

DOI:10.1145/1721654.1721674

Cryptographic protocols safeguard the privacy of user queries to public databases.

BY SERGEY YEKHANIN

Private Information Retrieval

THE UBIQUITY OF the Internet means a plethora of online public databases and an indispensable resource for retrieving up-to-date information. But it also poses a significant risk to user privacy, since a malicious database owner may monitor user queries and infer what the user is after. Indeed, in cases where user intentions are to be kept secret, users are often cautious about accessing public databases. For example, investors querying a stock-market database for the current market value of certain stocks might prefer not to reveal their interest in the stocks because it could inadvertently influence their price. Alternatively, companies might want to search for certain patents without revealing the patents' identities.

Private information retrieval (PIR) schemes are cryptographic protocols designed to safeguard the privacy of database users. They allow clients to retrieve records from public databases while completely hiding the identity of the retrieved records from database owners. The possibility of retrieving

database records without revealing their identities to the owner of the database may seem beyond hope. Note, however, that a trivial solution is available: When users want a single record, they can ask for a copy of the whole database. This solution involves enormous communication overhead and is likely to be unacceptable. It turns out⁷ that for users who want to keep their privacy fully protected (in the “information-theoretic” sense), this trivial solution is optimal.

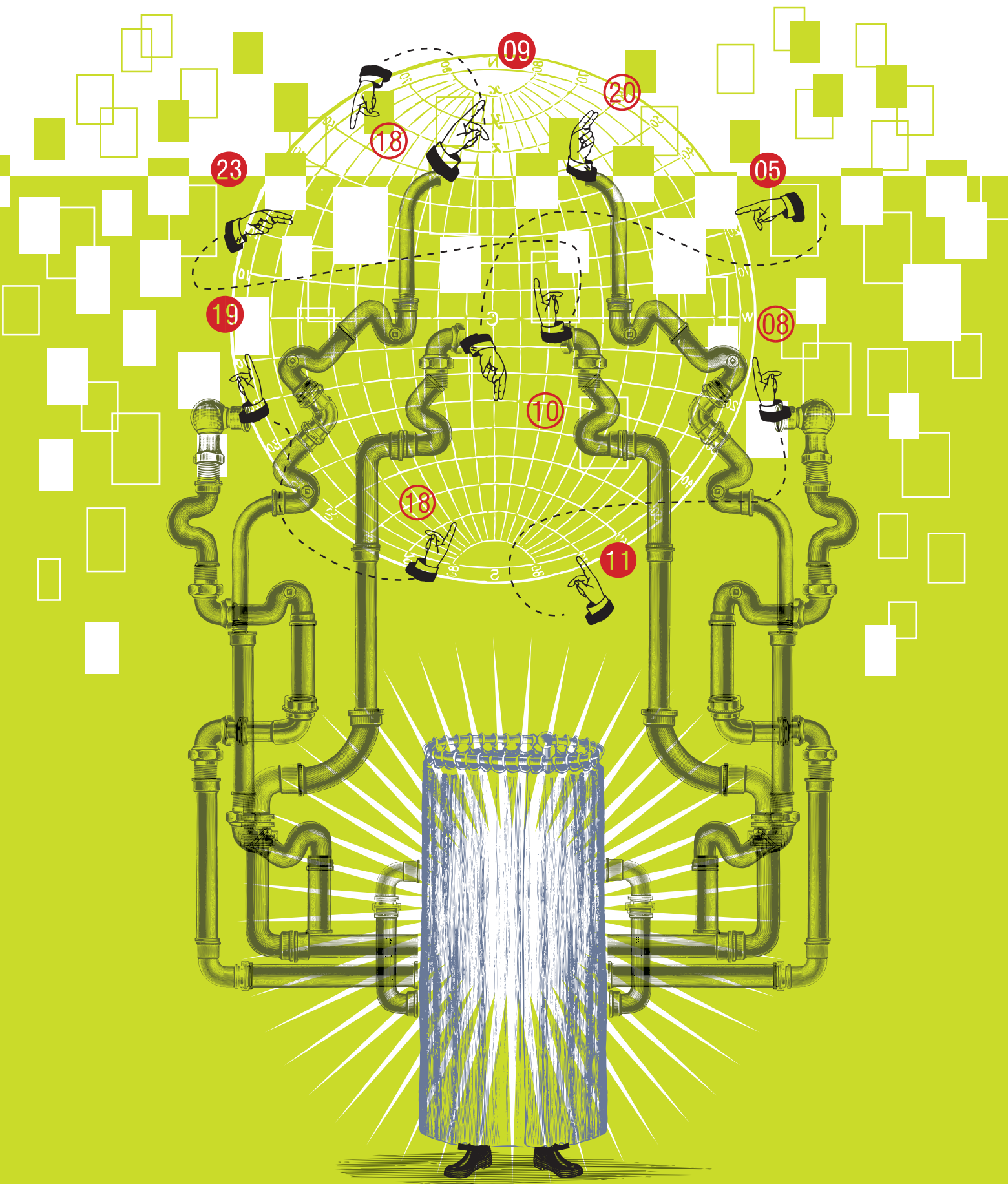
Fortunately, the negative result applies only to databases stored on a single server, rather than those replicated across several servers. In 1995, Chor et al.⁷ came up with PIR schemes that enable private retrieval of records from replicated databases, with a non-trivially small amount of communication. In such protocols, users query each server holding the database. The protocol ensures that each individual server (by observing only the query it receives) gets no information about the identity of the items of user interest. Chor et al.'s seminal paper⁷ triggered an important and extensive body of follow-up work.^{1,3,4,8,12,18,21,23,24}

Computational PIR. PIR schemes discussed here are often called “information theoretic” because they provide an absolute guarantee that each server participating in protocol execution gets no information about what users are after. PIR has also been studied in a computational setting.^{6,10,15,16} Computational PIR schemes are similar to their information-theoretic counterparts but provide a weaker

» key insights

- **User privacy is at risk whenever owners of publicly available network resources are able to trace, record, and mine user activity.**
- **The digital ecosystem involves not only such threats to personal privacy but also promising cures.**
- **Information-theoretic protocols decompose each user's query into several subqueries, ensuring they include no information about the user's intent.**

ILLUSTRATION BY CARL WIENS



guarantee. Specifically, they ensure only that a server cannot get any information about user intent unless it solves a certain computationally hard problem (such as factoring a large random integer). Providing privacy or security guarantees based on computational hardness assumptions is common in modern cryptography. In contrast to information-theoretic PIR schemes, computational PIR protocols with low communication overhead exist (under standard assumptions), even when a database is stored on a single server (see the figure here). However, for typical real-life parameters, known computational protocols are less efficient than known information-theoretic ones.

Locally decodable codes. PIR schemes are intimately related to a special class of error-correcting codes called “locally decodable codes,” or LDCs, that, on their own, are objects of interest. All recent constructions of information-theoretic PIR schemes work by first constructing LDCs, then converting them to PIRs.

Error-correcting codes help ensure reliable transmission of information over noisy channels, as well as reliable storage of information on a medium that may be partially corrupted over time or whose reading device is subject to errors. Such codes allow one to add redundancy, or bit strings, to messages, encoding them into longer bit strings, called “codewords,” in a way that the message can still be recovered even if a certain fraction of the codeword bits are corrupted. In typical applications of error-correcting codes the message is first partitioned into small blocks, each of which is then encoded sepa-

ately. This encoding strategy allows efficient random-access retrieval of the information, since one must decode only the portion of data in which one is interested. Unfortunately, this strategy yields poor noise resilience, since, when even a single block (out of possibly tens of thousands) is completely corrupted, some information is lost. In view of this limitation it would seem preferable to encode the whole message into a single codeword of an error-correcting code. Such a solution improves the robustness to noise but is hardly satisfactory, since one needs to look at the whole codeword in order to recover any particular bit of the message (at least when using classical error-correcting codes). Such decoding complexity is prohibitive for today’s massive data sets.

LDCs simultaneously provide efficient random-access retrieval and high noise resilience by allowing reliable reconstruction of an arbitrary bit of the message from looking at only a small number of randomly chosen codeword bits. Local decodability comes at the price of certain loss in terms of code efficiency. Specifically, LDCs require longer codeword lengths than their classical counterparts. Though LDCs were discussed in the literature^{2,19} in the early 1990s, the first formal definition of LDCs was given in 2000 by Katz and Trevisan.¹³ Further work on the efficiency of LDCs was covered in.^{3,4,8,12,14,18,24}

Outline. Computational and information theoretic PIR schemes rely on different sets of techniques. The main focus in this article is information-theoretic schemes, reserving the term PIR for information-theoretic protocols. In the following section we make

the notions of information-theoretic PIR and LDCs more concrete, explaining the relationship between them. Later, we demonstrate how nontrivial PIR is possible, offering an exposition of one of the simplest and earliest schemes. We then present the most basic PIR protocol of the current generation. Finally, we discuss computational PIR protocols and an early single-server scheme due to Kushilevitz and Ostrovsky.¹⁵

Preliminaries

Beginning with a (slightly informal) definition of PIR schemes, we model the database as an n -bit string x that is replicated between a small number k of non-communicating servers S_1, \dots, S_k . The user holds an index i (an integer between 1 and n) and is interested in obtaining the value of the bit x_i . To achieve this goal, the user tosses several random coins, queries each server, and receives replies from which the desired bit x_i is computed. The query to each server is distributed independently of i ; therefore, each server gets no information about what the user is after.

Note that user queries are not necessarily requests for particular individual bits or sets of database bits. Rather, they specify functions computed by the servers; for example, a query may specify a set of indices between 1 and n , and the server’s response may be the XOR of the database bits stored at these indices.

Historically, the main parameter of interest in PIR schemes was communication complexity, or a function of n measuring the total number of bits communicated between user and servers, maximized over all choices of x in $\{0, 1\}^n$, i in 1 to n , and the user’s random coin tosses.

The most efficient two-server PIR protocols available today have communication complexity of $O(n^{1/3})$. These protocols, which are due to Chor et al.,⁷ have never been improved upon. However, PIR schemes involving three or more servers have seen improvement.^{1,4,7,8,12,24} The best-known are due to Efremenko⁸ and Itoh and Suzuki.¹² Efremenko’s three-server PIR schemes require $f(n)$ bits of communication, where $f(n)$ is a certain function that grows slower than any



Two-server information-theoretic PIR scheme. To retrieve a database record, the user queries two servers, each of which stores a copy of the database, but individual queries carry no information about what the user is after. The desired record is obtained from the servers’ combined responses.

polynomial in n but faster than any power of the logarithm of n . The best lower bound for the communication complexity of two-server PIR is $5 \log n$ due to Wehner and de Wolf.²¹ Closing the gap between upper and lower bounds for PIR communication complexity is a major open problem.


Now we address a more detailed treatment of LDCs that have three parameters: k , δ , and ε . A (k, δ, ε) -LDC encodes n -bit messages x to N -bit codewords $C(x)$, such that for every i between 1 and n the bit x_i can be recovered with probability $1 - \varepsilon$ by a randomized decoding procedure that reads only k codeword bits, even after the codeword $C(x)$ is adversarially corrupted in up to δN locations. Here, the probability is only over the decoder's random coin tosses.

We illustrate the notion of LDCs by presenting a $(2, \delta, 2\delta)$ -(Hadamard) LDC that encodes n -bit messages to 2^n -bit codewords. In what follows, let $[n]$ denote the set $\{1, \dots, n\}$.


Hadamard code. Every coordinate in the Hadamard code corresponds to one (of 2^n) subsets of $[n]$ and stores the XOR of the corresponding bits of the message x . Let y be an (adversarially corrupted) encoding of x . Given an index $i \in [n]$ and y , the Hadamard decoder picks a set S in $[n]$ uniformly at random and outputs the XOR of the two coordinates of y corresponding to sets S and $S \Delta \{i\}$. (Here, Δ denotes the symmetric difference of sets (such as $\{1, 4, 5\} \Delta \{4\} = \{1, 5\}$, and $\{1, 4, 5\} \Delta \{2\} = \{1, 2, 4, 5\}$). It's not difficult to verify that both decoders' queries go to uncorrupted locations if y differs from the correct encoding of x in at most δ fraction of coordinates than with probability $1 - 2\delta$. In such cases, the decoder correctly recovers the i -th bit of x .

The Hadamard code allows for a super-fast recovery of the message bits (such as, given a codeword corrupted in 0.1 fraction of coordinates, one is able to recover any bit of the message with probability 0.8 by reading only two codeword bits) at a price of very large codeword length. Designing LDCs with optimal, or smallest possible, codeword length is a major challenge.

The definition of LDCs implies that every bit x_i of the message can



The protocol ensures that every individual server (by observing only the query it receives) gets no information about the identity of the items of user interest.



be recovered from many different k -tuples of codeword bits. Note that such k -tuples cannot all belong to a small ($o(N)$ -size) subset S of codeword coordinates, since corrupting the coordinates in S could lead to the high probability of a decoding error. Thus the distribution of every individual query of the decoder must be somewhat close to a uniform distribution on the codeword bits. Indeed, many known LDCs have decoders whose individual queries are distributed perfectly uniformly; such LDCs are called “perfectly smooth.”

There is a strong relationship between LDCs and PIR schemes. Short LDCs yield efficient PIR schemes and vice versa. Here, we demonstrate the flavor of this relationship, presenting a general procedure that obtains a k -server PIR scheme from any perfectly smooth k -query LDC.

Let C be a perfectly smooth LDC encoding n -bit messages to N -bit codewords. At the preprocessing stage, servers S_1, \dots, S_k encode the n -bit database x with the code C . Next, a user interested in obtaining the i -th bit of x tosses random coins and generates a k -tuple of queries (q_1, \dots, q_k) , such that x_i can be computed from $C(x)_{q_1}, \dots, C(x)_{q_k}$. For every j in $[k]$, the user sends the query q_j to the server S_j . Each server S_j responds with a one-bit answer $C(x)_{q_j}$. The user combines the servers' responses to obtain x_i .

Verifying that the protocol is private is straightforward, since for every j in $[k]$ the query q_j is uniformly distributed over the set of codeword coordinates; the total communication is given by $k(\log N + 1)$.

Early PIR Scheme

Let d be a small integer. Our goal here is to demonstrate how nontrivial PIR is possible by presenting a simple $(d+1)$ -server scheme with $O(n^{1/d})$ communication to access an n -bit database. The key idea behind this scheme is polynomial interpolation in a finite-field setting.

We begin with some technical background. Let $p > d$ be a prime. It is well known that addition and multiplication of numbers $\{0, \dots, p-1\}$ modulo p satisfy the standard identities that one is used to over the real numbers. That is, numbers $\{0, \dots, p-1\}$ form a finite

field with respect to these operations. This field is denoted by F_p . In what follows we deal with polynomials defined over finite fields. Such polynomials have all algebraic properties that one is used to with polynomials over the real numbers. Specifically, a univariate polynomial over F_p of degree d is uniquely determined by its values at any $d + 1$ points.

Let m be a large integer. Let E_1, \dots, E_n be a certain collection of n vectors over F_p of dimension m . The collection is fixed and independent of the n -bit database x . We assume the collection is known to both the servers and the user. On the preprocessing stage of the PIR protocol, each of $(d + 1)$ servers represents the database x by the same degree d polynomial f in m variables. The key property of such a polynomial is that for every i in $[n] : f(E_i) = x_i$. In order to ensure that such a polynomial f exists we choose m to be reasonably large compared to n . Setting $m = O(n^{1/d})$ suffices.

Now suppose the user wants to retrieve the i -th bit of the database and knows the collection of vectors E_1, \dots, E_n . The user's goal is thus to recover the value of the polynomial f (held by the servers) at E_i . Obviously, the user cannot explicitly request the value of f at E_i from any of the servers, since such a request would ruin the privacy of the protocol; that is, some server will get to know which database bit the user is after. Instead, the user obtains the value of $f(E_i)$ indirectly, relying on the rich structure of local dependencies between the evaluations of a low-degree polynomial f at multiple points. Specifically, the user generates a randomized collection of m -dimensional vectors P_1, \dots, P_{d+1} over F_p such that:

1. Each of the vectors P_λ is individually uniformly random and thus provides no information about E_i ; and
2. The values of any degree d polynomial (including the polynomial f) at P_1, \dots, P_{d+1} determine the value of the polynomial at E_i .

The user sends each server one of the vectors P_1, \dots, P_{d+1} . The servers then evaluate the polynomial f at the vectors they receive and return the values they obtain back to the user. The user combines the values $f(P_1), \dots, f(P_{d+1})$ to get the desired value $f(E_i)$. The protocol is perfectly private, and the com-

munication amounts to sending $(d + 1)$ vectors of dimension m to the servers and a single value back to the user. Here, we elaborate on how vectors P_1, \dots, P_{d+1} are chosen.

The user picks an m -dimensional vector V uniformly at random and for every λ between 1 and $d+1$ sets $P_\lambda = E_i + \lambda V$. Clearly, every individual vector P_λ is uniformly random. To see that values of $f(P_1), \dots, f(P_{d+1})$ determine the value of $f(E_i)$ consider a univariate polynomial $g(\lambda) = f(E_i + \lambda V)$. Note that the degree of g is at most d . Therefore the values of g at $d + 1$ points determine the polynomial g uniquely. It remains to notice that $g(\lambda) = f(P_\lambda)$ and $g(0) = f(E_i) = x_i$.

This PIR scheme is a classical example of PIR schemes. The ideas behind it can be traced back to Babai et al.² The idea of obtaining PIR schemes and LDCs through polynomials was used extensively thereafter.^{1,3,7} The most powerful constructions in this framework are due to Beimel et al.⁴; see also Woodruff and Yekhanin.²³

Modern LDCs and PIRs

In 2007, the author of the current article proposed a new approach to designing LDCs and PIR schemes,²⁴ leading to development of today's most effective codes and schemes.^{8,12,18} The new constructions are not based on polynomials; their key technical ingredient is a design of a large family of sets with restricted intersections. Here, we offer a bird's-eye view of the construction of LDCs. Such codes are easily turned into PIR schemes.

Let k be a small integer. Here, we design a binary k -query LDC C that encodes n -bit messages to codewords. This construction involves two steps: The first is a reduction to a problem of constructing a certain family of sets with restricted intersections; the second is an algebraic construction of the desired family.

Step 1. Construct an F_2 -linear code C , so:

► C is an F_2 -linear map. For any two messages x_1, x_2 in F_2^n we have $C(x_1 + x_2) = C(x_1) + C(x_2)$, where the sums of vectors are computed modulo two in each coordinate; and

► The decoding algorithm proceeds by tossing random coins, reading a certain k -tuple of coordinates of the cor-

rupted codeword and outputting the XOR of the values in these coordinates.

For i in $[n]$, let e_i denote a binary n -dimensional vector, whose unique non-zero coordinate is i . Observe that every linear LDC admits a combinatorial description. That is, to define a linear LDC it is sufficient to specify for every i in $[n]$:

► A set T_i of coordinates of $C(e_i)$ that are set to 1. Such sets completely specify the encoding, since for any message x , $C(x) = \sum_{i: x_i=1} C(e_i)$; and

► A family Q_i of k -size subsets of codeword coordinates that can be read by a decoding algorithm while reconstructing the i -th message bit.

Not every collection of sets T_i and families Q_i yields an LDC. Certain combinatorial constraints must be satisfied. The basic rationale for these constraints is the following:

1. The decoding must be correct in case no codeword bits have been corrupted. This implies that for every i, j in $[n]$ and any k -set S in Q_i the size of $S \cap T_j$ must be odd if $i = j$ and even otherwise; and

2. The distribution of individual queries of the decoding algorithm must be close to uniform. This implies that for every i in $[n]$, the union of the k -sets in Q_i must be large relative to the number of codeword coordinates.

Step 2. Design a collection of sets T_i and families Q_i that satisfy these constraints. This is where most of the technical work occurs. The construction is supported by a strong geometric intuition. We consider a bijection between the set of codeword coordinates and a set of m -dimensional vectors over a finite field of cardinality k . We choose sets T_i to be unions of certain parallel hyperplanes in the m -dimensional linear space over F_k and families Q_i to be certain families of lines. (Note that lines over F_k have only k points.) We use basic algebra to argue about the intersection sizes.

Computational PIR

Computational PIR schemes are attractive because they avoid the need to maintain replicated copies of a database and do not compromise user privacy against colluding servers. Here, we share a high-level overview (the basic version) of the Kushilevitz and

Ostrovsky scheme that relies on the standard hardness assumption¹⁵—the “quadratic residuosity assumption.”

Let m be a positive integer. A number a is called a “quadratic residue,” or QR, modulo m , if there exists an integer x such that $x^2 = a \bmod m$. Otherwise, a is called a “quadratic non-residue,” or QNR, modulo m . The QR assumption states that it is computationally hard to distinguish numbers that are QRs modulo m from those that are not, unless one knows the factorization of m .

The protocol. The server stores the n -bit database x in a square matrix of size s by s for $s = \sqrt{n}$.

$$\begin{bmatrix} x_{11} & \dots & x_{1j} & \dots & x_{1s} \\ \vdots & & \vdots & & \vdots \\ x_{i1} & \dots & x_{ij} & \dots & x_{is} \\ \vdots & & \vdots & & \vdots \\ x_{s1} & \dots & x_{sj} & \dots & x_{ss} \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ b_i \\ \vdots \\ a_s \end{bmatrix}$$

Suppose the database user is interested in obtaining the value x_{ij} for some i, j between 1 and s . The user would select a large integer m together with its factorization at random and generate $s - 1$ random QRs $a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_s$ modulo m , as well as a single random QNR b_i . The user would then send the string $a_1, \dots, a_{i-1}, b_i, a_{i+1}, \dots, a_s$ and the integer m to the server.

The QR assumption implies that the server cannot distinguish b_i from integers $\{a_l\}$ and thus observes only a string of s random-looking integers u_1, \dots, u_s modulo m (one per each row of the database). The server responds with s integers π_1, \dots, π_s (one per each column of the database). Here each π_h is equal to a product of integers u_l for all l such that $x_{lh} = 1$; formally, $\pi_h = \prod_{l: x_{lh}=1} u_l \bmod m$.

Verifying that the value of π_j is going to be a QR modulo m if $x_{ij} = 0$ and is going to be a QNR modulo m is not hard if $x_{ij} = 1$, since a product of two QRs is a QR and the product of a QR with a QNR is a QNR. The user need only check whether π_j is a QR, which is easy, since the user knows the factorization of m .


The total amount of communication in this PIR scheme is roughly $O(\sqrt{n})$. Better protocols are available; see Gentry and Ramzan¹⁰ and Lipmaa¹⁶ for the most efficient computational PIR schemes.

Conclusion

With nearly 15 years of development, the PIR field has grown large and deep, with many subareas and connections to other fields. For more details, see Ostrovsky and Sketch¹⁷ for a survey of computational PIR and Trevisan²⁰ and Yekhanin²⁵ for a survey of information theoretic PIR; see also Gasarch.⁹

Here, we have concentrated on a single (the most studied) aspect of PIR schemes—their communications complexity. Another important aspect of PIR schemes is the amount of computation servers must perform in order to respond to user queries.^{5,11,23} We are hopeful that further progress will lead to the wide practical deployment of these schemes.

Acknowledgments

We would like to thank Martin Abadi, Alex Andoni, Mihai Badiu, Yuval Ishai, Sumit Nain, and Madhu Sudan and the anonymous referees for helpful comments regarding this article. 

References

1. Ambainis, A. Upper bound on the communication complexity of private information retrieval. In *Proceedings of the Automata, Languages and Programming, 24th International Colloquium LNCS 1256* (Bologna, Italy, July 7–11). Springer, 1997, 401–407.
2. Babai, L., Fortnow, L., Levin, L., and Szegedy, M. Checking computations in polylogarithmic time. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing* (New Orleans, LA, May 5–8). ACM Press, New York, 1991, 21–31.
3. Beimel, A., Ishai, Y., and Kushilevitz, E. General constructions for information-theoretic private information retrieval. *Journal of Computer and System Sciences* 71, 2 (2005), 213–247.
4. Beimel, A., Ishai, Y., Kushilevitz, E., and Raymond, J.F. Breaking the $O(n^{1/(2k-1)})$ barrier for information-theoretic private information retrieval. In *Proceedings of the 43rd Symposium on Foundations of Computer Science* (Vancouver, B.C., Nov. 16–19). IEEE Computer Society, 2002, 261–270.
5. Beimel, A., Ishai, Y., and Malkin, T. Reducing the servers' computation in private information retrieval: PIR with preprocessing. In *Proceedings of the 20th Annual International Cryptology Conference LNCS 1880* (Santa Barbara, CA, Aug. 20–24). Springer, 2000, 55–73.
6. Cachin, C., Micali, S., and Stadler, M. Computationally private information retrieval with polylogarithmic communication. In *Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques LNCS 1592* (Prague, Czech Republic, May 2–6). Springer, 1999, 402–414.
7. Chor, B., Goldreich, O., Kushilevitz, E., and Sudan, M. Private information retrieval. *Journal of the ACM* 45, 6 (1998), 965–981.
8. Efremenko, K. 3-query locally decodable codes of subexponential length. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing* (Bethesda, MD, May 31–June 2). ACM Press, New York, 2009, 39–44.
9. Gasarch, W. Web page on private information retrieval; <http://www.cs.umd.edu/~gasarch/pir/pir.html>
10. Gentry, C. and Ramzan, Z. Single-database private information retrieval with constant communication rate. In *Proceedings of the 32nd International Colloquium on Automata, Languages and*

Programming LNCS 3580 (Lisbon, Portugal, July 11–15). Springer, 2005, 803–815.

11. Ishai, Y., Kushilevitz, E., Ostrovsky, R., and Sahai, A. Batch codes and their applications. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing* (Chicago, June 13–16). ACM Press, New York, 2004, 262–271.
12. Itoh, T. and Suzuki, Y. New constructions for query-efficient locally decodable codes of subexponential length. *IEICE Transactions on Information and Systems* E93-D, 2 (2010), 263–270.
13. Katz, J. and Trevisan, L. On the efficiency of local decoding procedures for error-correcting codes. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing* (Portland, OR, May 21–23). ACM Press, New York, 2000, 80–86.
14. Kerenidis, I. and de Wolf, R. Exponential lower bound for 2-query locally decodable codes via a quantum argument. *Journal of Computer and System Sciences* 69, 3 (2004), 395–420.
15. Kushilevitz, E. and Ostrovsky, R. Replication is not needed: Single-database computationally private information. In *Proceedings of the 11th Annual Symposium on Foundations of Computer Science* (Miami Beach, FL, Oct. 19–22). IEEE Computer Society, 1997, 364–373.
16. Lipmaa, H. An oblivious transfer protocol with log-squared communication. In *Proceedings of the 11th International Conference on Information Security LNCS 5222* (Taipei, Sept. 15–18). Springer, 2008, 314–328.
17. Ostrovsky, R. and Sketch, W.E. III A survey of single database private information retrieval: Techniques and applications. In *Proceedings of the 10th International Conference on Practice and Theory in Public-Key Cryptography LNCS 4450* (Beijing, Apr. 16–20). Springer, 2007, 393–411.
18. Raghavendra, P. A Note on Yekhanin's Locally Decodable Codes Technical Report TR07-016. Electronic Colloquium on Computational Complexity, 2007.
19. Sudan, M. *Efficient Checking of Polynomials and Proofs and the Hardness of Approximation Problems*. Ph.D. thesis, University of California, Berkeley, 1992.
20. Trevisan, L. Some applications of coding theory in computational complexity. *Quaderni di Matematica* 13 (2004), 347–424.
21. Wehner, S. and de Wolf, R. Improved lower bounds for locally decodable codes and private information retrieval. In *Proceedings of the 32nd International Colloquium on Automata, Languages, and Programming LNCS 3580* (Lisbon, Portugal, July 11–15). Springer, 2005, 1424–1436.
22. Woodruff, D. *New Lower Bounds for General Locally Decodable Codes Technical Report TR07-006*. Electronic Colloquium on Computational Complexity, 2007.
23. Woodruff, D. and Yekhanin, S. A geometric approach to information theoretic private information retrieval. *SIAM Journal on Computing* 47, 4 (2007), 1046–1056.
24. Yekhanin, S. Towards 3-query locally decodable codes of subexponential length. *Journal of the ACM* 55, 1 (2008), 1–14.
25. Yekhanin, S. *Locally Decodable Codes and Private Information Retrieval Schemes*. Ph.D. thesis. Massachusetts Institute of Technology, Cambridge, MA, 2007; <http://research.microsoft.com/en-us/um/people/yekhanin/Papers/phdthesis.pdf>

Sergey Yekhanin (yekhanin@microsoft.com) is a researcher in Microsoft Research Silicon Valley, Mountain View, CA.