Efficient Fuzzy Private Set Intersection from Secret-shared OPRF

Xinpeng Yang[1], Meng Hao[2*], Chenkai Weng[3], Robert H. Deng[2], Yonggang Wen[1], Tianwei Zhang[1]
[1]*Nanyang Technological University, xinpeng004@e.ntu.edu.sg, {ygwen, tianwei.zhang}@ntu.edu.sg*
[2]*Singapore Management University, menghao303@gmail.com, robertdeng@smu.edu.sg*
[3]*Arizona State University, chenkai.weng@asu.edu*

***Abstract*—Private set intersection (PSI) enables a sender holding a set Q of size m and a receiver holding a set W of size n to securely compute the intersection $Q \cap W$. Fuzzy PSI (FPSI) is a PSI variant where the receiver learns the items $q \in Q$ for which there exists some $w \in W$ satisfying $\mathsf{dist}(q, w) \leq \delta$ under a given distance metric. Although several FPSI works are proposed for L_p distance metrics with $p \in [1, \infty]$, they either heavily rely on expensive homomorphic encryptions, or incur undesirable complexity, e.g., exponential to the element dimension, both of which lead to poor practical efficiency.**

In this work, we propose efficient FPSI protocols for $L_{p\in[1,\infty]}$ distance metrics, primarily leveraging significantly cheaper symmetric-key operations. Our protocols achieve linear communication and computation complexity in the set sizes m, n, the dimension d, and the distance threshold δ. Our core building block is an oblivious programmable PRF with secret-shared outputs, which may be of independent interest. Furthermore, we incorporate a prefix technique that reduces the dependence on the distance threshold δ to logarithmic, which is particularly suitable for large δ.

We implement our FPSI protocols and compare them with state-of-the-art constructions. Experimental results demonstrate that our protocols consistently and significantly outperform existing works across all settings. Specifically, our protocols achieve a speedup of $12\sim145\times$ in running time and a reduction of $3\sim8\times$ in communication cost compared to Gao et al. (ASIACRYPT'24) and a speedup of $9\sim80\times$ in running time and a reduction of $5\sim19\times$ in communication cost compared to Dang et al. (CCS'25).

1. Introduction

Private set intersection (PSI) [1], [2], [3], [4], [5], [6], [7] enables two parties, the sender and the receiver, to compute the intersection of their sets without revealing additional information beyond the intersection itself. Standard PSI protocols focus on exact matching and have been widely employed in various scenarios such as password breach monitoring and genome matching [8], [9]. However, in some complicated applications with multi-dimensional inputs, requiring exact intersection is often impractical or even impossible. For example, in biometric authentication systems (e.g., fingerprint or face), two samples belonging to the same individual may differ due to environmental variations and feature extraction perturbations [10], which limits the application of standard PSI protocols.

To address this problem, fuzzy PSI extends PSI by allowing the receiver, holding the set W, to learn those items $q \in Q$ from the sender for which there exists some $w \in W$ satisfying $\mathsf{dist}(q, w) \leq \delta$ under a specified distance metric. Recently, a substantial body of works [11], [12], [13], [14], [15], [16], [17], [18], [19], [20] have proposed a variety of fuzzy PSI constructions. Among these, van Baarsen and Pu [13] introduced the first generic fuzzy PSI protocols supporting arbitrary $L_{p\in[1,\infty]}$ distance. Although several follow-up works [17], [18] further improve performance, these protocols still incur super-linear complexity, e.g., exponential in the element dimension, which becomes prohibitively expensive in high-dimensional settings.

More recently, Gao et al. [14] proposed the first fuzzy PSI protocols for $L_{p\in[1,\infty]}$ distance with linear complexity in the set size, the dimension, and the distance threshold. Dang et al. [16] further optimized this line of work by introducing the prefix representation [21] that reduces the dependence on the threshold to logarithmic, thereby reducing both computation and communication costs, particularly for large thresholds. Nevertheless, both schemes rely heavily on expensive additively homomorphic encryption operations, such as the ElGamal and Paillier schemes, which leads to poor concrete performance. Please refer to Table 1 and Section 1.2 for a more detailed discussion of related works.

Motivated by the above, we raise the following question:

> Can we construct concretely efficient fuzzy PSI protocols for general $L_{p\in[1,\infty]}$ distance with linear complexity in the set size and the element dimension?

1.1. Our Contribution

We answer the above question affirmatively and summarize our contributions as follows.

1) **Shared-output OPPRF.** We introduce a new building block, termed oblivious programmable PRF with secret-shared outputs (so-OPPRF), which enables oblivious evaluations of programmable PRF without revealing outputs to either party individually.

* Meng Hao is the corresponding author

TABLE 1: Asymptotic complexities of existing fuzzy PSI protocols for $L_{p\in[1,\infty]}$ distance, where the sender holds m elements and the receiver holds n elements in a d-dimensional space, δ is the distance threshold. We ignore multiplicative factors of the computational security parameter κ and statistical security parameter λ.

Metric	Protocol	Assumption	Communication	Computation	
				Sender	Receiver
L_∞	[13]	$\mathcal{R}, \min > 2\delta$	$O(\delta dn + 2^d m)$	$O(2^d dm)$	$O(\delta dn + 2^d m)$
		$\mathcal{R}, \min > 4\delta$	$O(\delta 2^d dn + m)$	$O(dm)$	$O(\delta 2^d dn + m)$
		$\mathcal{R}$, disj. proj.	$O((\delta d)^2 n + m)$	$O(d^2 m)$	$O((\delta d)^2 n + m)$
	[20]	$\mathcal{R}$, mini-univ.	$O((dn \log \delta + m(2 \log \delta)^d))$	$O(m(2 \log \delta)^d)$	$O((\log \delta)^d n + dm \log \delta)$
	[18]	$\mathcal{R} \wedge \mathcal{S}$, $\min > 4\delta$	$O(d(m + 2^d n) \log \delta)$	$O(d(m + 2^d n) \log \delta)$	$O(2^d n(d \log \delta + (\log \delta)^{d/2}))$
	[15]	$\mathcal{R} \wedge \mathcal{S}$, s-separate	$O(\delta^s \frac{d}{s}(m + n))$	$O(\delta^s \frac{d}{s} m + n)$	$O(\delta^s \frac{d}{s} n + m)$
	[17]	$\mathcal{S}$, disj. hash	$O(d(\delta m + 2^d n))$	$O(\delta dm)$	$O(d2^d n + m)$
	[19]	$\mathcal{S}$, disj. hash	$O(d \log \delta (n2^d + m2^{d-s}))$	$O(2^{d-s} dm \log \delta)$	$O(2^s dn \log \delta)$
	[14]	$\mathcal{R} \wedge \mathcal{S}$, disj. proj.	$O(\delta d(m + n))$	$O(\delta dm + n)$	$O(\delta dn + m)$
	[16]	$\mathcal{R} \wedge \mathcal{S}$, disj. proj.	$O\,(d(m + n) \log \delta)$	$O\,(d(m + n) \log \delta)$	$O\,(d(m + n) \log \delta)$
	Ours	$\mathcal{R} \wedge \mathcal{S}$, disj. proj.	$O\,(\delta d(m + n))$	$O\,(\delta dm + dn)$	$O\,(\delta dn + dm)$
	Ours-prefix	$\mathcal{R} \wedge \mathcal{S}$, disj. proj.	$O\,(d(m + n) \log \delta)$	$O\,(d(m + n) \log \delta)$	$O\,(d(m + n) \log \delta)$
$L_{p\in[1,\infty)}$	[13]	$\mathcal{R}, \min > 2\delta(d^{1/p} + 1)$	$O(\delta 2^d dn + \delta^p m)$	$O((d + \delta^p)m)$	$O(\delta 2^d dn + m)$
		$\mathcal{R}, \min > \delta/\rho$	$O(\delta dn^{1+\rho} + \delta^\rho mn^\rho \log n)$	$O((d + \delta^\rho)mn^\rho \log n)$	$O(\delta dn^{1+\rho} + mn^\rho \log n)$
	[18]	$\mathcal{R} \wedge \mathcal{S}$, $\min > 2\delta(d^{1/p} + 1)$	$O(d(\delta m + 2^d n) + p(m + 2^d n) \log \delta)$	$O(d(\delta m + 2^d n) + p(m + 2^d n) \log \delta)$	$O(2^d n(d + p \log \delta))$
	[15]	$\mathcal{R} \wedge \mathcal{S}$, s-separate	$O(\delta^s \frac{d}{s}(m + n) + pm \log \delta)$	$O((\delta^s \frac{d}{s} + p \log \delta)m + n)$	$O(\delta^s \frac{d}{s} n + pm \log \delta)$
	[17]	$\mathcal{S}$, disj. hash	$O(d(\delta m + 2^d n))$	$O(\delta dm)$	$O(d2^d n + m)$
	[14]	$\mathcal{R} \wedge \mathcal{S}$, disj. proj.	$O(\delta d(m + n) + pm \log \delta)$	$O(\delta dm + pm \log \delta + n)$	$O(\delta dn + pm \log \delta)$
	[16]	$\mathcal{R} \wedge \mathcal{S}$, disj. proj.	$O\,(dpn \log \delta + dm \log \delta)$	$O\,(dpm \log \delta + dn \log \delta)$	$O\,(dpn \log \delta + dm \log \delta)$
	Ours	$\mathcal{R} \wedge \mathcal{S}$, disj. proj.	$O\,(\delta d(m + n) + pm \log \delta)$	$O\,(\delta dm + dn + pm \log \delta)$	$O\,(\delta dn + dm + pm \log \delta)$
	Ours-prefix	$\mathcal{R} \wedge \mathcal{S}$, disj. proj.	$O\,(dpn \log \delta + dpm \log \delta)$	$O\,(dpm \log \delta + dpn \log \delta)$	$O\,(dpn \log \delta + dpm \log \delta)$

– $\mathcal{R}/\mathcal{S}$ denotes that the set of receiver/sender satisfies the assumption, and $\mathcal{R} \wedge \mathcal{S}$ indicates that both sets satisfy the assumption.
– disj. proj. is short for disjoint projection assumption, where each element has disjoint projections in some dimension with all other elements.
– disj. hash is short for disjoint hash assumption, which means the spatial hashing scheme maps at most one point to every possible target grid cell.
– mini-univ is short for mini-universe assumption, where every ball can be mapped to a distinct vertex of the space.
– s-separate means for every s dimensions, elements have disjoint projections in one of these s dimensions with all other elements where $1 \le s \le d$.
– $0 < \rho < 1/c$ is a parameter in locality-sensitive hashing where the distance between any receiver's two elements is greater than $c\delta$.
– $\min > l$ means that the minimum distance between any two elements of the set is greater than l.

2) **Modular fuzzy mapping.** We present a modular protocol design of fuzzy mapping, which is a core component of fuzzy PSI, from shared-output OPPRF and shared-input OPRF.
3) **Efficient fuzzy PSI.** We propose efficient fuzzy PSI protocols for $L_{p\in[1,\infty]}$ distance metrics based on our fuzzy mapping, primarily leveraging lightweight symmetric-key operations, with linear complexity in the set size, the dimension, and the distance threshold.
4) **Prefix optimizations.** We incorporate prefix techniques to optimize the overhead of fuzzy PSI protocols for large distance thresholds, which reduces the computational and communication complexity to logarithmic in the distance threshold.
5) **Extensive evaluations.** We conduct extensive experiments under various parameter settings. Experimental results demonstrate that our protocols consistently and significantly outperform the state-of-the-art works across all settings, particularly, up to $80\times$ faster computation and $19\times$ lower communication.

1.2. Related Work

Fuzzy PSI for Hamming distance. For a long time, research on fuzzy PSI primarily focused on the Hamming distance. Freedman et al. [2] first introduced the problem of secure fuzzy matching and proposed a protocol based on additively homomorphic encryption (AHE). Subsequent works [10], [21], [22], [23], [24], [25] improve the protocol's complexity based on different techniques and assumptions.

Fuzzy PSI for Minkowski distance. More recently, a series of works have explored fuzzy PSI for $L_{p\in[1,\infty]}$ metrics. Garimella et al. [11] introduced the notion of structure-aware PSI, in which one party holds a structured dataset while the other holds an unstructured set of points. Their construction can be extended to fuzzy PSI for L_∞ distance, but has a cost that scales with δ^d. Chakraborti et al. [21] studied the problem for L_1 distance and presented a useful technique known as prefix representation. Then, Garimella et al. [12] and Bui et al. [20] improved their previous work [11], reducing the complexity to $(\log \delta)^d$ with prefix representation.

van Baarsen and Pu [13] proposed the first fuzzy PSI protocol for general L_∞ and L_p distances based on the Decisional Diffie–Hellman (DDH) assumption. Their construct supposed each element has a minimum distance (2δ or 4δ) from other elements. They subsequently improved their design using lightweight primitives [18], which significantly reduced the overall computational overhead. Similarly, they used prefix representation to achieve a complexity of $(\log \delta)^d$.

Several follow-up works further improved the performance but still incurred exponential complexity. Piske et al. [17] constructed the fuzzy PSI protocol based on a newly introduced primitive, distance-aware OT. They followed the disjoint hash assumption [19] where the whole input space is cut into grid cells and the spatial hashing scheme maps at most one element to every possible target grid cell. Their techniques rely on symmetric cryptographic primitives; however, their construction remains exponential

in the dimension d. Moreover, their protocol operates over a relatively small domain and supports only two-dimensional inputs for the L_2 distance, which limits its practicality. Richardson et al. [19] generalized the approach of [26] and built a fuzzy PSI protocol using generic two-party computation, resulting in poor concrete efficiency. Chongchitmate et al. [27] proposed a protocol that achieves nearly linear complexity in the set size. Nevertheless, their design relies on garbled circuits and multiple PSI invocations, which significantly limits its practical efficiency.

Building on [13], Gao et al. [14] proposed the first fuzzy PSI for $L_{p\in[1,\infty]}$ metrics, achieving linear complexity with respect to all parameters. They assumed that the set of both parties satisfies the disjoint projection assumption, where each element has disjoint projections in some dimension from all other elements. Later, Dang et al. [16] further improved this approach by adopting prefix representation to reduce the complexity to $\log\delta$. Nevertheless, both Gao et al. [14] and Dang et al. [16] heavily rely on the AHE, incurring considerable computational overhead. Recently, Zhang et al. [15] improved [14] by introducing a new framework for fuzzy PSI based on a stronger "s-separate" assumption. Although [15] solely uses symmetric cryptographic primitives, it suffers from a relatively high exponential factor δ^s for both communication and computation overhead, especially for large δ.

Other Solutions. A closely related line of works study private record linkage (PRL) [28], [29], [30], [31], [32], [33], [34], [35], whose goal is to identify common entities across disparate datasets while protecting the privacy of non-matching records. Unlike our setting, these works typically do not rely on assumptions on the input distribution. Instead, many of them employ locality-preserving or locality-sensitive hashing techniques [36], [37] to project multi-dimensional records into lower-dimensional representations, often at the expense of matching accuracy.

Existing PRL protocols have been instantiated using a variety of privacy-enhancing techniques, including Bloom filters [38], differential privacy [39], and secure multiparty computation (MPC). Among them, only a small number of works [32], [33], [34], [35] provide cryptographic security without additional leakage. In particular, Khurram and Kerschbaum [32] proposed the first efficient cryptographically secure PRL protocol based on MPC. Their approach avoids the quadratic cost of all-pairs comparison by first securely sorting the records and then comparing only records that fall within the same comparison window. Nevertheless, the overall cost of their construction is dominated by the secure merge sorting procedure, resulting in an $O(n\log n)$ complexity, where n denotes the total number of records. Wei et al. [33] improved this line of works in both efficiency and empirical accuracy by incorporating locality-sensitive hashing [40]. However, their protocol leaks the number of comparisons, and its overall complexity remains $O(n\log n)$. To achieve stronger privacy guarantees, Stammler et al. [34] abandoned locality-based blocking and instead compare all record pairs, incurring a quadratic $O(n^2)$ complexity. Adir et al. [35] combined locality-sensitive hashing [37] with PSI to obtain linear complexity. However, replacing generic MPC-based circuit for distance computation with standard PSI significantly reduces the computational overhead, but this simplification comes at the cost of reduced matching accuracy.

While PRL protocols address the trade-off among security, efficiency, and accuracy in more general application settings, our goal is to design a protocol for structured sets satisfying certain assumptions, which allows us to avoid introducing false positives or negatives. Accordingly, we treat PRL as a closely related series of works, but center our subsequent discussion on prior fuzzy PSI protocols that realize the same functionality under certain assumptions.

In this work, we focus on constructing concretely efficient fuzzy PSI protocols with linear complexity in both the set size and the dimension, while avoiding expensive public-key primitives such as AHE used in existing linear-complexity constructions [14], [16]. Table 1 compares the asymptotic complexities of our protocols with prior works for $L_{p\in[1,\infty]}$ distance. Among prior works under the same assumptions, we report only the protocols with the best asymptotic complexity.

2. Technical overview

We begin by introducing the necessary notations. The sender $\mathcal{S}$ holds a set $Q = \{q_j\}_{j\in[m]}$ of size m, and the receiver $\mathcal{R}$ holds a set $W = \{w_i\}_{i\in[n]}$ of size n, both defined over a d-dimensional space. We use $q_{j,k}$ to denote the k-th coordinate of q_j. For any $q_j \in Q$ and $w_i \in W$, we say that q_j and w_i are δ-close if $\mathsf{dist}(q_j, w_i) \le \delta$, where δ denotes the distance threshold.

2.1. Fuzzy PSI Paradigm from Fuzzy Mapping

Recent works [13], [14], [16], [17] show that fuzzy PSI protocols typically consist of two phases: *coarse mapping* and *refined filtering*. (1) In the coarse mapping phase, each element in both parties' sets is mapped to an identifier (ID). Any two elements that are within distance δ, one from the sender and one from the receiver, are guaranteed to receive the same ID. This phase may introduce false positives, but no false negatives. (2) In the refined filtering phase, for each candidate pair with the same ID, the parties further perform an exact distance check to eliminate the false positives generated during the coarse mapping phase. The above idea is similar to hashing-based standard PSI protocols and reduces the number of necessary distance comparisons from $O(mn)$ to $O(n)$ or $O(m)$.

To realize the coarse mapping phase, Gao et al. [14] propose a generalized protocol, termed *fuzzy mapping*, which takes sets Q and W as input and outputs identifier sets ID_Q and ID_W for the sender and receiver, respectively. The protocol guarantees that for any two elements $q_j \in Q$ and $w_i \in W$ satisfying $\mathsf{dist}(q_j, w_i) \le \delta$, it holds that $\mathsf{ID}_{q_j} = \mathsf{ID}_{w_i}$. Subsequently, the refined filtering phase is carried out via *fuzzy matching*, which takes as input every

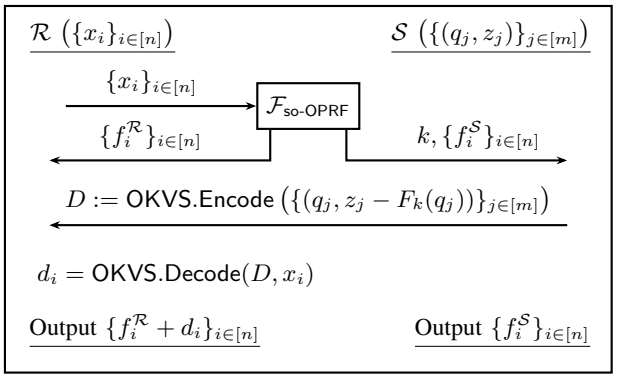

Figure 1: Construction of so-OPPRF from so-OPRF.

pair (q_j, w_i) sharing the same ID and returns q_j to the receiver if and only if $\mathsf{dist}(q_j, w_i) \leq \delta$.

In this work, we follow this fuzzy PSI paradigm, but introduce new techniques to realize the fuzzy mapping and fuzzy matching phases with linear complexity in the set size and dimension, primarily using lightweight symmetric-key operations[1].

2.2. OPPRF with Shared Outputs

An oblivious pseudorandom function (OPRF) allows a receiver to input a value x and obtain $\mathsf{PRF}(k, x)$, where the PRF key k is privately held by the sender. Programmable OPRF (OPPRF), introduced by Kolesnikov et al. [43], extends this functionality by allowing the sender to program the PRF on designated points: for each sender's chosen pair (y_i, z_i), the functionality enforces $\mathsf{PRF}(k, y_i) = z_i$, while all non-programmed inputs receive pseudorandom outputs.

In this work, we introduce a variant of OPPRF, termed *shared-output OPPRF* (so-OPPRF), which serves as a core building block in our fuzzy mapping and fuzzy PSI protocols. A so-OPPRF is identical to a standard OPPRF except that it outputs secret shares of $\mathsf{PRF}(k, x)$ to the two parties rather than revealing the PRF value to the receiver.

We construct an efficient so-OPPRF protocol by combining an oblivious key-value store (OKVS) [44] with a shared-output OPRF (so-OPRF) [45], [46], following the classical OPPRF paradigm [47], [48]. Concretely, the parties first run a so-OPRF, in which the sender inputs a PRF key k and the receiver inputs a set X, and they obtain secret shares $f_i^{\mathcal{S}}, f_i^{\mathcal{R}}$ of $\mathsf{PRF}(k, x_i)$ for each $x_i \in X$. To program the PRF on the chosen points $\{(q_j, z_j)\}_{j\in[m]}$, the sender uses OKVS to encode these points. OKVS [44] consists of two algorithms: Encode takes as input a set of key-value pairs $\{(k_j, v_j)\}_{j\in[m]}$ and outputs an encoding D while Decode takes as input a key k and D, and outputs a value v. The correctness ensures that $\mathsf{OKVS.Decode}(D, k_j)$ outputs v_j, and the obliviousness means that if the OKVS encodes random values, the encoding D is independent of the encoded keys. To this end, the sender computes an OKVS encoding D on $\{(q_j, z_j - \mathsf{PRF}(k, q_j))\}_{j\in[m]}$ and sends D to the receiver. Finally, the sender outputs $y_i^{\mathcal{S}} := f_i^{\mathcal{S}}$, while the receiver outputs $y_i^{\mathcal{R}} := f_i^{\mathcal{R}} + \mathsf{OKVS.Decode}(D, x_i)$. For each programmed point $x_i = q_j$, correctness holds immediately since $y_i^{\mathcal{S}} + y_i^{\mathcal{R}} = f_i^{\mathcal{S}} + f_i^{\mathcal{R}} + z_j - \mathsf{PRF}(k, q_j) = z_j$. The pseudorandomness of unprogrammed points follows from the functionality of so-OPRF.

The above steps are illustrated in Figure 1. In the following Sections 2.3 and 2.4, we will show how to design fuzzy PSI from so-OPPRF.

2.3. Fuzzy Mapping from so-OPPRF and si-OPRF

We present a modular framework of fuzzy mapping with the following two steps. (1) Local mapping: the sender constructs a private local mapping H_Q that maps the set Q's elements to unique local IDs, while guaranteeing that for each $w_i \in W$, if there exists q_j such that $\mathsf{dist}(q_j, w_i) \leq \delta$, $H_Q(w_i) = H_Q(q_j)$ holds. Similarly, the receiver generates a private local mapping H_W. (2) Global mapping: two parties interactively compute the global IDs as $\mathsf{ID}_{q_j} := \mathsf{PRF}_k(H_Q(q_j) + H_W(q_j))$ and $\mathsf{ID}_{w_i} := \mathsf{PRF}_k(H_Q(w_i) + H_W(w_i))$ without revealing any information about Q, W. Along with the correctness of local mapping, it ensures that if $\mathsf{dist}(q_j, w_i) \leq \delta$, it holds $\mathsf{ID}_{q_j} = \mathsf{ID}_{w_i}$. It is worth noting that compared to the existing fuzzy mapping [14], [16], our main contribution lies in the modular PRF-based global mapping design, which simplifies protocol comprehension and facilitates efficient protocol instantiation. We elaborate on these aspects and the underlying technical methods in the remainder of this section.

To instantiate the above framework, we present an efficient fuzzy mapping protocol for L_∞ distance. The following description focuses on the receiver's protocol due to the symmetric design. Specifically, in the first local mapping step, same as the prior works [14], [15], [16], for each $q_{j,k}$, the sender locally assigns a random value $r_{j,k}$ to the interval $[q_{j,k} - \delta, q_{j,k} + \delta]$ centered on $q_{j,k}$, formally represented as a set of the key-value pairs $L := \{(k\|(q_{j,k}+t), r_{j,k})\}_{j\in[m],k\in[d],t\in[-\delta,\delta]}$. We will explain how to handle overlaps later. The sender then defines the local mapping H_Q such that $H_Q(q_j + \xi) := \sum_{k\in[d]} r_{j,k}$ for each $q_j \in Q$ and $\xi \in [-\delta, \delta]^d$. Then, in the global mapping step, to obtain $H_Q(w_i)$ without revealing any information to each other, the receiver obliviously retrieves values $r_{w_i,k}$ from the sender's key-value list L with the inputs $k\|w_{i,k}$ for each dimension k. It needs to ensure that for each $k \in [d]$, if $|w_{i,k} - q_{j,k}| \leq \delta$, then $r_{w_i,k} = r_{j,k}$ holds and hence $H_Q(w_i) = \sum_{k\in[d]} r_{w_i,k} = H_Q(q_j)$; otherwise, $r_{w_i,k}$ is a random value.

We observe that the above functionality can be realized by invoking the OPPRF protocol, where the sender inputs the programmed points L, and the receiver inputs queries $k\|w_{i,k}$ and learns either the designated $r_{j,k}$ or

[1] Our protocols invoke OT and VOLE, instantiated via pseudorandom correlation generators (PCGs) [41], [42]. In PCGs, the seed setup phase requires a small number of base OTs implemented with public-key operations, while the seed expansion phase relies on linear codes. Both phases are highly efficient in practice [17], [18].

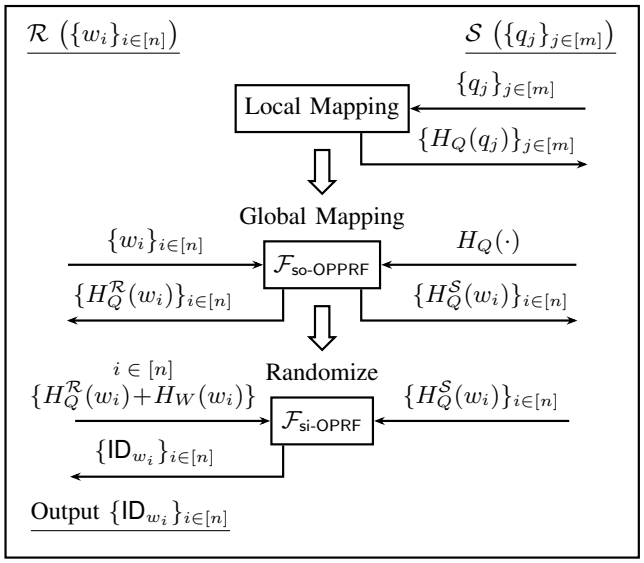

Figure 2: Construction of fuzzy mapping from so-OPPRF and si-OPRF. The sender and receiver can switch roles and repeat the process symmetrically.

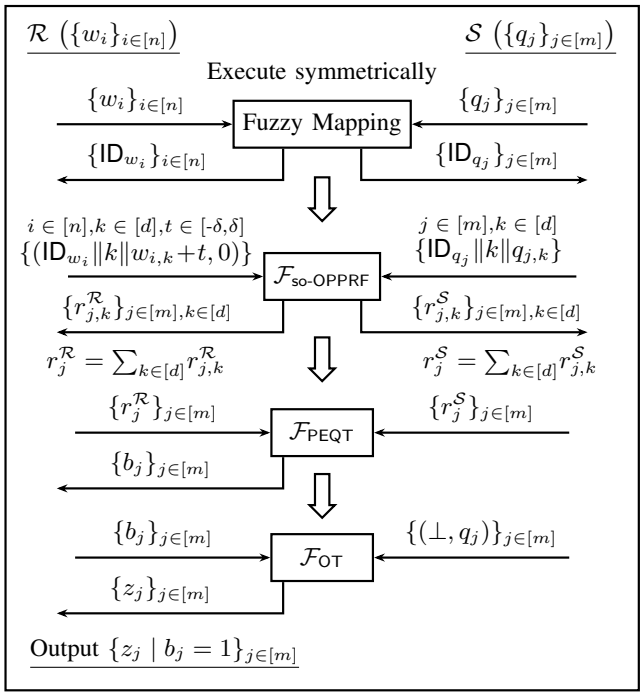

Figure 3: Construction of fuzzy PSI from so-OPPRF.

pseudorandom values. However, this will leak to the receiver additional information about partial matches on individual dimensions. That is, since the sender programs all points in the interval $[q_{j,k} - \delta, q_{j,k} + \delta]$ to the same $r_{j,k}$, the receiver learns the same OPPRF evaluation on two different $w_{i,k}, w_{i',k} \in [q_{j,k} - \delta, q_{j,k} + \delta]$, leaking the range of some $q_{j,k}$. Existing solutions [14], [16] employ AHE to avoid this leakage, but result in prohibitively large overhead.

Applying so-OPPRF. We adopt our so-OPPRF protocol to prevent the information leakage to the receiver while achieving better efficiency than AHE-based solutions. Specifically, with the same inputs L and $\{k\|w_{i,k}\}_{k\in[d]}$ as above, so-OPPRF returns the secret shares $\{r^{\mathcal{S}}_{w_i,k}\}_{k\in[d]}$ to the sender and $\{r^{\mathcal{R}}_{w_i,k}\}_{k\in[d]}$ to the receiver. Then, the sender sets $H_Q^{\mathcal{S}}(w_i) := \sum_{k\in[d]} r^{\mathcal{S}}_{w_i,k}$ and the receiver sets $H_Q^{\mathcal{R}}(w_i) := \sum_{k\in[d]} r^{\mathcal{R}}_{w_i,k}$, both of which are the secret shares of $H_Q(w_i)$.

However, $H_Q(w_i)$ can not be directly revealed to the receiver. The reason is that in the local mapping [14], when there are overlapped intervals, later random values will overwrite earlier values. This enlarges the valid radius-δ interval of $q_{j,k}$, thereby introducing false positives. This may result in two elements w_{i_1}, w_{i_2} satisfying $H_Q(w_{i_1}) = H_Q(w_{i_2})$ even if $\mathsf{dist}(w_{i_1}, w_{i_2}) > \delta$, which leaks the sender's information, i.e., there is a sender's element nearby w_{i_1}, w_{i_2}. Existing protocols [14], [16] employ a symmetric execution[2] to additionally compute $H_W(w_i)$ and derive ID_{w_i} from $H_Q(w_i) + H_W(w_i)$ with a customized combination of expensive AHE and Diffie–Hellman key-exchange.

Applying si-OPRF. To address this leakage, we introduce a new building block, called OPRF with secret-shared inputs (si-OPRF), and realize it with MPC-friendly PRF [46]. Different from so-OPRF, in si-OPRF, the PRF key k and the PRF input x are secret shared between two parties. After evaluation, it returns the plain PRF output $\mathsf{PRF}_k(x)$ to the receiver. To obtain the final IDs ID_{w_i}, the sender and the receiver input secret shares $H_Q^{\mathcal{S}}(w_i)$ and $H_Q^{\mathcal{R}}(w_i)+H_W(w_i)$ and secret shared k into si-OPRF. Then the receiver learns $\mathsf{ID}_{w_i} := \mathsf{PRF}_k(H_Q(w_i) + H_W(w_i))$. This ensures that even for $H_Q(w_{i_1}) = H_Q(w_{i_2})$, if local mapping ensures that $H_W(w_{i_1}) \neq H_W(w_{i_2})$, the receiver learns different $\mathsf{ID}_{w_{i_1}}$ and $\mathsf{ID}_{w_{i_2}}$. Similarly, the sender learns $\mathsf{ID}_{q_j} := \mathsf{PRF}_k(H_Q(q_j) + H_W(q_j))$, where the same secret-shared PRF key k is used by both parties in two si-OPRF invocations. This ensures that $\mathsf{ID}_{q_j} = \mathsf{ID}_{w_i}$ for $\mathsf{dist}(q_j, w_i) \leq \delta$ due to $H_Q(q_j) = H_Q(w_i)$ and $H_W(q_j) = H_W(w_i)$. The above processes are illustrated in Figure 2.

2.4. Fuzzy PSI from so-OPPRF

The fuzzy mapping protocol above guarantees that all δ-close elements between the sender and the receiver are assigned the same IDs, but it may introduce false positives. We further utilize our so-OPPRF protocol to perform a refined filtering that excludes elements that are not truly δ-close, thereby achieving fuzzy PSI for both the L_∞ and L_p distances with $p \in [1, \infty)$.

We here focus on the protocol for L_∞ distance. As shown in Figure 3, two parties first invoke fuzzy mapping procedure to get the IDs, and then execute the so-OPPRF protocol, where the receiver inputs $\{(\mathsf{ID}_{w_i}\|k\|(w_{i,k}+t),0)\}_{t\in[-\delta,\delta],i\in[n],k\in[d]}$ and the

[2] The method [16] lacks this symmetric execution and causes the above privacy leakage issue, which has been confirmed by the authors via our private communication. Table 1 gives the corrected complexity with the symmetric execution.

sender inputs $\{\mathsf{ID}_{q_j} \| k \| q_{j,k}\}_{j\in[m],k\in[d]}$. The sender receives $\{r^{\mathcal{S}}_{j,k}\}_{j\in[m],k\in[d]}$ and the receiver receives $\{r^{\mathcal{R}}_{j,k}\}_{j\in[m],k\in[d]}$. That means if $\mathsf{ID}_{q_j} = \mathsf{ID}_{w_i}$ and the distance between $q_{j,k}$ and $w_{i,k}$ is within δ, both parties will obtain the secret shares of 0 in dimension k; otherwise, they hold random values. Then, for $j \in [m]$, the sender computes $r^{\mathcal{S}}_j := \sum_{k\in[d]} r^{\mathcal{S}}_{j,k}$, and the receiver computes $r^{\mathcal{R}}_j := \sum_{k\in[d]} r^{\mathcal{R}}_{j,k}$. For each δ-close element pair with the same ID, so-OPPRF guarantees that $r^{\mathcal{S}}_j$ and $r^{\mathcal{R}}_j$ are secret shares of the designated value 0. In contrast, if ID_{q_j} is different from all IDs of W or there exists w_i with the same ID but the distance $|w_{i,k} - q_{j,k}| > \delta$ for some dimension k, then $r^{\mathcal{S}}_j$ and $r^{\mathcal{R}}_j$ will be secret shares of a random value according to the randomness property of so-OPPRF. After that, similar to prior works, both parties can employ a cheap Private Equality Test (PEQT) protocol to verify whether they hold secret shares of 0. Finally, the two parties invoke an Oblivious Transfer (OT) protocol, where the receiver selects the sender's elements based on outcomes of the PEQT protocol.

2.5. Fuzzy PSI with prefix optimizations

We further optimize our fuzzy PSI protocols using prefix techniques [12], [16], [18], [20] for large distance threshold δ. This modification reduces the communication and computation complexity from $O(\delta)$ to $O(\log \delta)$. The high-level idea to incorporate prefix techniques in our framework is that the sender in the so-OPPRF protocol does not need to program the entire interval $[q_{j,k} - \delta, q_{j,k} + \delta]$ of size $2\delta + 1$, but only $O(\log \delta)$ prefixes that together cover the interval. We present customized protocols to instantiate this optimization. In contrast to the state-of-the-art protocol [16] that employs expensive AHE, our protocols still use our so-OPPRF and the newly introduced equality-conditional selection protocol, which significantly improve overall efficiency. Please refer to Section 7 for more details.

3. Preliminary

3.1. Notation

We use κ and λ to denote the computational and statistical security parameters, respectively. Let $\mathsf{negl}(x)$ be a negligible function in x if it vanishes faster than the inverse of any polynomial in x. We use $[a, b]$ to denote the set $\{a, \ldots, b\}$ and $[a]$ to denote the set $\{1, \ldots, a\}$. For a set S, $|S|$ denotes the cardinality of S. By $r \leftarrow S$, we denote that r is sampled from the set S uniformly at random. We use $\mathbf{1}\{\mathsf{event}\}$ to denote an indicator function, which equals 1 if the event occurs and 0 otherwise. All protocols in this work are secure in the semi-honest model and the definition is deferred to Appendix A.1.

3.2. Functionality of Fuzzy PSI

We formally define the ideal functionality for fuzzy PSI in Figure 4. Same as the state-of-the-art linear-complexity fuzzy PSI works [14], [16], we utilize the disjoint projection assumption for the sets of both sender and receiver. That is, each element maintains a distance of more than 2δ on at least one dimension from the set's other elements. We define it formally as follows:

Definition 1 (Disjoint projection)**.** *A set $W \in \mathbb{U}^{n\times d}$ satisfies the δ-disjoint projection assumption, if for any $w_i \in W$, there exists $k \in [d]$ such that for any $w_j \in W$ for $j \neq i$ it holds $[w_{i,k} - \delta, w_{i,k} + \delta] \cap [w_{j,k} - \delta, w_{j,k} + \delta] = \emptyset$.*

As shown in the work [13], if the set's elements are uniformly distributed, the set satisfies the disjoint projection assumption defined in Definition 1 with probability $1 - \mathsf{negl}(d)$.

Functionality $\mathcal{F}_{\mathsf{FPSI}}$

Parameters: Sender $\mathcal{S}$ and receiver $\mathcal{R}$ with input sizes m, n, respectively. Distance metric $\mathsf{dist}(\cdot,\cdot)$ and threshold δ.

Protocol:

1) Wait for input $Q = \{q_j\}_{j\in[m]} \in \mathbb{U}^{d\times m}$ from $\mathcal{S}$ and $W = \{w_i\}_{i\in[n]} \in \mathbb{U}^{d\times n}$ from $\mathcal{R}$.
2) Output $Z := \{q_j | \exists i \in [n],\ \mathsf{dist}(q_j, w_i) \leq \delta\}$ to $\mathcal{R}$.

Figure 4: Functionality of fuzzy PSI.

3.3. Oblivious Key-Value Store

An oblivious key-value store (OKVS) [44] is a data structure consisting of two algorithms: Encode takes as input a set of key-value pairs and outputs a data structure, and Decode takes as input a key and the data structure and outputs a value. The obliviousness implies that if the OKVS encodes random values, the data structure is independent of the encoded key set.

Definition 2 (Oblivious Key-Value Store [44])**.** *An oblivious key-value store (OKVS) is parameterized by a key space $\mathcal{K}$, a value space $\mathcal{V}$, and the statistical security parameter λ, and consists of two algorithms:*

- Encode*: on input a set of key-value pairs $L \in (\mathcal{K}\times\mathcal{V})^n$, outputs a vector $D \in \mathcal{V}^m$ or a failure indicator $\perp$ with probability bounded by $1/2^\lambda$.*
- Decode*: on input a vector $D \in \mathcal{V}^m$ and a key $k \in \mathcal{K}$, outputs a value $v \in \mathcal{V}$.*

Correctness: For all $L \in (\mathcal{K} \times \mathcal{V})^n$ with distinct keys for which $D \leftarrow \mathsf{Encode}(L)$ and $D \neq \perp$, it holds that $\forall (k, v) \in L$, $\mathsf{Decode}(D, k) = v$.

Obliviousness: For any distinct $\{k_1, \ldots, k_n\} \in \mathcal{K}^n$ and $\{k'_1, \ldots, k'_n\} \in \mathcal{K}^n$, Encode does not output $\perp$ on $\{k_1, \ldots, k_n\}$ and $\{k'_1, \ldots, k'_n\}$, and then the following distributions are statistically indistinguishable

$$\{\mathsf{Encode}(\{(k_1, v_1), \ldots, (k_n, v_n)\}) \mid v_i \leftarrow \mathcal{V}, i \in [n]\} \approx_s$$
$$\{\mathsf{Encode}(\{(k'_1, v_1), \ldots, (k'_n, v_n)\}) \mid v_i \leftarrow \mathcal{V}, i \in [n]\}.$$

Double obliviousness: For any distinct $\{k_1, \ldots, k_n\} \in \mathcal{K}^n$ such that Encode does not output $\perp$ on $\{k_1, \ldots, k_n\}$, then

Functionality $\mathcal{F}_{\text{so-OPRF}}$

Parameters: Sender $\mathcal{S}$ and receiver $\mathcal{R}$. PRF $F : \mathbb{K} \times \mathbb{U} \to \mathbb{F}$.

Protocol:

1) Wait for input $X = \{x_1, \ldots, x_n\} \subseteq \mathbb{U}$ from $\mathcal{R}$.
2) Wait for input $k \in \mathbb{K}$ from $\mathcal{S}$.
3) For $i \in [n]$, compute $y_i := F_k(x_i)$ and sample $y_i^{\mathcal{S}}, y_i^{\mathcal{R}} \leftarrow \mathbb{F}$ such that $y_i^{\mathcal{S}} + y_i^{\mathcal{R}} = y_i \in \mathbb{F}$.
4) Output $\{y_i^{\mathcal{S}}\}_{i\in[n]}$ to $\mathcal{S}$ and $\{y_i^{\mathcal{R}}\}_{i\in[n]}$ to $\mathcal{R}$.

Figure 5: Functionality of oblivious PRF with secret-shared outputs.

Functionality $\mathcal{F}_{\text{si-OPRF}}$

Parameters: Sender $\mathcal{S}$ and receiver $\mathcal{R}$. PRF $F : \mathbb{K} \times \mathbb{U} \to \mathbb{F}$.

Protocol:

1) Wait for input $X^{\mathcal{S}} = \{x_1^{\mathcal{S}}, \ldots, x_n^{\mathcal{S}}\} \subseteq \mathbb{U}$ from $\mathcal{S}$ and $X^{\mathcal{R}} = \{x_1^{\mathcal{R}}, \ldots, x_n^{\mathcal{R}}\} \subseteq \mathbb{U}$ from $\mathcal{R}$ such that $x_i := x_i^{\mathcal{S}} + x_i^{\mathcal{R}} \in \mathbb{F}$.
2) Wait for input $k^{\mathcal{S}} \in \mathbb{K}$ from $\mathcal{S}$ and $k^{\mathcal{R}} \in \mathbb{K}$ from $\mathcal{R}$ such that $k := k^{\mathcal{S}} + k^{\mathcal{R}} \in \mathbb{K}$.
3) For $i \in [n]$, compute $y_i := F_k(x_i)$.
4) Output $\{y_i\}_{i\in[n]}$ to $\mathcal{R}$.

Figure 6: Functionality of oblivious PRF with secret-shared inputs.

$\{\mathsf{Encode}(\{(k_1, v_1), \ldots, (k_n, v_n)\}) \mid v_i \leftarrow \mathcal{V}, i \in [n]\}$ is statistically indistinguishable from uniform distribution over $\mathcal{V}^m$.

Independence: For any $L := \{(k_i, v_i)_{i\in[n]}\} \in (\mathcal{K} \times \mathcal{V})^n$ with distinct keys such that $D \leftarrow \mathsf{Encode}(L)$ and $D \neq \perp$, it holds that for any $k \notin \{k_i\}_{i\in[n]}$, $\mathsf{Decode}(D, k)$ is statistically indistinguishable from uniform distribution over $\mathcal{V}$.

3.4. Secret shared OPRF

We introduce two functionalities of OPRF with secret shared output (so-OPRF) and OPRF with secret shared key and input (si-OPRF) in Figure 5 and Figure 6, respectively. We instantiate the above two variants of OPRF using MPC-friendly alternating-moduli (weak) PRF, which was first proposed by Boneh et al. [49] and later optimized in [46], [50], [51]. At a high level, this construction takes as input a key k and a value x, and multiplies them by various matrices modulo two different primes, e.g., 2 and 3. More concretely, the PRF construction in [46] is defined as $F(k, x) := \mathbf{B} \cdot_2 (\mathbf{A} \cdot_3 [k \circ_2 (\mathbf{G} \cdot_2 [x||1])])$, where $\mathbf{G} \in \mathbb{F}_2^{n\times(d+1)}$, $\mathbf{A} \in \mathbb{F}_3^{m\times n}$, $\mathbf{B} \in \mathbb{F}_2^{t\times m}$ are uniformly distributed, and $\cdot_p$, $\circ_p$ are multiplication and component-wise multiplication modulo p. In particular, we make use of the parameterization $d = \kappa, n = 4\kappa, m = 2\kappa, t = \kappa$ which implies $\mathbf{G}$ is an expanding matrix and $\mathbf{B}$ is a compressing matrix. The security of this construction relies on the assumption [49] that linear operations over two different moduli result in a highly non-linear and unpredictable function when viewed over $\mathbb{F}_2$ or $\mathbb{F}_3$.

3.5. Functionalities of Building Blocks

We present the functionalities of the private equality/interval test in Figure 7. Chakraborti et al. [21] introduced an efficient private interval test protocol based on prefix representation. Both the communication and computation complexities are $O(\log \delta)$. Besides, we introduce the functionalities $\mathcal{F}_{\mathsf{MUX}}$ and $\mathcal{F}_{\mathsf{ssPEQT}}$, which are deferred to Appendix B.1.

Functionality $\mathcal{F}_{\mathsf{Interval}}$

Parameters: Sender $\mathcal{S}$ and receiver $\mathcal{R}$. Threshold δ.

Protocol:

1) Wait for input $x^{\mathcal{S}}$ from $\mathcal{S}$ and $x^{\mathcal{R}}$ from $\mathcal{R}$.
2) Output 1 to $\mathcal{R}$ if $x^{\mathcal{S}} + x^{\mathcal{R}} \leq \delta$, otherwise output 0.

Figure 7: Functionality of private interval test.

3.6. Prefix Representation

Prefix representation, first proposed by Chakraborti et al. [21] and further studied and formalized in [12], [16], [18], [20], is used to improve the efficiency of fuzzy PSI. Let $x = x_\ell x_{\ell-1} \cdots x_1 \in \{0,1\}^\ell$ be a binary string. We first introduce the following notations:

- $\mathsf{Prefix}(x_\ell x_{\ell-1} \cdots x_1, k) = x_\ell x_{\ell-1} \cdots x_{k+1}$
- $\mathsf{AllPrefix}(x_\ell x_{\ell-1} \cdots x_1, k) = \{x_\ell x_{\ell-1} \cdots x_{j+1}\}_{j\in[0,k]}$
- $\mathsf{UpBound}(x_\ell x_{\ell-1} \cdots x_k) = x_\ell x_{\ell-1} \cdots x_k \| 11 \cdots 1$
- $\mathsf{LowBound}(x_\ell x_{\ell-1} \cdots x_k) = x_\ell x_{\ell-1} \cdots x_k \| 00 \cdots 0$
- $\mathsf{Interval}(x_\ell x_{\ell-1} \cdots x_k) = \{x_\ell \cdots x_k \| x^*\}_{x^* \in \{0,1\}^{k-1}}$

Previous works [16], [18] gave the definition of $\mathsf{Decompose}$ as following:

Definition 3. *Given an integer interval $[a-\delta, a+\delta]$, there is an algorithm* $\mathsf{Decompose}$ *to succinctly encode the interval into a list of prefixes $\{p_i\}_{i\in[\hat{\mu}]}$ such that:*

- $[a - \delta, a + \delta] = \bigcup_{i\in[\hat{\mu}]} \mathsf{Interval}(p_i)$
- *For any $i \in [\hat{\mu}]$, if a binary string p is a prefix of p_i, $\mathsf{Interval}(p_i) \nsubseteq [a - \delta, a + \delta]$*

The algorithm $\mathsf{Decompose}$ [16], [18] has a computation complexity $O(\log \delta)$ and the number of prefixes $\hat{\mu}$ is also $O(\log \delta)$. The length of prefixes p_i is at least $\ell - \mu'$ where μ' is the number of wildcards (i.e., non-determined bit strings) and $\mu' = O(\log \delta)$. Precisely, if δ is a power of 2, the number of prefixes $\hat{\mu}$ has a lower bound of $\lceil \log(2\delta + 1) \rceil$ and μ' has an upper bound of $\lfloor \log(2\delta + 1) \rfloor$ [18]. In this paper, we set our threshold δ to a power of 2 for simplicity. Unless otherwise specified, we denote $\hat{\mu} = 2 + \log \delta$ and $\mu' = 1 + \log \delta$ or $\log \delta$. For a binary string $x \in \{0,1\}^\ell$, there are $\mu' + 1$ prefixes of length at least $\ell - \mu'$ and we denote $\mu = \mu' + 1$.

4. OPPRF with Shared Outputs

We introduce a new variant of OPPRF, termed so-OPPRF, whose outputs are secret-shared between two parties. So-OPPRF is useful in scenarios where outputs cannot

Functionality $\mathcal{F}_{\text{so-OPPRF}}$

Parameters: Sender $\mathcal{S}$ and receiver $\mathcal{R}$ with input sizes m, n, respectively.

Protocol:

1) Wait for input $L = \{(q_j, z_j)\}_{j\in[m]} \subseteq \mathbb{U} \times \mathbb{F}$ from $\mathcal{S}$ and $X = \{x_i\}_{i\in[n]} \subseteq \mathbb{U}$ from $\mathcal{R}$.
2) Sample a random function $F' : \mathbb{U} \to \mathbb{F}$ such that $F'(q) = z$ for $(q, z) \in L$.
3) For $i \in [n]$, compute $y_i := F'(x_i)$ and sample $y_i^{\mathcal{S}}, y_i^{\mathcal{R}} \leftarrow \mathbb{F}$ such that $y_i^{\mathcal{S}} + y_i^{\mathcal{R}} = y_i \in \mathbb{F}$.
4) Output $\{y_i^{\mathcal{S}}\}_{i\in[n]}$ and $\mathcal{O}^{F'}$ to $\mathcal{S}$ and $\{y_i^{\mathcal{R}}\}_{i\in[n]}$ to $\mathcal{R}$.

Figure 8: Functionality of oblivious programmable PRF with secret-shared outputs.

Protocol $\Pi_{\text{so-OPPRF}}$

Parameters: Sender $\mathcal{S}$ and receiver $\mathcal{R}$ with input sizes m, n, respectively. PRF $F : \mathbb{K} \times \mathbb{U} \to \mathbb{F}$.

Input: $\mathcal{S}$ inputs $L = \{(q_j, z_j)\}_{j\in[m]} \subseteq \mathbb{U} \times \mathbb{F}$. $\mathcal{R}$ inputs $X = \{x_i\}_{i\in[n]} \subseteq \mathbb{U}$.

Protocol:

1) $\mathcal{S}$ samples $k \leftarrow \mathbb{K}$. $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\text{so-OPRF}}$, where $\mathcal{S}$ inputs k and R inputs $X = \{x_i\}_{i\in[n]}$. $\mathcal{S}$ receives $\{f_i^{\mathcal{S}}\}_{i\in[n]}$, and $\mathcal{R}$ receives $\{f_i^{\mathcal{R}}\}_{i\in[n]}$, where $f_i := F_k(x_i) \in \mathbb{F}$.
2) $\mathcal{S}$ computes PRF values $F_k(q_j)$ for $j \in [m]$ and computes the OKVS encoding $D := \mathsf{OKVS.Encode}(L')$, where $L' := \{(q_j, z_j - F_k(q_j))\}_{j\in[m]}$.
3) $\mathcal{S}$ sends D to $\mathcal{R}$ and $\mathcal{R}$ computes $d_i := \mathsf{OKVS.Decode}(D, x_i)$ for $i \in [n]$.
4) $\mathcal{S}$ defines the function $F'(q) := F_k(q) + \mathsf{OKVS.Decode}(D, q)$.
5) $\mathcal{S}$ outputs $\{y_i^{\mathcal{S}} := f_i^{\mathcal{S}}\}_{i\in[n]}$ and $\mathcal{O}^{F'}$, and $\mathcal{R}$ outputs $\{y_i^{\mathcal{R}} := f_i^{\mathcal{R}} + d_i\}_{i\in[n]}$.

Figure 9: Protocol of oblivious programmable PRF with secret-shared outputs.

be revealed to either party individually, which may be of independent interest. As we have presented our idea in Section 2.2, here we illustrate the ideal functionality in Figure 8 and the detailed protocol in Figure 9.

We show the protocol's security in Theorem 1, and the security proof is deferred to Appendix A.2.

Theorem 1. *The protocol $\Pi_{\text{so-OPPRF}}$ in Figure 9 realizes the functionality $\mathcal{F}_{\text{so-OPPRF}}$ in Figure 8 against semi-honest adversaries in the ($\mathcal{F}_{\text{so-OPRF}}$)-hybrid model if* OKVS *satisfies the correctness, double obliviousness, and independence properties defined in Definition 2.*

5. Efficient Fuzzy Mapping

Fuzzy mapping (FMap), introduced by Gao et al. [14], enables two parties to assign identifiers to their respective set elements. If an element from the sender $\mathcal{S}$ and an element from the receiver $\mathcal{R}$ are sufficiently close, they will be mapped to the same identifier. We recall the formal definition from Gao et al. [14] as follows.

Procedure LocalMap

Parameters: Distance threshold is δ. Prefix parameter $\hat{\mu} = 2 + \log \delta$.

Input: A set $Q = \{q_j\}_{j\in[m]} \in \mathbb{U}^{d\times m}$.

Protocol:

1) Initialize $\mathsf{List} := \emptyset$ and $\mathsf{interval}_k := \emptyset$ for $k \in [d]$.
2) For each $j \in [m]$ and each $k \in [d]$:
   - Sample $r_{j,k}$ and define $U_{j,k} := [q_{j,k} - \delta, q_{j,k} + \delta]$.
   - For each $(U, r) \in \mathsf{interval}_k$ if $U_{j,k} \cap U \neq \emptyset$, remove (U, r) from $\mathsf{interval}_k$ and update $U_{j,k} = U_{j,k} \cup U$.
   - Set $\mathsf{interval}_k := \mathsf{interval}_k \cup \{(U_{j,k}, r_{j,k})\}$.
3) For each $j \in [m]$, set $\mathsf{pid}_{q_j} := \sum_{k=1}^{d} r_k$, where r_k satisfies that $q_{j,k} \in U_k$ and $(U_k, r_k) \in \mathsf{interval}_k$.
4) For each $k \in [d]$ and each $(U, r) \in \mathsf{interval}_k$:
   - Without prefix optimization: For each $x \in U$, set $\mathsf{List} := \mathsf{List} \cup \{(k\|x, r)\}$.
   - With prefix optimization: Partition interval U into consecutive, disjoint sub-intervals $\{U_j\}_{j\in[\epsilon]}$, such that $|U_j| = 2\delta + 1$ for $j \in [\epsilon - 1]$ and $|U_\epsilon| \leq 2\delta + 1$. For each $j \in [\epsilon]$, compute $\{x_{j,h}\}_{h\in[\hat{\mu}]} := \mathsf{Decompose}(U_j)$ and set $\mathsf{List} := \mathsf{List} \cup \{(k\|x_{j,h}, 0\|r)\}_{h\in[\hat{\mu}]}$.
5) Pad List with dummy items so that its size equals
   - Without prefix optimization: $md(2\delta + 1)$.
   - With prefix optimization: $md(\log \delta + 2)$.
6) Output $\{\mathsf{pid}_{q_j}\}_{j\in[m]}$ and List.

Figure 10: Procedure of local mapping and its variant with prefix optimization.

Definition 4 (Fuzzy mapping [14])**.** *A two-party protocol Π_{FMap}, where $\mathcal{S}$ inputs $Q = \{q_j\}_{j\in[m]} \in \mathbb{U}^{d\times m}$ and learns $\{\mathsf{ID}_{q_j}\}_{j\in[m]} \in \mathbb{F}^m$ and $\mathcal{R}$ inputs $W = \{w_i\}_{i\in[n]} \in \mathbb{U}^{d\times n}$ and learns $\{\mathsf{ID}_{w_i}\}_{i\in[n]} \in \mathbb{F}^n$, is a secure fuzzy mapping protocol for distance metric* dist *and threshold δ against semi-honest adversaries, if and only if it satisfies:*

1) ***Correctness:*** *For any $w \in W$ and $q \in Q$, if $\mathsf{dist}(w, q) \leq \delta$, then $\mathsf{ID}_q = \mathsf{ID}_w$.*
2) ***Distinctiveness:*** *For any $w_i, w_{i'} \in W$ and $i \neq i'$, $\mathsf{Pr}[\mathsf{ID}_{w_i} = \mathsf{ID}_{w_{i'}}] \leq \mathsf{negl}(\lambda)$.*
3) ***Security:*** *Considering a corrupted sender $\mathcal{S}$, for any $Q \in \mathbb{U}^{d\times m}$ and any $W, W' \in \mathbb{U}^{d\times n}$, it holds that $\mathsf{view}_{\mathcal{S}}^{\Pi}(Q, W) \approx_c \mathsf{view}_{\mathcal{S}}^{\Pi}(Q, W')$. Similarly, considering a corrupted receiver $\mathcal{R}$, for any $W \in \mathbb{U}^{d\times n}$ and any $Q, Q' \in \mathbb{U}^{d\times m}$, it holds that $\mathsf{view}_{\mathcal{R}}^{\Pi}(Q, W) \approx_c \mathsf{view}_{\mathcal{R}}^{\Pi}(Q', W)$.*

As we have presented our idea in Section 2.3, here we give the construction of the fuzzy mapping protocol for L_∞ distance in Figure 11 and its sub-procedure local mapping in Figure 10. We note that for any w, q, the fact that $\mathsf{dist}_\infty(q, w) \leq \mathsf{dist}_p(q, w)$ holds. Therefore, we will use fuzzy mapping for L_∞ distance in fuzzy PSI protocols for both L_∞ and L_p distance, since fuzzy mapping can tolerate false positives.

Protocol Π_{FMap}

Input: $\mathcal{S}$ inputs $Q = \{q_j\}_{j\in[m]} \in \mathbb{U}^{d\times m}$ and $\mathcal{R}$ inputs $W = \{w_i\}_{i\in[n]} \in \mathbb{U}^{d\times n}$.

Protocol:

1) $\mathcal{S}$ runs $\mathsf{LocalMap}(Q)$ and obtains $(\{\mathsf{pid}_{q_j}\}_{j\in[m]}, \mathsf{List}^{\mathcal{S}})$. $\mathcal{R}$ runs $\mathsf{LocalMap}(W)$ and obtains $(\{\mathsf{pid}_{w_i}\}_{i\in[n]}, \mathsf{List}^{\mathcal{R}})$.
2) $\mathcal{S}$ and $\mathcal{R}$ sample secret shares $k^{\mathcal{S}}, k^{\mathcal{R}}$ of the PRF key, respectively.
3) $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$, where $\mathcal{S}$ as the sender inputs $\mathsf{List}^{\mathcal{S}}$ and $\mathcal{R}$ as the receiver inputs $\{k\|w_{i,k}\}_{i\in[n],k\in[d]}$. $\mathcal{S}$ receives $\{r^{\mathcal{S}}_{w_i,k}\}_{i\in[n],k\in[d]}$ and $\mathcal{R}$ receives $\{r^{\mathcal{R}}_{w_i,k}\}_{i\in[n],k\in[d]}$.
4) For $i \in [n]$, $\mathcal{S}$ computes $r^{\mathcal{S}}_{w_i} := \sum_{k\in[d]} r^{\mathcal{S}}_{w_i,k}$, and $\mathcal{R}$ computes $r^{\mathcal{R}}_{w_i} := \sum_{k\in[d]} r^{\mathcal{R}}_{w_i,k}$.
5) $\mathcal{S}$ and $\mathcal{R}$ invoke $\mathcal{F}_{\mathsf{si\text{-}OPRF}}$, where $\mathcal{S}$ as the sender inputs $(k^{\mathcal{S}}, \{r^{\mathcal{S}}_{w_i}\}_{i\in[n]})$, and $\mathcal{R}$ as the receiver inputs $(k^{\mathcal{R}}, \{r^{\mathcal{R}}_{w_i} + \mathsf{pid}_{w_i}\}_{i\in[n]})$ and learns $\{\mathsf{ID}_{w_i}\}_{i\in[n]}$.
6) Symmetrically, $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$, where $\mathcal{R}$ as the sender inputs $\mathsf{List}^{\mathcal{R}}$ and $\mathcal{S}$ as the receiver inputs $\{k\|q_{j,k}\}_{j\in[m],k\in[d]}$. $\mathcal{S}$ receives $\{r^{\mathcal{S}}_{q_j,k}\}_{j\in[m],k\in[d]}$ and $\mathcal{R}$ receives $\{r^{\mathcal{R}}_{q_j,k}\}_{j\in[m],k\in[d]}$.
7) For $j \in [m]$, $\mathcal{S}$ computes $r^{\mathcal{S}}_{q_j} := \sum_{k\in[d]} r^{\mathcal{S}}_{q_j,k}$, and $\mathcal{R}$ computes $r^{\mathcal{R}}_{q_j} := \sum_{k\in[d]} r^{\mathcal{R}}_{q_j,k}$.
8) $\mathcal{S}$ and $\mathcal{R}$ invoke $\mathcal{F}_{\mathsf{si\text{-}OPRF}}$, where $\mathcal{R}$ as the sender inputs $(k^{\mathcal{R}}, \{r^{\mathcal{R}}_{q_j}\}_{j\in[m]})$, and $\mathcal{S}$ as the receiver inputs $(k^{\mathcal{S}}, \{r^{\mathcal{S}}_{q_j} + \mathsf{pid}_{q_j}\}_{j\in[m]})$ and learns $\{\mathsf{ID}_{q_j}\}_{j\in[m]} \in \mathbb{F}_{2^\ell}$.

Figure 11: Protocol of fuzzy mapping for L_∞ distance.

We show that the above protocol satisfies the fuzzy mapping definition in Theorem 2.

Theorem 2. *The Π_{FMap} protocol in Figure 11 satisfies the correctness, distinctiveness, and security properties defined in Definition 4 for L_∞ distance, if both parties' sets satisfy the disjoint projection assumption.*

Proof. **Correctness.** For any $w_i \in W$ and $q_j \in Q$, if $\mathsf{dist}(w_i, q_j) \le \delta$, then for any $k \in [d]$, $|w_{i,k} - q_{j,k}| \le \delta$ always holds. Therefore, all these $w_{i,k}$ are assigned in $\mathsf{List}^{\mathcal{S}}$ and $r^{\mathcal{S}}_{w_{i,k}} + r^{\mathcal{R}}_{w_{i,k}} = \mathsf{List}^{\mathcal{S}}[k\|w_{i,k}]$ according to the functionality $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$. Moreover, the $\mathsf{LocalMap}$ procedure ensures that the $2\delta + 1$ points on dimension k centered at $q_{j,k}$ all have the same assignment, hence $\sum_{k\in[d]}(r^{\mathcal{S}}_{w_{i,k}} + r^{\mathcal{R}}_{w_{i,k}}) = \mathsf{pid}_{q_j}$ holds. Similarly, it holds that $\sum_{k\in[d]}(r^{\mathcal{S}}_{q_{j,k}} + r^{\mathcal{R}}_{q_{j,k}}) = \mathsf{pid}_{w_i}$. The identifiers are computed as follows $\mathsf{ID}_{q_j} := \mathsf{PRF}_k(\sum_{k\in[d]}(r^{\mathcal{S}}_{q_{j,k}} + r^{\mathcal{R}}_{q_{j,k}}) + \mathsf{pid}_{q_j})$ and $\mathsf{ID}_{w_i} := \mathsf{PRF}_k(\sum_{k\in[d]}(r^{\mathcal{S}}_{w_{i,k}} + r^{\mathcal{R}}_{w_{i,k}}) + \mathsf{pid}_{w_i})$. Since the PRF key is the same for both PRF evaluations on q_j and w_i, then $\mathsf{ID}_{q_j} = \mathsf{ID}_{w_i}$ when $\mathsf{dist}(w_i, q_j) \le \delta$.

Distinctiveness. Since the $\mathcal{R}$'s set W satisfies the disjoint projection assumption, without loss of generality, for each w_i, assume the interval of $[w_{i,k_i} - \delta, w_{i,k_i} + \delta]$ on dimension k_i is disjoint from the length-$(2\delta + 1)$ intervals of other elements of W. Thus, the assignment $\mathsf{List}^{\mathcal{R}}[k_i\|w_{i,k_i}]$ is a uniformly random value and is independent of any other assignments. Then, we have that $\mathsf{ID}_{w_i} := \mathsf{PRF}(k, \mathsf{List}^{\mathcal{R}}[k_i\|w_{i,k_i}] + \ldots)$. According to the functionality of si-OPRF, ID_{w_i} is computationally indistinguishable from a uniformly random value. Therefore, a union bound shows that the probability of that there exists $i, i' \in [n]$ such that $i \ne i'$ and $\mathsf{ID}_{w_i} = \mathsf{ID}_{w_{i'}} \in \mathbb{F}$ is at most $n^2/|\mathbb{F}|$. With $|\mathbb{F}| \ge n^2 \cdot 2^\lambda$, the probability $n^2/|\mathbb{F}| \le 1/2^\lambda$ is negligible.

Security. Since the protocol is symmetric, we only consider the corrupted sender $\mathcal{S}$. Let $\mathsf{ID}_Q(W)$ denote the $\mathcal{S}$'s output from the real-world protocol where $\mathcal{S}$ inputs Q and $\mathcal{R}$ inputs W. We show the simulator $\mathsf{Sim}^{\mathsf{FMap}}_{\mathcal{S}}(Q, \mathsf{ID}_Q(W))$:

1) $\mathsf{Sim}^{\mathsf{FMap}}_{\mathcal{S}}$ runs $\mathsf{LocalMap}(Q)$ and records the sampled randomness.
2) $\mathsf{Sim}^{\mathsf{FMap}}_{\mathcal{S}}$ randomly samples the PRF key share $k^{\mathcal{S}}$.
3) $\mathsf{Sim}^{\mathsf{FMap}}_{\mathcal{S}}$ randomly samples $\{r^{\mathcal{S}}_{w_i,k}\}_{i\in[n],k\in[d]}$.
4) $\mathsf{Sim}^{\mathsf{FMap}}_{\mathcal{S}}$ randomly samples $\{r^{\mathcal{S}}_{q_j,k}\}_{j\in[m],k\in[d]}$.
5) $\mathsf{Sim}^{\mathsf{FMap}}_{\mathcal{S}}$ appends the above values and $(Q, \mathsf{ID}_Q(W))$ into the simulated view.

We show that the output by $\mathsf{Sim}^{\mathsf{FMap}}_{\mathcal{S}}$ is indistinguishable from the real protocol. The only difference is $\{r^{\mathcal{S}}_{w_i,k}\}_{i\in[n],k\in[d]}$ and $\{r^{\mathcal{S}}_{q_j,k}\}_{j\in[m],k\in[d]}$. They are random secret shares in the real protocol according to the so-OPPRF functionality, while they are randomly sampled in the simulated view with the same distribution. Therefore, for any Q, W, it holds that $\mathsf{view}^{\Pi}_{\mathcal{S}}(Q, W) \approx_c \mathsf{Sim}^{\mathsf{FMap}}_{\mathcal{S}}(Q, \mathsf{ID}_Q(W))$.

Moreover, since the $\mathcal{S}$'s set Q satisfies the disjoint projection assumption, as shown in the above analysis of distinctiveness, ID_Q is computationally indistinguishable from uniformly random values for any W. Then, for any Q, W, it holds that $\mathsf{Sim}^{\mathsf{FMap}}_{\mathcal{S}}(Q, \mathsf{ID}_Q(W)) \approx_c \mathsf{Sim}^{\mathsf{FMap}}_{\mathcal{S}}(Q, R \leftarrow \mathbb{F}^m)$.

Protocol $\Pi^{L_\infty}_{\mathsf{FPSI}}$

Parameters: Distance threshold δ.

Input: $\mathcal{S}$ inputs $Q = \{q_j\}_{j\in[m]} \in \mathbb{U}^{d\times m}$ and $\mathcal{R}$ inputs $W = \{w_i\}_{i\in[n]} \in \mathbb{U}^{d\times n}$.

Protocol:

1) $\mathcal{S}$ and $\mathcal{R}$ invoke sub-protocol Π_{FMap}, where $\mathcal{S}$ inputs Q and $\mathcal{R}$ inputs W. $\mathcal{S}$ obtains $\{\mathsf{ID}_{q_j}\}_{j\in[m]}$ and $\mathcal{R}$ obtains $\{\mathsf{ID}_{w_i}\}_{i\in[n]}$.
2) $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$, where $\mathcal{R}$ as the sender inputs $\{(\mathsf{ID}_{w_i}\|k\|(w_{i,k} + t), 0)\}_{t\in[-\delta,\delta],i\in[n],k\in[d]}$ and $\mathcal{S}$ as the receiver inputs $\{\mathsf{ID}_{q_j}\|k\|q_{j,k}\}_{j\in[m],k\in[d]}$. $\mathcal{S}$ receives $\{r^{\mathcal{S}}_{j,k}\}_{j\in[m],k\in[d]}$ and $\mathcal{R}$ receives $\{r^{\mathcal{R}}_{j,k}\}_{j\in[m],k\in[d]}$.
3) For $j \in [m]$, $\mathcal{S}$ computes $r^{\mathcal{S}}_j := \sum_{k\in[d]} r^{\mathcal{S}}_{j,k}$, and $\mathcal{R}$ computes $r^{\mathcal{R}}_j := \sum_{k\in[d]} r^{\mathcal{R}}_{j,k}$.
4) $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{PEQT}}$, where $\mathcal{S}$ inputs $\{r^{\mathcal{S}}_j\}_{j\in[m]}$ and $\mathcal{R}$ inputs $\{-r^{\mathcal{R}}_j\}_{j\in[m]}$. $\mathcal{R}$ receives $\{b_j\}_{j\in[m]}$, where $b_j := \mathbf{1}\{r^{\mathcal{S}}_j + r^{\mathcal{R}}_j = 0\}$.
5) $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{OT}}$, where $\mathcal{S}$ inputs $\{(\perp, q_j)\}_{j\in[m]}$, and $\mathcal{R}$ inputs $\{b_j\}_{j\in[m]}$ and receives $\{z_j\}_{j\in[m]}$.
6) $\mathcal{R}$ outputs $Z := \{z_j \mid b_j = 1 \text{ for } j \in [m]\}$.

Figure 12: Protocol of fuzzy PSI for L_∞ distance.

Therefore, we have $\mathsf{view}_{\mathcal{S}}^{\Pi}(Q,W) \approx_c \mathsf{Sim}_{\mathcal{S}}^{\mathsf{FMap}}(Q, R \leftarrow \mathbb{F}^m) \approx_c \mathsf{view}_{\mathcal{S}}^{\Pi}(Q,W')$. This completes the proof. □

6. Fuzzy PSI

6.1. Fuzzy PSI for L_∞ Distance

We have presented our idea in Section 2.4 and the detailed fuzzy PSI protocol for L_∞ distance is illustrated in Figure 12. We show the protocol's correctness as follows.

Proof. **Correctness.** For each $q_j \in Q$, if there exists $w_i \in W$ such that $\mathsf{dist}(q_j, w_i) \le \delta$, the correctness of the fuzzy mapping in Theorem 2 guarantees that $\mathsf{ID}_{q_j} = \mathsf{ID}_{w_i}$. Consequently, by the correctness of the functionalities $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$, $\mathcal{F}_{\mathsf{PEQT}}$, and $\mathcal{F}_{\mathsf{OT}}$, the receiver correctly obtains $b_j = 1$ and the corresponding element q_j.

On the other hand, for each $q_j \in Q$, if $\mathsf{dist}(q_j, w_i) > \delta$ for all $w_i \in W$, two cases arise due to the false positives introduced by the fuzzy mapping. (1) The first case is $\mathsf{ID}_{q_j} \neq \mathsf{ID}_{w_i}$ for any $w_i \in W$. According to the correctness of functionality $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$, the secret-shared outputs $r_{j,k} \in \mathbb{F}$ for $k \in [d]$ are uniformly random. (2) The second case is $\mathsf{ID}_{q_j} = \mathsf{ID}_{w_i}$ for some $w_i \in W$ and there exists some k^* such that $|q_{j,k^*} - w_{i,k^*}| > \delta$. According to the correctness of functionality $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$, the secret-shared output $r_{j,k^*} \in \mathbb{F}$ is uniformly random. In both cases, $r_j := \sum_{k\in[d]} r_{j,k} \in \mathbb{F}$ is uniformly random. According to the correctness of functionality $\mathcal{F}_{\mathsf{PEQT}}$, a union bound shows that the probability of there existing $j \in [m]$ such that $b_j = 1$ is at most $m/|\mathbb{F}|$. By setting $|\mathbb{F}| \ge m \cdot 2^\lambda$, the probability $m/|\mathbb{F}| \le 1/2^\lambda$ is negligible. The correctness of $\mathcal{F}_{\mathsf{OT}}$ ensures the receiver learns nothing about q_j. □

We show the protocol's security in Theorem 3 and the security proof is deferred to Appendix A.3.

Theorem 3. *The protocol $\Pi_{\mathsf{FPSI}}^{L_\infty}$ in Figure 12 realizes the functionality $\mathcal{F}_{\mathsf{FPSI}}$ for L_∞ distance in Figure 4 against semi-honest adversaries in the $(\mathcal{F}_{\mathsf{so\text{-}OPPRF}}, \mathcal{F}_{\mathsf{PEQT}}, \mathcal{F}_{\mathsf{OT}})$-hybrid model.*

6.2. Fuzzy PSI for L_p Distance

We present the construction of fuzzy PSI for L_p distance, which closely resembles that of L_∞ distance. Given the fact that $\mathsf{dist}_p(q,w) \le \delta$ implies $\mathsf{dist}_\infty(q,w) \le \delta$, this protocol reuses the fuzzy mapping protocol for L_∞ distance to generate IDs, with the only modification in the refined filtering.

Specifically, two parties invoke functionality $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$, where the receiver sets the key-value pairs as $\{(\mathsf{ID}_{w_i} \| k \| (w_{i,k}+t), |t|^p)\}_{i\in[n],k\in[d],t\in[-\delta,\delta]}$, and the sender prepares $\{\mathsf{ID}_{q_j} \| k \| q_{j,k}\}_{j\in[m],k\in[d]}$. Then, two parties get the secret sharing of the distance $|w_{i,k} - q_{j,k}|^p$ for each dimension $k \in [d]$ or a random value. The following process computes the distance $\mathsf{dist}_p(q_j, w_i)^p = \sum_{k\in[d]} |w_{i,k} - q_{j,k}|^p$. We note that the protocol includes a Boolean-to-arithmetic conversion (B2A) protocol, since the outputs of our so-OPPRF protocol reside in a binary field $\mathbb{F}_{2^\ell}$ that does not support computing arithmetic operations of the L_p distance. B2A converts the outputs of so-OPPRF into arithmetic shares, which are suitable for subsequent computation.

Protocol $\Pi_{\mathsf{FPSI}}^{L_p}$

Parameters: Distance threshold δ.

Input: $\mathcal{S}$ inputs $Q = \{q_j\}_{j\in[m]} \in \mathbb{U}^{d\times m}$ and $\mathcal{R}$ inputs $W = \{w_i\}_{i\in[n]} \in \mathbb{U}^{d\times n}$.

Protocol:

1) $\mathcal{S}$ and $\mathcal{R}$ invoke sub-protocol Π_{FMap}, where $\mathcal{S}$ inputs Q and $\mathcal{R}$ inputs W. $\mathcal{S}$ obtains $\{\mathsf{ID}_{q_j}\}_{j\in[m]}$ and $\mathcal{R}$ obtains $\{\mathsf{ID}_{w_i}\}_{i\in[n]}$.
2) $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$, where $\mathcal{R}$ as the sender inputs $\{(\mathsf{ID}_{w_i} \| k \| (w_{i,k}+t), |t|^p)\}_{t\in[-\delta,\delta],i\in[n],k\in[d]}$ and $\mathcal{S}$ as the receiver inputs $\{\mathsf{ID}_{q_j} \| k \| q_{j,k}\}_{j\in[m],k\in[d]}$. $\mathcal{S}$ receives $\{r_{j,k}^{\mathcal{S}}\}_{j\in[m],k\in[d]}$ and $\mathcal{R}$ receives $\{r_{j,k}^{\mathcal{R}}\}_{j\in[m],k\in[d]}$.
3) (Optional) $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{B2A}}$, where $\mathcal{S}$ inputs $\{r_{j,k}^{\mathcal{S}}\}_{j\in[m],k\in[d]}$ and $\mathcal{R}$ inputs $\{r_{j,k}^{\mathcal{R}}\}_{j\in[m],k\in[d]}$. $\mathcal{S}$ receives $\{d_{j,k}^{\mathcal{S}}\}_{j\in[m],k\in[d]}$ and $\mathcal{R}$ receives $\{d_{j,k}^{\mathcal{R}}\}_{j\in[m],k\in[d]}$.
4) For $j \in [m]$, $\mathcal{S}$ computes $d_j^{\mathcal{S}} := \sum_{k\in[d]} d_{j,k}^{\mathcal{S}}$, and $\mathcal{R}$ computes $d_j^{\mathcal{R}} := \sum_{k\in[d]} d_{j,k}^{\mathcal{R}}$.
5) $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{Interval}}$, where $\mathcal{S}$ inputs $\{d_j^{\mathcal{S}}\}_{j\in[m]}$ and $\mathcal{R}$ inputs $\{d_j^{\mathcal{R}}\}_{j\in[m]}$. $\mathcal{R}$ receives $\{b_j\}_{j\in[m]}$, where $b_j := \mathbf{1}\{d_j^{\mathcal{S}} + d_j^{\mathcal{R}} \le \delta^p\}$.
6) $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{OT}}$, where $\mathcal{S}$ inputs $\{(\perp, q_j)\}_{j\in[m]}$, and $\mathcal{R}$ inputs $\{b_j\}_{j\in[m]}$ and receives $\{z_j\}_{j\in[m]}$.
7) $\mathcal{R}$ outputs $Z := \{z_j \mid b_j = 1 \text{ for } j \in [m]\}$.

Figure 13: Protocol of fuzzy PSI for L_p distance.

In addition, unlike the private equality test used in protocol $\Pi_{\mathsf{FPSI}}^{L_\infty}$, here we invoke the private interval test protocol [21] to determine whether the distance falls below the threshold δ^p. The detailed construction of the protocol is illustrated in Figure 13. We show the correctness as follows.

Proof. **Correctness.** For each $q_j \in Q$, if there exists $w_i \in W$ such that $\mathsf{dist}_p(q_j, w_i) \le \delta$, then by the correctness of fuzzy mapping for L_∞ distance in Theorem 2 and the fact that $\mathsf{dist}_\infty(q_j, w_i) \le \mathsf{dist}_p(q_j, w_i) \le \delta$ for any q_j, w_i, it holds $\mathsf{ID}_{q_j} = \mathsf{ID}_{w_i}$. Then, by the correctness of functionalities $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$ and $\mathcal{F}_{\mathsf{B2A}}$, it holds $d_{j,k}^{\mathcal{S}} + d_{j,k}^{\mathcal{R}} = |w_{i,k} - q_{j,k}|^p$, and hence $d_j^{\mathcal{S}} + d_j^{\mathcal{R}} = \sum_{k\in[d]} (d_{j,k}^{\mathcal{S}} + d_{j,k}^{\mathcal{R}}) = \sum_{k\in[d]} |w_{i,k} - q_{j,k}|^p \le \delta^p$. Therefore, by the correctness of functionalities $\mathcal{F}_{\mathsf{Interval}}$ and $\mathcal{F}_{\mathsf{OT}}$, the receiver correctly obtains $b_j = 1$ and the corresponding element q_j.

On the other hand, for each $q_j \in Q$, if $\mathsf{dist}_p(q_j, w_i) > \delta$ in L_p distance for all $w_i \in W$, there are three cases due to the false positive of fuzzy mapping. (1) The first case is $\mathsf{ID}_{q_j} \neq \mathsf{ID}_{w_i}$ for any $w_i \in W$. According to the correctness of functionality $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$, the output $r_{j,k}^{\mathcal{S}} + r_{j,k}^{\mathcal{R}} \in \mathbb{F}$ is uniformly random for $k \in [d]$. Therefore, $r_j^{\mathcal{S}} + r_j^{R} := \sum_{k\in[d]} (r_{j,k}^{\mathcal{S}} + r_{j,k}^{\mathcal{R}}) \in \mathbb{F}$ is uniformly random. (2) The

Protocol Π_{EQSel}

Parameters: Ideal functionality $\mathcal{F}_{\mathsf{MUX}}$ and $\mathcal{F}_{\mathsf{ssPEQT}}$.
Input: $\mathcal{S}$ inputs $\{(e_i^{\mathcal{S}}, v_i^{\mathcal{S}})\}_{i\in[h]}$ and $\mathcal{R}$ inputs $\{(e_i^{\mathcal{R}}, v_i^{\mathcal{R}})\}_{i\in[h]}$.
Protocol:

1) $\mathcal{S}$ and $\mathcal{R}$ invoke $\mathcal{F}_{\mathsf{ssPEQT}}$, where $\mathcal{S}$ inputs $\{e_i^{\mathcal{S}}\}_{i\in[h]}$ and $\mathcal{R}$ inputs $\{e_i^{\mathcal{R}}\}_{i\in[h]}$. $\mathcal{S}$ receives $\{b_i^{\mathcal{S}}\}_{i\in[h]} \in \{0,1\}^h$, and $\mathcal{R}$ receives $\{b_i^{\mathcal{R}}\}_{i\in[h]} \in \{0,1\}^h$.
2) $\mathcal{S}$ and $\mathcal{R}$ invoke $\mathcal{F}_{\mathsf{MUX}}$, where $\mathcal{S}$ inputs $\{(b_i^{\mathcal{S}}, v_i^{\mathcal{S}})\}_{i\in[h]}$ and $\mathcal{R}$ inputs $\{(b_i^{\mathcal{R}}, v_i^{\mathcal{R}})\}_{i\in[h]}$. $\mathcal{S}$ receives $\{t_i^{\mathcal{S}}\}_{i\in[h]}$ and $\mathcal{R}$ receives $\{t_i^{\mathcal{R}}\}_{i\in[h]}$.
3) $\mathcal{S}$ computes $b^{\mathcal{S}} := \bigoplus_{i\in[h]} b_i^{\mathcal{S}}$, $t^{\mathcal{S}} := \sum_{i\in[h]} t_i^{\mathcal{S}}$ and $\mathcal{R}$ computes $b^{\mathcal{R}} := \bigoplus_{i\in[h]} b_i^{\mathcal{R}}$, $t^{\mathcal{R}} := \sum_{i\in[h]} t_i^{\mathcal{R}}$.
4) $\mathcal{S}$ randomly samples $r^{\mathcal{S}}$ and $\mathcal{R}$ randomly samples $r^{\mathcal{R}}$.
5) $\mathcal{S}$ and $\mathcal{R}$ invoke $\mathcal{F}_{\mathsf{MUX}}$, where $\mathcal{S}$ inputs $(b^{\mathcal{S}}, t^{\mathcal{S}} - r^{\mathcal{S}})$ and $\mathcal{R}$ inputs $(b^{\mathcal{R}}, t^{\mathcal{R}} - r^{\mathcal{R}})$. $\mathcal{S}$ receives $m^{\mathcal{S}}$ and $\mathcal{R}$ receives $m^{\mathcal{R}}$.
6) $\mathcal{S}$ outputs $z^{\mathcal{S}} := m^{\mathcal{S}} + r^{\mathcal{S}}$. $\mathcal{R}$ outputs $z^{\mathcal{R}} := m^{\mathcal{R}} + r^{\mathcal{R}}$.

Figure 14: Protocol of random equality-conditional selection.

second case is $\mathsf{ID}_{q_j} = \mathsf{ID}_{w_i}$ but $\mathsf{dist}_\infty(q_j, w_i) > \delta$. There exists $k^* \in [d]$ such that $|q_{j,k^*} - w_{i,k^*}| > \delta$. Then, according to the correctness of functionality $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$, the output $r^{\mathcal{S}}_{j,k^*} + r^{\mathcal{R}}_{j,k^*} \in \mathbb{F}$ is uniformly random. Similarly, $r_j^S + r_j^R := \sum_{k\in[d]}(r^{\mathcal{S}}_{j,k} + r^{\mathcal{R}}_{j,k}) \in \mathbb{F}$ is uniformly random. As a result, in the first two cases, the probability of $r_j^S + r_j^R \le \delta$ is at most $\delta^p/|\mathbb{F}|$. By the correctness of functionality $\mathcal{F}_{\mathsf{interval}}$, for any $j \in [m]$, a union bound shows that the probability of there existing $j \in [m]$ such that $b_j = 1$ is at most $m\delta^p/|\mathbb{F}|$. By setting $|\mathbb{F}| \ge m\delta^p 2^\lambda$, the probability $m\delta^p/|\mathbb{F}| \le 1/2^\lambda$ is negligible. The correctness of functionality $\mathcal{F}_{\mathsf{OT}}$ ensures the receiver learns nothing about q_j. (3) The third case is $\mathsf{ID}_{q_j} = \mathsf{ID}_{w_i}$ and $\mathsf{dist}_\infty(q_j, w_i) \le \delta$ for some $w_i \in W$. By the correctness of functionalities $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$ and $\mathcal{F}_{\mathsf{B2A}}$, $d^{\mathcal{S}}_{j,k} + d^{\mathcal{R}}_{j,k} = |w_{i,k} - q_{j,k}|^p$, it holds $d_j^{\mathcal{S}} + d_j^{\mathcal{R}} = \sum_{k\in[d]}(d^{\mathcal{S}}_{j,k} + d^{\mathcal{R}}_{j,k}) = \sum_{k\in[d]}|w_{i,k} - q_{j,k}|^p = \mathsf{dist}_p(q_j, w_i)^p > \delta^p$. The correctness of $\mathcal{F}_{\mathsf{Interval}}$ and $\mathcal{F}_{\mathsf{OT}}$ ensure the receiver learns nothing about q_j. □

We show the protocol's security in Theorem 4, and the security proof is deferred to Appendix A.4.

Theorem 4. *The protocol $\Pi^{L_p}_{\mathsf{FPSI}}$ in Figure 13 realizes the functionality $\mathcal{F}_{\mathsf{FPSI}}$ for L_p distance in Figure 4 against semi-honest adversaries in the $(\mathcal{F}_{\mathsf{so\text{-}OPPRF}}, \mathcal{F}_{\mathsf{B2A}}, \mathcal{F}_{\mathsf{Interval}}, \mathcal{F}_{\mathsf{OT}})$-hybrid model.*

7. Fuzzy PSI with Prefix Optimization

In this section, we present how to optimize our fuzzy PSI framework using prefix techniques [12], [16], [18] for the large distance threshold δ. In the previous section, our fuzzy PSI achieves linear complexity with respect to δ in both communication and computation. With prefix optimizations, we reduce the complexity to $O(\log \delta)$.

Protocol $\Pi_{\mathsf{FMap\text{-}Prefix}}$

Parameters: Threshold δ. Prefix parameter $\mu' = 1 + \log \delta$ and $\mu = 2 + \log \delta$.
Input: $\mathcal{S}$ inputs $Q = \{q_j\}_{j\in[m]} \in \mathbb{U}^{d\times m}$ and $\mathcal{R}$ inputs $W = \{w_i\}_{i\in[n]} \in \mathbb{U}^{d\times n}$.
Protocol:

1) $\mathcal{S}$ runs $\mathsf{LocalMap}^{\mathsf{Prefix}}(Q)$ and obtains $(\{\mathsf{pid}_{q_j}\}_{j\in[m]}, \mathsf{List}^{\mathcal{S}})$. $\mathcal{R}$ runs $\mathsf{LocalMap}^{\mathsf{Prefix}}(W)$ and obtains $(\{\mathsf{pid}_{w_i}\}_{i\in[n]}, \mathsf{List}^{\mathcal{R}})$.
2) $\mathcal{S}$ and $\mathcal{R}$ sample secret shares $k^{\mathcal{S}}, k^{\mathcal{R}}$ of the PRF key, respectively.
3) For $j \in [m], k \in [d]$, $\mathcal{S}$ computes $\{q_{j,k,h}\}_{h\in[\mu]} := \mathsf{AllPrefix}(q_{j,k}, \mu')$. $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$, where $\mathcal{R}$ as the sender inputs $\mathsf{List}^{\mathcal{R}}$ and $\mathcal{S}$ as the receiver inputs $\{k\|q_{j,k,h}\}_{j\in[m],k\in[d],h\in[\mu]}$. $\mathcal{S}$ receives $\{e^{\mathcal{S}}_{q_j,k,h}\|v^{\mathcal{S}}_{q_j,k,h}\}_{j\in[m],k\in[d],h\in[\mu]}$ and $\mathcal{R}$ receives $\{e^{\mathcal{R}}_{q_j,k,h}\|v^{\mathcal{R}}_{q_j,k,h}\}_{j\in[m],k\in[d],h\in[\mu]}$.
4) For $j \in [m], k \in [d]$, $\mathcal{S}$ and $\mathcal{R}$ invoke sub-protocol Π_{EQSel}, where $\mathcal{S}$ inputs $\{(e^{\mathcal{S}}_{q_j,k,h}, v^{\mathcal{S}}_{q_j,k,h})\}_{h\in[\mu]}$ and receives $r^{\mathcal{S}}_{q_j,k}$, and $\mathcal{R}$ inputs $\{(e^{\mathcal{R}}_{q_j,k,h}, v^{\mathcal{R}}_{q_j,k,h})\}_{h\in[\mu]}$ and receives $r^{\mathcal{R}}_{q_j,k}$.
5) For $j \in [m]$, $\mathcal{S}$ computes $r^{\mathcal{S}}_{q_j} := \sum_{k\in[d]} r^{\mathcal{S}}_{q_j,k}$, and $\mathcal{R}$ computes $r^{\mathcal{R}}_{q_j} := \sum_{k\in[d]} r^{\mathcal{R}}_{q_j,k}$.
6) $\mathcal{S}$ and $\mathcal{R}$ invoke $\mathcal{F}_{\mathsf{si\text{-}OPRF}}$, where $\mathcal{R}$ as the sender inputs $(k^{\mathcal{R}}, \{r^{\mathcal{R}}_{q_j}\}_{j\in[m]})$, and $\mathcal{S}$ as the receiver inputs $(k^{\mathcal{S}}, \{r^{\mathcal{S}}_{q_j} + \mathsf{pid}_{q_j}\}_{j\in[m]})$ and learns $\{\mathsf{ID}_{q_j}\}_{j\in[m]}$.
7) Symmetrically, for $i \in [n], k \in [d]$, $\mathcal{R}$ computes $\{w_{i,k,h}\}_{h\in[\mu]} := \mathsf{AllPrefix}(w_{i,k}, \mu')$. $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$, where $\mathcal{S}$ as the sender inputs $\mathsf{List}^{\mathcal{S}}$ and $\mathcal{R}$ as the receiver inputs $\{k\|w_{i,k,h}\}_{i\in[n],k\in[d],h\in[\mu]}$. $\mathcal{S}$ receives $\{e^{\mathcal{S}}_{w_i,k,h}\|v^{\mathcal{S}}_{w_i,k,h}\}_{i\in[n],k\in[d],h\in[\mu]}$ and $\mathcal{R}$ receives $\{e^{\mathcal{R}}_{w_i,k,h}\|v^{\mathcal{R}}_{w_i,k,h}\}_{i\in[n],k\in[d],h\in[\mu]}$.
8) For $i \in [n], k \in [d]$, $\mathcal{S}$ and $\mathcal{R}$ invoke sub-protocol Π_{EQSel}, where $\mathcal{S}$ inputs $\{(e^{\mathcal{S}}_{w_i,k,h}, v^{\mathcal{S}}_{w_i,k,h})\}_{h\in[\mu]}$ and receives $r^{\mathcal{S}}_{w_i,k}$, and $\mathcal{R}$ inputs $\{(e^{\mathcal{R}}_{w_i,k,h}, v^{\mathcal{R}}_{w_i,k,h})\}_{h\in[\mu]}$ and receives $r^{\mathcal{R}}_{w_i,k}$.
9) For $i \in [n]$, $\mathcal{S}$ computes $r^{\mathcal{S}}_{w_i} := \sum_{k\in[d]} r^{\mathcal{S}}_{w_i,k}$, and $\mathcal{R}$ computes $r^{\mathcal{R}}_{w_i} := \sum_{k\in[d]} r^{\mathcal{R}}_{w_i,k}$.
10) $\mathcal{S}$ and $\mathcal{R}$ invoke $\mathcal{F}_{\mathsf{si\text{-}OPRF}}$, where $\mathcal{S}$ as the sender inputs $(k^{\mathcal{S}}, \{r^{\mathcal{S}}_{w_i}\}_{i\in[n]})$, and $\mathcal{R}$ as the receiver inputs $(k^{\mathcal{R}}, \{r^{\mathcal{R}}_{w_i} + \mathsf{pid}_{w_i}\}_{i\in[n]})$ and learns $\{\mathsf{ID}_{w_i}\}_{i\in[n]}$.

Figure 15: Protocol of fuzzy mapping with prefix optimization.

7.1. Optimized Fuzzy Mapping

At a high level, by incorporating prefix techniques [12], [16], [18] into our framework, the sender in the so-OPPRF protocol does not need to program the entire interval $[q_{j,k} - \delta, q_{j,k} + \delta]$ of size $2\delta + 1$, but only $O(\log \delta)$ prefixes that together cover the interval. On the receiver side, this modification introduces an additional $O(\log \delta)$ evaluations of so-OPPRF, since there are $O(\log \delta)$ candidate prefixes that may match. Consequently, the overall complexity is reduced from $O(\delta)$ to $O(\log \delta)$.

Protocol $\Pi^{L_\infty}_{\mathsf{FPSI\text{-}Prefix}}$

Parameters: Distance threshold δ. Prefix parameters $\mu' = 1 + \log \delta$ and $\mu = \hat{\mu} = 2 + \log \delta$.

Input: $\mathcal{S}$ inputs $Q = \{q_j\}_{j\in[m]} \in \mathbb{U}^{d\times m}$ and $\mathcal{R}$ inputs $W = \{w_i\}_{i\in[n]} \in \mathbb{U}^{d\times n}$.

Protocol:

1) $\mathcal{S}$ and $\mathcal{R}$ invoke sub-protocol $\Pi_{\mathsf{FMap\text{-}Prefix}}$, where $\mathcal{S}$ inputs Q and $\mathcal{R}$ inputs W. $\mathcal{S}$ obtains $\{\mathsf{ID}_{q_j}\}_{j\in[m]}$ and $\mathcal{R}$ obtains $\{\mathsf{ID}_{w_i}\}_{i\in[n]}$.
2) For $i \in [n], k \in [d]$, $\mathcal{R}$ computes $\{w_{i,k,h}\}_{h\in[\hat{\mu}]} := \mathsf{Decompose}(w_{i,k} - \delta, w_{i,k} + \delta)$.
3) For $j \in [m], k \in [d]$, $\mathcal{S}$ computes $\{q_{j,k,h}\}_{h\in[\mu]} := \mathsf{AllPrefix}(q_{j,k}, \mu')$.
4) $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$, where $\mathcal{R}$ inputs $\{(\mathsf{ID}_{w_i}\|k\|w_{i,k,h}, 0)\}_{i\in[n],k\in[d],h\in[\hat{\mu}]}$ and $\mathcal{S}$ inputs $\{\mathsf{ID}_{q_j}\|k\|q_{j,k,h}\}_{j\in[m],k\in[d],h\in[\mu]}$. $\mathcal{S}$ receives $\{e^{\mathcal{S}}_{j,k,h}\|v^{\mathcal{S}}_{j,k,h}\}_{j\in[m],k\in[d],h\in[\mu]}$ and $\mathcal{R}$ receives $\{e^{\mathcal{R}}_{j,k,h}\|v^{\mathcal{R}}_{j,k,h}\}_{j\in[m],k\in[d],h\in[\mu]}$.
5) For $j \in [m], k \in [d]$, $\mathcal{S}$ and $\mathcal{R}$ invoke sub-protocol Π_{EQSel}, where $\mathcal{S}$ inputs $\{(e^{\mathcal{S}}_{j,k,h}, v^{\mathcal{S}}_{j,k,h})\}_{h\in[\mu]}$ and receives $t^{\mathcal{S}}_{j,k}$, and $\mathcal{R}$ inputs $\{(e^{\mathcal{R}}_{j,k,h}, v^{\mathcal{R}}_{j,k,h})\}_{h\in[\mu]}$ and receives $t^{\mathcal{R}}_{j,k}$.
6) For $j \in [m]$, $\mathcal{S}$ computes $t^{\mathcal{S}}_j := \sum_{k\in[d]} t^{\mathcal{S}}_{j,k}$, and $\mathcal{R}$ computes $t^{\mathcal{R}}_j := \sum_{k\in[d]} t^{\mathcal{R}}_{j,k}$.
7) $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{PEQT}}$, where $\mathcal{S}$ inputs $\{t^{\mathcal{S}}_j\}_{j\in[m]}$ and $\mathcal{R}$ inputs $\{-t^{\mathcal{R}}_j\}_{j\in[m]}$. $\mathcal{R}$ receives $\{b_j\}_{j\in[m]}$ where $b_j := \mathbf{1}\{t^{\mathcal{S}}_j = -t^{\mathcal{R}}_j\}$.
8) $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{OT}}$, where $\mathcal{S}$ inputs $\{(\perp, q_j)\}_{j\in[m]}$, and $\mathcal{R}$ inputs $\{b_j\}_{j\in[m]}$ and receives $\{z_j\}_{j\in[m]}$.
9) $\mathcal{R}$ outputs $Z := \{z_j \mid b_j = 1 \text{ for } j \in [m]\}$.

Figure 16: Protocol of fuzzy PSI with prefix optimization for L_∞ distance.

We first introduce a new building block, denoted as random equality-conditional selection (EQSel). As illustrated in Figure 14, EQSel takes as input a set of secret shares of $\{(e_i, v_i)\}_{i\in[h]}$ from two parties, where e_i represents a condition and v_i denotes a payload. If there exists $i \in [h]$ such that $e_i = 0$, the protocol outputs the associated payload v_i; otherwise, it outputs a random value. Note that in our applications, there exists at most one e_i equal to 0. We instantiate this protocol using secret-shared private equality test [48] and secret-shared multiplexer [52].

We present our optimized fuzzy mapping protocol in Figure 15. The main difference over our non-prefix fuzzy mapping protocol is that for each $w_{i,k}$, the receiver needs to prepare μ candidate prefixes for the invocation of so-OPPRF, where the sender sets the programmed values of $q_{j,k}$'s prefixes as $0\|r_{j,k}$. Subsequently, both parties receive μ secret shares from so-OPPRF and then invoke EQSel on them to select $r_{j,k}$ or a random value depending on whether there is a matched prefix. The remaining part of the protocol keeps the same as Π_{FMap}.

7.2. Optimized Fuzzy PSI for $L_{p\in[1,\infty]}$ Distance

We present our fuzzy PSI protocol with prefix optimization for L_∞ distance. The optimized protocol for L_p distance is presented in Appendix B.3. Since our $\Pi_{\mathsf{FMap\text{-}prefix}}$ achieves the same functionality as Π_{FMap}, we only need to redesign the remaining part of the fuzzy PSI protocol (i.e., the refined filtering phase).

Similarly, we use the prefix representation of the interval of size $2\delta + 1$ to reduce the number of key-value pairs for the refined filtering. With prefix optimization, the sender extends its every single input to μ prefixes in order to match the prefixes of the receiver. Therefore, both parties obtain μ outputs from the so-OPPRF for each dimension of a single element where at most one of these μ prefixes will match. We then use our EQSel protocol again to select the matched prefix, and the output is either the associated value or a random value.

The detailed protocol is shown in Figure 16. The security proof of protocol $\Pi^{L_\infty}_{\mathsf{FPSI\text{-}Prefix}}$ follows that of protocol $\Pi^{L_\infty}_{\mathsf{FPSI}}$ in Section 6.1. We omit it due to limited space.

8. Evaluation

8.1. Experimental Setup

We implement our protocols in C++ and our code is available at https://github.com/Th0masAndy/FPSI. All experiments are conducted on a server running Ubuntu 22.04, equipped with two AMD EPYC 9555 64-core processors and 512 GB of RAM. The sender and receiver are emulated as two separate threads within a single process. Each experiment is repeated five times, and the average values are reported. We set the computational security parameter to $\kappa = 128$ and the statistical security parameter to $\lambda = 40$. Network conditions are simulated using the Linux `tc` command, configured with a bandwidth of 10 Gbps and a latency of 0.02 ms. In Appendix C, we further evaluate the performance under varying bandwidth and latency settings.

We compare our protocols with the state-of-the-art linear-complexity fuzzy PSI schemes [14], [16], as they rely on the same assumption as ours and support high-dimensional inputs. For a fair comparison, we re-execute their publicly available implementations under the same experimental environment as ours. Below, we detail the implementation components of our protocols. For OKVS, we adopt the implementation from [6]. The implementations of si-OPRF and so-OPRF are built upon the MPC-friendly PRF proposed by [46]. We use silent OT from the libOTe library [53], and the implementations of ssPEQT and PEQT are based on the protocols from [48]. For the prefix-related algorithms, we use the implementation of [16].

8.2. Performance of Fuzzy PSI

We evaluate the performance of our fuzzy PSI protocols in Section 6 and compare them with the state-of-the-art

TABLE 2: The communication (MB) and running time (s) of fuzzy PSI for L_∞, L_1 and L_2. "—" indicates out-of-memory. "Ours" denotes our fuzzy PSI protocols in Section 6. The best results are marked in green.

$m=n$	Protocol	(d,δ)											
		$(4,16)$		$(8,16)$		$(16,16)$		$(4,32)$		$(8,32)$		$(16,32)$	
		Comm.	Time	Comm.	Time	Comm.	Time	Comm.	Time	Comm.	Time	Comm.	Time
L_∞ Distance													
2^8	Gao et al. [14]	23.2	4.4	46.2	8.6	92.1	16.6	45.5	7.8	90.7	15.1	181.2	30.5
	Dang et al. [16]	43.5	3.1	86.9	5.4	173.6	11.8	46.8	2.9	93.5	5.7	186.9	12.1
	Ours	7.3	0.3	10.6	0.3	17.1	0.4	9.4	0.3	14.7	0.4	25.2	0.5
2^{12}	Gao et al. [14]	370.9	73.2	738.5	144.6	1473.9	282.5	727.3	132.7	1451.3	253.9	2899.5	507.2
	Dang et al. [16]	695.8	43.3	1389.5	85.4	2777.0	181.2	749.3	46.4	1496.5	91.3	2990.9	190.7
	Ours	60.5	1.3	112.3	2.0	215.8	3.0	93.2	1.4	177.6	2.2	346.5	3.7
2^{16}	Gao et al. [14]	5934.1	1195.7	11816.7	2269.5	23581.9	4637.7	11636.5	2132.4	23221.5	4197.1	—	—
	Dang et al. [16]	11132.6	707.3	22232.2	1406.1	44431.4	2945.4	11988.4	737.6	23943.8	1447.8	47854.6	3150.7
	Ours	908.7	17.2	1737.6	25.0	3396.0	42.0	1432.0	19.0	2784.7	28.9	5491.5	49.7
L_1 Distance													
2^8	Gao et al. [14]	23.3	4.2	46.3	8.3	92.2	16.6	45.6	7.7	90.8	14.9	181.3	30.4
	Dang et al. [16]	61.2	3.8	122.2	7.2	244.2	14.7	71.2	4.0	142.2	7.6	284.2	15.6
	Ours	7.8	0.3	11.3	0.4	18.4	0.5	9.9	0.3	15.4	0.5	26.6	0.5
2^{12}	Gao et al. [14]	372.4	70.7	740.1	143.2	1475.4	282.6	729.2	129.6	1453.3	258.1	2901.4	506.9
	Dang et al. [16]	978.2	57.9	1954.8	116.7	3908.1	223.5	1138.3	61.3	2274.6	120.4	4547.1	244.3
	Ours	66.1	1.4	122.5	2.1	235.3	3.5	98.7	1.6	187.8	2.5	366.0	4.1
2^{16}	Gao et al. [14]	5959.1	1210.6	11841.7	2299.7	23606.9	4593.3	11667.5	2091.3	23252.5	4165.0	—	—
	Dang et al. [16]	15650.7	930.8	31276.8	1833.3	62529.1	3646.7	18212.9	995.8	36393.2	1954.8	72753.9	3888.1
	Ours	990.9	17.9	1893.8	27.8	3700.2	46.9	1516.2	20.2	2942.9	31.5	5797.7	55.8
L_2 Distance													
2^8	Gao et al. [14]	23.4	4.4	46.4	8.6	92.3	16.6	45.7	7.9	91.0	15.1	181.5	30.6
	Dang et al. [16]	70.1	4.0	140.0	7.6	279.6	14.5	84.8	4.6	169.0	8.9	337.4	16.0
	Ours	7.8	0.4	11.4	0.4	18.4	0.5	9.9	0.4	15.5	0.4	26.6	0.6
2^{12}	Gao et al. [14]	373.9	72.7	741.6	143.2	1476.9	283.8	731.1	132.5	1455.2	259.9	2903.3	509.4
	Dang et al. [16]	1122.0	63.1	2239.4	117.7	4474.2	231.3	1356.0	72.6	2703.5	132.6	5398.4	253.3
	Ours	66.3	1.4	122.7	2.2	235.5	3.5	99.2	1.6	188.2	2.5	366.4	4.0
2^{16}	Gao et al. [14]	5983.1	1223.9	11865.7	2321.1	23630.9	4585.5	11697.5	2127.9	23282.5	4134.8	—	—
	Dang et al. [16]	17951.5	1009.4	35830.4	1930.9	71588.3	3799.7	21696.1	1159.7	43255.6	2115.1	86374.7	4332.7
	Ours	998.8	18.6	1901.8	27.8	3708.2	47.7	1526.1	20.4	2952.8	31.7	5807.7	55.1

protocols [14], [16]. The results are presented in Table 2, where we set the input sizes to $m = n = 2^8, 2^{12}, 2^{16}$, the dimension to $d = 4, 8, 16$, and the distance threshold to $\delta = 16, 32$. We do not use prefix optimizations, since for these small distance thresholds our non-prefix protocols are more efficient. The performance of prefix optimizations for large thresholds is presented in the next section.

As shown in Table 2, our protocol for L_∞ distance achieves a $3\sim13\times$ improvement in communication efficiency and a $9\sim145\times$ improvement in computational efficiency, outperforming all previous protocols. Specifically, when $m = n = 2^{16}$, $d = 8$, and $\delta = 16$, our protocol consumes $13\times$ less communication than [16] and $7\times$ less than [14]. When $m = n = 2^{16}$, $d = 8$, and $\delta = 32$, our protocol is $50\times$ faster than [16] and $145\times$ faster than [14]. We note that our protocols achieve better improvement for larger-scale inputs, since the overhead of the underlying silent OT [41] can be amortized.

We also present the results for L_1 and L_2 distances, which are commonly used metrics. Compared with [16], our protocol achieves up to an $80\times$ improvement in running time and a $19\times$ reduction in communication cost. Moreover, our protocols for L_1 and L_2 distances incur only a minor overhead compared to that of L_∞ distance, whereas [16] nearly doubles its overall cost.

Under limited network conditions, where the time for communication dominates the total running time, we provide the results in Appendix C.

8.3. Performance of Fuzzy PSI with Prefix Optimization

We report the performance of our fuzzy PSI protocols with prefix optimization for large distance threshold δ. We set relatively large δ as $\{2^4, 2^6, 2^8, 2^{10}\}$. As shown in Table 3, our optimized protocols exhibit desirable performance gains for large distance thresholds δ. Overall, compared with the state-of-the-art work [16], we achieve a $7\sim10\times$ improvement in communication efficiency and a $13\sim38\times$ improvement in computational efficiency. We note that our non-prefix protocols still exhibit competitive performance compared to [16], especially in computation efficiency.

We also compare our optimized protocols with our non-prefix protocols. For $\delta \geq 64$, our optimized protocols significantly reduce the communication overhead. For instance, when $\delta = 1024$, it lowers the communication cost by $11\times$ compared to the non-prefix protocols, while reducing the running time by approximately half. However, our non-prefix protocols still achieve better runtime performance for $\delta \leq 256$, since the optimized protocols introduce additional operations such as conditional selection.

We provide the performance under different network settings in Appendix C. Since our optimized protocol significantly reduces communication overhead, its overall efficiency further improves under limited network conditions.

TABLE 3: The communication (MB) and running time (s) of fuzzy PSI with prefix optimizations for large threshold δ. We fix $m = n = 2^{12}$ and $d = 8$. "—" indicates out-of-memory. "Ours" denotes our fuzzy PSI protocols in Section 6 and "Ours-prefix" denotes our protocols with prefix optimizations in Section 7. The best results are marked in green.

Metric	Protocol	$\delta = 16$		$\delta = 64$		$\delta = 256$		$\delta = 1024$	
		Comm.	Time	Comm.	Time	Comm.	Time	Comm.	Time
L_∞	Gao et al. [14]	738.5	151.5	2876.9	496.6	11430.5	1850.6	—	—
	Dang et al. [16]	1389.5	86.2	1605.1	95.6	2581.0	156.9	2795.0	164.6
	Ours	112.3	1.8	308.3	2.6	1093.1	7.2	4235.8	19.7
	Ours-prefix	184.8	5.4	225.7	5.7	289.1	7.9	366.2	10.0
L_1	Gao et al. [14]	740.1	143.4	2879.3	483.3	11433.6	1840.1	—	—
	Dang et al. [16]	1954.8	113.4	2487.7	152.6	3695.2	216.6	4479.7	241.5
	Ours	122.5	2.1	318.5	2.9	1103.5	7.0	4246.5	20.3
	Ours-prefix	252.2	7.0	295.9	7.2	367.7	9.6	456.6	10.9
L_2	Gao et al. [14]	741.6	144.3	2881.5	490.6	11436.6	1856.3	—	—
	Dang et al. [16]	2239.4	126.6	2818.4	175.5	4248.0	303.1	5572.5	572.7
	Ours	122.7	2.1	319.4	2.9	1104.6	7.3	4247.8	21.0
	Ours-prefix	385.2	9.7	440.3	9.7	534.0	11.9	682.6	15.1

9. Discussion

Limitations. Our work still has the following limitations. First, similar to prior fuzzy PSI protocols [13], [14], [16], [17], [18], our protocol relies on certain assumptions about the input distribution. These assumptions are typically parameterized by the distance threshold δ. While such parameters are well-defined in theoretical analysis, determining appropriate values for real-world datasets remains challenging. Second, it remains unclear how to obtain malicious security without substantially compromising efficiency. A natural approach is to instantiate our construction using maliciously secure OT [54], [55] and authenticated secret sharing based on IT-MACs [56], [57]. However, a key technical challenge is how to efficiently verify that the OKVS encoding is well-formed and consistent.

Future Work. We outline several directions for future research. First, it is of interest to design fuzzy PSI protocols under weaker or more flexible assumptions, such as relaxing distribution constraints or supporting one-sided assumptions, while maintaining comparable efficiency. Second, although existing fuzzy PSI protocols guarantee correctness under ideal assumptions, it remains an open problem to quantify and evaluate the impact when these assumptions are violated in practice (e.g., when elements are closer than expected or exhibit collisions). Developing robustness and accuracy guarantees in such settings is an important direction for future work.

10. Conclusion

In this work, we present a novel modular design for fuzzy PSI based on symmetric primitives. Our protocol is built upon a new building block termed so-OPPRF, achieving linear complexity with respect to the input size m, n, the dimension d, and the distance threshold δ. We further optimize the protocol by incorporating prefix techniques, and reduce its complexity to logarithmic in δ. Experimental results demonstrate that our protocols outperform linear-complexity state-of-the-art works, significantly improving both computation and communication efficiency.

11. Acknowledgment

This work was supported by the Nanyang Technological University Centre in Computational Technologies for Finance (NTU-CCTF). It was also supported by Lee Kong Chian Chair Professorship, Singapore Management University. It was also supported by the National Research Foundation, Singapore, and Cyber Security Agency of Singapore under its National Cybersecurity R&D Programme and CyberSG R&D Cyber Research Programme Office. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of NTU-CCTF, National Research Foundation, Singapore, Cyber Security Agency of Singapore as well as CyberSG R&D Programme Office, Singapore.

12. Ethics considerations

This work provides effective solutions for the fuzzy private set intersection task, encouraging researchers to pay more attention to the privacy of data analysis. The experiments in this paper are all based on public datasets and do not contain any personal or illegal information. We believe that our research was done ethically.

13. LLM usage considerations

This paper used the LLM solely to assist with grammar checking and language polishing. The LLM did not contribute any conceptual ideas, methodological innovations, experimental designs, or analytical insights to this work. All intellectual contributions originate from the authors. All data provided to the LLM for linguistic refinement contain no sensitive, personal, or ethically problematic information.

References

[1] C. Meadows, "A more efficient cryptographic matchmaking protocol for use in the absence of a continuously available third party," in *1986 IEEE Symposium on Security and Privacy*. IEEE, 1986, pp. 134–134.

[2] M. J. Freedman, K. Nissim, and B. Pinkas, "Efficient private matching and set intersection," in *International conference on the theory and applications of cryptographic techniques*. Springer, 2004, pp. 1–19.

[3] C. Dong, L. Chen, and Z. Wen, "When private set intersection meets big data: an efficient and scalable protocol," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, 2013, pp. 789–800.

[4] B. Pinkas, T. Schneider, G. Segev, and M. Zohner, "Phasing: Private set intersection using permutation-based hashing," in *24th USENIX Security Symposium (USENIX Security 15)*, 2015, pp. 515–530.

[5] B. Pinkas, M. Rosulek, N. Trieu, and A. Yanai, "Psi from paxos: Fast, malicious private set intersection," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2020, pp. 739–767.

[6] S. Raghuraman and P. Rindal, "Blazing fast psi from improved okvs and subfield vole," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 2505–2517.

[7] M. Hao, W. Liu, L. Peng, H. Li, C. Zhang, H. Chen, and T. Zhang, "Unbalanced circuit-psi from oblivious key-value retrieval," in *33rd USENIX Security Symposium (USENIX Security 24)*, 2024, pp. 6435–6451.

[8] Google, "Better password protections in chrome - how it works," 2019, last accessed 11 November 2025. [Online]. Available: https://security.googleblog.com/2019/12/better-password-protections-in-chrome.html

[9] L. Shen, X. Chen, D. Wang, B. Fang, and Y. Dong, "Efficient and private set intersection of human genomes," in *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2018, pp. 761–764.

[10] E. Uzun, S. P. Chung, V. Kolesnikov, A. Boldyreva, and W. Lee, "Fuzzy labeled private set intersection with applications to private real-time biometric search," in *30th USENIX Security Symposium (USENIX Security 21)*, 2021, pp. 911–928.

[11] G. Garimella, M. Rosulek, and J. Singh, "Structure-aware private set intersection, with applications to fuzzy matching," in *Annual International Cryptology Conference*. Springer, 2022, pp. 323–352.

[12] G. Garimella, B. Goff, and P. Miao, "Computation efficient structure-aware psi from incremental function secret sharing," in *Annual International Cryptology Conference*. Springer, 2024, pp. 309–345.

[13] A. van Baarsen and S. Pu, "Fuzzy private set intersection with large hyperballs," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2024, pp. 340–369.

[14] Y. Gao, L. Qi, X. Liu, Y. Luo, and L. Wang, "Efficient fuzzy private set intersection from fuzzy mapping," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2024, pp. 36–68.

[15] C. Zhang, Y. Chen, Y. Cao, Y. Bai, S. Li, J. Lin, A. Wang, and X. Wang, "Fast fuzzy psi from symmetric-key techniques," *Cryptology ePrint Archive*, 2025.

[16] C. Dang, X. Zhou, and B. Liang, "Efficient fuzzy psi based on prefix representation," in *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*, 2025, pp. 2204–2218.

[17] L. Piske, J. Singh, N. Trieu, V. Kolesnikov, and V. Zikas, "Distance-aware ot with application to fuzzy psi," in *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*, 2025, pp. 4679–4691.

[18] A. van Baarsen and S. Pu, "Fuzzy private set intersection from vole," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2025, pp. 327–360.

[19] D. Richardson, M. Rosulek, and J. Xu, "Fuzzy psi via oblivious protocol routing," *Cryptology ePrint Archive*, 2024.

[20] D. Bui, G. Garimella, P. Miao, and P. V. L. Pham, "New framework for structure-aware psi from distributed function secret sharing," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2025, pp. 294–326.

[21] A. Chakraborti, G. Fanti, and M. K. Reiter, "Distance-aware private set intersection," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 319–336.

[22] L. Chmielewski and J.-H. Hoepman, "Fuzzy private matching," in *2008 Third International Conference on Availability, Reliability and Security*. IEEE, 2008, pp. 327–334.

[23] Q. Ye, R. Steinfeld, J. Pieprzyk, and H. Wang, "Efficient fuzzy matching and intersection on private datasets," in *International Conference on Information Security and Cryptology*. Springer, 2009, pp. 211–228.

[24] P. Indyk and D. Woodruff, "Polylogarithmic private approximations and efficient matching," in *Theory of Cryptography Conference*. Springer, 2006, pp. 245–264.

[25] E.-O. Blass and G. Noubir, "Assumption-free fuzzy psi via predicate encryption," *Cryptology ePrint Archive*, 2025.

[26] C. Cho, D. Dachman-Soled, and S. Jarecki, "Efficient concurrent covert computation of string equality and set intersection," in *Cryptographers' Track at the RSA Conference*. Springer, 2016, pp. 164–179.

[27] W. Chongchitmate, S. Lu, and R. Ostrovsky, "Approximate psi with near-linear communication," *Cryptology ePrint Archive*, 2024.

[28] T. De Vries, H. Ke, S. Chawla, and P. Christen, "Robust record linkage blocking using suffix arrays and bloom filters," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 5, no. 2, pp. 1–27, 2011.

[29] E. A. Durham, M. Kantarcioglu, Y. Xue, C. Toth, M. Kuzu, and B. Malin, "Composite bloom filters for secure record linkage," *IEEE transactions on knowledge and data engineering*, vol. 26, no. 12, pp. 2956–2968, 2013.

[30] X. He, A. Machanavajjhala, C. Flynn, and D. Srivastava, "Composing differential privacy and secure computation: A case study on scaling private record linkage," in *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, 2017, pp. 1389–1406.

[31] A. Inan, M. Kantarcioglu, G. Ghinita, and E. Bertino, "Private record matching using differential privacy," in *Proceedings of the 13th International Conference on Extending Database Technology*, 2010, pp. 123–134.

[32] B. Khurram and F. Kerschbaum, "Sfour: a protocol for cryptographically secure record linkage at scale," in *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 2020, pp. 277–288.

[33] R. Wei and F. Kerschbaum, "Cryptographically secure private record linkage using locality-sensitive hashing," *Proceedings of the VLDB Endowment*, vol. 17, no. 2, pp. 79–91, 2023.

[34] S. Stammler, T. Kussel, P. Schoppmann, F. Stampe, G. Tremper, S. Katzenbeisser, K. Hamacher, and M. Lablans, "Mainzelliste secureepilinker (mainsel): privacy-preserving record linkage using secure multi-party computation," *Bioinformatics*, vol. 38, no. 6, pp. 1657–1668, 2022.

[35] A. Adir, E. Aharoni, N. Drucker, E. Kushnir, R. Masalha, M. Mirkin, and O. Soceanu, "Privacy-preserving record linkage using local sensitive hash and private set intersection," in *International Conference on Applied Cryptography and Network Security*. Springer, 2022, pp. 398–424.

[36] K. Zhao, H. Lu, and J. Mei, "Locality preserving hashing," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 28, no. 1, 2014.

[37] J. Leskovec, A. Rajaraman, and J. D. Ullman, *Mining of massive data sets*. Cambridge university press, 2020.

[38] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[39] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography conference*. Springer, 2006, pp. 265–284.

[40] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher, "Min-wise independent permutations," in *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, 1998, pp. 327–336.

[41] E. Boyle, G. Couteau, N. Gilboa, Y. Ishai, L. Kohl, P. Rindal, and P. Scholl, "Efficient two-round ot extension and silent non-interactive secure computation," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 291–308.

[42] S. Raghuraman, P. Rindal, and T. Tanguy, "Expand-convolute codes for pseudorandom correlation generators from lpn," in *Annual International Cryptology Conference*. Springer, 2023, pp. 602–632.

[43] V. Kolesnikov, N. Matania, B. Pinkas, M. Rosulek, and N. Trieu, "Practical multi-party private set intersection from symmetric-key techniques," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1257–1272.

[44] G. Garimella, B. Pinkas, M. Rosulek, N. Trieu, and A. Yanai, "Oblivious key-value stores and amplification for private set intersection," in *Advances in Cryptology–CRYPTO 2021: 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16–20, 2021, Proceedings, Part II 41*. Springer, 2021, pp. 395–425.

[45] A. van Baarsen and M. Stevens, "Amortizing circuit-PSI in the multiple sender/receiver setting," *IACR Communications in Cryptology*, vol. 1, no. 3, 2024.

[46] N. Alamati, G.-V. Policharla, S. Raghuraman, and P. Rindal, "Improved alternating-moduli prfs and post-quantum signatures," in *Annual International Cryptology Conference*. Springer, 2024, pp. 274–308.

[47] B. Pinkas, T. Schneider, O. Tkachenko, and A. Yanai, "Efficient circuit-based psi with linear communication," in *Advances in Cryptology–EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part III 38*. Springer, 2019, pp. 122–153.

[48] P. Rindal and P. Schoppmann, "Vole-psi: fast oprf and circuit-psi from vector-ole," in *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 2021, pp. 901–930.

[49] D. Boneh, Y. Ishai, A. Passelègue, A. Sahai, and D. J. Wu, "Exploring crypto dark matter: New simple prf candidates and their applications," in *Theory of Cryptography Conference*. Springer, 2018, pp. 699–729.

[50] I. Dinur, S. Goldfeder, T. Halevi, Y. Ishai, M. Kelkar, V. Sharma, and G. Zaverucha, "Mpc-friendly symmetric cryptography from alternating moduli: Candidates, protocols, and applications," in *Annual International Cryptology Conference*. Springer, 2021, pp. 517–547.

[51] M. R. Albrecht, A. Davidson, A. Deo, and D. Gardham, "Crypto dark matter on the torus: Oblivious prfs from shallow prfs and tfhe," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2024, pp. 447–476.

[52] D. Rathee, M. Rathee, N. Kumar, N. Chandran, D. Gupta, A. Rastogi, and R. Sharma, "Cryptflow2: Practical 2-party secure inference," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 325–342.

[53] P. Rindal and L. Roy, "libOTe: an efficient, portable, and easy to use Oblivious Transfer Library," https://github.com/osu-crypto/libOTe.

[54] K. Yang, C. Weng, X. Lan, J. Zhang, and X. Wang, "Ferret: Fast extension for correlated ot with small communication," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 1607–1626.

[55] L. Roy, "Softspokenot: Quieter ot extension from small-field silent vole in the minicrypt model," in *Annual international cryptology conference*. Springer, 2022, pp. 657–687.

[56] R. Bendlin, I. Damgård, C. Orlandi, and S. Zakarias, "Semi-homomorphic encryption and multiparty computation," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2011, pp. 169–188.

[57] J. B. Nielsen, P. S. Nordholt, C. Orlandi, and S. S. Burra, "A new approach to practical active-secure two-party computation," in *Annual Cryptology Conference*. Springer, 2012, pp. 681–700.

Appendix A. Threat Model and Security Proof

A.1. Threat Model

Similar to prior works [13], [14], [16], [18], we consider static semi-honest probabilistic polynomial-time (PPT) adversaries. Namely, a PPT adversary $\mathcal{A}$ passively corrupts either the sender $\mathcal{S}$ or the receiver $\mathcal{R}$ at the beginning of the protocol and honestly follows the protocol specification. We use the standard simulation-based security definition for secure two-party computation. Our construction invokes multiple sub-protocols, and we use the *hybrid model* to describe them. By convention, a protocol invoking a functionality $\mathcal{F}$ is referred to as the $\mathcal{F}$-hybrid model. We give the formal security definition as follows.

Definition 5. *Let* $\mathsf{view}^{\Pi}_{\mathcal{S}}(x,y)$ *and* $\mathsf{view}^{\Pi}_{\mathcal{R}}(x,y)$ *be the views of* $\mathcal{S}$ *and* $\mathcal{R}$ *in a protocol* Π*, respectively, where* x *is the input of* $\mathcal{S}$ *and* y *is the input of* $\mathcal{R}$*. Let* $\mathsf{out}(x,y)$ *be the protocol's output of both parties and* $\mathcal{F}(x,y)$ *be the functionality's output.* Π *is said to securely compute a functionality* $\mathcal{F}$ *in the semi-honest model if for every PPT adversary* $\mathcal{A}$ *there exists PPT simulators* $\mathsf{Sim}_{\mathcal{S}}$ *and* $\mathsf{Sim}_{\mathcal{R}}$ *such that for all inputs* x *and* y*,*

$$\{\mathsf{view}^{\Pi}_{\mathcal{S}}(x,y), \mathsf{out}(x,y)\} \approx_c \{\mathsf{Sim}_{\mathcal{S}}(x, \mathcal{F}_{\mathcal{S}}(x,y)), \mathcal{F}(x,y)\},$$
$$\{\mathsf{view}^{\Pi}_{\mathcal{R}}(x,y), \mathsf{out}(x,y)\} \approx_c \{\mathsf{Sim}_{\mathcal{R}}(x, \mathcal{F}_{\mathcal{R}}(x,y)), \mathcal{F}(x,y)\}.$$

A.2. Proof of so-OPPRF

We give the detailed proof of Theorem 1.

Proof. If both parties behave honestly, correctness on programmed points $(q,z) \in L$ follows from the correctness of OKVS, since $y_i^{\mathcal{S}} + y_i^{\mathcal{R}} = \mathsf{OKVS.Decode}(D,q) + f_i^{\mathcal{S}} + f_i^{\mathcal{R}} = z - F_k(q) + F_k(q) = z$. On unprogrammed points $x \in X$, the OKVS encoding D is independent from $F_k(x)$, and thus the value $y_i^{\mathcal{S}} + y_i^{\mathcal{R}} := F_k(x) + \mathsf{OKVS.Decode}(D,x)$ is indistinguishable from a uniformly random distribution, since $F_k(\cdot)$ is indistinguishable from a random function by definition of $\mathcal{F}_{\mathsf{OPRF}}$. We next consider a corrupted sender or receiver.

Corrupted sender. We show the simulator $\mathsf{Sim}_{\mathcal{S}}(L, \{y_i^{\mathcal{S}}\}_{i\in[n]}, \mathcal{O}^{F'})$:

1) $\mathsf{Sim}_{\mathcal{S}}$ computes $D \leftarrow \mathsf{OKVS.Encode}(\{(q_j, z_j - F(q_j))\}_{j\in[m]})$, where $F(q_j)$ are randomly sampled.

Functionality $\mathcal{F}_{\mathsf{MUX}}$

Parameters: Sender $\mathcal{S}$ and receiver $\mathcal{R}$.

Protocol:

1) Wait for input (x_0, b_0) from $\mathcal{S}$ and (x_1, b_1) from $\mathcal{R}$.
2) Select r_0 randomly, and set $r_1 = x_0 + x_1 - r_0$ if $b_0 + b_1 = 1$ otherwise set $r_1 = -r_0$.
3) Output r_0 to $\mathcal{R}$ and r_1 to $\mathcal{S}$.

Figure 17: Functionality of MUX.

Functionality $\mathcal{F}_{\mathsf{ssPEQT}}$

Parameters: Sender $\mathcal{S}$ and receiver $\mathcal{R}$.

Protocol:

1) Wait for input x_0 from $\mathcal{S}$ and x_1 from $\mathcal{R}$.
2) Select r_0 randomly, and set $r_1 = 1 - r_0$ if $x_0 = x_1$ otherwise set $r_1 = -r_0$.
3) Output r_0 to $\mathcal{R}$ and r_1 to $\mathcal{S}$.

Figure 18: Functionality of secret-shared PEQT.

For each query $x \notin \{q_j\}_{j\in[m]}$, $\mathsf{Sim}_{\mathcal{S}}$ defines $F(x) := F'(x) - \mathsf{OKVS.Decode}(D, x)$.

2) $\mathsf{Sim}_{\mathcal{S}}$ appends $(\{f_i^{\mathcal{S}} := y_i^{\mathcal{S}}\}_{i\in[n]}, \mathcal{O}^F, \mathcal{O}^{F'})$ in the view.

We show that the view of adversary simulated by Sim has the identical distribution as its view in the real-world execution. For programmed points q_j, it holds $F'(q_j) = F(q_j) + \mathsf{OKVS.Decode}(D, q_j)$. For unprogrammed points x, the output $F(x) := F'(x) - \mathsf{OKVS.Decode}(D, x)$ is uniformly random according to the independence property of OKVS.

Corrupted receiver. We show the simulator $\mathsf{Sim}_{\mathcal{R}}(X, \{y_i^{\mathcal{R}}\}_{i\in[n]})$:

1) $\mathsf{Sim}_{\mathcal{R}}$ samples a random OKVS encoding D on m random key-value pairs and appends D in the view.
2) $\mathsf{Sim}_{\mathcal{R}}$ appends $\{f_i^{\mathcal{R}}\}_{i\in[n]}$ in the view, where $f_i^{\mathcal{R}} := y_i^{\mathcal{R}} - \mathsf{OKVS.Decode}(D, x_i)$ for $i \in [n]$.

We show that the output by $\mathsf{Sim}_{\mathcal{R}}$ is indistinguishable from the real protocol. By the double obliviousness property of OKVS and the randomness of so-OPRF, the randomly sampled OKVS encoding D is statistically indistinguishable from a real protocol execution. Moreover, $f_i^{\mathcal{R}}$ has the same distribution in both the real and simulated protocols such that $y_i^{\mathcal{R}} = f_i^{\mathcal{R}} + \mathsf{OKVS.Decode}(D, x_i)$.

□

A.3. Proof of Fuzzy PSI for L_∞ distance

We give the detailed proof of Theorem 3.

Proof. **Corrupted sender.** We show the simulator $\mathsf{Sim}_{\mathcal{S}}(Q)$, which invokes the sub-simulator $\mathsf{Sim}_{\mathcal{S}}^{\mathsf{FMap}}$:

1) $\mathsf{Sim}_{\mathcal{S}}$ invokes $\mathsf{Sim}_{\mathcal{S}}^{\mathsf{FMap}}(Q, \mathsf{ID}_Q)$ and appends the output to the view, where ID_Q is randomly sampled.
2) $\mathsf{Sim}_{\mathcal{S}}$ randomly samples $\{r_{j,k}^{\mathcal{S}}\}_{j\in[m],k\in[d]}$ and appends them to the view.

Protocol $\Pi_{\mathsf{getList_p}}$

Parameters: $\{\mathsf{ID}_{w_i}\}_{i\in[n]}, \{w_i\}_{i\in[n]}, p$. Denote $\mu = \log \delta$.

Protocol:

1) For $i \in [n], k \in [d]$, compute $\{w_{0,i,k,h}\}_{h\in[\mu]} := \mathsf{Decompose}([w_{i,k} - \delta, w_{i,k}])$ and $\{w_{1,i,k,h}\}_{h\in[\mu]} := \mathsf{Decompose}([w_{i,k} + 1, w_{i,k} + \delta])$.
2) If $p = 1$:
   - $\mathsf{List_p} := \{(\mathsf{ID}_{w_i} \| k \| \sigma \| w_{\sigma,i,k,h}, 0 \| \| w^* - w_{i,k}|)\}_{i\in[n],k\in[d],\sigma\in[0,1],h\in[\mu]}$, where $w^* := \mathsf{UpBound}(w_{\sigma,i,k,h})$ if $\sigma = 0$ and $w^* := \mathsf{LowBound}(w_{\sigma,i,k,h})$ otherwise.
3) If $p = 2$:
   - $\mathsf{List_p} := \{(\mathsf{ID}_{w_i} \| k \| \sigma \| w_{\sigma,i,k,h}, 0 \| \| w^* - w_{i,k}| \| | w^* - w_{i,k}|^2)\}_{i\in[n],k\in[d],\sigma\in[0,1],h\in[\mu]}$, where $w^* := \mathsf{UpBound}(w_{\sigma,i,k,h})$ if $\sigma = 0$ and $w^* := \mathsf{LowBound}(w_{\sigma,i,k,h})$ otherwise.
4) Pad $\mathsf{List_p}$ into size $n \cdot d \cdot 2 \cdot \mu$ and outputs $\mathsf{List_p}$

Figure 19: Protocol for computing list for $p = 1$ and $p = 2$. The protocol can extend to arbitrary p.

Protocol $\Pi_{\mathsf{getDistance_p}}$

Parameters: Sender inputs $\{s_h^{\mathcal{S}}\}_{h\in[\mu]} \in \mathbb{F}^{\mu\times p}$, $q \in \mathbb{U}$, $\sigma \in \{0, 1\}$. Receiver inputs $\{s_h^{\mathcal{R}}\}_{h\in[\mu]} \in \mathbb{F}^{\mu\times p}$. Functionality $\mathcal{F}_{\mathsf{Mult}}$. Denote $\mu = \log \delta$.

Protocol:

1) $\mathcal{S}$ computes $\{q_h\}_{h\in[\mu]} := \mathsf{AllPrefix}(q, \mu)$.
2) For $h \in [\mu]$, $\mathcal{S}$ computes $e_h := |q - \mathsf{UpBound}(q_h)|$ if $\sigma = 0$ and $e_h = |q - \mathsf{LowBound}(q_h)|$ otherwise.
3) If $p = 1$:
   - For $h \in [\mu]$, $\mathcal{S}$ computes $d_h^{\mathcal{S}} := e_h + s_h^{\mathcal{S}}$ and $\mathcal{R}$ computes $d_h^{\mathcal{R}} := s_h^{\mathcal{R}}$.
4) If $p = 2$:
   - For $h \in [\mu]$, $\mathcal{S}$ parses $s_h^{\mathcal{S}} = s_{h,1}^{\mathcal{S}} \| s_{h,2}^{\mathcal{S}} \in \mathbb{F}^2$ and $\mathcal{R}$ parses $s_h^{\mathcal{R}} = s_{h,1}^{\mathcal{R}} \| s_{h,2}^{\mathcal{R}} \in \mathbb{F}^2$.
   - For $h \in [\mu], i \in [1, p]$, $\mathcal{S}$ and $\mathcal{R}$ invoke $\mathcal{F}_{\mathsf{Mult}}$ where $\mathcal{S}$ inputs e_h and $\mathcal{R}$ inputs $s_{h,1}^{\mathcal{R}}$. $\mathcal{S}$ receives $m_h^{\mathcal{S}}$ and $\mathcal{R}$ receives $m_h^{\mathcal{R}}$.
   - For $h \in [\mu]$, $\mathcal{S}$ computes $d_h^{\mathcal{S}} := e_h^2 + 2e_h \cdot s_{h,1}^{\mathcal{S}} + 2m_h^{\mathcal{S}} + s_{h,2}^{\mathcal{S}}$ and $d_h^{\mathcal{R}} := 2m_h^{\mathcal{R}} + s_{h,2}^{\mathcal{R}}$.
5) $\mathcal{S}$ outputs $\{d_h^{\mathcal{S}}\}_{h\in[\mu]}$ and $\mathcal{R}$ outputs $\{d_h^{\mathcal{R}}\}_{h\in[\mu]}$.

Figure 20: Protocol for computing distance for $p = 1$ and $p = 2$. The protocol can extend to arbitrary p.

We show that the output by $\mathsf{Sim}_{\mathcal{S}}$ is indistinguishable from the real protocol. According to the security property of fuzzy mapping in Theorem 2, the simulated view of $\mathsf{Sim}_{\mathcal{S}}^{\mathsf{FMap}}$ is indistinguishable from the real execution. Besides, the only difference is $\{r_{j,k}^{\mathcal{S}}\}_{j\in[m],k\in[d]}$. They are random secret shares in the real protocol according to the so-OPPRF functionality, while they are randomly sampled in the simulated view with the same distribution. Therefore, the output by $\mathsf{Sim}_{\mathcal{S}}$ is indistinguishable from the real protocol.

Corrupted receiver. We show the simulator $\mathsf{Sim}_{\mathcal{R}}(W, Z)$, which invokes the sub-simulator $\mathsf{Sim}_{\mathcal{R}}^{\mathsf{FMap}}$:

1) $\mathsf{Sim}_{\mathcal{R}}$ randomly samples ID_W. $\mathsf{Sim}_{\mathcal{R}}$ invokes $\mathsf{Sim}_{\mathcal{R}}^{\mathsf{FMap}}(W, \mathsf{ID}_W)$ and appends the output to the view.
2) $\mathsf{Sim}_{\mathcal{R}}$ randomly samples $\{r^{\mathcal{R}}_{j,k}\}_{j\in[m],k\in[d]}$ and samples a random function F such that $F(\mathsf{ID}_{w_i}\|k\|(w_{i,k}+t)) = 0$ for $t \in [-\delta, \delta], i \in [n], k \in [d]$. $\mathsf{Sim}_{\mathcal{R}}$ appends $(\{r^{\mathcal{R}}_{j,k}\}_{j\in[m],k\in[d]}, \mathcal{O}^F)$ to the view.
3) $\mathsf{Sim}_{\mathcal{R}}$ uses $\perp$ to pad Z to m elements, randomly shuffles the set Z, and sets $b_j := 0$ if $z_j = 0$ and $b_j := 1$ otherwise for $j \in [m]$. $\mathsf{Sim}_{\mathcal{R}}$ appends $(\{b_j\}_{j\in[m]}, Z)$ to the view.

We show that the output by $\mathsf{Sim}_{\mathcal{R}}$ is indistinguishable from the real protocol. According to the security property of fuzzy mapping in Theorem 2, the simulated view of $\mathsf{Sim}_{\mathcal{R}}^{\mathsf{FMap}}$ is indistinguishable from the real execution. Moreover, the way $\mathcal{R}$ obtains the elements in Z is identical to the real execution since the elements in Z are randomly shuffled. Besides, the only difference is $(\{r^{\mathcal{R}}_{j,k}\}_{j\in[m],k\in[d]}, \mathcal{O}^F)$. $\{r^{\mathcal{R}}_{j,k}\}_{j\in[m],k\in[d]}$ are random secret shares in the real protocol according to the so-OPPRF functionality, while they are randomly sampled in the simulated view with the same distribution. Then, the function F is constructed in the same manner in the real and simulated executions. Therefore, the output by $\mathsf{Sim}_{\mathcal{R}}$ is indistinguishable from the real protocol. □

A.4. Proof of Fuzzy PSI for L_p distance

We give the detailed proof of Theorem 4, which is similar to that of Theorem 3.

Proof. **Corrupted sender.** We show the simulator $\mathsf{Sim}_{\mathcal{S}}(Q)$, which invokes the sub-simulator $\mathsf{Sim}_{\mathcal{S}}^{\mathsf{FMap}}$:

1) $\mathsf{Sim}_{\mathcal{S}}$ randomly samples ID_Q. $\mathsf{Sim}_{\mathcal{S}}$ invokes $\mathsf{Sim}_{\mathcal{S}}^{\mathsf{FMap}}(Q, \mathsf{ID}_Q)$ and appends the output to the view.
2) $\mathsf{Sim}_{\mathcal{S}}$ randomly samples $\{r^{\mathcal{S}}_{j,k}\}_{j\in[m],k\in[d]}$ and appends it to the view.
3) $\mathsf{Sim}_{\mathcal{S}}$ randomly samples $\{d^{\mathcal{S}}_{j,k}\}_{j\in[m],k\in[d]}$ and appends it to the view.

We show that the output by $\mathsf{Sim}_{\mathcal{S}}$ is indistinguishable from the real protocol. According to the security property of fuzzy mapping in Theorem 2, the simulated view of $\mathsf{Sim}_{\mathcal{S}}^{\mathsf{FMap}}$ is indistinguishable from the real execution. Besides, the only differences are $\{r^{\mathcal{S}}_{j,k}\}_{j\in[m],k\in[d]}$ and $\{d^{\mathcal{S}}_{j,k}\}_{j\in[m],k\in[d]}$. They are random secret shares in the real protocol according to the functionalities $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$ and $\mathcal{F}_{\mathsf{B2A}}$, while they are randomly sampled in the simulated view with the same distribution. Therefore, the output by $\mathsf{Sim}_{\mathcal{S}}$ is indistinguishable from the real protocol.

Corrupted receiver. We show the simulator $\mathsf{Sim}_{\mathcal{R}}(W, Z)$, which invokes the sub-simulator $\mathsf{Sim}_{\mathcal{R}}^{\mathsf{FMap}}$:

1) $\mathsf{Sim}_{\mathcal{R}}$ randomly samples ID_W. $\mathsf{Sim}_{\mathcal{R}}$ invokes $\mathsf{Sim}_{\mathcal{R}}^{\mathsf{FMap}}(W, \mathsf{ID}_W)$ and appends the output to the view.
2) $\mathsf{Sim}_{\mathcal{R}}$ randomly samples $\{r^{\mathcal{R}}_{j,k}\}_{j\in[m],k\in[d]}$ and samples a random function F such that $F(\mathsf{ID}_{w_i}\|k\|(w_{i,k}+t)) = |t|^p$ for $t \in [-\delta, \delta], i \in [n], k \in [d]$. $\mathsf{Sim}_{\mathcal{R}}$ appends $(\{r^{\mathcal{R}}_{j,k}\}_{j\in[m],k\in[d]}, \mathcal{O}^F)$ to the view.
3) $\mathsf{Sim}_{\mathcal{R}}$ randomly samples $\{d^{\mathcal{R}}_{j,k}\}_{j\in[m],k\in[d]}$ and appends them to the view.
4) $\mathsf{Sim}_{\mathcal{R}}$ uses $\perp$ to pad Z to m elements, randomly shuffles the set Z, and sets $b_j := 0$ if $z_j = 0$ and $b_j := 1$ otherwise for $j \in [m]$. $\mathsf{Sim}_{\mathcal{R}}$ appends $(\{b_j\}_{j\in[m]}, Z)$ to the view.

We show that the output by $\mathsf{Sim}_{\mathcal{R}}$ is indistinguishable from the real protocol. According to the security property of fuzzy mapping in Theorem 2, the simulated view of $\mathsf{Sim}_{\mathcal{R}}^{\mathsf{FMap}}$ is indistinguishable from the real execution. Moreover, the way $\mathcal{R}$ obtains the elements in Z is identical to the real execution since the elements in Z are randomly shuffled. Besides, the only difference is

Protocol $\Pi^{L_p}_{\mathsf{FPSI\text{-}Prefix}}$

Parameters: Distance threshold δ. Prefix parameters $\mu' = \log\delta$ and $\mu = 1 + \log\delta$.
Input: $\mathcal{S}$ inputs $Q = \{q_j\}_{j\in[m]} \in \mathbb{U}^{d\times m}$ and $\mathcal{R}$ inputs $W = \{w_i\}_{i\in[n]} \in \mathbb{U}^{d\times n}$.

Protocol:

1) $\mathcal{S}$ and $\mathcal{R}$ invoke sub-protocol $\Pi_{\mathsf{FMap\text{-}Prefix}}$, where $\mathcal{S}$ inputs Q and $\mathcal{R}$ inputs W. $\mathcal{S}$ obtains $\{\mathsf{ID}_{q_j}\}_{j\in[m]}$ and $\mathcal{R}$ obtains $\{\mathsf{ID}_{w_i}\}_{i\in[n]}$.
2) $\mathcal{R}$ invokes $\Pi_{\mathsf{getList_p}}(\{\mathsf{ID}_{w_i}\}_{i\in[n]}, \{w_i\}_{i\in[n]}, p)$ and gets $\mathsf{List_p}$.
3) For $j \in [m], k \in [d]$, $\mathcal{S}$ computes $\{q_{j,k,h}\}_{h\in[\mu]} := \mathsf{AllPrefix}(q_{j,k}, \mu)$.
4) $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$, where $\mathcal{R}$ inputs $\mathsf{List_p}$ and $\mathcal{S}$ inputs $\{\mathsf{ID}_{q_j}\|k\|\sigma\|q_{j,k,h}\}_{\sigma\in[0,1],j\in[m],k\in[d],h\in[\mu]}$. $\mathcal{S}$ receives $\{e^{\mathcal{S}}_{\sigma,j,k,h}\|v^{\mathcal{S}}_{\sigma,j,k,h}\}_{\sigma\in[0,1],j\in[m],k\in[d],h\in[\mu]}$ and $\mathcal{R}$ receives $\{e^{\mathcal{R}}_{\sigma,j,k,h}\|v^{\mathcal{R}}_{\sigma,j,k,h}\}_{\sigma\in[0,1],j\in[m],k\in[d],h\in[\mu]}$.
5) For $\sigma \in [0,1], j \in [m], k \in [d], h \in [\mu]$, $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{B2A}}$, where $\mathcal{S}$ inputs $v^{\mathcal{S}}_{\sigma,j,k,h}$ and $\mathcal{R}$ inputs $v^{\mathcal{R}}_{\sigma,j,k,h}$. $\mathcal{S}$ receives $s^{\mathcal{S}}_{\sigma,j,k,h}$ and $\mathcal{R}$ receives $s^{\mathcal{R}}_{\sigma,j,k,h}$.
6) For $\sigma \in [0,1], j \in [m], k \in [d]$, $\mathcal{S}$ and $\mathcal{R}$ invoke $\Pi_{\mathsf{getDistance_p}}$, where $\mathcal{S}$ inputs $(\{s^{\mathcal{S}}_{\sigma,j,k,h}\}_{h\in[\mu]}, q_{j,k}, \sigma)$, $\mathcal{R}$ inputs $\{s^{\mathcal{R}}_{\sigma,j,k,h}\}_{h\in[\mu]}$. $\mathcal{S}$ receives $\{d^{\mathcal{S}}_{\sigma,j,k,h}\}_{h\in[\mu]}$ and $\mathcal{R}$ receives $\{d^{\mathcal{R}}_{\sigma,j,k,h}\}_{h\in[\mu]}$.
7) For $j \in [m], k \in [d]$, $\mathcal{S}$ and $\mathcal{R}$ invoke $\mathcal{F}_{\mathsf{EQSel}}$, where $\mathcal{S}$ inputs $\{e^{\mathcal{S}}_{\sigma,j,k,h}, d^{\mathcal{S}}_{\sigma,j,k,h}\}_{\sigma\in[0,1],h\in[\mu]}$ and $\mathcal{R}$ inputs $\{e^{\mathcal{R}}_{\sigma,j,k,h}, d^{\mathcal{R}}_{\sigma,j,k,h}\}_{\sigma\in[0,1],h\in[\mu]}$. $\mathcal{S}$ receives $r^{\mathcal{S}}_{j,k}$ and $\mathcal{R}$ receives $r^{\mathcal{R}}_{j,k}$.
8) For $j \in [m]$, $\mathcal{S}$ computes $r^{\mathcal{S}}_j := \sum_{k\in[d]} r^{\mathcal{S}}_{j,k}$, and $\mathcal{R}$ computes $r^{\mathcal{R}}_j := \sum_{k\in[d]} r^{\mathcal{R}}_{j,k}$.
9) For $j \in [m]$, $\mathcal{S}$ and $\mathcal{R}$ invoke functionality $\mathcal{F}_{\mathsf{Interval}}$, where $\mathcal{S}$ inputs $r^{\mathcal{S}}_j$ and $\mathcal{R}$ inputs $r^{\mathcal{R}}_j$. $\mathcal{R}$ receives $b_j := \mathbf{1}(r^{\mathcal{S}}_j + r^{\mathcal{R}}_j \le \delta^p)$.
10) For $j \in [m]$, $\mathcal{R}$ and $\mathcal{S}$ invoke functionality $\mathcal{F}_{\mathsf{OT}}$, where $\mathcal{R}$ inputs b_j and $\mathcal{S}$ inputs $(\perp, q_j)$. $\mathcal{R}$ receives OT outputs z_j.
11) $\mathcal{R}$ outputs $Z := \{z_j \mid b_j = 1 \text{ for } j \in [m]\}$.

Figure 21: Protocol of fuzzy PSI with prefix optimization for L_p distance.

TABLE 4: The communication (MB) and running time (s) of fuzzy PSI in Section 6 under different network settings.

$m=n$	δ	d	L_∞				L_1				L_2			
			Comm.	10Gbps	1Gbps	100Mbps	Comm.	10Gbps	1Gbps	100Mbps	Comm.	10Gbps	1Gbps	100Mbps
256	16	4	7.3	0.3	0.4	4.8	7.8	0.3	0.5	5.2	7.8	0.4	0.6	5.2
		8	10.6	0.3	0.5	5.3	11.3	0.4	0.6	5.8	11.4	0.4	0.6	5.8
		16	17.1	0.4	0.7	5.8	18.4	0.5	0.8	6.2	18.4	0.5	0.7	6.2
	32	4	9.4	0.3	0.6	5.1	9.9	0.3	0.6	5.5	9.9	0.4	0.6	5.5
		8	14.7	0.4	0.7	5.7	15.4	0.5	0.7	6.4	15.5	0.4	0.7	6.4
		16	25.2	0.5	0.8	6.7	26.6	0.5	0.9	7.1	26.6	0.6	0.9	7.1
4096	16	4	60.5	1.3	1.8	10.4	66.1	1.4	2.0	11.3	66.3	1.4	2.0	11.3
		8	112.3	2.0	2.7	15.1	122.5	2.1	3.0	16.5	122.7	2.2	3.0	16.5
		16	215.8	3.0	4.4	25.0	235.3	3.5	5.0	27.5	235.5	3.5	4.9	27.6
	32	4	93.2	1.4	2.2	13.4	98.7	1.6	2.4	14.2	99.2	1.6	2.4	14.3
		8	177.6	2.2	3.5	21.4	187.8	2.5	3.8	22.7	188.2	2.5	3.9	22.7
		16	346.5	3.7	6.1	37.0	366.0	4.1	6.7	40.3	366.4	4.0	6.6	39.2
65536	16	4	908.7	17.2	22.4	96.9	990.9	17.9	24.4	105.7	998.8	18.6	24.5	106.4
		8	1737.6	25.0	36.8	172.1	1893.8	27.8	39.2	190.4	1901.8	27.8	39.5	188.3
		16	3396.0	42.0	65.0	319.4	3700.2	46.9	71.0	351.1	3708.2	47.7	71.3	355.5
	32	4	1432.0	19.0	28.5	143.0	1516.2	20.2	30.5	151.1	1526.1	20.4	31.1	155.6
		8	2784.7	28.9	49.8	271.1	2942.9	31.5	53.9	284.3	2952.8	31.7	53.8	286.6
		16	5491.5	49.7	92.0	511.3	5797.7	55.8	99.2	541.9	5807.7	55.1	99.8	541.4

TABLE 5: The communication (MB) and running time (s) of fuzzy PSI under different network settings. We fix $m = n = 2^{12}$ and $d = 8$. "Ours-prefix" denotes our optimized protocol in Section 7 and "Ours" denotes our non-optimized protocol in Section 6.

δ	Protocol	L_∞				L_1				L_2			
		Comm.	10Gbps	1Gbps	100Mbps	Comm.	10Gbps	1Gbps	100Mbps	Comm.	10Gbps	1Gbps	100Mbps
16	Ours	112.3	1.8	2.6	15.1	122.5	2.1	3.0	16.4	122.7	2.1	3.0	16.4
	Ours-prefix	184.8	5.4	6.3	27.4	252.2	7.0	8.4	35.6	385.2	9.7	11.6	49.3
64	Ours	308.3	2.6	5.1	33.7	318.5	2.9	5.4	35.5	319.4	2.9	5.6	35.3
	Ours-prefix	225.7	5.7	6.8	31.2	295.9	7.2	8.9	39.2	440.3	9.7	12.4	54.0
256	Ours	1093.1	7.2	15.1	104.5	1103.5	7.0	15.7	109.3	1104.6	7.3	15.5	106.3
	Ours-prefix	289.1	7.9	9.0	38.4	367.7	9.6	11.1	46.9	534.0	11.9	15.1	63.1
1024	Ours	4235.8	19.7	55.3	389.7	4246.5	20.3	55.9	386.5	4247.8	21.0	56.0	388.9
	Ours-prefix	366.2	10.0	9.7	42.8	456.6	10.9	13.1	55.5	682.6	15.1	18.9	78.7

$(\{r^{\mathcal{R}}_{j,k}\}_{j\in[m],k\in[d]}, \mathcal{O}^F, \{d^{\mathcal{R}}_{j,k}\}_{j\in[m],k\in[d]})$. $\{r^{\mathcal{R}}_{j,k}\}_{j\in[m],k\in[d]}$ and $\{d^{\mathcal{R}}_{j,k}\}_{j\in[m],k\in[d]}$ are random secret shares in the real protocol according to the functionalities $\mathcal{F}_{\mathsf{so\text{-}OPPRF}}$ and $\mathcal{F}_{\mathsf{B2A}}$, while they are randomly sampled in the simulated view with the same distribution. Then, the function F is constructed in the same manner in the real and simulated executions. Therefore, the output by $\mathsf{Sim}_{\mathcal{R}}$ is indistinguishable from the real protocol.

□

Appendix B. Details of Fuzzy PSI with Prefix Optimizations

B.1. Other Building Blocks

We present the functionalities of $\mathcal{F}_{\mathsf{MUX}}$ and $\mathcal{F}_{\mathsf{ssPEQT}}$ in Figure 17 and Figure 18, respectively.

B.2. Sub-protocols for $\Pi^{L_p}_{\mathsf{FPSI\text{-}Prefix}}$

We present the sub-protocols of our optimized fuzzy PSI protocol $\Pi^{L_p}_{\mathsf{FPSI\text{-}Prefix}}$ in Figure 19 and Figure 20.

B.3. Optimized Fuzzy PSI for L_p Distance

With prefix optimization, it is non-trivial to convert L_∞ distance to L_p distance. As the interval is broken into prefixes, we cannot directly compute the distance for every point within the interval; instead, we only know the distance associated with each prefix. Dang et al. [16] addressed this issue by decomposing the distance into two parts. Specifically, the distance between points x and y can be expressed as $|x - x^*| + |x^* - y|$, where x^* represents either the upper or lower bound of the prefix that x falls into. The value $|x - x^*|$ can be computed by the sender, while $|x^* - y|$ can be computed by the receiver, and the two parties then jointly compute the sum of these two distances.

We adopt a similar method but realize the same functionality using more efficient MPC primitives to keep consistent with our framework, whereas Dang et al. [16] heavily rely on Paillier encryption. Our modified construction for the key-value pairs is presented in Figure 19 and Figure 20.

The detailed protocol for $\Pi^{L_p}_{\mathsf{FPSI\text{-}Prefix}}$ is shown in Figure 21 and its security proof follows Appendix A.4.

Appendix C. Performance under Different Network Settings

We provide the performance of our fuzzy PSI protocols and the variants with prefix optimizations under different network settings in Table 4 and Table 5. We set the network to 1 Gbps bandwidth with 40 ms latency and 100 Mbps with 80 ms latency.

Appendix D.
Meta-Review

The following meta-review was prepared by the program committee for the 2026 IEEE Symposium on Security and Privacy (S&P) as part of the review process as detailed in the call for papers.

D.1. Summary

The paper improves the efficiency of a known construction of Fuzzy-PSI where two parties match similar but not necessarily identical elements. It contributes the primitive of an Oblivious Programmable Pseudo-Random Function (OPPRF) with secret shared outputs.

D.2. Scientific Contributions

- Provides a Valuable Step Forward in an Established Field.

D.3. Reasons for Acceptance

1) The paper provides a valuable step forward in the established field of Fuzzy-PSI. It significantly improves the performance of a known construction by contributing a new primitive - an OPPRF with secret shared outputs - which may be of independent interest.