Multi-query Computationally-Private Information Retrieval with Constant Communication Rate

Jens Groth[1], Aggelos Kiayias[2], and Helger Lipmaa[3,4]

[1] University College London, UK
j.groth@ucl.ac.uk
[2] Department of Informatics, University of Athens, Greece
aggelos@di.uoa.gr
[3] Cybernetica AS, Estonia
lipmaa@research.cyber.ee
[4] Tallinn University, Estonia

Abstract. A fundamental privacy problem in the client-server setting is the retrieval of a record from a database maintained by a server so that the computationally bounded server remains oblivious to the index of the record retrieved while the overall communication between the two parties is smaller than the database size. This problem has been extensively studied and is known as computationally private information retrieval (CPIR). In this work we consider a natural extension of this problem: a multi-query CPIR protocol allows a client to extract m records of a database containing n ℓ-bit records. We give an information-theoretic lower bound on the communication of any multi-query information retrieval protocol. We then design an efficient non-trivial multi-query CPIR protocol that matches this lower bound. This means we settle the multi-query CPIR problem optimally up to a constant factor.

Keywords: Computationally private information retrieval, multi-query CPIR, lower bound on communication.

1 Introduction

A (single-server) computationally-private information retrieval (CPIR) protocol enables a client to query a database without revealing which data it is extracting. Several cryptographic techniques based on various computational hardness assumptions have been proposed for CPIR with sublinear communication. In this paper, we go beyond the well-studied single-query case and investigate communication-efficient CPIR protocols for the case where the client has multiple queries. A multi-query CPIR protocol allows a client to extract m records of a database containing n records of ℓ bits each. We give an information-theoretic lower bound on the communication and design a multi-query CPIR protocol that matches this lower bound (up to a constant factor). Our focus

P.Q. Nguyen and D. Pointcheval (Eds.): PKC 2010, LNCS 6056, pp. 107–123, 2010.

in this paper is on the theory of multi-query CPIR giving an asymptotically communication-optimal multi-query CPIR protocol under a reasonable cryptographic assumption; we leave the tasks of optimizing the computational overhead and the constant factor gap between the upper and lower bounds on the communication as open problems.

oblivious transfer (OT) or symmetric CPIR (SCPIR) protocol. A two-message SCPIR protocol is usually required to be secure in the sense of semisimulatability, first defined by Naor and Pinkas [12].

1.1 Background

Private Information Retrieval was introduced in [3]. Kushilevitz and Ostrovsky [9] showed that it is possible to do CPIR with sublinear communication. Cachin, Micali and Stadler [2] gave the first CPIR-protocol for retrieving one bit out a database where the communication complexity is polylogarithmic in the database size. The communication-wise best single-query CPIR protocols up-to-date are by Gentry and Ramzan [6] and Lipmaa [10] that allow the retrieval of an ℓ-bit record from the database.

Turning to our problem, there are three trivial solutions to m-query CPIR: One option is parallel repetition of a single-query CPIR. In the case of repeating Gentry and Ramzan's CPIR [6] this would result in a communication of $\Theta(m \cdot \log n + m \cdot \ell + m \cdot k)$, where k is a security parameter specifying the length of an RSA-modulus. As we will see this is *not* optimal.

Another option is to use a single-query CPIR protocol to fetch one $m\ell$-bit element from an $\binom{n}{m}$-element database. As our lower bound shows, this solution has asymptotically optimal communication $\Theta(m \cdot \log_2(n/m) + m \cdot \ell + k)$ when combined with Gentry and Ramzan's CPIR, but unfortunately increases the server's computation to $\Omega(\binom{n}{m})$, which for many choices of n and m is superpolynomial in the security parameter.

A third option is to transmit the entire database to the client and is inefficient in terms of communication.

Ishai, Kushilevitz, Ostrovsky and Sahai [7] proposed batch-codes for encoding a database over many separate blocks such that a client can extract m records by querying only a smaller number of records from each block. Our solution uses a related strategy and part of this paper consists of encoding the database in separate blocks that can be queried by separate smaller CPIR protocols. The batch-codes by Ishai, Kushilevitz, Ostrovsky and Sahai, however, do not apply directly to our problem. One reason is that their batch-codes are optimized with respect to keeping the total number of records in all the blocks low in order to keep the computational complexity low, whereas our solution actually uses an encoding where the total number of records in all the blocks becomes quite large. Another difference between the works is that they only consider the case where the database and the blocks use the same alphabet, for instance ℓ-bit strings, while we in some instances will encode the database into blocks of records from a different alphabet.

1.2 Our Contribution

We design a computationally efficient two-message multi-query CPIR protocol with $\Theta(m\ell + m \cdot \log_2(n/m) + k)$ bits of communication, where k is a security parameter specifying the size of an RSA modulus. Server computation is dominated by $\Theta(n\ell)$ group operations, where in both cases the constant in big-Θ is reasonably small. The client's privacy is based on a variant of the Φ-hiding assumption [2,6]. In our construction, we use a multi-query CPIR variant of Gentry and Ramzan's single-query CPIR [6] that works for a restricted set of parameters (m, n, ℓ). We present a reduction demonstrating that any multi-query CPIR protocol that works for a restricted set of parameters can be used as a building block to construct a communication-optimal CPIR protocol for *any* set of parameters.

We also prove that any perfectly correct multi-query (non-private) information retrieval protocol has an information theoretical lower bound of $\Omega(m \cdot \log_2(n/m) + m\ell)$ bits of communication. Thus, our proposed multi-query CPIR has optimal communication complexity up to a constant factor.

1.3 Challenges and Techniques

Known techniques suffice for communication-optimal CPIR in the extreme cases, where the number of queries is very small or very large. If $m = \Omega(n)$ the server can send the entire database to the client in the clear, giving a communication complexity of $n\ell = O(m\ell + m\log(n/m))$ bits. If $m = O(1)$ we can invoke Gentry and Ramzan's single-query CPIR m times in parallel to get a communication complexity of $O(\ell + k) = O(m\ell + m\log(n/m) + k)$ bits. We are interested in finding a communication-efficient CPIR protocol for the case where m is in between the two extremes. Indeed, if $m = o(n)$, then downloading the entire database at a cost of $n\ell$ bits would be sub-optimal and when $m = \omega(1)$ simply repeating Gentry and Ramzan's protocol has an additive overhead of $\Omega(mk)$ bits, which would make it a sub-optimal choice.

A first step towards resolving this issue is Gentry and Ramzan's observation [6] that while they focused on the single-query case, it is also possible to get a restricted multi-query CPIR protocol with their techniques. We will use such a restricted multi-query CPIR scheme as a building block in our construction. The restricted protocol is only communication-optimal for certain choices of (m, n, ℓ) though. It encodes the queries as hidden prime-powers, however, when n grows, the size of these primes grows as well. When $\ell = \Omega(\log n)$ or $m = O(n^{\epsilon})$ for a constant $\epsilon > 0$ this turns out not to be a problem, but when ℓ is small and $m = \omega(n^{\epsilon})$ the increase of the prime size causes a loss of bandwidth of up to a factor $\log n$.

To eliminate the up to a factor $\log n$ overhead in the communication complexity we will encode the database in such a way that it can be split into smaller pieces that can be processed by the restricted multi-query CPIR protocol. One part of this encoding consists of dividing the database into smaller blocks that will be treated separately. With smaller blocks, we need smaller primes in the

Gentry-Ramzan CPIR to specify a particular index of a record and this improves the communication complexity. To spread the queries evenly on the blocks, we first let the client choose a random permutation of the database. To preserve the sublinear communication complexity, the client does this by sending the server a seed for a pseudorandom number generator from which the longer full permutation can be generated.

Another part of our encoding is best explained by an example. Suppose we have a database of 4 one-bit records and the client wants retrieve two records. We can encode the database as a 6-record database containing 2-bit elements for each possible pair of queries $(1, 2)$, $(1, 3)$, $(1, 4)$, $(2, 3)$, $(2, 4)$, $(3, 4)$ the client could have. This encoding increases the size of the database and the size of the records, but reduces the number of queries the client needs to make. When the client's encoding of queries as primes permits the extraction of many bits at a time, then this encoding improves bandwidth since fewer queries are needed. Batch-codes [7] address a related encoding problem, however, as explained in the section on related work neither of their batch-codes suffice for minimizing communication in our scheme.

With respect to the lower bound on communication, the challenge is that we must consider all possible multi-query CPIR protocols. Most known CPIR protocols consist of encoding the queries in a single message that is sent to the server, which information-theoretically leads to a lower bound of $\log_2(\binom{n}{m}) = \Omega(m \cdot \log(m/n))$. Furthermore, obviously m ℓ-bit strings cannot be communicated using less than $m\ell$ bits. For these protocols it is therefore straightforward to get a lower bound of $\Omega(m{\cdot}\log_2(n/m){+}m\ell)$ bits. However, the lower bound also needs to cover the case of multi-query CPIR protocols that work in a different way and may use more rounds. This makes proving the lower bound non-trivial; we do not know of prior work giving such a lower bound even in the single-query case.

1.4 Roadmap

In Sect. 2, we present the necessary preliminaries. In Sect. 3, we prove a lower bound for the communication complexity of multi-query CPIR (even when privacy is not required). In Sect. 4, we construct a basic restricted multi-query CPIR protocol based on Gentry and Ramzan's work [6]. In Sect. 5, we design a new multi-query CPIR protocol for any parameter values.

2 Preliminaries

Notation. All our algorithms take as input a security parameter k. In the following we say a function f is negligible if $f(k) = k^{-\omega(1)}$. We write $f(k) \approx g(k)$ if $|f(k) - g(k)|$ is negligible. We write $(\mathrm{out}_C, \mathrm{out}_D) \leftarrow \langle C(x), D(y) \rangle$ if C on input x and D on input y output respectively out_C and out_D after interacting with each other.

Multi-query CPIR. Consider a database with records $x_1, \ldots, x_n \in \{0, 1\}^\ell$. Informally, a multi-query CPIR protocol is a protocol that allows a client to extract m different records $x_{i_1}, \ldots, x_{i_m}$ from the database, without revealing which records it extracted. Formally, a multi-query CPIR protocol consists of two interactive polynomial time Turing machines C and D that we call respectively the client and the server. Both parties get as input a security parameter k written in unary and additional parameters m, n, ℓ. The server takes as an input n elements $x_1, \ldots, x_n \in \{0,1\}^\ell$. The client takes as an input a set of m different indexes $i_1, \ldots, i_n \in \{1, \ldots, n\}$. (Note that since we are interested in minimal communication in the case of fixed m, we can assume that all m indexes are different.) The client and server interact, and in the end the client outputs $y_1, \ldots, y_m \in \{0,1\}^\ell$ or a special failure symbol $\perp$. (C, D) is a multi-query CPIR protocol if it satisfies the standard correctness and privacy properties as defined below. Intuitively, correctness means that in the case of an honest client and an honest server, the client always retrieves correct elements $x_{i_1}, \ldots, x_{i_m}$. Privacy is defined in the sense of indistinguishability: given two input tuples $\boldsymbol{i^0}$, $\boldsymbol{i^1}$ chosen by a malicious server, the server should not be able to guess which of the two tuples the client actually uses.

Definition 1 (Perfect correctness). *A multi-query CPIR protocol* (C, D) *has perfect correctness if for any* k, *any* $m, n, \ell = \mathrm{poly}(k)$ *and any* $\boldsymbol{i} = (i_1, \ldots, i_n)$ *and* $\boldsymbol{x} = (x_1, \ldots, x_n)$ *with* $i_j \in \{1, \ldots, n\}$ *and* $x_j \in \{0,1\}^\ell$, *we have that if* $(\mathrm{out}_C, \mathrm{out}_D) \leftarrow \langle C(1^k, m, n, \ell, \boldsymbol{i}), D(1^k, m, n, \ell, \boldsymbol{x})\rangle$, *then* $\mathrm{out}_C = (x_{i_1}, \ldots, x_{i_m})$.

Definition 2 (Computational privacy). *A multi-query CPIR protocol has computational privacy if for all non-uniform polynomial time adversaries* $\mathcal{A}$ *we have*

$$\Pr\left[\begin{array}{l} b \leftarrow \{0,1\}, (m, n, \ell, \boldsymbol{i^0}, \boldsymbol{i^1}, \mathrm{state}) \leftarrow \mathcal{A}(1^k), \\ (\mathrm{out}_C, \mathrm{out}_\mathcal{A}) \leftarrow \langle C(1^k, m, n, \ell, \boldsymbol{i^b}), \mathcal{A}(\mathrm{state})\rangle : \mathrm{out}_\mathcal{A} = b \end{array}\right] \approx \frac{1}{2},$$

where $m, n, \ell = \mathrm{poly}(k)$ *and* $\boldsymbol{i^j} = (i^j_1, \ldots, i^j_m)$ *with* $1 \le i^j_1 < \ldots < i^j_m \le n$.

3 Lower Bound on (m, n, ℓ)-CPIR Communication

Let the database contain n records of size ℓ. An (m, n, ℓ) information retrieval is a two-party protocol between a client and a server that enables the client to receive any m out of the n records. In this section we will establish a lower bound for any perfectly correct (m, n, ℓ) information retrieval protocol, private or not. If the protocol consists of the user indicating the desired indices and the server sending the elements of those indices a straightforward lower bound of $\log_2 \binom{n}{m} + m\ell$ bits applies. Establishing that the same lower bound applies also in the general case requires more work. We show that $\Omega(m \cdot \log(n/m) + m \cdot \ell)$ is in fact the lower bound for any perfectly correct information retrieval protocol. The lower bound is information theoretical and holds even when the client and server are computationally unbounded. The lower bound assumes the protocol to have perfect correctness, so any choice of fixed random tapes also gives a perfectly correct protocol, which means it suffices to prove the lower bound for any fixed pair of random tapes. We will therefore in the following without loss of generality assume that the client and server are deterministic.

Denote by X the set of all subsets of n elements of size m and by Y the set of all n-tuples over the alphabet $\Sigma = \{0,1\}^\ell$. The output of any (m,n,ℓ) information retrieval protocol belongs to the set Z of m-tuples over Σ. We denote by $f : X \times Y \to Z$ the output of the information retrieval.

Any protocol computing f can be represented by a binary tree (cf. [8]) so that each internal node v is labeled by a function $c_v : X \to \{0,1\}$ or $s_v : Y \to \{0,1\}$. The root of the tree is the initial node of the protocol and an execution involves following a path from root to leaf according to the functions of the nodes. Note that the program of the client is determined by all the $c_v(\cdot)$ functions where the program of the server is determined by all the $s_v(\cdot)$ functions. For the purpose of obtaining the most general lower bound we make no assumptions on the complexity of these functions. Finally, each leaf holds a value $z \in Z = \{0,1\}^{\ell m}$ which is the output of the client.

For any such protocol we define the equivalence relation between two inputs $(x_1,y_1) \sim (x_2,y_2)$ if they lead the protocol to the same output leaf. For each leaf λ there is a different equivalence class R_λ, and the set of all equivalence classes of $\sim$ is thus parameterized by the set of all leaves. It holds that for any λ, the set R_λ is a combinatorial rectangle: $(x_1,y_1) \in R_\lambda, (x_2,y_2) \in R_\lambda$ implies that $(x_1,y_2),(x_2,y_1) \in R_\lambda$. This follows from the fact that the leaves define unique paths from the root and in each node the path the protocol takes only depends on one of the two inputs. A *fooling set* F_z for some $z \in Z$, on the other hand, is a subset of $X \times Y$ for which it holds that for any $(x_1,y_1),(x_2,y_2) \in F_z$ with $f(x_1,y_1) = f(x_2,y_2) = z$ we have that either $f(x_1,y_2) \neq z$, or $f(x_2,y_1) \neq z$. Fooling sets are useful for lower bounds as they can only be covered by as many f-monochromatic rectangles as their cardinality (a rectangle R is f-monochromatic iff $\exists z : (x,y) \in R \Rightarrow f(x,y) = z$). The number of monochromatic rectangles in turn yields a lower bound on the number of leaves in any protocol tree which then implies a lower bound on the tree's height (which is equal to the total communication) [8]. In a nutshell, the number of leaves in the protocol tree for any protocol computing the function f must be at least $\sum_{z \in Z} |F_z|$ where F_z is a fooling set for the output value $z \in Z$.

Lemma 1. *Fix n, m, ℓ. Let $z = (z_1, \ldots, z_m)$ be such that $\{z_1, \ldots, z_m\} \subsetneq \{0,1\}^\ell$. Define $L(z) := \mathrm{lexmin}\left(\{0,1\}^\ell \setminus \{z_j\}_{j=1}^m\right)$, where the function $\mathrm{lexmin}(A)$ denotes the lexicographically smallest element of the set of strings A. The set*

$$F_z = \left\{(I, y_1, \ldots, y_n) \mid I = \{i_1, \ldots, i_m\} \subseteq \{1, \ldots, n\}, y_{i_j} = z_j, y_{i'} = L(z)\right\}$$

is a fooling set of size $\binom{n}{m}$, where the indexes have the ranges $j = 1, \ldots, m, i' \in \{1, \ldots, n\} \setminus \{i_1, \ldots, i_m\}$, and $1 \leq i_1 < \cdots < i_m \leq n$.

Proof. It is obvious that $|F_z| = \binom{n}{m}$ and that any input $(x,y) \in F_z$ satisfies that $f(x,y) = z$. We next show that F_z is a fooling set. Let $(I; y_1, \ldots, y_n), (I'; y'_1, \ldots, y'_n) \in F_z$. Observe that it should be $I \neq I'$, thus there is at least one location in I that is not in I'. Without loss of generality, say $i_1 \in I \setminus I'$. It follows that $f(I; y'_1, \ldots, y'_n) = (z'_1, \ldots, z'_m) \neq z$ since $z'_1 = L(z) \neq z_1$. □

Observe that if $2^\ell > m$ then trivially $\{z_1, \ldots, z_m\} \subsetneq \{0,1\}^\ell$ for any possible output tuple $z \in Z = \{0,1\}^{\ell m}$, i.e., there would be $2^{\ell m}$ possible outputs for which the lemma above applies. On the other hand, when $2^\ell \leq m$ the number of possible outputs for which the lemma applies is at least $(2^\ell - 1)^m$. This follows from the fact that there are at least that many m tuples that ommit a specific ℓ-bitstring. While this lower bound can be made more tight it will be sufficient for our communication complexity argument.

Theorem 1. *Consider parameters n, m, ℓ with $n \geq m$. The communication complexity of any protocol solving the (m, n, ℓ) information retrieval problem is $\Omega(m \cdot \log_2(n/m) + m \cdot \ell)$.*

Proof. Consider first the case $\ell > 1$. There are at least $(2^\ell - 1)^m$ possible outputs of the information retrieval protocol for which the corresponding fooling set has cardinality $\binom{n}{m}$ according to lemma 1. It follows that the communication complexity is lower-bounded by

$$\left\lceil \log_2 \left(\binom{n}{m} \cdot (2^\ell - 1)^m \right) \right\rceil \geq \left\lceil \log_2 \binom{n}{m} + m \cdot \log_2(2^\ell - 1) \right\rceil$$
$$\geq \left\lceil \log_2 \binom{n}{m} \right\rceil + m(\ell - 1) \ .$$

In order to obtain the asymptotic bound, we use the fact $\binom{tm}{m} \geq t^m$ for any $t, m \geq 1$ and by setting $t = n/m$ we obtain the statement of the theorem. Next, consider the case $\ell = 1$ and $2m < n$. In this case, there are 2 choices where the fooling set has cardinality $\binom{n}{m}$. It follows that the communication complexity would be at least $\log_2 \binom{n}{m} = \Omega(m \cdot \log(n/m))$. Finally in case $\ell = 1$ and $2m \geq n$ trivially the communication is at least m bits, from which the statement of the theorem follows. □

4 Restricted Multi-query CPIR

In Sect. 5, we will construct a multi-query CPIR with communication complexity $O(m\ell + m \cdot \log(n/m) + k)$, where k is a security parameter. As a building block, it will use a multi-query CPIR protocol (C, D) with communication complexity k. Since communication is bounded by k, such a CPIR cannot handle all choices of (m, n, ℓ). What we will need from the building block is that it can be used whenever

$$m \leq \frac{\alpha k}{\ell + \log_2 n} \ , \tag{1}$$

for *some* constant $0 < \alpha < 1$.

In this section, we provide a construction of such a building block—that we call a *restricted multi-query CPIR*—based on the previous single-query CPIR of Gentry and Ramzan [6] and their observation that it can be extended to the multi-query setting. The security of the restricted multi-query CPIR relies on the Φ-hiding assumption by Cachin, Micali and Stadler [2]. It is a 2-message multi-query CPIR protocol, so we will describe it by three algorithms $(\mathrm{Query}, \mathrm{Response}, \mathrm{Extract})$ that generate respectively the client's query, the server's response, and finally allow the client to extract the records from the response.

In the following, Π will be a deterministic polynomial-time algorithm that takes n as input and generates n (small) prime numbers. K will be a probabilistic polynomial time key generator that takes as input the security parameter k and an integer $\pi < 2^{2\alpha k}$ with factors in the list generated by Π, where α is a constant parameter. On such an input it generates a triple (G, g, q), such that G is a group with efficiently computable operations, q is a positive integer, and g is an element of this group with $\mathrm{ord}(g) = \pi q$. We require that the description of G and group elements are at most $k/3$ bits each and that K satisfies the following assumption:

Definition 3 (Decision Subgroup Assumption). *There exists some $\alpha \in (0, 1)$ such that for all probabilistic polynomial-time $\mathcal{A}$,*

$$\Pr\left[\begin{matrix} b \leftarrow \{0,1\}, (1^n, \pi_0, \pi_1, \mathrm{state}) \leftarrow \mathcal{A}(1^k), (G, g, q) \leftarrow K(1^k, \pi_b) : \\ \mathcal{A}(G, g, \mathrm{state}) = b \end{matrix}\right] \approx \frac{1}{2} \enspace ,$$

where the adversary outputs positive integers $\pi_0, \pi_1 < 2^{2\alpha k}$ with factors in $\Pi(n)$.

As an example, we may choose $N = PQ$ as a $k/3$-bit RSA modulus, where $P = 2\pi r + 1$ and $Q = 2st + 1$ and r, s, and t are large random positive integers, and select g as a random element of $\mathbb{Z}_N^*$ that satisfies $\mathrm{ord}(g) = \pi q$ for some q with $\gcd(\pi, q) = 1$. When Π generates a set $p_1, \ldots, p_n$ where $2n < p_1 < \cdots < p_n$ it is shown by Gentry and Ramzan [6] that the assumption above reduces to a variant of the Φ-hiding assumption[1]:

$$\Pr\left[\begin{matrix} b \leftarrow \{0,1\}, (1^n, \pi_0, \pi_1, \mathrm{state}) \leftarrow \mathcal{A}(1^k), N \leftarrow \mathrm{RSAK}(1^k, \pi_b) : \\ \mathcal{A}(N, \mathrm{state}) = b \end{matrix}\right] \approx \frac{1}{2} \enspace ,$$

where $\mathcal{A}$ outputs π_0 and π_1 as described above, and RSAK outputs a $k/3$-bit RSA-modulus N. To avoid factorization attacks due to Coppersmith when a large factor of $\phi(N)$ is known, in this instantiation, the parameter α should be appropriately selected. Specifically, due to the fact that when a factor of $\phi(N)$ that is larger than $N^{1/4}$ is known then it is possible to factor N (Coppersmith [5,4], cf. [1] and the related discussion in [6]) we need to choose

$$\alpha \leq 1/25 \enspace .$$

[1] Strictly speaking Gentry and Ramzan only show this for π being a prime power, however, their proof carries over without change to the more general case where π can be a composite number.

Indeed, with this choice we have that $\pi < 2^{2k/25}$ which is smaller than $2^{(k/3-1)/4} \leq N^{1/4}$ as long as $k \geq 75$.

Given (Π, K), we can construct a 2-message (m, n, ℓ)-CPIR following the protocol of Gentry and Ramzan. This restricted (m, n, ℓ)-CPIR protocol works for choices of the parameters that satisfy Eq. (1):

Query: Let $p_1, \ldots, p_n$ be the primes generated by Π. Let $\pi_1, \ldots, \pi_n$ be the smallest prime powers of $p_1, \ldots, p_n$ that are larger than 2^ℓ, i.e., $\pi_i = p_i^{\lceil \ell / \log_2 p_i \rceil}$. Let $i_1, \ldots, i_m$ be different indexes of the elements the client wants to extract from the database. Define $\pi = \prod_{j=1}^m \pi_{i_j}$ and run $(G, g, q) \leftarrow K(1^k, \pi)$. Send (G, g) to the server, and store q for later use.

Response: Given a database of ℓ-bit elements $x_1, \ldots, x_n$, use the Chinese remainder theorem to compute x' so $x' \equiv x_i \mod \pi_i$ for $1 \leq i \leq n$ and send $c = g^{x'}$ to the client.

Extract: For each $1 \leq j \leq m$, compute $c_j = c^{q\pi/\pi_{i_j}}$ and $g_j = g^{q\pi/\pi_{i_j}}$, and find x_{i_j} so that $c_j = g_j^{x_{i_j}}$. Output $(x_{i_1}, \ldots, x_{i_m})$.

Observe that the extraction step requires solving m instances of the discrete logarithm problem within the cyclic groups $\langle g_j \rangle$ for $j = 1, \ldots, m$, where the order of each such subgroup is π_{i_j}. Given that π_{i_j} is a power of the prime p_{i_j}, the extraction requires $O(m\sqrt{p_n}(\ell / \log_2 p_1))$ steps using Giant-Step Baby-Step techniques when the user computes the x_i-s as is done in the Pohlig-Hellman algorithm [13]. The server's computation consists of one $\Theta(\ell n)$-bit exponentiation as in [6].

Now that we have given the protocol, let us explain the constraints on the parameters. By definition we have $1 \leq m, n, \ell = k^{O(1)}$ but we need more limiting constraints since we use only k bits of communication. Since π_i are chosen as the smallest prime powers of p_i that are larger than 2^ℓ we have $\pi_i < 2^\ell p_n = 2^{\ell + \log_2 p_n}$. This means $\pi < 2^{m(\ell + \log_2 p_n)}$ so we have $\pi < 2^{2\alpha k}$ whenever

$$m \leq \frac{2\alpha k}{\ell + \log_2 p_n} \ .$$

Using the constraints in the example given by Gentry and Ramzan based on RSA moduli, we may use Π that generates the first n primes larger than $2n$. We can use the following crude bound on the primes $2n < p_1 < \cdots < p_n < 2n^2$ for $n \geq 2$. For $n \geq 2$ we therefore can use the restricted multi-query CPIR protocol whenever Eq. (1) holds.

We observe that when $n = 1$ we do not need any security assumption, since the client only has one choice of index to query and therefore privacy is not a concern. We also remark that if the key generation algorithm has negligible failure probability, we still have computational privacy if the client reveals the indices $i_1, \ldots, i_m$ on key generation failure. This means we can get perfect correctness in the CPIR. In conclusion, we have the following theorem:

Theorem 2. *If the decision subgroup assumption (definition 3) holds for a constant* $0 < \alpha < 1$*, there exists a 2-message multi-query CPIR with perfect* *correctness, computational privacy and k bits of communication for parameters* (m, n, ℓ) *satisfying Eq.* (1).

Proof. Follows from discussion above. □

Discussion. Recall that by Eq. (1), $k = \Omega(m \log_2 n + m\ell)$, thus the restricted protocol has communication $\Omega(m \log_2 n + m\ell)$. It may seem that we achieve no gain over the m-times parallel repetition of Gentry-Ramzan's CPIR protocol that has communication $\Theta(m \log_2 n + m\ell + mk)$ for some security parameter k. However, the gain is actually quite significant, especially when $k \gg \log n$.

For example, consider the case $\ell = 1$. Then, m-times repetition of the Gentry-Ramzan protocol gives us a multi-query protocol with communication $m \cdot k$. Now suppose that $m = \sqrt{k}$ and $n = k^{2/3}$. The number of bits used in the transcript is $k^{3/2}$. On the other hand, when we use the restricted multi-query protocol, $\sqrt{k} \leq \frac{\alpha k}{1+\log_2 k}$ and thus we get a protocol with communication k. Thus for large values of m the protocol of this section outperforms the m-times repetition of Gentry-Ramzan's single-query CPIR protocol.

5 Communication-Optimal Perfectly Correct Multi-query CPIR

Our optimal communication reduction of arbitrary multi-query CPIR to the restricted multi-query CPIR will use the restricted CPIR protocol from [6] described above and a pseudorandom number generator PRG. Our transformation operates in four different modes depending on the choice of the parameters (m, n, ℓ). We examine these modes of operation in the following four subsections. The most challenging case is the one that m is relatively large but not as large as to enable the trivial protocol that sends the whole database to be a good solution. We start with the easier cases first.

5.1 Multi-query (m, n, ℓ)-CPIR for Constant n/m

When $n = O(m)$ it is asymptotically communication-optimal to send the entire database to the client. For concreteness, we fix the implicit constant in the big-O notation to be 9 and send the entire database to the client whenever $n \leq 9m$. It is obvious this is a 1-message multi-query CPIR protocol that has perfect correctness and privacy, and optimal communication of $n\ell \leq 9m\ell = O(m\ell)$ bits. In this case, the server does not do any computation except what is needed for the transmission of the database.

5.2 Multi-query (m, n, ℓ)-CPIR for Small m

We will now give a simple extension of the restricted multi-query CPIR that is communication-optimal when $m \leq k^{2/3}$ and $n > 9m$. We do this by chopping the ℓ-bit records into smaller pieces of size e. This gives us $\lceil \ell/e \rceil$ databases containing e-bit strings. We run the restricted multi-query CPIR protocol (C, D) to extract

m records in each of these databases. In order to do this we have to select the parameter e suitably so that the parameter restriction for the restricted CPIR is satisfied.

1. Define $e = \min(\ell, \lfloor \alpha k/m - \log_2 n \rfloor)$.
2. The server splits $(x_1, \dots, x_n)$ into $\lceil \ell/e \rceil$ databases $\{(x_{h,1}, \dots, x_{h,n})\}_{h=1}^{\lceil \ell/e \rceil}$, where all $x_{h,i}$ are e-bit strings and x_i is the concatenation of $x_{1,i}, \dots, x_{\lceil \ell/e \rceil, i}$.
3. The client and server run $\lceil \ell/e \rceil$ restricted multi-query CPIR protocols in parallel for $h \in \{1, \dots, \lceil \ell/e \rceil\}$: $(x_{h,i_1}, \dots, x_{h,i_m}) \leftarrow \langle C(1^k, m, n, e, i_1, \dots, i_m), D(1^k, m, n, e, x_{h,1}, \dots, x_{h,n}) \rangle$.
4. The client computes $x_{i_1}, \dots, x_{i_m}$ by concatenating the restricted multi-query CPIR outputs $\{(x_{h,i_1}, \dots, x_{h,i_m})\}_{h=1}^{\lceil \ell/e \rceil}$ for each index.
5. The client outputs $(x_{i_1}, \dots, x_{i_m})$.

The above protocol runs in the same number of rounds as the restricted multi-query CPIR protocol. If $\ell \leq \lfloor \alpha k/m - \log_2 n \rfloor$ we just need one copy of the restricted protocol, so we get a communication complexity of k bits. If $\ell > \lfloor \alpha k/m - \log_2 n \rfloor$ we get a communication complexity of

$$\lceil \ell/(\lfloor \alpha k/m - \log_2 n \rfloor) \rceil \cdot k < \Big(\frac{\ell}{\alpha k/2m} + 1 \Big) k = \frac{2m\ell}{\alpha} + k = O(k + m\ell) \ ,$$

provided

$$\lfloor \frac{\alpha k}{m} - \log_2 n \rfloor \geq \frac{\alpha k}{2m} \ .$$

The latter condition holds for large enough k, because $m \leq k^{2/3}$ and $\log_2 n = O(\log_2 k)$ implies

$$k \geq \frac{2m(1 + \log_2 n)}{\alpha}$$

asymptotically, which in turn implies

$$\frac{\alpha k}{m} - \log_2 n - 1 \geq \frac{\alpha k}{2m} \ .$$

Note that one can further optimize this protocol, since for each h the client's uses the same indices and therefore may choose to use the same initial query every time.

Lemma 2. *The multi-query (m, n, ℓ)-CPIR protocol described above for $m \leq k^{2/3}$ is correct and private under the assumption that the underlying restricted CPIR protocol satisfies these properties. Moreover, if the restricted CPIR protocol has perfect correctness the CPIR protocol above has perfect correctness as well.*

Proof. By the choice of e we guarantee that

$$m \leq \frac{\alpha k}{e + \log_2 n}$$

as required by Eq. (1). A hybrid argument shows that (perfect) correctness follows from the (perfect) correctness of the restricted multi-query CPIR protocol. Another hybrid argument shows that if an adversary has advantage ϵ in breaking the privacy of the CPIR protocol, then we can break the restricted CPIR protocol with probability

$$\frac{\epsilon}{\lceil \ell/e \rceil} > \frac{\epsilon}{(\ell+1)}$$

where the last inequality holds for values of

$$k \geq \frac{2m(1+\log_2 n)}{\alpha} \enspace .$$

5.3 Multi-query (m,n,ℓ)-CPIR for Large Values of m and $\ell \leq \log_2(n/m)$

We will now consider the case, where $9m < n \wedge k^{2/3} < m \wedge \ell \leq \log_2(n/m)$. Let $b, d \in \mathbb{N}$ be two parameters to be specified below. We split the database into $\lceil \frac{n}{bd} \rceil$ blocks of size bd, and on each of these blocks we will use the restricted multi-query CPIR protocol. Note that if it happens that the clients' queries are evenly distributed then we only need to extract an average of $\frac{mbd}{n}$ records from each of these blocks.

To ensure the uniformity of its queries, the client will choose a seed $s \leftarrow \{0,1\}^k$ for a pseudorandom number generator. From this pseudorandomness seed, the client and the server can generate a pseudorandom permutation of the n elements. From now on we can therefore assume that the client's indices $i_1, \ldots, i_m$ are randomly distributed. Still, we cannot expect that each block has exactly $\frac{mbd}{n}$ records that need to be extracted. We will therefore choose $a = bm/n$ and extract $2ad$ records from each block. We will choose b, d such that ad is large enough to give us negligible probability that the pseudorandom permutation places more than $2ad$ records in any single block.

Recall that the restricted multi-query CPIR lets us extract $2ad$ records from each block provided that, following Eq. (1),

$$2ad \leq \frac{\alpha k}{\ell + \log_2(bd)} \enspace .$$

When ℓ is small, for instance when $\ell = 1$, this means that we need k bits to extract

$$2ad \leq \frac{\alpha k}{\ell + \log_2(bd)} < \frac{\alpha k}{\log_2(bd)}$$

database bits, giving us a non-constant communication rate.

We will get around this problem by using an encoding of the block that gives a more efficient utilization of the bandwidth. The encoding divides each block of size bd into d segments of b records. We then encode each segment by enumerating all possible combinations of a elements that can be drawn from this segment. This gives us a segment of $\binom{b}{a}$ strings of length $a\ell$. On average we desire to extract two $(a\ell)$-bit records from each of the d segments. In reality, the $2ad$ records we need to extract from the block are pseudorandomly distributed on the d segments, but by extracting $3d$ $(a\ell)$-bit strings from the d segments, we are guaranteed to cover any distribution of $2ad$ records in the block. This is an immediate corollary of the following simple counting lemma:

Lemma 3. *Let $a, b, d \in \mathbb{N}$ and let $S_1, \ldots, S_d$ be disjoint sets with $|S_i| = b$. For any $A \subseteq \cup_{i=1}^{d} S_i$ with $|A| = 2ad$, there exists a family of sets $G_1, \ldots, G_t$ such that (i) for each G_j there is some S_i with $G_j \subseteq S_i$, (ii) $|G_j| = a$, (iii) $A \subseteq \cup_{j=1}^{t} G_j$ and (iv) $t \leq 3d$.*

Proof. Let $A_1, \ldots, A_d$ be the partition of A across $S_1, \ldots, S_d$ with $|A_i| = a_i$ and $\sum_{i=1}^{d} a_i = 2ad$. Each A_i can be covered by $\lceil \frac{a_i}{a} \rceil$ subsets of size a from S_i. It follows that we can cover A with a number of sets that equals $\sum_{i=1}^{d} \lceil \frac{a_i}{a} \rceil \leq d + (\sum_{i=1}^{d} a_i)/a = 3d$.

In conclusion, on each block we use the restricted multi-query CPIR to extract $3d$ out of $d \cdot \binom{b}{a}$ possible $(a\ell)$-bit strings. According to Eq. (1), we can use the restricted multi-query CPIR protocol to do this if we choose b, d such that

$$3d \leq \frac{\alpha k}{a\ell + \log_2(d\binom{b}{a})} \quad . \tag{2}$$

Let us now give the constraints we have on the choices of b, d and give a possible choice of variables that gives us optimal communication complexity:

– We want $ad = mb/n \cdot d$ to be so large that there is negligible probability of more than $2ad$ records falling into the same block.
– We need Eq. (2) in order to use the restricted multi-query CPIR protocol.
– Finally, we want $d \cdot \binom{b}{a}$ to be polynomial in k so that the encoded database contains $k^{O(1)}$ elements and hence it is processed in polynomial time in k.

We first use a Chernoff-bound on the probability that for any given bd block there will be more than $2ad$ records that we want to extract. For a fixed bd block, the probability of more than $2ad$ indices needing extraction is smaller than the probability of more than $2ad$ indices ending up in the same block if we allow repetition. The latter probability is $\Pr[X > 2ad]$ where X is a random variable with $X = \sum_{i=1}^{bd} X_i$ and $X_1, \ldots, X_{bd}$ are independent Bernoulli trials with probability $p = m/n$. By using a Chernoff bound we get

$$\Pr[X > 2ad] < e^{-ad/3} \quad . \tag{3}$$

For the latter condition to hold, we choose $a = \lceil \frac{\log k}{\log(n/m)} \rceil$ and $b = a\lceil n/m \rceil$ giving

$$\binom{b}{a} \leq (e\frac{b}{a})^a \leq (e\lceil \frac{n}{m} \rceil)^{\frac{\log k}{\log(n/m)}+1} < e^{\log k + 1} \cdot 2^{\log(n/m) \cdot (\frac{\log k}{\log(n/m)}+1)} = k^{O(1)}.$$

When including $d = k^{O(1)}$ we will therefore have $ad = k^{O(1)}$ so the server will run in polynomial time. We observe for future use that at the same time

$$\binom{b}{a} \geq (\frac{b}{a})^a \geq \lceil \frac{n}{m} \rceil^{\frac{\log k}{\log(n/m)}} \geq \max(k, \frac{n}{m}) \geq k.$$

As we will see the above constraints on the parameters will be sufficient to get an optimal communication complexity. Note that in order to get perfect correctness, the client can check whether indeed all blocks need extraction of at most $2ad$ records. In the unlikely case this is not the case, the client can send the indices that it wants to extract in the clear. This latter protocol is obviously not private, but is only invoked with negligible probability. We have the following protocol construction:

1. Set $a = \lceil \frac{\log k}{\log(n/m)} \rceil$ and $b = a\lceil n/m \rceil$ and $d = \lceil \min(\frac{m}{a}, \frac{\alpha k/4}{a\ell + 2\log\binom{b}{a}}) \rceil$.
2. The client generates a seed $s \leftarrow \{0,1\}^k$ for the pseudorandom generator and checks that $\psi = \mathrm{PRG}(s)$ is a permutation of the indices so at most $2ad$ records need to be extracted from each block of size bd.
3. In the unlikely event ψ does place more than $2ad$ records to be extracted in the same block, the client sends $i_1, \ldots, i_m$ in clear to the server (encoded so it uses approximately $\log\binom{n}{m}$ bits of communication). The server responds with $(x_{i_1}, \ldots, x_{i_m})$, which the client outputs and halts.
4. The client sends s to the server and the server permutes the indices according to $\psi = \mathrm{PRG}(s)$.
5. The server divides the database into blocks of bd consecutive records and encodes each block as a database consisting of $d\binom{b}{a}$ records of length $a\ell$ such that each segment of $\binom{b}{a}$ records contains all possible choices of a ℓ-bit records from the corresponding segment of b records in the block.
6. The client and the server run the restrict multi-query CPIR protocol (C, D) on the $\lceil n/bd \rceil$ encoded blocks of $d\binom{b}{a}$ records to get $3d$ $(a\ell)$-bit strings. This corresponds to extracting the up to $2ad$ records from each of the original blocks.
7. The client decodes the output and reverses the permutation of the indices to get the output $(x_{i_1}, \ldots, x_{i_m})$.

First, the bound on the error probability given in Eq. 3 is negligible as it is bounded by $e^{-ad/3}$ and it holds that $ad = a \cdot \lceil \min(m/a, (\alpha k/4)/(a\ell + 2\log\binom{b}{a})) \rceil > k^{2/3}$ since $m > k^{2/3}$ and $\ell \leq \log(n/m)$ and $\log\binom{b}{a} = \log(k^{O(1)})$.

Regarding communication complexity, let us first compute it when

$$\frac{m}{a} > \frac{\alpha k/4}{a\ell + 2\log\binom{b}{a}}$$

so $d = \lceil (\alpha k/4)/(a\ell + 2\log\binom{b}{a})\rceil$. We send the pseudorandom seed of length k and run the CPIR protocol $\lceil n/bd \rceil$ times for a total communication of

$$\begin{aligned}(\lceil n/bd\rceil + 1)\,k <& \frac{nk}{bd} + 2k \leq nk/(b \cdot \frac{\alpha k/4}{a\ell + 2\log_2\binom{b}{a}}) + 2k\\ =&\frac{4n}{\alpha b}\cdot (a\ell + 2\log_2\binom{b}{a}) + 2k \leq \frac{20}{\alpha}\cdot\frac{na}{b}\cdot\log_2\frac{n}{m} + 2k\\ \leq&\frac{40}{\alpha}\cdot m\log_2\frac{n}{m} + 2k\ \ ,\end{aligned}$$

where we have used that $a\ell + 2\log\binom{b}{a} \leq a\log(n/m) + 2\log((eb/a)^a) \leq a\log(n/m) + 2a\log(e\lceil n/m\rceil) \leq 5a\log(n/m)$.

Next, we look at the case $d = \lceil m/a \rceil$. We have a communication complexity of

$$(\lceil n/bd\rceil + 1)\,k < \frac{nk}{bd} + 2k \leq \frac{nka}{bm} + 2k \leq 4k.$$

Also, in the rare cases where the client ends up sending the indices in the clear we have a communication complexity of $\log\binom{n}{m} + m\cdot\ell = O(m\cdot\log_2(n/m) + k)$.

Lemma 4. *The CPIR protocol for $n > 9m, m > k^{2/3}, \ell \leq \log_2(n/m)$ is correct and private. It has perfect correctness if the restricted multi-query CPIR protocol has perfect correctness.*

Proof. The protocol is perfectly correct because the restricted CPIR protocol is correct. We just need to verify that the restricted protocol can actually be applied, i.e., for sufficiently large k we have

$$3d \leq \frac{\alpha k}{a\ell + \log_2 d\cdot\binom{b}{a}}.$$

To see this holds, observe $d \leq k$ because

$$\alpha/4 + \frac{a\ell + 2\log_2\binom{b}{a}}{k} \leq 1$$

which follows from $\alpha/4 < 1$ and $a\ell + 2\log_2\binom{b}{a} = O(\log^2 k)$ (for sufficiently large k). From the choice of d in the protocol we now get

$$d \leq \left\lceil\frac{\alpha k/4}{a\ell + 2\log\binom{b}{a}}\right\rceil \leq \left\lceil\frac{\alpha k/4}{a\ell + \log_2 d\cdot\binom{b}{a}}\right\rceil < \frac{\alpha k/3}{a\ell + \log_2 d\cdot\binom{b}{a}}$$

where the second inequality follows from $d \leq k \leq \binom{b}{a}$ (for sufficiently large choice of k.)

With the choice of parameters we are guaranteed that the restricted multi-query CPIR of communication complexity k bits can be used on each block of size bd. An adversary with a probability of ϵ of breaking the privacy of the protocol can therefore be converted into an adversary that breaks the restricted multi-query CPIR with probability $\frac{\epsilon}{\lceil n/bd \rceil}$ except for the negligible probability that the privacy breach is due to a bad pseudorandom seed. Similarly, a hybrid argument shows that the multi-query protocol is correct. When the pseudorandom seed is bad, we step down to a non-private but perfectly correct CPIR. Therefore, if the restricted multi-query CPIR has perfect correctness, then we have perfect correctness of our CPIR. □

5.4 Multi-query CPIR for $\ell > \log_2(n/m)$

The final case is where $9m < n \wedge k^{2/3} < m \wedge \ell > \log_2(n/m)$. We split each database record into $\ell' := \lceil \ell / \lceil \log_2(n/m) \rceil \rceil$ records of length $\lceil \log_2(n/m) \rceil$ bits each. We now need to extract $\ell' \cdot m$ out of $\ell' \cdot n$ records of length $\lceil \log_2(n/m) \rceil$. Using the previous construction, we get a multi-query CPIR protocol that can do this with communication complexity $O\left(\ell' m \cdot \log_2\left(\frac{\ell' n}{\ell' m}\right) + k\right) = O(m\ell + k)$.

5.5 Summary: Communication-Optimal Multi-query CPIR

Combining the four protocols, we get a communication-optimal multi-query CPIR:

1. If $n \leq 9m$ send the entire database to the client
2. Else if $m \leq k^{2/3}$ use the CPIR protocol from Section 5.2 with communication complexity $O(m\ell + k)$
3. Else if $\ell \leq \log_2(n/m)$ use the CPIR protocol from Section 5.3 with communication complexity $O(m \cdot \log_2(n/m) + k)$
4. Else if $\ell > \log_2(n/m)$ use the CPIR protocol from Section 5.4 with communication complexity $O(m\ell + k)$

For sufficiently large k this protocol works for all choices of (m, n, ℓ). The communication complexity is $O(m\ell + m \cdot \log_2(n/m) + k)$, which is optimal up to a constant for perfectly correct CPIR. As a corollary to the lemmas in this section, we get the following:

Theorem 3. *The CPIR protocol given above is correct and private. It has perfect correctness if the restricted multi-query CPIR protocol has perfect correctness.*

Acknowledgments. Several reviewers have offered helpful comments on this paper. We would like in particular to thank an anonymous reviewer from ICALP 2009 for a long and insightful review.

The first author was supported by Engineering and Physical Sciences Research Council grant number EP/G013829/1. The second author performed the work at the University of Connecticut, Department of Computer Science and Engineering, partly supported by NSF grants 0447808,0831304,0831306. The third author was supported by Estonian Science Foundation grant #8058 and the European Union through the European Regional Development Fund.

References

1. Blömer, J., May, A.: A Tool Kit for Finding Small Roots of Bivariate Polynomials over the Integers. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 251–267. Springer, Heidelberg (2005)
2. Cachin, C., Micali, S., Stadler, M.: Computational Private Information Retrieval with Polylogarithmic Communication. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 402–414. Springer, Heidelberg (1999)
3. Chor, B., Goldreich, O., Kushilevitz, E., Sudan, M.: Private Information Retrieval. In: 36th Annual Symposium on Foundations of Computer Science, Milwaukee, Wisconsin, October 23–25, pp. 41–50. IEEE, Los Alamitos (1995)
4. Coppersmith, D.: Finding a Small Root of a Bivariate Integer Equation; Factoring with High Bits Known. In: Maurer [11], pp. 178–189
5. Coppersmith, D.: Finding a Small Root of a Univariate Modular Equation. In: Maurer [11], pp. 155–165
6. Gentry, C., Ramzan, Z.: Single-Database Private Information Retrieval with Constant Communication Rate. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 803–815. Springer, Heidelberg (2005)
7. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Batch codes and their applications. In: Proceedings of the Thirty-Fifth Annual ACM Symposium on the Theory of Computing, Chicago, IL, USA, June 13–16, pp. 262–271. ACM Press, New York (2004)
8. Kushilevitz, E., Nisan, N.: Communication Complexity. Cambridge University Press, Cambridge (1997)
9. Kushilevitz, E., Ostrovsky, R.: Replication is Not Needed: Single Database, Computationally-Private Information Retrieval. In: 38th Annual Symposium on Foundations of Computer Science, Miami Beach, Florida, October 20–22, pp. 364–373. IEEE Computer Society, Los Alamitos (1997)
10. Lipmaa, H.: An Oblivious Transfer Protocol with Log-Squared Communication. In: Zhou, J., López, J., Deng, R.H., Bao, F. (eds.) ISC 2005. LNCS, vol. 3650, pp. 314–328. Springer, Heidelberg (2005)
11. Maurer, U. (ed.): EUROCRYPT 1996. LNCS, vol. 1070. Springer, Heidelberg (1996)
12. Naor, M., Pinkas, B.: Oblivious Transfer and Polynomial Evaluation. In: Proceedings of the Thirty-First Annual ACM Symposium on the Theory of Computing, Atlanta, Georgia, USA, May 1–4, pp. 245–254. ACM Press, New York (1999)
13. Pohlig, S., Hellman, M.: An Improved Algorithm for Computing Logarithms over GF(p) and Its Cryptographic Significance. IEEE Transactions on Information Theory 24, 106–110 (1978)