The Communication Complexity of Threshold Private Set Intersection

Satrajit Ghosh* and Mark Simkin*

Aarhus University, Denmark
{satrajit, simkin}@cs.au.dk

Abstract. Threshold private set intersection enables Alice and Bob who hold sets S_A and S_B of size n to compute the intersection $S_\mathsf{A} \cap S_\mathsf{B}$ if the sets do not differ by more than some threshold parameter t. In this work, we investigate the communication complexity of this problem and we establish the first upper and lower bounds. We show that any protocol has to have a communication complexity of $\Omega(t)$. We show that an almost matching upper bound of $\tilde{\mathcal{O}}(t)$ can be obtained via fully homomorphic encryption. We present a computationally more efficient protocol based on weaker assumptions, namely additively homomorphic encryption, with a communication complexity of $\tilde{\mathcal{O}}(t^2)$. For applications like biometric authentication, where a given fingerprint has to have a large intersection with a fingerprint from a database, our protocols may result in significant communication savings.
Prior to this work, all previous protocols had a communication complexity of $\Omega(n)$. Our protocols are the first ones with communication complexities that mainly depend on the threshold parameter t and only logarithmically on the set size n.

1 Introduction

Private set intersection enables two mutually distrustful parties Alice and Bob to compute the intersection $S_\mathsf{A} \cap S_\mathsf{B}$ of their respective sets S_A and S_B without revealing any other information. Efficient protocols have numerous applications ranging from botnet detection [NMH+10], through online advertising [PSSZ15], to private contact discovery [Mar14]. The first solution to this problem was given by Meadows [Mea86] and since then, a long line of work [FNP04, KS05, DT10, DCW13, PSZ14, PSSZ15, KKRT16, HV17, KMP+17, RR17a, RR17b, CLR17, GN17, KLS+17, PSWW18] has considered the problem in the two-party, the multi-party, and the server-aided setting with both passive and active security. Beyond private set intersection, several works [KS05, HW06, CGT12, DD15, EFG+15, PSWW18] have also considered protocols for privately computing the size of the set intersection, rather than the intersection itself. Freedman et al. [FNP04] proved a lower bound of $\Omega(n)$ on the communication complexity of any private set intersection protocol, where n is the size of the smallest input set. This lower bound directly extends to the case of protocols that only compute the intersection size and it constitutes a fundamental barrier to the efficiency of these protocols.

In certain scenarios we do not require the full power of private set intersection. For example, for the case of biometric authentication we may want to check whether a given fingerprint reading matches a fingerprint from a database. In this setting, we are neither interested in the concrete intersection nor in the exact size of the intersection. All we care about is a binary answer telling us whether the fingerprints have a large intersection or not. In the case of privacy-preserving ridesharing [HOS17] two users only want to share a ride if large parts of their trajectories on a map intersect. In this case, the users may be interested in the concrete intersection of their routes, but only if the intersection is large. Yet another example can be found in the online dating world, where two potential love birds Alice and Bob are only interested in learning the intersection of their dating preferences if the intersection thereof is sufficiently large. Speaking more abstractly, this problem is known as threshold private set intersection, where Alice and Bob hold sets of size n each and only want to learn the intersection if their sets do not differ by more than t elements. Only a few works [FNP04, HOS17, GN17, PSWW18, ZC18] have considered this problem and all of them present

* Supported by the European Unions's Horizon 2020 research and innovation program under grant agreement No 669255 (MPCPRO) and No 731583 (SODA).

solutions, whose communication complexity scales at least linearly in the size of the smaller input set. This seems to be somewhat inherent to these works, since all of them start from a private set intersection protocol and then massage it until it becomes a threshold private set intersection protocol. In this work we ask:

What is the communication complexity of threshold private set intersection?

Answering this question is both theoretically and practically relevant. As explained above, threshold and regular private set intersection protocols have many applications. A better understanding of their communication complexities and their qualitative differences provides us with a better understanding of this research area. It enables us to pick the right tool for a given job and it allows us to have a firm understanding of the communication complexities that we can expect. From a practical perspective, overcoming the private set intersection lower bound of $\Omega(n)$ may result in significant efficiency gains for applications that only require threshold private set intersection. For example, in the biometric authentication setting one usually only allows for a very small difference between a stored and a given fingerprint. We show that using threshold private set intersection protocols, the communication complexity can be almost completely independent of the total size of the fingerprints and instead only depends on the maximum allowed difference between the two fingerprints.

1.1 Our Contribution

We initiate the study of sublinear (in the set size) threshold private set intersection and provide a first characterization of its communication complexity. We prove a lower bound of $\Omega(t)$ on the communication complexity of any protocol that computes the intersection of two sets that do not differ by more than t elements. We present an almost matching upper bound of $\tilde{\mathcal{O}}(t)$ based on fully homomorphic encryption. We show how to avoid the use of fully homomorphic encryption by presenting a computationally more efficient protocol based on weaker assumptions, namely additively homomorphic encryption, with communication complexity of $\tilde{\mathcal{O}}(t^2)$. For applications, where the set intersection has to be large and thus t is small, our protocols may result in significant improvements over the state-of-the-art in terms of communication complexity.

Along the way we also present a communication efficient protocol for private intersection cardinality testing, which privately computes whether two sets differ by more than a given threshold t or not. We believe that this protocol may be of independent interest. From a conceptual perspective, our paper highlights somewhat surprising connections between threshold private set intersection, set reconciliation protocols [MTZ03] from distributed systems, and sparse polynomial interpolation [BOT88], which have to the best of our knowledge not been known before.

What this paper is not about. Most existing works on private set intersection aim to develop the most practically efficient protocols. At the same time, many basic theoretical questions about private set intersection remain unanswered. The goal of this work to provide first answers to one such question. We hope that the research direction initiated in this work will eventually lead to asymptotically optimal *and* practically efficient protocols. The results in this paper present several novel techniques to provide the first non-trivial feasibility results for sublinear threshold private set intersection, which we believe to be of theoretical importance, but we do not claim them to be practically useful yet.

1.2 Technical Overview

Our main threshold private set intersection protocol can be split into two subprotocols. One for testing, whether two given sets are "similar enough" and one for computing the set intersection of two such similar sets. Here we highlight some of the main ideas underlying our protocols.

Private Intersection Cardinality Testing. The goal of private intersection cardinality testing is to enable Alice and Bob, who hold sets S_A and S_B of elements from a field $\mathbb{F}_p$, to determine, whether their sets are similar or not. More formally, we have some similarity threshold parameter t and we would like to test whether $|(S_\mathsf{A} \setminus S_\mathsf{B}) \cup (S_\mathsf{B} \setminus S_\mathsf{A})| \leq 2t$ without revealing any other information about the sets. Our solution to this problem is based on the idea of encoding sets as polynomials over a field as has been done in numerous previous works [BK89, MTZ03, FNP04, KS05]. However, in contrast to previous works, which encode the elements of a set into the roots of a polynomial, we encode the elements into separate monomials of a polynomial. Our encoding procedure encodes a set $S_\mathsf{A} = \{a_1, \ldots, a_n\}$ as a polynomial $\mathsf{p}_\mathsf{A}(x) = \sum_{i=1}^{n} x^{a_i}$. The main idea behind this encoding is that, given two encoded sets $\mathsf{p}_\mathsf{A}(x)$ and $\mathsf{p}_\mathsf{B}(x)$, the number of monomials in the polynomial $\mathsf{p}(x) = \mathsf{p}_\mathsf{A}(x) - \mathsf{p}_\mathsf{B}(x)$ corresponds to the size of the symmetric set difference between S_A and S_B. In particular, if $|(S_\mathsf{A} \setminus S_\mathsf{B}) \cup (S_\mathsf{B} \setminus S_\mathsf{A})| \leq 2t$, then $\mathsf{p}(x)$ has at most $2t$ monomials. Encoding the sets in such a way, allows us to make use of the polynomial sparsity test of Grigorescu et al. [GJR10], which itself is heavily based on the seminal work of Ben-Or and Tiwari [BOT88]. A polynomial $\mathsf{p}(x)$ is called t-sparse if it has at most t monomials. Grigorescu et al. present a randomized algorithm that only requires $2t$ evaluations of $\mathsf{p}(x)$ to determine, whether the polynomial is t-sparse or not. To obtain our private intersection cardinality test, we combine the ideas above with additively homomorphic encryption and the privacy-preserving linear algebra techniques of Kiltz et al. [KMWF07]. Our resulting protocol has a communication complexity of $\tilde{\mathcal{O}}(t^2)$.

Threshold Private Set Intersection. For the problem of threshold private set intersection, our starting point is the set reconciliation protocol by Minsky et al. [MTZ03], where Alice and Bob hold sets S_A and S_B and would like to compute the set union $S_\mathsf{A} \cup S_\mathsf{B}$ in a communication efficient manner. As shown by Minsky et al., Alice and Bob can do this with communication complexity proportional to the size of the symmetric set difference, that is, with communication complexity roughly $\tilde{\mathcal{O}}((|S_{\mathsf{A}\setminus\mathsf{B}}| + |S_{\mathsf{B}\setminus\mathsf{A}}|) \log p)$ bits. This is asymptotically close to optimal, since at the very least both parties need to exchange the data elements that are not part of the intersection $S_\mathsf{A} \cap S_\mathsf{B}$. The set reconciliation protocol by Minsky et al. starts by encoding both sets as monic polynomials, where the roots of the polynomial correspond to the elements of the set. For a set $S_\mathsf{A} = \{a_1, \ldots, a_n\}$, the corresponding polynomial is $\mathsf{p}_\mathsf{A}(x) = \prod_{i=1}^{n} (x - a_i)$. The degree $\mathsf{deg}(\mathsf{p}_\mathsf{A})$ of the polynomial equals the set size n and since p_A is monic, it can be interpolated from n evaluation points. The main observation behind Minsky et al.'s protocol is that

$$\mathsf{p}(x) := \frac{\mathsf{p}_\mathsf{A}(x)}{\mathsf{p}_\mathsf{B}(x)} = \frac{\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(x)}{\mathsf{p}_{\mathsf{B}\setminus\mathsf{A}}(x)}$$

If we divide the two polynomials representing the sets, then the common factors of $\mathsf{p}_\mathsf{A}(x)$ and $\mathsf{p}_\mathsf{B}(x)$ cancel out and what remains is a rational polynomial[1], where the numerator represents the elements exclusively contained in S_A and the denominator represents the elements only contained in S_B. It is straightforward to see that if S_A and S_B do not differ by more than $2t$ elements, that is if $|S_{\mathsf{A}\setminus\mathsf{B}}| + |S_{\mathsf{B}\setminus\mathsf{A}}| \leq 2t$, then $\mathsf{deg}(\mathsf{p}) = \mathsf{deg}(\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}) + \mathsf{deg}(\mathsf{p}_{\mathsf{B}\setminus\mathsf{A}}) \leq 2t$ and we can interpolate p from $2t$ evaluation points via rational function interpolation[2]. The second observation behind Minsky et al.'s protocol is that we can compute evaluation points of $\mathsf{p}(x)$ from evaluation points of $\mathsf{p}_\mathsf{A}(x)$ and $\mathsf{p}_\mathsf{B}(x)$. To evaluate p at location α, both Alice and Bob first separately evaluate $\mathsf{p}_\mathsf{A}(x)$ and $\mathsf{p}_\mathsf{B}(x)$ at α and then jointly compute $\mathsf{p}(\alpha) = \frac{\mathsf{p}_\mathsf{A}(\alpha)}{\mathsf{p}_\mathsf{B}(\alpha)}$.

Based on these observations the set union protocol by Minsky et al. roughly works as follows. Let us assume that we already know that the sets do not differ by more than $2t$ elements. First, both Alice and Bob encode their sets as polynomials as described above. Both parties separately evaluate their polynomials on some pre-agreed set of evaluation points $\{\alpha_1, \ldots, \alpha_{2t}\}$ to obtain $\{\mathsf{p}_\mathsf{A}(\alpha_1), \ldots, \mathsf{p}_\mathsf{A}(\alpha_{2t})\}$ and $\{\mathsf{p}_\mathsf{B}(\alpha_1), \ldots, \mathsf{p}_\mathsf{B}(\alpha_{2t})\}$. After exchanging their sets of polynomial evaluations, both parties use rational interpolation to compute the polynomial $\mathsf{p}(x) = \frac{\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(x)}{\mathsf{p}_{\mathsf{B}\setminus\mathsf{A}}(x)}$. Given $\mathsf{p}(x)$, for example Alice, learns the denominator $\mathsf{p}_{\mathsf{B}\setminus\mathsf{A}}(x)$ and computes

[1] A rational polynomial is the fraction of two polynomials. See Section 2.1 for details.
[2] See [MTZ03] for details on rational function interpolation over a field.

an encoding of the set union $\mathsf{p}_{\mathsf{A}\cup\mathsf{B}}(x) = \mathsf{p}_{\mathsf{A}}(x) \cdot \mathsf{p}_{\mathsf{B}\setminus\mathsf{A}}(x)$. Importantly for us we observe that apart from computing the set union, Alice can also compute the set intersection by computing $\mathsf{p}_{\mathsf{A}\cap\mathsf{B}}(x) = \frac{\mathsf{p}_{\mathsf{A}}(x)}{\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(x)}$. The key observation here is that in order to compute the intersection, it is sufficient for Alice to learn which elements are exclusive to her set. In case of a "large" intersection, this quantity is much smaller than the size of the sets or the size of the intersection.

Given Minsky et al.'s protocol, one possible approach towards constructing a sublinear private set intersection protocol (for similar sets) would be to combine it with a generic protocol for secure two-party computation. Both parties input evaluation points of their polynomials, using a secure computation protocol we interpolate $\mathsf{p}(x)$, and finally output $\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(x)$ and $\mathsf{p}_{\mathsf{B}\setminus\mathsf{A}}(x)$ to Alice and Bob respectively. Unfortunately, this does not seem to result in a practically or asymptotically efficient protocol. In order to interpolate $\mathsf{p}(x)$, one would have to perform a gaussian elimination inside the secure computation protocol. For a system of linear equations with $\mathcal{O}(t)$ unknowns, this requires $\mathcal{O}(t^3)$ operations.

We take a very different approach. We only make minimal use of generic secure computation to obtain "noisy" evaluation points of p. Using these points, Alice can then in plain interpolate a rational polynomial $\frac{\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(x)}{U(x)}$, where $U(x)$ is a uniformly random polynomial. From this polynomial Alice can learn $\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(x)$ and therefore $\mathsf{p}_{\mathsf{A}\cap\mathsf{B}}(x)$, but nothing else beyond that.

2 Preliminaries

Notation. Let λ be the computational and κ the statistical security parameter. For a set S, we write $v \leftarrow S$ to denote that v is chosen uniformly at random from S. For a possibly randomized algorithm A, we write $v \leftarrow A(x)$ to denote a run of A on input x that produces output v. For $n \in \mathbb{N}$, we write $[n] := \{1, 2, \ldots, n\}$. We write $|S|$ for the number of elements in S. We use $\tilde{\mathcal{O}}(\cdot)$ as a variant of the big-O notation that ignores polylog factors.

Sets. Throughout most of the paper we will assume that the sets of Alice and Bob are of equal size n. We show how to deal with sets of different sizes in Section 6.4. We assume that the set elements come from a field $\mathbb{F}_p$, where p is a $\Theta(\kappa)$-bit prime.

Size of the Intersection vs. Size of the Symmetric Set Difference. We will measure the "similarity" of two sets S_{A} and S_{B} in terms of size of their symmetric set difference. In some scenarios it may be more convenient to measure the similarity of two sets in terms of intersection size. These two measures are equivalent. A *lower* bound t_{min} on the intersection set size $|S_{\mathsf{A}} \cap S_{\mathsf{B}}|$, corresponds to a *upper* bound $t_{\mathsf{max}} = 2\,(n - t_{\mathsf{min}})$ on the size of the symmetric set difference $|\,(S_{\mathsf{A}} \setminus S_{\mathsf{B}}) \cup (S_{\mathsf{B}} \setminus S_{\mathsf{A}})\,|$.

2.1 Linear Algebra

We recall some terminology and definitions from linear algebra.

Matrices. Let $\mathbb{F}_p^{k\times k}$ be the set of k-by-k square matrices with entries from $\mathbb{F}_p$. A matrix $M \in \mathbb{F}_p^{k\times k}$ is said to be invertible, if there exists a matrix M^{-1}, such that $M \cdot M^{-1} = I$, where I is the identity matrix. A matrix that is not invertible is called singular. A matrix M is singular if and only if it has determinant 0.

Polynomials. Let $\mathsf{p}(x) = \sum_{i=0}^{n} a_i x^i$ be a polynomial. We call $\{a_0, \ldots, a_n x^n\}$ the monomials and $\{a_0, \ldots, a_n\}$ the coefficients of the polynomial. The degree $\mathsf{deg}(\mathsf{p})$ of a polynomial $\mathsf{p}(x)$ is the the largest i, such that the monomial $a_i x^i \neq 0$. A polynomial is said to be monic if for $i = \mathsf{deg}(\mathsf{p})$, we have $a_i = 1$. We write $\mathbb{F}_p[X]$ to denote the set of polynomials with coefficients from the field $\mathbb{F}_p$. A polynomial $\mathsf{p}(x) \in \mathbb{F}_p[X]$ of degree d is uniquely defined and can be efficiently interpolated form $d+1$ evaluation points $\{(\alpha_1, \mathsf{p}(\alpha_1)), \ldots, (\alpha_{d+1}, \mathsf{p}(\alpha_{d+1}))\}$

via Lagrange interpolation. If $\mathsf{p}(x)$ is monic, then d points suffice. A polynomial $h(x) = \frac{\mathsf{p}(x)}{q(x)}$, where $\mathsf{p}(x)$, $\mathsf{q}(x)$ are polynomials of degree n and m, is called a rational polynomial or rational function. It can be interpolated, uniquely up to constants, from $n+m+1$ points [MTZ03]. If $\mathsf{p}(x)$ and $\mathsf{q}(x)$ are monic, then $n+m$ points suffice. A polynomial $\mathsf{p}(x)$ is said to be ℓ-sparse if has at most ℓ monomials, i.e. if $|\{a_i x^i \mid a_i \neq 0\}| \leq \ell$.

Our main construction in Section 6 will make use of an observation about polynomials due to Kissner and Song [KS05]. For the sake of concreteness we restate their lemma[3] here in a slightly less general fashion, which is tailored to our needs.

Lemma 1 ([KS05]). *Let p be a prime. Let $\mathsf{p}(x), \mathsf{q}(x) \in \mathbb{F}_p[X]$ be polynomials of degree $d \leq t$ with $\mathsf{gcd}(\mathsf{p}(x), \mathsf{q}(x)) = 1$. Let $\mathsf{R}_1(x), \mathsf{R}_2(x) \in \mathbb{F}_p[X]$ be two uniformly random polynomials of degree t. Then $\mathsf{U}(x) = \mathsf{p}(x) \cdot \mathsf{R}_1(x) + \mathsf{q}(x) \cdot \mathsf{R}_2(x)$ is a uniformly random polynomial of degree at most $2t$.*

Another basic observation about polynomials that we will need, is captured in Lemma 2. Simply speaking it states that for some given polynomial $\mathsf{p}(x)$ of degree d_p and some uniformly random polynomial $\mathsf{R}(x)$ of degree d_R, the probability that the polynomials share a common root negligible in the statistical security parameter κ.

Lemma 2. *Let p be a $\Theta(\kappa)$-bit prime. Let $\mathsf{p}(x) \in \mathbb{F}_p[X]$ be an arbitrary but fixed non-zero polynomial of degree at most d_p and let $\mathsf{R}(x) \in \mathbb{F}_p[X]$ be a uniformly random polynomial of degree at most d_R. Then*

$$\Pr[\mathsf{gcd}(\mathsf{p}(x), \mathsf{R}(x)) \neq 1] \leq \mathsf{negl}(\kappa)$$

Proof (sketch). The gcd of $\mathsf{p}(x)$ and $\mathsf{R}(x)$ equals to one if and only if the two polynomials share no common roots. A uniformly random polynomial $\mathsf{R}(x)$ of degree d_R has at most d_R roots, which are distributed uniformly at random. The probability of picking one random root that is not a root of $\mathsf{p}(x)$ is $1 - \frac{d_p}{p}$. It follows that

$$\begin{aligned}\Pr[\mathsf{gcd}(\mathsf{p}(x), \mathsf{R}(x)) \neq 1] &= 1 - \Pr[\mathsf{gcd}(\mathsf{p}(x), \mathsf{R}(x)) = 1] \\ &= 1 - (1 - \frac{d_p}{p})^{d_R} \\ &\leq \mathsf{negl}(\kappa)\end{aligned}$$

2.2 Secure Two-Party Computation

Our security definitions are given in the universal composability (UC) framework of Canetti [Can01]. We provide a brief overview here and refer the reader to [MQU07, CDN15] for a more complete summary of the security model.

We consider a two-party protocol Π that is supposed to implement some ideal functionality $\mathcal{F}$. Security is defined by comparing two processes. In the real process the two parties execute the protocol Π. The protocol itself is allowed to make use of an idealized functionality $\mathcal{G}$. An environment $\mathcal{Z}$ chooses the inputs of all parties, it models everything that is external to the protocol, and it represents the adversary, who attacks the protocol. $\mathcal{Z}$ may corrupt a party and get access to that party's internal tapes. In the ideal process, two dummy parties send their inputs to the ideal functionality $\mathcal{F}$ and get back the output of the computation. In such an ideal process, a simulator $\mathcal{S}$, also known as the ideal world adversary, emulates $\mathcal{Z}$'s view of a real protocol execution. $\mathcal{S}$ has full control of the corrupted dummy party. $\mathcal{S}$ emulates $\mathcal{Z}$'s view of that party as well as its communication with $\mathcal{G}$. At the end of both executions $\mathcal{Z}$ outputs a single bit. Let $\mathsf{REAL}_\lambda[\mathcal{Z}, \Pi, \mathcal{G}]$, respectively $\mathsf{IDEAL}_\lambda[\mathcal{Z}, S, \mathcal{F}]$, be the random variable denoting $\mathcal{Z}$'s final output bit in the real, respectively ideal, process. We say Π securely implements $\mathcal{F}$, if no environment $\mathcal{Z}$ can distinguish whether it has been part of a real or ideal process.

[3] The lemma we are referring to here is Lemma 2 in the paper of Kissner and Song.

Definition 1. *Π securely implements functionality $\mathcal{F}$ with respect to a class of environments Env in the $\mathcal{G}$-hybrid model, if there exists a simulator $\mathcal{S}$ such that for all $\mathcal{Z} \in Env$ we have*

$$|\Pr[\mathsf{REAL}_\lambda[\mathcal{Z}, \Pi, \mathcal{G}] = 1] - \Pr[\mathsf{IDEAL}_\lambda[\mathcal{Z}, S, \mathcal{F}] = 1]| \leq \mathsf{negl}(\lambda)$$

In this paper, we focus on static passive adversaries. We consider environments $\mathcal{Z}$ that get full read-only access to a corrupted party's internal tapes. The corrupted party follows the protocol honestly.

2.3 Additively Homomorphic Encryption.

We recall the definition of additively homomorphic encryption and the associated IND-CPA security notion.

Definition 2 (Public Key Encryption Scheme). *A public key encryption scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ consists of three algorithms:*

$\mathsf{KeyGen}(1^\lambda)$**:** The key generation algorithm takes as input the security parameter 1^λ and outputs a key pair $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Enc}(\mathsf{pk}, m)$**:** The encryption algorithm takes as input the public key pk, a message $m \in \mathcal{M}$, and outputs a ciphertext c.

$\mathsf{Dec}(\mathsf{sk}, c)$**:** The decryption algorithm takes as input the secret key sk, the ciphertext $c' \in \mathcal{C}$, and outputs a plaintext m.

We say $\mathcal{E}$ is additively homomorphic if we can add encrypted values and multiply them by plaintext constants. Concretely, if there exist operations $\boxplus$ and $\boxdot$, such that for any $a, b \in \mathcal{M}$ and any two ciphertexts $c_1 = \mathsf{Enc}(\mathsf{pk}, m_1)$ and $c_2 = \mathsf{Enc}(\mathsf{pk}, m_2)$, it holds that $(a \boxdot c_1) \boxplus (b \boxdot c_2) = \mathsf{Enc}(\mathsf{pk}, a \cdot m_1 + b \cdot m_2)$. For the sake of simplicity and readability we will use the same notation for algebraic operations on the plaintext and algebraic operations on the ciphertext space. We assume that it will be clear from the context which one is meant. Possible instantiations of such a cryptosystem are the Paillier cryptosystem [Pai99] or its generalization the Damgård-Jurik cryptosystem [DJ01]. We will furthermore assume that the message space of the encryption scheme is a field[4].

Definition 3 (Indistinguishability under Chosen Plaintext Attacks). *Let $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be a (homomorphic) encryption scheme and let $\mathcal{A}$ be a PPT adversary. We say $\mathcal{E}$ is* IND-CPA*-secure if for all PPT adversaries $\mathcal{A}$ it holds that*

$$\Pr\left[b = b' : \begin{array}{c} (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda) \\ (m_0, m_1) \leftarrow \mathcal{A}(\mathsf{pk}) \\ b \leftarrow \{0, 1\} \\ c \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b) \\ b' \leftarrow \mathcal{A}(c) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(x)$$

2.4 Oblivious Linear Function Evaluation.

Oblivious linear function evaluation allows a receiver to obliviously evaluate a linear function that is only known to the sender. Concretely, the sender has two input values $a, b \in \mathbb{F}$ that determine a linear function $f(x) = a \cdot x + b$ over $\mathbb{F}$ and the receiver holds input $x \in \mathbb{F}$. The receiver will learn only $f(x)$, and the sender learns nothing about the evaluation point x. The corresponding ideal functionality $\mathcal{F}_{\text{OLE}}$ is depicted in Figure 1. Several efficient instantiations, both in the passive and malicious settings, exist [NP99, IPS09, ADI+17, GNN17].

[4] For the case of the Paillier cryptosystem this is strictly speaking not the case, since not every element from the message space has an inverse. However, finding an element that does not have an inverse is as hard as breaking the security of the cryptosystem. Therefore, we can treat the message space as if it was an actual field.

$\mathcal{F}_{\text{OLE}}$

The sender has input $(a, b) \in \mathbb{F}^2$ and the receiver has input $x \in \mathbb{F}$.

- Upon receiving a message $(\texttt{inputS}, (a, b))$ from the sender with a, b, store a and b.
- Upon receiving a message $(\texttt{inputR}, x)$ from the receiver with x, store x.
- Compute $y = a \cdot x + b$ and send $(\texttt{output}, y)$ to the receiver.

Fig. 1. Oblivious Linear Function Evaluation Functionality.

3 Lower and Upper Bounds

To provide a better understanding of what is possible and what is not, we present upper and a lower bounds for the communication complexity of threshold private set intersection protocols. We prove unconditionally that any threshold private set intersection protocol has to have a communication complexity of $\Omega(t)$, where t is an upper bound on the size of the symmetric set difference. We show how to obtain an almost matching upper bound of $\tilde{\mathcal{O}}(t)$ using fully homomorphic encryption [RAD78, Gen09, BGV12]. Due to its computational complexity, this bound seems to be mainly of theoretical interest. We sketch a construction based on simpler assumptions, namely garbled circuits [Yao86], with a communication complexity of $\tilde{\mathcal{O}}(t^3)$. In light of these results, our main protocol, which we will describe in the following sections, places itself in between those bounds. It has a communication complexity of $\tilde{\mathcal{O}}(t^2)$ and is thus asymptotically more efficient than the garbled circuit solution. It is based on weaker assumptions, namely additively homomorphic encryption, and is computationally more efficient than the construction based on fully homomorphic encryption. A visual illustration of these results can be found in Figure 2.

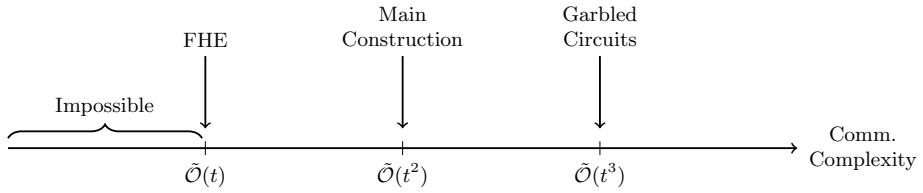

Fig. 2. An illustration what is possible and what is not in terms of communication complexity of threshold private intersection protocols. t is the upper bound on the symmetric set difference of the two sets.

Lower Bound for Threshold Private Set Intersection. To prove our lower bound for threshold private set intersection, we will make use of a known lower bound for the disjointness problem. In the disjointness problem, Alice and Bob hold two n-bit vectors a and b, and would like to compute the function

$$\text{Dis}(a, b) = \begin{cases} 0 & \text{if } \exists i : a_i = b_i = 1 \\ 1 & \text{Otherwise} \end{cases}$$

A series of results [BFS86, KS92, Raz90, BYJKS04] have established that the communication complexity of this function is $\Theta(n)$. Freedman et al. [FNP04] observed that these results directly yield a lower bound of $\Omega(n)$ on the communication complexity of any set intersection protocol for sets of size n. We sketch how these results also provide a lower bound for threshold private set intersection. Assume towards contradiction that for sets of size n', which have a intersection of size at least $n' - t$, there exists a protocol that computes their

intersection with communication complexity $o(t)$. We can use such a protocol to construct a set intersection protocol for sets of size n with complexity $o(n)$. We interpret n as the parameter t and let both parties pad their sets with some common pre-agreed dummy elements such that their their total set sizes are n'. The parties then run the threshold set intersection protocol. Since both parties are guaranteed to have $n' - t$ elements in common, the protocol will compute the intersection of their sets with communication complexity $o(t) = o(n)$, which leads to a contradiction.

Upper Bound from Fully Homomorphic Encryption. We sketch how to combine the set reconciliation protocol of Minsky et al. [MTZ03] with fully homomorphic encryption [RAD78, Gen09, BGV12] to obtain an almost matching upper bound of $\tilde{\mathcal{O}}(t)$. We provide a high-level description of the construction here and leave the details to the interested reader. Fully homomorphic encryption allows anyone to evaluate arbitrary circuits over encrypted data without being able to decrypt. Known instantiations are based on lattice based assumptions, such as learning with errors [BV11a] or the ring learning with errors [BV11b]. Fully homomorphic encryption leads to a conceptually very simple and communication efficient solution for general secure two party computation. Alice encrypts her data and sends it to Bob. Bob encrypts his data and homomorphically evaluates the desired function on their joint encrypted data. He sends back the result to Alice, who can decrypt the result of the computation. The communication complexity only depends on the size of the inputs and the size of the output, but importantly it does not depend on the size of the evaluated circuit.

Using fully homomorphic encryption, we let Alice and Bob execute a variation of Minsky et al.'s protocol. Alice encodes her set S_A as a polynomial $\mathsf{p}_\mathsf{A}(x) = \prod_{i=1}^{n}(x - a_i)$ and sends Bob encrypted evaluations $\{\mathsf{p}_\mathsf{A}(\alpha_1), \ldots, \mathsf{p}_\mathsf{A}(\alpha_{2t})\}$ as well as an additional encrypted evaluation $\mathsf{p}_\mathsf{A}(z)$ and the uniformly random z itself in the clear. Bob evaluates his set as a polynomial on the same points and then homomorphically interpolates the rational function $\frac{\mathsf{p}_\mathsf{A}(x)}{\mathsf{p}_\mathsf{B}(x)} = \frac{\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(x)}{\mathsf{p}_{\mathsf{B}\setminus\mathsf{A}}(x)}$, where the gcd of numerator and denominator is 1, using the first $2t$ encrypted points to obtain a candidate polynomial p. Bob computes

$$(\mathsf{p}(x), \mathsf{p}_\mathsf{A}(z), \mathsf{p}_\mathsf{B}(z), z) \mapsto \begin{cases} \mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(x) & \text{if } \mathsf{p}(z) = \frac{\mathsf{p}_\mathsf{A}(z)}{\mathsf{p}_\mathsf{B}(z)} \\ \perp & \text{Otherwise} \end{cases}$$

on the encrypted data and sends back the result to Alice. The correctness of this approach directly follows from the correctness of Minsky et al.'s protocol. Security follows from the security of fully homomorphic encryption. The total communication consists of Alice sending $2t + 1$ ciphertexts to Bob and him sending the coefficients of the polynomial in the numerator, i.e. t ciphertexts, to Alice. Assuming the ciphertexts are larger than the corresponding plaintexts by at most a multiplicative constant and assuming that the set elements are drawn from $\mathbb{F}_p$, we can conclude that the total communication complexity is $\mathcal{O}(t \log p)$ bits. Despite its nice communication complexity, this solution has two drawbacks. From a theoretical perspective, it relies on fully homomorphic encryption and thus can only be instantiated from lattice based assumptions. From a practical perspective, it does not seem to be anywhere near practical due to the fact that one has to homomorphically perform a rational polynomial interpolation on the ciphertexts, which leads to a high computational complexity.

Using Garbled Circuits. A simple, but asymptotically inefficient solution based on one-way functions and oblivious transfer can be obtained by using garbled circuits [Yao86] instead of fully homomorphic encryption. For garbled circuits, the communication complexity corresponds to the size of the circuit that is being evaluated. Following the same approach as above, the size of the circuit is dominated by the rational interpolation logic. Using gaussian elimination this step requires $\mathcal{O}(t^3)$ operations, which leads to a total communication complexity of at least $\tilde{\mathcal{O}}(t^3)$ bits.

4 Intersection Cardinality Testing

An important building block for our threshold private set intersection protocol in Section 6, is a intersection cardinality testing protocol, which enables two parties to check whether their sets differ by more than a given threshold $2t$ with communication complexity $\tilde{\mathcal{O}}(t)$. We present a non-private solution based on polynomial sparsity testing here and show how to obtain a privacy-preserving version thereof in Section 5. We believe that the non-private as well as the private intersection cardinality test may be of independent interest.

From a conceptual perspective, our protocol is very simple. It is basically a direct application of the polynomial sparsity test of Grigorescu et al. [GJR10] to an appropriate encoding of sets as polynomials. We encode a set $S_{\mathsf{A}} = \{a_1, \ldots, a_n\}$ as a polynomial $\mathsf{p}_{\mathsf{A}}(x) = \sum_{i=1}^{n} x^{a_i}$. The main idea behind this encoding is that the sparsity of the polynomial $\mathsf{p}_{\mathsf{A}}(x) - \mathsf{p}_{\mathsf{B}}(x)$ corresponds to the size of the symmetric set difference of S_{A} and S_{B}. The protocol Π_{ICT}^{2t} is described in Figure 3.

Π_{ICT}^{2t}

Alice and Bob have as input set $S_{\mathsf{A}} = \{a_1, \ldots, a_n\} \in \mathbb{F}_p^n$ and $S_{\mathsf{B}} = \{b_1, \ldots, b_n\} \in \mathbb{F}_p^n$ respectively.

Protocol:

1. Alice and Bob encode their sets as polynomials $\mathsf{p}_{\mathsf{A}}(x) = \sum_{i=1}^{n} x^{a_i}$ and $\mathsf{p}_{\mathsf{B}}(x) = \sum_{i=1}^{n} x^{b_i}$ in $\mathbb{F}_q[X]$.
2. Alice picks uniformly random $u \leftarrow \mathbb{F}_q$.
3. Alice computes the Hankel matrix

$$H_{\mathsf{A}} = \begin{bmatrix} \mathsf{p}_{\mathsf{A}}(u^0) & \mathsf{p}_{\mathsf{A}}(u^1) & \ldots & \mathsf{p}_{\mathsf{A}}(u^{2t}) \\ \mathsf{p}_{\mathsf{A}}(u^1) & \mathsf{p}_{\mathsf{A}}(u^2) & \ldots & \mathsf{p}_{\mathsf{A}}(u^{2t+1}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathsf{p}_{\mathsf{A}}(u^{2t}) & \mathsf{p}_{\mathsf{A}}(u^{2t+1}) & \ldots & \mathsf{p}_{\mathsf{A}}(u^{4t}) \end{bmatrix}$$

   and sends it along with u to Bob.
4. Bob, using his own Hankel matrix H_{B}, computes $H_{\mathsf{C}} = H_{\mathsf{A}} - H_{\mathsf{B}}$.
5. If $\det(H_{\mathsf{C}}) = 0$, then Bob outputs `similar`, otherwise he outputs `different`.

Fig. 3. Protocol for intersection cardinality testing based on the polynomial sparsity testing protocol of Grigorescu et al. [GJR10].

Theorem 1. *Let S_{A} and S_{B} be subsets of $\mathbb{F}_p$. Let $q > (4t^2+2t)(p-1)2^{\kappa}$ be a prime power. Π_{ICT}^{2t} has a communication overhead of $4t+1$ field elements from $\mathbb{F}_q$. If $|S_{\mathsf{A}\setminus\mathsf{B}}|+|S_{\mathsf{B}\setminus\mathsf{A}}| \leq 2t$, then $\Pr[\Pi_{\mathrm{ICT}}^{2t}$ outputs $\mathsf{similar}] = 1$ and if $|S_{\mathsf{A}\setminus\mathsf{B}}| + |S_{\mathsf{B}\setminus\mathsf{A}}| > 2t$, then $\Pr[\Pi_{\mathrm{ICT}}^{2t}$ outputs $\mathsf{similar}] \leq 1 - 2^{-\kappa}$.*

Proof. The original algorithm of Grigorescu et al. [GJR10] takes an arbitrary polynomial p as its input, computes the corresponding Hankel matrix H, and then computes the determinant thereof. We essentially directly apply their algorithm to the polynomial $\mathsf{p}_{\mathsf{C}}(x) = \mathsf{p}_{\mathsf{A}}(x) - \mathsf{p}_{\mathsf{B}}(x)$. We exploit the fact that we can compute the Hankel matrix H_C of $\mathsf{p}_{\mathsf{C}}(x)$ by first computing the Hankel matrices H_A and H_B. The correctness and the parameters of the randomized polynomial sparsity testing protocol directly follow from the test of Grigorescu et al.[5]

It remains to show that the sparsity of the computed polynomial does indeed reflect the size of the symmetric set difference. If S_{A} and S_{B} have an intersection of size k, then the polynomial $\mathsf{p}_{\mathsf{A}}(x) - \mathsf{p}_{\mathsf{B}}(x)$ will

[5] See Theorem 3 in their work.

have exactly $2(n-k)$ monomials. Thus, if $|S_{\mathsf{A}\setminus\mathsf{B}}| + |S_{\mathsf{B}\setminus\mathsf{A}}| < 2t$, then $k > n - t$ and therefore $\mathsf{p}_{\mathsf{A}}(x) - \mathsf{p}_{\mathsf{B}}(x)$ will be a $2t$-sparse polynomial. □

Efficiency. To get a better idea of what this theorem means in terms of concrete efficiency, it is worth looking at some common real world parameter settings. For instance, for sets of 64-bit integers, a statistical security parameter $\kappa = 40$, and a threshold t of size at most 2^{20}, we roughly require a 128-bit modulus q.

5 Private Intersection Cardinality Testing

We obtain a privacy-preserving version of the intersection cardinality test from Section 4 via a combination of homomorphic encryption and the matrix singularity test due to Kiltz et al. [KMWF07]. The singularity test enables Alice, who holds a encrypted matrix over a finite field, and Bob, who holds the decryption key, to test whether the matrix is singular or not. Recall, that a matrix being singular and it having determinant 0 are equivalent statements. Let $\mathcal{F}_{\text{INV}}$ be the corresponding ideal functionality, which either returns `singular` or `invertible`. Kiltz et al. show how to securely and efficiently implement such an functionality using additively homomorphic encryption.

Theorem 2 ([KMWF07]). *Let $M \in \mathbb{F}_q^{k\times k}$ be the encrypted matrix. Assuming* IND-CPA*-secure additively homomorphic encryption, the ideal functionality $\mathcal{F}_{\text{INV}}$ can be realized securely with communication complexity $\mathcal{O}(k^2 \log q \log k)$ in $\mathcal{O}(\log k)$ rounds with security against a passive adversary. The protocol is correct with probability $1 - \frac{k+1}{q}$, which for a q chosen as in Theorem 1 is overwhelming in κ.*

For our choice of q (see Theorem 1) the protocol of Kiltz et al. fails with negligible (in κ) probability. In the following, for the sake of simplicity, we will assume that the corresponding ideal functionality $\mathcal{F}_{\text{INV}}$ has perfect correctness. All of our protocols and proofs trivially extend to the case, where the ideal functionality errs with a negligible probability.

5.1 Ideal Functionality

The ideal functionality $\mathcal{F}_{\text{PICT}}^{2t}$ for private intersection cardinality testing is depicted in Figure 4. Alice and Bob send their input sets S_{A} and S_{B} to the ideal functionality, which checks whether the sets differ by more than $2t$ elements. It outputs `different` if this is the case and it outputs `similar` otherwise. Note that our ideal functionality is size hiding in the sense that the environment $\mathcal{Z}$ does not learn the size of the input sets of Alice or Bob.

$\mathcal{F}_{\text{PICT}}^{2t}$

Alice and Bob have as input set S_{A} and S_{B} respectively.

- Upon receiving message $(\texttt{inputA}, S_{\mathsf{A}})$ from Alice, store S_{A}.
- Upon receiving message $(\texttt{inputB}, S_{\mathsf{B}})$ from Bob, store S_{B}.
- If $|\,(S_{\mathsf{A}} \setminus S_{\mathsf{B}}) \cup (S_{\mathsf{B}} \setminus S_{\mathsf{A}})\,| \leq 2t$, then the functionality outputs `similar`, otherwise `different`, to Alice and Bob.

Fig. 4. Ideal functionality for private intersection cardinality testing.

5.2 Protocol

Our private intersection cardinality test Π^{2t}_{PICT} closely follows its non-private counterpart Π^{2t}_{ICT} from Section 4. The main difference is that we now encrypt the Hankel matrix of Alice before sending it to Bob. Upon receiving Alice's encrypted matrix, Bob exploits the homomorphic properties of the encryption scheme to compute the Hankel matrix that corresponds to the polynomial encoding of the symmetric set difference. Using $\mathcal{F}_{\text{INV}}$, Bob learns whether the matrix is singular or invertible and thus learns, whether the intersection of the two sets is large enough. The protocol Π^{2t}_{PICT} is depicted in Figure 5.

Π^{2t}_{PICT}

Alice and Bob have as input set $S_{\mathsf{A}} = \{a_1, \ldots, a_n\} \in \mathbb{F}_p^n$ and $S_{\mathsf{B}} = \{b_1, \ldots, b_n\} \in \mathbb{F}_p^n$ respectively.

Protocol:

1. Alice and Bob encode their sets as polynomials $\mathsf{p}_{\mathsf{A}}(x) = \sum_{i=1}^{n} x^{a_i}$ and $\mathsf{p}_{\mathsf{B}}(x) = \sum_{i=1}^{n} x^{b_i}$ in $\mathbb{F}_q[X]$.
2. Alice picks uniformly random $u \leftarrow \mathbb{F}_q$.
3. Alice samples an encryption key $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ and computes an encrypted version of her Hankel matrix
$$H_{\mathsf{A}} = \begin{bmatrix} c_0 & c_1 & \ldots & c_{2t} \\ c_1 & c_2 & \ldots & c_{2t+1} \\ \vdots & \vdots & \ddots & \vdots \\ c_{2t} & c_{2t+1} & \ldots & c_{4t} \end{bmatrix}$$
where $c_i \leftarrow \mathsf{Enc}(pk, \mathsf{p}_{\mathsf{A}}(u^i))$.
4. Alice sends H_{A}, u, and pk to Bob.
5. Bob computes his own encrypted Hankel matrix H_{B} and computes $H_{\mathsf{C}} = H_{\mathsf{A}} - H_{\mathsf{B}}$.
6. Alice sends sk and Bob sends H_{C} to the ideal functionality $\mathcal{F}_{\text{INV}}$.
7. If Bob gets back `singular` from $\mathcal{F}_{\text{INV}}$, then he sends `similar` to Alice and outputs `similar` himself, otherwise he sends and outputs `different`.

Fig. 5. Protocol for private intersection cardinality testing.

5.3 Security

Theorem 3. *Let q be as in Theorem 1. Let $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be a* IND-CPA *secure additively homomorphic encryption scheme. Then Π^{2t}_{PICT} securely implements $\mathcal{F}^{2t}_{\text{PICT}}$ with communication complexity $\tilde{\mathcal{O}}(t^2)$ in the $\mathcal{F}_{\text{INV}}$-hybrid model with security against a passive adversary and overwhelming (in κ) correctness.*

Proof. Either Alice or Bob can be corrupted. We consider the two cases separately.

Alice corrupt. In this case, security holds trivially. The environment corrupting Alice learns nothing beyond the input and output of the computation.

Bob corrupt. The simulator $\mathcal{S}$ sends Bob's input to the ideal functionality and obtains $\mathsf{result} \in \{\mathsf{similar}, \mathsf{different}\}$. $\mathcal{S}$ picks a uniformly random $u \leftarrow \mathbb{F}_q$ and samples an encryption key $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$. It computes

$c_i \leftarrow \mathsf{Enc}(pk, 0)$ for $0 \leq i \leq 4t$.

$$H_{\mathsf{A}} = \begin{bmatrix} c_0 & c_1 & \dots & c_{2t} \\ c_1 & c_2 & \dots & c_{2t+1} \\ \vdots & \vdots & \ddots & \vdots \\ c_{2t} & c_{2t+1} & \dots & c_{4t} \end{bmatrix}$$

The simulator leaks u, pk, and H_{A} to $\mathcal{Z}$. At this point Bob would send some matrix H_{C} to the ideal functionality $\mathcal{F}_{\text{INV}}$, which is also simulated by $\mathcal{S}$. If $\mathsf{result} = \mathsf{similar}$, then the simulator leaks $\mathsf{singular}$ to $\mathcal{Z}$. Otherwise the simulator leaks $\mathsf{invertible}$. The only difference between the environment's view in a real and a simulated protocol execution is the matrix H_{A}. In a real execution it contains encrypted evaluations of Alice's polynomial. In the simulated execution it contains encryptions of 0. Indistinguishability of the real and ideal process follows directly from the $\mathsf{IND\text{-}CPA}$ security of the encryption scheme. □

6 Threshold Private Set Intersection

In this section we present our threshold private set intersection protocol, which proceeds as follows. First, Alice and Bob use $\mathcal{F}_{\text{PICT}}^{2t}$ to determine whether their sets differ by more than $2t$ elements. If the ideal functionality outputs $\mathsf{different}$, the parties output $\perp$. If it outputs $\mathsf{similar}$, the parties engage in a secure set intersection protocol, which has a communication complexity of $\tilde{\mathcal{O}}(t)$ bits.

6.1 Ideal Functionality

The ideal functionality $\mathcal{F}_{\text{TPSI}}^{2t}$ for threshold private set intersection is depicted in Figure 6. Alice and Bob send their input sets S_{A} and S_{B} to the ideal functionality, which checks whether the sets differ by more than $2t$ elements. If this is the case, the functionality returns $\perp$ to both parties. If this is not the case, the functionality returns the set intersection $S_{\mathsf{A}} \cap S_{\mathsf{B}}$ to both Alice and Bob.

$\mathcal{F}_{\text{TPSI}}^{2t}$

Alice and Bob have as input sets S_{A} and S_{B} respectively.

- Upon receiving message $(\texttt{inputA}, S_{\mathsf{A}})$ from Alice, store S_{A} and leak $(\texttt{inputA}, |S_{\mathsf{A}}|)$ to the environment $\mathcal{Z}$.
- Upon receiving message $(\texttt{inputB}, S_{\mathsf{B}})$ from Bob, store S_{B} and leak $(\texttt{inputB}, |S_{\mathsf{B}}|)$ to the environment $\mathcal{Z}$.
- If $|\,(S_{\mathsf{A}} \setminus S_{\mathsf{B}}) \cup (S_{\mathsf{B}} \setminus S_{\mathsf{A}})\,| \leq 2t$, then the functionality outputs $A \cap B$, otherwise $\perp$, to Alice and Bob.

Fig. 6. Threshold Private Set Intersection functionality.

6.2 Protocol

Our protocol loosely follows the approach of Minsky et al.'s [MTZ03] set reconciliation protocol. Assume that the sets of Alice and Bob do not differ by more than t elements. Both Alice and Bob encode their sets as polynomials over a field, where the roots of the polynomials are the elements of the corresponding set. Let $\mathsf{p}_{\mathsf{A}}(x) = \prod_{i=1}^{n}(x - a_i)$ and $\mathsf{p}_{\mathsf{B}}(x) = \prod_{i=1}^{n}(x - b_i)$ be those polynomials. Ideally, we would like to directly apply Minsky et al.'s protocol to interpolate $\mathsf{p}(x) = \frac{\mathsf{p}_{\mathsf{B}}(x)}{\mathsf{p}_{\mathsf{A}}(x)} = \frac{\mathsf{p}_{\mathsf{B}\setminus\mathsf{A}}(x)}{\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(x)}$ from which both Alice and Bob could

compute the intersection of their sets. For example, Alice could extract[6] $\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(x)$ from $\mathsf{p}(x)$ and compute the intersection polynomial as $\frac{\mathsf{p}_{\mathsf{A}}(x)}{\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(x)}$. Unfortunately, Alice would learn more information than she should, since she could also simply extract $\mathsf{p}_{\mathsf{B}\setminus\mathsf{A}}(x)$ and learn Bob's entire set.

As discussed before, one possible solution is to use generic secure two-party computation for interpolating p and separating the numerator and denominator. Due to the complexity of the computational task, this does not seem to result in a asymptotically or practically efficient solution. Our protocol takes a different approach and only makes minimal use of generic secure two-party computation. We only use it to transform evaluation points of p into a noisy versions thereof. Using these noisy evaluation points, Alice and Bob can perform the interpolation in plain to compute the set intersection without learning the other party's input.

In our construction, we will make use of a noisy polynomial addition functionality functionality $\mathcal{F}_{\text{NPA}}^{(3t+1,\,t)}$, which takes the polynomials $\mathsf{p}_{\mathsf{A}}(x)$ and $\mathsf{p}_{\mathsf{B}}(x)$ of Alice and Bob as its input and outputs noisy evaluation points $\{\mathsf{V}(\alpha_1),\ldots,\mathsf{V}(\alpha_{3t+1})\}$, where $\mathsf{V}(x) = \mathsf{p}_{\mathsf{A}}(x)\cdot\mathsf{R}_1(x) + \mathsf{p}_{\mathsf{B}}(x)\cdot\mathsf{R}_2(x)$. The polynomials R_1 and R_2 are uniformly random polynomials of degree t. We show how to efficiently instantiate this functionality with communication complexity $\tilde{\mathcal{O}}(t)$ in Section 7. Our protocol is presented in Figure 7.

Π_{TPSI}^{2t}

Alice and Bob have as input set $S_{\mathsf{A}} = \{a_1,\ldots,a_n\} \in \mathbb{F}_p^n$ and $S_{\mathsf{B}} = \{b_1,\ldots,b_n\} \in \mathbb{F}_p^n$ respectively.
The values $\alpha_1,\ldots,\alpha_{3t+1} \in \mathbb{F}_p$ are fixed and publicly known.

Protocol:

1. Alice and Bob send S_{A} and S_{B} to $\mathcal{F}_{\text{PICT}}^{2t}$.
2. If $\mathcal{F}_{\text{PICT}}^{2t}$ returns `different`, both parties output $\perp$.
3. Alice and Bob encode their sets as polynomials $\mathsf{p}_{\mathsf{A}}(x) = \prod_{i=1}^{n}(x-a_i)$ and $\mathsf{p}_{\mathsf{B}}(x) = \prod_{i=1}^{n}(x-b_i)$.
4. Alice and Bob send $\mathsf{p}_{\mathsf{A}}(x)$ and $\mathsf{p}_{\mathsf{B}}(x)$ to $\mathcal{F}_{\text{NPA}}^{(3t+1,\,t)}$.
5. Both receive evaluation points $(\mathsf{V}(\alpha_1),\ldots,\mathsf{V}(\alpha_{3t+1}))$, where $\mathsf{V}(\alpha_i) = \mathsf{p}_{\mathsf{A}}(\alpha_i)\cdot\mathsf{R}_1(\alpha_i) + \mathsf{p}_{\mathsf{B}}(\alpha_i)\cdot\mathsf{R}_2(\alpha_i)$ and $\mathsf{R}_1,\mathsf{R}_2$ are uniformly random polynomials of degree t.
6. Alice computes $\mathsf{p}(\alpha_i) = \frac{\mathsf{V}(\alpha_i)}{\mathsf{p}_{\mathsf{A}}(\alpha_i)} = \frac{\mathsf{U}(\alpha_i)}{\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(\alpha_i)}$ for each $i \in [3t+1]$ and interpolates p with gcd of numerator and denominator being 1 from these points.
7. Alice performs the following steps (Bob works analogously):
   (a) Alice sets $S_{\mathsf{A}\cap\mathsf{B}} := S_{\mathsf{A}}$.
   (b) For each $a_i \in S_{\mathsf{A}}$, if $\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(a_i) = 0$, Alices removes a_i from $S_{\mathsf{A}\cap\mathsf{B}}$.
   (c) Alice outputs $S_{\mathsf{A}\cap\mathsf{B}}$.

Fig. 7. Protocol for securely computing the intersection of two sets that do not differ by more than $2t$ points.

6.3 Security

Theorem 4. *Protocol Π_{TPSI}^{2t} securely implements $\mathcal{F}_{\text{TPSI}}^{2t}$ with communication complexity $\tilde{\mathcal{O}}(t)$ in the ($\mathcal{F}_{\text{PICT}}^{2t}$, $\mathcal{F}_{\text{NPA}}^{(3t+1,\,t)}$)-hybrid model with security against a passive adversary.*

Proof. We first show that our protocol indeed produces the correct result and we then go on to prove its security.

[6] separating the numerator and denominator from a given rational polynomial is easy here, because we obtain the coefficients of both separately during the interpolation step.

Correctness. If S_A and S_B differ by more than $2t$ elements, then both parties output $\perp$ and terminate in step 2 of the protocol. If on the other hand $|\,(S_\mathsf{A} \setminus S_\mathsf{B}) \cup (S_\mathsf{B} \setminus S_\mathsf{A})\,| \leq 2t$, then since $|S_\mathsf{A}| = |S_\mathsf{B}|$, it follows that $|\,(S_\mathsf{A} \setminus S_\mathsf{B})\,| \leq t$ and $|\,(S_\mathsf{B} \setminus S_\mathsf{A})\,| \leq t$. Alice computes polynomial

$$\begin{aligned}\mathsf{p}(\alpha_i) = \frac{\mathsf{V}(\alpha_i)}{\mathsf{p}_\mathsf{A}(\alpha_i)} &= \frac{\mathsf{p}_\mathsf{A}(\alpha_i) \cdot \mathsf{R}_1(\alpha_i) + \mathsf{p}_\mathsf{B}(\alpha_i) \cdot \mathsf{R}_2(\alpha_i)}{\mathsf{p}_\mathsf{A}(\alpha_i)} \\ &= \frac{\mathsf{p}_{\mathsf{A}\cap\mathsf{B}}(\alpha_i) \cdot \mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(\alpha_i) \cdot \mathsf{R}_1(\alpha_i) + \mathsf{p}_{\mathsf{A}\cap\mathsf{B}}(\alpha_i) \cdot \mathsf{p}_{\mathsf{B}\setminus\mathsf{A}}(\alpha_i) \cdot \mathsf{R}_2(\alpha_i)}{\mathsf{p}_{\mathsf{A}\cap\mathsf{B}}(\alpha_i) \cdot \mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(\alpha_i)} \\ &= \frac{\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(\alpha_i) \cdot \mathsf{R}_1(\alpha_i) + \mathsf{p}_{\mathsf{B}\setminus\mathsf{A}}(\alpha_i) \cdot \mathsf{R}_2(\alpha_i)}{\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(\alpha_i)}\end{aligned}$$

The numerator is a polynomial of degree at most $2t$ and the denominator is a polynomial of degree at most t. It follows that she can interpolate $\mathsf{p}(x)$ from $3t+1$ points. The polynomial in the denominator encodes the elements that are only in Alice's set and thus she can learn the intersection by removing those elements from her set S_A. By Lemma 2 we are certain that, with overwhelming probability, no root in the denominator $\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}$ will be cancelled out by accident from the remaining random numerator.

Security. We assume that Alice is corrupt. The proof, where Bob is corrupt is completely symmetrical. The simulator sends Alice's input set to the ideal functionality $\mathcal{F}_{\text{TPSI}}^{2t}$ and either obtains $\mathsf{result} = \perp$ or the intersection $\mathsf{result} = S_{\mathsf{A}\cap\mathsf{B}}$. In the first step of the protocol, Alice would send her set S_A to the ideal functionality $\mathcal{F}_{\text{PICT}}^{2t}$, which is simulated by the simulator $\mathcal{S}$. If $\mathsf{result} = \perp$, then $\mathcal{S}$ returns $\mathsf{different}$ as the ideal functionality's answer to $\mathcal{Z}$. Otherwise, $\mathcal{S}$ answers with $\mathsf{similar}$. In case the protocol did not terminate, Alice would continue by sending $\mathsf{p}_\mathsf{A}(x)$ to $\mathcal{F}_{\text{NPA}}^{(3t+1,\,t)}$. At this point, the simulator needs to construct a polynomial $\mathsf{V}(x)$ for responding to Alice's query. In a real protocol execution the polynomial would be

$$\mathsf{V}(x) = \mathsf{p}_{\mathsf{A}\cap\mathsf{B}}(x) \underbrace{\left(\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(x) \cdot \mathsf{R}_1(x) + \mathsf{p}_{\mathsf{B}\setminus\mathsf{A}}(x) \cdot \mathsf{R}_2(x)\right)}_{\mathsf{U}(x):=}$$

Since $|S_\mathsf{A}| = |S_\mathsf{B}|$, it follows that $|S_{\mathsf{A}\setminus\mathsf{B}}| = |S_{\mathsf{B}\setminus\mathsf{A}}|$ and thus $\deg\big(\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}\big) = \deg\big(\mathsf{p}_{\mathsf{B}\setminus\mathsf{A}}\big)$. Furthermore, we know that $\deg\big(\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}(x)\big) = |S_\mathsf{A}| - |S_{\mathsf{A}\cap\mathsf{B}}|$. From these observations we can conclude that the degree of $\deg(\mathsf{U}) = |S_\mathsf{A}| - |S_{\mathsf{A}\cap\mathsf{B}}| + t$. The simulator $\mathcal{S}$ picks a uniformly random polynomial $\mathsf{U}(x)$ of that degree and for $1 \leq i \leq 3t+1$, it computes the polynomial evaluations $\mathsf{V}(\alpha_i) = \mathsf{U}(\alpha_i) \cdot \mathsf{p}_{\mathsf{A}\cap\mathsf{B}}(\alpha_i)$ and leaks them to $\mathcal{Z}$ as the output of $\mathcal{F}_{\text{NPA}}^{(3t+1,\,t)}$. The environment's view in a real and in a simulated process only differs in the way we choose the polynomial $\mathsf{V}(x)$. We know that $\deg\big(\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}\big) = \deg\big(\mathsf{p}_{\mathsf{B}\setminus\mathsf{A}}\big) \leq t$, that $\gcd\big(\mathsf{p}_{\mathsf{A}\setminus\mathsf{B}}, \mathsf{p}_{\mathsf{B}\setminus\mathsf{A}}\big) = 1$, and that $\deg(\mathsf{R}_1) = \deg(\mathsf{R}_2) = t$. Indistinguishability of the real and the simulated process directly follows from Lemma 1. □

6.4 Dealing with Sets of Different Sizes

Throughout the paper we have so far assumed that the sets of Alice and Bob are of the same size. This was done for the sake of simplicity, but is not necessary in general. Consider two sets S_A and S_B, where, without loss of generality, we assume that $|S_\mathsf{B}| > |S_\mathsf{A}|$. Independently of their actual intersection size, the symmetric set difference of the two sets will be at least $t_{\mathsf{min}} := |S_\mathsf{B}| - |S_\mathsf{A}|$. This means that the threshold parameter t in our privacy-preserving protocols would need to be at least t_{min}. Since the set sizes are known, we can simply pad S_A to the size of S_B with dummy elements and adapt our difference threshold to $t_{\mathsf{new}} := t + |S_\mathsf{B}| - |S_\mathsf{A}| \leq 2t$ accordingly. The simple approach of padding the smaller set to the size of the larger one would thus increase the communication complexity of our protocols by at most a small constant factor. The relation between the size of the symmetric set difference and total set sizes is illustrated in Figure 8.

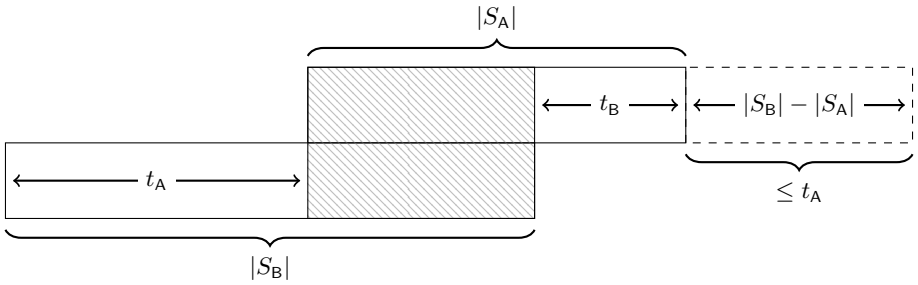

Fig. 8. An illustration of how the size of the symmetric set difference behaves for sets of different sizes. The set intersection between the sets S_A and S_B is indicated by the shaded gray area. The size of the symmetric set difference $(S_{\mathsf{A}\setminus\mathsf{B}} \cup S_{\mathsf{B}\setminus\mathsf{A}})$ is $t_\mathsf{A} + t_\mathsf{B}$. The dotted rectangle on the right illustrates the amount of padding we would have to perform to make the two sets be of the same size. Padding S_A to the size of S_B would increase the symmetric set difference by at most t_A.

7 Instantiation

We show how to efficiently instantiate the noisy polynomial addition functionality $\mathcal{F}_{\text{NPA}}^{\ell,\,t}$ from Section 6 using oblivious linear function evaluation.

7.1 Ideal Functionality

The ideal functionality $\mathcal{F}_{\text{NPA}}^{\ell,\,t}$, depicted in Figure 9, for noisy polynomial addition takes polynomials $\mathsf{p}_\mathsf{A}(x)$ and $\mathsf{p}_\mathsf{B}(x)$ of degree n from Alice and Bob as input, and returns back ℓ evaluation points of $\mathsf{p}_\mathsf{A}(x) \cdot \mathsf{R}_1(x) + \mathsf{p}_\mathsf{B}(x) \cdot \mathsf{R}_2(x)$, where $\mathsf{R}_1(x)$ and $\mathsf{R}_2(x)$ are uniformly random polynomials of degree t.

$\mathcal{F}_{\text{NPA}}^{\ell,\,t}$

Alice and Bob have inputs $\mathsf{p}_\mathsf{A}(x)$ and $\mathsf{p}_\mathsf{B}(x)$ of degree n.
Let $\{\alpha_1, \ldots, \alpha_\ell\}$ be a set of publicly known distinct points in $\mathbb{F}$.

- Upon receiving message $(\texttt{inputS}, \mathsf{p}_\mathsf{A}(x))$ from Alice, where $\mathsf{p}_\mathsf{A}(x) \in \mathbb{F}[X]$, store the polynomial $\mathsf{p}_\mathsf{A}(x)$.
- Upon receiving message $(\texttt{inputR}, \mathsf{p}_\mathsf{B}(x))$ from Bob, where $\mathsf{p}_\mathsf{B}(x) \in \mathbb{F}[X]$, store polynomial $\mathsf{p}_\mathsf{B}(x)$.
- Once both parties have submitted their polynomials, pick uniformly random polynomials $\mathsf{R}_1(x), \mathsf{R}_2(x) \in \mathbb{F}[X]$ of degree t and set $\mathsf{p}(x) = \mathsf{p}_\mathsf{A}(x) \cdot \mathsf{R}_1(x) + \mathsf{p}_\mathsf{B}(x) \cdot \mathsf{R}_2(x)$.
- Send ℓ evaluation points $\{\mathsf{p}(\alpha_1), \cdots, \mathsf{p}(\alpha_\ell)\}$ to Alice and Bob.

Fig. 9. Noisy Polynomial Addition Functionality.

7.2 Protocol

Our starting point is a protocol by Ghosh and Nilges [GN17], which implements a very similar functionality in the $\mathcal{F}_{\text{OLE}}$-hybrid model. In their protocol the sender inputs a polynomial $\mathsf{p}_\mathsf{A}(x)$ and random polynomial $\mathsf{R}(x)$, the receiver inputs a polynomial $\mathsf{p}_\mathsf{B}(x)$ and gets back a noisy polynomial $\mathsf{p}_\mathsf{A}(x) + \mathsf{R}(x) \cdot \mathsf{p}_\mathsf{B}(x)$. We use a modified version of their protocol to instantiate our $\mathcal{F}_{\text{NPA}}^{\ell,\,t}$ functionality.

In our protocol, both Alice and Bob evaluate their input polynomials on the evaluation points $\{\alpha_1, \ldots, \alpha_\ell\}$. Alice picks two uniformly random polynomials $\mathsf{R}_1^{\mathsf{A}}(x), \mathsf{R}_2^{\mathsf{A}}(x)$ of degree t and a random polynomial $\mathsf{U}_{\mathsf{A}}(x)$ of degree ℓ. Bob picks two random polynomials $\mathsf{R}_1^{\mathsf{B}}(x), \mathsf{R}_2^{\mathsf{B}}(x)$ of degree t and a random polynomial $\mathsf{U}_{\mathsf{B}}(x)$ of degree ℓ. Now Alice and Bob will invoke the $\mathcal{F}_{\text{OLE}}$ ideal functionality 2ℓ times, where Alice will act as the receiver in the first ℓ and as the sender in the last ℓ invocations. In the first ℓ instances, for each $i \in [\ell]$, Alice inputs evaluation points $\mathsf{p}_{\mathsf{A}}(\alpha_i)$, Bob inputs $(\mathsf{R}_1^{\mathsf{B}}(\alpha_i), \mathsf{U}_{\mathsf{B}}(\alpha_i))$, and Alice receives back $\mathsf{s}_{\mathsf{A}}(\alpha_i) = \mathsf{p}_{\mathsf{A}}(\alpha_i) \cdot \mathsf{R}_1^{\mathsf{B}}(\alpha_i) + \mathsf{U}_{\mathsf{B}}(\alpha_i)$. In the next ℓ instances, for each $i \in [\ell]$, Bob inputs $\mathsf{p}_{\mathsf{B}}(\alpha_i)$, Alice inputs $(\mathsf{R}_2^{\mathsf{A}}(\alpha_i), \mathsf{U}_{\mathsf{A}}(\alpha_i))$, and Bob receives back $\mathsf{s}_{\mathsf{B}}(\alpha_i) = \mathsf{p}_{\mathsf{B}} \cdot \mathsf{R}_2^{\mathsf{A}}(\alpha_i) + \mathsf{U}_{\mathsf{A}}(\alpha_i)$. For each $i \in [\ell]$, Alice sends $\mathsf{s}'_{\mathsf{A}}(\alpha_i) = \mathsf{s}_{\mathsf{A}}(\alpha_i) + \mathsf{p}_{\mathsf{A}}(\alpha_i) \cdot \mathsf{R}_1^{\mathsf{A}}(\alpha_i) - \mathsf{U}_{\mathsf{A}}(\alpha_i)$ to Bob, who can then compute

$$
\begin{aligned}
&\mathsf{p}_{\mathsf{A}}(\alpha_i) \cdot \mathsf{R}_1(\alpha_i) + \mathsf{p}_{\mathsf{B}} \cdot \mathsf{R}_2(\alpha_i) := \\
&\mathsf{s}_{\mathsf{B}}(\alpha_i) + \mathsf{s}'_{\mathsf{A}}(\alpha_i) + \mathsf{p}_{\mathsf{B}}(\alpha_i) \cdot \mathsf{R}_2^{\mathsf{B}}(\alpha_i) - \mathsf{U}_{\mathsf{B}}(\alpha_i) = \\
&\mathsf{p}_{\mathsf{A}}(\alpha_i) \cdot \left(\mathsf{R}_1^{\mathsf{A}}(\alpha_i) + \mathsf{R}_1^{\mathsf{B}}(\alpha_i)\right) + \mathsf{p}_{\mathsf{B}} \cdot \left(\mathsf{R}_2^{\mathsf{A}}(\alpha_i) + \mathsf{R}_2^{\mathsf{B}}(\alpha_i)\right)
\end{aligned}
$$

In a completely symmetrical fashion, Bob sends $\mathsf{s}'_{\mathsf{B}}(\alpha_i) = \mathsf{s}_{\mathsf{B}}(\alpha_i) + \mathsf{p}_{\mathsf{B}}(\alpha_i) \cdot \mathsf{R}_1^{\mathsf{B}}(\alpha_i) - \mathsf{U}_{\mathsf{B}}(\alpha_i)$ to Alice, who can then compute the same evaluation points of their noisy polynomial addition $\mathsf{p}_{\mathsf{A}}(\alpha_i) \cdot \mathsf{R}_1(\alpha_i) + \mathsf{p}_{\mathsf{B}} \cdot \mathsf{R}_2(\alpha_i)$. Our protocol is described formally in Figure 10.

Theorem 5. *$\Pi_{\text{NPA}}^{\ell,t}$ implements $\mathcal{F}_{\text{NPA}}^{\ell,t}$ in the $\mathcal{F}_{\text{OLE}}$-hybrid model with security against a passive adversary.*

Proof (Sketch). We assume that Alice is corrupt. The proof, where Bob is corrupt is completely symmetrical. The simulator sends Alice's input p_{A} to the ideal functionality $\mathcal{F}_{\text{NPA}}^{\ell,t}$ and gets back $\{\mathsf{p}(\alpha_1), \cdots, \mathsf{p}(\alpha_\ell)\}$. The simulator picks the polynomials $\mathsf{U}_{\mathsf{B}}(x)$ and $\mathsf{U}_{\mathsf{A}}(x)$ of degree ℓ uniformly at random. It then picks $\mathsf{p}_{\mathsf{B}}(x)$ of degree n and $\mathsf{R}_1^{\mathsf{A}}(x), \mathsf{R}_2^{\mathsf{A}}(x), \mathsf{R}_1^{\mathsf{B}}(x), \mathsf{R}_2^{\mathsf{B}}(x)$ of degree t uniformly at random under the constraint that

$$
\mathsf{p}(\alpha_i) = \mathsf{p}_{\mathsf{A}}(\alpha_i) \cdot \left(\mathsf{R}_1^{\mathsf{A}}(\alpha_i) + \mathsf{R}_1^{\mathsf{B}}(\alpha_i)\right) + \mathsf{p}_{\mathsf{B}}(\alpha_i) \cdot \left(\mathsf{R}_2^{\mathsf{A}}(\alpha_i) + \mathsf{R}_2^{\mathsf{B}}(\alpha_i)\right)
$$

Using these values the simulator computes $\mathsf{s}_{\mathsf{A}}(\alpha_i), \mathsf{s}_{\mathsf{B}}(\alpha_i), \mathsf{s}'_{\mathsf{A}}(\alpha_i), \mathsf{s}'_{\mathsf{B}}(\alpha_i)$ as in the protocol description.

During the first ℓ invocations of $\mathcal{F}_{\text{OLE}}$, Alice would send $(\texttt{inputR}, \mathsf{p}_{\mathsf{A}}(\alpha_i))$ to the ideal functionality $\mathcal{F}_{\text{OLE}}$. The simulator leaks $\mathsf{s}_{\mathsf{A}}(\alpha_i)$ to $\mathcal{Z}$ as the response that Alice would receive from $\mathcal{F}_{\text{OLE}}$. During the next ℓ invocations of $\mathcal{F}_{\text{OLE}}$ Alice would send $(\texttt{inputS}, (\mathsf{R}_2^{\mathsf{A}}(\alpha_i), \mathsf{U}_{\mathsf{A}}(\alpha_i)))$, but does not receive anything back, hence we do not need to simulate anything here. Finally we leak $\mathsf{s}'_{\mathsf{B}}(\alpha_i)$ to $\mathcal{Z}$ as the message that she would receive in step 6. The only difference between a real protocol execution and our simulation is the choice of $\mathsf{p}_{\mathsf{B}}(x)$, which in turn influences the value of $\mathsf{s}'_{\mathsf{B}}(\alpha_i)$. However, since

$$
\mathsf{p}(\alpha_i) = \mathsf{s}_{\mathsf{A}}(\alpha_i) + \mathsf{s}'_{\mathsf{B}}(\alpha_i) + \mathsf{p}_{\mathsf{A}}(\alpha_i) \cdot \mathsf{R}_1^{\mathsf{A}}(\alpha_i) - \mathsf{U}_{\mathsf{A}}(\alpha_i)
$$

we have that

$$
\mathsf{s}'_{\mathsf{B}}(\alpha_i) = \mathsf{p}(\alpha_i) - \mathsf{p}_{\mathsf{A}}(\alpha_i) \cdot \mathsf{R}_1^{\mathsf{A}}(\alpha_i) - \mathsf{p}_{\mathsf{A}}(\alpha_i) \cdot \mathsf{R}_1^{\mathsf{B}}(\alpha_i) + \mathsf{U}_{\mathsf{A}}(\alpha_i) - \mathsf{U}_{\mathsf{B}}(\alpha_i)
$$

At this point we observe that the values $\mathsf{s}'_{\mathsf{B}}(\alpha_i)$ are distributed uniformly at random, since we only learn ℓ evaluation points and since $\mathsf{U}_{\mathsf{B}}(x)$ is a uniformly random polynomial of degree ℓ, which is not known to Alice. □

Efficiency. The communication complexity of $\Pi_{\text{NPA}}^{\ell,t}$ essentially depends on the communication complexity of the $\mathcal{F}_{\text{OLE}}$ functionality. Using a passively secure instantiations of $\mathcal{F}_{\text{OLE}}$ with constant communication overhead [NP99, IPS09, ADI+17], we obtain a instantiation for $\mathcal{F}_{\text{NPA}}^{\ell,t}$ with communication complexity $\mathcal{O}(\ell \log p)$.

$\Pi^{\ell,t}_{\text{NPA}}$

Let $\{\alpha_1, \ldots, \alpha_\ell\}$ be a set of publicly known distinct points in $\mathbb{F}_p$. Alice and Bob have inputs $\mathsf{p}_\mathsf{A}(x) \in \mathbb{F}_p[X]$ and $\mathsf{p}_\mathsf{B}(x) \in \mathbb{F}_p[X]$ of degree n each.

Protocol:

1. Alice picks $\mathsf{R}^\mathsf{A}_1(x), \mathsf{R}^\mathsf{A}_2(x) \in \mathbb{F}_p[X]$ of degree t and $\mathsf{U}_\mathsf{A}(x) \in \mathbb{F}_p[X]$ of degree ℓ uniformly at random.
2. Bob picks $\mathsf{R}^\mathsf{B}_1(x), \mathsf{R}^\mathsf{B}_2(x) \in \mathbb{F}_p[X]$ of degree t and $\mathsf{U}_\mathsf{B}(x) \in \mathbb{F}[X]$ of degree ℓ uniformly at random.
3. For each $i \in [\ell]$
   - Alice sends $(\texttt{inputR}, \mathsf{p}_\mathsf{A}(\alpha_i))$ to $\mathcal{F}_{\text{OLE}}$.
   - Bob sends $(\texttt{inputS}, (\mathsf{R}^\mathsf{B}_1(\alpha_i), \mathsf{U}_\mathsf{B}(\alpha_i)))$ to $\mathcal{F}_{\text{OLE}}$.
   - Alice receives back $\mathsf{s}_\mathsf{A}(\alpha_i) = \mathsf{p}_\mathsf{A}(\alpha_i) \cdot \mathsf{R}^\mathsf{B}_1(\alpha_i) + \mathsf{U}_\mathsf{B}(\alpha_i)$.
4. For each $i \in [\ell]$
   - Bob sends $(\texttt{inputR}, \mathsf{p}_\mathsf{B}(\alpha_i))$ to $\mathcal{F}_{\text{OLE}}$.
   - Alice sends $(\texttt{inputS}, (\mathsf{R}^\mathsf{A}_2(\alpha_i), \mathsf{U}_\mathsf{A}(\alpha_i)))$ to $\mathcal{F}_{\text{OLE}}$.
   - Alice receives back $\mathsf{s}_\mathsf{B}(\alpha_i) = \mathsf{p}_\mathsf{B}(\alpha_i) \cdot \mathsf{R}^\mathsf{A}_2(\alpha_i) + \mathsf{U}_\mathsf{A}(\alpha_i)$.
5. For each $i \in [\ell]$, Alice sends to Bob
$$\mathsf{s}'_\mathsf{A}(\alpha_i) = \mathsf{s}_\mathsf{A}(\alpha_i) + \mathsf{p}_\mathsf{A}(\alpha_i) \cdot \mathsf{R}^\mathsf{A}_1(\alpha_i) - \mathsf{U}_\mathsf{A}(\alpha_i)$$
6. For each $i \in [\ell]$, Bob sends to Alice
$$\mathsf{s}'_\mathsf{B}(\alpha_i) = \mathsf{s}_\mathsf{B}(\alpha_i) + \mathsf{p}_\mathsf{B}(\alpha_i) \cdot \mathsf{R}^\mathsf{B}_2(\alpha_i) - \mathsf{U}_\mathsf{B}(\alpha_i)$$
7. Alice outputs the evaluation points
$$\mathsf{p}(\alpha_i) = \mathsf{s}_\mathsf{A}(\alpha_i) + \mathsf{s}'_\mathsf{B}(\alpha_i) + \mathsf{p}_\mathsf{A}(\alpha_i) \cdot \mathsf{R}^\mathsf{A}_1(\alpha_i) - \mathsf{U}_\mathsf{A}(\alpha_i)$$
8. Bob outputs the evaluation points
$$\mathsf{p}(\alpha_i) = \mathsf{s}_\mathsf{B}(\alpha_i) + \mathsf{s}'_\mathsf{A}(\alpha_i) + \mathsf{p}_\mathsf{B}(\alpha_i) \cdot \mathsf{R}^\mathsf{B}_2(\alpha_i) - \mathsf{U}_\mathsf{B}(\alpha_i)$$

Fig. 10. Protocol for computing evaluation points of the noisy polynomial addition of $\mathsf{p}_\mathsf{A}(x)$ and $\mathsf{p}_\mathsf{B}(x)$ in the $\mathcal{F}_{\text{OLE}}$-hybrid model.

8 Conclusion and Open Problems

In this work we have initiated the study of sublinear threshold private set intersection. We have established a lower bound, showing that any protocol has to have a communication complexity of at least $\Omega(t)$, where t is the maximum allowed size of the symmetric set difference. We have shown an almost matching upper bound of $\tilde{\mathcal{O}}(t)$ based on fully homomorphic encryption and we have shown how to obtain a protocol with communication complexity $\tilde{\mathcal{O}}(t^2)$ based on additively homomorphic encryption. Our work poses several exciting open questions. From a theoretical perspective, it remains an open problem to construct a protocol with communication complexity $\tilde{\mathcal{O}}(t)$ from weaker assumptions than fully homomorphic encryption. Since our intersection protocol in Section 6 already has the desired complexity, one "only" needs to find a protocol for private set intersection cardinality testing with the same communication complexity. From a practical perspective, it is an open question to develop protocols that are practically, rather than just asymptotically, efficient.

References

ADI+17. Benny Applebaum, Ivan Damgård, Yuval Ishai, Michael Nielsen, and Lior Zichron. Secure arithmetic computation with constant computational overhead. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 223–254. Springer, Heidelberg, August 2017.

BFS86. László Babai, Peter Frankl, and Janos Simon. Complexity classes in communication complexity theory (preliminary version). In *27th FOCS*, pages 337–347. IEEE Computer Society Press, October 1986.

BGV12. Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In Shafi Goldwasser, editor, *ITCS 2012*, pages 309–325. ACM, January 2012.

BK89. Manuel Blum and Sampath Kannan. Designing programs that check their work. In *21st ACM STOC*, pages 86–97. ACM Press, May 1989.

BOT88. Michael Ben-Or and Prasoon Tiwari. A deterministic algorithm for sparse multivariate polynominal interpolation (extended abstract). In *20th ACM STOC*, pages 301–309. ACM Press, May 1988.

BV11a. Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In Rafail Ostrovsky, editor, *52nd FOCS*, pages 97–106. IEEE Computer Society Press, October 2011.

BV11b. Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 505–524. Springer, Heidelberg, August 2011.

BYJKS04. Ziv Bar-Yossef, Thathachar S Jayram, Ravi Kumar, and D Sivakumar. An information statistics approach to data stream and communication complexity. *Journal of Computer and System Sciences*, 68(4):702–732, 2004.

Can01. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145. IEEE Computer Society Press, October 2001.

CDN15. Ronald Cramer, Ivan Damgård, and Jesper Buus Nielsen. *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press, 2015.

CGT12. Emiliano De Cristofaro, Paolo Gasti, and Gene Tsudik. Fast and private computation of cardinality of set intersection and union. In Josef Pieprzyk, Ahmad-Reza Sadeghi, and Mark Manulis, editors, *CANS 12*, volume 7712 of *LNCS*, pages 218–231. Springer, Heidelberg, December 2012.

CLR17. Hao Chen, Kim Laine, and Peter Rindal. Fast private set intersection from homomorphic encryption. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 17*, pages 1243–1255. ACM Press, October / November 2017.

DCW13. Changyu Dong, Liqun Chen, and Zikai Wen. When private set intersection meets big data: an efficient and scalable protocol. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 13*, pages 789–800. ACM Press, November 2013.

DD15. Sumit Kumar Debnath and Ratna Dutta. Secure and efficient private set intersection cardinality using bloom filter. In Javier Lopez and Chris J. Mitchell, editors, *ISC 2015*, volume 9290 of *LNCS*, pages 209–226. Springer, Heidelberg, September 2015.

DJ01. Ivan Damgård and Mats Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In Kwangjo Kim, editor, *PKC 2001*, volume 1992 of *LNCS*, pages 119–136. Springer, Heidelberg, February 2001.

DT10. Emiliano De Cristofaro and Gene Tsudik. Practical private set intersection protocols with linear complexity. In Radu Sion, editor, *FC 2010*, volume 6052 of *LNCS*, pages 143–159. Springer, Heidelberg, January 2010.

EFG+15. Rolf Egert, Marc Fischlin, David Gens, Sven Jacob, Matthias Senker, and Jörn Tillmanns. Privately computing set-union and set-intersection cardinality via bloom filters. In Ernest Foo and Douglas Stebila, editors, *ACISP 15*, volume 9144 of *LNCS*, pages 413–430. Springer, Heidelberg, June / July 2015.

FNP04. Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient private matching and set intersection. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 1–19. Springer, Heidelberg, May 2004.

Gen09. Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 169–178. ACM Press, May / June 2009.

GJR10. Elena Grigorescu, Kyomin Jung, and Ronitt Rubinfeld. A local decision test for sparse polynomials. *Inf. Process. Lett.*, 110(20):898–901, 2010.

GN17. Satrajit Ghosh and Tobias Nilges. An algebraic approach to maliciously secure private set intersection. Cryptology ePrint Archive, Report 2017/1064, 2017. https://eprint.iacr.org/2017/1064.

GNN17. Satrajit Ghosh, Jesper Buus Nielsen, and Tobias Nilges. Maliciously secure oblivious linear function evaluation with constant overhead. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 629–659. Springer, Heidelberg, December 2017.

HOS17. Per A. Hallgren, Claudio Orlandi, and Andrei Sabelfeld. Privatepool: Privacy-preserving ridesharing. In *30th IEEE Computer Security Foundations Symposium, CSF 2017, Santa Barbara, CA, USA, August 21-25, 2017*, pages 276–291, 2017.

HV17. Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. Scalable multi-party private set-intersection. In Serge Fehr, editor, *PKC 2017, Part I*, volume 10174 of *LNCS*, pages 175–203. Springer, Heidelberg, March 2017.

HW06. Susan Hohenberger and Stephen A. Weis. Honest-verifier private disjointness testing without random oracles. In *Privacy Enhancing Technologies, 6th International Workshop, PET 2006, Cambridge, UK, June 28-30, 2006, Revised Selected Papers*, pages 277–294, 2006.

IPS09. Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Secure arithmetic computation with no honest majority. In Omer Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 294–314. Springer, Heidelberg, March 2009.

KKRT16. Vladimir Kolesnikov, Ranjit Kumaresan, Mike Rosulek, and Ni Trieu. Efficient batched oblivious PRF with applications to private set intersection. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 16*, pages 818–829. ACM Press, October 2016.

KLS+17. Ágnes Kiss, Jian Liu, Thomas Schneider, N Asokan, and Benny Pinkas. Private set intersection for unequal set sizes with mobile applications. *Proceedings on Privacy Enhancing Technologies*, 2017(4):177–197, 2017.

KMP+17. Vladimir Kolesnikov, Naor Matania, Benny Pinkas, Mike Rosulek, and Ni Trieu. Practical multi-party private set intersection from symmetric-key techniques. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 17*, pages 1257–1272. ACM Press, October / November 2017.

KMWF07. Eike Kiltz, Payman Mohassel, Enav Weinreb, and Matthew K. Franklin. Secure linear algebra using linearly recurrent sequences. In Salil P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 291–310. Springer, Heidelberg, February 2007.

KS92. Bala Kalyanasundaram and Georg Schintger. The probabilistic communication complexity of set intersection. *SIAM Journal on Discrete Mathematics*, 5(4):545–557, 1992.

KS05. Lea Kissner and Dawn Xiaodong Song. Privacy-preserving set operations. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 241–257. Springer, Heidelberg, August 2005.

Mar14. Moxie Marlinspike. The difficulty of private contact discovery. whispersystems.org/blog/contact-discovery., 2014.

Mea86. Catherine A. Meadows. A more efficient cryptographic matchmaking protocol for use in the absence of a continuously available third party. In *Proceedings of the 1986 IEEE Symposium on Security and Privacy, Oakland, California, USA, April 7-9, 1986*, pages 134–137, 1986.

MQU07. Jörn Müller-Quade and Dominique Unruh. Long-term security and universal composability. In Salil P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 41–60. Springer, Heidelberg, February 2007.

MTZ03. Yaron Minsky, Ari Trachtenberg, and Richard Zippel. Set reconciliation with nearly optimal communication complexity. *IEEE Transactions on Information Theory*, 49(9):2213–2218, 2003.

NMH+10. Shishir Nagaraja, Prateek Mittal, Chi-Yao Hong, Matthew Caesar, and Nikita Borisov. Botgrep: Finding P2P bots with structured graph analysis. In *19th USENIX Security Symposium, Washington, DC, USA, August 11-13, 2010, Proceedings*, pages 95–110, 2010.

NP99. Moni Naor and Benny Pinkas. Oblivious transfer and polynomial evaluation. In *31st ACM STOC*, pages 245–254. ACM Press, May 1999.

Pai99. Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 223–238. Springer, Heidelberg, May 1999.

PSSZ15. Benny Pinkas, Thomas Schneider, Gil Segev, and Michael Zohner. Phasing: Private set intersection using permutation-based hashing. In *24th USENIX Security Symposium, USENIX Security 15, Washington, D.C., USA, August 12-14, 2015.*, pages 515–530, 2015.

PSWW18. Benny Pinkas, Thomas Schneider, Christian Weinert, and Udi Wieder. Efficient circuit-based PSI via cuckoo hashing. In *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part III*, pages 125–157, 2018.

PSZ14. Benny Pinkas, Thomas Schneider, and Michael Zohner. Faster private set intersection based on OT extension. In *Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, August 20-22, 2014.*, pages 797–812, 2014.

RAD78. Ronald L Rivest, Len Adleman, and Michael L Dertouzos. On data banks and privacy homomorphisms. *Foundations of secure computation*, 4(11):169–180, 1978.

Raz90. Alexander A. Razborov. Applications of matrix methods to the theory of lower bounds in computational complexity. *Combinatorica*, 10(1):81–93, 1990.

RR17a. Peter Rindal and Mike Rosulek. Improved private set intersection against malicious adversaries. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 235–259. Springer, Heidelberg, April / May 2017.

RR17b. Peter Rindal and Mike Rosulek. Malicious-secure private set intersection via dual execution. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 17*, pages 1229–1242. ACM Press, October / November 2017.

Yao86. Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th FOCS*, pages 162–167. IEEE Computer Society Press, October 1986.

ZC18. Yongjun Zhao and Sherman S. M. Chow. Can you find the one for me? privacy-preserving matchmaking via threshold PSI. Cryptology ePrint Archive, Report 2018/184, 2018. https://eprint.iacr.org/2018/184.