Electronic Colloquium on Computational Complexity, Report No. 94 (2014)

2-Server PIR with sub-polynomial communication

Zeev Dvir* Sivakanth Gopi†

Abstract

A 2-server Private Information Retrieval (PIR) scheme allows a user to retrieve the ith bit of an n-bit database replicated among two servers (which do not communicate) while not revealing any information about i to either server. In this work we construct a 1-round 2-server PIR with total communication cost $n^{O(\sqrt{\log\log n/\log n})}$. This improves over the currently known 2-server protocols which require $O(n^{1/3})$ communication and matches the communication cost of known 3-server PIR schemes. Our improvement comes from reducing the number of servers in existing protocols, based on Matching Vector Codes, from 3 or 4 servers to 2. This is achieved by viewing these protocols in an algebraic way (using polynomial interpolation) and extending them using partial derivatives.

1 Introduction

Private Information Retrieval (PIR) was first introduced by Chor, Goldreich, Kuzhelevtiz and Sudan [CKGS98]. In a k-server PIR scheme, a user can retrieve the ith bit a_i of a n-bit database $\mathbf{a} = (a_1, \cdots, a_n) \in \{0,1\}^n$ replicated among k servers (which do not communicate) while giving no information about i to any server. The goal is to design PIR schemes that minimize the communication cost which is the worst case number of bits transferred between the user and the servers in the protocol. The trivial solution which works even with one server is to ask a server to send the entire database $\mathbf{a}$, which has communication cost n.

When $k = 1$ the trivial solution cannot be improved [CKGS98]. But when $k \geq 2$, the communication cost can be brought down significantly. In [CKGS98], a 2-server PIR scheme with communication cost $O(n^{1/3})$ and a k-server PIR scheme with cost $O\left(k^2 \log k n^{1/k}\right)$ were presented. The k-server PIR schemes were improved further in subsequent papers [Amb97, BI01, BIKR02]. In [BIKR02], a k-server PIR scheme with cost $n^{O\left(\frac{\log\log k}{k\log k}\right)}$ was obtained. This was the best for a long time until the breakthrough result of Yekhanin[Yek08] who gave the first 3-server scheme with sub-polynomial communication (assuming a number theoretic conjecture). Later, Efremenko[Efr09] gave an unconditional k-server PIR scheme with sub-polynomial cost for $k \geq 3$ which were slightly improved in [IS10] and [CFL+13]. These new PIR schemes follow from the constructions of constant query smooth Locally Decodable Codes

*Department of Computer Science and Department of Mathematics, Princeton University. Email: zeev.dvir@gmail.com. Research supported by NSF grants CCF-1217416 and CCF-0832797.

†Department of Computer Science, Princeton University. Email: sgopi@cs.princeton.edu.

ISSN 1433-8092

(LDCs) of sub-exponential length called Matching Vector Codes (MVCs)[DGY10]. A k-query LDC [KT00] is an error correcting code which allows the receiver of a corrupted encoding of a message to recover the ith bit of the message using only k (random) queries. In a *smooth* LDC, each query of the reconstruction algorithm is uniformly distributed among the code word symbols. Given a k-query smooth LDC, one can construct a k-server PIR scheme by letting each server simulate one of the queries. Despite the advances in 3-server PIR schemes, the 2-server PIR case is still stuck at $O(n^{1/3})$ since 2-query LDCs provably require exponential size encoding [KdW03] (which translates to $\Omega(n)$ communication cost in the corresponding PIR scheme). For more information on the relation between PIR and LDC and the constructions of sub-exponential LDCs and sub-polynomial cost PIR schemes with more than 2 servers we refer to the survey [Yek12].

On the lower bounds side, there is very little known. The best known lower bound for the communication cost of a 2-server PIR is $5 \log n$ [WdW05] whereas the trivial lower bound is $\log n$. In [CKGS98], a lower bound of $\Omega(n^{1/3})$ is conjectured. In [RY06], an $\Omega(n^{1/3})$ lower bound was proved for a restricted model of 2-server PIR called bilinear group based PIR. This model encompasses all the previously known constructions which achieve $O(n^{1/3})$ cost for 2-server PIR. We elaborate more on the relation between this model and our construction after we present our results below.

PIR is extensively studied and there are several variants of PIR in literature. The most important variant with cryptographic applications is called Computationally Private Information Retrieval (CPIR). In CPIR, the privacy guarantee is based on computational hardness of certain functions i.e. a computationally bounded server cannot gain any information about the user's query. In this case, non-trivial schemes exist even in the case of one server under some cryptographic hardness assumptions. For more information on these variants of PIR see [Gar, Gas04, Lip]. In this paper, we are only concerned with information theoretic privacy i.e. even a computationally unbounded server cannot gain any information about the user's query which is the strongest form of privacy.

1.1 Our Results

We start with a formal definition of a 2-server PIR scheme. A 2-server PIR scheme involves two servers $\mathcal{S}_1$ and $\mathcal{S}_2$ and a user $\mathcal{U}$. A database $\mathbf{a} = (a_1, \cdots, a_n) \in \{0,1\}^n$ is replicated between the servers $\mathcal{S}_1$ and $\mathcal{S}_2$. We assume that the servers cannot communicate with each other. The user $\mathcal{U}$ wants to retrieve the ith bit of the database a_i without revealing any information about i to either server. The following definition is from [CKGS98]:

Definition 1.1. *A 2-server PIR protocol is a triplet of algorithms $\mathcal{P} = (\mathcal{Q}, \mathcal{A}, \mathcal{R})$. At the beginning, the user $\mathcal{U}$ obtains a random string r. Next $\mathcal{U}$ invokes $\mathcal{Q}(i, r)$ to generate a pair of queries (q_1, q_2). $\mathcal{U}$ sends q_1 to $\mathcal{S}_1$ and q_2 to $\mathcal{S}_2$. Each server S_j responds with an answer $ans_j = \mathcal{A}(j, \mathbf{a}, q_j)$. Finally, $\mathcal{U}$ computes its output by applying the recovery algorithm $\mathcal{R}(ans_1, ans_2, i, r)$. The protocol should satisfy the following conditions:*

- ***Correctness :*** *For any n, $\mathbf{a} \in \{0,1\}^n$ and $i \in [n]$, the user the outputs the correct value of a_i with probability 1 (where the probability is over the random strings r) i.e.*

$\mathcal{R}(ans_1, ans_2, i, r) = a_i$

- ***Privacy :*** *Each server individually learns no information about i i.e. for any fixed database* $\mathbf{a}$ *and for $j = 1, 2$, the distributions of $q_j(i_1, r)$ and $q_j(i_2, r)$ are identical for all $i_1, i_2 \in [n]$ when r is randomly chosen.*

The communication cost of the protocol is the total number of bits exchanged between the user and the servers in the worst case.

k-server PIR is similarly defined, with the database replicated among k servers which cannot communicate between themselves. We only defined 1-round PIR i.e. there is only one round of interaction between the user and the servers. All known constructions of PIR schemes are 1-round and it is an interesting open problem to find if interaction helps. We now state our main theorem:

Theorem 1. *There exists a 2-server PIR scheme with communication cost* $n^{O\left(\sqrt{\frac{\log\log n}{\log n}}\right)}$.

The definition of a 2-server PIR scheme can be generalized in an obvious manner to any number of servers. In [Efr09] a 2^r-server PIR schemes was given with $n^{O\left((\log\log n/\log n)^{1-1/r}\right)}$ communication cost for any $r \geq 2$. Using our techniques, we can reduce the number of servers in this scheme by a factor of two. That is, we prove the following stronger form of Theorem 1.

Theorem 2. *For any $r \geq 2$, there exists a 2^{r-1}-server PIR scheme with communication cost* $n^{O\left((\log\log n/\log n)^{1-1/r}\right)}$.

We note that the proof of Theorem 2 actually allows the database symbols to be in the larger alphabet $\mathbb{Z}_m$, where m is the composite over which we construct the MV family.

There was some work on decreasing the 2^r query complexity of the construction of Matching Vector Codes in [Efr09]. A query complexity of $9 \cdot 2^{r-4}$ for $r \geq 6$ was achieved in [IS10] while keeping the encoding length the same. This was improved in [CFL+13] to $3^{\lceil r/2 \rceil}$ for $2 \leq r \leq 103$ and $(\frac{3}{4})^{51} \cdot 2^r$ for $r \geq 104$. Using these LDCs directly to get a PIR scheme will do better than our scheme when the number of servers is more than 26, whereas our scheme will do better than these when the number of servers are less than 9.

1.2 Related work

Polynomial lower bounds for bilinear group based PIR: In [RY06], an $\Omega(n^{1/3})$ lower bound was shown for a restricted model of 2-server PIR schemes. This lower bound holds for schemes that are both *bilinear* and *group based*. Our scheme can be made into a bilinear scheme* (see section 4.1) over the field $\mathbb{F}_3$ of three elements. However, it does not satisfy the property of being group based as defined in [RY06]. Our scheme does satisfy a weaker notion of *employing a group-based secret sharing scheme* (another technical term defined in [RY06]). The difference between these two notions (of being group based as opposed to employing a

*Our scheme can infact be made linear and using a simple transformation given in [RY06], any linear scheme can be converted to a bilinear scheme

group based secret sharing scheme) is akin to the difference between LCCs and LDCs (LCCs being the stronger notion). In group based PIR, the database is represented by the values of a function over a subset of a group but the user should be able to recover the value of that function at *every* group element. Our scheme encodes the database as a function over a group and the user will only be able to recover the bits of the database from the function.

2-query LDCs over large alphabet: The reader familiar with the exponential lower bounds for 2-query LDCs [KdW03] would wonder why our construction does not violate these bounds. The reason is that, when one translates 2-server PIR schemes into LDC, the resulting alphabet of the code can be quite large. Formally, a scheme with communication cost s will translate into an LDC $C : \{0,1\}^n \mapsto (\{0,1\}^s)^{2^s}$ (with the blocks corresponding to all possible answers by the servers). Thus, each one of the two queries used by the decoder is a string of s bits. The known lower bounds for such LDCs are exponential only as long as $s << \log(n)$ and so our construction does not violate them. Hence, our main theorem also gives the first construction of a sub-exponential 2-query LDC over an alphabet of size $2^{n^{o(1)}}$.

1.3 Proof Overview

On a very high level, the new protocol combines the existing 2-server scheme of [WY05], which uses polynomial interpolation using derivatives, with Matching Vector Codes (MV Codes) [Yek08, Efr09]. In particular, we make use of the view of MV codes as polynomial codes, developed in [DGY10]. This short overview is meant as a guide to the ideas in the construction and assumes some familiarity with [WY05] and [DGY10] (a detailed description will follow in the next sections). The 2-server scheme of [WY05] works by embedding the database $\mathbf{a} = (a_1, \ldots, a_n)$ as evaluations of a degree 3 polynomial $F(x_1, \ldots, x_k)$ over a small finite field $\mathbb{F}_q$ with $k \sim n^{1/3}$. To recover the value $a_i = F(P_i)$ the user passes a random line through the point $P_i \in \mathbb{F}_q^k$, picks two random points Q_1, Q_2 on that line and sends the point Q_i to the ith server. Each server responds with the value of F at Q_i and the values of all partial derivatives $\partial F/\partial x_j, j = 1, \ldots, k$ at that point. The restriction of F to the line is a univariate degree 3 polynomial and the user can recover the values of this polynomial at two points as well as the value of its derivative at these points. These four values (two evaluations plus two derivatives) are enough to recover the polynomial and so its value at P_i. The key is that each server's response only depends on the point Q_i (which is completely random). The user can compute the derivatives of the restricted polynomial from these values (knowing the line equation).

To see how MV codes come into the picture we have to describe them in some detail. An MV family is a pair of lists $\mathcal{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_n)$, $\mathcal{V} = (\mathbf{v}_1, \ldots, \mathbf{v}_n)$ with each list element $\mathbf{u}_i$ and $\mathbf{v}_j$ belonging to $\mathbb{Z}_m^k$ and m is a small integer. These lists must satisfy the condition that $\langle \mathbf{u}_i, \mathbf{v}_j \rangle$ (taken mod m) is zero iff $i = j$. When m is a composite, one can construct such families of vectors of size $n = k^{\omega(1)}$ [Gro99] (this is impossible if m is prime). From such a family we can construct an m-query LDC as follows: given a message $\mathbf{a} = (a_1, \ldots, a_n) \in \{0,1\}^n$ define the polynomial $F(x_1, \ldots, x_k) = \sum_{i=1}^n a_i \mathbf{x}^{\mathbf{u}_i}$ (we denote $\mathbf{x}^{\mathbf{c}} = x_1^{c_1} \ldots x_k^{c_k}$). We think of F as a polynomial with coefficients in some finite field $\mathbb{F}_q$ containing an element $\gamma \in \mathbb{F}_q$ of order m. The final encoding of $\mathbf{a}$ is the evaluations of F over all points in $\mathbb{F}_q^k$ of the form

$\gamma^{\mathbf{c}} = (\gamma^{c_1}, \ldots, \gamma^{c_k})$ for all $\mathbf{c} \in \mathbb{Z}_m^k$. To recover a_i in a 'smooth' way, we pick a random $\mathbf{z} \in \mathbb{Z}_m^k$ and consider the restriction of F to the 'multiplicative line' given by $L = \{\gamma^{\mathbf{z}+t\mathbf{v}_i} \,|\, t \in \mathbb{Z}_m\}$. That is, we denote $G(t) = F(\gamma^{\mathbf{z}+t\mathbf{v}_i})$. In [DGY10] it was observed that this restriction can be seen as a polynomial $g(T)$ of degree at most $m-1$ in the new 'variable' $T = \gamma^t$ and so can be reconstructed from the m values on the line $g(\gamma^t) = G(t), t = 0, 1, \ldots, m-1$. The final observation is that $g(0)$ is a nonzero multiple of a_i (since the only contribution to the free coefficient comes from the monomial $a_i\mathbf{x}^{\mathbf{u}_i}$) and so we can recover it if we know $g(T)$.

Our new protocol combines these two constructions by using the MV code construction and then asking each server for the evaluations of F at a point, as well as the values of a certain differential operator (similar to first order derivatives) at these points. For this to work we need two ingredients. The first is to replace the field $\mathbb{F}_q$ with a certain ring which has characteristic m and an element of order m (we only use $m = 6$ and can take the polynomial ring $\mathbb{Z}_m[\gamma]/(\gamma^6 - 1)$). The second is an observation that, in known MV families constructions [Gro99], the inner products $\langle \mathbf{u}_i, \mathbf{v}_j \rangle$ that are nonzero (that is, when $i \neq j$) can be made to fall in a small set. More precisely, over $\mathbb{Z}_6$, the inner products are either zero or in the set $\{1, 3, 4\}$. This means that the restricted polynomial only has nonzero coefficients corresponding to powers of T coming from the set $\{0, 1, 3, 4\}$. Such a polynomial has four degrees of freedom and can be recovered from two evaluations and two derivatives (of order one). We are also able to work with arbitrary MV families by using second order derivatives at two points (which are sufficient to recover a degree 5 polynomial).

1.4 Organization

In section 2 we give some preliminary definitions and notations. In section 3, we review the construction of a 2-server PIR scheme with $O(n^{1/3})$ communication cost which is based on polynomial interpolation with partial derivatives [WY05]. In section 4, we present our new construction of sub-polynomial 2-server PIR schemes and some of its variants. Then, in Section 5 we analyze the generalization to more servers. We conclude in Section 6 with some remarks on future directions.

2 Preliminaries

Notations: We will use bold letters like $\mathbf{u}, \mathbf{v}, \mathbf{z}$ etc. to denote vectors. The inner product between two vectors $\mathbf{u} = (u_1, \cdots, u_k), \mathbf{v} = (v_1, \cdots, v_k)$ is denoted by $\langle \mathbf{u}, \mathbf{v} \rangle = \sum_{i=1}^k u_i v_i$. For a commutative ring $\mathcal{R}$ we will denote by $\mathcal{R}[x_1, \cdots, x_k]$ the ring of polynomials in formal variables $x_1, \ldots, x_k$ with coefficients in $\mathcal{R}$. We will use the notation $\mathbf{x}^{\mathbf{z}}$ with $\mathbf{x} = (x_1, \cdots, x_k)$, $\mathbf{z} = (z_1, \cdots, z_k) \in \mathbb{Z}^k$ to denote the monomial $\prod_{i=1}^k x_i^{z_i}$. So any polynomial $F(\mathbf{x}) \in \mathcal{R}[x_1, \cdots, x_k]$ can be written as $F(\mathbf{x}) = \sum_{\mathbf{z}} c_{\mathbf{z}} \mathbf{x}^{\mathbf{z}}$.

$\mathbb{Z}_m = \mathbb{Z}/m\mathbb{Z}$ is the ring of integers modulo m. When $\mathbf{u} \in \mathbb{Z}_m^k$, $\mathbf{x}^{\mathbf{u}}$ denotes $\mathbf{x}^{\tilde{\mathbf{u}}}$ where $\tilde{\mathbf{u}} \in \{0, 1, \cdots, m-1\}^k$ is the unique vector such that $\mathbf{u} \equiv \tilde{\mathbf{u}} \mod m$. $\mathbb{F}_q$ denotes the finite field of size q.

2.1 The rings $\mathcal{R}_{m,r}$

For our construction it will be convenient (although not absolutely necessary, see Section 4.1) to work over a ring which has characteristic 6 and contains an element of order 6. We now discuss how to construct such a ring in general.

Let $m > 1$ be an integer and let γ be a formal variable. We denote by

$$\mathcal{R}_{m,r} = \mathbb{Z}_m[\gamma]/(\gamma^r - 1)$$

the ring of univariate polynomials $\mathbb{Z}_m[\gamma]$ in γ with coefficients in $\mathbb{Z}_m$ modulo the identity $\gamma^r = 1$.[†] More formally, each element $f \in \mathcal{R}_{m,r}$ is represented by a degree $\leq r-1$ polynomial $f(\gamma) = \sum_{\ell=0}^{r-1} c_\ell \gamma^\ell$ with coefficients $c_i \in \mathbb{Z}_m$. Addition is done as in $\mathbb{Z}_m[\gamma]$ (coordinate wise modulo m) and multiplication is done over $\mathbb{Z}_m[\gamma]$ but using the identity $\gamma^r = 1$ to reduce higher order monomials to degree $\leq r-1$. It is easy to see that this reduction is uniquely defined: to obtain the coefficient of γ^ℓ we sum all the coefficients of powers of γ that are of the form $\ell + km$ for some integer $k \geq 0$. This implies the following lemma.

Lemma 2.1. *Let $f = \sum_{\ell=0}^{r-1} c_\ell \gamma^\ell$ be an element in $\mathcal{R}_{m,r}$. Then, $f = 0$ in the ring $\mathcal{R}_{m,r}$ iff $c_i = 0$ (in $\mathbb{Z}_m$) for all $0 \leq i \leq r-1$.*

Remark 2.2. *For any $t \in \{0, 1, \cdots, r-1\}$, γ^t is not a zero divisor of the ring $\mathcal{R}_{m,r}$. This holds since the coefficients of $\gamma^t \cdot f(\gamma)$ are the same as those of $f(\gamma)$ (shifted cyclicly t positions).*

2.2 Matrices over Commutative Rings

Let $\mathcal{R}$ be a commutative ring (with unity). Let $M \in \mathcal{R}^{n \times n}$ be an $n \times n$ matrix with entries from $\mathcal{R}$. Most of the classical theory of determinants can be derived in this setting in exactly the same way as over fields. One particularly useful piece of this theory is the Adjugate (or Classical Adjoint) matrix. For an $n \times n$ matrix $M \in \mathcal{R}^{n \times n}$ the Adjugate matrix is denoted by $\mathrm{adj}(M) \in \mathcal{R}^{n \times n}$ and has the (j,i) cofactor of A as its (i,j)th entry (recall that the cofactor is the determinant of the matrix obtained from M after removing the ith row and jth column multiplied by $(-1)^{i+j}$). A basic fact in matrix theory is the following identity.

Lemma 2.3 (Theorem 1.7 from [McD84])**.** *Let $M \in \mathcal{R}^{n \times n}$ with $\mathcal{R}$ a commutative ring with identity. Then $M \cdot \mathrm{adj}(M) = \mathrm{adj}(M) \cdot M = \det(M) \cdot I_n$ where I_n is the $n \times n$ identity matrix.*

The way we will use this fact is as follows:

Remark 2.4. *Suppose $M \in \mathcal{R}^{n \times n}$ has non-zero determinant and let $\mathbf{a} = (a_1, \ldots, a_n)^t \in \mathcal{R}^n$ be some column vector where $a_1 = 0$ or $a_1 = c$, where c is not a zero-divisor. Then we can determine the value of a_1 (i.e., tell whether its 0 or c) from the product $M \cdot \mathbf{a}$. The way to do it is to multiply $M \cdot \mathbf{a}$ from the left by $\mathrm{adj}(M)$ and to look at the first entry. This will give us $\det(M) \cdot a_1$ which is zero iff a_1 is (since $\det(M) \cdot c$ is always nonzero).*

[†]The rings $\mathcal{R}_{m,r}$ are sometimes denoted by $\mathbb{Z}_m[C_r]$ and referred to as the *group ring* of the cyclic group C_r with coefficients in $\mathbb{Z}_m$. See e.g., [KS13, HH11] for some recent applications of these rings in cryptography.

2.3 Matching Vector Families

Definition 2.5 (Matching Vector Family)**.** *Let $S \subset \mathbb{Z}_m \setminus \{0\}$ and let $\mathcal{F} = (\mathcal{U}, \mathcal{V})$ where $\mathcal{U} = (\mathbf{u}_1, \cdots, \mathbf{u}_n), \mathcal{V} = (\mathbf{v}_1, \cdots, \mathbf{v}_n)$ and $\forall i\ \mathbf{u}_i, \mathbf{v}_i \in \mathbb{Z}_m^k$. Then $\mathcal{F}$ is called an S-matching vector family of size n and dimension k if $\forall\ i, j$,*

$$\langle \mathbf{u}_i, \mathbf{v}_j \rangle \begin{cases} = 0 & \text{if } i = j \\ \in S & \text{if } i \neq j \end{cases}$$

If S is omitted, it implies that $S = \mathbb{Z}_m \setminus \{0\}$.

Theorem 2.6 (Theorem 1.4 in [Gro99])**.** *Let $m = p_1 p_2 \cdots p_r$ where $p_1, p_2 \cdots, p_r$ are distinct primes with $r \geq 2$, then there exists an explicitly constructible S-matching vector family $\mathcal{F}$ in $\mathbb{Z}_m^k$ of size $n \geq \exp\left(\Omega\left(\frac{(\log k)^r}{(\log\log k)^{r-1}}\right)\right)$ where $S = \{a \in \mathbb{Z}_m : a \mod p_i \in \{0,1\}\ \forall\ i \in [r]\} \setminus \{0\}$.*

Remark 2.7. *The size of S in the above theorem is $2^r - 1$ by the Chinese Remainder Theorem. Thus, there are matching vector families of size super-polynomial in the dimension of the space with inner products restricted to a set of size $2^r = |S \cup \{0\}|$.*

In the special case when $p_1 = 2, p_2 = 3$, we have $m = 6$ and the following corollary:

Corollary 2.8. *There is an explicitly constructible S-matching vector family $\mathcal{F}$ in $\mathbb{Z}_6^k$ of size $n \geq \exp\left(\Omega\left(\frac{(\log k)^2}{\log\log k}\right)\right)$ where $S = \{1, 3, 4\} \subset \mathbb{Z}_6$*

2.4 A number theoretic lemma

We will need the following simple lemma. Recall that the *order* of an element a in a finite multiplicative group G is the smallest integer $w \geq 1$ so that $a^w = 1$.

Lemma 2.9. *Let $\mathbb{F}_p$ be a field of prime order p and let $k \geq 1$ be an integer co-prime to p. Then, the algebraic closure of $\mathbb{F}_p$ contains an element ζ of order k.*

Proof. Since k, p are co-prime, $p \in \mathbb{Z}_k^*$ which is the multiplicative group of invertible elements in $\mathbb{Z}_k$. Let $w \geq 1$ be the order of p in the group $\mathbb{Z}_k^*$, so k divides $p^w - 1$. Consider the extension field $\mathbb{F}_{p^w}$, which is a sub field of the algebraic closure of $\mathbb{F}_p$. The multiplicative group $\mathbb{F}_{p^w}^*$ of this field is a cyclic group of size $p^w - 1$. Since k divides this size, there must be an element in $\mathbb{F}_{p^w}$ of order k. □

3 Review of $O(n^{1/3})$ cost 2-server PIR

There are several known constructions of 2-server PIR with $O(n^{1/3})$ communication cost. We will recall here in detail a particular construction due to [WY05] which uses polynomial interpolation using derivatives (over a field). In the next section we will replace the field with a ring and see how to use matching vector families to reduce the communication cost.

Let $\mathbf{a} = (a_1, \cdots, a_n)$ be the database, choose k to be smallest integer such that $n \leq \binom{k}{3}$. Let $\mathbb{F}_q$ be a finite field with $q > 3$ elements and suppose for simplicity that q is prime (so that partial derivatives behave nicely for polynomials of degree at most 3). Let $\phi : [n] \mapsto \{0,1\}^k \subset \mathbb{F}_q^k$ be an embedding of the n coordinates into points in $\{0,1\}^k$ of Hamming weight 3. Such an embedding exists since $n \leq \binom{k}{3}$.

Define $F(x_1, \cdots, x_k) = F(\mathbf{x}) \in \mathbb{F}_q[x_1, \cdots, x_k]$ as

$$F(\mathbf{x}) = \sum_{i=1}^{n} a_i \left(\prod_{j:\phi(i)_j=1} x_j \right)$$

Note that $F(\mathbf{x})$ is a degree 3 polynomial satisfying $F(\phi(i)) = a_i \; \forall \; i \in [n]$. Fix any two nonzero field elements $t_1 \neq t_2 \in \mathbb{F}_q \setminus \{0\}$.

Suppose the user $\mathcal{U}$ wants to recover the bit a_τ. The protocol is as follows: The user picks a uniformly random element $\mathbf{z} \in \mathbb{F}_q^k$ and sends $\phi(\tau) + t_1\mathbf{z}$ to $\mathcal{S}_1$ and $\phi(\tau) + t_2\mathbf{z}$ to $\mathcal{S}_2$. Each server S_i then replies with the value of F at the point received $F(\phi(\tau) + t_i\mathbf{z})$ as well as the values of the k partial derivatives of F at the same point

$$\nabla F(\phi(\tau) + t_i\mathbf{z}) = \left(\frac{\partial F}{\partial z_1}(\phi(\tau) + t_i\mathbf{z}), \cdots, \frac{\partial F}{\partial z_k}(\phi(\tau) + t_i\mathbf{z}) \right)$$

The partial derivatives here are defined in the same way as for polynomials over the real numbers.

$\mathcal{U}$: Picks a uniformly random $\mathbf{z} \in \mathbb{F}_q^k$
$\mathcal{U} \to \mathcal{S}_i : \phi(\tau) + t_i\mathbf{z}$
$\mathcal{S}_i \to \mathcal{U} : F(\phi(\tau) + t_i\mathbf{z}), \nabla F(\phi(\tau) + t_i\mathbf{z})$

The protocol is private since $\phi(\tau) + t\mathbf{z}$ is uniformly distributed in $\mathbb{F}_q^k$ for any τ and $t \neq 0$.

Consider the univariate polynomial

$$g(t) = F(\phi(\tau) + t\mathbf{z}).$$

Observe that, be the chain rule,

$$g'(t) = \langle \nabla F(\phi(\tau) + t\mathbf{z}), \mathbf{z} \rangle.$$

Thus the user can recover the values $g(t), g'(t)$ for $t = t_1, t_2$ from the server's responses. From this information the user needs to find $g(0) = F(\phi(\tau)) = a_\tau$. Since F is a degree 3 polynomial, $g(t)$ is a univariate degree 3 polynomial, let $g(t) = \sum_{\ell=0}^{3} c_\ell t^\ell$. Therefore we have the following matrix equation:

$$\begin{bmatrix} g(t_1) \\ g'(t_1) \\ g(t_2) \\ g'(t_2) \end{bmatrix} = \begin{bmatrix} 1 & t_1 & t_1^2 & t_1^3 \\ 0 & 1 & 2t_1 & 3t_1^2 \\ 1 & t_2 & t_2^2 & t_2^3 \\ 0 & 1 & 2t_2 & 3t_2^2 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = M \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

The matrix M has determinant $det(M) = (t_2 - t_1)^4$ and so M is invertible as long as $t_1 \neq t_2$. Thus the user can find $c_0 = g(0) = F(\phi(\tau)) = a_\tau$ by multiplying by the inverse of M.

The communication cost of this protocol is $O(k) = O(n^{1/3})$ since the user sends a vector in $\mathbb{F}_q^k$ to each server and each server sends an element in $\mathbb{F}_q$ and a vector in $\mathbb{F}_q^k$ to the user.

4 The new 2-server scheme

In this section we describe our main construction which proves Theorem 1. Before describing the construction we set up some of the required ingredients and notations.

The first ingredient is a matching vector family over $\mathbb{Z}_6$ as in Corollary 2.8. That is, we construct an $S = \{1, 3, 4\}$- matching vector family $\mathcal{F} = (\mathcal{U}, \mathcal{V})$ where $\mathcal{U} = (\mathbf{u}_1, \cdots, \mathbf{u}_n), \mathcal{V} = (\mathbf{v}_1, \cdots, \mathbf{v}_n)$ have elements in $\mathbb{Z}_6^k$. Corollary 2.8 tells us that this can be done with $n = \exp(\Omega(\log^2 k / \log\log k))$ or $k = \exp(O\left(\sqrt{\log n \ \log\log n}\right))$.

We will work with polynomials over the ring

$$\mathcal{R} = \mathcal{R}_{6,6} = \mathbb{Z}_6[\gamma]/(\gamma^6 - 1)$$

(see Section 2). We will denote the vector $(\gamma^{z_1}, \gamma^{z_2}, \cdots, \gamma^{z_k})$ by $\gamma^{\mathbf{z}}$ where $\mathbf{z} = (z_1, \cdots, z_k) \in \mathbb{Z}_6^k$. We will need to extend the notion of partial derivatives to polynomials in $\mathcal{R}[x_1, \ldots, x_k]$. This will be a non standard definition, but it will satisfy all the properties we will need. Instead of defining each partial derivative separately, we define one operator that will include all of them.

Definition 4.1. *Let $\mathcal{R}$ be a commutative ring and let $F(\mathbf{x}) = \sum c_{\mathbf{z}} \mathbf{x}^{\mathbf{z}} \in \mathcal{R}[x_1, \ldots, x_k]$. We define $F^{(1)} \in (\mathcal{R}^k)[x_1, \ldots, x_k]$ to be*

$$F^{(1)}(\mathbf{x}) := \sum (c_{\mathbf{z}} \cdot \mathbf{z}) \mathbf{x}^{\mathbf{z}}$$

For example, when $F(x_1, x_2) = x_1^2 x_2 + 4x_1 x_2 + 3x_2^2$ (with integer coefficients),

$$F^{(1)}(x_1, x_2) = \begin{bmatrix} 2 \\ 1 \end{bmatrix} x_1^2 x_2 + \begin{bmatrix} 4 \\ 4 \end{bmatrix} x_1 x_2 + \begin{bmatrix} 0 \\ 6 \end{bmatrix} x_2^2$$

One can think of $F^{(1)}$ both as a polynomial with coefficients in $\mathcal{R}^k$ as well as a k-tuple of polynomials in $\mathcal{R}[x_1, \ldots, x_k]$. This will not matter much since the only operation we will perform on $F^{(1)}$ is to evaluate it at a point in $\mathcal{R}^k$.

The Protocol: Let $\mathbf{a} = (a_1, a_2 \cdots, a_n) \in \{0, 1\}^n$ be an n-bit database shared by two servers $\mathcal{S}_1$ and $\mathcal{S}_2$. The user $\mathcal{U}$ wants to find the bit a_τ without revealing any information about τ to either server. For the rest of this section, $\mathcal{R} = \mathcal{R}_{6,6} = \mathbb{Z}_6[\gamma]/(\gamma^6 - 1)$. The servers represent the database as a polynomial $F(\mathbf{x}) \in \mathcal{R}[\mathbf{x}] = \mathcal{R}[x_1, \cdots, x_k]$ given by

$$F(\mathbf{x}) = F(x_1, \cdots, x_k) = \sum_{i=1}^{n} a_i \mathbf{x}^{\mathbf{u}_i},$$

where $\mathcal{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_n)$ are given by the matching vector family $\mathcal{F} = (\mathcal{U}, \mathcal{V})$.

The user samples a uniformly random $\mathbf{z} \in \mathbb{Z}_6^k$ and then sends $\mathbf{z} + t_1\mathbf{v}_\tau$ to $\mathcal{S}_1$ and $\mathbf{z} + t_2\mathbf{v}_\tau$ to $\mathcal{S}_2$ where we fix $t_1 = 0$ and $t_2 = 1$ (other choices of values would also work). $\mathcal{S}_i$ then responds with the value of F at the point $\gamma^{\mathbf{z}+t_i\mathbf{v}_\tau}$, that is with $F(\gamma^{\mathbf{z}+t_i\mathbf{v}_\tau})$ and the value of the 'first order derivative' at the same point $F^{(1)}(\gamma^{\mathbf{z}+t_i\mathbf{v}_\tau})$. Notice that the protocol is private since $\mathbf{z} + t\mathbf{v}_\tau$ is uniformly distributed over $\mathbb{Z}_6^k$ for any fixed τ and t.

$$\begin{array}{|r|}\hline \mathcal{U} : \text{Picks a uniformly random } \mathbf{z} \in \mathbb{Z}_6^k \\ \mathcal{U} \to \mathcal{S}_i : \mathbf{z} + t_i\mathbf{v}_\tau \\ \mathcal{S}_i \to \mathcal{U} : F(\gamma^{\mathbf{z}+t_i\mathbf{v}_\tau}), F^{(1)}(\gamma^{\mathbf{z}+t_i\mathbf{v}_\tau}) \\ \hline \end{array}$$

Recovery: Define

$$G(t) := F(\gamma^{\mathbf{z}+t\mathbf{v}_\tau}) = \sum_{i=1}^{n} a_i \gamma^{\langle \mathbf{z}, \mathbf{u}_i \rangle + t\langle \mathbf{v}_\tau, \mathbf{u}_i \rangle}$$

Using the fact that $\gamma^6 = 1$, we can rewrite $G(t)$ as:

$$G(t) = \sum_{\ell=0}^{5} c_\ell \cdot \gamma^{t\ell},$$

with each $c_\ell \in \mathcal{R}$ given by

$$c_\ell = \sum_{i:\langle \mathbf{u}_i, \mathbf{v}_\tau \rangle = \ell \mod 6} a_i \gamma^{\langle \mathbf{z}, \mathbf{u}_i \rangle}.$$

Since

$$\langle \mathbf{u}_i, \mathbf{v}_\tau \rangle \mod 6 \begin{cases} = 0 & \text{if } i = \tau \\ \in S = \{1, 3, 4\} & \text{if } i \neq \tau \end{cases}$$

we can conclude that $c_0 = a_\tau \gamma^{\langle \mathbf{u}_\tau, \mathbf{z} \rangle}$ and $c_2 = c_5 = 0$. Therefore

$$G(t) = c_0 + c_1\gamma^t + c_3\gamma^{3t} + c_4\gamma^{4t}.$$

Next, consider the polynomial

$$g(T) = c_0 + c_1 T + c_3 T^3 + c_4 T^4 \in \mathcal{R}[T].$$

By definition we have

$$g(\gamma^t) = G(t) = F(\gamma^{\mathbf{z}+t\mathbf{v}_\tau})$$

$$g^{(1)}(\gamma^t) = \sum_{\ell=0}^{5} \ell c_\ell \gamma^{t\ell} = \langle F^{(1)}(\gamma^{\mathbf{z}+t\mathbf{v}_\tau}), \mathbf{v}_\tau \rangle,$$

where the last equality holds since $c_2 = c_5 = 0$ and

$$\langle F^{(1)}(\gamma^{\mathbf{z}+t\mathbf{v}_\tau}), \mathbf{v}_\tau \rangle = \left\langle \sum_{i=1}^{n} a_i \mathbf{u}_i \gamma^{\langle \mathbf{z}, \mathbf{u}_i \rangle + t\langle \mathbf{v}_\tau, \mathbf{u}_i \rangle}, \mathbf{v}_\tau \right\rangle$$

$$= \sum_{i=1}^{n} a_i \langle \mathbf{u}_i, \mathbf{v}_\tau \rangle \gamma^{\langle \mathbf{z}, \mathbf{u}_i \rangle + t\langle \mathbf{v}_\tau, \mathbf{u}_i \rangle}$$

$$= \sum_{\ell=0}^{5} \ell \left(\sum_{i: \langle \mathbf{u}_i, \mathbf{v}_\tau \rangle = \ell \mod 6} a_i \gamma^{\langle \mathbf{z}, \mathbf{u}_i \rangle} \right) \gamma^{t\ell} = \sum_{\ell=0}^{5} \ell c_\ell \gamma^{t\ell}$$

So the user can find the values of $g(\gamma^t), g^{(1)}(\gamma^t)$ for $t = t_1, t_2$. Since $t_1 = 0, t_2 = 1$, we obtain the following matrix equation:

$$\begin{bmatrix} g(1) \\ g^{(1)}(1) \\ g(\gamma) \\ g^{(1)}(\gamma) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 3 & 4 \\ 1 & \gamma & \gamma^3 & \gamma^4 \\ 0 & \gamma & 3\gamma^3 & 4\gamma^4 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_3 \\ c_4 \end{bmatrix} = M \begin{bmatrix} c_0 \\ c_1 \\ c_3 \\ c_4 \end{bmatrix}$$

The determinant (over $\mathcal{R}$) of the matrix M is

$$\det(M) = \gamma(\gamma - 1)^4(\gamma^2 + 4\gamma + 1) = 3\gamma^5 + 4\gamma^4 + 3\gamma^3 + 2\gamma \tag{1}$$

and so, by Lemma 2.1, is a non-zero element of the ring $\mathcal{R}$. Since $c_0 = a_\tau \gamma^{\langle \mathbf{u}_\tau, \mathbf{z} \rangle}$, either $c_0 = 0$ or $c_0 = \gamma^{\langle \mathbf{u}_\tau, \mathbf{z} \rangle}$ which is not a zero-divisor by remark 2.2. Hence, by Remark 2.4, the user can find whether $c_0 = 0$ from the vector $[g(1), g^{(1)}(1), g(\gamma), g^{(1)}(\gamma)]^t$ by multiplying it from the left by $\mathrm{adj}(M)$. Since $c_0 = a_\tau \gamma^{\langle \mathbf{u}_\tau, \mathbf{z} \rangle}$, a_τ will be zero iff c_0 is and so the user can recover $a_\tau \in \{0, 1\}$.

Communication Cost: The user sends a vector in $\mathbb{Z}_6^k$ to each server. Each server sends a element of $\mathcal{R}$ and a vector in $\mathcal{R}^k$ to the user. Since elements of $\mathcal{R}$ have constant size description, the total communication cost is $O(k) = n^{o(1)}$.

4.1 Working over $\mathbb{Z}_6$ or $\mathbb{F}_3$

Using the ring $\mathcal{R}_{6,6} = \mathbb{Z}_6[\gamma]/(\gamma^6 - 1)$ in the above construction makes the presentation clearer but is not absolutely necessary. Observing the proof, we see that one can replace it with any ring $\mathcal{R}$ as long as there is a homomorphism from $\mathcal{R}_{6,6}$ to $\mathcal{R}$ such that the determinant of the matrix M (Eq. 1) doesn't vanish under this homomorphism.

For example, we can work over the ring $\mathbb{Z}_6$ and use the element -1 as a substitute for γ. Since $(-1)^6 = 1$ all of the calculations we did with γ carry through. In addition, the resulting determinant of M is non zero when setting $\gamma = -1$ and so we can complete the recovery process. More formally, define the homomorphism $\tau : \mathbb{Z}_6[\gamma]/(\gamma^6 - 1) \mapsto \mathbb{Z}_6$ by extending the identity homomorphsim on $\mathbb{Z}_6$ using $\tau(\gamma) = -1$. Observe that the determinant of the matrix M in Eq. (1) doesn't vanish under this homomorphism, $\tau(\det(M)) = -4 = 2$.

A more interesting example is the ring of integers modulo 3, which we denote by $\mathbb{F}_3$ to highlight that it is also a field. We can use the homomorphsim $\phi : \mathbb{Z}_6[\gamma]/(\gamma^6 - 1) \mapsto \mathbb{F}_3$ by extending the natural homomorphsim from $\mathbb{Z}_6$ to $\mathbb{F}_3$ (given by reducing each element modulo 3) using $\phi(\gamma) = -1$. Again the determinant in Eq. (1) doesn't vanish. This also shows that

our scheme can be made to be *bilinear*, as defined in [RY06], since the answers of each server become linear combinations of database entries over a field.

4.2 An Alternative Construction

In the construction above we used the special properties of Grolmusz's construction, namely that the non-zero inner products are in the special set $S = \{1, 3, 4\}$. Here we show how to make the construction work with any matching vector family (over $\mathbb{Z}_6$). This construction also introduces higher order differential operators, which could be of use if one is to generalize this work further.

Suppose we run our protocol (with $\mathcal{R} = \mathcal{R}_{6,6}$) using a matching vector family with $S = \mathbb{Z}_6 \setminus \{0\}$. Then, we cannot claim that $c_2 = c_5 = 0$, but we still have $c_0 = a_\tau \gamma^{\langle \mathbf{u}_\tau, \mathbf{z} \rangle}$. We can proceed by asking for the 'second order' derivative of $F(\mathbf{x}) = \sum_{i=0}^{n} a_i \mathbf{x}^{\mathbf{u}_i}$ which we define as

$$F^{(2)}(\mathbf{x}) := \sum c_{\mathbf{z}} \ (\mathbf{z} \otimes \mathbf{z}) \ \mathbf{x}^{\mathbf{z}}$$

where $\mathbf{z} \otimes \mathbf{z}$ is the $k \times k$ matrix defined by $(\mathbf{z} \otimes \mathbf{z})_{ij} = z_i z_j$. For example, when $P(x_1, x_2) = x_1^2 x_2 + 4x_1 x_2 + 3x_2^2$,

$$P^{(2)}(x_1, x_2) = \begin{bmatrix} 4 & 2 \\ 2 & 1 \end{bmatrix} x_1^2 x_2 + 4 \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} x_1 x_2 + 3 \begin{bmatrix} 0 & 0 \\ 0 & 4 \end{bmatrix} x_2^2.$$

The final protocol is:

$\mathcal{U}$: Picks a uniformly random $\mathbf{z} \in \mathbb{Z}_m^k$
$\mathcal{U} \to \mathcal{S}_i : \mathbf{z} + t_i \mathbf{v}_\tau$
$\mathcal{S}_i \to \mathcal{U} : F(\gamma^{\mathbf{z}+t_i\mathbf{v}_\tau}), F^{(1)}(\gamma^{\mathbf{z}+t_i\mathbf{v}_\tau}), F^{(2)}(\gamma^{\mathbf{z}+t_i\mathbf{v}_\tau})$

Notice that privacy is maintained and the communication is $O(k^2) = n^{o(1)}$ as before.

For recovery, define $g(T) \in \mathcal{R}[T]$ as before and notice that, in addition to the identities

$$g(\gamma^t) = \sum_{\ell=0}^{5} c_\ell \gamma^{t\ell} = F(\gamma^{\mathbf{z}+t\mathbf{v}_\tau})$$
$$g^{(1)}(\gamma^t) = \sum_{\ell=0}^{5} \ell c_\ell \gamma^{t\ell} = \langle F^{(1)}(\gamma^{\mathbf{z}+t\mathbf{v}_\tau}), \mathbf{v}_\tau \rangle,$$

we also get the second order derivative of g from

$$g^{(2)}(\gamma^t) = \sum_{\ell=0}^{5} \ell^2 c_\ell \gamma^{t\ell} = \langle F^{(2)}(\gamma^{\mathbf{z}+t\mathbf{v}_\tau}), \mathbf{v}_\tau \otimes \mathbf{v}_\tau \rangle,$$

where the inner product of matrices is taken entry-wise and using the identity $\langle \mathbf{u} \otimes \mathbf{u}, \mathbf{v} \otimes \mathbf{v} \rangle = \langle \mathbf{u}, \mathbf{v} \rangle^2$.

By choosing $t_1 = 0, t_2 = 1$, we have the following matrix equation:

$$\begin{bmatrix} g(1) \\ g^{(1)}(1) \\ g^{(2)}(1) \\ g(\gamma) \\ g^{(1)}(\gamma) \\ g^{(2)}(\gamma) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 & 4 & 5 \\ 0 & 1 & 4 & 9 & 16 & 25 \\ 1 & \gamma & \gamma^2 & \gamma^3 & \gamma^4 & \gamma^5 \\ 0 & \gamma & 2\gamma^2 & 3\gamma^3 & 4\gamma^4 & 5\gamma^5 \\ 0 & \gamma & 4\gamma^2 & 9\gamma^3 & 16\gamma^4 & 25\gamma^5 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix} = M \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix}$$

$\det(M) = 4\gamma^3(\gamma - 1)^9 = 4 + 2\gamma^3 \neq 0$ and so we can use recover a_τ as before.

5 Generalization to more servers

In this section we prove Theorem 2. As was mentioned in the introduction, we will allow the database symbols to belong to a slightly larger alphabet $\mathbb{Z}_m$.

Let $q = 2^{r-1}$ denote the number of servers $\mathcal{S}_1, \cdots, \mathcal{S}_q$ for some $r \geq 2$. Let $m = p_1 p_2 \cdots p_r$ where $p_1, p_2, \cdots, p_r$ are distinct primes. By theorem 2.6, there is an explicit S-matching vector family $\mathcal{F} = (\mathcal{U}, \mathcal{V})$ of size n and dimension $k = n^{O\left((\log\log n/\log n)^{1-1/r}\right)}$ where $S = \{a \in \mathbb{Z}_m : a \bmod p_i \in \{0,1\}\ \forall\ i \in [r]\} \setminus \{0\}$. By remark 2.7, $|S \cup \{0\}| = 2^r = 2q$.

The Protocol: We will work over the ring $\mathcal{R} = \mathcal{R}_{m,m} = \mathbb{Z}_m[\gamma]/(\gamma^m - 1)$. The servers represent the database $\mathbf{a} = (a_1, \cdots, a_n) \in \mathbb{Z}_m^n$ as a polynomial $F(\mathbf{x}) \in \mathcal{R}[\mathbf{x}] = \mathcal{R}[x_1, \cdots, x_k]$ given by

$$F(\mathbf{x}) = F(x_1, \cdots, x_k) = \sum_{i=1}^{n} a_i \mathbf{x}^{\mathbf{u}_i},$$

where $\mathcal{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_n)$ are given by the matching vector family $\mathcal{F} = (\mathcal{U}, \mathcal{V})$.

The user samples a uniformly random $\mathbf{z} \in \mathbb{Z}_m^k$ and then sends $\mathbf{z} + t_i \mathbf{v}_\tau$ to $\mathcal{S}_i$ for $i \in [q]$ where $t_i = i - 1$. $\mathcal{S}_i$ then responds with the value of F at the point $\gamma^{\mathbf{z}+t_i\mathbf{v}_\tau}$, that is with $F(\gamma^{\mathbf{z}+t_i\mathbf{v}_\tau})$ and the value of the 'first order derivative' at the same point $F^{(1)}(\gamma^{\mathbf{z}+t_i\mathbf{v}_\tau})$. Notice that the protocol is private since $\mathbf{z} + t\mathbf{v}_\tau$ is uniformly distributed over $\mathbb{Z}_m^k$ for any fixed τ and t.

$\mathcal{U}$: Picks a uniformly random $\mathbf{z} \in \mathbb{Z}_m^k$
$\mathcal{U} \to \mathcal{S}_i : \mathbf{z} + t_i \mathbf{v}_\tau$
$\mathcal{S}_i \to \mathcal{U} : F(\gamma^{\mathbf{z}+t_i\mathbf{v}_\tau}), F^{(1)}(\gamma^{\mathbf{z}+t_i\mathbf{v}_\tau})$

Recovery: Similarly to the 2-server analysis, we define

$$G(t) := F(\gamma^{\mathbf{z}+t\mathbf{v}_\tau}) = \sum_{i=1}^{n} a_i \gamma^{\langle \mathbf{z}, \mathbf{u}_i \rangle + t\langle \mathbf{v}_\tau, \mathbf{u}_i \rangle} = c_0 + \sum_{\ell \in S} c_\ell \gamma^{t\ell},$$

and

$$g(T) = c_0 + \sum_{\ell \in S} c_\ell T^\ell \in \mathcal{R}[T],$$

so that $c_0 = a_\tau \gamma^{\langle \mathbf{u}_\tau, \mathbf{z} \rangle}$ and

$$g(\gamma^t) = G(t) = F(\gamma^{\mathbf{z} + t\mathbf{v}_\tau})$$
$$g^{(1)}(\gamma^t) = \sum_{\ell=0}^{m-1} \ell c_\ell \gamma^{t\ell} = \langle F^{(1)}(\gamma^{\mathbf{z}+t\mathbf{v}_\tau}), \mathbf{v}_\tau \rangle,$$

Hence, the user can calculate the values of $g(\gamma^t), g^{(1)}(\gamma^t)$ for $t = t_1, \cdots, t_q$ and we end up with the following (square) system of equations:

$$\begin{bmatrix} g(\gamma^{t_1}) \\ g^{(1)}(\gamma^{t_1}) \\ \vdots \\ g(\gamma^{t_q}) \\ g^{(1)}(\gamma^{t_q}) \end{bmatrix} = \begin{bmatrix} 1 & \cdots & \gamma^{t_1 \ell} & \cdots \\ 0 & \cdots & \ell \gamma^{t_1 \ell} & \cdots \\ \vdots & & \vdots & \vdots \\ 1 & \cdots & \gamma^{t_q \ell} & \cdots \\ 0 & \cdots & \ell \gamma^{t_q \ell} & \cdots \end{bmatrix} \begin{bmatrix} c_0 \\ \vdots \\ c_\ell \\ \vdots \end{bmatrix} = M \begin{bmatrix} c_0 \\ \vdots \\ c_\ell \\ \vdots \end{bmatrix}$$

where the $2^r = 2q$ columns are indexed by $\ell \in \{0\} \cup S$. Instead of computing the determinant (and the Adjugate matrix), we will use the following Lemma (proven below).

Lemma 5.1. *There exists a row vector*

$$\boldsymbol{\lambda} = [\alpha_1, \beta_1, \cdots, \alpha_q, \beta_q] \in \mathcal{R}^{2q}$$

such that $\boldsymbol{\lambda} M = [\mu, 0, \cdots, 0]$ *for some* $\mu \in \mathcal{R}$ *where* $\mu \mod p_i \neq 0 \ \forall i \in [r]$.

Using this Lemma, the user can recover a_τ as follows. We have

$$\nu := \boldsymbol{\lambda} \begin{bmatrix} g(\gamma^{t_1}) \\ g^{(1)}(\gamma^{t_1}) \\ \vdots \\ g(\gamma^{t_q}) \\ g^{(1)}(\gamma^{t_q}) \end{bmatrix} = \boldsymbol{\lambda} M \begin{bmatrix} c_0 \\ \vdots \\ c_\ell \\ \vdots \end{bmatrix} = [\mu, 0, \cdots, 0] \begin{bmatrix} c_0 \\ \vdots \\ c_\ell \\ \vdots \end{bmatrix} = \mu c_0$$

Taking this equation modulo p_i we get,

$$(\nu \mod p_i) = (\mu c_0 \mod p_i) = (\mu \mod p_i)(a_\tau \mod p_i)\gamma^{\langle \mathbf{u}_\tau, \mathbf{z} \rangle}$$

Let $\mu = \sum_{j=0}^{m-1} \mu_j \gamma^j$ and $\nu = \sum_{j=0}^{m-1} \nu_j \gamma^j$. Since $\mu \mod p_i \neq 0$, there exists j such that $\mu_j \mod p_i \neq 0$. So $(a_\tau \mod p_i) = (\mu_j \mod p_i)^{-1}(\nu_{j+\langle \mathbf{u}_\tau, \mathbf{z} \rangle} \mod p_i)$. So we can find $a_\tau \mod p_i$ for each $i \in [r]$. Finally we use Chinese Remainder Theorem to find $a_\tau \in \mathbb{Z}_m$.

5.1 Proof of Lemma 5.1

For any $\boldsymbol{\lambda} = [\alpha_1, \beta_1, \cdots, \alpha_q, \beta_q] \in \mathcal{R}^{2q}$ we can define a function $h : S \cup \{0\} \mapsto \mathcal{R}$ as:

$$h(\ell) = (\boldsymbol{\lambda} M)_\ell = \left(\sum_{i=1}^{q} \alpha_i \gamma^{t_i \ell}\right) + \ell \left(\sum_{i=1}^{q} \beta_i \gamma^{t_i \ell}\right).$$

Our goal is then to construct an h of this form such that

$$h(\ell) \begin{cases} = 0 & \text{if } \ell \in S \\ = \mu & \text{if } \ell = 0 \end{cases}$$

where $(\mu \mod p_i) \neq 0 \ \forall i \in [r]$.

Notice that, by Chinese Remaindering,

$$\mathcal{R} = \mathcal{R}_{m,m} \cong \mathcal{R}_{p_1,m} \times \ldots \times \mathcal{R}_{p_r,m}, \tag{2}$$

where we recall that $\mathcal{R}_{p_i,m} = \mathbb{Z}_{p_i}[\gamma]/(\gamma^m - 1)$. Therefore, we also get that, for a formal variable x, the rings of univariate polynomials also satisfy

$$\mathcal{R}[x] \cong \mathcal{R}_{p_1,m}[x] \times \ldots \times \mathcal{R}_{p_r,m}[x].$$

In other words, any family of polynomials $f_i \in \mathcal{R}_{p_i,m}[x]$, $i \in [r]$ can be 'lifted' to a single polynomial $f \in \mathcal{R}[x]$ so that $(f \mod p_i) = f_i$ for all i (reducing f mod p_i is done coordinate-wise). Moreover, since this lift is done coefficient-wise (using Eq.2), we get that the degree of f is equal to the maximum of the degrees of the f_i's.

We begin by constructing, for each $i \in [r]$ the following polynomial $f_i(x) \in \mathcal{R}_{p_i,m}[x]$:

$$f_i(x) = \prod_{\ell \in S,\ \ell = 0 \mod p_i} (x - \gamma^\ell)$$

The degree of f_i is $2^{r-1} - 1 = q - 1$ so, by the above comment, we can find a polynomial $f(x) \in \mathcal{R}[x]$ of degree $q - 1$ such that $f(x) \equiv f_i(x) \mod p_i$ for all $i \in [r]$. Define $\alpha_i, i \in [q]$ to be the coefficients of the polynomial f so that $f(x) = \sum_{i=1}^{q} \alpha_i x^{i-1}$. Since we defined $t_i = i - 1$, we have $f(x) = \sum_{i=1}^{q} \alpha_i x^{t_i}$. Define $\beta_i = -\alpha_i$ for all $i \in [q]$. Our final construction of h is thus

$$h(\ell) = f(\gamma^\ell) - \ell f(\gamma^\ell)$$

Claim 5.2. $h(\ell) = 0 \ \forall \ell \in S$

Proof. Since $0 \notin S$, $\ell \neq 0$. We will look at $h(\ell)$ modulo each of the primes.

$$h(\ell) \mod p_i = f_i(\gamma^\ell) - (\ell \mod p_i) f_i(\gamma^\ell) = \begin{cases} f_i(\gamma^\ell) = 0 & \text{if } \ell = 0 \mod p_i \\ f_i(\gamma^\ell) - f_i(\gamma^\ell) = 0 & \text{if } \ell = 1 \mod p_i \end{cases}$$

Therefore, using Chinese Remaindering, $h(\ell) = 0 \ \forall \ell \in S$. □

Claim 5.3. $(h(0) \mod p_j) \neq 0$ *for all* $j \in [r]$

Proof. Suppose in contradiction that $(h(0) \mod p_j) = 0$, then

$$h(0) \mod p_j = f_j(1) = \prod_{\ell \in S,\ \ell = 0 \mod p_j} (1 - \gamma^\ell) = 0.$$

The above equation holds in the ring $\left(\mathbb{Z}_{p_j}[\gamma]/(\gamma^m - 1)\right)$.Therefore, if we consider what happens in the ring $\mathbb{Z}_{p_i}[\gamma] \cong \mathbb{F}_{p_i}[x]$ (we replace the formal variable γ with x to highlight the fact that x does not satisfy any relation) we get that

$$\prod_{\ell \in S,\ \ell = 0 \mod p_j} (1 - x^\ell) = (x^m - 1)\theta(x) \tag{3}$$

for some polynomial $\theta(x) \in \mathbb{F}_{p_j}[x]$. The above equation is an identity in the ring $\mathbb{F}_{p_j}[x]$. So we can check its validity by substituting values for x from the algebraic closure of $\mathbb{F}_{p_j}$. Let $m' = m/p_j$ and let ζ be an element in the algebraic closure of $\mathbb{F}_{p_j}$ of order m' (so $\zeta^\ell = 1$ iff m' divides ℓ). Since m' and p_j are co-prime, such an element exists by Lemma 2.9. If we substitute ζ into Eq. 3, the RHS is zero (since m' divides m). However, each term in the LHS product is nonzero, since if $\ell = 0 \mod p_j$ and m' divides ℓ then $\ell = 0 \mod m$ but we know that $0 \notin S$. Since we are working over the algebraic closure of $\mathbb{F}_{p_j}$ which is a field, the product of nonzero elements is nonzero. This is a contradiction, and so Eq. 3 does not hold. □

6 Concluding remarks

In this work we presented the first 2-server PIR scheme (information theoretic) with sub-polynomial cost. It is unclear what is the optimal communication cost of 2-server schemes and we conjecture that our protocol is far from optimal.

One approach to decrease the communication cost is to take m to be a product of $r > 2$ prime factors in theorem 2.6 to get a larger S-matching vector family where $S = \{a \in \mathbb{Z}_m : a \mod p_i \in \{0, 1\}\ \forall\ i \in [r]\} \setminus \{0\}$ which is of size $2^r - 1$. So we need 2^{r-1} independent equations from each server to find c_0. We can ask the servers for derivatives of F at $\gamma^{\mathbf{z}+t\mathbf{v}_\tau}$ up to order $2^{r-1} - 1$. If these equations are 'independent' i.e. the determinant of the coefficient matrix doesn't vanish then we can find c_0. If we can do this, we can decrease the cost to $n^{O\left(2^r (\log\log n/\log n)^{1-1/r}\right)}$. But observe that for each $l \in S$, $l^2 = l \mod m$ since $l \mod p_i \in \{0, 1\}\ \forall i \in [r]$. So higher order derivatives of g are equal to the first order derivative and we get repeated rows in the coefficient matrix M. One avenue for improvement could be by trying to construct S such that elements of S doesn't satisfy a low-degree monic polynomial.

7 Acknowledgements

We would like to thank Klim Efremenko and Sergey Yekhanin for helpful comments.

References

[Amb97] Andris Ambainis. Upper bound on communication complexity of private information retrieval. In *ICALP*, pages 401–407, 1997.

[BI01] Amos Beimel and Yuval Ishai. Information-theoretic private information retrieval: A unified construction. In *ICALP*, pages 912–926, 2001.

[BIKR02] Amos Beimel, Yuval Ishai, Eyal Kushilevitz, and Jean-François Raymond. Breaking the $o(n^{1/(2k-1)})$ barrier for information-theoretic private information retrieval. In *FOCS*, pages 261–270, 2002.

[CFL+13] Yeow Meng Chee, Tao Feng, San Ling, Huaxiong Wang, and Liang Feng Zhang. Query-efficient locally decodable codes of subexponential length. *Computational Complexity*, 22(1):159–189, 2013.

[CKGS98] Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. Private information retrieval. *J. ACM*, 45(6):965–981, 1998.

[DGY10] Zeev Dvir, Parikshit Gopalan, and Sergey Yekhanin. Matching vector codes. In *FOCS*, pages 705–714, 2010.

[Efr09] Klim Efremenko. 3-query locally decodable codes of subexponential length. In *STOC*, pages 39–44, 2009.

[Gar] William Gararch. A webpage on private information retrieval. `https://www.cs.umd.edu/~gasarch/TOPICS/pir/pir.html`.

[Gas04] William I. Gasarch. A survey on private information retrieval (column: Computational complexity). *Bulletin of the EATCS*, 82:72–107, 2004.

[Gro99] Vince Grolmusz. Superpolynomial size set-systems with restricted intersections mod 6 and explicit ramsey graphs. *Combinatorica*, 20:2000, 1999.

[HH11] Barry Hurley and Ted Hurley. Group ring cryptography. *CoRR*, abs/1104.1724, 2011.

[IS10] Toshiya Itoh and Yasuhiro Suzuki. Improved constructions for query-efficient locally decodable codes of subexponential length. *IEICE Transactions*, 93-D(2):263–270, 2010.

[KdW03] Iordanis Kerenidis and Ronald de Wolf. Exponential lower bound for 2-query locally decodable codes via a quantum argument. In *STOC*, pages 106–115, 2003.

[KS13] C. Koupparis Kahrobaei and V. Shpilrain. Public key exchange using matrices over group rings. *Groups, Complexity, and Cryptology*, 5:97–115, 2013.

[KT00] Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *32nd ACM Symposium on Theory of Computing (STOC)*, pages 80–86, 2000.

[Lip] Helger Lipmaa. A webpage on oblivious transfer or private information retrieval. `http://www.cs.ut.ee/~lipmaa/crypto/link/protocols/oblivious.php`.

[McD84] B. R. McDonald. *Linear Algebra Over Commutative Rings*. Pure and Applied Mathematics #87. Marcel Dekker, New York, 1984.

[RY06] Alexander A. Razborov and Sergey Yekhanin. An $\Omega(n^{1/3})$ lower bound for bilinear group based private information retrieval. In *FOCS*, pages 739–748, 2006.

[WdW05] Stephanie Wehner and Ronald de Wolf. Improved lower bounds for locally decodable codes and private information retrieval. In *ICALP*, pages 1424–1436, 2005.

[WY05] David P. Woodruff and Sergey Yekhanin. A geometric approach to information-theoretic private information retrieval. In *IEEE Conference on Computational Complexity*, pages 275–284, 2005.

[Yek08] Sergey Yekhanin. Towards 3-query locally decodable codes of subexponential length. *J. ACM*, 55(1), 2008.

[Yek12] Sergey Yekhanin. Locally decodable codes. *Foundations and Trends in Theoretical Computer Science*, 6(3):139–255, 2012.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de