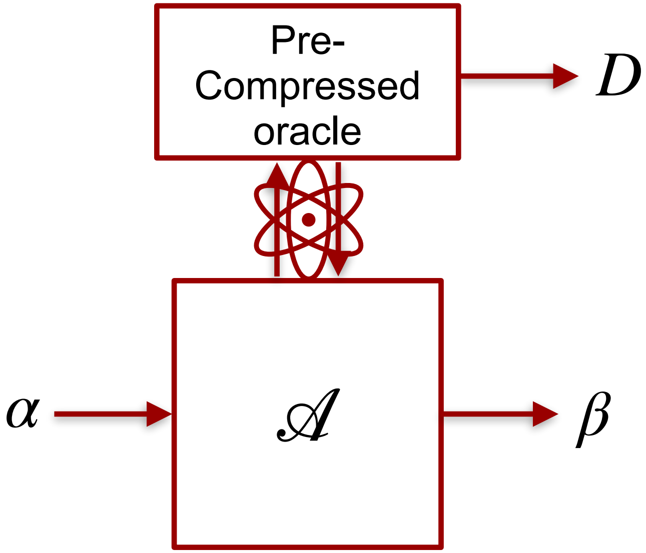
- Initial state: $\left(|\perp\rangle^{\otimes 2^n}\right)_D$, "database register" $D = D_{0\ldots000}\, D_{0\ldots001}\, D_{0\ldots010} \,\cdots\, D_{1\ldots1}$
- Compression: Sparse representation with default symbol $\perp$



Pre-compressed oracle

Properties of the pre-compressed oracle for random n-bit to n-bit function f

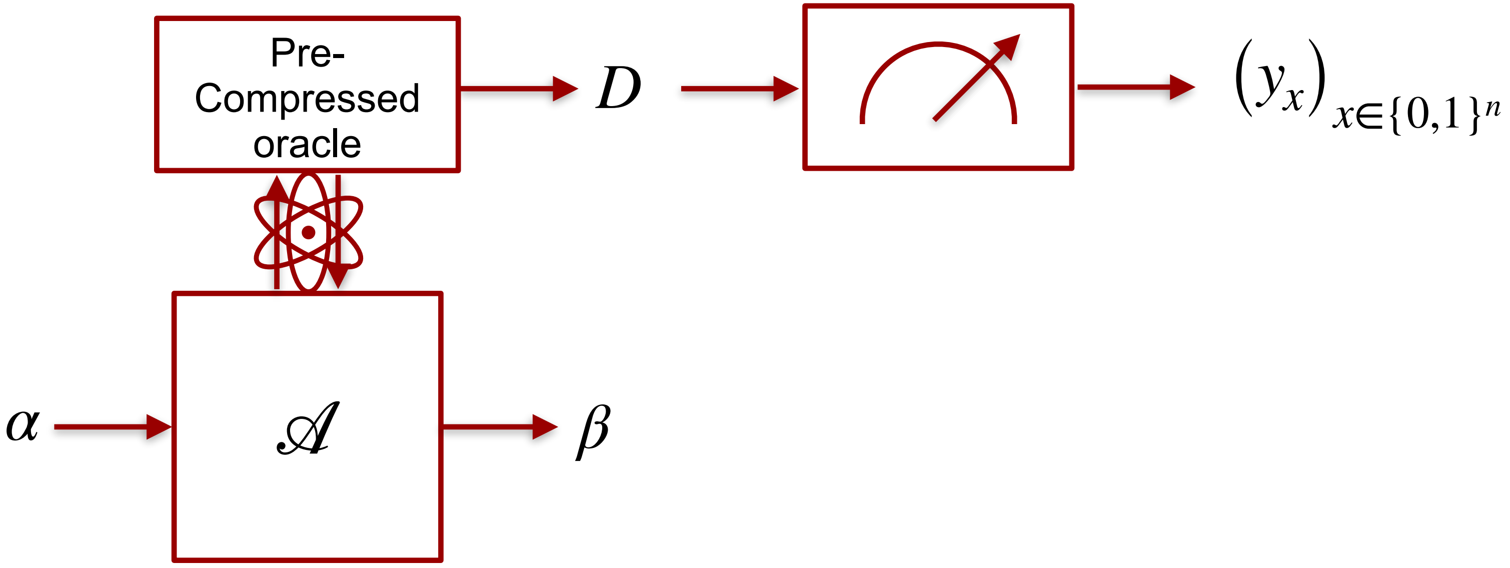
- Initial state: $\left(|\perp\rangle^{\otimes 2^n}\right)_D$, "database register" $D = D_{0\ldots000}\, D_{0\ldots001}\, D_{0\ldots010} \,\cdots\, D_{1\ldots1}$
- Compression: Sparse representation with default symbol $\perp$



Pre-compressed oracle

Properties of the pre-compressed oracle for random n-bit to n-bit function f

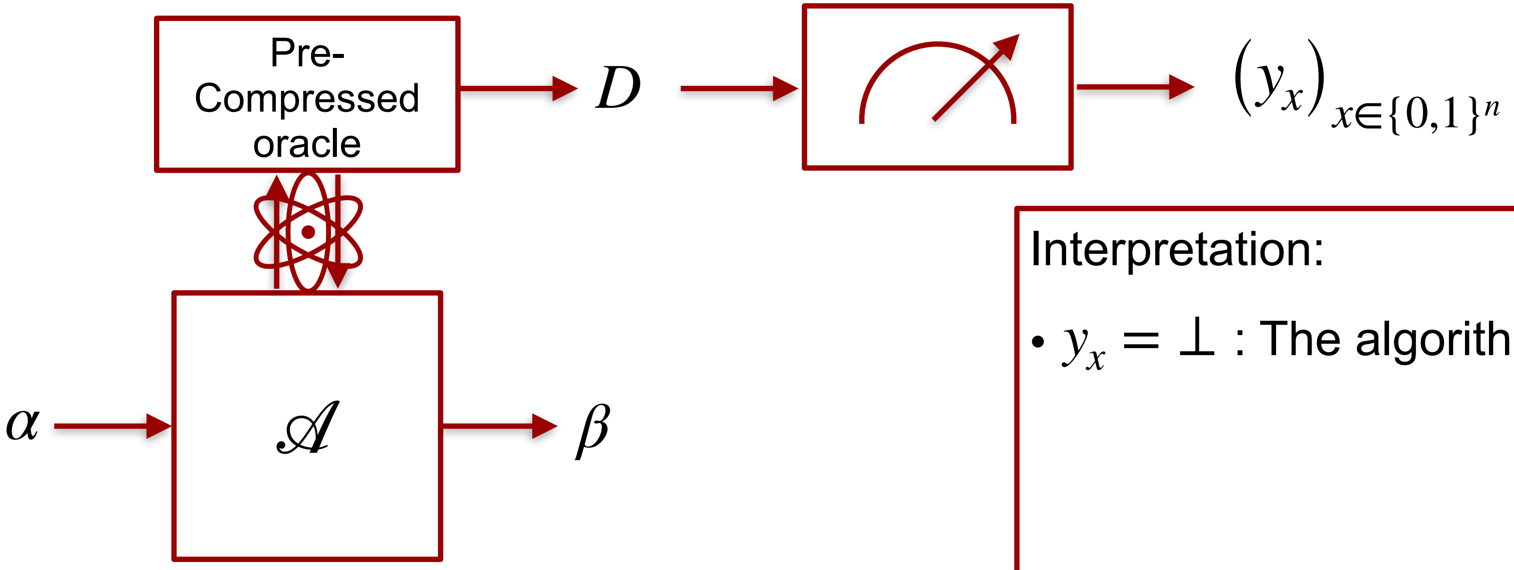
- Initial state: $\left(|\perp\rangle^{\otimes 2^n}\right)_D$, "database register" $D = D_{0\ldots000}\,D_{0\ldots001}\,D_{0\ldots010}\,\cdots\,D_{1\ldots1}$
- Compression: Sparse representation with default symbol $\perp$



Pre-compressed oracle

Properties of the pre-compressed oracle for random n-bit to n-bit function f

- Initial state: $\left(|\perp\rangle^{\otimes 2^n}\right)_D$, "database register" $D = D_{0\ldots000}\, D_{0\ldots001}\, D_{0\ldots010} \cdots D_{1\ldots1}$
- Compression: Sparse representation with default symbol $\perp$



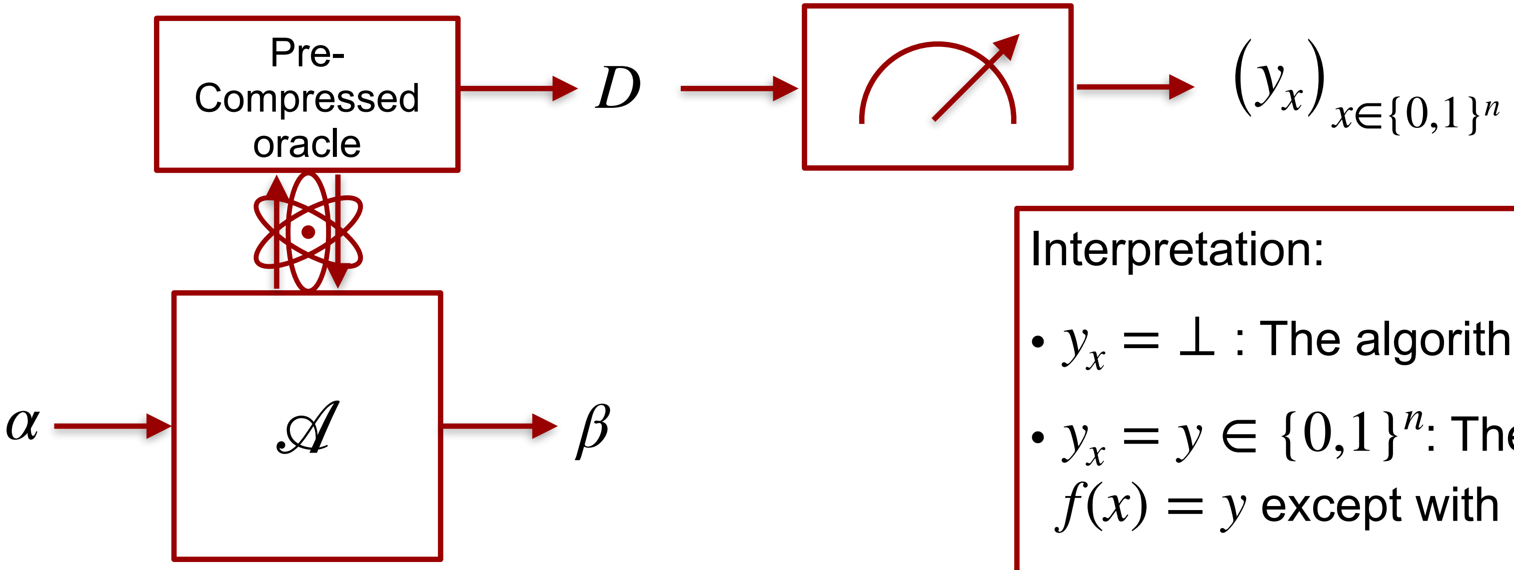
Interpretation:

- $y_x = \perp$: The algorithm $\mathscr{A}$ has no info about $f(x)$

Pre-compressed oracle

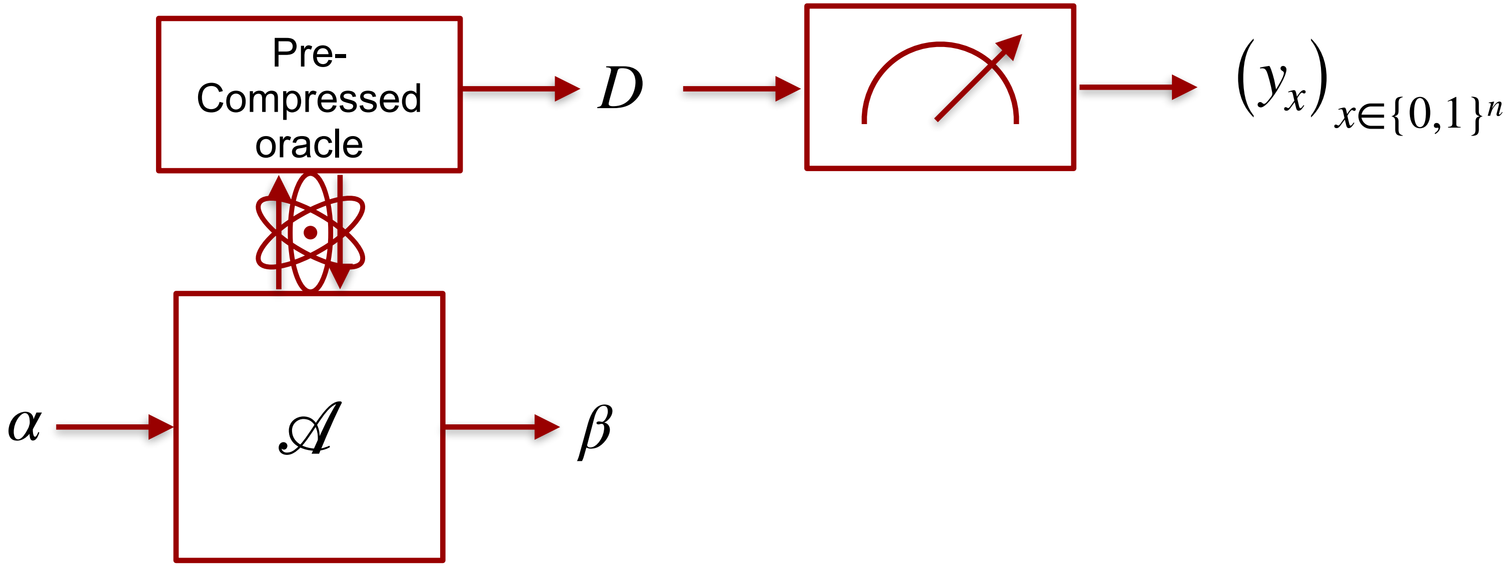
Properties of the pre-compressed oracle for random n-bit to n-bit function f

- Initial state: $\left(|\perp\rangle^{\otimes 2^n}\right)_D$, "database register" $D = D_{0\ldots000}\, D_{0\ldots001}\, D_{0\ldots010} \cdots D_{1\ldots1}$
- Compression: Sparse representation with default symbol $\perp$



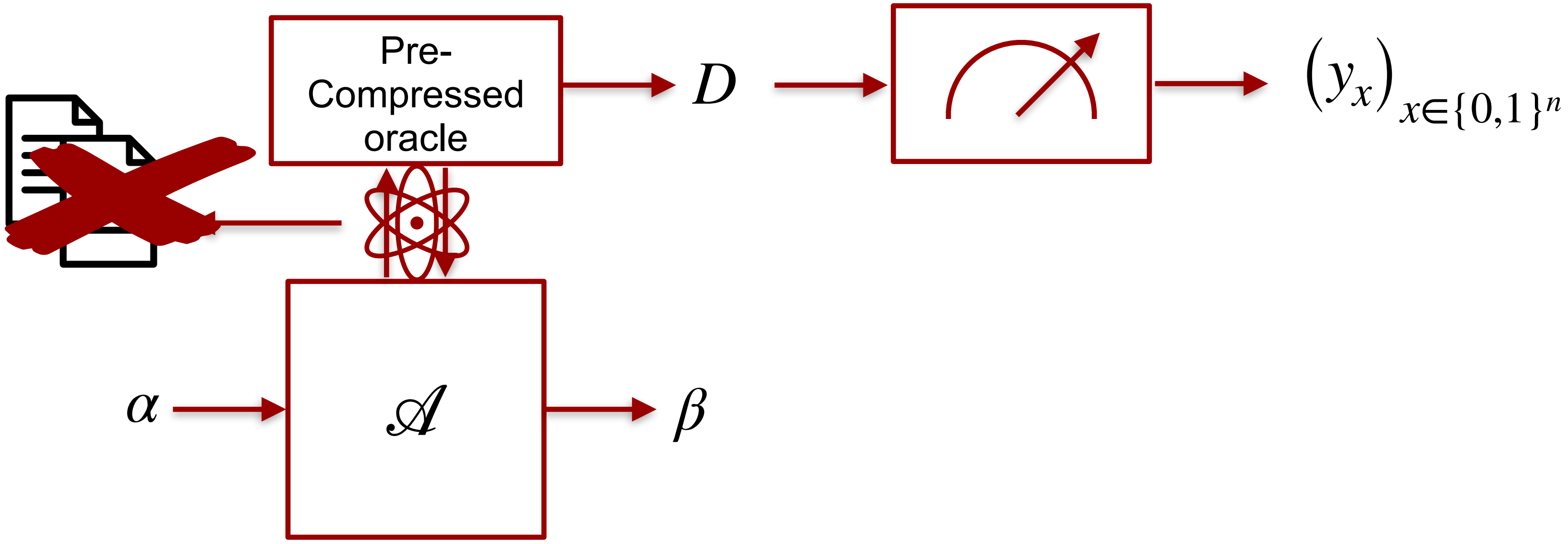
Interpretation:

- $y_x = \perp$: The algorithm $\mathscr{A}$ has no info about $f(x)$
- $y_x = y \in \{0,1\}^n$: The algorithm has queried x, and $f(x) = y$ except with negligible probability.



Interpretation:

- $y_x = \bot$: The algorithm $\mathscr{A}$ has no info about $f(x)$
- $y_x = y \in \{0,1\}^n$: The algorithm has queried x, and $f(x) = y$ except with negligible probability.

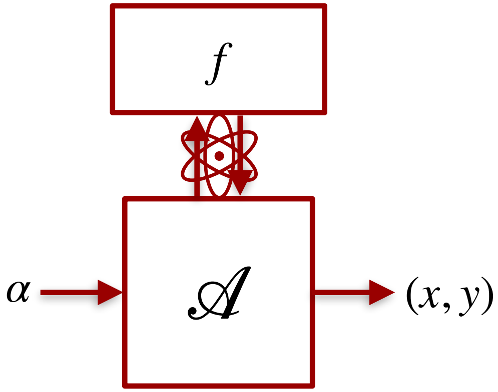


Interpretation:

- $y_x = \perp$: The algorithm $\mathscr{A}$ has no info about $f(x)$
- $y_x = y \in \{0,1\}^n$: The algorithm has queried x, and $f(x) = y$ except with negligible probability.

The fundamental lemma

i)

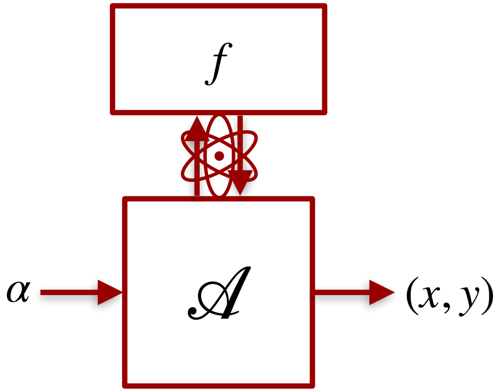


The fundamental lemma

Lemma (Zhandry ’18, slightly informal):

Let $\mathscr{A}$ be a quantum oracle algorithm and R a relation, and let f be a random function. Consider the two experiments

i)

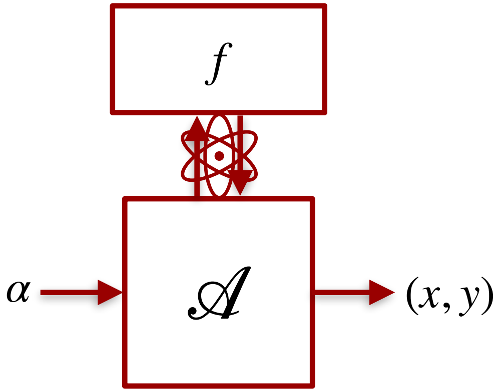


The fundamental lemma

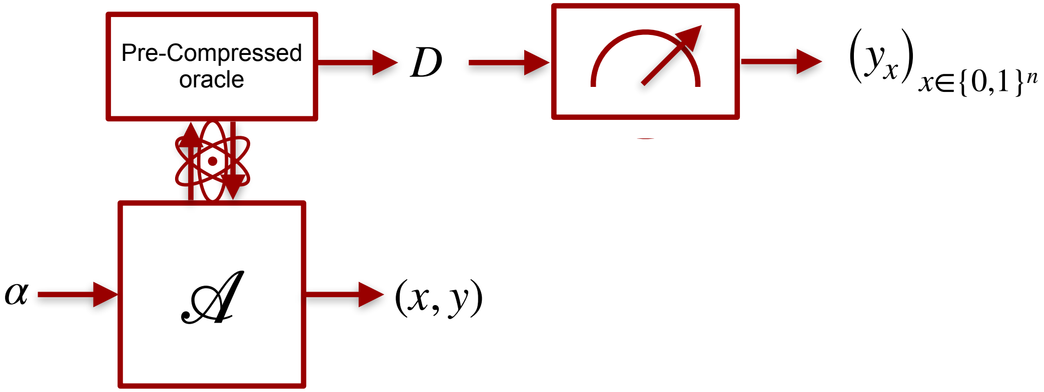
Lemma (Zhandry '18, slightly informal):

Let $\mathscr{A}$ be a quantum oracle algorithm and R a relation, and let f be a random function. Consider the two experiments

i)



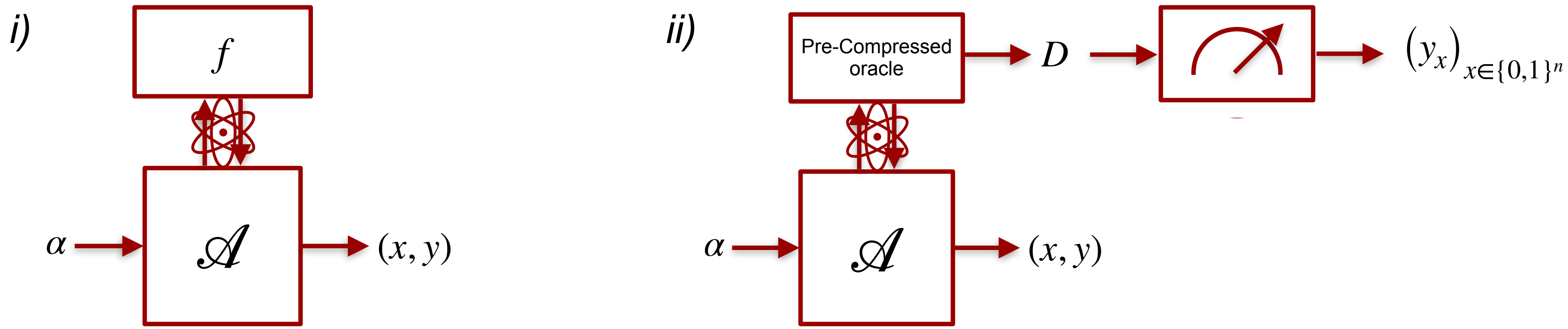
ii)



The fundamental lemma

Lemma (Zhandry '18, slightly informal):

Let $\mathscr{A}$ be a quantum oracle algorithm and R a relation, and let f be a random function. Consider the two experiments



Then

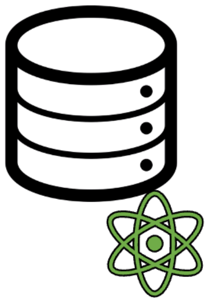
$$\Pr_{i)}[y = f(x) \wedge (x,y) \in R] \leq \Pr_{ii)}[y = y_x \wedge (x,y) \in R] + 2^{-n/2}\,.$$

Query complexity from compressed oracles

Basic idea

Query Lower Bounds

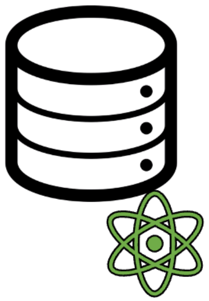
- Intuition: The quantum queries are recorded in the database, an adversary can only learn about the function what is recorded there



- **Theorem:** For any quantum player making q queries, if the database D is measured after the q queries, the probability that it contains a pair $(x, 0^n)$ is at most $O\left(\frac{q^2}{2^n}\right)$.
- **Idea:** Track the norm of the state projected onto D containing a zero. It starts at 0, and every query increases it by at most $\frac{1}{2^{n/2}}$. After q queries, its norm is at most $\frac{q}{2^{n/2}}$. $\square$
- Using newer tools from [Chung Fehr Huang Liao 21], such reasoning is almost classical.

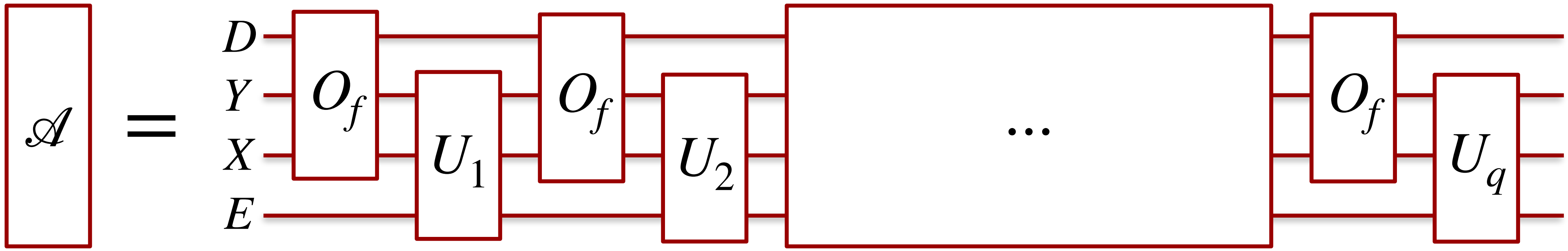
Basic idea

Query Lower Bounds

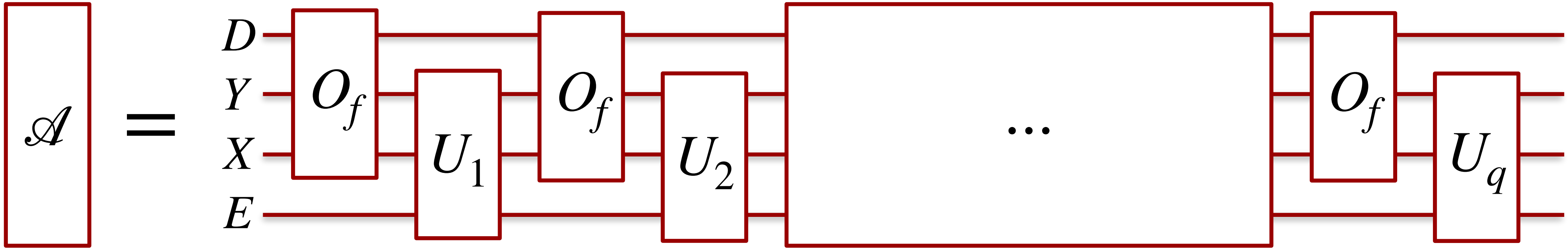


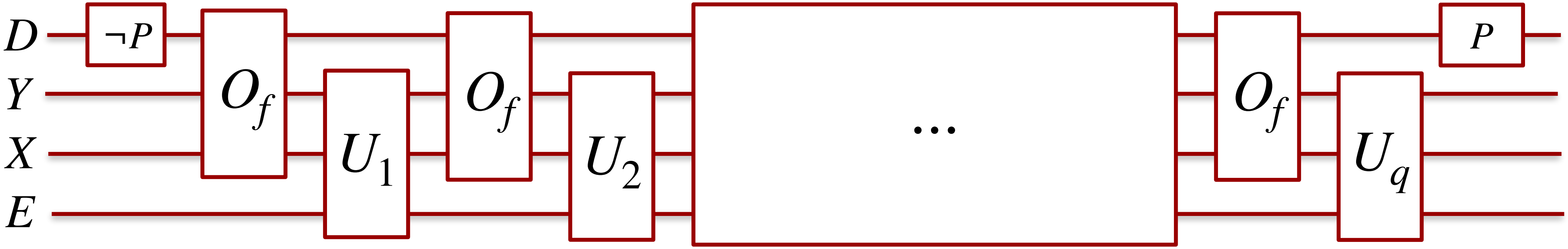
- Intuition: The quantum queries are recorded in the database, an adversary can only learn about the function what is recorded there
- **Theorem:** For any quantum player making q queries, if the database D is measured after the q queries, the probability that it contains a pair $(x, 0^n)$ is at most $O\left(\frac{q^2}{2^n}\right)$.
- **Idea:** Track the norm of the state projected onto D containing a zero. It starts at 0, and every query increases it by at most $\frac{1}{2^{n/2}}$. After q queries, its norm is at most $\frac{q}{2^{n/2}}$. $\square$
- Using newer tools from [Chung Fehr Huang Liao 21], such reasoning is almost classical.

Some more detail (CHFL21)



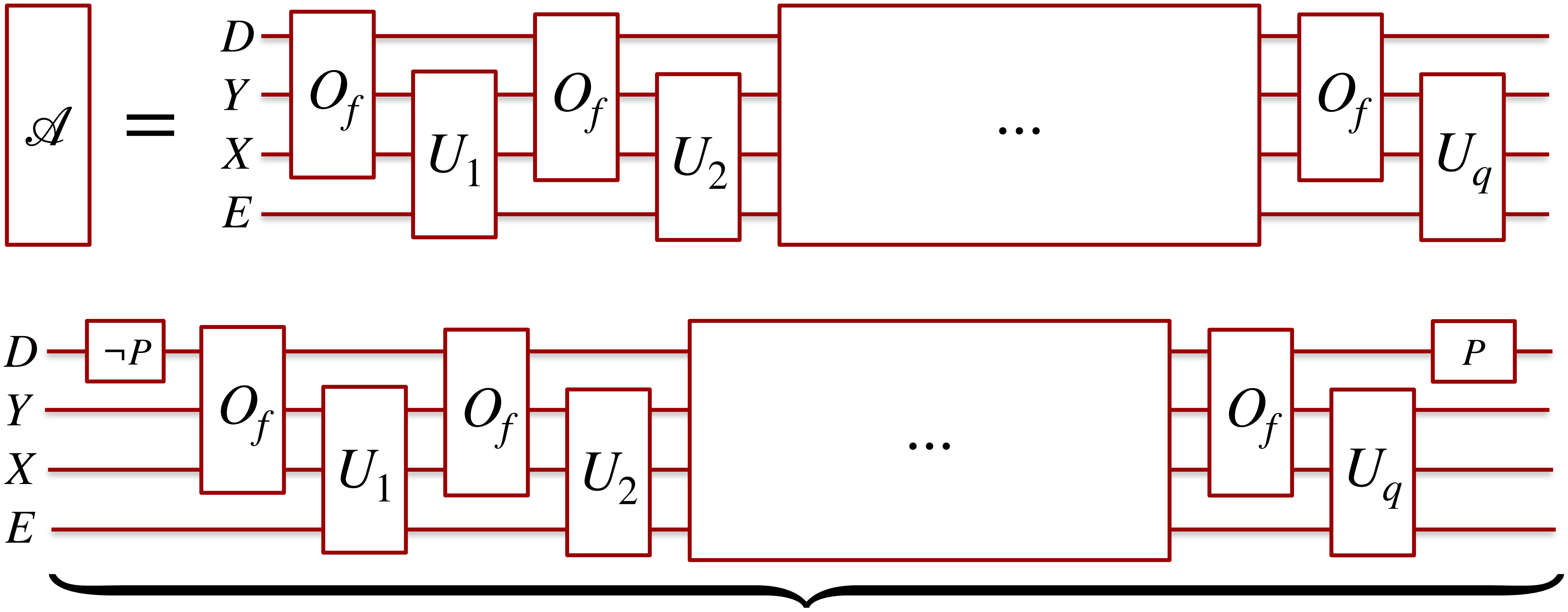
Some more detail (CHFL21)





Example: P the projector on databases containing an output 0^n

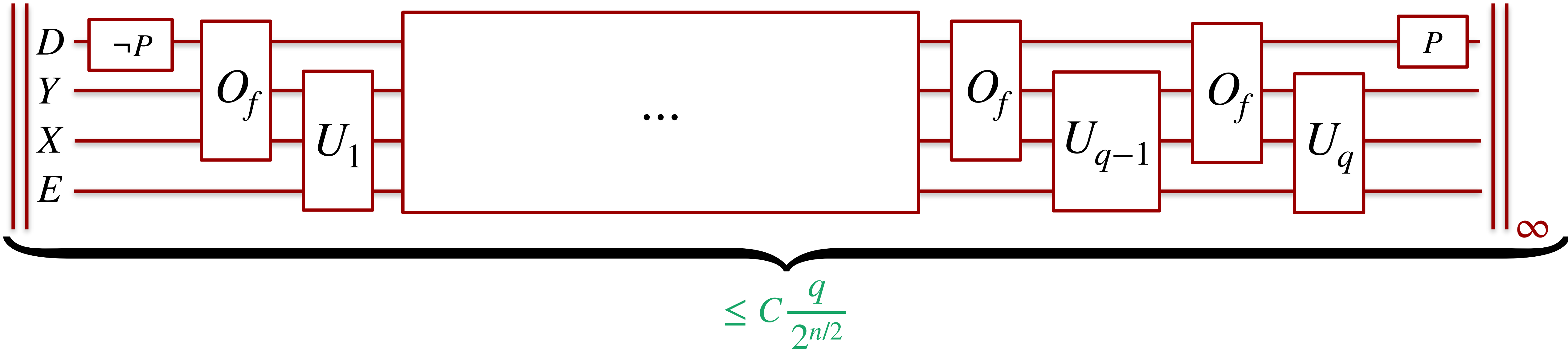
Some more detail (CHFL21)



$$\leq C\frac{q}{2^{n/2}}$$

Example: P the projector on databases containing an output 0^n

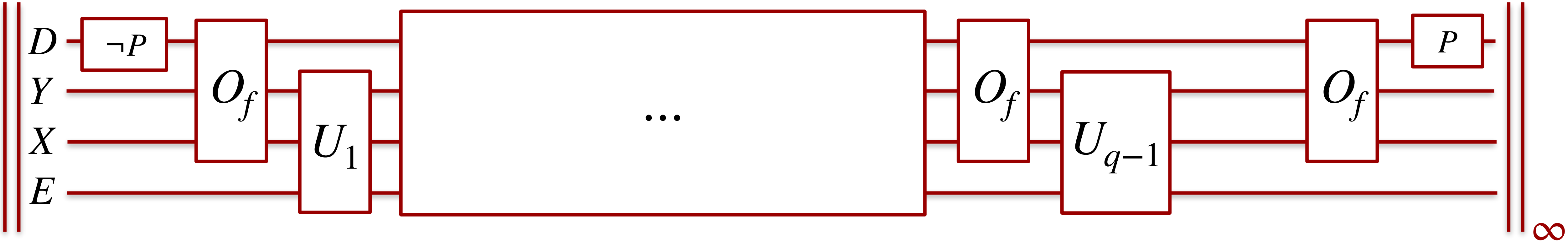
Some more detail (CHFL21)



Some more detail (CHFL21)

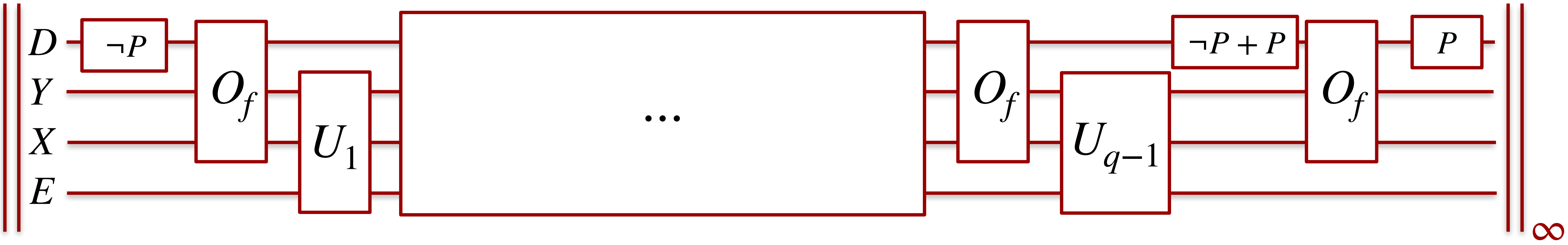
Some more detail (CHFL21)

Some more detail (CHFL21)

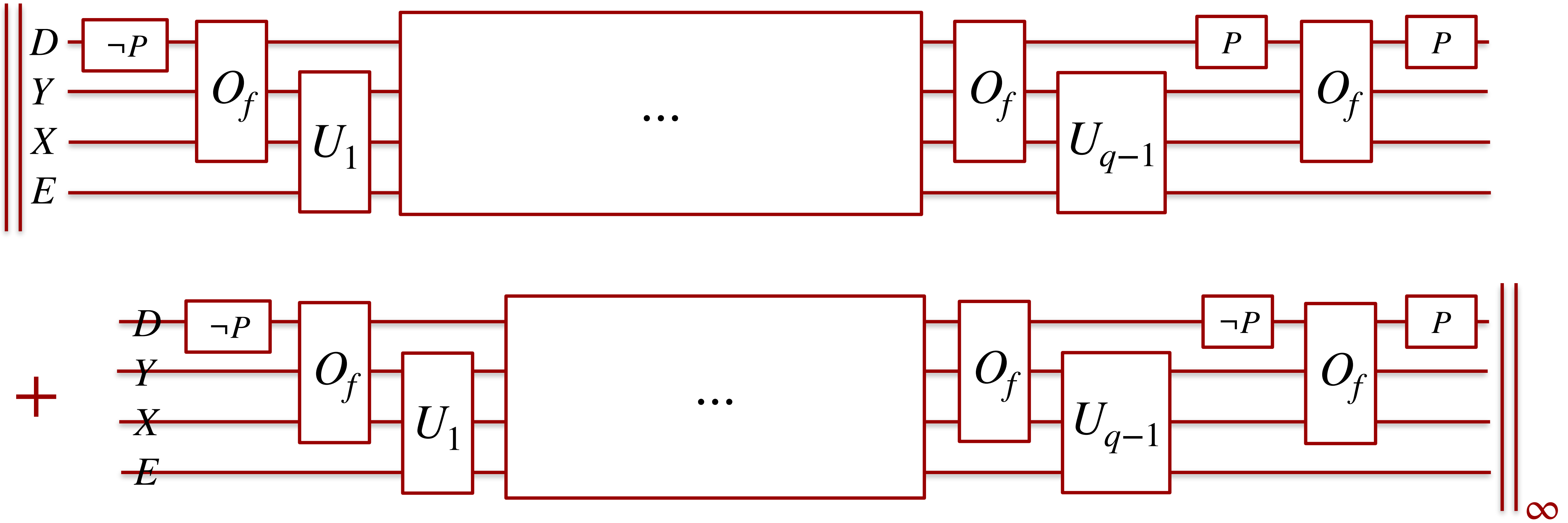


Some more detail (CHFL21)

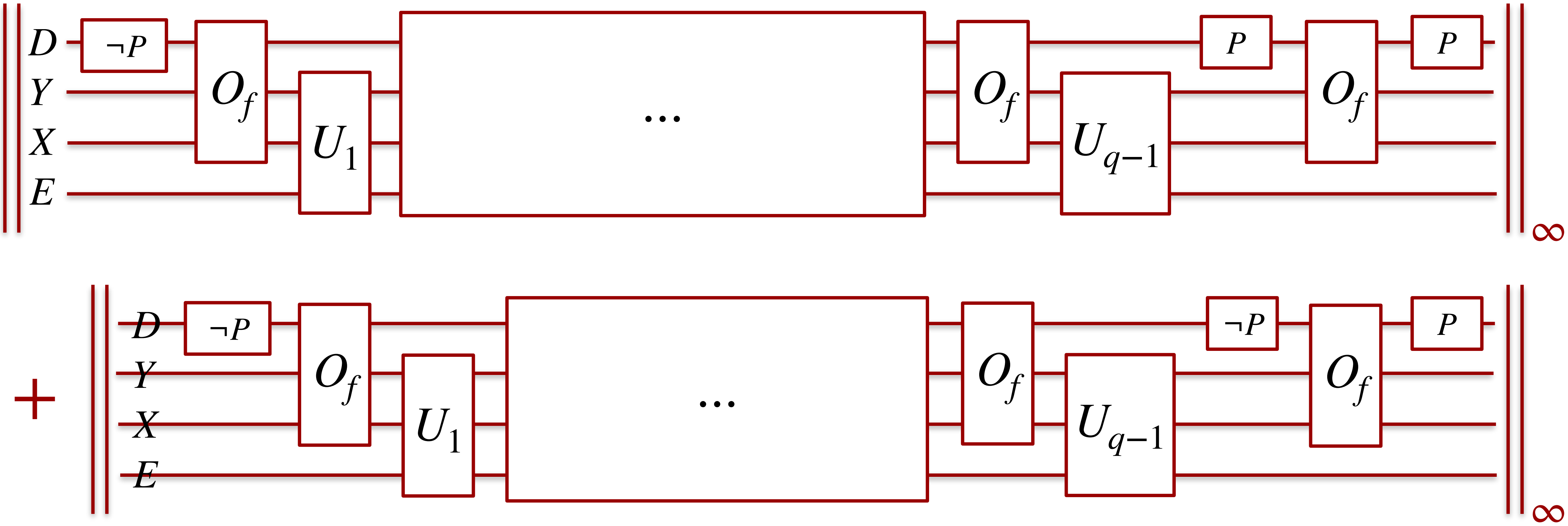
Some more detail (CHFL21)



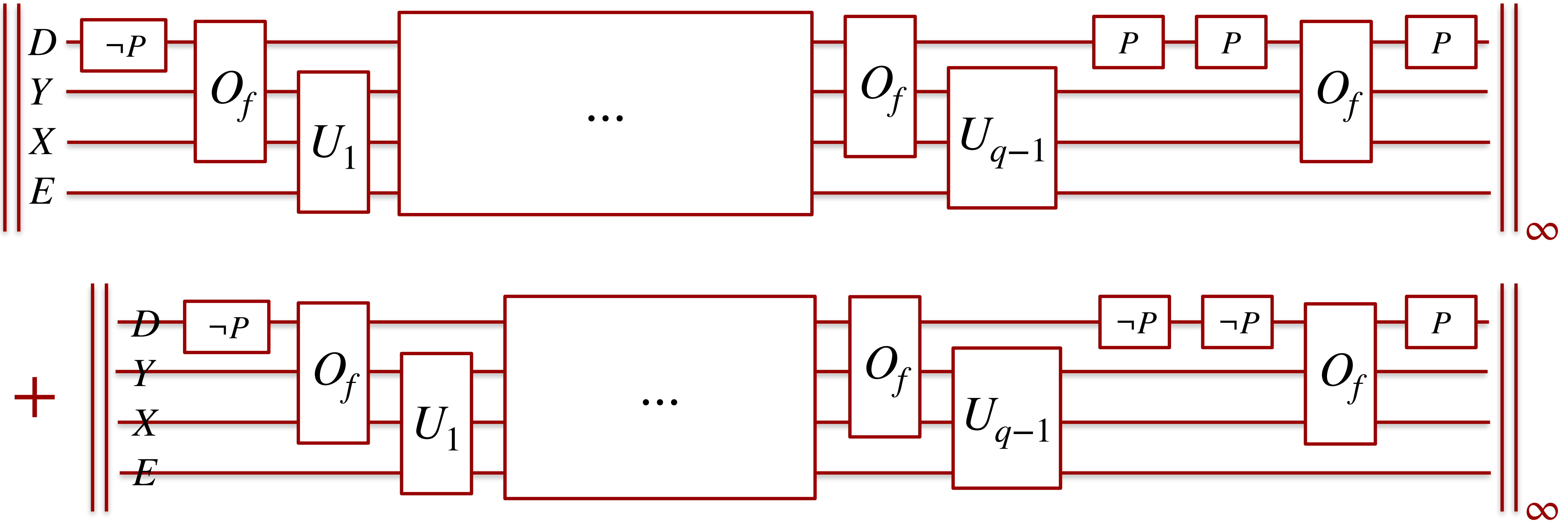
Some more detail (CHFL21)



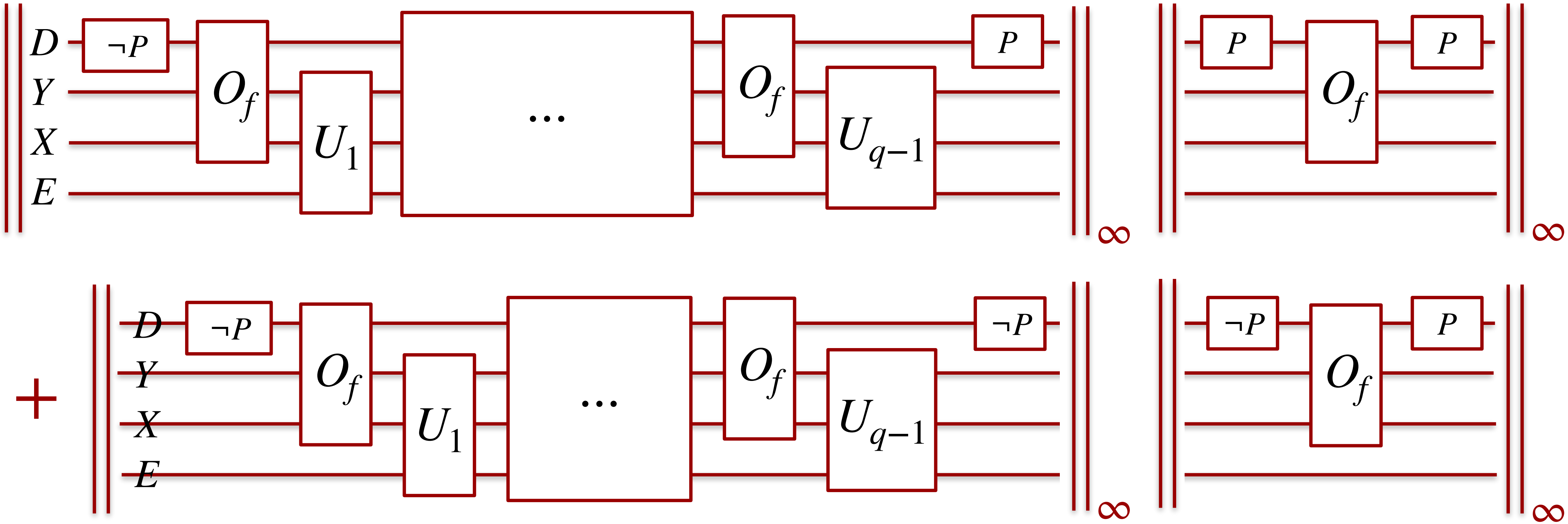
Some more detail (CHFL21)



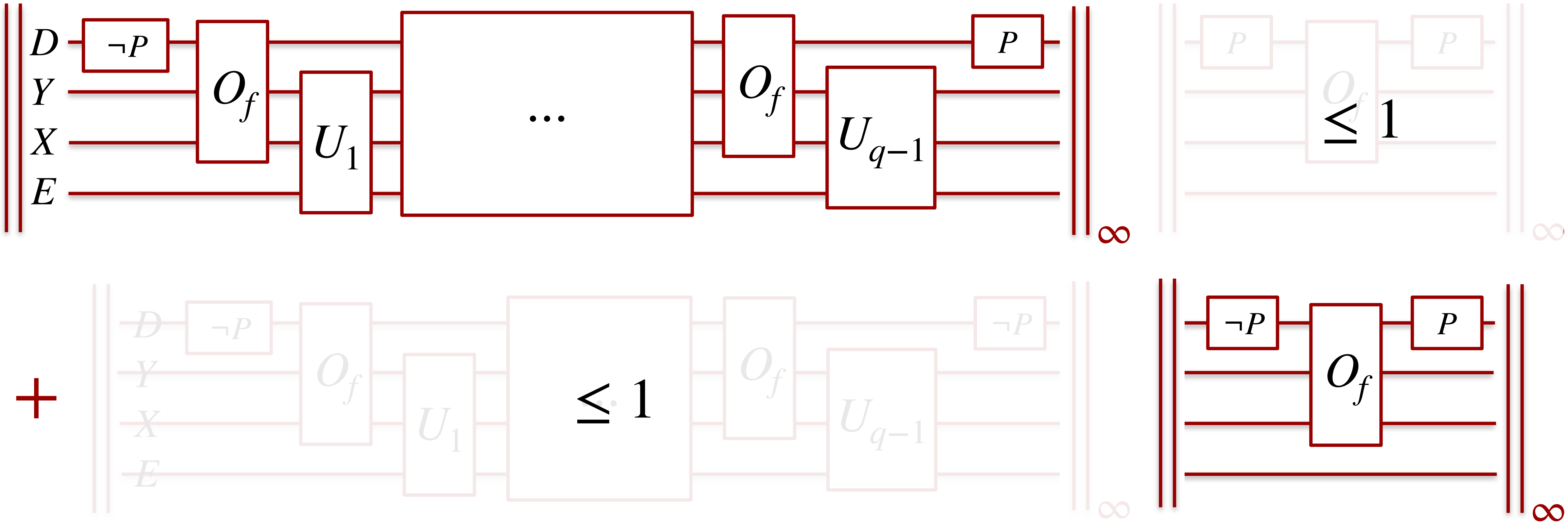
Some more detail (CHFL21)



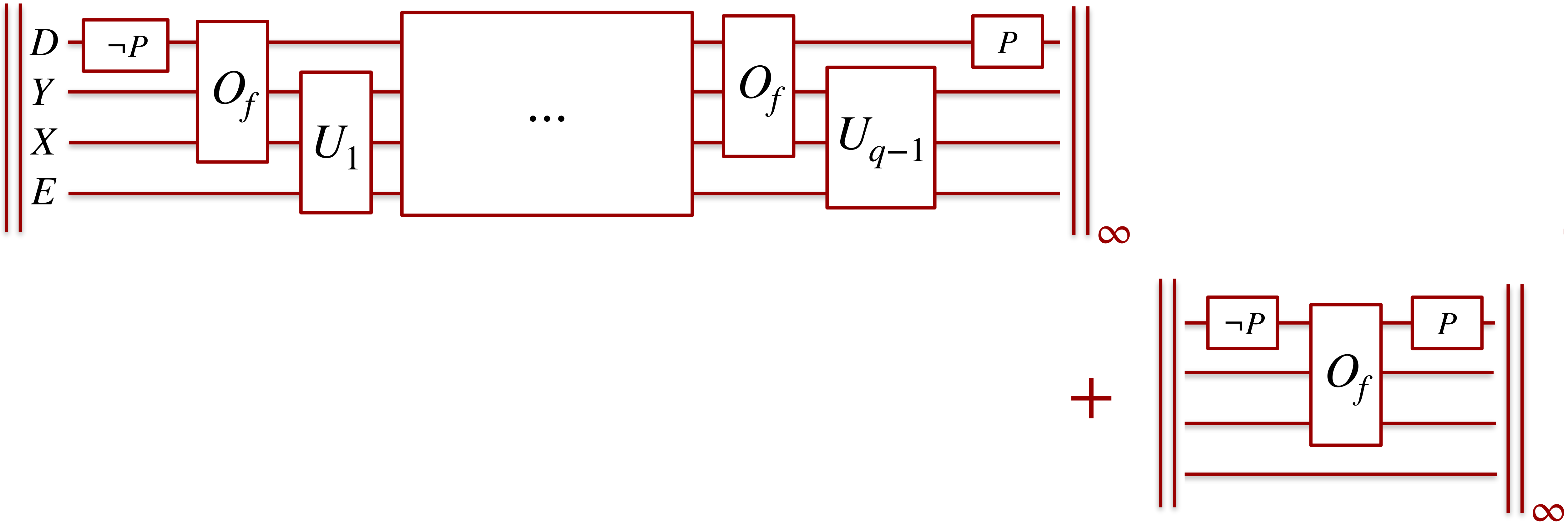
Some more detail (CHFL21)



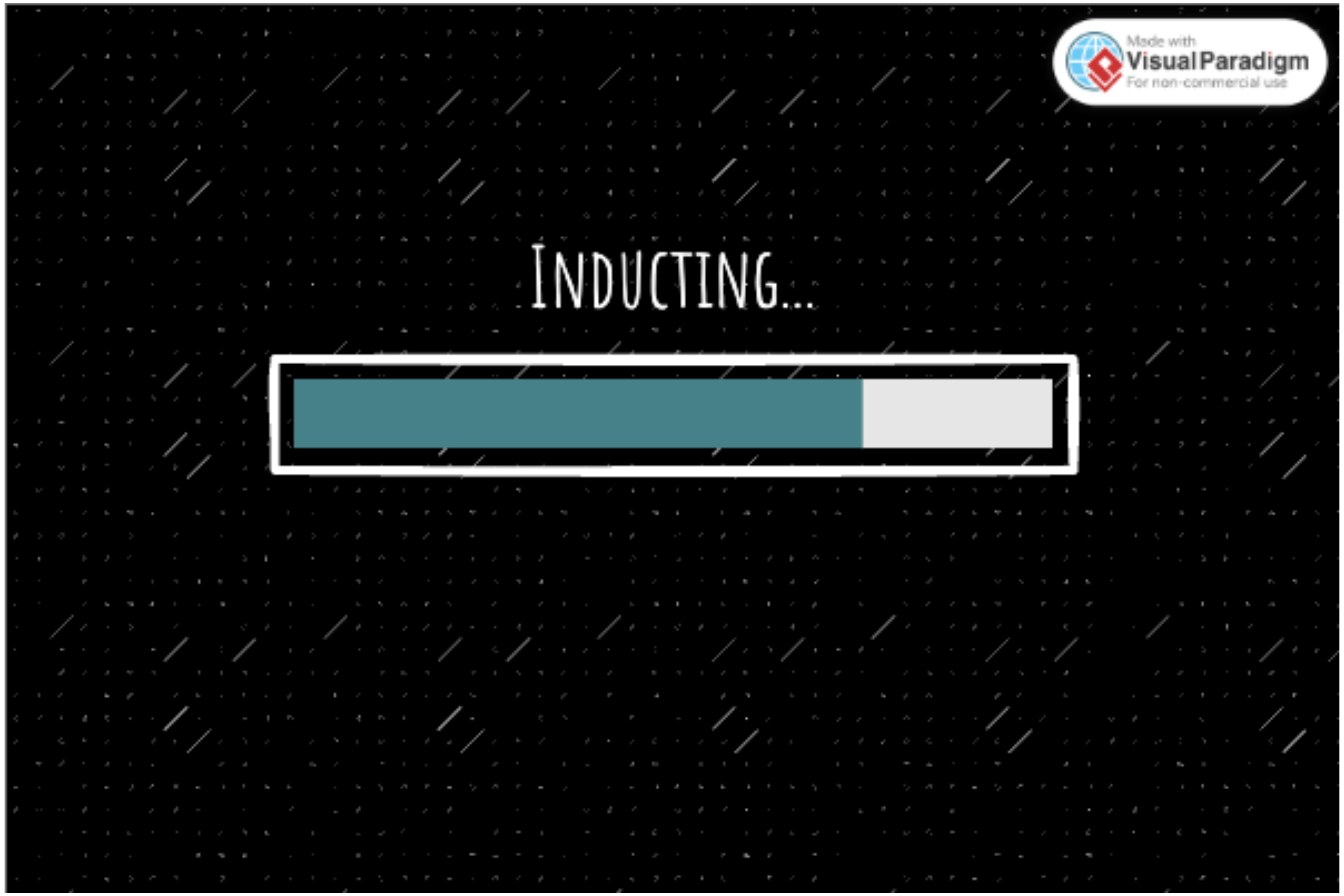
Some more detail (CHFL21)



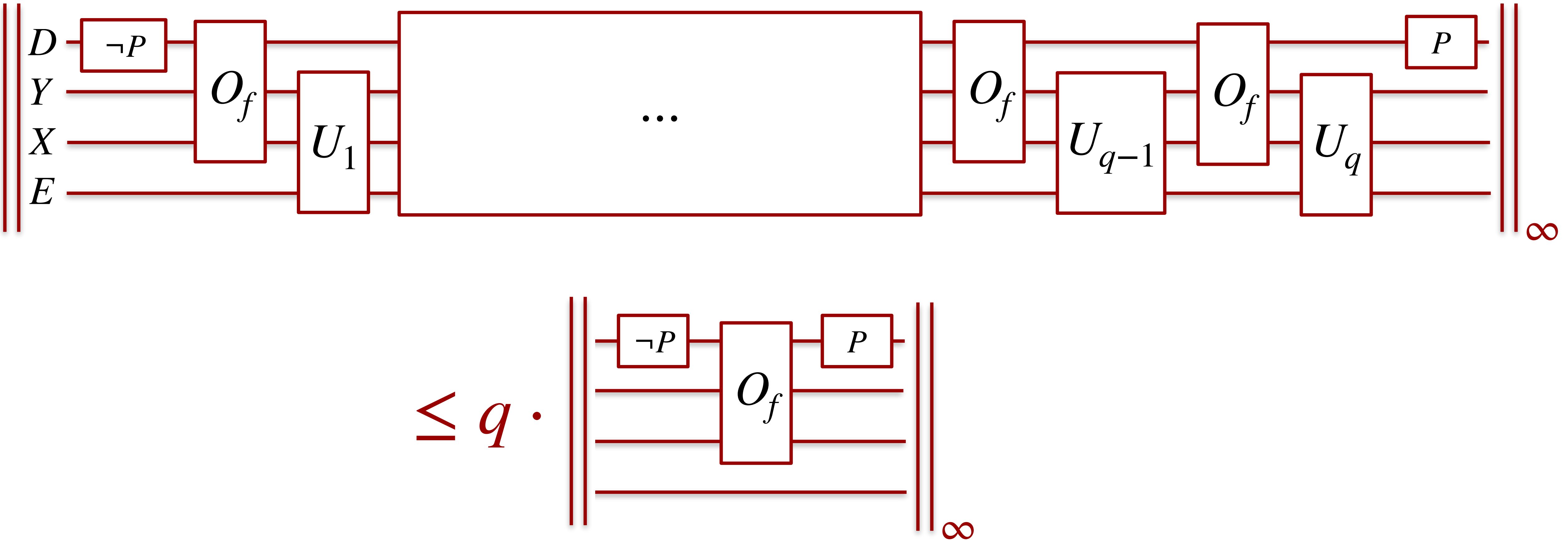
Some more detail (CHFL21)



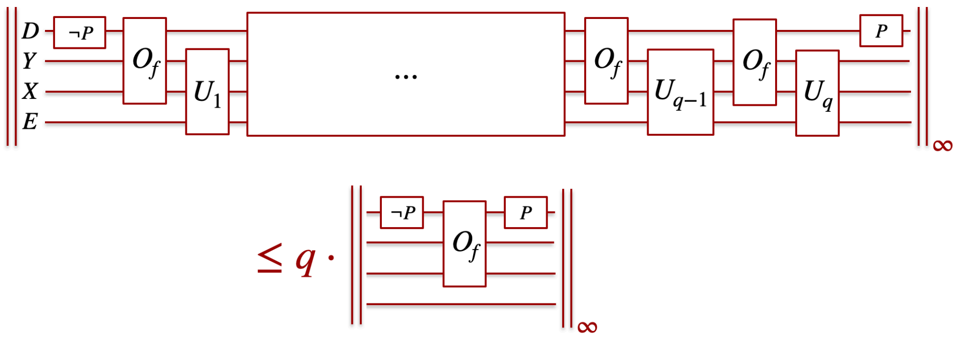
Some more detail (CHFL21)



Some more detail (CHFL21)



Applications



Query complexity of

- preimage finding
- collision finding
- finding (several) (multi-)collisions (Liu&Zhandry ’19)

Allows analyzing

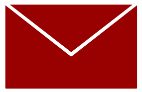
- Proofs of Sequential Work (CHFL21)
- Space-time trade-offs (Hamoudi&Magniez 23)
- NIZKs (Chiesa Manohar Spooner ’19, Don, Fehr, M, Schaffner ’22)

Extractable commitments in the QROM

Commitment schemes

Alice

Bob



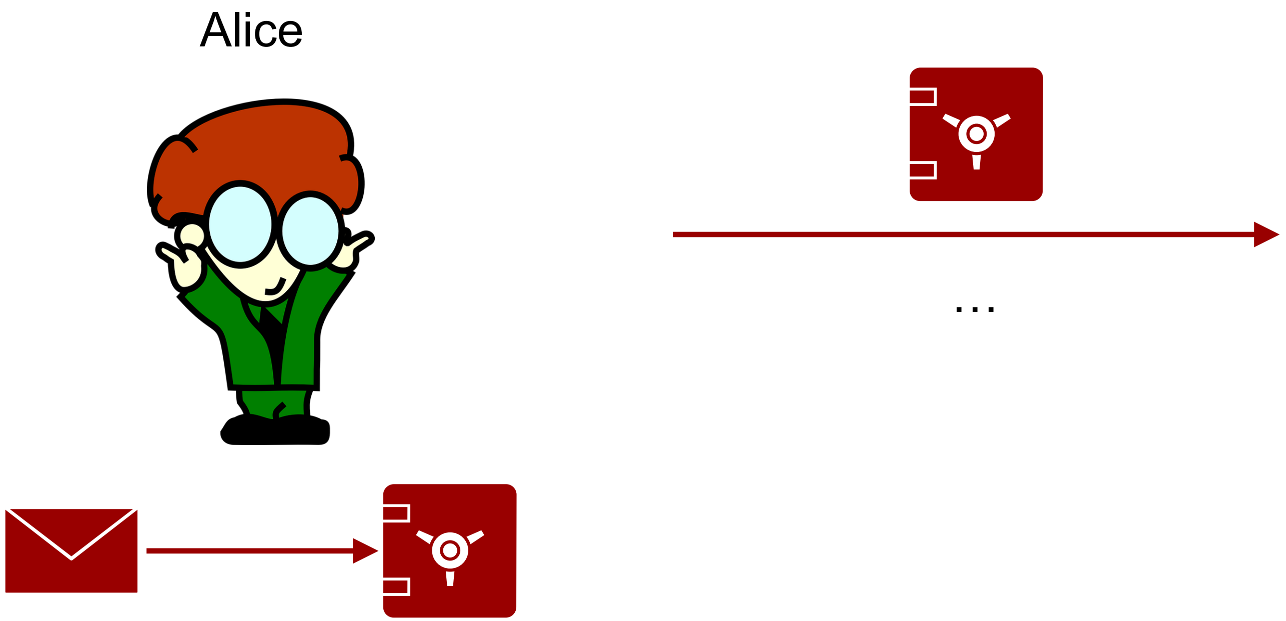
Commitment schemes

Alice

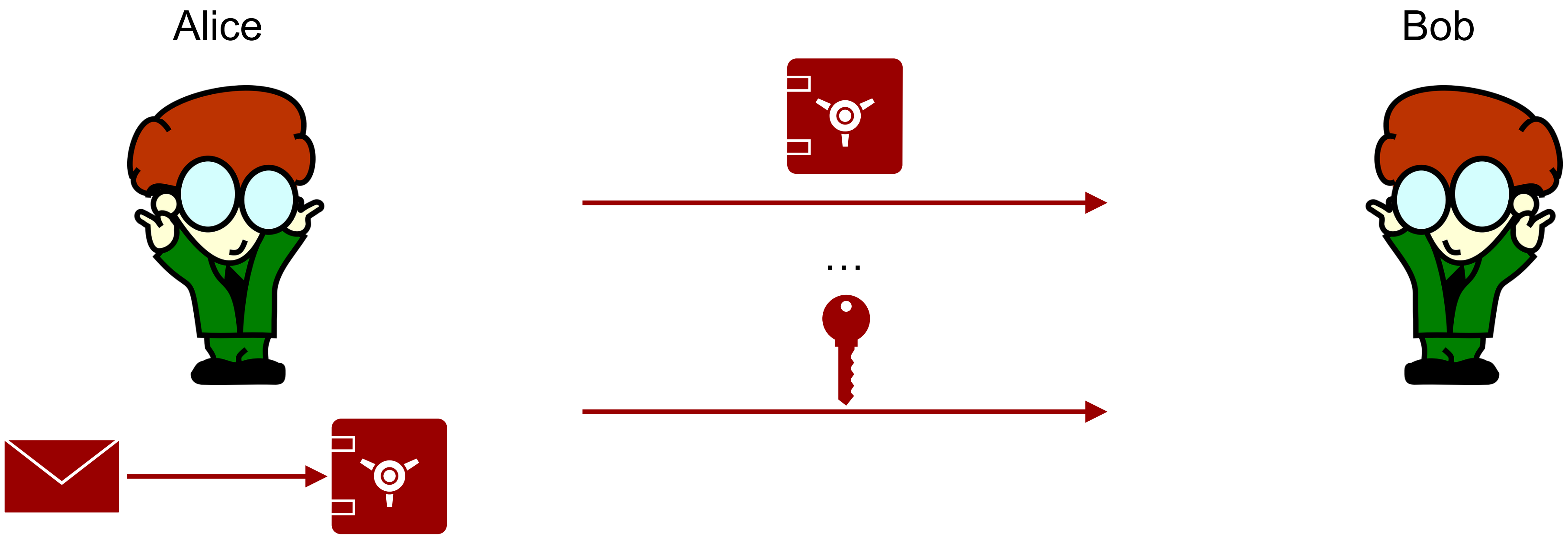
Bob

Commitment schemes

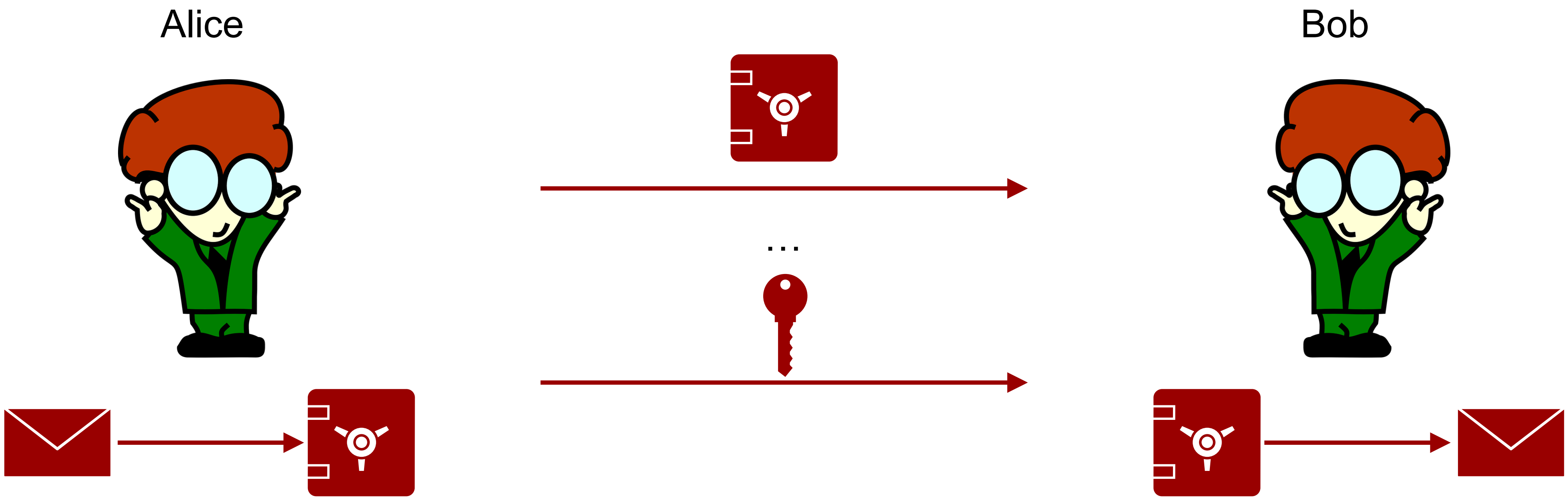
Commitment schemes



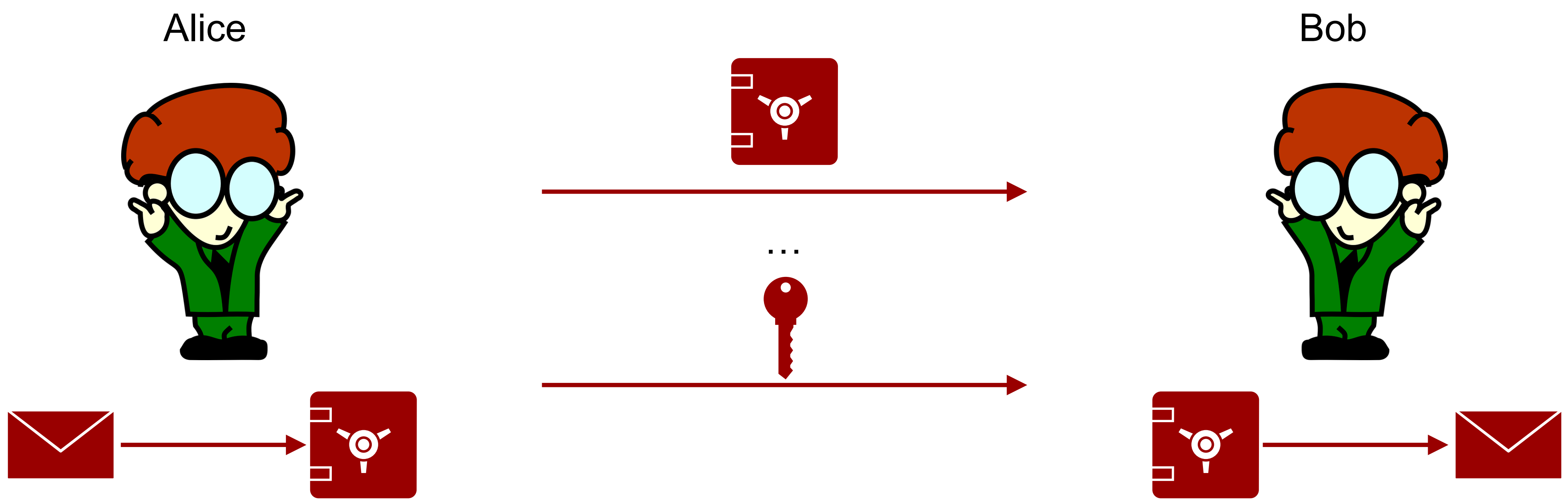
Commitment schemes



Commitment schemes



Commitment schemes



Properties

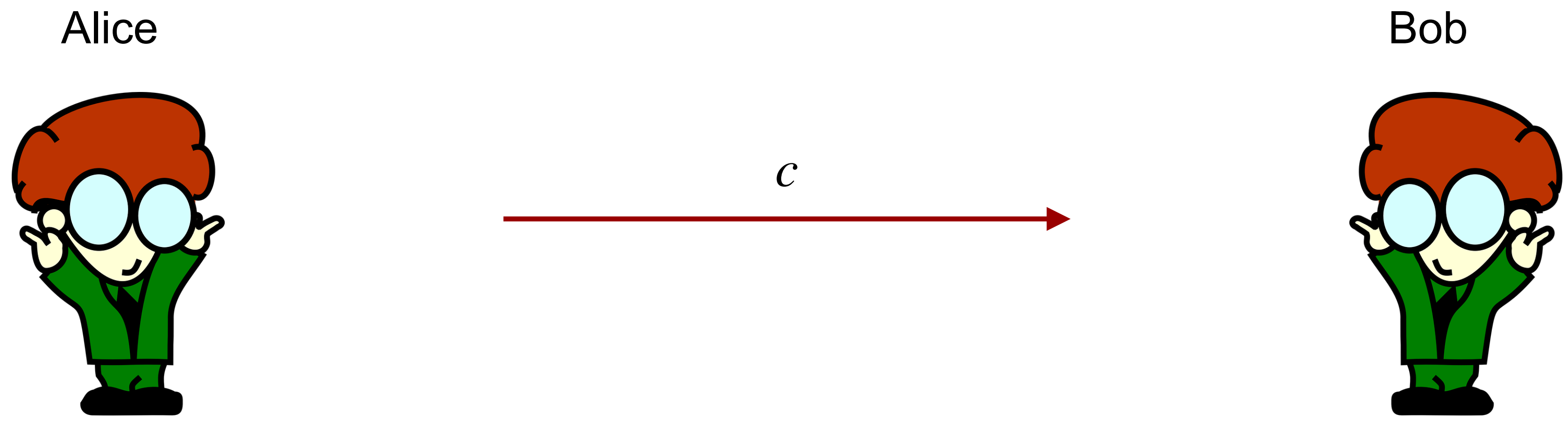
- Hiding: Bob cannot learn without
- Binding: Alice cannot change content of after sending

Hash-based commitments

Alice

Bob

Hash-based commitments



$$r \leftarrow \{0,1\}^{\ell}$$

$$c = H(m, r)$$

Hash-based commitments

Alice

$r \leftarrow \{0,1\}^\ell$

$c = H(m, r)$

c

…

Bob

Hash-based commitments

Alice

$r \leftarrow \{0,1\}^{\ell}$

$c = H(m, r)$

c →

$\dots$

(m, r) →

Bob

$c = H(m, r)?$

Hash-based commitments

Alice

$r \leftarrow \{0,1\}^\ell$

$c = H(m, r)$

c →

$\dots$

(m, r) →

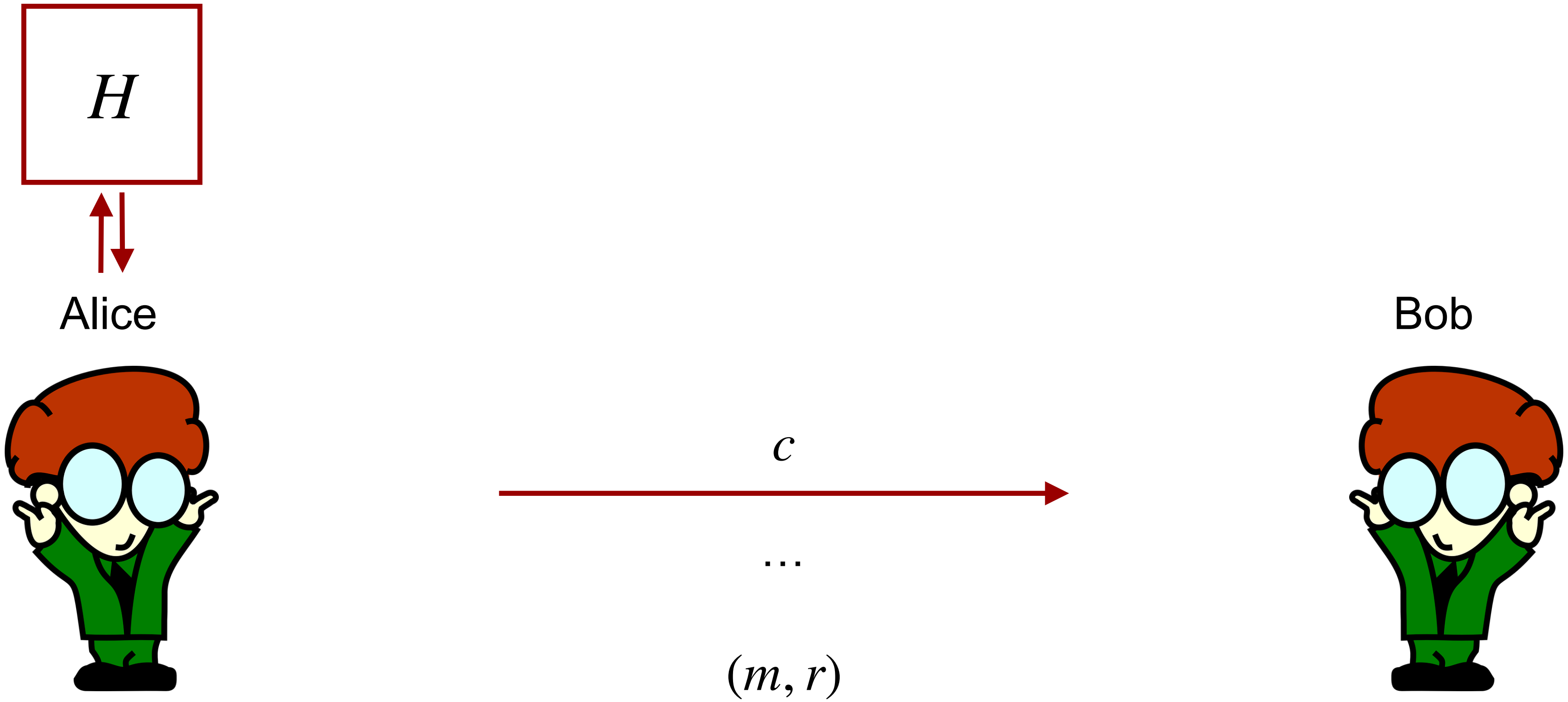
Bob

$c = H(m, r)?$

Properties

- Hiding: Hard to find m given c (preimage resistance)
- Binding: Hard to find pairs (m, r), (m', r') with $m \neq m'$ and $H(m, r) = H(m', r')$ (collision resistance)

Extractable commitments in the ROM

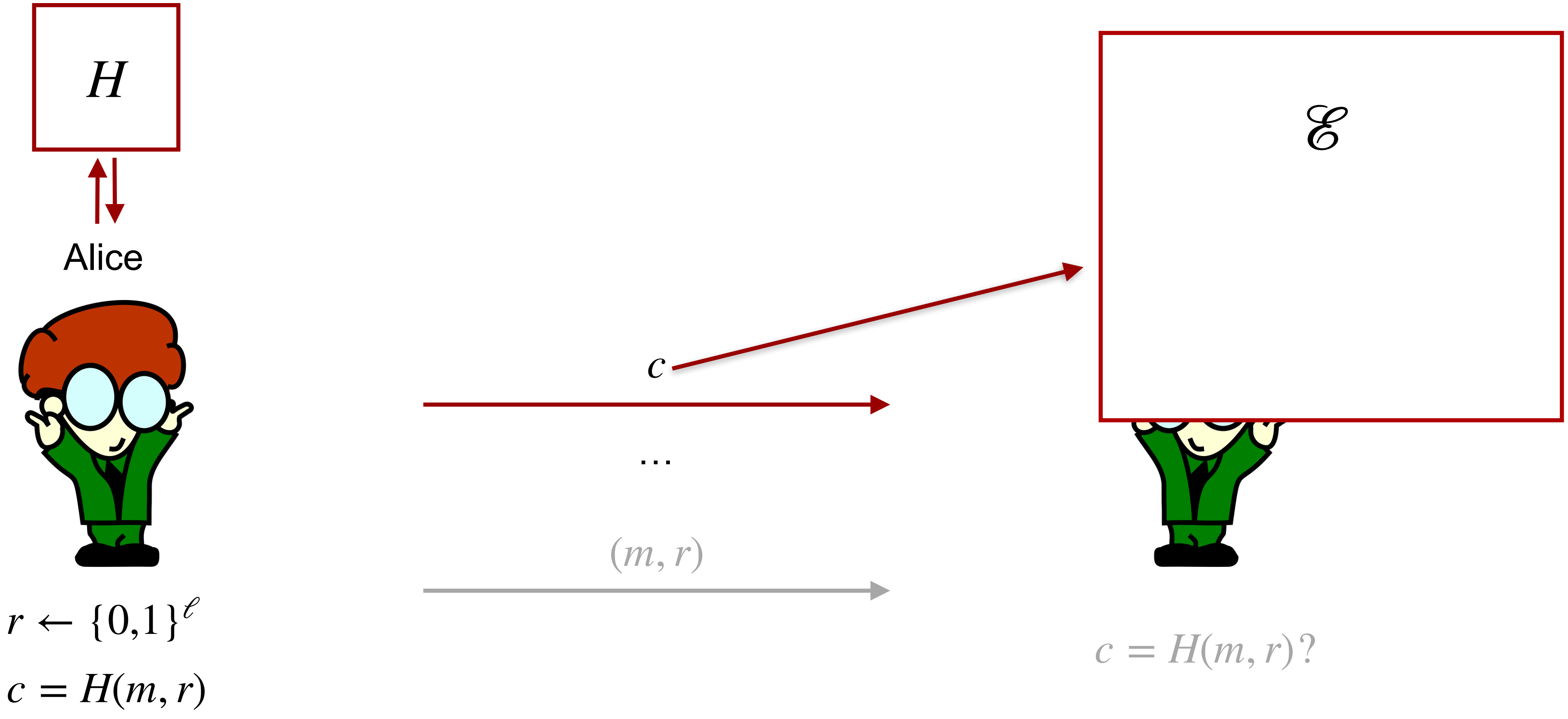


$r \leftarrow \{0,1\}^\ell$

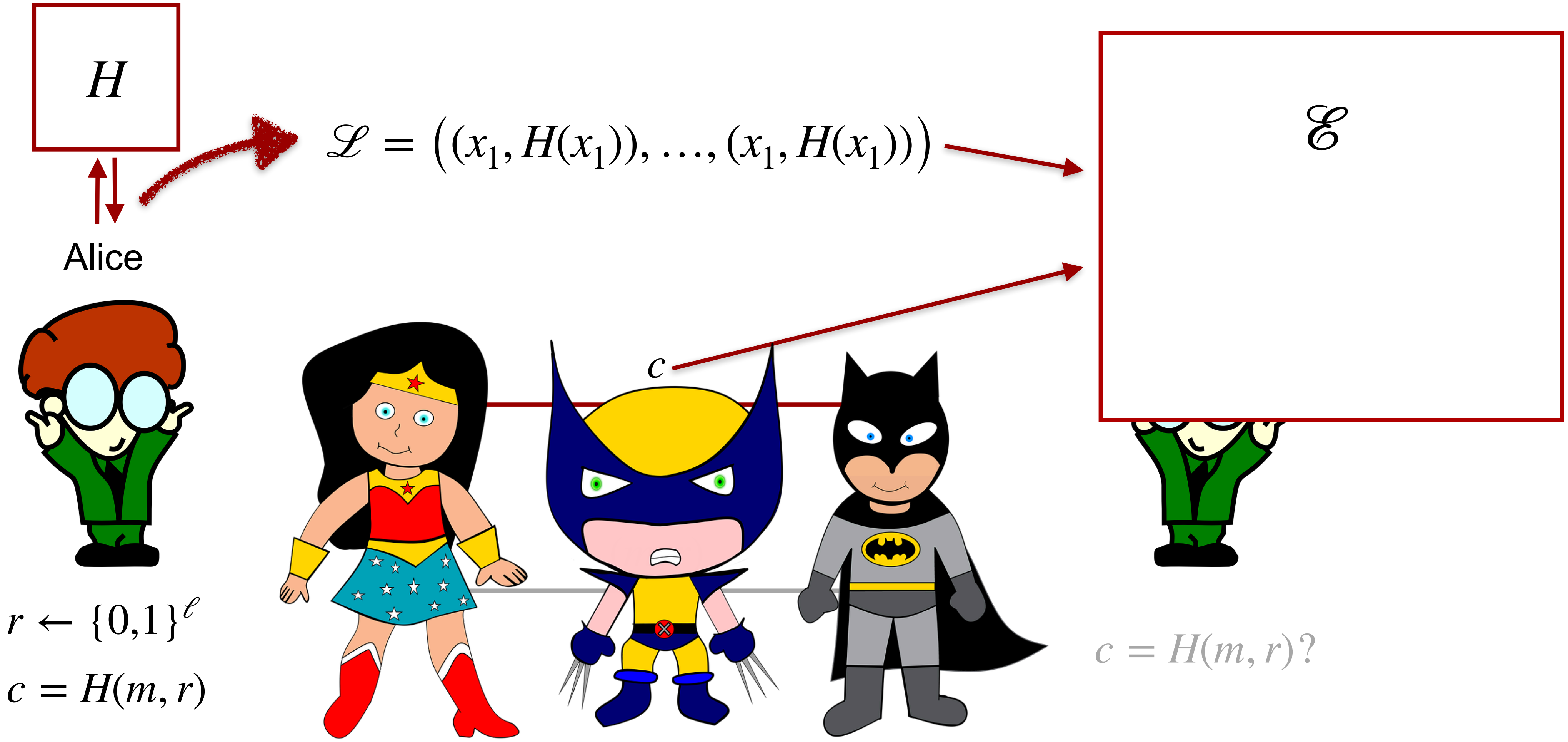
$c = H(m, r)$

$c = H(m, r)?$

Extractable commitments in the ROM



Extractable commitments in the ROM



Extractable commitments in the ROM

H

Alice

$\mathscr{L} = \big((x_1, H(x_1)), \ldots, (x_1, H(x_1))\big)$

c

$\ldots$

(m, r)

$r \leftarrow \{0,1\}^\ell$

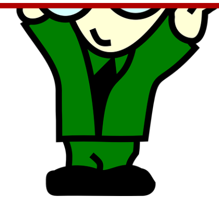
$c = H(m, r)$

$\mathscr{E}$:

Find query

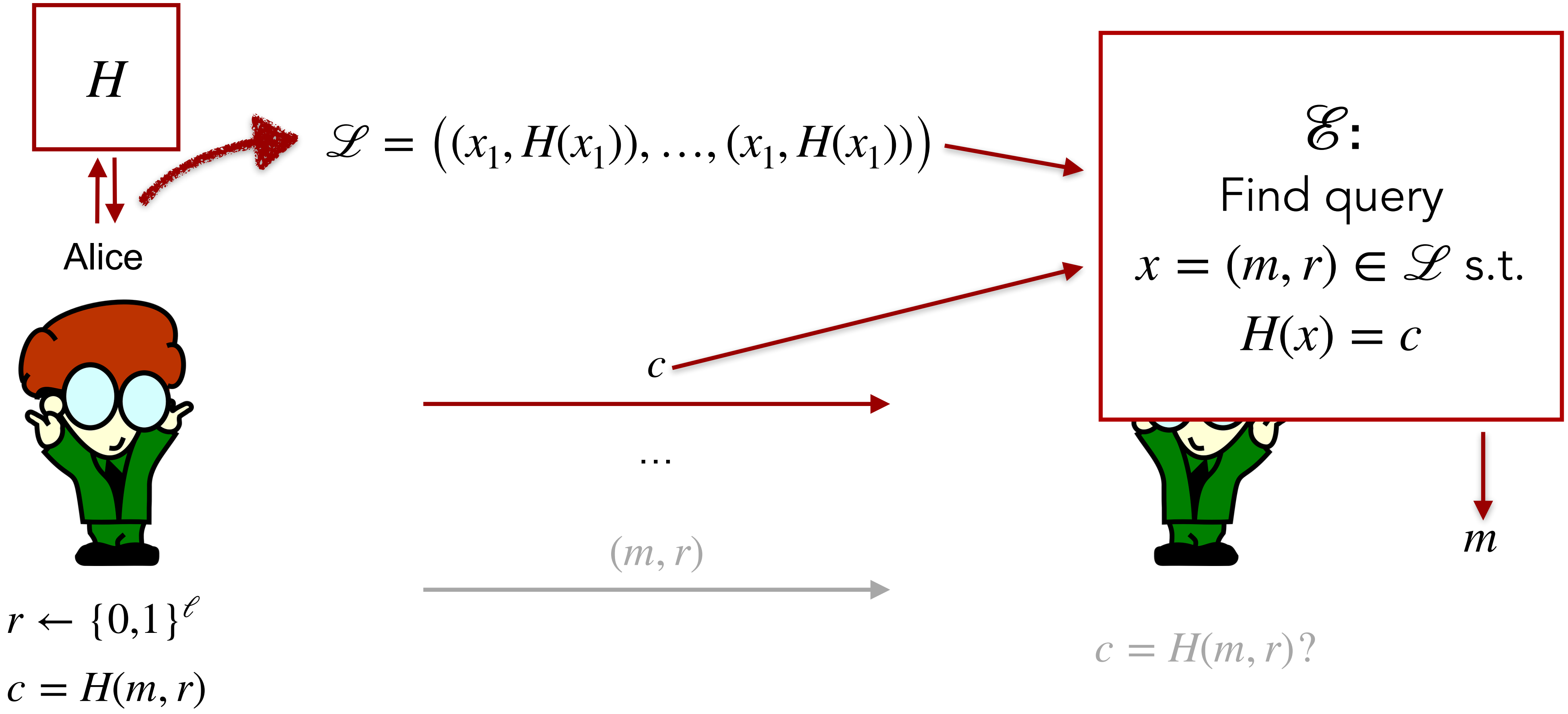
$x = (m, r) \in \mathscr{L}$ s.t.

$H(x) = c$

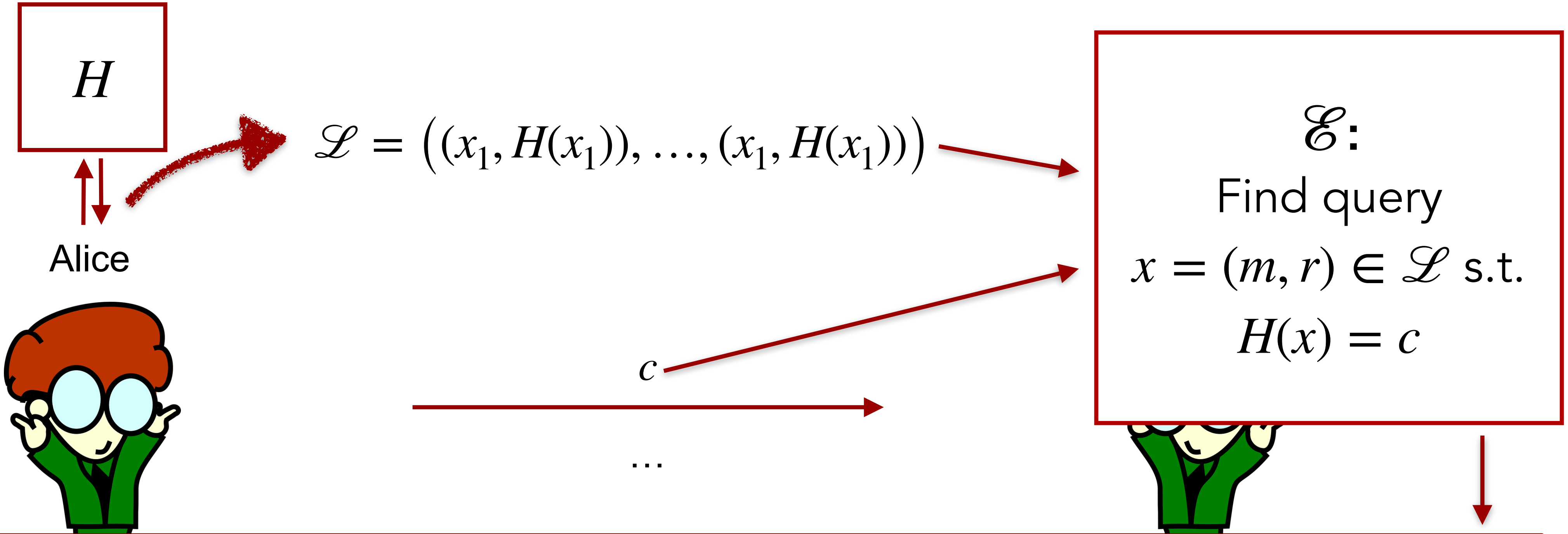


$c = H(m, r)?$

Extractable commitments in the ROM



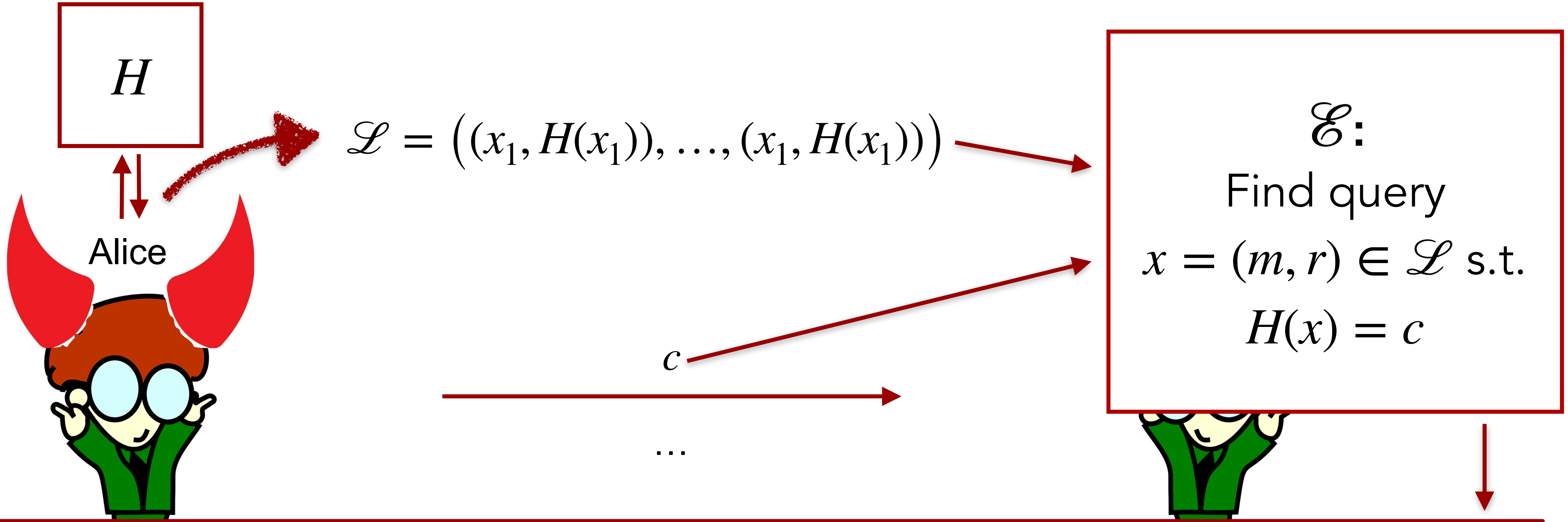
Extractable commitments in the ROM



Why does it work?

- Hiding: Hard to find m given c (preimage resistance)
- Binding: Hard to find pairs (m, r), (m', r') with $m \neq m'$ and $H(m, r) = H(m', r')$ (collision resistance)

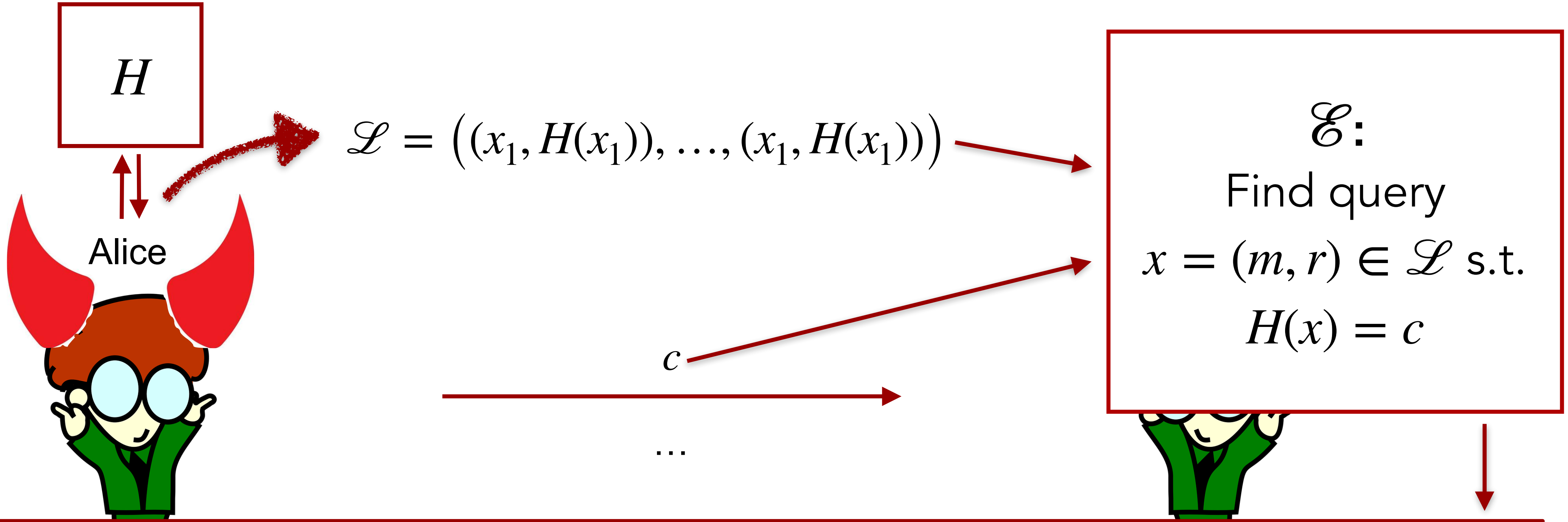
Extractable commitments in the ROM



Why does it work?

- Hiding: Hard to find m given c (preimage resistance)
- Binding: Hard to find pairs (m, r), (m', r') with $m \neq m'$ and $H(m, r) = H(m', r')$ (collision resistance)

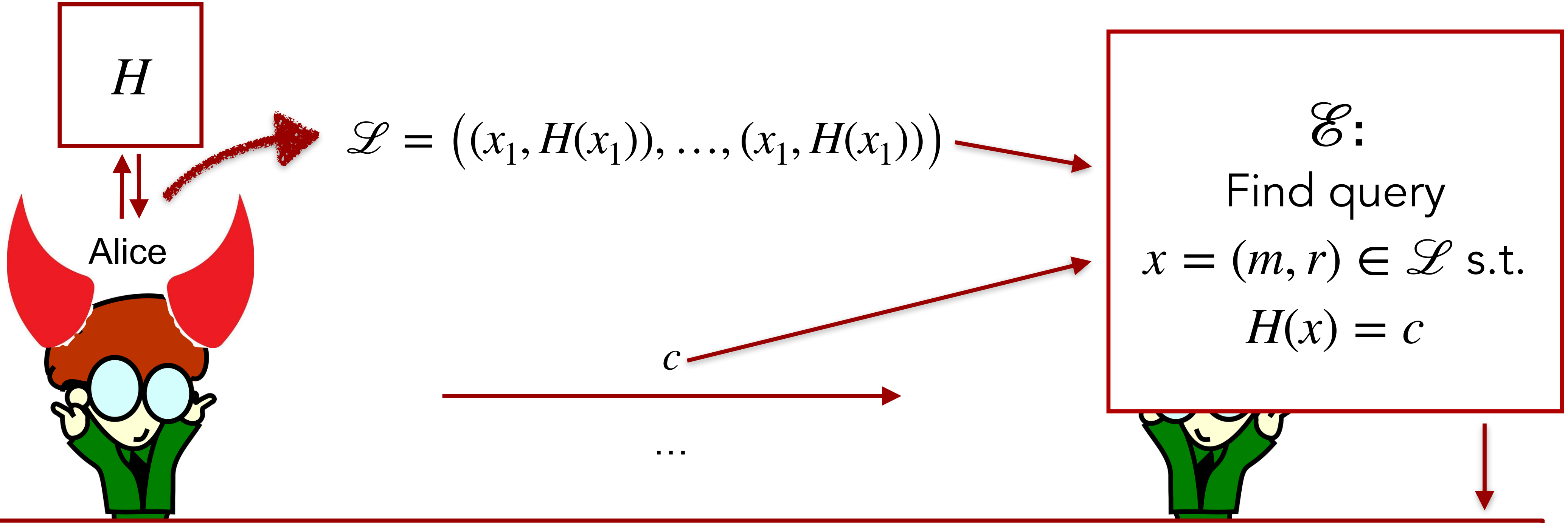
Extractable commitments in the ROM



Why does it work?

- Hiding: Hard to find m given c (preimage resistance) $\Rightarrow$ if m is not in $\mathscr{L}$, Alice can't open.
- Binding: Hard to find pairs (m, r), (m', r') with $m \neq m'$ and $H(m, r) = H(m', r')$ (collision resistance)

Extractable commitments in the ROM



Why does it work?

- Hiding: Hard to find m given c (preimage resistance) $\Rightarrow$ if m is not in $\mathscr{L}$, Alice can't open.
- Binding: Hard to find pairs (m, r), (m', r') with $m \neq m'$ and $H(m, r) = H(m', r')$ (collision resistance) $\Rightarrow$ extracted m is unique.

Extractable commitments in the QROM

H

Alice

$r \leftarrow \{0,1\}^\ell$

$c = H(m, r)$

c

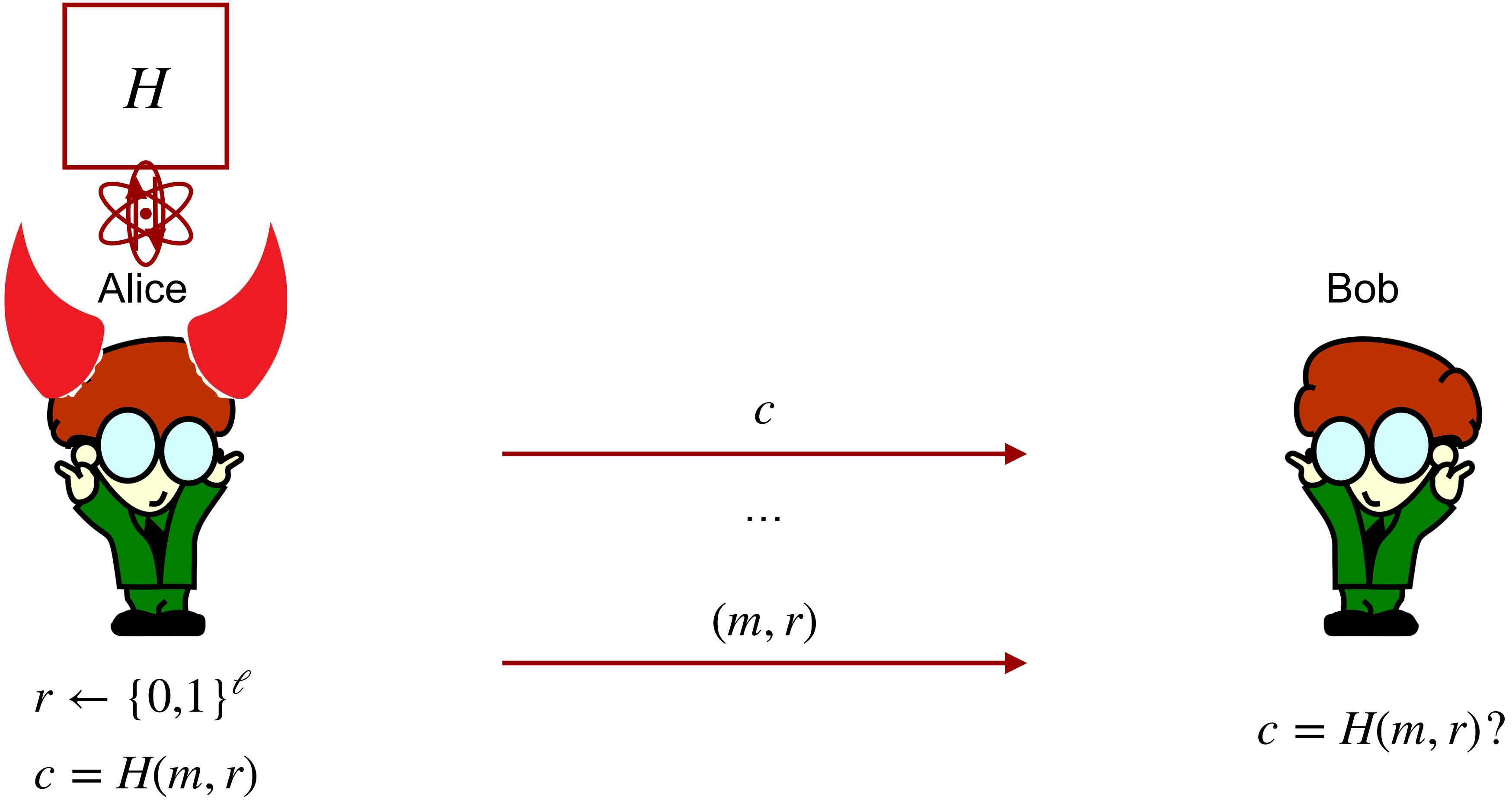
…

(m, r)

Bob

$c = H(m, r)?$

Extractable commitments in the QROM

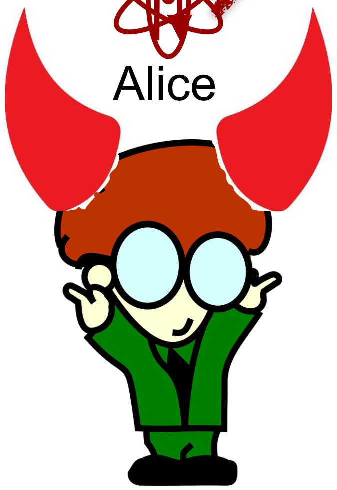


Extractable commitments in the QROM

H

$\mathscr{L} = \big((x_1, H(x_1)), \ldots, (x_1, H(x_1))\big)$?????

Alice



Bob

c

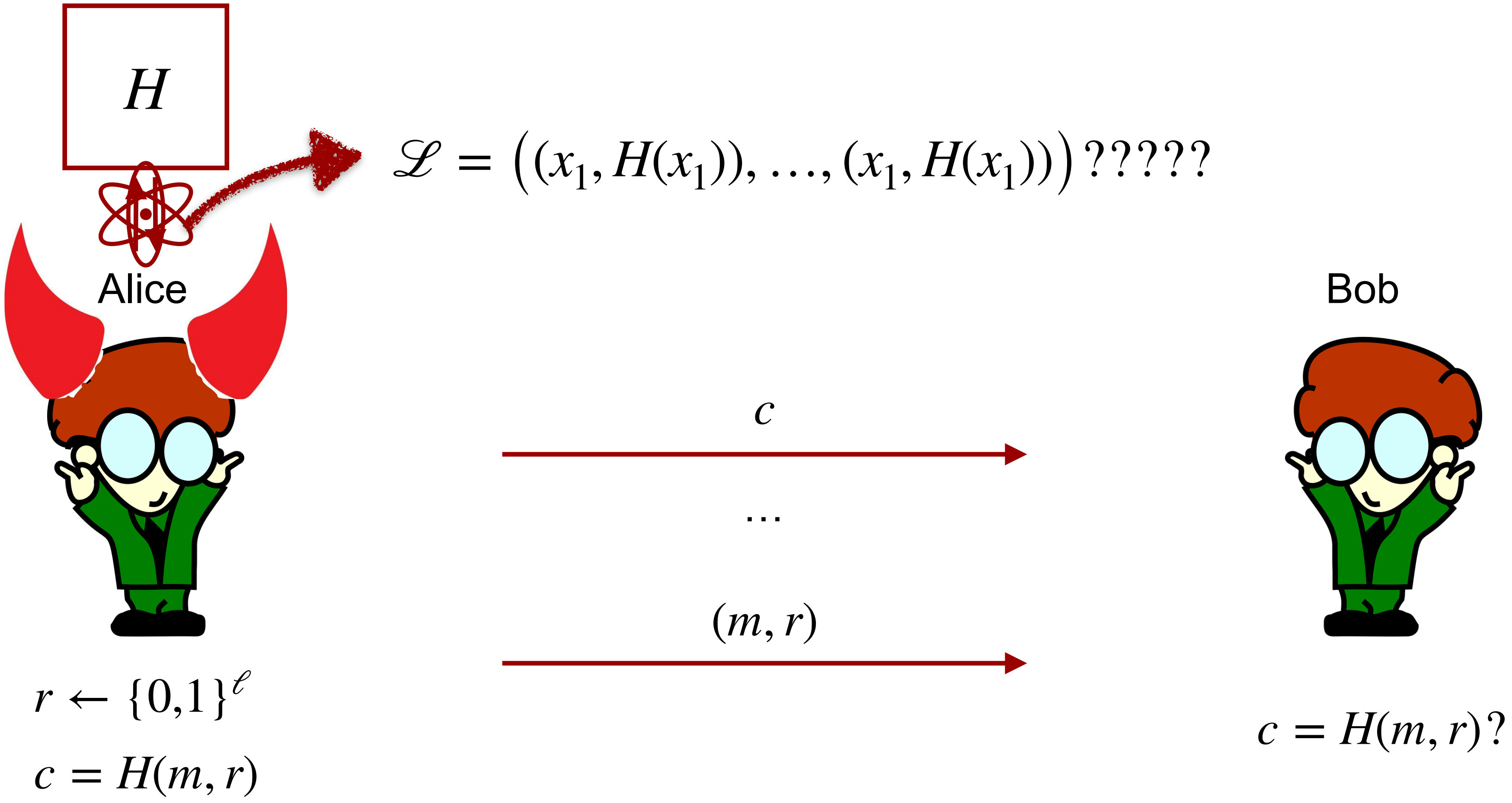
$\ldots$

(m, r)

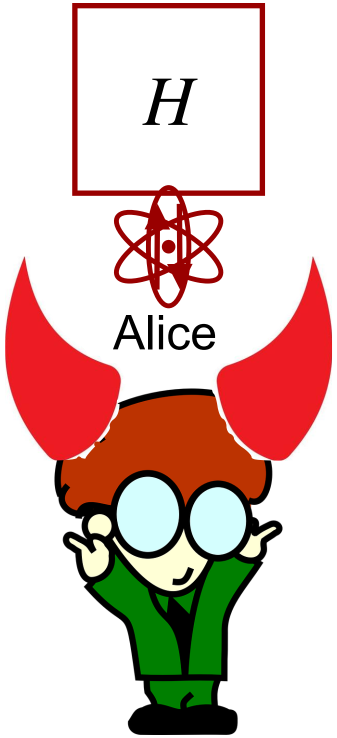
$r \leftarrow \{0,1\}^{\ell}$

$c = H(m, r)?$

Extractable commitments in the QROM



Extractable commitments in the QROM



$r \leftarrow \{0,1\}^{\ell}$

$c = H(m, r)$

c →

…

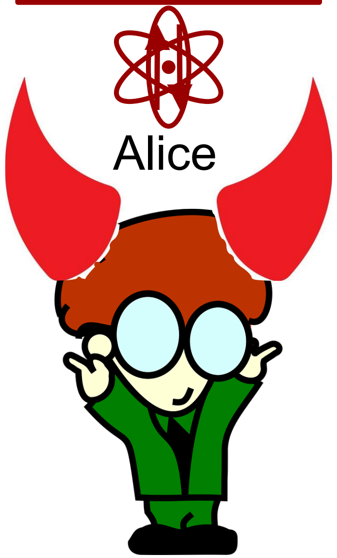
(m, r) →

Bob

$c = H(m, r)?$

Extractable commitments in the QROM

Pre-Compressed oracle



Bob

c →

…

(m, r) →

$r \leftarrow \{0,1\}^{\ell}$

$c = H(m, r)$

$c = H(m, r)?$

Extractable commitments in the QROM

Pre-Compressed oracle D

Alice

$r \leftarrow \{0,1\}^\ell$

$c = H(m, r)$

c

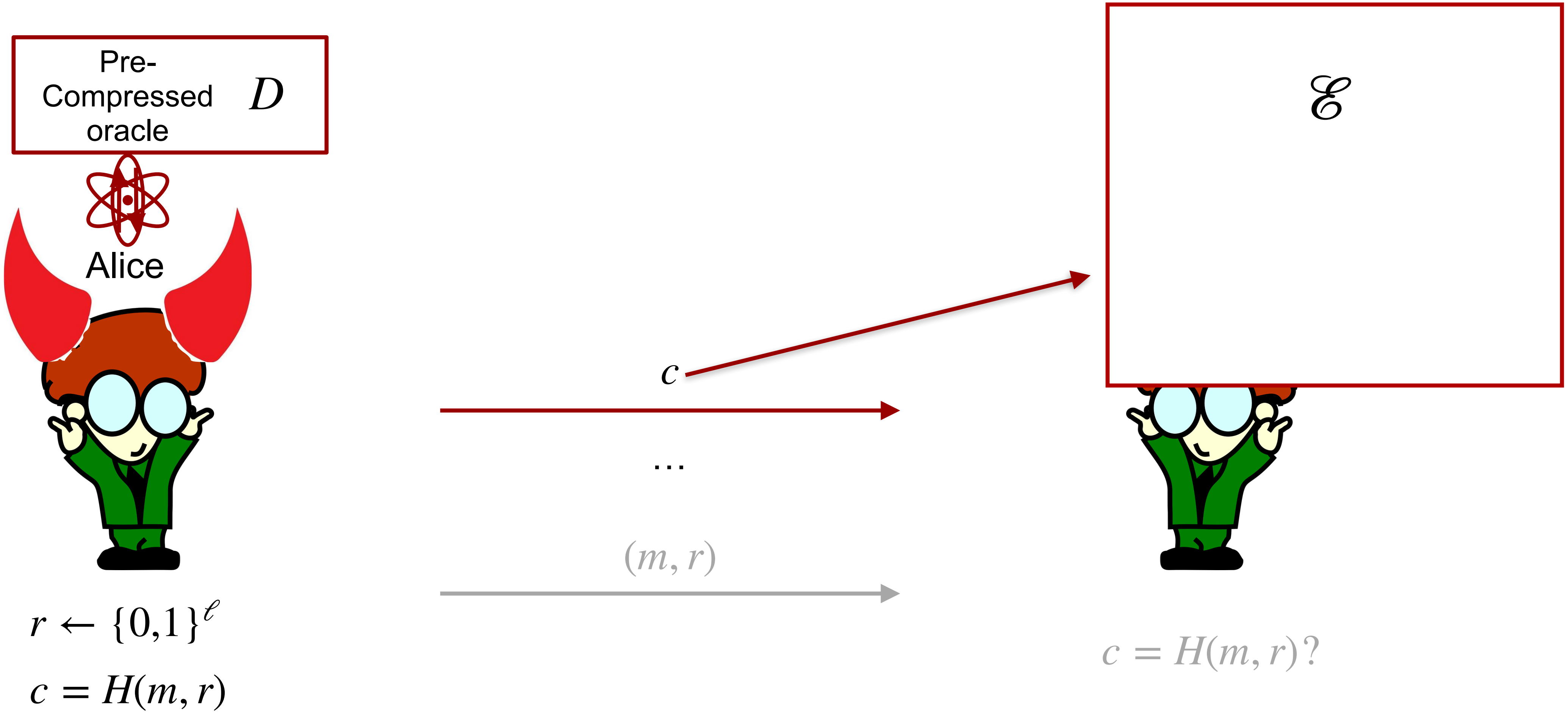
…

(m, r)

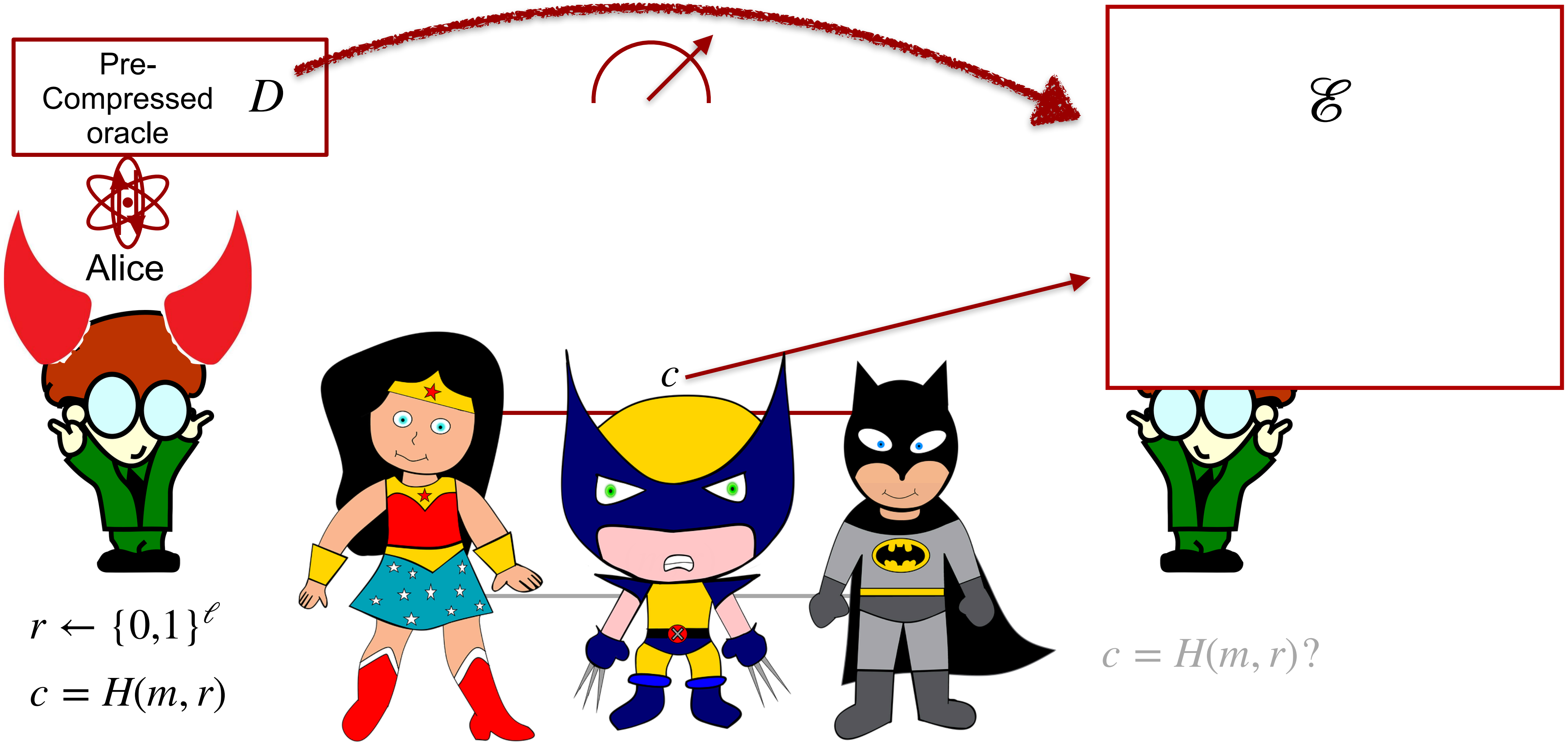
Bob

$c = H(m, r)?$

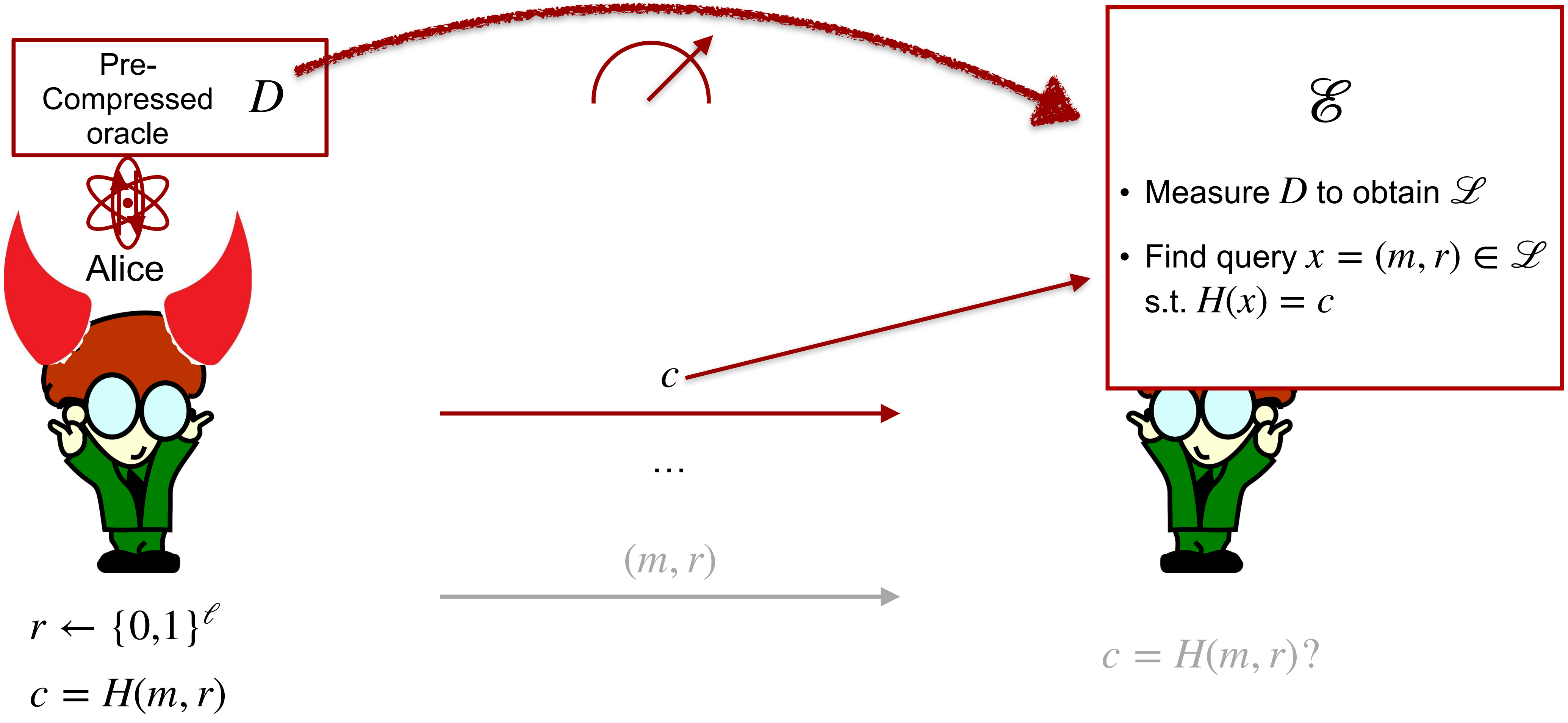
Extractable commitments in the QROM



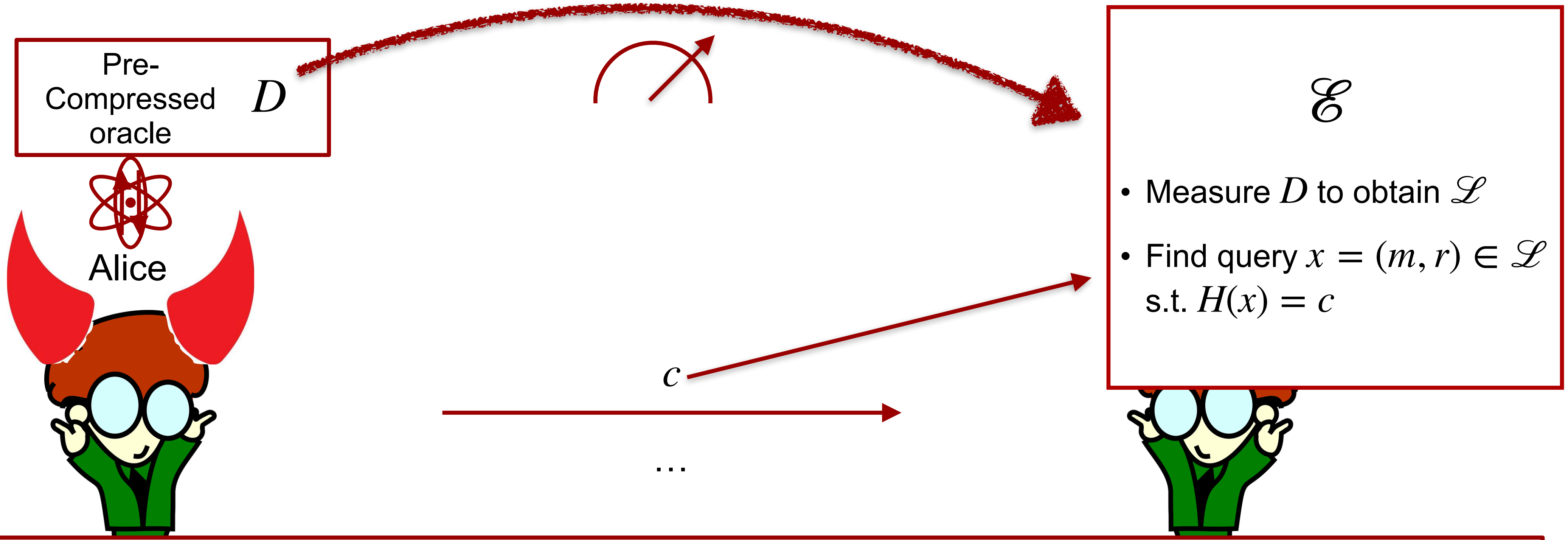
Extractable commitments in the QROM



Extractable commitments in the QROM



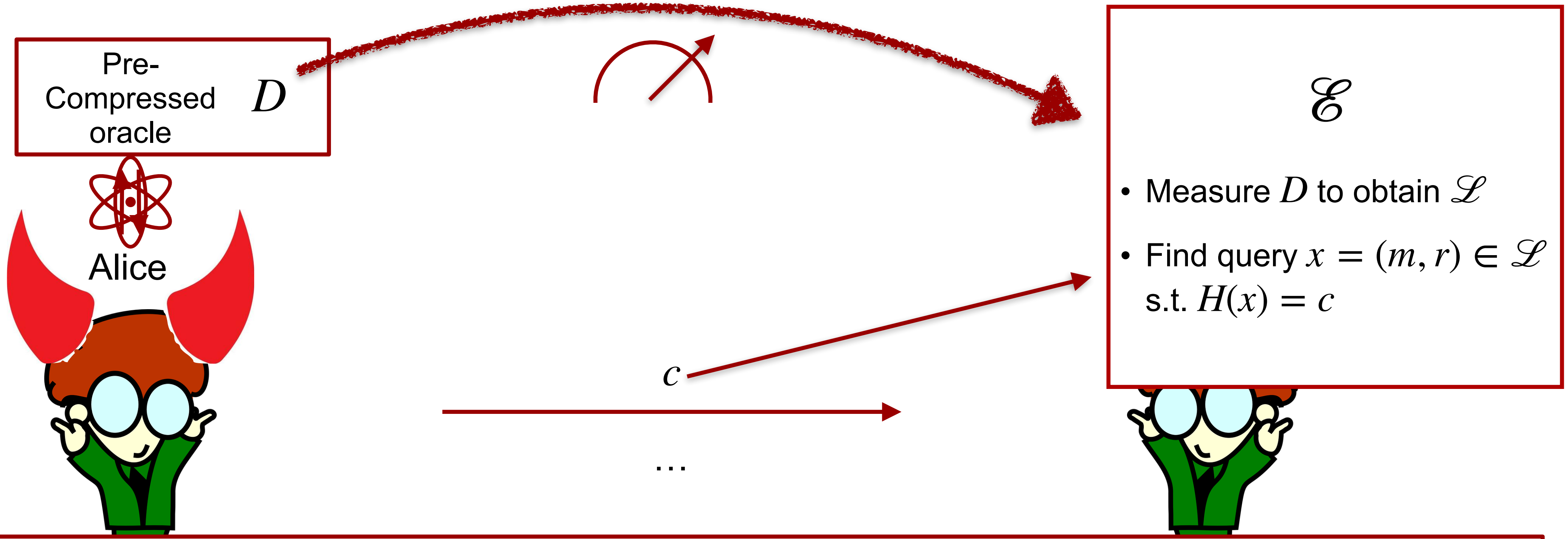
Extractable commitments in the QROM



Why does it work?

- Hiding: Hard to find m given c (preimage resistance) $\Rightarrow$ if m is not in $\mathscr{L}$, Alice can't open????
- Binding: Hard to find pairs (m, r), (m', r') with $m \neq m'$ and $H(m, r) = H(m', r')$ (collision resistance) $\Rightarrow$ extracted m is unique?????

Extractable commitments in the QROM



Why does it work?

- Hiding: Hard to find m given c (preimage resistance) $\Rightarrow$ if m is not in $\mathscr{L}$, Alice can't open????
- Binding: Hard to find pairs (m, r), (m', r') with $m \neq m'$ and $H(m, r) = H(m', r')$ (collision resistance, query bound e.g. via CFHL21) $\Rightarrow$ extracted m is unique.

Extractable commitments in the QROM

Why does it work?

- Hiding: Hard to find m given c (preimage resistance) $\Rightarrow$ if m is not in $\mathscr{L}$, Alice can't open????

Extractable commitments in the QROM

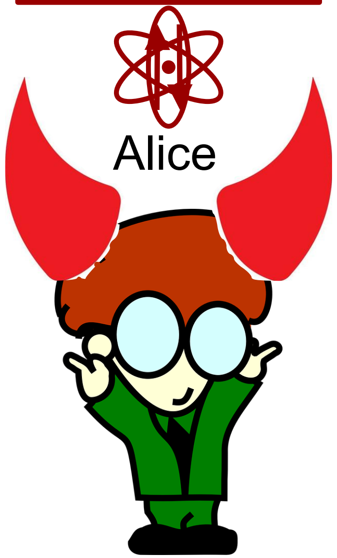
Why does it work?

- Hiding: Hard to find m given c (preimage resistance) $\Rightarrow$ if m is not in $\mathscr{L}$, Alice can't open????

Counterfactual argument!

Extractable commitments in the QROM

Pre-Compressed oracle



Alice

Bob

c →

$\ldots$

(m, r) →

$r \leftarrow \{0,1\}^{\ell}$

$c = H(m, r)$

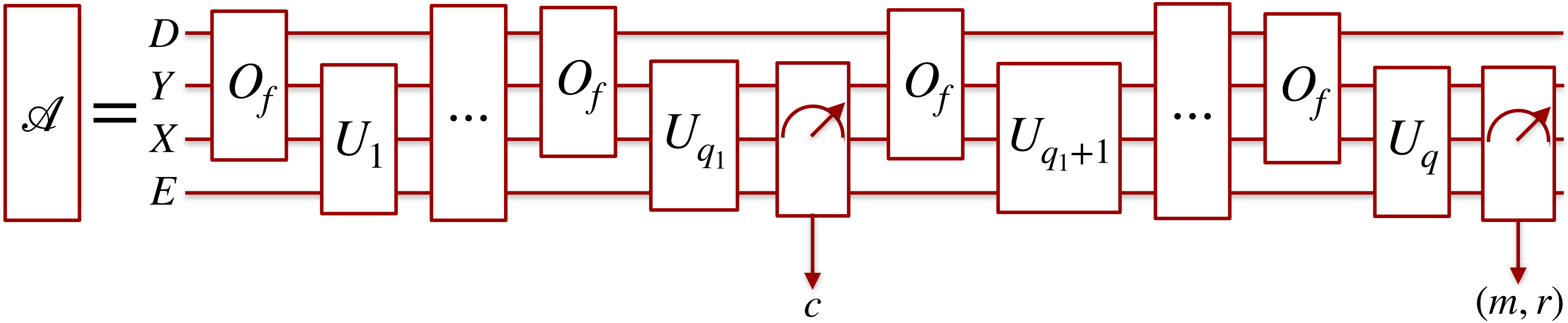
$c = H(m, r)?$

Extractable commitments in the QROM

Why does it work?

- Hiding: Hard to find m given c (preimage resistance) $\Rightarrow$ if m is not in $\mathscr{L}$, Alice can't open????

Counterfactual argument!

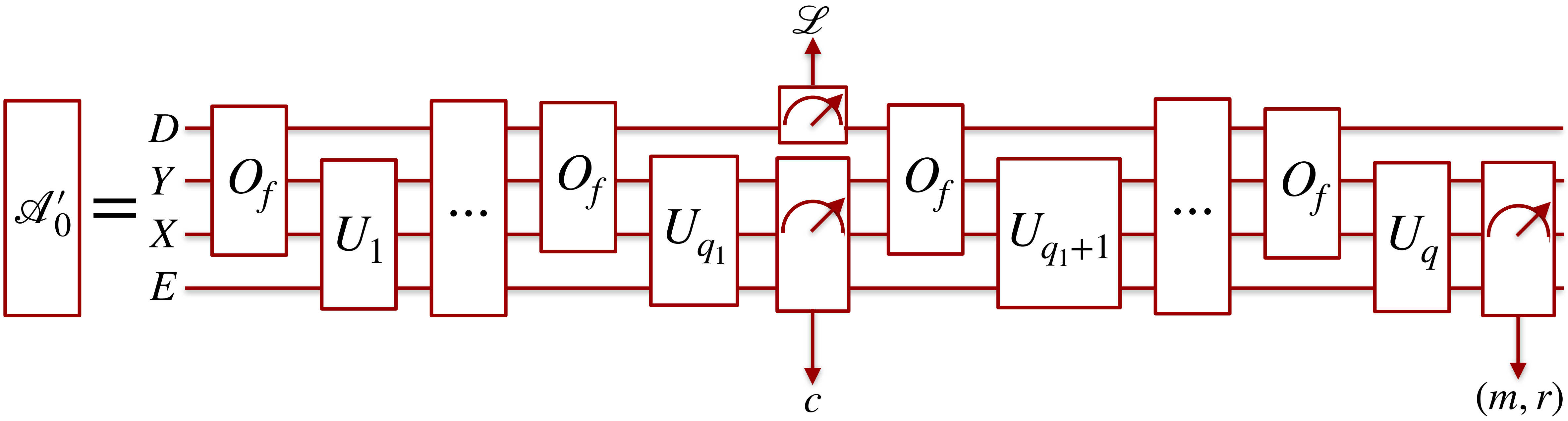


Extractable commitments in the QROM

Why does it work?

- Hiding: Hard to find m given c (preimage resistance) $\Rightarrow$ if m is not in $\mathscr{L}$, Alice can't open????

Counterfactual argument!

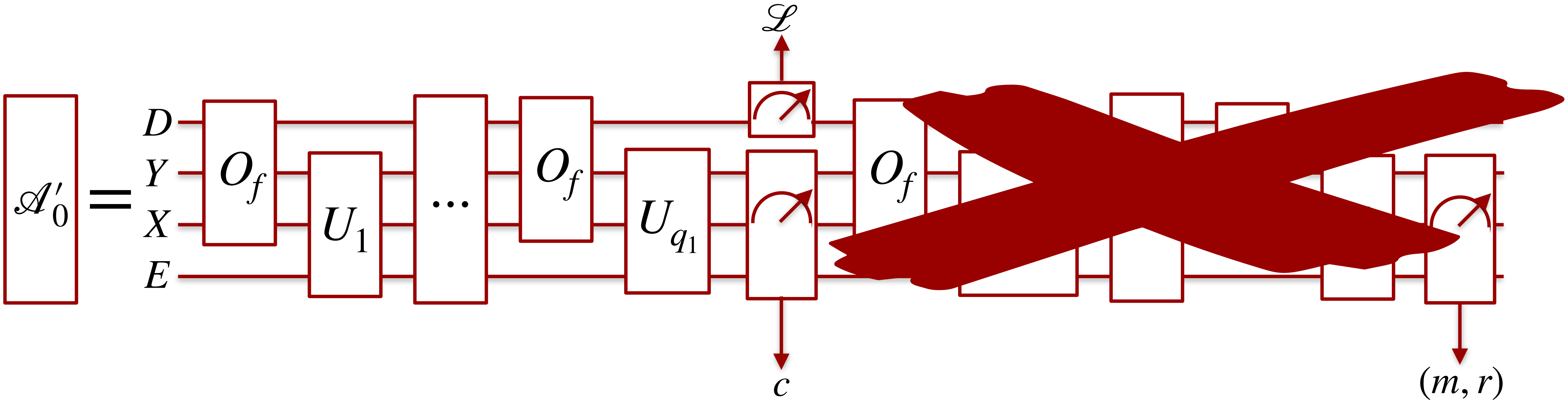


Extractable commitments in the QROM

Why does it work?

- Hiding: Hard to find m given c (preimage resistance) $\Rightarrow$ if m is not in $\mathscr{L}$, Alice can't open????

Counterfactual argument!

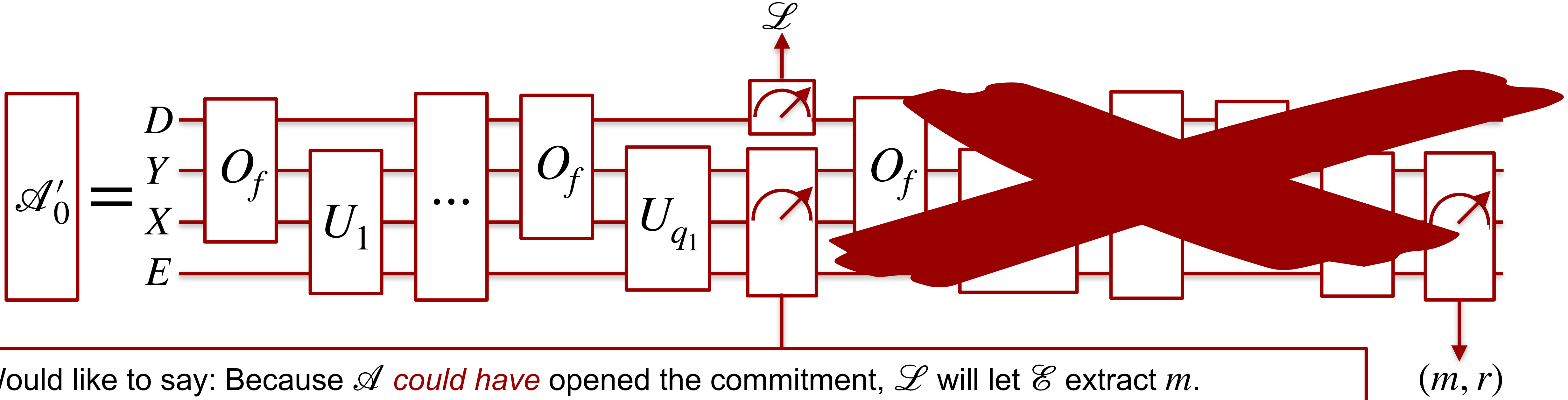


Extractable commitments in the QROM

Why does it work?

- Hiding: Hard to find m given c (preimage resistance) $\Rightarrow$ if m is not in $\mathscr{L}$, Alice can't open????

Counterfactual argument!



Would like to say: Because $\mathscr{A}$ *could have* opened the commitment, $\mathscr{L}$ will let $\mathscr{E}$ extract m.

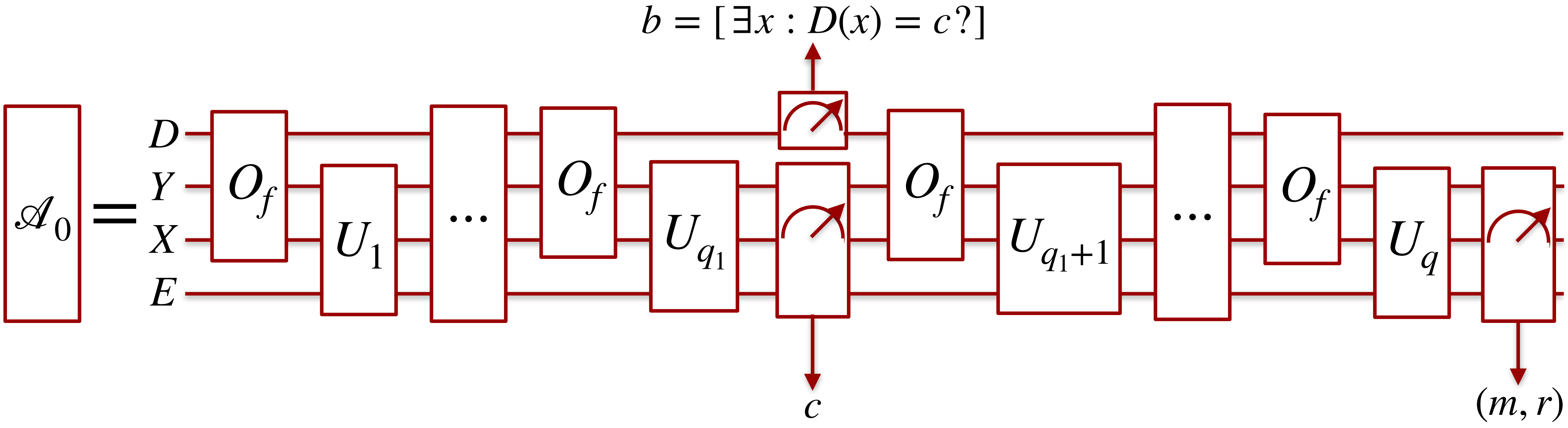
Missing ingredient: Pinching Lemma

Lemma (Pinching, in this form: Boneh and Zhandry '13):

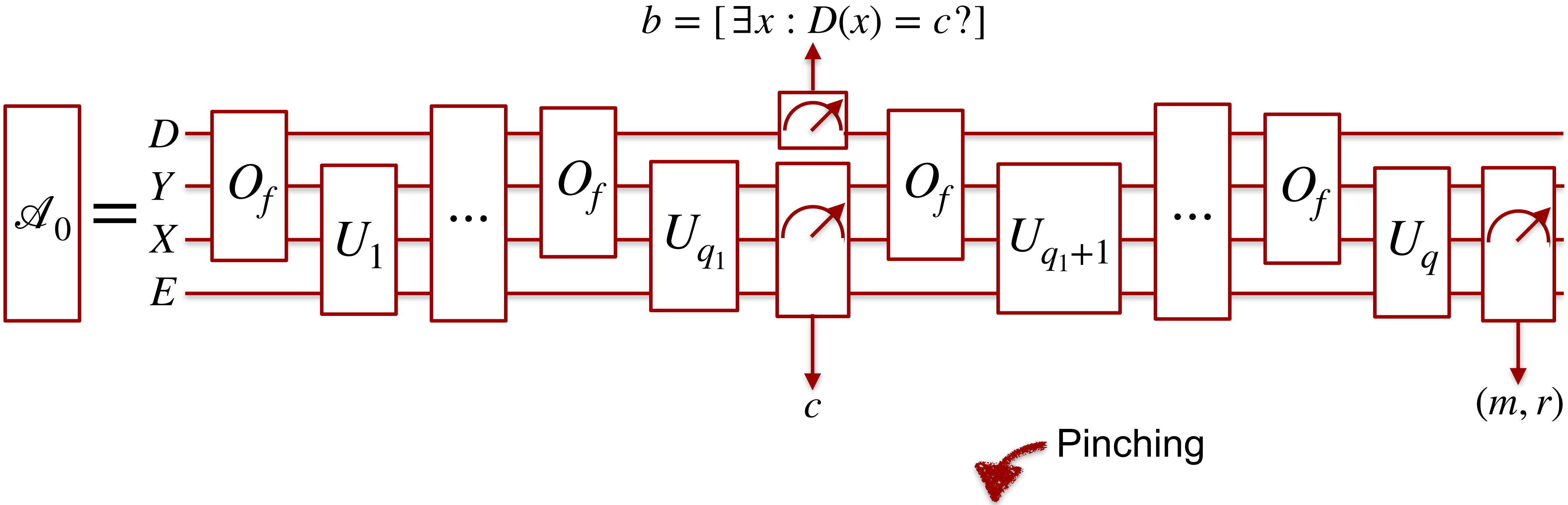
Let $\mathscr{A}$ be a quantum algorithm and $x' \in \{0,1\}^n$. Let $\mathscr{A}_0$ be another quantum algorithm obtained from $\mathscr{A}$ by pausing $\mathscr{A}$ at an arbitrary stage of execution, performing a partial measurement that obtains one of k outcomes, and then resuming $\mathscr{A}$. Then

$$\Pr_{x \leftarrow \mathscr{A}_0()}[x = x'] \geq \Pr_{x \leftarrow \mathscr{A}()}[x = x']/k$$

Extractable commitments in the QROM

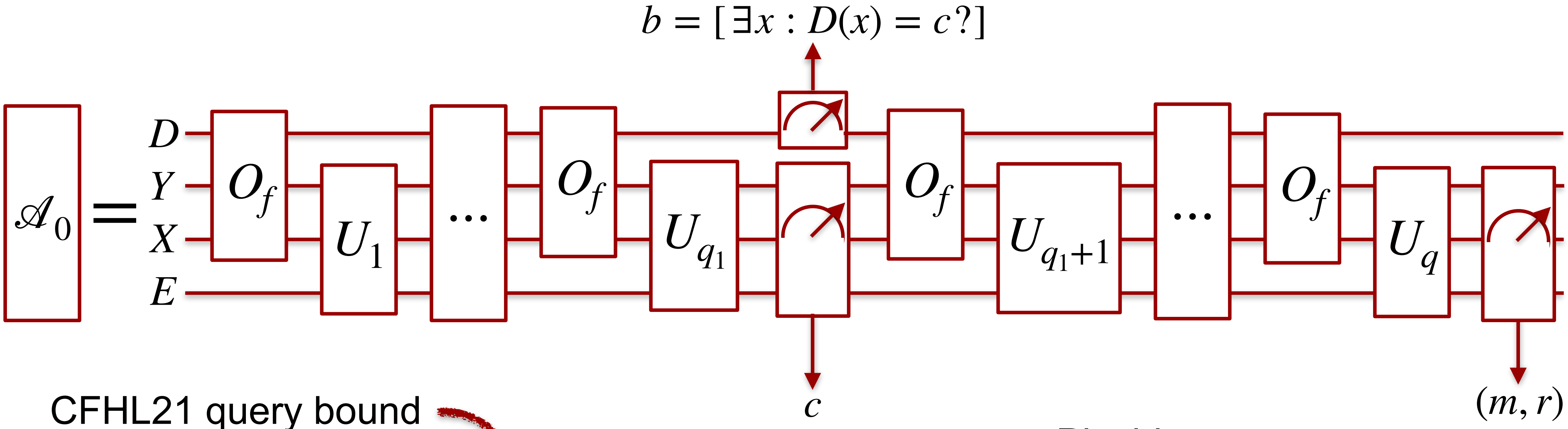


Extractable commitments in the QROM



$$\Pr_{(c,m,r)\leftarrow\mathscr{A}_0()}[O_f(m,r)=c] \geq \Pr_{(c,m,r)\leftarrow\mathscr{A}()}[O_f(m,r)=c]/2$$

Extractable commitments in the QROM



CFHL21 query bound

Pinching

$$\Pr[b=1] + O\left(\frac{(q-q_1)^2}{2^n}\right) \geq \Pr_{(c,m,r)\leftarrow\mathscr{A}_0()}[O_f(m,r)=c] \geq \Pr_{(c,m,r)\leftarrow\mathscr{A}()}[O_f(m,r)=c]/2$$

Extractable commitments in the QROM

Theorem (Extractable Commitments in the QROM, informal):

Let $\mathscr{A}^H$ be an interactive quantum oracle algorithm with access to a random oracle H that first outputs a commitment c, and later opening information (m, r). There exists an extractor $\mathscr{E}$ that simulates $\mathscr{A}$'s oracle H and after $\mathscr{A}$ outputs c, outputs (m', r') such that $H(m', r') = c$.

Extractable commitments in the QROM

Extractable commitments in the QROM

Using slightly more quantum techniques [DF**M**S22] we can…

- …get rid of the factor of 2

Extractable commitments in the QROM

Using slightly more quantum techniques [DF**M**S22] we can…

- …get rid of the factor of 2
- …prove that if we continue to run $\mathscr{A}$ after extraction, it will still output the same opening (up to small additive loss)

Extractable commitments in the QROM

Using slightly more quantum techniques [DF**M**S22] we can…

- …get rid of the factor of 2
- …prove that if we continue to run $\mathscr{A}$ after extraction, it will still output the same opening (up to small additive loss)

Uses reasoning via operator norm of commutators…

Extractable commitments in the QROM

Using slightly more quantum techniques [DF**M**S22] we can…

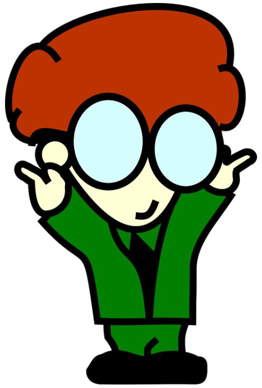
- …get rid of the factor of 2
- …prove that if we continue to run $\mathscr{A}$ after extraction, it will still output the same opening (up to small additive loss)

Uses reasoning via operator norm of commutators…

Applications: straightline extraction for certain sigma protocols, Fujisaki Okamoto

Applications

Sigma protocols

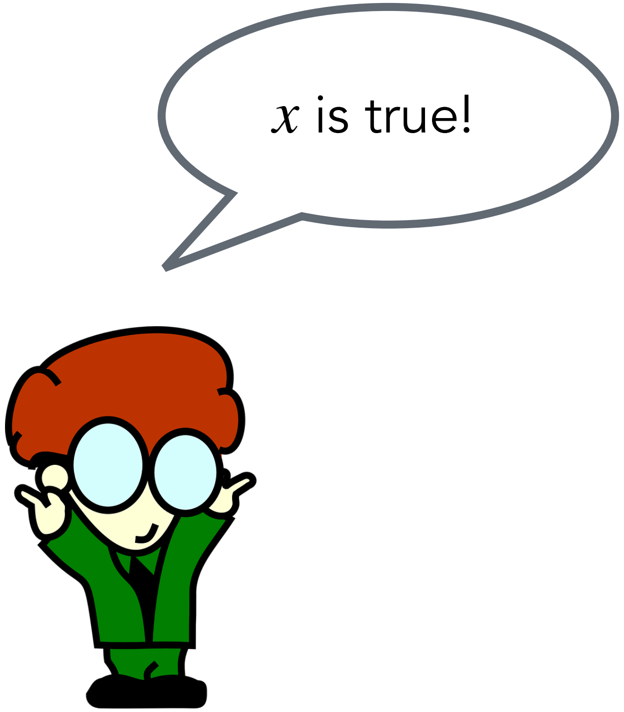


Prover

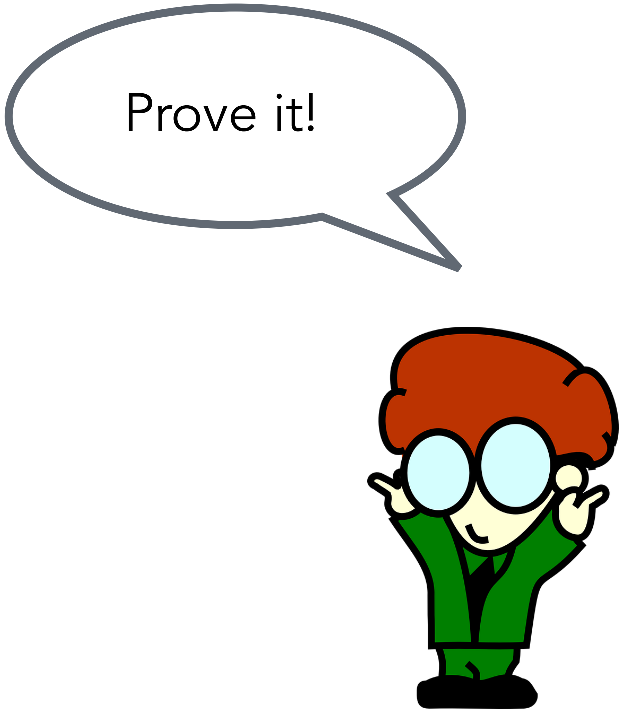
Verifier

Sigma protocols

Sigma protocols



Prover

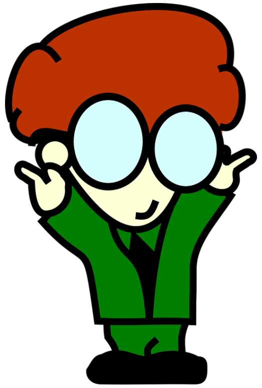


Verifier

Sigma protocols

x is true!

Prove it!

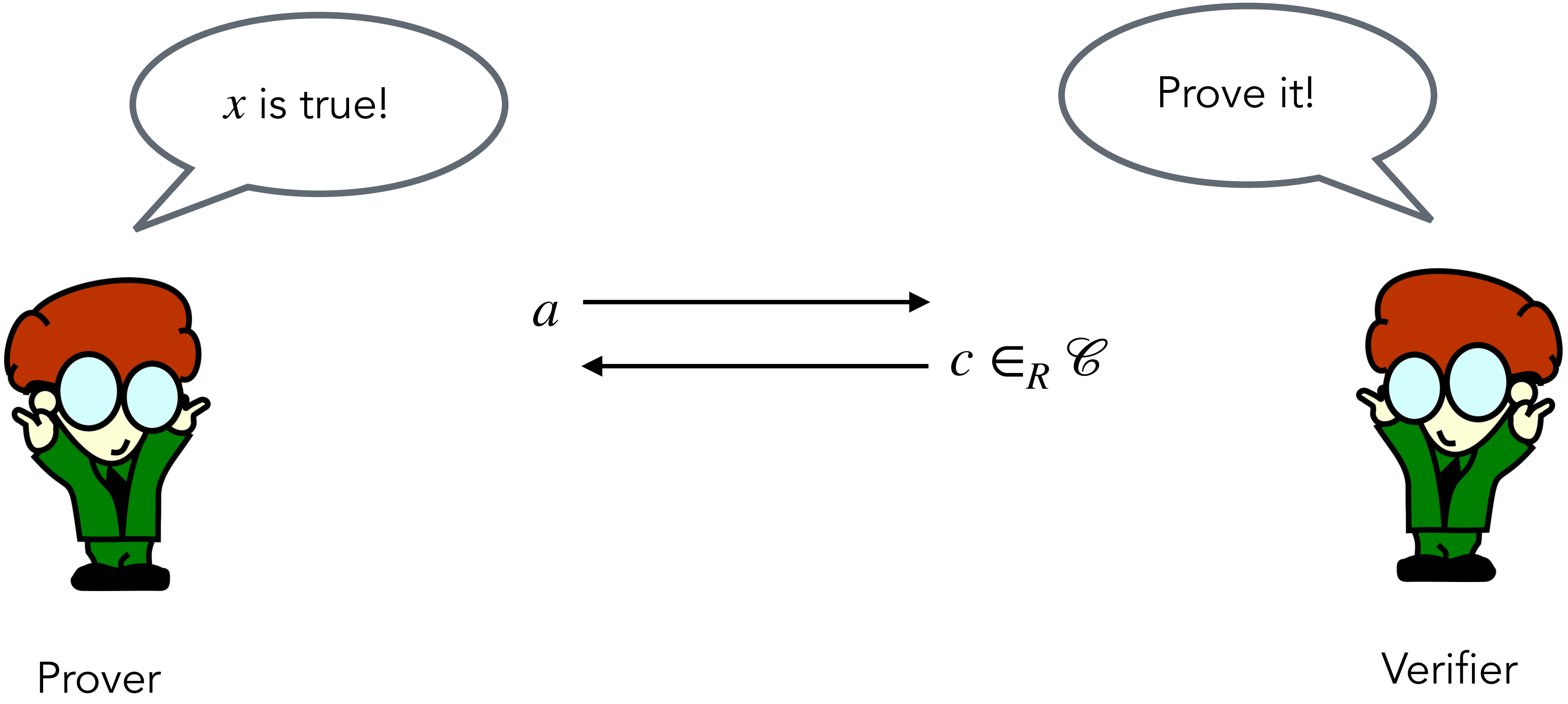


a →

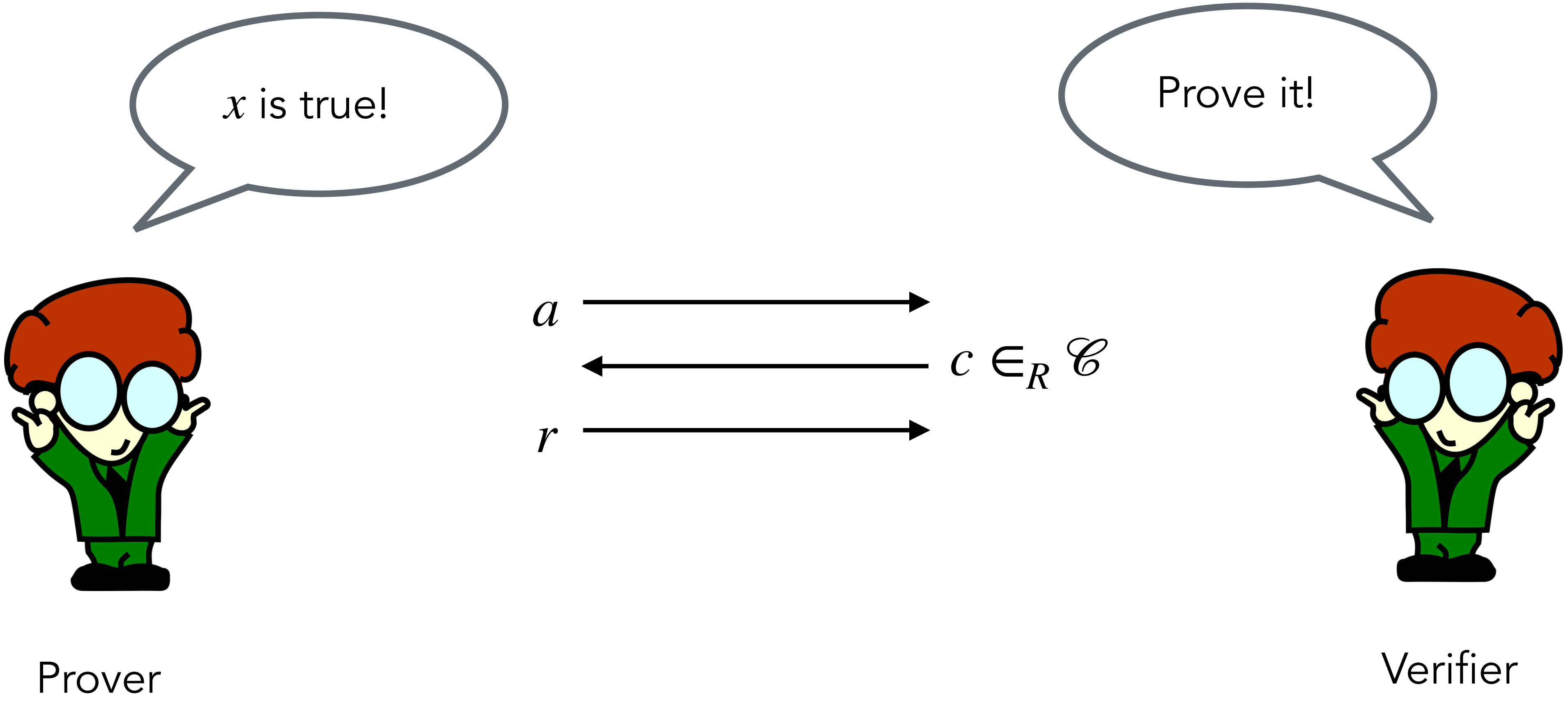
Prover

Verifier

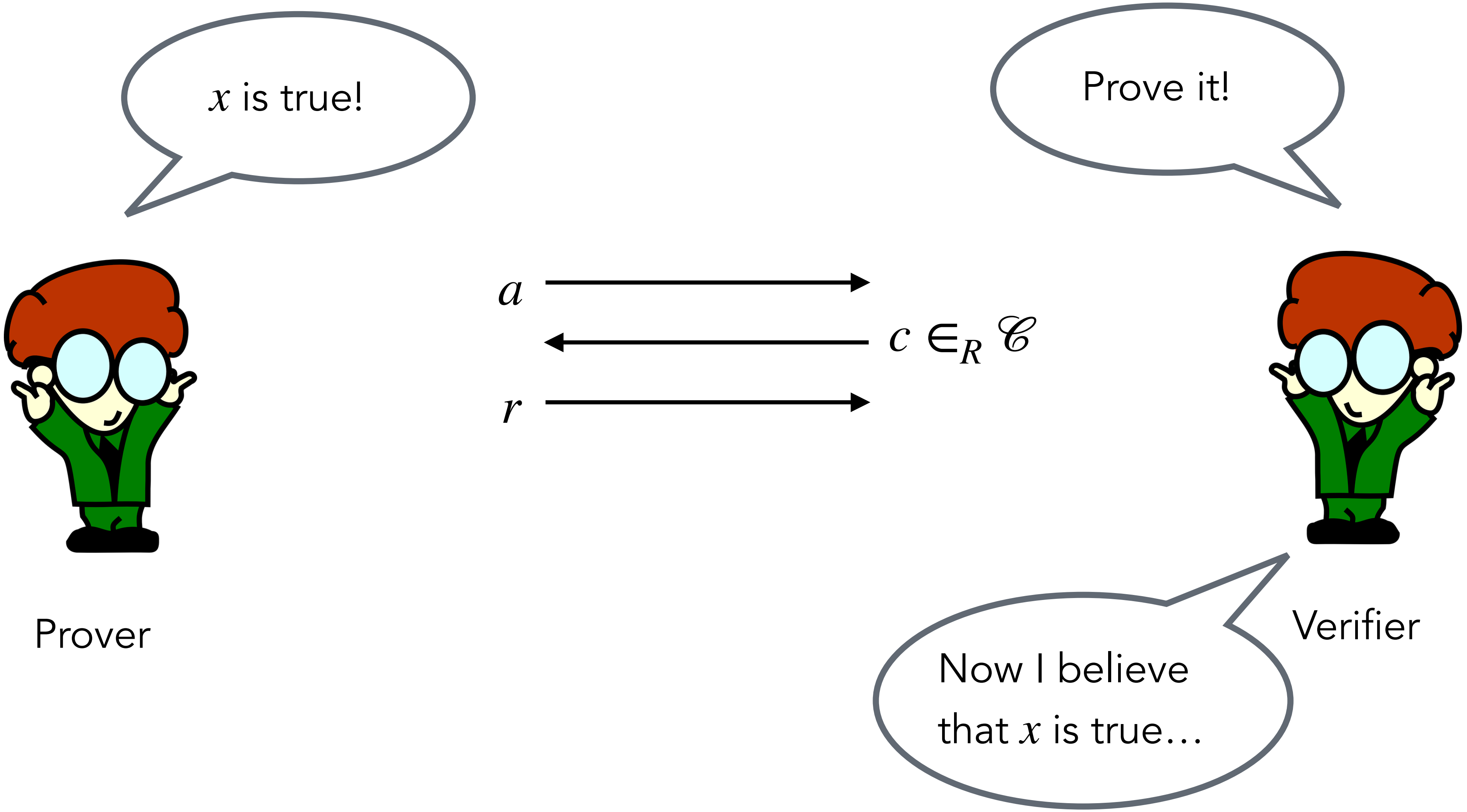
Sigma protocols



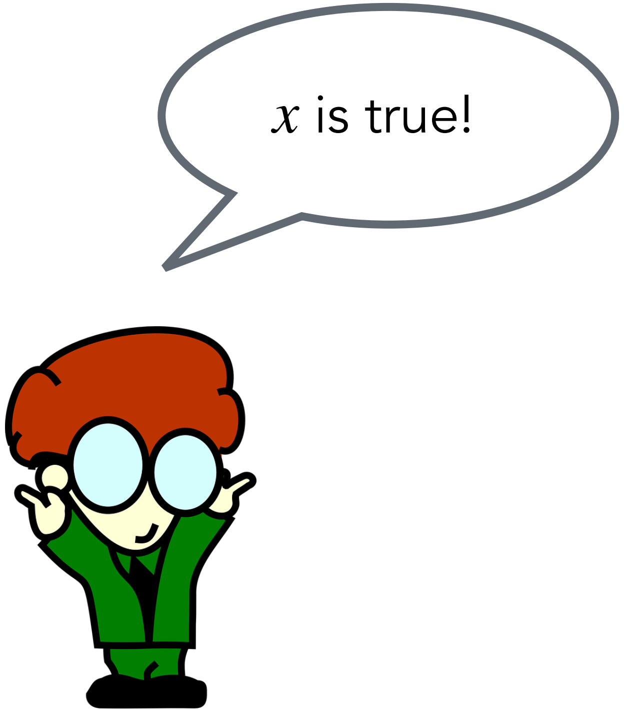
Sigma protocols



Sigma protocols



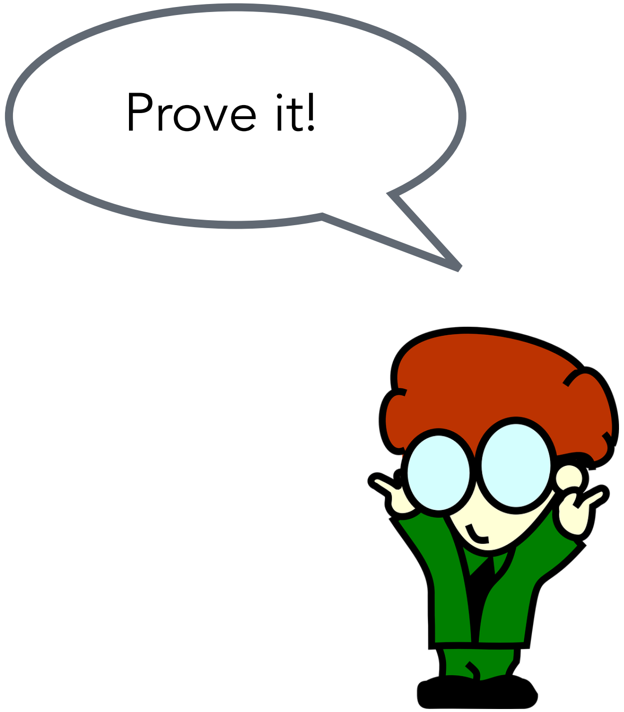
Commit-and-open sigma protocols



Prover

$m_1, m_2, \ldots, m_\ell$

$y_i = H(m_i, r_i)$



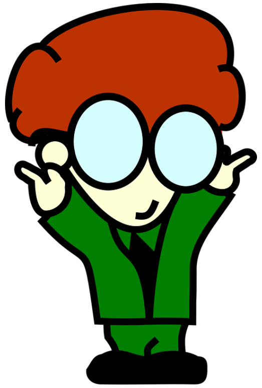
Verifier

Commit-and-open sigma protocols

x is true!

Prove it!

$y_1, y_2, \ldots, y_\ell$ →



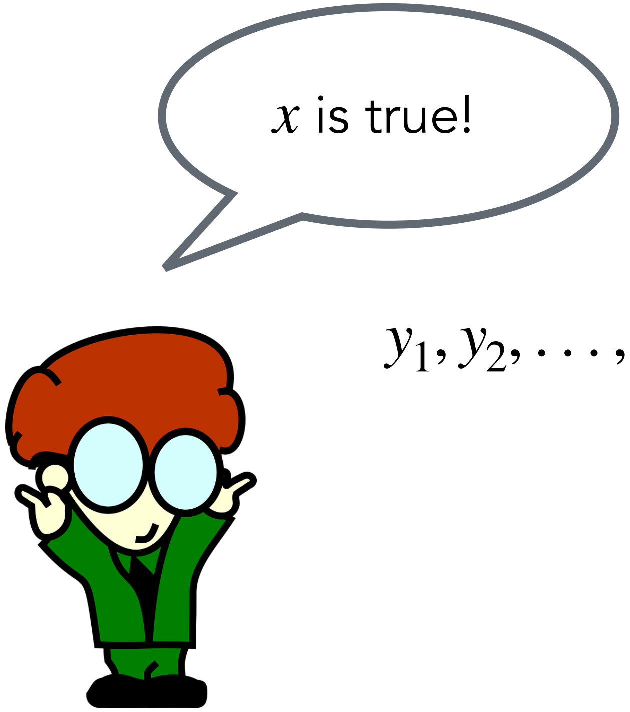
Prover

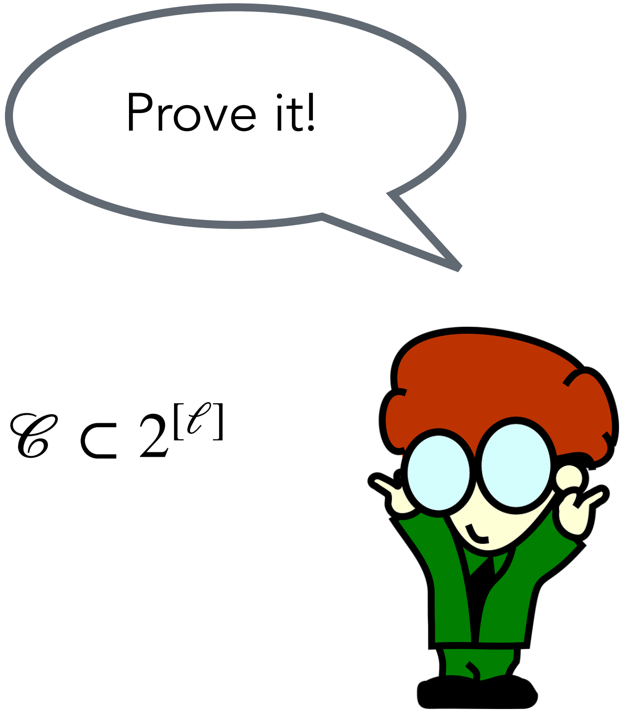
$m_1, m_2, \ldots, m_\ell$

$y_i = H(m_i, r_i)$

Verifier

Commit-and-open sigma protocols





$y_1, y_2, \ldots, y_\ell \longrightarrow$

$\longleftarrow c \in_R \mathscr{C} \subset 2^{[\ell]}$

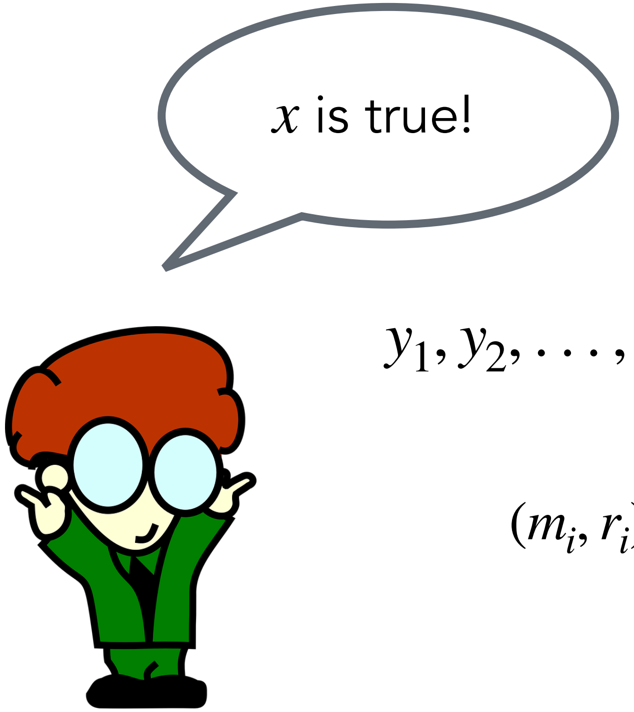
Prover

$m_1, m_2, \ldots, m_\ell$

$y_i = H(m_i, r_i)$

Verifier

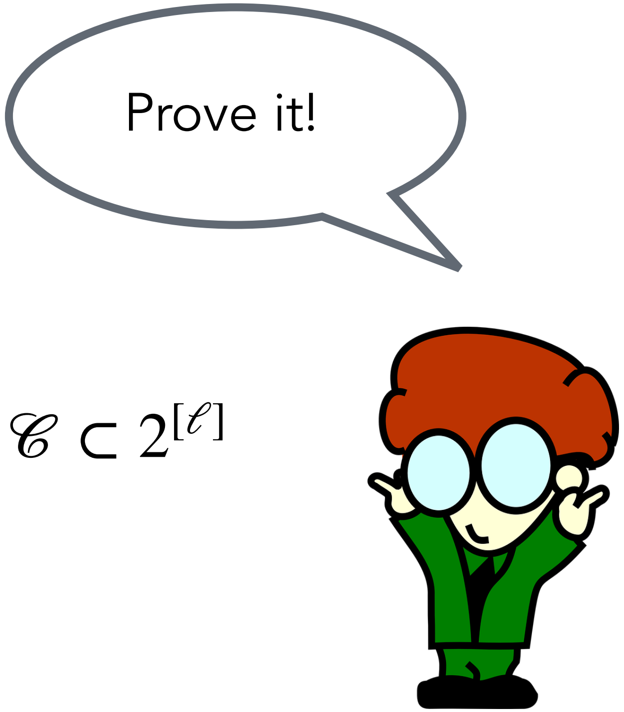
Commit-and-open sigma protocols



$y_1, y_2, \ldots, y_\ell$ →

← $c \in_R \mathscr{C} \subset 2^{[\ell]}$

$(m_i, r_i)_{i \in c}$ →



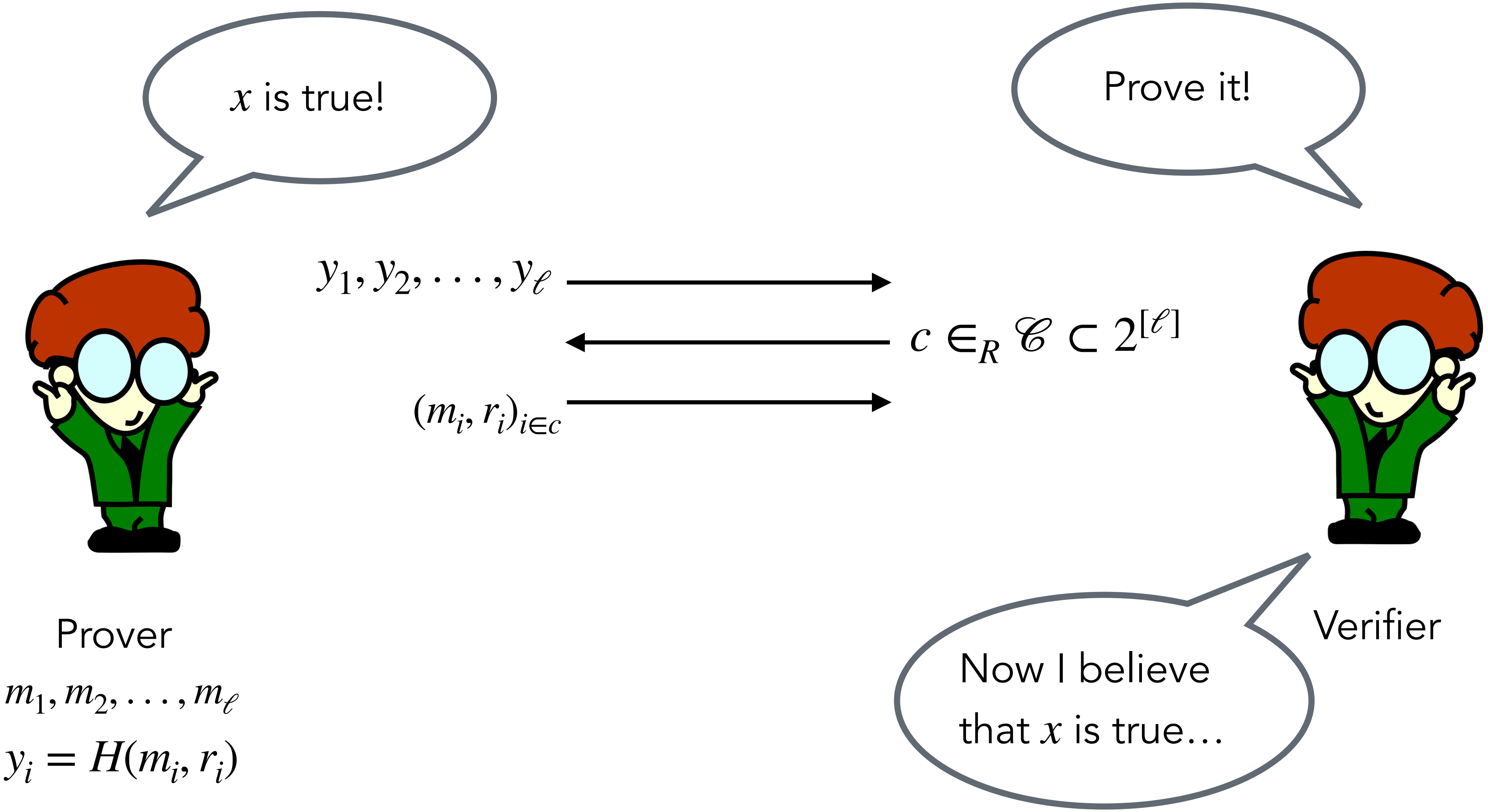
Prover

$m_1, m_2, \ldots, m_\ell$

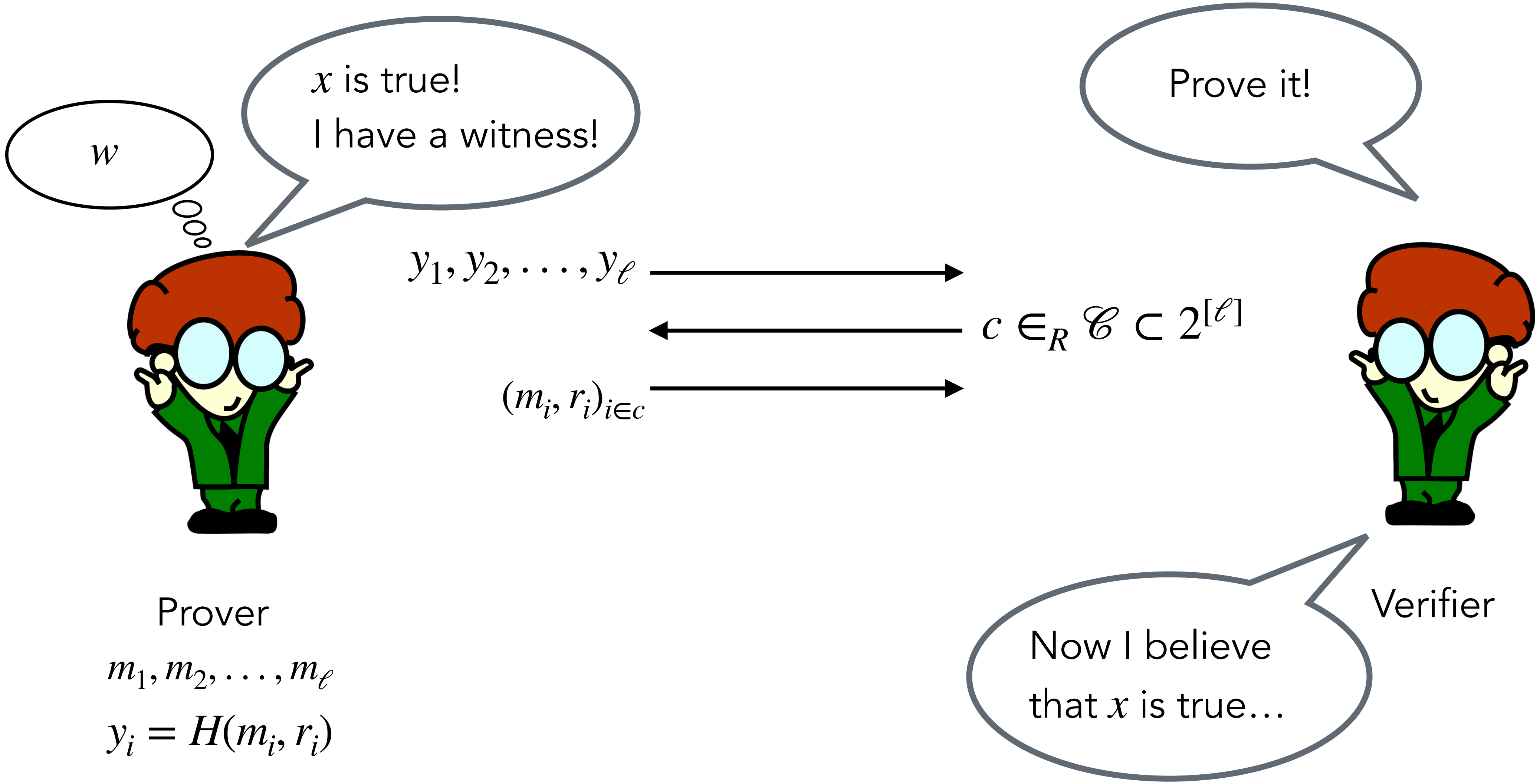
$y_i = H(m_i, r_i)$

Verifier

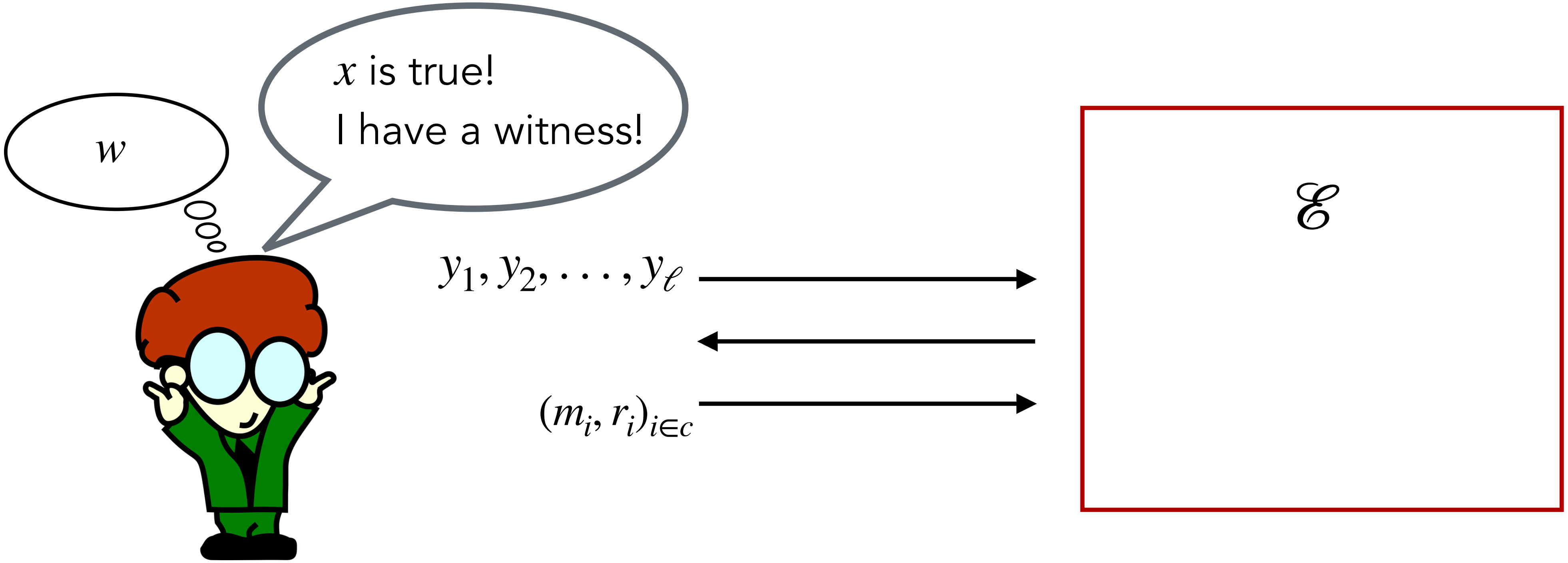
Commit-and-open sigma protocols



Proof of knowledge



Proof of knowledge

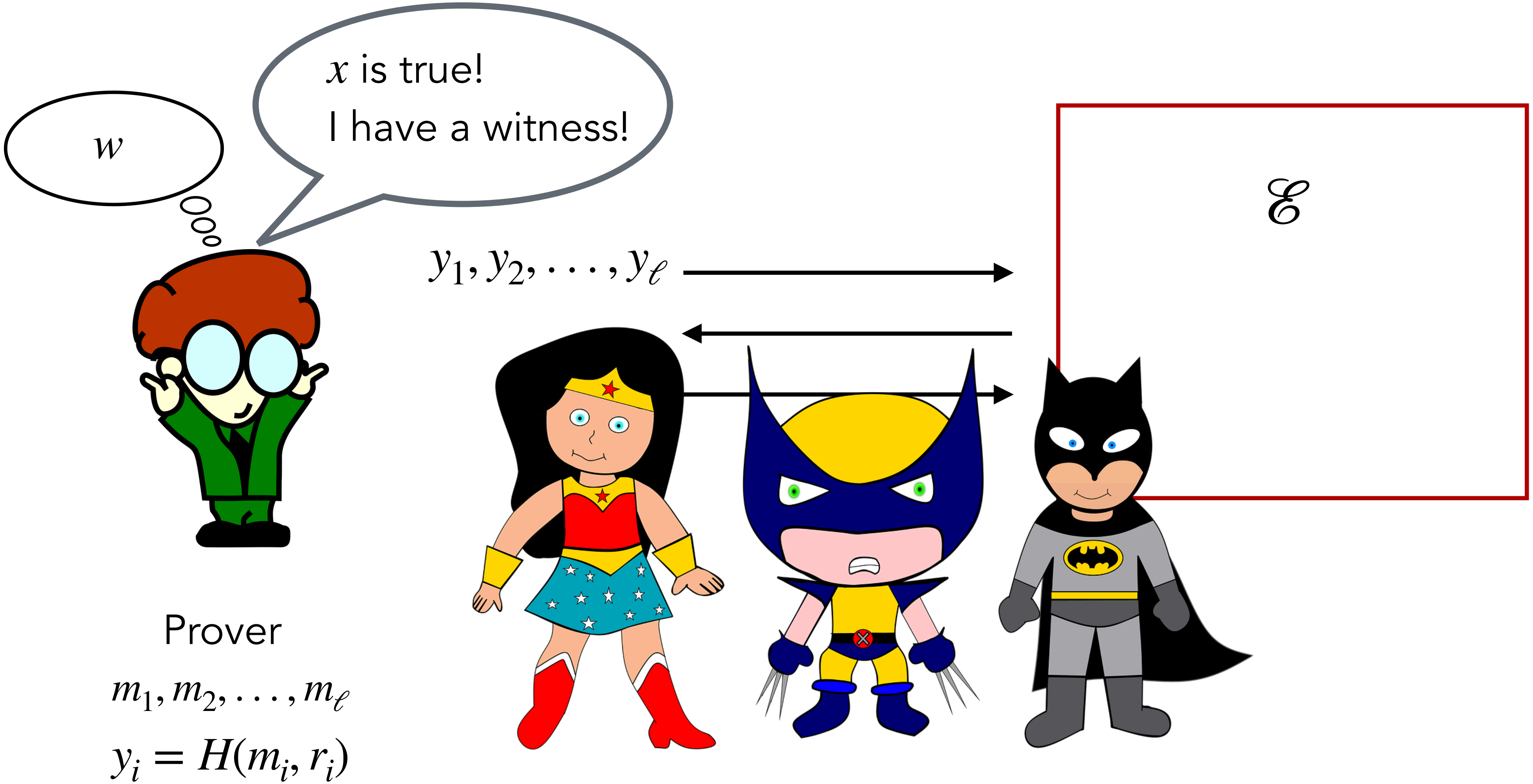


Prover

$m_1, m_2, \ldots, m_\ell$

$y_i = H(m_i, r_i)$

Proof of knowledge



Online Extraction

Theorem (Online extraction for special-sound commit-and-open sigma protocols, DF**M**S22):
For a special-sound commit-and-open Σ-protocol in the QROM, there exists an extractor $\mathscr{E}$ that simulates the quantum-accessible random oracle for any adversary $\mathscr{A}$ such that

$$\Pr[\mathscr{E} \text{ succeeds}] \geq \Pr[\mathscr{A} \text{ succeeds}] - \frac{1}{\ell} - \text{negl}$$

Online Extraction

Theorem (Online extraction for special-sound commit-and-open sigma protocols, DF**M**S22):
For a special-sound commit-and-open Σ-protocol in the QROM, there exists an extractor $\mathscr{E}$ that simulates the quantum-accessible random oracle for any adversary $\mathscr{A}$ such that

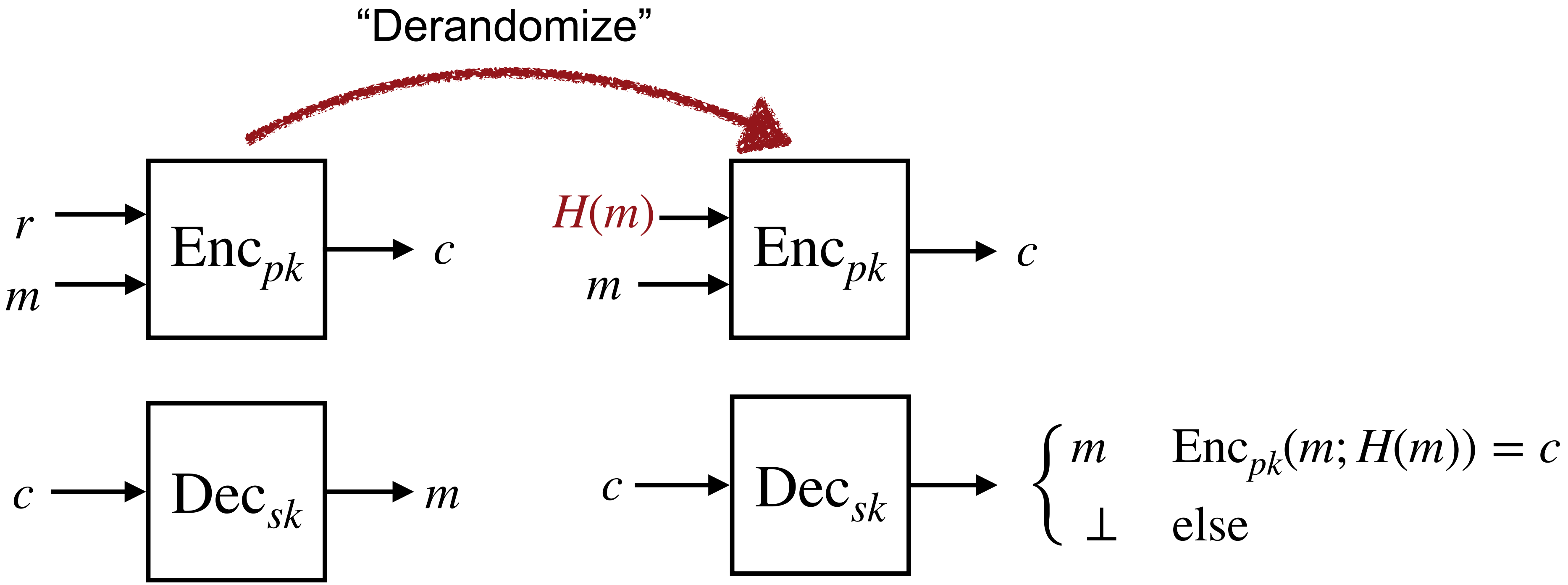
$$\Pr[\mathscr{E} \text{ succeeds}] \geq \Pr[\mathscr{A} \text{ succeeds}] - \frac{1}{\ell} - \text{negl}$$

Accounts for computational binding of commitments

Fujisaki Okamoto

Upgrades weak security to chosen-ciphertext security for key encapsulation

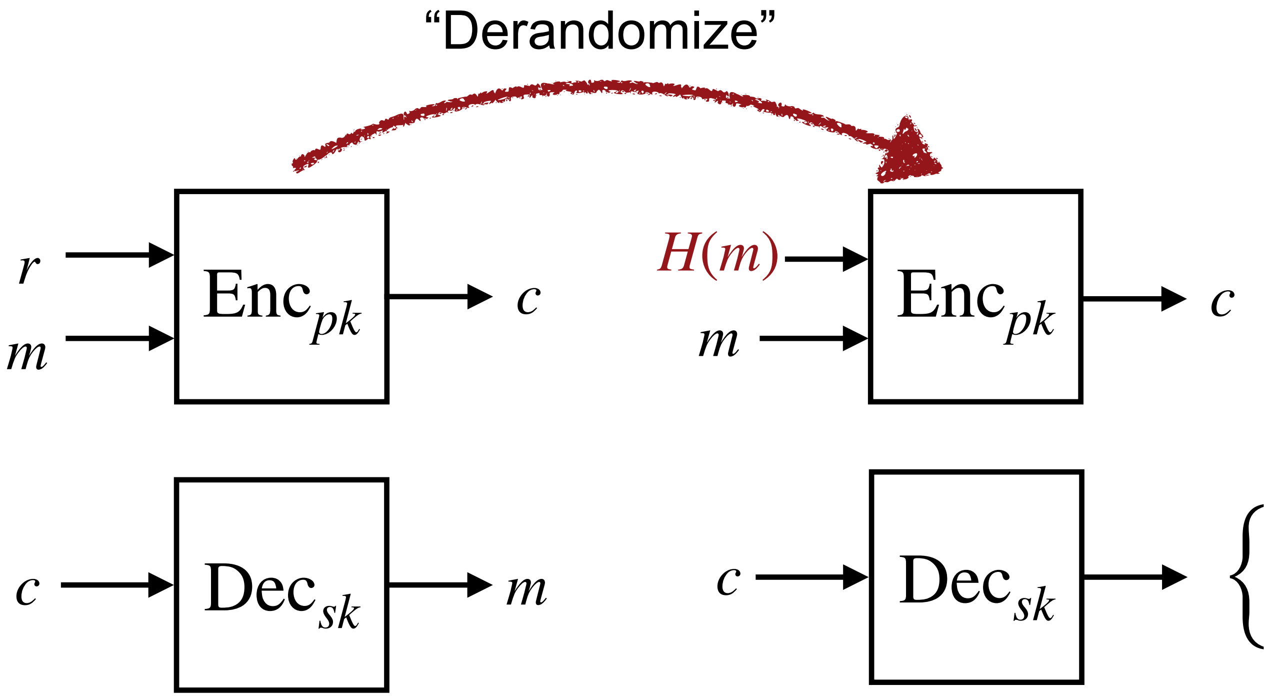
“Derandomize, then Hash”



Fujisaki Okamoto

Upgrades weak security to chosen-ciphertext security for key encapsulation

"Derandomize, then Hash"



$$\begin{cases} m & \text{Enc}_{pk}(m; H(m)) = c \\ \perp & \text{else} \end{cases}$$

Encryption now uses H…
Ciphertext is commitment!

$\Rightarrow$ Extract to simulate decryption

Fujisaki Okamoto

Theorem (Vanilla FO, Zhandry 19', DF**M**S22):
The Fujisaki-Okamoto transformation with explicit rejection applied to a public-key encryption scheme with one-wayness security that is genuinely randomized yields a CCA-secure key encapsulation mechanism. Explicit security bound:

$$\mathsf{ADV}[\mathcal{A}]^{\text{IND-CCA}}_{\mathrm{KEM}} \leq 2q\sqrt{\mathsf{ADV}^{\text{OW-CPA}}_{\mathrm{PKE}}[\mathcal{B}]} + 24q^2\sqrt{\delta} + 24q\sqrt{qq_D}\cdot 2^{-\gamma/4}\,.$$

Summary

- The compressed oracle technique allows random-oracle-based extraction in the post-quantum setting
- Applications include digital signatures and CCA secure key encapsulation

Open questions

- Cryptographers like permutations — the compressed oracle technique doesn't? (First step: eprint 2024/1140)
- Plenty of classical RO-based extractors that have not been made quantum yet: Forking Lemma, Masny-Rindal OT…

The End