Information-Theoretic PIR: Constructions and Applications

Seyoon Ragavan

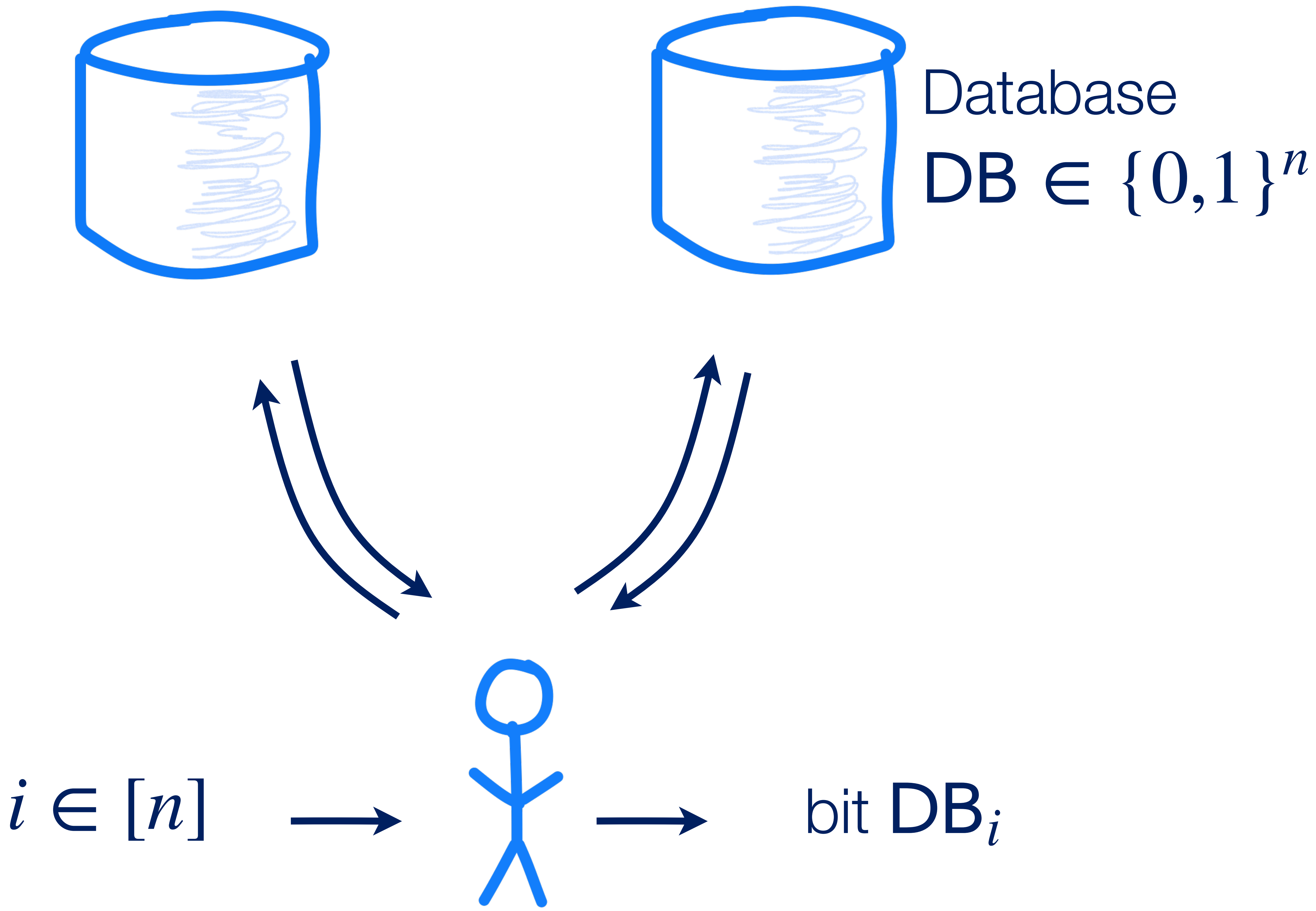
Based on joint works with Alexandra Henzinger and (if time permits) Ted Pyne

Thanks Alexandra for some of these slides!

ePrint: 2025/2008, 2026/265

Private Information Retrieval [CGKS95,KO97]

Goal: privately read an entry from a remote database



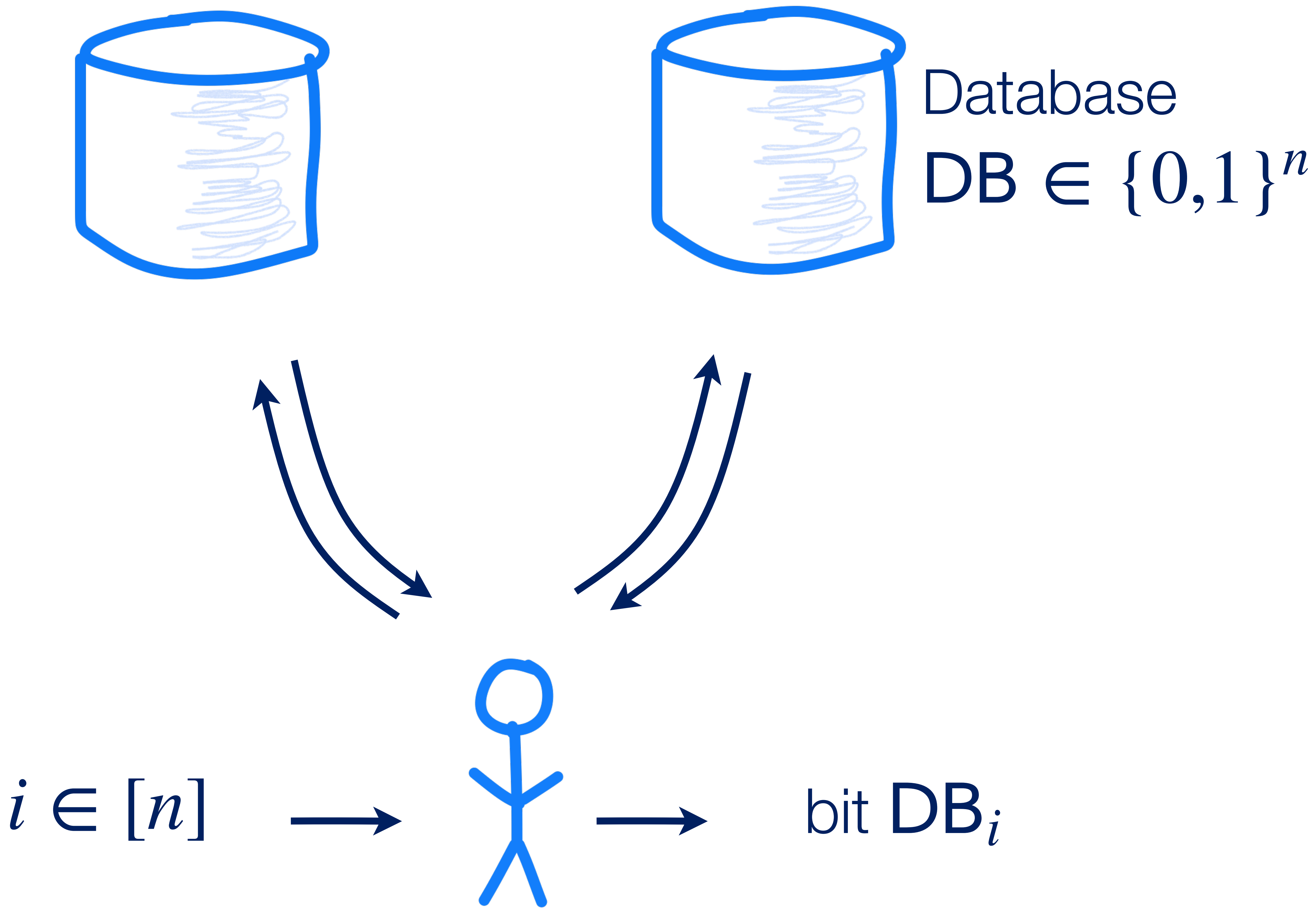
Database

DB $\in \{0,1\}^n$

$i \in [n]$ ⟶ ⟶ bit DB_i

Private Information Retrieval [CGKS95,KO97]

Goal: privately read an entry from a remote database



Database

DB $\in \{0,1\}^n$

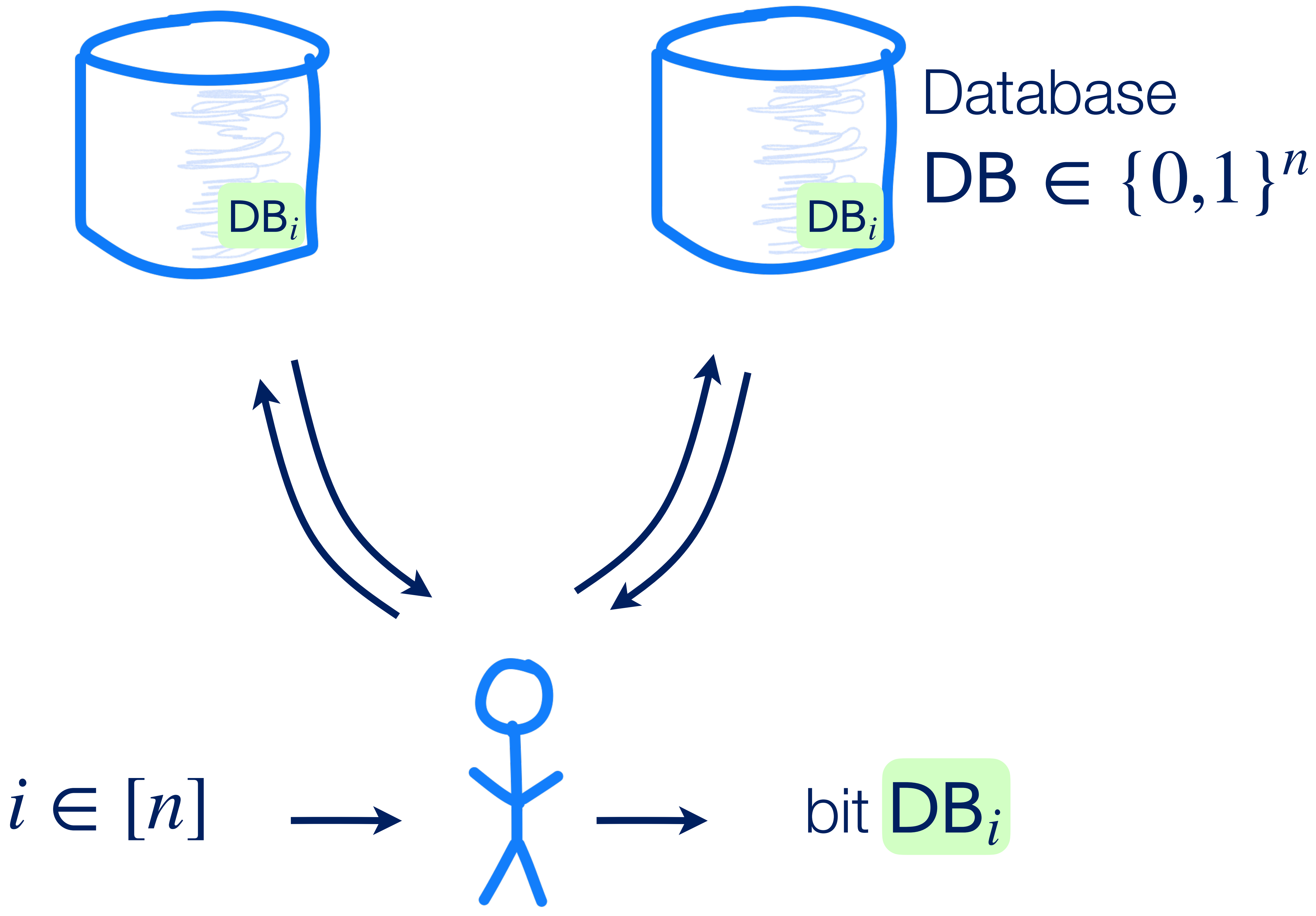
$i \in [n]$ $\longrightarrow$ $\longrightarrow$ bit DB$_i$

Correctness:
for all DB $\in \{0,1\}^n$ and $i \in [n]$, a user interacting with two honest servers learns DB$_i$.

Privacy:
an attacker compromising one server learns nothing about i, even if malicious.

Private Information Retrieval [CGKS95,KO97]

Goal: privately read an entry from a remote database

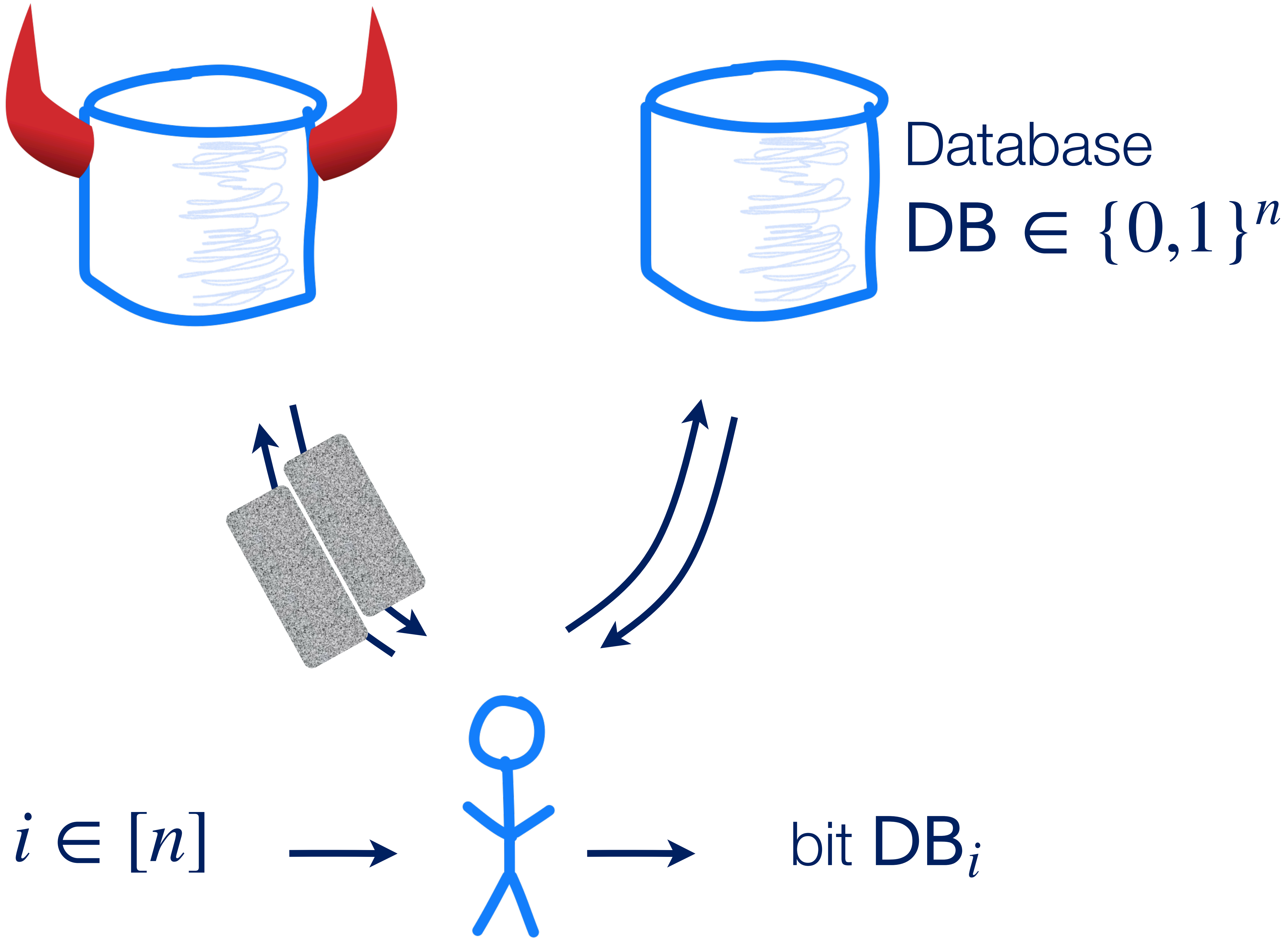


Correctness:
for all $\mathsf{DB} \in \{0,1\}^n$ and $i \in [n]$, a user interacting with two honest servers learns DB_i.

Privacy:
an attacker compromising one server learns nothing about i, even if malicious.

Private Information Retrieval [CGKS95,KO97]

Goal: privately read an entry from a remote database



Database

DB $\in \{0,1\}^n$

$i \in [n]$ ⟶ ⟶ bit DB$_i$

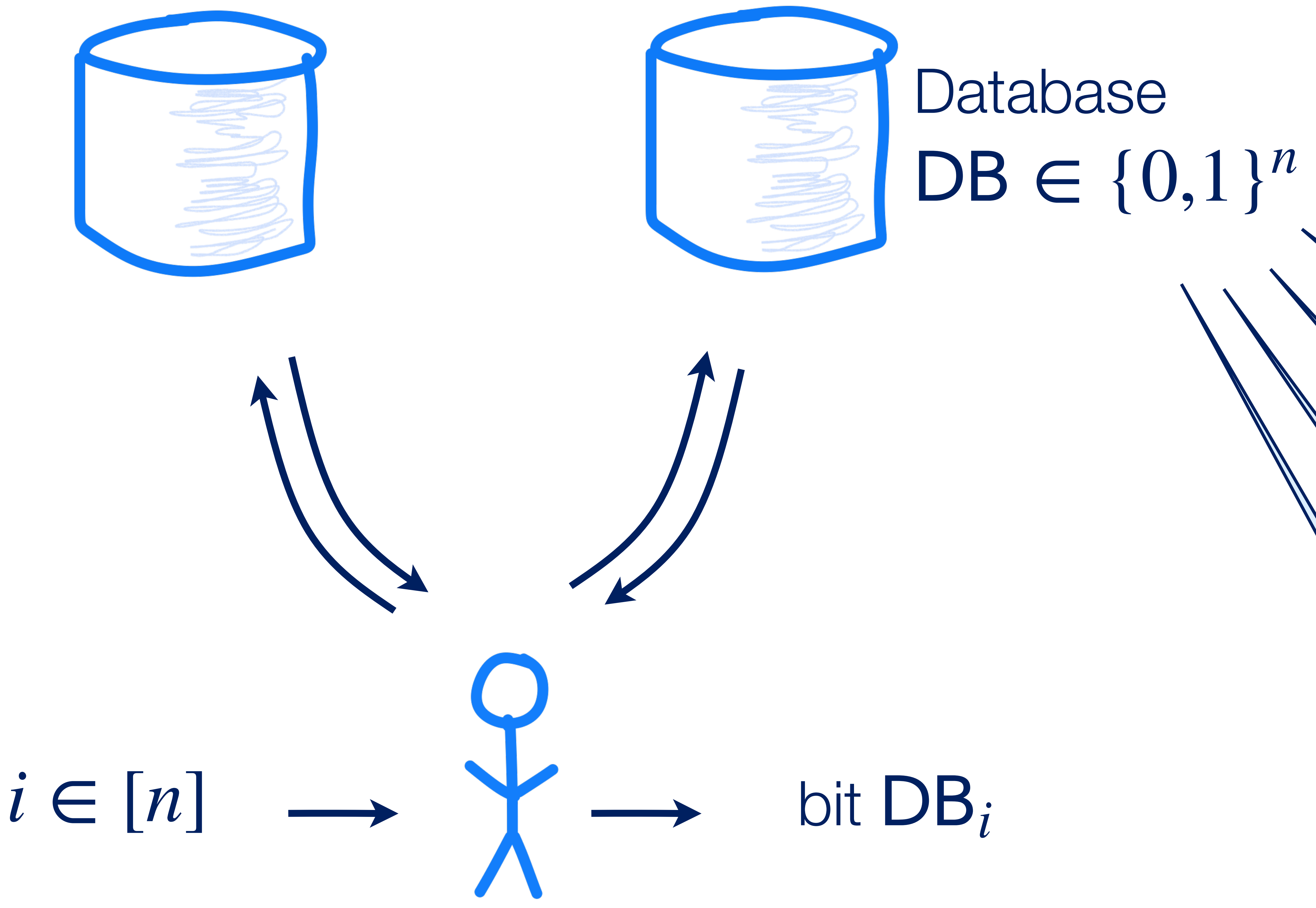
Correctness:
for all DB $\in \{0,1\}^n$ and $i \in [n]$,
a user interacting with two honest
servers learns DB$_i$.

Privacy:
an attacker compromising one
server learns nothing about i,
even if malicious.

Formally: information-theoretic/computational indistinguishability of the server's view, on reads to any two indices i or i'.

Private Information Retrieval [CGKS95,KO97]

Goal: privately read an entry from a remote database



Database

$\mathsf{DB} \in \{0,1\}^n$

$i \in [n]$ → → bit DB_i

Private web search
[Coeus'21, Tiptoe'23, Wally'24]

Private advertising
[J'01,ObliviAd'12,AdVeil'20…]

Private media delivery
[Popcorn'16,PIRSONA'21,CrowdSurf'25,…]

Metadata-hiding messaging
[PIRTor'11,Pung'16,Talek'20,Addra'21,…]

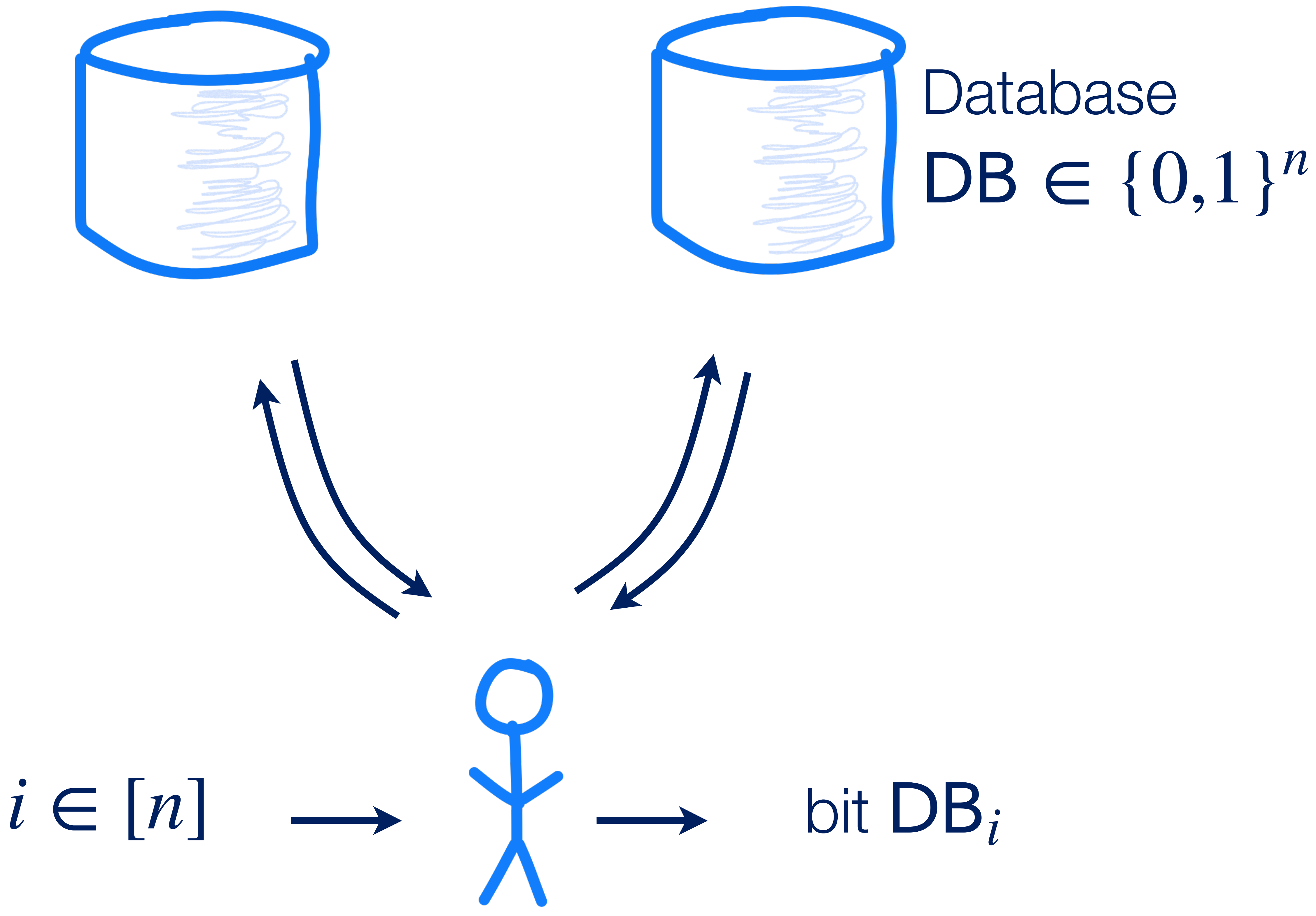
Private auditing and blocklists
[Checklist'20,SimplePIR'23,HPW'25,…]

Private file storage
[Dory'20,Peer2PIR'25,…]

On your iPhone: Private caller-ID

Private Information Retrieval [CGKS95,KO97]

Goal: privately read an entry from a remote database



Modern PIR needs very little communication:

- No privacy: $\log n + 1$
- Info-theoretic privacy: $n^{o(1)}$
- Comp. privacy: $O(\lambda \cdot \log n)$
  [KO97,CMS99,DG16,BGI16]

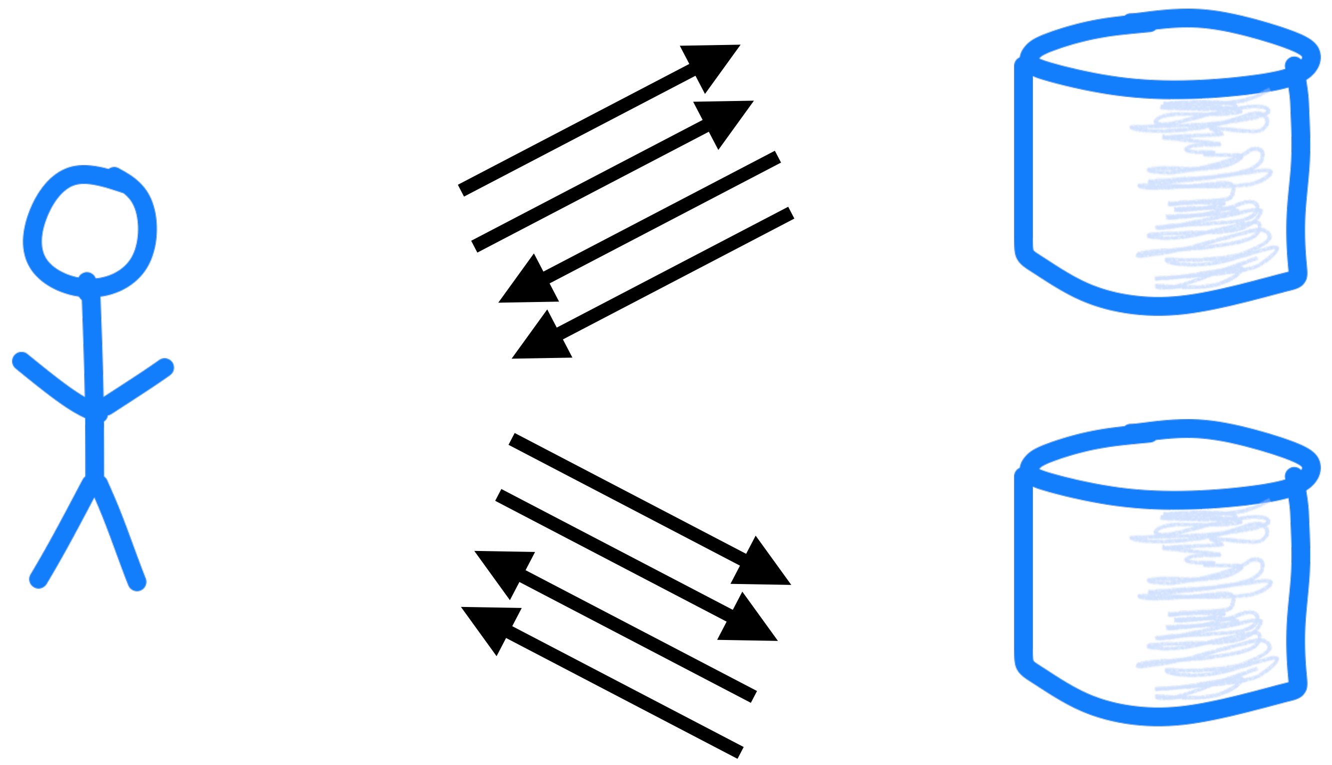
…but lots of server work:

- No privacy: $O(1)$ time
- With privacy: $\Omega(n)$ time
  [BIM00, PY22]

Solution: Change the PIR model to get sublinear time

Solution: Change the PIR model to get sublinear time

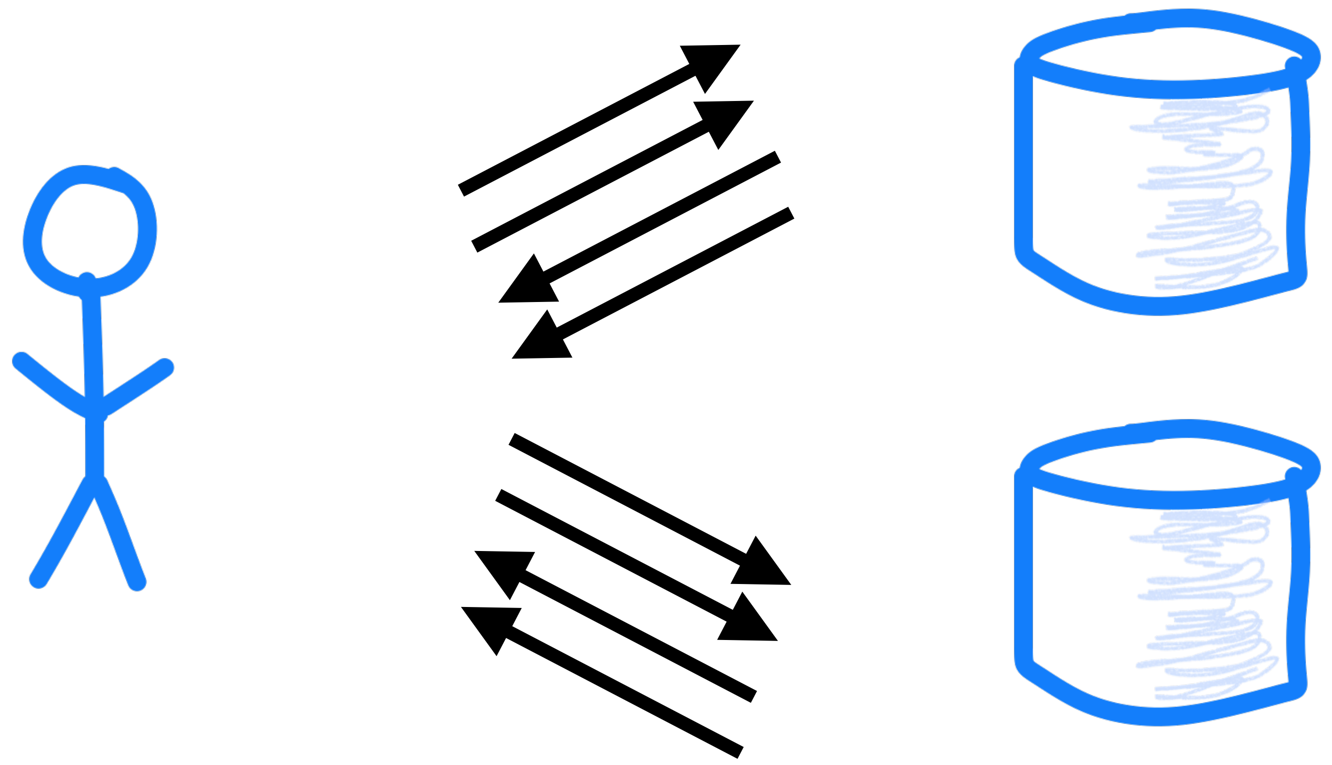
Batch PIR with many, non-adaptive queries



[IKOS’04,HHG’13,GKL’10,LG’15,AS’16,H’16,ACLS’18,CHLR’18]

Solution: Change the PIR model to get sublinear time

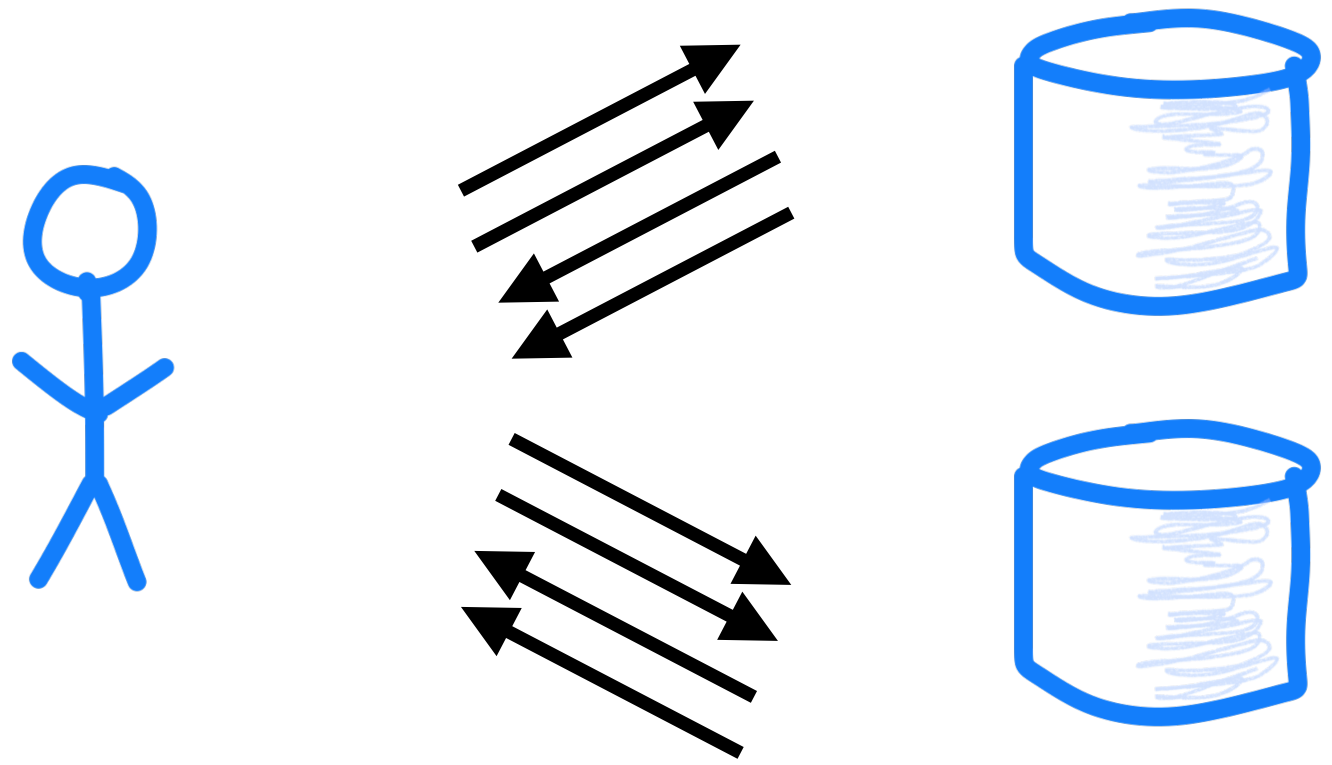
Batch PIR with many, non-adaptive queries



[IKOS’04,HHG’13,GKL’10,LG’15,AS’16,H’16,ACLS’18,CHLR’18]

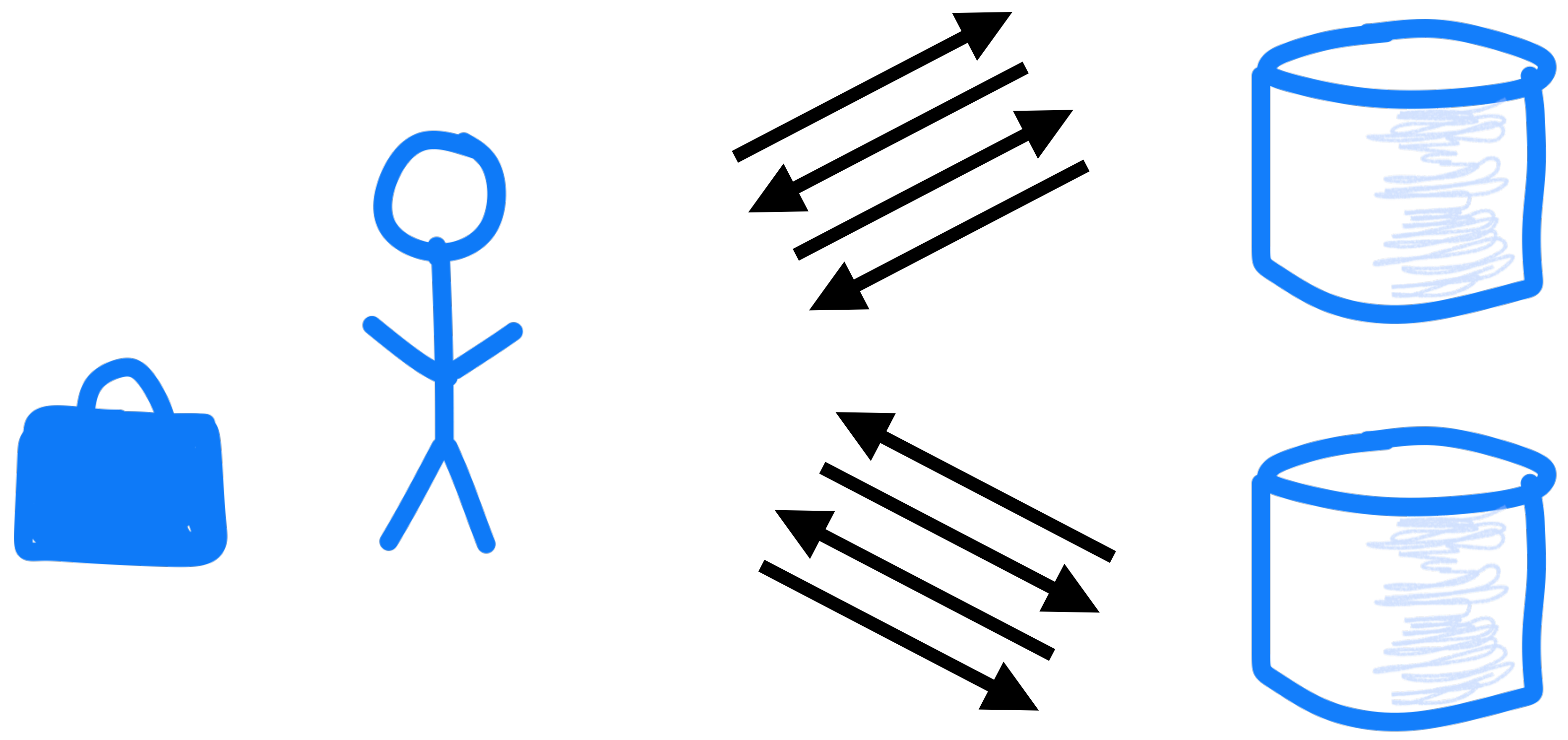
Solution: Change the PIR model to get sublinear time

Batch PIR with many, non-adaptive queries



[IKOS’04,HHG’13,GKL’10,LG’15,AS’16,H’16,ACLS’18,CHLR’18]

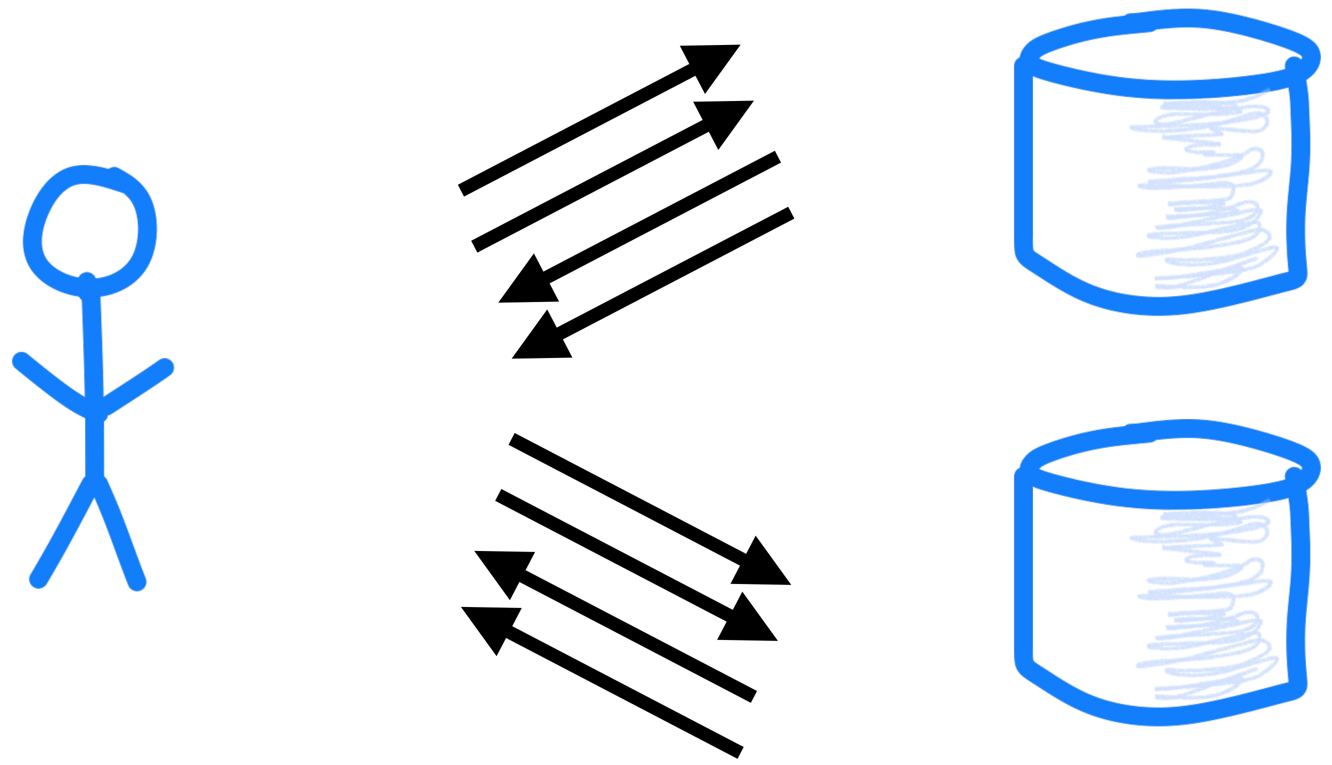
Offline/online PIR with stateful clients + many queries



[CK’20,SACM’21,KC’21,CHK’22,LP’23,ZLS’23,GZS’24,ISW’24,RMS’24,…]

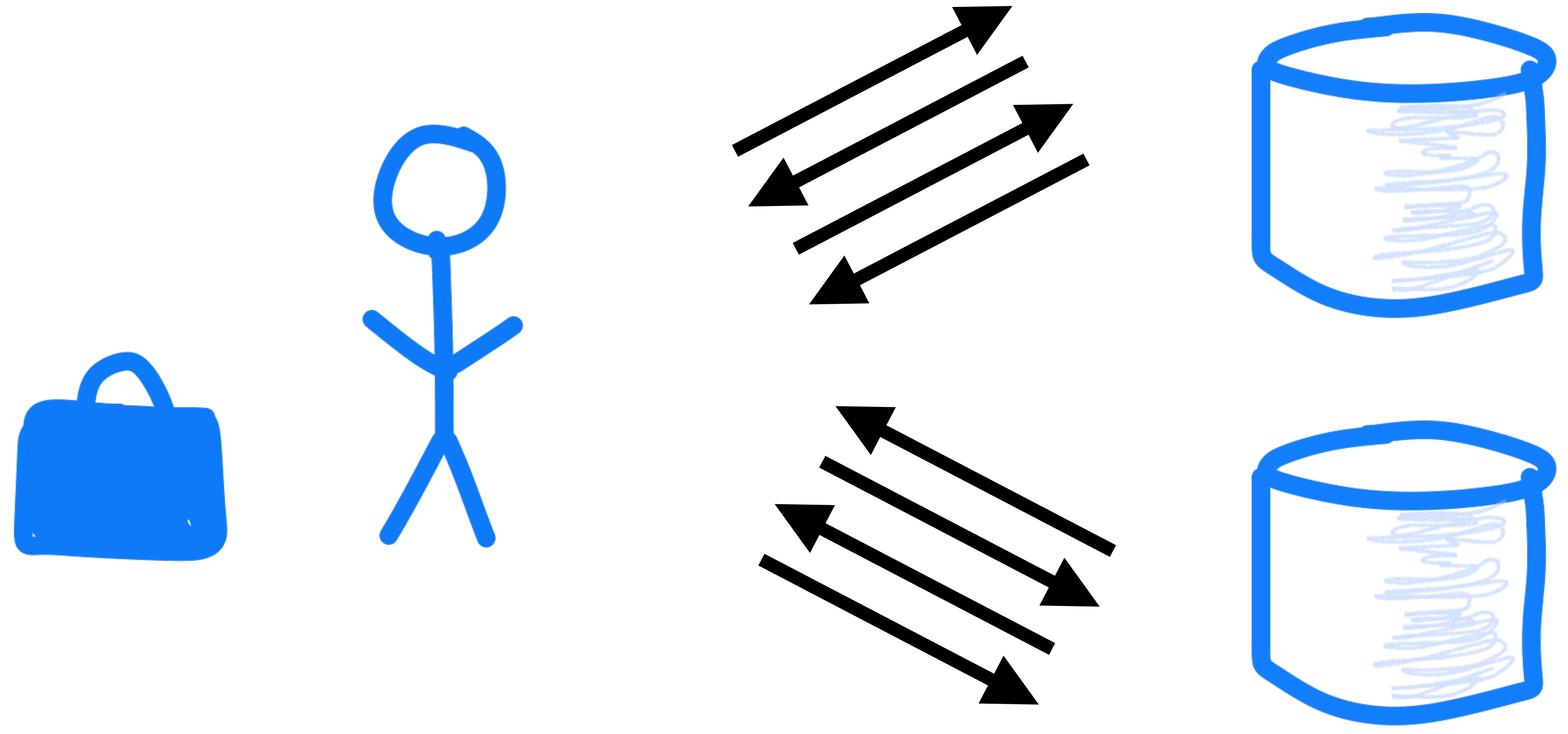
Solution: Change the PIR model to get sublinear time

Batch PIR with many, non-adaptive queries



[IKOS'04,HHG'13,GKL'10,LG'15,AS'16,H'16,ACLS'18,CHLR'18]

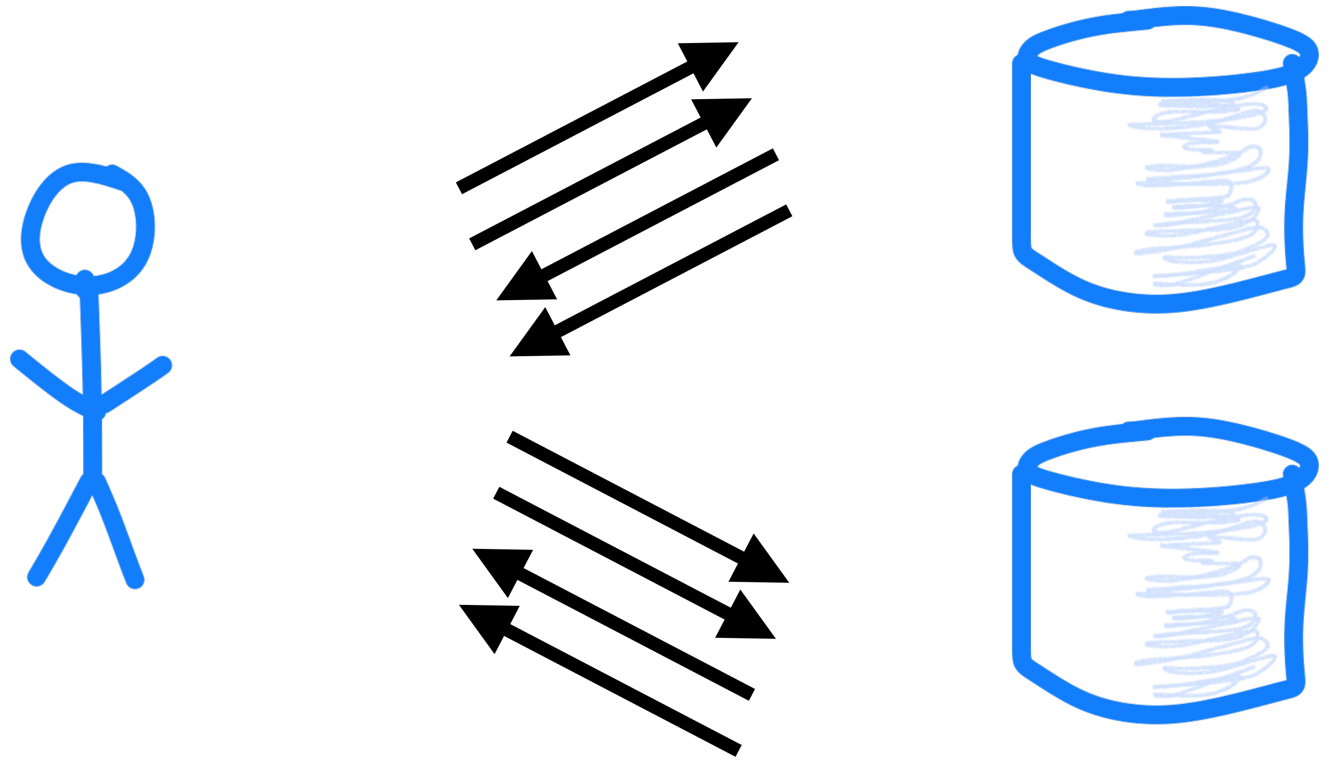
Offline/online PIR with stateful clients + many queries



[CK'20,SACM'21,KC'21,CHK'22,LP'23,ZLS'23,GZS'24,ISW'24,RMS'24,…]

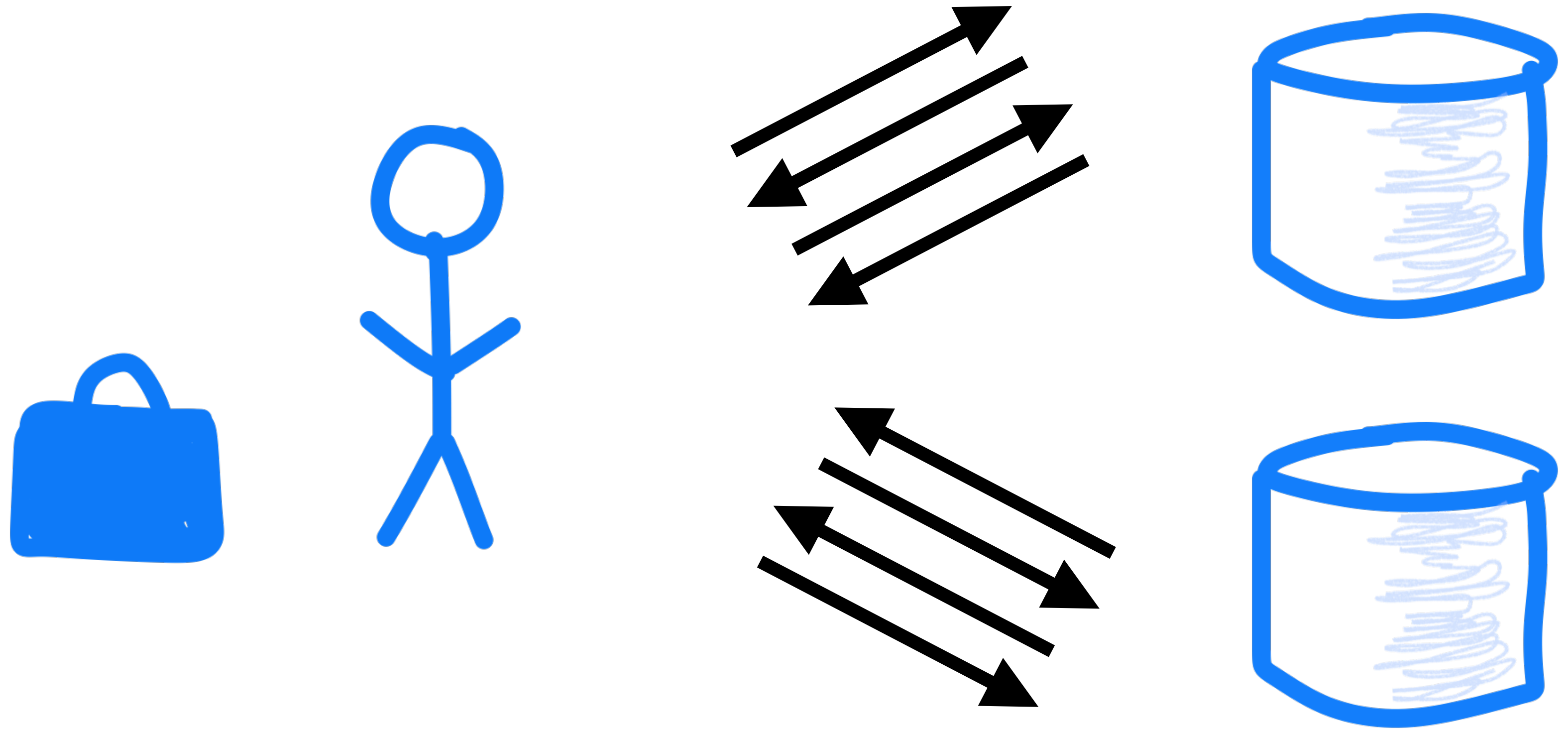
Solution: Change the PIR model to get sublinear time

Batch PIR with many, non-adaptive queries



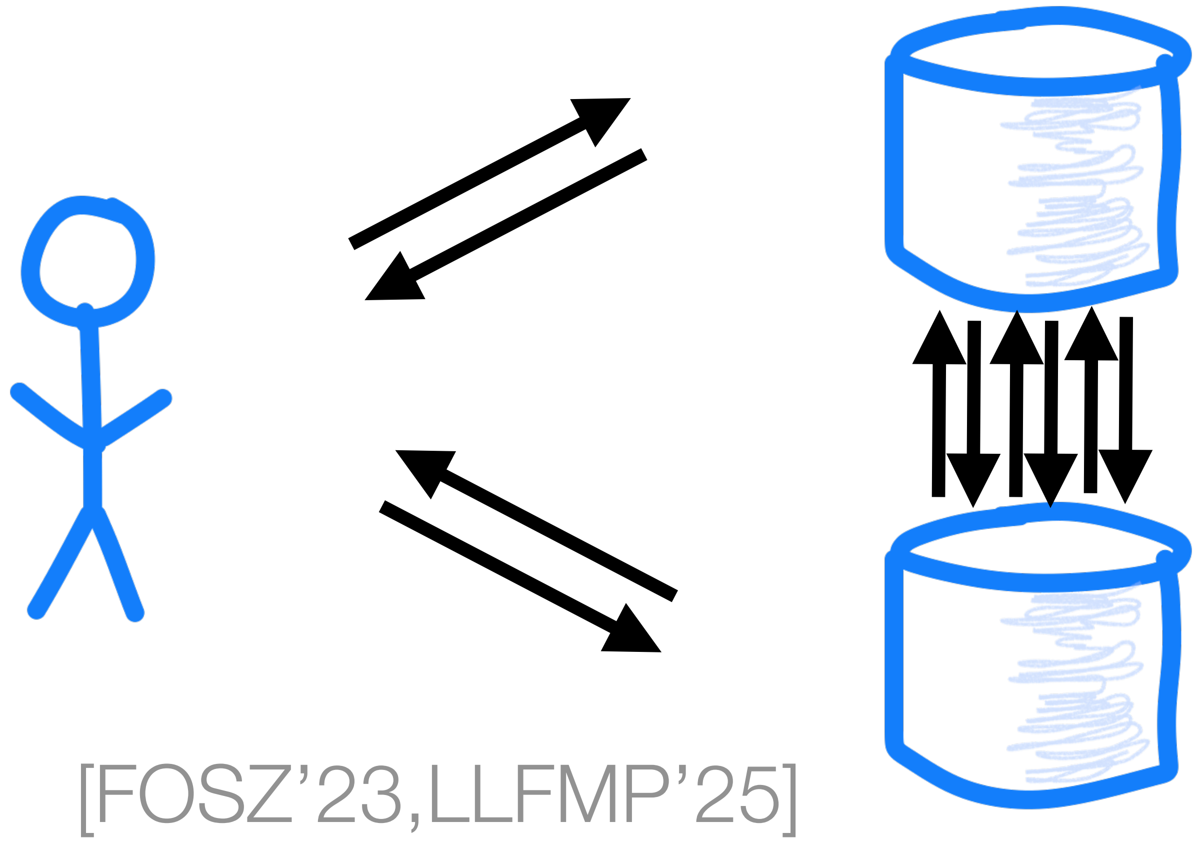
[IKOS’04,HHG’13,GKL’10,LG’15,AS’16,H’16,ACLS’18,CHLR’18]

Offline/online PIR with stateful clients + many queries



[CK’20,SACM’21,KC’21,CHK’22,LP’23,ZLS’23,GZS’24,ISW’24,RMS’24,…]

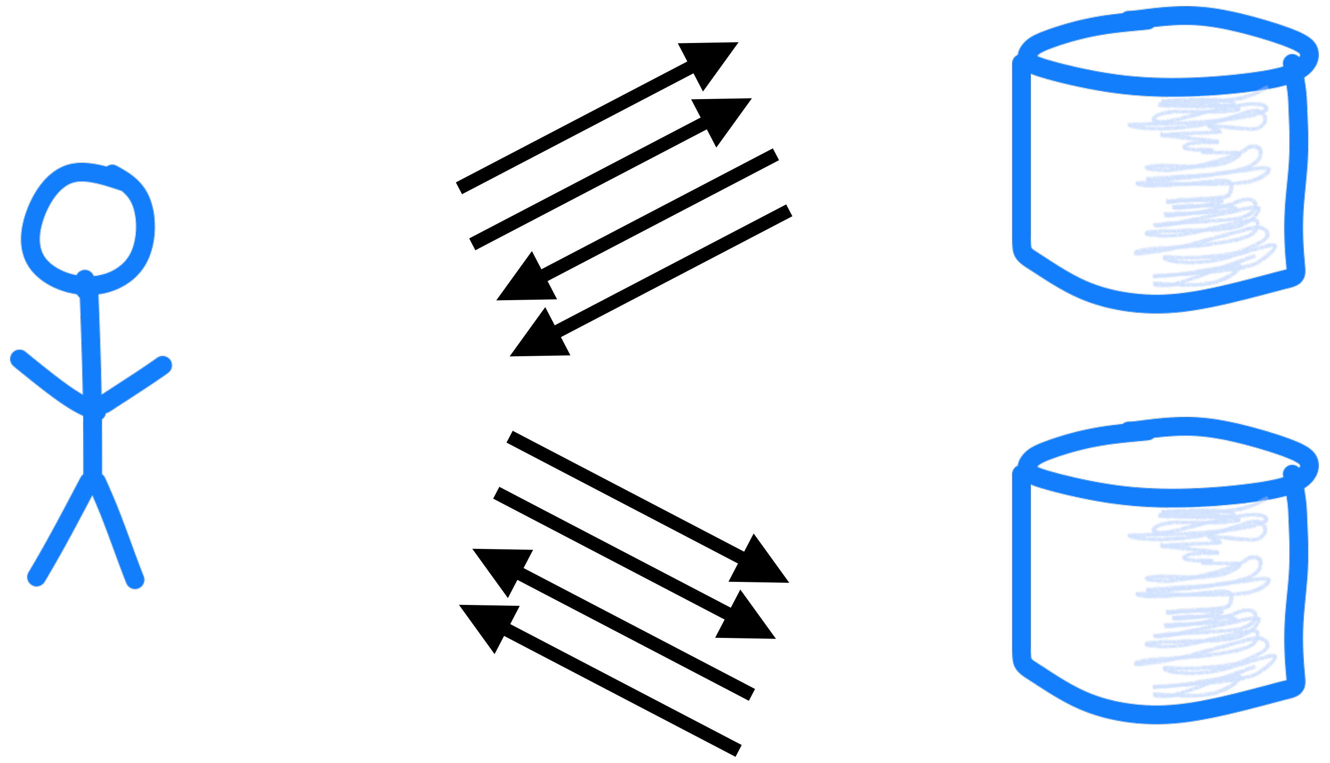
Distributed ORAM with communicating servers



[FOSZ’23,LLFMP’25]

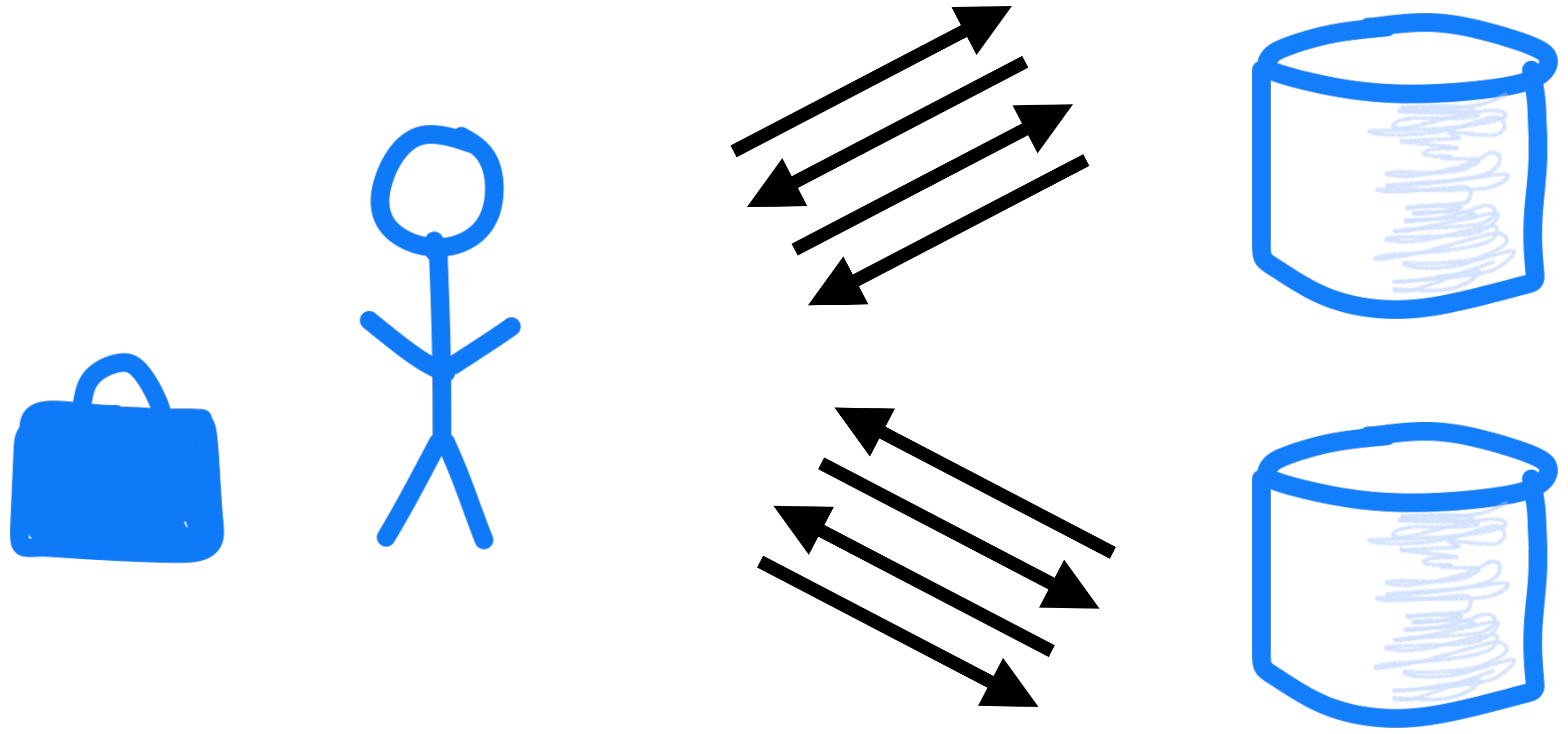
Solution: Change the PIR model to get sublinear time

Batch PIR with many, non-adaptive queries



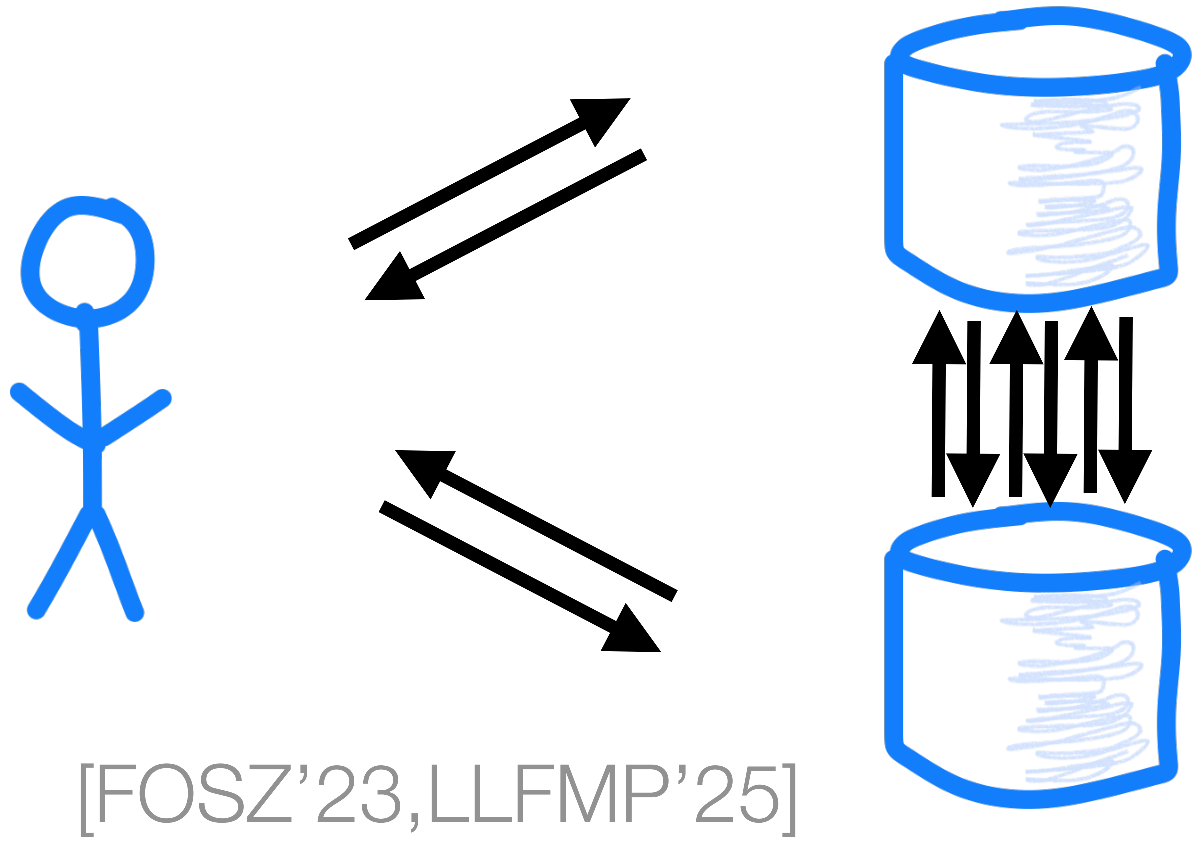
[IKOS'04,HHG'13,GKL'10,LG'15,AS'16,H'16,ACLS'18,CHLR'18]

Offline/online PIR with stateful clients + many queries



[CK'20,SACM'21,KC'21,CHK'22,LP'23,ZLS'23,GZS'24,ISW'24,RMS'24,…]

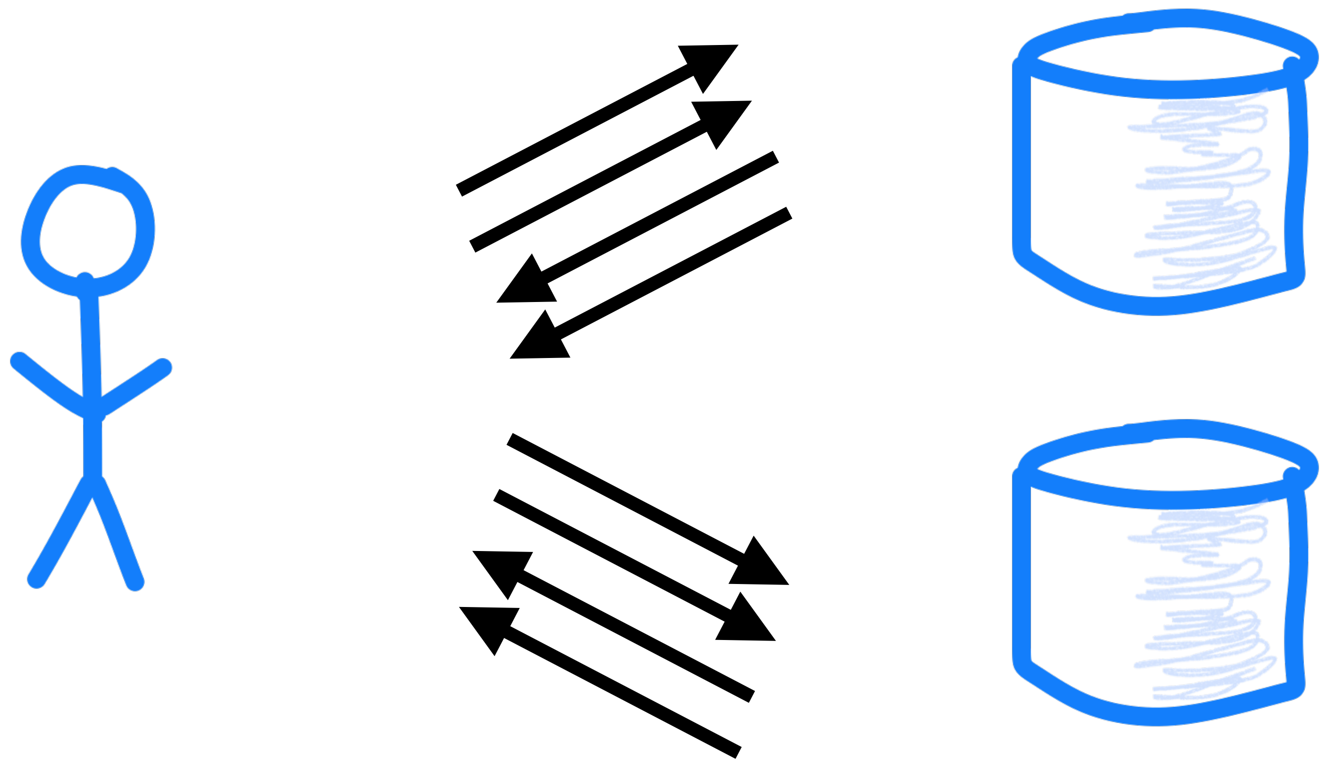
Distributed ORAM with communicating servers



[FOSZ'23,LLFMP'25]

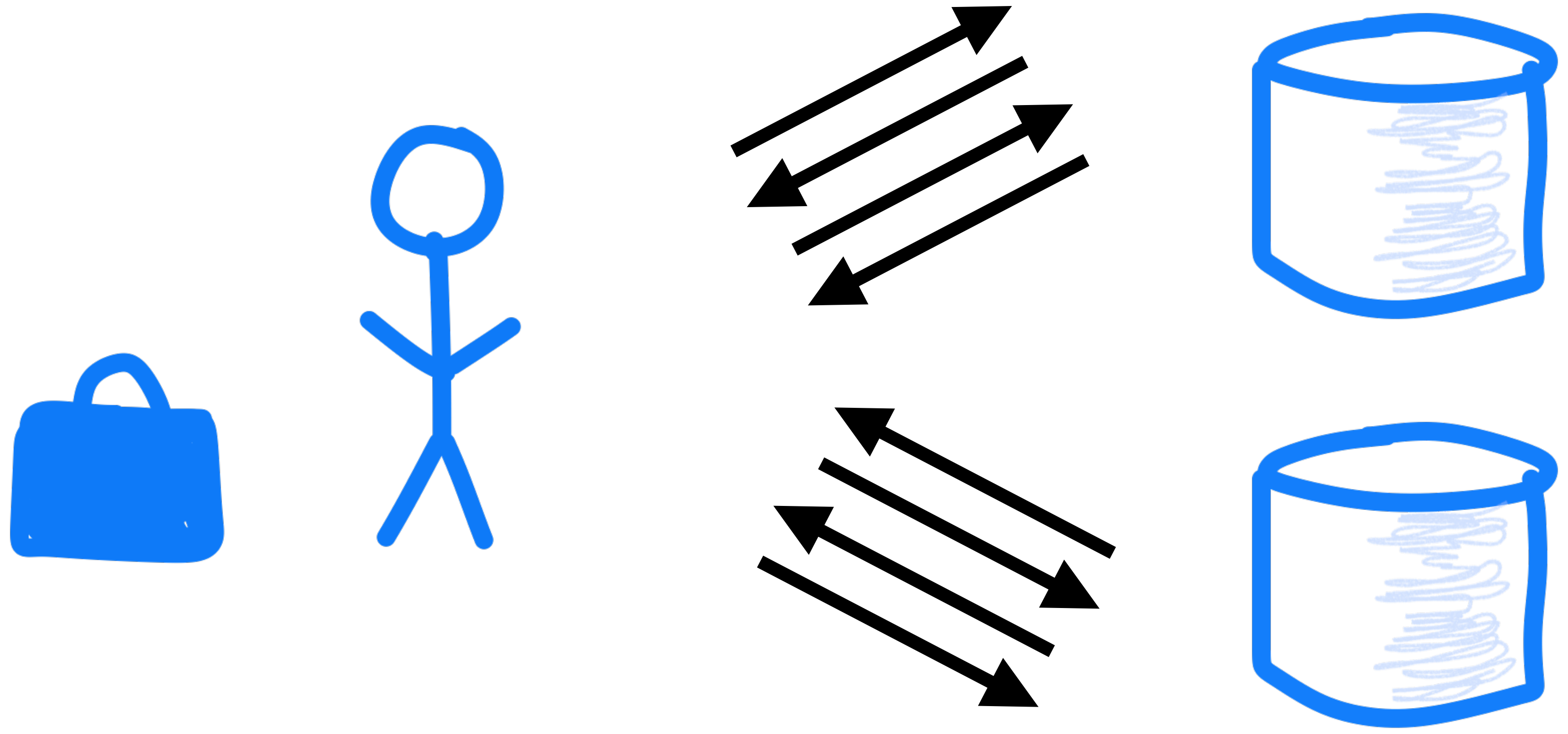
Solution: Change the PIR model to get sublinear time

Batch PIR with many, non-adaptive queries



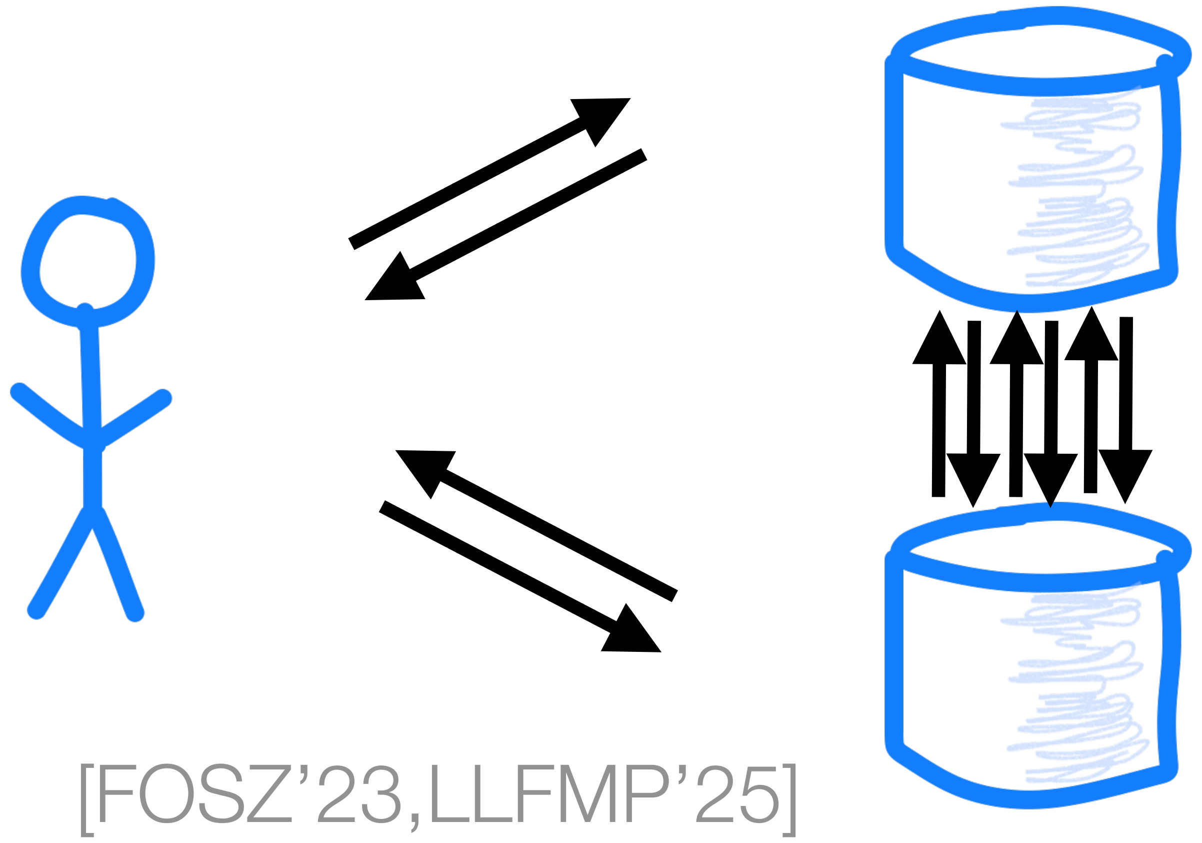
[IKOS'04,HHG'13,GKL'10,LG'15,AS'16,H'16,ACLS'18,CHLR'18]

Offline/online PIR with stateful clients + many queries



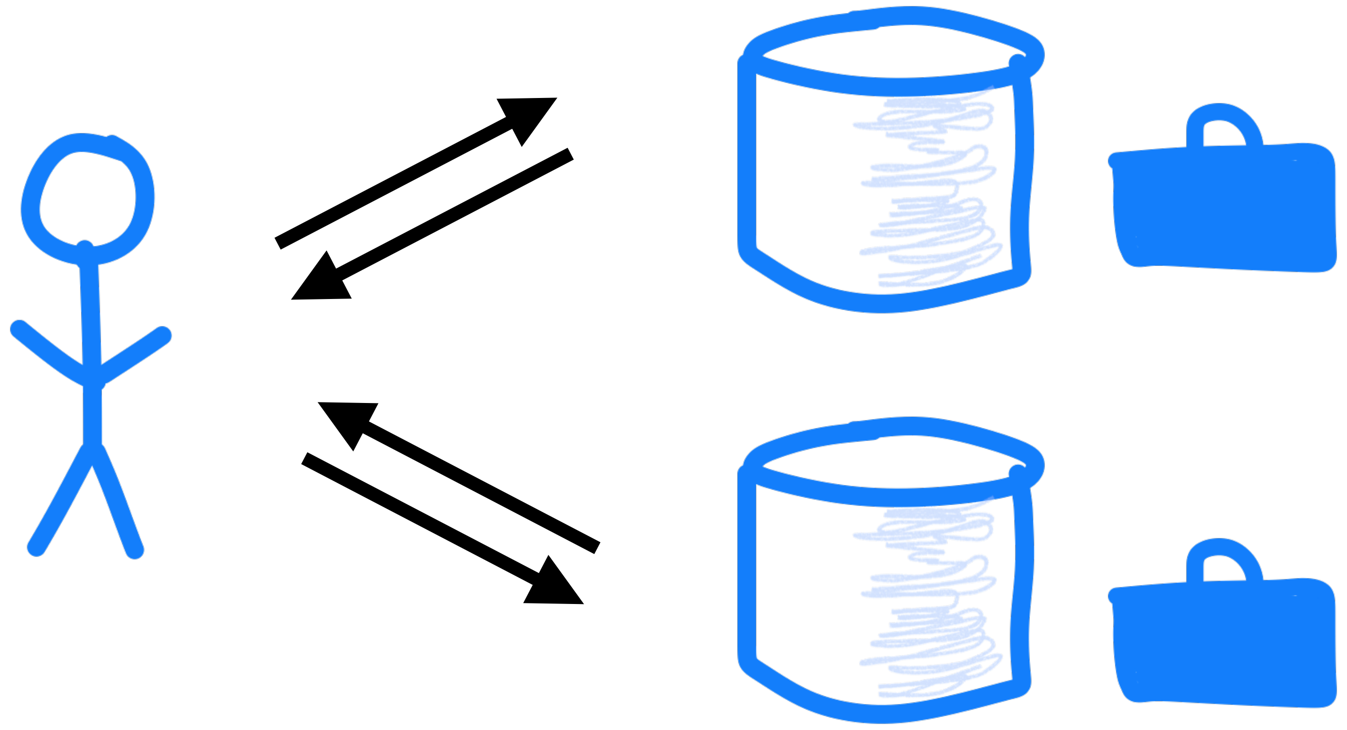
[CK'20,SACM'21,KC'21,CHK'22,LP'23,ZLS'23,GZS'24,ISW'24,RMS'24,…]

Distributed ORAM with communicating servers



[FOSZ'23,LLFMP'25]

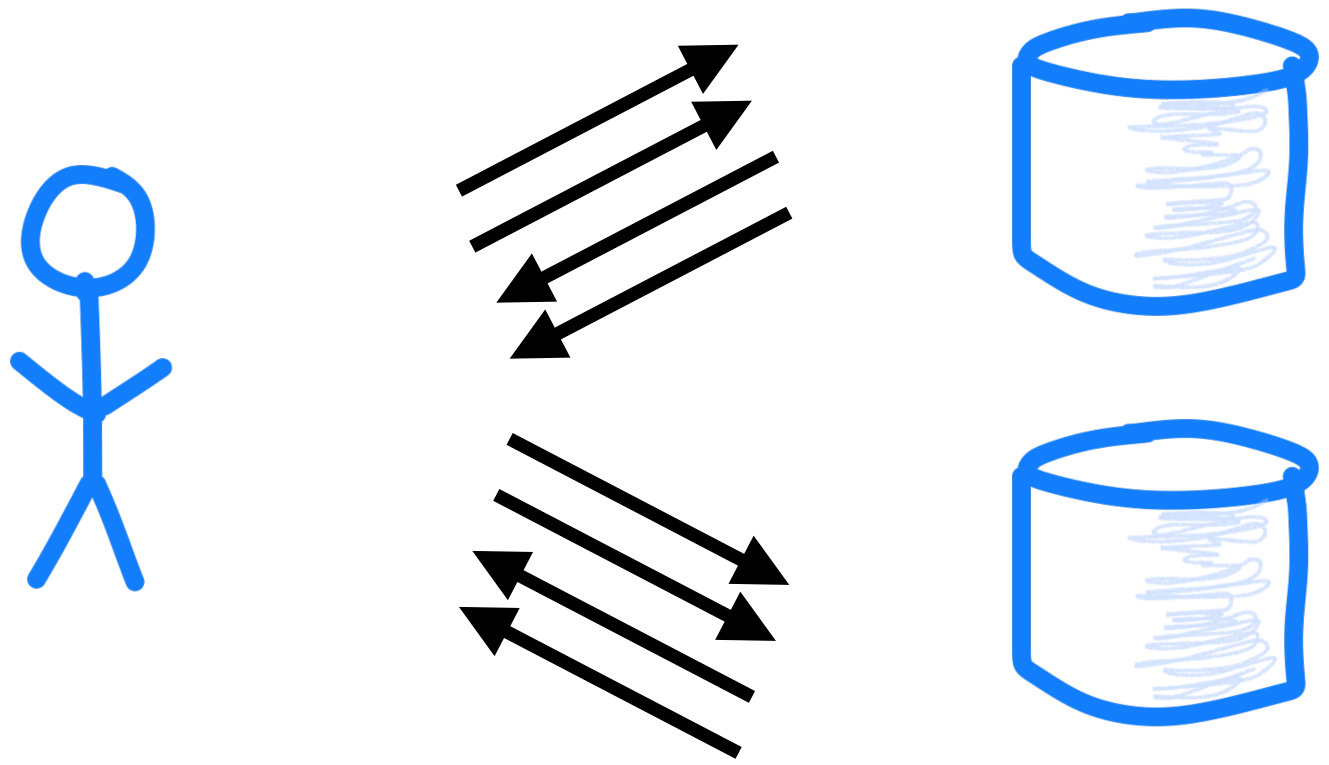
PIR with preprocessing



[BIM'00,BIPW'17,CHR'17,HOWW'18,LMW'23,GLMDS25]

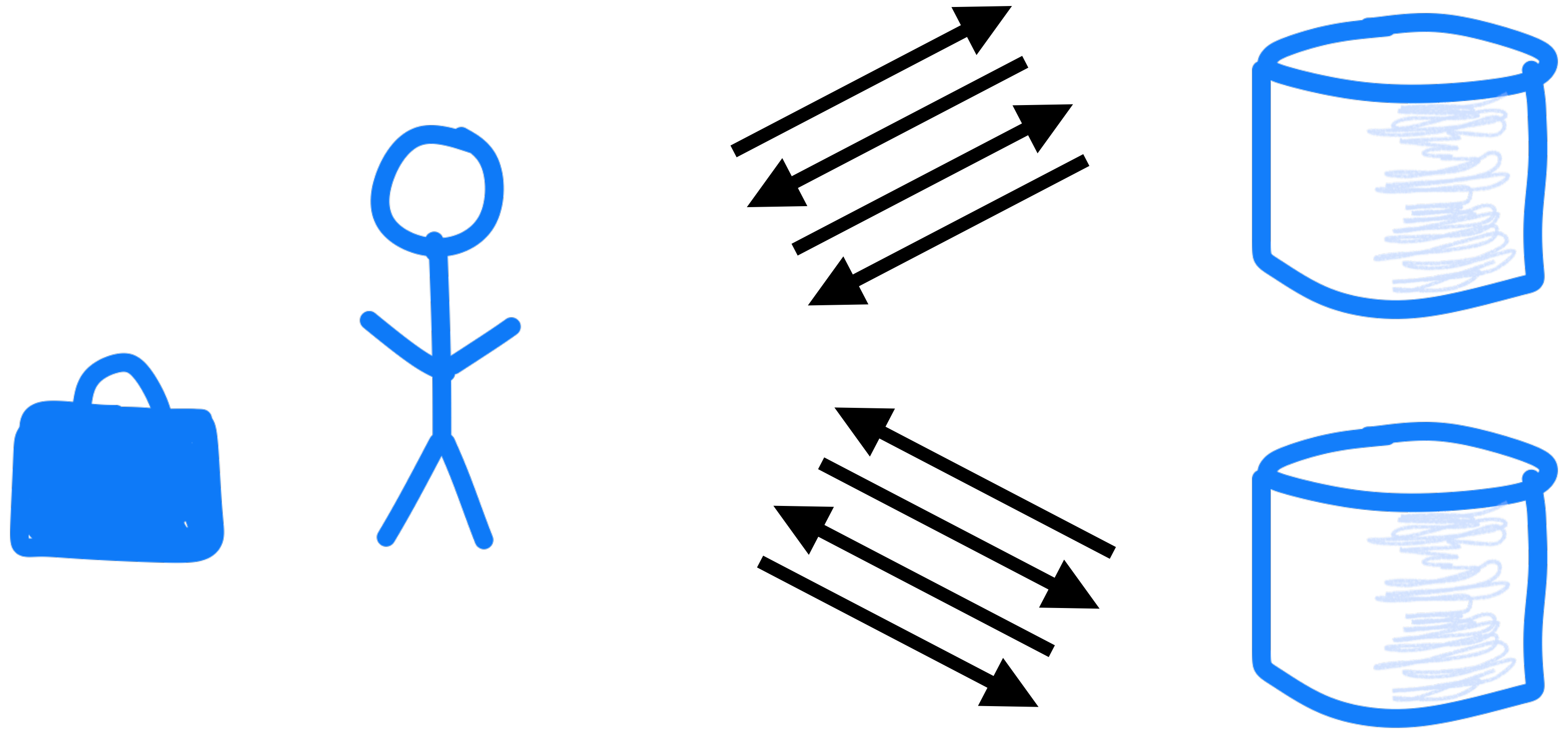
Solution: Change the PIR model to get sublinear time

Batch PIR with many, non-adaptive queries



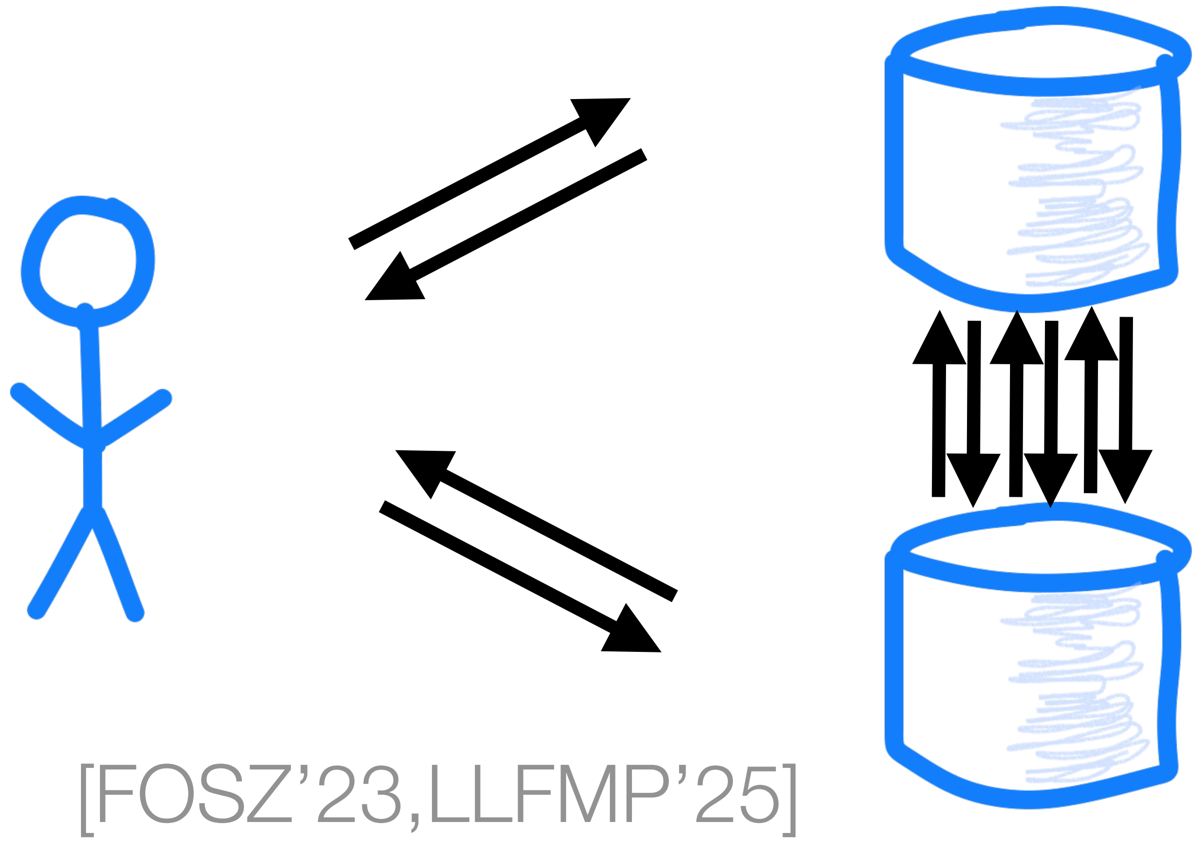
[IKOS'04,HHG'13,GKL'10,LG'15,AS'16,H'16,ACLS'18,CHLR'18]

Offline/online PIR with stateful clients + many queries



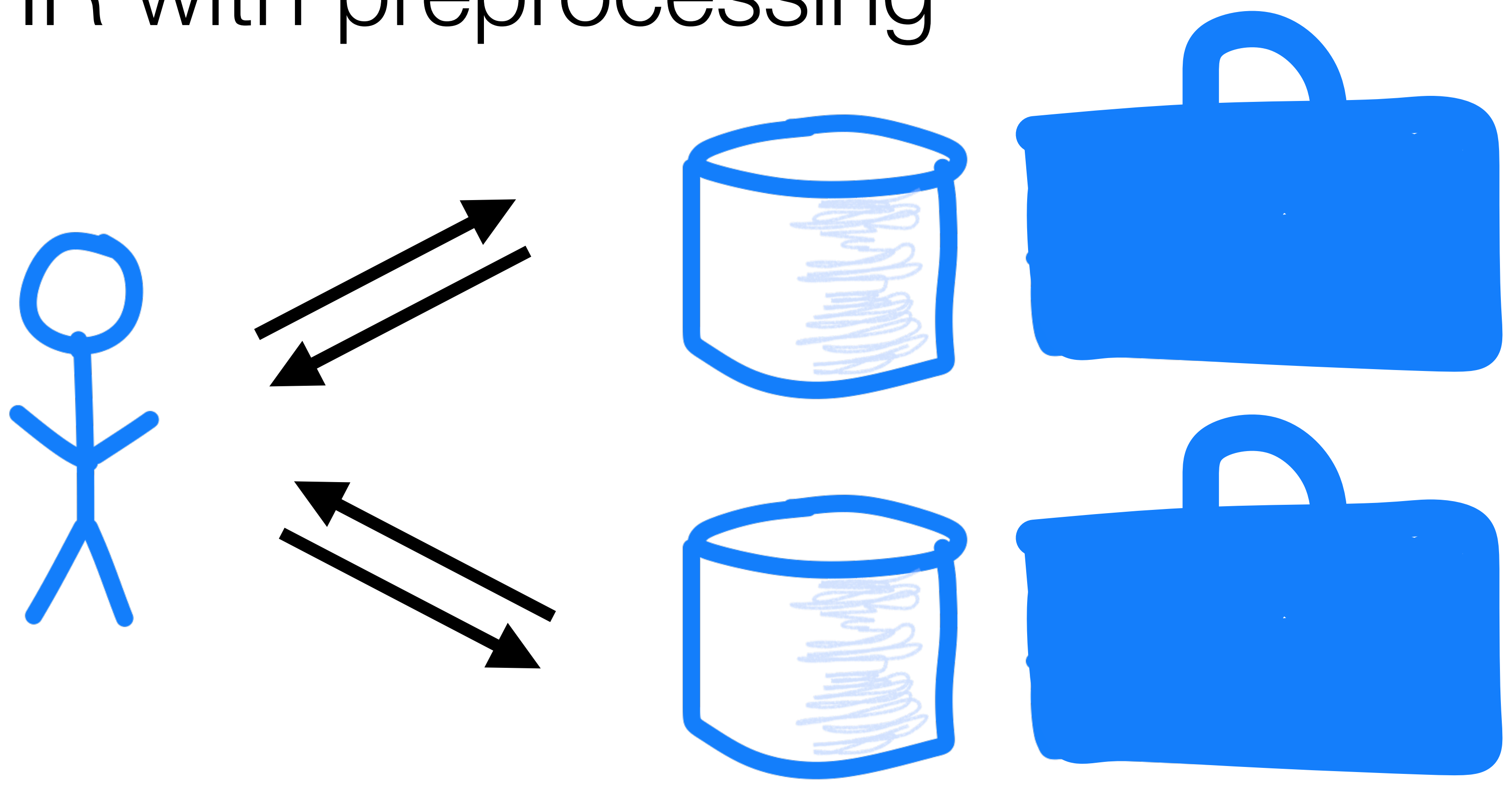
[CK'20,SACM'21,KC'21,CHK'22,LP'23,ZLS'23,GZS'24,ISW'24,RMS'24,…]

Distributed ORAM with communicating servers



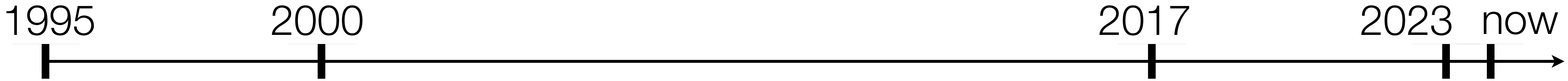
[FOSZ'23,LLFMP'25]

PIR with preprocessing



[BIM'00,BIPW'17,CHR'17,HOWW'18,LMW'23,GLMDS25]

25 years of work on PIR with preprocessing



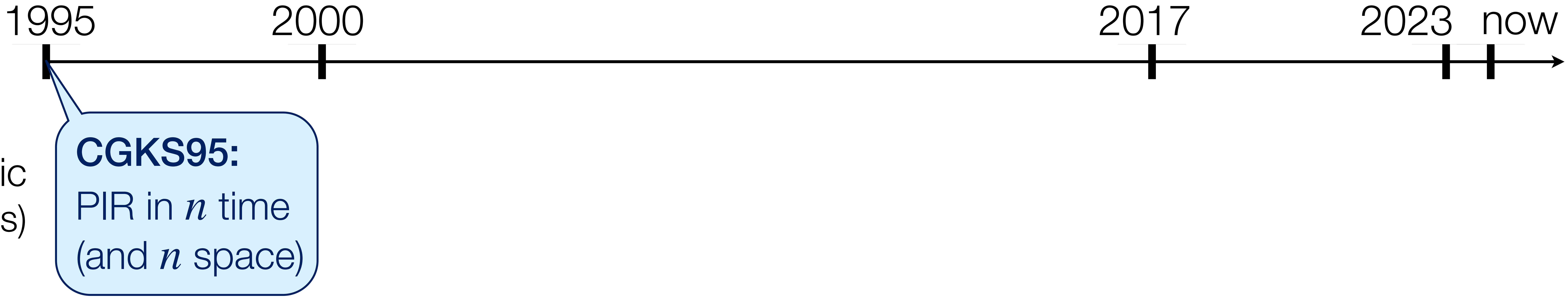
Info-theoretic
(2 servers)

Computational
(1 server)

Concrete storage
(2 GB database)

*Ignoring polylog savings in time

25 years of work on PIR with preprocessing



CGKS95:
PIR in n time
(and n space)

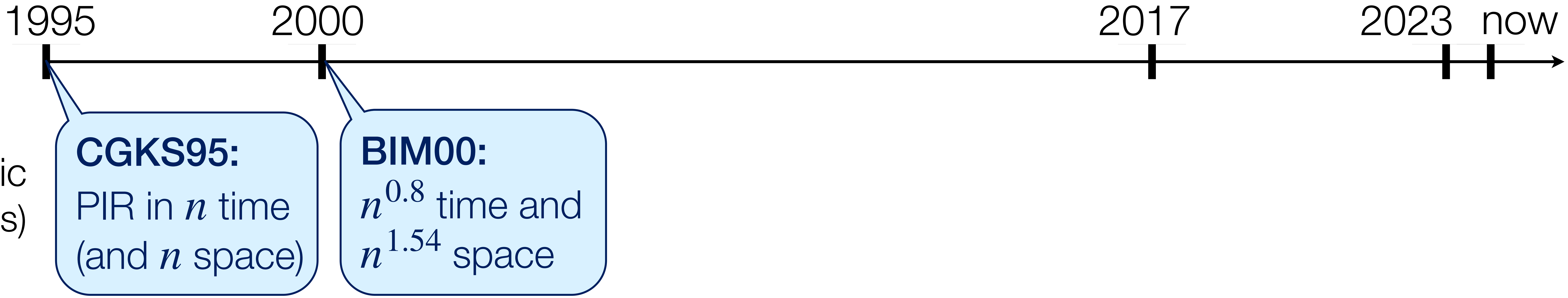
Info-theoretic
(2 servers)

Computational
(1 server)

Concrete storage
(2 GB database)

*Ignoring polylog savings in time

25 years of work on PIR with preprocessing



Info-theoretic
(2 servers)

CGKS95:
PIR in n time
(and n space)

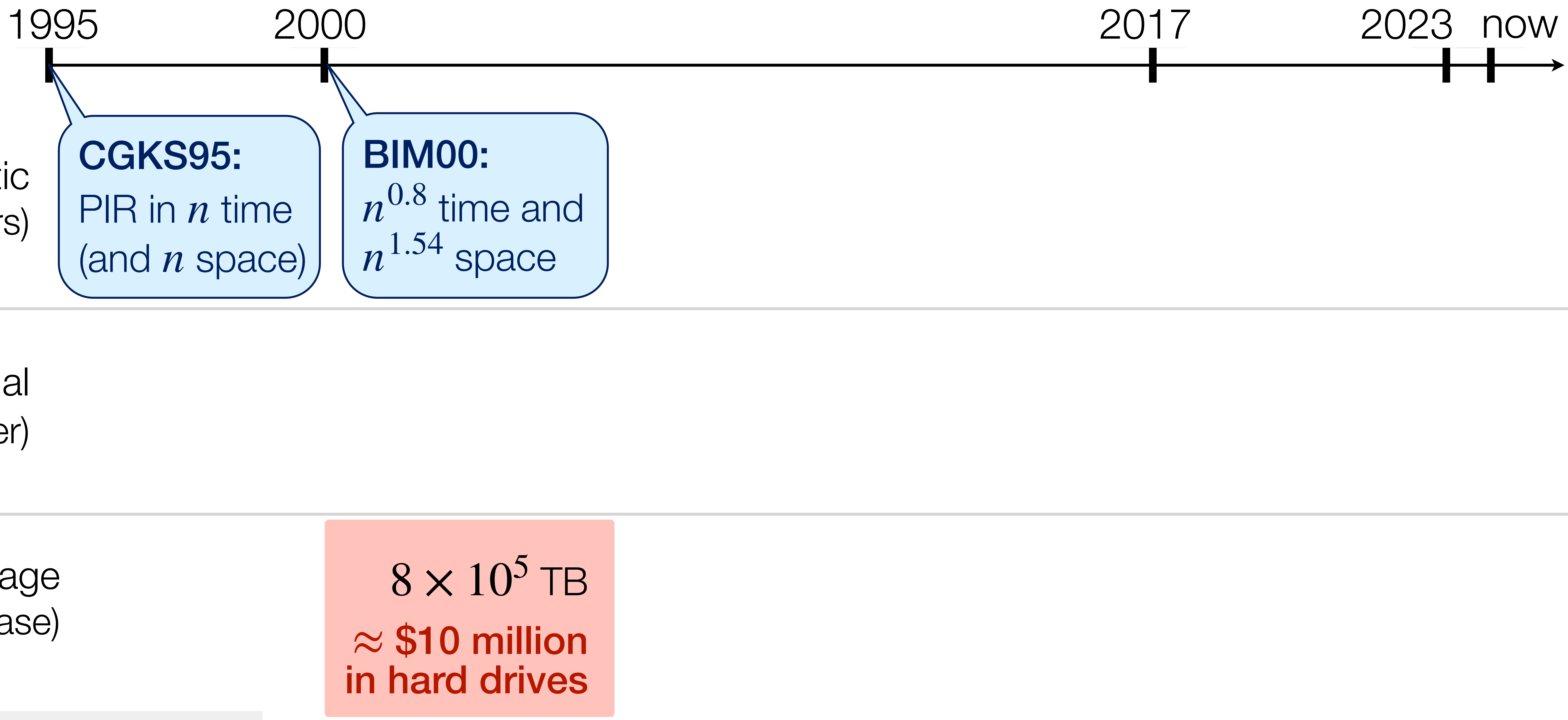
BIM00:
$n^{0.8}$ time and
$n^{1.54}$ space

Computational
(1 server)

Concrete storage
(2 GB database)

*Ignoring polylog savings in time

25 years of work on PIR with preprocessing



1995 2000 2017 2023 now

Info-theoretic
(2 servers)

CGKS95:
PIR in n time
(and n space)

BIM00:
$n^{0.8}$ time and
$n^{1.54}$ space

Computational
(1 server)

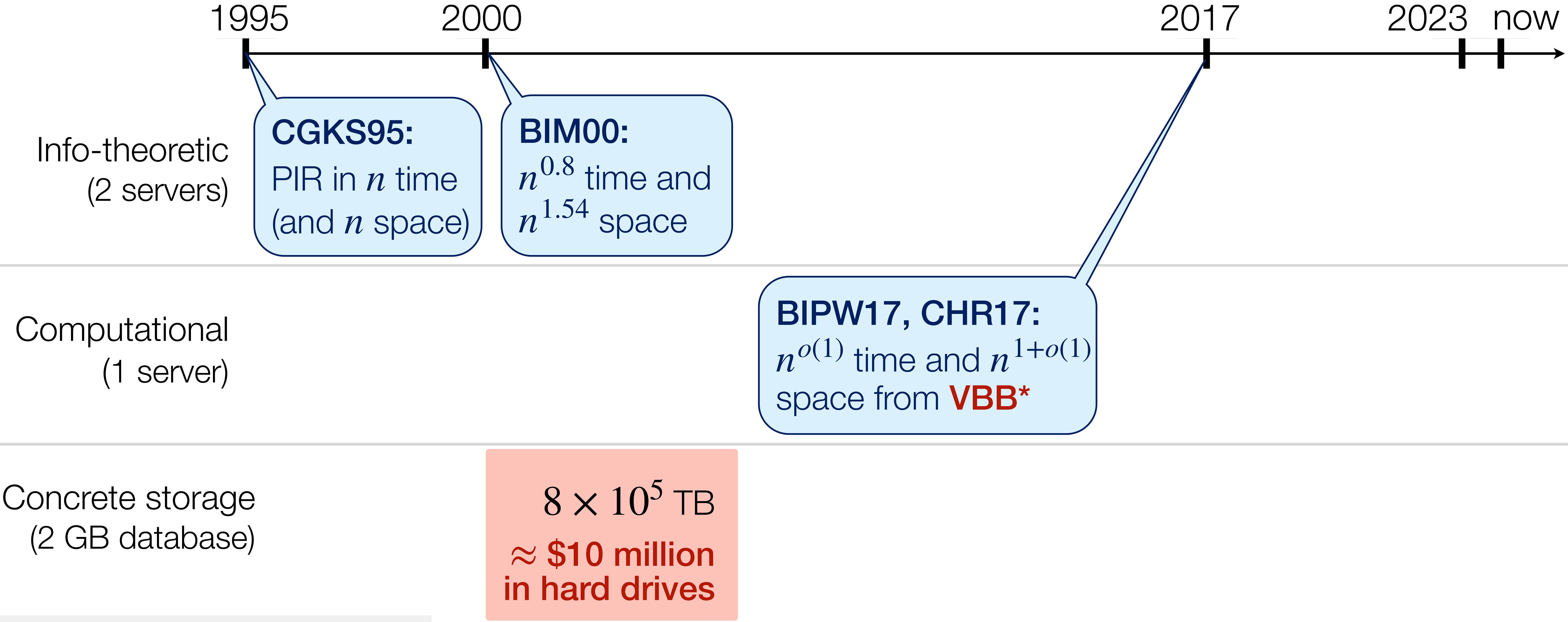
Concrete storage
(2 GB database)

8×10^5 TB

**≈ $10 million
in hard drives**

*Ignoring polylog savings in time

25 years of work on PIR with preprocessing



Info-theoretic (2 servers)

CGKS95: PIR in n time (and n space)

BIM00: $n^{0.8}$ time and $n^{1.54}$ space

Computational (1 server)

BIPW17, CHR17: $n^{o(1)}$ time and $n^{1+o(1)}$ space from **VBB***

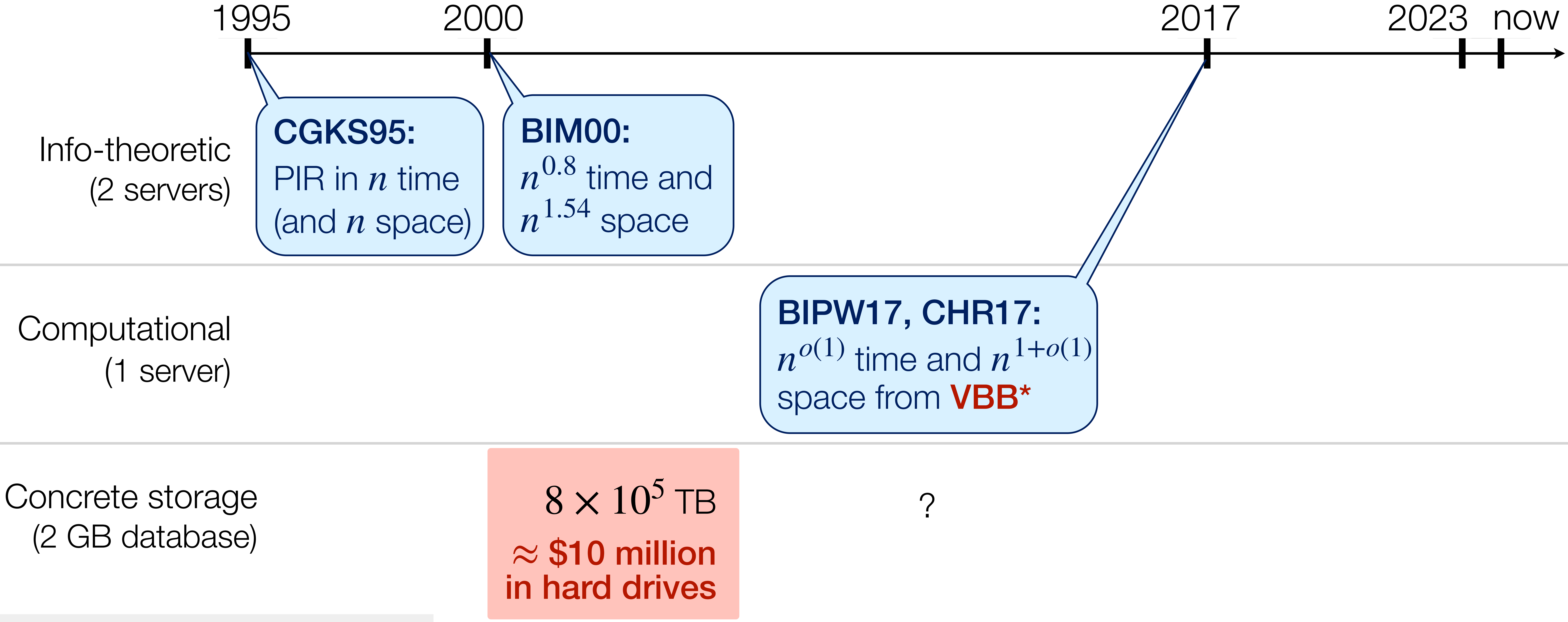
Concrete storage (2 GB database)

8×10^5 TB

≈ $10 million in hard drives

*Ignoring polylog savings in time

25 years of work on PIR with preprocessing



1995 2000 2017 2023 now

Info-theoretic (2 servers)

CGKS95: PIR in n time (and n space)

BIM00: $n^{0.8}$ time and $n^{1.54}$ space

Computational (1 server)

BIPW17, CHR17: $n^{o(1)}$ time and $n^{1+o(1)}$ space from **VBB***

Concrete storage (2 GB database)

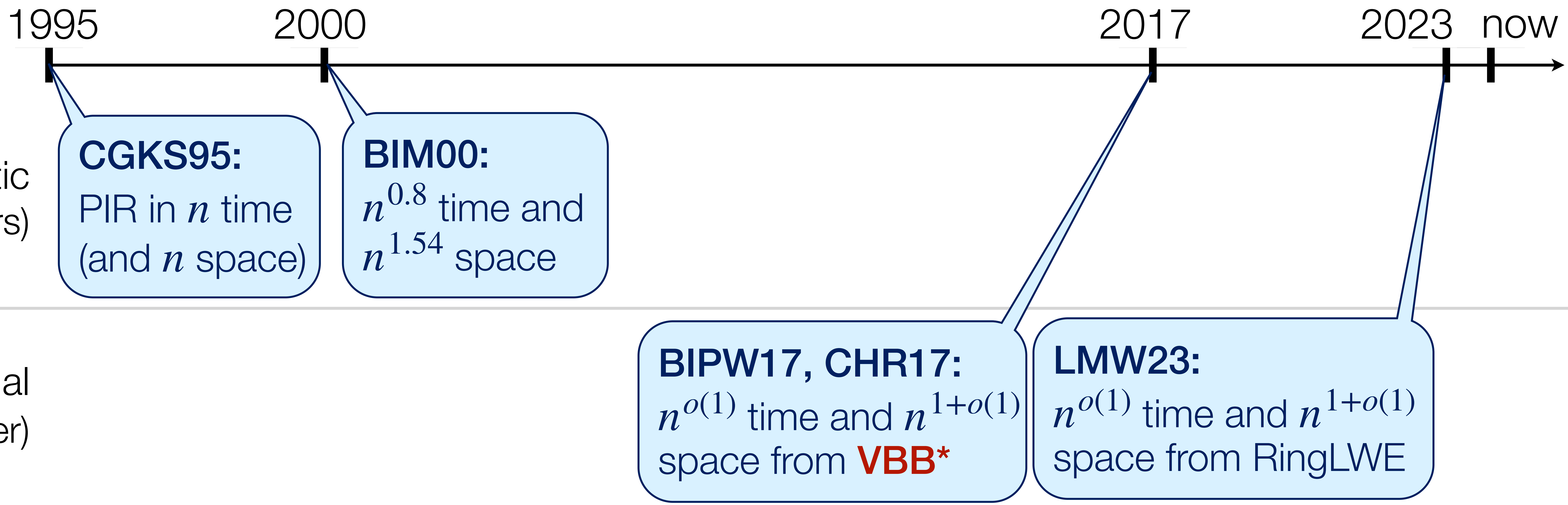
8×10^5 TB

≈ $10 million in hard drives

?

*Ignoring polylog savings in time

25 years of work on PIR with preprocessing



Info-theoretic
(2 servers)

CGKS95:
PIR in n time
(and n space)

BIM00:
$n^{0.8}$ time and
$n^{1.54}$ space

Computational
(1 server)

BIPW17, CHR17:
$n^{o(1)}$ time and $n^{1+o(1)}$
space from **VBB***

LMW23:
$n^{o(1)}$ time and $n^{1+o(1)}$
space from RingLWE

Concrete storage
(2 GB database)

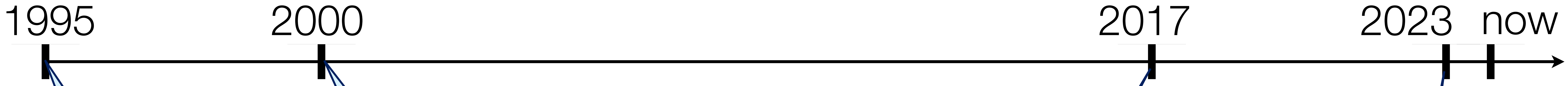
8×10^5 TB

**≈ $10 million
in hard drives**

?

*Ignoring polylog savings in time

25 years of work on PIR with preprocessing



Info-theoretic
(2 servers)

CGKS95:
PIR in n time
(and n space)

BIM00:
$n^{0.8}$ time and
$n^{1.54}$ space

Computational
(1 server)

BIPW17, CHR17:
$n^{o(1)}$ time and $n^{1+o(1)}$
space from **VBB***

LMW23:
$n^{o(1)}$ time and $n^{1+o(1)}$
space from RingLWE

Concrete storage
(2 GB database)

8×10^5 TB
**≈ $10 million
in hard drives**

?

2×10^6 TB
**≈ $20 million
in hard drives**

*Ignoring polylog savings in time

25 years of work on PIR with preprocessing

1995 | 2000 | 2017 | 2023 | now

Info-theoretic (2 servers)

CGKS95: PIR in n time (and n space)

BIM00: $n^{0.8}$ time and $n^{1.54}$ space

This talk: $n^{0.82}$ time and $n^{1+o(1)}$ space

Computational (1 server)

BIPW17, CHR17: $n^{o(1)}$ time and $n^{1+o(1)}$ space from **VBB***

LMW23: $n^{o(1)}$ time and $n^{1+o(1)}$ space from RingLWE

Concrete storage (2 GB database)

8×10^5 TB

≈ $10 million in hard drives

?

2×10^6 TB

≈ $20 million in hard drives

*Ignoring polylog savings in time

25 years of work on PIR with preprocessing

1995
2000
2017
2023
now

Info-theoretic (2 servers)

CGKS95: PIR in n time (and n space)

BIM00: $n^{0.8}$ time and $n^{1.54}$ space

This talk: $n^{0.82}$ time and $n^{1+o(1)}$ space

Computational (1 server)

BIPW17, CHR17: $n^{o(1)}$ time and $n^{1+o(1)}$ space from **VBB***

LMW23: $n^{o(1)}$ time and $n^{1+o(1)}$ space from RingLWE

Concrete storage (2 GB database)

8×10^5 TB

≈ $10 million in hard drives

?

2×10^6 TB

≈ $20 million in hard drives

1 TB

≈ $10 in hard drives

*Ignoring polylog savings in time

Theorem. For any database of $n > 10^6$ bits, there exists information-theoretic, two-server PIR with preprocessing with:

- $1.5 \cdot \sqrt{\log_2 n} \cdot n$ bits of storage,
- $12 \cdot n^{0.82}$ server RAM lookups per query,
- $12 \cdot n^{0.82}$ bits of communication.

Theorem. For any database of $n > 10^6$ bits, there exists information-theoretic, two-server PIR with preprocessing with:

- $1.5 \cdot \sqrt{\log_2 n} \cdot n$ bits of storage,
- $12 \cdot n^{0.82}$ server RAM lookups per query,
- $12 \cdot n^{0.82}$ bits of communication.

First info-theoretic PIR with

1. constant number of servers,
2. quasilinear storage, and
3. polynomially-sublinear time.

Theorem. For any database of $n > 10^6$ bits, there exists information-theoretic, two-server PIR with preprocessing with:

- $1.5 \cdot \sqrt{\log_2 n} \cdot n$ bits of storage,
- $12 \cdot n^{0.82}$ server RAM lookups per query,
- $12 \cdot n^{0.82}$ bits of communication.

First info-theoretic PIR with

1. constant number of servers,
2. quasilinear storage, and
3. polynomially-sublinear time.

Corollary 1: with two servers and compact LHE [known from DDH, DCR, QR, LWE], the server time is $n^{0.82} \cdot \mathsf{poly}(\lambda)$ and the communication is $n^{0.31} \cdot \mathsf{poly}(\lambda)$.

Theorem. For any database of $n > 10^6$ bits, there exists information-theoretic, two-server PIR with preprocessing with:

- $1.5 \cdot \sqrt{\log_2 n} \cdot n$ bits of storage,
- $12 \cdot n^{0.82}$ server RAM lookups per query,
- $12 \cdot n^{0.82}$ bits of communication.

First info-theoretic PIR with

1. constant number of servers,
2. quasilinear storage, and
3. polynomially-sublinear time.

Corollary 1: with two servers and compact LHE [known from DDH, DCR, QR, LWE], the server time is $n^{0.82} \cdot$ poly(λ) and the communication is $n^{0.31} \cdot$ poly(λ).

Corollary 2: with two servers and compact FHE*, the server time is $n^{0.82} \cdot$ poly(λ) and the communication is $\log(n) \cdot$ poly(λ).

Theorem. For any database of $n > 10^6$ bits, there exists information-theoretic, two-server PIR with preprocessing with:

- $1.5 \cdot \sqrt{\log_2 n} \cdot n$ bits of storage,
- $12 \cdot n^{0.82}$ server RAM lookups per query,
- $12 \cdot n^{0.82}$ bits of communication.

First info-theoretic PIR with

1. constant number of servers,
2. quasilinear storage, and
3. polynomially-sublinear time.

Corollary 1: with two servers and compact LHE [known from DDH, DCR, QR, LWE], the server time is $n^{0.82} \cdot \mathsf{poly}(\lambda)$ and the communication is $n^{0.31} \cdot \mathsf{poly}(\lambda)$.

Corollary 2: with two servers and compact FHE*, the server time is $n^{0.82} \cdot \mathsf{poly}(\lambda)$ and the communication is $\log(n) \cdot \mathsf{poly}(\lambda)$.

Our schemes support a broader time-space tradeoff, that strictly improves on prior work.

This talk

1. **Background:** PIR with preprocessing
2. **[HR26] New two-server PIR:** sublinear time, quasilinear space
3. **Evaluation:** what does this mean for practice?
4. **Bonuses** 😁
    - Reducing communication using crypto
    - **[HPR26]** Connecting multi-server PIR to complexity theory

This talk

1. **Background:** PIR with preprocessing

2. **[HR26] New two-server PIR:** sublinear time, quasilinear space
3. **Evaluation:** what does this mean for practice?
4. **Bonuses** 😁
    - Reducing communication using crypto
    - **[HPR26]** Connecting multi-server PIR to complexity theory

Prior information-theoretic PIR

[BIM00,GLMDS25]

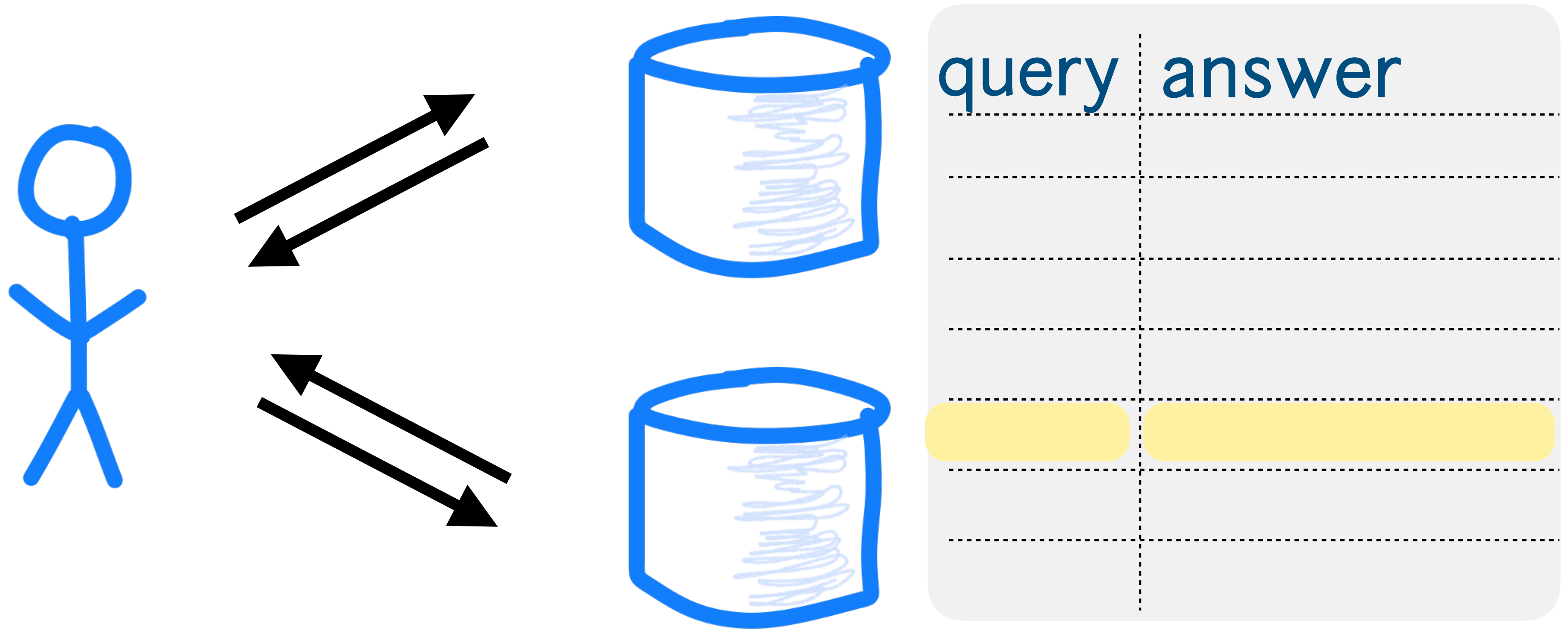
Prior information-theoretic PIR

[BIM00,GLMDS25]

1. Build "imbalanced" PIR with tiny queries
    - Query length $= (1 + o(1)) \cdot \log n$
    - Answer length $= \ell = O(n^{0.82})$

2. Precompute the answer to every query
    - To answer a query: read 1 location of length ℓ

➡ PIR in $n^{1.82+o(1)}$ space and $n^{0.82}$ time



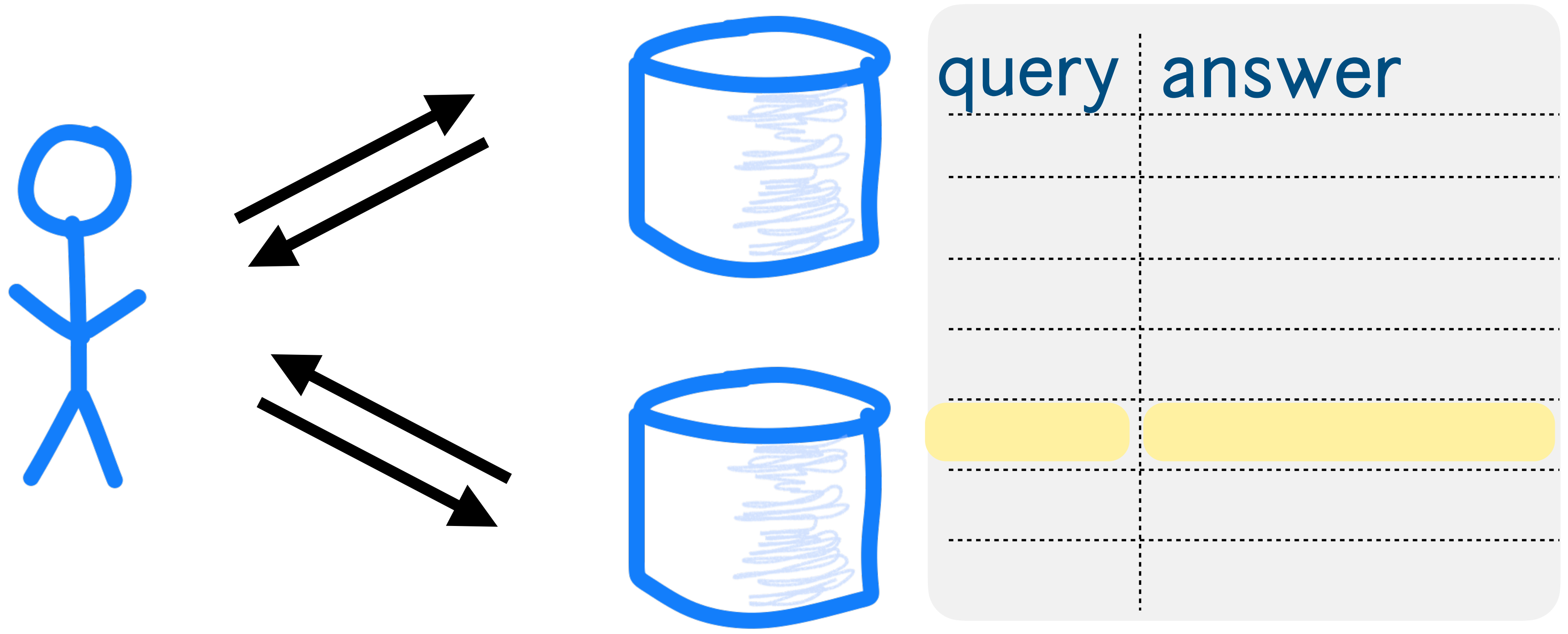
Prior information-theoretic PIR

[BIM00,GLMDS25]

1. Build "imbalanced" PIR with tiny queries
    - Query length $= (1 + o(1)) \cdot \log n$
    - Answer length $= \ell = O(n^{0.82})$

2. Precompute the answer to every query
    - To answer a query: read 1 location of length ℓ

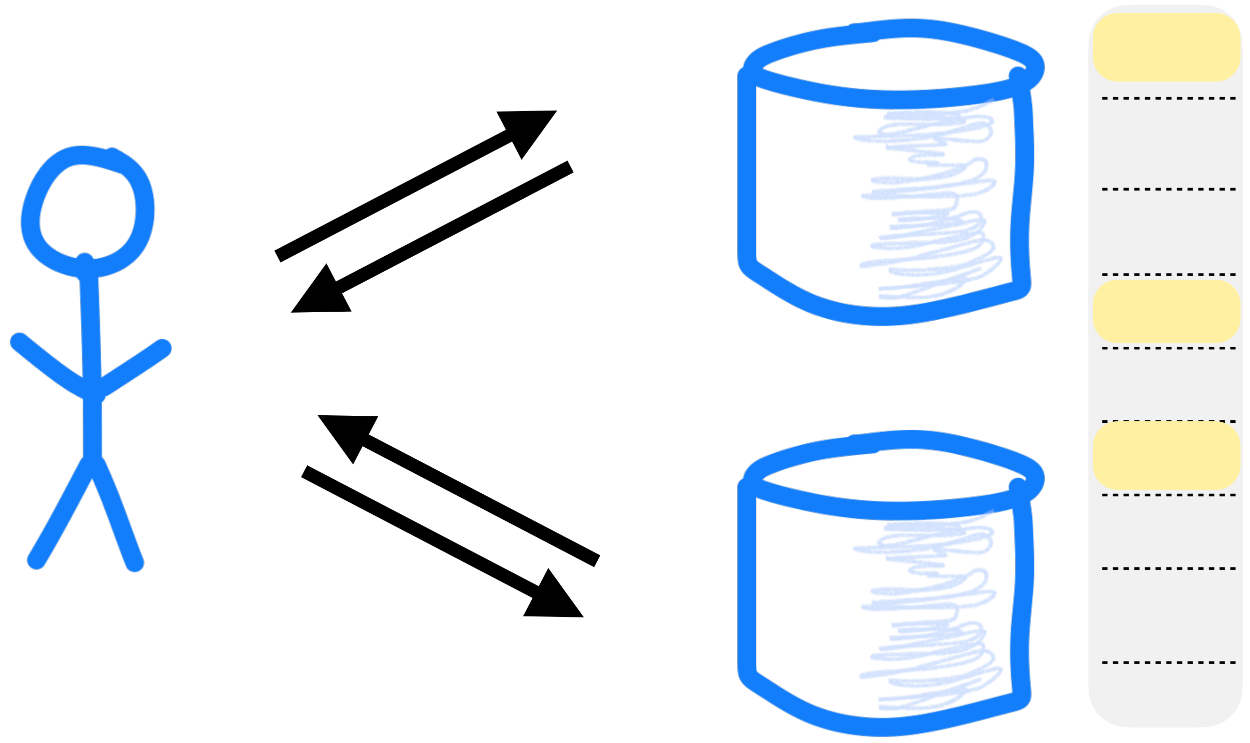
➡ PIR in $n^{1.82+o(1)}$ space and $n^{0.82}$ time



This work

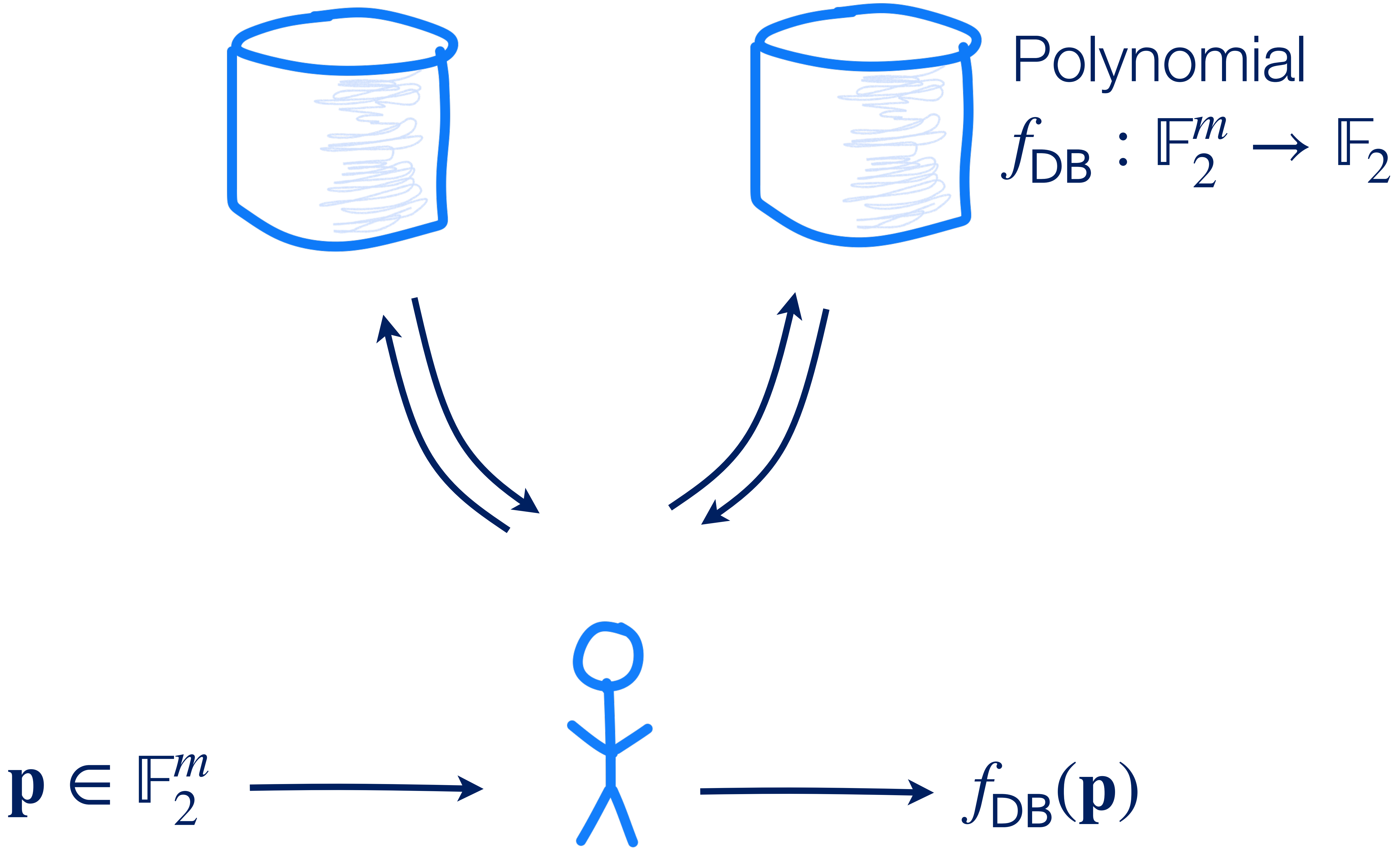
2. New data structure
    - To answer a query: read ℓ locations of length 1

➡ PIR in $n^{1+o(1)}$ space and $n^{0.82}$ time



Starting point: PIR from private polynomial evaluation

A common step in [BIM00,BIKR02,BIK05,WY05,BV11…]



Polynomial

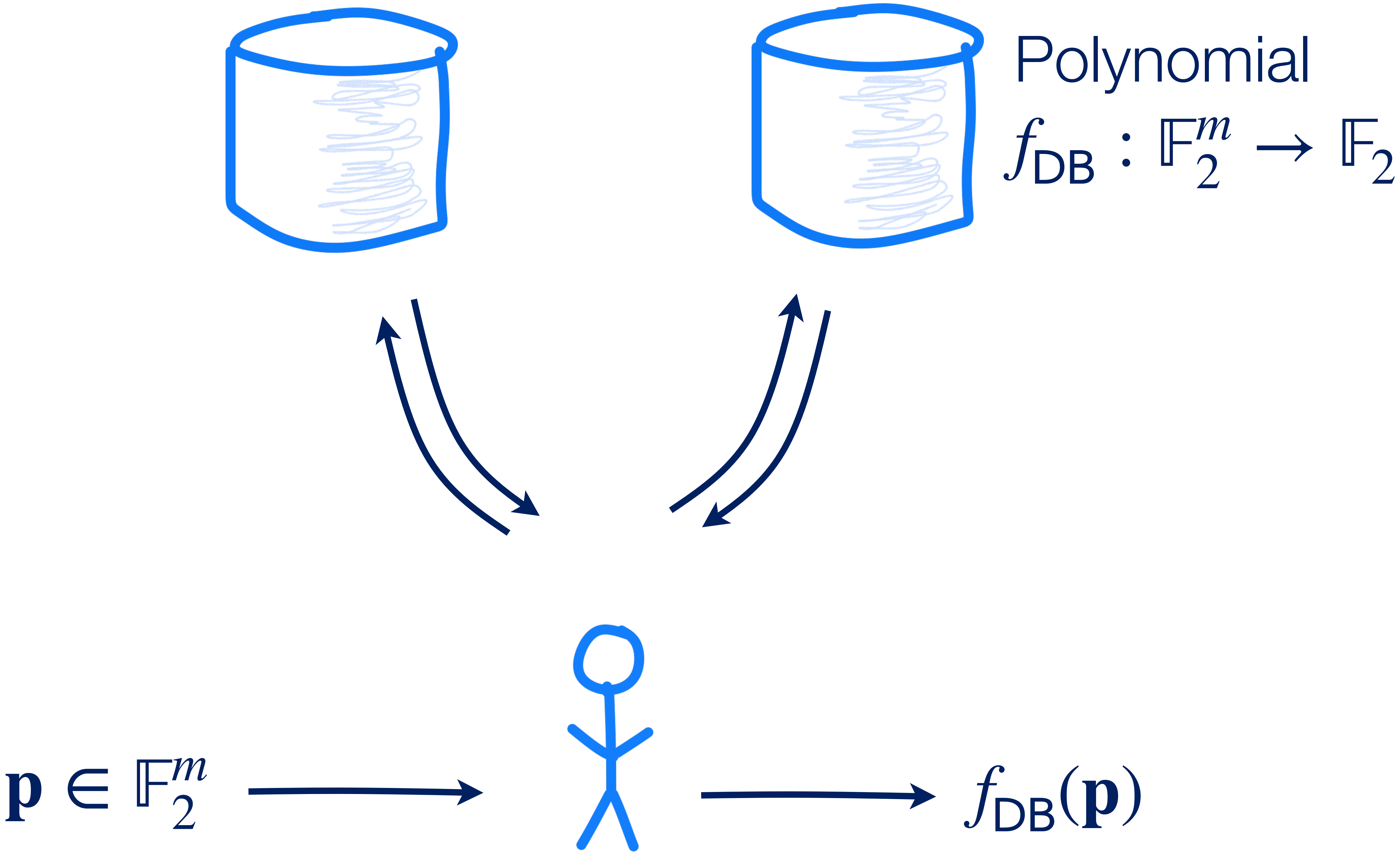
$f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$

$\mathbf{p} \in \mathbb{F}_2^m$

$f_{\mathsf{DB}}(\mathbf{p})$

Starting point: PIR from private polynomial evaluation

A common step in [BIM00,BIKR02,BIK05,WY05,BV11…]

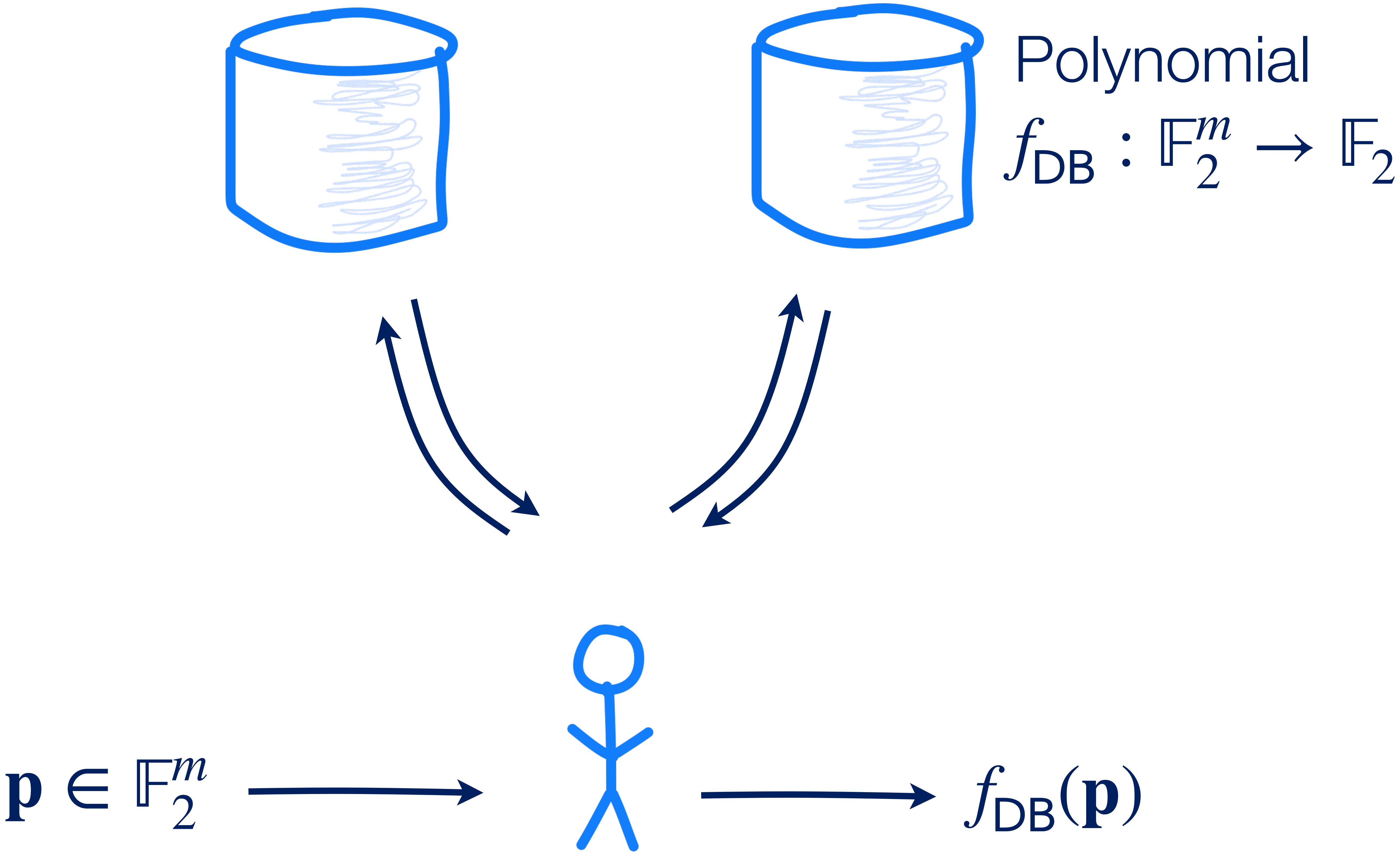

Polynomial
$f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$
$\mathbf{p} \in \mathbb{F}_2^m$
$f_{\mathsf{DB}}(\mathbf{p})$

We need: f_{DB} encodes the database

1. f_{DB} is homogenous and degree-D
2. $\binom{m}{D} \geq n$
3. $\mathbf{E}(j) = j$-th point in $\mathbb{F}_2^m$ of weight D
4. $\forall j \in [n], \quad f_{\mathsf{DB}}(\mathbf{E}(j)) = \mathsf{DB}_j$

Starting point: PIR from private polynomial evaluation

A common step in [BIM00,BIKR02,BIK05,WY05,BV11…]



Polynomial

$f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$

$\mathbf{p} \in \mathbb{F}_2^m$

$f_{\mathsf{DB}}(\mathbf{p})$

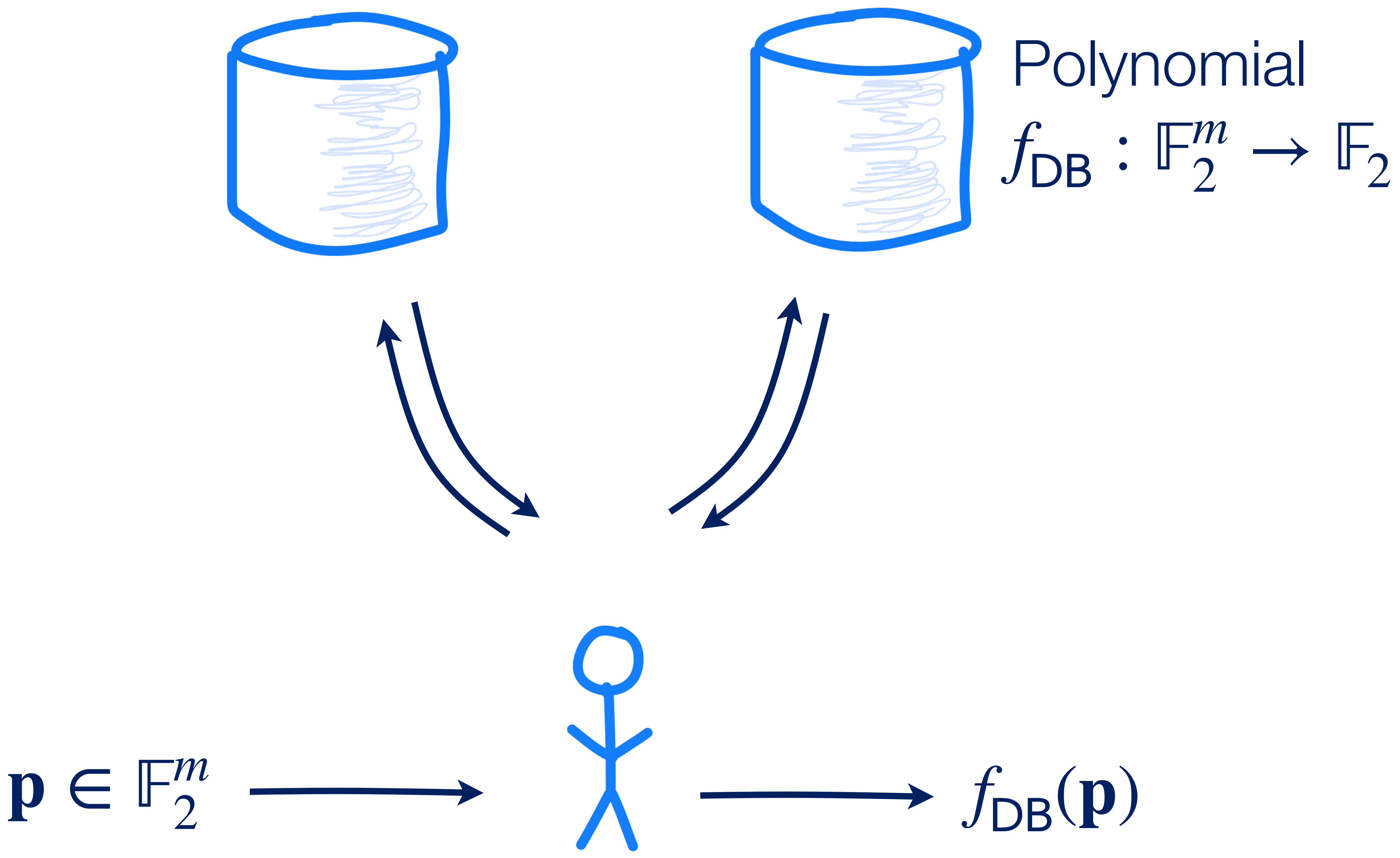
We need: f_{DB} encodes the database

1. f_{DB} is homogenous and degree-D
2. $\binom{m}{D} \geq n$
3. $\mathbf{E}(j) = j$-th point in $\mathbb{F}_2^m$ of weight D
4. $\forall j \in [n], \quad f_{\mathsf{DB}}(\mathbf{E}(j)) = \mathsf{DB}_j$

Correctness: for any f_{DB} and point $\mathbf{p}$, a user interacting with two honest servers learns $f_{\mathsf{DB}}(\mathbf{p})$.

Starting point: PIR from private polynomial evaluation

A common step in [BIM00,BIKR02,BIK05,WY05,BV11…]


Polynomial
$f_{\text{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$
$\mathbf{p} \in \mathbb{F}_2^m$
$f_{\text{DB}}(\mathbf{p})$

We need: f_{DB} encodes the database

1. f_{DB} is homogenous and degree-D
2. $\binom{m}{D} \geq n$
3. $\mathbf{E}(j) = j$-th point in $\mathbb{F}_2^m$ of weight D
4. $\forall j \in [n], \quad f_{\text{DB}}(\mathbf{E}(j)) = \text{DB}_j$

Correctness: for any f_{DB} and point $\mathbf{p}$, a user interacting with two honest servers learns $f_{\text{DB}}(\mathbf{p})$.

Privacy: each server learns nothing about $\mathbf{p}$, even if the server is malicious.

Starting point: PIR from private polynomial evaluation

A common step in [BIM00,BIKR02,BIK05,WY05,BV11…]

More explicitly:

$$f_{\mathsf{DB}}(\mathbf{y} \in \mathbb{F}_2^m) = \sum_{j=1}^{n} \mathsf{DB}_j \cdot \mathbf{y}^{\mathbf{E}(j)}, \text{ where}$$

$$\mathbf{a}^{\mathbf{b}} = \prod_{i=1}^{m} a_i^{b_i}.$$

We need: f_{DB} encodes the database

1. f_{DB} is homogenous and degree-D
2. $\binom{m}{D} \geq n$
3. $\mathbf{E}(j) = j$-th point in $\mathbb{F}_2^m$ of weight D
4. $\forall j \in [n], \quad f_{\mathsf{DB}}(\mathbf{E}(j)) = \mathsf{DB}_j$

Correctness: for any f_{DB} and point $\mathbf{p}$, a user interacting with two honest servers learns $f_{\mathsf{DB}}(\mathbf{p})$.

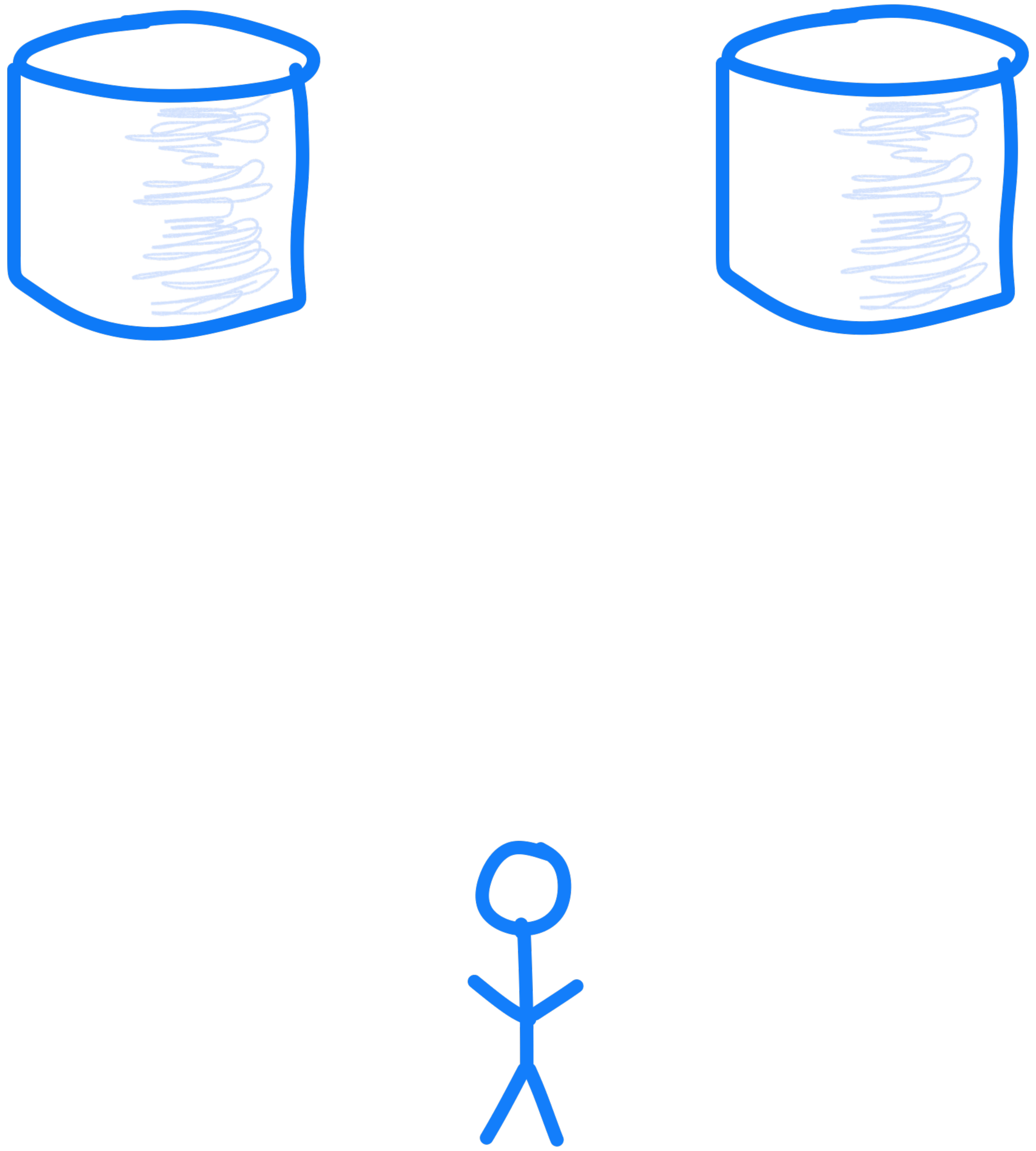
Privacy: each server learns nothing about $\mathbf{p}$, even if the server is malicious.

PIR from private polynomial evaluation

Scheme 1a: from Lagrange Interpolation [Sha79]

PIR from private polynomial evaluation

Scheme 1a: from Lagrange Interpolation [Sha79]



Homogenous degree-D

$f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$

PIR from private polynomial evaluation

Scheme 1a: from Lagrange Interpolation [Sha79]

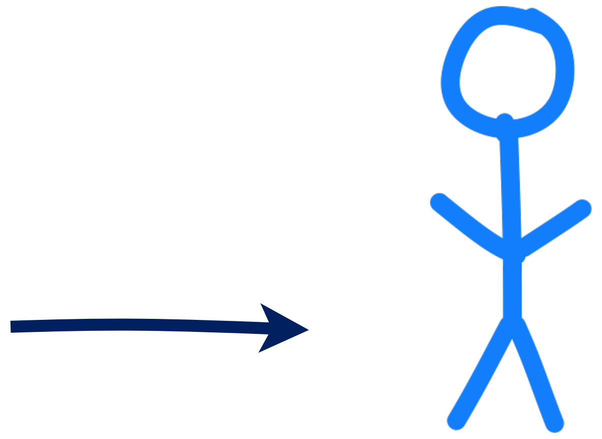
Homogenous degree-D

$f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$

Point $\mathbf{p} \in \mathbb{F}_2^m$

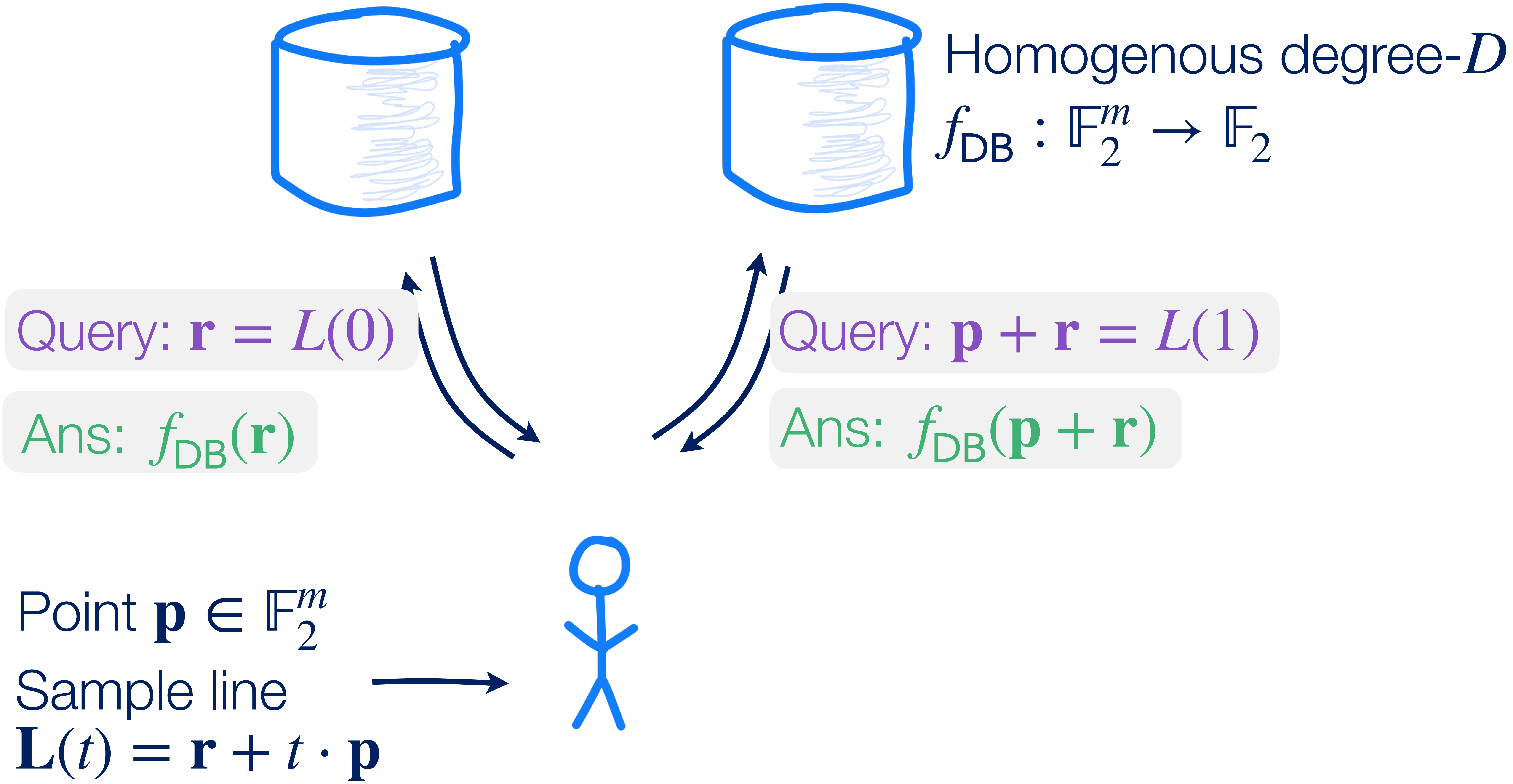
Sample line

$\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$



PIR from private polynomial evaluation

Scheme 1a: from Lagrange Interpolation [Sha79]



Homogenous degree-D

$f_{\mathsf{DB}} : \mathbb{F}_2^m \rightarrow \mathbb{F}_2$

Query: $\mathbf{r} = L(0)$

Ans: $f_{\mathsf{DB}}(\mathbf{r})$

Query: $\mathbf{p} + \mathbf{r} = L(1)$

Ans: $f_{\mathsf{DB}}(\mathbf{p} + \mathbf{r})$

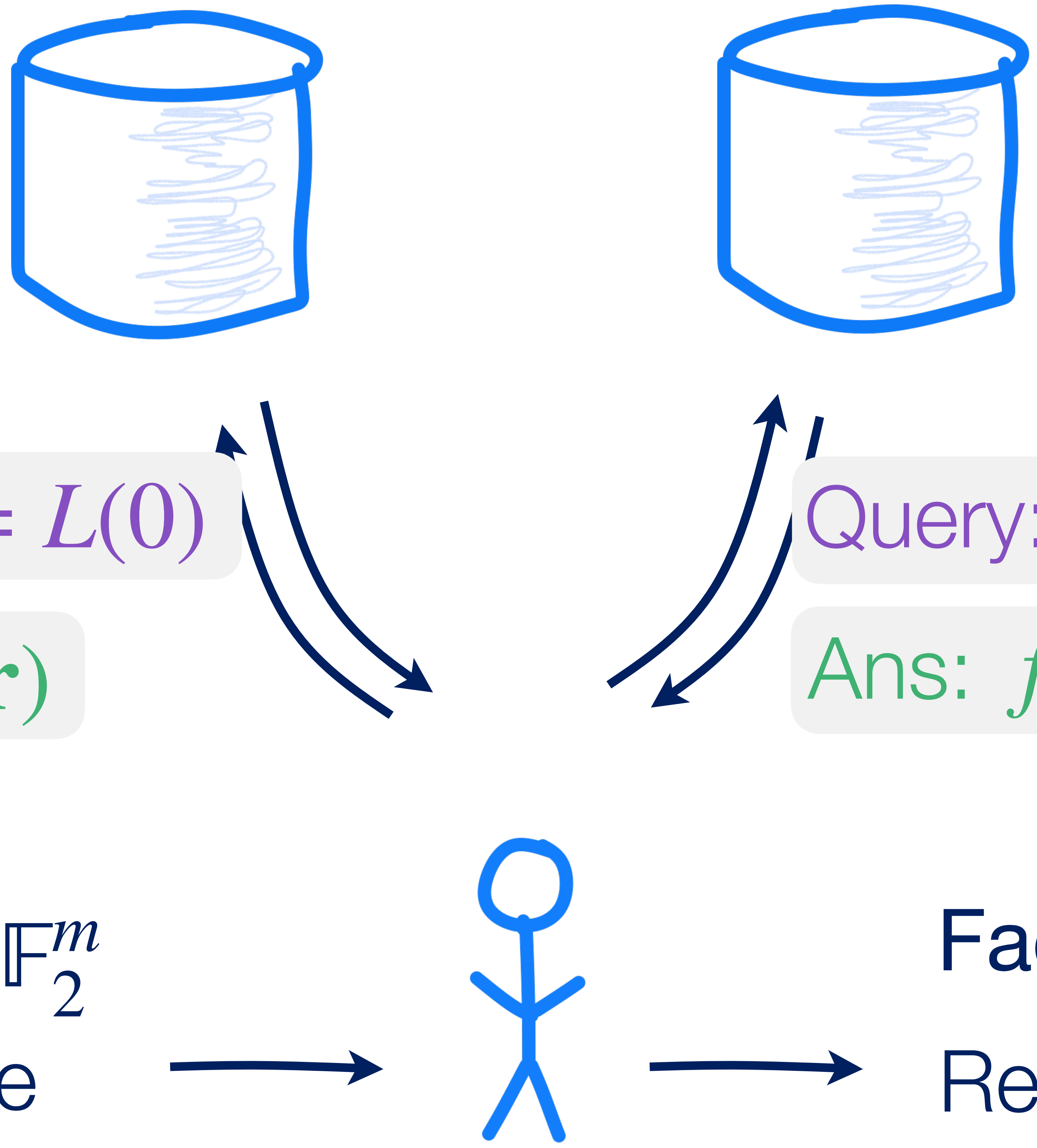
Point $\mathbf{p} \in \mathbb{F}_2^m$

Sample line

$\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

PIR from private polynomial evaluation

Scheme 1a: from Lagrange Interpolation [Sha79]



Homogenous degree-D

$f_{\text{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$

Query: $\mathbf{r} = L(0)$

Ans: $f_{\text{DB}}(\mathbf{r})$

Query: $\mathbf{p} + \mathbf{r} = L(1)$

Ans: $f_{\text{DB}}(\mathbf{p} + \mathbf{r})$

Point $\mathbf{p} \in \mathbb{F}_2^m$

Sample line

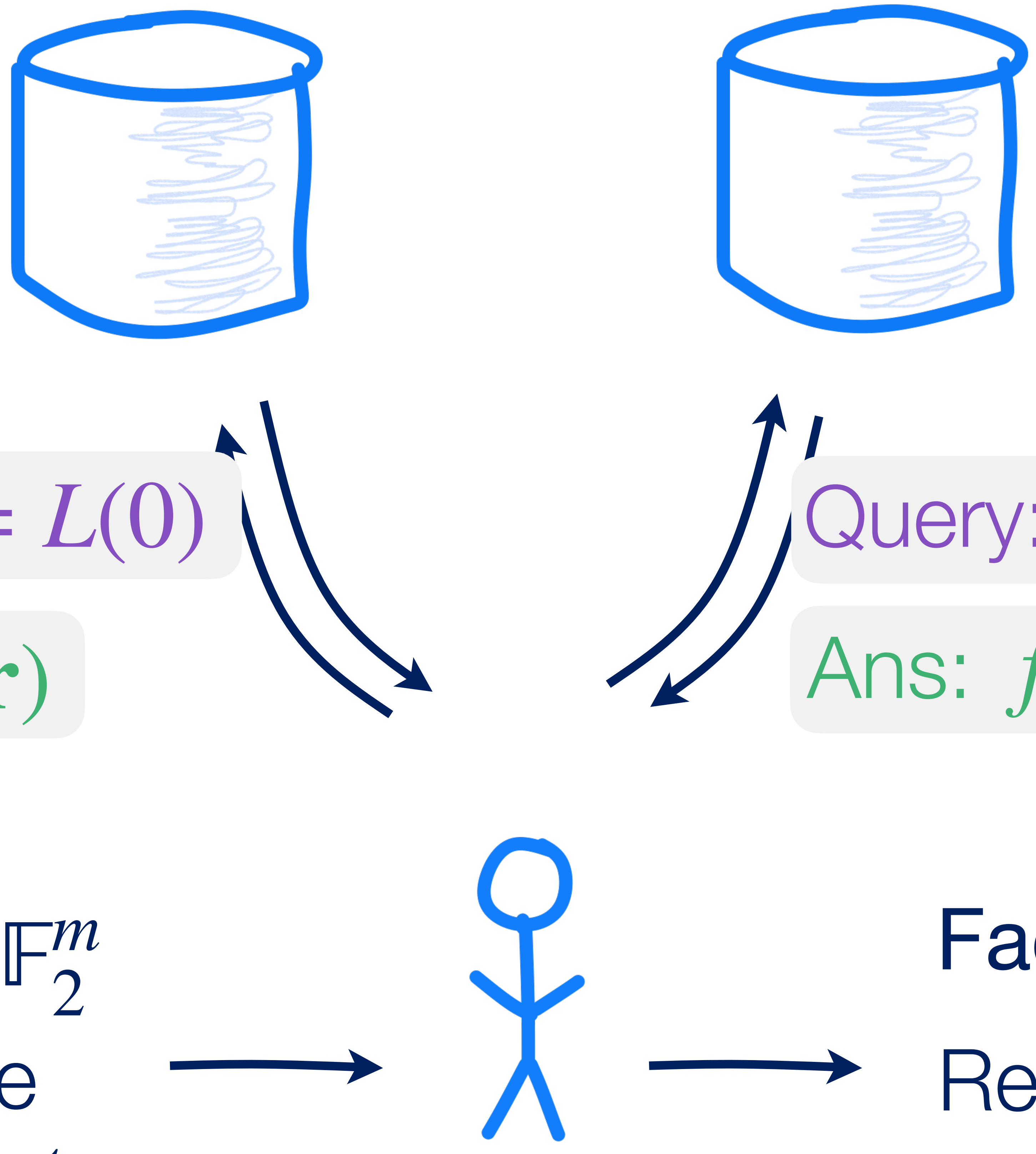
$\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

Fact: leading coefficient of $f_{\text{DB}}(\mathbf{L}(t))$ is $f_{\text{DB}}(\mathbf{p})$

Recover $f_{\text{DB}} \circ \mathbf{L}$ from its evaluations at $D + 1$ points, via Lagrange interpolation

PIR from private polynomial evaluation

Scheme 1a: from Lagrange Interpolation [Sha79]



Homogenous degree-D

$f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$

Query: $\mathbf{r} = L(0)$

Ans: $f_{\mathsf{DB}}(\mathbf{r})$

Query: $\mathbf{p} + \mathbf{r} = L(1)$

Ans: $f_{\mathsf{DB}}(\mathbf{p} + \mathbf{r})$

Point $\mathbf{p} \in \mathbb{F}_2^m$

Sample line $\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

Fact: leading coefficient of $f_{\mathsf{DB}}(\mathbf{L}(t))$ is $f_{\mathsf{DB}}(\mathbf{p})$

Recover $f_{\mathsf{DB}} \circ \mathbf{L}$ from its evaluations at $D + 1$ points, via Lagrange interpolation

With 2 servers, forces $D = 1$!

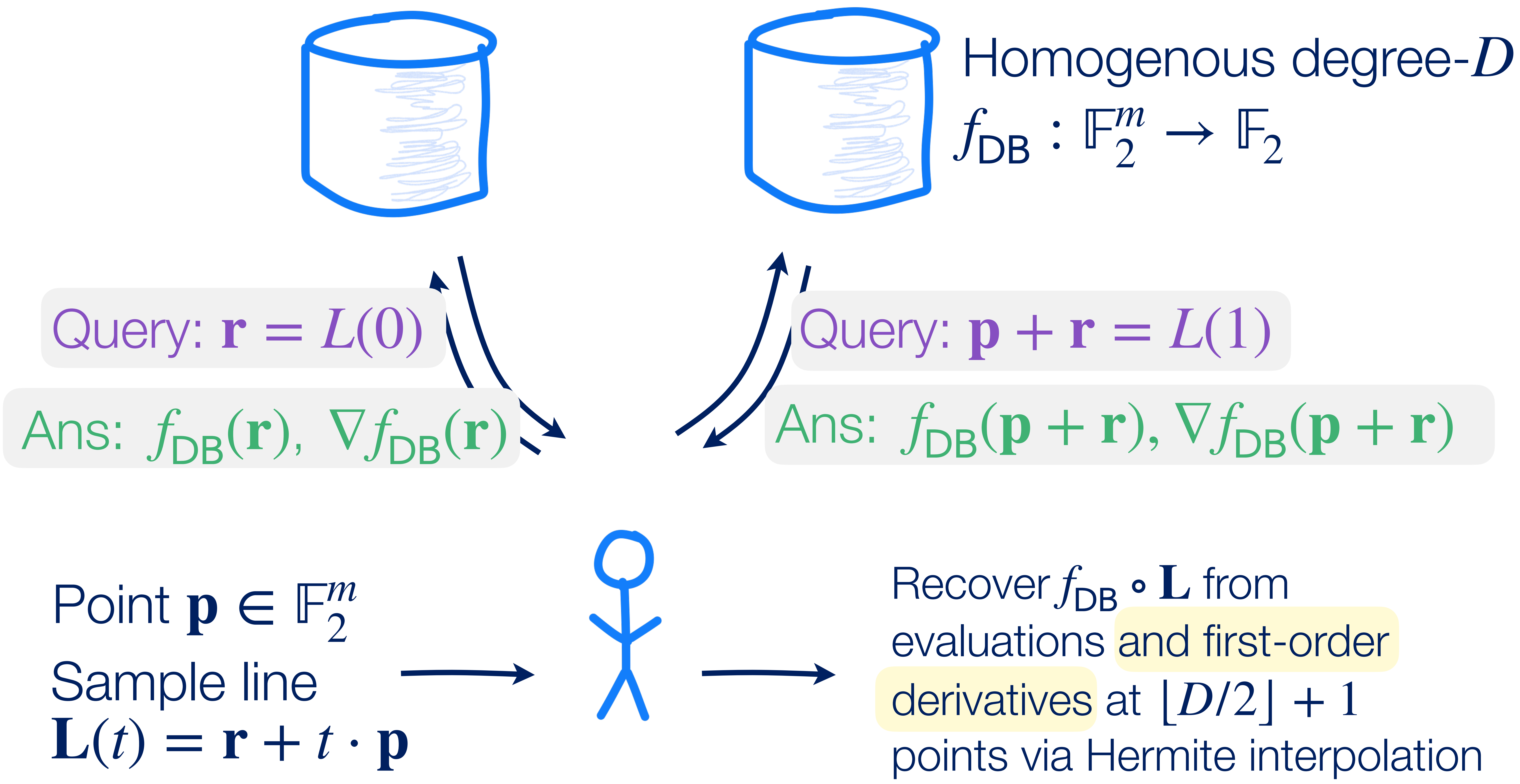
→ gives "trivial" PIR with

➡ $\binom{m}{D} \geq n \implies m = n$

➡ upload n and download 1

PIR from private polynomial evaluation

Scheme 1b: add derivatives [WY05]



Homogenous degree-D $f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$

Query: $\mathbf{r} = L(0)$

Ans: $f_{\mathsf{DB}}(\mathbf{r}),\ \nabla f_{\mathsf{DB}}(\mathbf{r})$

Query: $\mathbf{p} + \mathbf{r} = L(1)$

Ans: $f_{\mathsf{DB}}(\mathbf{p} + \mathbf{r}), \nabla f_{\mathsf{DB}}(\mathbf{p} + \mathbf{r})$

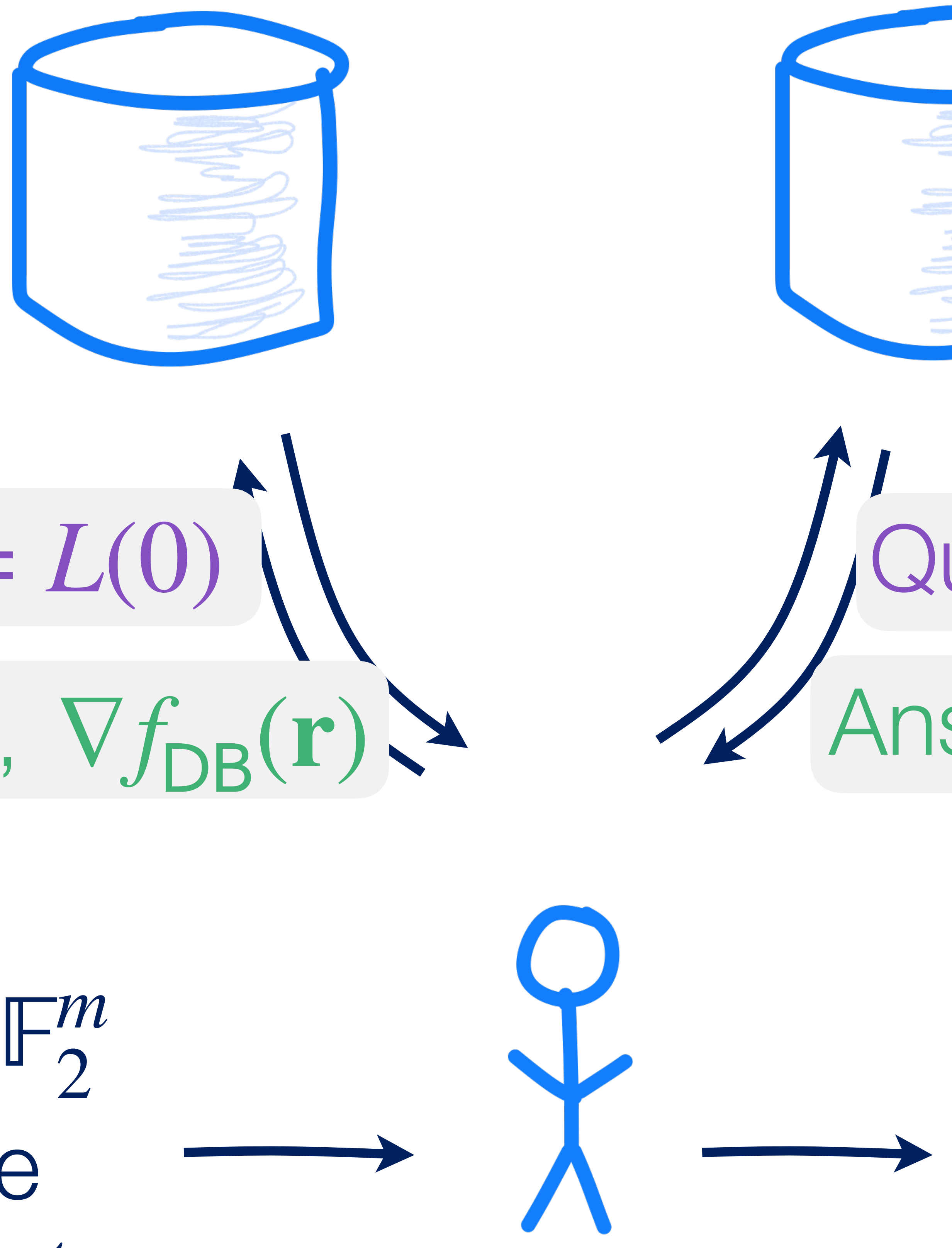
Point $\mathbf{p} \in \mathbb{F}_2^m$

Sample line $\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

Recover $f_{\mathsf{DB}} \circ \mathbf{L}$ from evaluations and first-order derivatives at $\lfloor D/2 \rfloor + 1$ points via Hermite interpolation

PIR from private polynomial evaluation

Scheme 1b: add derivatives [WY05]



Homogenous degree-D

$f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$

Query: $\mathbf{r} = L(0)$

Ans: $f_{\mathsf{DB}}(\mathbf{r}), \nabla f_{\mathsf{DB}}(\mathbf{r})$

Query: $\mathbf{p} + \mathbf{r} = L(1)$

Ans: $f_{\mathsf{DB}}(\mathbf{p} + \mathbf{r}), \nabla f_{\mathsf{DB}}(\mathbf{p} + \mathbf{r})$

Point $\mathbf{p} \in \mathbb{F}_2^m$

Sample line

$\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

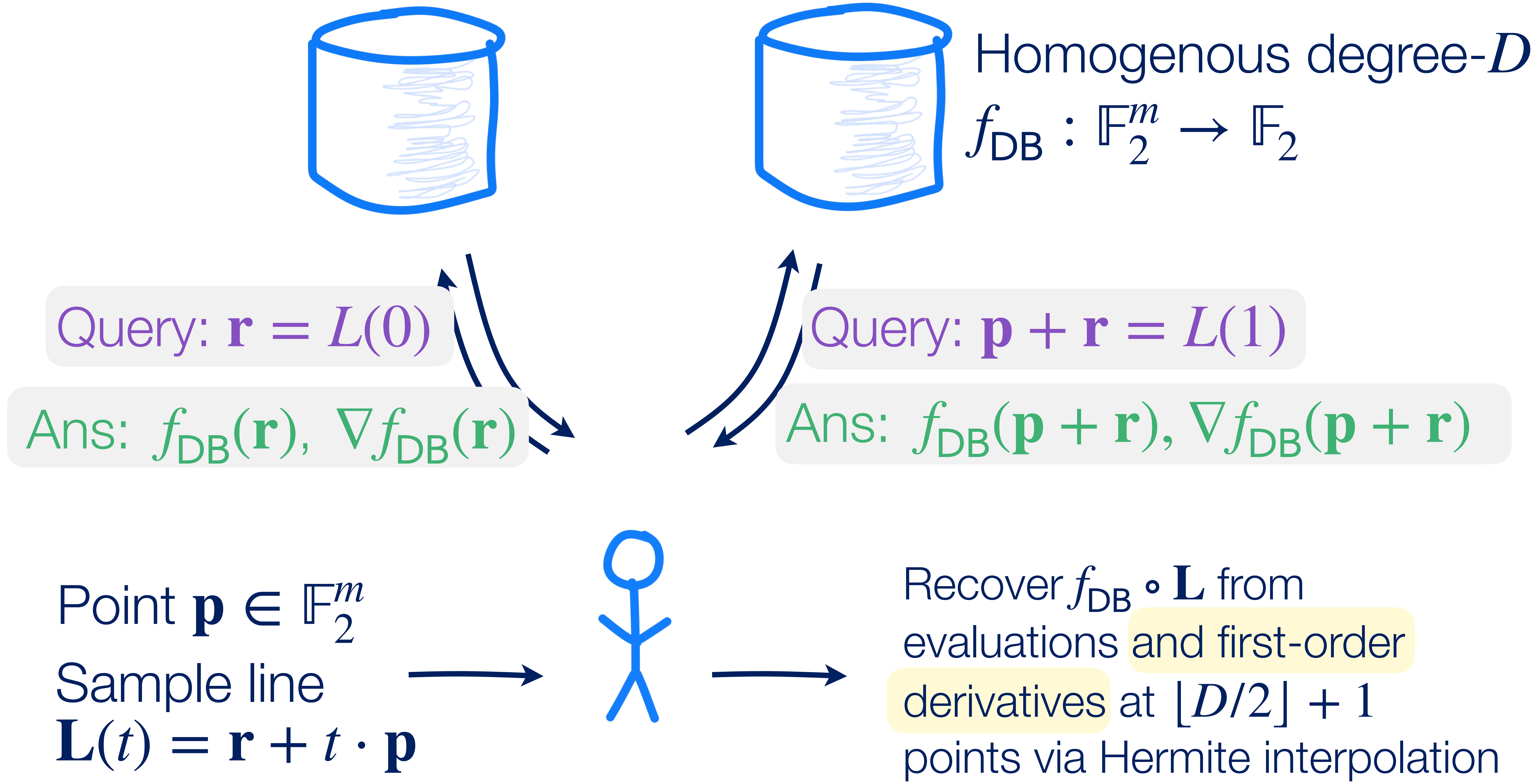
Recover $f_{\mathsf{DB}} \circ \mathbf{L}$ from evaluations and first-order derivatives at $\lfloor D/2 \rfloor + 1$ points via Hermite interpolation

Warning! Servers can't know the slope of $\mathbf{L}$ so they can't send derivatives of $f_{\mathsf{DB}} \circ \mathbf{L}$

- Instead, they send all first-order partial derivatives of f_{DB}
- Client will recover derivatives of $f_{\mathsf{DB}} \circ \mathbf{L}$ using chain rule

PIR from private polynomial evaluation

Scheme 1b: add derivatives [WY05]



Homogenous degree-D $f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$

Query: $\mathbf{r} = L(0)$

Ans: $f_{\mathsf{DB}}(\mathbf{r}),\ \nabla f_{\mathsf{DB}}(\mathbf{r})$

Query: $\mathbf{p} + \mathbf{r} = L(1)$

Ans: $f_{\mathsf{DB}}(\mathbf{p} + \mathbf{r}), \nabla f_{\mathsf{DB}}(\mathbf{p} + \mathbf{r})$

Point $\mathbf{p} \in \mathbb{F}_2^m$

Sample line

$\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

Recover $f_{\mathsf{DB}} \circ \mathbf{L}$ from evaluations and first-order derivatives at $\lfloor D/2 \rfloor + 1$ points via Hermite interpolation

With 2 servers, gives "balanced" PIR with

➡ $D = 3$

➡ $\binom{m}{D} \geq n \implies m = n^{1/3}$

➡ upload $n^{1/3}$ and download $n^{1/3}$

Warning! Servers can't know the slope of $\mathbf{L}$ so they can't send derivatives of $f_{\mathsf{DB}} \circ \mathbf{L}$

- Instead, they send all first-order partial derivatives of f_{DB}
- Client will recover derivatives of $f_{\mathsf{DB}} \circ \mathbf{L}$ using chain rule

PIR from private polynomial evaluation

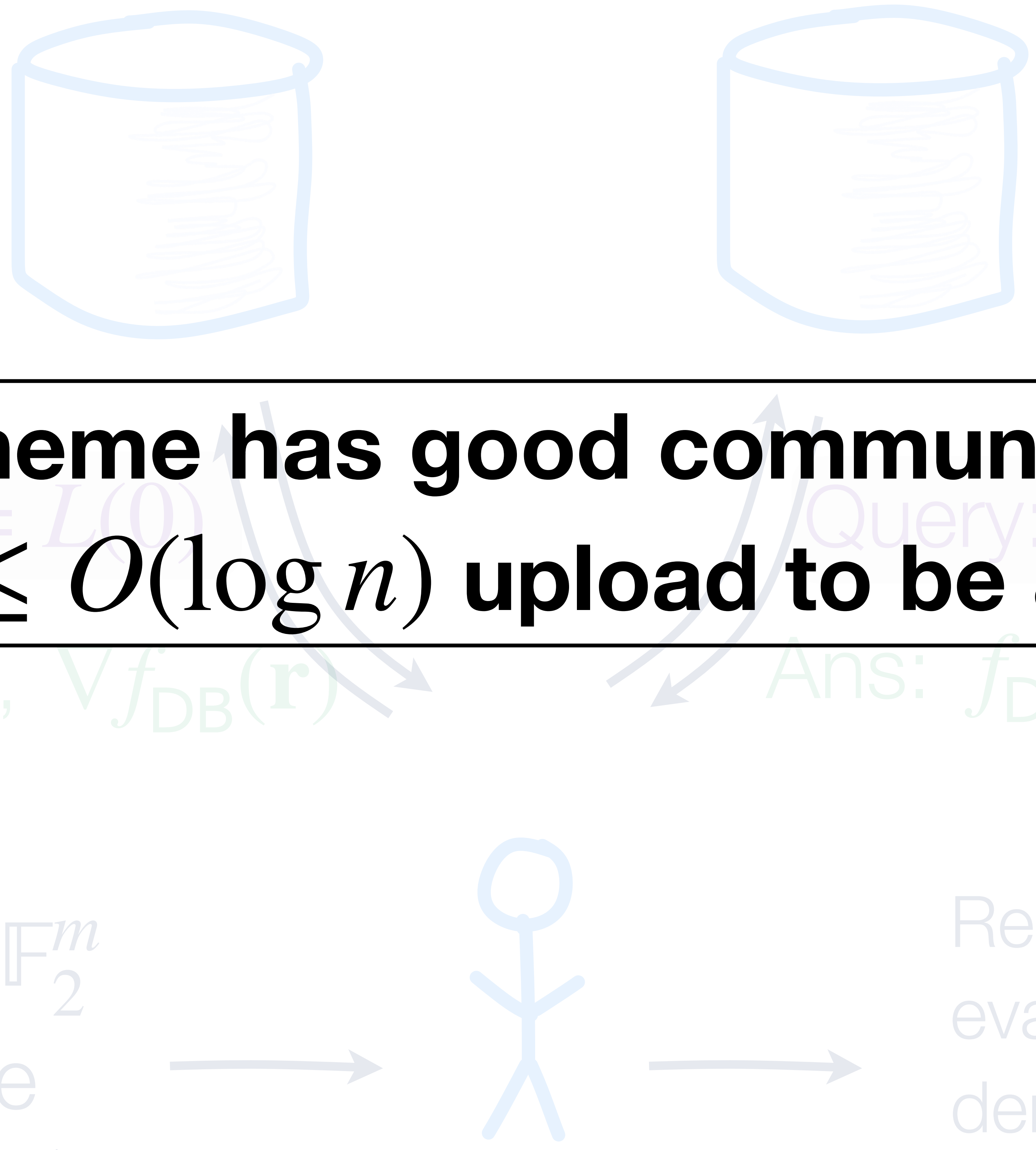
Scheme 1b: add derivatives [WY05]

With 2 servers, gives "balanced" PIR with

➡ $D = 3$

➡ $\binom{m}{D} \geq n \implies m = n^{1/3}$

➡ upload $n^{1/3}$ and download $n^{1/3}$



Homogenous degree-D $f_{\text{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$

Query: $\mathbf{r} = L(0)$

Ans: $f_{\text{DB}}(\mathbf{r})$, $\nabla f_{\text{DB}}(\mathbf{r})$

Query: $\mathbf{p} + \mathbf{r} = L(1)$

Ans: $f_{\text{DB}}(\mathbf{p}+\mathbf{r})$, $\nabla f_{\text{DB}}(\mathbf{p}+\mathbf{r})$

Point $\mathbf{p} \in \mathbb{F}_2^m$

Sample line $\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

Recover $f_{\text{DB}} \circ \mathbf{L}$ from evaluations and first-order derivatives at $\lfloor D/2 \rfloor + 1$ points via Hermite interpolation

Warning! Servers can't know the slope of $\mathbf{L}$ so they can't send derivatives of $f_{\text{DB}} \circ \mathbf{L}$

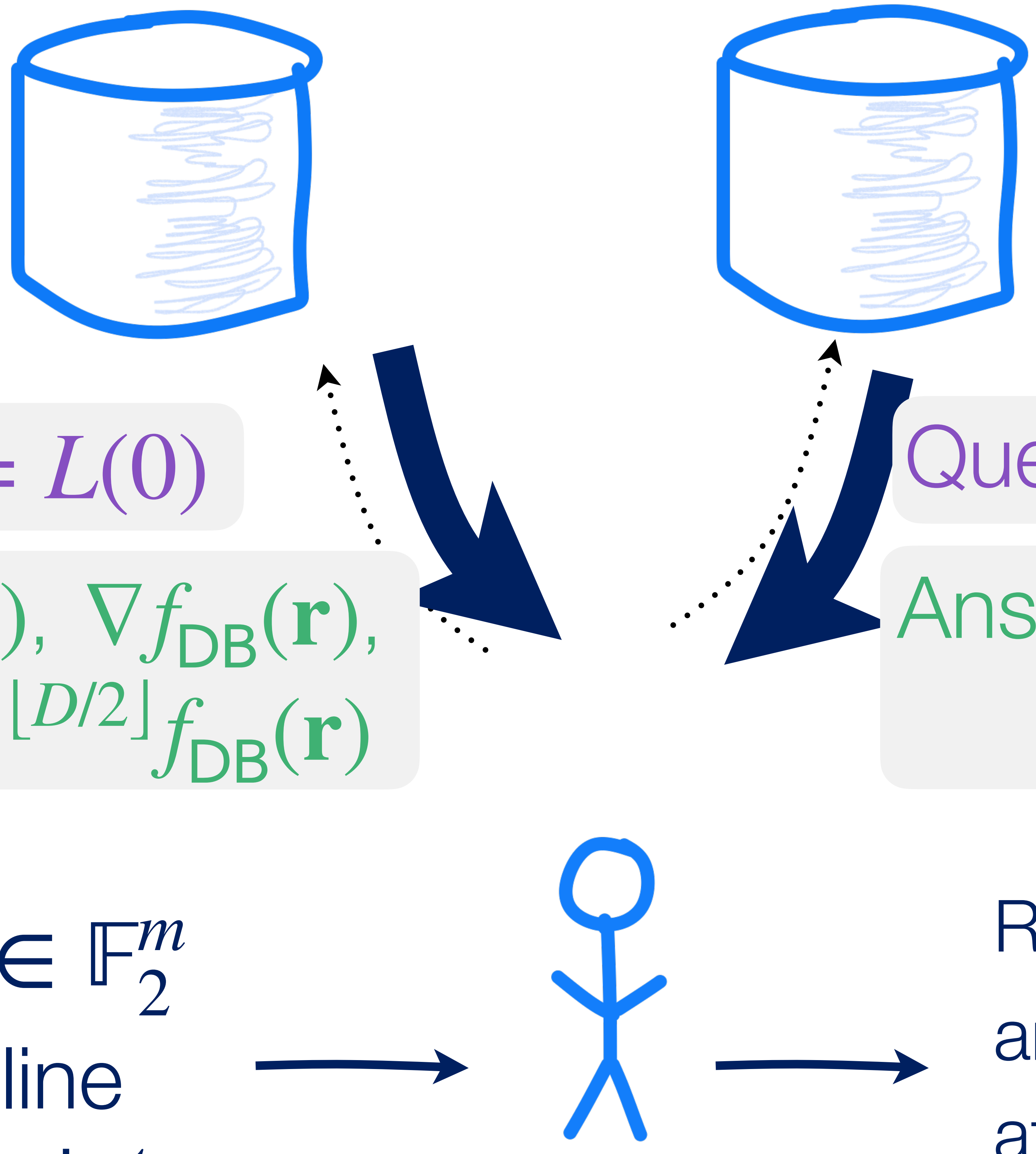
- Instead, they send all first-order partial derivatives of f_{DB}
- Client will recover derivatives of $f_{\text{DB}} \circ \mathbf{L}$ using chain rule

This scheme has good communication but it is not preprocessing-friendly!

Need $\leq O(\log n)$ upload to be able to precompute answers to all queries

PIR from private polynomial evaluation

Make it “imbalanced”: more derivatives [BIM00,GLM+25]



Homogenous degree-D

$f_{\text{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$

Query: $\mathbf{r} = L(0)$

Ans: $f_{\text{DB}}(\mathbf{r}),\ \nabla f_{\text{DB}}(\mathbf{r}),\ \ldots,\ \nabla^{\lfloor D/2 \rfloor} f_{\text{DB}}(\mathbf{r})$

Query: $\mathbf{p} + \mathbf{r} = L(1)$

Ans: $f_{\text{DB}}(\mathbf{p}+\mathbf{r}),\ \nabla f_{\text{DB}}(\mathbf{p}+\mathbf{r}),\ \ldots,\ \nabla^{\lfloor D/2 \rfloor} f_{\text{DB}}(\mathbf{p}+\mathbf{r})$

Point $\mathbf{p} \in \mathbb{F}_2^m$

Sample line

$\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

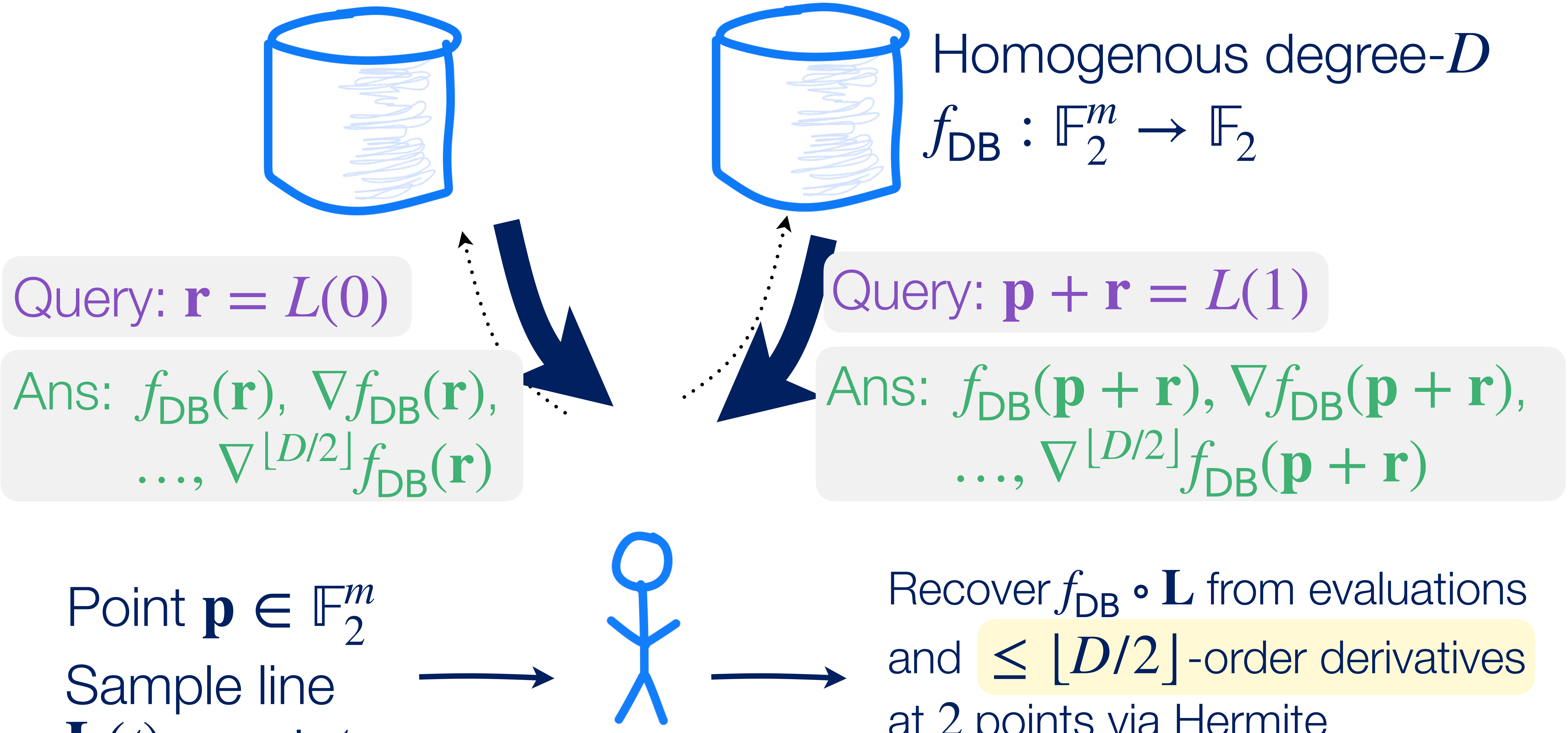
Recover $f_{\text{DB}} \circ \mathbf{L}$ from evaluations and $\leq \lfloor D/2 \rfloor$-order derivatives at 2 points via Hermite interpolation

Warning! Servers can’t know the slope of $\mathbf{L}$ so they can’t send derivatives of $f_{\text{DB}} \circ \mathbf{L}$

- Instead, they send all partial derivatives of f_{DB} to order $\leq \lfloor D/2 \rfloor$
- Client will recover derivatives of $f_{\text{DB}} \circ \mathbf{L}$ using chain rule

PIR from private polynomial evaluation

Make it “imbalanced”: more derivatives [BIM00,GLM+25]



Homogenous degree-D $f_{\text{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$

Query: $\mathbf{r} = L(0)$

Ans: $f_{\text{DB}}(\mathbf{r})$, $\nabla f_{\text{DB}}(\mathbf{r})$, $\ldots, \nabla^{\lfloor D/2 \rfloor} f_{\text{DB}}(\mathbf{r})$

Query: $\mathbf{p} + \mathbf{r} = L(1)$

Ans: $f_{\text{DB}}(\mathbf{p}+\mathbf{r})$, $\nabla f_{\text{DB}}(\mathbf{p}+\mathbf{r})$, $\ldots, \nabla^{\lfloor D/2 \rfloor} f_{\text{DB}}(\mathbf{p}+\mathbf{r})$

Point $\mathbf{p} \in \mathbb{F}_2^m$

Sample line

$\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

Recover $f_{\text{DB}} \circ \mathbf{L}$ from evaluations and $\leq \lfloor D/2 \rfloor$-order derivatives at 2 points via Hermite interpolation

With 2 servers, gives “imbalanced” PIR with

➡ $m = (1 + o(1)) \cdot \log n$

➡ $D = m/2$

➡ upload $m \approx \log n$ and download $\binom{m}{\lfloor D/2 \rfloor} \approx n^{0.82}$

Warning! Servers can’t know the slope of $\mathbf{L}$ so they can’t send derivatives of $f_{\text{DB}} \circ \mathbf{L}$

- Instead, they send all partial derivatives of f_{DB} to order $\leq \lfloor D/2 \rfloor$
- Client will recover derivatives of $f_{\text{DB}} \circ \mathbf{L}$ using chain rule

Prior information-theoretic PIR

[BIM00,GLMDS25]

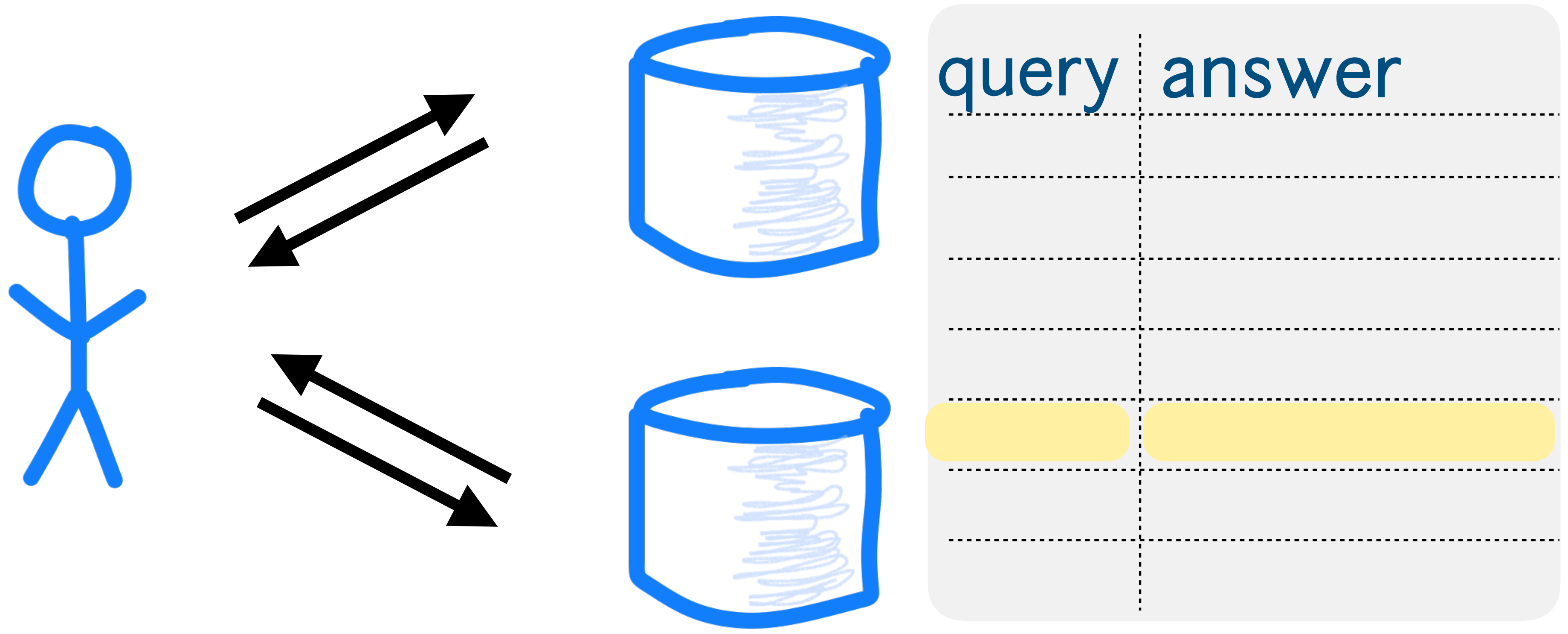
✓ 1. Build “imbalanced” PIR with tiny queries

- Query length = $(1 + o(1)) \cdot \log n$
- Answer length = $\ell = O(n^{0.82})$

2. Precompute the answer to every query

- To answer a query: read 1 location of length ℓ

➡ PIR in $n^{1.82+o(1)}$ space and $n^{0.82}$ time

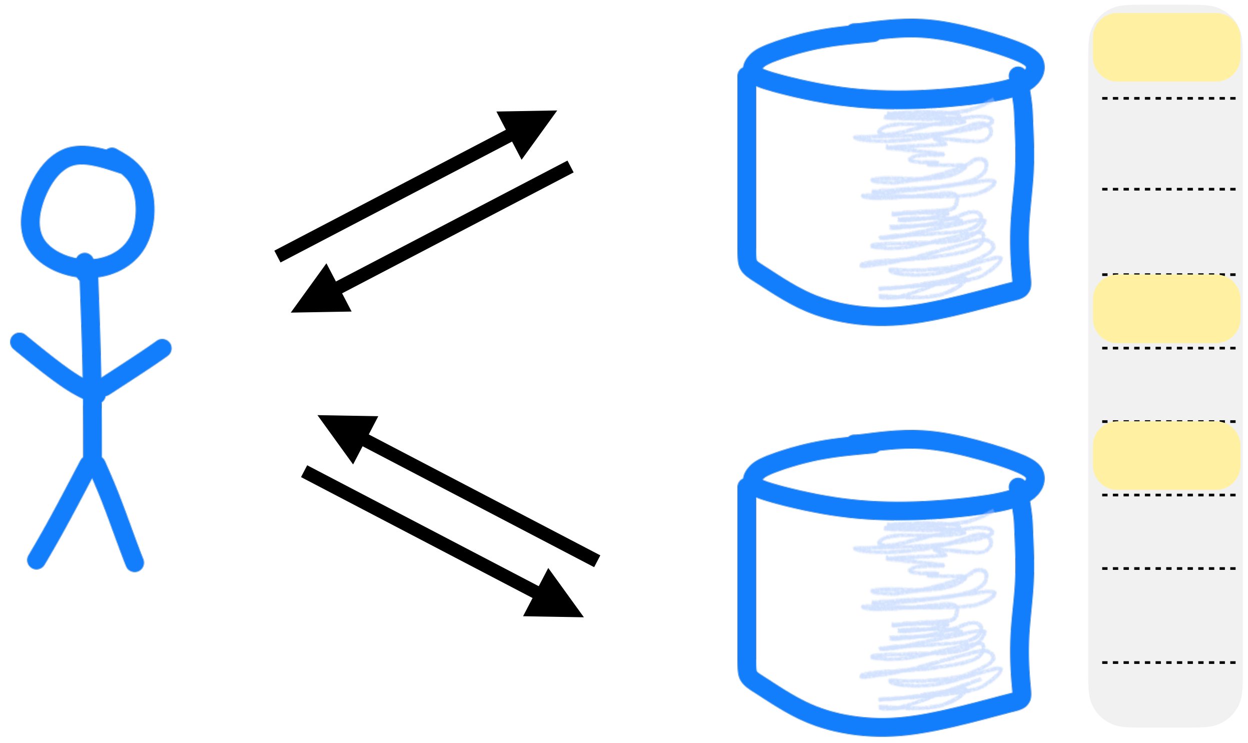


This work

2. New data structure

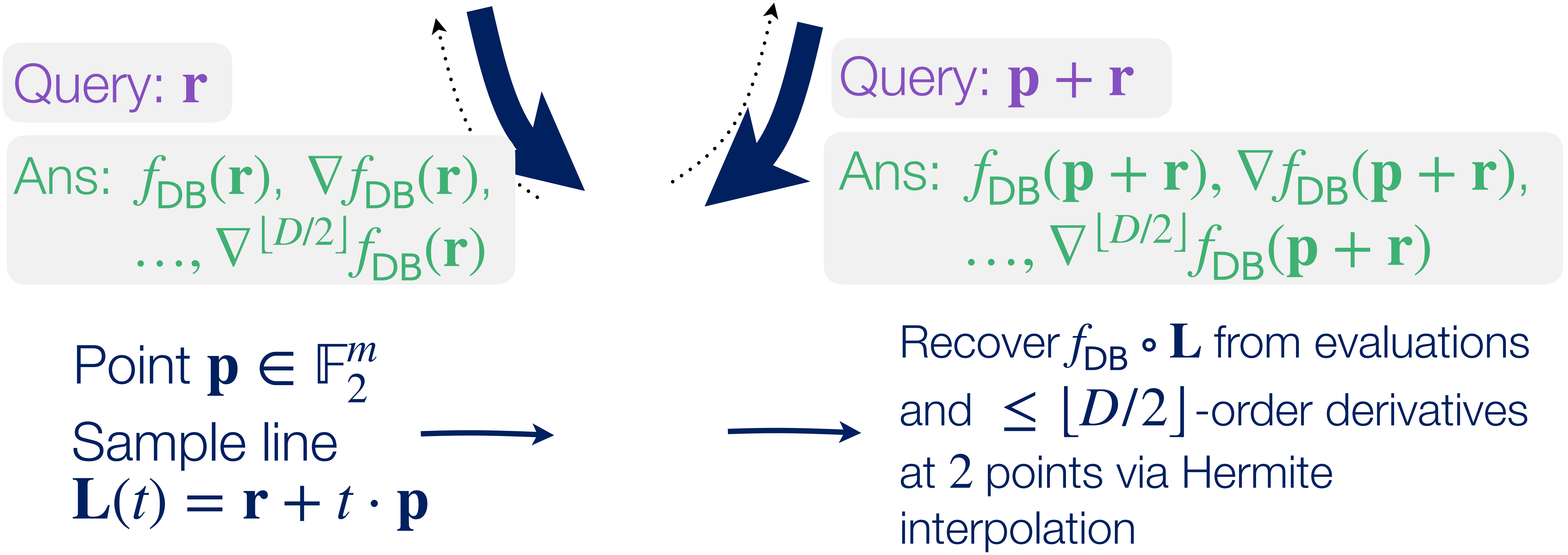
- To answer a query: read ℓ locations of length 1

➡ PIR in $n^{1+o(1)}$ space and $n^{0.82}$ time



PIR with Preprocessing

Prior work: Precompute every answer [BIM00,GLM+25]



Query: $\mathbf{r}$

Ans: $f_{\mathsf{DB}}(\mathbf{r}),\ \nabla f_{\mathsf{DB}}(\mathbf{r}),\ \ldots,\ \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{r})$

Query: $\mathbf{p} + \mathbf{r}$

Ans: $f_{\mathsf{DB}}(\mathbf{p} + \mathbf{r}), \nabla f_{\mathsf{DB}}(\mathbf{p} + \mathbf{r}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{p} + \mathbf{r})$

Point $\mathbf{p} \in \mathbb{F}_2^m$

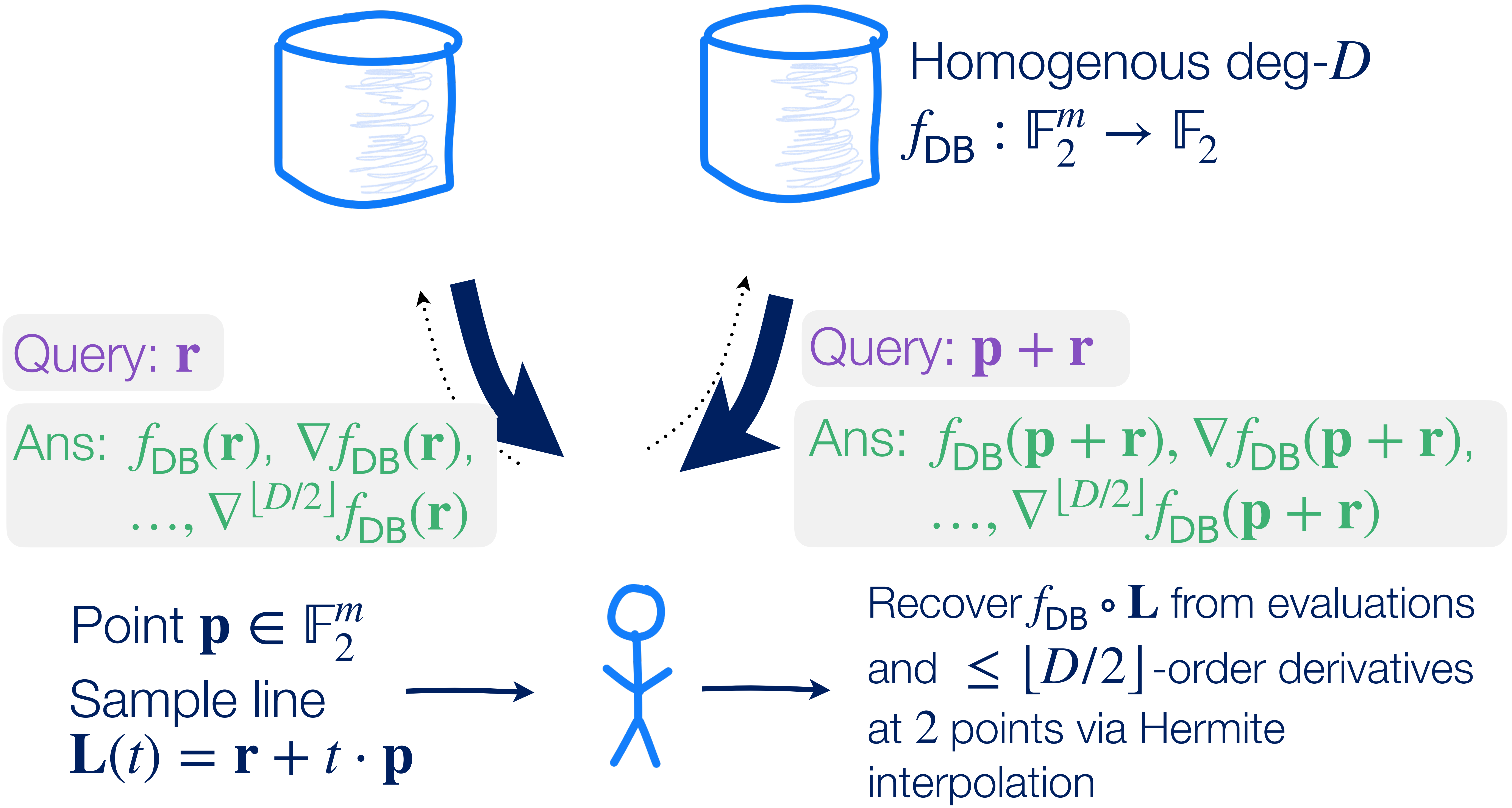
Sample line

$\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

Recover $f_{\mathsf{DB}} \circ \mathbf{L}$ from evaluations and $\leq \lfloor D/2 \rfloor$-order derivatives at 2 points via Hermite interpolation

PIR with Preprocessing

Prior work: Precompute every answer [BIM00,GLM+25]



Homogenous deg-D

$f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$

Query: $\mathbf{r}$

Ans: $f_{\mathsf{DB}}(\mathbf{r}),\ \nabla f_{\mathsf{DB}}(\mathbf{r}),\ \ldots,\ \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{r})$

Query: $\mathbf{p} + \mathbf{r}$

Ans: $f_{\mathsf{DB}}(\mathbf{p} + \mathbf{r}),\ \nabla f_{\mathsf{DB}}(\mathbf{p} + \mathbf{r}),\ \ldots,\ \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{p} + \mathbf{r})$

Point $\mathbf{p} \in \mathbb{F}_2^m$

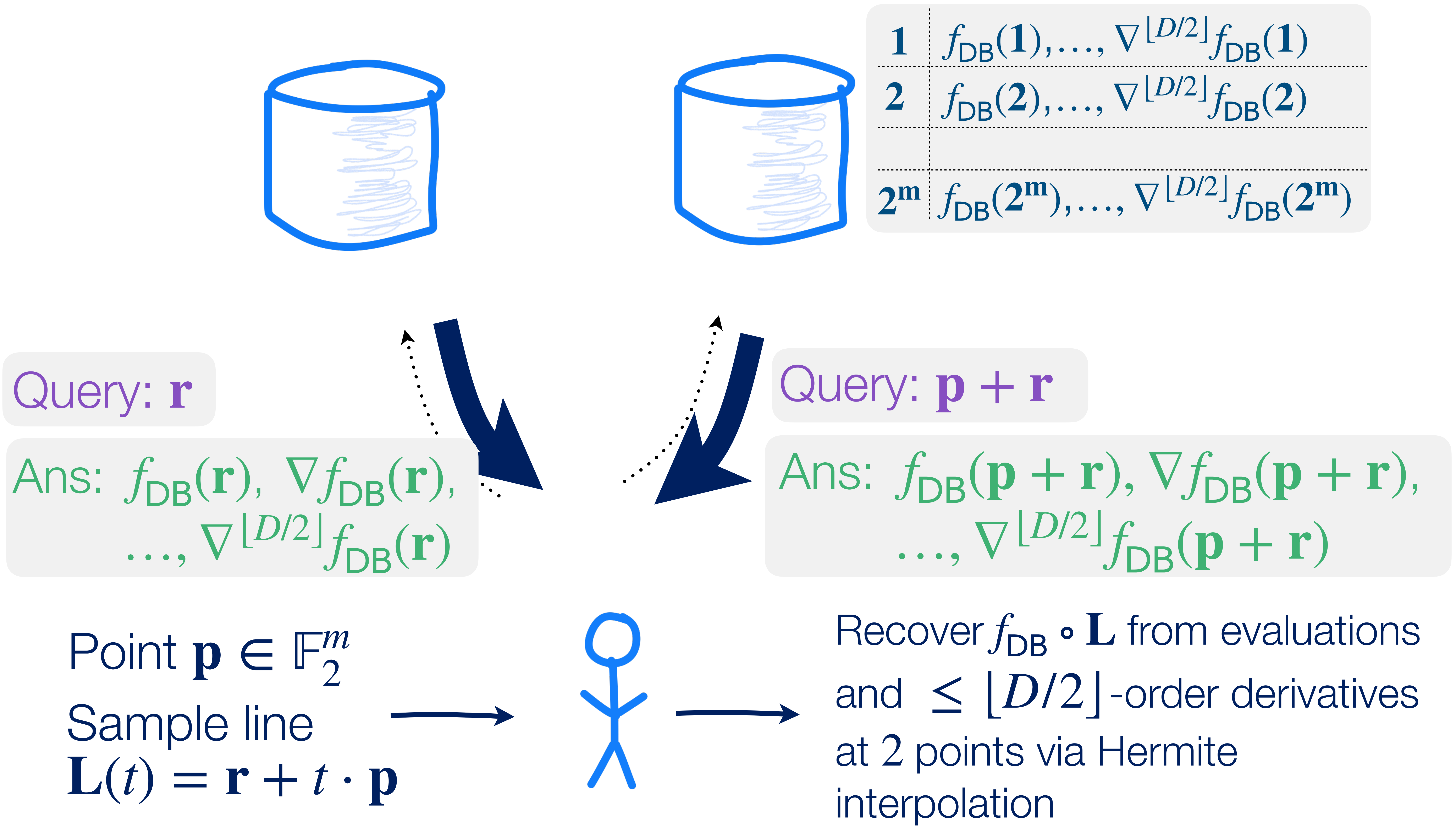
Sample line

$\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

Recover $f_{\mathsf{DB}} \circ \mathbf{L}$ from evaluations and $\leq \lfloor D/2 \rfloor$-order derivatives at 2 points via Hermite interpolation

PIR with Preprocessing

Prior work: Precompute every answer [BIM00,GLM+25]



$\mathbf{1}$	$f_{\mathsf{DB}}(\mathbf{1}),\ldots,\nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{1})$
$\mathbf{2}$	$f_{\mathsf{DB}}(\mathbf{2}),\ldots,\nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{2})$
$\mathbf{2^m}$	$f_{\mathsf{DB}}(\mathbf{2^m}),\ldots,\nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{2^m})$

Query: $\mathbf{r}$

Ans: $f_{\mathsf{DB}}(\mathbf{r}),\ \nabla f_{\mathsf{DB}}(\mathbf{r}),\ \ldots,\ \nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{r})$

Query: $\mathbf{p}+\mathbf{r}$

Ans: $f_{\mathsf{DB}}(\mathbf{p}+\mathbf{r}),\ \nabla f_{\mathsf{DB}}(\mathbf{p}+\mathbf{r}),\ \ldots,\ \nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{p}+\mathbf{r})$

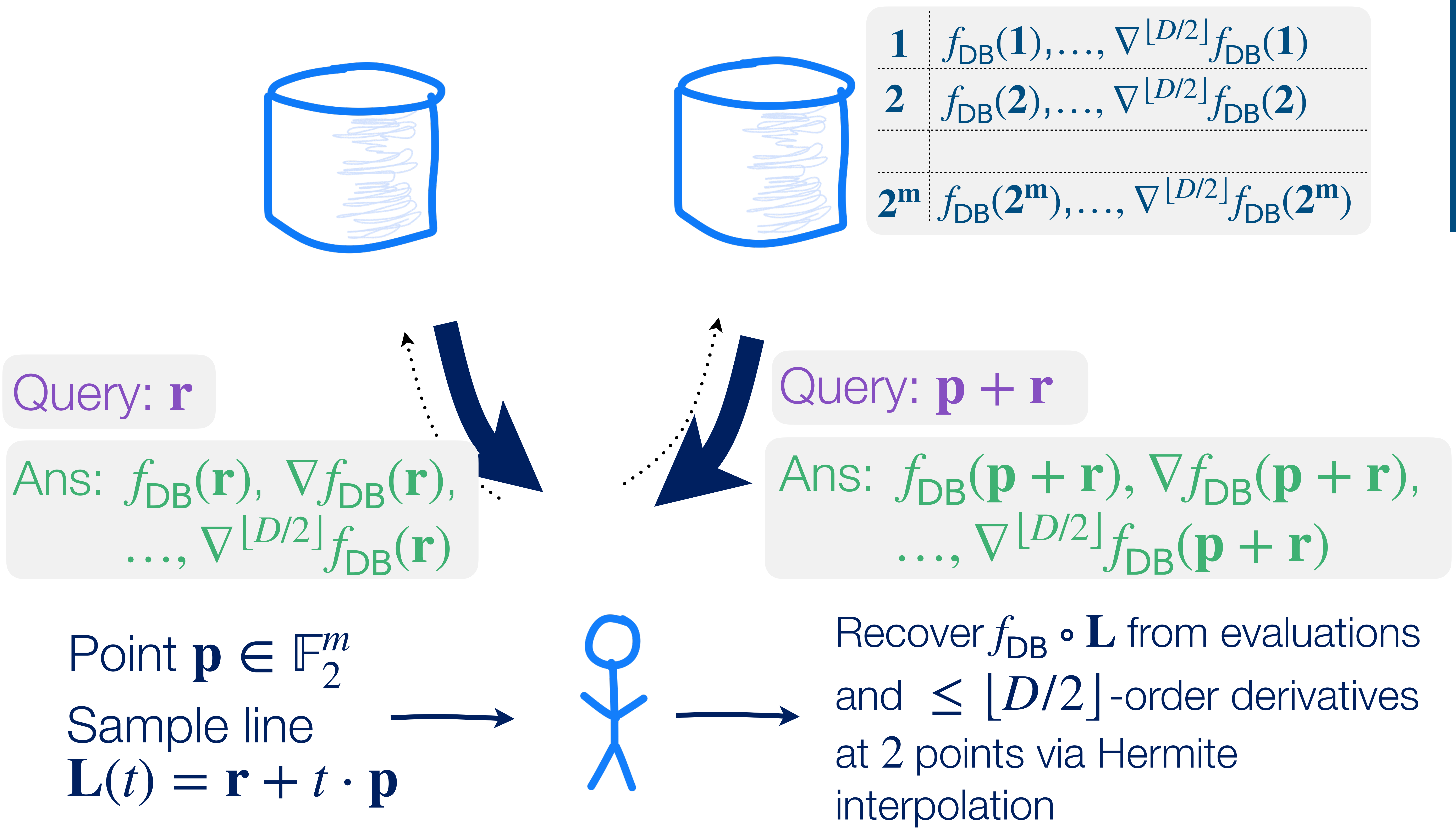
Point $\mathbf{p}\in\mathbb{F}_2^m$

Sample line $\mathbf{L}(t)=\mathbf{r}+t\cdot\mathbf{p}$

Recover $f_{\mathsf{DB}}\circ\mathbf{L}$ from evaluations and $\leq\lfloor D/2\rfloor$-order derivatives at 2 points via Hermite interpolation

PIR with Preprocessing

Prior work: Precompute every answer [BIM00,GLM+25]



$\mathbf{1}$	$f_{\mathsf{DB}}(\mathbf{1}),\ldots,\nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{1})$
$\mathbf{2}$	$f_{\mathsf{DB}}(\mathbf{2}),\ldots,\nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{2})$
$\mathbf{2^m}$	$f_{\mathsf{DB}}(\mathbf{2^m}),\ldots,\nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{2^m})$

Query: $\mathbf{r}$

Ans: $f_{\mathsf{DB}}(\mathbf{r}),\ \nabla f_{\mathsf{DB}}(\mathbf{r}),\ \ldots,\ \nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{r})$

Query: $\mathbf{p}+\mathbf{r}$

Ans: $f_{\mathsf{DB}}(\mathbf{p}+\mathbf{r}), \nabla f_{\mathsf{DB}}(\mathbf{p}+\mathbf{r}), \ldots, \nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{p}+\mathbf{r})$

Point $\mathbf{p}\in\mathbb{F}_2^m$

Sample line $\mathbf{L}(t)=\mathbf{r}+t\cdot\mathbf{p}$

Recover $f_{\mathsf{DB}}\circ\mathbf{L}$ from evaluations and $\leq\lfloor D/2\rfloor$-order derivatives at 2 points via Hermite interpolation

With 2 servers, gives preprocessing PIR with

- ➡ $O(\log n)$ upload and $n^{0.82}$ download
- ➡ $2^m\cdot n^{0.82}=n^{1.82+o(1)}$ server storage
- ➡ $n^{0.82}$ server time

Prior information-theoretic PIR

[BIM00,GLMDS25]

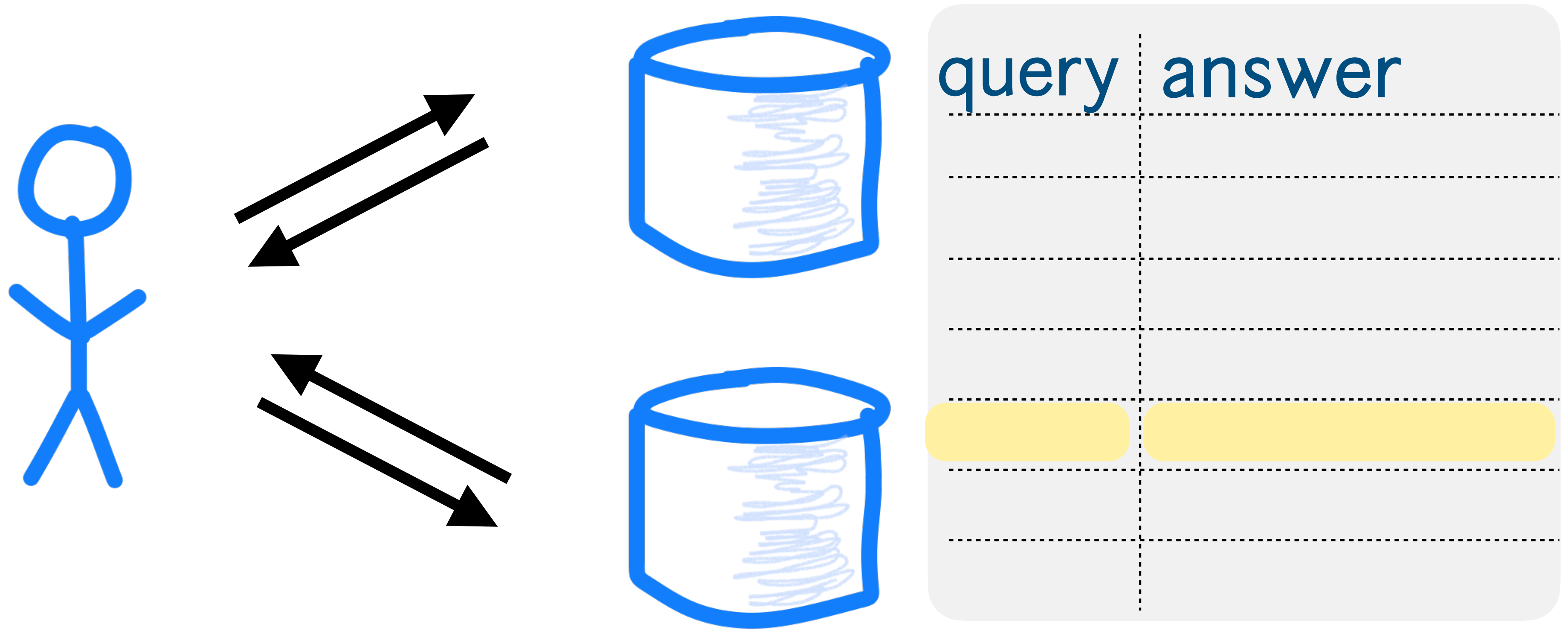
✓ 1. Build “imbalanced” PIR with tiny queries

- Query length = $(1 + o(1)) \cdot \log n$
- Answer length = $\ell = O(n^{0.82})$

✓ 2. Precompute the answer to every query

- To answer a query: read 1 location of length ℓ

➡ PIR in $n^{1.82+o(1)}$ space and $n^{0.82}$ time

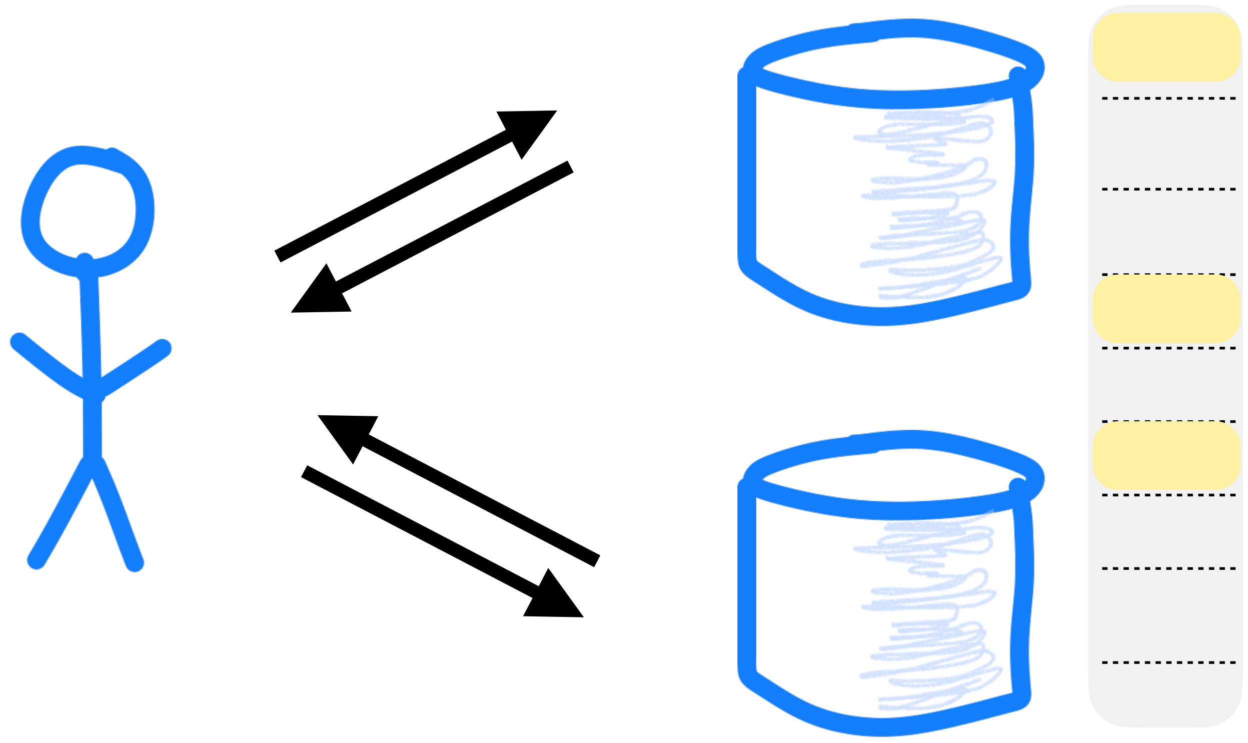


This work

2. New data structure

- To answer a query: read ℓ locations of length 1

➡ PIR in $n^{1+o(1)}$ space and $n^{0.82}$ time



This talk

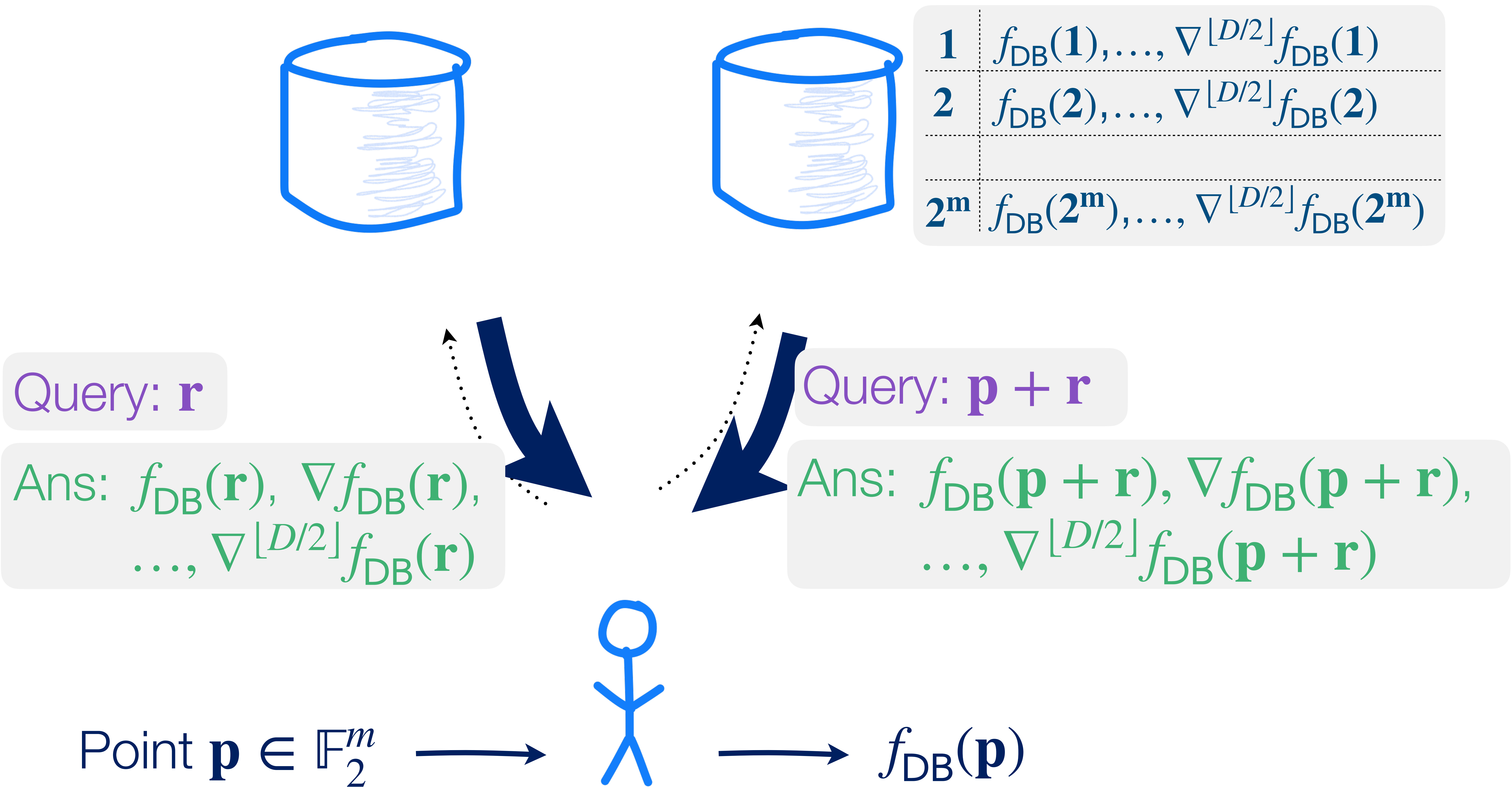
1. **Background:** PIR with preprocessing
2. **[HR26] New two-server PIR:** sublinear time, quasilinear space
3. **Evaluation:** what does this mean for practice?
4. **Bonuses** 😁
    - Reducing communication using crypto
    - **[HPR26]** Connecting multi-server PIR to complexity theory

This talk

1. **Background:** PIR with preprocessing
2. **[HR26] New two-server PIR:** sublinear time, quasilinear space
3. **Evaluation:** what does this mean for practice?
4. **Bonuses** 😁
    - Reducing communication using crypto
    - **[HPR26]** Connecting multi-server PIR to complexity theory

Prior work: Precompute every possible PIR answer

[BIM00,GLMDS25]



$\mathbf{1}$	$f_{\mathsf{DB}}(\mathbf{1}),\ldots,\nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{1})$
$\mathbf{2}$	$f_{\mathsf{DB}}(\mathbf{2}),\ldots,\nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{2})$
$\mathbf{2^m}$	$f_{\mathsf{DB}}(\mathbf{2^m}),\ldots,\nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{2^m})$

Query: $\mathbf{r}$

Ans: $f_{\mathsf{DB}}(\mathbf{r}),\ \nabla f_{\mathsf{DB}}(\mathbf{r}),\ \ldots,\ \nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{r})$

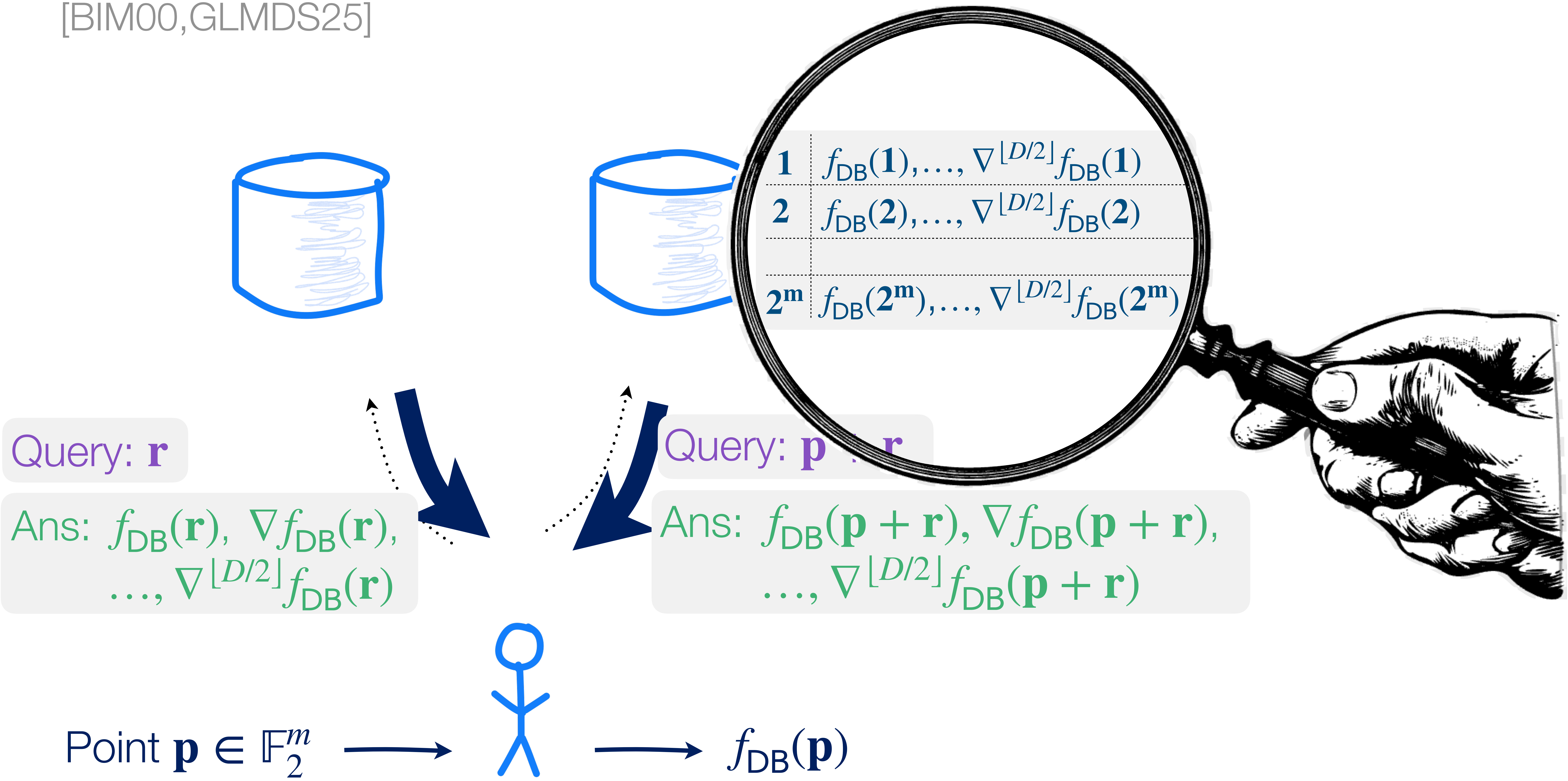
Query: $\mathbf{p}+\mathbf{r}$

Ans: $f_{\mathsf{DB}}(\mathbf{p}+\mathbf{r}),\ \nabla f_{\mathsf{DB}}(\mathbf{p}+\mathbf{r}),\ \ldots,\ \nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{p}+\mathbf{r})$

Point $\mathbf{p}\in\mathbb{F}_2^m \longrightarrow$ $\longrightarrow f_{\mathsf{DB}}(\mathbf{p})$

Prior work: Precompute every possible PIR answer

[BIM00,GLMDS25]



$\mathbf{1}$	$f_{\mathsf{DB}}(\mathbf{1}),\ldots,\nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{1})$
$\mathbf{2}$	$f_{\mathsf{DB}}(\mathbf{2}),\ldots,\nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{2})$
$\mathbf{2^m}$	$f_{\mathsf{DB}}(\mathbf{2^m}),\ldots,\nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{2^m})$

Query: $\mathbf{r}$

Ans: $f_{\mathsf{DB}}(\mathbf{r}),\ \nabla f_{\mathsf{DB}}(\mathbf{r}),\ \ldots,\ \nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{r})$

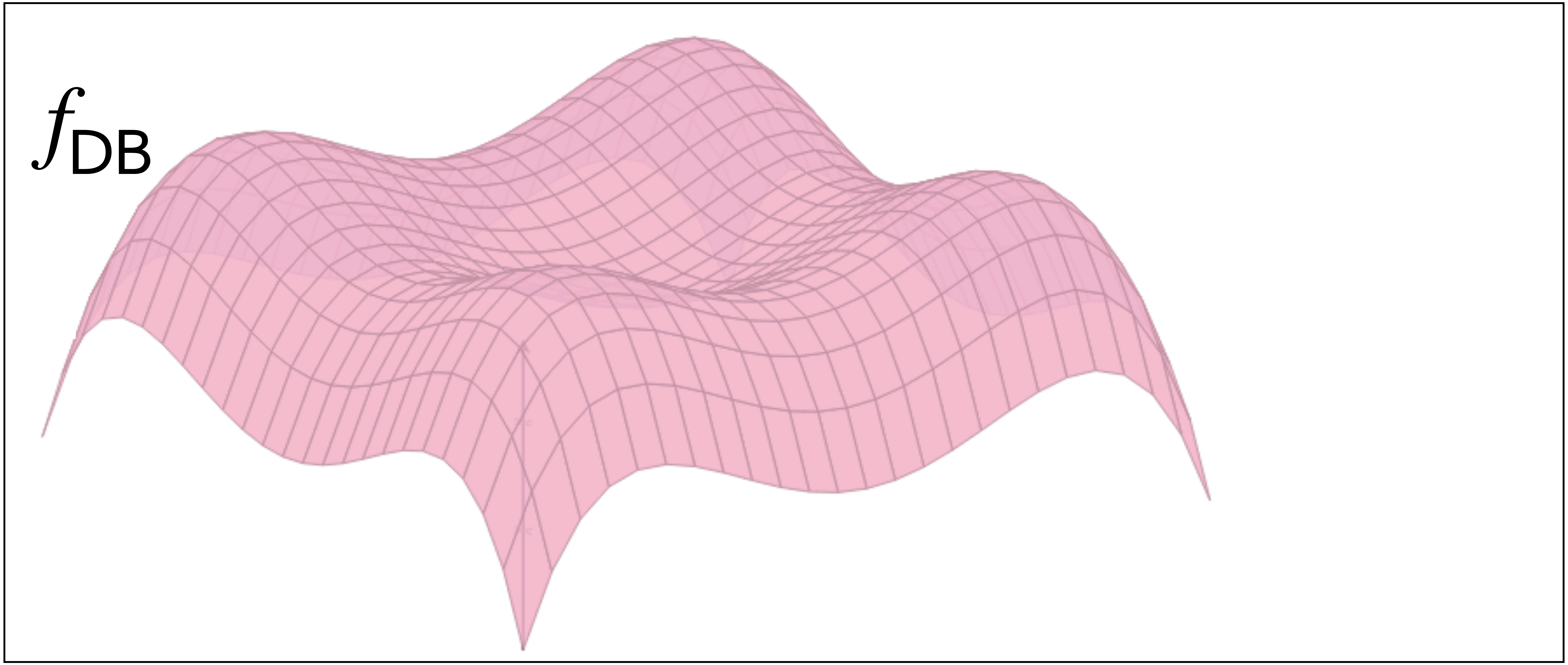
Query: $\mathbf{p} + \mathbf{r}$

Ans: $f_{\mathsf{DB}}(\mathbf{p}+\mathbf{r}), \nabla f_{\mathsf{DB}}(\mathbf{p}+\mathbf{r}),\ \ldots,\ \nabla^{\lfloor D/2\rfloor}f_{\mathsf{DB}}(\mathbf{p}+\mathbf{r})$

Point $\mathbf{p} \in \mathbb{F}_2^m \longrightarrow$ $\longrightarrow f_{\mathsf{DB}}(\mathbf{p})$

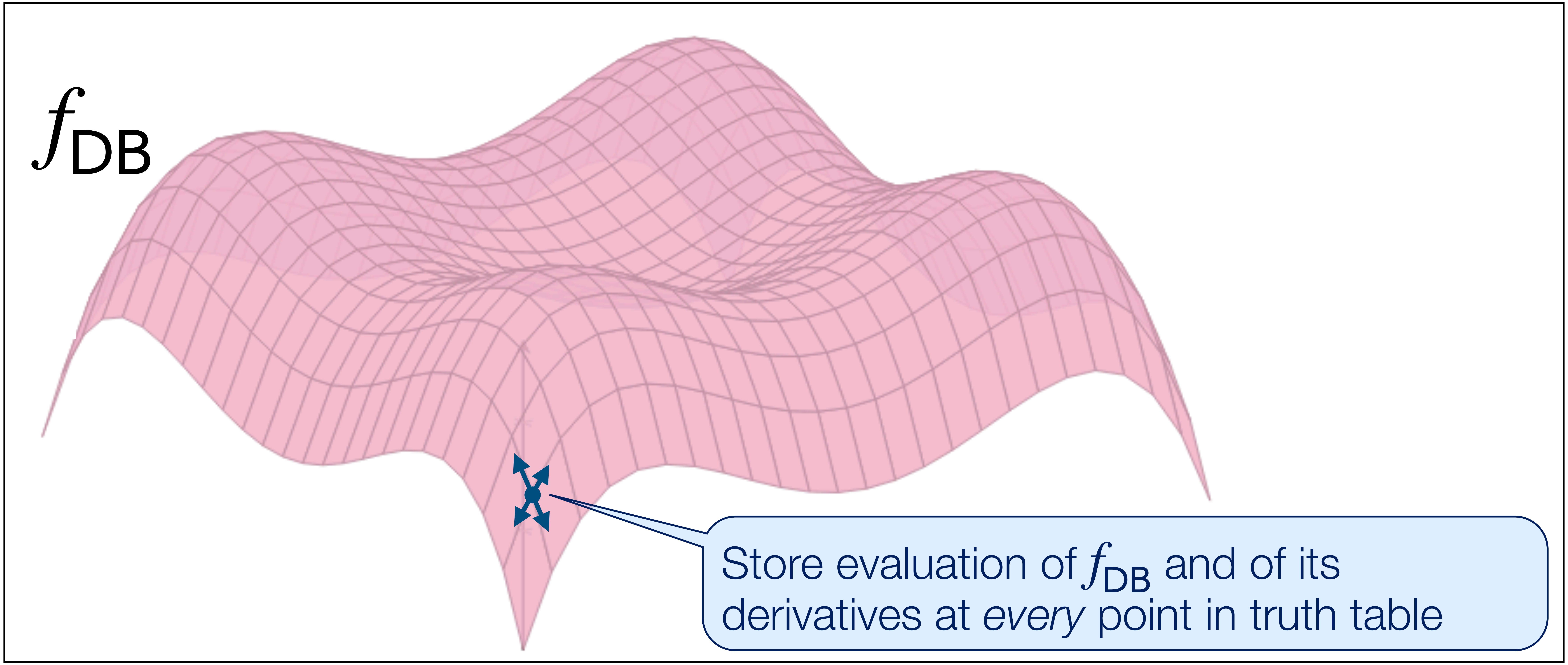
Prior work: Precompute every possible PIR answer

[BIM00,GLMDS25]



Prior work: Precompute every possible PIR answer

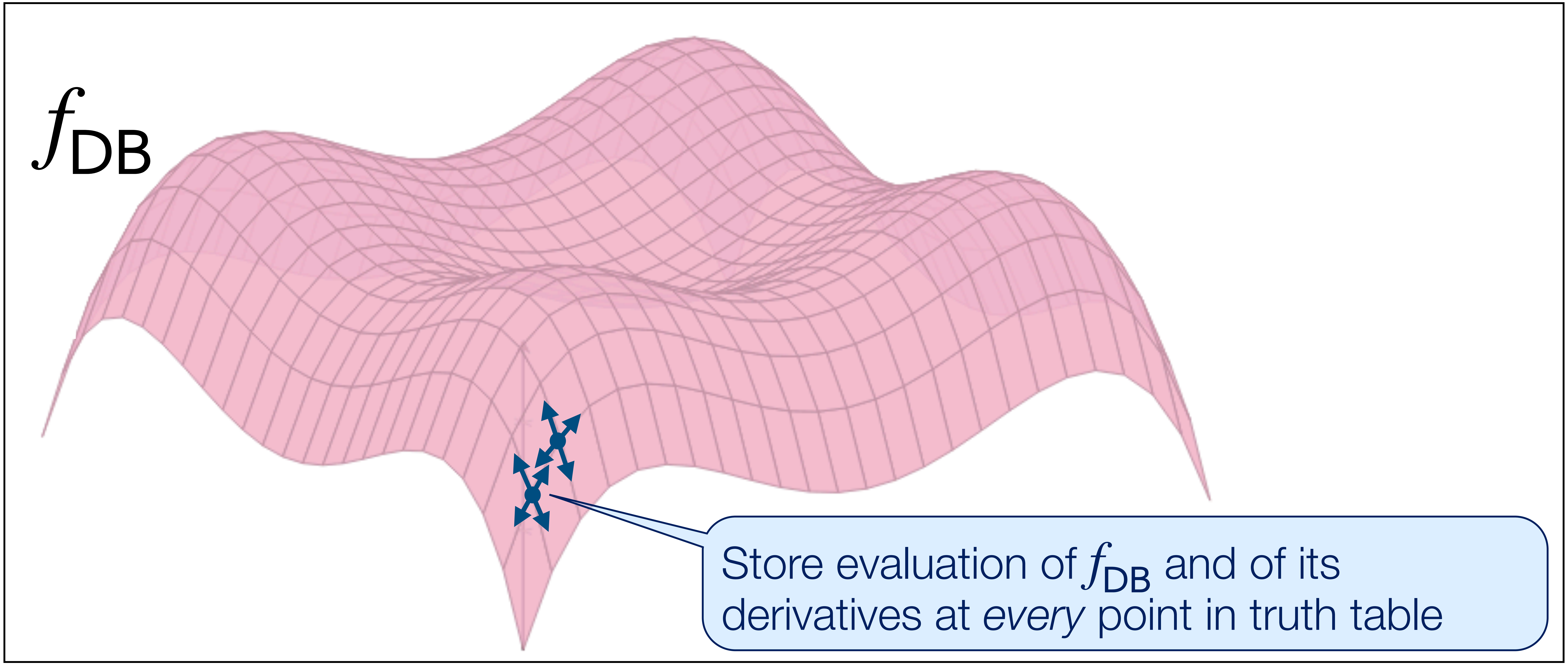
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

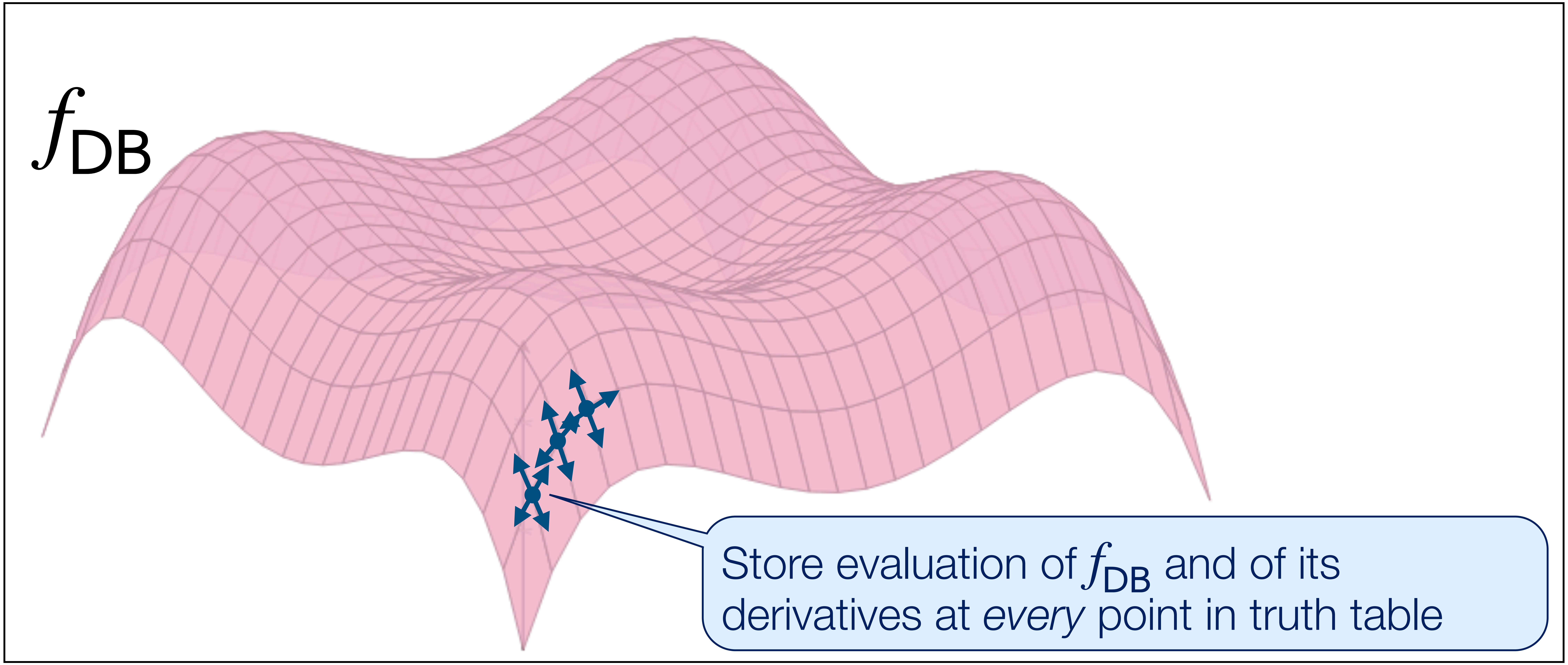
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

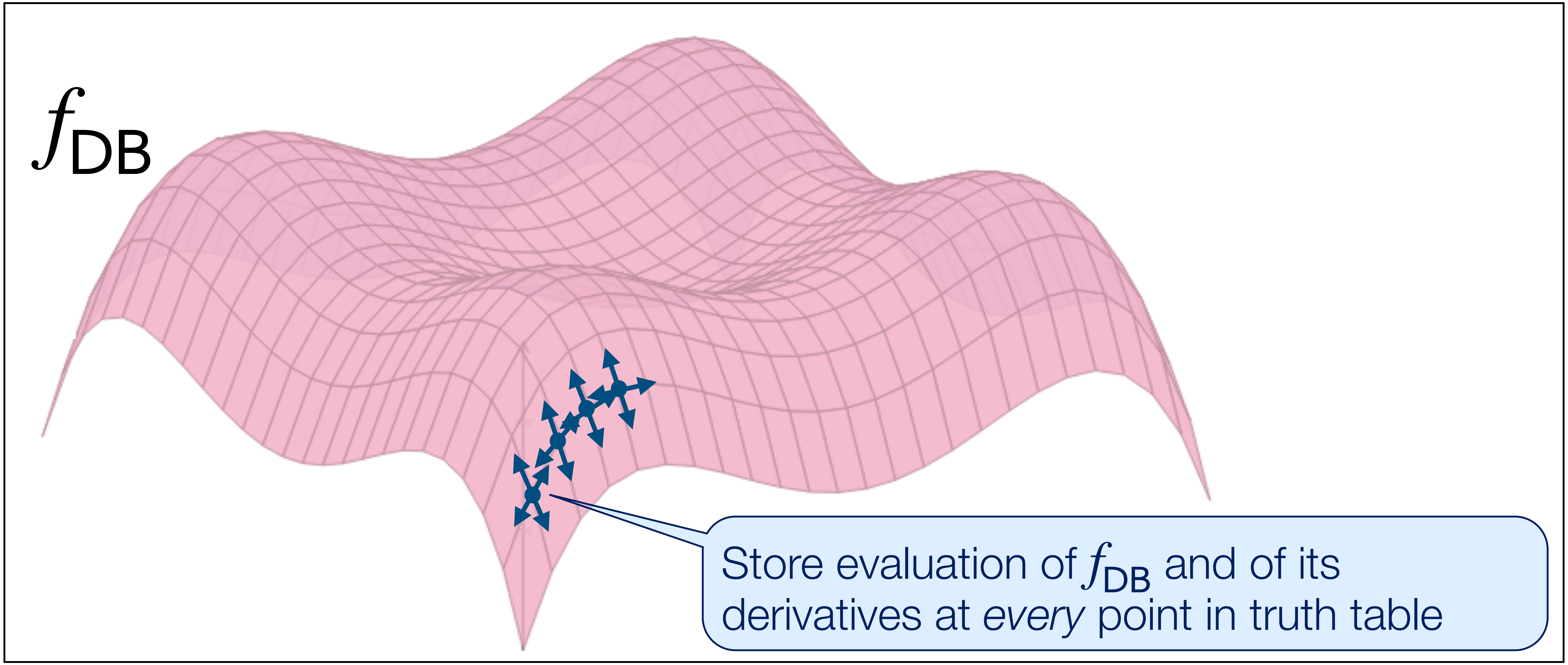
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

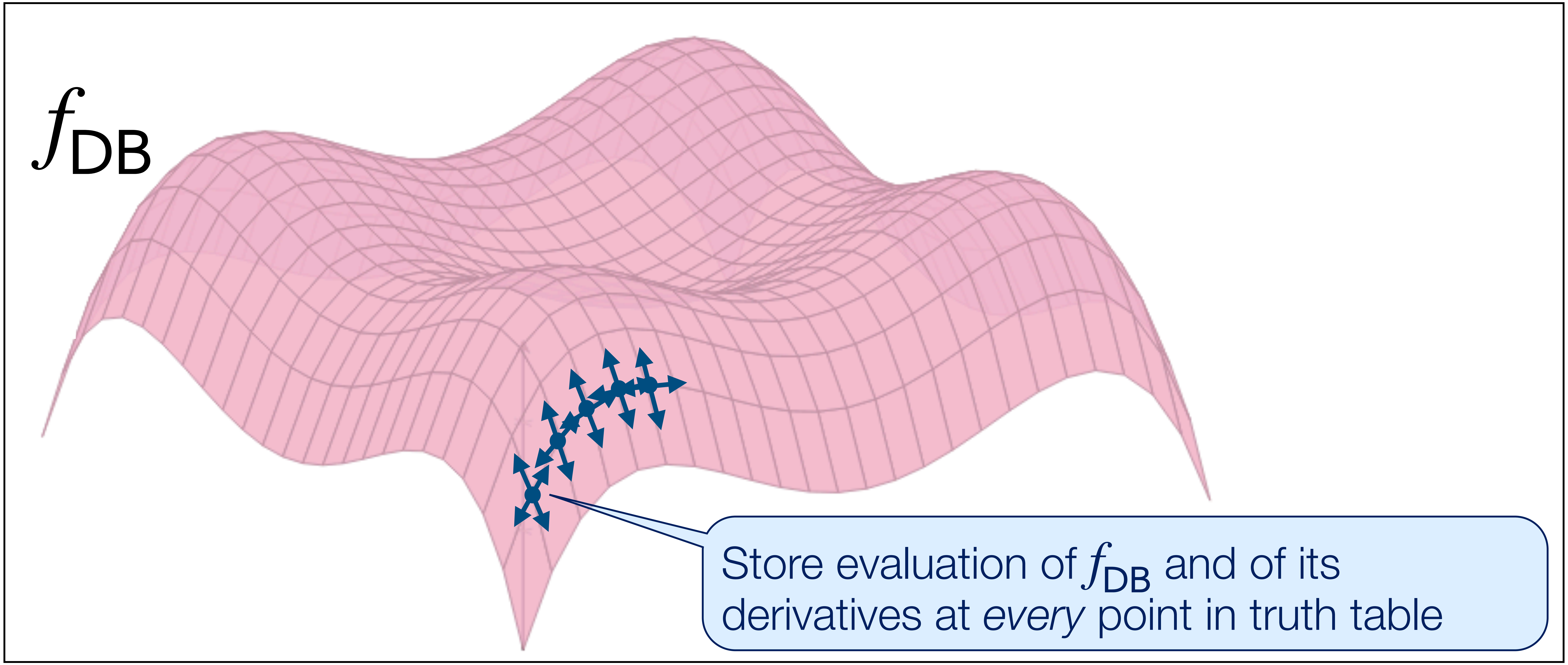
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

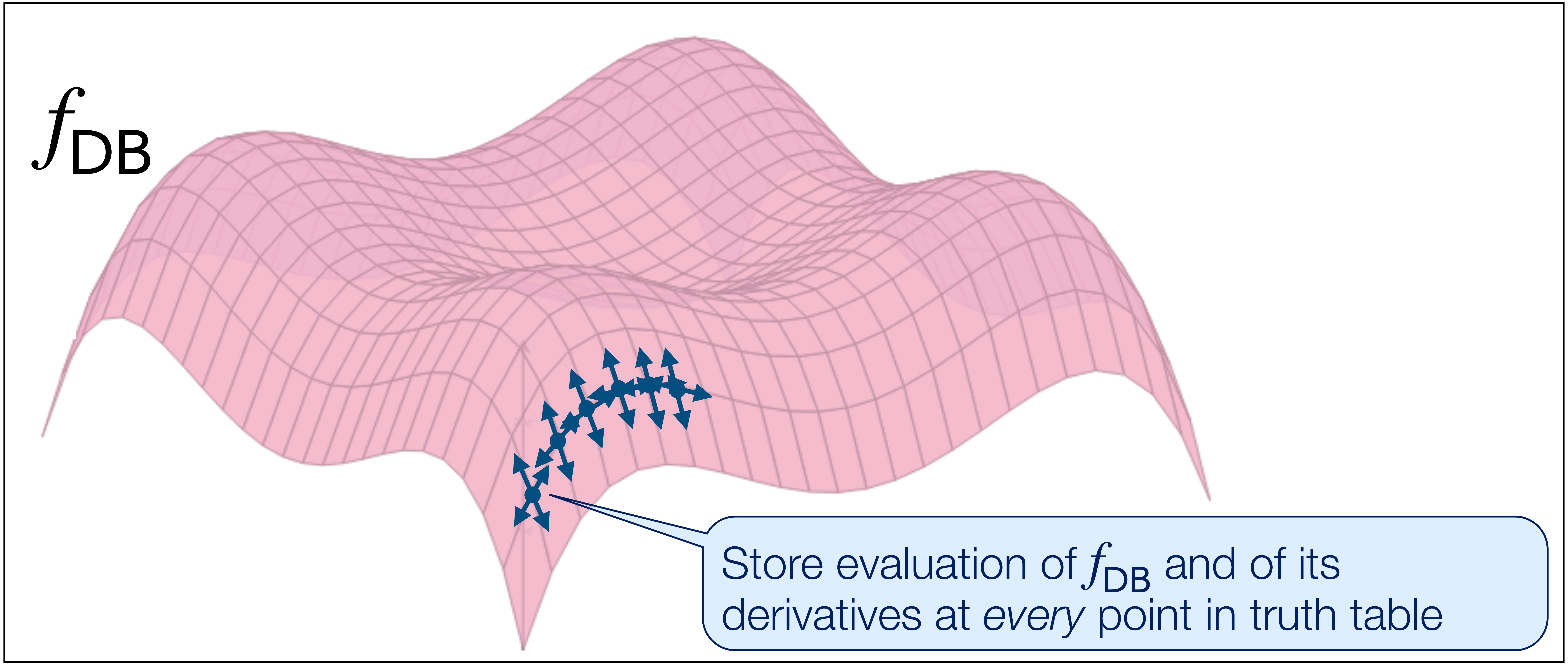
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

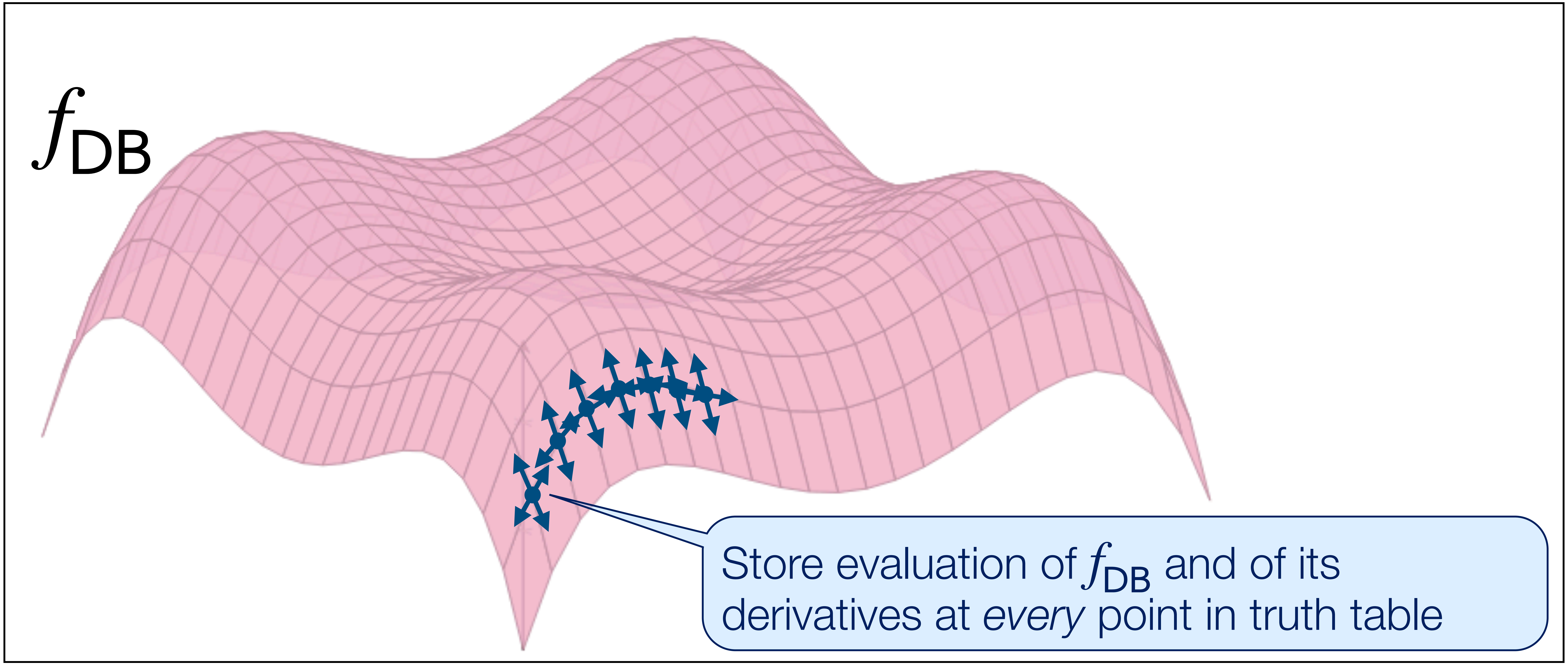
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

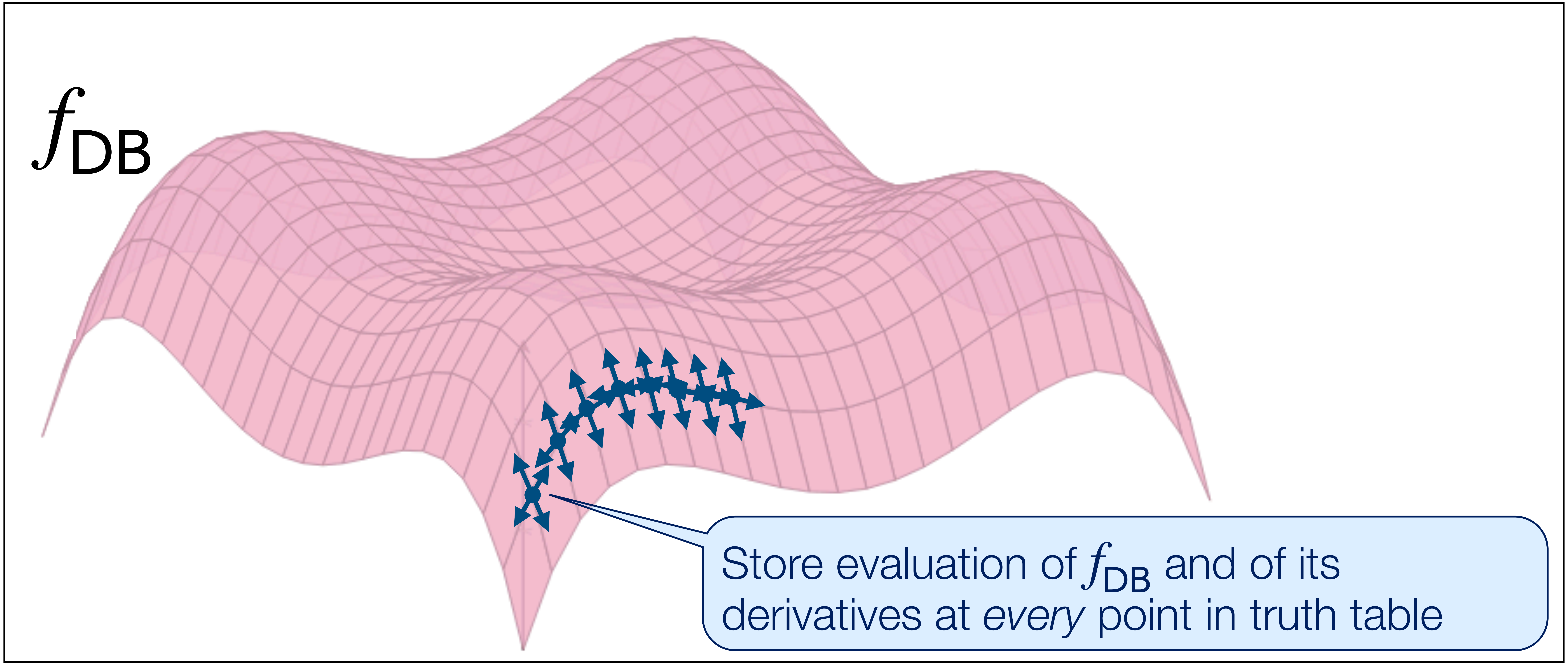
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

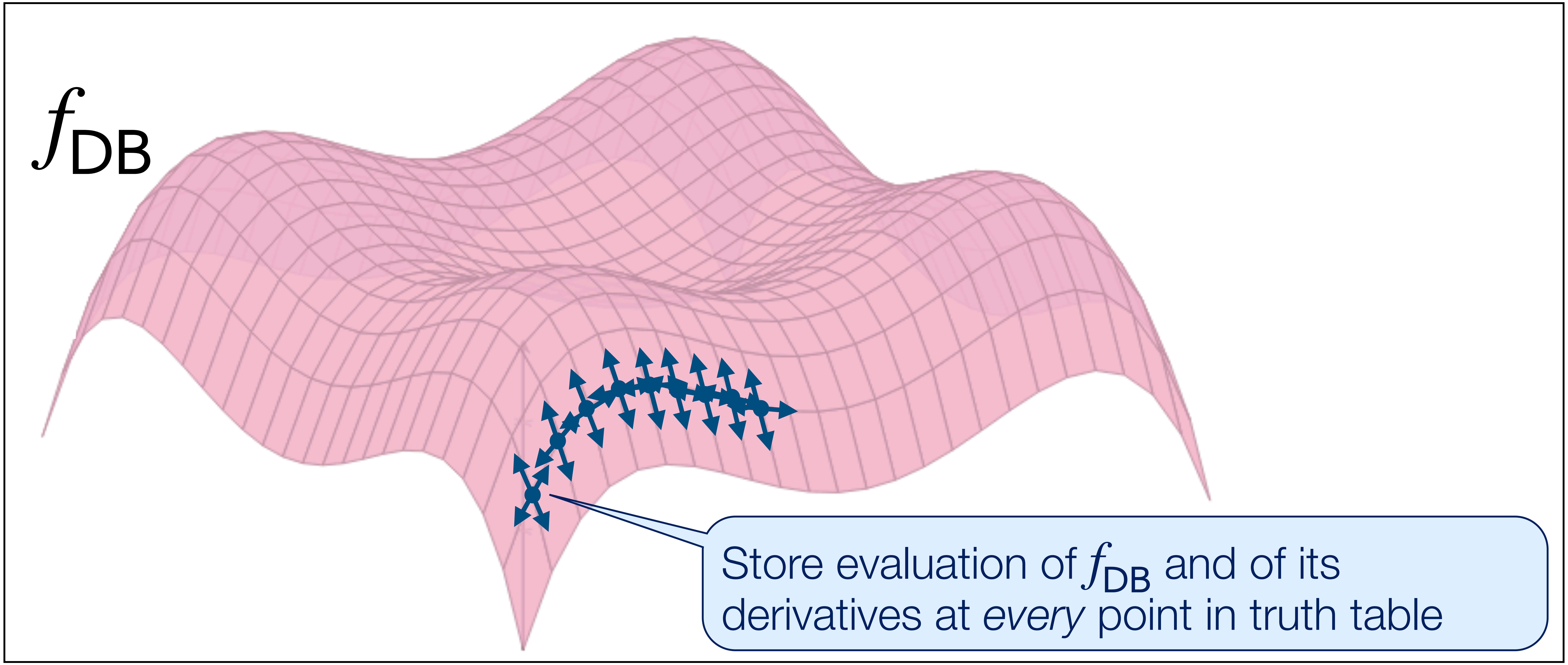
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

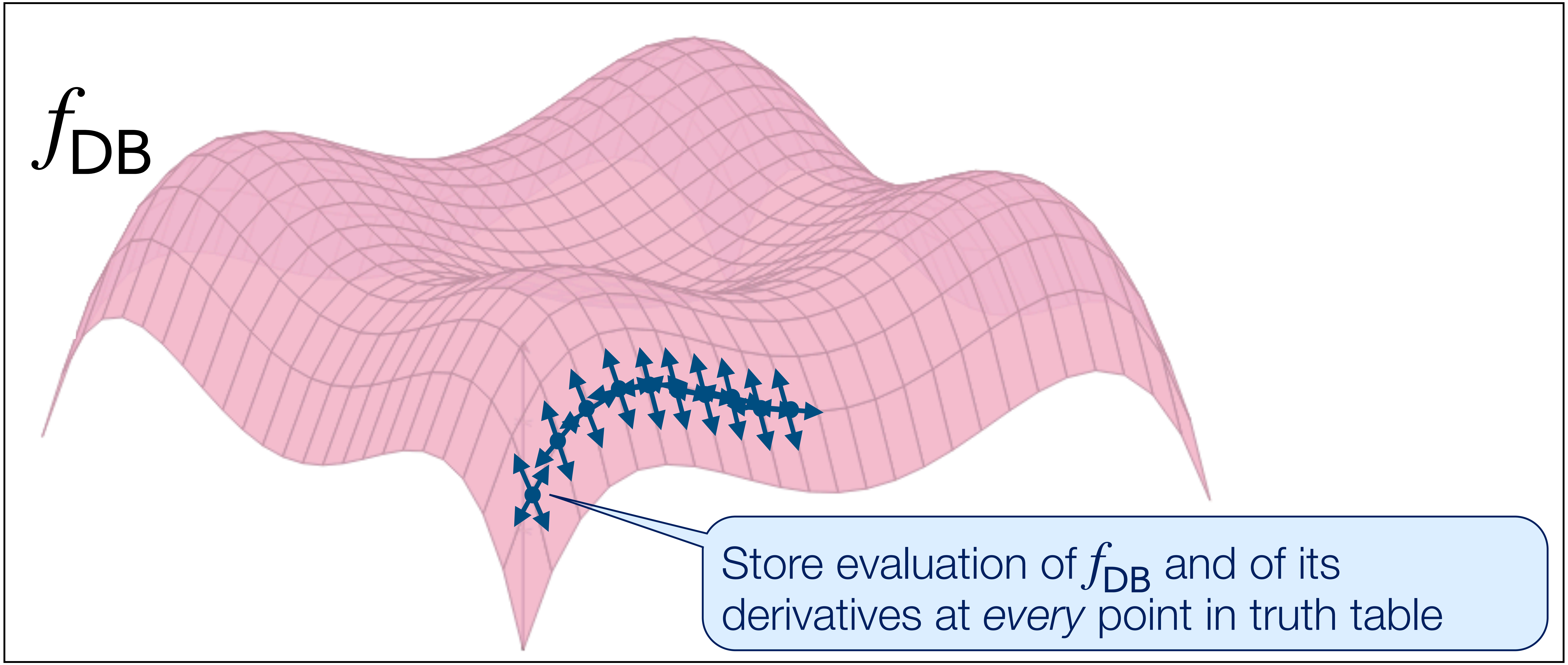
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

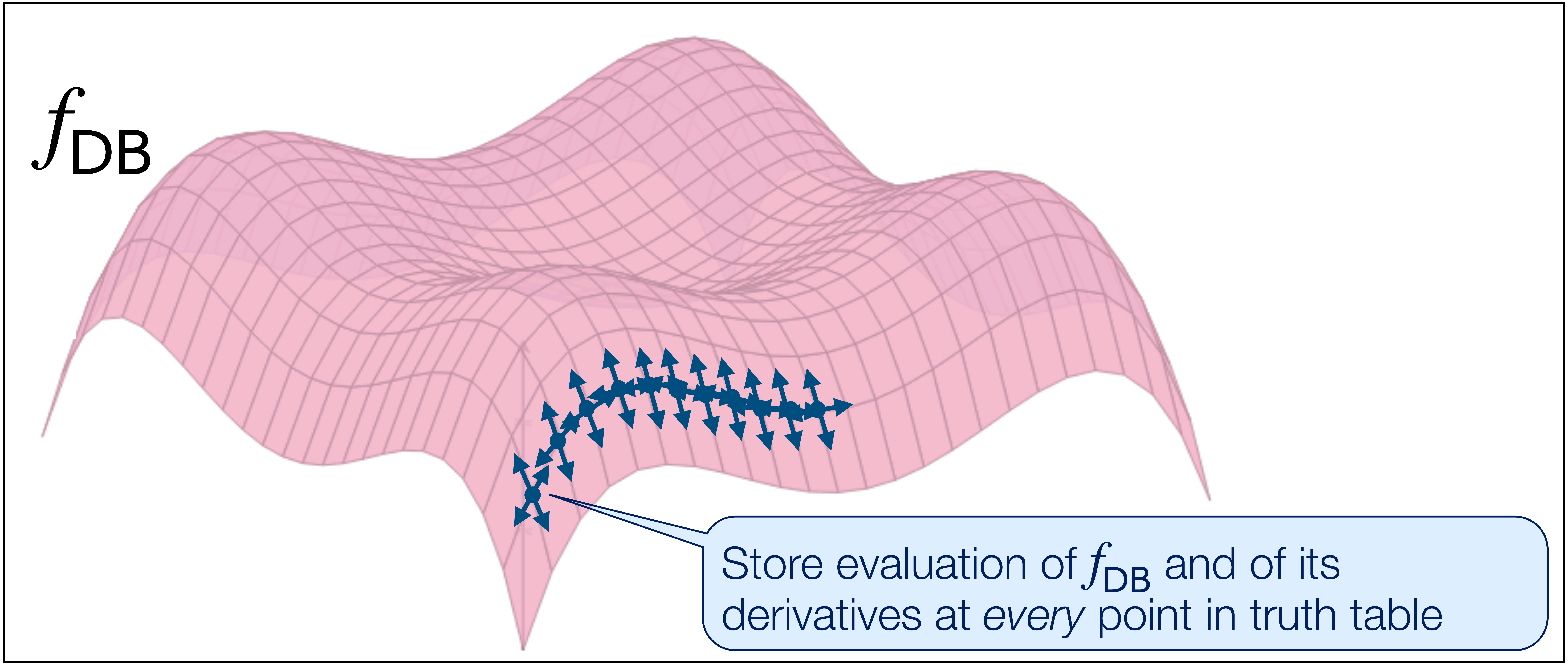
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

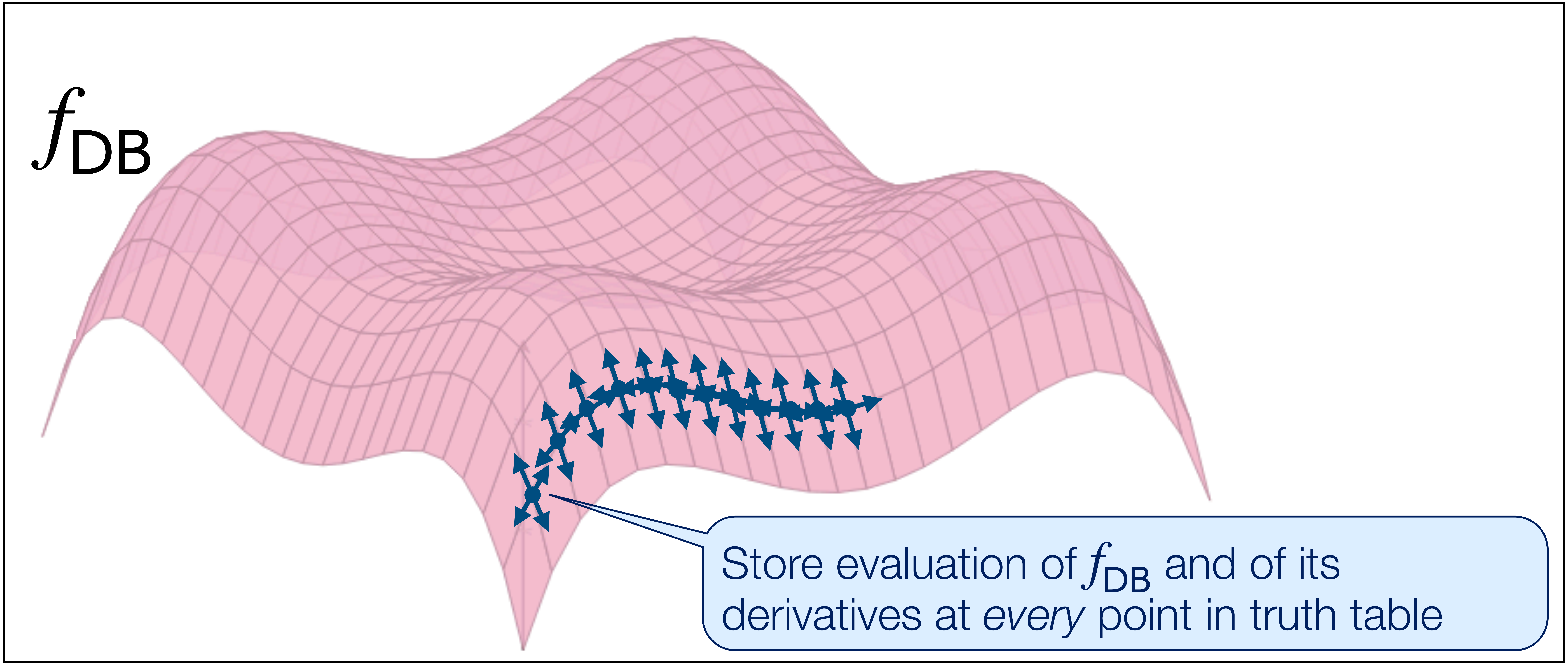
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

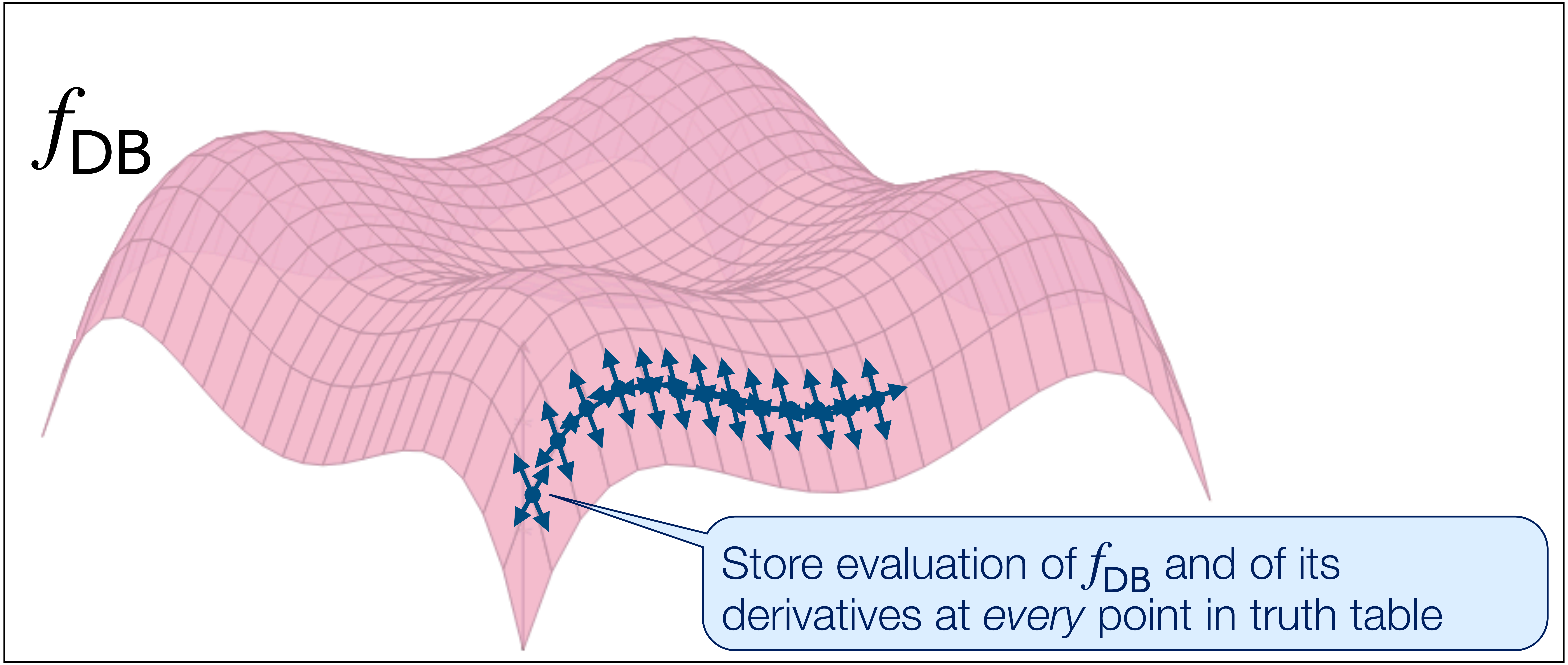
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

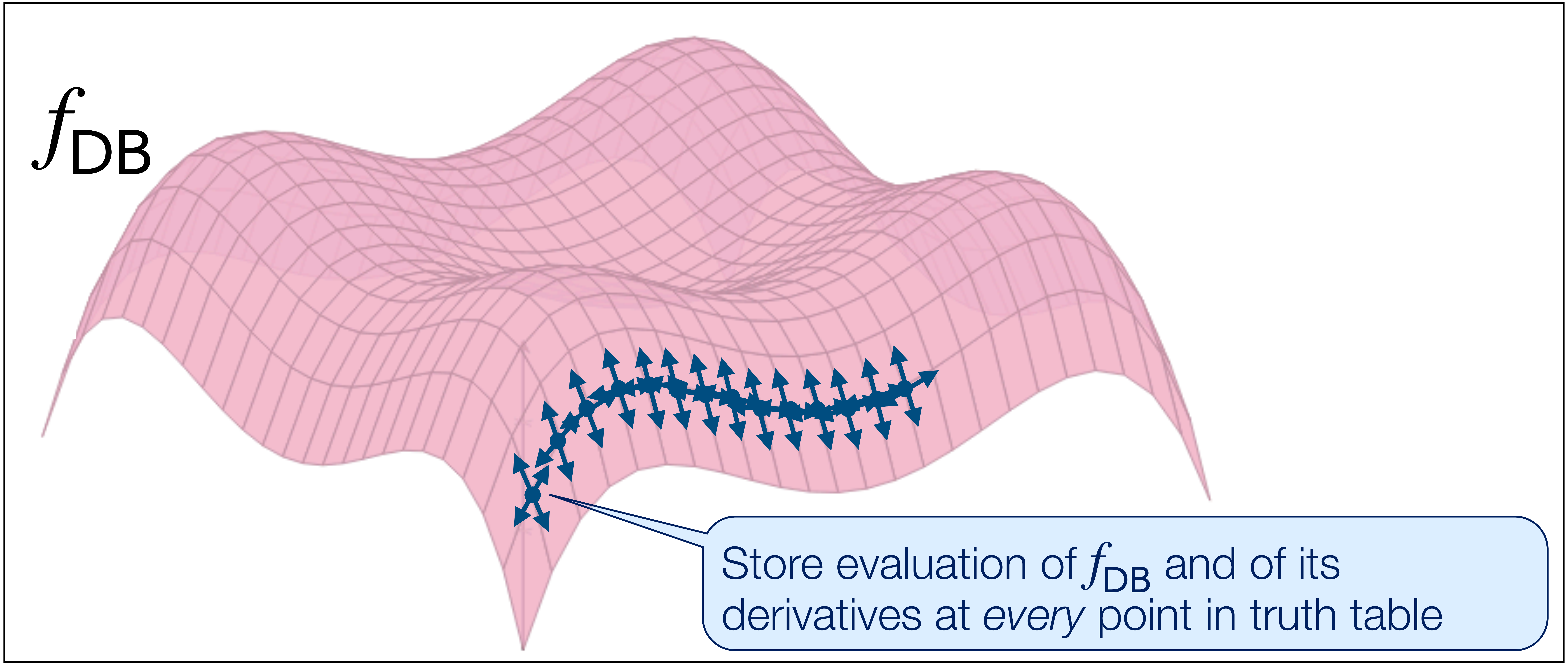
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

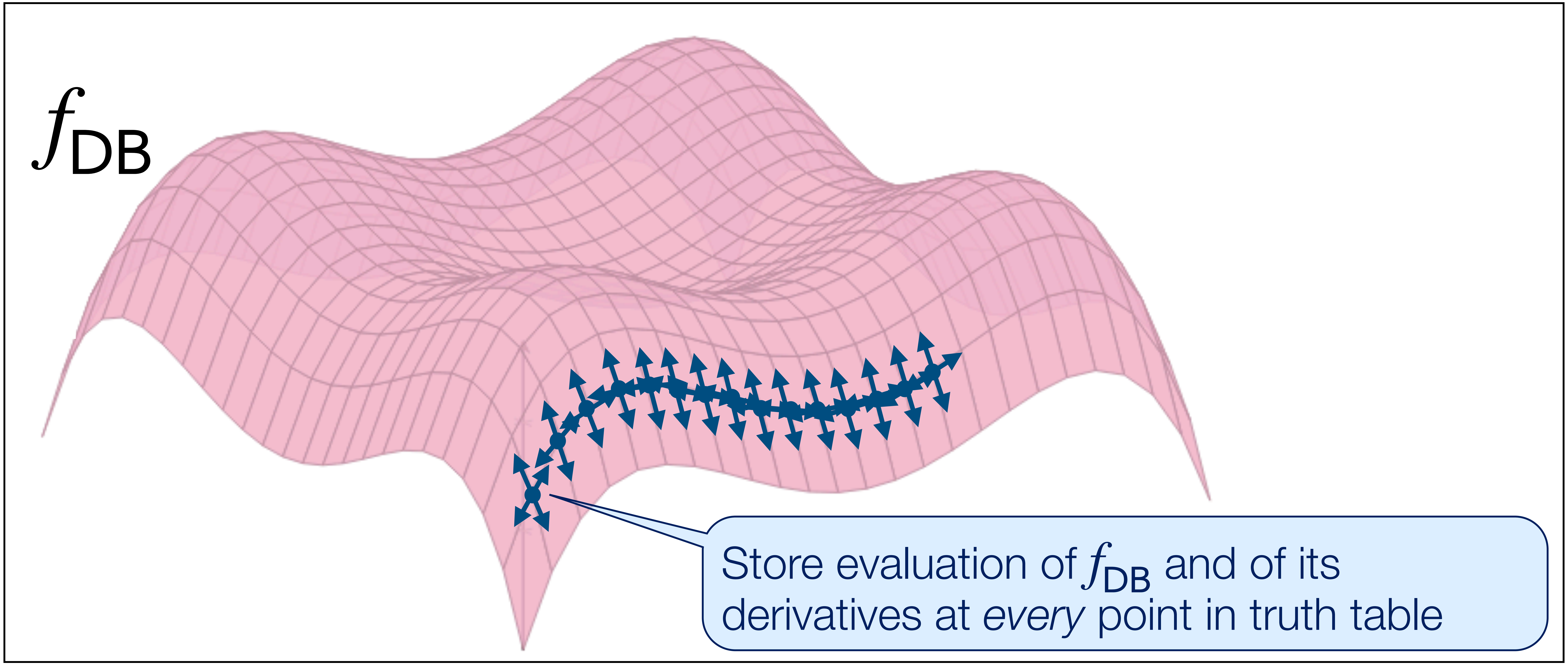
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

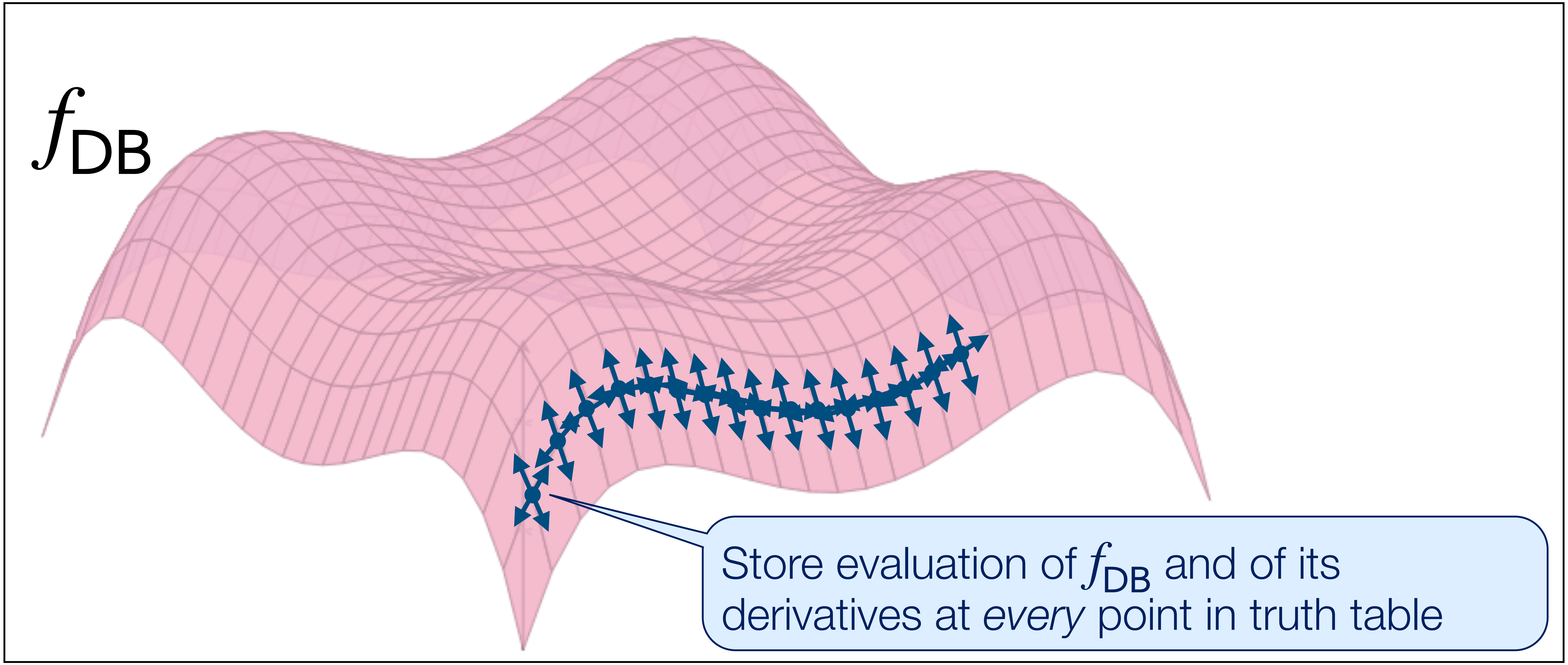
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

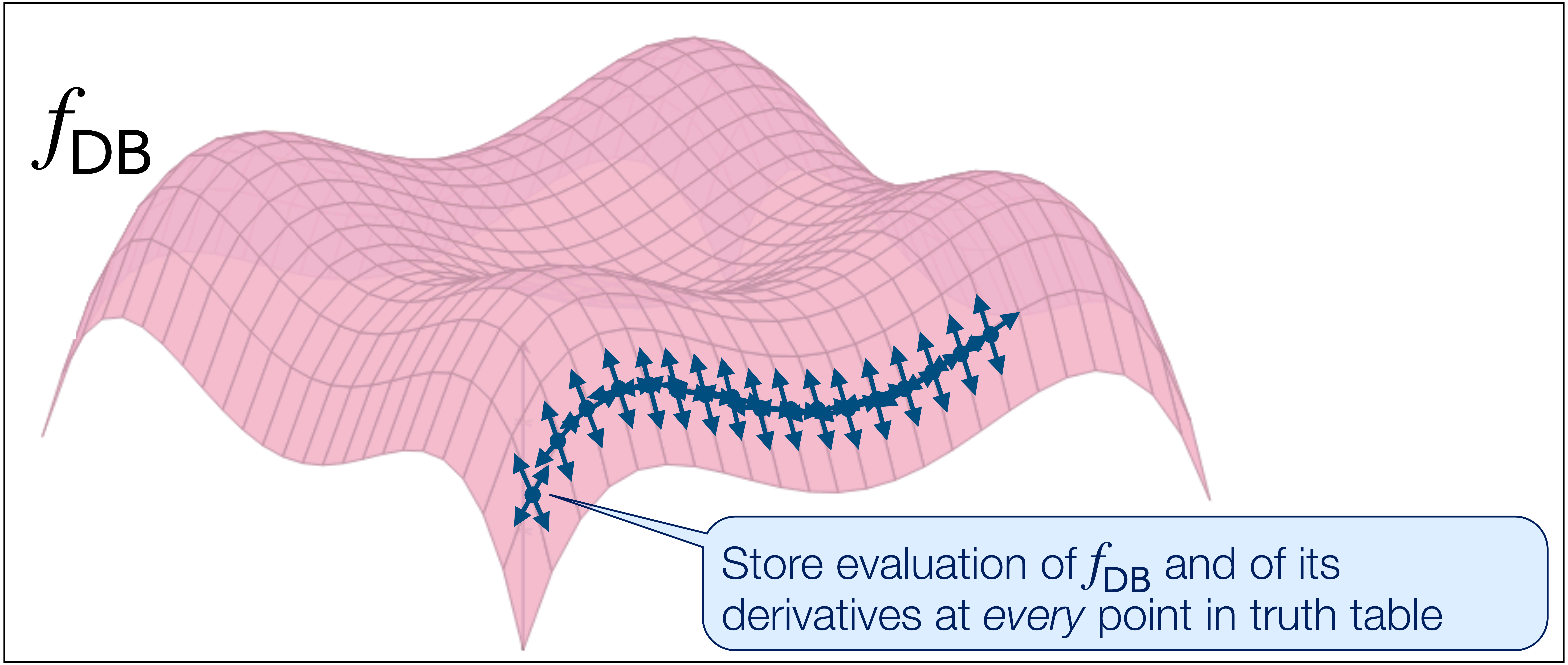
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

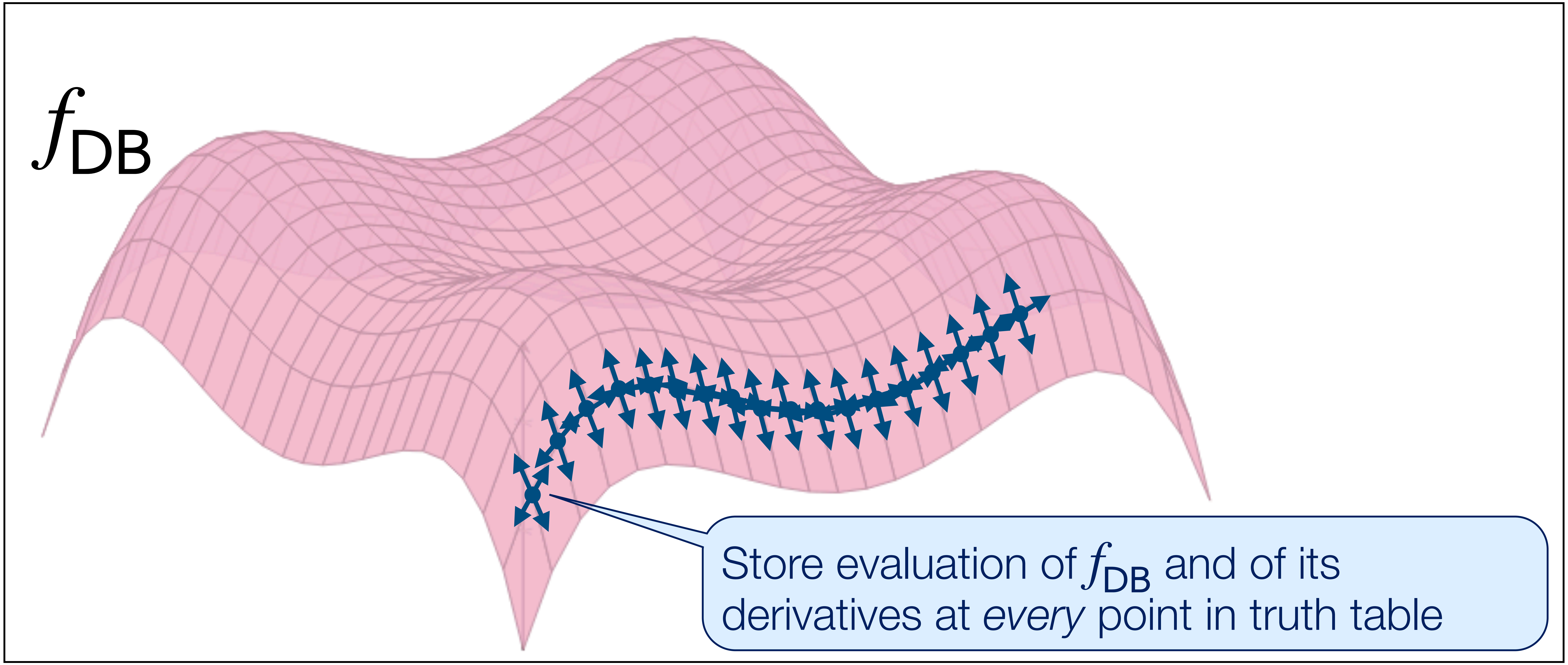
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

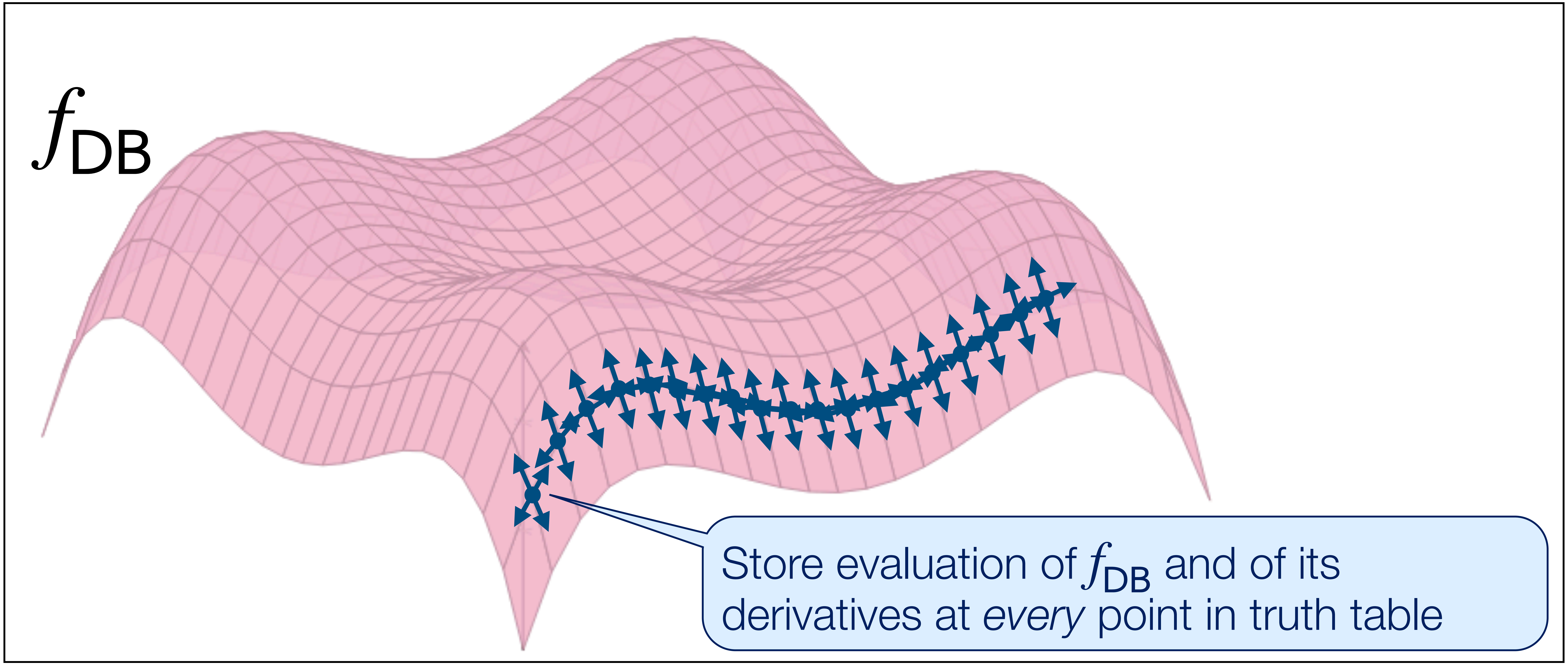
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

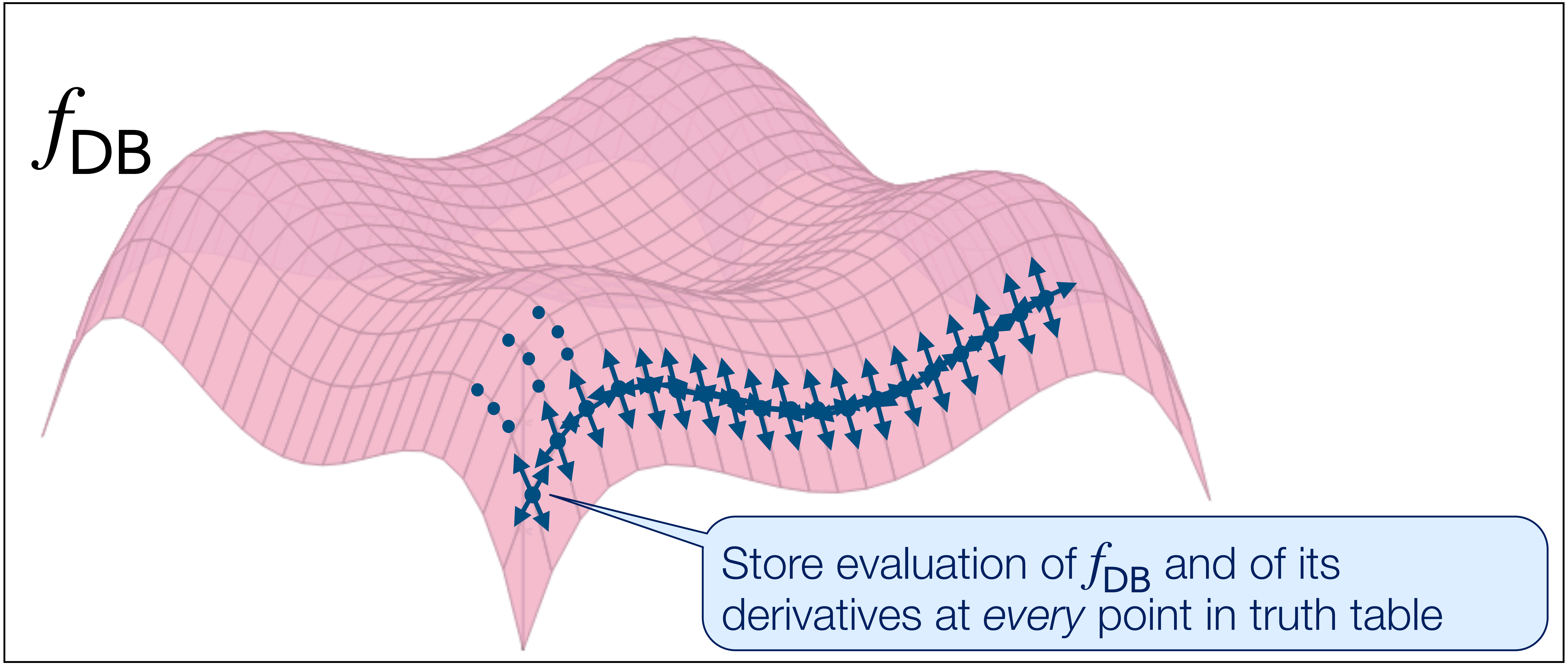
[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Prior work: Precompute every possible PIR answer

[BIM00,GLMDS25]



Store evaluation of f_{DB} and of its derivatives at *every* point in truth table

Fact 0. For a linear function $f : \mathbb{F} \to \mathbb{F}$, we have

$$f'(x) = f(x + 1) - f(x) \,.$$

Fact 0. For a linear function $f : \mathbb{F} \to \mathbb{F}$, we have

$$f'(x) = f(x+1) - f(x)\,.$$

Fact 1. Since f_{DB} is multilinear, for any evaluation point $\mathbf{x} \in \mathbb{F}_2^m$,

$$\nabla f_{\mathsf{DB}}(\mathbf{x}) = \begin{bmatrix} \vdots \\ f_{\mathsf{DB}}(\mathbf{x} + \mathbf{u}_i) - f_{\mathsf{DB}}(\mathbf{x}) \\ \vdots \end{bmatrix} \in \mathbb{F}_2^m$$

Fact 0. For a linear function $f : \mathbb{F} \to \mathbb{F}$, we have

$$f'(x) = f(x+1) - f(x)\,.$$

Fact 1. Since f_{DB} is multilinear, for any evaluation point $\mathbf{x} \in \mathbb{F}_2^m$,

$$\nabla f_{\mathsf{DB}}(\mathbf{x}) = \begin{bmatrix} \vdots \\ f_{\mathsf{DB}}(\mathbf{x} + \mathbf{u}_i) - f_{\mathsf{DB}}(\mathbf{x}) \\ \vdots \end{bmatrix} \in \mathbb{F}_2^m$$

In other words: anyone can deduce $\nabla f_{\mathsf{DB}}(\mathbf{x})$ from the evaluations of f_{DB} in a Hamming ball of radius 1 around point $\mathbf{x}$.

Fact 0. For a linear function $f : \mathbb{F} \to \mathbb{F}$, we have

$$f'(x) = f(x+1) - f(x)\,.$$

Fact 1. Since f_{DB} is multilinear, for any evaluation point $\mathbf{x} \in \mathbb{F}_2^m$,

$$\nabla f_{\mathsf{DB}}(\mathbf{x}) = \begin{bmatrix} \vdots \\ f_{\mathsf{DB}}(\mathbf{x} + \mathbf{u}_i) - f_{\mathsf{DB}}(\mathbf{x}) \\ \vdots \end{bmatrix} \in \mathbb{F}_2^m$$

Fact 2. Since f_{DB} is multilinear, for any evaluation point $\mathbf{x} \in \mathbb{F}_2^m$,

$$\nabla^2 f_{\mathsf{DB}}(\mathbf{x}) = \begin{bmatrix} f_{\mathsf{DB}}(\mathbf{x} + \mathbf{u}_i + \mathbf{u}_j) - f_{\mathsf{DB}}(\mathbf{x} + \mathbf{u}_i) - f_{\mathsf{DB}}(\mathbf{x} + \mathbf{u}_j) + f_{\mathsf{DB}}(\mathbf{x}) & \cdots \\ \vdots & \ddots \\ & \end{bmatrix}$$

Fact 0. For a linear function $f : \mathbb{F} \to \mathbb{F}$, we have

$$f'(x) = f(x+1) - f(x)\,.$$

Fact 1. Since f_{DB} is multilinear, for any evaluation point $\mathbf{x} \in \mathbb{F}_2^m$,

$$\begin{bmatrix} \vdots \end{bmatrix}$$

In other words: anyone can deduce $\nabla^2 f_{\mathsf{DB}}(\mathbf{x})$ from the evaluations of f_{DB} in a Hamming ball of radius 2 around point $\mathbf{x}$.

Fact 2. Since f_{DB} is multilinear, for any evaluation point $\mathbf{x} \in \mathbb{F}_2^m$,

$$\nabla^2 f_{\mathsf{DB}}(\mathbf{x}) = \begin{bmatrix} f_{\mathsf{DB}}(\mathbf{x}+\mathbf{u}_i+\mathbf{u}_j) - f_{\mathsf{DB}}(\mathbf{x}+\mathbf{u}_i) - f_{\mathsf{DB}}(\mathbf{x}+\mathbf{u}_j) + f_{\mathsf{DB}}(\mathbf{x}) & \dots \\ \vdots & \ddots \\ & \end{bmatrix}$$

Fact 0. For a linear function $f : \mathbb{F} \to \mathbb{F}$, we have

$$f'(x) = f(x+1) - f(x)\,.$$

Fact 1. Since f_{DB} is multilinear, for any evaluation point $\mathbf{x} \in \mathbb{F}_2^m$,

$$\nabla f_{\mathsf{DB}}(\mathbf{x}) = \begin{bmatrix} \vdots \\ f_{\mathsf{DB}}(\mathbf{x} + \mathbf{u}_i) - f_{\mathsf{DB}}(\mathbf{x}) \\ \vdots \end{bmatrix} \in \mathbb{F}_2^m$$

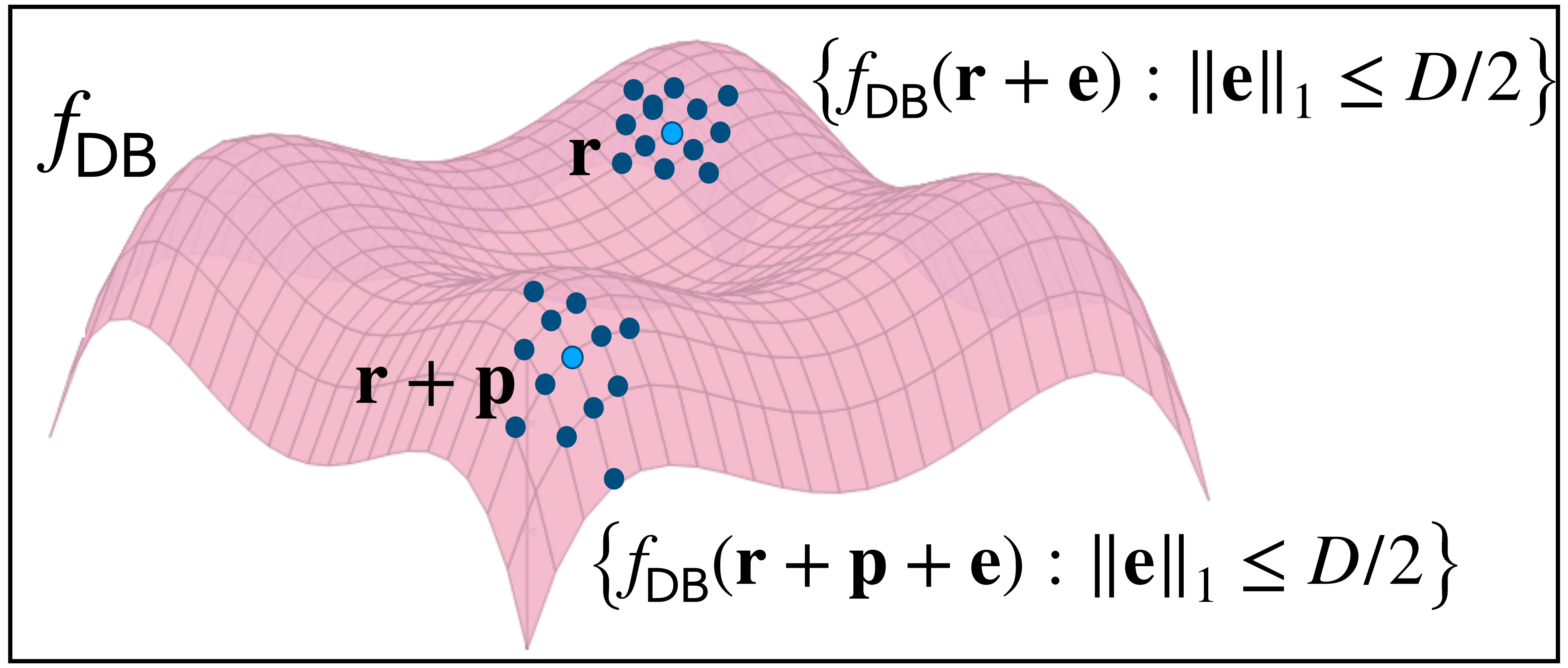
Fact 2. Since f_{DB} is multilinear, for any evaluation point $\mathbf{x} \in \mathbb{F}_2^m$,

$$\nabla^2 f_{\mathsf{DB}}(\mathbf{x}) = \begin{bmatrix} f_{\mathsf{DB}}(\mathbf{x} + \mathbf{u}_i + \mathbf{u}_j) - f_{\mathsf{DB}}(\mathbf{x} + \mathbf{u}_i) - f_{\mathsf{DB}}(\mathbf{x} + \mathbf{u}_j) + f_{\mathsf{DB}}(\mathbf{x}) & \cdots \\ \vdots & \ddots \\ & \end{bmatrix}$$

Idea: Save storage by using evaluations in Hamming balls instead of derivatives.

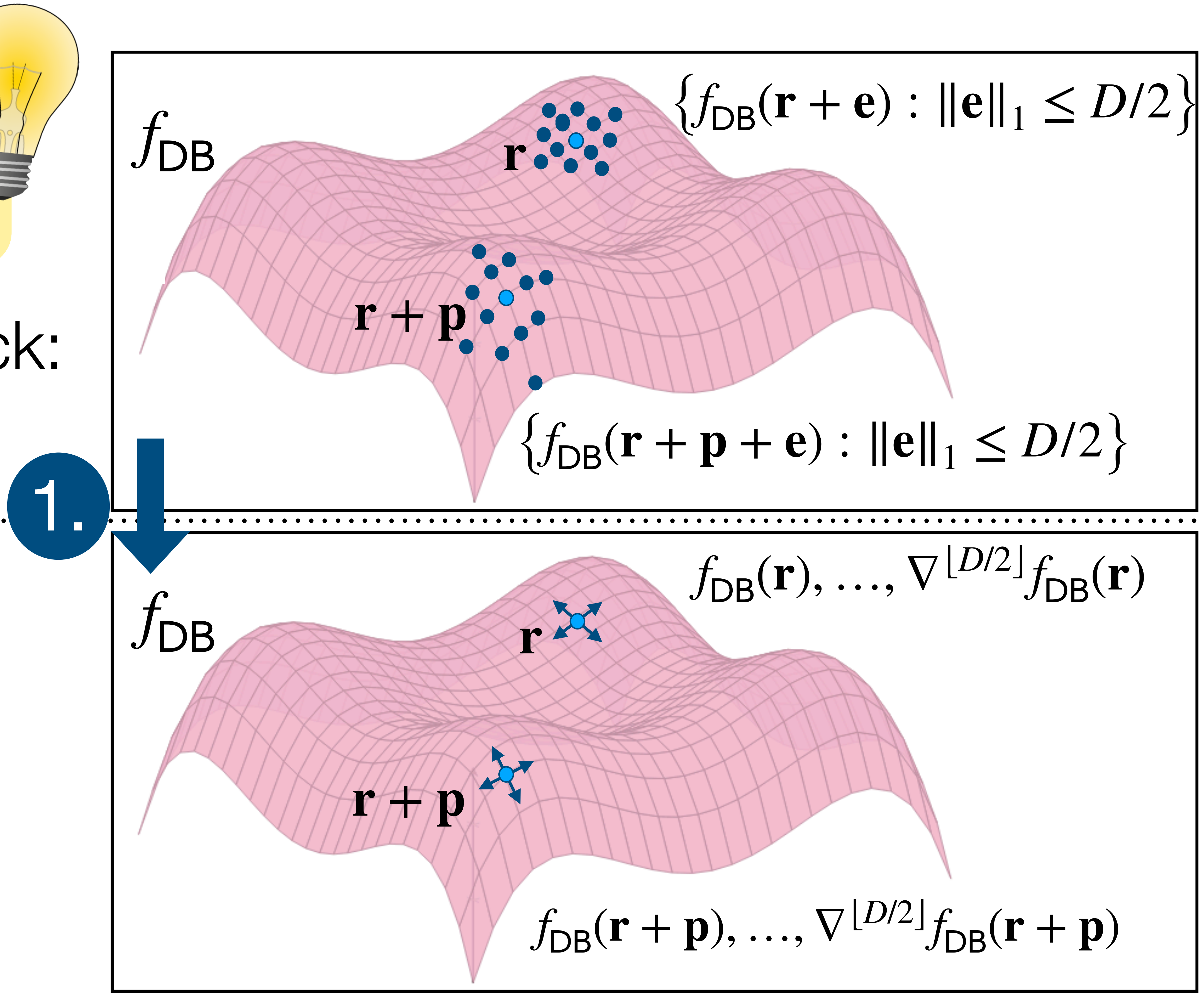
Idea: Save storage by using evaluations in Hamming balls instead of derivatives.

On query points $\mathbf{r}$ and $\mathbf{r} + \mathbf{p}$, the servers send back:


f_{DB}
$\mathbf{r}$
$\{f_{\mathsf{DB}}(\mathbf{r}+\mathbf{e}) : \|\mathbf{e}\|_1 \leq D/2\}$
$\mathbf{r}+\mathbf{p}$
$\{f_{\mathsf{DB}}(\mathbf{r}+\mathbf{p}+\mathbf{e}) : \|\mathbf{e}\|_1 \leq D/2\}$

Idea: Save storage by using evaluations in Hamming balls instead of derivatives.

On query points $\mathbf{r}$ and $\mathbf{r} + \mathbf{p}$, the servers send back:


f_{DB}
$\mathbf{r}$
$\{f_{\mathsf{DB}}(\mathbf{r} + \mathbf{e}) : \|\mathbf{e}\|_1 \leq D/2\}$
$\mathbf{r} + \mathbf{p}$
$\{f_{\mathsf{DB}}(\mathbf{r} + \mathbf{p} + \mathbf{e}) : \|\mathbf{e}\|_1 \leq D/2\}$
1.
f_{DB}
$f_{\mathsf{DB}}(\mathbf{r}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{r})$
$\mathbf{r}$
$\mathbf{r} + \mathbf{p}$
$f_{\mathsf{DB}}(\mathbf{r} + \mathbf{p}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{r} + \mathbf{p})$

From these replies, the user computes:

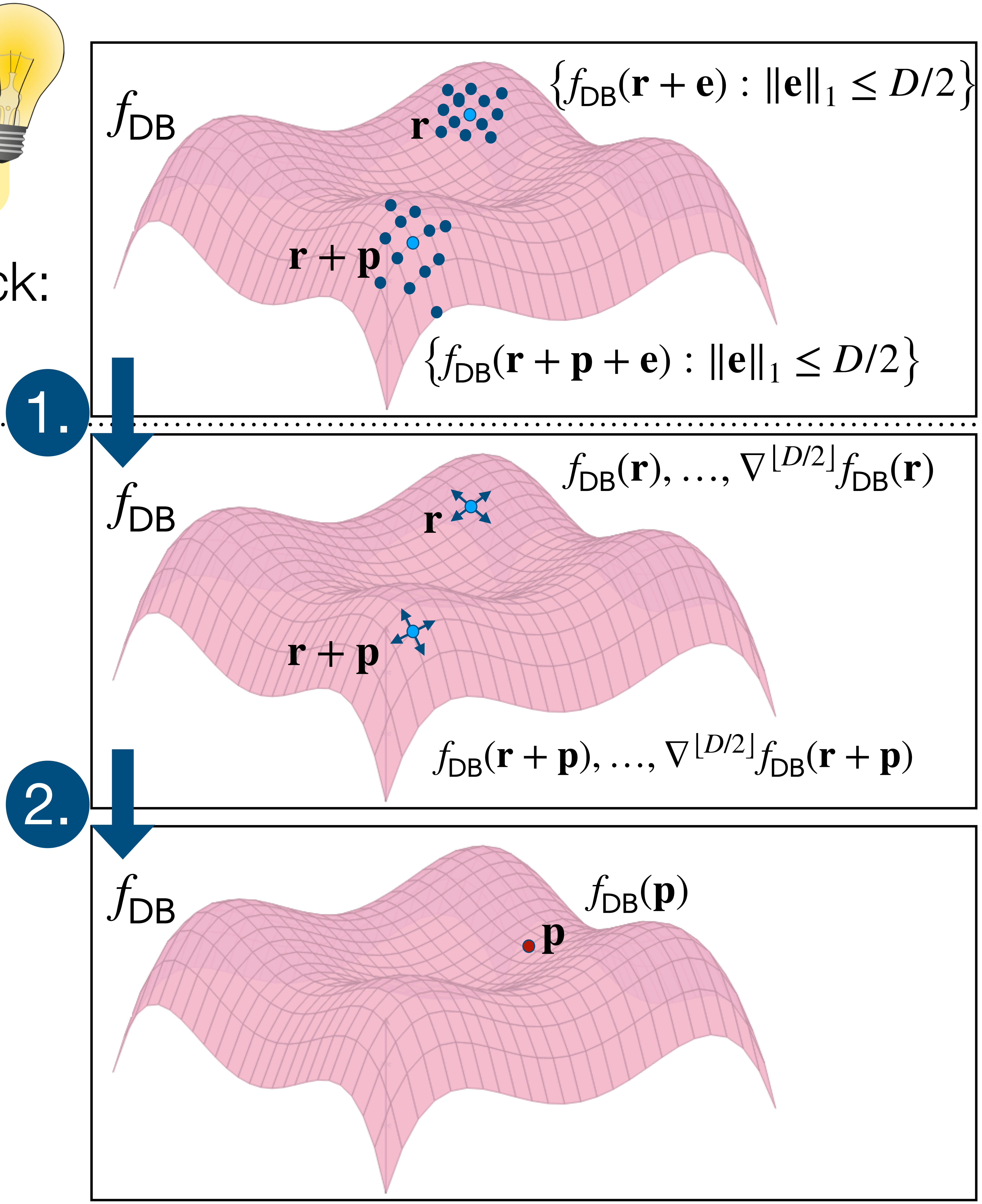
1. Finite differences

Idea: Save storage by using evaluations in Hamming balls instead of derivatives.

On query points $\mathbf{r}$ and $\mathbf{r}+\mathbf{p}$, the servers send back:

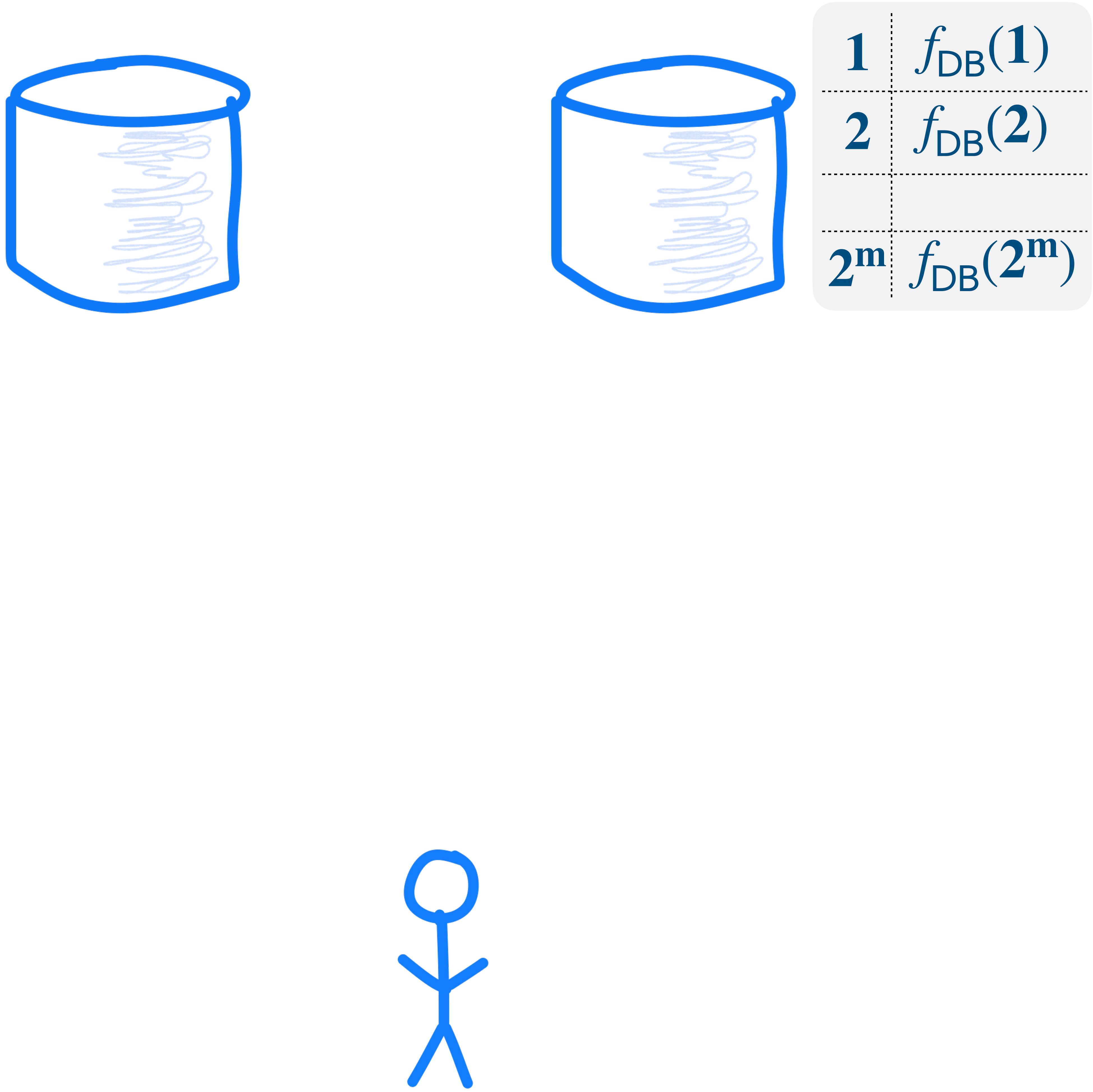
From these replies, the user computes:

1. Finite differences
2. Chain rule and Hermite interpolation


f_{DB}
$\mathbf{r}$
$\{f_{\mathrm{DB}}(\mathbf{r}+\mathbf{e}) : \|\mathbf{e}\|_1 \le D/2\}$
$\mathbf{r}+\mathbf{p}$
$\{f_{\mathrm{DB}}(\mathbf{r}+\mathbf{p}+\mathbf{e}) : \|\mathbf{e}\|_1 \le D/2\}$
1.
f_{DB}
$f_{\mathrm{DB}}(\mathbf{r}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathrm{DB}}(\mathbf{r})$
$\mathbf{r}$
$\mathbf{r}+\mathbf{p}$
$f_{\mathrm{DB}}(\mathbf{r}+\mathbf{p}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathrm{DB}}(\mathbf{r}+\mathbf{p})$
2.
f_{DB}
$f_{\mathrm{DB}}(\mathbf{p})$
$\mathbf{p}$

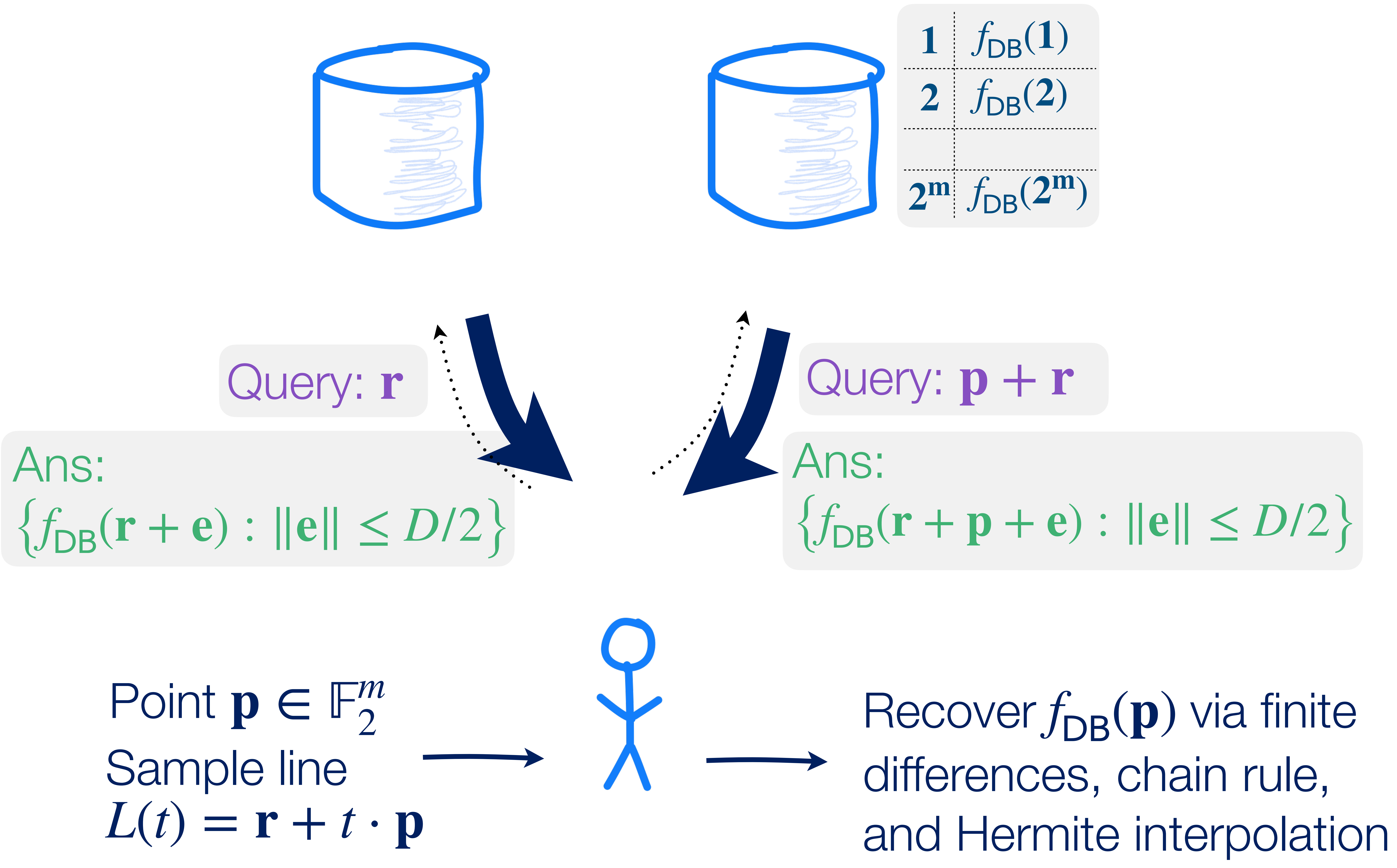
New PIR with preprocessing

New PIR with preprocessing



$\mathbf{1}$	$f_{\mathsf{DB}}(\mathbf{1})$
$\mathbf{2}$	$f_{\mathsf{DB}}(\mathbf{2})$
$\mathbf{2^m}$	$f_{\mathsf{DB}}(\mathbf{2^m})$

New PIR with preprocessing



$\mathbf{1}$	$f_{\mathsf{DB}}(\mathbf{1})$
$\mathbf{2}$	$f_{\mathsf{DB}}(\mathbf{2})$
$\mathbf{2^m}$	$f_{\mathsf{DB}}(\mathbf{2^m})$

Query: $\mathbf{r}$

Ans: $\{f_{\mathsf{DB}}(\mathbf{r}+\mathbf{e}) : \|\mathbf{e}\| \le D/2\}$

Query: $\mathbf{p}+\mathbf{r}$

Ans: $\{f_{\mathsf{DB}}(\mathbf{r}+\mathbf{p}+\mathbf{e}) : \|\mathbf{e}\| \le D/2\}$

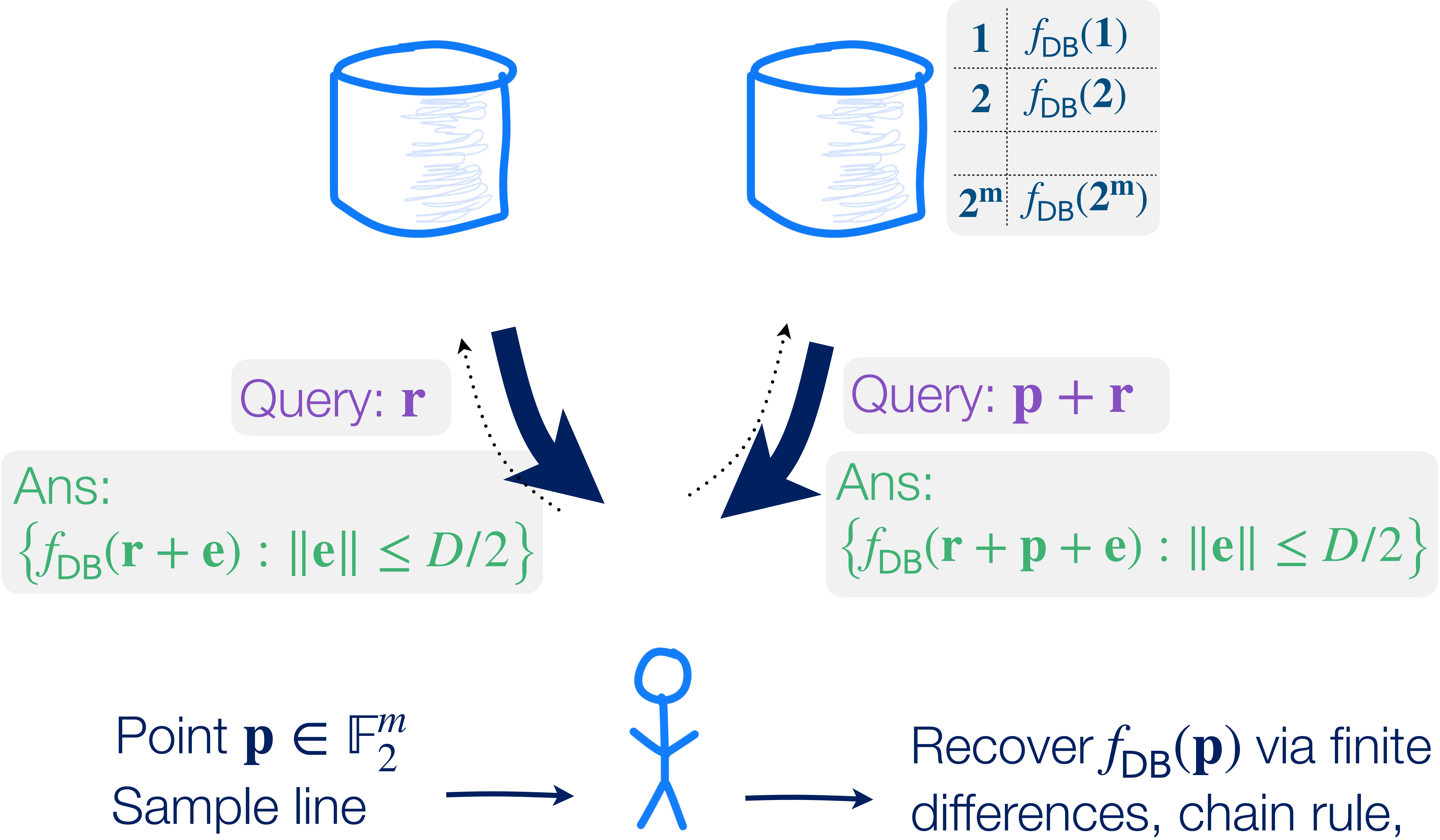
Point $\mathbf{p} \in \mathbb{F}_2^m$

Sample line

$L(t) = \mathbf{r} + t \cdot \mathbf{p}$

Recover $f_{\mathsf{DB}}(\mathbf{p})$ via finite differences, chain rule, and Hermite interpolation

New PIR with preprocessing



$\mathbf{1}$	$f_{\text{DB}}(\mathbf{1})$
$\mathbf{2}$	$f_{\text{DB}}(\mathbf{2})$
$\mathbf{2^m}$	$f_{\text{DB}}(\mathbf{2^m})$

Query: $\mathbf{r}$

Ans: $\{f_{\text{DB}}(\mathbf{r}+\mathbf{e}) : \|\mathbf{e}\| \leq D/2\}$

Query: $\mathbf{p}+\mathbf{r}$

Ans: $\{f_{\text{DB}}(\mathbf{r}+\mathbf{p}+\mathbf{e}) : \|\mathbf{e}\| \leq D/2\}$

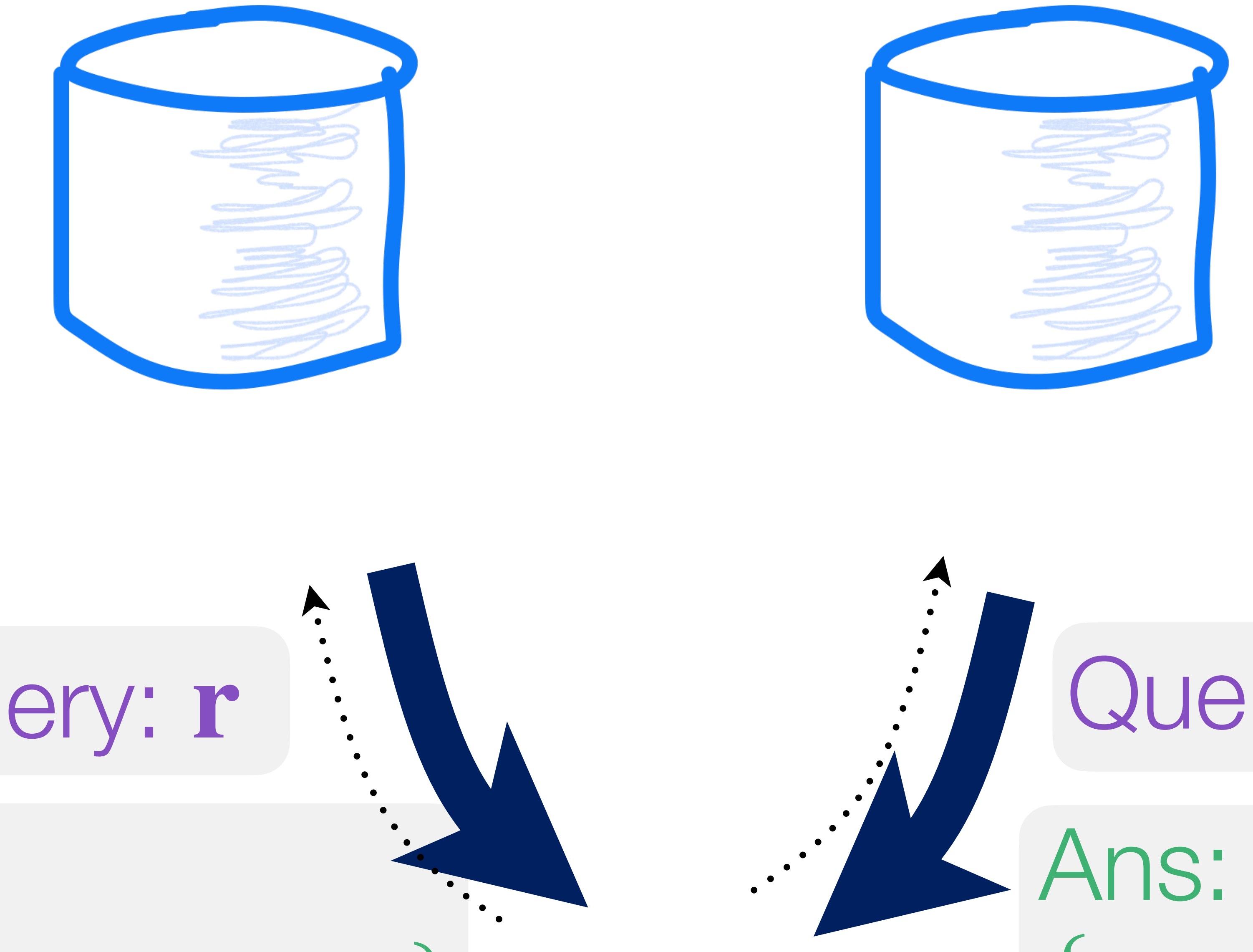
Point $\mathbf{p} \in \mathbb{F}_2^m$
Sample line
$L(t) = \mathbf{r} + t \cdot \mathbf{p}$

Recover $f_{\text{DB}}(\mathbf{p})$ via finite differences, chain rule, and Hermite interpolation

With 2 servers, gives preprocessing PIR with

- ➡ Same comm. as [BIM00]:
  $O(\log n)$ upload
  $n^{0.82}$ download
- ➡ Same time as [BIM00]:
  $O(n^{0.82})$ work
- ➡ Quasilinear space:
  $2^m = n^{1+o(1)}$ bits

New PIR with preprocessing



$\mathbf{1}$	$f_{\mathsf{DB}}(\mathbf{1})$
$\mathbf{2}$	$f_{\mathsf{DB}}(\mathbf{2})$
$\mathbf{2^m}$	$f_{\mathsf{DB}}(\mathbf{2^m})$

Query: $\mathbf{r}$

Ans: $\{f_{\mathsf{DB}}(\mathbf{r}+\mathbf{e}) : \|\mathbf{e}\| \le D/2\}$

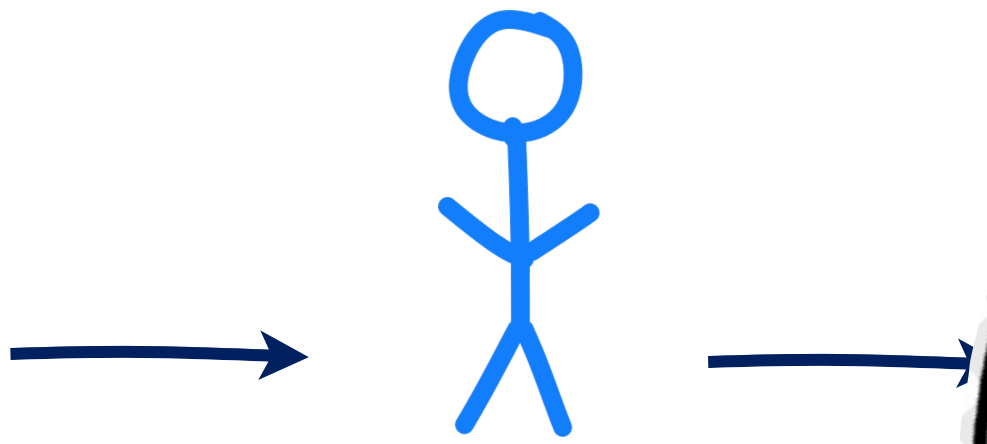
Query: $\mathbf{p}+\mathbf{r}$

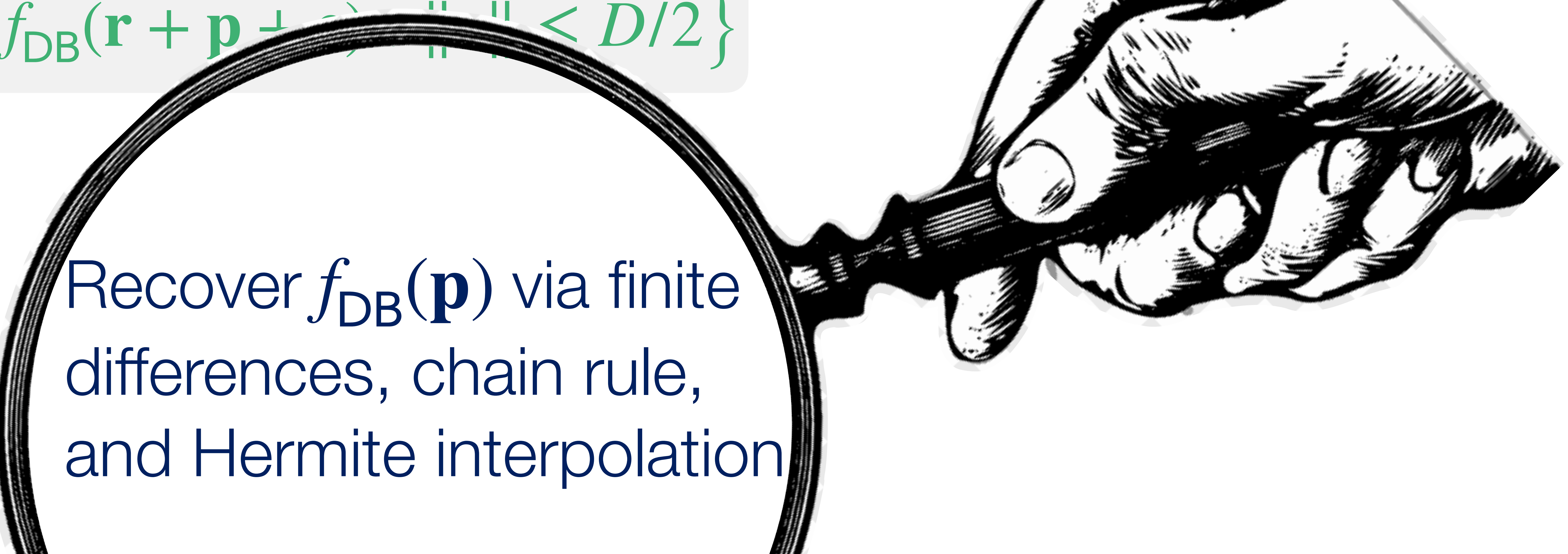
Ans: $\{f_{\mathsf{DB}}(\mathbf{r}+\mathbf{p}+\mathbf{e}) : \|\mathbf{e}\| \le D/2\}$

Point $\mathbf{p} \in \mathbb{F}_2^m$

Sample line

$L(t) = \mathbf{r} + t \cdot \mathbf{p}$



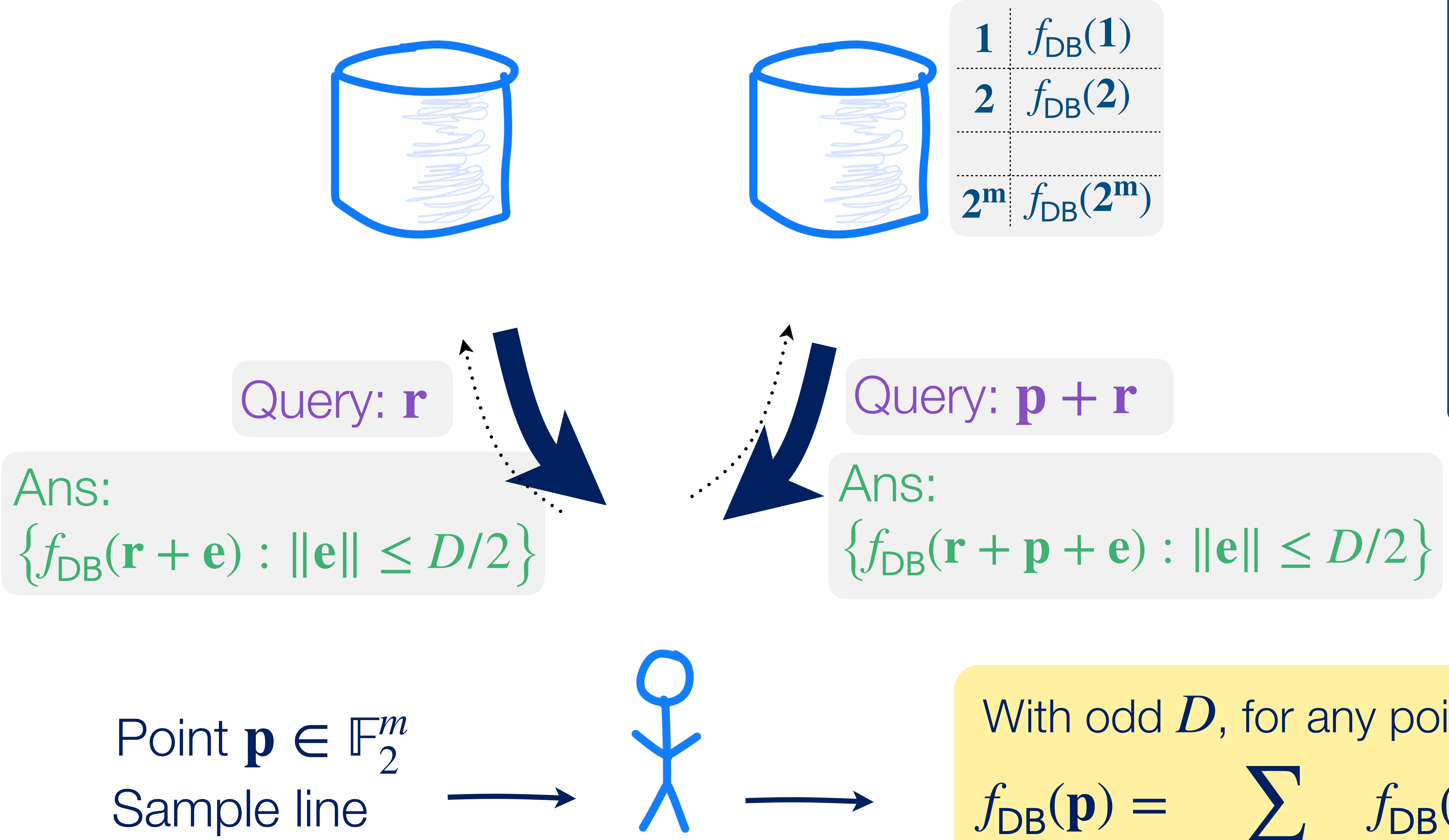


Recover $f_{\mathsf{DB}}(\mathbf{p})$ via finite differences, chain rule, and Hermite interpolation

With 2 servers, gives preprocessing PIR with

- ➡ Same comm. as [BIM00]:
  $O(\log n)$ upload
  $n^{0.82}$ download
- ➡ Same time as [BIM00]:
  $O(n^{0.82})$ work
- ➡ Quasilinear space:
  $2^m = n^{1+o(1)}$ bits

New PIR with preprocessing



1	$f_{\mathsf{DB}}(\mathbf{1})$
2	$f_{\mathsf{DB}}(\mathbf{2})$
$\mathbf{2^m}$	$f_{\mathsf{DB}}(\mathbf{2^m})$

Query: $\mathbf{r}$

Ans: $\{f_{\mathsf{DB}}(\mathbf{r}+\mathbf{e}) : \|\mathbf{e}\| \leq D/2\}$

Query: $\mathbf{p}+\mathbf{r}$

Ans: $\{f_{\mathsf{DB}}(\mathbf{r}+\mathbf{p}+\mathbf{e}) : \|\mathbf{e}\| \leq D/2\}$

Point $\mathbf{p} \in \mathbb{F}_2^m$

Sample line

$L(t) = \mathbf{r} + t \cdot \mathbf{p}$

With odd D, for any point $\mathbf{p}$ with $\|\mathbf{p}\| = D$:

$$f_{\mathsf{DB}}(\mathbf{p}) = \sum_{\substack{\|\mathbf{e}\| \leq \lfloor D/2 \rfloor \\ \mathbf{e} \leq \mathbf{p}}} f_{\mathsf{DB}}(\mathbf{r}+\mathbf{e}) + f_{\mathsf{DB}}(\mathbf{p}+\mathbf{r}+\mathbf{e})$$

With 2 servers, gives preprocessing PIR with

➡ Same comm. as [BIM00]:

$O(\log n)$ upload

$n^{0.82}$ download

➡ Same time as [BIM00]:

$O(n^{0.82})$ work

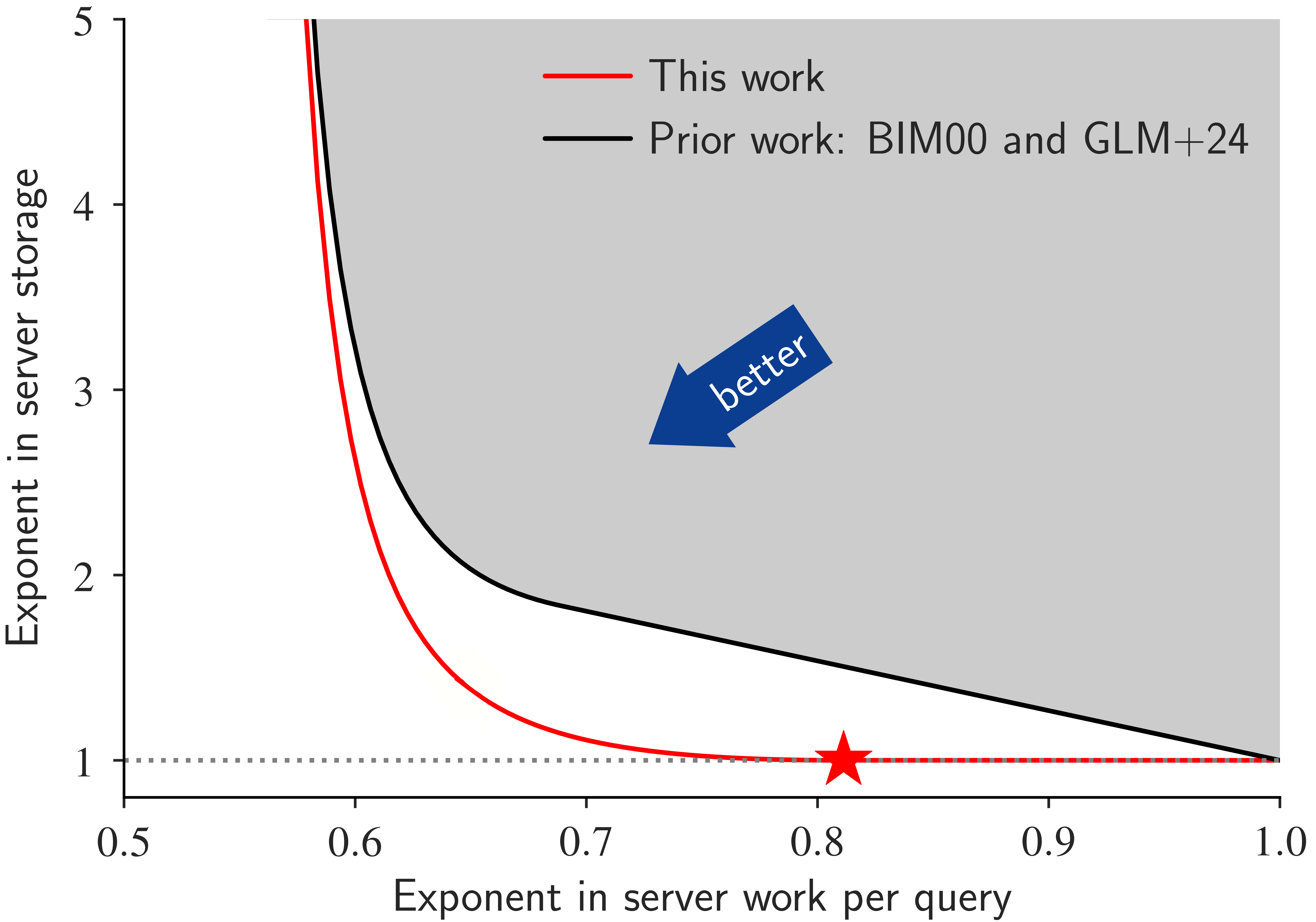
➡ Quasilinear space:

$2^m = n^{1+o(1)}$ bits

Theorem. On any database of $n > 10^6$ bits, there exists information-theoretic, two-server PIR with preprocessing with:

- $1.5 \cdot \sqrt{\log n} \cdot n$ bits of server storage,
- $12 \cdot n^{0.82}$ server RAM lookups per query, and
- $12 \cdot n^{0.82}$ bits of communication per query.

✓


This work
Prior work: BIM00 and GLM+24
better
5
4
3
2
1
0.5
0.6
0.7
0.8
0.9
1.0
Exponent in server storage
Exponent in server work per query

This talk

1. **Background:** PIR with preprocessing
2. **[HR26] New two-server PIR:** sublinear time, quasilinear space
3. **Evaluation:** what does this mean for practice?
4. **Bonuses** 😁
    - Reducing communication using crypto
    - **[HPR26]** Connecting multi-server PIR to complexity theory

This talk

1. **Background:** PIR with preprocessing
2. **[HR26] New two-server PIR:** sublinear time, quasilinear space
3. **Evaluation:** what does this mean for practice?
4. **Bonuses** 😁
   - Reducing communication using crypto
   - **[HPR26]** Connecting multi-server PIR to complexity theory

Concrete Evaluation: Storage

Compared to prior PIR with preprocessing [BIM00,GLM+25]

DB size (GB) with 1-byte records	Communication (MB)	Our storage (TB)	BIM00 storage (TB)
2	0.7	1	7.6×10^5
11	4.4	1	4.4×10^6
37	22.2	1	4.9×10^6
82	95.5	1	1.3×10^6

Concrete Evaluation: Space-Time Tradeoff

Compared to fastest two-server, linear-time PIR with $\sqrt{n}$ communication

DB size (GB) with 1-byte records	**Storage blowup**	**Communication blowup**	**Memory accesses saved**	**Throughput improvement**
2	512x	14x	2,926x	10.2x
11	93x	37x	2,560x	9.0x
37	28x	101x	1,707x	5.5x
82	12x	298x	879x	1.8x

This talk

1. **Background:** PIR with preprocessing
2. **[HR26] New two-server PIR:** sublinear time, quasilinear space
3. **Evaluation:** what does this mean for practice?
4. **Bonuses** 😁
    - Reducing communication using crypto
    - **[HPR26]** Connecting multi-server PIR to complexity theory

This talk

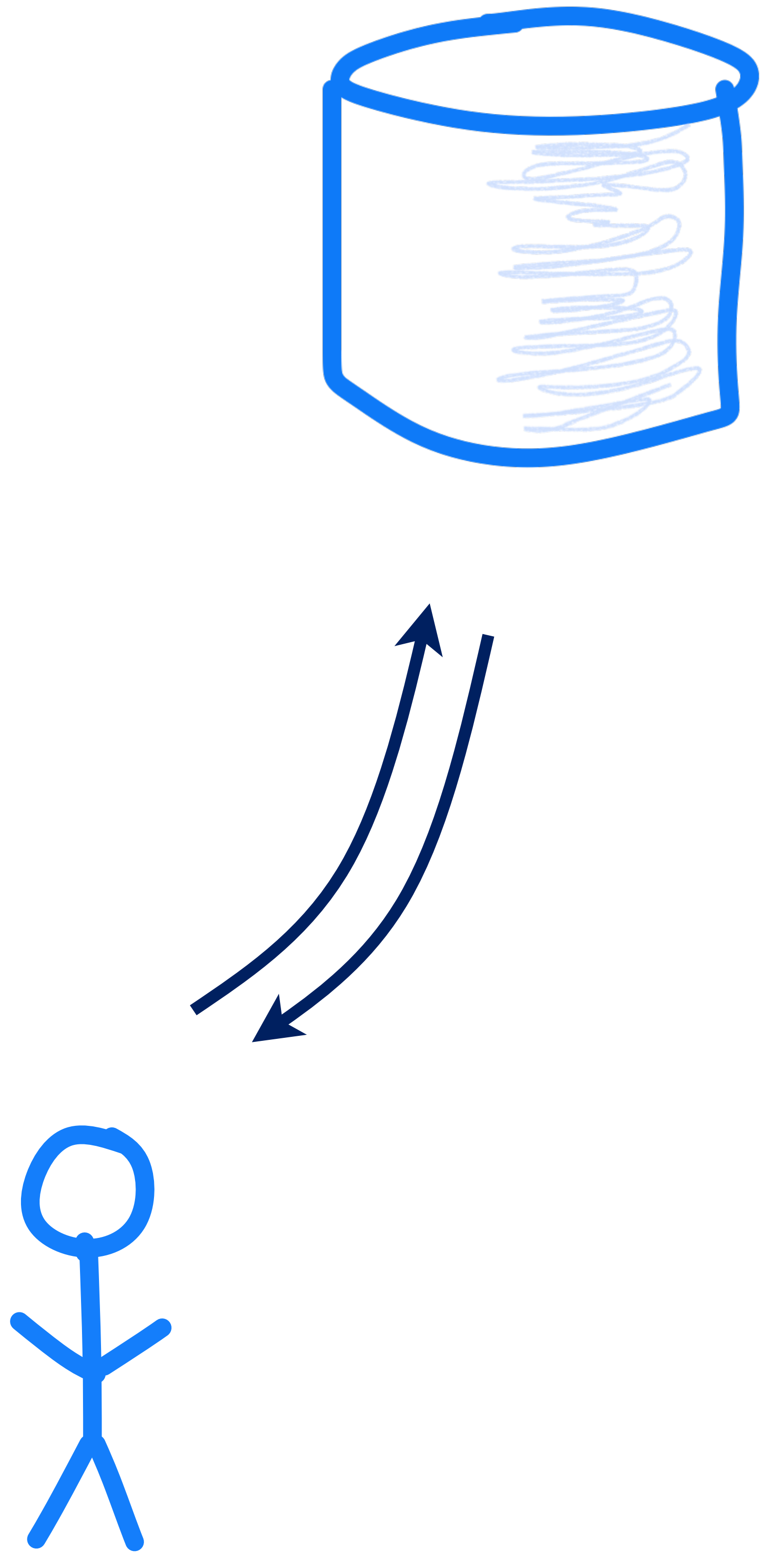
1. **Background:** PIR with preprocessing

2. **[HR26] New two-server PIR:** sublinear time, quasilinear space
3. **Evaluation:** what does this mean for practice?
4. **Bonuses** 😁
    - Reducing communication using crypto
    - **[HPR26]** Connecting multi-server PIR to complexity theory

This talk

1. **Background:** PIR with preprocessing
2. **[HR26] New two-server PIR:** sublinear time, quasilinear space
3. **Evaluation:** what does this mean for practice?
4. **Bonuses** 😁
   - ➡ Reducing communication using crypto
   - **[HPR26]** Connecting multi-server PIR to complexity theory

Homomorphic Encryption for Single-Server PIR

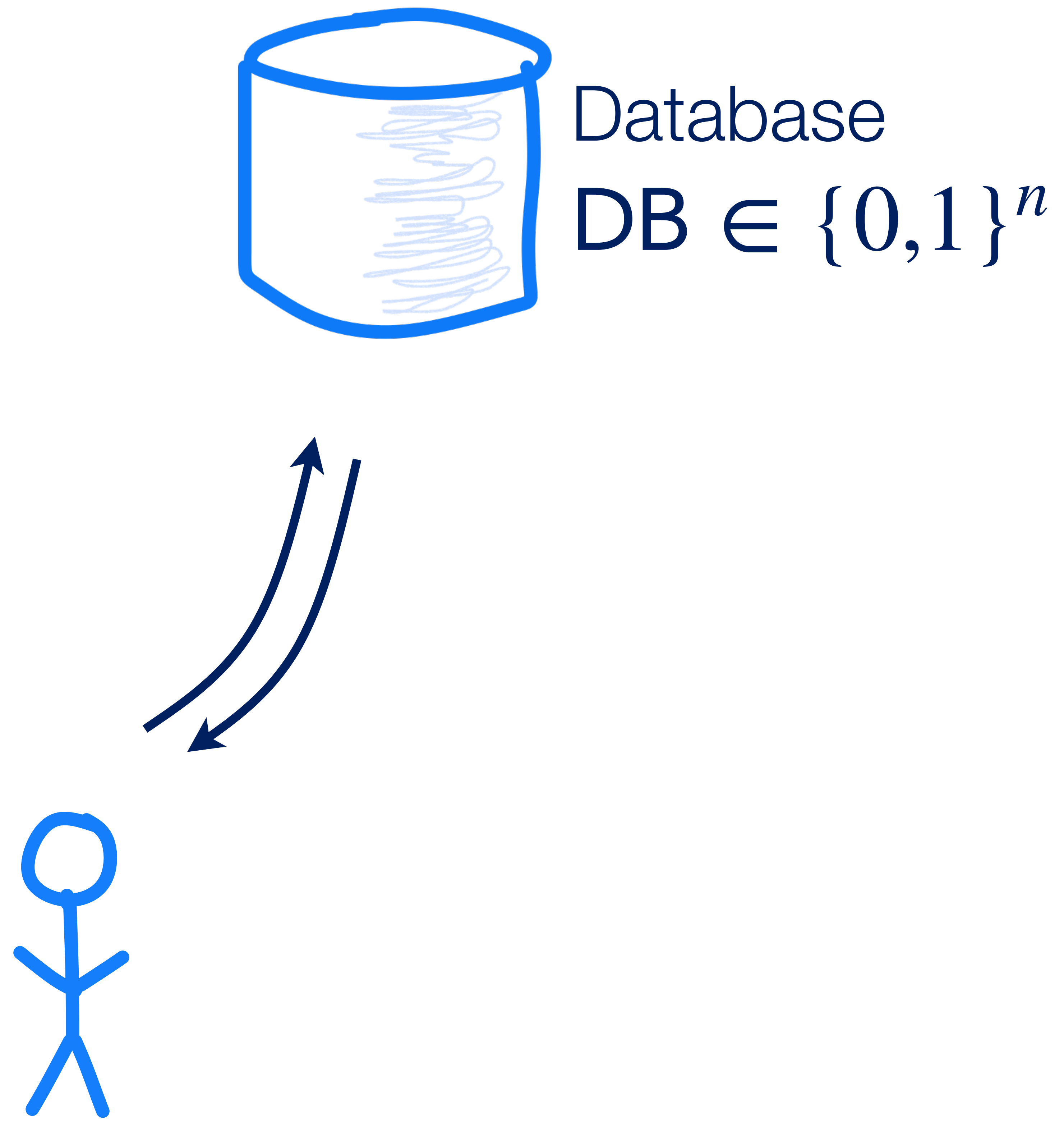


Database

DB $\in \{0,1\}^n$

Homomorphic Encryption for Single-Server PIR

- PIR: computing $\mathsf{DB}[i]$ without revealing i

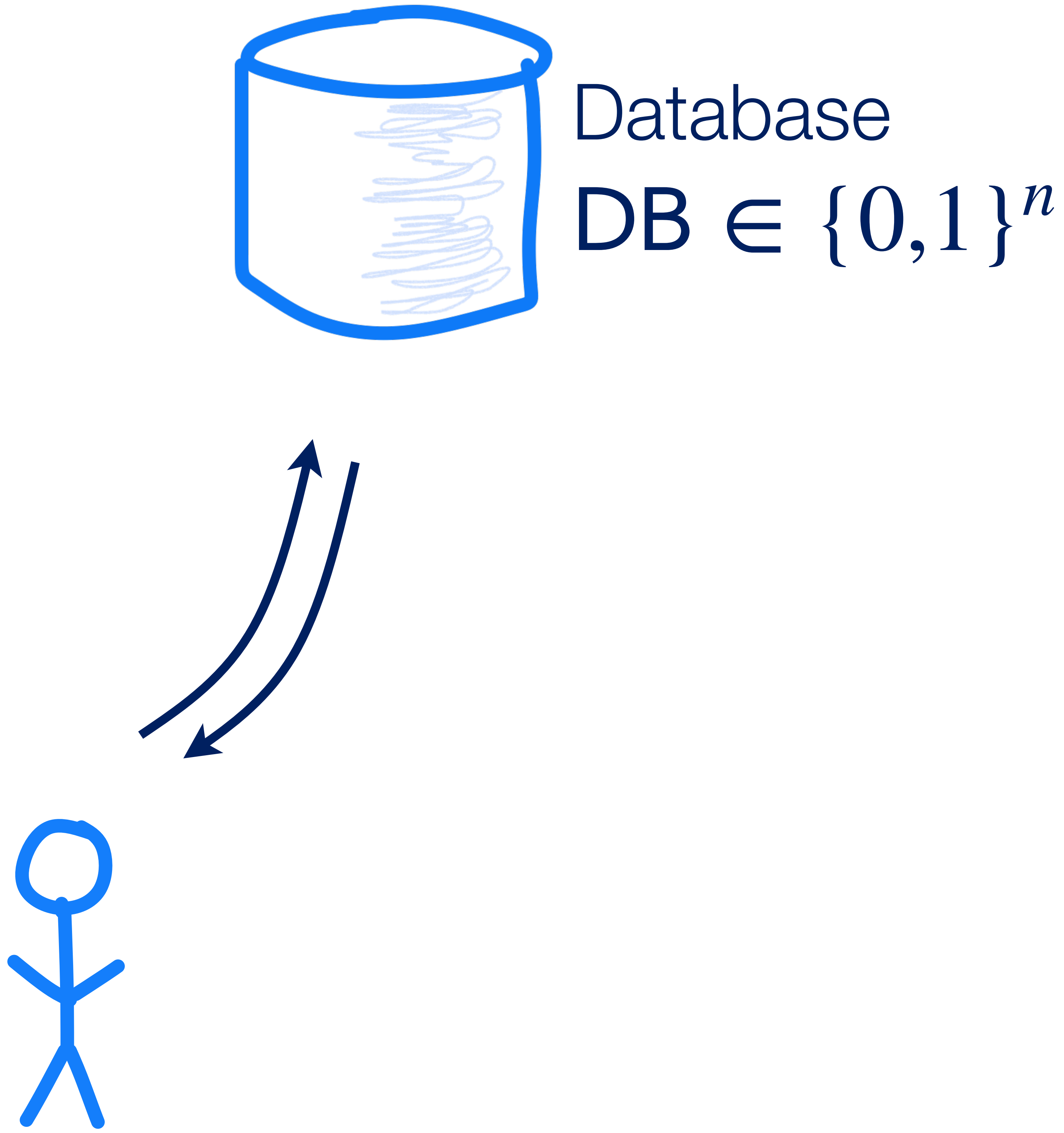


Database

$\mathsf{DB} \in \{0,1\}^n$

Homomorphic Encryption for Single-Server PIR

- PIR: computing $\mathsf{DB}[i]$ without revealing i
- Naive FHE solution:

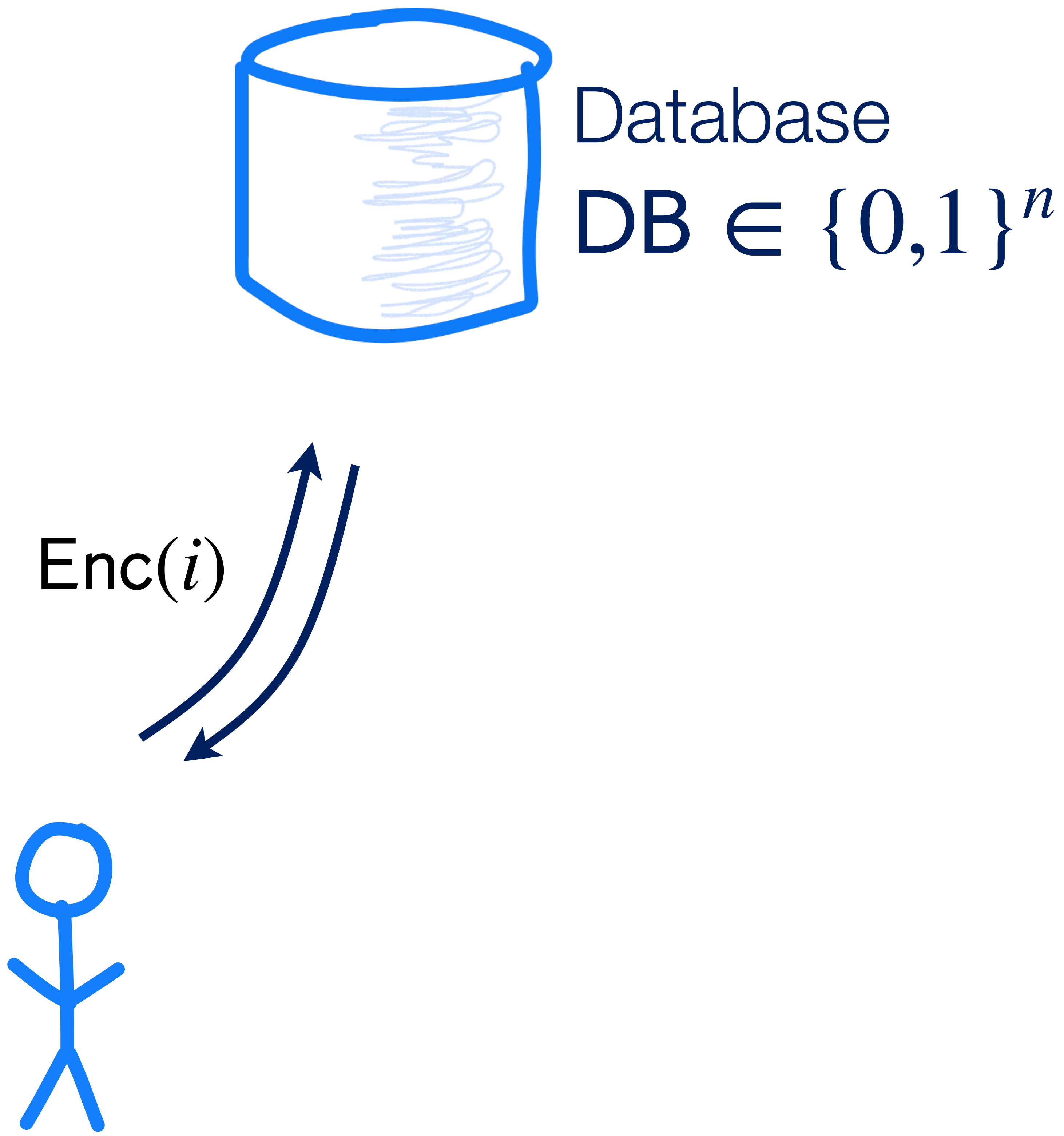


Database

$\mathsf{DB} \in \{0,1\}^n$

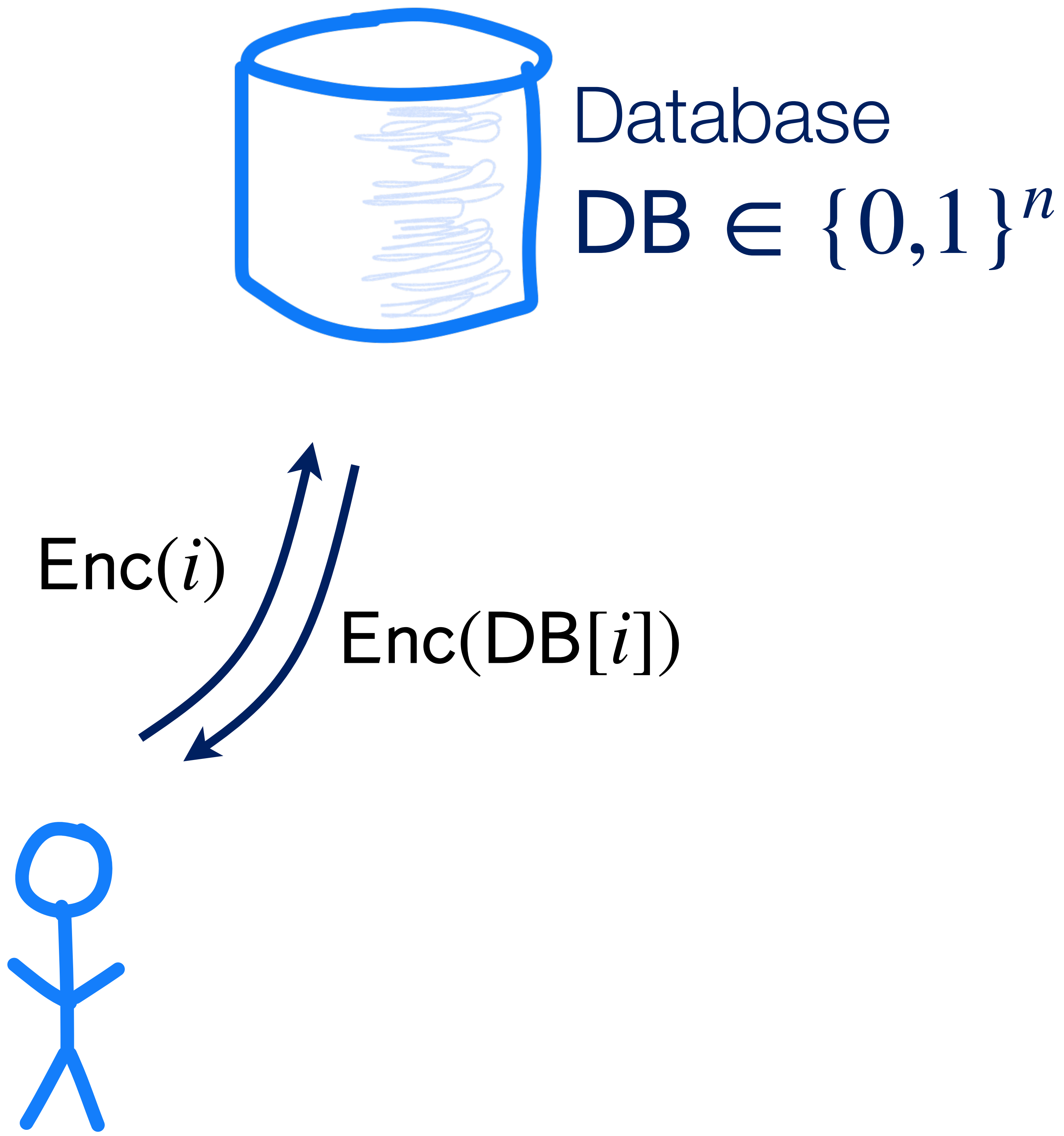
Homomorphic Encryption for Single-Server PIR

- PIR: computing $\mathsf{DB}[i]$ without revealing i
- Naive FHE solution:
  - Query: $\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(i)$



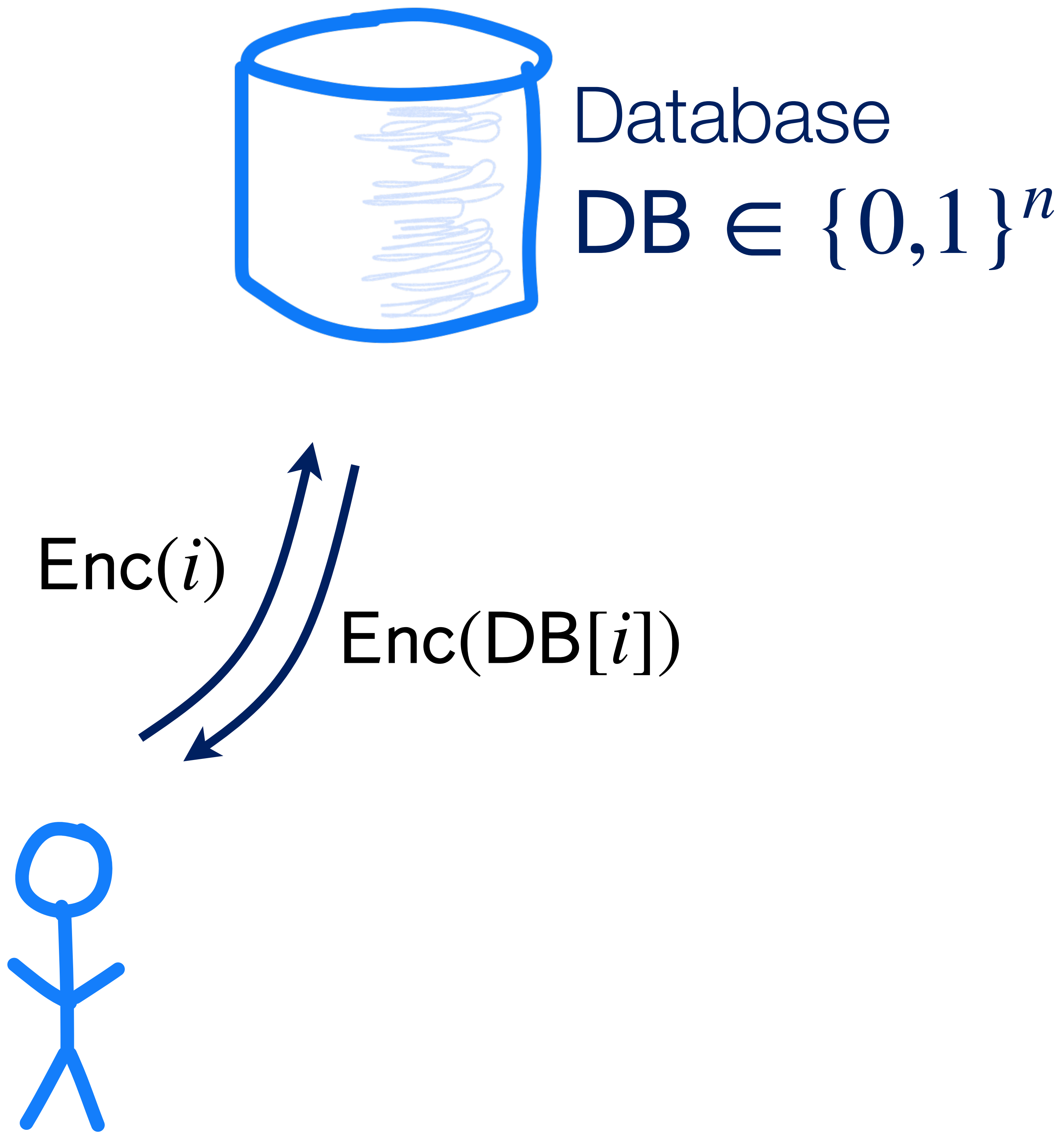
Homomorphic Encryption for Single-Server PIR

- PIR: computing $\mathsf{DB}[i]$ without revealing i
- Naive FHE solution:
  - Query: $\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(i)$
  - Server answer: $\mathsf{FHE.Eval}(\mathsf{ct}, \mathsf{DB}[\,\cdot\,])$



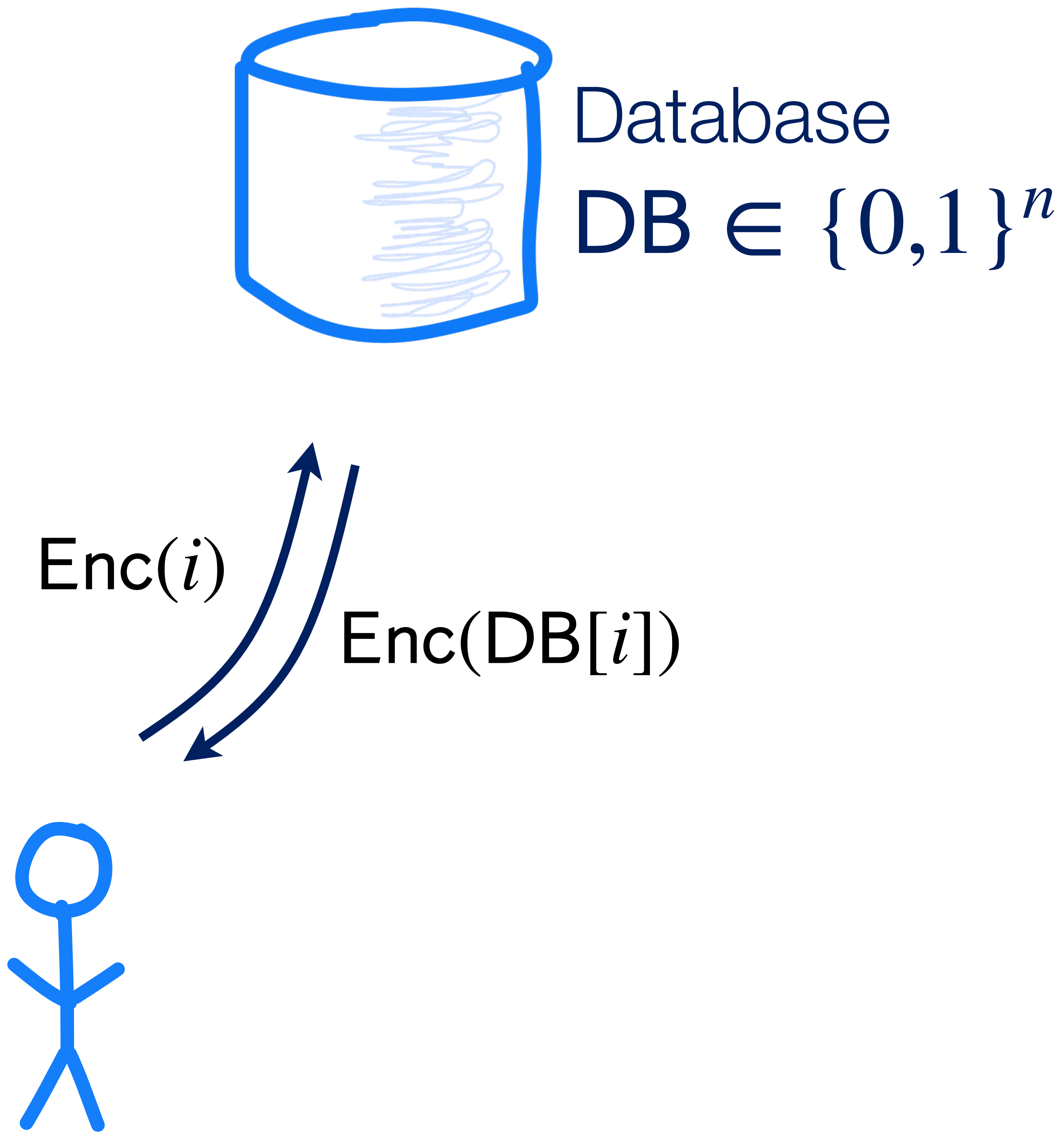
Homomorphic Encryption for Single-Server PIR

- PIR: computing $\mathsf{DB}[i]$ without revealing i
- Naive FHE solution:
  - Query: $\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(i)$
  - Server answer: $\mathsf{FHE.Eval}(\mathsf{ct}, \mathsf{DB}[\cdot])$
  - User decrypts to learn $\mathsf{DB}[i]$



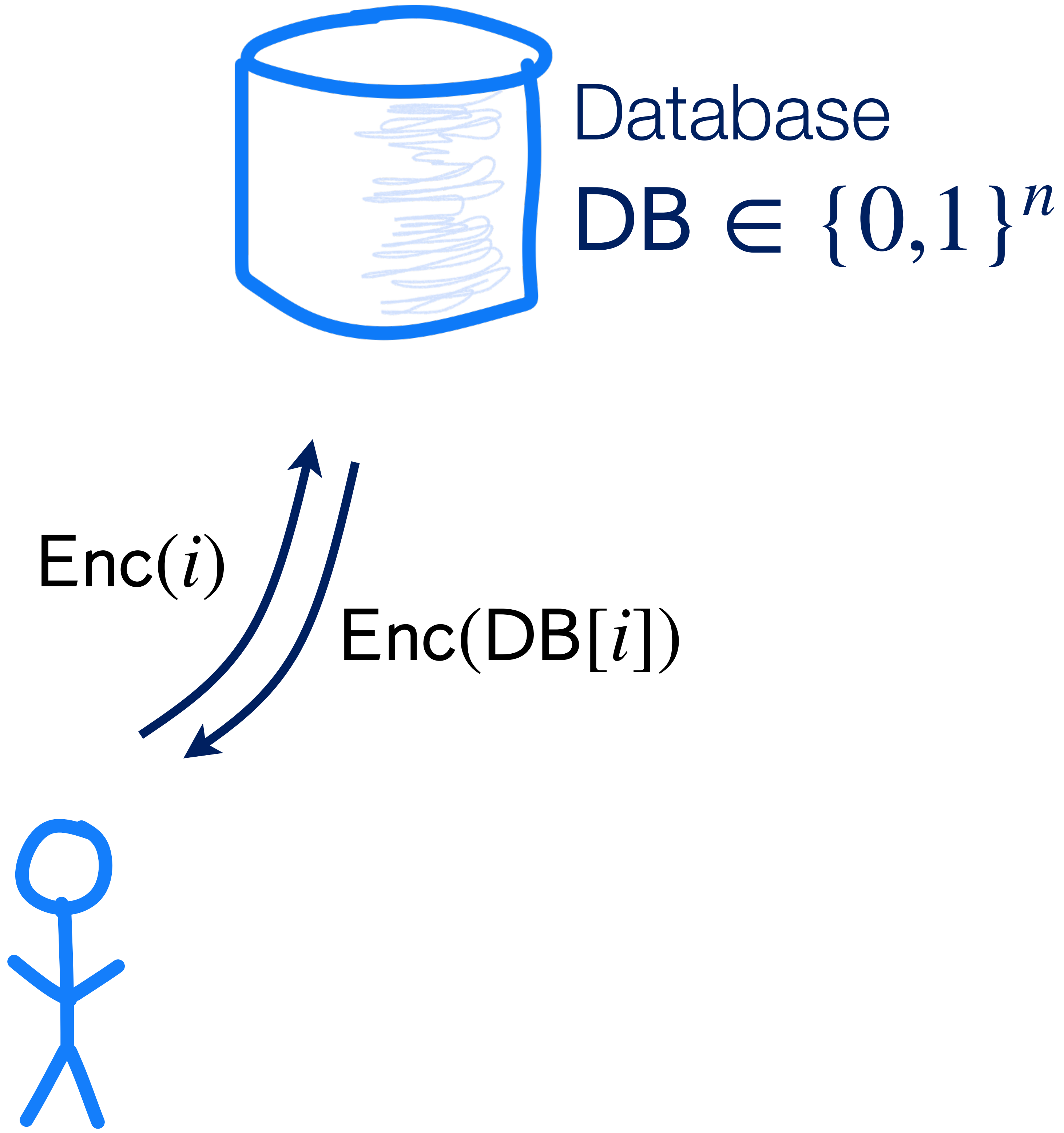
Homomorphic Encryption for Single-Server PIR

- PIR: computing $\mathsf{DB}[i]$ without revealing i
- Naive FHE solution:
  - Query: $\mathsf{ct} \leftarrow \mathsf{FHE}.\mathsf{Enc}(i)$
  - Server answer: $\mathsf{FHE}.\mathsf{Eval}(\mathsf{ct}, \mathsf{DB}[\,\cdot\,])$
  - User decrypts to learn $\mathsf{DB}[i]$
- But $\mathsf{DB}[\,\cdot\,]$ is a size $O(n)$ circuit!

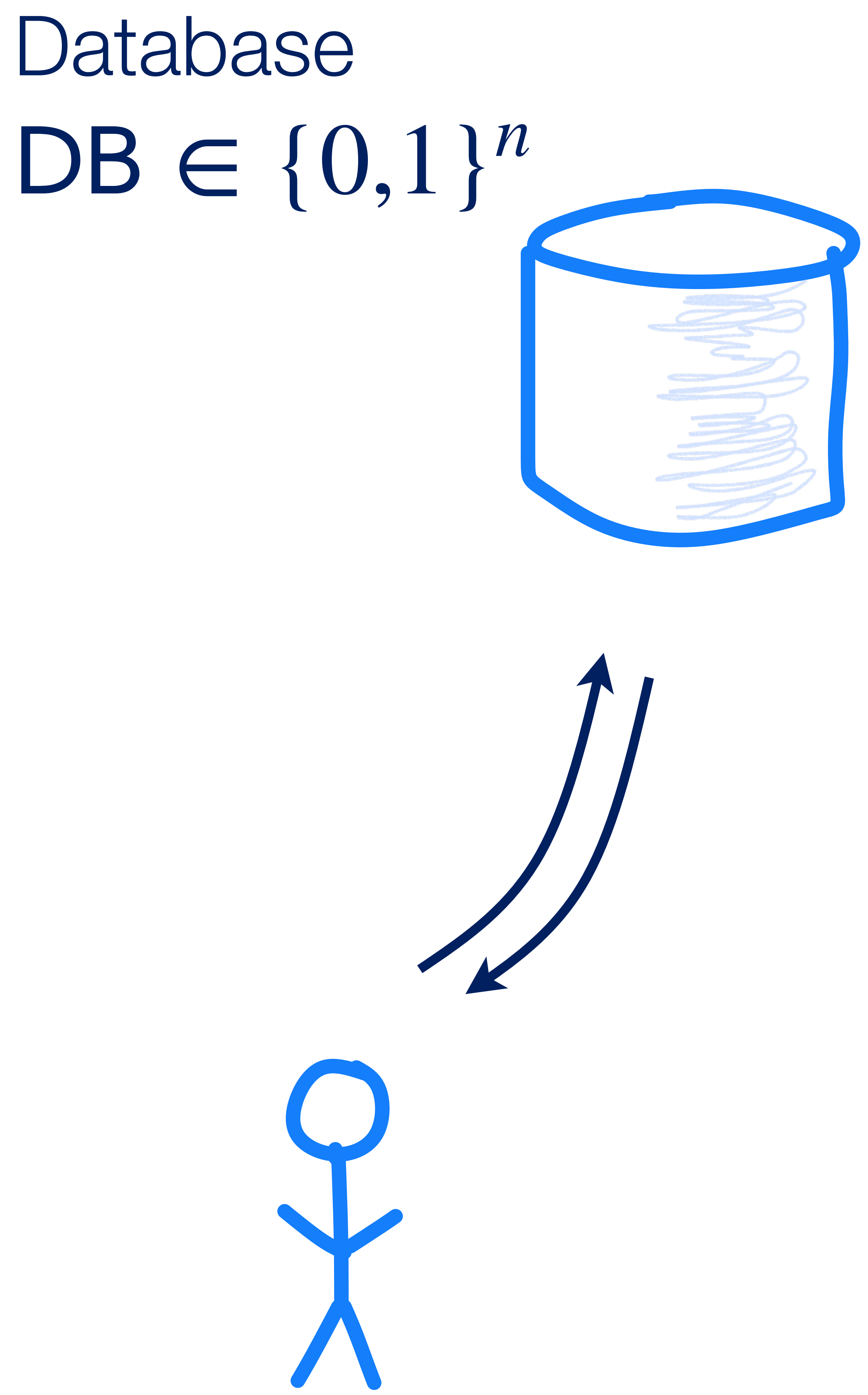


Homomorphic Encryption for Single-Server PIR

- PIR: computing $\mathsf{DB}[i]$ without revealing i
- Naive FHE solution:
  - Query: $\mathsf{ct} \leftarrow \mathsf{FHE}.\mathsf{Enc}(i)$
  - Server answer: $\mathsf{FHE}.\mathsf{Eval}(\mathsf{ct}, \mathsf{DB}[\,\cdot\,])$
  - User decrypts to learn $\mathsf{DB}[i]$
- But $\mathsf{DB}[\,\cdot\,]$ is a size $O(n)$ circuit!
  - Server time per query: $O(n)$



Homomorphic Encryption for 2-Server PIR

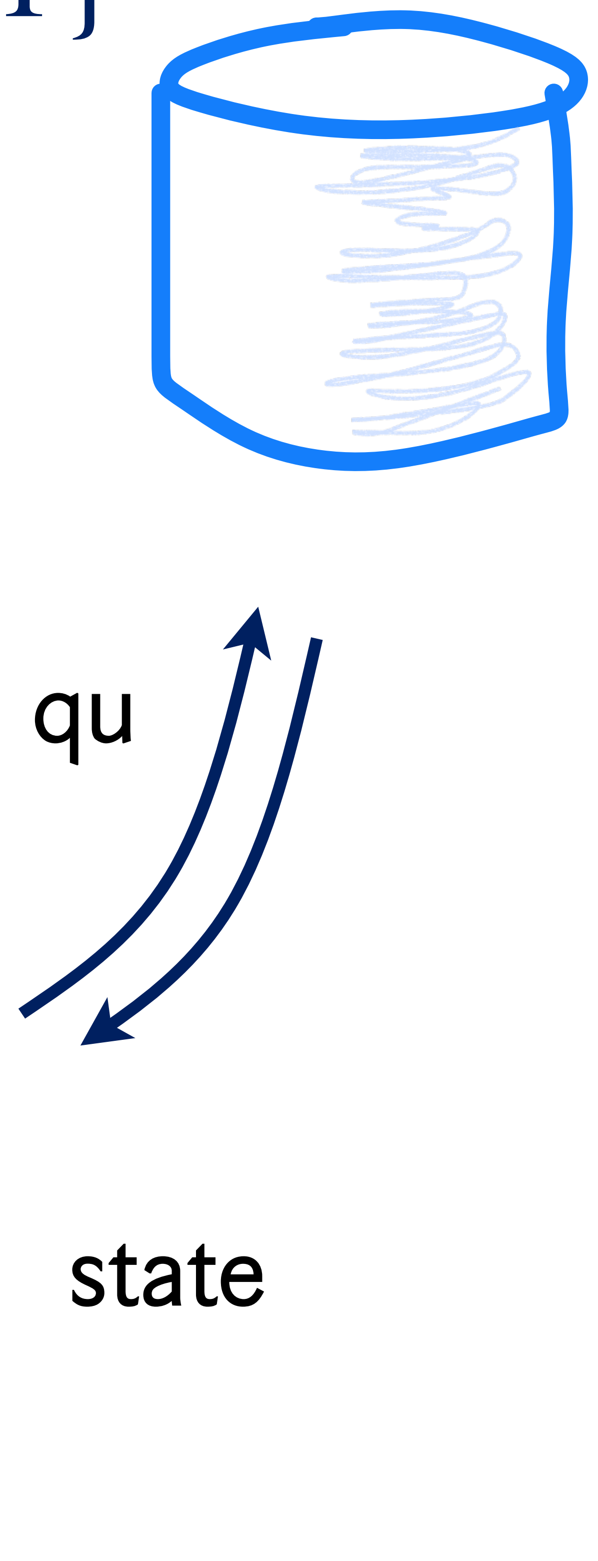


Homomorphic Encryption for 2-Server PIR

- Three phases:
    - Query: $\mathsf{state}, \mathsf{qu}_1 \leftarrow \mathsf{Query}(i)$

Database

$\mathsf{DB} \in \{0,1\}^n$

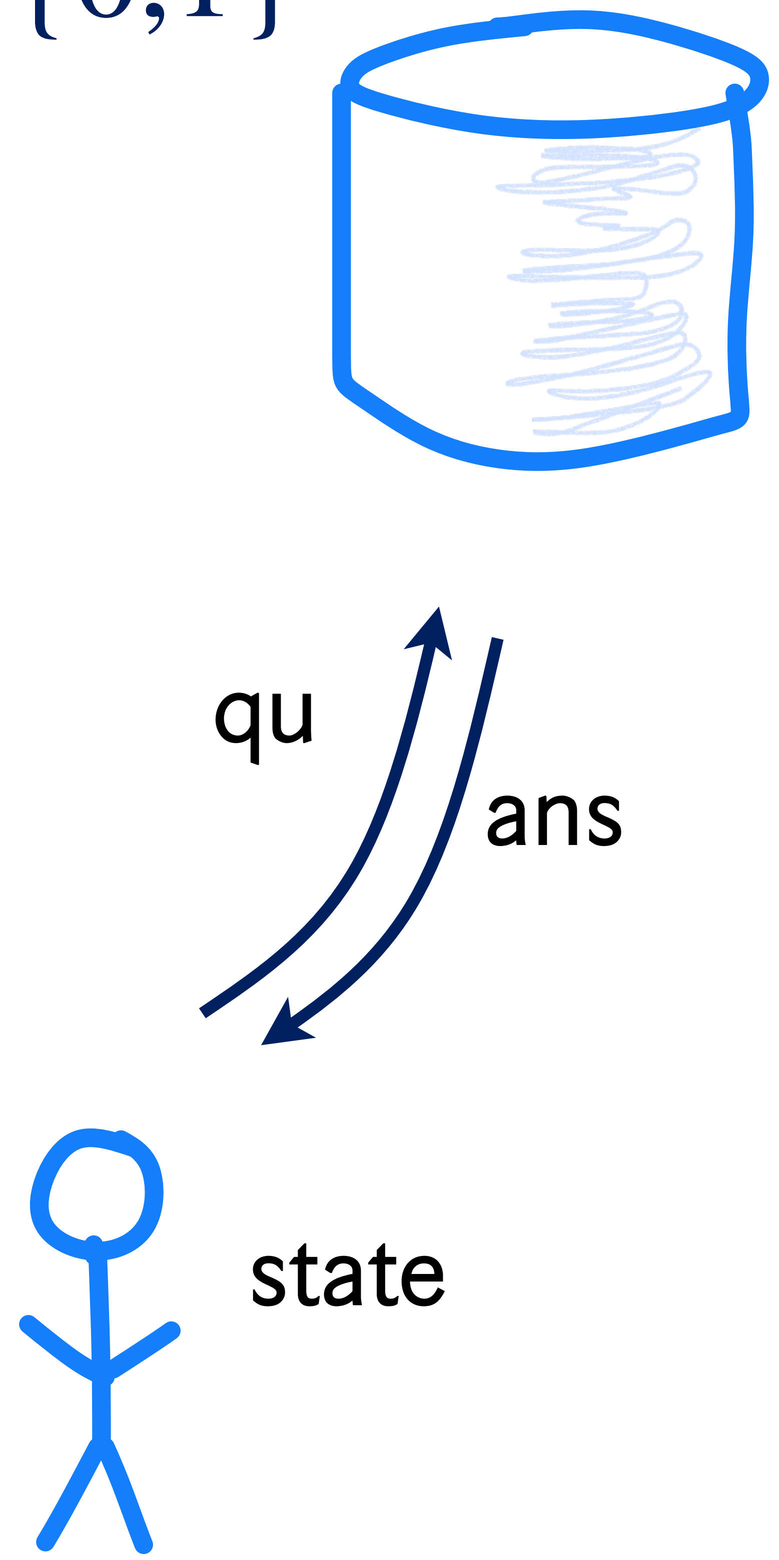


Homomorphic Encryption for 2-Server PIR

- Three phases:
  - Query: $\mathsf{state}, \mathsf{qu}_1 \leftarrow \mathsf{Query}(i)$
  - Answer: $\mathsf{ans}_1 = \mathsf{Answer}(\mathsf{DB}, \mathsf{qu}_1)$

Database

$\mathsf{DB} \in \{0,1\}^n$

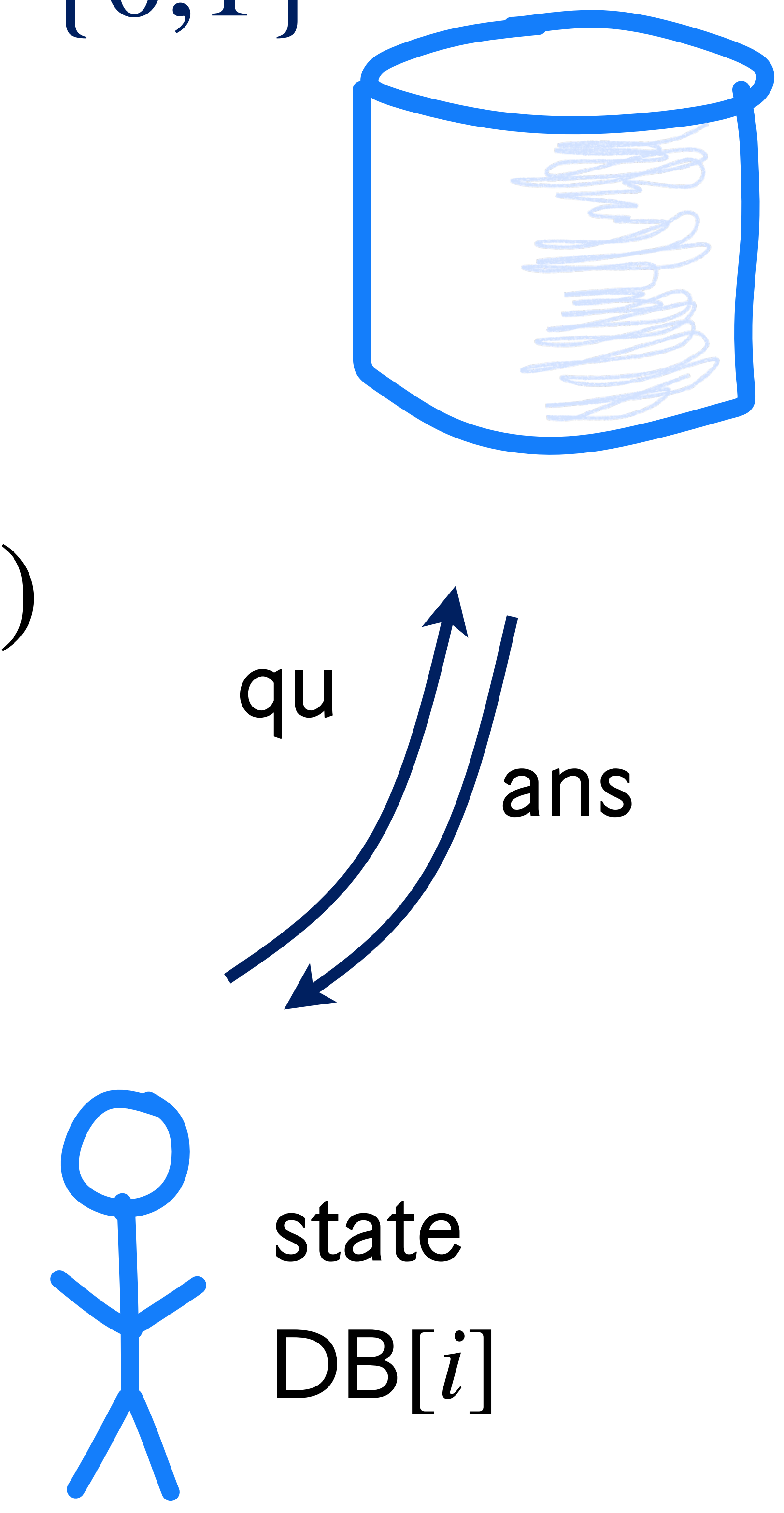


Homomorphic Encryption for 2-Server PIR

- Three phases:
  - Query: $\mathsf{state}, \mathsf{qu}_1 \leftarrow \mathsf{Query}(i)$
  - Answer: $\mathsf{ans}_1 = \mathsf{Answer}(\mathsf{DB}, \mathsf{qu}_1)$
  - Reconstruction: $\mathsf{DB}[i] = \mathsf{Reconstruct}(\mathsf{state}, \mathsf{ans}_1) + \mathsf{Reconstruct}(\mathsf{state}, \mathsf{ans}_2)$

Database

$\mathsf{DB} \in \{0,1\}^n$

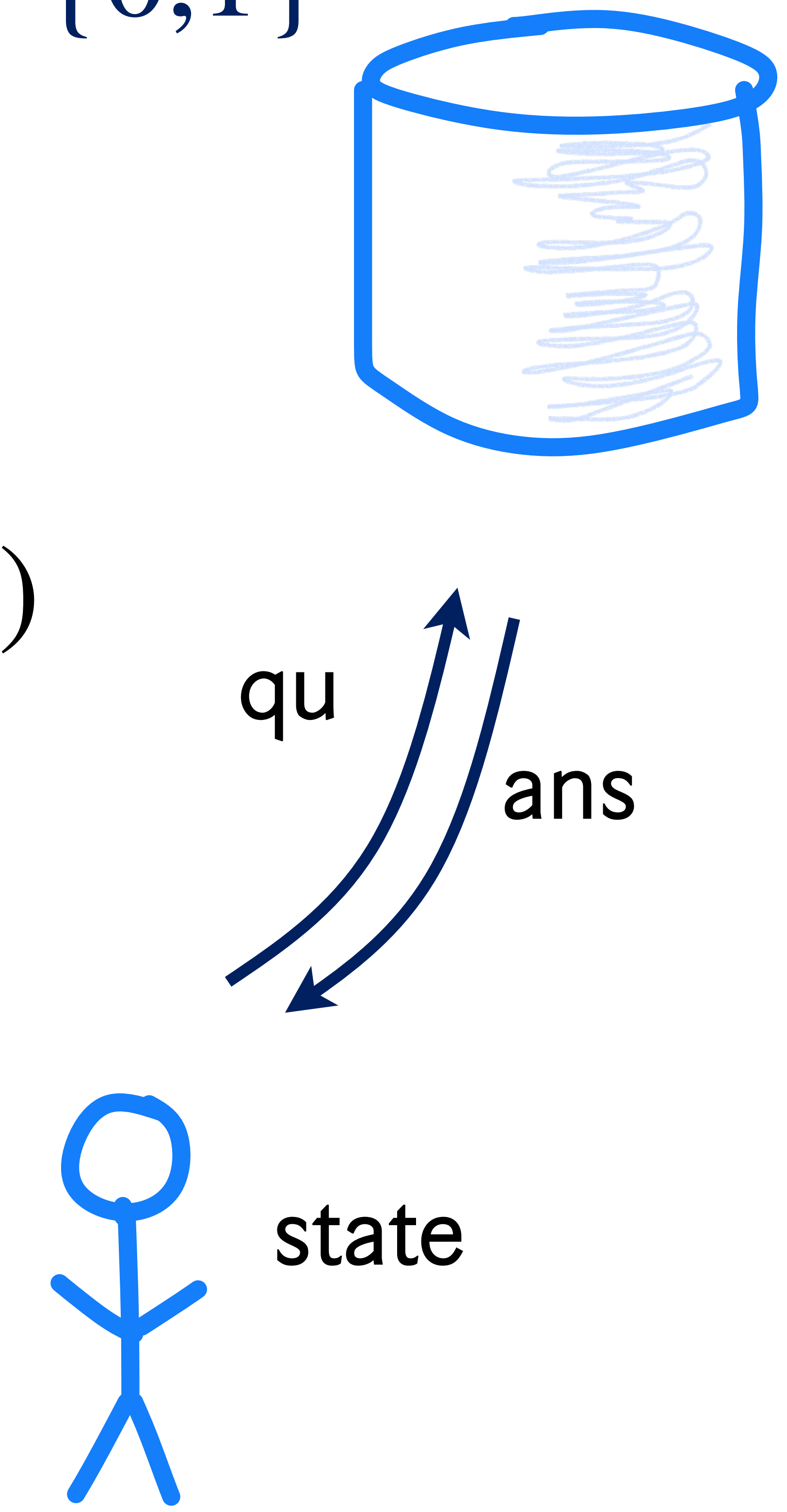


Homomorphic Encryption for 2-Server PIR

- Three phases:
  - Query: $\mathsf{state}, \mathsf{qu}_1 \leftarrow \mathsf{Query}(i)$
  - Answer: $\mathsf{ans}_1 = \mathsf{Answer}(\mathsf{DB}, \mathsf{qu}_1)$
  - Reconstruction: $\mathsf{DB}[i] = \mathsf{Reconstruct}(\mathsf{state}, \mathsf{ans}_1) + \mathsf{Reconstruct}(\mathsf{state}, \mathsf{ans}_2)$
- Observation (coming up): **Reconstruct** is a small circuit!

Database

$\mathsf{DB} \in \{0,1\}^n$

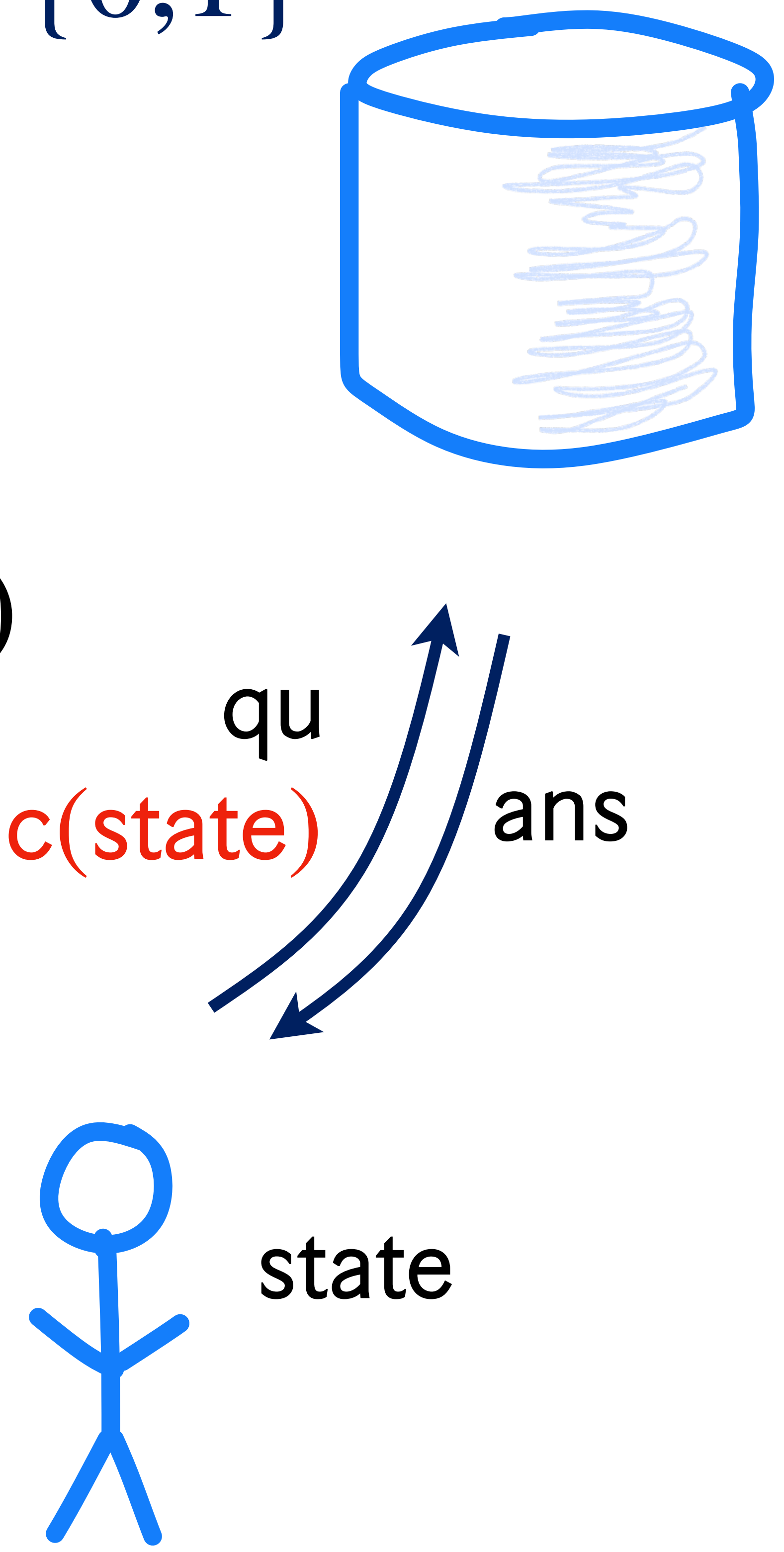


Homomorphic Encryption for 2-Server PIR

- Three phases:
  - Query: $\mathsf{state}, \mathsf{qu}_1 \leftarrow \mathsf{Query}(i)$
  - Answer: $\mathsf{ans}_1 = \mathsf{Answer}(\mathsf{DB}, \mathsf{qu}_1)$
  - Reconstruction: $\mathsf{DB}[i] = \mathsf{Reconstruct}(\mathsf{state}, \mathsf{ans}_1) + \mathsf{Reconstruct}(\mathsf{state}, \mathsf{ans}_2)$
- Observation (coming up): $\mathsf{Reconstruct}$ is a small circuit!
  - Include $\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{state})$ in query

Database

$\mathsf{DB} \in \{0,1\}^n$

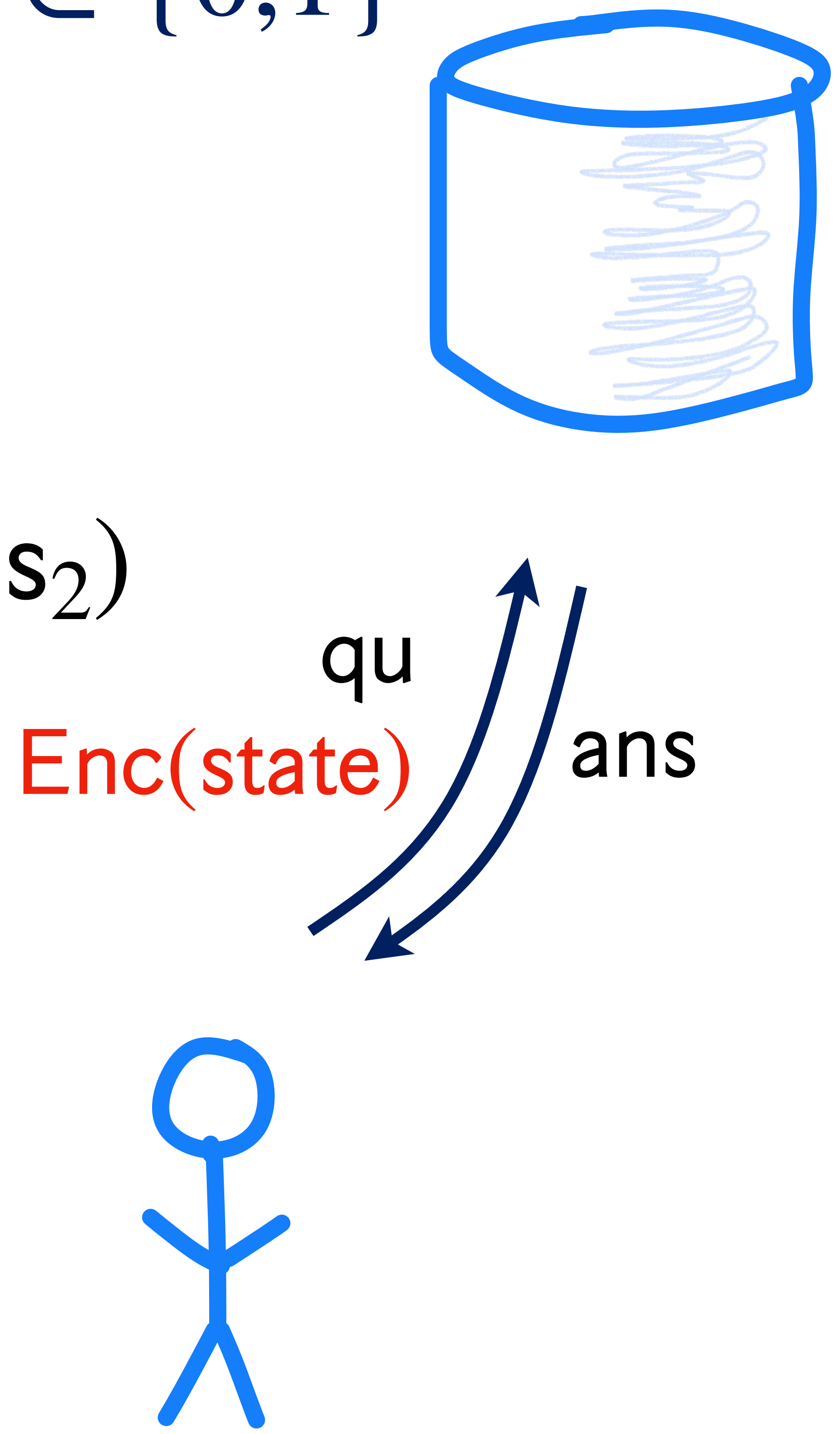


Homomorphic Encryption for 2-Server PIR

- Three phases:
  - Query: $\mathsf{state}, \mathsf{qu}_1 \leftarrow \mathsf{Query}(i)$
  - Answer: $\mathsf{ans}_1 = \mathsf{Answer}(\mathsf{DB}, \mathsf{qu}_1)$
  - Reconstruction: $\mathsf{DB}[i] = \mathsf{Reconstruct}(\mathsf{state}, \mathsf{ans}_1) + \mathsf{Reconstruct}(\mathsf{state}, \mathsf{ans}_2)$
- Observation (coming up): **Reconstruct** is a small circuit!
  - Include $\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{state})$ in query

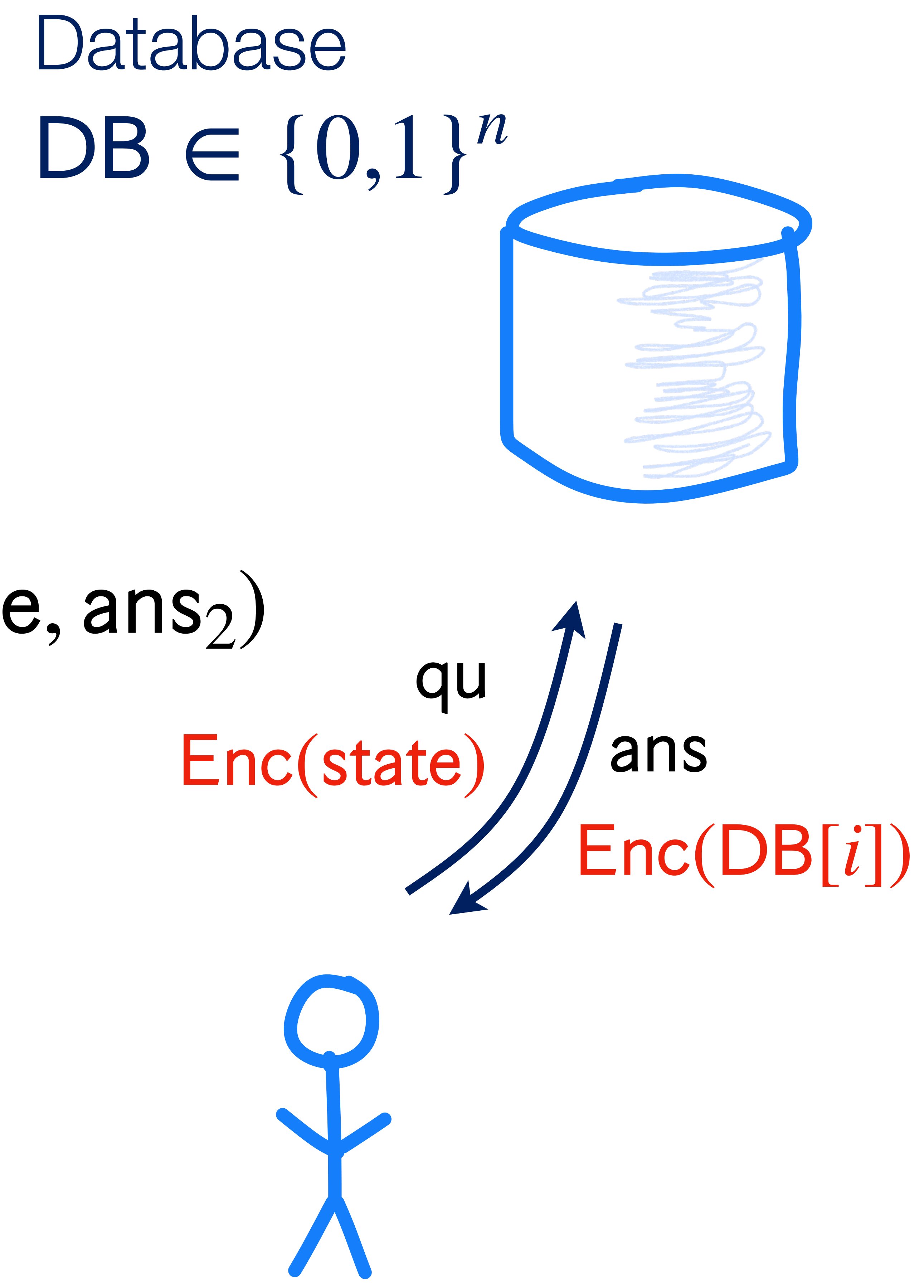
Database

$\mathsf{DB} \in \{0,1\}^n$



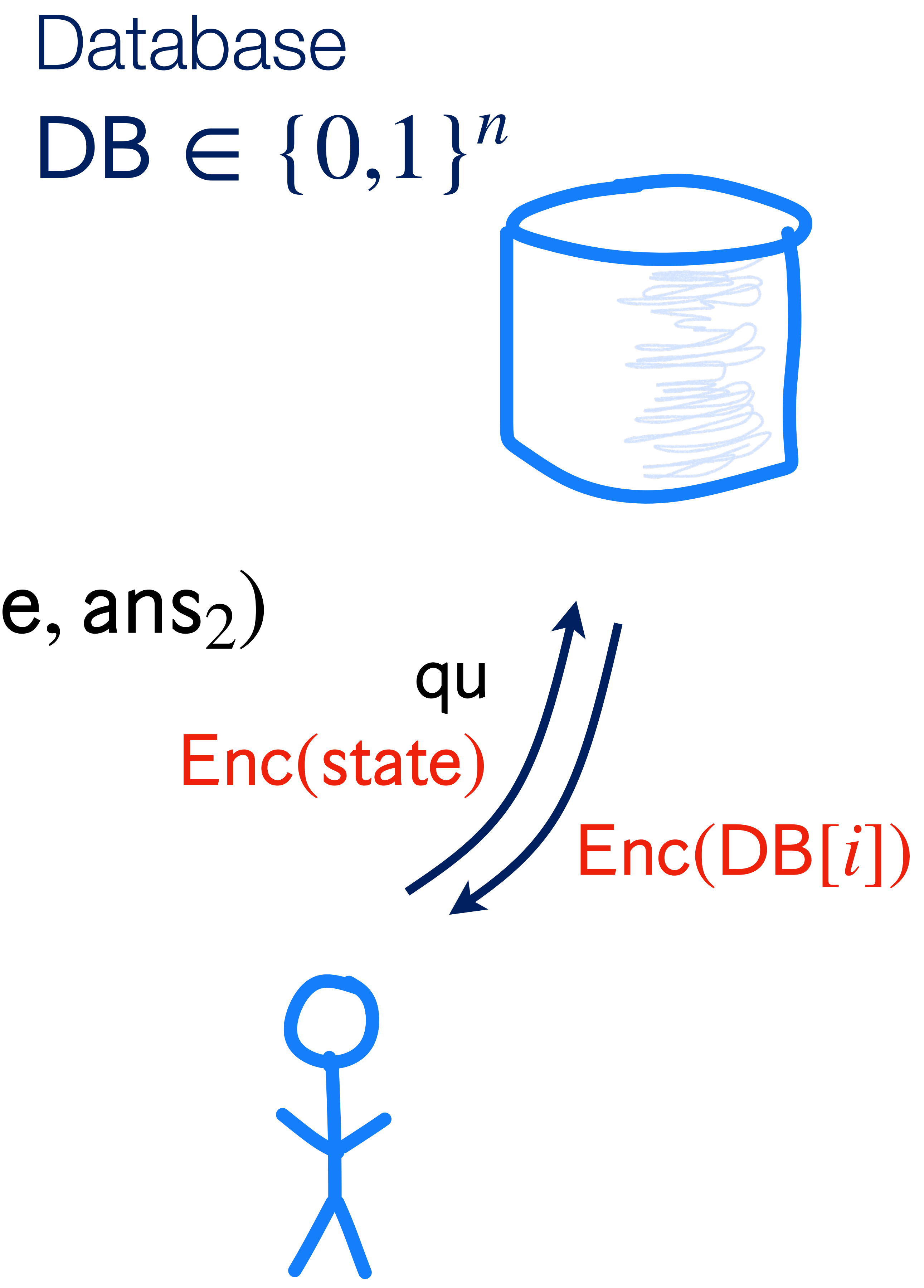
Homomorphic Encryption for 2-Server PIR

- Three phases:
  - Query: $\mathsf{state}, \mathsf{qu}_1 \leftarrow \mathsf{Query}(i)$
  - Answer: $\mathsf{ans}_1 = \mathsf{Answer}(\mathsf{DB}, \mathsf{qu}_1)$
  - Reconstruction: $\mathsf{DB}[i] = \mathsf{Reconstruct}(\mathsf{state}, \mathsf{ans}_1) + \mathsf{Reconstruct}(\mathsf{state}, \mathsf{ans}_2)$
- Observation (coming up): $\mathsf{Reconstruct}$ is a small circuit!
  - Include $\mathsf{ct} \leftarrow \mathsf{FHE}.\mathsf{Enc}(\mathsf{state})$ in query
  - Server i computes ans_i then $\mathsf{FHE}.\mathsf{Eval}(\mathsf{ct}, \mathsf{Reconstruct}(\mathsf{ans}_i, \cdot\,))$



Homomorphic Encryption for 2-Server PIR

- Three phases:
  - Query: $\mathsf{state}, \mathsf{qu}_1 \leftarrow \mathsf{Query}(i)$
  - Answer: $\mathsf{ans}_1 = \mathsf{Answer}(\mathsf{DB}, \mathsf{qu}_1)$
  - Reconstruction: $\mathsf{DB}[i] = \mathsf{Reconstruct}(\mathsf{state}, \mathsf{ans}_1) + \mathsf{Reconstruct}(\mathsf{state}, \mathsf{ans}_2)$
- Observation (coming up): $\mathsf{Reconstruct}$ is a small circuit!
  - Include $\mathsf{ct} \leftarrow \mathsf{FHE}.\mathsf{Enc}(\mathsf{state})$ in query
  - Server i computes ans_i then $\mathsf{FHE}.\mathsf{Eval}(\mathsf{ct}, \mathsf{Reconstruct}(\mathsf{ans}_i, \cdot\,))$

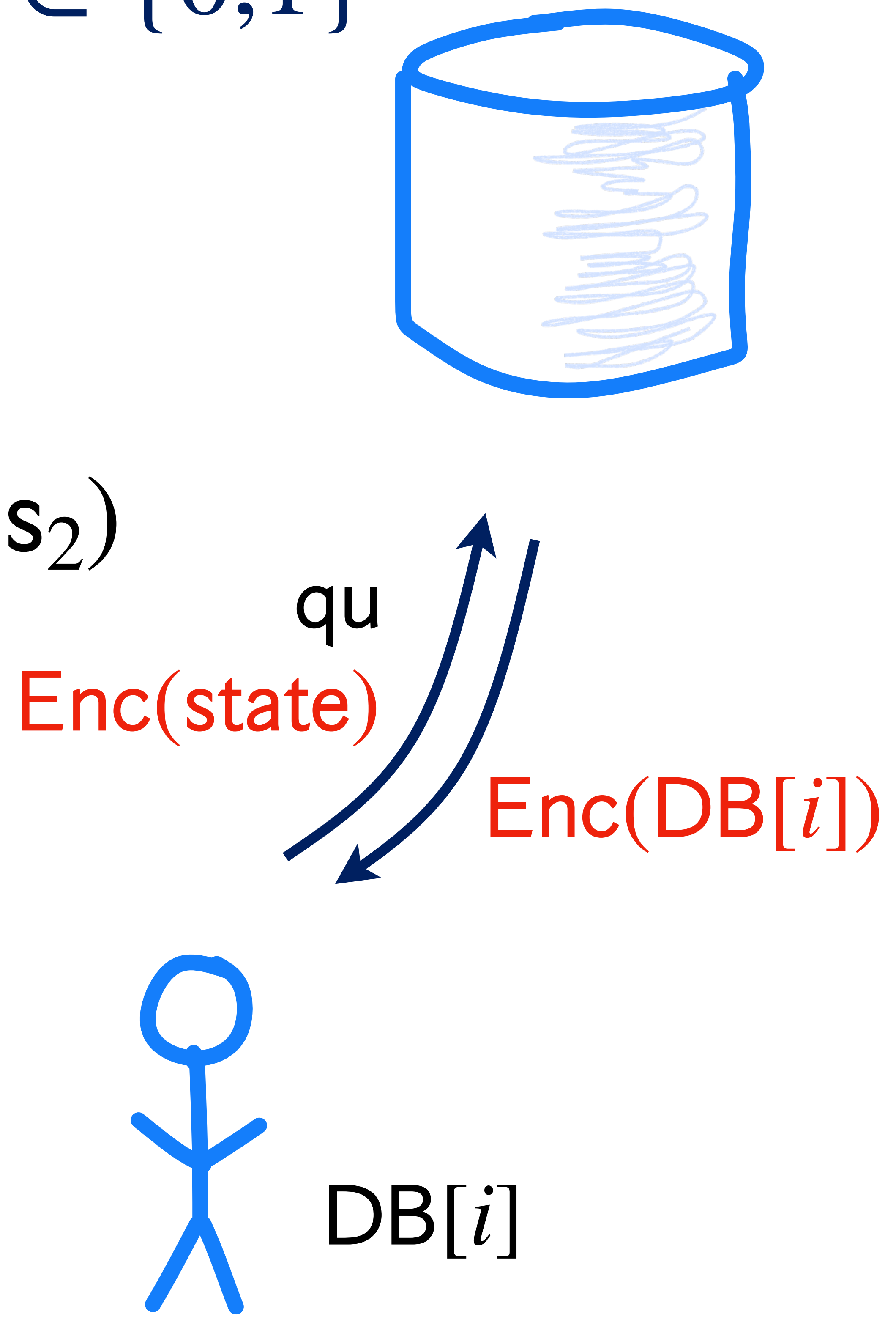


Homomorphic Encryption for 2-Server PIR

- Three phases:
  - Query: $\mathsf{state}, \mathsf{qu}_1 \leftarrow \mathsf{Query}(i)$
  - Answer: $\mathsf{ans}_1 = \mathsf{Answer}(\mathsf{DB}, \mathsf{qu}_1)$
  - Reconstruction: $\mathsf{DB}[i] = \mathsf{Reconstruct}(\mathsf{state}, \mathsf{ans}_1) + \mathsf{Reconstruct}(\mathsf{state}, \mathsf{ans}_2)$
- Observation (coming up): $\mathsf{Reconstruct}$ is a small circuit!
  - Include $\mathsf{ct} \leftarrow \mathsf{FHE}.\mathsf{Enc}(\mathsf{state})$ in query
  - Server i computes ans_i then $\mathsf{FHE}.\mathsf{Eval}(\mathsf{ct}, \mathsf{Reconstruct}(\mathsf{ans}_i, \cdot\,))$
  - Client decrypts!

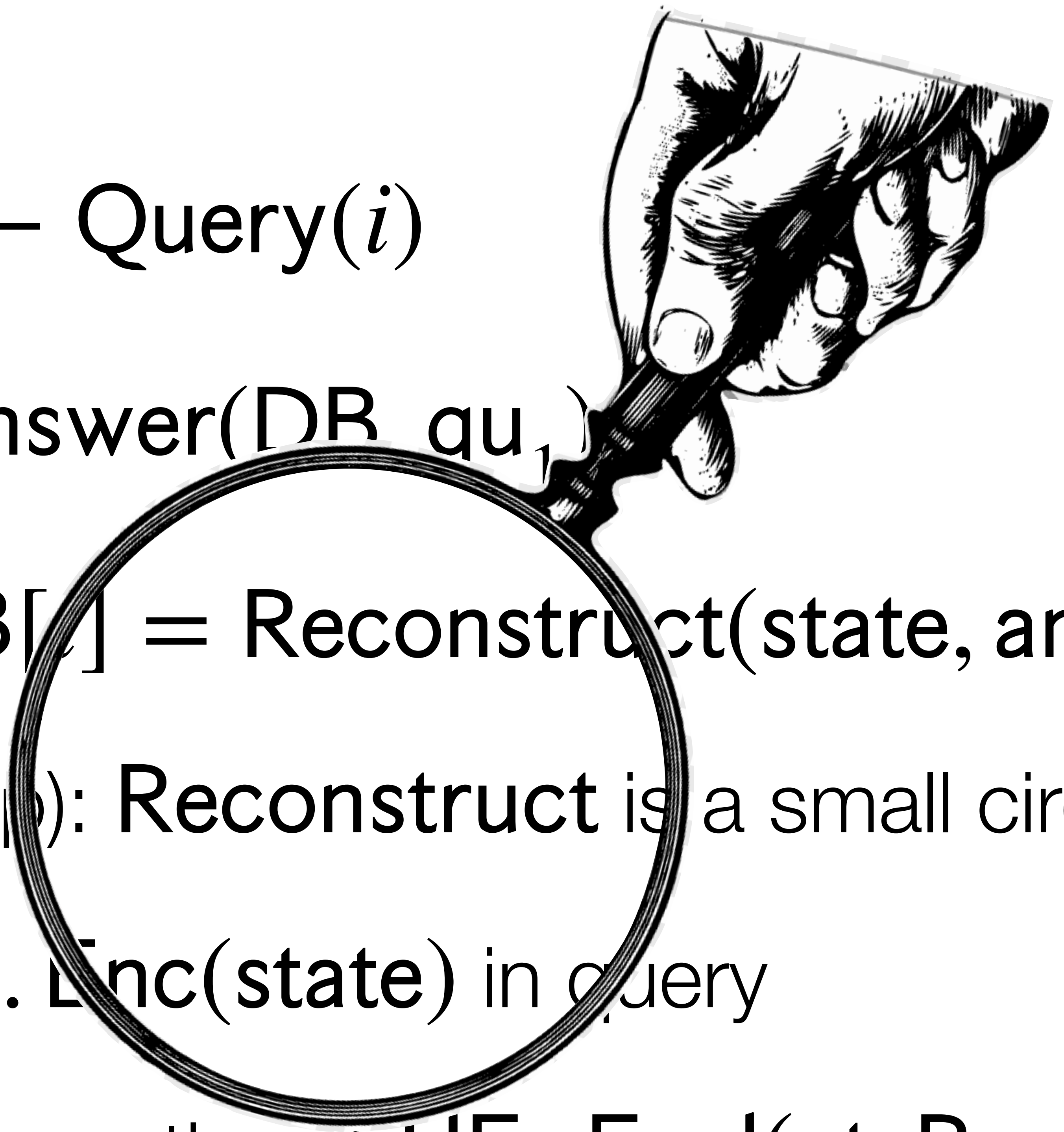
Database

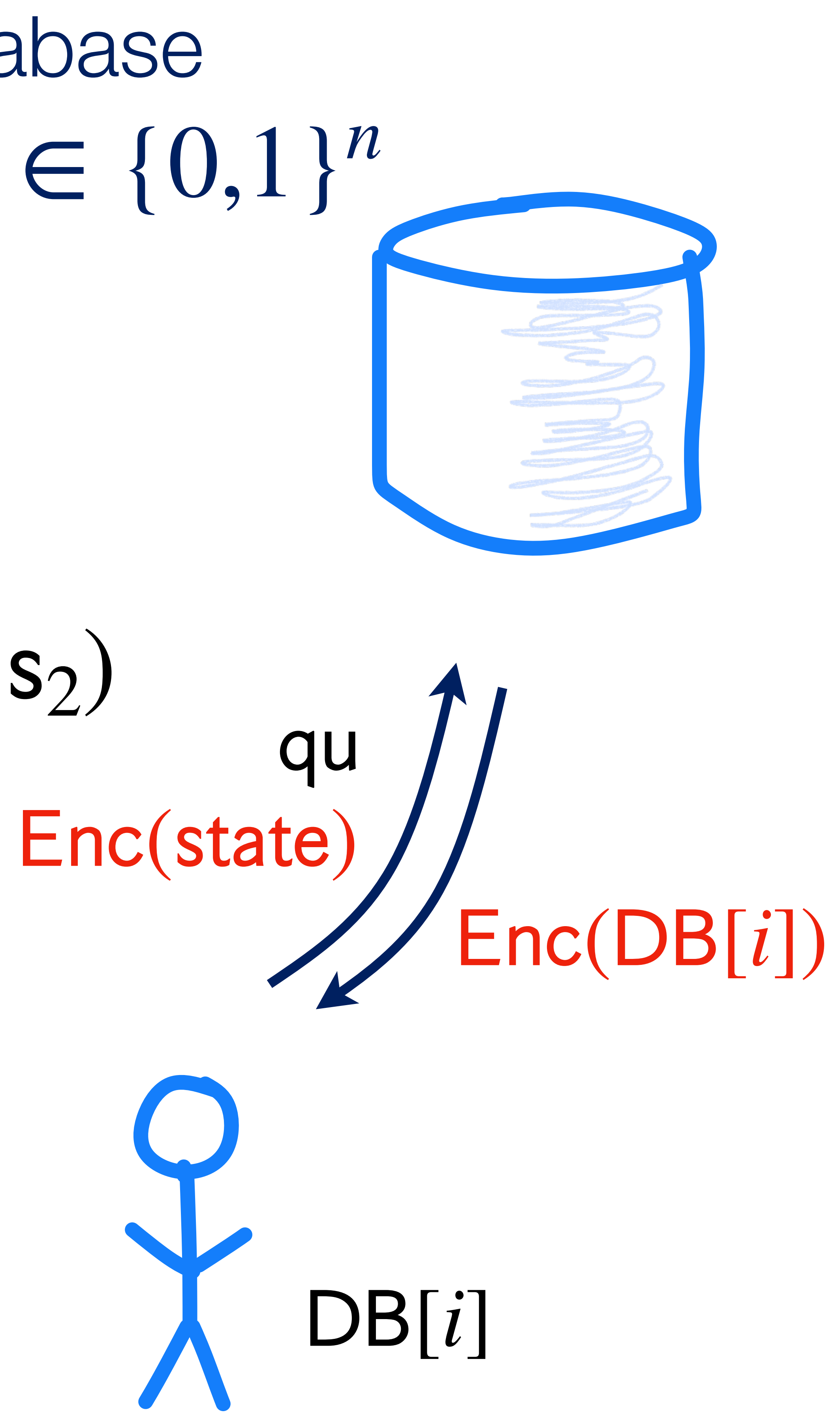
$\mathsf{DB} \in \{0,1\}^n$



Homomorphic Encryption for 2-Server PIR

- Three phases:
  - Query: $\mathsf{state}, \mathsf{qu}_1 \leftarrow \mathsf{Query}(i)$
  - Answer: $\mathsf{ans}_1 = \mathsf{Answer}(\mathsf{DB}, \mathsf{qu}_1)$
  - Reconstruction: $\mathsf{DB}[i] = \mathsf{Reconstruct}(\mathsf{state}, \mathsf{ans}_1) + \mathsf{Reconstruct}(\mathsf{state}, \mathsf{ans}_2)$
- Observation (coming up): **Reconstruct** is a small circuit!
  - Include $\mathsf{ct} \leftarrow \mathsf{FHE}.\mathsf{Enc}(\mathsf{state})$ in query
  - Server i computes ans_i then $\mathsf{FHE}.\mathsf{Eval}(\mathsf{ct}, \mathsf{Reconstruct}(\mathsf{ans}_i, \cdot\,))$
  - Client decrypts!





Database

$\mathsf{DB} \in \{0,1\}^n$

Our PIR Reconstruction

Cheatsheet

$m \approx \log n$

$D \approx m/2$

Our PIR Reconstruction

Cheatsheet

$m \approx \log n$

$D \approx m/2$

- $f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$ of degree D

Our PIR Reconstruction

Cheatsheet

$m \approx \log n$

$D \approx m/2$

- $f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$ of degree D
- Queries: $\mathbf{r}, \mathbf{r} + \mathbf{p}$
  - $\mathsf{state} = \mathbf{p}$

Our PIR Reconstruction

Cheatsheet

$m \approx \log n$

$D \approx m/2$

- $f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$ of degree D
- Queries: $\mathbf{r}, \mathbf{r} + \mathbf{p}$
  - $\mathsf{state} = \mathbf{p}$
- Answers: $\left\{f(\mathbf{x} + \mathbf{e}) : \mathbf{x} \in \{\mathbf{r}, \mathbf{r} + \mathbf{p}\}, \|\mathbf{e}\| \leq D/2\right\}$

Our PIR Reconstruction

Cheatsheet
$m \approx \log n$
$D \approx m/2$

- $f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$ of degree D
- Queries: $\mathbf{r}, \mathbf{r} + \mathbf{p}$
  - state $= \mathbf{p}$
- Answers: $\left\{ f(\mathbf{x} + \mathbf{e}) : \mathbf{x} \in \{\mathbf{r}, \mathbf{r} + \mathbf{p}\}, \|\mathbf{e}\| \le D/2 \right\}$
- Reconstruction:

$$\mathsf{DB}_i = f_{\mathsf{DB}}(\mathbf{p}) = \sum_{\substack{\| \mathbf{e} \| \ \le \lfloor D/2 \rfloor \\ \mathbf{e} \le \mathbf{p}}} \left(\underbrace{f(\mathbf{r} + \mathbf{e})}_{\text{from server 1's answer}} + \underbrace{f(\mathbf{p} + \mathbf{r} + \mathbf{e})}_{\text{from server 2's answer}} \right)$$

Our PIR Reconstruction

Cheatsheet

$m \approx \log n$

$D \approx m/2$

- $f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$ of degree D
- Queries: $\mathbf{r}, \mathbf{r} + \mathbf{p}$
  - state $= \mathbf{p}$
- Answers: $\left\{ f(\mathbf{x} + \mathbf{e}) : \mathbf{x} \in \{\mathbf{r}, \mathbf{r} + \mathbf{p}\}, \|\mathbf{e}\| \le D/2 \right\}$
- Reconstruction:

$$\mathsf{DB}_i = f_{\mathsf{DB}}(\mathbf{p}) = \sum_{\substack{\| \mathbf{e} \| \ \le \lfloor D/2 \rfloor \\ \mathbf{e} \le \mathbf{p}}} \left(\underbrace{f(\mathbf{r} + \mathbf{e})}_{\text{from server 1's answer}} + \underbrace{f(\mathbf{p} + \mathbf{r} + \mathbf{e})}_{\text{from server 2's answer}} \right)$$

- For each $\mathbf{e}$, $\mathbf{1}[\mathbf{e} \le \mathbf{p}] = \mathbf{p}^{\mathbf{e}} = \prod_{i \in [m]} p_i^{e_i}$, which is degree $\le D/2$ in $\mathbf{p}$

Abstract Setup

Cheatsheet

$m \approx \log n$

$D \approx m/2$

- Two polynomials $g_1, g_2 : \mathbb{F}_2^m \to \mathbb{F}_2$ of degree $D/2$ for servers 1 and 2 respectively

$$g_1(\mathbf{x}) = \sum_{\|\mathbf{e}\| \le \lfloor D/2 \rfloor} \left(\underbrace{f(\mathbf{r} + \mathbf{e})}_{\text{from server 1's answer}} \right) \mathbf{x}^{\mathbf{e}}$$

$$g_2(\mathbf{x}) = \sum_{\|\mathbf{e}\| \le \lfloor D/2 \rfloor} \left(\underbrace{f(\mathbf{p} + \mathbf{r} + \mathbf{e})}_{\text{from server 2's answer}} \right) \mathbf{x}^{\mathbf{e}}$$

Abstract Setup

Cheatsheet

$m \approx \log n$

$D \approx m/2$

Abstract Setup

Cheatsheet

$m \approx \log n$

$D \approx m/2$

- Two polynomials $g_1, g_2 : \mathbb{F}_2^m \to \mathbb{F}_2$ of degree $D/2$ for servers 1 and 2 respectively
- User's goal: evaluate $g_1(\mathbf{p}) + g_2(\mathbf{p})$ for $\|\mathbf{p}\| = D$, without revealing $\mathbf{p} \in \mathbb{F}_2^m$

Abstract Setup

Cheatsheet

$m \approx \log n$

$D \approx m/2$

- Two polynomials $g_1, g_2 : \mathbb{F}_2^m \to \mathbb{F}_2$ of degree $D/2$ for servers 1 and 2 respectively
- User's goal: evaluate $g_1(\mathbf{p}) + g_2(\mathbf{p})$ for $\|\mathbf{p}\| = D$, without revealing $\mathbf{p} \in \mathbb{F}_2^m$
- All we need to do is get server 1 to help the user evaluate $g_1(\mathbf{p})$ and likewise for server 2

Succinct PIR from FHE

Cheatsheet

$m \approx \log n$

$D \approx m/2$

$\binom{m}{\alpha m} \approx 2^{mH(\alpha)} \approx n^{H(\alpha)}$

Succinct PIR from FHE

Cheatsheet

$m \approx \log n$

$D \approx m/2$

$\binom{m}{\alpha m} \approx 2^{mH(\alpha)} \approx n^{H(\alpha)}$

- Communication: upload $m \cdot \mathsf{poly}(\lambda) = \log n \cdot \mathsf{poly}(\lambda)$, download $\mathsf{poly}(\lambda)$

Succinct PIR from FHE

Cheatsheet

$m \approx \log n$

$D \approx m/2$

$\binom{m}{\alpha m} \approx 2^{mH(\alpha)} \approx n^{H(\alpha)}$

- Communication: upload $m \cdot \mathsf{poly}(\lambda) = \log n \cdot \mathsf{poly}(\lambda)$, download $\mathsf{poly}(\lambda)$
- Server computation: same as before + evaluating a degree $D/2$ polynomial in m variables

Succinct PIR from FHE

Cheatsheet

$m \approx \log n$

$D \approx m/2$

$\binom{m}{\alpha m} \approx 2^{mH(\alpha)} \approx n^{H(\alpha)}$

- Communication: upload $m \cdot \mathsf{poly}(\lambda) = \log n \cdot \mathsf{poly}(\lambda)$, download $\mathsf{poly}(\lambda)$
- Server computation: same as before + evaluating a degree $D/2$ polynomial in m variables
  - Runtime $\approx \binom{m}{D/2} \cdot \mathsf{poly}(\lambda) \approx n^{H(1/4)} \cdot \mathsf{poly}(\lambda) \approx n^{0.82} \cdot \mathsf{poly}(\lambda)$

Succinct PIR from FHE

Cheatsheet

$m \approx \log n$

$D \approx m/2$

$\binom{m}{\alpha m} \approx 2^{mH(\alpha)} \approx n^{H(\alpha)}$

- Communication: upload $m \cdot \mathsf{poly}(\lambda) = \log n \cdot \mathsf{poly}(\lambda)$, download $\mathsf{poly}(\lambda)$
- Server computation: same as before + evaluating a degree $D/2$ polynomial in m variables
    - Runtime $\approx \binom{m}{D/2} \cdot \mathsf{poly}(\lambda) \approx n^{H(1/4)} \cdot \mathsf{poly}(\lambda) \approx n^{0.82} \cdot \mathsf{poly}(\lambda)$
    - $H(\alpha) \in [0,1]$: binary entropy of $\alpha \in [0,1]$

Succinct PIR from FHE

Cheatsheet

$m \approx \log n$

$D \approx m/2$

$\binom{m}{\alpha m} \approx 2^{mH(\alpha)} \approx n^{H(\alpha)}$

- Communication: upload $m \cdot \textsf{poly}(\lambda) = \log n \cdot \textsf{poly}(\lambda)$, download $\textsf{poly}(\lambda)$
- Server computation: same as before + evaluating a degree $D/2$ polynomial in m variables
  - Runtime $\approx \binom{m}{D/2} \cdot \textsf{poly}(\lambda) \approx n^{H(1/4)} \cdot \textsf{poly}(\lambda) \approx n^{0.82} \cdot \textsf{poly}(\lambda)$
  - $H(\alpha) \in [0,1]$: binary entropy of $\alpha \in [0,1]$

Theorem: with compact **fully homomorphic encryption***, we get 2-server PIR with server storage $n^{1+o(1)}$, time per query $O(n^{0.82})$ and communication $O(\log n)$.

✓

What About Linear Homomorphism?

What About Linear Homomorphism?

- Goal: compute $g(\mathbf{p})$ for g of degree $D/2$ and $||\mathbf{p}|| = D$, without revealing $\mathbf{p}$

What About Linear Homomorphism?

- Goal: compute $g(\mathbf{p})$ for g of degree $D/2$ and $||\mathbf{p}|| = D$, without revealing $\mathbf{p}$
- Naive attempt: linearise the computation by including $\mathsf{LHE}\,.\,\mathsf{Enc}(\mathbf{p}^{\otimes D/2})$ in the query

What About Linear Homomorphism?

- Goal: compute $g(\mathbf{p})$ for g of degree $D/2$ and $||\mathbf{p}|| = D$, without revealing $\mathbf{p}$
- Naive attempt: linearise the computation by including $\mathsf{LHE}\,.\,\mathsf{Enc}(\mathbf{p}^{\otimes D/2})$ in the query
  - Communication cost: $\binom{m}{D/2} \cdot \mathsf{poly}(\lambda) \approx n^{0.82} \cdot \mathsf{poly}(\lambda)$, same as before 😭

What About Linear Homomorphism?

- Goal: compute $g(\mathbf{p})$ for g of degree $D/2$ and $\|\mathbf{p}\| = D$, without revealing $\mathbf{p}$
- Naive attempt: linearise the computation by including $\mathsf{LHE}\,.\,\mathsf{Enc}(\mathbf{p}^{\otimes D/2})$ in the query
  - Communication cost: $\binom{m}{D/2} \cdot \mathsf{poly}(\lambda) \approx n^{0.82} \cdot \mathsf{poly}(\lambda)$, same as before 😭
  - Very imbalanced: uploading $n^{0.82}$ ciphertexts and downloading just one → can rebalance!

What About Linear Homomorphism?

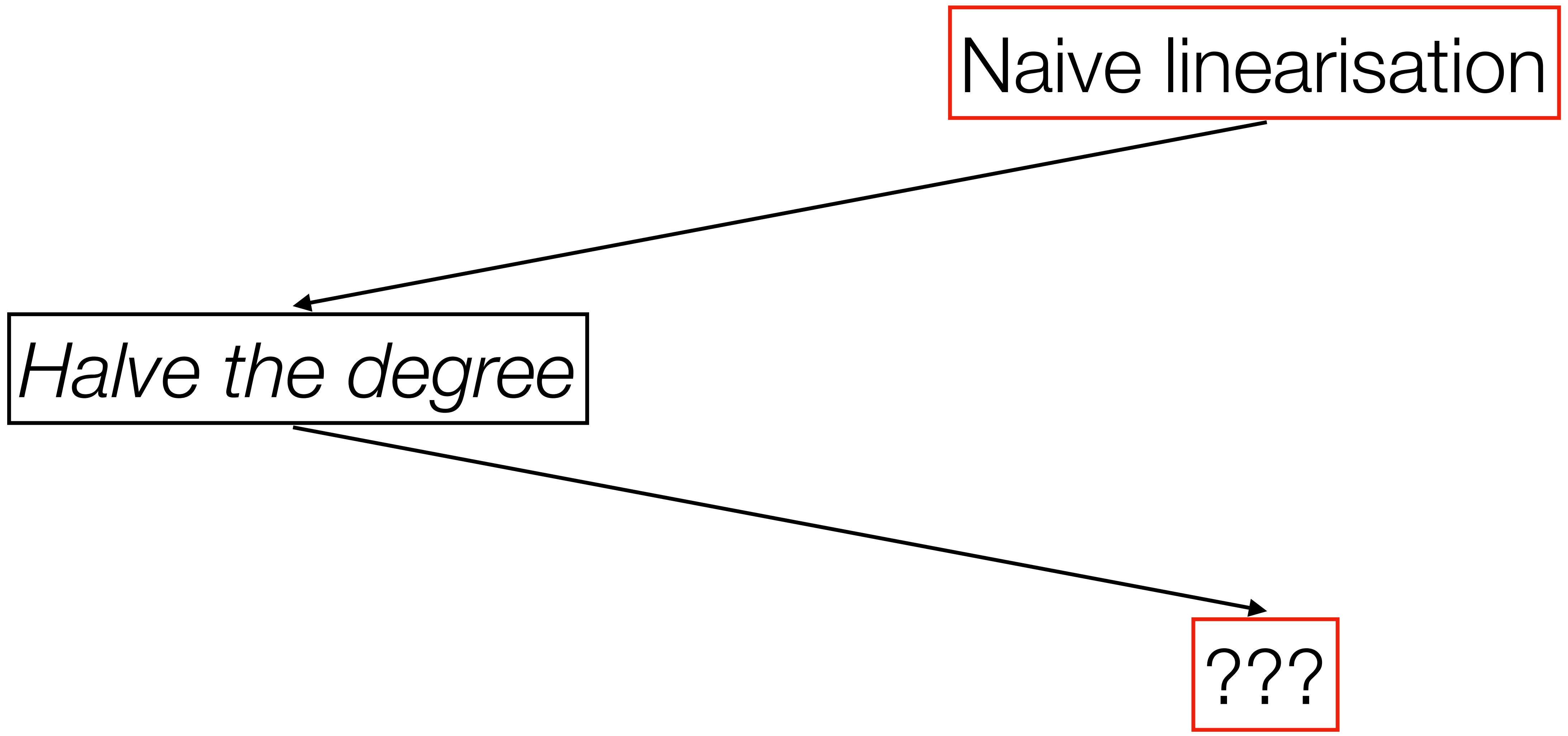
- Goal: compute $g(\mathbf{p})$ for g of degree $D/2$ and $||\mathbf{p}|| = D$, without revealing $\mathbf{p}$
- Naive attempt: linearise the computation by including $\mathsf{LHE.Enc}(\mathbf{p}^{\otimes D/2})$ in the query
  - Communication cost: $\binom{m}{D/2} \cdot \mathsf{poly}(\lambda) \approx n^{0.82} \cdot \mathsf{poly}(\lambda)$, same as before 😭
  - Very imbalanced: uploading $n^{0.82}$ ciphertexts and downloading just one → can rebalance!

Naive linearisation

???

What About Linear Homomorphism?

- Goal: compute $g(\mathbf{p})$ for g of degree $D/2$ and $||\mathbf{p}|| = D$, without revealing $\mathbf{p}$
- Naive attempt: linearise the computation by including $\mathsf{LHE}.\mathsf{Enc}(\mathbf{p}^{\otimes D/2})$ in the query
  - Communication cost: $\binom{m}{D/2} \cdot \mathsf{poly}(\lambda) \approx n^{0.82} \cdot \mathsf{poly}(\lambda)$, same as before 😭
  - Very imbalanced: uploading $n^{0.82}$ ciphertexts and downloading just one → can rebalance!



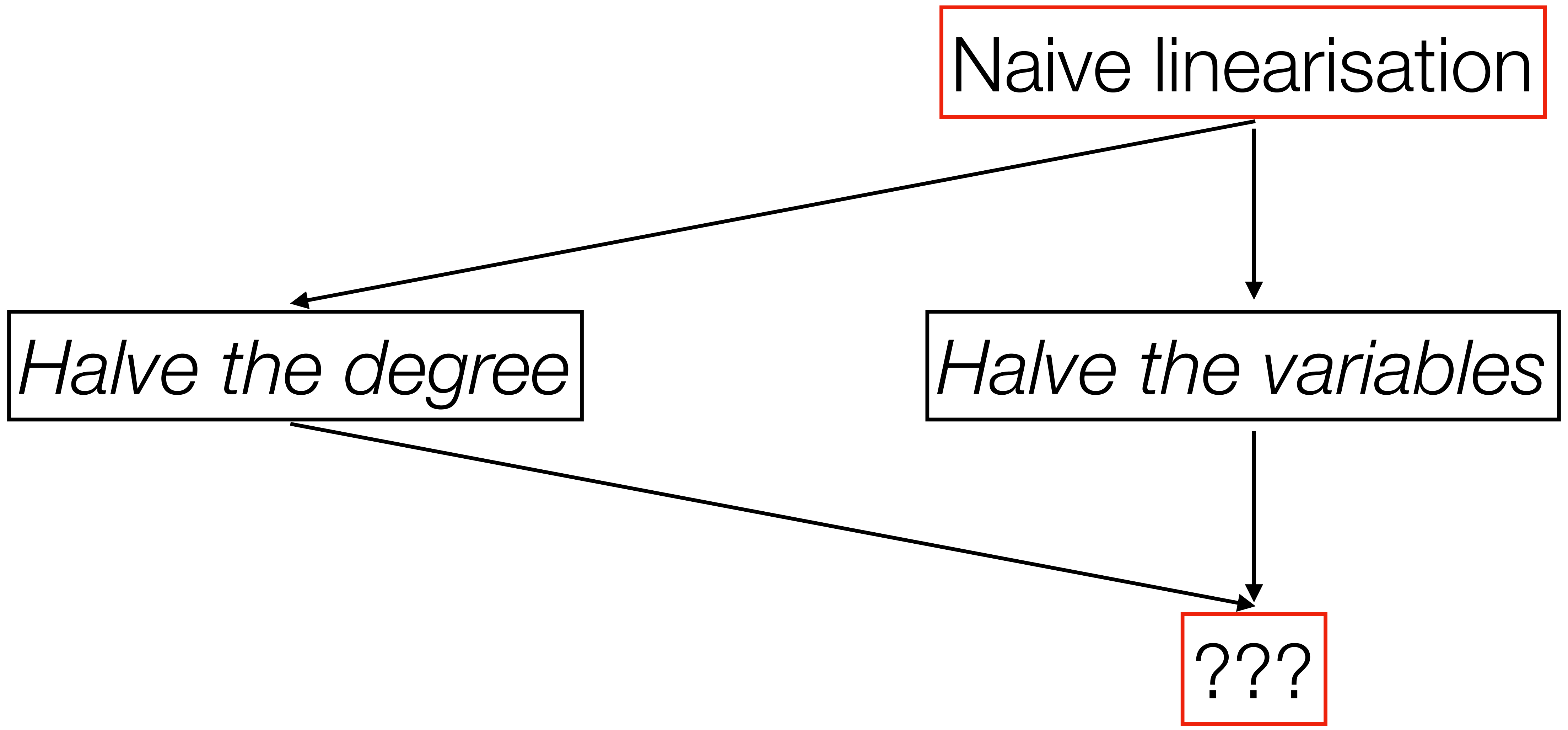
Naive linearisation

Halve the degree

???

What About Linear Homomorphism?

- Goal: compute $g(\mathbf{p})$ for g of degree $D/2$ and $||\mathbf{p}|| = D$, without revealing $\mathbf{p}$
- Naive attempt: linearise the computation by including $\mathsf{LHE}\,.\,\mathsf{Enc}(\mathbf{p}^{\otimes D/2})$ in the query
  - Communication cost: $\binom{m}{D/2} \cdot \mathsf{poly}(\lambda) \approx n^{0.82} \cdot \mathsf{poly}(\lambda)$, same as before 😭
  - Very imbalanced: uploading $n^{0.82}$ ciphertexts and downloading just one → can rebalance!



Naive linearisation

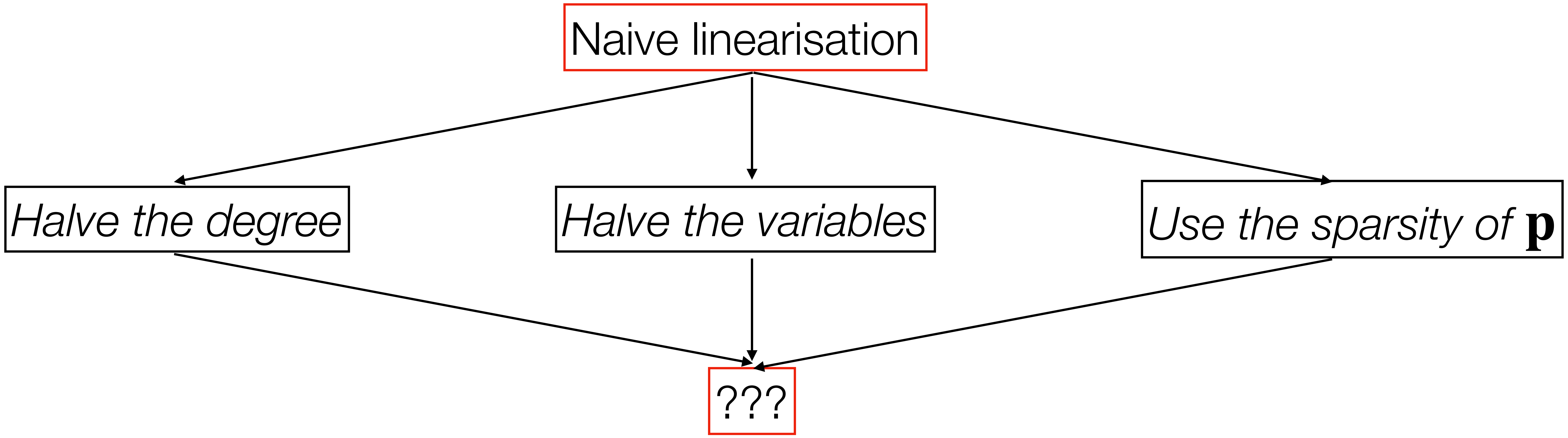
Halve the degree

Halve the variables

???

What About Linear Homomorphism?

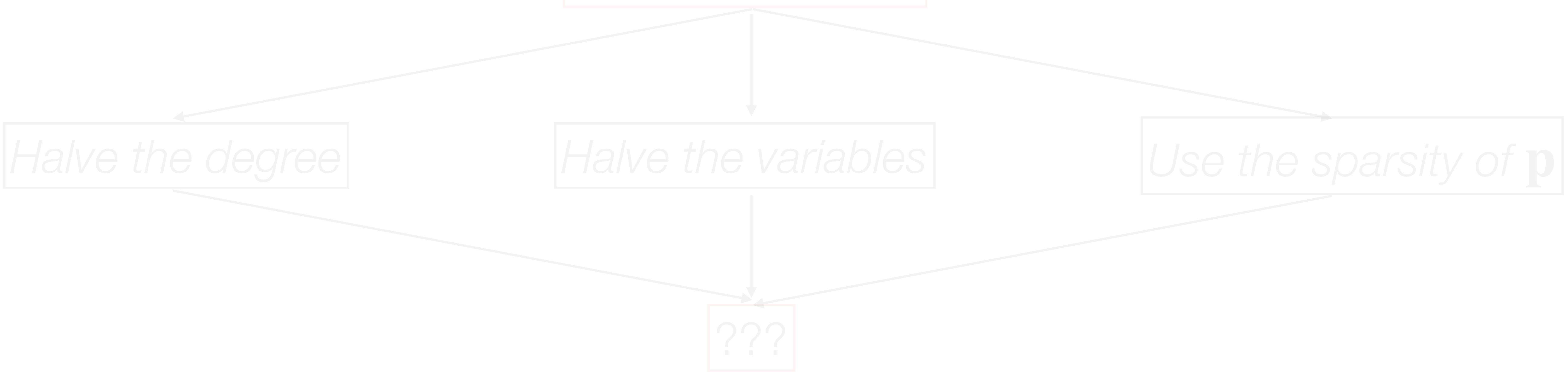
- Goal: compute $g(\mathbf{p})$ for g of degree $D/2$ and $||\mathbf{p}|| = D$, without revealing $\mathbf{p}$
- Naive attempt: linearise the computation by including $\mathsf{LHE.Enc}(\mathbf{p}^{\otimes D/2})$ in the query
  - Communication cost: $\binom{m}{D/2} \cdot \mathsf{poly}(\lambda) \approx n^{0.82} \cdot \mathsf{poly}(\lambda)$, same as before 😭
  - Very imbalanced: uploading $n^{0.82}$ ciphertexts and downloading just one → can rebalance!



What About Linear Homomorphism?

- Goal: compute $g(\mathbf{p})$ for g of degree $D/2$ and $||\mathbf{p}|| = D$, without revealing $\mathbf{p}$
- Naive attempt: linearise the computation by including LHE . Enc($\mathbf{p}^{\otimes D/2}$) in the query
- Communication cost: $\binom{m}{D/2} \cdot$ poly(λ) $\approx n^{0.82} \cdot$ poly(λ), same as before 😱
- Ver : uploading $n^{0.82}$ ciphertexts and downloading just one → can rebalance!

Natural target: reduce communication from $n^{0.82}$ to $\sqrt{n^{0.82}} = n^{0.41}$



Naive linearisation

Halve the degree

Halve the variables

Use the sparsity of $\mathbf{p}$

???

LHE → Computing Deg 2 Polynomials

LHE → Computing Deg 2 Polynomials

- **Claim** [KO97]: assume LHE. Then if the user has $\mathbf{x}, \mathbf{y} \in \mathbb{F}^{\ell}$ and the server has $\mathbf{A} \in \mathbb{F}^{\ell \times \ell}$, the user can learn $\mathbf{x}^{\top}\mathbf{A}\mathbf{y}$ without revealing $\mathbf{x}, \mathbf{y}$, with $\ell \cdot \mathsf{poly}(\lambda)$ communication.

LHE → Computing Deg 2 Polynomials

- **Claim** [KO97]: assume LHE. Then if the user has $\mathbf{x}, \mathbf{y} \in \mathbb{F}^{\ell}$ and the server has $\mathbf{A} \in \mathbb{F}^{\ell \times \ell}$, the user can learn $\mathbf{x}^{\top}\mathbf{A}\mathbf{y}$ without revealing $\mathbf{x}, \mathbf{y}$, with $\ell \cdot \mathsf{poly}(\lambda)$ communication.
- **Proof:**
  - User sends $\mathsf{LHE}.\mathsf{Enc}(\mathbf{y})$

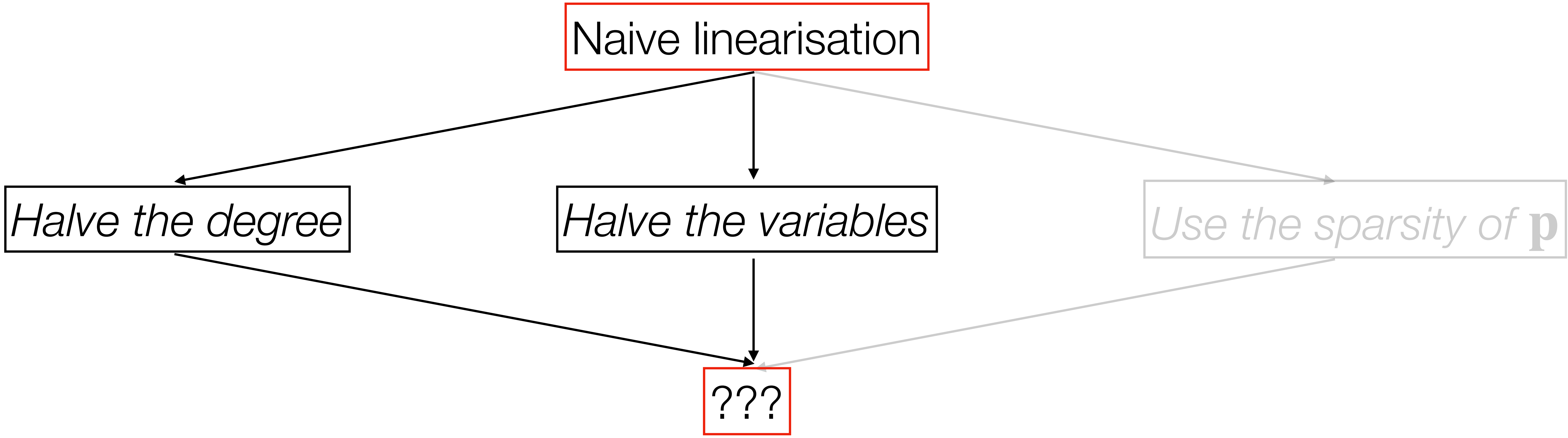
LHE → Computing Deg 2 Polynomials

- **Claim** [KO97]: assume LHE. Then if the user has $\mathbf{x}, \mathbf{y} \in \mathbb{F}^{\ell}$ and the server has $\mathbf{A} \in \mathbb{F}^{\ell \times \ell}$, the user can learn $\mathbf{x}^{\top}\mathbf{A}\mathbf{y}$ without revealing $\mathbf{x}, \mathbf{y}$, with $\ell \cdot \mathsf{poly}(\lambda)$ communication.
- **Proof:**
    - User sends $\mathsf{LHE}.\mathsf{Enc}(\mathbf{y})$
    - Server replies with $\mathsf{LHE}.\mathsf{Enc}(\mathbf{A}\mathbf{y})$

LHE → Computing Deg 2 Polynomials

- **Claim** [KO97]: assume LHE. Then if the user has $\mathbf{x}, \mathbf{y} \in \mathbb{F}^{\ell}$ and the server has $\mathbf{A} \in \mathbb{F}^{\ell \times \ell}$, the user can learn $\mathbf{x}^{\top}\mathbf{A}\mathbf{y}$ without revealing $\mathbf{x}, \mathbf{y}$, with $\ell \cdot \mathsf{poly}(\lambda)$ communication.
- **Proof:**
  - User sends $\mathsf{LHE}.\mathsf{Enc}(\mathbf{y})$
  - Server replies with $\mathsf{LHE}.\mathsf{Enc}(\mathbf{A}\mathbf{y})$
  - User decrypts and locally takes the inner product with $\mathbf{x}$

Rebalancing



Naive linearisation

Halve the degree

Halve the variables

Use the sparsity of $\mathbf{p}$

???

Halving the Degree/Variables

Cheatsheet

$\mathbf{p} \in \mathbb{F}_2^m$

$D \approx m/2$

Evaluating $g(\mathbf{p})$

$\deg g \le D/2$

$\binom{m}{\alpha m} \approx 2^{mH(\alpha)} \approx n^{H(\alpha)}$

Halving the Degree/Variables

- Want to write $g(\mathbf{p})$ as a degree 2 polynomial in as few variables as possible

Cheatsheet

$\mathbf{p} \in \mathbb{F}_2^m$

$D \approx m/2$

Evaluating $g(\mathbf{p})$

$\deg g \leq D/2$

$\binom{m}{\alpha m} \approx 2^{mH(\alpha)} \approx n^{H(\alpha)}$

Halving the Degree/Variables

Cheatsheet

$\mathbf{p} \in \mathbb{F}_2^m$

$D \approx m/2$

Evaluating $g(\mathbf{p})$

$\deg g \leq D/2$

$\binom{m}{\alpha m} \approx 2^{mH(\alpha)} \approx n^{H(\alpha)}$

- Want to write $g(\mathbf{p})$ as a degree 2 polynomial in as few variables as possible
- Idea 1: use $\mathbf{p}^{\otimes D/4}$

Halving the Degree/Variables

- Want to write $g(\mathbf{p})$ as a degree 2 polynomial in as few variables as possible
- Idea 1: use $\mathbf{p}^{\otimes D/4}$
    - Number of variables (and communication): $\binom{m}{D/4} \approx n^{H(1/8)} \approx n^{0.54}$

Cheatsheet

$\mathbf{p} \in \mathbb{F}_2^m$

$D \approx m/2$

Evaluating $g(\mathbf{p})$

$\deg g \leq D/2$

$\binom{m}{\alpha m} \approx 2^{mH(\alpha)} \approx n^{H(\alpha)}$

Halving the Degree/Variables

Cheatsheet

$\mathbf{p} \in \mathbb{F}_2^m$

$D \approx m/2$

Evaluating $g(\mathbf{p})$

$\deg g \leq D/2$

$\binom{m}{\alpha m} \approx 2^{mH(\alpha)} \approx n^{H(\alpha)}$

- Want to write $g(\mathbf{p})$ as a degree 2 polynomial in as few variables as possible
- Idea 1: use $\mathbf{p}^{\otimes D/4}$
  - Number of variables (and communication): $\binom{m}{D/4} \approx n^{H(1/8)} \approx n^{0.54}$
- Idea 2: use $\mathbf{p}_{1:m/2}^{\otimes D/2}$ and $\mathbf{p}_{m/2+1:m}^{\otimes D/2}$

Halving the Degree/Variables

Cheatsheet

$\mathbf{p} \in \mathbb{F}_2^m$

$D \approx m/2$

Evaluating $g(\mathbf{p})$

$\deg g \leq D/2$

$\binom{m}{\alpha m} \approx 2^{mH(\alpha)} \approx n^{H(\alpha)}$

- Want to write $g(\mathbf{p})$ as a degree 2 polynomial in as few variables as possible
- Idea 1: use $\mathbf{p}^{\otimes D/4}$
  - Number of variables (and communication): $\binom{m}{D/4} \approx n^{H(1/8)} \approx n^{0.54}$
- Idea 2: use $\mathbf{p}_{1:m/2}^{\otimes D/2}$ and $\mathbf{p}_{m/2+1:m}^{\otimes D/2}$
  - Number of variables: $\binom{m/2}{D/2} \approx 2^{mH(1/2)/2} \approx n^{0.5}$

Halving the Degree/Variables

Cheatsheet

$\mathbf{p} \in \mathbb{F}_2^m$

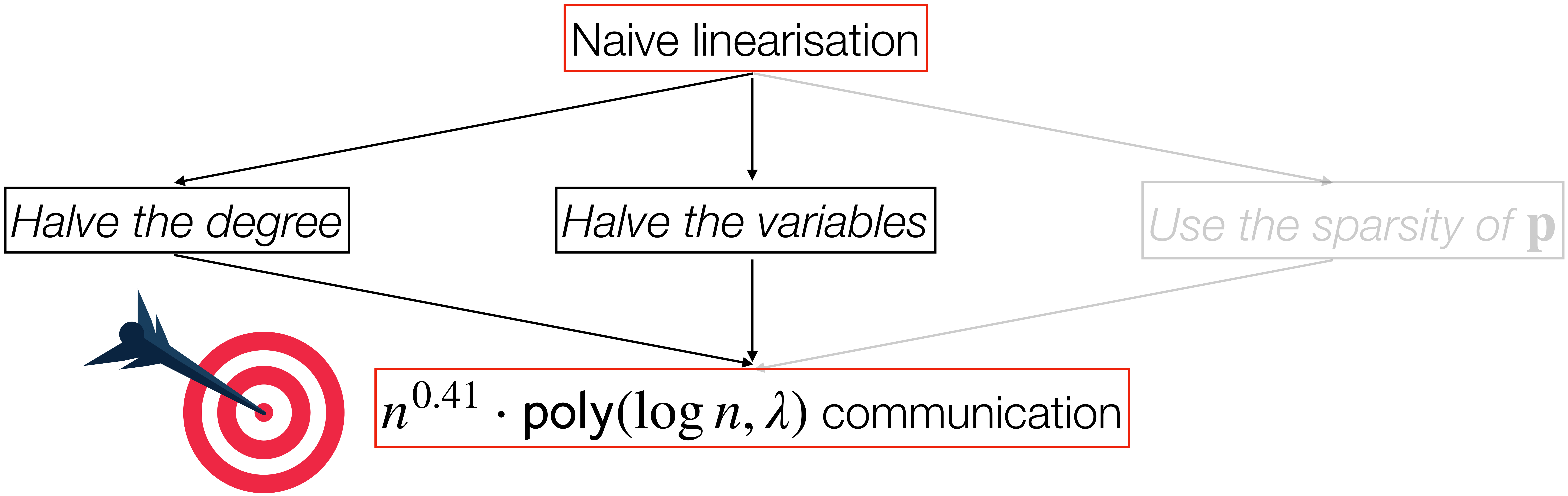
$D \approx m/2$

Evaluating $g(\mathbf{p})$

$\deg g \leq D/2$

$\binom{m}{\alpha m} \approx 2^{mH(\alpha)} \approx n^{H(\alpha)}$

- Want to write $g(\mathbf{p})$ as a degree 2 polynomial in as few variables as possible
- Idea 1: use $\mathbf{p}^{\otimes D/4}$
  - Number of variables (and communication): $\binom{m}{D/4} \approx n^{H(1/8)} \approx n^{0.54}$
- Idea 2: use $\mathbf{p}_{1:m/2}^{\otimes D/2}$ and $\mathbf{p}_{m/2+1:m}^{\otimes D/2}$
  - Number of variables: $\binom{m/2}{D/2} \approx 2^{mH(1/2)/2} \approx n^{0.5}$
- Careful combination of these ideas: $\approx \binom{m/2}{D/4} \approx 2^{mH(1/4)/2} \approx n^{0.41}$



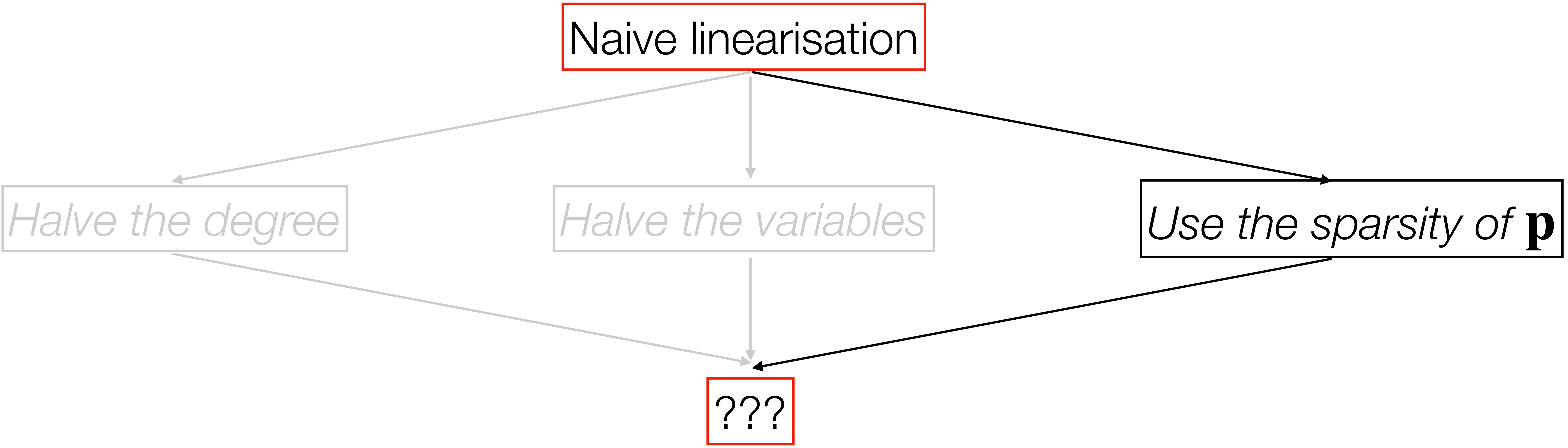
Naive linearisation

Halve the degree

Halve the variables

Use the sparsity of $\mathbf{p}$

$n^{0.41} \cdot \mathsf{poly}(\log n, \lambda)$ communication



Naive linearisation

Halve the degree

Halve the variables

Use the sparsity of $\mathbf{p}$

???

Leveraging the Sparsity of $\mathbf{p}$ [BIM04]

Cheatsheet

$\mathbf{p} \in \mathbb{F}_2^m$

$D \approx m/2$

Evaluating $g(\mathbf{p})$

$\deg g \leq D/2$

$\|\mathbf{p}\| = D$

Leveraging the Sparsity of $\mathbf{p}$ [BIM04]

Cheatsheet

$\mathbf{p} \in \mathbb{F}_2^m$

$D \approx m/2$

Evaluating $g(\mathbf{p})$

$\deg g \leq D/2$

$\|\mathbf{p}\| = D$

- Original naive idea: compute $\langle \mathbf{p}^{\otimes D/2}, \mathsf{coefs}(g) \rangle$ under the hood

Leveraging the Sparsity of $\mathbf{p}$ [BIM04]

- Original naive idea: compute $\langle \mathbf{p}^{\otimes D/2}, \mathsf{coefs}(g) \rangle$ under the hood

Cheatsheet

$\mathbf{p} \in \mathbb{F}_2^m$

$D \approx m/2$

Evaluating $g(\mathbf{p})$

$\deg g \leq D/2$

$\|\mathbf{p}\| = D$

Leveraging the Sparsity of $\mathbf{p}$ [BIM04]

Cheatsheet

$\mathbf{p} \in \mathbb{F}_2^m$

$D \approx m/2$

Evaluating $g(\mathbf{p})$

$\deg g \leq D/2$

$\|\mathbf{p}\| = D$

- Original naive idea: compute $\langle \mathbf{p}^{\otimes D/2}, \mathsf{coefs}(g) \rangle$ under the hood
- Observation: $\mathbf{p}$ only has D nonzero entries → $\mathbf{p}^{\otimes D/2}$ has $\leq 2^D \ll \binom{m}{D/2}$ nonzero entries!

Leveraging the Sparsity of $\mathbf{p}$ [BIM04]

Cheatsheet

$\mathbf{p} \in \mathbb{F}_2^m$

$D \approx m/2$

Evaluating $g(\mathbf{p})$

$\deg g \leq D/2$

$\|\mathbf{p}\| = D$

- Original naive idea: compute $\langle \mathbf{p}^{\otimes D/2}, \mathsf{coefs}(g) \rangle$ under the hood
- Observation: $\mathbf{p}$ only has D nonzero entries → $\mathbf{p}^{\otimes D/2}$ has $\leq 2^D \ll \binom{m}{D/2}$ nonzero entries!
- Idea: view $\mathsf{coefs}(g)$ as a "mini-database" and run a single-server "mini-PIR" protocol (KO97, IKOS04) to retrieve $\mathsf{coefs}(g)$ in just these 2^D positions

Leveraging the Sparsity of $\mathbf{p}$ [BIM04]

Cheatsheet

$\mathbf{p} \in \mathbb{F}_2^m$

$D \approx m/2$

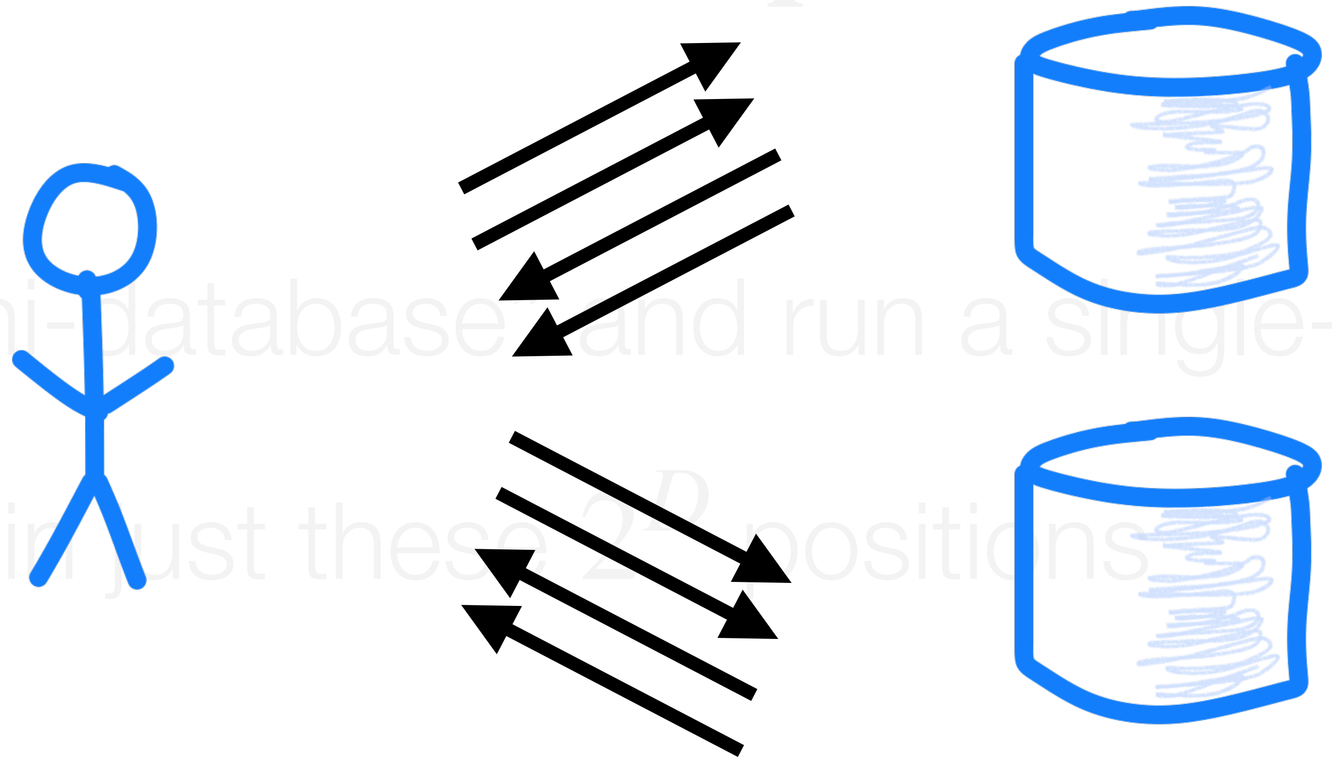
Evaluating $g(\mathbf{p})$

$\deg g \leq D/2$

$\|\mathbf{p}\| = D$

- Original naive idea: compute $\langle \mathbf{p}^{\otimes D/2}, \textbf{coefs}(g) \rangle$ under the hood
- Observation: $\mathbf{p}$ only has D nonzero entries $\Rightarrow \mathbf{p}^{\otimes D/2}$ has $\leq 2^D \ll \binom{m}{D/2}$ nonzero entries!
- Idea: view $\textbf{coefs}(g)$ as a “mini-database” and run a single-server “mini-PIR” protocol (KO97, IKOS04) to retrieve $\textbf{coefs}(g)$ in just these 2^D positions

Batch PIR with many, non-adaptive queries



[IKOS’04,HHG’13,GKL’10,AS’16,H’16,ACLS’18,CHLR’18]

Leveraging the Sparsity of $\mathbf{p}$ [BIM04]

Cheatsheet

$\mathbf{p} \in \mathbb{F}_2^m$

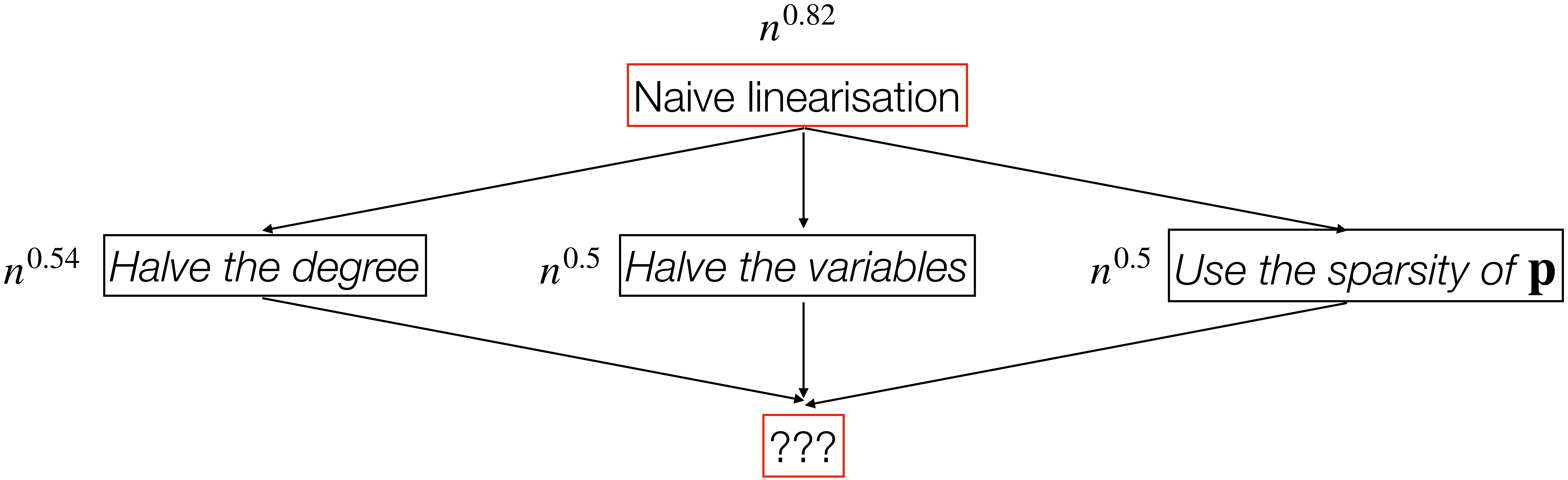
$D \approx m/2$

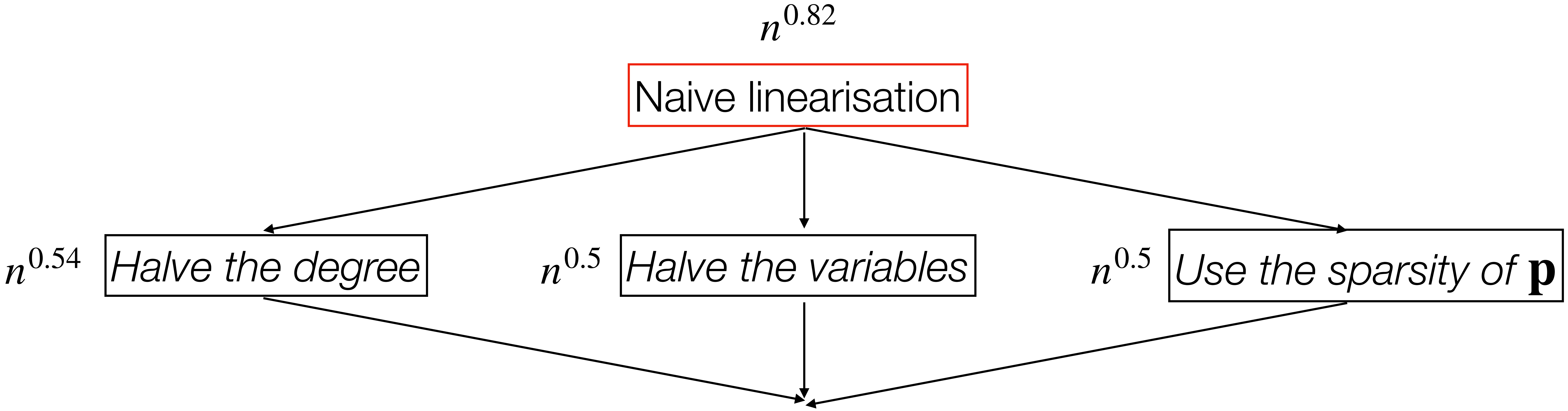
Evaluating $g(\mathbf{p})$

$\deg g \leq D/2$

$\|\mathbf{p}\| = D$

- Original naive idea: compute $\langle \mathbf{p}^{\otimes D/2}, \mathsf{coefs}(g) \rangle$ under the hood
- Observation: $\mathbf{p}$ only has D nonzero entries → $\mathbf{p}^{\otimes D/2}$ has $\leq 2^D \ll \binom{m}{D/2}$ nonzero entries!
- Idea: view $\mathsf{coefs}(g)$ as a "mini-database" and run a single-server "mini-PIR" protocol (KO97, IKOS04) to retrieve $\mathsf{coefs}(g)$ in just these 2^D positions
- Communication: $\approx 2^D \approx 2^{m/2} \approx n^{0.5}$


$n^{0.82}$
Naive linearisation
$n^{0.54}$
Halve the degree
$n^{0.5}$
Halve the variables
$n^{0.5}$
Use the sparsity of $\mathbf{p}$
???



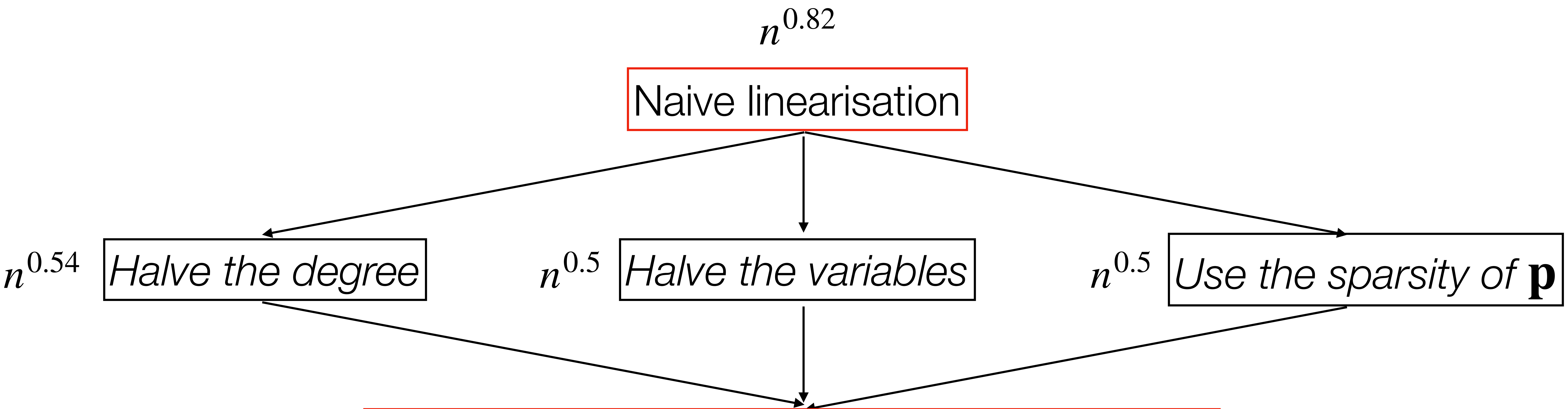
$n^{0.82}$

Naive linearisation

$n^{0.54}$ *Halve the degree*

$n^{0.5}$ *Halve the variables*

$n^{0.5}$ *Use the sparsity of* $\mathbf{p}$



$n^{0.82}$ Naive linearisation

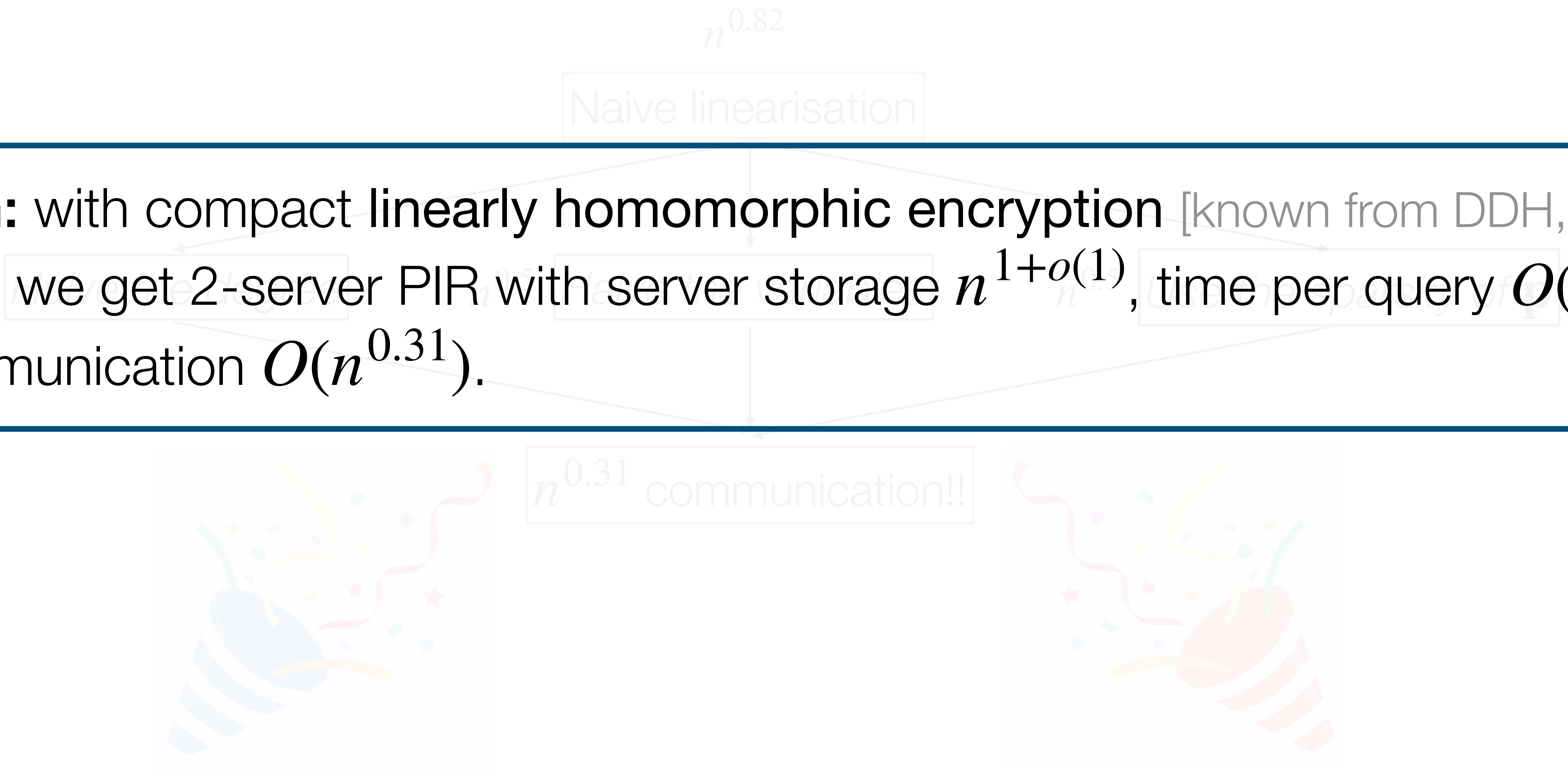
$n^{0.54}$ *Halve the degree*

$n^{0.5}$ *Halve the variables*

$n^{0.5}$ *Use the sparsity of* $\mathbf{p}$

$n^{0.31}$ communication!!

Even better than the $n^{0.41}$ we were aiming for!



Theorem: with compact **linearly homomorphic encryption** [known from DDH, DCR, QR, LWE], we get 2-server PIR with server storage $n^{1+o(1)}$, time per query $O(n^{0.82})$ and communication $O(n^{0.31})$.

✓

This talk

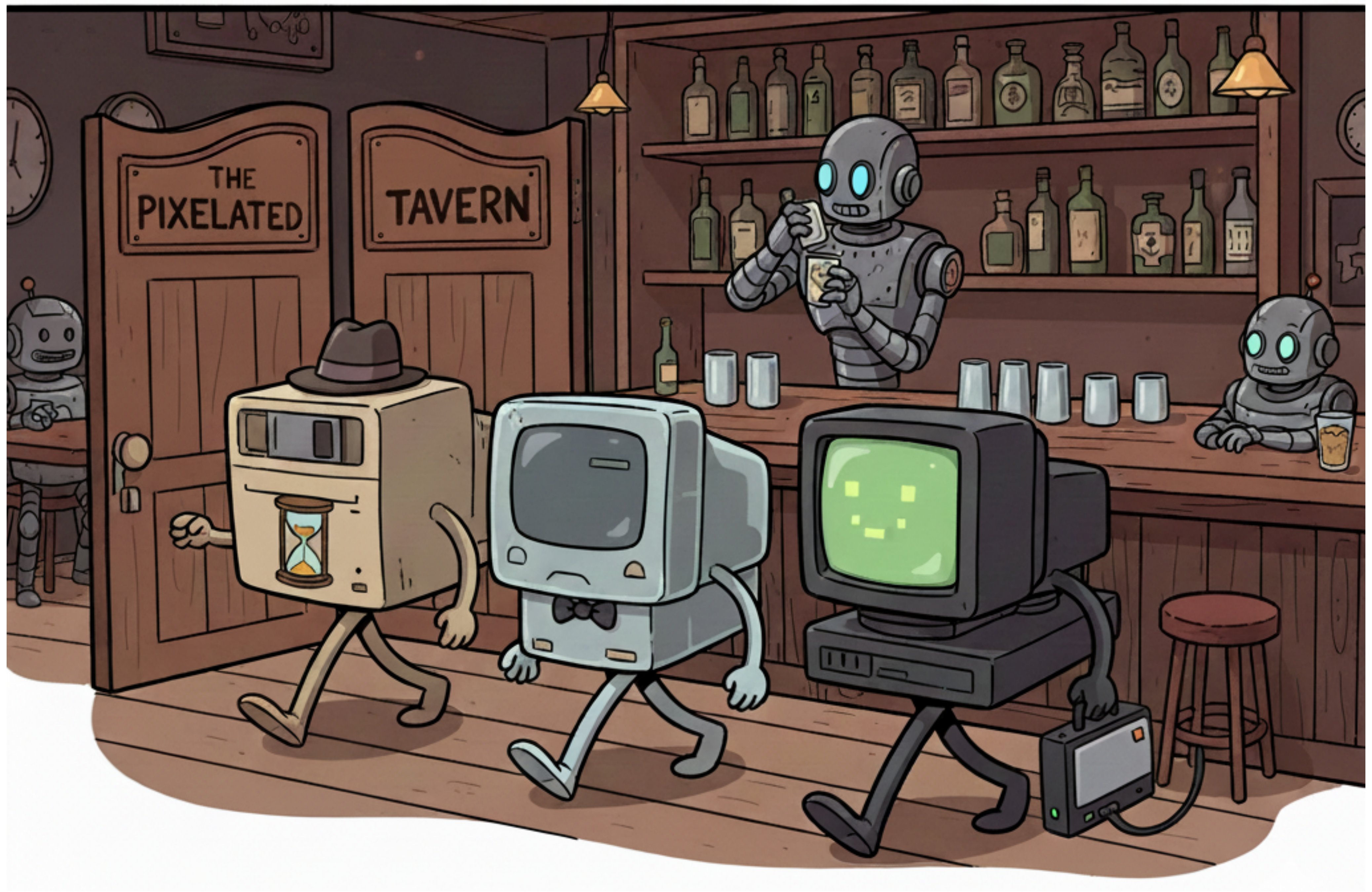
1. **Background:** PIR with preprocessing
2. **[HR26] New two-server PIR:** sublinear time, quasilinear space
3. **Evaluation:** what does this mean for practice?
4. **Bonuses** 😁
    - Reducing communication using crypto
    - **[HPR26]** Connecting multi-server PIR to complexity theory

This talk

1. **Background:** PIR with preprocessing

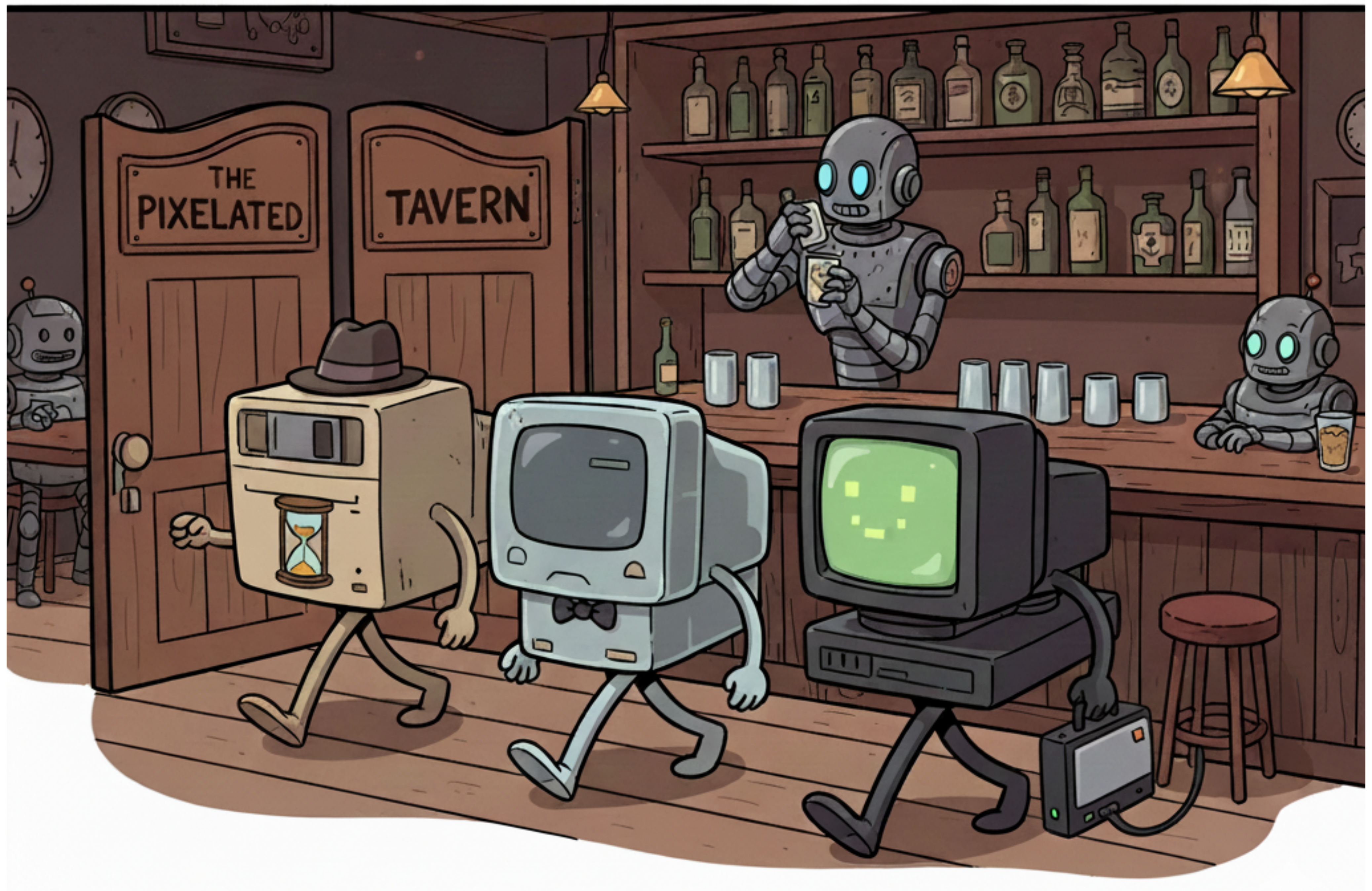
2. **[HR26] New two-server PIR:** sublinear time, quasilinear space
3. **Evaluation:** what does this mean for practice?
4. **Bonuses** 😁
    - Reducing communication using crypto
    - ➡ **[HPR26]** Connecting multi-server PIR to complexity theory

Three Turing Machines Walk Into a Bar



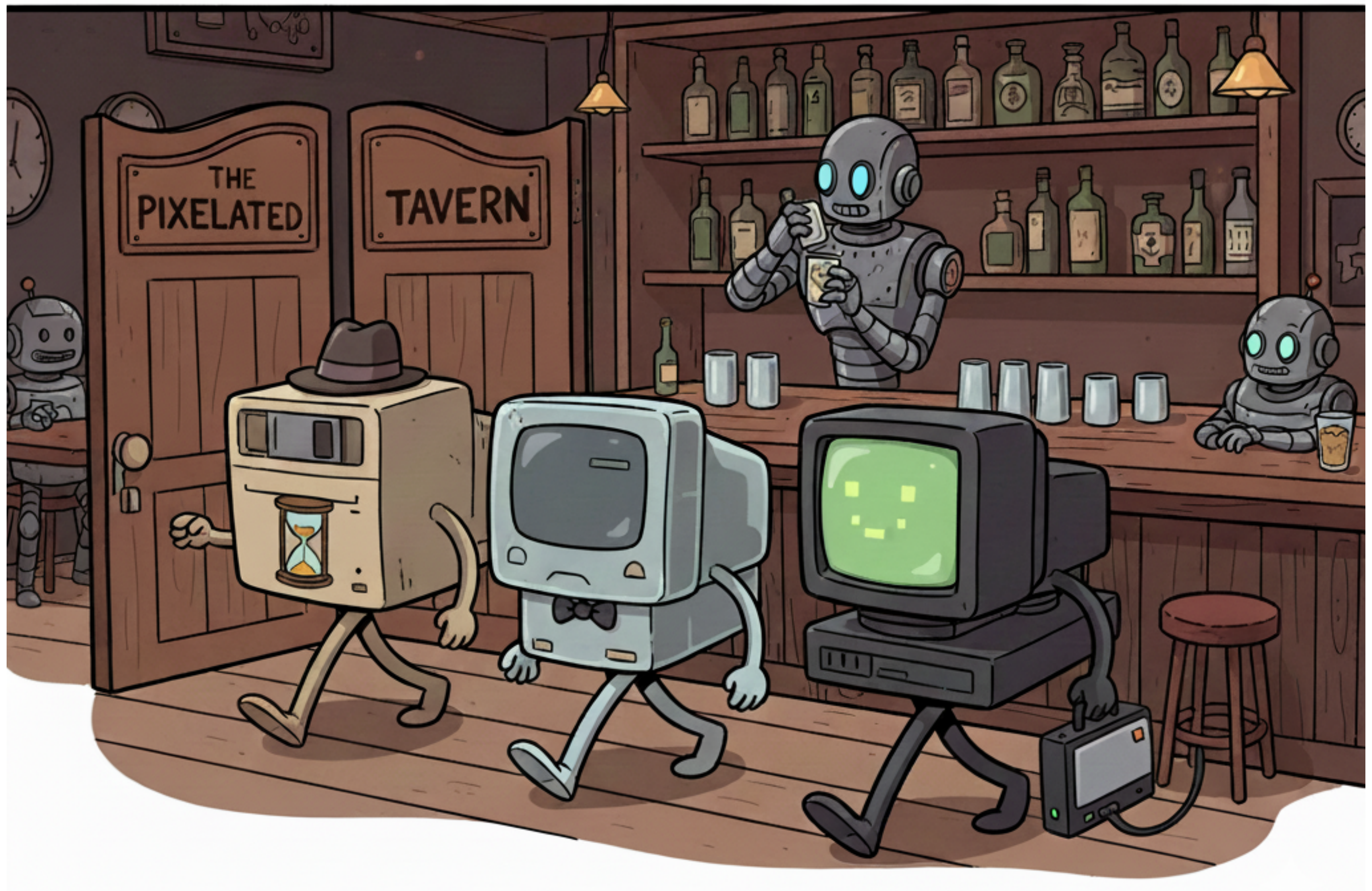
Three Turing Machines Walk Into a Bar

- Computer A: has to run in time $\leq T$, but has unlimited memory



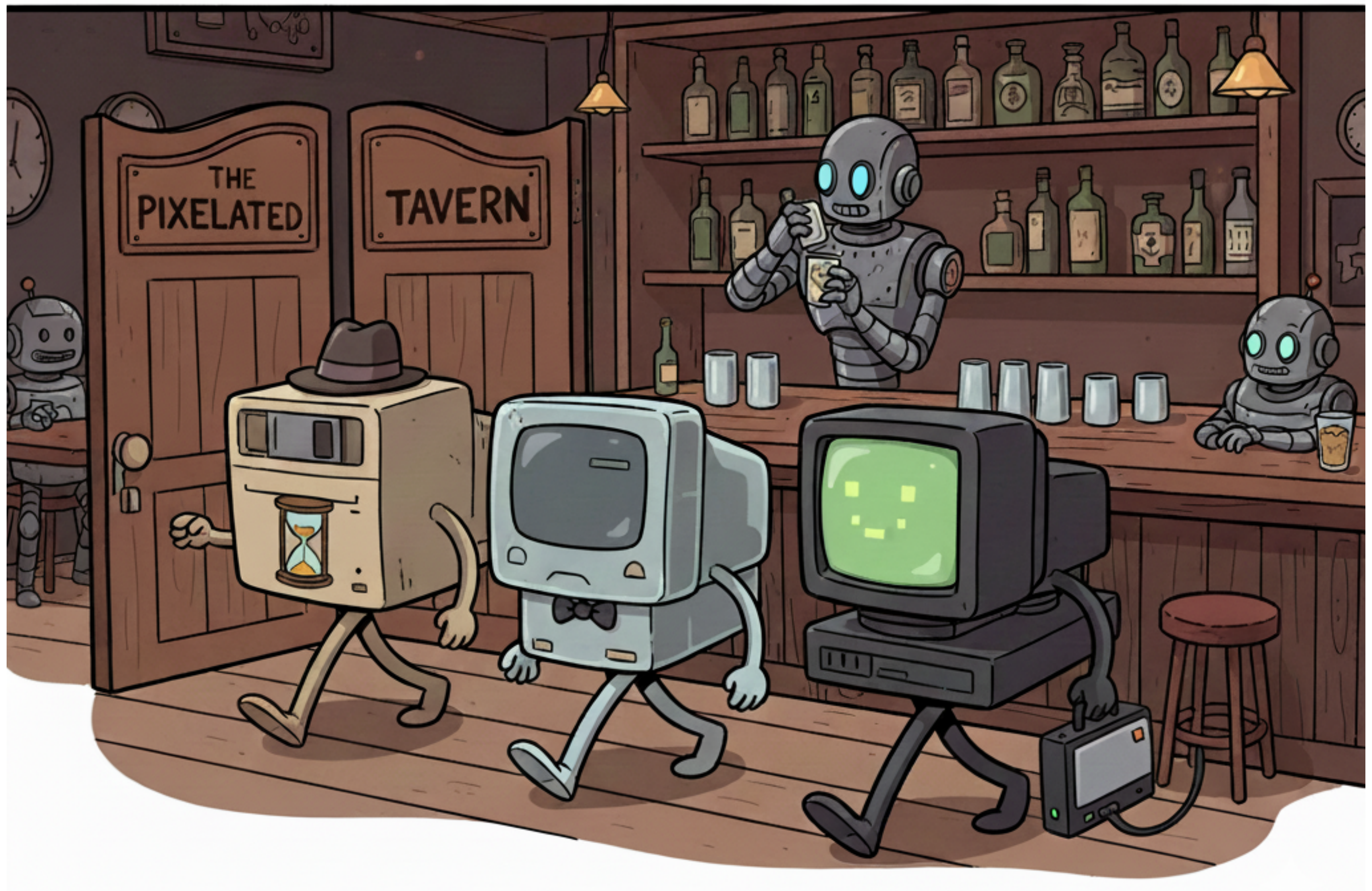
Three Turing Machines Walk Into a Bar

- Computer A: has to run in time $\leq T$, but has unlimited memory
- Computer B: has memory $\sqrt{T \log T}$, but can run in unlimited time



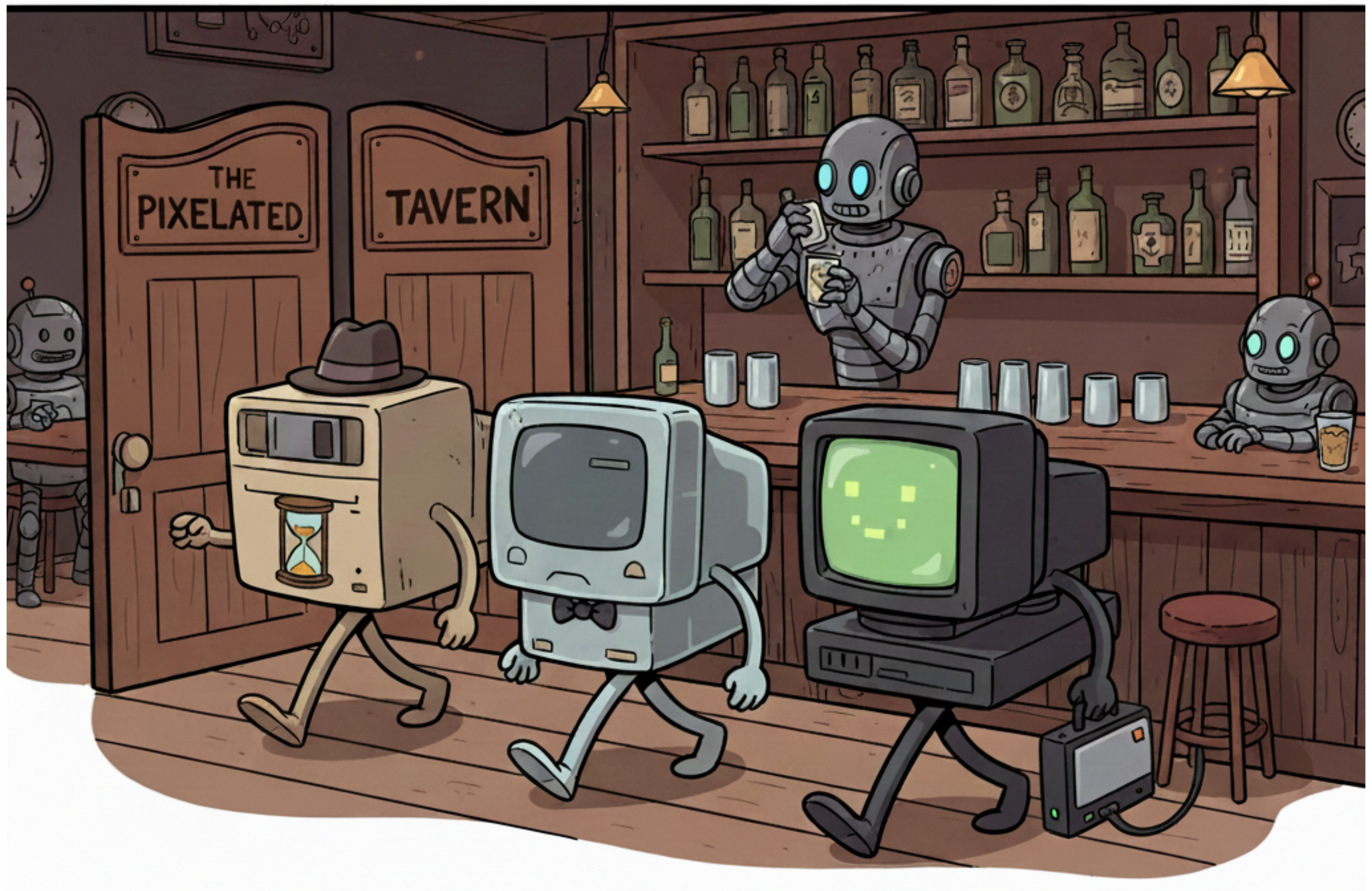
Three Turing Machines Walk Into a Bar

- Computer A: has to run in time $\leq T$, but has unlimited memory
- Computer B: has memory $\sqrt{T \log T}$, but can run in unlimited time
- Computer C: has memory $\sqrt{T}$, unlimited time, and a full “catalytic” hard drive of size $2^{T^{0.001}}$



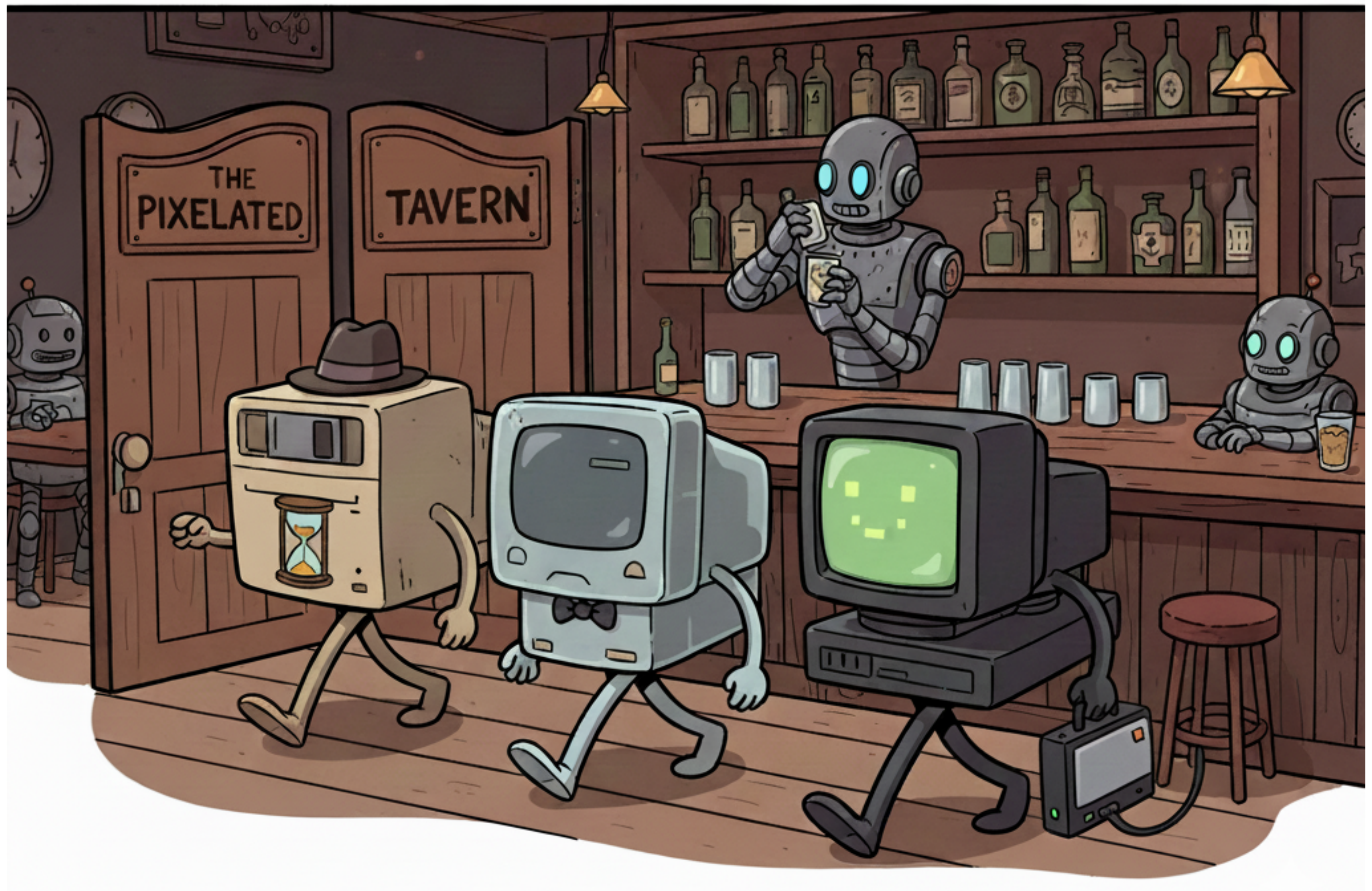
Three Turing Machines Walk Into a Bar

- Computer A: has to run in time $\leq T$, but has unlimited memory
- Computer B: has memory $\sqrt{T \log T}$, but can run in unlimited time
- Computer C: has memory $\sqrt{T}$, unlimited time, and a full “catalytic” hard drive of size $2^{T^{0.001}}$
  - Can temporarily modify the hard drive’s contents
  - Must restore to the original state at the end!



Three Turing Machines Walk Into a Bar

- Computer A: has to run in time $\leq T$, but has unlimited memory
- Computer B: has memory $\sqrt{T \log T}$, but can run in unlimited time
- Computer C: has memory $\sqrt{T}$, unlimited time, and a full "catalytic" hard drive of size $2^{T^{0.001}}$
  - Can temporarily modify the hard drive's contents
  - Must restore to the original state at the end!



Q: How do the powers of these computers compare?

Is one of them "strictly weakest" i.e. the other two computers could do absolutely anything that it could?

50 Years of Time-Space Tradeoffs

- Computer A: time $\leq T$, unlimited memory
- Computer B: memory $\sqrt{T \log T}$, unlimited time
- Computer C: memory $\sqrt{T}$, unlimited time, full hard drive of size $2^{T^{0.001}}$

50 Years of Time-Space Tradeoffs

- **Folklore:** if Computer B had memory T, then it would be more powerful than Computer A

- Computer A: time $\leq T$, unlimited memory
- Computer B: memory $\sqrt{T \log T}$, unlimited time
- Computer C: memory $\sqrt{T}$, unlimited time, <u>full</u> hard drive of size $2^{T^{0.001}}$

50 Years of Time-Space Tradeoffs

- **Folklore:** if Computer B had memory T, then it would be more powerful than Computer A
- **Hopcroft-Paul-Valiant ’77:** if Computer B instead had memory $T/\log T$, then it would be more powerful than Computer A

- Computer A: time $\leq T$, unlimited memory
- Computer B: memory $\sqrt{T \log T}$, unlimited time
- Computer C: memory $\sqrt{T}$, unlimited time, <u>full</u> hard drive of size $2^{T^{0.001}}$

50 Years of Time-Space Tradeoffs

- **Folklore:** if Computer B had memory T, then it would be more powerful than Computer A
- **Hopcroft-Paul-Valiant '77:** if Computer B instead had memory $T/\log T$, then it would be more powerful than Computer A
- **Cook-Mertz '23, Williams '25:** Computer B is more powerful than Computer A!

- Computer A: time $\leq T$, unlimited memory
- Computer B: memory $\sqrt{T \log T}$, unlimited time
- Computer C: memory $\sqrt{T}$, unlimited time, full hard drive of size $2^{T^{0.001}}$

50 Years of Time-Space Tradeoffs

- **Folklore:** if Computer B had memory T, then it would be more powerful than Computer A
- **Hopcroft-Paul-Valiant '77:** if Computer B instead had memory $T/\log T$, then it would be more powerful than Computer A
- **Cook-Mertz '23, Williams '25:** Computer B is more powerful than Computer A!
- **Henzinger-Pyne-R '26, combining Cook-Mertz-Williams with PIR techniques:** Computer C is also more powerful than Computer A!

- Computer A: time $\leq T$, unlimited memory
- Computer B: memory $\sqrt{T \log T}$, unlimited time
- Computer C: memory $\sqrt{T}$, unlimited time, full hard drive of size $2^{T^{0.001}}$

50 Years of Time-Space Tradeoffs

- **Folklore:** if Computer B had memory T, then it would be more powerful than Computer A
- **Hopcroft-Paul-Valiant ’77:** if Computer B instead had memory $T/\log T$, then it would be more powerful than Computer A
- **Cook-Mertz ’23, Williams ’25:** Computer B is more powerful than Computer A!
- **Henzinger-Pyne-R ’26, combining Cook-Mertz-Williams with PIR techniques:** Computer C is also more powerful than Computer A!

- Computer A: time $\leq T$, unlimited memory
- Computer B: memory $\sqrt{T \log T}$, unlimited time
- Computer C: memory $\sqrt{T}$, unlimited time, full hard drive of size $2^{T^{0.001}}$

Q: How do the powers of these computers compare?
A: Computer A is strictly weakest

Generalising Cook-Mertz-Williams

Generalising Cook-Mertz-Williams

- Insight: the workhorse of Cook-Mertz-Williams is the Reed-Muller PIR protocol in disguise
    - Uses $O(\log n)$ servers for a database of size n

Generalising Cook-Mertz-Williams

- Insight: the workhorse of Cook-Mertz-Williams is the Reed-Muller PIR protocol in disguise
    - Uses $O(\log n)$ servers for a database of size n
- Our work: replace this with an existing low-communication PIR protocol using $O(1)$ servers

Generalising Cook-Mertz-Williams

- Insight: the workhorse of Cook-Mertz-Williams is the Reed-Muller PIR protocol in disguise
  - Uses $O(\log n)$ servers for a database of size n
- Our work: replace this with an existing low-communication PIR protocol using $O(1)$ servers

TLDR: techniques from info-theoretic PIR connect to complexity theory in surprising ways!

Hopefully many more connections to come 😁

Summary

Summary

- [HR26] First information-theoretic PIR with $n^{1+o(1)}$ storage, $n^{1-\Omega(1)}$ server time, and $O(1)$ servers

Summary

- [HR26] First information-theoretic PIR with $n^{1+o(1)}$ storage, $n^{1-\Omega(1)}$ server time, and $O(1)$ servers
    - $s = 2$: server time $n^{0.82}$

Summary

- [HR26] First information-theoretic PIR with $n^{1+o(1)}$ storage, $n^{1-\Omega(1)}$ server time, and $O(1)$ servers
  - $s = 2$: server time $n^{0.82}$
  - Communication not ideal, but can be shrunk with linearly or fully homomorphic encryption

Summary

- [HR26] First information-theoretic PIR with $n^{1+o(1)}$ storage, $n^{1-\Omega(1)}$ server time, and $O(1)$ servers
  - $s = 2$: server time $n^{0.82}$
  - Communication not ideal, but can be shrunk with linearly or fully homomorphic encryption
  - Seems like it could be the first practically feasible PIR with preprocessing!

Summary

- [HR26] First information-theoretic PIR with $n^{1+o(1)}$ storage, $n^{1-\Omega(1)}$ server time, and $O(1)$ servers
  - $s = 2$: server time $n^{0.82}$
  - Communication not ideal, but can be shrunk with linearly or fully homomorphic encryption
  - Seems like it could be the first practically feasible PIR with preprocessing!
- [HPR26] New results on catalytic space using techniques from information-theoretic PIR

Wishlist: Constructing Info-Theoretic PIR with Preprocessing

$O(1)$ servers, $n^{1+o(1)}$ storage, $n^{1/2-\Omega(1)}$ server time?

Wishlist: Constructing Info-Theoretic PIR with Preprocessing

$O(1)$ servers, $n^{1+o(1)}$ storage, $n^{1/2-\Omega(1)}$ server time?

$O(1)$ servers, **poly**(n) storage, $n^{1-\Omega(1)}$ server time, communication $n^{o(1)}$?

Wishlist: Constructing Info-Theoretic PIR with Preprocessing

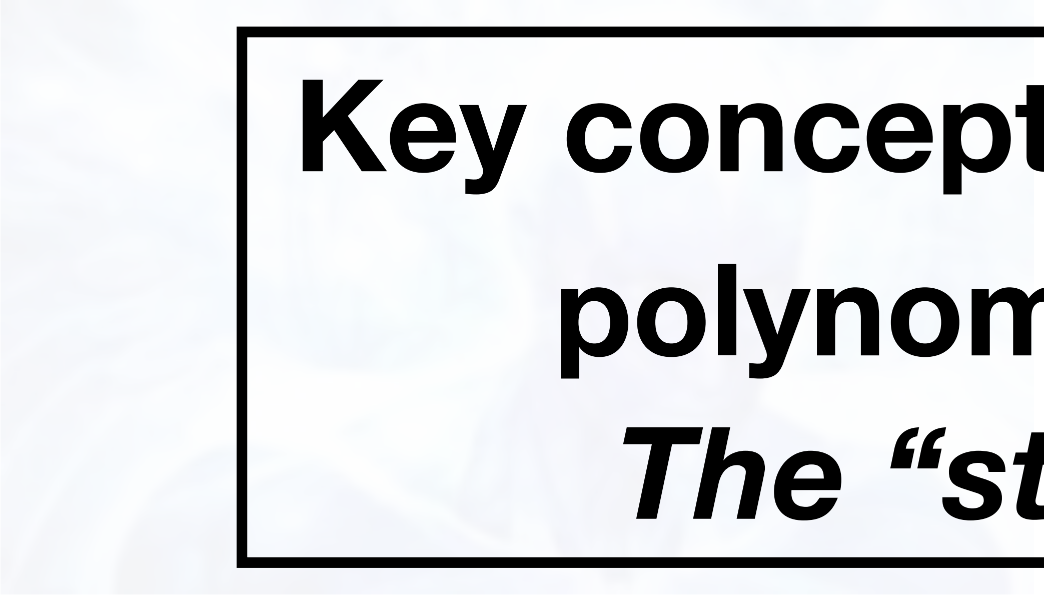
$O(1)$ servers, $n^{1+o(1)}$ storage, $n^{1/2-\Omega(1)}$ server time?

$O(1)$ servers, **poly**(n) storage, $n^{1-\Omega(1)}$ server time, communication $n^{o(1)}$?

$O(1)$ servers, **poly**(n) storage, $n^{o(1)}$ server time?

Wishlist: Constructing Info-Theoretic PIR with Preprocessing

$O(1)$ servers, $n^{1+o(1)}$ storage, $n^{1/2-\Omega(1)}$ server time?



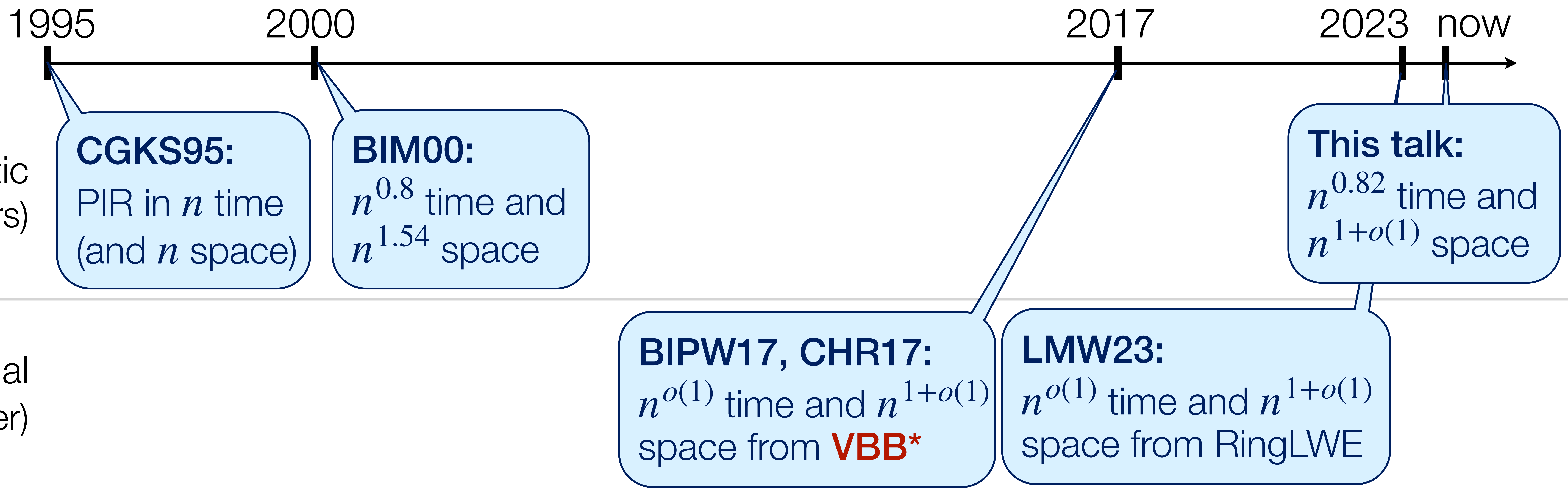
$O(1)$ servers, poly(n) storage, $n^{1-\Omega(1)}$ server time, communication $n^{o(1)}$?

Key conceptual challenge: better data structures for evaluating polynomials with many ($\gg \log n$ or even n^{ϵ}) variables.

The "store all evaluations" solution no longer cuts it!

$O(1)$ servers, poly(n) storage, $n^{o(1)}$ server time?

Thank you! Questions?



Info-theoretic (2 servers)

CGKS95: PIR in n time (and n space)

BIM00: $n^{0.8}$ time and $n^{1.54}$ space

This talk: $n^{0.82}$ time and $n^{1+o(1)}$ space

Computational (1 server)

BIPW17, CHR17: $n^{o(1)}$ time and $n^{1+o(1)}$ space from **VBB***

LMW23: $n^{o(1)}$ time and $n^{1+o(1)}$ space from RingLWE

Concrete storage (2 GB database)

8×10^5 TB **≈ $10 million in hard drives**

?

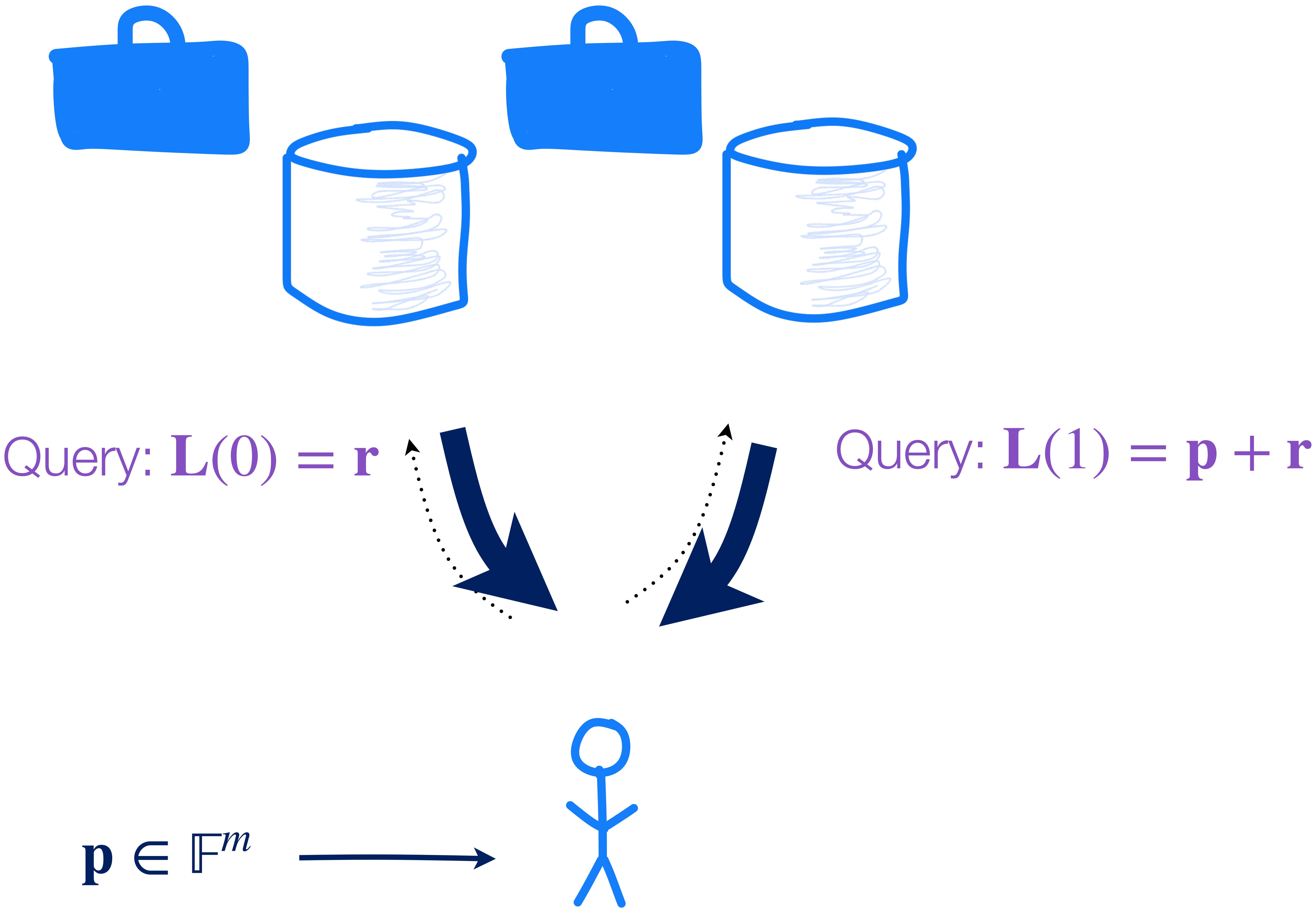
2×10^6 TB **≈ $20 million in hard drives**

1 TB **≈ $10 in hard drives**

*Ignoring polylog savings in time

Bonus Slides on ≥ 3 Servers

$s = 2$ Servers: A Refresher



Query: $\mathbf{L}(0) = \mathbf{r}$

Query: $\mathbf{L}(1) = \mathbf{p} + \mathbf{r}$

$\mathbf{p} \in \mathbb{F}^m$

Cheatsheet

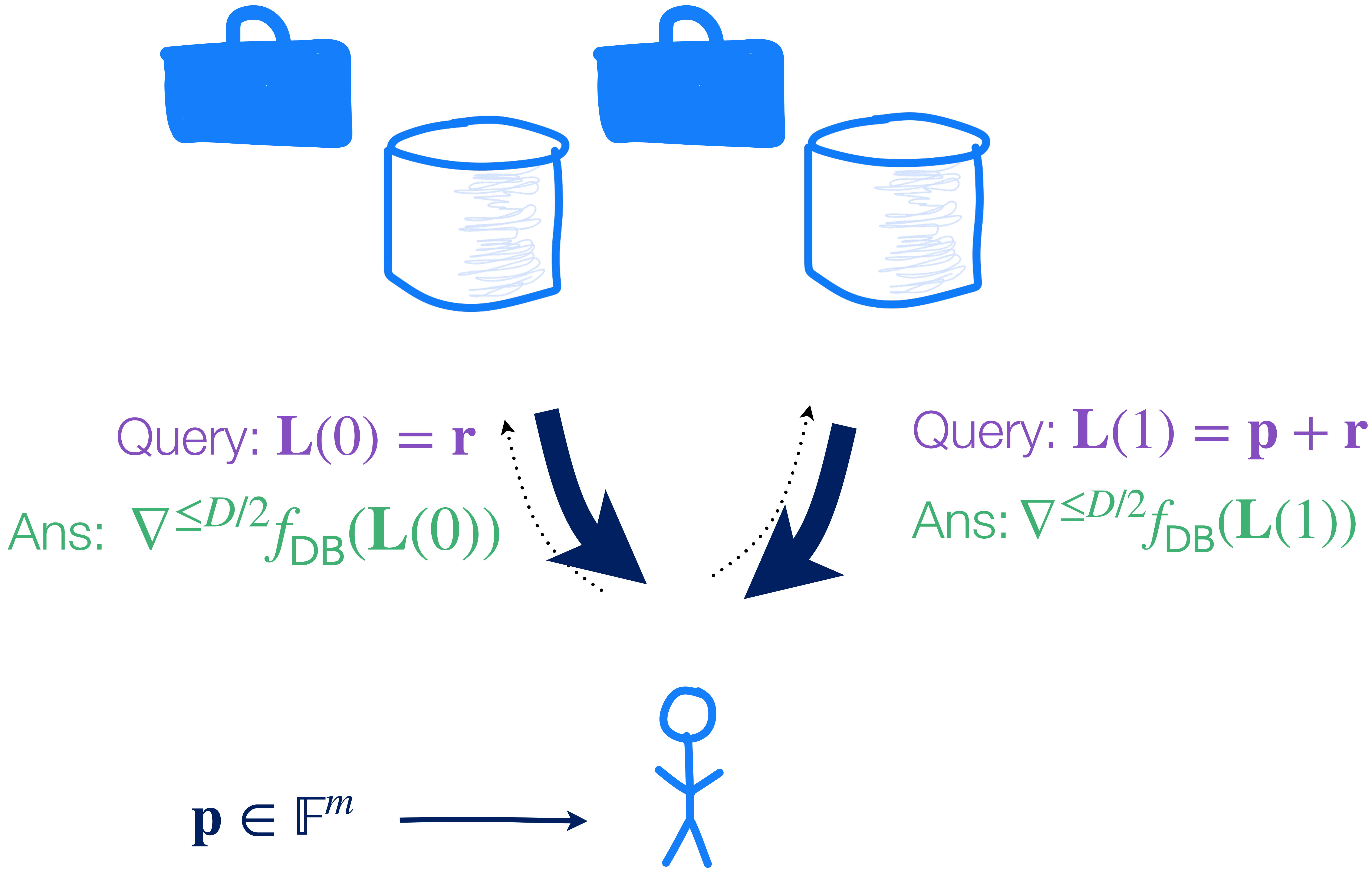
Field: $\mathbb{F}_2$

f_{DB} multilinear

$\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

$\binom{m}{D} \geq n$

$s = 2$ Servers: A Refresher



Query: $\mathbf{L}(0) = \mathbf{r}$

Ans: $\nabla^{\leq D/2} f_{\mathsf{DB}}(\mathbf{L}(0))$

Query: $\mathbf{L}(1) = \mathbf{p} + \mathbf{r}$

Ans: $\nabla^{\leq D/2} f_{\mathsf{DB}}(\mathbf{L}(1))$

$\mathbf{p} \in \mathbb{F}^m$

Cheatsheet

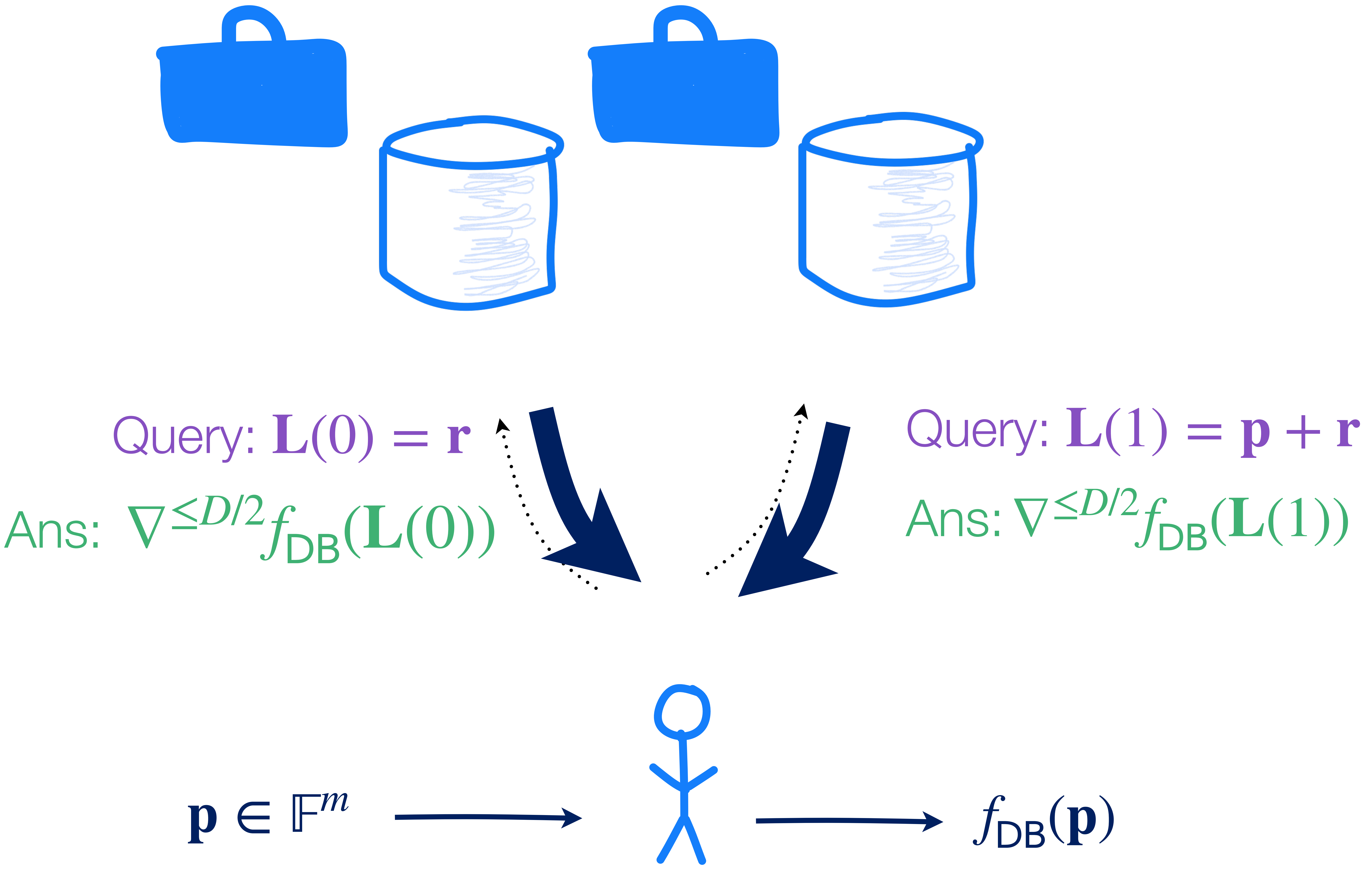
Field: $\mathbb{F}_2$

f_{DB} multilinear

$\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

$\binom{m}{D} \geq n$

$s = 2$ Servers: A Refresher



Query: $\mathbf{L}(0) = \mathbf{r}$

Ans: $\nabla^{\leq D/2} f_{\mathsf{DB}}(\mathbf{L}(0))$

Query: $\mathbf{L}(1) = \mathbf{p} + \mathbf{r}$

Ans: $\nabla^{\leq D/2} f_{\mathsf{DB}}(\mathbf{L}(1))$

$\mathbf{p} \in \mathbb{F}^m \longrightarrow$ $\longrightarrow f_{\mathsf{DB}}(\mathbf{p})$

Cheatsheet

Field: $\mathbb{F}_2$

f_{DB} multilinear

$\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

$\binom{m}{D} \geq n$

$s > 2$ Servers: What Changes? [GLMDS25]

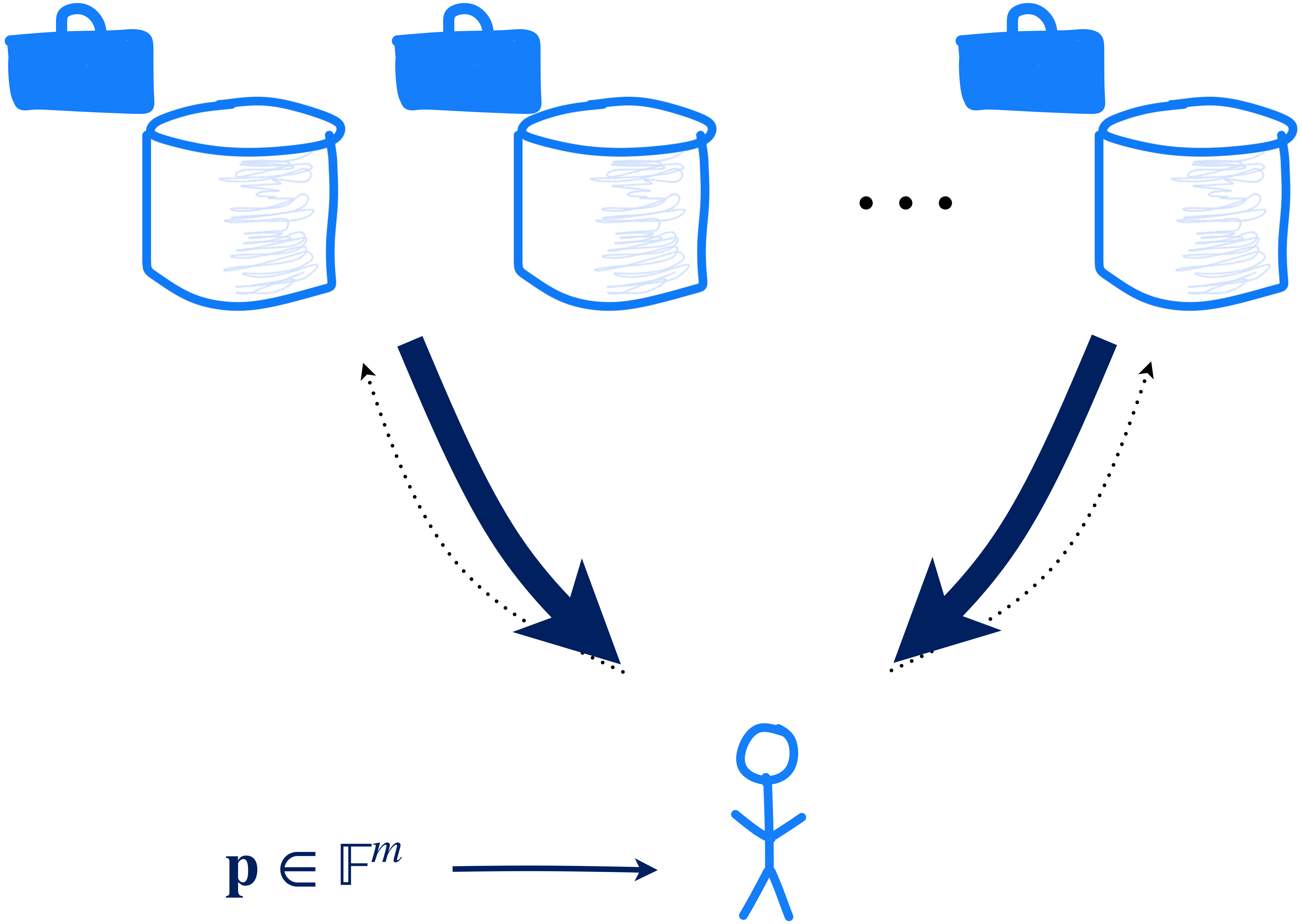
Cheatsheet

Field: $\mathbb{F}_q$ (for $q \geq s$)

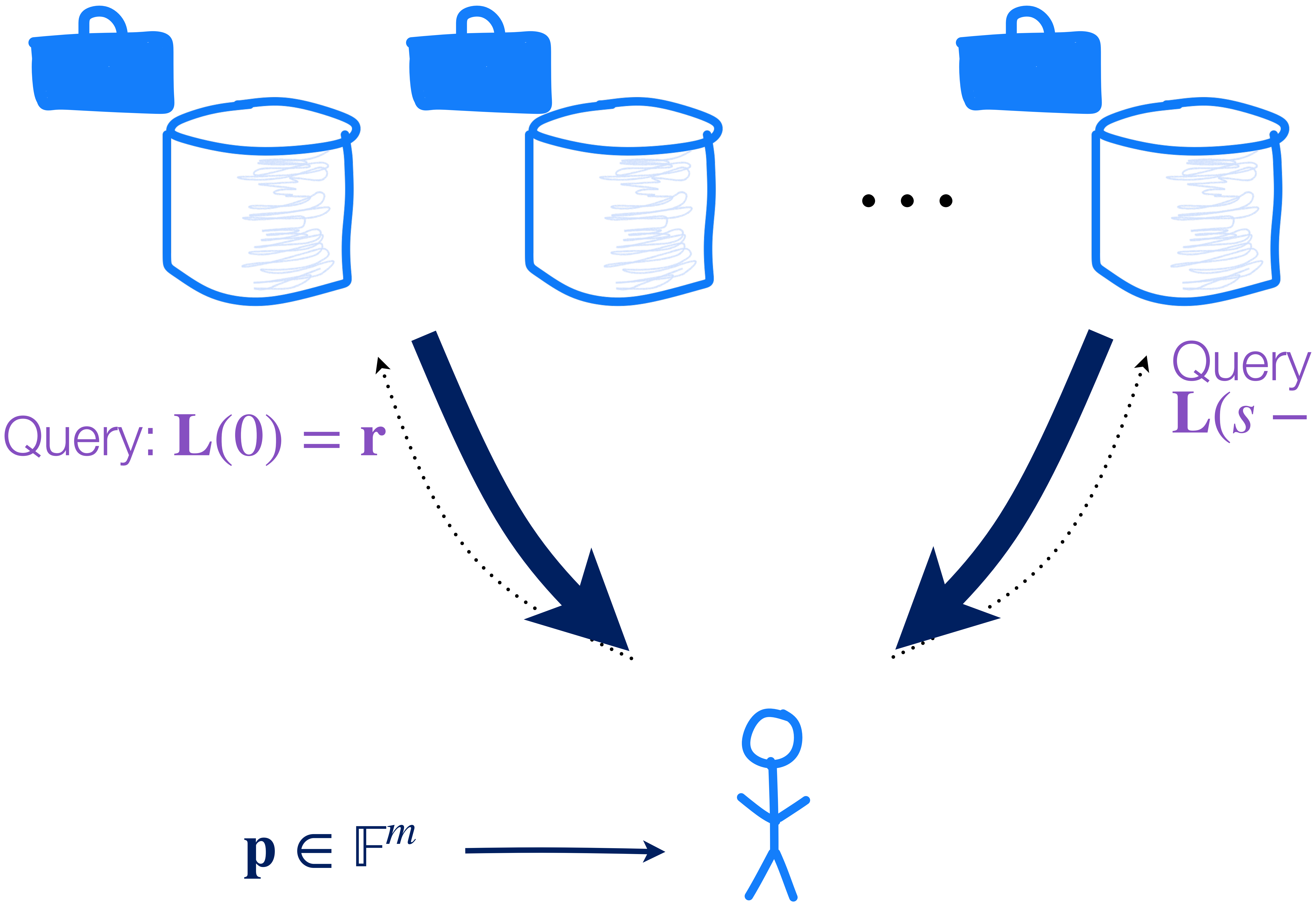
f_{DB} multilinear

$\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

$\binom{m}{D} \geq n$



$s > 2$ Servers: What Changes? [GLMDS25]



Query: $\mathbf{L}(0) = \mathbf{r}$

Query: $\mathbf{L}(s-1) = \mathbf{r} + (s-1) \cdot \mathbf{p}$

$\mathbf{p} \in \mathbb{F}^m$

Cheatsheet

Field: $\mathbb{F}_q$ (for $q \geq s$)

f_{DB} multilinear

$\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

$\binom{m}{D} \geq n$

$s > 2$ Servers: What Changes? [GLMDS25]

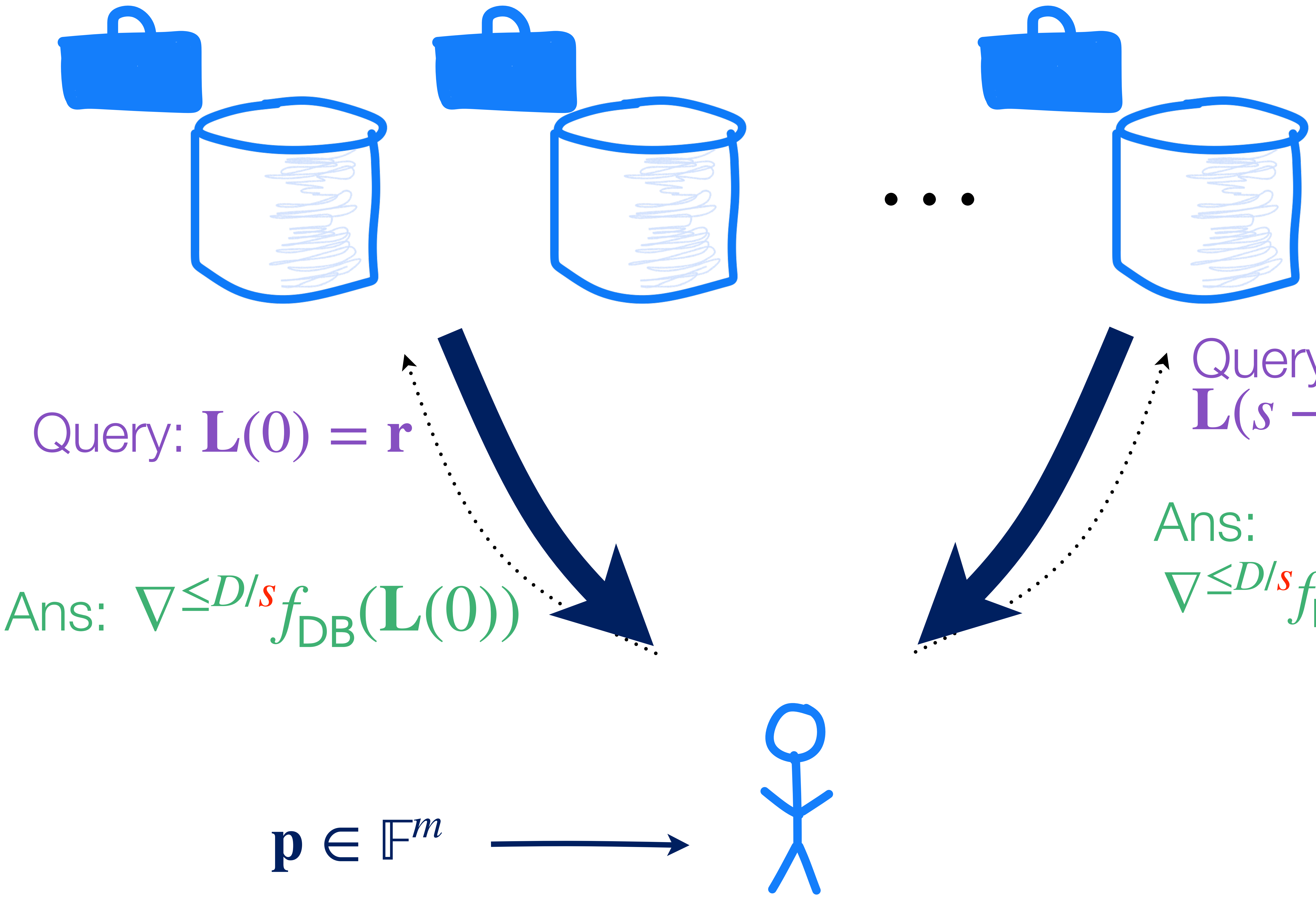
Cheatsheet

Field: $\mathbb{F}_q$ (for $q \geq s$)

f_{DB} multilinear

$\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

$\binom{m}{D} \geq n$



Query: $\mathbf{L}(0) = \mathbf{r}$

Ans: $\nabla^{\leq D/s} f_{\text{DB}}(\mathbf{L}(0))$

Query: $\mathbf{L}(s-1) = \mathbf{r} + (s-1) \cdot \mathbf{p}$

Ans: $\nabla^{\leq D/s} f_{\text{DB}}(\mathbf{L}(s-1))$

$\mathbf{p} \in \mathbb{F}^m$

$s > 2$ Servers: What Changes? [GLMDS25]

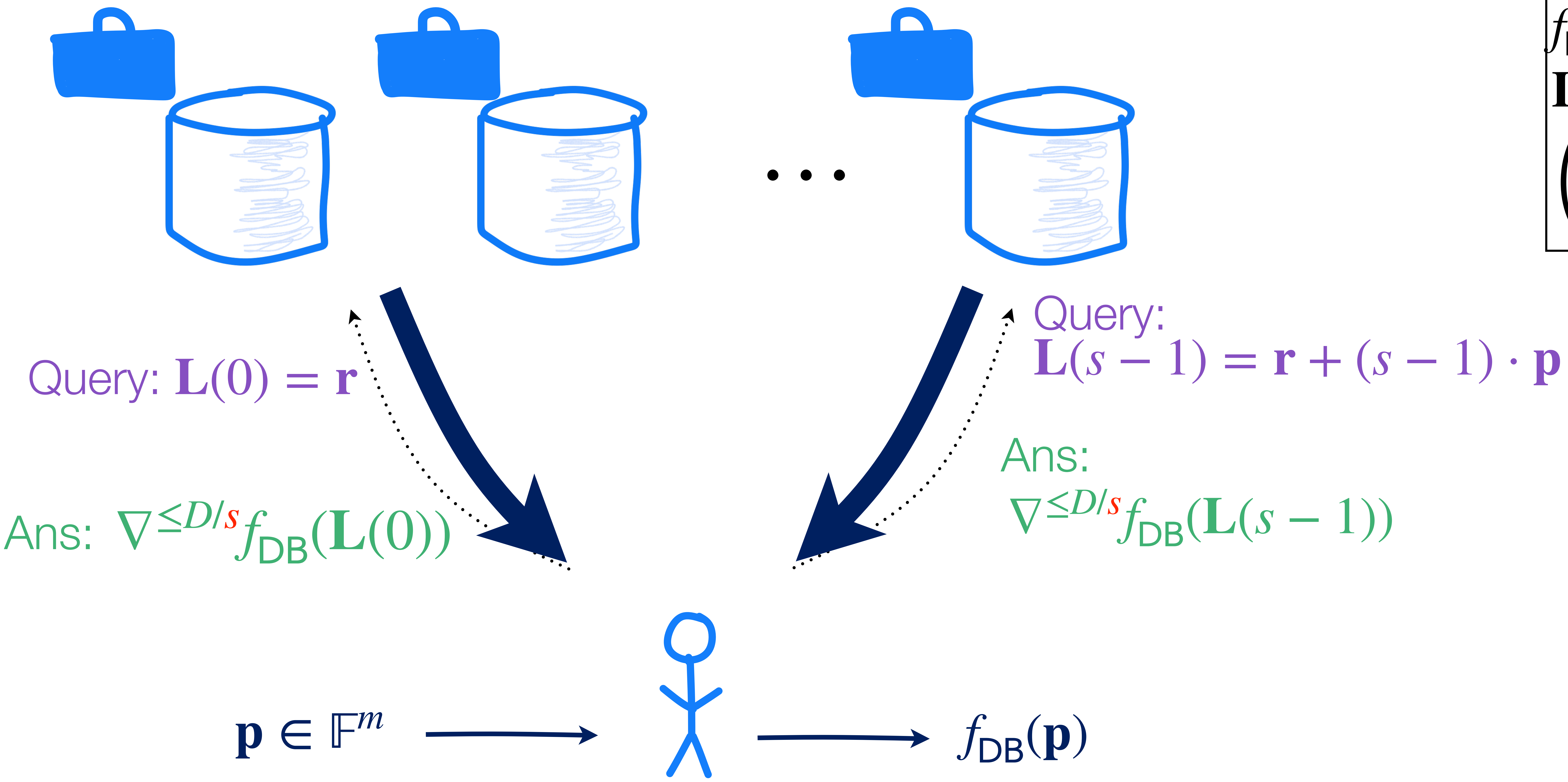
Cheatsheet

Field: $\mathbb{F}_q$ (for $q \geq s$)

f_{DB} multilinear

$\mathbf{L}(t) = \mathbf{r} + t \cdot \mathbf{p}$

$\binom{m}{D} \geq n$



Query: $\mathbf{L}(0) = \mathbf{r}$

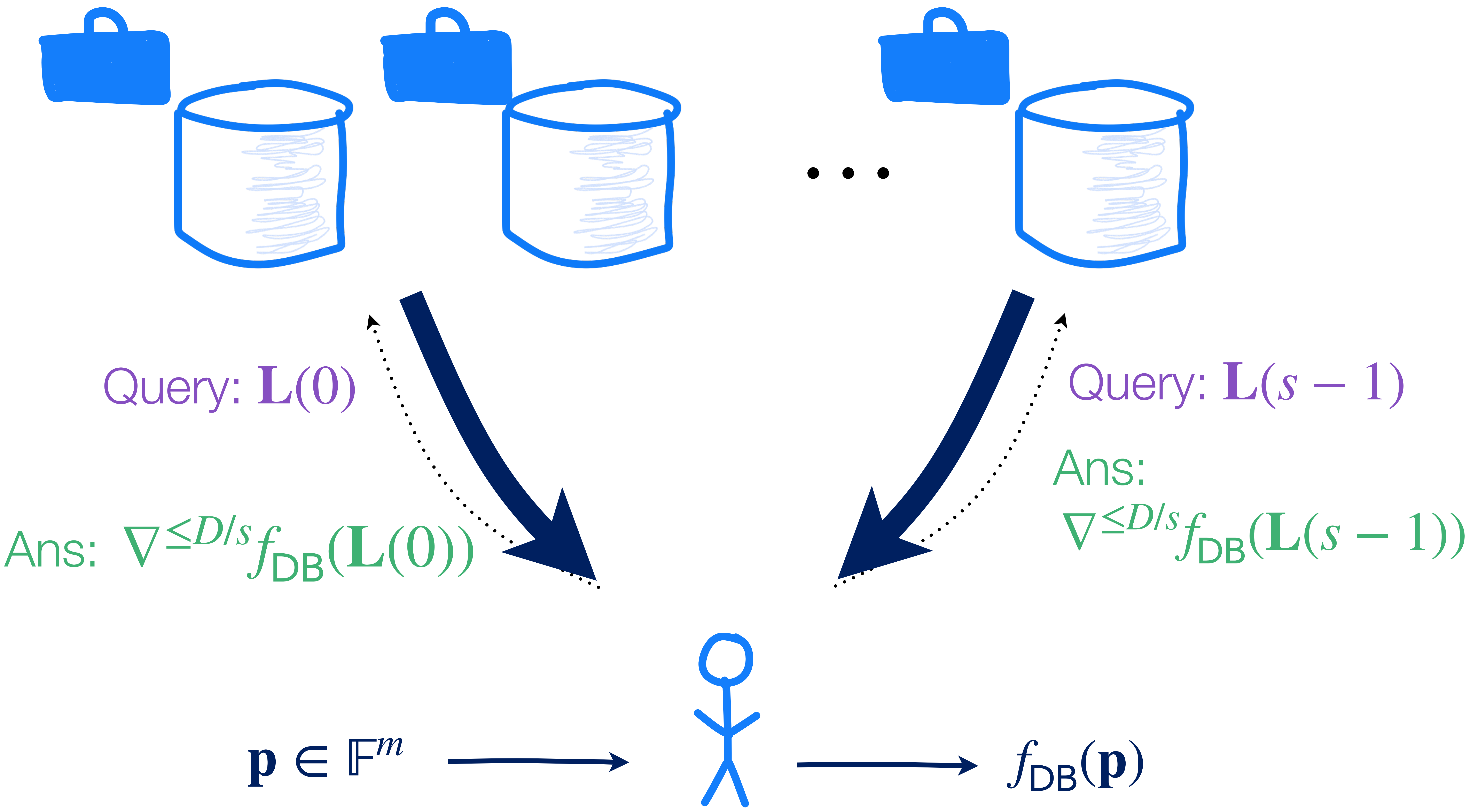
Ans: $\nabla^{\leq D/s} f_{\text{DB}}(\mathbf{L}(0))$

Query: $\mathbf{L}(s-1) = \mathbf{r} + (s-1) \cdot \mathbf{p}$

Ans: $\nabla^{\leq D/s} f_{\text{DB}}(\mathbf{L}(s-1))$

$\mathbf{p} \in \mathbb{F}^m \longrightarrow \quad \longrightarrow f_{\text{DB}}(\mathbf{p})$

Collusion Resistance via Shamir Secret Sharing [BIK05]



Query: $\mathbf{L}(0)$

Ans: $\nabla^{\leq D/s} f_{\mathsf{DB}}(\mathbf{L}(0))$

Query: $\mathbf{L}(s-1)$

Ans: $\nabla^{\leq D/s} f_{\mathsf{DB}}(\mathbf{L}(s-1))$

$\mathbf{p} \in \mathbb{F}^m$

$f_{\mathsf{DB}}(\mathbf{p})$

Cheatsheet

Field: $\mathbb{F}_q$ (for $q \geq s$)

f_{DB} multilinear

$\binom{m}{D} \geq n$

Collusion Resistance via Shamir Secret Sharing [BIK05]

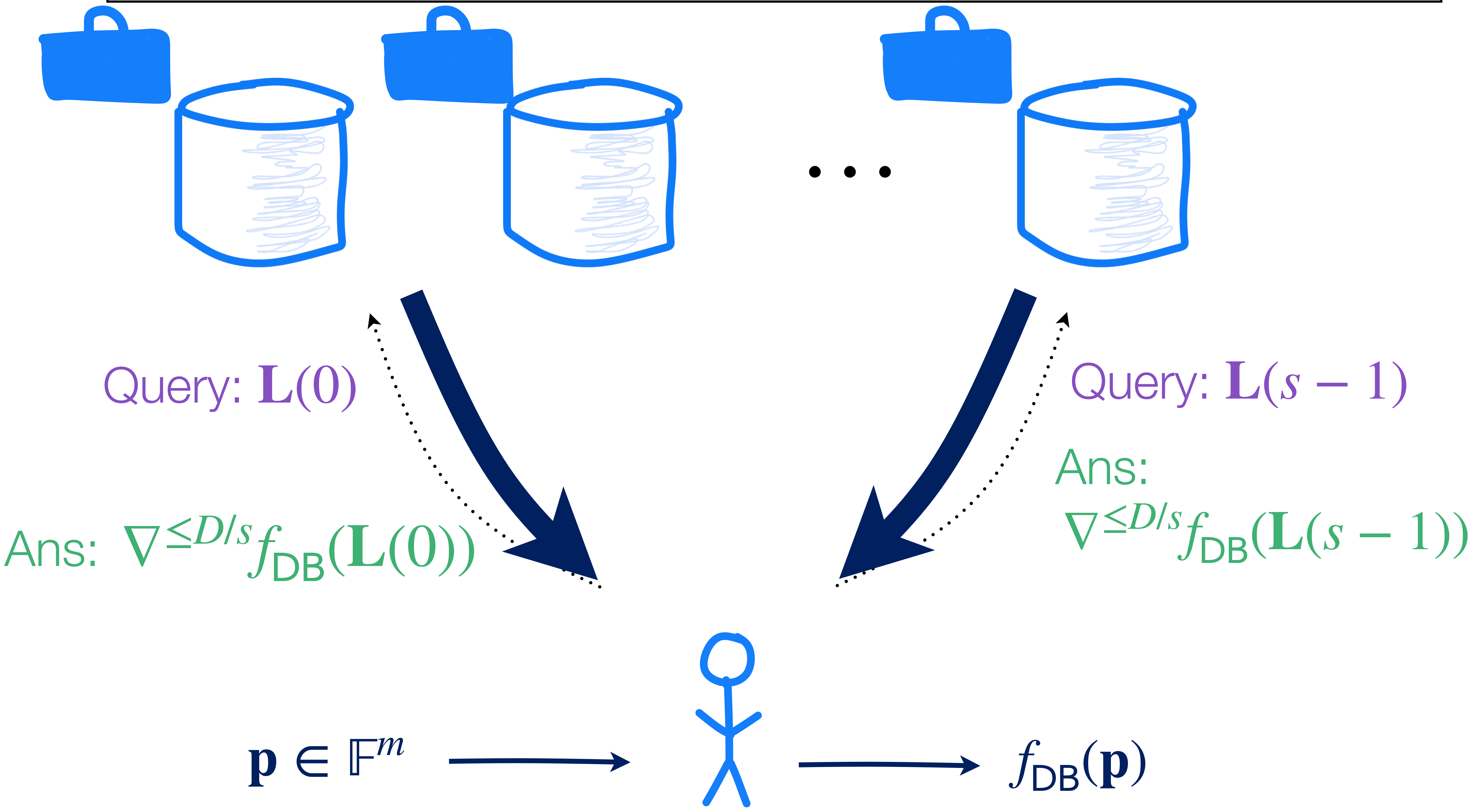
Security against 1 server: use a line of slope $\mathbf{p}$

Cheatsheet

Field: $\mathbb{F}_q$ (for $q \geq s$)

f_{DB} multilinear

$\binom{m}{D} \geq n$



Query: $\mathbf{L}(0)$

Ans: $\nabla^{\leq D/s} f_{\mathsf{DB}}(\mathbf{L}(0))$

Query: $\mathbf{L}(s-1)$

Ans: $\nabla^{\leq D/s} f_{\mathsf{DB}}(\mathbf{L}(s-1))$

$\mathbf{p} \in \mathbb{F}^m$

$f_{\mathsf{DB}}(\mathbf{p})$

Collusion Resistance via Shamir Secret Sharing [BIK05]

Security against 1 server: use a line of slope $\mathbf{p}$

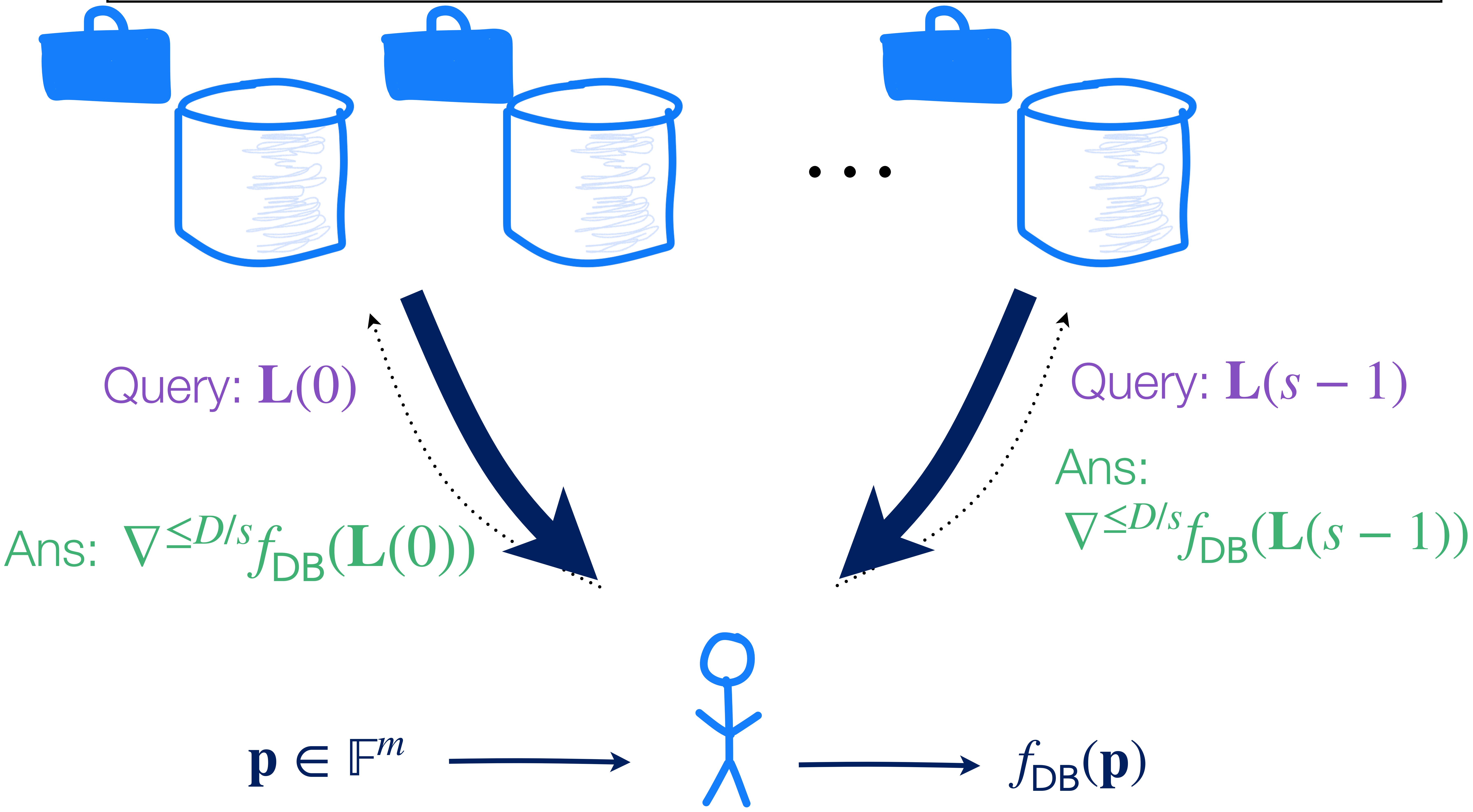
Security against c colluding servers: use a degree $\mathbf{c}$ curve of “slope” $\mathbf{p}$

Cheatsheet

Field: $\mathbb{F}_q$ (for $q \geq s$)

f_{DB} multilinear

$$\binom{m}{D} \geq n$$



Query: $\mathbf{L}(0)$

Ans: $\nabla^{\leq D/s} f_{\mathrm{DB}}(\mathbf{L}(0))$

Query: $\mathbf{L}(s-1)$

Ans: $\nabla^{\leq D/s} f_{\mathrm{DB}}(\mathbf{L}(s-1))$

$\mathbf{p} \in \mathbb{F}^m \longrightarrow$ $\longrightarrow f_{\mathrm{DB}}(\mathbf{p})$

Collusion Resistance via Shamir Secret Sharing [BIK05]

Security against 1 server: use a line of slope $\mathbf{p}$

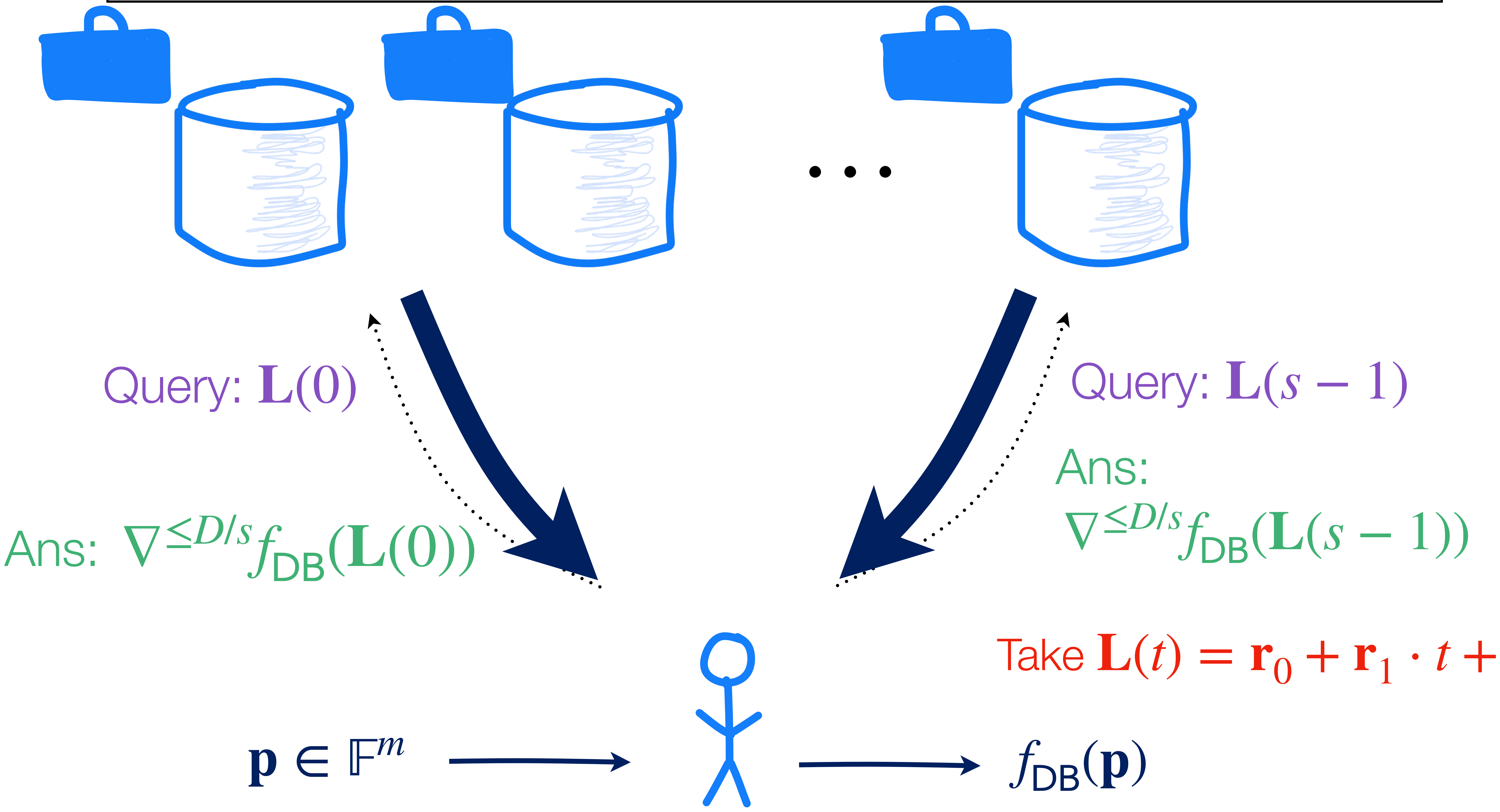
Security against c colluding servers: use a degree $\mathbf{c}$ curve of "slope" $\mathbf{p}$

Cheatsheet

Field: $\mathbb{F}_q$ (for $q \geq s$)

f_{DB} multilinear

$\binom{m}{D} \geq n$



Query: $\mathbf{L}(0)$

Ans: $\nabla^{\leq D/s} f_{\text{DB}}(\mathbf{L}(0))$

Query: $\mathbf{L}(s-1)$

Ans: $\nabla^{\leq D/s} f_{\text{DB}}(\mathbf{L}(s-1))$

Take $\mathbf{L}(t) = \mathbf{r}_0 + \mathbf{r}_1 \cdot t + \ldots + \mathbf{r}_{c-1} \cdot t^{c-1} + \mathbf{p} \cdot t^c$

$\mathbf{p} \in \mathbb{F}^m \longrightarrow$ $\longrightarrow f_{\text{DB}}(\mathbf{p})$

Collusion Resistance via Shamir Secret Sharing [BIK05]

Security against 1 server: use a line of slope $\mathbf{p}$

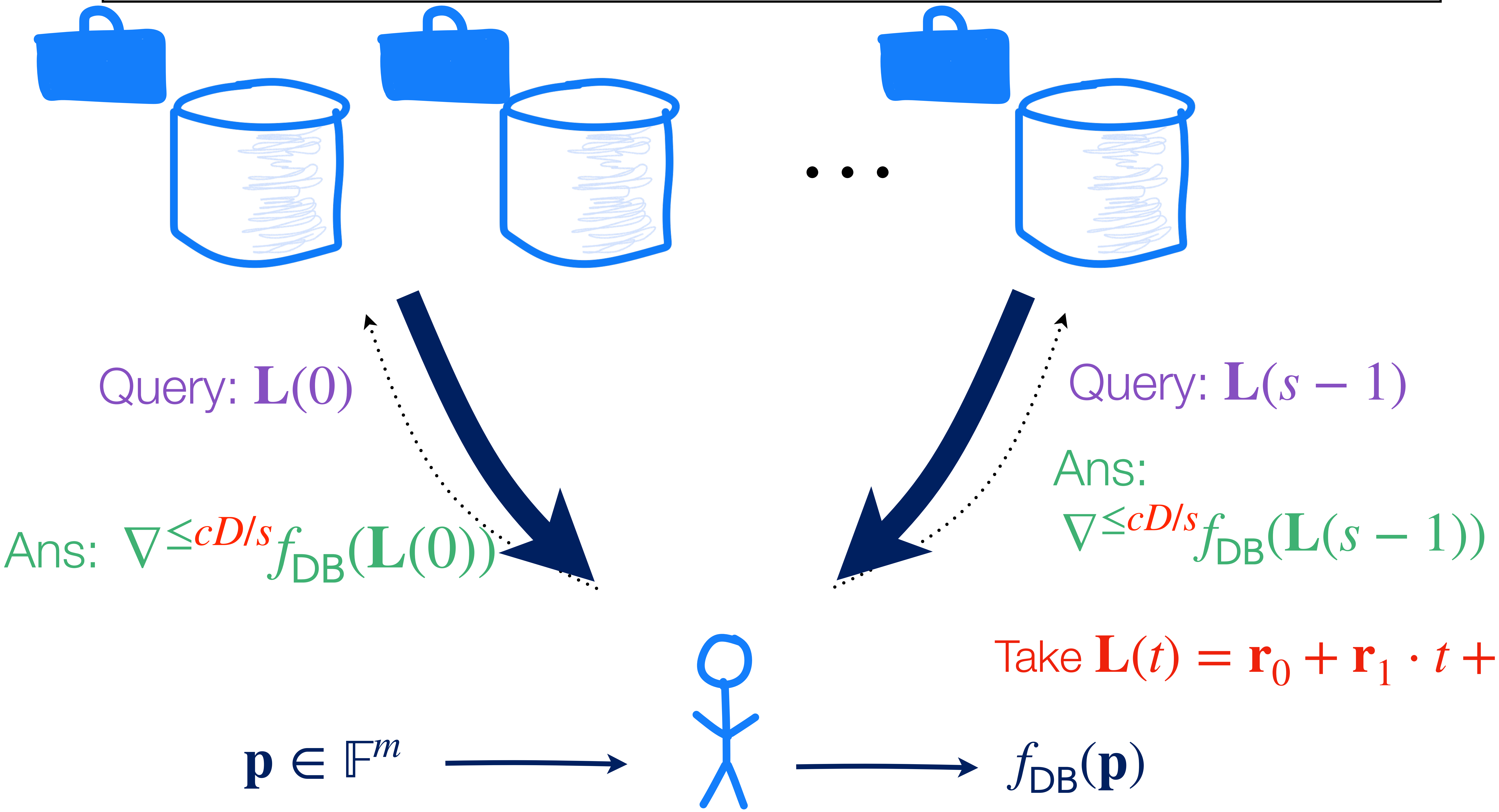
Security against c colluding servers: use a degree $\mathbf{c}$ curve of “slope” $\mathbf{p}$

Cheatsheet

Field: $\mathbb{F}_q$ (for $q \geq s$)

f_{DB} multilinear

$\binom{m}{D} \geq n$



Query: $\mathbf{L}(0)$

Ans: $\nabla^{\leq cD/s} f_{\mathsf{DB}}(\mathbf{L}(0))$

Query: $\mathbf{L}(s-1)$

Ans: $\nabla^{\leq cD/s} f_{\mathsf{DB}}(\mathbf{L}(s-1))$

Take $\mathbf{L}(t) = \mathbf{r}_0 + \mathbf{r}_1 \cdot t + \ldots + \mathbf{r}_{c-1} \cdot t^{c-1} + \mathbf{p} \cdot t^c$

$\mathbf{p} \in \mathbb{F}^m \longrightarrow$ $\longrightarrow f_{\mathsf{DB}}(\mathbf{p})$

$s > 2$ Servers: Our Improvements

$s > 2$ Servers: Our Improvements

- New idea 1: the same finite differences technique as in the 2-server case!

$s > 2$ Servers: Our Improvements

- New idea 1: the same finite differences technique as in the 2-server case!
- New idea 2: vary the **individual** degree of f_{DB}

Role of Individual Degree

Role of Individual Degree

- Up to now: $f_{\mathsf{DB}} : \mathbb{F}^m \to \mathbb{F}$ multilinear with total degree D

Role of Individual Degree

- Up to now: $f_{\mathsf{DB}} : \mathbb{F}^m \to \mathbb{F}$ multilinear with total degree D
  - $\binom{m}{D}$ degrees of freedom → $\binom{m}{D} \geq n$

Role of Individual Degree

- Up to now: $f_{\mathsf{DB}} : \mathbb{F}^m \to \mathbb{F}$ multilinear with total degree D
  - $\binom{m}{D}$ degrees of freedom → $\binom{m}{D} \geq n$
  - Number of derivatives: $\binom{m}{D/s}$

Role of Individual Degree

- Up to now: $f_{\mathsf{DB}} : \mathbb{F}^m \to \mathbb{F}$ multilinear with total degree D
  - $\binom{m}{D}$ degrees of freedom → $\binom{m}{D} \geq n$
  - Number of derivatives: $\binom{m}{D/s}$
- Generalisation: what if we let f_{DB} have higher individual degree $d > 1$?

Role of Individual Degree

- Up to now: $f_{\mathsf{DB}} : \mathbb{F}^m \rightarrow \mathbb{F}$ multilinear with total degree D
  - $\binom{m}{D}$ degrees of freedom → $\binom{m}{D} \geq n$
  - Number of derivatives: $\binom{m}{D/s}$
- Generalisation: what if we let f_{DB} have higher individual degree $d > 1$?
  - 😁: more degrees of freedom → larger database!

Role of Individual Degree

- Up to now: $f_{\mathsf{DB}} : \mathbb{F}^m \to \mathbb{F}$ multilinear with total degree D
  - $\binom{m}{D}$ degrees of freedom → $\binom{m}{D} \geq n$
  - Number of derivatives: $\binom{m}{D/s}$
- Generalisation: what if we let f_{DB} have higher individual degree $d > 1$?
  - 😁: more degrees of freedom → larger database!
  - 😭: also more derivatives to send → more time and communication per query

Role of Individual Degree

- Up to now: $f_{\mathsf{DB}} : \mathbb{F}^m \to \mathbb{F}$ multilinear with total degree D
  - $\binom{m}{D}$ degrees of freedom → $\binom{m}{D} \geq n$
  - Number of derivatives: $\binom{m}{D/s}$
- Generalisation: what if we let f_{DB} have higher individual degree $d > 1$?
  - 😁: more degrees of freedom → larger database!
  - 😭: also more derivatives to send → more time and communication per query
- **TLDR: the sweet spot for d increases as the number of servers increases**

Special Case: Storage $n^{1+o(1)}$, Individual Degree $s-1$

Special Case: Storage $n^{1+o(1)}$, Individual Degree $s-1$

Theorem: for constant prime s, we get information-theoretic s-server PIR with:

Special Case: Storage $n^{1+o(1)}$, Individual Degree $s - 1$

Theorem: for constant prime s, we get information-theoretic s-server PIR with:

- server storage $n^{1+o(1)}$ and

Special Case: Storage $n^{1+o(1)}$, Individual Degree $s-1$

Theorem: for constant prime s, we get information-theoretic s-server PIR with:

- server storage $n^{1+o(1)}$ and
- communication and time per query $n^{\alpha+o(1)}$, where

Special Case: Storage $n^{1+o(1)}$, Individual Degree $s-1$

Theorem: for constant prime s, we get information-theoretic s-server PIR with:

- server storage $n^{1+o(1)}$ and
- communication and time per query $n^{\alpha+o(1)}$, where

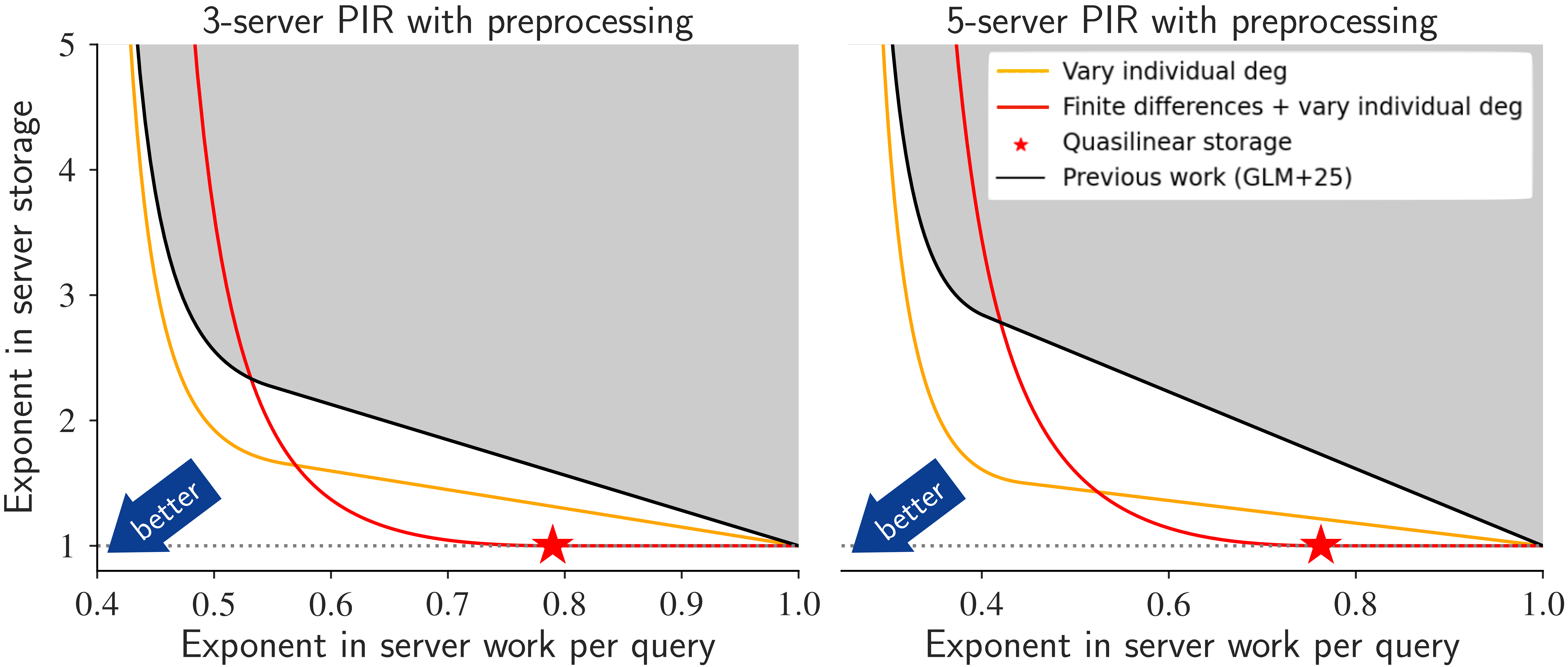
$$\alpha = 1 + \frac{1}{\log s} - \left(\frac{s+1}{2s}\right)\frac{\log(s+1)}{\log s} \approx \frac{1}{2} + \frac{1}{\log s} \text{ as } s \text{ grows}$$

Special Case: Storage $n^{1+o(1)}$, Individual Degree $s - 1$

Theorem: for constant prime s, we get information-theoretic s-server PIR with:

- server storage $n^{1+o(1)}$ and
- communication and time per query $n^{\alpha+o(1)}$, where

$$\alpha = 1 + \frac{1}{\log s} - \left(\frac{s+1}{2s}\right)\frac{\log(s+1)}{\log s} \approx \frac{1}{2} + \frac{1}{\log s} \text{ as } s \text{ grows}$$


3-server PIR with preprocessing
5-server PIR with preprocessing
Exponent in server storage
5
4
3
2
1
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Exponent in server work per query
0.4
0.6
0.8
1.0
Exponent in server work per query
better
better
Vary individual deg
Finite differences + vary individual deg
Quasilinear storage
Previous work (GLM+25)

Bonus Slides on Locally Decodable Codes

Smooth Locally Decodable Codes [KT00]

Smooth Locally Decodable Codes [KT00]

- Goal: encode a message in such a way that we can recover any bit of the message with a small number of accesses to the codeword

Smooth Locally Decodable Codes [KT00]

- Goal: encode a message in such a way that we can recover any bit of the message with a small number of accesses to the codeword
- Three functions:

Smooth Locally Decodable Codes [KT00]

- Goal: encode a message in such a way that we can recover any bit of the message with a small number of accesses to the codeword
- Three functions:
  - Encode(msg $\in \{0,1\}^n) \rightarrow c \in \Sigma^\ell$

Cheatsheet

n: message length

ℓ: codeword length

Σ: alphabet

q: locality

Smooth Locally Decodable Codes [KT00]

- Goal: encode a message in such a way that we can recover any bit of the message with a small number of accesses to the codeword
- Three functions:
    - $\mathsf{Encode}(\mathsf{msg} \in \{0,1\}^n) \to c \in \Sigma^\ell$
    - Randomised $\mathsf{Query}(i \in [n]) \to \mathsf{qu} \in [\ell]^q$

Cheatsheet

n: message length

ℓ: codeword length

Σ: alphabet

q: locality

Smooth Locally Decodable Codes [KT00]

- Goal: encode a message in such a way that we can recover any bit of the message with a small number of accesses to the codeword
- Three functions:
  - Encode(msg $\in \{0,1\}^n$) $\rightarrow c \in \Sigma^\ell$
  - Randomised Query($i \in [n]$) $\rightarrow$ qu $\in [\ell]^q$
  - Decode($i \in [n]$, qu $\in [\ell]^q$, c[qu]) $\rightarrow$ msg[i]

Cheatsheet

n: message length

ℓ: codeword length

Σ: alphabet

q: locality

Smooth Locally Decodable Codes [KT00]

- Goal: encode a message in such a way that we can recover any bit of the message with a small number of accesses to the codeword
- Three functions:
  - $\mathsf{Encode}(\mathsf{msg} \in \{0,1\}^n) \rightarrow c \in \Sigma^\ell$
  - Randomised $\mathsf{Query}(i \in [n]) \rightarrow \mathsf{qu} \in [\ell]^q$
  - $\mathsf{Decode}(i \in [n], \mathsf{qu} \in [\ell]^q, c[\mathsf{qu}]) \rightarrow \mathsf{msg}[i]$
- **Smoothness:** marginal distribution of qu_j is uniform over $[\ell]$ for all j

Cheatsheet

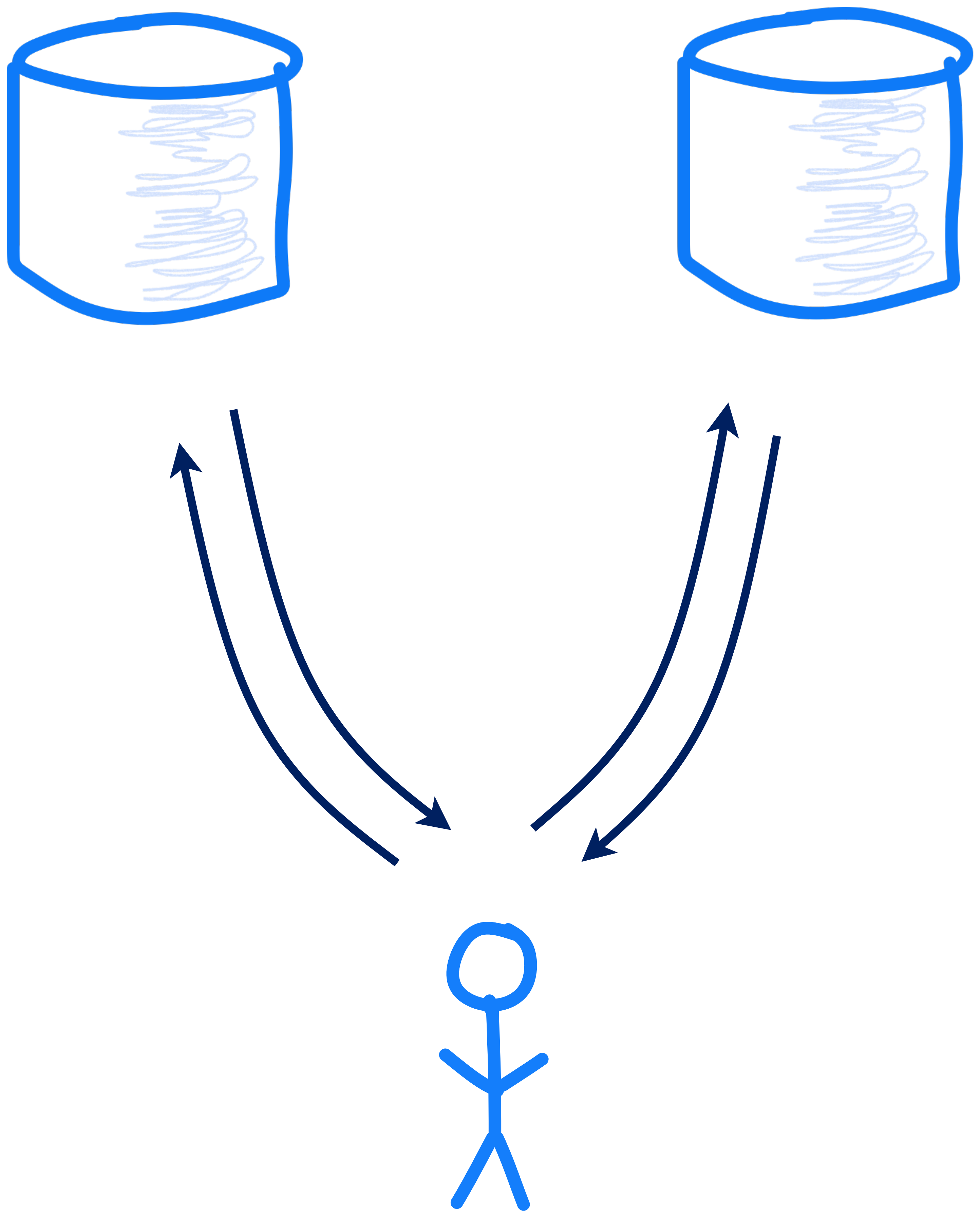
n: message length

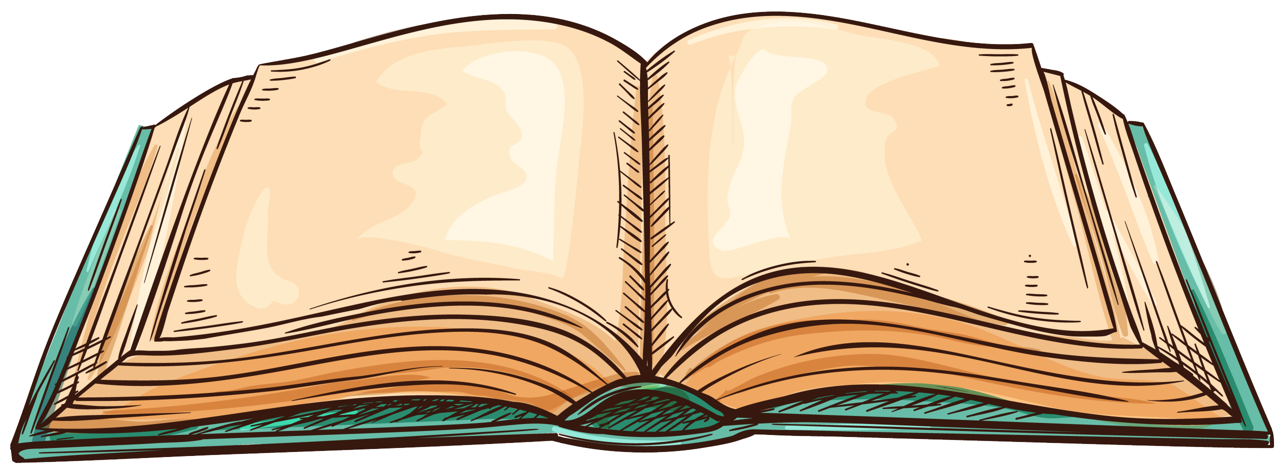
ℓ: codeword length

Σ: alphabet

q: locality

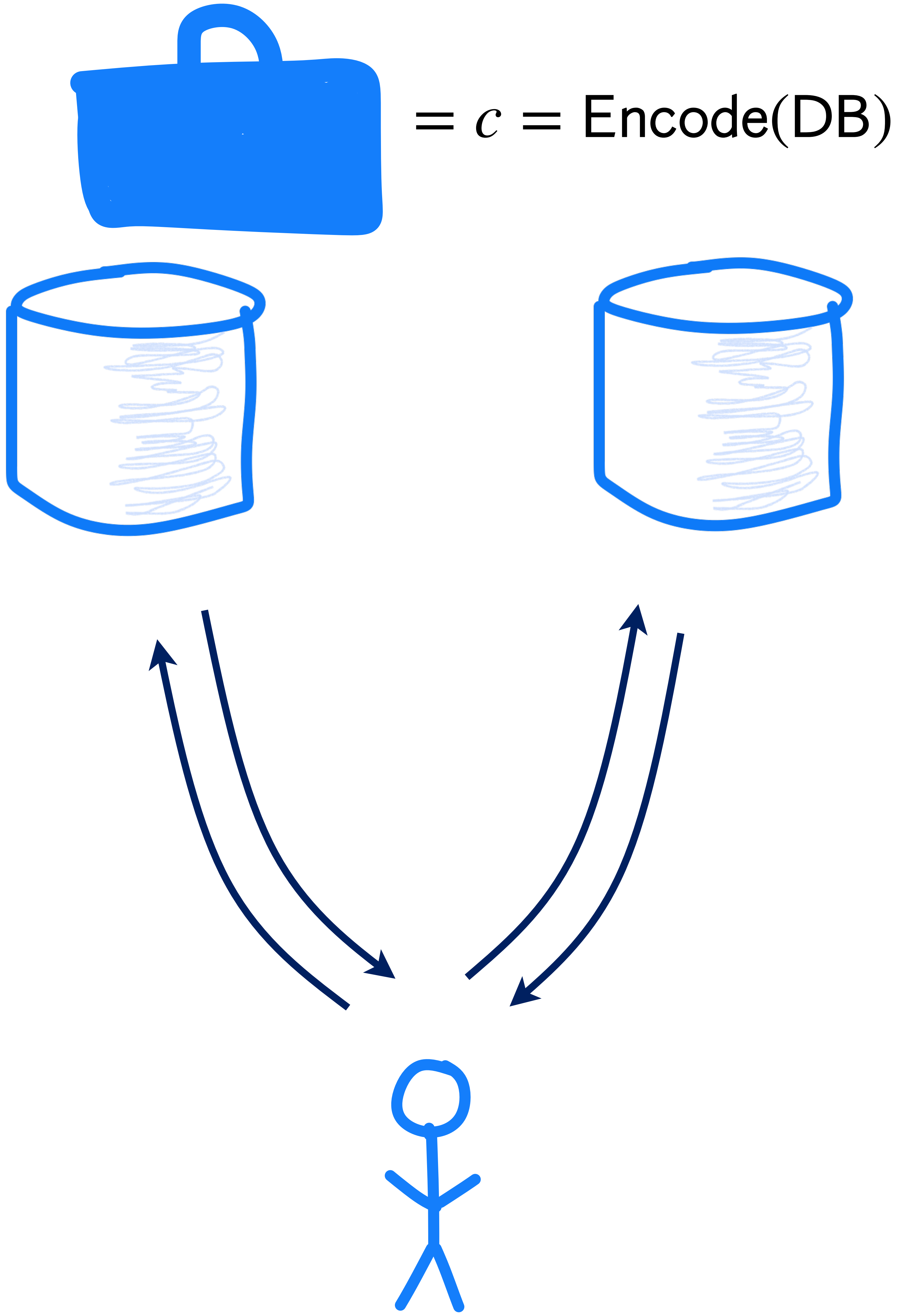
2-query LDCs → 2-server PIR



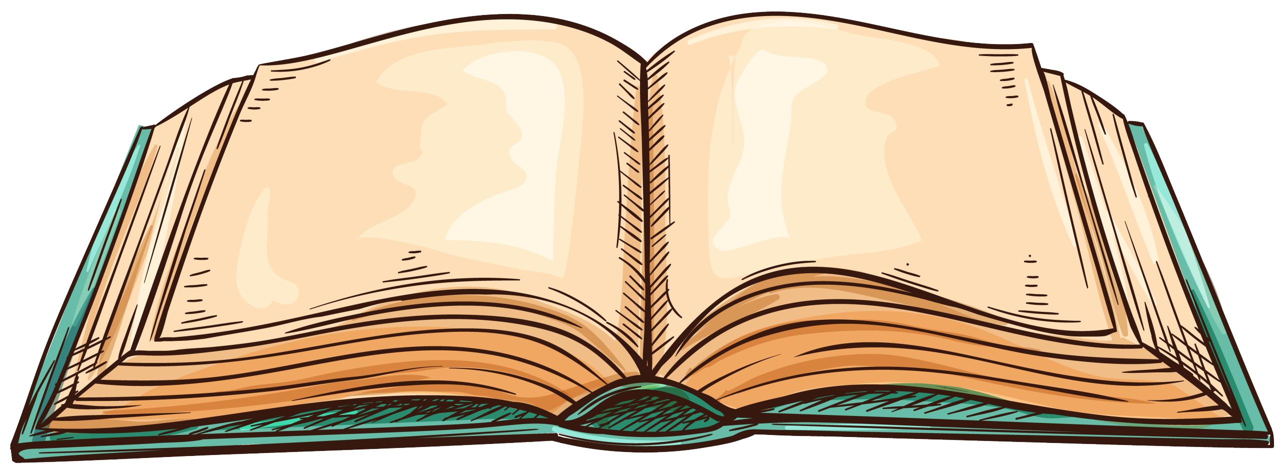


LDCs	PIR
Message	DB
Encoding	Preprocessing
Decoding	Reconstruction

2-query LDCs → 2-server PIR

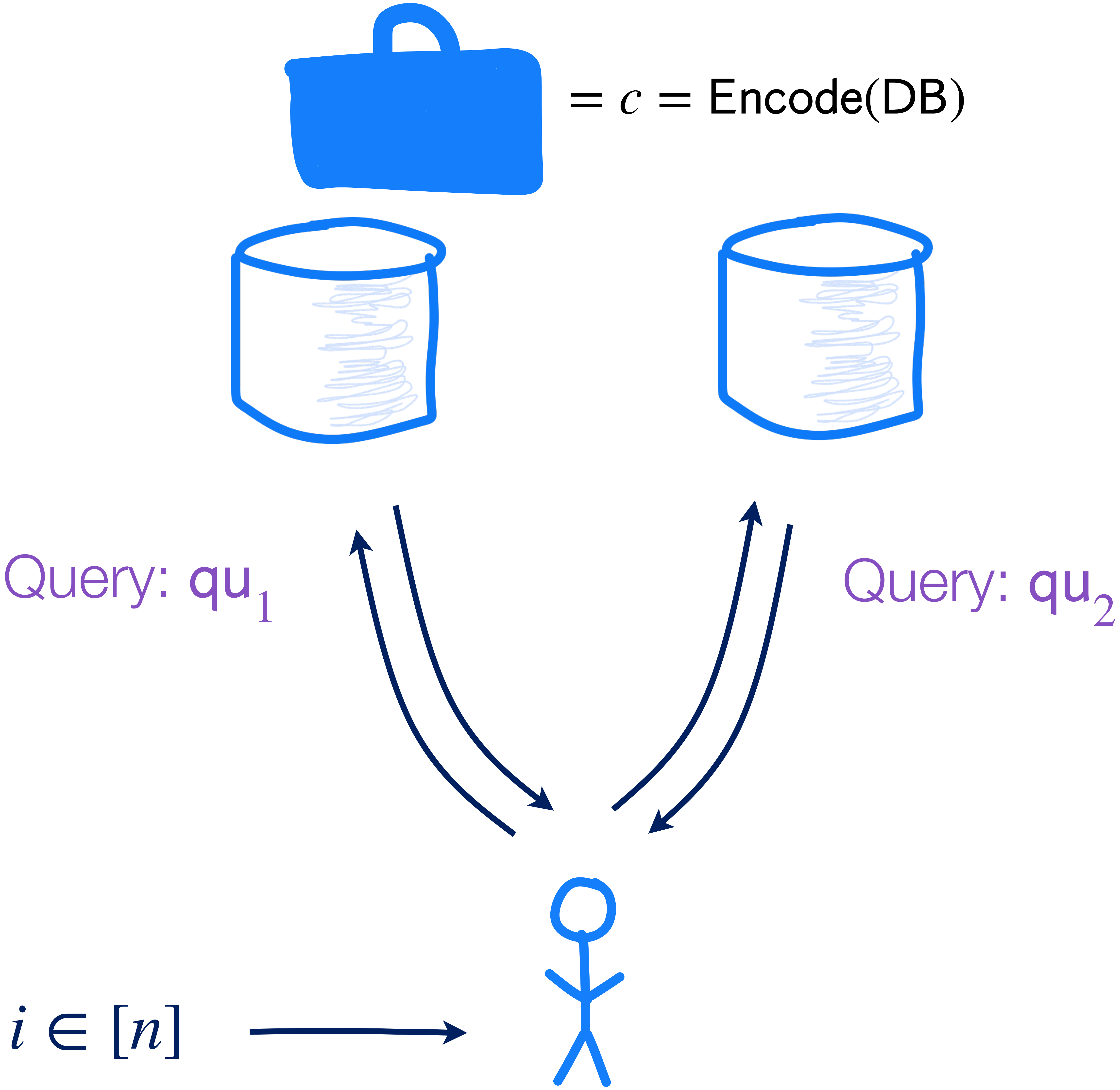


$= c =$ Encode(DB)



LDCs	PIR
Message	DB
Encoding	Preprocessing
Decoding	Reconstruction

2-query LDCs → 2-server PIR

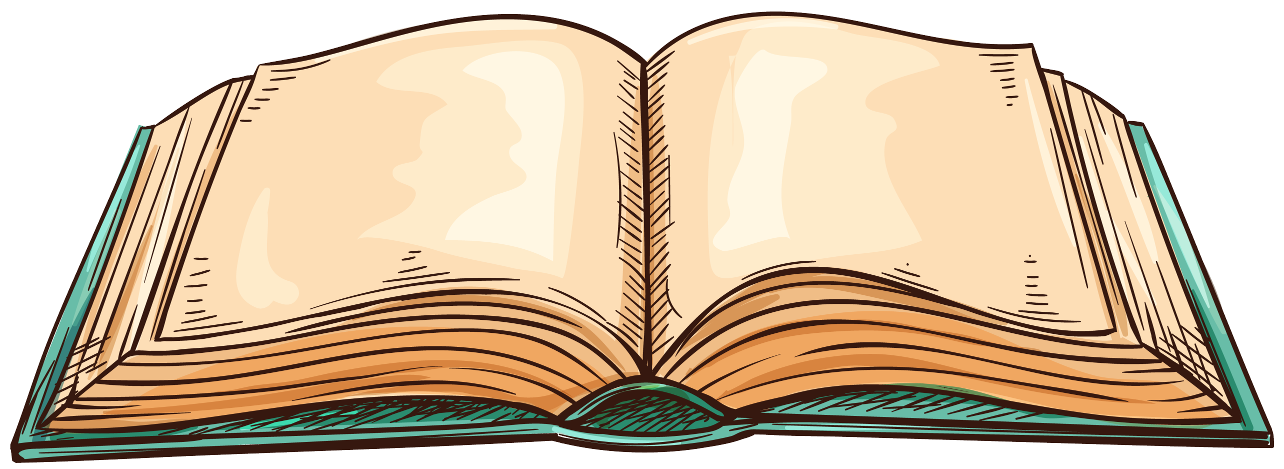


$= c =$ Encode(DB)

Query: $\mathbf{qu}_1$

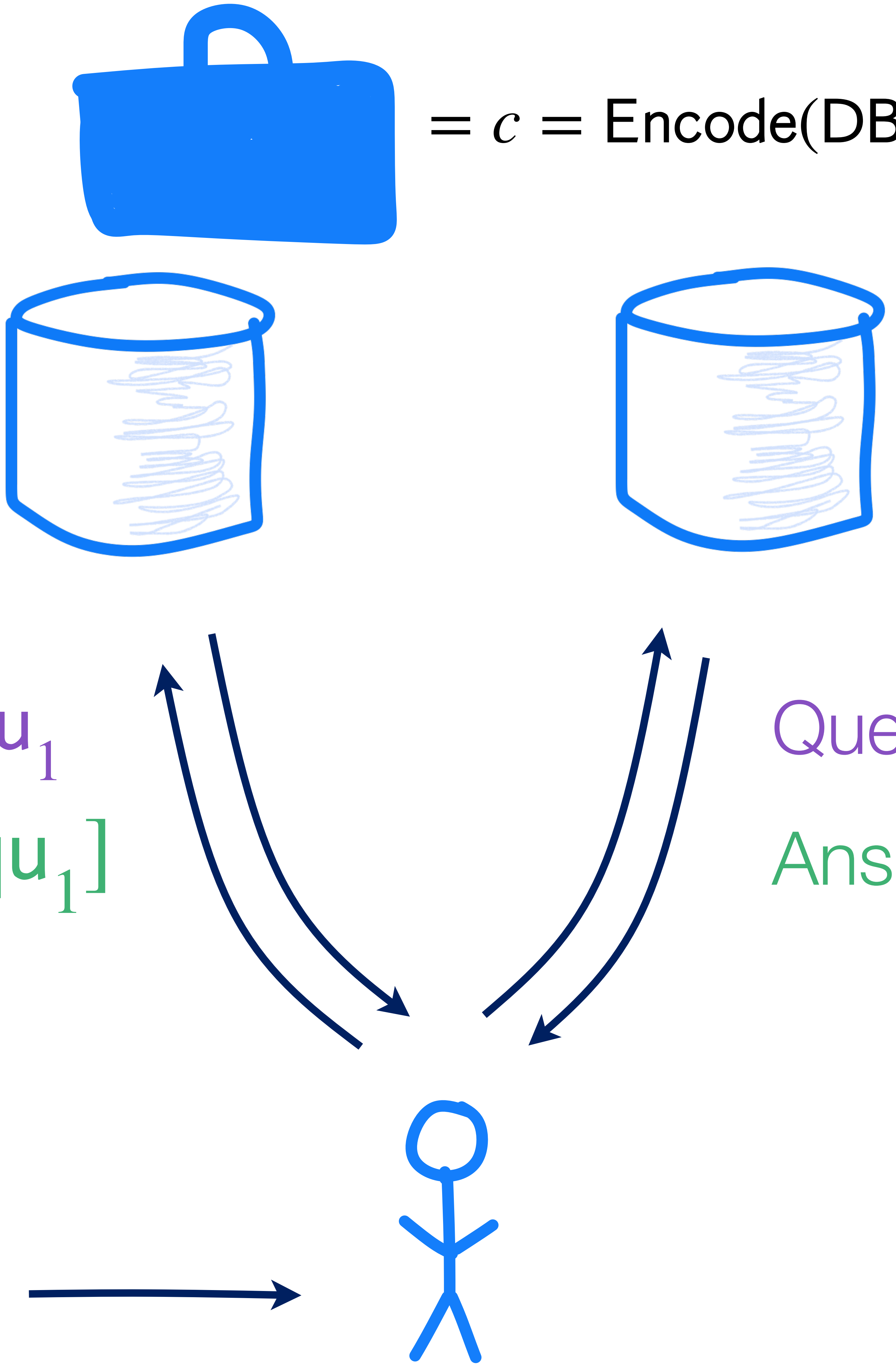
Query: $\mathbf{qu}_2$

$i \in [n]$



LDCs	PIR
Message	DB
Encoding	Preprocessing
Decoding	Reconstruction

2-query LDCs → 2-server PIR



$= c =$ Encode(DB)

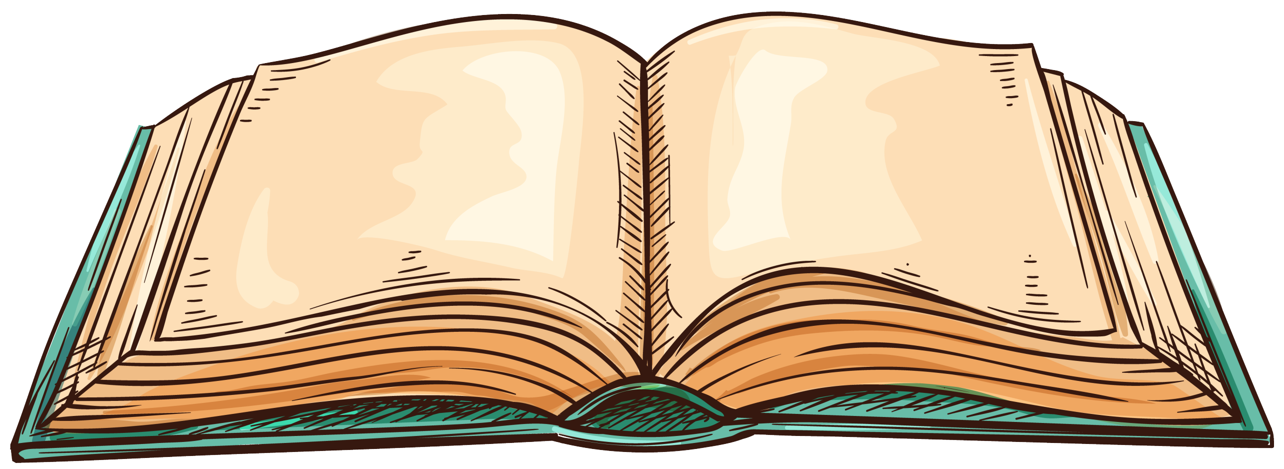
Query: $\mathbf{qu}_1$

Ans: $c[\mathbf{qu}_1]$

Query: $\mathbf{qu}_2$

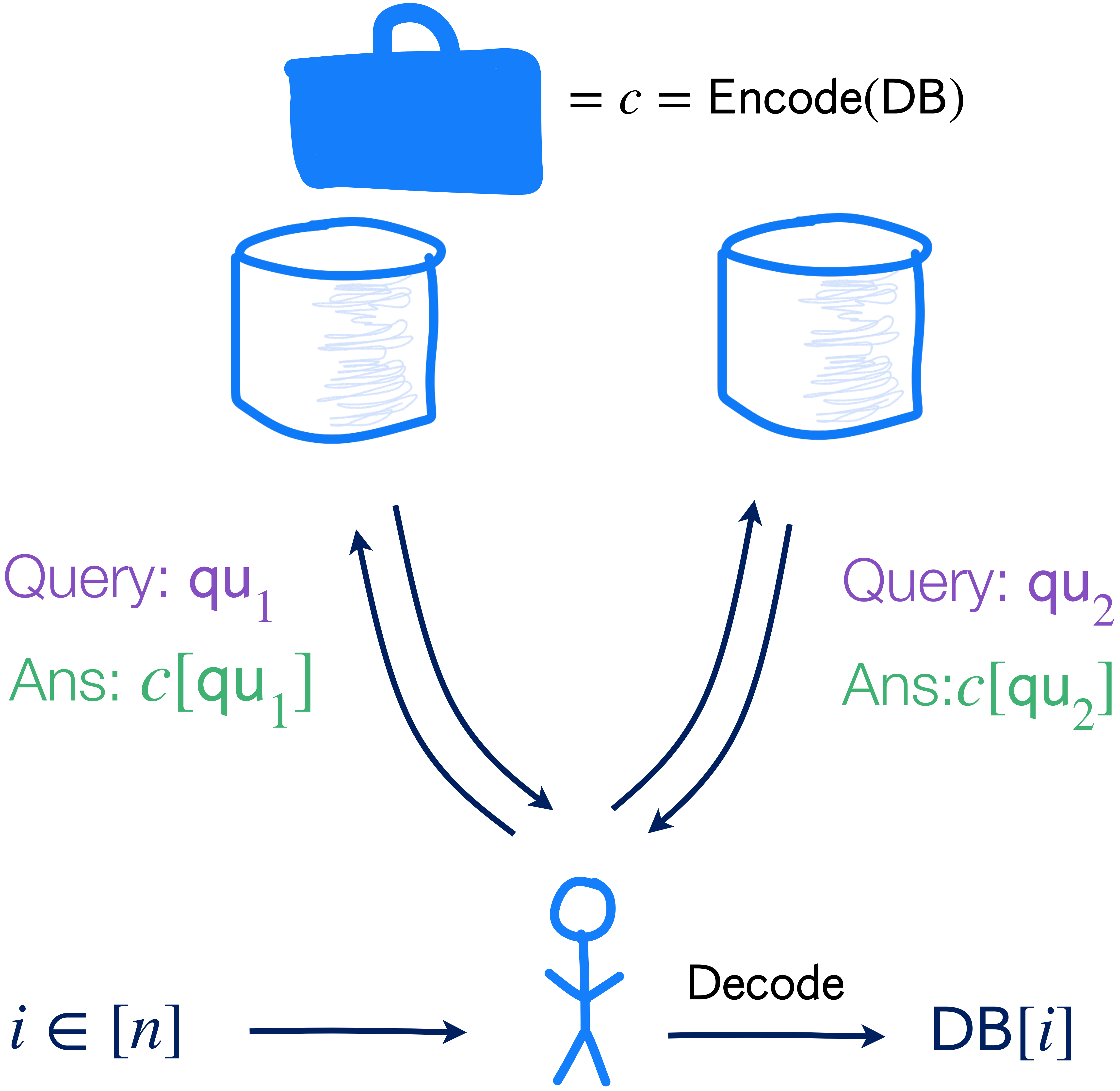
Ans:$c[\mathbf{qu}_2]$

$i \in [n]$



LDCs	PIR
Message	DB
Encoding	Preprocessing
Decoding	Reconstruction

2-query LDCs → 2-server PIR



$= c =$ Encode(DB)

Query: $\mathbf{qu}_1$

Ans: $c[\mathbf{qu}_1]$

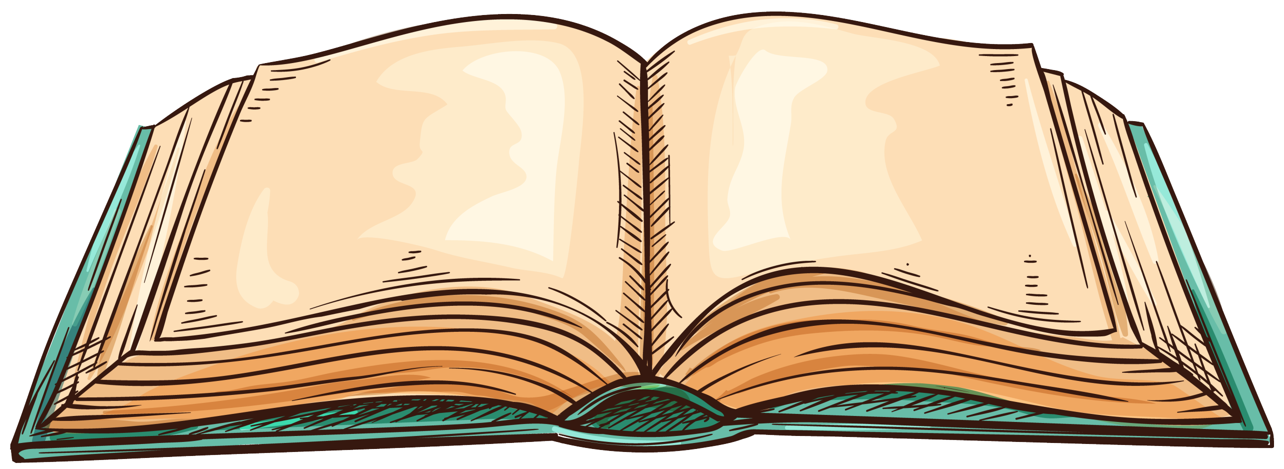
Query: $\mathbf{qu}_2$

Ans:$c[\mathbf{qu}_2]$

$i \in [n]$

Decode

DB$[i]$



LDCs	PIR
Message	DB
Encoding	Preprocessing
Decoding	Reconstruction

Casting BIM00 PIR as an LDC

$\mathbf{1}$	$f_{\mathsf{DB}}(\mathbf{1}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{1})$
$\mathbf{2}$	$f_{\mathsf{DB}}(\mathbf{2}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{2})$
$\mathbf{2^m}$	$f_{\mathsf{DB}}(\mathbf{2^m}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{2^m})$

Casting BIM00 PIR as an LDC

- Encode($\mathsf{DB} \in \{0,1\}^n$):
  - Interpolate $f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$ of total degree D
  - $\ell = 2^m, \Sigma = \{0,1\}^{\binom{m}{D/2}}$

$\mathbf{1}$	$f_{\mathsf{DB}}(\mathbf{1}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{1})$
$\mathbf{2}$	$f_{\mathsf{DB}}(\mathbf{2}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{2})$
$\mathbf{2^m}$	$f_{\mathsf{DB}}(\mathbf{2^m}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{2^m})$

Casting BIM00 PIR as an LDC

- Encode(DB $\in \{0,1\}^n$):
  - Interpolate $f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$ of total degree D
  - $\ell = 2^m, \Sigma = \{0,1\}^{\binom{m}{D/2}}$
- Query($i \in [n]$): calculate $\mathbf{p} = \mathsf{E}(i) \in \mathbb{F}_2^m$ and query $\mathbf{r}, \mathbf{r} + \mathbf{p}$

$\mathbf{1}$	$f_{\mathsf{DB}}(\mathbf{1}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{1})$
$\mathbf{2}$	$f_{\mathsf{DB}}(\mathbf{2}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{2})$
$\mathbf{2^m}$	$f_{\mathsf{DB}}(\mathbf{2^m}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{2^m})$

Casting BIM00 PIR as an LDC

- Encode($\mathsf{DB} \in \{0,1\}^n$):
  - Interpolate $f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$ of total degree D
  - $\ell = 2^m, \Sigma = \{0,1\}^{\binom{m}{D/2}}$
- Query($i \in [n]$): calculate $\mathbf{p} = \mathsf{E}(i) \in \mathbb{F}_2^m$ and query $\mathbf{r}, \mathbf{r} + \mathbf{p}$
- Decode: Hermite interpolation

$\mathbf{1}$	$f_{\mathsf{DB}}(\mathbf{1}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{1})$
$\mathbf{2}$	$f_{\mathsf{DB}}(\mathbf{2}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{2})$
$\mathbf{2^m}$	$f_{\mathsf{DB}}(\mathbf{2^m}), \ldots, \nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{2^m})$

Casting BIM00 PIR as an LDC

- Encode$(\mathsf{DB} \in \{0,1\}^n)$:
  - Interpolate $f_{\mathsf{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$ of total degree D
  - $\ell = 2^m, \Sigma = \{0,1\}^{\binom{m}{D/2}}$
- Query$(i \in [n])$: calculate $\mathbf{p} = \mathsf{E}(i) \in \mathbb{F}_2^m$ and query $\mathbf{r}, \mathbf{r} + \mathbf{p}$
- **Decode**: Hermite interpolation

$\mathbf{1}$	$f_{\mathsf{DB}}(\mathbf{1}),\ldots,\nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{1})$
$\mathbf{2}$	$f_{\mathsf{DB}}(\mathbf{2}),\ldots,\nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{2})$
$\mathbf{2^m}$	$f_{\mathsf{DB}}(\mathbf{2^m}),\ldots,\nabla^{\lfloor D/2 \rfloor} f_{\mathsf{DB}}(\mathbf{2^m})$

Q: Does our finite differences technique imply a new 2-query LDC?

Casting BIM00 PIR as an LDC

- Encode($\mathrm{DB} \in \{0,1\}^n$):
  - Interpolate $f_{\mathrm{DB}} : \mathbb{F}_2^m \to \mathbb{F}_2$ of total degree D
  - $\ell = 2^m, \Sigma = \{0,1\}^{\binom{m}{D/2}}$
- Query($i \in [n]$): calculate $\mathbf{p} = \mathrm{E}(i) \in \mathbb{F}_2^m$ and query $\mathbf{r}, \mathbf{r}+\mathbf{p}$
- Decode: Hermite interpolation

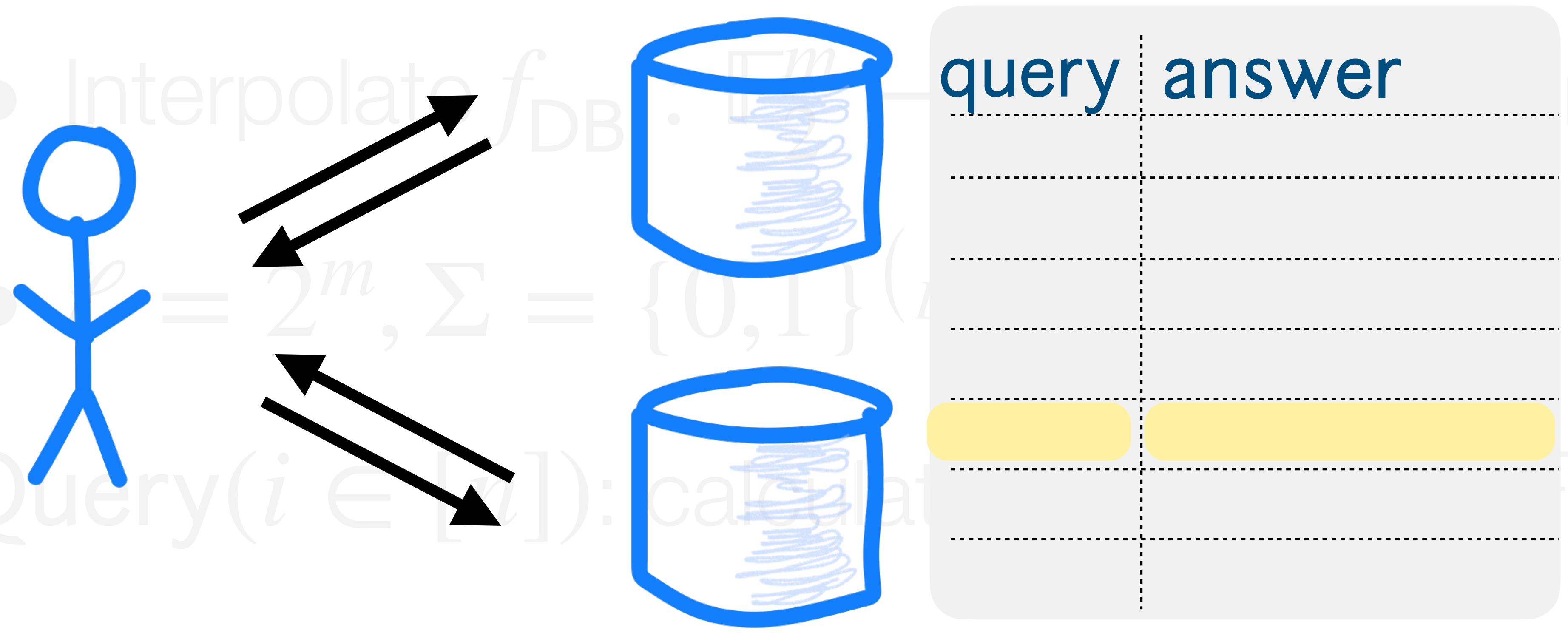
$\mathbf{1}$	$f_{\mathrm{DB}}(\mathbf{1}),\ldots,\nabla^{\lfloor D/2 \rfloor} f_{\mathrm{DB}}(\mathbf{1})$
$\mathbf{2}$	$f_{\mathrm{DB}}(\mathbf{2}),\ldots,\nabla^{\lfloor D/2 \rfloor} f_{\mathrm{DB}}(\mathbf{2})$
$\mathbf{2^m}$	$f_{\mathrm{DB}}(\mathbf{2^m}),\ldots,\nabla^{\lfloor D/2 \rfloor} f_{\mathrm{DB}}(\mathbf{2^m})$

Q: Does our finite differences technique imply a new 2-query LDC?
A: Actually no! LDCs are rigid: they require you to separately write out the answer for every query

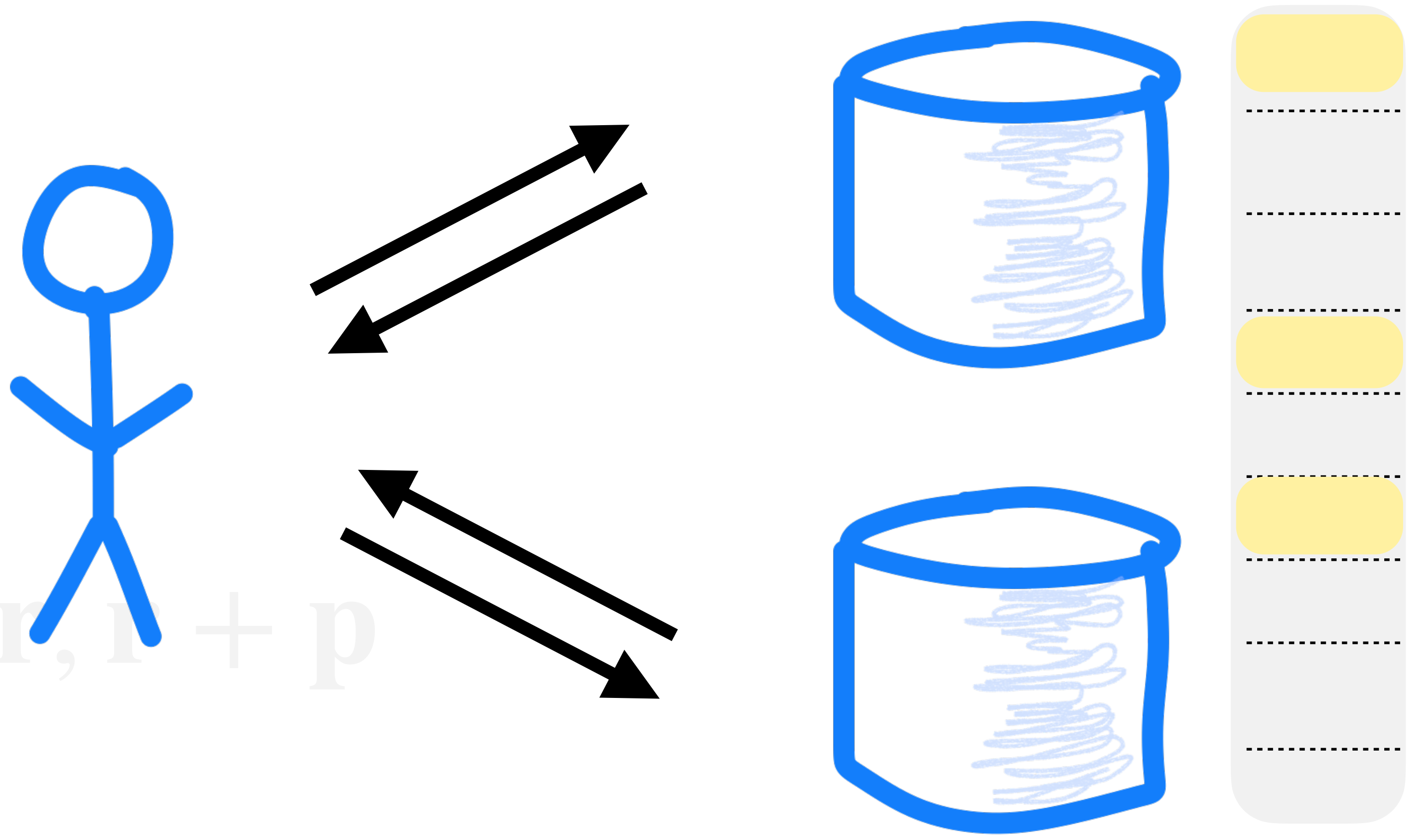
Casting BIM00 PIR as an LDC

- Encode(DB $\in \{0,1\}^n$):
  - Interpolate f_{DB} ... degree D
  - $\ell = 2^m, \Sigma = \{0,1\}^{(...)}$
- Query($i \in [n]$): calculat... $\mathbb{F}_2^m$ and query $\mathbf{r}, \mathbf{r}+\mathbf{p}$
- Decode: Hermite interpolation

LDCs and BIM00 PIR



Our PIR



Q: Does our finite differences technique imply a new 2-query LDC?
A: Actually no! LDCs are rigid: they require you to separately write out the answer for every query

Defining Batch-Smooth LDCs

Cheatsheet

n: message length

ℓ: codeword length

Σ: alphabet

q: locality

b: number of batches

Defining Batch-Smooth LDCs

- Three functions:
  - Encode(msg $\in \{0,1\}^n$) → $c \in \Sigma^\ell$
  - Randomised Query($i \in [n]$) → qu $\in [\ell]^q$
  - Decode($i \in [n]$, qu $\in [\ell]^q$, c[qu]) → msg[i]
- **Smoothness:** marginal distribution of qu$_j$ is uniform over $[\ell]$ for all j

Cheatsheet

n: message length

ℓ: codeword length

Σ: alphabet

q: locality

b: number of batches

Defining Batch-Smooth LDCs

- Three functions:
  - $\mathsf{Encode}(\mathsf{msg} \in \{0,1\}^n) \to c \in \Sigma^\ell$
  - Randomised $\mathsf{Query}(i \in [n]) \to \mathsf{qu} \in [\ell]^q$
  - $\mathsf{Decode}(i \in [n], \mathsf{qu} \in [\ell]^q, c[\mathsf{qu}]) \to \mathsf{msg}[i]$
- **Smoothness:** marginal distribution of qu_j is uniform over $[\ell]$ for all j
- **Batch-smoothness:** reorganise the queries into b batches:

1	$\mathsf{qu}_1, \ldots, \mathsf{qu}_{q/b}$
2	$\mathsf{qu}_{q/b+1}, \ldots, \mathsf{qu}_{2q/b}$
b	$\mathsf{qu}_{(b-1)q/b+1}, \ldots, \mathsf{qu}_q$

Cheatsheet

n: message length

ℓ: codeword length

Σ: alphabet

q: locality

b: number of batches

Defining Batch-Smooth LDCs

- Three functions:
  - Encode(msg $\in \{0,1\}^n$) $\rightarrow c \in \Sigma^\ell$
  - Randomised Query($i \in [n]$) $\rightarrow$ qu $\in [\ell]^q$
  - Decode($i \in [n]$, qu $\in [\ell]^q$, c[qu]) $\rightarrow$ msg[i]
- **Smoothness:** marginal distribution of qu$_j$ is uniform over $[\ell]$ for all j
- **Batch-smoothness:** reorganise the queries into b batches:

1	$\mathsf{qu}_1, \ldots, \mathsf{qu}_{q/b}$
2	$\mathsf{qu}_{q/b+1}, \ldots, \mathsf{qu}_{2q/b}$
b	$\mathsf{qu}_{(b-1)q/b+1}, \ldots, \mathsf{qu}_q$

Then each row's distribution should be independent of the index i being queried

Cheatsheet

n: message length

ℓ: codeword length

Σ: alphabet

q: locality

b: number of batches

BIM00 PIR as a Batch-Smooth LDC

Cheatsheet

n: message length

$\binom{m}{D} \geq n$

BIM00 PIR as a Batch-Smooth LDC

- Number of batches: $b = 2$

Cheatsheet

n: message length

$\binom{m}{D} \geq n$

BIM00 PIR as a Batch-Smooth LDC

- Number of batches: $b = 2$
- Alphabet: $\Sigma = \{0,1\}$

Cheatsheet

n: message length

$\binom{m}{D} \geq n$

BIM00 PIR as a Batch-Smooth LDC

Cheatsheet

n: message length

$\binom{m}{D} \geq n$

- Number of batches: $b = 2$
- Alphabet: $\Sigma = \{0,1\}$
- Codeword length: $\ell = 2^m \cdot \binom{m}{D/2}$

BIM00 PIR as a Batch-Smooth LDC

Cheatsheet

n: message length

$\binom{m}{D} \geq n$

- Number of batches: $b = 2$
- Alphabet: $\Sigma = \{0,1\}$
- Codeword length: $\ell = 2^m \cdot \binom{m}{D/2}$
- Number of queries: $q = 2\binom{m}{D/2}$

Our PIR as a Batch-Smooth LDC

Cheatsheet

n: message length

$\binom{m}{D} \geq n$

- Number of batches: $b = 2$
- Alphabet: $\Sigma = \{0,1\}$
- Codeword length: $\ell = 2^m \cdot \cancel{\binom{m}{D/2}} \xrightarrow{\text{Finite differences}} 2^m$
- Number of queries: $q = 2\binom{m}{D/2}$

Our PIR as a Batch-Smooth LDC

Cheatsheet

n: message length

$\binom{m}{D} \geq n$

- Number of batches: $b = 2$
- Alphabet: $\Sigma = \{0,1\}$
- Codeword length: $\ell = 2^m \cdot \cancel{\binom{m}{D/2}} \xrightarrow{\text{Finite differences}} 2^m$
- Number of queries: $q = 2\binom{m}{D/2}$

Consequence: the first batch-smooth LDC with constant alphabet size, constant number of batches b, codeword length $n^{1+o(1)}$, and polynomially sublinear number of queries $q = n^{1-\Omega(1)}$