Fuzzy Private Set Intersection

with Large Hyperballs

Aron van Baarsen

CWI & Leiden University

Sihang Pu†

CNRS-IRIF

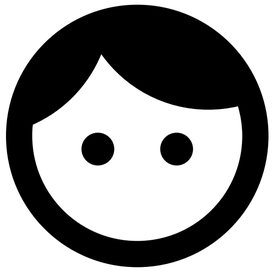
†*Work was done at CISPA Helmholtz Center.*

Private Set Intersection (PSI)

Traditional Definition

Alice

Bob



Private Set Intersection (PSI)

Traditional Definition

Alice

N items

{ Hello, Zürich }

Bob

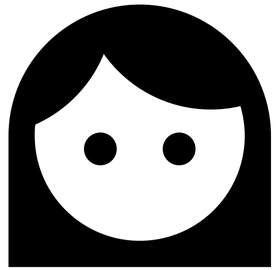
M items

{ Hello, World, ! }

Private Set Intersection (PSI)

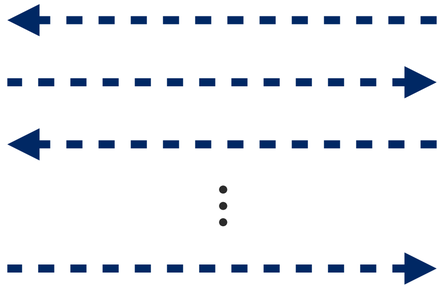
Traditional Definition

Alice

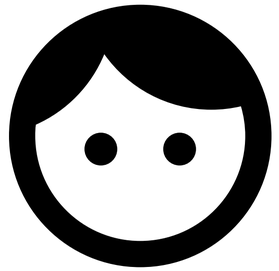


N items

{ Hello, Zürich }



Bob

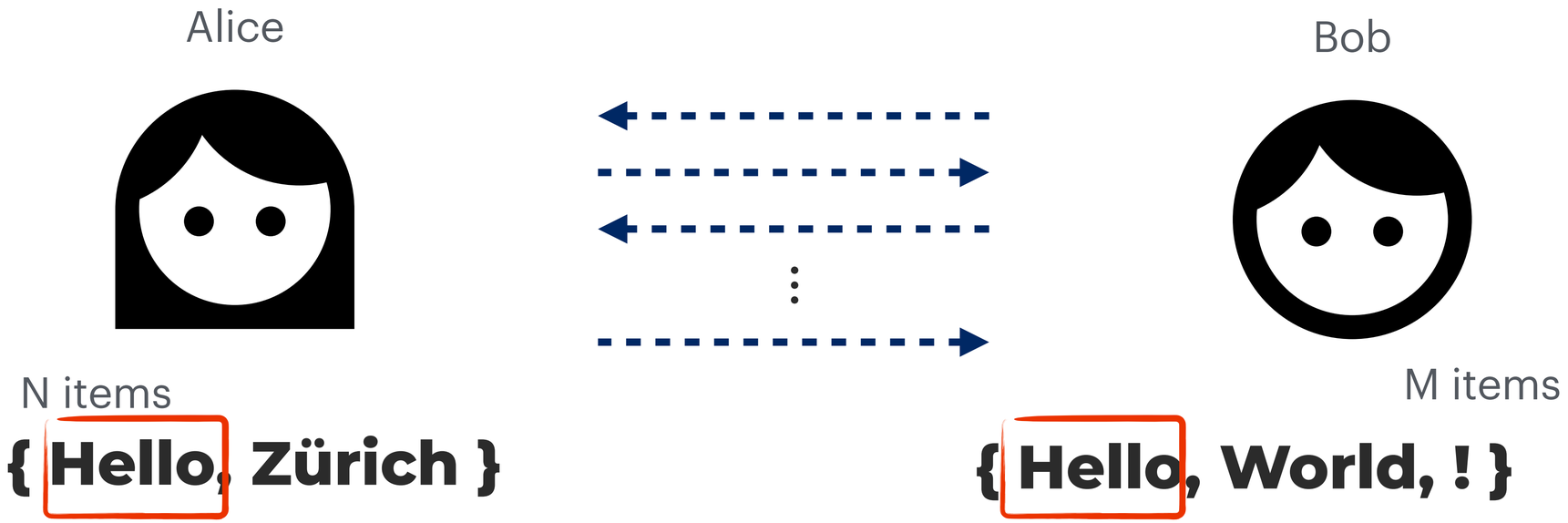


M items

{ Hello, World, ! }

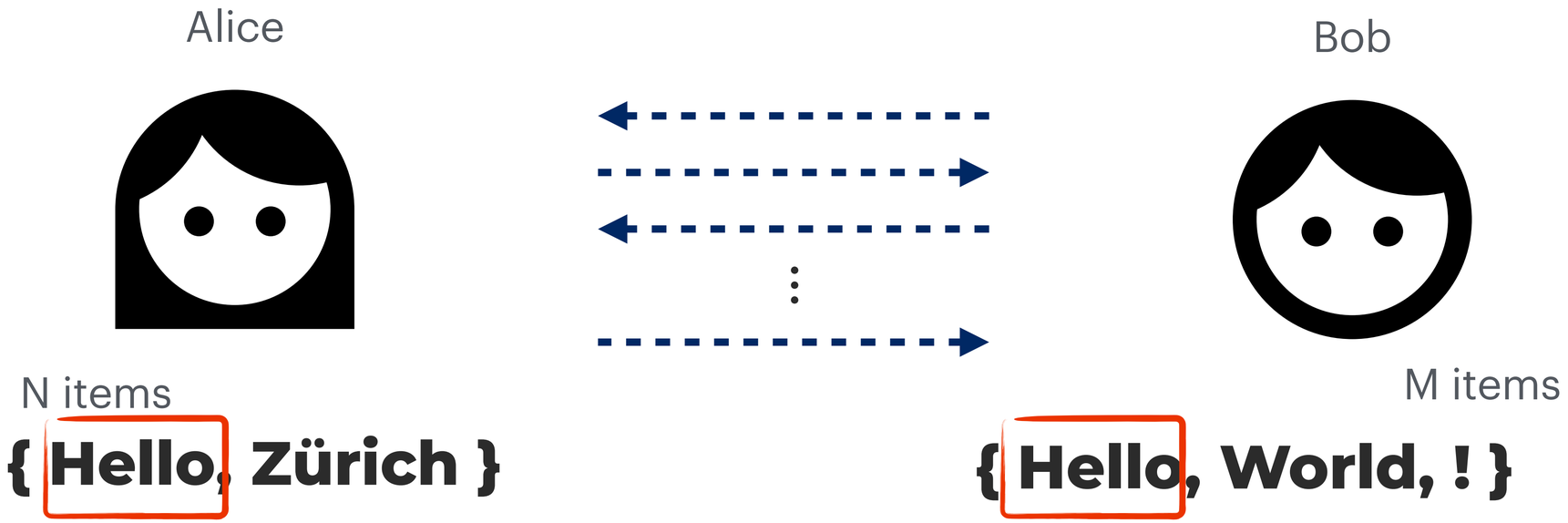
Private Set Intersection (PSI)

Traditional Definition



Private Set Intersection (PSI)

Traditional Definition



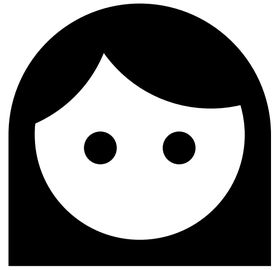
A special case of 2-PC:

- Constant rounds
- Extremely efficient, O(N+M) comm. and comp.

Private Set Intersection (PSI)

Traditional Definition

Alice (**Receiver**)



N items

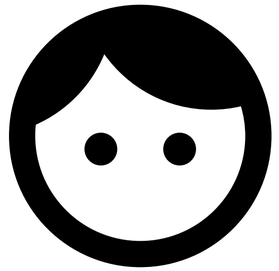
{ Hello, Zürich }

Learns the intersection

Two Rounds

Semi-honest Security

Bob (**Sender**)



M items

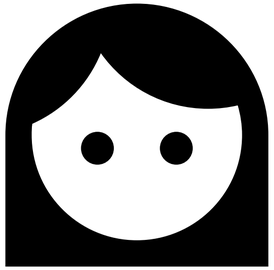
{ Hello, World, ! }

Nothing learnt except N

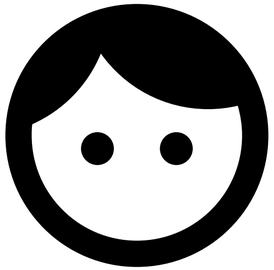
Private Set Intersection (PSI)

What if some letters are mistyped?

Alice



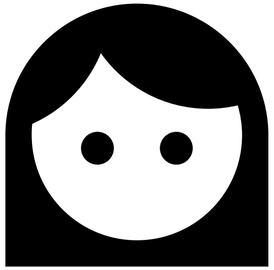
Bob



Private Set Intersection (PSI)

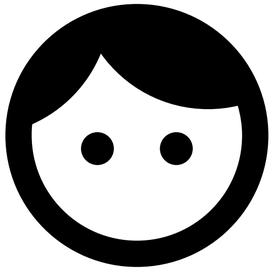
What if some letters are mistyped?

Alice



{ Hello, Zürich }

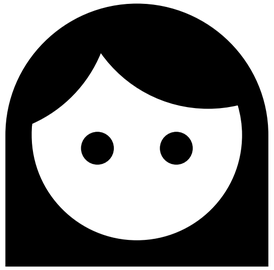
Bob



Private Set Intersection (PSI)

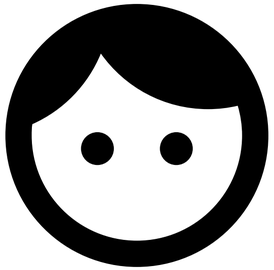
What if some letters are mistyped?

Alice



{ Hello, Zürich }

Bob

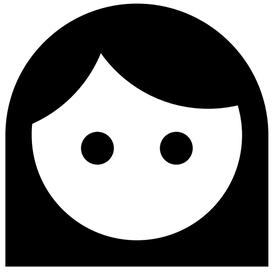


{ Helli, World }

Private Set Intersection (PSI)

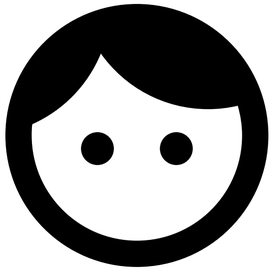
What if some letters are mistyped?

Alice

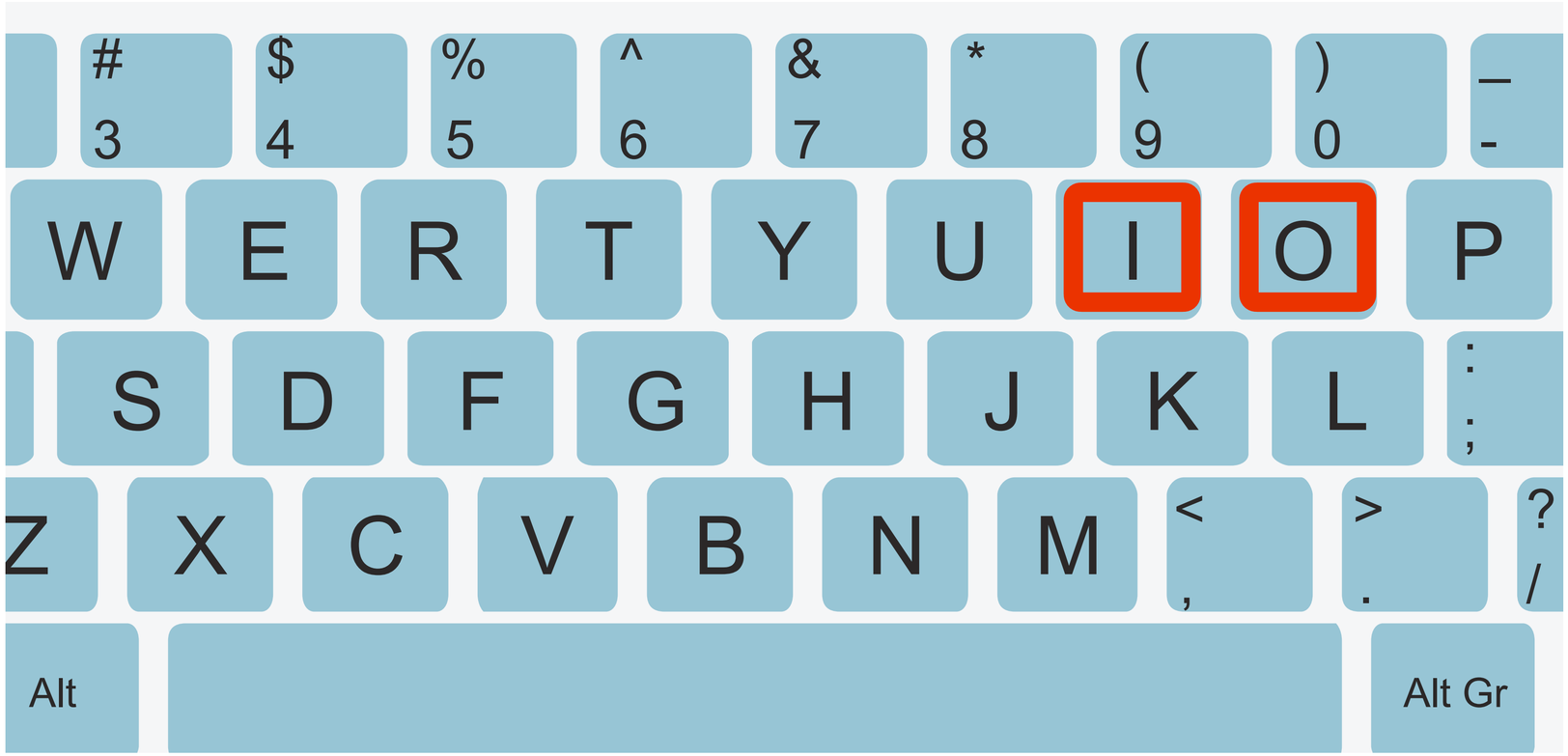


{ Hello, Zürich }

Bob



{ Helli, World }



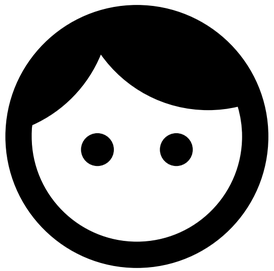
Private Set Intersection (PSI)

What if some letters are mistyped?

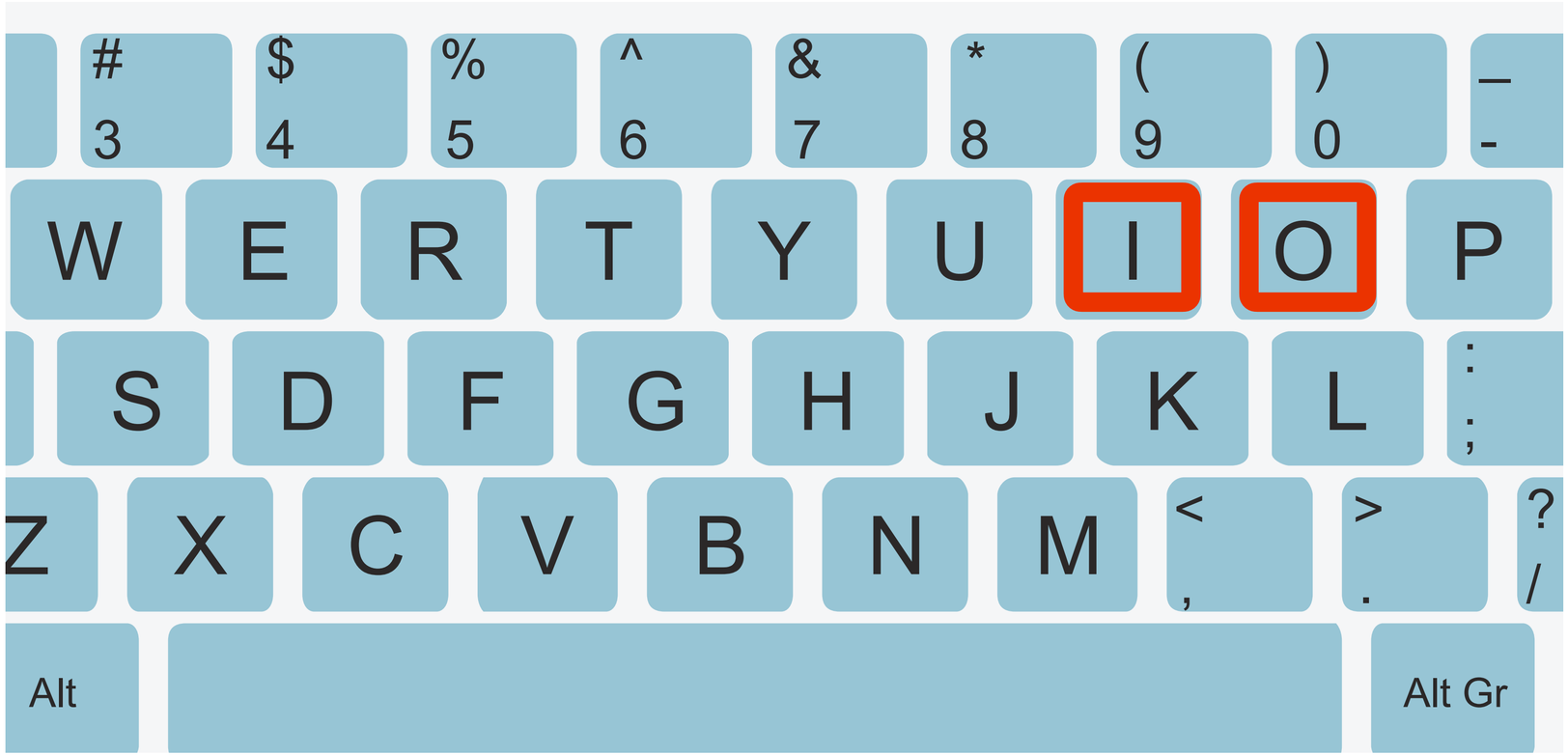
Alice

{ Hello, Zürich }

Bob



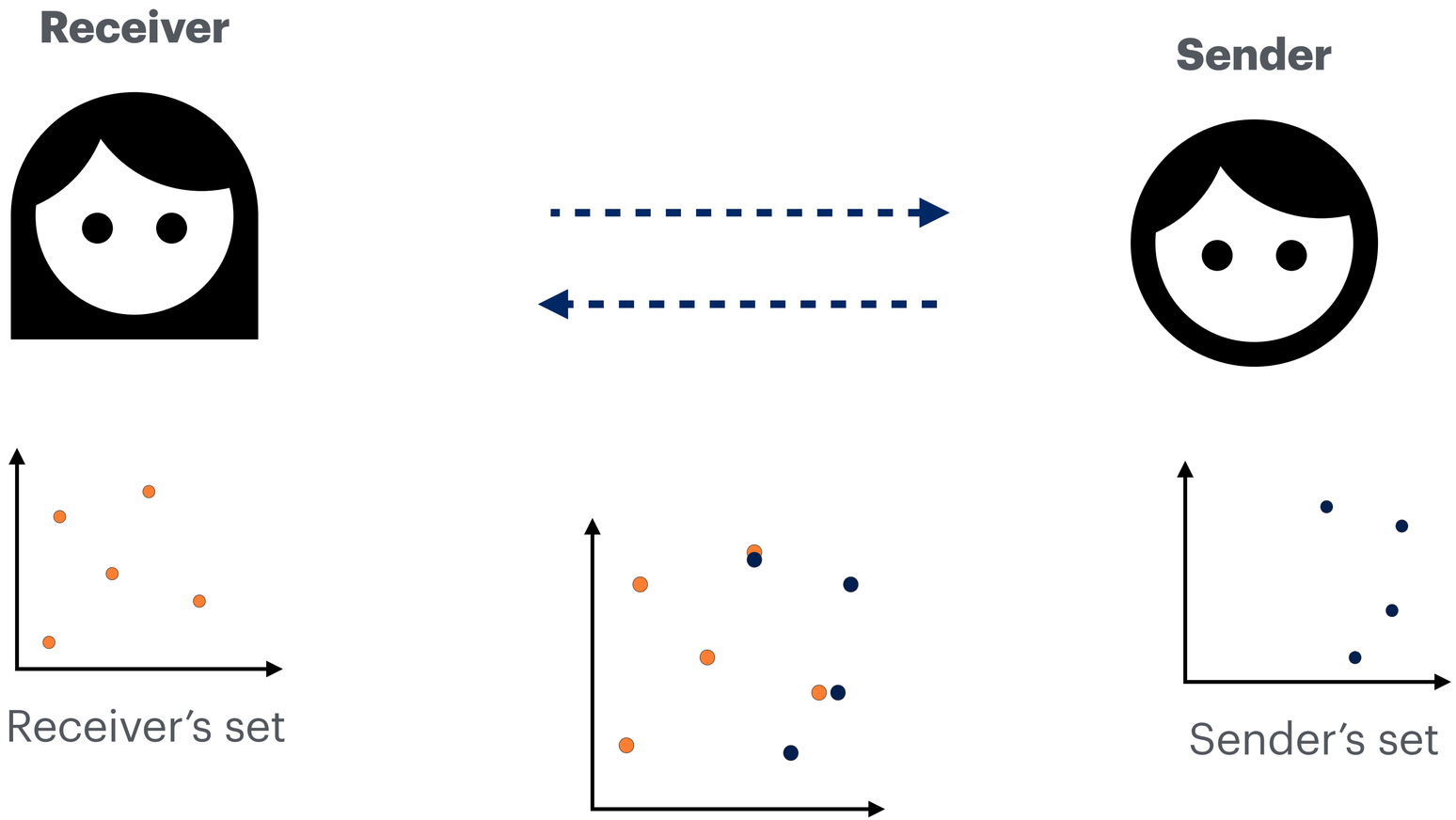
{ Helli, World }



Naive approach: enumerate all nearby points, then use the traditional PSI

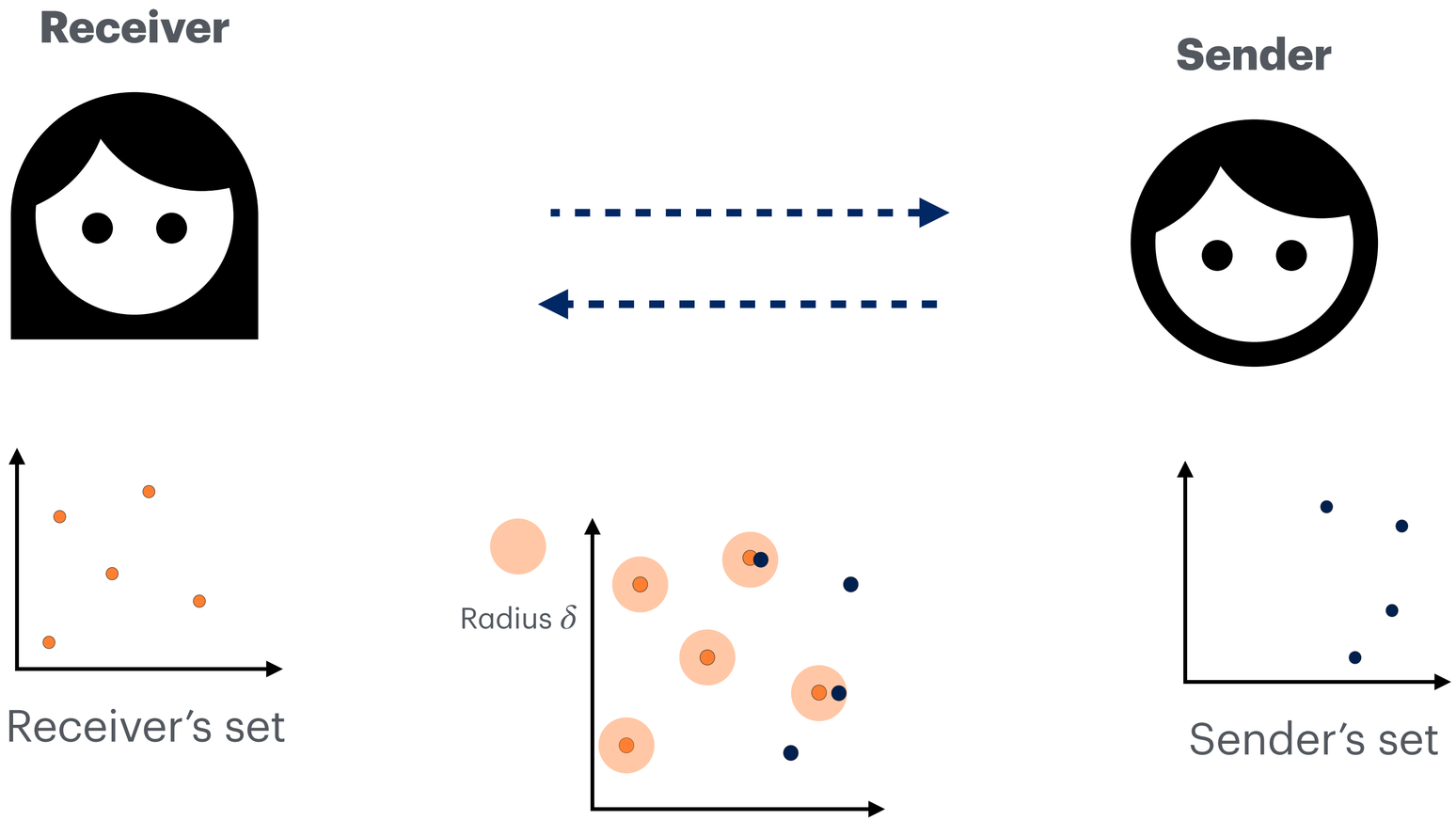
Fuzzy PSI (fPSI)

Definitions



Fuzzy PSI (fPSI)

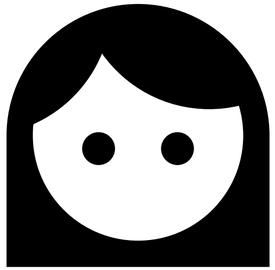
Definitions

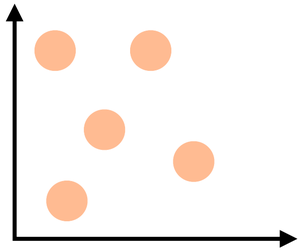


Fuzzy PSI (fPSI)

Definitions

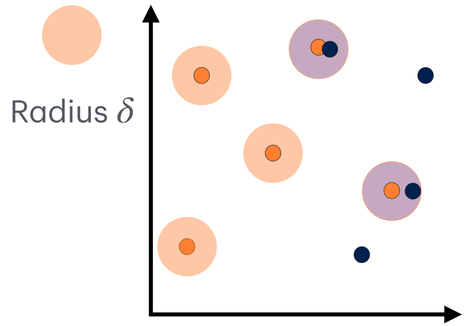
Receiver



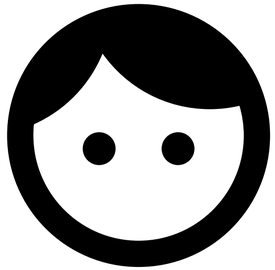


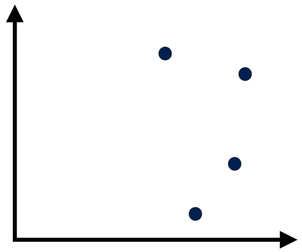
Learns ● , if and only if

- ● intersects ●



Sender

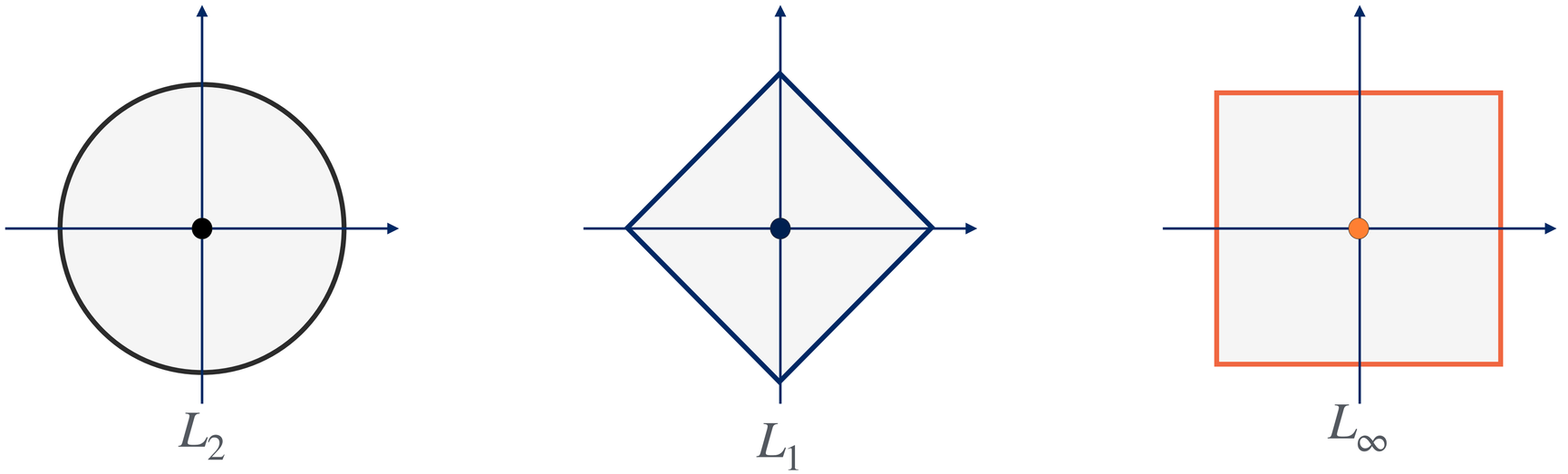




Sender's set

Fuzzy PSI (fPSI)

Distance Functions



We only consider **exact** distance functions, e.g., without using metric embeddings

Prior Approaches

Prior Approaches

- Either reduce to **traditional** PSI, or use **generic** approaches [IW06]

Prior Approaches

- Either reduce to **traditional** PSI, or use **generic** approaches [IW06]
  - **Exponential** blowup, only suitable for **tiny** balls

Prior Approaches

- Either reduce to **traditional** PSI, or use **generic** approaches [IW06]
  - **Exponential** blowup, only suitable for **tiny** balls
  - **Heavy** machinery, only suitable for a **small number** of points

Prior Approaches

- Either reduce to **traditional** PSI, or use **generic** approaches [IW06]
  - **Exponential** blowup, only suitable for **tiny** balls
  - **Heavy** machinery, only suitable for a **small number** of points
- FSS + OT [GRS22], [GRS23]

Prior Approaches

- Either reduce to **traditional** PSI, or use **generic** approaches [IW06]
  - **Exponential** blowup, only suitable for **tiny** balls
  - **Heavy** machinery, only suitable for a **small number** of points
- FSS + OT [GRS22], [GRS23]
  - Communication: **sublinear** to the volume

Prior Approaches

- Either reduce to **traditional** PSI, or use **generic** approaches [IW06]
  - **Exponential** blowup, only suitable for **tiny** balls
  - **Heavy** machinery, only suitable for a **small number** of points
- FSS + OT [GRS22], [GRS23]
  - Communication: **sublinear** to the volume
  - Computation: basically **linear** to the volume

Prior Approaches

- Either reduce to **traditional** PSI, or use **generic** approaches [IW06]
  - **Exponential** blowup, only suitable for **tiny** balls
  - **Heavy** machinery, only suitable for a **small number** of points
- FSS + OT [GRS22], [GRS23]
  - Communication: **sublinear** to the volume
  - Computation: basically **linear** to the volume
  - Three-round

Prior Approaches

- Either reduce to **traditional** PSI, or use **generic** approaches [IW06]
  - **Exponential** blowup, only suitable for **tiny** balls
  - **Heavy** machinery, only suitable for a **small number** of points
- FSS + OT [GRS22], [GRS23]
  - Communication: **sublinear** to the volume
  - Computation: basically **linear** to the volume
  - Three-round
  - Support L_∞ and L_1 distance (L_1 is way more expensive)

Prior Approaches

- Either reduce to **traditional** PSI, or use **generic** approaches [IW06]
  - **Exponential** blowup, only suitable for **tiny** balls
  - **Heavy** machinery, only suitable for a **small number** of points
- FSS + OT [GRS22], [GRS23]
  - Communication: **sublinear** to the volume
  - Computation: basically **linear** to the volume
  - Three-round
  - Support L_∞ and L_1 distance (L_1 is way more expensive)
  - The receiver learns **precise input** of the sender

Prior Approaches

- Either reduce to **traditional** PSI, or use **generic** approaches [IW06]
  - **Exponential** blowup, only suitable for **tiny** balls
  - **Heavy** machinery, only suitable for a **small number** of points
- FSS + OT [GRS22], [GRS23]
  - Communication: **sublinear** to the volume
  - Computation: basically **linear** to the volume
  - Three-round
  - Support L_∞ and L_1 distance (L_1 is way more expensive)
  - The receiver learns **precise input** of the sender
    - Not a problem for traditional PSI

Prior Approaches

- Either reduce to **traditional** PSI, or use **generic** approaches [IW06]
  - **Exponential** blowup, only suitable for **tiny** balls
  - **Heavy** machinery, only suitable for a **small number** of points
- FSS + OT [GRS22], [GRS23]
  - Communication: **sublinear** to the volume
  - Computation: basically **linear** to the volume
  - Three-round
  - Support L_∞ and L_1 distance (L_1 is way more expensive)
  - The receiver learns **precise input** of the sender
    - Not a problem for traditional PSI

Our Goals

With **negligible** correctness errors,

- Can we support *large volume* ?
- Can we support *generalized distances*?
- Can we support *different functionalities*?

Our Results

Our Results

- **Negligible** correctness error, i.e. no false positives or false negatives

Our Results

- **Negligible** correctness error, i.e. no false positives or false negatives
- **Two-round** protocols
  - Specific applications, e.g., illegal content detection [BGJP23]

Our Results

- **Negligible** correctness error, i.e. no false positives or false negatives
- **Two-round** protocols
  - Specific applications, e.g., illegal content detection [BGJP23]
- **Sublinear** communication and computation
  - Low-dim: $O(2^d)$, scales **sub-linearly** to the volume, e.g., $O((2\delta)^d)$
  - High-dim: Get rid of the **exponential dependence** on dimensions

Our Results

- **Negligible** correctness error, i.e. no false positives or false negatives
- **Two-round** protocols
  - Specific applications, e.g., illegal content detection [BGJP23]
- **Sublinear** communication and computation
  - Low-dim: $O(2^d)$, scales **sub-linearly** to the volume, e.g., $O((2\delta)^d)$
  - High-dim: Get rid of the **exponential dependence** on dimensions
- **Generalized** distance metrics
  - L_p for any $p \in [1, \infty]$

Our Results

- **Negligible** correctness error, i.e. no false positives or false negatives
- **Two-round** protocols
  - Specific applications, e.g., illegal content detection [BGJP23]
- **Sublinear** communication and computation
  - Low-dim: $O(2^d)$, scales **sub-linearly** to the volume, e.g., $O((2\delta)^d)$
  - High-dim: Get rid of the **exponential dependence** on dimensions
- **Generalized** distance metrics
  - L_p for any $p \in [1,\infty]$
- **Generalized** functionalities
  - Various leakages allowed: sender's points, receiver's points, labels, cardinality…
  - Circuit fuzzy PSI

Technical Part

Recap: [GRS22]

Why it fails on sublinear computations?

Recap: [GRS22]

Why it fails on sublinear computations?

- Function Secret Sharing (FSS, [BGI15])
  - **Share**$(F) \to (F_0, F_1)$
  - **Eval**$(F_0, q) \oplus$ **Eval**$(F_1, q) = F(q)$

Recap: [GRS22]

Why it fails on sublinear computations?

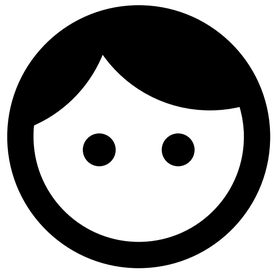
- Function Secret Sharing (FSS, [BGI15])
  - **Share**$(F) \to (F_0, F_1)$
  - **Eval**$(F_0, q) \oplus$ **Eval**$(F_1, q) = F(q)$
- Both parties secretly share a **membership test** function $F_{\mathbf{W}}(\cdot)$
  - $\mathbf{W}$ is the **point set** containing every point inside the **Receiver's balls**
  - If $\mathbf{q} \in \mathbf{W} \implies F_{\mathbf{W}}(\mathbf{q}) = 0$, i.e., **Eval**$(F_0, q) =$ **Eval**(F_1, q)
  - If $\mathbf{q} \notin \mathbf{W} \implies F_{\mathbf{W}}(\mathbf{q}) \neq 0$, i.e., **Eval**$(F_0, q) \neq$ **Eval**(F_1, q)

Recap: [GRS22]

Why it fails on sublinear computations?

Receiver

Sender



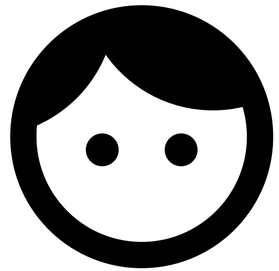
Recap: [GRS22]

Why it fails on sublinear computations?

Share $(F_{\mathbf{W}}) \rightarrow ($ 0 , 1 $)$

Receiver

Sender



Recap: [GRS22]

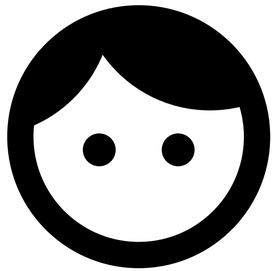
Why it fails on sublinear computations?

Share $(F_W) \rightarrow ($ 0 , 1 $)$

Receiver

Oblivious Transfer

Sender



Recap: [GRS22]

Why it fails on sublinear computations?

Share $(F_{\mathbf{W}}) \rightarrow ($ 0 , 1 $)$

Recap: [GRS22]

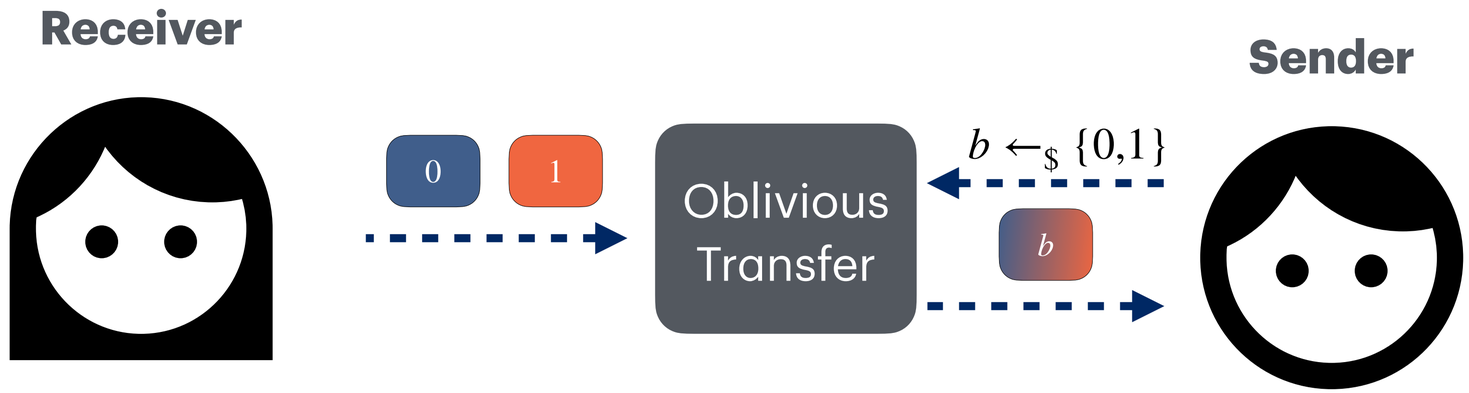
Why it fails on sublinear computations?

Share $(F_{\mathbf{W}}) \rightarrow ($ 0 , 1 $)$

Recap: [GRS22]

Why it fails on sublinear computations?

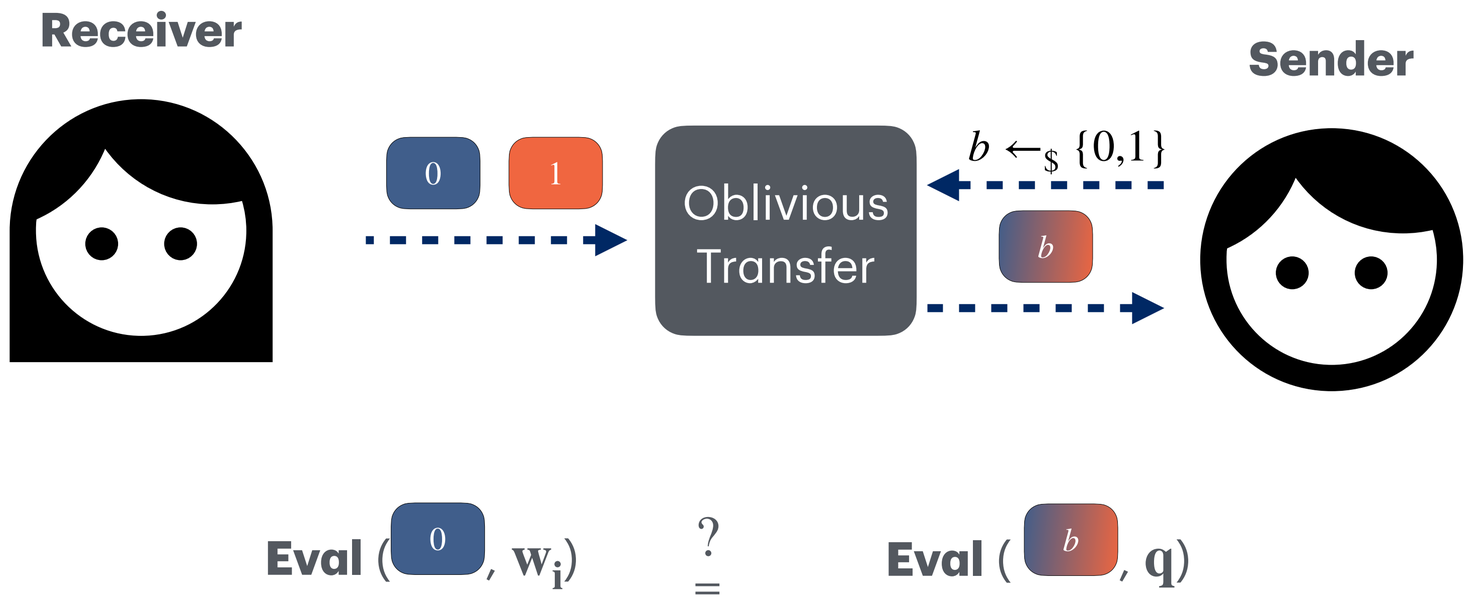
Share $(F_{\mathbf{W}}) \rightarrow ($ 0 , 1 $)$



Recap: [GRS22]

Why it fails on sublinear computations?

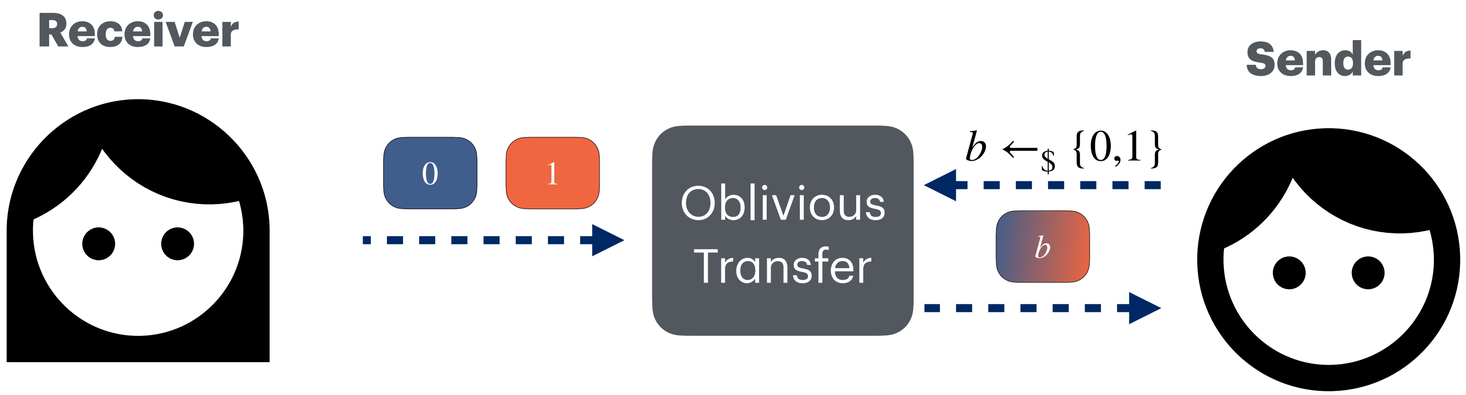
Share ($F_{\mathbf{W}}$) → (0 , 1)



Recap: [GRS22]

Why it fails on sublinear computations?

Share ($F_{\mathbf{W}}$) → (0 , 1)



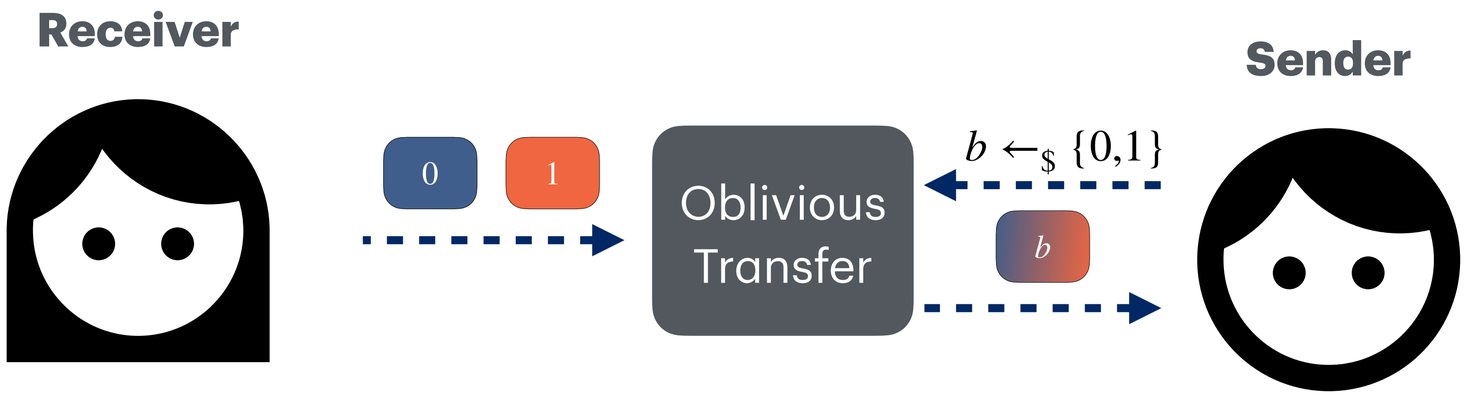
Eval (0 , $\mathbf{w_i}$) $\overset{?}{=}$ **Eval** (b , $\mathbf{q}$)

If $\mathbf{w_i} = \mathbf{q}$, the equation holds no matter what b is.

Recap: [GRS22]

Why it fails on sublinear computations?

Share ($F_{\mathbf{W}}$) → (0 , 1)



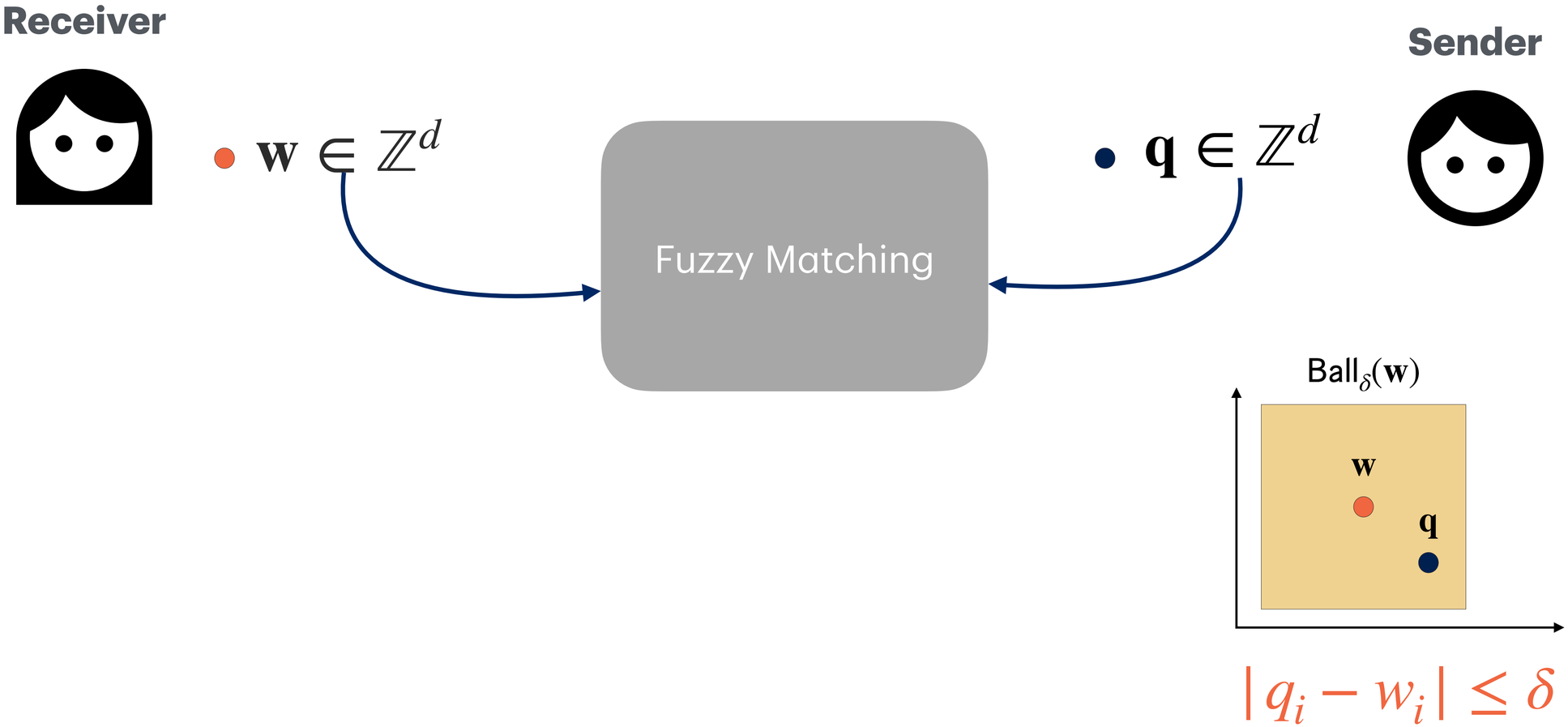
The receiver has to ***enumerate each point*** contained in the ball $\mathbf{w_i} \in \mathbf{W}$

Eval (0 , $\mathbf{w_i}$) $\overset{?}{=}$ **Eval** (b , $\mathbf{q}$)

If $\mathbf{w_i} = \mathbf{q}$, the equation holds no matter what b is.

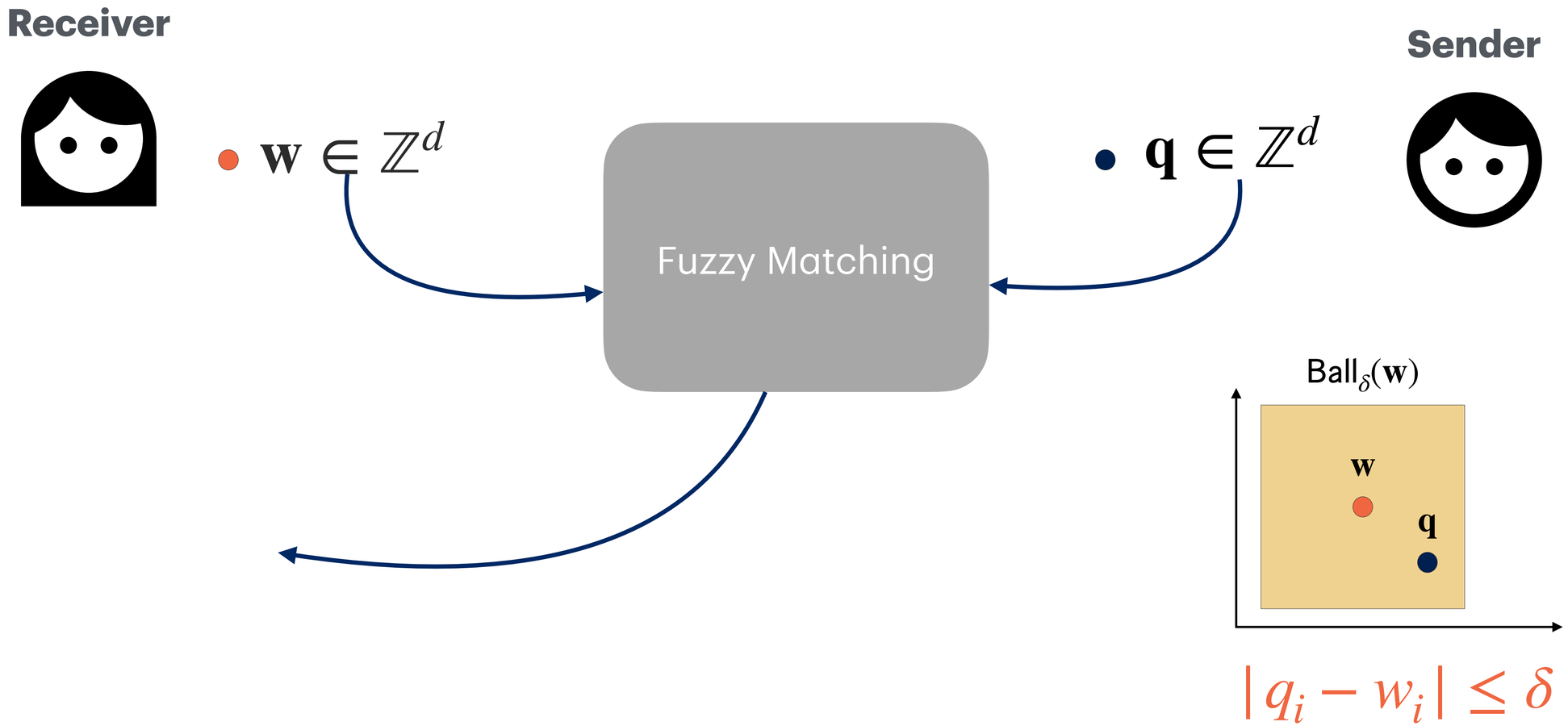
Core Block: Fuzzy Matching

Abstract the functionality mentioned above



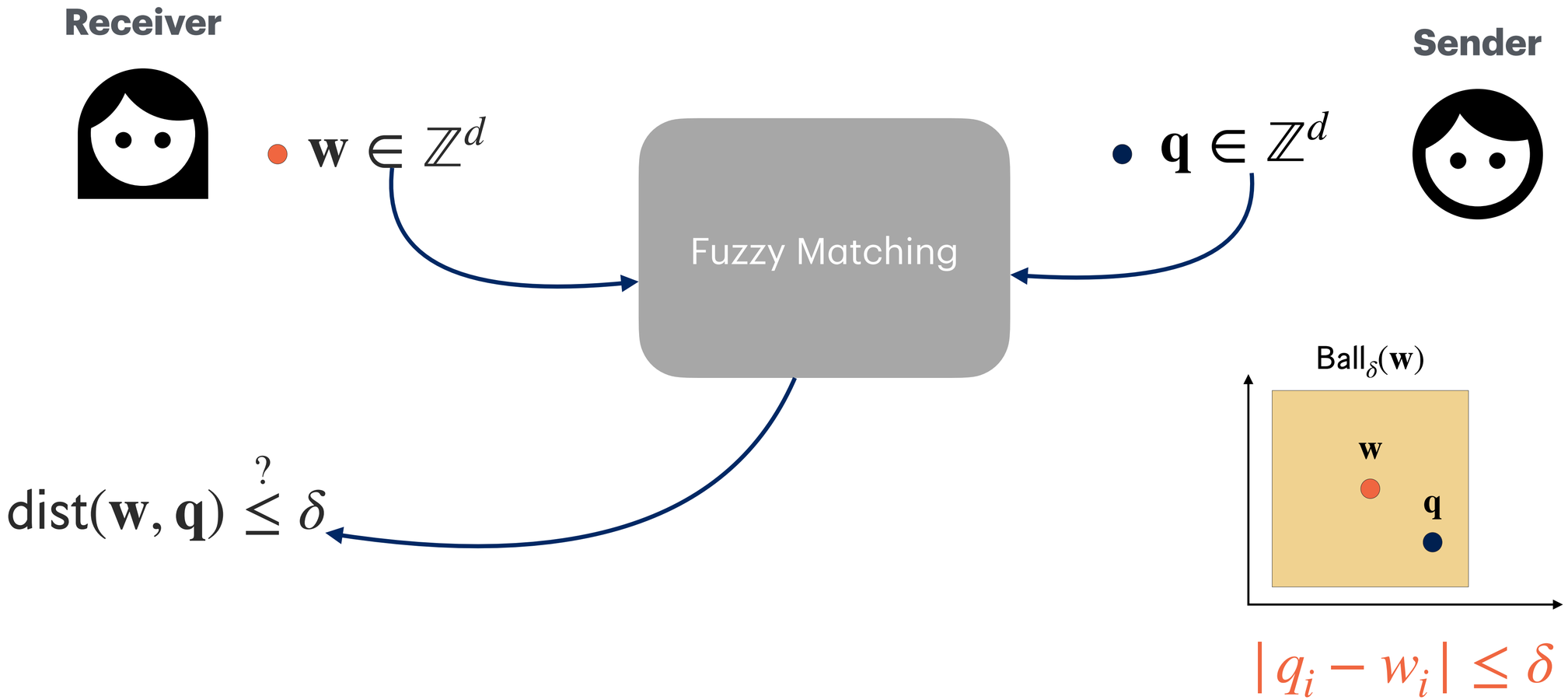
Core Block: Fuzzy Matching

Abstract the functionality mentioned above



Core Block: Fuzzy Matching

Abstract the functionality mentioned above

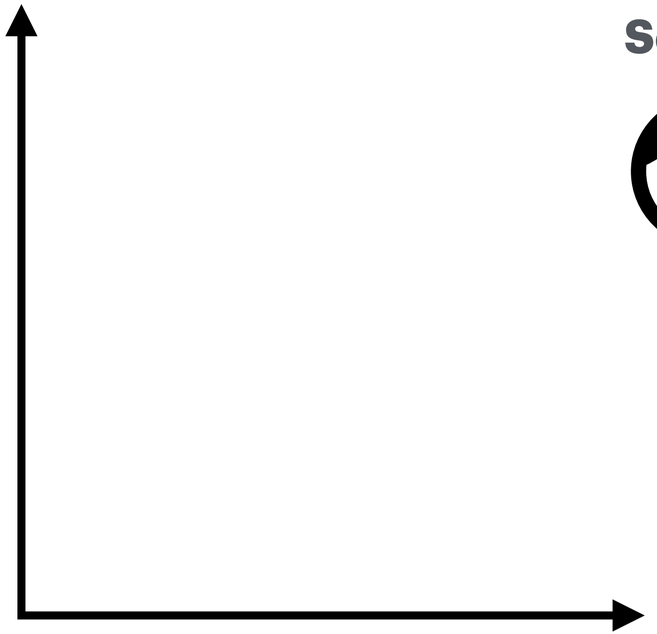


Fuzzy Matching for L_∞

High-level ideas to get sublinear computations

Receiver

sk  $\mathbf{w} \in \mathbb{Z}^2$



Sender

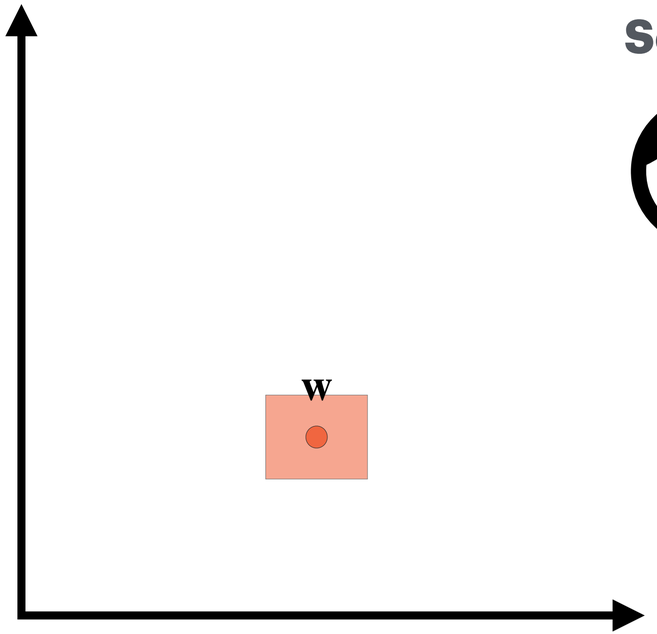
 $\mathbf{q} \in \mathbb{Z}^2$

Fuzzy Matching for L_∞

High-level ideas to get sublinear computations

Receiver

sk  $\mathbf{w} \in \mathbb{Z}^2$



Sender

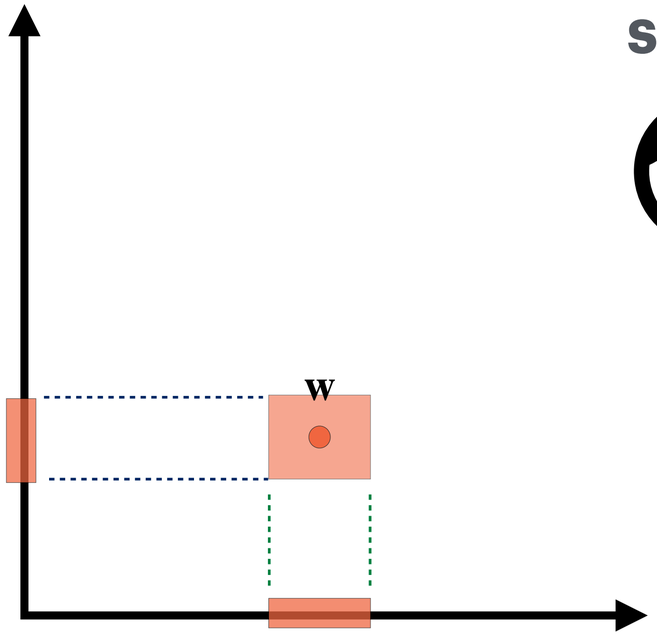
 $\mathbf{q} \in \mathbb{Z}^2$

Fuzzy Matching for L_∞

High-level ideas to get sublinear computations

Receiver

sk  $\mathbf{w} \in \mathbb{Z}^2$



Sender

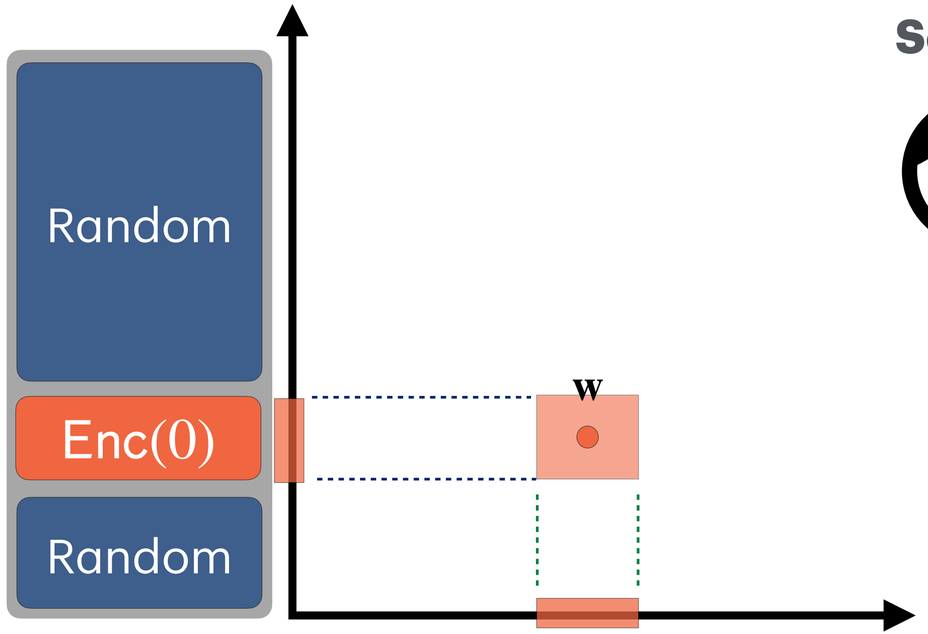
 $\mathbf{q} \in \mathbb{Z}^2$

Fuzzy Matching for L_∞

High-level ideas to get sublinear computations

Receiver

$\mathbf{w} \in \mathbb{Z}^2$



Sender

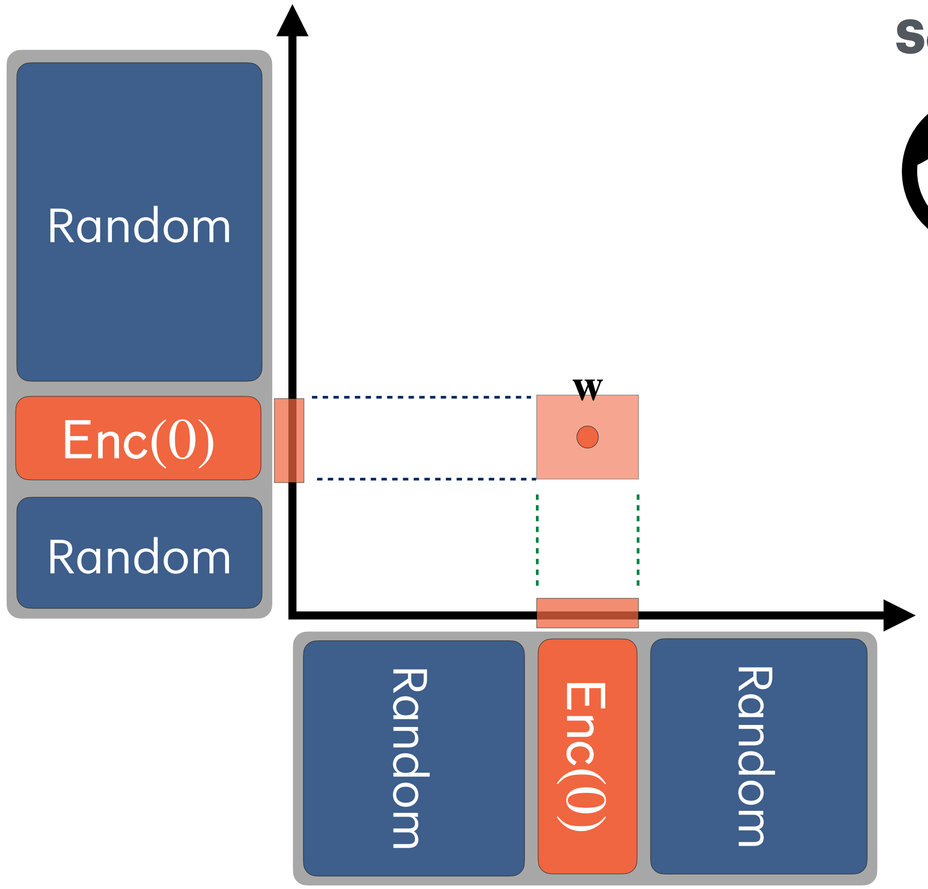
$\mathbf{q} \in \mathbb{Z}^2$

Fuzzy Matching for L_∞

High-level ideas to get sublinear computations

Receiver

sk  $\mathbf{w} \in \mathbb{Z}^2$



Sender

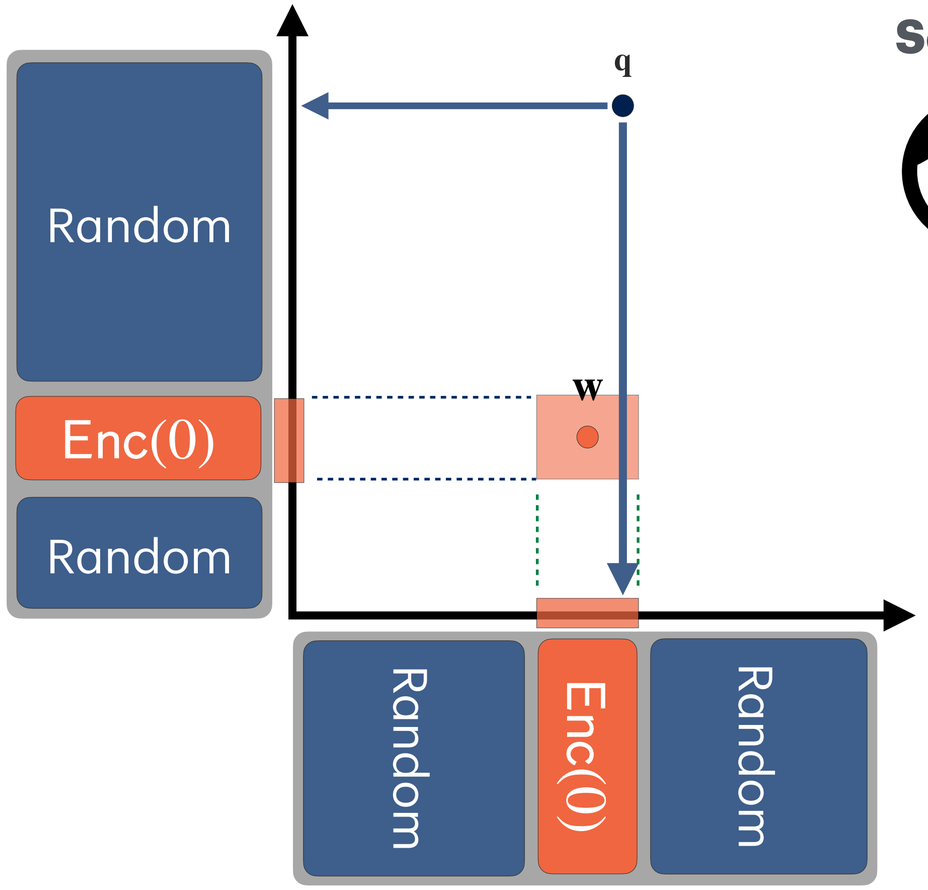
 $\mathbf{q} \in \mathbb{Z}^2$

Fuzzy Matching for L_∞

High-level ideas to get sublinear computations

Receiver

$\mathbf{w} \in \mathbb{Z}^2$



Sender

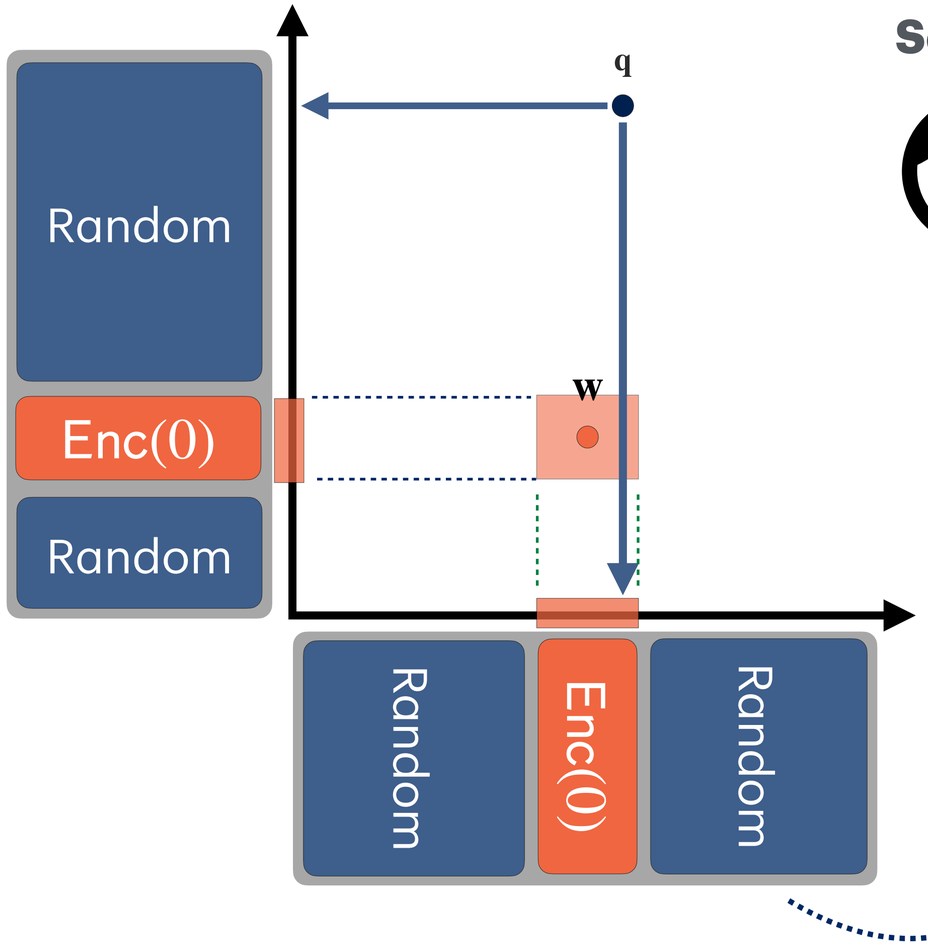
$\mathbf{q} \in \mathbb{Z}^2$

Fuzzy Matching for L_∞

High-level ideas to get sublinear computations

Receiver

$\mathbf{w} \in \mathbb{Z}^2$



Sender

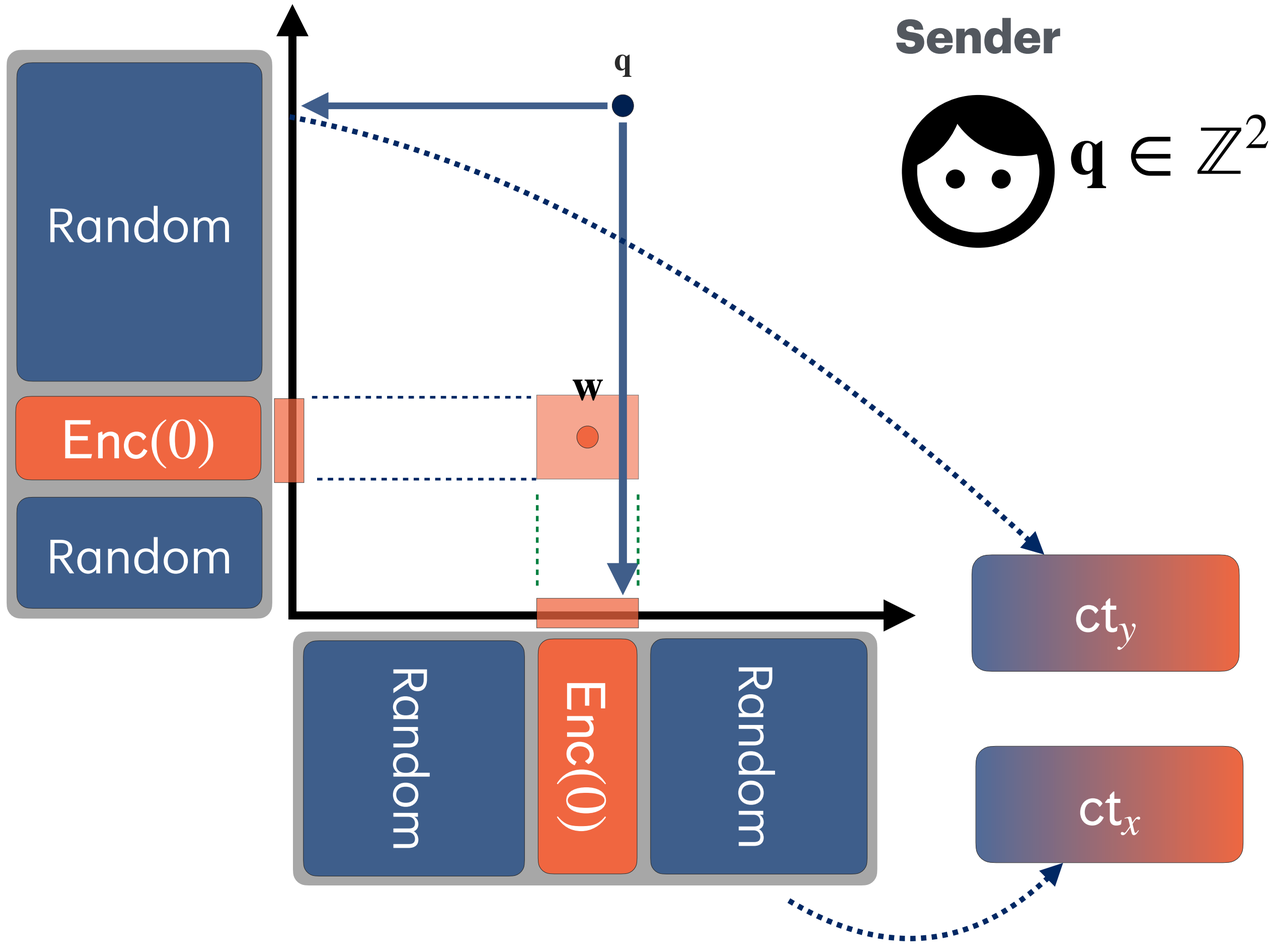
$\mathbf{q} \in \mathbb{Z}^2$

Fuzzy Matching for L_∞

High-level ideas to get sublinear computations

Receiver

sk $\mathbf{w} \in \mathbb{Z}^2$



Sender

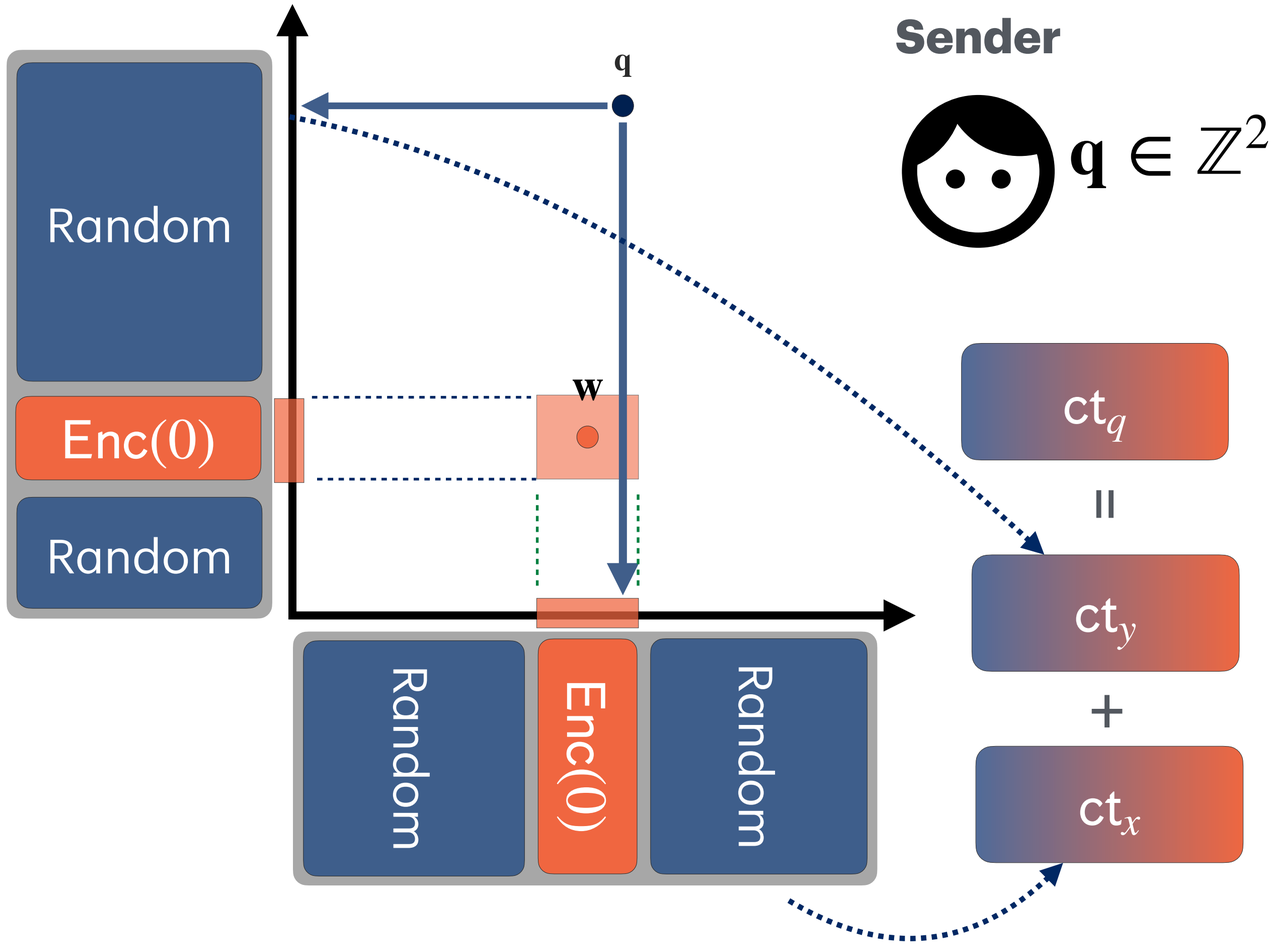
$\mathbf{q} \in \mathbb{Z}^2$

Fuzzy Matching for L_∞

High-level ideas to get sublinear computations

Receiver

sk  $\mathbf{w} \in \mathbb{Z}^2$



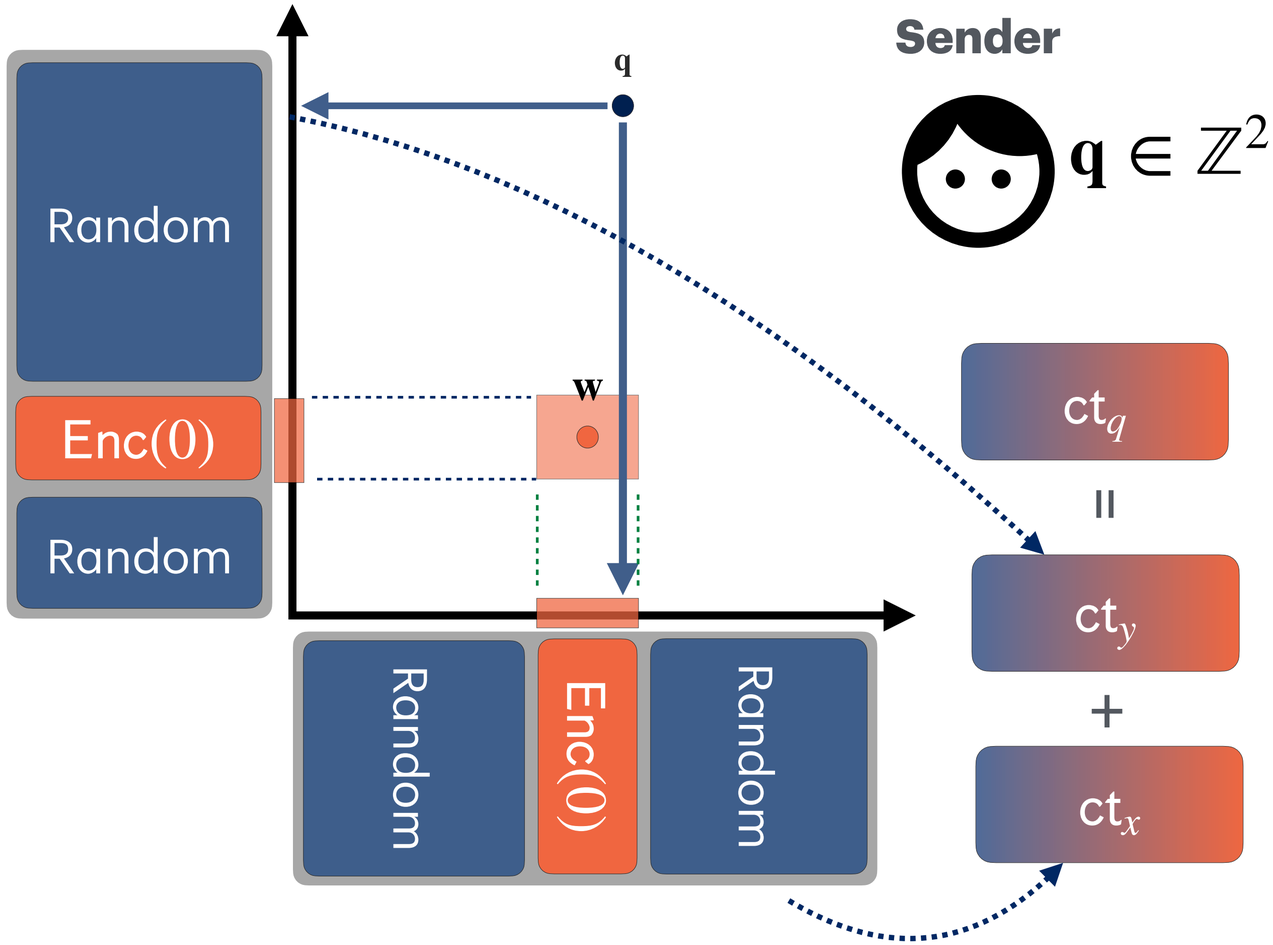
Fuzzy Matching for L_∞

High-level ideas to get sublinear computations

Receiver

sk  $\mathbf{w} \in \mathbb{Z}^2$

Dec(sk, ct_q) $\stackrel{?}{=} 0$



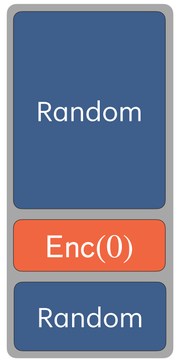
Fuzzy Matching for L_∞

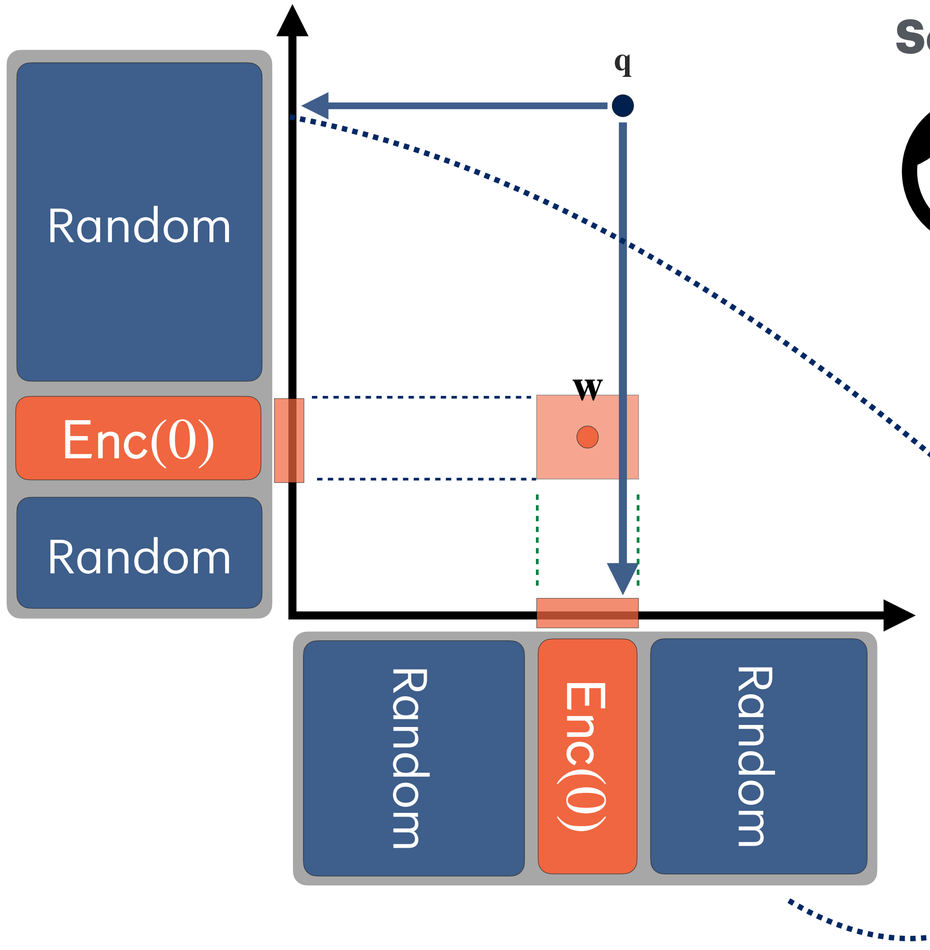
High-level ideas to get sublinear computations

Receiver

sk  $\mathbf{w} \in \mathbb{Z}^2$

Dec(sk, ct_q) $\stackrel{?}{=} 0$





Sender

 $\mathbf{q} \in \mathbb{Z}^2$

ct_q

=

ct_y

+

ct_x

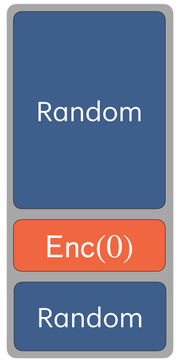
Fuzzy Matching for L_∞

High-level ideas to get sublinear computations

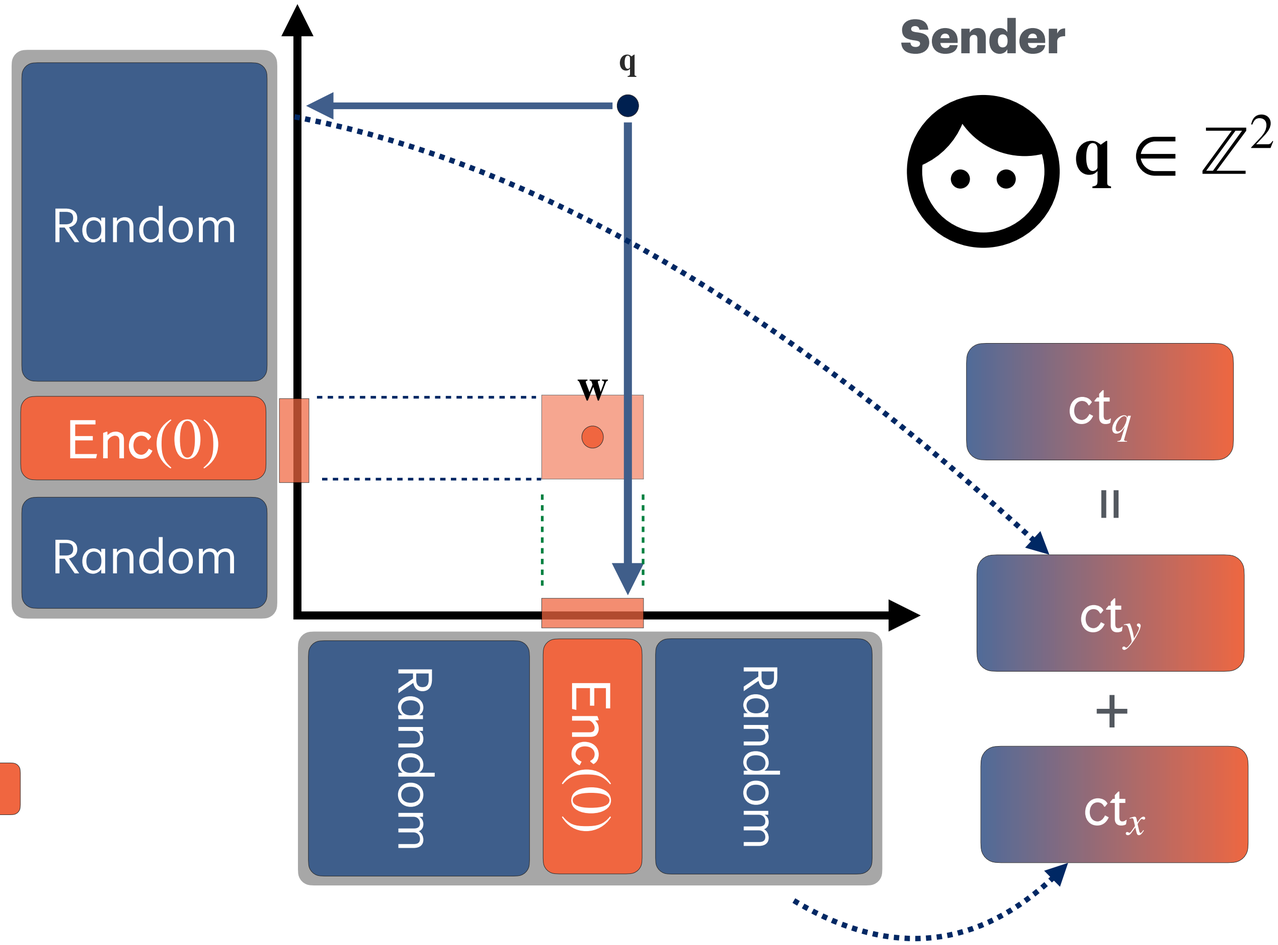
Receiver

sk  $\mathbf{w} \in \mathbb{Z}^2$

Dec(sk, ct_q) $\stackrel{?}{=} 0$

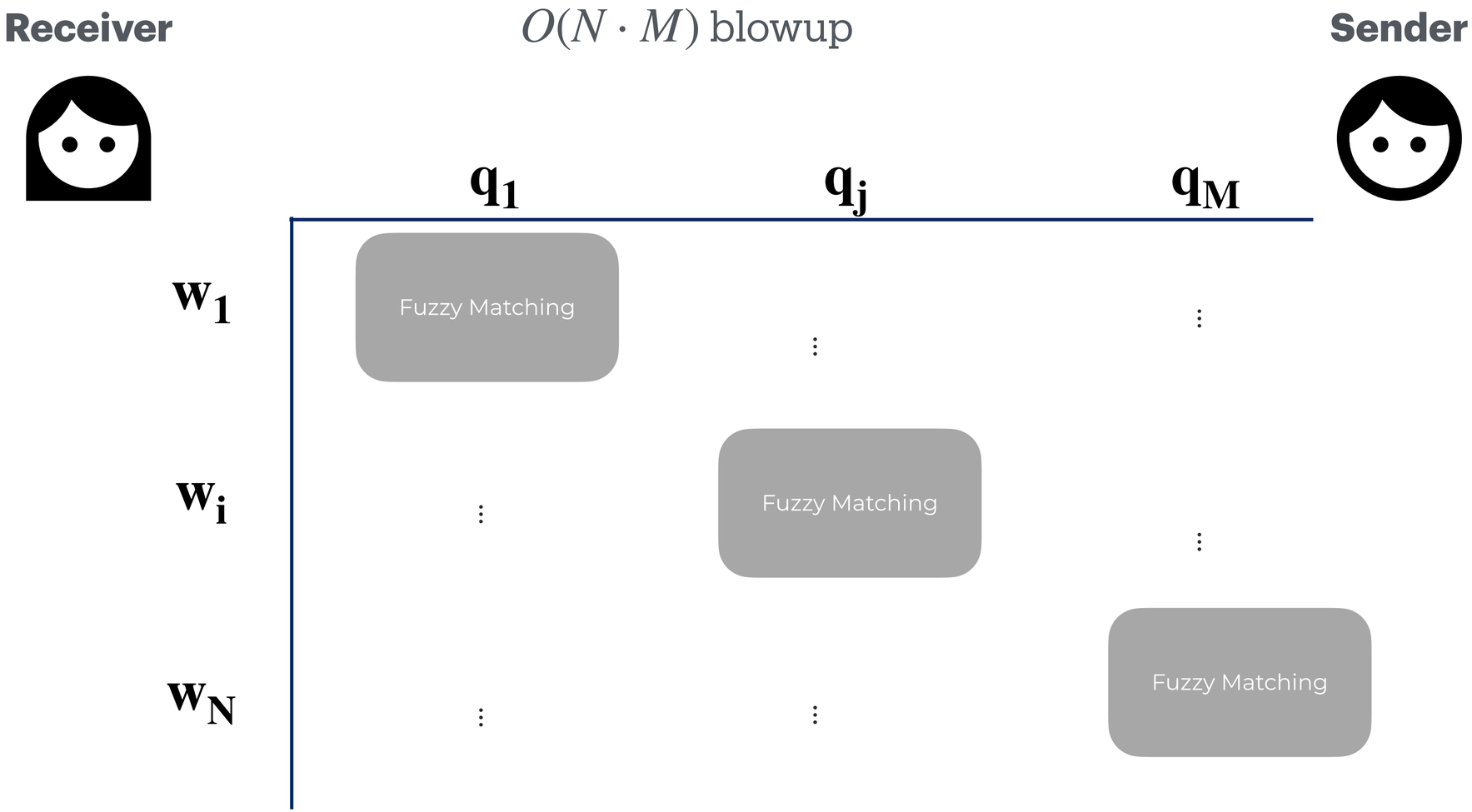


- Encoding time and size only depend on Enc(0)
- Decoding time is constant



Full Picture

From Fuzzy Matching to Fuzzy PSI



Fuzzy PSI

For Low Dimensions

Fuzzy PSI

For Low Dimensions

- If the dimension is **low**
  - Use **Spatial Hashing** Techniques from prior works [GRS22,23]
  - Tile the space with **hypercubes**
  - Avoid the quadratic blowup
  - Both comm. and comp. costs scale with $O(2^d)$
    - This is already **sublinear** to the volume

Fuzzy PSI

For High Dimensions

Fuzzy PSI

For High Dimensions

- For **high** dimensions, the factor $O(2^d)$ is still expensive.

Fuzzy PSI

For High Dimensions

- For **high** dimensions, the factor $O(2^d)$ is still expensive.
- We provide two approaches.

Fuzzy PSI

For High Dimensions

- For **high** dimensions, the factor $O(2^d)$ is still expensive.
- We provide two approaches.
- Assume the **receiver's points** satisfying **some property,** we can
  - Reduce the factor from $O(2^d)$ to $O(d^2)$
  - Have **linear** blowup on parties' set size

Fuzzy PSI

For High Dimensions

- For **high** dimensions, the factor $O(2^d)$ is still expensive.
- We provide two approaches.
- Assume the **receiver's points** satisfying **some property,** we can
  - Reduce the factor from $O(2^d)$ to $O(d^2)$
  - Have **linear** blowup on parties' set size
- Relying on **Locality Sensitive Hashing**, we can
  - Reduce the factor from $O(2^d)$ to $O(d)$
  - Have **sub-quadratic** blowup on parties' set size

Summary

Summary

- Fuzzy ***Matching*** for two points

Summary

- Fuzzy ***Matching*** for two points
- From Fuzzy Matching to Fuzzy ***PSI***

Summary

- Fuzzy ***Matching*** for two points
- From Fuzzy Matching to Fuzzy ***PSI***
- Refer to ***https://ia.cr/2024/330*** for:
  - Detailed techniques and constructions
  - ***Generalized distance*** functions
  - ***Generalized functionalities*** and variants

Summary

Future Works

- Fuzzy ***Matching*** for two points
- From Fuzzy Matching to Fuzzy ***PSI***
- Refer to ***https://ia.cr/2024/330*** for:
  - Detailed techniques and constructions
  - ***Generalized distance*** functions
  - ***Generalized functionalities*** and variants

Summary

- Fuzzy ***Matching*** for two points
- From Fuzzy Matching to Fuzzy ***PSI***
- Refer to ***https://ia.cr/2024/330*** for:
  - Detailed techniques and constructions
  - ***Generalized distance*** functions
  - ***Generalized functionalities*** and variants

Future Works

- An obvious one:
  - ***Malicious*** security
- Improvement in high dimensions, without using LSH
- Can we minimize the ***public-key operations***?

Thanks for Listening!