




# Fuzzy Private Set Intersection with Large Hyperballs

Aron van Baarsen<sup>1,2</sup> and Sihang Pu<sup>3</sup>(✉) 

<sup>1</sup> Cryptology Group, CWI, Amsterdam, The Netherlands  
aronvanbaarsen@gmail.com

<sup>2</sup> Mathematical Institute, Leiden University, Leiden, The Netherlands

<sup>3</sup> CISPA Helmholtz Center for Information Security, Saarbrücken, Germany  
sihang.pu@gmail.com

**Abstract.** Traditional private set intersection (PSI) involves a receiver and a sender holding sets  $X$  and  $Y$ , respectively, with the receiver learning only the intersection  $X \cap Y$ . We turn our attention to its fuzzy variant, where the receiver holds  $|X|$  hyperballs of radius  $\delta$  in a metric space and the sender has  $|Y|$  points. Representing the hyperballs by their center, the receiver learns the points  $x \in X$  for which there exists  $y \in Y$  such that  $\text{dist}(x, y) \leq \delta$  with respect to some distance metric. Previous approaches either require general-purpose multi-party computation (MPC) techniques like garbled circuits or fully homomorphic encryption (FHE), leak details about the sender’s precise inputs, support limited distance metrics, or scale poorly with the hyperballs’ volume.

This work presents the first black-box construction for fuzzy PSI (including other variants such as PSI cardinality, labeled PSI, and circuit PSI), which can handle polynomially large radius and dimension (i.e., a potentially exponentially large volume) in two interaction messages, supporting general  $L_{p \in [1, \infty]}$  distance, without relying on garbled circuits or FHE. The protocol excels in both asymptotic and concrete efficiency compared to existing works. For security, we solely rely on the assumption that the Decisional Diffie-Hellman (DDH) holds in the random oracle model.

## 1 Introduction

Private set intersection (PSI) is a cryptographic primitive that allows two parties to compute the intersection  $X \cap Y$  of their private datasets  $X$  and  $Y$ , without revealing any information about items not in the intersection. The first PSI protocol is often dated back to Meadows [30] and many modern protocols still have the same structure using an oblivious pseudorandom function (OPRF) [28, 35, 36]. Recent PSI protocols are very practical and can for example compute the intersection of sets of  $2^{20}$  elements in  $\approx 0.37$  s [35]. Many richer PSI functionalities have also been explored, such as: *PSI cardinality* [16, 18, 27], where only the cardinality of the intersection is revealed; *labeled PSI* [10, 13, 14], which allows

---

A. van Baarsen—Research partially funded by NWO/TKI Grant 628.009.014.

© International Association for Cryptologic Research 2024

M. Joye and G. Leander (Eds.): EUROCRYPT 2024, LNCS 14655, pp. 340–369, 2024.

[https://doi.org/10.1007/978-3-031-58740-5\\_12](https://doi.org/10.1007/978-3-031-58740-5_12)

the parties to learn labels associated to the items in the intersection; *circuit PSI* [24, 34, 36], which only reveals secret shares of the intersection and allows the parties to securely evaluate any function on the intersection.

Recently Garimella et al. [20, 21] introduced the concept of *structure-aware PSI*, where the receiver’s input set has some publicly known structure. For example, the receiver holds  $N$  balls of radius  $\delta$  and dimension  $d$  and the sender holds a set of  $M$  points, and the sender learns which of the sender’s points lie within one of their balls. This special case is often referred to as *fuzzy PSI* and is the focus of our work. Using a standard PSI protocol for this task leads to a rather inefficient solution since the communication and computation complexity usually scale at least linearly in the cardinality of the input sets, i.e., the total volume of the balls  $N \cdot \delta^d$ . Garimella et al. can overcome this barrier in terms of communication in the semi-honest [20] as well as in the malicious [21] setting. However, the receiver’s computation is still proportional to the total volume of the input balls, which makes their protocols scale poorly with the dimension  $d$ . Moreover, their protocols are limited to the  $L_\infty$  and  $L_1$ <sup>1</sup> distance and only realize a *standard PSI* functionality, where the receiver learns exactly which of the sender’s points lie in the intersection. Other works are either limited to the Hamming distance [37], Hamming and  $L_2$  distance [26] or Hamming distance and one-dimensional  $L_1$  distance [8], and often require heavy machinery or yield non-negligible correctness error.

In this work, we present fuzzy PSI protocols in the semi-honest setting for general  $L_\infty$  and  $L_p$  distance with  $p \in [1, \infty)$ , and present several optimized variants for low as well as high dimensions. Notably, the communication as well as computation complexity of our high-dimension protocols scales linearly or quadratically with the dimension  $d$ . We moreover extend our protocols to various richer fuzzy PSI functionalities including PSI cardinality, labeled PSI, PSI with sender privacy, and circuit PSI. Our protocols have comparable performance to [20] in the low-dimensional setting and significantly outperform other approaches when the dimension increases. Finally, our protocols rely only on the decisional Diffie-Hellman (DDH) assumption.

### 1.1 Our Contributions

**Fuzzy Matching.** The main building block for our fuzzy PSI constructions is a fuzzy matching protocol, which on input a point  $\mathbf{w} \in \mathbb{Z}^d$  from the receiver and a point  $\mathbf{q} \in \mathbb{Z}^d$  from the sender, outputs 1 to the receiver if  $\text{dist}(\mathbf{w}, \mathbf{q}) \leq \delta$  and 0 otherwise. Here  $\text{dist}$  can either be  $L_\infty$  distance or  $L_p$  distance for  $p \in [1, \infty)$ . It results in a two-message protocol for  $L_\infty$  distance with communication complexity  $O(\delta d)$ , computation complexity  $O(\delta d)$  for the receiver and  $O(d)$  for the sender when  $\text{dist} = L_\infty$ ; For  $L_p$  distance, it has communication complexity  $O(\delta d + \delta^p)$ , computation complexity  $O(\delta d)$  for the receiver, and  $O(d + \delta^p)$  for the sender.

---

<sup>1</sup> The overhead of  $L_1$  balls would be  $\frac{2^d}{d}$  times larger than that of  $L_\infty$  balls in their protocols.

**Table 1.** Asymptotic complexities of fuzzy PSI protocols, where the receiver holds  $N$  hyperballs of radius  $\delta$  and the sender holds  $M$  points in  $\mathbb{Z}^d$ .  $\rho \leq 1/c$  is a parameter to the LSH scheme if the receiver’s points are distance  $> c\delta$  apart. We ignore multiplicative factors of the computational security parameter  $\lambda$  and statistical parameter  $\kappa$ .

Setting [protocol]	Communication	Comp. (receiver)	Comp. (sender)
$L_\infty$ trad. PSI [35]	$O((2\delta)^d N + M)$	$O((2\delta)^d N)$	$O((2\delta)^d N + M)$
$> 2\delta$ [ours]	$\mathbf{O}(\delta dN + 2^d M)$	$\mathbf{O}(\delta dN + 2^d M)$	$\mathbf{O}(2^d dM)$
$> 2\delta$ [20]	$O((4 \log \delta)^d N + M)$	$O((2\delta)^d N)$	$O((2 \log \delta)^d M)$
$> 4\delta$ [ours]	$O(\delta 2^d dN + M)$	$\mathbf{O}(\delta 2^d dN + M)$	$\mathbf{O}(dM)$
$> 4\delta$ [20]	$O(2^d dN \log \delta + M)$	$O((2\delta)^d N)$	$O(dM \log \delta)$
$> 8\delta$ [21]	$O(dN \log \delta + M)$	$O((2\delta)^d N)$	$O(dM \log \delta)$
$\exists$ disj. proj. [ours]	$O((\delta d)^2 N + M)$	$O((\delta d)^2 N + M)$	$O(d^2 M)$
$\forall$ disj. proj. [20]	$O(dN \log \delta + M)$	$O((2\delta)^d N)$	$O(dM \log \delta)$
$L_p$ $> 2\delta(d^{1/p} + 1)$ [ours]	$O(\delta 2^d dN + \delta^p M)$	$O(\delta 2^d dN + M)$	$O((d + \delta^p)M)$
LSH [ours]	$O(\delta dN^{1+p} + \delta^p MN^\rho \log N)$	$O(\delta dN^{1+p} + MN^\rho \log N)$	$O((d + \delta^p)MN^\rho \log N)$

**Fuzzy PSI in Low-Dimensions.** Using a fuzzy matching protocol we can trivially obtain a fuzzy PSI protocol by letting the sender and receiver run the fuzzy matching protocol for every combination of inputs, but this leads to an undesirable  $N \cdot M$  blowup in communication and computation complexity. To circumvent this blowup for a low dimension  $d$ , we develop a new spatial hashing technique for disjoint  $L_\infty$  balls which incurs only a  $O(2^d)$  factor to the receiver’s communication and sender’s computational complexity. To support  $L_p$  balls, we extend the “shattering” idea from [20] to generalized  $L_p$  setting. The asymptotic complexities are given in Table 1. It is worth noting that, unlike to [20, 21], the computation complexity of our protocols scale *sublinearly* to the volume of balls.

**Fuzzy PSI in High-Dimensions.** Unfortunately, the above spatial hashing approaches still yield a  $2^d$  factor in the communication and computation complexities, which becomes prohibitive for large dimensions  $d$ . The earlier work [20] proposes a protocol that can overcome this factor, for communication costs, in the  $L_\infty$  setting under the *globally disjoint* assumption that the projections  $[w_{k,i} - \delta, w_{k,i} + \delta]$  of the sender’s balls  $k \in [N]$  are disjoint *for all* dimensions  $i \in [d]$ , which the authors themselves mention is a somewhat artificial assumption. We present a two-message protocol with comparable communication and much lower computation complexity under a milder assumption that for each  $k \in [N]$  *there exists* a dimension  $i \in [d]$  where the projection is disjoint from all other  $k' \in [N]$ , namely, not necessary to be globally disjoint. We argue that this is a more realistic assumption since points in high dimensions tend to be sparser and show that it is satisfied with a high probability if the points  $\mathbf{w}_k$  are uniformly distributed.

We moreover present a two-message protocol in the  $L_p$  setting which can circumvent this exponential factor in the dimension  $d$ , while achieving sub-quadratic complexity in the number of inputs. The key idea of this protocol

is to use locality-sensitive hashing (LSH) to perform a coarse mapping such that points close to each other end up in the same bucket with high probability, and subsequently use our fuzzy matching protocol for  $L_p$  distance to compare the items in each bucket. See Table 1 for the asymptotic complexities of our protocols.

**Extensions to Broader Functionalities.** By default, all our protocols except for the LSH-based protocol realize the stricter PSI functionality where the receiver only learns how many of the sender’s points lie close to any of the receiver’s points, which we call *PSI cardinality* (PSI-CA). Earlier works [20, 21] realize the functionality where the receiver learns exactly which of the sender’s points are in the intersection. We refer to this functionality as *standard PSI*.

For all of our protocols discussed above, except for the LSH-based protocol, we show that we can extend them to realize the following functionalities: *standard PSI*; *PSI with sender privacy* (PSI-SP), where the receiver only learns which of the receiver’s balls are in the intersection; *labeled PSI*, where the receiver only learns some label associated to the sender’s points in the intersection; *circuit PSI*, where the parties only learn secret shares of the intersection and optional data associated to each input point, which they can use as the input to any secure follow-up computation. We can realize these extensions without increasing the asymptotic complexities of the protocols and without needing to introduce additional computational assumptions. With the only exception that for the circuit PSI extension we need a generic MPC functionality to compute a secure comparison circuit at the end of the protocol, which is common for traditional (non-fuzzy) circuit PSI protocols [34, 36].

**Performance.** Our experimental results demonstrate that it requires only 1.2 GB bandwidth and 432 s in total to complete a standard fuzzy PSI protocol when parties have thousands of  $L_\infty$  balls and millions of points in a 5-dimensional space. As a comparison, prior works need  $\gg 4300$  s (conservative estimate).

## 1.2 Related Work

Traditional PSI protocols have become very efficient [9, 28, 35], but are optimized for the setting where the parties’ input sets have approximately the same size, and their communication and computation costs scale linearly with the input size. This leads to an inefficient fuzzy PSI protocol since the receiver’s input size is  $N \cdot \delta^d$  when the receiver holds  $N$  hyperballs of dimension  $d$  and radius  $\delta$ . Asymmetric (or unbalanced) PSI protocols [1, 10, 11, 14] target the setting where one party’s set is much larger than the other’s and can achieve communication complexity sublinear in the large set size, but  $O(\sqrt{N} \cdot \delta^{d/2})$  computational complexity, using fully homomorphic encryption [14]. For traditional PSI there exist many efficient protocols realizing richer functionalities such as PSI cardinality [16, 27], labeled PSI [10, 14] and circuit PSI [24, 34, 36], but all of these suffer from the same limitations as discussed above when applied to the fuzzy PSI setting.

There exists another line of works concerning threshold PSI [3, 7, 22, 23], where the fuzziness is measured by the number of *exact* matches between items.

Secure fuzzy matching was introduced by Freedman et al. [18] as the problem of identifying when two tuples have a Hamming distance below a certain threshold. They propose a protocol based on additively homomorphic encryption and polynomial interpolation, which was later shown to be insecure [12]. Follow-up works focus on the Hamming distance as well and use similar oblivious polynomial evaluation techniques [12, 38]. Indyk and Woodruff [26] construct a fuzzy PSI protocol for  $L_2$  and Hamming distance using garbled circuits. Uzun et al. [37] give a protocol for fuzzy labeled PSI for Hamming distance using garbled circuits and fully homomorphic encryption. Chakraborti et al. [8] propose a fuzzy PSI protocol for Hamming distance based on additively homomorphic encryption and vector oblivious linear evaluation (VOLE), which has a non-negligible false positive rate. They moreover present a protocol for one-dimensional  $L_1$  distance, which can be constructed using any  $O(N)$  communication PSI protocol for sets of size  $N$ , and has resulting communication complexity  $O(N \log \delta)$  [8]. It is an interesting question whether their techniques can be extended to higher dimensions. Since the focus of our work is to construct fuzzy PSI protocols for general  $L_p$  and  $L_\infty$  distances, general dimension  $d$ , and with negligible error rate, it is not possible to make a meaningful comparison with these works.

Garimella et al. [20, 21] initiated the study of structure-aware PSI, which covers fuzzy PSI as a special case. They introduce the definition of weak boolean function secret sharing (bFSS) for set membership testing and give a general protocol for structure-aware PSI from bFSS. They develop several new bFSS techniques, focusing on the case where the input set is the union of  $N$  balls of radius  $\delta$  with respect to the  $L_\infty$  norm in  $d$ -dimensional space, which results in a fuzzy PSI protocol as the ones we concern ourselves with in this work. The techniques used in their protocols are fundamentally different from ours, except that we use similar spatial hashing techniques to obtain efficient fuzzy matching protocols in the low-dimensional setting. Moreover, their protocols are limited to the  $L_\infty$  and  $L_1$  distance setting and only realize the standard PSI functionality where the receiver learns the exact sender's points in the intersection. Finally, the receiver's computational complexity in their protocols scales as  $O((2\delta)^d N)$ , which makes them unsuitable in the high-dimensional setting. See Table 1 for a more detailed comparison of communication and computational complexities.

### 1.3 Applications

**Private Proximity Detection.** There exist certain contexts where individuals need to know the proximity of others for varying purposes: In the realm of contact tracing, where individuals may seek to determine if they are in the vicinity of an infected person; Within the scope of ride-sharing platforms, users might wish to identify available vehicles in their surroundings. In both scenarios, the privacy of all involved parties should be preserved and fuzzy PSI protocols provide a direct solution to this problem.

**Biometric and Password Identification.** Fuzzy matching could also be useful in authentication or identification scenarios. Notable applications of this technique can be observed in the matching of similar passwords to enhance usability or security. A case in point is Facebook’s authentication protocol, which auto-corrects the capitalization of the initial character in passwords [31]. Similarly, it can be useful to check if a user’s password is similar to a leaked password [33]. Furthermore, fuzzy matching can be employed to match biometric data, such as fingerprint and iris scans, thereby facilitating a blend of convenience and security [17]. In general, a fuzzy unbalanced PSI protocol is more useful since the server usually holds a large database of clients’ passwords (or biometric samples).

**Illegal Content Detection.** Recently, Bartusek et al. [4] introduced the study of illegal content detection within the framework of end-to-end secure messaging, focusing particularly on the detection of child sexual abuse material, encompassing photographs and videos. Central to their protocol is a two-message PSI protocol, wherein the initial message is reusable and published once for the receiver’s database. After this, the computational overhead for both parties is rendered independent of the database size. The research notably leverages Apple’s PSI protocol [5], which, while only facilitating exact matches, serves its purpose effectively. Ideally, matching should be sufficiently fuzzy to ensure that illegal images remain detectable even following rotation or mild post-processing. Our fuzzy PSI constructions, encapsulated within two-round protocols and featuring a reusable initial message, may find potential applicability in such contexts.

## 2 Technical Overview

Before heading for the details of our fuzzy matching and PSI protocols, let us start by discussing a standard PSI protocol proposed by Apple [5].

### 2.1 Recap: Apple’s PSI Protocol

We simplify Apple’s PSI protocol to the basic setting where the receiver holds a set  $W := \{w_1, \dots, w_n\}$ , the sender holds an item  $q$ , and the receiver wants to learn  $q$  if  $q \in W$  and nothing otherwise. Their main idea is a novel usage of random self-reduction of DDH tuples from Naor and Reingold [32] in PSI contexts. Given a cyclic group  $\mathbb{G} := \langle g \rangle$  of prime order  $p$ , the tuple  $(g, h, h_1, h_2)$  can be re-randomized into  $(g, h, u, v)$  such that  $u, v$  are uniformly random over  $\mathbb{G}$  as long as  $(g, h, h_1, h_2)$  is *not* a well-formed DDH tuple (i.e., there is no  $s \in \mathbb{Z}_p^*$  to satisfy  $g^s = h \wedge h_1^s = h_2$ ). Otherwise, both  $(g, h, h_1, h_2)$  and the re-randomized tuple  $(g, h, u, v)$  are valid DDH tuples. This re-randomization basically utilizes two random coins  $a, b \leftarrow_s \mathbb{Z}_p^*$  to output

$$(u := g^a h_1^b, v := h^a h_2^b).$$

Now to obtain a PSI protocol, the receiver could sample  $s \leftarrow_{\$} \mathbb{Z}_p^*$  and publish

$$(g, h := g^s, H(w_1)^s, \dots, H(w_n)^s),$$

where  $H$  is a hash-to-group function. Then the sender returns pairs

$$(u_i, \text{ct}_i := \text{Enc}_{v_i}(q))_{i \in [n]},$$

where  $(u_i, v_i)$  is the re-randomization output for each tuple  $(g, h, H(q), H(w_i)^s)$ , and  $\text{Enc}$  is some symmetric-key encryption scheme (e.g., a one-time pad). The receiver can try to decrypt each  $\text{ct}_i$  using the key  $u_i^s$  to learn  $q$ . For the sender’s privacy, the random self-reduction of DDH tuples guarantees that when  $q \neq w_i$ , the secret key  $v_i$  is uniformly random from the receiver’s view and thus  $q$  is hidden according to the security of this symmetric-key encryption. For the receiver’s privacy,  $(H(w_1)^s, \dots, H(w_n)^s)$  is pseudorandom according to the generalized DDH assumption when  $H$  is modelled as a random oracle.

### 2.2 Fuzzy Matching for Infinity Distance

Our crucial observation is that the above approach can be naturally applied in fuzzy matching protocols where the receiver holds a point  $\mathbf{w} \in \mathbb{Z}^d$  in a  $d$ -dimensional space, the sender holds a point  $\mathbf{q} \in \mathbb{Z}^d$ , and the receiver learns if  $\text{dist}(\mathbf{w}, \mathbf{q}) \leq \delta$ . Here,  $\delta$  is the maximal allowed distance between  $\mathbf{w}$  and  $\mathbf{q}$ . For the moment, let us focus on the simplest case where the distance is calculated over  $L_\infty$ , which means, the receiver gets 1 if

$$\forall i \in [d] : w_i - \delta \leq q_i \leq w_i + \delta,$$

and gets 0 otherwise. This problem is equivalent to the following: The receiver holds  $d$  sets  $\{W_1, \dots, W_d\}$  where  $W_i := \{w_i - \delta, \dots, w_i + \delta\}$ , the sender holds  $d$  items  $\{q_1, \dots, q_d\}$ , and they want to run a membership test for each dimension simultaneously, without leaking the results for individual dimensions. Though the receiver can publish  $H(w_i + j)^s$  for each  $i \in [d], j \in [-\delta, +\delta]$  as above, the sender has to use random self-reduction for each possible match, which yields too much communication and computation effort for the sender. Namely, the entire volume of a  $d$ -dimensional  $\delta$ -radius ball  $O((2\delta + 1)^d)$ .

**Reducing the Complexity.** There is a standard trick to significantly reduce the complexity by using an oblivious key-value store (OKVS). Recall that an OKVS [19] will encode a key-value list  $\{(\text{key}_j, \text{val}_j)\}_{j \in [n]}$  into a data structure  $E$ , such that decoding with a correct  $\text{key}_*$  returns the corresponding  $\text{val}_*$ , where the encoding time scales linearly to the list size and decoding a single key takes only a constant number of operations. So the above protocol can be improved as follows:

- 1) The receiver publishes  $(g, h, E_i \leftarrow \text{Encode}(\{(w_i + j, H(w_i + j)^s)\}_{j \in [-\delta, +\delta]}))$  for each  $i \in [d]$ ;

- 2) The sender retrieves  $h_i \leftarrow \text{Decode}(E_i, q_i)$  for each  $i \in [d]$  and sends the rerandomized tuple  $(u := g^a \prod_{i=1}^d H(q_i)^b, v := h^a \prod_{i=1}^d h_i^b)$ , where  $a, b \leftarrow_{\$} \mathbb{Z}_p^*$ , to the receiver;
- 3) The receiver checks if  $(g, h, u, v)$  is a valid DDH tuple.

The protocol is correct when  $\text{dist}(\mathbf{q}, \mathbf{w}) \leq \delta$ , according to the correctness of the underlying OKVS scheme, which says that decoding the structure  $E_i$  with a correct encoding key  $q_i$  will return the encoded value  $h_i := H(q_i)^s$ ; When  $\text{dist}(\mathbf{q}, \mathbf{w}) > \delta$ , we typically need to rely on the independence property of OKVS, which says that decoding with a non-encoded key will yield a uniformly random result. Therefore, in this case, there exists at least one  $h_{i^*}$  that is uniformly random; hence  $(g, h, H(q_{i^*}), h_{i^*})$  is not a DDH tuple except with negligible probability. The sender’s privacy can be established as before from the random self-reduction of DDH tuples. To argue the receiver’s privacy, we rely on the obliviousness property of the OKVS, namely, the encoded keys  $\{w_i + j\}$  are completely hidden as long as the encoded values  $\{H(w_i + j)^s\}$  are uniformly random. Since  $(h, H(w_i + j), H(w_i + j)^s)$  is pseudorandom by the DDH assumption, then according to the obliviousness of OKVS, the receiver’s message can be simulated by encoding random key-value pairs.

Note that our real construction shown in Sect. 5.1, is slightly different from what we described here. We encode the OKVS “over the exponent” to reduce heavy public-key operations over  $\mathbb{G}$  because our encoded values are pseudorandom over a *structured* group  $\mathbb{G}$  (i.e., the elliptic curves).

So far, we have obtained a two-message fuzzy matching protocol for  $L_\infty$  distance, with  $O(d\delta)$  communication and computation complexity.

### 2.3 Generalized Distance Functions

When the distance function is calculated in  $L_p$ , the receiver would get 1 if

$$\text{dist}_p(\mathbf{w}, \mathbf{q}) := \left( \sum_{i=1}^d |w_i - q_i|^p \right)^{1/p} \leq \delta,$$

and 0 otherwise. To make the problem easier, we consider the  $p$ -powered  $L_p$  distance, namely, we check if  $\sum_{i=1}^d |w_i - q_i|^p \leq \delta^p$ . Thanks to the homomorphism of DDH tuples, the sender can homomorphically evaluate the distance function. Moreover, since an  $L_{p \geq 1}$  ball must be confined in an  $L_\infty$  ball, namely,  $|w_i - q_i| \leq \delta$  for any  $i \in [d]$  if  $\text{dist}_p(\mathbf{w}, \mathbf{q}) \leq \delta$ , the protocol could work as follows:

- 1) The receiver publishes

$$\left( g, h, E_i \leftarrow \text{Encode} \left( \left\{ (w_i + j, H(w_i + j)^s \cdot g^{|j|^p}) \right\}_{j \in [-\delta: \delta]} \right) \right),$$

for each  $i \in [d]$ ;



2) The sender retrieves  $h_i \leftarrow \text{Decode}(E_i, q_i)$  for each  $i \in [d]$ , and computes

$$\left( u := g^a \prod_{i=1}^d H(q_i)^b, v := h^a \prod_{i=1}^d h_i^b \right),$$

for random  $a, b \leftarrow_s \mathbb{Z}_p^*$ ;

3) The sender generates a list  $\text{list} := \{g^{b \cdot j}\}_{j \in [0:\delta^p]}$  and outputs  $(u, v, \text{list})$ ;

4) The receiver checks if there is any  $x \in \text{list}$  such that  $v = u^s \cdot x$ .

Denote  $t := \text{dist}_p(\mathbf{w}, \mathbf{q})$ . If  $\forall i \in [d], |w_i - q_i| \leq \delta$ , then the correctness holds naturally since each retrieved  $h_i := H(q_i)^s \cdot g^{|w_i - q_i|^p}$ , implying that

$$\frac{v}{u^s} = g^{b \cdot t^p},$$

which would be included in  $\text{list}$  if and only if  $t \leq \delta^2$ . On the other hand, if there exists  $i^* \in [d]$  such that  $|w_{i^*} - q_{i^*}| > \delta$ , then according to the independence property of OKVS the decoded  $h_{i^*}$  as well as  $v$  would be uniformly random over  $\mathbb{G}$ , such that  $v/u^s \in \text{list}$  with only negligible probability.

**Subtle Issues and the Fix.** The receiver's privacy is almost the same as before, relying on the generalized DDH assumption and the obliviousness of OKVS. It is a little bit subtle to argue the sender's privacy: Currently,  $\text{list}$  would leak information on the sender's input. Precisely, given  $(u, v, \text{list})$ , the receiver could check, for example, if  $\frac{v}{u^s \cdot g^b} \in \text{list}$  to learn if  $t^p = \delta^p + 1$  or not, since  $g^b \in \text{list}$ . Moreover, even in the case that  $t \leq \delta$ , the receiver could still deduce the *exact*  $t$  by checking which index is matched. The latter can be solved by shuffling the list, so we focus on the former issue. One approach is to hash each list item as  $\text{list} := \{H'(g^{b \cdot j})\}_{j \in [0:\delta^p]}$ . By modeling  $H' : \mathbb{G} \mapsto \{0, 1\}^*$  as a random oracle, the group structure is erased and the adversary cannot utilize  $g^{b \cdot j}$  anymore. However, the issue still exists since the adversary could check if  $H'((\frac{v}{u^s})^{1/\alpha}) \in \text{list}$  to learn if  $t^p \in \{0, \alpha, 2\alpha, \dots, \delta^p \alpha\}$  for any  $\alpha$ . Therefore, we have to apply a random linear function over  $t^p$  to make sure that  $\frac{v}{u^s} = g^{b \cdot t^p + c}$  where  $b, c$  are random scalars. The details can be found in Sect. 5.2.

Regarding the complexity, the communication and computation are increased by an additive term  $O(\delta^p)$  from the infinity distance setting.

## 2.4 Fuzzy PSI in Low Dimensions

For the moment, let us consider the fuzzy PSI *cardinality* problem, where the receiver holds a union of  $d$ -dimensional balls of radius  $\delta$  represented by their centers  $\{\mathbf{w}_1, \dots, \mathbf{w}_N\}$ , the sender holds a set of points  $\{\mathbf{q}_1, \dots, \mathbf{q}_M\}$  in the same space, and the receiver learns *the number of* sender's points located inside the balls. When the dimension  $d$  of the space is low, e.g.,  $O(\log(\lambda))$ , we can exploit

<sup>2</sup> Assuming the group order  $p$  is large enough that  $t^p < p$ .

the geometric structure of the space to efficiently match balls and points to avoid the quadratic blowup mentioned in the introduction. The high-level idea is to tile the entire space by  $d$ -dimensional hypercubes of side-length  $2\delta$  (also called *cells*, together a *grid*), then the receiver can encode a ball (represented by its center  $\mathbf{w}_i$ ) in a way that the sender can efficiently match it with a point  $\mathbf{q}_j$ , without enumerating all balls. After that, both parties can run a fuzzy matching protocol between  $\mathbf{w}_i$  and  $\mathbf{q}_j$  as before.

The idea of Garimella et al. [20] is to “shatter” each receiver’s ball into its intersected cells, however, to guarantee each cell is intersected with a single ball (otherwise collisions appear during encoding an OKVS<sup>3</sup>), the receiver’s balls typically need to be at least  $4\delta$  apart from each other. To tackle the case of disjoint balls, the authors improved their techniques [21] by observing that each grid cell can only contain the center of a single receiver’s ball. Thus, the receiver could encode the identifier of each cell which contains a ball center, and the sender can try to decode the OKVS by iterating over all neighborhood cells<sup>4</sup> surrounding its point. This approach yields a  $O(3^d)$  factor for the sender’s computation and communication costs: Given a point  $\mathbf{q}$ , the center of any  $L_\infty$  ball intersected with  $\mathbf{q}$  is located in at most  $3^d$  cells surrounding the cell containing  $\mathbf{q}$ .

**New Spatial Hashing Ideas.** Here we provide a new hashing technique to reduce this blowup from  $3^d$  to  $2^d$ . Note that the  $3^d$  factor comes from the fact that the entire neighborhood of the point  $\mathbf{q}$  is too large (i.e., a hypercube of side-length  $6\delta$ ), but we only need to care about the neighbor cells that intersected with the receiver’s balls already. Specifically, if the grid is set properly, an  $L_\infty$  ball will intersect *exactly*  $2^d$  cells, which constitute a hypercube of side-length  $4\delta$ , denoted as a *block*. Our crucial observation is that each block is *unique* for each disjoint ball, i.e., two disjoint balls must be associated with different blocks, as detailed in Lemma 6. Given this, the receiver could encode the identifier of each block, and the sender would decode by iterating all potential blocks. There are in total  $2^d$  possible blocks for each sender’s point due to each block being comprised of  $2^d$  cells and each cell contains at most a single ball’s center.

**Compatible with  $L_p$  Balls.** Though we only considered  $L_\infty$  balls so far, we can generalize the “shattering” idea from [20] to  $L_p$  balls as well. We still tile the space with hypercubes, but we show that as long as  $L_p$  balls are at least  $2\delta(d^{1/p} + 1)$  apart from each other, then each grid cell intersects at most one  $L_p$  ball, as detailed in Lemma 5. Particularly, when  $p = \infty$ ,  $2\delta(d^{1/p} + 1)$  degrades to the original  $4\delta$ . Combining it with our fuzzy matching protocols for  $L_p$  distance, we immediately obtain fuzzy PSI for precise  $L_p$  distance (i.e., without approximation or metric embedding). One important step different from the  $L_\infty$  setting is to pad the key-value list to size  $2^d N$  with random pairs since an  $L_p$  ball could intersect with a various number of cells. Otherwise, the receiver’s privacy would be compromised.

<sup>3</sup> Note that this is not a problem in their setting as they use the function secret sharing (FSS) to handle each grid cell.

<sup>4</sup> The neighborhood is a hypercube of side-length  $6\delta$ .

### 2.5 Extending to High Dimensions

To overcome the  $2^d$  factor in complexities, we first focus our attention on  $L_\infty$  distances: Ideally, if the receiver’s balls are *globally disjoint* on every dimension, namely, the projection of the balls on each dimension never overlaps, then the “collision” issue mentioned above would disappear. In this way, for each dimension  $i \in [d]$ , the receiver could encode the OKVS as

$$E_i \leftarrow \text{Encode}\left(\left\{\left(w_{k,i} + j, \quad H(w_{k,i} + j)^s\right)\right\}_{j \in [-\delta, +\delta]}\right),$$

where  $w_{k,i}$  is the projection of the ball center  $\mathbf{w}_k$  on dimension  $i$ . The sender just behaves the same as in Sect. 2.2. This approach results in  $O(\delta dN + M)$  communication and computation costs. However, as stated in [20], this ideal setting is somewhat artificial and unrealistic.

**Weaker Assumptions by Leveraging Dummy OKVS Instances.** After taking a closer look at this approach, we realized that the global disjointness is not necessary to be satisfied on *every* dimension, as we actually could tolerate some collisions. Specifically, the value  $h_i$  decoded from  $E_i$  for some point  $\mathbf{q}$  (lying in one of the receiver’s balls) would constitute a tuple  $(g, h, H(q_i), h_i)$ . However, this tuple does not necessarily need to be a DDH tuple. We only need the final product

$$\left(g, h, \prod_{i=1}^d H(q_i), \prod_{i=1}^d h_i\right)$$

to be a valid DDH tuple for correctness.

Suppose there exists *at least one* dimension on which the projections of *all* balls are disjoint. This gives each ball a unique way to identify it from others. Our idea is to leverage OKVS instances recursively: For each ball  $\mathbf{w}_k$ , the receiver encodes an *outer* OKVS for dimension  $i$  by

$$E_i \leftarrow \text{Encode}\left(\left\{\left(w_{k,i} + j, \quad \text{val}_{k,i,j}\right)\right\}_{j \in [-\delta, +\delta]}\right),$$

where  $\text{val}_{k,i,j}$  differs in two cases:

- If the current dimension  $i$  is the globally separated dimension for all balls, then  $\text{val}_{k,i,j}$  is an *inner* OKVS instance for fuzzy matching with  $\mathbf{w}_k$ , namely,

$$\text{val}_{k,i,j} \leftarrow \text{Encode}\left(\left\{\left(i' \parallel w_{k,i'} + j', \quad h_{i',j'} \parallel h_{i',j'}^s\right)\right\}_{i' \in [d], j' \in [-\delta, +\delta]}\right),$$

where  $h_{i',j'} \leftarrow_{\mathbb{S}} \mathbb{G}$ ;

- Otherwise, the  $\text{val}_{k,i,j} := (\mathbf{r} \parallel \mathbf{r}^s)$  is a *dummy* instance where  $\mathbf{r} \leftarrow_{\mathbb{S}} \mathbb{G}^m$  and  $m$  is the size of the inner OKVS instance.

For each point  $\mathbf{q}$ , the sender first decodes the outer OKVS to obtain a list  $\{\text{val}_1, \dots, \text{val}_d\}$ , then runs the decoding function on each  $\text{val}_{j \in [d]}$  to get

$$(u_j \parallel v_j) := \prod_{i=1}^d \text{Decode}(\text{val}_j, q_i).$$

In the end, the sender re-randomizes the result from the tuple  $(g, h, \prod_{j=1}^d u_j, \prod_{j=1}^d v_j)$ .

For correctness, we expect that decoding a dummy instance on any key would output a valid DDH pair all the time. This can be guaranteed if the inner OKVS has a linear decoding function. Clearly, in this way, each  $\mathbf{q}$  would get either an inner OKVS instance or random garbage from *the globally separated* dimension. The latter results in valid DDH tuples with negligible probability, so we focus on the case that the sender gets an inner OKVS instance in the end. This reduces the fuzzy PSI problem to the fuzzy matching problem as other dummy instances won't affect the correctness. For security, the inner OKVS has to be doubly oblivious, namely, the encoded structure itself is uniformly random. Regarding the complexity, the receiver's communication and computation costs would be  $O(d\delta)$  times larger.

**Further Weaken the Assumption.** The above assumption is weaker and milder than what was used in prior works, but it is still somewhat artificial. Here we show that we can even weaken this assumption to the following: For each ball, there *exists* at least one dimension on which *its* projection is separated from others. Note that the above approach doesn't work yet in this setting: There might exist a point whose projection on *each* dimension is inside the projection of a *non-separated* interval from some ball. In other words, the sender would get a list of dummy instances after decoding the outer OKVS. This results in a false positive since dummy instances always output a match. To rule out these false positives, we realize that we could encode additional information into each  $\text{val}_{k,i,j}$ .

For simplicity, let's assume the decoding function of the OKVS is determined by a binary vector with some *fixed hamming weight*, that is, given an instance  $\mathbf{r} \in \mathbb{G}^m$  and some *key*, the decoding function outputs

$$\text{Decode}(\mathbf{r}, \text{key}) = \langle \mathbf{d}, \mathbf{r} \rangle = \prod_{i=1}^m r_i^{d_i},$$

where  $\mathbf{d} \in \{0,1\}^m$  is deterministically sampled by the *key*, and  $\text{HammingWeight}(\mathbf{d}) = t$ . The receiver samples two random shares  $\zeta_{\perp}, \zeta_{\top}$  such that  $\zeta_{\perp} \cdot \zeta_{\top} = 1$ . We denote as  $I_k$  the *first* dimension on which  $\mathbf{w}_k$  projects a separated interval. Then the receiver could set  $\text{val}_{k,i,j}$  for each  $\mathbf{w}_k$  in this way:

- If the current dimension  $i = I_k$ , then  $\text{val}_{k,i,j}$  is an inner OKVS instance defined by

$$\text{val}_{k,i,j} \leftarrow \text{Encode} \left( \left\{ (i' \parallel w_{k,i'} + j', \quad h_{i',j'} \parallel h_{i',j'}^s \cdot \zeta_{\perp}^{t \cdot (d-1)}) \right\}_{i' \in [d], j' \in [-\delta, +\delta]} \right),$$

- where  $h_{i',j'} \leftarrow_s \mathbb{G}$  and  $t$  is the hamming weight of  $\mathbf{d}$ ;
- Otherwise, the  $\text{val}_{k,i,j} := (\mathbf{r} \parallel \mathbf{r}^s \cdot \zeta_{\top})$  for  $\mathbf{r} \leftarrow_s \mathbb{G}^m$ .

The security follows as before, whereas the correctness is non-trivial. First, consider the sender’s point  $\mathbf{q}$  intersecting some receiver’s ball. After decoding the inner OKVS instance, the sender gets a pair  $(u_* \parallel u_*^s \cdot \zeta_{\perp}^{td \cdot (d-1)})$  for some  $u_*$ ; After decoding a dummy instance, the sender gets  $(r_* \parallel r_*^s \cdot \zeta_{\top}^{td})$  for some  $r_*$  instead. Now, by multiplying them together, the final tuple

$$(g, h, v \parallel v^s \cdot \zeta_{\perp}^{td \cdot (d-1)} \cdot \zeta_{\top}^{td \cdot (d-1)}) = (g, h, v \parallel v^s)$$

is a valid DDH tuple for some  $v \in \mathbb{G}$ .

Then consider the case that the sender’s point  $\mathbf{q}$  is outside of all balls. The only way to report a match is to get a list of all dummy instances after decoding the outer OKVS instance, otherwise the inner OKVS instance will output a random garbage result. However, since dummy instances only encode  $\zeta_{\top}$ , the product of them equals 1 with negligible probability due to  $\zeta_{\top}$  being randomly sampled and  $td^2 \ll p$  if  $t = O(\kappa)$ .

Recall that we assume the decoding vector  $\mathbf{d}$  to have fixed hamming weight. This is not ideal since most modern OKVS instantiations (e.g., [6, 19, 35]) don’t satisfy this requirement, whereas the only exception is the garbled bloom filters [15] whose efficiency is not satisfactory. We managed to get rid of this assumption in our real protocol in the end, please refer to Sect. 7.1 for details.

**Locality-Sensitive Hashing.** The above approaches are heavily tailored to the  $L_{\infty}$  distance. To support  $L_p$  distance in high dimensions, we utilize locality-sensitive hashing (LSH) to identify matching balls. An LSH family with parameters  $(\delta, c\delta, p_1, p_2)$  guarantees the following:

- If two points  $\mathbf{w}$  and  $\mathbf{q}$  are close enough, i.e.,  $\text{dist}_p(\mathbf{w}, \mathbf{q}) \leq \delta$ , they would be hashed into the same bucket with at least  $p_1$  probability;
- If they are far apart, i.e.,  $\text{dist}_p(\mathbf{w}, \mathbf{q}) > c\delta$ , then the probability of hashing them into the same bucket is at most  $p_2$ .

In other words, an LSH family bounds the false-positive and false-negative probability to  $p_2$  and  $1 - p_1$ , respectively. Usually, false-positive and false-negative cannot be reduced to negligible simultaneously. However, given the existence of our fuzzy matching protocols, we can tolerate false positives by running fuzzy matching on each positive match. Therefore, the high-level strategy is that the receiver hashes each ball center via LSH to some LSH entry, and the sender would identify multiple positive LSH entries for each of its points. If we set the parameters properly, the total number of false positives for each sender’s point can be upper-bounded by  $O(N^{\rho})$  for some  $\rho < 1$  which gives us just a *sub-quadratic* blowup in total communication and computation complexities.

One caveat is that there is a constant gap between the calculation of false positives and false negatives mentioned above, namely, false positives are calculated when points are  $c\delta$ -apart, whereas false negatives are calculated when

points are  $\delta$ -close. Fortunately, when the receiver’s balls are disjoint (i.e., centers are  $2\delta$ -part), this gap can be filled by setting  $c = 2$ . Another caveat is that this approach does not support fuzzy PSI cardinality anymore due to the rationale behind the LSH: To guarantee a negligible false-negative rate, we typically have to prepare multiple LSH tables where a true positive might appear more than once.

For the formal details of this construction, we refer to the full version of the paper [2].

### 3 Preliminaries

We represent the computational security parameter as  $\lambda \in \mathbb{N}$ , the statistical security parameter as  $\kappa \in \mathbb{N}$  and the output of the algorithm  $\mathcal{A}$  on input  $\text{in}$  using  $r \leftarrow \{0, 1\}$  as its randomness by  $x \leftarrow \mathcal{A}(\text{in}; r)$ . The randomness is often omitted and only explicitly mentioned when necessary. Efficient algorithms are considered to be *probabilistic polynomial time* (PPT) machines. We use  $\approx_c$  to denote computational indistinguishability and  $\approx_s$  to denote statistical indistinguishability of probability distributions. The notation  $[n]$  signifies a set  $\{1, \dots, n\}$  and  $[a : b]$  the set  $\{a, a + 1, \dots, b - 1, b\}$ . We use  $\mathbf{c}[i : j]$  to represent a vector with a defined length of  $[c_i, \dots, c_j]$  and  $\mathbf{c}$  to indicate a vector of  $c$ .

All the protocols presented in this work are two-party protocols. Security is proven against semi-honest adversaries via the standard simulation-based paradigm (see e.g., [29]).

#### 3.1 Oblivious Key-Value Store (OKVS)

The concept of an oblivious key-value store (OKVS) was introduced by Garimella et al. [19] to capture the properties of data structures commonly used in PSI protocols. Subsequent works proposed OKVS constructions offering favorable trade-offs between encoding/decoding time and encoding size [6, 35].

**Definition 1 (Oblivious Key-Value Store).** *An oblivious key-value store OKVS is parameterized by a key space  $\mathcal{K}$ , a value space  $\mathcal{V}$ , computational and statistical security parameters  $\lambda, \kappa$ , respectively, and consists of two algorithms:*

- **Encode** : *takes as input a set of key-value pairs  $L \in (\mathcal{K} \times \mathcal{V})^n$  and randomness  $\theta \in \{0, 1\}^\lambda$ , and outputs a vector  $P \in \mathcal{V}^m$  or a failure indicator  $\perp$ .*
- **Decode** : *takes as input a vector  $P \in \mathcal{V}^m$ , a key  $k \in \mathcal{K}$  and randomness  $\theta \in \{0, 1\}^\lambda$ , and outputs a value  $v \in \mathcal{V}$ .*

*That satisfies:*

- **Correctness**: *For all  $L \in (\mathcal{K} \times \mathcal{V})^n$  with distinct keys and  $\theta \in \{0, 1\}^\lambda$  for which  $\text{Encode}(L; \theta) = P \neq \perp$ , it holds that  $\forall (k, v) \in L: \text{Decode}(P, k; \theta) = v$ .*
- **Low failure probability**: *For all  $L \in (\mathcal{K} \times \mathcal{V})^n$  with distinct keys:*  
 $\Pr_{\theta \leftarrow \{0, 1\}^\lambda}[\text{Encode}(L; \theta) = \perp] \leq 2^{-\kappa}$ .

- **Obliviousness:** For any  $\{k_1, \dots, k_n\}, \{k'_1, \dots, k'_n\} \subseteq \mathcal{K}$  of  $n$  distinct keys and any  $\theta \in \{0, 1\}^\lambda$ , if  $\text{Encode}$  does not output  $\perp$ , then for  $v_1, \dots, v_n \leftarrow_s \mathcal{V}$ :  $\{P \leftarrow \text{Encode}(\{(k_i, v_i)_{i \in [n]}\}; \theta)\} \approx_c \{P' \leftarrow \text{Encode}(\{(k'_i, v_i)_{i \in [n]}\}; \theta)\}$ .
- **Double obliviousness:** For all sets of  $n$  distinct keys  $\{k_1, \dots, k_n\} \subseteq \mathcal{K}$  and  $n$  values  $\{v_1, \dots, v_n\} \leftarrow_s \mathcal{V}$ , there is  $\text{Encode}(\{(k_i, v_i)_{i \in [n]}\}; \theta)$  statistically indistinguishable from uniformly random element from  $\mathcal{V}^m$ .

The efficiency of OKVS is characterized by: (1) the time it takes to encode  $n$  key-value pairs; (2) the time it takes to decode a single key; (3) the ratio  $n/m$  between the number of key-value pairs  $n$  and the encoding size  $m$ , also called the *rate*. Recent OKVS constructions [6, 19, 35] achieve: (1) encoding time  $O(n\kappa)$ ; (2) decoding time  $O(\kappa)$ ; (3) constant rate.

For this work, we will need OKVS to support the value space  $\mathcal{V}$  being equal to a cyclic group  $\mathbb{G}$  of prime order  $p$ . A sufficient condition for this, which is satisfied by the efficient constructions of [6, 19, 35] is:

- **$\mathbb{F}_p$ -Linear:** There exists a function  $\text{dec} : \mathcal{K} \times \{0, 1\}^\lambda \rightarrow \mathbb{F}_p^m$  such that for all  $P \in \mathbb{G}^m, k \in \mathcal{K}$  and  $\theta \in \{0, 1\}^\lambda$  it holds that  $\text{Decode}(P, k; \theta) := \langle \text{dec}(k; \theta), P \rangle$ , where for  $\mathbf{d} \in \mathbb{F}_p^m$  and  $\mathbf{g} \in \mathbb{G}^m$  we define  $\langle \mathbf{d}, \mathbf{g} \rangle := g_1^{d_1} \cdots g_m^{d_m}$ .

**Lemma 1 (Independence).** *If OKVS satisfies  $\mathbb{F}_p$ -linearity and  $\text{negl}(\kappa)$  failure probability, and  $\theta$  is uniformly randomly chosen, then for any  $L := \{(k_i, v_i)_{i \in [n]}\}$  with distinct keys, and any key  $k \notin \{k_i\}_{i \in [n]}$ , it holds that  $\text{Decode}(\text{Encode}(L; \theta), k)$  is indistinguishable from random.*

### 3.2 Random Self-reductions of DDH Tuples

The well-known decisional Diffie-Hellman (DDH) assumption for a cyclic group  $\mathbb{G} = \langle g \rangle$  of prime order  $p$  states that the distribution of Diffie-Hellman (DH) tuples  $(g, h := g^s, h_1, h_2 := h_1^s)$ , where  $s \leftarrow_s \mathbb{Z}_p, h_1 \leftarrow_s \mathbb{G}$ , is computationally indistinguishable from the distribution of random tuples  $(g, h := g^s, h_1, h_2)$ , where  $s \leftarrow_s \mathbb{Z}_p, h_1, h_2 \leftarrow_s \mathbb{G}$ . Naor and Reingold [32] show that deciding whether an arbitrary tuple  $(g, h, h_1, h_2)$  with  $h, h_1, h_2 \in \mathbb{G}$  is a DH tuple can be reduced to breaking the DDH assumption. For this work, we consider a special case of this reduction where  $h := g^s$  is fixed.

**Lemma 2 (Random Self-Reduction [32]).** *Let  $\mathbb{G} := \langle g \rangle$  be a cyclic group of order  $p$ , let  $h := g^s$  for  $s \in \mathbb{Z}_p$  and  $h_1, h_2 \in \mathbb{G}$ . If  $h'_1 := g^a \cdot h_1^b$  and  $h'_2 := h^a \cdot h_2^b$ , where  $a, b \leftarrow_s \mathbb{Z}_p$ , then:*

- $h'_1$  is uniformly random in  $\mathbb{G}$  and  $h'_2 = (h'_1)^s$  if  $h_2 = h_1^s$ .
- $(h'_1, h'_2)$  is a uniformly random pair of group elements otherwise.

$\mathcal{F}_{\text{FUZZYMATCH}}$
<hr/> <b>Parameters :</b> dimension $d$ , radius $\delta$ , and a distance function $\text{dist}(\cdot, \cdot)$ .
<b>Functionality :</b>
<ul style="list-style-type: none"> <li>- RECEIVER inputs <math>\mathbf{w} \in \mathbb{Z}^d</math>.</li> <li>- SENDER inputs <math>\mathbf{q} \in \mathbb{Z}^d</math>.</li> <li>- Output 1 to RECEIVER if <math>\text{dist}(\mathbf{w}, \mathbf{q}) \leq \delta</math>, and 0 otherwise.</li> </ul>

Possible Distance Functions
<hr/> $\text{dist}(\mathbf{w}, \mathbf{q})$ is defined as:
<ul style="list-style-type: none"> <li>- <math>L_\infty</math> Distance: <math>\text{dist}(\mathbf{w}, \mathbf{q}) = \max_{i \in [d]}  w_i - q_i </math></li> <li>- Hamming Distance: <math>\text{dist}(\mathbf{w}, \mathbf{q}) = \sum_{i=1}^d (w_i \neq q_i)</math></li> <li>- Conjunction of Hamming Distance and <math>L_\infty</math> on <math>\delta_\infty</math>:  <math>\text{dist}(\mathbf{w}, \mathbf{q}) = \sum_{i=1}^d ( w_i - q_i  &gt; \delta_\infty)</math></li> <li>- <math>L_p</math> Distance: <math>\text{dist}(\mathbf{w}, \mathbf{q}) = \left( \sum_{i=1}^d  w_i - q_i ^p \right)^{1/p}</math></li> </ul>

Fig. 1. Ideal Functionality of Fuzzy Matching

## 4 Definitions and Functionalities

We define the two-message protocol as below, consisting of three algorithms:

- $\text{Receiver}_1(\text{INPUT}_R)$ : The algorithm takes the RECEIVER’s  $\text{INPUT}_R$ , outputs the first message  $\text{msg}_1$  and its secret state  $\text{st}$ ;
- $\text{Sender}_1(\text{INPUT}_S, \text{msg}_1)$ : The algorithm takes the SENDER’s  $\text{INPUT}_S$  and  $\text{msg}_1$ , outputs the second message  $\text{msg}_2$ ;
- $\text{Receiver}_2(\text{st}, \text{msg}_2)$ : The algorithm takes the state  $\text{st}$  and the second message  $\text{msg}_2$ , outputs the final OUTPUT.

### 4.1 Definition of Fuzzy Matching

We define the functionality of fuzzy matching between two points in Fig. 1, with different distance functions including both infinity ( $L_\infty$ ) and Minkowski ( $L_p$ ) distance where  $p \in [1, \infty)$ .

### 4.2 Definition of Fuzzy (Circuit) Private Set Intersection

We define the functionality of fuzzy PSI and fuzzy circuit PSI in Figs. 2 and 3, respectively. Note that for standard fuzzy PSI, we also consider a slightly stronger functionality (compared to prior works) where the receiver only learns whether their points are in the intersection, but not the sender’s exact points, which we call PSI with sender privacy (PSI-SP). We extend the functionality of fuzzy PSI to many closely related variants including PSI cardinality (PSI-CA), labeled PSI, and circuit PSI.



$\mathcal{F}_{\text{FUZZYPSI}}$
<hr/> <b>Parameters</b> : dimension $d$ , radius $\delta$ , cardinality of sets $N, M$ , a distance function $\text{dist}(\cdot, \cdot)$ , a leakage function $\text{leakage}(\cdot, \cdot)$ , label length $\sigma$ , and a concise description for receiver's and sender's points $\mathcal{D}_R, \mathcal{D}_S$ , respectively.
<b>Functionality</b> :
<ul style="list-style-type: none"> <li>- RECEIVER inputs <math>\mathbf{W} \in \mathbb{Z}^{d \times N}</math> according to <math>\mathcal{D}_R</math>.</li> <li>- SENDER inputs <math>\mathbf{Q} \in \mathbb{Z}^{d \times M}</math> according to <math>\mathcal{D}_S</math>.</li> <li>  For Labeled PSI, SENDER inputs <math>\text{Label}_{\mathbf{Q}} \in \{0, 1\}^{\sigma \times M}</math>.</li> <li>- Return <math>\text{leakage}(\mathbf{W}, \mathbf{Q})</math> to RECEIVER.</li> </ul>

Possible Leakage Functions
<hr/> $\text{leakage}(\mathbf{W}, \mathbf{Q})$ is defined as:
<ul style="list-style-type: none"> <li>- PSI-CA: <math>\text{leakage}(\mathbf{W}, \mathbf{Q}) = \sum_{i \in [N], j \in [M]} (\text{dist}(\mathbf{w}_i, \mathbf{q}_j) \leq \delta)</math>.</li> <li>- PSI: <math>\text{leakage}(\mathbf{W}, \mathbf{Q}) = \{\mathbf{q}_j \mid \exists i \in [N], \text{dist}(\mathbf{w}_i, \mathbf{q}_j) \leq \delta\}</math>.</li> <li>- PSI-SP: <math>\text{leakage}(\mathbf{W}, \mathbf{Q}) = \{\mathbf{w}_i \mid \exists j \in [M], \text{dist}(\mathbf{w}_i, \mathbf{q}_j) \leq \delta\}</math>.</li> <li>- Labeled PSI: <math>\text{leakage}(\mathbf{W}, \mathbf{Q}) = \{\text{label}_j \mid \exists i \in [N], \text{dist}(\mathbf{w}_i, \mathbf{q}_j) \leq \delta\}</math>, where <math>\text{label}_j</math> is the label associated with <math>\mathbf{q}_j</math>.</li> </ul>

Fig. 2. Ideal Functionality of Fuzzy PSI

$\mathcal{F}_{\text{FUZZYCPSI}}$
<hr/> <b>Parameters</b> : dimension $d$ , radius $\delta$ , cardinality of sets $N, M$ , a distance function $\text{dist}(\cdot, \cdot)$ , a leakage function $\text{leakage}(\cdot, \cdot)$ , associated data length $\sigma$ , and a concise description for receiver's and sender's points $\mathcal{D}_R, \mathcal{D}_S$ , respectively.
<b>Functionality</b> :
<ul style="list-style-type: none"> <li>- RECEIVER inputs <math>\mathbf{W} \in \mathbb{Z}^{d \times N}</math> according to <math>\mathcal{D}_R</math> and associated data <math>\tilde{\mathbf{W}} \in \{0, 1\}^{\sigma \times N}</math>.</li> <li>- SENDER inputs <math>\mathbf{Q} \in \mathbb{Z}^{d \times M}</math> according to <math>\mathcal{D}_S</math> and associated data <math>\tilde{\mathbf{Q}} \in \{0, 1\}^{\sigma \times M}</math>.</li> <li>- For each <math>j \in [M]</math>, sample <math>r_j, s_j \leftarrow_{\\$} \{0, 1\}^{1+2\sigma}</math> such that:  <math>r_j \oplus s_j = 1 \parallel \tilde{\mathbf{w}}_i \parallel \tilde{\mathbf{q}}_j</math> if <math>\exists i \in [N]</math> s.t. <math>\text{dist}(\mathbf{w}_i, \mathbf{q}_j) \leq \delta</math>, and <math>r_j \oplus s_j = 0^{1+2\sigma}</math> otherwise.</li> <li>- Return <math>(r_j)_{j \in [M]}</math> to RECEIVER and <math>(s_j)_{j \in [M]}</math> to SENDER.</li> </ul>

Fig. 3. Ideal Functionality of Fuzzy Circuit PSI

## 5 Fuzzy Matching

We start by presenting a fuzzy matching protocol for two points in hyperspace with infinity distance ( $L_\infty$ ) and hamming distance, then we extend it into a more general setting with Minkowski distance ( $L_{p \in [1, \infty)}$ ).

### 5.1 Fuzzy Matching for Infinity Distance

We provide the protocol for infinity distance in Fig. 4. We also show how to generalize the above approach to support the conjunction of infinity and hamming distance in the full version of the paper [2]. The proofs of the following theorems can be found in the full version of the paper [2].

**Theorem 1 (Correctness).** *The protocol provided in Fig. 4 is correct with  $1 - \text{negl}(\kappa)$  probability if OKVS satisfies perfect correctness defined in Sect. 3.1*

<u>GetList<sub>∞</sub>(<i>h, s, w, Δ<sub>w</sub></i>)</u>	<u>GetTuple<sub>∞</sub>(<i>g, h, q, Δ<sub>q</sub>, E</i>)</u>
For each $i = 1 \dots d$ : For each $j = -\delta \dots \delta$ : Set $\text{key}_j \leftarrow H_\gamma(\Delta_w \  w_i + j)$ Set $h_j \leftarrow \mathbb{G}$ Set $\text{val}_j = (h_j \  h_j^s)$ Set $\text{list}_i = \{(\text{key}_j, \text{val}_j)_{j \in [-\delta:\delta]}\}$ Return $\text{list}_1, \dots, \text{list}_d$	For each $i = 1 \dots d$ : Set $\text{key}_i \leftarrow H_\gamma(\Delta_q \  q_i)$ $(u_i \  v_i) \leftarrow \text{Decode}(E_i, \text{key}_i)$ Sample $a, b \leftarrow \mathbb{Z}_p$ Set $u_* = g^a \cdot \left(\prod_{i=1}^d u_i\right)^b$ Set $v_* = h^a \cdot \left(\prod_{i=1}^d v_i\right)^b$ Set $v_* \leftarrow H_{\kappa'}(v_*)$ Return $(u_*, v_*)$

<u>Receiver<sub>1</sub>(<i>w</i> ∈ ℤ<sup><i>d</i></sup>)</u>	<u>Sender<sub>1</sub>(<i>q</i> ∈ ℤ<sup><i>d</i></sup>, <i>msg</i><sub>1</sub>)</u>
Sample $g \leftarrow \mathbb{G}, s \leftarrow \mathbb{Z}_p$ Compute $h = g^s$ Get $\{\text{list}_i\}_{i \in [d]} \leftarrow \text{GetList}_\infty(h, s, \mathbf{w}, 0^\kappa)$ Set $E_i \leftarrow \text{Encode}(\text{list}_i)$ for each $i \in [d]$ Set $\mathbf{E} = \{E_1, \dots, E_d\}$ Output $\text{msg}_1 := (g, h, \mathbf{E}), \text{st} := s$	Parse $\text{msg}_1 := (g, h, \mathbf{E})$ $(u_*, v_*) \leftarrow \text{GetTuple}_\infty(g, h, \mathbf{q}, 0^\kappa, \mathbf{E})$ Output $\text{msg}_2 := (u_*, v_*)$
	<u>Receiver<sub>2</sub>(<i>st, msg</i><sub>2</sub>)</u>
	Parse $\text{msg}_2 := (u_*, v_*)$ and $\text{st} := s$ Output 1 if $H_{\kappa'}(u_*^s) = v_*$ and 0 otherwise

**Fig. 4.** Fuzzy Matching for  $L_\infty$  Distance

and independence property from Lemma 1, and  $H_\gamma : \{0, 1\}^* \mapsto \{0, 1\}^\gamma, H_{\kappa'} : \mathbb{G} \mapsto \{0, 1\}^{\kappa'}$  are universal hash functions where  $\gamma = \kappa + \log \delta$  and  $\kappa' = \kappa$ .

**Theorem 2 (Security).** *The protocol provided in Fig. 4 realizes the functionality defined in Fig. 1 for  $L_\infty$  distance function against semi-honest adversaries if OKVS is oblivious and the DDH assumption holds.*

**Theorem 3 (Complexity).** *The communication complexity is  $O(2\delta d\lambda + \lambda + \kappa)$  where  $\lambda, \kappa$  are the security and statistical parameters; The computational complexity is  $O(2\delta d)$  for the receiver and  $O(d)$  for the sender.*

### 5.2 Fuzzy Matching for Minkowski Distance

We provide the protocol for  $L_p$  distance where  $1 \leq p < \infty$  in Fig. 5. For simplicity, we assume  $p$  is an integer for the moment. The proofs of the following theorems can be found in the full version of the paper [2].

**Theorem 4 (Correctness).** *The protocol provided in Fig. 5 is correct with  $1 - \text{negl}(\kappa)$  probability if OKVS satisfies perfect correctness defined in Sect. 3.1 and independence property from Lemma 1, and  $H_\gamma : \{0, 1\}^* \mapsto \{0, 1\}^\gamma, H_\kappa : \mathbb{G} \mapsto \{0, 1\}^\kappa$  are universal hash functions where  $\gamma = \kappa + \log \delta$  and  $\kappa' = \kappa + p \log \delta$ .*

<u>GetList<sub>p</sub>(<math>h, s, \mathbf{w}, \Delta_w</math>)</u>	<u>GetTuple<sub>p</sub>(<math>g, h, \mathbf{q}, \Delta_q, \mathbf{E}</math>)</u>
For each $i = 1 \dots d$ : For each $j = -\delta \dots \delta$ : Set $\text{key}_j \leftarrow H_\gamma(\Delta_w \  w_i + j)$ Set $h_j \leftarrow \mathbb{G}$ Set $\text{val}_j = (h_j \  h_j^s \cdot g^{ j p})$ Set $\text{list}_i = \{(\text{key}_j, \text{val}_j)_{j \in [-\delta:\delta]}\}$ Return $\text{list}_1, \dots, \text{list}_d$	For each $i = 1 \dots d$ : Set $\text{key}_i \leftarrow H_\gamma(\Delta_q \  q_i)$ $(u_i \  v_i) \leftarrow \text{Decode}(E_i, \text{key}_i)$ Sample $a, b, c \leftarrow \mathbb{Z}_p$ Set $f_* = g^c \cdot \left(\prod_{i=1}^d u_i\right)^b$ Set $h_* = h^c \cdot \left(\prod_{i=1}^d v_i\right)^b \cdot g^a$ Set $\text{list}_* = \emptyset$ For each $j = 0 \dots \delta^p$ : Set $x_j = H_{\kappa'}(g^{a+b \cdot j})$ Set $\text{list}_* = \text{list}_* \cup x_j$ Shuffle $\text{list}_*$ Return $(f_*, h_*, \text{list}_*)$

<u>Receiver<sub>1</sub>(<math>\mathbf{w} \in \mathbb{Z}^d</math>)</u>	<u>Sender<sub>1</sub>(<math>\mathbf{q} \in \mathbb{Z}^d, \text{msg}_1</math>)</u>
Sample $g \leftarrow \mathbb{G}, s \leftarrow \mathbb{Z}_p$ Compute $h = g^s$ Get $\{\text{list}_{i \in [d]}\} \leftarrow \text{GetList}_p(h, s, \mathbf{w}, 0^\kappa)$ Get $E_i \leftarrow \text{Encode}(\text{list}_i)$ for each $i \in [d]$ Set $\mathbf{E} = \{E_1, \dots, E_d\}$ Output $\text{msg}_1 := (g, h, \mathbf{E}), \text{st} := s$	Parse $\text{msg}_1 := (g, h, \mathbf{E})$ $(f_*, h_*, \text{list}_*) \leftarrow \text{GetTuple}_p(g, h, \mathbf{q}, 0^\kappa, \mathbf{E})$ Output $\text{msg}_2 := (f_*, h_*, \text{list}_*)$
	<u>Receiver<sub>2</sub>(<math>\text{st}, \text{msg}_2</math>)</u> Parse $\text{msg}_2 := (f_*, h_*, \text{list}_*)$ and $\text{st} := s$ Set $x = H_{\kappa'}(f_*^{-s} \cdot h_*)$ Output 1 if $x \in \text{list}_*$ and 0 otherwise

**Fig. 5.** Fuzzy Matching for  $L_p$  Distance

**Theorem 5 (Security).** *The protocol provided in Fig. 5 realizes the functionality defined in Fig. 1 for  $L_p$  distance function, against semi-honest adversaries if OKVS is oblivious, the hash function  $H_{\kappa'} : \mathbb{G} \mapsto \{0, 1\}^{\kappa'}$  is modeled as a random oracle, and the DDH assumption holds.*

**Theorem 6 (Complexity).** *The communication complexity is  $O(2\delta d\lambda + 2\lambda + \delta^p \kappa)$  where  $\lambda, \kappa$  are the security and statistical parameters; The computational complexity is  $O(2\delta d)$  for the receiver and  $O(d + \delta^p)$  for the sender.*

## 6 Fuzzy PSI in Low-Dimension Space

Clearly, with a fuzzy matching protocol in hand, we could straightforwardly execute a protocol instance for every pair of points from both the sender and receiver. Yet, this approach would lead to a quadratic increase in computational and communicative overheads. In the following sections, we depict some

methods to circumvent this quadratic overhead, addressing both low-dimensional (in Sect. 6) and high-dimensional (in Sect. 7) spaces separately. We will deal with PSI-CA first (i.e., only let the receiver learn the cardinality of the intersection), then show how to extend PSI-CA to broader functionalities in Sect. 8, including standard PSI, labeled PSI, and circuit PSI.

### 6.1 Spatial Hashing Techniques

Consider the case that points are located in a low-dimension space  $\mathcal{U}^d$  (e.g.,  $d = o(\log(\lambda))$ ) where  $\mathcal{U}$  is the universe for each dimension. We use a similar idea from [20] to tile the entire space into hypercubes with side length  $2\delta$ , but we consider a more general  $L_p$  distance setting. That is, we consider  $L_p$  distance over a space tiled by  $L_\infty$  hypercubes. We denote each hypercube as a *cell*. Specifically, given a point  $\mathbf{w} \in \mathcal{U}^d$ , the index  $\text{id}_i$  of each cell  $\mathcal{C}$  on each dimension  $i \in [d]$  is determined by  $\text{id}_i = \lfloor \frac{w_i}{2\delta} \rfloor$  and each cell is labeled by  $\text{id}_0 \parallel \dots \parallel \text{id}_d$ . The proofs of the following results are given in the full version of the paper [2].

**Lemma 3 (Maximal Distance in a Cell).** *Given two points  $\mathbf{w}, \mathbf{q} \in \mathcal{U}^d$  located in the same cell with side length  $2\delta$ , then the distance between them is  $\text{dist}_p(\mathbf{w}, \mathbf{q}) < 2\delta d^{\frac{1}{p}}$  where  $p \in [1, \infty]$ . Specifically, if  $p = \infty$ ,  $\text{dist}_\infty(\mathbf{w}, \mathbf{q}) < 2\delta$ .*

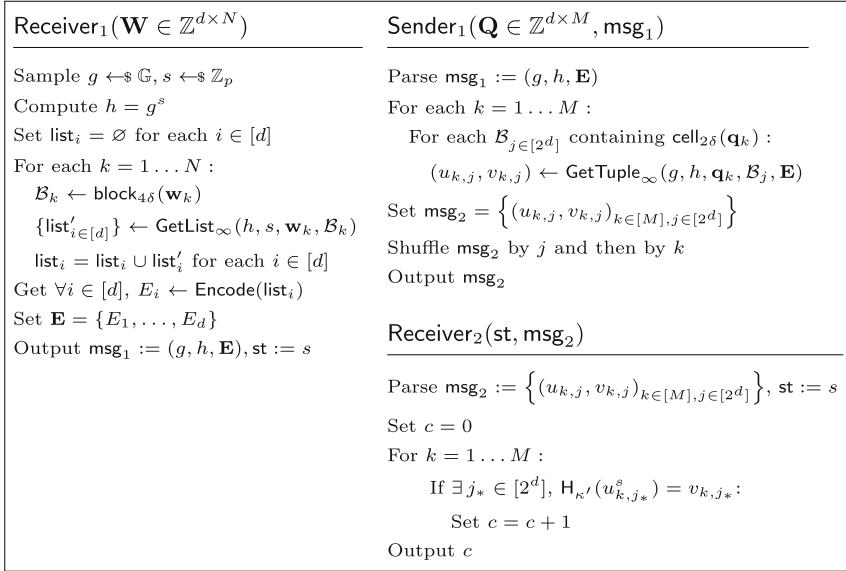
**Lemma 4 (Unique Center).** *Suppose there are multiple  $L_p$  balls ( $p \in [1, \infty]$ ) with radius  $\delta$  lying in a  $d$ -dimension space which is tiled by hypercubes (i.e., cells) with side length  $2\delta$ . If these balls' centers are at least  $2\delta d^{\frac{1}{p}}$  apart, then for each cell, there is at most one center of the balls lying in this cell. Specifically, if  $p = \infty$ , then the unique center holds for disjoint balls since  $2\delta d^{\frac{1}{p}}$  degrades to  $2\delta$  in this case.*

**Lemma 5 (Unique Ball).** *Suppose there are multiple  $\delta$ -radius  $L_p$  balls ( $p \in [1, \infty]$ ) distributed in a  $d$ -dimension space which is tiled by hypercubes (cells) of side length  $2\delta$ . If these balls' centers are at least  $2\delta(d^{\frac{1}{p}} + 1)$  apart from each other, then there exists at most one ball intersecting with the same cell. Specifically, if  $p = \infty$ , this holds for  $L_\infty$  balls with  $4\delta$ -apart centers.*

**Lemma 6 (Unique Block).** *Any  $L_\infty$  ball with radius  $\delta$  will intersect with **exactly**  $2^d$  cells with side length  $2\delta$  in a  $d$ -dimension space. Moreover, if we denote such  $2^d$  cells together as a block (which is a hypercube with side length  $4\delta$ ), then each block is unique for each disjoint ball. In other words, any two disjoint balls must be associated with different blocks.*

### 6.2 Fuzzy PSI-CA for Infinity Distance

We provide the detailed protocol in Fig. 6 realizing fuzzy PSI-CA for infinity distance where the receiver's points are  $2\delta$  apart from each other (i.e., the receiver's  $\delta$ -radius balls are disjoint). In the figure,  $\text{block}_{4\delta}$  returns the label of the block of side-length  $4\delta$ ,  $\text{cell}_{2\delta}$  returns the label of the cell of side-length



**Fig. 6.** Fuzzy PSI-CA, infinity distance, receiver’s points are  $2\delta$  apart (i.e., disjoint balls)

$2\delta$ , and `GetList`, `GetTuple` are provided in Fig. 4. The proofs of the following theorems can be found in the full version of the paper [2], and we also generalize this approach to the setting where both parties hold a structured set of hyperballs.

**Theorem 7 (Correctness).** *The protocol presented in Fig. 6 is correct with probability  $1 - \text{negl}(\kappa)$  if OKVS satisfies perfect correctness defined in Sect. 3.1 and the independence property from Lemma 1,  $\mathbf{H}_\gamma : \{0, 1\}^* \mapsto \{0, 1\}^\gamma, \mathbf{H}_{\kappa'} : \mathbb{G} \mapsto \{0, 1\}^{\kappa'}$  used in `GetList`, `GetTuple` are universal hash functions where  $\gamma = \kappa + \log(MN\delta)$ ,  $\kappa' = \kappa + d \log M$ , and the receiver’s points are  $2\delta$  apart.*

**Theorem 8 (Security).** *The protocol presented in Fig. 6 realizes the fuzzy PSI-CA functionality defined in Fig. 2 for infinity distance against semi-honest adversaries if OKVS is oblivious, and the DDH assumption holds.*

**Theorem 9 (Complexity).** *The protocol provided in Fig. 6 has communication complexity  $O(2\delta dN\lambda + 2^d M(\lambda + \kappa'))$  where  $\lambda, \kappa = \kappa' - d \log M$  are the security and statistical parameters; The computational complexity is  $O(2\delta dN + 2^d M)$  for the receiver and  $O(2^d dM)$  for the sender.*

### 6.3 Fuzzy PSI-CA for Minkowski Distance

Assuming that the receiver’s points are spaced  $2\delta(d^{\frac{1}{p}} + 1)$  apart, we can allow the receiver to iterate through each possible location, as depicted in Fig. 7. The proofs of the following theorems can be found in the full version of the paper [2]

Receiver <sub>1</sub> ( $\mathbf{W} \in \mathbb{Z}^{d \times N}$ )	Sender <sub>1</sub> ( $\mathbf{Q} \in \mathbb{Z}^{d \times M}$ , $\text{msg}_1$ )
Sample $g \leftarrow \mathbb{G}$ , $s \leftarrow \mathbb{Z}_p$ Compute $h = g^s$ Set $\text{list}_i = \emptyset$ for each $i \in [d]$ For each $k = 1 \dots N$ : For each cell $C_{k'}$ intersecting $\text{ball}_\delta(\mathbf{w}_k)$ : $\{\text{list}'_{i \in [d]}\} \leftarrow \text{GetList}_p(h, s, \mathbf{w}_k, C_{k'})$ $\text{list}_i = \text{list}_i \cup \text{list}'_i$ for each $i \in [d]$ Pad $\text{list}_i$ to size $2^d N$ with random pairs Get $\forall i \in [d]$ , $E_i \leftarrow \text{Encode}(\text{list}_i)$ Set $\mathbf{E} = \{E_1, \dots, E_d\}$ Output $\text{msg}_1 := (g, h, \mathbf{E})$ , $\text{st} := s$	Parse $\text{msg}_1 := (g, h, \mathbf{E})$ For each $k = 1 \dots M$ : $C_k \leftarrow \text{cell}_{2\delta}(\mathbf{q}_k)$ $t_k \leftarrow \text{GetTuple}_p(g, h, \mathbf{q}_k, C_k, \mathbf{E})$ Set $\text{msg}_2 = \{t_{k \in [M]}\}$ and shuffle Output $\text{msg}_2$ <hr/> Receiver <sub>2</sub> ( $\text{st}$ , $\text{msg}_2$ ) <hr/> Parse $\text{msg}_2 := \{t_{k \in [M]}\}$ , $\text{st} := s$ Set $c = 0$ For $k = 1 \dots M$ : If $\mathbf{p} = \infty$ : Parse $t_k := (u_k, v_k)$ Set $c = c + 1$ if $H_{\kappa'}(u_k^s) = v_k$ Else: Parse $t_k := (f_k, h_k, \mathcal{X}_k)$ Set $c = c + 1$ if $H_{\kappa'}(f_k^{-s} h_k) \in \mathcal{X}_k$ Output $c$

**Fig. 7.** Fuzzy PSI-CA,  $L_p$  distance with  $\mathbf{p} \in [1, \infty]$ , receiver's points are  $2\delta(d^{\frac{1}{\mathbf{p}}} + 1)$  apart

**Theorem 10 (Correctness).** *The protocol presented in Fig. 7 is correct with probability  $1 - \text{negl}(\kappa)$  if OKVS satisfies the perfect correctness defined in Sect. 3.1 and the independence property from Lemma 1,  $H_\gamma : \{0, 1\}^* \mapsto \{0, 1\}^\gamma$ ,  $H_{\kappa'} : \mathbb{G} \mapsto \{0, 1\}^{\kappa'}$  used in GetList, GetTuple are universal hash functions where  $\gamma = \kappa + d \log(\delta N) + \log M$ ,  $\kappa' = \kappa + \mathbf{p} \log(M\delta)$  if  $\mathbf{p} < \infty$  and  $\kappa' = \kappa + \log M$  if  $\mathbf{p} = \infty$ , and the receiver's points are  $2\delta(d^{\frac{1}{\mathbf{p}}} + 1)$  apart for  $\mathbf{p} \in [1, \infty]$ .*

**Theorem 11 (Security).** *The protocol presented in Fig. 7 realizes the fuzzy PSI-CA functionality defined in Fig. 2 for  $L_{\mathbf{p} \in [1, \infty]}$  distance against semi-honest adversaries if OKVS is oblivious and the DDH assumption holds. Additionally, if  $\mathbf{p} < \infty$ , the hash function  $H_{\kappa'} : \mathbb{G} \mapsto \{0, 1\}^{\kappa'}$  is modeled as a random oracle.*

**Theorem 12 (Complexity).** *The protocol provided in Fig. 7 has communication complexity  $O(2\delta d 2^d N \lambda + M(\lambda + \kappa'))$  when  $L_{\mathbf{p}} = L_\infty$  and  $O(2\delta d 2^d N \lambda + M(2\lambda + \delta^{\mathbf{p}} \kappa'))$  when  $\mathbf{p} \in [1, \infty)$  where  $\lambda, \kappa$  are the security and statistical parameters. Specifically,  $\kappa = \kappa' - \log M$  if  $\mathbf{p} = \infty$  and  $\kappa = \kappa' - \mathbf{p} \log(M\delta)$  otherwise. The receiver's computational complexity is  $O(2\delta d 2^d N + M)$ ; The sender's computational complexity is  $O(dM)$  if  $\mathbf{p} = \infty$ , and  $O(dM + \delta^{\mathbf{p}})$  otherwise.*

## 7 Fuzzy PSI in High-Dimension Space

In this section, we construct an efficient fuzzy PSI protocol in a high-dimensional space, i.e., of a polynomially large dimension. For infinity distance, we provide a fuzzy PSI-CA protocol in Sect. 7.1 and extend it to richer functionalities

in Sect. 8; For Minkowski distance, please refer to the full version of the paper [2] for details.

### 7.1 Infinity Distance

Suppose we assume the receiver’s set has good distribution in a high-dimensional space, particularly if each ball has disjoint projections (i.e., separated) from others on *at least one* dimension. In this case, we can get communication and computation complexity both scaling *polynomially* in the dimension. For instance, if balls are uniformly distributed, then it satisfies this predicate with overwhelming probability. The proof of this can be found in the full version of the paper [2].

**Definition 2 (Separated Balls).** *The set of  $\delta$ -radius balls are separated in a  $d$ -dimension space if and only if the projections are separated on at least one dimension for each ball. Specifically, for the center  $\mathbf{w}_k$  of each ball in the set, there exists some dimension  $i_* \in [d]$  such that*

$$\forall j \in [-\delta : \delta], w_{k,i_*} + j \notin \{w_{k',i_*} + j'\}_{k' \neq k, j' \in [-\delta:\delta]},$$

where  $\{w_{k',i_*} + j'\}$  is the set of projections from other balls.

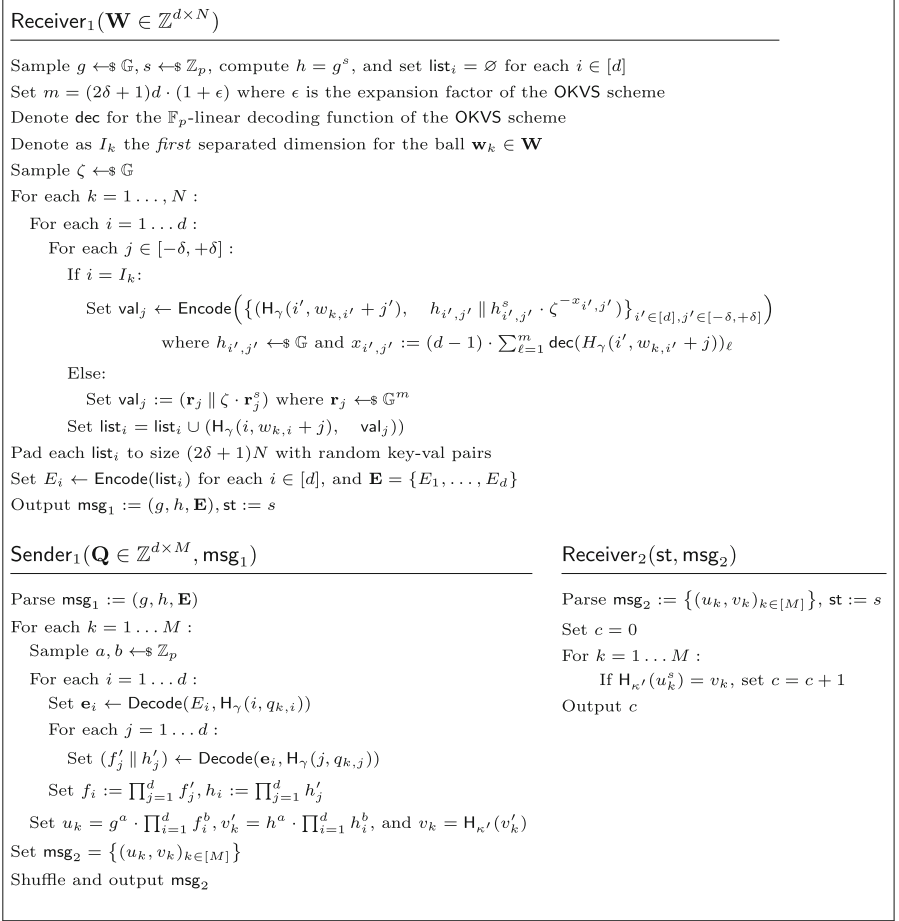
**Lemma 7 (Uniform Distribution).** *If centers of the balls are uniformly distributed ( $\mathbf{W} \leftarrow_{\mathcal{S}} \mathcal{U}^{d \times N}$ ) where  $\mathcal{U} := \mathbb{Z}_{2^u}$ , then it has the property defined in Definition 2 with probability  $1 - \text{negl}(d)$ .*

Given the receiver’s balls are separated as defined in Definition 2, we provide an efficient protocol in Fig. 8 which gets rid of the  $2^d$  term for both communication and computation. The proofs of the following theorems can be found in the full version of the paper [2].

**Theorem 13 (Correctness).** *The protocol presented in Fig. 8 is correct with probability  $1 - \text{negl}(\kappa)$  if OKVS satisfies the perfect correctness,  $\mathbb{F}_p$ -linearity defined in Sect. 3.1 and the independence property from Lemma 1,  $H_\gamma : \{0, 1\}^* \mapsto \{0, 1\}^\gamma$ ,  $H_{\kappa'} : \mathbb{G} \mapsto \{0, 1\}^{\kappa'}$  are universal hash functions where  $\gamma = \kappa + \log NM\delta$ ,  $\kappa' = \kappa + \log M$ , and the receiver’s set are separated as defined in Definition 2. Particularly, we require that the decoding vector satisfies  $\text{dec}(\cdot) \in \{0, 1\}^m$  and  $\text{HammingWeight}(\text{dec}(\cdot)) = O(\kappa)$  where  $m$  is the size of the OKVS.*

**Theorem 14 (Security).** *The protocol presented in Fig. 8 satisfies the fuzzy PSI-CA functionality defined in Fig. 2 for infinity distance against semi-honest adversaries if OKVS is doubly oblivious, and the DDH assumption holds.*

**Theorem 15 (Complexity).** *The protocol presented in Fig. 8 has communication complexity  $O((2\delta d)^2 N \lambda + M(\lambda + \kappa))$  where  $\lambda, \kappa = \kappa' - \log M$  are computational and statistical parameters; The computational complexity is  $O((2\delta d)^2 N + M)$  for the receiver and  $O(2d^2 M)$  for the sender.*



**Fig. 8.** Fuzzy PSI-CA, infinity distance, each ball is separated on at least one dimension

## 8 Extending to Broader Functionalities

We show above protocols can be extended to a broader class of functionalities, including standard PSI, PSI with sender privacy, labeled PSI, and circuit PSI, with small tweaks and therefore preserving the efficiency. We describe extensions for all protocols in this work except for the  $L_p$  distance protocol in high dimensional space since currently, the simulator for a corrupt receiver needs to know the points of the sender that lie in the intersection, i.e., only works for the standard PSI functionality. We describe the main idea behind the extensions and give the formal details for the different protocol settings (including PSI with sender privacy and circuit PSI) in the full version of the paper [2].



**Labeled PSI.** For labeled PSI, the sender has some labels  $\text{label}_k \in \{0, 1\}^\sigma$  attached to their input points  $\mathbf{q}_k$ ,  $k \in [M]$ , and the receiver wishes to learn the labels of the points for which there exists an  $i \in [N]$  such that  $\text{dist}(\mathbf{w}_i, \mathbf{q}_k) \leq \delta$  (see Fig. 2 for the ideal functionality). It can be realized for the protocol in Fig. 6 (and similar for the protocols in Figs. 7 and 8 by ignoring the index  $j$  in these cases) by letting the sender use  $v_{k,j}$  as a one-time pad to encrypt  $\text{label}_k$  together with a special prefix, e.g.,  $0^\kappa$ , indicating that the label belongs to a valid match. For the protocol in Fig. 7 with  $\mathbf{p} \neq \infty$ , the sender instead uses the  $x_{k,j} \in \mathcal{X}_k$  as a one-time pad to encrypt  $0^\kappa \parallel \text{label}_k$ .

**Standard PSI.** By letting the labels be a description of the sender’s points, we can realize standard PSI, where the receiver learns the sender’s points  $\mathbf{q}_k$  for which there exists an  $i \in [N]$  such that  $\text{dist}(\mathbf{w}_i, \mathbf{q}_k) \leq \delta$  (see Fig. 2 for the ideal functionality).

## 9 Performance Evaluation

In this section, we provide a micro-benchmark for our fuzzy PSI protocols for  $L_{\mathbf{p} \in \{1, 2, \infty\}}$  in low-dimension settings.

**Implementation.** We implement the standard fuzzy PSI variant (i.e., the receiver learns the sender’s points in the intersection) in three different metrics ( $L_\infty, L_1, L_2$ ) in a  $d$ -dimension space where  $d = \{2, 3, 5, 10\}$ , following the Figs. 6, and 7. The proof-of-concept implementation<sup>5</sup> is written in Rust, with less than 1000 lines of code. We use Ristretto and curve25519-dalek to instantiate the underlying group  $\mathbb{G}$ , use FxHash and Blake3 to instantiate the hash function  $H_\gamma, H_{\kappa'}$ . We choose the security parameter  $\lambda = 128$  and statistical parameter  $\kappa = 40$  as usual. To instantiate the OKVS, we follow the construction from [6] but working in  $\mathbb{F}_p$  and the expansion rate  $\epsilon = 0.5$  to make sure we have  $2^{-\kappa}$  correctness error rate. Though it can be optimized to  $\epsilon = 0.1 \sim 0.25$  to have a more compact size, the encoding and decoding time would also increase accordingly.

**Environment.** We run the experiments on an ordinary laptop over a single thread: Macbook Air (M1 2020) with 8 GB RAM and a 2.1 GHz CPU, without using SIMD (e.g., AVX, NEON) optimizations. We measure the entire protocol time in a local network setting (i.e., LAN-like) without considering latency.

### 9.1 Concrete Performance

**Fuzzy PSI.** We mainly consider three cases for fuzzy PSI protocols: The receiver’s points are  $2\delta$ -apart (shown in Table 2), and  $2\delta(d^{\frac{1}{p}} + 1)$ -apart (shown in Table 3). It is worth noting that, any distribution of the receiver’s points can

<sup>5</sup> The open-sourced repository: [https://github.com/sihangpu/fuzzy\\_PSI](https://github.com/sihangpu/fuzzy_PSI).

**Table 2.** Fuzzy PSI when points are  $>2\delta$  (i.e., disjoint balls)

Metric	Radius $\delta$	Dimension $d$	Receiver's $N$	Sender's $M$	Bandwidth	Total Time
$L_\infty$ [20]	30	2	$2^{11}$	$2^{20}$	$\approx 9865^b$ MB	$\gg 1500^a$ s
$L_\infty$	30	2	$2^{11}$	$2^{20}$	173 MB	257.25 s
$L_\infty$	30	5	$2^{13}$	$2^{11}$	231 MB	177.18 s
$L_\infty$	1000	2	$2^{11}$	$2^{11}$	753 MB	303.59 s

<sup>a</sup> Estimated by assuming each PRG evaluation is about 4.8 ns and hash evaluation is about 4.8 ns/byte [25]. Only consider the computational costs at the receiver's side.

<sup>b</sup> Estimated by the concrete bFSS size provided in [20].

**Table 3.** Fuzzy PSI when points are  $>2\delta(d^{\frac{1}{d}} + 1)$ 

Metric	Radius $\delta$	Dimension $d$	Receiver's $N$	Sender's $M$	Bandwidth	Total Time
$L_\infty$ [20]	10	5	$2^{11}$	$2^{20}$	-	$\gg 4300^a$ s
$L_\infty$ [20]	30	10	$2^5$	$2^{20}$	$\approx 10^{11}$ <sup>b</sup> MB	$\gg 10^{13}$ <sup>a</sup> s
$L_\infty$	30	2	$2^{11}$	$2^{20}$	134 MB	99.28 s
$L_\infty$	10	5	$2^{11}$	$2^{20}$	1240 MB	432.42 s
$L_\infty$	30	10	$2^5$	$2^{20}$	1844 MB	1135.63 s
$L_1$	10	2	$2^{11}$	$2^{20}$	107 MB	94.10 s
$L_1$	30	2	$2^{11}$	$2^{20}$	369 MB	111.07 s
$L_2$	10	2	$2^{11}$	$2^{20}$	467 MB	97.37 s
$L_2$	30	2	$2^{11}$	$2^{20}$	3727 MB	121.21 s

<sup>a</sup> Estimated by assuming each PRG evaluation is about 4.8 ns and hash evaluation is about 4.8 ns/byte [25]. Only consider the computational costs at the receiver's side.

<sup>b</sup> Estimated by the concrete bFSS size provided in [20].

be reduced to the disjoint setting by varying the radius. Specifically, for the  $L_\infty$  metric, the second case degrades to  $4\delta$ -apart points; For the  $L_{\{1,2\}}$  metric, our protocol only supports the second case. Our protocols can support large volume balls since our computation and communication cost scaled only sub-linearly to the total volume. In the full version of the paper [2] we also explore the setting where both receiver and sender hold a structured set consisting of hyperballs.

For comparison, we estimate the concrete communication cost for [20] based on their concrete bFSS size table reported in the paper. For the disjoint balls setting we use the reported share sizes for their `spatial hash ◦ sum ◦ tensor ◦ ggm (0.5, 1)-bFSS`, assume bFSS evaluation to cost  $(2 \log \delta)^d$  PRG calls, estimate PRG calls to take 10 machine cycles using AES-NI, and put  $\ell = 440$ . For the distance  $> 4\delta$  setting we use the reported share sizes for their `spatial hash ◦ concat ◦ tt (1 - 1/2d, d)-bFSS`, assume bFSS evaluation to cost 1 machine cycle and put  $\ell = 162$  for dimension  $d = 5$ ,  $\ell = 139$  for dimension  $d = 10$ . In all settings we estimate the correlation-robust hash calls at the end of the protocol to take around 10 machine cycles/byte, based on the fastest performance reported in [25] on 64-byte inputs. We assume a universe size of 32-bit integers for each

dimension. Note that here we report the most conservative estimates for their running time and can only be considered as a loose lower bound.

## 10 Conclusion

In this work, we explored the fuzzy PSI in a more general setting, including higher dimensional space, comprehensive  $L_p$  distance metric, and extended functionality variants. We also demonstrate the practicality of our protocols by experimental results. However, there are still many open problems to be solved, such as, our  $L_p$  protocols have an additional  $O(\delta^p)$  communication overhead for each sender's point which might be expensive when  $\delta$  or  $p$  is too large. Another interesting problem to think is how to get a more efficient protocol in polynomially large dimension space for  $L_2$  distance, or if we can weaken the separated assumption further for  $L_\infty$  distance? We leave them as well as the concrete efficiency optimization to future works. Also, current fuzzy PSI protocols with negligible correctness error require disjoint balls at least. What if the receiver's balls are intersected? Any non-trivial approaches without quadratic overhead would be interesting to explore.

## References

1. Alamati, N., Branco, P., Döttling, N., Garg, S., Hajiabadi, M., Pu, S.: Laconic private set intersection and applications. In: Nissim, K., Waters, B. (eds.) TCC 2021. LNCS, vol. 13044, pp. 94–125. Springer, Cham (2021). [https://doi.org/10.1007/978-3-030-90456-2\\_4](https://doi.org/10.1007/978-3-030-90456-2_4)
2. van Baarsen, A., Pu, S.: Fuzzy private set intersection with large hyperballs. Cryptology ePrint Archive, Paper 2024/330 (2024). <https://eprint.iacr.org/2024/330>
3. Badrinarayanan, S., Miao, P., Raghuraman, S., Rindal, P.: Multi-party threshold private set intersection with sublinear communication. In: Garay, J.A. (ed.) PKC 2021, Part II. LNCS, vol. 12711, pp. 349–379. Springer, Cham (2021). [https://doi.org/10.1007/978-3-030-75248-4\\_13](https://doi.org/10.1007/978-3-030-75248-4_13)
4. Bartusek, J., Garg, S., Jain, A., Policharla, G.V.: End-to-end secure messaging with traceability only for illegal content. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023. LNCS, Part V, Germany, Lyon, France, 23–27 April 2023, vol. 14008, pp. 35–66. Springer, Heidelberg (2023). [https://doi.org/10.1007/978-3-031-30589-4\\_2](https://doi.org/10.1007/978-3-031-30589-4_2)
5. Bhowmick, A., Boneh, D., Myers, S., Talwar, K., Tarbe, K.: The Apple PSI system (2021). <https://www.apple.com/child-safety/pdf/Apple.PSI.System.Security.Protocol.and.Analysis.pdf>
6. Bienstock, A., Patel, S., Seo, J.Y., Yeo, K.: Near-optimal oblivious key-value stores for efficient PSI, PSU and volume-hiding multi-maps. In: USENIX Security Symposium, pp. 301–318. USENIX Association (2023)
7. Branco, P., Döttling, N., Pu, S.: Multiparty cardinality testing for threshold private intersection. In: Garay, J. (ed.) 24th International Conference on Theory and Practice of Public Key Cryptography, PKC 2021. LNCS, Part II, Virtual Event, 10–13 May 2021, vol. 12711, pp. 32–60. Springer, Heidelberg (2021). [https://doi.org/10.1007/978-3-030-75248-4\\_2](https://doi.org/10.1007/978-3-030-75248-4_2)

8. Chakraborti, A., Fanti, G., Reiter, M.K.: Distance-aware private set intersection. In: USENIX Security Symposium. USENIX Association (2023)
9. Chase, M., Miao, P.: Private set intersection in the internet setting from lightweight oblivious PRF. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part III. LNCS, vol. 12172, pp. 34–63. Springer, Cham (2020). [https://doi.org/10.1007/978-3-030-56877-1\\_2](https://doi.org/10.1007/978-3-030-56877-1_2)
10. Chen, H., Huang, Z., Laine, K., Rindal, P.: Labeled PSI from fully homomorphic encryption with malicious security. In: Lie, D., Mannan, M., Backes, M., Wang, X. (eds.) 25th Conference on Computer and Communications Security, ACM CCS 2018, Toronto, ON, Canada, 15–19 October 2018, pp. 1223–1237. ACM Press (2018). <https://doi.org/10.1145/3243734.3243836>
11. Chen, H., Laine, K., Rindal, P.: Fast private set intersection from homomorphic encryption. In: Thuraisingham, B.M., Evans, D., Malkin, T., Xu, D. (eds.) 24th Conference on Computer and Communications Security, ACM CCS 2017, Dallas, TX, USA, 31 October–November 2 2017, pp. 1243–1255. ACM Press (2017). <https://doi.org/10.1145/3133956.3134061>
12. Chmielewski, L., Hoepman, J.: Fuzzy private matching (extended abstract). In: ARES, pp. 327–334. IEEE Computer Society (2008)
13. Chor, B., Gilboa, N., Naor, M.: Private information retrieval by keywords. Cryptology ePrint Archive, Report 1998/003 (1998). <https://eprint.iacr.org/1998/003>
14. Cong, K., et al.: Labeled PSI from homomorphic encryption with reduced computation and communication. In: Vigna, G., Shi, E. (eds.) 28th Conference on Computer and Communications Security, ACM CCS 2021, Virtual Event, Republic of Korea, 15–19 November 2021, pp. 1135–1150. ACM Press (2021). <https://doi.org/10.1145/3460120.3484760>
15. Dong, C., Chen, L., Wen, Z.: When private set intersection meets big data: an efficient and scalable protocol. In: Sadeghi, A.R., Gligor, V.D., Yung, M. (eds.) 20th Conference on Computer and Communications Security, ACM CCS 2013, Berlin, Germany, 4–8 November 2013, pp. 789–800. ACM Press (2013). <https://doi.org/10.1145/2508859.2516701>
16. Duong, T., Phan, D.H., Trieu, N.: Catalic: delegated PSI cardinality with applications to contact tracing. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, Part III. LNCS, vol. 12493, pp. 870–899. Springer, Cham (2020). [https://doi.org/10.1007/978-3-030-64840-4\\_29](https://doi.org/10.1007/978-3-030-64840-4_29)
17. Dupont, P.-A., Hesse, J., Pointcheval, D., Reyzin, L., Yakoubov, S.: Fuzzy password-authenticated key exchange. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018. LNCS, vol. 10822, pp. 393–424. Springer, Cham (2018). [https://doi.org/10.1007/978-3-319-78372-7\\_13](https://doi.org/10.1007/978-3-319-78372-7_13)
18. Freedman, M.J., Nissim, K., Pinkas, B.: Efficient private matching and set intersection. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 1–19. Springer, Heidelberg (2004). [https://doi.org/10.1007/978-3-540-24676-3\\_1](https://doi.org/10.1007/978-3-540-24676-3_1)
19. Garimella, G., Pinkas, B., Rosulek, M., Trieu, N., Yanai, A.: Oblivious key-value stores and amplification for private set intersection. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part II. LNCS, vol. 12826, pp. 395–425. Springer, Cham (2021). [https://doi.org/10.1007/978-3-030-84245-1\\_14](https://doi.org/10.1007/978-3-030-84245-1_14)
20. Garimella, G., Rosulek, M., Singh, J.: Structure-aware private set intersection, with applications to fuzzy matching. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology, CRYPTO 2022, Part I. LNCS, Santa Barbara, CA, USA, 15–18 August 2022, vol. 13507, pp. 323–352. Springer, Heidelberg (2022). [https://doi.org/10.1007/978-3-031-15802-5\\_12](https://doi.org/10.1007/978-3-031-15802-5_12)

21. Garimella, G., Rosulek, M., Singh, J.: Malicious secure, structure-aware private set intersection. In: Handschuh, H., Lysyanskaya, A. (eds.) *Advances in Cryptology, CRYPTO 2023, Part I*. LNCS, Santa Barbara, CA, USA, 20–24 August 2023, vol. 14081, pp. 577–610. Springer, Heidelberg (2023). [https://doi.org/10.1007/978-3-031-38557-5\\_19](https://doi.org/10.1007/978-3-031-38557-5_19)
22. Ghosh, S., Simkin, M.: The communication complexity of threshold private set intersection. In: Boldyreva, A., Micciancio, D. (eds.) *CRYPTO 2019, Part II*. LNCS, vol. 11693, pp. 3–29. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-26951-7\\_1](https://doi.org/10.1007/978-3-030-26951-7_1)
23. Ghosh, S., Simkin, M.: Threshold private set intersection with better communication complexity. In: Boldyreva, A., Kolesnikov, V. (eds.) *26th International Conference on Theory and Practice of Public Key Cryptography, PKC 2023, Part II*. LNCS, Atlanta, GA, USA, 7–10 May 2023, vol. 13941, pp. 251–272. Springer, Heidelberg (2023). [https://doi.org/10.1007/978-3-031-31371-4\\_9](https://doi.org/10.1007/978-3-031-31371-4_9)
24. Huang, Y., Evans, D., Katz, J.: Private set intersection: are garbled circuits better than custom protocols? In: *ISOC Network and Distributed System Security Symposium, NDSS 2012, San Diego, CA, USA, 5–8 February 2012*. The Internet Society (2012)
25. *ECRYPT II: eBACS: ECRYPT Benchmarking of Cryptographic Systems (2023)*. <https://bench.cr.yp.to/results-sha3>
26. Indyk, P., Woodruff, D.P.: Polylogarithmic private approximations and efficient matching. In: Halevi, S., Rabin, T. (eds.) *3rd Theory of Cryptography Conference, TCC 2006*. LNCS, New York, NY, USA, 4–7 March 2006, vol. 3876, pp. 245–264. Springer, Heidelberg (2006). [https://doi.org/10.1007/11681878\\_13](https://doi.org/10.1007/11681878_13)
27. Ion, M., et al.: On deploying secure computing: private intersection-sum-with-cardinality. In: *EuroS&P*, pp. 370–389. IEEE (2020)
28. Kolesnikov, V., Kumaresan, R., Rosulek, M., Trieu, N.: Efficient batched oblivious PRF with applications to private set intersection. In: Weippl, E.R., Katzenbeisser, S., Kruegel, C., Myers, A.C., Halevi, S. (eds.) *23rd Conference on Computer and Communications Security, ACM CCS 2016, Vienna, Austria, 24–28 October 2016*, pp. 818–829. ACM Press (2016). <https://doi.org/10.1145/2976749.2978381>
29. Lindell, Y.: How to simulate it - a tutorial on the simulation proof technique. *Cryptology ePrint Archive, Report 2016/046* (2016). <https://eprint.iacr.org/2016/046>
30. Meadows, C.: A more efficient cryptographic matchmaking protocol for use in the absence of a continuously available third party. In: *S&P*, pp. 134–137. IEEE Computer Society (1986)
31. Muffett, A.: Facebook: password hashing & authentication (2015). <https://rwc.iacr.org/2015/program.html>
32. Naor, M., Reingold, O.: Number-theoretic constructions of efficient pseudo-random functions. In: *38th Annual Symposium on Foundations of Computer Science, Miami Beach, Florida, 19–22 October 1997*, pp. 458–467. IEEE Computer Society Press (1997). <https://doi.org/10.1109/SFCS.1997.646134>
33. Pal, B., et al.: Might I Get Pwned: a second generation compromised credential checking service. In: *USENIX Security Symposium*, pp. 1831–1848. USENIX Association (2022)
34. Pinkas, B., Schneider, T., Tkachenko, O., Yanai, A.: Efficient circuit-based PSI with linear communication. In: Ishai, Y., Rijmen, V. (eds.) *EUROCRYPT 2019*. LNCS, vol. 11478, pp. 122–153. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-17659-4\\_5](https://doi.org/10.1007/978-3-030-17659-4_5)

35. Raghuraman, S., Rindal, P.: Blazing fast PSI from improved OKVS and subfield VOLE. In: Yin, H., Stavrou, A., Cremers, C., Shi, E. (eds.) 29th Conference on Computer and Communications Security, ACM CCS 2022, Los Angeles, CA, USA, 7–11 November 2022, pp. 2505–2517. ACM Press (2022). <https://doi.org/10.1145/3548606.3560658>
36. Rindal, P., Schoppmann, P.: VOLE-PSI: fast OPRF and circuit-PSI from vector-OLE. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021, Part II. LNCS, vol. 12697, pp. 901–930. Springer, Cham (2021). [https://doi.org/10.1007/978-3-030-77886-6\\_31](https://doi.org/10.1007/978-3-030-77886-6_31)
37. Uzun, E., Chung, S.P., Kolesnikov, V., Boldyreva, A., Lee, W.: Fuzzy labeled private set intersection with applications to private real-time biometric search. In: Bailey, M., Greenstadt, R. (eds.) 30th USENIX Security Symposium, USENIX Security 2021, 11–13 August 2021, pp. 911–928. USENIX Association (2021)
38. Ye, Q., Steinfeld, R., Pieprzyk, J., Wang, H.: Efficient fuzzy matching and intersection on private datasets. In: Lee, D., Hong, S. (eds.) ICISC 2009. LNCS, vol. 5984, pp. 211–228. Springer, Heidelberg (2010). [https://doi.org/10.1007/978-3-642-14423-3\\_15](https://doi.org/10.1007/978-3-642-14423-3_15)