A Simpler Rate-Optimal CPIR Protocol

Helger Lipmaa and Kateryna Pavlyk

University of Tartu, Tartu, Estonia

Abstract. In PETS 2015, Kiayias, Leonardos, Lipmaa, Pavlyk, and Tang proposed the first $(n, 1)$-CPIR protocol with rate $1 - o(1)$. They use advanced techniques from multivariable calculus (like the Newton-Puiseux algorithm) to establish optimal rate among a large family of different CPIR protocols. It is only natural to ask whether one can achieve similar rate but with a much simpler analysis. We propose parameters to the earlier $(n, 1)$-CPIR protocol of Lipmaa (ISC 2005), obtaining a CPIR protocol that is asymptotically almost as communication-efficient as the protocol of Kiayias *et al.* However, for many relevant parameter choices it is slightly more communication-efficient, due to the cumulative rounding errors present in the protocol of Kiayias *et al.* Importantly, the new CPIR protocol is simpler to understand, implement, and analyze.

Keywords: Communication complexity, computationally-private information retrieval, cryptographic protocols, optimal rate

1 Introduction

A computationally private information retrieval ($(n, 1)$-CPIR, [11]) protocol enables the receiver to obtain an ℓ-bit element from sender's database of n elements, without the sender getting to know which element was obtained. An efficient CPIR protocol has to be implemented by virtually any two-party privacy-preserving database application, and hence CPIR protocols have received significant attention in the literature.

Since there exists a trivial CPIR protocol with linear communication ℓn where the sender just forwards the whole database to the receiver, a major requirement in the design of new CPIR protocols is their communication efficiency. The first CPIR protocol with sublinear communication was proposed by Kushilevitz and Ostrovsky [11], and slightly optimized by Stern [16]. The first CPIR protocol with polylogarithmic-in-n communication was proposed by Cachin, Micali and Stadler [3]. The first CPIR protocols with *asymptotically* truly efficient communication complexity were proposed by Lipmaa [12,13] and Gentry and Ramzan [6].

All mentioned papers were concerned in the communication complexity as a function of n. However, optimizing the communication complexity of a CPIR protocol as a function of ℓ is also important, especially in applications where the database elements are very long, e.g., movies. Optimizing the rate — defined as the size of useful information ($\log n + \ell$ in the case of an $(n, 1)$-CPIR protocol)

divided by the actual communication complexity of the protocol — is also an interesting theoretical question. Indeed, achieving optimal rate (while still having acceptable computational complexity) is a central question in many areas of computer science and engineering.

The first constant-rate CPIR protocol was proposed by Gentry and Ramzan [6] (ICALP 2005, rate 1/4) and Lipmaa [12] (ISC 2005, rate 1/2). Lipmaa devised another variant of his protocol with optimized results; the resulting CPIR protocol from [13] had rate $1-1/a+o(1)$ for some positive constant $a > 1$. However, the drawback of the latter variant (see Sect. 3.3 for its full description) is an additive term $a\kappa \log_2^2 n$ in the communication complexity (here, κ is the security parameter), which means that the optimal value of a is actually quite small unless ℓ is very huge. Moreover, a cannot depend on ℓ (i.e., it has to be constant), and thus this CPIR protocol does not achieve rate $1 - o(1)$.

In a recent paper, Kiayias *et al.* [10] proposed a general parameterized family of so called leveled LBP-homomorphic encryption schemes with rate $1 - o(1)$. Here, LBP denotes the complexity class of functions implementable by polynomial-size (leveled) large-output branching programs, [17]. They then used the fact [8,13] that such an encryption scheme can be used to efficiently implement CPIR.

However, achieving optimal rate required the authors of [10] to perform extensive technical analysis. More precisely, following earlier papers like [11,12,13], the $(n, 1)$-CPIR protocol of Kiayias *et al.* is recursive. First, [10] constructs a (leveled) homomorphic encryption scheme that allows to compute an arbitrary function f by constructing a w-ary branching program (for some small $w \ll n$, e.g., $w = 2$) that computes f. Following [8], this homomorphic encryption scheme privately implements the $(w, 1)$ multiplexer function, needed in every internal node of a branching program, by using a simple $(w, 1)$-CPIR protocol that has minimal (i.e., rate $1 - o(1)$) *sender-side* communication. However, it has linear client-side (and hence, total) communication.

In addition, at every internal node, the $(n, 1)$-CPIR protocol of [10] applies a precisely defined operation of splitting and concatenating, that guarantees that at the level d of the branching program, the $(w, 1)$-CPIR protocol operates with database elements of length $s_d\kappa$, where s_d is a parameter to be optimized. More precisely, the outputs of the CPIR protocol from level $d-1$ are cut into some t_d pieces of length $s_d\kappa$. By using this recursive construction, a suitable $(w, 1)$-CPIR protocol can be used to securely implement any function from LBP.

Kiayias *et al.* [10] showed, by using an intricate analysis, that the optimal communication is achieved when $s_1 = \ldots = s_m =: s$, where m is the length of the branching program. In a nutshell, they used multivariable calculus to show that the communication complexity of their CPIR protocol is optimized when s is equal to a root of a certain degree-$(m + 1)$ polynomial f_m. Then, they used Galois theory to show that f_m cannot be solved in radicals. Finally, they used the theory of Newton-Puiseux series to numerically compute an approximation of the optimal s. As the end result, they obtained a CPIR protocol of rate $1 - 1.72\sqrt{\kappa/\ell} \log_2 n + O(\ell^{-1})$.

Hence, the analysis used in [10] is (very) complicated, resulting in (a) a CPIR protocol with a complex description, and (b) an optimal parameter choice that, while it can be done efficiently, seems to be difficult to analyze. For example, the optimal value of s in [10] is given by a series. After that, [10] proves that given the so computed s, the communication complexity will be given by another explicit series. However, in practice one needs to compute an integer approximation of s efficiently. While [10] proposed an efficient algorithm for computing such an approximation, it is unclear how this will influence the precise value of the communication complexity in the general case.

Moreover, one problem of their scheme is due to "rounding errors". First, the claimed rate corresponds to the case when s is a real root while in practice s must be an integer. To deal with this requirement, Kiayias *et al.* presented an $O(\log\log n)$-time algorithm to compute an integer approximation of s. Second, recall that each $(w,1)$-CPIR protocol at every layer in [10] requires plaintexts of the same length $s\kappa$. However, in the optimal construction of [10], there is no guarantee that the total output length of the previous layer divides by s and hence at every layer one has to round up the length of each plaintext. This means that at every layer, there will be some undue increase in the number of applied $(w,1)$-CPIR protocols, which increases the actual communication complexity of the resulting $(n,1)$-CPIR protocol.

The authors of [10] did not compute precise upper bounds on the communication of their CPIR protocol after s is rounded to an integer and one adds up the rounding errors. Instead, [10] provided empirical data (see Sect. 7.1.1 in [10], or Fig. 1 in the current paper) that the increase in communication is insignificant when ℓ is large, at least for some practically relevant values of ℓ and n.

Our Contribution. We show how to achieve *almost* the same communication complexity and rate as in the protocol of Kiayias *et al.* [10]. We provide precise analysis and comparison in Sect. 5, where we show that the difference between the communication of the "ideal" CPIR protocol of [10] (that does not take into account rounding errors) and the new CPIR protocol is $O(\ell^{1/2})$. After taking into account the rounding errors, the new protocol will be *slightly* more communication-efficient for all values of ℓ and n analysed in [10]. (See Fig. 1.)

We use the CPIR protocol proposed by Lipmaa in ISC 2005 [12] and ICISC 2009 [13] but with parameters that we optimize in the current paper. In particular, we consider general w-ary decision trees instead of just binary contrary to [12,13]. Alternatively, the proposed protocol is an instantiation of the the CPIR protocol family of Kiayias *et al.* [10] but with different parameter set, namely, with the values t_d being constant, $t_1 = \cdots = t_m =: t$, and the values s_d being slightly increasing. This means that the new CPIR protocol can be seen as a t-times parallel implementation — each for $\lceil \ell/t \rceil$-bit databases — of the CPIR protocol from [12], for an optimized value of t. The new analysis is significantly simpler than the multi-page analysis of [10] but surprisingly enough delivers almost the same results. (Intuitively, this happens since in [10], in different layers

ℓ/κ	Communication				
	No privacy	Kiayas *et al.* [10]		This work	
		Theoretical	With rounding	Theoretical	With rounding
10^3	2 048 017	4 079 561	4 220 928	4 090 880	4 090 880
10^4	20 480 017	26 439 497	26 759 168	26 443 776	26 443 776
10^5	204 800 017	223 161 724	223 942 656	223 163 148	223 163 343
10^6	2 048 000 017	2 105 572 921	2 107 731 968	2 105 573 376	2 105 573 376
10^7	20 480 000 017	20 661 566 883	20 664 602 624	20 661 567 027	20 661 569 161
10^8	204 800 000 017	205 373 669 331	205 394 259 968	205 373 669 376	205 373 669 376

Fig. 1. Comparison with [10], for $\kappa = 2048$, $w = 5$, $n = 5^7$. The protocol from [10] offers better communication if rounding is not taken into account. However, in all cases, the current work offers better communication in practice (i.e., when parameters have been rounded correctly)

one uses parameters $(s, s, s, \dots)$ while in the new protocol, one uses parameters $(s, s+1, s+2, \dots)$. Since ℓ and s both are considered to be large, $s + 1 \approx s$.)

To show that our analysis is really simple, we will very briefly outline it next. The communication function of the w-ary generalization of the CPIR from [12] depends on n (the size of the database), ℓ (the length of database elements), κ (the security parameter), t (the parallelism factor) and w (the arity of the decision tree). Here, t and w are the values to be optimized. First, we use simple univariate analysis to derive the optimal value $t_{opt} = \sqrt{(w-1)\ell/\kappa}$ of t for any w. Given the value of t_{opt}, we then "near optimize" (see Sect. 4) the value of w. Here, near optimizing means that we write the communication function as a series in ℓ, and then choose the *integer* value of w (namely, $w = 5$) that minimizes the *most significant* coefficients of this series. Since t_{opt} is a function of ℓ, the layout of the series crucially depends on the fact that we first fix t_{opt}.

We show that under these values of t and w, the *asymptotic* communication of the resulting CPIR protocol is practically the same as in the optimal case in [10]. On the other hand, for *interesting*[1] values of ℓ, the proposed variant will have slightly better communication. More precisely, in the new CPIR protocol, the communication complexity function, written down as a series in ℓ coincides with the one of the CPIR from [10] in the first three terms. The communication complexity of the optimal CPIR of [10] has a tailing element $O_\ell(1/\ell)$ that makes their construction asymptotically slightly more efficient. However, the difference is not big: for example, in a concrete case where the database elements are $10^6\kappa$ bits long and the database has $n = 5^7$ elements (here, $\kappa = 2048$ is the currently recommended security parameter), the CPIR of [10] is — when ignoring rounding

[1] Here, by interesting we mean values of ℓ that correspond to the length of an audio or video file; this was also the motivating example given in [10]. If ℓ is much shorter, then optimizing the communication complexity as a function of ℓ is not relevant.

errors — more efficient than the new CPIR by 683 bytes out of more than 3 billion. See Fig. 1 for more examples.

However, this comparison is purely theoretical since it operates with the "ideal" communication function and does not take into account rounding errors. Compared to [10], we do not run into rounding errors at *every* layer of the construction. Intuitively, this is the case since in our construction, each ciphertext of the previous layer is considered to be the plaintext of the next layer and hence the length of the plaintexts increases by κ bits at each layer. On the other hand, in [10], at each layer, the concatenation of t ciphertexts (of total length $(s+1)t_d\kappa$) is divided into new plaintexts, each of length $s\kappa$. The rounding error (at every layer) is caused by the fact that for an s that is chosen optimally by the analysis of [10], $(s+1)t_d\kappa$ is essentially never divisible by s.

In fact, in the new construction, it is only important that $s \mid \ell$ (or else we get a *one-time* rounding error at the very bottom of the protocol construction). This means, as we show numerically, that in practice, the new CPIR protocol achieves slightly better communication complexity than the CPIR of [10], while being much simpler. See Fig. 1 for a communication efficiency comparison. To demonstrate the (relative) simplicity of the new construction, we will give a full description of the new CPIR protocol on Fig. 3; the only important distinction from the well-known CPIR protocol of [12], as modified by [13], is in the first line (the choice of the paramrters). A comparable full description of the CPIR protocol of [10] is significantly longer, albeit mostly due to the more complicated procedure for selecting optimal parameters. In fact, [10] does *not* give a self-contained description of their CPIR protocol. Fig. 3 in [10] describes their new LHE scheme (that then has to be modified to become a CPIR protocol), but the choice of all parameters is described later in that paper, together with the issues rising from rounding the parameters.

Extensions And Applications. Based on the ideas of [8,10] and of the current paper, one can construct a rate $1 - o(1)$ homomorphic encryption scheme that computes any function that has a polynomial-size large-output branching program. All known fully homomorphic encryption schemes have a very low rate. (See [7] for insights on why achieving good rate fully homomorphic encryption scheme might be difficult.) Since the generalization from binary decision trees, that are used to construct the new CPIR protocol, to arbitrary branching programs is straightforward yet necessitates introducing a lot of branching program-related terminology, we will omit further discussion and refer to [10].

Similarly, one can build a rate $1-o(1)$ oblivious transfer, given the new CPIR protocol and known transformations, see [10] for discussion. Finally, based on their CPIR protocol, [10] proposed a new rate $1-o(1)$ strong conditional oblivious transfer protocol [1], and based on the later, [9] constructed the first optimal rate asymmetric fingerprinting protocol. One can plug in the CPIR protocol of the current paper to those constructions obtaining simpler yet slightly more communication-efficient protocols for (strong conditional) oblivious transfer and asymmetric fingerprinting.

2 Preliminaries

Notation. For a predicate, let $[P(x)] \in \{0,1\}$ denote the truth value of $P(x)$, e.g., $[x = y]$ is equal to 1 iff $x = y$ and to 0 otherwise. The Lambert's W function is defined by the equation $z = W(z)e^{W(z)}$. Asymptotically, $W(z) \approx \ln z - \ln\ln z$. Let κ be the security parameter; in our case it corresponds to the key length in bits, so $\kappa \geq 2048$.

Public-Key Cryptosystem. A length-flexible cryptosystem $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ [4,5] consists of three efficient algorithms, Gen for key generation, Enc for encryption, and Dec for decryption. The public key pk fixes the plaintext space, the randomizer space $\mathfrak{R}_{\mathsf{pk}}$, and the ciphertext space. For a public key pk, plaintext m (of bitlength $\ell = |m|$), a positive integer length parameter $s := \lceil \ell/\kappa \rceil$, and a randomizer $r \in \mathfrak{R}_{\mathsf{pk}}$ we have $c = \mathsf{Enc}^s_{\mathsf{pk}}(m; r)$ and $m = \mathsf{Dec}^s_{\mathsf{sk}}(c)$, and it is required that $\mathsf{Dec}^s_{\mathsf{sk}}(\mathsf{Enc}^s_{\mathsf{pk}}(m; r)) = m$.

A length-flexible cryptosystem has to satisfy the usual IND-CPA security requirement [4]. That is, no efficient adversary should be able to distinguish between ciphertexts corresponding to m_0 and m_1 encrypted by using the same integer length parameter, even if m_0 and m_1 were chosen by her.

Let the *rate* of the cryptosystem be $|m|/|c|$, i.e., the ratio between the number of useful bits and the actual transmission length. A length-flexible cryptosystem is *optimal rate* if $|m|/|c| = 1 - o(1)$ when $|m|$ increases.

A cryptosystem is *additively homomorphic* if $\mathsf{Dec}^s_{\mathsf{sk}}(\mathsf{Enc}^s_{\mathsf{pk}}(m_1; r_1) \cdot \mathsf{Enc}^s_{\mathsf{pk}}(m_2; r_2)) = m_1 + m_2$. In [4,5], Damgård and Jurik constructed two IND-CPA secure optimal-rate length-flexible additively homomorphic cryptosystems. See also [2]. An additively homomorphic cryptosystem is also required to be *rerandomizable* in the sense that $\mathsf{Enc}^s_{\mathsf{pk}}(m; r) \cdot \mathsf{Enc}^s_{\mathsf{pk}}(0; r'_1)$ is computationally indistinguishable from $\mathsf{Enc}^s_{\mathsf{pk}}(0; r'_2)$, for uniformly random $r'_1, r'_2 \leftarrow_r \mathfrak{R}_{\mathsf{pk}}$.

More precisely, in the cryptosystem of [4], the public key is a well-chosen RSA modulus $N = pq$, the secret key is (p, q), and for a positive integer s, $\mathsf{Enc}^s_{\mathsf{pk}}(m; r) = (1+N)^m r^{N^s} \mod N^{s+1}$, for $m \in \mathbb{Z}_{N^s}$ and $r \in \mathbb{Z}^*_N$. Hence, if the plaintext is of length $s\kappa$, the cryptosystem of [4] has ciphertext of length $(s+1)\kappa$. The rate of this cryptosystem is

$$\frac{\ell}{\ell + \kappa} = 1 - \frac{\kappa}{\ell} + \frac{\kappa^2}{\ell^2} + O_\ell(\ell^{-3}) \ .$$

This is intuitively optimal (up to the choice of κ) since κ bits are needed to randomize the ciphertext. The Damgård-Jurik cryptosystem from [4] is IND-CPA secure under the DCR assumption [15].

If pk and r are understood in the context (or if their precise value is not relevant), we will not write them down explicitly.

Computationally-Private Information Retrieval (CPIR). Assume $n > 1$ and ℓ are positive integers, with $n, \ell = poly(\kappa)$. An $(n, 1)$-CPIR protocol [11] for

ℓ-bit strings allows the receiver on input $x \in \{0,\ldots,n-1\}$ to obtain $f_x \in \{0,1\}^\ell$ out of the sender's database $\boldsymbol{f} = (f_0,\ldots,f_{n-1})$ without the sender getting any information about x.

In a *two-message CPIR protocol*, the receiver first generates a public and secret key pair $(\mathsf{pk},\mathsf{sk})$, then sends a query $Q \leftarrow \mathsf{Query}_{\mathsf{pk}}(n,\ell;x)$ and pk to the sender, who answers with a reply $R \leftarrow \mathsf{Reply}_{\mathsf{pk}}(n,\ell;\boldsymbol{f},Q)$. After that, the receiver uses a function $\mathsf{Answer}_{\mathsf{sk}}(n,\ell;x,R)$ to recover f_x.

The receiver's communication is equal to $|Q|$, the sender's communication is equal to $|R|$, and the *total communication* is equal to $\mathsf{com} := |Q| + |R|$. A non-private CPIR protocol consists of two messages, $Q = x$ (of $\log_2 n$ bits) from the receiver to the sender, and $R = f_x$ (of ℓ bits) from the sender to the receiver. We do not count pk as part of the communication, since (a) it is short, and (b) it can —and will — be reused between many instances of the CPIR protocol. The *rate* of a CPIR protocol is equal to $(\log_2 n + \ell)/\mathsf{com}$.

A two-message CPIR protocol is *IND-CPA secure* if no efficient adversary $\mathcal{A}$ can distinguish between queries corresponding to x_0 and x_1, even if x_0 and x_1 were chosen by her. That is,

$$\Pr\left[\begin{array}{l}(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^\kappa), (x_0,x_1) \leftarrow \mathcal{A}_{\mathsf{pk}}(1^\kappa,n,\ell), b \leftarrow_r \{0,1\}, \\ Q \leftarrow \mathsf{Query}_{\mathsf{pk}}(n,\ell;x_b) : \mathcal{A}_{\mathsf{pk}}(n,\ell;Q) = b\end{array}\right]$$

is negligible in κ, for each probabilistic polynomial-time $\mathcal{A}$ and polynomially large n and ℓ.

3 Related Work

There are very few conceptually different approaches for constructing communication-efficient $(n,1)$-CPIR protocols. The $(n,1)$-CPIR protocol by Kiyaias et al. [10], following earlier protocols [11,16,12,8,13], homomorphically executes a branching program, by using a $(w,1)$-CPIR at every internal node of the branching program. Here, w is a small constant. See [3,6] for a different approach that however results in rate that cannot be better than 1/4; see [3,6] for a discussion.

3.1 Linear-Communication $(w,1)$-CPIR Protocol

Recall that s is a positive integer. The concrete underlying $(w,1)$-CPIR protocol used in [12,8,13,10] is a simple linear-communication CPIR protocol from [12][2]. To transfer one $\ell = s\kappa$-bit database element, the receiver sends to the sender $w-1$ ciphertexts, and the sender responds with one ciphertext, where the length of each ciphertext is $(s+1)\kappa$ bits. More precisely, the receiver sends to the sender $w-1$ ciphertexts C_i encrypting $[x=i]$ for $i \in \{0,\ldots,w-2\}$, $C_i = \mathsf{Enc}^s([x=$

[2] As shown in [14], linear communication is the best one can hope when building a CPIR protocol on top of an additively homomorphic cryptosystem while *not* using recursion.

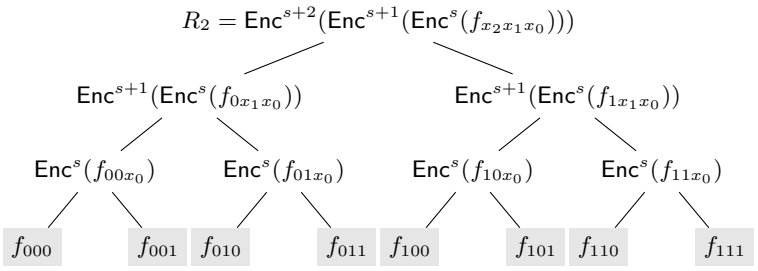

Fig. 2. Using Lipmaa's $(w, 1)$-CPIR from [12] with $w = 2$ and $n = 8$. The receiver sends $\mathsf{Enc}^{s}(x_0)$, $\mathsf{Enc}^{s+1}(x_1)$, $\mathsf{Enc}^{s+2}(x_2)$ to the sender. The sender computes recursively the values at intermediate nodes, and then replies with R_2.

$i]; r_i)$ for a random $r_i \leftarrow_r \mathfrak{R}_{\mathsf{pk}}$. From $\{C_i\}_{i=0}^{w-2}$, by using additive homomorphism, the sender obtains the ciphertext C_{w-1} encrypting $[x = w-1] = 1 - \sum_{i=0}^{w-2}[x = i]$. Hence, $(C_0, \ldots, C_{w-1})$ encrypts the x-th unit vector, $x \in \{0, \ldots, w-1\}$. Then, she uses $\{C_i\}_{i=0}^{w-1}$ to homomorphically compute a randomized ciphertext encrypting $\sum_{i=1}^{n}[x = i]f_i = f_x$. That is, $Q = \mathsf{Query}_{\mathsf{pk}}(n, \ell; x) = (C_0, \ldots, C_{w-2})$, $C_{w-1} = \mathsf{Enc}^{s}(1; 0)/\prod_{i=0}^{w-2} C_i$, and $R = \mathsf{Reply}_{\mathsf{pk}}(n, \ell; \boldsymbol{f}, Q) = \prod_{i=0}^{w-1} C_i^{f_i} \cdot \mathsf{Enc}^{s}(0; r)$ for a random r. The receiver just computes $\mathsf{Answer}_{\mathsf{sk}}(n, \ell; x, R) = \mathsf{Dec}^{s}_{\mathsf{sk}}(R)$. This CPIR protocol is IND-CPA secure given that the underlying Damgård-Jurik cryptosystem is IND-CPA secure, i.e., under the DCR assumption.

While this $(w, 1)$-CPIR has linear communication, importantly its sender-side communication consists of only one ciphertext and thus has near-optimal rate $(\log_2 n + \ell)/(\ell + \kappa) = 1 - (\kappa - \log_2 n)/\ell + O(\ell^{-2}) = 1 - o(1)$.

3.2 Lipmaa's Recursive $(n, 1)$-CPIR Protocol from [12]

W.l.o.g., assume that n is a power of w, $n = w^m$ for some m, where w is a small positive integer. (In the general case, one can add dummy elements to the database.) The $(n, 1)$-CPIR protocols of [11,12,13,10] are built on top of a $(w, 1)$-CPIR, $w \ll n$, in a recursive manner.

Let $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be an optimal-rate length-flexible additively homomorphic cryptosystem like the one proposed by Damgård and Jurik [4] and $(\mathsf{Query}, \mathsf{Reply}, \mathsf{Answer})$ be the $(w, 1)$-CPIR protocol of Sect. 3.1. In the $(n, 1)$-CPIR protocol of Lipmaa from ISC 2005 [12], a w-ary decision tree of length $m := \log_w n$ is constructed on top of a database of n elements. Then, the internal nodes are assigned labels starting from bottom. Let $x = \sum_{i=0}^{m-1} x_i w^i$, i.e., x_i is the ith w-ary digit of x. For an internal node v that has distance i to the leafs, the label of v is equal to the reply of the $(w, 1)$-CPIR protocol, given a query $\mathsf{Query}(w, s\kappa; x_i)$ and a database $(f_0, \ldots, f_{w-1})$ consisting of the labels of the children of v. (See Fig. 2.) Finally, the sender replies with the label of the root of the binary decision tree, and the receiver applies to it m times the Answer function to recover f_x.

Since we use the $(w,1)$-CPIR protocol of Sect. 3.1, if the labels of the children of v are say $(f_{v0},\ldots,f_{v1})$, then the label of v is going to be $\mathsf{Enc}_{\mathsf{pk}}^{s+i-1}(f_{vx_i})$ (as in Fig. 2), and each application of Answer consists of a single decryption.

The receiver's message in the $(n,1)$-CPIR protocol corresponds to one $(w,1)$-CPIR receiver's message for each length parameter $s+i$, $i\in\{1,\ldots,\log_w n\}$, while the sender's message corresponds to one $(w,1)$-CPIR sender's message for the length parameter $s+\log_w n$. The resulting receiver's communication is

$$\begin{aligned}\mathsf{rec}_1(w,n,\ell,\kappa) :=& \sum_{i=1}^{\log_w n}(w-1)(\ell/\kappa+i)\kappa\\ =&(w-1)(\ell/\kappa+(\log_w n+1)/2)\log_w n\cdot\kappa\\ =&(w-1)(\ell+(\log_w n+1)\kappa/2)\log_w n\end{aligned}$$

and the sender's communication is

$$\mathsf{sen}_1(w,n,\ell,\kappa) := (\ell/\kappa+\log_w n)\kappa = \ell+\kappa\log_w n\ .$$

(Recall that communication is always measured in bits.) Hence, the total communication $\mathsf{com}_1(w,n,\ell,\kappa)=\mathsf{rec}_1(w,n,\ell,\kappa)+\mathsf{sen}_1(w,n,\ell,\kappa)$ of the CPIR protocol from [12] is equal to

$$\mathsf{com}_1(w,n,\ell,\kappa) = ((w-1)\log_w n+1)\ell+\frac{\kappa\log_w n\cdot((w-1)\log_w n+(w+1))}{2}\ .$$

Its rate is $(\log_2 n+\ell)/\mathsf{com}_1(w,n,\ell,\kappa)\approx 1/((w-1)\log_w n+1)$. For large ℓ, $\mathsf{com}_1(\cdot,n,\ell,\kappa)$ is clearly minimal when $w=2$, with

$$\mathsf{com}_1(2,n,\ell,\kappa) = (\log_2 n+1)\ell+\frac{\kappa\log_2 n\cdot(\log_2 n+3)}{2}$$

and rate $\approx 1/(\log_2 n+1)$.

3.3 Optimizing the Communication by Data-Parallelization

In [12], Lipmaa additionally noted that one can reduce the communication (assuming $\ell/\kappa\gg\log_2 n$) by executing the protocol from Sect. 3.2 separately and in parallel on every (ℓ/t)-bit chunk of the database elements, where $t\geq 1$, $t\mid\ell$, is a positive integer. This results in optimized total communication since in the $(n,1)$-CPIR protocol of Sect. 3.2, the receiver's communication is much larger than the sender's communication. If $t>1$, then the same receiver's message can be used in all t parallel invocations of the protocol from Sect. 3.2, while the sender has to respond with t messages. Crucially, the bitlength of database elements in each invocation is divided by t and thus every single message of the receiver and the sender becomes shorter.

More precisely, assuming again $t\mid\ell$, the parallelized $(n,1)$-CPIR protocol of [12] has the receiver's communication, the sender's communication, and the total communication

$$\begin{aligned}\mathsf{rec}_2(w,n,\ell,\kappa,t) :=&\mathsf{rec}_1(w,n,\ell/t,\kappa) = (w-1)(\ell/t+(\log_w n+1)\kappa/2)\log_w n\ ,\\ \mathsf{sen}_2(w,n,\ell,\kappa,t) :=&t\cdot\mathsf{sen}_1(w,n,\ell/t,\kappa) = t(\ell/t+\kappa\log_w n) = \ell+t\kappa\log_w n\ ,\end{aligned}$$

$$\mathsf{com}_2(w,n,\ell,\kappa,t) = (w-1)(\ell/t + (\log_w n + 1)\kappa/2)\log_w n + \ell + t\kappa\log_w n \ . \quad (1)$$

If $t \nmid \ell$, then one has to round ℓ/t upwards.

In ISC 2005 [12], Lipmaa considered parameter settings that resulted in rate $\approx 1/2$. In ICISC 2009 [13], Lipmaa considered the following parameter settings: $w = 2$ and $t = a\log_2 n$ for large a. In this case,

$$\mathsf{com}_2(2,n,\ell,\kappa,a\log_2 n) = \left(\frac{1}{a}+1\right)\ell + \frac{(2a+1)\kappa\log_2^2 n}{2} + \frac{\kappa\log_2 n}{2} \ . \quad (2)$$

Thus with such parameters the parallelized $(n,1)$-CPIR protocol has rate

$$\frac{\log_2 n + \ell}{\mathsf{com}_2(2,n,\ell,\kappa,a\log_2 n)} = \frac{a}{a+1} + O(\ell^{-2}) \le 1 - \frac{1}{a} + O(\ell^{-2}) \ .$$

However, for this estimate to hold, it is needed that $a = \Theta_\ell(1)$ does not depend on ℓ. Moreover, due to the additive term $\Theta(a)\kappa\log_2^2 n$ in Eq. (2), the communication complexity will actually increase if a is too large. Hence, by using the parameters proposed in [13], the parallelized $(n,1)$-CPIR protocol from [12] cannot achieve rate $1 - o(1)$.

3.4 The CPIR Protocol of Kiayias *et al.*

Kiayas *et al.* [10] proposed another twist on top of the CPIR protocol of Lipmaa [12]. In a nutshell, during the recursive procedure, the parallelized CPIR protocol of Sect. 3.3 stores at every childrens' node the concatenation of t plaintexts. The label of the parent node is defined to be equal to the concatenation of t individual ciphertexts. In [10], each childrens' node also stores the concatenation of t plaintexts each being (say) L bits long. However, this concatenation is then redivided into t' equal-length new plaintexts (each of length $\lceil tL/t' \rceil$). The new plaintexts are then encrypted individually and the resulting ciphertexts concatenated as the label of the parent node. The major contribution in [10] is the computation of optimal values t and t' (for each layer of the CPIR tree) and establishing that one can choose those values so as to obtain a CPIR protocol of rate $1 - o(1)$.

4 Simple Optimal-Rate CPIR Protocol

We now propose a different setting of the parameters for the parallelized $(n,1)$-CPIR protocol from Sect. 3.3, motivated by the approach of [10]. We first continue the analysis of [12,13], and find optimal values of the parameters. After that, for the sake of completeness, we will give a full description of the resulting CPIR protocol together with a security proof.

4.1 Optimization of Parameters

Recall that the communication complexity of Lipmaa's parallelized $(n, 1)$-CPIR protocol is given by Eq. (1). It depends on three variables (κ, ℓ, and n) that are fixed, and two variables (w and t) that can be optimized. We were unable to find the global optimum of com_2, due to the complicated form of $\partial\mathsf{com}_2/\partial w$,

$$\frac{\partial\mathsf{com}_2}{\partial w} = \frac{\ln n \cdot \ln w \cdot (w \ln w(2\ell + kt) - 2\ell(w-1) - kt(2t + w - 1))}{2tw \ln^3 w} + \frac{\ln^2 n \cdot kt(-2w + w \ln w + 2)}{2tw \ln^3 w} \ .$$

Instead, we will first optimize com_2 as a function of t, and then we will "near optimize" the result as a function of w. By doing so, we obtain a CPIR protocol that has a rate very close to the rate of [10], but with a much simpler analysis.

We will find the optimal value of t by requiring that

$$\frac{\partial\mathsf{com}_2}{\partial t} = \frac{(t^2\kappa - (w-1)\ell)\log_w n}{t^2} = 0 \ .$$

Since $n \neq 0$, this holds if

$$t = t_{opt} := \sqrt{(w-1)\ell/\kappa} \ .$$

Clearly,

$$\mathsf{com}_2(w,n,\ell,\kappa,t_{opt}) = \ell + \frac{2\sqrt{w-1}}{\log_2 w} \cdot \sqrt{\ell\kappa} \cdot \log_2 n + \frac{(w-1)(\log_w n + 1)\log_w n}{2} \cdot \kappa \ . \tag{3}$$

Finding a value of w that optimizes this function seems to be also complicated. Hence, as in [10], we now choose w that just minimizes the most significant term in com_2 that depends on w, i.e., the second term, hoping that the result w will be close to the optimal. The second additive term in the right hand side of Eq. (3) is minimized when

$$\frac{d}{dw}\frac{\sqrt{w-1}}{\log_2 w} = \frac{(w \ln w - 2w + 2)\ln 2}{2\sqrt{w-1} \cdot w \ln^2 w} = 0 \ ,$$

that is, when

$$w = -\frac{2}{W(-2/e^2)} \approx 4.92 \ . \tag{4}$$

Since w has to be an integer, we take $w = 5$, exactly as in [10]. Then, $t_{opt} = 2\sqrt{\ell/\kappa}$. Thus, recalling that $\ell = t \cdot s\kappa$, we get that

$$s = \frac{\ell}{t_{opt}\kappa} = \frac{\ell}{2\sqrt{\ell/\kappa} \cdot \kappa} = \frac{1}{2} \cdot \sqrt{\ell/\kappa} \ .$$

Parameters: $\kappa, n, \ell, t = \lceil 2\sqrt{\ell/\kappa} \rceil, s = \lceil \ell/(t\kappa) \rceil, w = 5, m = \lceil \log_w n \rceil$.
Receiver's $\mathsf{Query}^{new}(n, \ell; x)$**:**
Generate a new public and secret key pair $(\mathsf{pk}, \mathsf{sk})$ for the Damgård-Jurik cryptosystem.
Write $x = \sum_{d=0}^{m-1} x_d w^d$ for $x_d \in \{0, \ldots, w-1\}$.
For $d = 0$ to $m-1$:
1. For $j = 0$ to $w-2$:
   (a) Generate a new randomizer $r_{dj} \leftarrow \mathfrak{R}_{\mathsf{pk}}$
   (b) Let $Q_{dj} \leftarrow \mathsf{Enc}_{\mathsf{pk}}^{s+d-1}([x_d = j]; r_{dj})$
2. Compute $Q_{d,w-1} \leftarrow \mathsf{Enc}_{\mathsf{pk}}^{s+d-1}(1;1) / \prod_{j=0}^{w-2} Q_{dj}$

Send pk and $\mathsf{Query}_{\mathsf{pk}}(n, \ell, x) := \boldsymbol{Q} = (Q_{dj})_{d \in [0,m-1], j \in [0,w-2]}$ to the sender
Sender's $\mathsf{Reply}_{\mathsf{pk}}^{new}(n, \ell; \boldsymbol{f}, \boldsymbol{Q})$:
For $i - 0$ to $n-1$:
1. Denote $L_{0,i} = f_i$
2. Write $L_{0,i} = (L_{0,i,0}, \ldots, L_{0,i,t-1})$, with $|L_{0,i,z}| = s\kappa$

For $d = 0$ to $m-1$:
1. Compute $Q_{d,w-1} \leftarrow \mathsf{Enc}_{\mathsf{pk}}^{s+d-1}(1;1) / \prod_{j=0}^{w-2} Q_{dj}$
2. For $i = 0$ to $n/w^{d+1} - 1$:
   (a) For $z = 0$ to $t-1$:
      i. $L_{d+1,i,z} = \mathsf{Enc}_{\mathsf{pk}}^{s+d-1}(0; r'_{diz}) \cdot \prod_{j=0}^{w-1} Q_{dj}^{L_{d,iw+j,z}}$ for random $r'_{diz} \leftarrow \mathfrak{R}_{\mathsf{pk}}$

Let $\boldsymbol{R} = (R_0, \ldots, R_{t-1}) := (L_{m,0,0}, \ldots, L_{m,0,t-1})$.
Return $\mathsf{Reply}_{\mathsf{pk}}(n, \ell; \boldsymbol{f}, \boldsymbol{Q}) = \boldsymbol{R}$.
Receiver's $\mathsf{Answer}_{\mathsf{sk}}^{new}(n, \ell; \boldsymbol{R})$:
For $d = m-1$ downto 0:
1. For $z = 0$ to $t-1$: $R_z \leftarrow \mathsf{Dec}_{\mathsf{sk}}^{s+d}(R_z)$

Return $f_x = (R_0, \ldots, R_{t-1})$

Fig. 3. Full description of the new $(n, 1)$-CPIR protocol

4.2 Full Protocol

Before giving a full efficiency analysis (it will be done in Sect. 5), we now take a step back and give a detailed description of the resulting $(n, 1)$-CPIR protocol. In the description below we do not assume that (say) n is a power of w, hence we will use the $\lceil \cdot \rceil$ function to compute intermediate parameters. See Fig. 3 for a full description. We emphasize that — except the different choice of parameters — this is the same protocol as described in Sect. 3.3 and hence we omit repeating the intuition.

4.3 Security Proof

Lemma 1. *Assume that the underlying public-key cryptosystem is IND-CPA secure. Then, the new CPIR protocol is IND-CPA secure.*

Proof (Sketch). The sender, not having access to the secret key, only sees a vector of ciphertexts $(Q_{00}, \ldots, Q_{m-1,w-2})$. Hence, the security of the CPIR protocol is guaranteed by the IND-CPA security of the cryptosystem via a standard hybrid argument. □

5 Communication Efficiency Analysis

5.1 Asymptotic Analysis

The given parameter choice results in the following theorem.

Theorem 1. *Assume that* $s = \sqrt{\ell/\kappa}/2$ *and* $\log_1 n$ *are integers. There exists an* $(n, 1)$*-CPIR protocol for* ℓ*-bit strings with communication complexity*

$$\mathsf{com}_2(5, n, \ell, \kappa, 2\sqrt{\ell/\kappa}) = \ell + \frac{4}{\log_2 5} \cdot \sqrt{\ell\kappa} \cdot \log_2 n + 2\left(\log_1^2 n + \log_1 n\right)\kappa \ .$$

Proof. The result follows from preceding discussion. □

Note that $4/\log_2 5 \approx 1.72$. Note also that

$$\begin{aligned}\mathsf{rec}_2(5, n, \ell, \kappa, 2\sqrt{\ell/\kappa}) &= \frac{2}{\log_2 5} \cdot \sqrt{\ell\kappa} \cdot \log_2 n + 2\left(\log_1^2 n + \log_1 n\right)\kappa \ , \\ \mathsf{sen}_2(5, n, \ell, \kappa, 2\sqrt{\ell/\kappa}) &= \ell + \frac{2}{\log_2 5} \cdot \sqrt{\ell\kappa} \cdot \log_2 n \ ,\end{aligned}$$

and hence rec_2 is sublinear in ℓ.

To compare, the $(n, 1)$-CPIR protocol of [10] (see Cor. 1 therein) achieves communication complexity

$$\ell + \frac{4}{\log_2 5} \cdot \sqrt{\ell\kappa} \cdot \log_2 n + 2\left(\log_1^2 n + \log_1 n\right)\kappa + O(\ell^{-1/2}) \ .$$

Thus, the $(n, 1)$-CPIR protocol from the current paper has *essentially* the same communication as in [10] (the first three terms of the series expansion of the communication function com are the same as in [10]), but with a much simpler analysis (and construction).

5.2 Optimization w.r.t. n

Consider now the task of optimization com_2 (as in Eq. (1)) as a function of n.

First, finding of the optimal t_{opt} does not depend on whether we optimize as a function of ℓ or n. Hence, we will assume that $t_{opt} = \sqrt{(w-1)\ell/\kappa}$, as before. Writing down the expression for com_2 as a — finite — series in $\log_2 n$, we get

$$\begin{aligned}\mathsf{com}_2(w, n, \ell, \kappa, t_{opt}) = \ell &+ \frac{(w-1)\kappa}{2\log_2^2 w} \cdot \log_2^2 n + \\ &\frac{4\sqrt{w-1}\sqrt{\ell\kappa} + (w-1)\kappa}{2\log_2 w} \cdot \log_2 n \ .\end{aligned}$$

Interestingly enough, the second additive term of this expression is minimized when $w = -\frac{2}{W(-2/e^2)} \approx 4.92 \approx 5$, which seems to hint that this value of w may be close to the global minimum.

5.3 Rate

Assume again that s and $\log_1 n$ are integers. By dividing the length of useful information, $\log_2 n + \ell$, with the communication (3), we get that the new CPIR has rate

$$\begin{aligned} R =& \frac{\log_2 n + \ell}{\mathsf{com}_2(w, n, \ell, \kappa, t_{opt})} \\ =& 1 - 2\sqrt{(w-1)\kappa/\ell}\log_w n + \frac{2\log_2 n + (w-1)\kappa \log_w n(7\log_w n - 1)}{2\ell} + \\ & O(\ell^{-3/2}) \ . \end{aligned} \tag{5}$$

Indeed, the communication function

$$\mathsf{com}_2(w, n, \ell, \kappa, t_{opt}) = \sum_{i=0}^{\infty} a_i \ell^{1-i/2}$$

is given by Eq. (3), where $a_0 = 1$, $a_1 = 2\sqrt{(w-1)\kappa}\log_w n$, $a_2 = ((w-1)\kappa(\log_w n + 1)\log_w n)/2$, $a_i = 0$, where $i \geq 3$. Let

$$R = \sum_{i=0}^{\infty} b_i \ell^{1-i/2} \ .$$

We find b_i from the condition $\mathsf{com}_2(w, n, \ell, \kappa, t_{opt}) \cdot R = \log_2 n + \ell$ comparing coefficients of different powers:

$$\begin{aligned} \ell^2 :&\quad a_0 b_0 = 0 \Rightarrow && b_0 = 0 \ , \\ \ell^{3/2} :&\quad a_0 b_1 + a_1 b_0 = 0 \Rightarrow && b_1 = 0 \ , \\ \ell :&\quad a_0 b_2 + a_1 b_1 + a_2 b_0 = 1 \Rightarrow && b_2 = 1 \ , \\ \ell^{1/2} :&\quad a_0 b_3 + a_1 b_2 + a_2 b_1 = 0 \Rightarrow && b_3 = -a_1 \ , \\ \ell^0 :&\quad a_0 b_4 + a_1 b_3 + a_2 b_2 = \log_2 n \Rightarrow && b_4 = \log_2 n + a_1^2 - a_2 \ , \\ \ell^i \ , i < 0 :&\quad \sum_{i=0}^{n} a_i b_{n-i} = 0 \Rightarrow && b_i \ . \end{aligned}$$

Thus we arrive to Eq. (5).

One can verify that the second term of Eq. (5) is minimized when w is as in Eq. (4). Assuming $w = 5$, the rate is

$$1 - \frac{4}{\log_2 5} \cdot \sqrt{\kappa/\ell} \cdot \log_2 n + ((14\kappa \log_1 n - 2\kappa + \log_2 5)\log_1 n) \cdot \frac{1}{\ell} + O(\ell^{-3/2}) \ .$$

See Sect. 5.4 for a figure showing how the rate grows as a function of ℓ/κ for a concrete value of n.

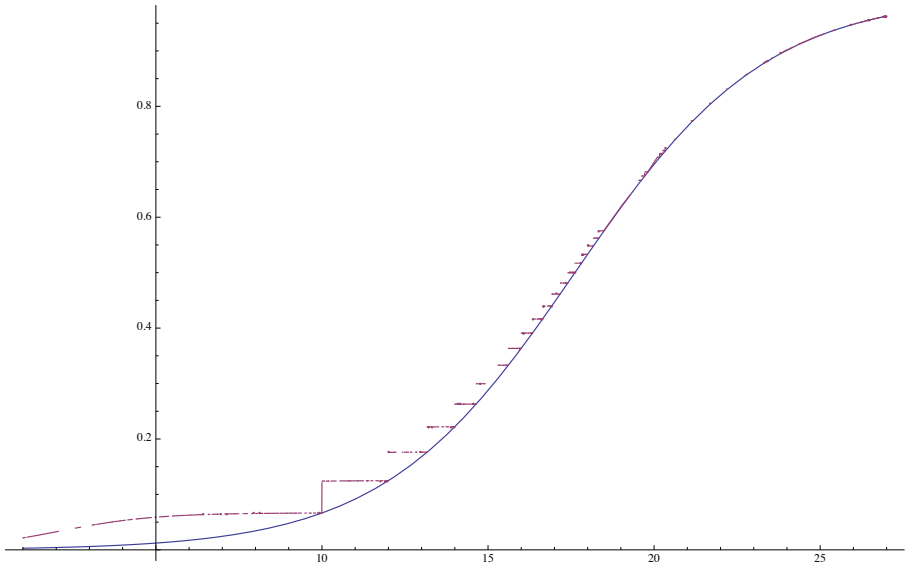

Fig. 4. The rate of the new CPIR protocol as a function of $\log_2(\ell/\kappa)$, i.e., on logarithmic scale, for $w = 5$, $n = 5^7$ and $\kappa = 2048$. The smooth (blue) line corresponds to the case without rounding errors. The jumpy (purple) line corresponds to the case with rounding errors; note that it also rounds up the non-private case, i.e., it uses $\ell + \lceil \log_2 n \rceil$ as the amount of useful information. This explains why the case with rounding errors usually has a better rate than the case without

5.4 Concrete Analysis

If the prerequisites of the theorem are not fulfilled (e.g., n is not a power of w), we need to use ceiling function in the computation of the communication function, that is, we are interested in the function $\lceil \mathsf{com}_2(\dots) \rceil := \lceil \mathsf{rec}_2(\dots) \rceil + \lceil \mathsf{sen}_2(\dots) \rceil$.

Kiayias et al [10] gave a few numerical examples of the efficiency of their CPIR protocol. In Fig. 1, we will give a comparison with the current work; the columns "theoretical" give the value of the function com_2, while the columns "With rounding" give the value of the function $\lceil \mathsf{com}_2 \rceil$. In all cases, $\kappa = 2048$ and $n = w^m = 5^7$. As we can see, due to the rounding errors present in the protocol of [10], the current work achieves always slightly better efficiency.

On Fig. 4, we depict the rate of the $\lceil \mathsf{com}_2 \rceil$ of the new CPIR protocol as a function of $\log_2(\ell/\kappa)$. In particular, the rate of the protocol from the current paper (when rounding included) is 0.917714 for $\ell = 10^5\kappa$ and 0.997207κ for $\ell = 10^8\kappa$. Computing a similar graphic for the CPIR protocol of [10] would be quite time consuming.

If n is arbitrary (not a power of w), then a standard approach is to add to the database a number of dummy elements so as to increase the database size to the next power of w. This will incur similar — very small! — penalties for the protocols of [10] and of the current paper. For example, consider the cases

$\kappa = 2048$, $\ell = 10^5\kappa$, and $w = 5$. If $n = 5^7$ is increased to $n = 5^7 + 1$ (the worst case, since one has to add $5^7 - 1$ dummy elements), the rate will decrease from 0.917714 to 0.906919.

Finally, the problem of optimizing the protocol for small values of ℓ is clearly out of scope for the current work since we try to decrease rate for *large* values of ℓ. See, e.g., Sect. 3 of [12] for a discussion of the case of small ℓ.

6 Open Problems

A major open problem left by the current work is to construct a CPIR protocol where the rate function grows faster than Eq. (5), or to show that this is not possible. An impossibility proof might be possible in some restricted model.

The second open problem is to construct a rate-optimal CPIR protocol with the better computational complexity. (See [10] for a detailed discussion about the computational complexity.)

Acknowledgment. The authors were supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No 653497 (project PANORAMIX). The first (resp., the second) author was supported by institutional research funding IUT2-1 (resp., IUT20-57) of the Estonian Ministry of Education and Research.

References

1. Blake, I.F., Kolesnikov, V.: Strong Conditional Oblivious Transfer and Computing on Intervals. In: ASIACRYPT 2004. LNCS, vol. 3329, pp. 515–529
2. Bresson, E., Catalano, D., Pointcheval, D.: A Simple Public-Key Cryptosystem with a Double Trapdoor Decryption Mechanism and Its Applications. In: ASIACRYPT 2003. LNCS, vol. 2894, pp. 37–54
3. Cachin, C., Micali, S., Stadler, M.: Computational Private Information Retrieval with Polylogarithmic Communication. In: EUROCRYPT 1999. LNCS, vol. 1592, pp. 402–414
4. Damgård, I., Jurik, M.: A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System. In: PKC 2001. LNCS, vol. 1992, pp. 119–136
5. Damgård, I., Jurik, M.: A Length-Flexible Threshold Cryptosystem with Applications. In: ACISP 2003. LNCS, vol. 2727, pp. 350–364
6. Gentry, C., Ramzan, Z.: Single-Database Private Information Retrieval with Constant Communication Rate. In: ICALP 2005. LNCS, vol. 3580, pp. 803–815
7. Gjøsteen, K., Strand, M.: Can there be efficient and natural FHE schemes? Technical Report 2016/105, IACR (2016) `http://eprint.iacr.org/2016/105`, last accessed version from June 2016.
8. Ishai, Y., Paskin, A.: Evaluating Branching Programs on Encrypted Data. In: TCC 2007. LNCS, vol. 4392, pp. 575–594
9. Kiayias, A., Leonardos, N., Lipmaa, H., Pavlyk, K., Tang, Q.: Communication Optimal Tardos-based Asymmetric Fingerprinting. In: CT-RSA 2015. LNCS, vol. 9048, pp. 469–486

10. Kiayias, A., Leonardos, N., Lipmaa, H., Pavlyk, K., Tang, Q.: Optimal Rate Private Information Retrieval from Homomorphic Encryption. Proceedings on Privacy Enhancing Technologies **2015**(2) (2015) pp. 222–243
11. Kushilevitz, E., Ostrovsky, R.: Replication is Not Needed: Single Database, Computationally-Private Information Retrieval. In: FOCS 1997, pp. 364–373
12. Lipmaa, H.: An Oblivious Transfer Protocol with Log-Squared Communication. In: ISC 2005. LNCS, vol. 3650, pp. 314–328
13. Lipmaa, H.: First CPIR Protocol with Data-Dependent Computation. In: ICISC 2009. LNCS, vol. 5984, pp. 193–210
14. Ostrovsky, R., Skeith III, W.E.: Communication Complexity in Algebraic Two-Party Protocols. In: CRYPTO 2008. LNCS, vol. 5157, pp. 379–396
15. Paillier, P.: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In: EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238
16. Stern, J.P.: A New And Efficient All Or Nothing Disclosure of Secrets Protocol. In: ASIACRYPT 1998. LNCS, vol. 1514, pp. 357–371
17. Wegener, I.: Branching Programs and Binary Decision Diagrams: Theory and Applications. Monographs on Discrete Mathematics and Applications. Society for Industrial Mathematics (2000)