Poster: Faster Optimal-Rate Many-Server PIR

Syed Mahbub Hafiz
Indiana University Bloomington
shafiz@indiana.edu

Ryan Henry
University of Calgary
ryan.henry@ucalgary.ca

***Abstract*—We present a novel family of so-called 1-private, multi-server PIR protocols exhibiting unprecedented performance with respect to *every* cost metric—download, upload, computation, and round complexity—typically considered in the PIR literature. With two servers, our protocols match the performance of the fastest previously known protocols; with three servers, they are already about *twice* as fast. And as the number of servers grows larger, so too does the speedup relative to prior work.**

***Index Terms*—Private information retrieval; optimality.**

I. Introduction

We propose a new family of multiserver private information retrieval (PIR) protocols, which we call "one-extra-word" protocols. The new protocols extend a recent PIR protocol of Shah, Rashmi, and Ramchandran [6] that exhibits a truly remarkable property: to fetch a b-bit record, the client need only download $b + 1$ bits total; hence, we name it "one-extra-bit" construction. We find that allowing "a bit more" download (and optionally introducing computational assumptions) yields a family of protocols offering very attractive tradeoffs. In addition to Shah et al.'s protocol, this family includes as special cases (2-server instances of) the seminal protocol of Chor, Kushilevitz, Gilboa, and Sahai [4] and the recent DPF-based protocol of Boyle, Gilboa, and Ishai [3]. An implicit folklore "axiom" dogmatically permeating the research literature on multiserver PIR holds that the latter protocols are the "most efficient protocols possible" in the perfectly and computationally private settings, respectively. Yet our findings soundly refute this "axiom": These special cases are (by far) the *least* performant representatives of our family, with essentially *all other* parameter settings yielding instances that are *significantly* faster.

II. "One-extra-word" protocols

The database $\boldsymbol{D}$ is an $r \times s$ matrix over a finite field $\mathbb{F}$, in which each of the r rows is an s-word block of fetchable data. For each $j \in [1..s+1]$, let $\vec{\boldsymbol{e}}_j$ denote the jth standard basis vector of $\mathbb{F}^{s+1}$. Denote by $\mathbf{M}^{(r,s)} \subseteq \mathbb{F}^{r\times(s+1)}$ the set of all height-r matrices whose rows are vectors from the standard basis $\{\vec{\boldsymbol{e}}_1, \dots, \vec{\boldsymbol{e}}_{s+1}\}$, and consider the family (indexed by $i \in [1..r]$) of equivalence relations $\equiv_i$ defined on the $\boldsymbol{A}, \boldsymbol{B} \in \mathbf{M}^{(r,s)}$ as $\boldsymbol{A} \equiv_i \boldsymbol{B}$ iff $\mathrm{Row}_{i^*}(\boldsymbol{A} - \boldsymbol{B}) \neq \vec{\boldsymbol{0}}$ implies $i^* = i$, where $\mathrm{Row}_{i^*}(\boldsymbol{A} - \boldsymbol{B})$ denotes the i^*th row of $\boldsymbol{A}-\boldsymbol{B}$ and $\vec{\boldsymbol{0}}$ denotes the zero vector in $\mathbb{F}^{s+1}$. Equivalence class $\mathrm{Eq}(i;\, \boldsymbol{A})$ comprises of $s+1$ number of such matrices who differ at the ith row only.

Encoding the database: The database encoding algorithm takes as input the database $\boldsymbol{D} \in \mathbb{F}^{r\times s}$, a vector $\vec{\boldsymbol{v}} \in \mathbb{F}^s$, and a (surjective) mapping function $\varphi\colon \mathbf{M}^{(r,s)} \to [1..\ell]$; it outputs a collection of ℓ buckets—one per server. Denote the augmented database by $\boldsymbol{D}^* \coloneqq \boldsymbol{D} \| (\boldsymbol{D}\,\vec{\boldsymbol{v}}^{\mathrm{T}}) \in \mathbb{F}^{r\times(s+1)}$. Next, to populate the buckets, the algorithm computes, for each of the $(s+1)^r$ matrices $\boldsymbol{A}$ in $\mathbf{M}^{(r,s)}$, the *Frobenius inner product*[1], $\langle \boldsymbol{D}^*, \boldsymbol{A}\rangle_{\mathrm{F}}$, of $\boldsymbol{D}^*$ with $\boldsymbol{A}$, and then it places the result in the bucket indexed by $\varphi(\boldsymbol{A})$.

Fetching a block: The client fetches $\vec{\boldsymbol{D}}_i$ by first selecting a uniform random matrix $\boldsymbol{A} \in_{\mathrm{R}} \mathbf{M}^{(r,s)}$, and then retrieving $\langle \boldsymbol{D}^*, \boldsymbol{B}_j\rangle_{\mathrm{F}}$ from bucket $\varphi(\boldsymbol{B}_j)$ for each $\boldsymbol{B}_j \in \mathrm{Eq}(i;\, \boldsymbol{A})$. When the vector $\vec{\boldsymbol{v}}$ and mapping φ satisfy certain (easy) properties, one can prove that (i) the desired record is the unique solution to the system of linear equations

$$\begin{pmatrix} \vec{\boldsymbol{e}}_1 - \vec{\boldsymbol{v}} \\ \vdots \\ \vec{\boldsymbol{e}}_s - \vec{\boldsymbol{v}} \end{pmatrix} \vec{\boldsymbol{D}}_i^{\mathrm{T}} = \begin{pmatrix} \langle \boldsymbol{D}^*, \boldsymbol{B}_1\rangle_{\mathrm{F}} - \langle \boldsymbol{D}^*, \boldsymbol{B}_{s+1}\rangle_{\mathrm{F}} \\ \vdots \\ \langle \boldsymbol{D}^*, \boldsymbol{B}_s\rangle_{\mathrm{F}} - \langle \boldsymbol{D}^*, \boldsymbol{B}_{s+1}\rangle_{\mathrm{F}} \end{pmatrix},$$

and (ii) the protocol reveals no information about the query for i to any server.

Moreover, we prove that when $\mathbb{F}$ is a binary field, the computation cost of our protocols matches Beimel, Ishai, and Malkin's [1] lower bound for the computation cost of *any* PIR protocol; likewise, we prove that the download cost of our protocols matches Blackburn, Etzion, and Paterson's [2] lower bound for the download cost of *any* PIR protocol. The computational variants of our protocol have upload cost in $\Theta(\lg r \lg \ell)$, where ℓ is the number of servers.

III. Perfectly 1-private "Bit-more-than-a-bit" protocols

We now describe our new "bit-more-than-a-bit" construction, a subfamily of perfectly 1-private one-extra-word protocols parametrized by $\ell \geq 2$ and s. Each member of this family uses a binary field $\mathbb{F} = \mathbf{GF}(2^w)$ where $w = \lceil \frac{b}{s} \rceil$, the all-0s vector $\vec{\boldsymbol{v}} = \vec{\boldsymbol{0}}$, and the mapping $\varphi\colon \mathbf{M}^{(r,s)} \to [1..\ell]$ defined in Equation (1) below.

$$\varphi(\boldsymbol{A}) \coloneqq \textstyle\sum_{i=1}^{r} (s+1)^{i-1}\, \mathrm{Ord}_i(\boldsymbol{A}) \bmod \ell. \tag{1}$$

This mapping induces ℓ buckets, compared with $(s+1)^{r-1}$ buckets for the one-extra-bit mapping; hence, relative to the one-extra-bit construction, it reduces ℓ from $(b+1)^{r-1}$—which is super-exponential in $|\boldsymbol{D}|$—to an arbitrary $\ell \geq 2$.

IV. Computationally 1-private "Bit-more-than-a-bit"

This section presents our most efficient construction having $\ell = 2^L$ servers for any positive integer L which reduces the per-server upload cost from rL bits of Section III construction to just $(\lambda+2)\lceil \lg \frac{r}{\lambda} \rceil L$ bits, where λ is a security parameter. The download cost remains unchanged and the computation cost increases only modestly in practice (employing AES-NI instruction set.) This construction replaces the *uniform random* query strings from our perfectly 1-private construction

[1] The *Frobenius inner product* of $\boldsymbol{D}^*$ and $\boldsymbol{A}$ is $\langle \boldsymbol{D}^*, \boldsymbol{A}\rangle_{\mathrm{F}} \coloneqq \mathrm{tr}(\boldsymbol{D}^*\boldsymbol{A}^{\mathrm{T}})$ or, equivalently, the sum of the products of each pair of corresponding components in $\boldsymbol{D}^*$ and $\boldsymbol{A}$.

with *pseudorandom* query strings generated by an L-tuple of 2*-out-of-*2 *distributed point functions* [5].

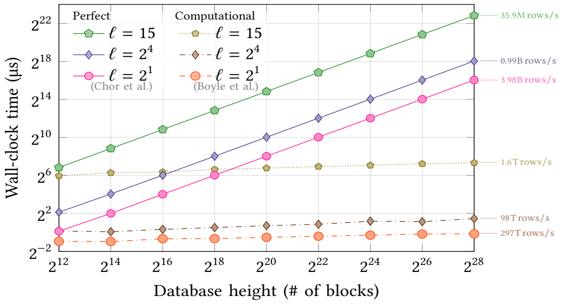

Fig. 1: Wall-clock time for (client-side) query construction in both computationally and perfectly 1-private protocols. When ℓ is not a power of 2, the smoothing parameter is $poly(\lambda) = \lambda = 128$.

We obtain the required "1-out-of-ℓ" privacy of the PIR queries from "1-out-of-2" secrecy of the $(2,2)$-DPFs by using an L-tuple of $(2,2)$-DPF key pairs, which the client samples independently and distributes to the servers. The efficiency of our construction also benefits from the fast *full-domain evaluation* algorithm of Boyle et al. [3; §3.2.1], which expands a $(2,2)$-DPF key k into the length-r bit vector $\mathsf{EvalFull}(k) := \langle \mathsf{Eval}(k,1), \mathsf{Eval}(k,2), \dots, \mathsf{Eval}(k,r) \rangle$. We also introduce the '$\Vert$' operator to denote the component-wise concatenation of length-r vectors; in particular, if $\vec{u} = \langle u_0, \dots, u_n \rangle$ and $\vec{v} = \langle v_0, \dots, v_n \rangle$, then $\vec{u} \Vert \vec{v} := \langle u_0 \| v_0, \dots, u_n \| v_n \rangle$.

A. Power-of-2 number of servers, $\ell = 2^L$

Query construction ("DPF key distribution"): Assign to each of the ℓ servers a numeric label j between 0 and $\ell - 1$ and then, for each $j \in [0\,..\,\ell-1]$, consider the L-bit binary representation $(j_{L-1} \cdots j_1 j_0)_2$ of j, where j_e denotes the eth-least-significant bit of j. To query for block $\vec{\boldsymbol{D}}_i$, the client samples L independent $(2,2)$-DPF key pairs, say

$$(k_0^{(L-1)}, k_1^{(L-1)}), \dots, (k_0^{(0)}, k_1^{(0)}) \leftarrow \mathsf{Gen}(1^\lambda; i) \times \cdots \times \mathsf{Gen}(1^\lambda; i),$$

and then, to each server $j \in [0\,..\,\ell-1]$, it sends the query string $q_j := \big(k_{j_{L-1}}^{(L-1)}, \dots, k_{j_0}^{(0)}\big)$.

Query expansion: Upon receiving q_j from the client, server j parses it as a sequence of DPF keys and then it performs a full-domain evaluation on each of the keys and concatenates the resulting bit vectors component-wise to obtain a length-r vector of L-bit integers; that is, it computes $\tilde{q}_j := \mathsf{EvalFull}(k^{(L-1)}) \Vert \cdots \Vert \mathsf{EvalFull}(k^{(0)})$. From here, the server proceeds exactly as it would upon receiving the query string $\tilde{q}_j$ directly from the client in Section III.

B. Arbitrary number of servers, $\ell \nmid 2^L$

We now describe how to extend the computationally 1-private bit-more-than-a-bit protocol to work with arbitrary number of servers, at the expense of some additional upload and computation overhead. Let a smoothing parameter $poly\colon \mathbb{N} \to \mathbb{N}$ be some positive integer-valued polynomial. Set $L = \lceil \lg \ell \rceil + poly(\lambda)$ and, as before, have the client sample an L-fold sequence of DPF key pairs and the server perform L full-domain evaluations and an L-ary component-wise concatenation as like Section IV-A. Another extra step to be performed by the server is to reduce the vector component-wise modulo ℓ to obtain the desired length-r vector of integers in $[0\,..\,\ell-1]$. The additional $poly(\lambda)$ DPF keys serve to "smooth out" the distribution, reducing the modulo bias to a negligible level.

V. Implementation and evaluation

This section evaluates performance of our constructions compared to the "folklore" protocols of Chor et al. and Boyle et al. We conducted all experiments on a workstation running Red Hat 7.6 on a quad-core Intel(R) Core(TM) i7-4770 CPU @ 3.40 GHz with 16 GiB of RAM. Figure 1 compares the cost of (client-side) query generation for DPF-based computationally 1-private protocol instances versus perfectly 1-private protocol instances. We observe a notable performance penalty when ℓ not a power of 2. As expected, the computationally 1-private protocols scale quite well (indeed, logarithmically) relative to the perfectly 1-private protocols. Figure 2 compares the time required for each of $\ell = 2^L$ servers to respond to a query.

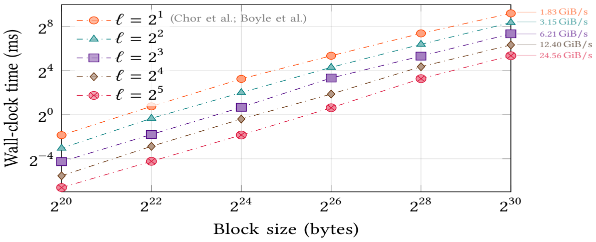

Fig. 2: Wall-clock time for (server-side) response generation as block size (b) grows. All experiments fix $r = 8$.

VI. Conclusion and future work

We transformed the one-extra-bit PIR construction of Shah et al. from a (highly impractical) theoretical result into what we believe to be the most efficient PIR protocol currently in existence, albeit under a very strong trust assumption. For future work, we plan to explore how our approach extends to the setting of computationally t-private protocols for thresholds $t > 1$.

Acknowledgements.: This work is supported by the NSF Grant No. 1718475 and an NSERC Discovery Grant.

References

[1] Amos Beimel, Yuval Ishai, and Tal Malkin. Reducing the servers' computation in private information retrieval: PIR with preprocessing. *Journal of Cryptology*, 17(2):125–151 (March 2004).

[2] Simon R. Blackburn, Tuvi Etzion, and Maura B. Paterson. PIR schemes with small download complexity and low storage requirements. In *Proceedings of ISIT 2017*, pages 146–150, Aachen, Germany (June 2017).

[3] Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing: Improvements and extensions. In *Proceedings of CCS 2016*, pages 1292–1303, Vienna, Austria (October 2016).

[4] Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. Private information retrieval. *Journal of the ACM (JACM)*, 45(6):965–981 (November 1998).

[5] Niv Gilboa and Yuval Ishai. Distributed point functions and their applications. In *Advances in Cryptology: Proceedings of EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 640–658, Copenhagen, Denmark (May 2014).

[6] Nihar B. Shah, K. V. Rashmi, and Kannan Ramchandran. One extra bit of download ensures perfectly private information retrieval. In *Proceedings of ISIT 2014*, pages 856–860, Honolulu, HI, USA (June–July 2014).