Electronic Colloquium on Computational Complexity, Report No. 15 (2001)

Information-Theoretic Private Information Retrieval: A Unified Construction

Amos Beimel[*] Yuval Ishai[†]

February 9, 2001

Abstract

A Private Information Retrieval (PIR) protocol enables a user to retrieve a data item from a database while hiding the identity of the item being retrieved. In a *t-private, k-server* PIR protocol the database is replicated among k servers, and the user's privacy is protected from any collusion of up to t servers. The main cost-measure of such protocols is the *communication complexity* of retrieving a single bit of data.

This work addresses the *information-theoretic* setting for PIR, in which the user's privacy should be unconditionally protected from collusions of servers. We present a unified general construction, whose abstract components can be instantiated to yield both old and new families of PIR protocols. A main ingredient in the new protocols is a generalization of a solution by Babai, Kimmel, and Lokam to a communication complexity problem in the so-called *simultaneous messages* model.

Our construction strictly improves upon previous constructions and resolves some previous anomalies. In particular, we obtain: (1) t-private k-server PIR protocols with communication $O(n^{1/\lfloor(2k-1)/t\rfloor})$, where n is the database size. For $t > 1$, this is a substantial asymptotic improvement over the previous state of the art; (2) a constant-factor improvement in the communication complexity of 1-private PIR, providing the first improvement to the 2-server case since PIR protocols were introduced; (3) efficient PIR protocols with logarithmic query length. The latter protocols have applications to the construction of efficient families of *locally decodable codes* over large alphabets and to PIR protocols with reduced work by the servers.

Keywords. private information retrieval, distributed databases, information-theoretic cryptography, locally decodable codes.

[*] Dept. of Computer Science, Ben-Gurion University, Beer-Sheva 84105, Israel. E-mail: beimel@cs.bgu.ac.il. Homepage: www.cs.bgu.ac.il/∼beimel.

[†] DIMACS and AT&T Labs – Research, USA. E-mail: yuval@dimacs.rutgers.edu.

ISSN 1433-8092

1 Introduction

A Private Information Retrieval (PIR) protocol allows a user to retrieve a data item of his choice from a database, such that the server storing the database does not gain information on the identity of the item being retrieved. For example, an investor might want to know the value of a certain stock in the stock-market without revealing which stock she is interested in. The problem was introduced by Chor, Goldreich, Kushilevitz, and Sudan [10], and has since then attracted a considerable amount of attention. In formalizing the problem, it is convenient to model the database by an n-bit string x, where the user, holding some *retrieval index* i, wishes to learn the i-th data bit x_i. This default setting can be easily extended to handle more general scenarios, e.g., of larger data items, several users, or several retrieved items per user.

A trivial solution to the PIR problem is to send the entire database x the user. However, while being perfectly private, the *communication complexity* of this solution may be prohibitively large. Note that if the privacy constraint is lifted, an optimal solution to the retrieval problem is to have the user explicitly send i to the server and receive x_i in return. This non-private solution requires only $\lceil \log_2 n \rceil + 1$ bits of communication, whereas the trivial private solution mentioned above requires n communication bits. Thus, the most significant goal of PIR-related research has been to minimize the communication overhead imposed by the privacy constraint.

Unfortunately, if the server is not allowed to gain *any* information about the identity of the retrieved bit, then the linear communication complexity of the trivial solution is optimal [10]. To overcome this problem, Chor et al. [10] suggested that the user accesses k replicated copies of the database kept on different servers, requiring that each *individual* server gets absolutely no information on i. PIR in this setting is referred to as *information-theoretic* PIR.[1] The above privacy requirement naturally generalizes to *t-private* PIR, which keeps i private from any collusion of (at most) t out of the k servers.

The best 1-private PIR protocols known to date are summarized below: (1) a 2-server protocol with communication complexity of $O(n^{1/3})$ bits [10]; (2) a k-server protocol with communication complexity of $O(n^{1/(2k-1)})$ bits, for any constant k (Ambainis [1] improving on [10], see also Ishai and Kushilevitz [15]); and (3) an $O(\log n)$-server protocol with communication complexity of $O(\log^2 n \log\log n)$ bits ([10], and implicitly in Beaver and Feigenbaum [4] and Beaver, Feigenbaum, Kilian, and Rogaway [5]). For the more general case of t-private PIR, the best previous bounds were obtained in [15], improving on [10]. To present these bounds, it is convenient to use the following alternative formulation of the question:

> Given positive integers d and t, what is the smallest number of servers, k, for which there exists a t-private PIR protocol with communication complexity $O(n^{1/d})$?

In [15] it was shown that $k = \min\left(\lfloor dt - (d+t-3)/2 \rfloor\ ,\ dt - t + 1 - (d \bmod 2)\right)$ servers are sufficient. If t is fixed and d grows, the number of servers in this bound is roughly $(t - \frac{1}{2})d$.

No strong general lower bounds on PIR are known. Mann [22] obtained a constant-factor improvement over the trivial $\log_2 n$ bound, for any constant k. In the 2-server case, much stronger lower bounds can be shown under the restriction that the user reconstructs x_i by computing the exclusive-or of a *constant* number of answer bits, whose identity may depend on i (Karloff and Schulman [18], see also Goldreich and Trevisan [14]). These results still leave an exponential gap between known upper bounds and lower bounds. For a list of other PIR-related works the reader can consult, e.g., [6].

A different approach for reducing the communication complexity of PIR is to settle for *computational* privacy, i.e., privacy against computationally bounded servers. Following an initial 2-server solution by

[1] In principle, the term "information-theoretic PIR" may also refer to protocols which leak a negligible amount of information on i. However, there is still no evidence that such a relaxation is useful.

Chor and Gilboa [9], Kushilevitz and Ostrovsky [20] proved that in this setting one server suffices. Under a standard number theoretic intractability assumption they construct, for every constant $\epsilon > 0$, a *single*-server protocol with communication complexity of $O(n^{\epsilon})$ bits. Subsequently, Cachin, Micali, and Stadler [8] constructed, based on a new number theoretic intractability assumption, a single-server protocol with polylogarithmic communication.[2] From a practical point of view, single-server solutions are preferable to multi-server solutions for obvious reasons. However, they have some *inherent* limitations which can only be avoided in a multi-server setting. For instance, it is impossible for a (sublinear) single-server PIR protocol to have very short queries (say $O(\log n)$-bit long) sent from the user to the server, or very short answers (say, one bit long) sent in return. These two extreme types of PIR protocols, which can be realized in the information-theoretic setting, have found different applications (Di-Crescenzo, Ishai, and Ostrovsky [11], Beimel, Ishai, and Malkin [6]) and therefore serve as an additional motivation for studying information-theoretic PIR. A different, coding-related, motivation is discussed in Section 1.2.

1.1 Our Results

We present a unified general framework for the construction of PIR protocols, whose abstract components can be instantiated to meet or beat all previously known upper bounds. In particular we obtain:

- t-private k-server PIR protocols with communication complexity $O(n^{1/\lfloor (2k-1)/t \rfloor})$. In other words, $k > dt/2$ is sufficient for the existence of a t-private k-server PIR protocol with $O(n^{1/d})$ communication. For $t > 1$, this is a substantial asymptotic improvement over the previous state of the art [15]. For example, for $t = 2$ the communication complexity of our protocol is $O(n^{1/(k-1)})$, while the communication complexity of the best previous protocol [15] is $O(n^{1/\lfloor 2k/3 \rfloor})$. Our bound is essentially the best one could hope for without asymptotically improving the 1-private bounds.

- A constant-factor improvement in the communication complexity compared to the 2-server protocol of [10] and its 1-private k-server generalizations from [1, 15]. In the 2-server case, this provides the first improvement since the problem was introduced in [10].

- Efficient PIR protocols with logarithmic query length. Specifically, we construct a t-private k-server PIR protocol with $O(\log n)$ query bits and $O(n^{t/k+\epsilon})$ answer bits, for every constant $\epsilon > 0$. The 1-private protocols from this family were used in [6] to save computation in PIR via preprocessing, and have an interesting application, discussed in Section 1.2 below, to the construction of efficient *locally decodable codes* over large alphabets.

It is interesting to note that in contrast to previous PIR protocols, in which the user can recover x_i by reading only a *constant* number of answer bits (whose location depends only on i), most instances of our construction require the user to read *all* answer bits and remember either the queries or the randomness used to generate them. It is open whether the previous constructions of [15] (in particular, the t-private protocols for $t > 1$) can be improved if one insists on the above "easy reconstruction" feature, which allows the user's algorithm to be implemented using logarithmic space.

1.2 Locally Decodable Codes

Recently, information-theoretic PIR protocols have found a different flavor of application, to the construction of *locally decodable codes*. A locally decodable code allows to encode a database x into a string y over an

[2] For practical sizes of databases and security parameter the communication complexity of the single-server protocols of [20, 8] is inferior to that of known multi-server protocols, even in the 2-server case.

alphabet Σ, such that even if a large fraction of y is corrupted by an adversary, then each bit of x can still be decoded *with high probability* by probing *few* (randomly selected) locations in y. More formally, a code $C : \{0,1\}^n \to \Sigma^m$ is said to be (k, δ, ρ)-locally decodable, if every bit x_i of x can be decoded from $y = C(x)$ with success probability $1/2 + \rho$ by probing k entries of y, even if up to a δ-fraction of the m entries of y are corrupted. Katz and Trevisan [19] have shown an intimate relation between such codes and information-theoretic PIR. In particular, any information-theoretic PIR protocol can be converted into a locally decodable code with related efficiency by concatenating the answers of all servers on all possible queries. This motivates the construction of PIR protocols with short queries.[3]

The short-query instantiations of our PIR construction have an interesting interpretation in terms of locally decodable codes. The main focus in the works [19, 14] has been on the following question. Suppose that ρ, δ are restricted to be greater than some positive constant. Given a constant number of queries k and a *constant-size* (say, binary) alphabet Σ, what is the minimal asymptotic growth of the code length? Generalizing a PIR lower bound of [22], it is proved in [19] that for any constant k the code length must be super-linear. For the case of a linear code with $k = 2$ (non-adaptive) queries, an exponential lower bound on $m(n)$ has been obtained in [14]. While no super-polynomial lower bounds are known for the case $k > 2$, the best known upper bound (obtained from PIR protocols with a single answer bit per server, see Section 6.2) is $m(n) = 2^{O(n^{1/(k-1)})}$, which is exponential in n. Our construction answers the following dual question: Suppose that we insist on the code being *efficient*, namely of polynomial length. Then, how small can the alphabet Σ be? More precisely, given a constant k, how small can $\Sigma_k(n)$ be such that the code length $m(n)$ is polynomial and, as before, $\rho(n), \delta(n)$ are kept constant? The short-query variants of our construction imply the following upper bound: for any constants $k \geq 2$ and $\epsilon > 0$ it suffices to let $\Sigma_k(n) = \{0,1\}^{\beta(n)}$, where $\beta(n) = O(n^{1/k+\epsilon})$.

ORGANIZATION. In Section 2 we give an overview of our unified approach for constructing PIR protocols. In Section 3 we provide some necessary definitions. In Section 4 we describe a meta-construction of PIR protocols, in Section 5 we instantiate one of its crucial ingredients, and in Section 6 we derive new and old families of PIR protocols as instances of the meta-construction from Section 4. Finally, in Section 7 we obtain an optimization of previous protocols.

2 Overview of Techniques

At the heart of our constructions is a combination of two techniques. While neither of the two techniques is new, it is only their proper combination which results in the new improved protocols.[4]

2.1 Reduction to Polynomial Evaluation

A first technique is a reduction of the retrieval problem to the problem of multivariate polynomial evaluation. Specifically, the retrieval of x_i, where the servers hold x and the user holds i, is reduced to an evaluation of a multivariate polynomial p_x, held by the servers, on a point $E(i)$, which the user determines based on i. We refer to $E(i)$ as the *encoding* of i. As observed in [5] and, more generally, in [10], the degree of p_x can be decreased by increasing the length of the encoding $E(i)$. Originating in [4], different variants of

[3] For constructing locally decodable codes, a relaxed information-theoretic notion of PIR is sufficient (allowing some limited information leakage of i). However, to date there is no evidence that this type of relaxation can significantly help, as all known constructions of locally decodable codes directly correspond to known (perfect) PIR protocols.

[4] A restricted use of the same approach has been made in the companion work [6].

this reduction have been (implicitly or explicitly) used in virtually every PIR-related construction. In fact, even the seemingly "combinatorial" constructions from [10, 1] can be cast in this terminology. Interestingly, encodings realizing the optimal length-degree tradeoff, which were utilized in [10, 11] to construct special families of PIR protocols with short answer length, could not be used to realize the best known bounds on the *total* communication complexity. In [10, 1, 15] it seemed necessary to use a more redundant encoding for obtaining the best protocols. This situation is remedied in the current work, where the best communication-efficient constructions are obtained using an optimal encoding. Consequently, we get at least a constant-factor improvement to the communication complexity of all previous constructions, including in the 2-server case.

2.2 Simultaneous Messages Protocols for Polynomial Evaluation

A main ingredient in our new protocols is a generalization of a solution by Babai, Kimmel, and Lokam [3] to a communication complexity problem of computing the *generalized addressing function* in the so-called *simultaneous messages* (SM) model. Interestingly, this problem was motivated by circuit lower bounds questions, completely unrelated to privacy or coding. Towards solving their problem, they consider the following scenario. A degree-d m-variate polynomial p is known to k players, and k points $y_1, y_2, \ldots, y_k$ (each being an m-tuple of field elements) are distributed among them such that the j-th player knows all points except y_j. An external referee knows *all* k points y_j but does not know p. How efficiently can the value $p(y_1 + y_2 + \ldots + y_k)$ be communicated to the referee if the players are restricted to simultaneously sending messages to the referee?

A naive solution to the above problem is to have one of the players send an entire description of p to the referee. Knowing all y_j, the referee can then easily compute the required output. A key observation made in [3] is that it is in fact possible to do much better. By decomposing $p(y_1 + y_2 + \ldots + y_k)$ into terms and assigning each term to a player having the least number of unknown values, it is possible to write p as the sum of k *lower degree* polynomials in the inputs, each known to one of the players. More precisely, the j-th player can locally compute from its inputs a degree-$\lfloor d/k \rfloor$ polynomial p_j in its *unknown* inputs, such that $p(y_1 + \ldots + y_n) = p_1(y_1) + p_2(y_2) + \ldots + p_k(y_k)$. Then, by letting the j-th player communicate the (much shorter) description of p_j, the referee can compute the required output. The amount of savings obtained by this degree reduction technique depends on the values of the parameters m,d, and k. In [3, 2], due to constraints imposed by the specific problem they consider, the degree-reduction technique is applied with rather inconvenient choices of parameters. Thus, in their setting the full savings potential of the technique has not been realized. It turns out that in the PIR context, where there is more freedom in the choice of parameters, the full spectrum of possible tradeoffs is revealed.

It is instructive to look at three useful choices of parameters: (1) If $d = 2k - 1$, then the degree of each polynomial p_j is only $\lfloor (2k-1)/k \rfloor = 1$. When $m \gg d$, this $2k - 1$ savings factor in the degree makes the description size of each p_j roughly the $(2k - 1)$-th root of the description size of p. (2) If $d = k - 1$, the degree of each p_j becomes 0, and consequently communicating each p_j requires sending a single field element. (3) Finally, if $m \gg d$ and $d \gg k$, then the cost of communicating p_j is roughly the k-th root of that of communicating p. These three examples, respectively, turn out to imply the existence of k-server PIR protocols with: (1) both queries and answers of length $O(n^{1/(2k-1)})$; (2) queries of length $O(n^{1/(k-1)})$ and answers of length $O(1)$; (3) queries of length $O(\log n)$ and answers of length $O(n^{1/k+\epsilon})$, for an arbitrarily small constant $\epsilon > 0$. The fact that a single compact explanation fits all these seemingly unrelated expressions is quite remarkable.

2.3 Combining the Two Techniques

In the case of 1-private PIR, the two techniques can be combined in the following natural way. On input i, the user computes an encoding $y = E(i)$ and the servers compute a degree-d polynomial p_x such that $x_i = p_x(E(i))$. To generate his queries, the user "secret-shares" $E(i)$ among the servers by first breaking it into otherwise-random vectors $y_1, \ldots, y_k$ which add up to y, and then sending to each server $\mathcal{S}_j$ all vectors except y_j. Using the SM communication protocol described in the previous section, the servers communicate $x_i = p_x(y)$ to the user.

This simple combination of the two techniques is already sufficient to yield some of the improved constructions. In the remainder of this work we generalize and improve the above solution in several different ways. First, we abstract its crucial components and formulate a generic "meta-construction" in these abstract terms. Second, we instantiate the abstract components to accommodate more general scenarios, such as the one required for dealing with t-private PIR. Third, for both the 1-private and the t-private case, we attempt at optimizing the amount of replication in the setting of [3] while maintaining the quality of the solution (that is, we use a more efficient secret-sharing scheme for distributing $E(i)$). These generalizations motivate various extensions of the SM communication model as described above, which may be of independent interest.

3 Definitions

Notation. By $[k]$ we denote the set $\{1, \ldots, k\}$, and by $\binom{[k]}{t}$ all subsets of $[k]$ of size t. For a k-tuple v and a set $T \subseteq [k]$, let v_T denote the restriction of v to its T-entries. That is, if $T = \{i_1, \ldots, i_t\}$ and $v = (v_1, \ldots, v_k)$ then $v_T = (v_{i_1}, \ldots, v_{i_t})$. By Y_j for some j we represent a variable, while by the lower letter y_j we represent an assignment to the former variable. By H we denote the binary entropy function; that is, $H(p) = -p \log p - (1-p)\log(1-p)$, where in this paper all logarithms are taken to the base 2.

Polynomials. Let GF(q) denote the finite field of q elements. By $F[Y_1, \ldots, Y_m]$ we denote the linear space of all polynomials in the indeterminates $Y_1, \ldots, Y_m$ over the field F, and by $F_d[Y_1, \ldots, Y_m]$ its subspace consisting of all polynomials whose *total* degree is *at most d*, and whose degree in each indeterminate is at most $|F| - 1$. (The last restriction guarantees that each polynomial in $F_d[Y_1, \ldots, Y_m]$ represents a distinct function $p : F^m \to F$.) A natural basis for this linear space consists of all *monic monomials* satisfying the above degree restrictions. The case $F = \mathrm{GF}(2)$ will be the most useful in this work. In this case, the natural basis consists of all products of at most d distinct indeterminates. Hence, $dim(F_d[Y_1, \ldots, Y_m]) = \sum_{w=0}^{d} \binom{m}{w}$ for $F = \mathrm{GF}(2)$. We denote this dimension by $\Lambda(m, d) \stackrel{\mathrm{def}}{=} \sum_{w=0}^{d} \binom{m}{w}$. We will also be interested in $F_d[Y_1, \ldots, Y_m]$ where $|F| > d$. In this case, the dimension of the space is $\binom{m+d}{d}$.

3.1 PIR Protocols

We define 1-round information-theoretic PIR protocols.[5] A k-server PIR protocol involves k servers $\mathcal{S}_1, \ldots, \mathcal{S}_k$, each holding the same n-bit string x (the database), and a user who wants to retrieve a bit x_i of the database.

Definition 3.1 (PIR) *A k-server PIR protocol $\mathcal{P} = (\mathcal{R}, \mathcal{Q}_1, \ldots, \mathcal{Q}_k, \mathcal{A}_1, \ldots, \mathcal{A}_k, \mathcal{C})$ consists of a probability distribution $\mathcal{R}$ and three types of algorithms: query algorithms $\mathcal{Q}_j(\cdot,\cdot)$, answering algorithms $\mathcal{A}_j(\cdot,\cdot)$,*

[5] All the protocols constructed in this paper, as well as all previous information-theoretic PIR protocols, require a single round of queries and answers. This definition may be extended to multi-round PIR in the natural way.

*and a reconstruction algorithm $\mathcal{C}(\cdot,\cdot,\ldots,\cdot)$ ($\mathcal{C}$ has $k+2$ arguments). At the beginning of the protocol, the user picks a random string r from the distribution $\mathcal{R}$. For $j=1,\ldots,k$, it computes a query $q_j=\mathcal{Q}_j(i,r)$ and sends it to server $\mathcal{S}_j$. Each server responds with an answer $a_j=\mathcal{A}_j(q_j,x)$ (the answer is a function of the query and the database; without loss of generality, the servers are deterministic). Finally, the user computes the bit x_i by applying the reconstruction algorithm $\mathcal{C}(i,r,a_1,\ldots,a_k)$. A k-server protocol as above is a t-*private *PIR protocol, if it satisfies the following requirements:*

Correctness. *The user always computes the correct value of x_i. Formally, for every $i\in\{1,...,n\}$, every random string r, and every database $x\in\{0,1\}^n$, $\mathcal{C}(i,r,\mathcal{A}_1(\mathcal{Q}_1(i,r),x),\ldots,\mathcal{A}_k(\mathcal{Q}_k(i,r),x))=x_i$.*

t-**Privacy.** *Each collusion of up to t servers has no information about the bit that the user tries to retrieve: For every two indices $i_1,i_2\in[n]$ and for every $T\subseteq[k]$, where $|T|\le t$, the distributions $\mathcal{Q}_T(i_1,\cdot)$ and $\mathcal{Q}_T(i_2,\cdot)$ are identical.*

A PIR protocol is called *linear* over a field F if, when viewing x as a vector in F^n, the following condition holds: for every $j\in[k]$ and query q_j the answer function $\mathcal{A}_j(\cdot,q_j)$ is a linear mapping from F^n to F^{β_j} for some integer β_j. All PIR protocols constructed in this work are linear.

3.2 Linear Secret-Sharing

A *t-private secret-sharing scheme* allows a dealer to distribute a secret s among k players, such that any set of at most t players learns nothing on s from their joint shares, and any set of at least $t+1$ players can completely recover s from their shares. A secret-sharing scheme is said to be *linear over a field F* if $s\in F$, and the share received by each player consists of one or more linear combinations of the secret and r independently random field-elements (where the same random field-elements are used for generating all shares). A linear secret-sharing scheme is formally defined by a k-tuple $L=(L_1,\ldots,L_k)$ such that each L_j maps from $F\times F^r$ to F^{ℓ_j}, where ℓ_j is the j-th player share length. Finally, given a linear secret sharing scheme as above, a vector in F^m will be shared by independently sharing each of its m entries. We next define two linear secret sharing schemes that will be useful in the paper.

Definition 3.2 [Shamir's scheme [23]]: Let $F=\mathrm{GF}(q)$, where $q>k$, and let $\omega_1,\ldots,\omega_k$ be distinct nonzero elements of F. To t-privately share a secret $s\in F$, the dealer chooses t random elements $a_1,\ldots,a_t$, which together with the secret s define a univariate polynomial $p(Y)\overset{\text{def}}{=}a_tY^t+a_{t-1}Y^{t-1}+\ldots+a_1Y+s$. Observe that $p(0)=s$. The share of the j-th player is $p(\omega_j)$. This share is a linear combination of the random inputs and the secret. Each set of at least $t+1$ players can recover $p(Y)$ by interpolation, and hence can also reconstruct $s=p(0)$. On the other hand, every set of t players learns nothing on s from their shares.

Definition 3.3 [The CNF scheme [17]]: This scheme may work over any finite field (in fact, over any finite group), and proceeds as follows. To t-privately share a secret $s\in F$:

- *Additively* share s into $\binom{k}{t}$ shares, each labeled by a different set from $\binom{[k]}{t}$; that is, $s=\sum_{T\in\binom{[k]}{t}}r_T$, where the shares r_T are otherwise-random field elements. (Equivalently, all r_T except one may be chosen uniformly at random, and the last is determined so that they all sum up to s.)
- Distribute to each player P_j all shares r_T such that $j\notin T$.

The t-privacy of the above scheme follows from the fact that every t players miss exactly one additive share r_T (namely, the one labeled by their index set). Every set of $t+1$ players views all shares, thus, can reconstruct the secret. The share of each party consists of $\binom{k-1}{t}$ field elements.[6]

4 The Meta-Construction

In this section we describe our construction in terms of its abstract general components, and specify some useful instantiations for each of these components. In the next section several combinations of these instantiations will be used for obtaining different families of PIR protocols.

4.1 Building Blocks

Three parameters which are common to all of our constructions are: (1) a finite *field* F, (2) a *degree* parameter d, and (3) an *encoding length* parameter m. The database x will always be viewed as a vector in F^n. Some variants of our construction will use an additional *block length* parameter ℓ.

All variants of our construction (as well as previous PIR protocols) can be cast in terms of the following abstract building blocks:

Linear space of polynomials. Let $V \subseteq F_d[Y_1, \ldots, Y_m]$ be a linear space of degree-d m-variate polynomials such that $\dim(V) \geq n$. The three most useful special cases are:

V1: The space $F_d[Y_1, \ldots, Y_m]$ where $F =$ GF(2); in this case, m and d must satisfy $\Lambda(m,d) \geq n$.

V2: The space $F_d[Y_1, \ldots, Y_m]$ where $|F| > d$; in this case, m and d must satisfy $\binom{m+d}{d} \geq n$.

V3: The linear subspace of $F_d[Y_1, \ldots, Y_m]$ such that $F = \mathrm{GF}(2)$ and V is spanned by the following basis of monomials. Let ℓ be an additional *block length* parameter, and let $m = \ell d$. We label the m indeterminate by $Y_{g,h}$, where $g \in [d]$ and $h \in [\ell]$. The basis of V will include all monic monomials containing exactly one indeterminate from each block, i.e., all monomials of the form $Y_{1,h_1} Y_{2,h_2} \cdots Y_{d,h_d}$. Since the number of such monomials is ℓ^d, the restriction on the parameters in this case is $\ell^d \geq n$.

Low-degree encoding. A *low-degree encoding* (with respect to the polynomial space V) is a mapping $E : [n] \to F^m$ satisfying the following requirement: There exist m-variate polynomials $p_1, p_2, \ldots, p_n \in V$ such that $\forall i, j \in [n], p_i(E(j))$ is 1 if $i = j$ and is 0 otherwise. By elementary linear algebra, $\dim(V) \geq n$ is a necessary and sufficient condition for the existence of such an encoding. Given a low-degree encoding E and polynomials $p_1, p_2, \ldots, p_n$ as above, we will associate with each database $x \in F^n$ the polynomial $p_x \in V$ defined by $p_x(Y_1, \ldots, Y_m) = \sum_{i=1}^n x_i p_i$. In the above x is fixed, and $x_1, \ldots, x_n$ are fixed coefficients (and not variables). Note that $p_x(E(i)) = x_i$ for every $i \in [n]$ and $x \in F^n$.

With each of the above linear spaces we associate a natural low-degree encoding.[7] Specifically, we use:

[6] This scheme has also been referred to as *replication-based* secret-sharing [15, 13]. It may be viewed as a special case of the formula-based secret-sharing construction from [7], obtained by using the canonic CNF representation of the threshold function.

[7] Since the *existence* of an appropriate encoding is implied by dimension arguments [11, Lemma 6], the specific encoding being employed will usually not matter. In some cases, however, the encoding can make a difference. Such a case is discussed in Section 5.3.

E1: Let $E(i)$ be the i-th vector in GF$(2)^m$ of Hamming weight *at most d*. A proof of validity of this encoding, that is, the existence of appropriate polynomials $p_1, \ldots, p_n$, appears in Appendix B.

E2: Let $\omega_0, \ldots, \omega_d$ be distinct field elements. Then, $E(i)$ is the i-th vector of the form $(\omega_{f_1}, \ldots, \omega_{f_m})$ such that $\sum_{j=1}^{m} f_j \leq d$. A proof of the validity of this encoding may be found in [11, Lemma 6].

E3: Let $(i_1, \ldots, i_d)$ be the d-digit base-ℓ representation of i (that is, $i = \sum_{j=1}^{d} i_j \ell^{j-1}$). Then, $E(i)$ is a concatenation of the length-ℓ unit vectors $e_{i_1}, e_{i_2}, \ldots, e_{i_d}$. The validity of this encoding follows by letting $p_i = Y_{1,i_1} \cdot \ldots \cdot Y_{d,i_d}$.

Linear secret-sharing scheme. Denoted by L. The following t-private schemes will be useful.

L1 : The t-private CNF construction from Definition 3.3.

L2 : The t-private Shamir construction from Definition 3.2.

L3 : A slight optimization of the CNF construction, whose details will be discussed in Section 7.

Simultaneous messages communication protocol (abbreviated SM protocol). The fourth and most crucial building block is a protocol for the following promise problem, defined by the instantiations of the previous components V, E, and L. The problem generalizes the scenario described in Section 2. The protocol, denoted P, involves a user $\mathcal{U}$ and k servers $\mathcal{S}_1, \ldots, \mathcal{S}_k$.

- *User's inputs:* Valid L-shares $y^1, \ldots, y^k$ of a point $y \in F^m$. (That is, the k vectors y^j can be obtained by applying L to each entry of y, and collecting the shares of each player.) Moreover, it may be useful to rely on the following additional promise: $y = E(i)$ for some $i \in [n]$. Most of the protocols constructed in this paper do not make use of this additional promise.

- *Servers' inputs:* All k servers hold a polynomial $p \in V$. In addition, each $\mathcal{S}_j$ holds the share vector y^j.

- *Communication pattern:* Each server $\mathcal{S}_j$ sends a single message to $\mathcal{U}$ based on its inputs p, y^j. We let β_j denote a bound on the length of the message sent by $\mathcal{S}_j$.

- *Output:* $\mathcal{U}$ should output $p(y)$.

In Section 5 we will describe our constructions of SM protocols P corresponding to some choices of the space of polynomials V, the low degree encoding E, and the linear secret sharing scheme L.

4.2 Putting the Pieces Together

A 4-tuple (V, E, L, P) instantiating the above 4 primitives uniquely defines a PIR protocol PIR(V, E, L, P). The protocol proceeds as follows.

- $\mathcal{U}$ lets $y = E(i)$.

- $\mathcal{U}$ shares y according to L among the k servers. Let y^j denote the vector of shares received by $\mathcal{S}_j$.

- Each server $\mathcal{S}_j$ lets $p = p_x$, and sends a message to $\mathcal{U}$ as specified by protocol P on inputs (p, y^j).

- $\mathcal{U}$ reconstructs $x_i = p(y)$ by applying the reconstruction function specified in P to $y^1, \ldots, y^k$ and the k messages it received.

The following lemma summarizes some easily verifiable properties of the above protocol.

Lemma 4.1 $\mathrm{PIR}(V, E, L, P)$ *is a (linear) t-private k-server PIR protocol, in which the user sends $m\ell_j$ field elements to each server $\mathcal{S}_j$ and receives β_j bits in return from each server (where ℓ_j is the share size defined by L and β_j is the length of message sent by $\mathcal{S}_j$ in P).*

Note that the only information that a server gets is a share of the encoding $E(i)$; the t-privacy of the secret sharing scheme ensures that a collusion of t servers learns nothing on i. For the query complexity, recall that $y = E(i) \in F^m$ and the user shares each of the m coordinates of y independently. Thus, the share size of server $\mathcal{S}_j$ is $m\ell_j$, where ℓ_j is the share size defined by L for sharing one coordinate (field element).

Some perspective concerning a typical choice of parameters is in place. In the typical case where k is viewed as a constant, all ℓ_j are also constant, and so the query complexity of $\mathrm{PIR}(V, E, L, P)$ is $O(m)$. If d is constant then, for any of the three vector spaces $\mathbf{V1}, \mathbf{V2}, \mathbf{V3}$, letting $m = O(n^{1/d})$ suffices to meet the dimension requirements. Thus, when both d, k are constants, the length of the queries in $\mathrm{PIR}(V, E, L, P)$ is $O(n^{1/d})$ and the length of the answers is determined by P.

In principle, the SM component in our construction could be replaced by a more general interactive protocol. However, there is yet no evidence that such an additional interaction may be helpful. Moreover, in defining an interactive variant of the fourth primitive one would have to take special care that the privacy requirement is not violated by the interaction. In the current non-interactive framework, the privacy requirement is automatically taken care of.

5 Simultaneous Messages Protocols

We next describe SM protocols corresponding to useful combinations of V, E, and L. These may be viewed as the core of the PIR protocol. Some extensions are described in Section 7.

5.1 Protocol $\mathbf{P1}$

Protocol $\mathbf{P1}$ will serve as our default protocol. It may be viewed as a natural generalization of the protocol from [3]. The ingredients of this protocols are the polynomial space $\mathbf{V1} = F_d[Y_1, \ldots, Y_m]$ where $F = \mathrm{GF}(2)$, the encoding $\mathbf{E1}$ which encodes i as a vector in $\mathrm{GF}(2)^m$ of Hamming weight *at most d*, and the linear secret sharing $\mathbf{L1}$ which is the CNF sharing.

Lemma 5.1 *For $V = \mathbf{V1}$, $E = \mathbf{E1}$, and $L = \mathbf{L1}$, there exists an SM protocol $\mathbf{P1}$ with message complexity $\beta_j = \Lambda(m, \lfloor dt/k \rfloor)\binom{k-1}{t-1}^{\lfloor dt/k \rfloor}$.*

Proof: Before describing the protocol with the specified complexity we will need some notation. Let $y = \sum_T y_T$ be an additive sharing of y induced by the CNF sharing, such that the input y^j of $\mathcal{S}_j$ is $(y_T)_{j \notin T}$. The servers' goal is to communicate $p(y) = p(\sum_T y_T)$ to $\mathcal{U}$ using at most $\Lambda(m, \lfloor dt/k \rfloor)\binom{k-1}{t-1}^{\lfloor dt/k \rfloor}$ communication bits per server.

Define a $\binom{k}{t}m$-variate polynomial $q\left(Y_{T,b} : T \in \binom{[k]}{t}, b \in [m]\right) \overset{\text{def}}{=} p\left(\sum_{T \in \binom{[k]}{t}} Y_{T,1}, \ldots, \sum_{T \in \binom{[k]}{t}} Y_{T,m}\right)$. The degree of q is the same as the degree of p. We consider the explicit representation of q as the sum of monomials. We claim that for every monomial $Y_{T_1,b_1} Y_{T_2,b_2} \ldots Y_{T_{d'},b_{d'}}$ of degree $d' \leq d$ there exist some

$j \in [k]$ such that at most $\lfloor dt/k \rfloor$ variables $Y_{T,b}$ with $j \in T$ appear in the monomial. Consider the multi-set $T_1 \cup T_2 \cup \ldots \cup T_{d'}$. This multi-set contains $d't \leq dt$ elements, thus there must be some $j \in [k]$ that appears at most $\lfloor dt/k \rfloor$ times in the multi-set.

We partition the monomials of q to k polynomials $q_1, \ldots, q_k$ such that q_j contains only monomials in which the number of the variables $Y_{T,b}$ with $j \in T$ is at most $\lfloor dt/k \rfloor$. Each monomial of q is in exactly one polynomial q_j, therefore, $q\left(Y_{T,b} : T \in \binom{[k]}{t}, b \in [m]\right) = \sum_{j=1}^{k} q_j \left(Y_{T,b} : T \in \binom{[k]}{t}, b \in [m]\right)$.

We are now ready to describe the protocol $\mathbf{P1}$. Each server $\mathcal{S}_j$ substitutes the values of the variables that it knows in q_j to obtain the polynomial:

$$\hat{q}_j \left(Y_{T,b} : T \in \binom{[k]}{t}, j \in T, b \in [m]\right)$$
$$\stackrel{\text{def}}{=} \quad q\left(y_{T,b} : T \in \binom{[k]}{t}, j \notin T, b \in [m],\ Y_{T,b} : T \in \binom{[k]}{t}, j \in T, b \in [m]\right).$$

The message of server $\mathcal{S}_j$ is the list of all coefficients of $\hat{q}_j$. The user, who knows the assignments to all variables, computes $\hat{q}_j(y_{T,b} : T \in \binom{[k]}{t}, j \in T, b \in [m])$ and sums these k values to obtain

$$\begin{aligned}\sum_{j=1}^{k} \hat{q}_j \left(y_{T,b} : T \in \binom{[k]}{t}, j \in T, b \in [m]\right) &= \sum_{j=1}^{k} q_j \left(y_{T,b} : T \in \binom{[k]}{t}, b \in [m]\right) \\ &= q\left(y_{T,b} : T \in \binom{[k]}{t}, b \in [m]\right) \ = \ p(y_1, \ldots, y_m).\end{aligned}$$

Thus, the user reconstructs the desired value.

We next analyze the message complexity of the protocol. Recall that $\hat{q}_j$ is a degree-$\lfloor dt/k \rfloor$ multivariate polynomial with $m\binom{k}{t-1}$ variables. By the definition of q, not all monomials are possible: no monomial contains two variables $Y_{T_1,b}$ and $Y_{T_2,b}$ for some $b \in [m]$ and $T_1 \neq T_2$. Thus, to describe a possible monomial we need, for some $w \in \{0, \ldots, \lfloor dt/k \rfloor\}$, to choose w indices in $[m]$ and w sets of size t that contain j. Therefore, the number of possible monomials of $\hat{q}_j$ is at most $\sum_{w=0}^{\lfloor dt/k \rfloor} \binom{m}{w} \binom{k-1}{t-1}^w \leq \Lambda(m, \lfloor dt/k \rfloor) \binom{k-1}{t-1}^{\lfloor dt/k \rfloor}$. Since each coefficient is from GF(2), the communication is as promised. $\square$

Remark 5.2 To define the protocol $\mathbf{P1}$ precisely, one should specify how to partition the polynomial q to the polynomials $q_1, \ldots, q_k$ (that is, what to do with monomials that can be assigned to more than one polynomial). More generally, instead of *partitioning* the monomials, one may break q to any k polynomials which add up to q and satisfy the requirement utilized above. However, a partition of the monomials as described above is sufficient for our purposes.

Remark 5.3 An important question is whether the communication complexity of the protocol $\mathbf{P1}$ is optimal. Using a similar lower bound technique to the one used in [3], it is possible to show that under the promise of the particular (low-weight) encoding $\mathbf{E1}$, the communication complexity of $\mathbf{P1}$ is indeed optimal up to a constant factor [16]. A similar statement holds for the the encoding $\mathbf{E3}$. It is not clear, however, whether this is true for an arbitrary encoding.

5.2 Protocol $\mathrm{P2}$

Protocol $\mathbf{P2}$ will be mainly useful for the construction of efficient PIR protocols with short answers (see Section 6.2). Unlike protocol $\mathbf{P1}$, which can be used with any combination of the parameters k, d, t, the applicability of $\mathbf{P2}$ will be restricted to the case $k > dt$. That is, $k = dt+1$ is the minimal sufficient number of servers. The first part of the following lemma is implicit in [4, 5, 10] and a special case of the second part is implicit in [11, 12].

Lemma 5.4 *For $V = \mathbf{V2}$, $E = \mathbf{E2}$, and $L = \mathbf{L2}$, and assuming that $k > dt$ and $|F| > k$, there exists an SM protocol $\mathbf{P2}$ in which each server sends a single field element. Moreover, given the promise that $p(y) \in F'$ for some subfield F' of F, it suffices for each server to send a single element of F'.*

Proof: Recall that $k > dt$ and $|F| > k$. All servers hold a degree-d m-variate polynomial $p(Y)$ over F, and each server $\mathcal{S}_j$ holds a Shamir-share $y^j \in F^m$ of a vector $y = (y_1, \ldots, y_m) \in F^m$. Recall that in the definition of Shamir's scheme, a field element s is shared by evaluating a degree-t polynomial, whose free coefficient is s, on k distinct points ω_j, where $j \in [k]$. That is, the user chooses m univariate polynomials $p_1, \ldots, p_m$ each one of degree t such that $y = (p_1(0), \ldots, p_m(0))$ and $y^j = (p_1(\omega_j), \ldots, p_m(\omega_j))$ for $j \in [k]$. The goal of the servers is to communicate the value $p(y)$ to $\mathcal{U}$ using a single field element per server (or a single element of a subfield F' given the promise that $p(y) \in F'$).

We first describe the protocol in which each server sends a single *field element*: the message sent by $\mathcal{S}_j$ in Protocol $\mathbf{P2}$ is $m_j = p(y^j)$. Thus, the points (ω_j, m_j) lie on the degree-dt univariate polynomial $q(Z) \stackrel{\text{def}}{=} p(p_1(Z), \ldots, p_m(Z))$. Furthermore, $q(0) = p(y)$. Since $k > dt$, the user can reconstruct $q(Z)$ by interpolation and evaluate $q(0) = p(y)$, that is, there exist "interpolation coefficients" $c_1, \ldots, c_k$ such that $p(y)$ can be reconstructed from the messages $m_1, \ldots, m_k$ by applying the fixed linear combination $\sum c_j m_j$.

We now describe how to reduce the messages to be elements of a subfield F' given the promise that $p(y) \in F'$. Suppose first that each server modifies its message to $m'_j = c_j m_j$. Then, $\mathcal{U}$ may reconstruct $p(y)$ by *adding* the k messages m'_j it receives. To reduce the answers to a single bit, let $H : F \to F'$ be a homomorphism such that $H(\alpha) = \alpha$ for all $\alpha \in F'$. Since H is a homomorphism, for any $a, b \in F$, it holds that $H(a+b) = H(a) + H(b)$. Now, instead of sending m'_j, the j-th server will send the F'-element $H(m'_j)$. From the properties of H and from the promise that $p(y) \in F'$ we may conclude that by adding the k answers (over F') the correct value $p(y)$ is obtained. □

A special case of interest is when $F' = \mathrm{GF}(2)$ and F is a sufficiently large extension field of F'. In this case, each message in the SM protocol consists of a single bit. Note that when $k > dt$ the messages of $\mathbf{P1}$ are also one-bit long. However, for this choice of parameters $\mathbf{P2}$ is superior to $\mathbf{P1}$ in that it relies on the (ideal) Shamir secret-sharing scheme, whereas $\mathbf{P1}$ relies on the highly redundant CNF-scheme.

We do not know (and view it as an interesting problem) to prove nontrivial upper or lower bounds on the SM complexity of the $\mathbf{P2}$ setting when $k \le dt$. The lower bound proof technique of [3] and its generalizations from [22, 19] do not seem to imply good bounds in this setting. Good upper bounds will enable to eliminate $\binom{k}{t}$ factors in our t-private constructions.

5.3 Protocol P3

Special cases of the protocol $\mathbf{P3}$ are implicit in the 2-server PIR construction from [10] and its k-server generalization from [15]. A useful feature of this protocol is that it allows the user to compute his output by probing a small number of bits from the received messages. We will only formulate this protocol for the 1-private case. Restricted generalizations to t-privacy may be obtained, using the approach of [15]. However, unlike the previous protocols, we do not know a "smooth" generalization to t-privacy.

Lemma 5.5 *For $V = \mathbf{V3}$, $E = \mathbf{E3}$, and $L = \mathbf{L1}$, there exists an SM protocol $\mathbf{P3}$ with message complexity $\beta_j = \ell^{\lfloor d/k \rfloor}\binom{d}{\lfloor d/k \rfloor}$ such that the user needs to read only $\binom{d}{\lfloor d/k \rfloor}$ bits from each message.*

Proof: We first specify the inputs to the protocol. Let p be a polynomial in $\mathbf{V3}$, and let $y \in \{0,1\}^{\ell d}$ be an encoding of some index i under the **E3** encoding, that is, $y = w_1 \circ \ldots \circ w_d$, where $\circ$ denote concatenation and w_g is some unit vector from the space $\{0,1\}^{\ell}$ for each $g \in [d]$. Furthermore, let $w_g = \sum_{j=1}^{k} w_{j,g}$ be an additive sharing of w_g induced by the 1-private CNF sharing. The input y^j of $\mathcal{S}_j$ is $(w_{a,g} : a \in [k] \backslash \{j\}, g \in [d])$. The servers' goal is to communicate $p(w_1 \circ \ldots \circ w_j) = p((\sum_{a=1}^{k} w_{a,1}) \circ \ldots \circ (\sum_{a=1}^{k} w_{a,d}))$ to $\mathcal{U}$ efficiently such that the user needs to read only few bits from each answer in order to reconstruct $p(y)$.

The fact that $p \in \mathbf{V3}$, that is, each monomial in p contains one variable from each block, implies the following fact:

Fact 5.6 $p(w_1 \circ \ldots \circ w_d) = p((\sum_{a=1}^{k} w_{a,1}) \circ \ldots \circ (\sum_{a=1}^{k} w_{a,d})) = \sum_{j_1 \in [k], \ldots, j_d \in [k]} p(w_{j_1,1} \circ \ldots \circ w_{j_d,d})$.

We partition the responsibility for evaluating each of these k^d values to the servers, such that server $\mathcal{S}_j$ is responsible for evaluating the values $p(w_{j_1,1} \circ \ldots \circ w_{j_d,d})$ for vectors in which $j_g = j$ for at most $\lfloor d/k \rfloor$ indices g. The server does not know the values of at most $\lfloor d/k \rfloor$ coordinates of such a vector, and it tries to guess them in a clever way. This is done using the knowledge that each w_g is unit vector. Therefore, $w_{j,g} = e_i - \sum_{a \in [k] \backslash \{j\}} w_{a,g}$ for some unit vector $e_i \in \{0,1\}^{\ell}$, and hence there are ℓ possible values of $w_{j,g}$ for each $g \in [d]$. This already implies a protocol where each server sends $k^d \ell^{\lfloor d/k \rfloor}$ bits: There are k^d values of p that need to be evaluated. For each such value $p(w_{j_1,1} \circ \ldots \circ w_{j_d,d})$, the server that is responsible for it, evaluates p using the coordinates that it knows and each of the guesses for the coordinates it does know; there are at most $\ell^{\lfloor d/k \rfloor}$ such guesses. The server sends this list of values to the user. The user, who knows all the values of the shares, sums the value of p evaluated at the k^d true values.

We next optimize this protocol. Instead of sending a list of length $\ell^{\lfloor d/k \rfloor}$ for every one of the k^d values of p, each server sends only $\binom{d}{\lfloor d/k \rfloor}$ lists, one for each set $A \in \binom{[d]}{\lfloor d/k \rfloor}$. This is done by assigning each value $p(w_{j_1,1} \circ \ldots \circ w_{j_d,d})$ which is under the responsibility of $\mathcal{S}_j$ to some set $A \in \binom{[d]}{\lfloor d/k \rfloor}$ such that $\{g : j_g = j\} \subseteq A$. Now, each of the $\ell^{\lfloor d/k \rfloor}$ possible "guesses" of $(w_{j_g,g})_{g \in A}$ allows $\mathcal{S}_j$ to uniquely determine all values $p(w_{j_1,1} \circ \ldots \circ w_{j_d,d})$ which are assigned to A, and therefore also their sum. Thus, $\mathcal{S}_j$ can send, for each of the $\ell^{\lfloor d/k \rfloor}$ guesses and each set A, a single bit containing this sum. Finally, note that the user needs to read only one bit from each list (correspond to the correct guess), giving a total of $\binom{d}{\lfloor d/k \rfloor}$ bits. □

In the protocol $\mathbf{P3}$ we heavily rely on the promise that y is an encoding under **E3** of some index i. This should be contrasted with the protocols $\mathbf{P1}$ and $\mathbf{P2}$ where we did not rely on any properties of the encoding.

6 Families of PIR Protocols Obtained via the Meta-Construction

We next describe and analyze several families of PIR protocols which are special cases of the meta-construction.

6.1 Main Family

Our main family of PIR protocols uses $\mathbf{V1}$, **E1**, $\mathbf{L1}$, and $\mathbf{P1}$. Protocols from this family yield our main improvements to the known upper bounds. We start with the general result, and then consider some interesting special cases.

Theorem 6.1 *Let m and d be positive integers such that $\Lambda(m,d) \geq n$. Then, for any k and t, where $k \geq 2$ and $1 \leq t < k$, there exists a t-private k-server PIR protocol with $\binom{k-1}{t}m$ query bits and $\Lambda(m, \lfloor dt/k \rfloor)\binom{k-1}{t-1}^{\lfloor dt/k \rfloor}$ answer bits per server.*

Proof: The condition $\Lambda(m,d) \geq n$ is sufficient (and necessary) to guarantee the existence of an encoding $\mathbf{E1}$ as required. We use the $\mathbf{L1}$ secret sharing scheme, and the length of each share, i.e., the query complexity, is $\binom{k-1}{t}m$. Finally, we use $\mathbf{P1}$, and by Lemma 5.1 the length of the answer of each server is $\Lambda(m, \lfloor dt/k \rfloor)\binom{k-1}{t-1}^{\lfloor dt/k \rfloor}$. Thus, by Lemma 4.1, the communication complexity is as promised. □

The first interesting case, in which we solve an open problem of [15], minimizes the total communication.

Corollary 6.2 *Let k and t, where $k \geq 2$ and $1 \leq t < k$, be integers. There exists a t-private k-server PIR protocol with total communication $O(\frac{k^2}{t}\binom{k}{t}n^{1/\lfloor(2k-1)/t\rfloor})$.*

Proof: To guarantee that $\Lambda(m,d) \geq n$ it suffices to let $m = O(dn^{1/d})$. Fix $d = \lfloor(2k-1)/t\rfloor$, so that $\lfloor dt/k \rfloor = 1$. Now we can apply Theorem 6.1 to obtain a protocol whose query length is $\binom{k-1}{t}m$ per server and the answer length is $\Lambda(m,1)\binom{k-1}{t-1}^1$ per server. Therefore, the total communication is as promised. □

The next corollaries contain an exact analysis for the 2-server case, and a somewhat cruder analysis for the 1-private k-server case.

Corollary 6.3 *There is a 1-private 2-server PIR protocol with total communication $4(6n)^{1/3}+2 \approx 7.27n^{1/3}$.*

Proof: Fix $m = (6n)^{1/3}$ and $d = 3$. First notice that $\Lambda(m,3) > m^3/6 = n$. Thus, we can apply Theorem 6.1 to obtain a protocol whose query length is m bits per server and the answer length is $\Lambda(m, \lfloor 3/2 \rfloor) = m+1$ bits per server. Therefore, the total communication is $4m+2 = 4(6n)^{1/3}+2 \approx 7.27n^{1/3}$. □

In comparison, the communication in the best previously known 2-server protocol [10] is roughly $12n^{1/3}$.

Corollary 6.4 *For every $k > 2$ there exists a 1-private k-server PIR protocol with total communication complexity of $k^2((2k-1)!n)^{1/(2k-1)} + k + k^3 = O(k^3 n^{1/(2k-1)})$.*

Proof: Fix $m = ((2k-1)!\,n)^{1/(2k-1)} + k$ and $d = 2k-1$. By Eq. (1) (appearing in Appendix A), $\Lambda(m,d) \geq n$. Thus, we can apply Theorem 6.1 to obtain a protocol whose query length is $(k-1)m$ per server and the answer length is $\Lambda(m, \lfloor(2k-1)/k\rfloor) = \Lambda(m,1) = m+1$ per server. Therefore, the total communication is $k^2m + k = k^2((2k-1)!\,n)^{1/(2k-1)} + k^3 + k = O(k^3n^{1/(2k-1)})$. □

This protocol should be compared to the protocol of [15] whose communication complexity is $k^2(2k-1)n^{1/(2k-1)}$. Thus, when k is large, our protocol improves over the protocol of [15] by the constant factor $1/\alpha_d$ (where α_d is the constant defined after Eq. (1)), which tends to e as k grows.

Another interesting case, discussed and used in [6], is when queries are short, i.e., of length $O(\log n)$; in this case we use denser assignments in which the *relative* weight d/m is fixed as some constant θ, where $0 < \theta \leq 1/2$. Substituting $m = (1/H(\theta) + o(1))\log n$ and $d = \lfloor \theta m \rfloor$ in Theorem 6.1 we obtain:

Corollary 6.5 *For any integers k, t, where $k \geq 2$ and $1 \leq t < k$, and any constant $0 < \theta \leq 1/2$, there exists a t-private k-server PIR protocol with $\binom{k-1}{t}(1/H(\theta) + o(1))\log n$ query bits and*

$$n^{(H(\theta t/k)+\theta\frac{t}{k}\log\binom{k-1}{t-1})/H(\theta)+o(1)}$$

answer bits per server. When $t = 1$, we get $(k-1)(1/H(\theta)+o(1))\log n$ query bits and $n^{H(\theta/k)/H(\theta)+o(1)}$ answer bits.

Since $\lim_{\theta\to 0}\frac{H(\theta t/k)}{H(\theta)} = t/k$ and $\lim_{\theta\to 0}\frac{\theta}{H(\theta)} = 0$ we get:

Corollary 6.6 *For any constant integers k, t, where $k \geq 2$ and $1 \leq t < k$, and any constant $\epsilon > 0$, there exists a t-private k-server PIR protocol with $O(\log n)$ query bits and $O(n^{t/k+\epsilon})$ answer bits.*

As shown by [19], a 1-private k-server PIR protocol with query length α and answer length β can be turned into a locally decodable code of length $k \cdot 2^{\alpha}$ over the alphabet $\Sigma = \{0,1\}^{\beta}$: A string $x \in \{0,1\}^n$ is encoded by concatenating the answers of all servers on all possible queries, where x is viewed as the database. If $\alpha = O(\log n)$, then the code length is polynomial. Thus, by substituting $t = 1$ in Corollary 6.6 and applying the above transformation we get:

Corollary 6.7 *For any constant integer $k \geq 2$ and constant $\epsilon > 0$, there exists a family $C(n)$ of* polynomial-length *(k, δ_k, ρ_k)-locally decodable codes over $\Sigma(n) = \{0,1\}^{\beta(n)}$, where $\beta(n) = O(n^{1/k+\epsilon})$, for some positive constants δ_k, ρ_k.*

6.2 Boolean Family

In this section we derive the construction of the most efficient known PIR protocols with a single answer bit per server using $\mathbf{V2}$, $\mathbf{E2}$, $\mathbf{L2}$, and $\mathbf{P2}$. These protocols, appearing in [11] (see also [12]) optimize similar protocols from [4, 5, 10] in which each answer consists of a single element from a moderately sized field. While the asymptotic communication complexity of protocols from this family is worse than that of the best unrestricted protocols, these protocols have found various applications. In particular they imply: (1) the most efficient constructions of binary locally decodable codes known to date; (2) very efficient PIR protocols for retrieving large records or "streams" of data; (3) PIR protocols with optimal amount of total *on-line* communication (see [11]); (4) PIR protocols with poly-logarithmic amount of *on-line* work by the servers (see [6]).

Theorem 6.8 (Implicit in [11]) *Let m and d be positive integers such that $\binom{m+d}{d} \geq n$. Then, for any $t \geq 1$, there exists a t-private k-server PIR protocol with $k = dt + 1$ servers, $\lceil \log(k+1) \rceil m$ query bits per server, and a* single *answer bit per server.*

Proof: A PIR protocol as required is obtained by letting $E = \mathbf{E2}, V = \mathbf{V2}, L = \mathbf{L2}$, and $P = \mathbf{P2}$, where F is GF$(2^{\lceil \log(k+1) \rceil})$ – the smallest GF(2)-extension with at least $k+1$ elements, and the subfield F' used by $\mathbf{P2}$ is GF(2). □

Corollary 6.9 *For any constant $d, t \geq 1$ there is a t-private PIR protocol with $k = dt + 1$ servers, $O(n^{1/d})$ query bits, and a single answer bit per server.*

6.3 Cube Family

Our last family of protocols generalizes the 2-server protocol from [10] and its k-server generalization from [15]. It relies on $\mathbf{V3}$, $\mathbf{E3}$, $\mathbf{L1}$, and $\mathbf{P3}$ as building blocks. The communication in these protocols is not optimal, but it has the property that the user needs to read fewer bits from the answers. These protocols have the interpretation of utilizing the "combinatorial cubes" geometry which was first used in [10]. Again, we start with the general result, and then consider interesting special cases.

Theorem 6.10 (Generalizing [10, 15]) *Let d and ℓ be positive integers such that $\ell^d \geq n$. Then, for any $k \geq 2$ there exists a 1-private k-server PIR protocol with $(k-1)d\ell$ query bits per server and $\ell^{\lfloor d/k \rfloor}\binom{d}{\lfloor d/k \rfloor}$ answer bits per server, in which the user needs to read only $\binom{d}{\lfloor d/k \rfloor}$ bits from each answer.*

Proof: The condition $\ell^d \geq n$ guarantees the existence of an encoding $\mathbf{E3}$ as required. We use the 1-private CNF secret-sharing scheme, thus the length of each share, i.e., the query complexity, is $(k-1)d\ell$. Finally, we use $\mathbf{P3}$, therefore, by Lemma 5.5, the length of the answer of each server and the number of bits the user needs to read from each answer is as promised. □

The first corollary, which already appears in [10, 15], minimizes the total communication.

Corollary 6.11 ([10, 15]) *Let $k \geq 2$ be an integer. There exists a 1-private k-server PIR protocol with total communication complexity of $O(k^3 n^{1/(2k-1)})$ such that the user reads only $2k-1$ bits from each answer.*

Proof: Fix $d = 2k-1$ and $\ell = n^{1/d}$. First notice that $\lfloor d/k \rfloor = 1$. By Theorem 6.10, we obtain a protocol whose query length is $(k-1)d\ell = O(k^2 n^{1/(2k-1)})$ per server, the answer length is $d\ell = O(kn^{1/(2k-1)})$ per server, and the user needs to read $d = 2k-1$ bits from each answer. □

The second corollary, which is utilized in [6], considers the case where the query length is logarithmic.

Corollary 6.12 *Let $k \geq 2$ be an integer and $\delta < 1$. There exists a 1-private k-server PIR protocol with query complexity $O(k2^{1/\delta}\delta \log n)$ and answer complexity $O(n^{1/k+H(1/k)\delta})$ in which the user reads only $O(n^{H(1/k)\delta})$ bits from each answer.*

Proof: Fix $d = \delta \log n$, and $\ell = n^{1/d} = 2^{1/\delta}$. By Theorem 6.10, we obtain a protocol whose query length is $(k-1)d\ell = O(k2^{1/\delta}\delta \log n)$ per server, the answer length is $\ell^{d/k}\Lambda(d, \lfloor d/k \rfloor) = O(n^{1/k+H(1/k)\delta})$ per server (this follows from Eq. (1) appearing in Appendix A), and the user needs to read $\Lambda(d, \lfloor d/k \rfloor) = O(n^{H(1/k)\delta})$ bits from each answer. □

7 Optimized CNF Secret Sharing

In the families of PIR protocols we described up to now we used two secret-sharing schemes: Shamir's scheme and the CNF scheme. Shamir's scheme has the smallest shares possible – the size of each share is the maximum of the secret size and the logarithm of the number of players. In contrast, the size of the each share in the CNF scheme is $\binom{k-1}{t}$ times the size of the secret. In some sense, the CNF sharing gives more redundancy to the share-holders. The protocols of [15] and our main family of protocols (described in Section 6.1) exploit this redundancy to improve the communication complexity with respect to the protocols of [10]. This raises the question if we can maintain the better communication complexity without paying the penalty of the redundancy, i.e., with shorter queries. This penalty can be quite big in the t-private protocols.

We indicate that some savings are possible. Specifically, we construct a secret-sharing scheme $\mathbf{L3}$, whose share complexity improves on that of the CNF scheme by roughly a factor of $t+1$ when $k \gg t$; yet, an SM protocol with identical communication to that of $\mathbf{P1}$ (and significantly better computation) can be based on $\mathbf{L3}$. This results in a similar improvement to the query length of our main family. An additional feature of the optimized construction is a significant reduction in the *computation* required by the servers. For instance, in the 1-private case its dependence on k is reduced from k^{2k-1} to roughly $k!$. Our construction generalizes an optimization which was suggested in [15] for the 1-private case, and significantly improves its computational complexity (the dependence on k is reduced from $2^{O(k^2)}$, which is even worse than the k^{2k-1} dependence of our main family, to roughly $k!$).

Definition 7.1 [1-Private optimized CNF scheme] This scheme may work over any finite field (in fact, over any finite group), and proceeds as follows. To 1-privately share a secret $s \in F$:

- *Additively* share s into k shares $r_1, \ldots, r_k$; that is, $s = \sum_{i=1}^{k} r_i$, where the shares r_i are otherwise-random field elements.
- Define $z_i = \sum_{j=i+1}^{k} r_j$.
- Distribute to each player P_j the shares $r_1, \ldots, r_{j-1}, z_j$.

The 1-privacy of the above scheme follows from the fact that each share contains less information than the share of the same party in Scheme $\mathbf{L1}$. On the other hand, every pair of parties, say P_{j_1}, P_{j_2} where $j_1 < j_2$, can reconstruct the secret s by computing $\sum_{j=1}^{j_1} r_j + z_{j_1}$ (the r_j's in the sum are held by P_{j_2}). The total share size summed over all parties in this scheme is $(k+2)(k-1)/2$ field elements. Asymptotically, this improves the total share size by a factor of 2 with respect to $\mathbf{L1}$, where the total share size is $k(k-1)$ field elements.

Lemma 7.2 *For $V = \mathbf{V1}$, $E = \mathbf{E1}$, and the 1-private optimized CNF sharing $\mathbf{L3}$, there exists an SM protocol $\mathbf{P4}$ with message complexity $\beta_j = \Lambda(m, \lfloor d/k \rfloor)$.*

Proof: We present an SM protocol $\mathbf{P4}$ as required. The description uses the notation of Protocol $\mathbf{P1}$ presented in the proof of Lemma 5.1; we consider only the case when $t = 1$, and denote $Y_{\{j\},b}$ by $Y_{j,b}$. In Protocol $\mathbf{P1}$ the servers hold an m-variate polynomial p, which defines a km-variate polynomial q. The monomials of q are partitioned to k polynomials $q_1, \ldots, q_k$ such that q_j contains only monomials in which the number of the variables $Y_{j,1}, \ldots, Y_{j,m}$ in each monomial is at most $\lfloor d/k \rfloor$. As discussed in Remark 5.2, in $\mathbf{P1}$ we did not specify the exact partition, that is, how to assign monomials that could be assigned to more than one polynomial q_j. For our next construction it is essential to require that a monomial is assigned to the polynomial q_j, where j is the smallest index such that the number of the variables $Y_{j,1}, \ldots, Y_{j,m}$ in the monomial is at most $\lfloor d/k \rfloor$.

Fix any j and consider the server $\mathcal{S}_j$. In the Protocol $\mathbf{P1}$ Server $\mathcal{S}_j$ substitutes the values of the variables that it knows in q_j to obtain the polynomial $\hat{q}_j \stackrel{\text{def}}{=} q(y_{1,1}, \ldots, y_{j-1,m}, Y_{j,1}, \ldots Y_{j,m}, y_{j+1,1}, \ldots, y_{k,m})$. We claim that the new shares of $\mathbf{L3}$ suffice for $\mathcal{S}_j$ to compute his original answer. Recall that every monomial in q is multi-linear, and furthermore, if a variable $Y_{j,\ell}$ appears in some monomial of q then for every $j' \neq j$ the variable $Y_{j',\ell}$ does not appear in that monomial. We define an equivalence relation between multi-linear monomials over the variables $Y_{1,1}, \ldots, Y_{1,m}, \ldots, Y_{k,1}, \ldots, Y_{k,m}$. (For every j we define a different equivalence relation.) We say that two monomials M_1 and M_2 are in the relation if:

1. For every $\ell \in [m]$ and $h \in \{1, \ldots, j\}$ the variable $Y_{h,\ell}$ appears in M_1 if and only if it appears in M_2, and
2. For every $\ell \in [m]$, there is some index $h_1 \in \{j+1, \ldots, k\}$ such that the variable $Y_{h_1,\ell}$ appears in M_1 if and only if there is some $h_2 \in \{j+1, \ldots, k\}$ such that the variable $Y_{h_2,\ell}$ appears in M_2, and
3. For every $h \in \{1, \ldots, j-1\}$ the monomial M_1 contains more than $\lfloor d/k \rfloor$ variables from the set $Y_{h,1}, \ldots, Y_{h,m}$. (By Item (1) this is also true for M_2.)

Also, if the condition in Item (3) does not hold for two monomials then they are in the relation. Notice that if a monomial M is assigned to q_j then all the monomials in its equivalent class appear in q and are assigned to q_j. By the multi-linearity, the sum of the monomials in each equivalence class can be expressed as a new monomial is the variables $Y_{1,1}, \ldots, Y_{1,m}, \ldots, Y_{j,1}, \ldots, Y_{j,m}$ and new variables $Z_{j,1}, \ldots, Z_{j,m}$ where $Z_{j,\ell} \stackrel{\text{def}}{=} \sum_{h=j+1}^{k} Y_{h,\ell}$. Furthermore, the polynomial q_j is the sum of the new monomials. By the definition of

$\mathbf{L3}$, server $\mathcal{S}_j$ knows the values $z_{j,1}, \ldots, z_{j,m}$ to be assigned to of $Z_{j,1}, \ldots, Z_{j,m}$. Thus, $\mathcal{S}_j$ can compute the coefficients of $\hat{q}_j$ and send them to the user as in Protocol $\mathbf{P1}$. □

We can generalize the optimized CNF scheme for arbitrary thresholds of privacy. We only describe the secret sharing scheme; the details of the appropriate SM protocol are the same as in the above described Protocol $\mathbf{P4}$.

Definition 7.3 [The t-private optimized CNF scheme] This scheme may work over any finite field (in fact, over any finite group), and proceeds as follows. To t-privately share a secret $s \in F$:

- *Additively* share s into $\binom{k}{t}$ shares, each labeled by a different set from $\binom{[k]}{t}$; that is, $s = \sum_{T \in \binom{[k]}{t}} r_T$, where the shares r_T are otherwise-random field elements.
- For every $j \in [k]$ and every $A \subseteq [j-1]$ such that $|A| \leq t$ and $|A| \geq t + j - k$ define
$$z_{j,A} = \sum_{T: j \notin T,\ T \cap [j-1] = A} r_T .$$
- Distribute to each player P_j all shares $z_{j,A}$ such that $A \subseteq [j-1]$ such that $|A| \leq t$ and $|A| \geq t+j-k$.

The t-privacy of the above scheme follows from the t-privacy of the $\mathbf{L1}$ scheme. (While not necessary for our purposes, it can also be verified that any set $C \subseteq [k]$ such that $|C| > t$ can reconstruct the secret.)

When $k \gg t$, this scheme improves the total share size by a factor of $t+1$ compared to $\mathbf{L1}$.

Acknowledgments. We thank Eyal Kushilevitz, Tal Malkin, Mike Saks, Yoav Stahl, and Xiaodong Sun for helpful related discussions.

References

[1] A. Ambainis. Upper bound on the communication complexity of private information retrieval. In *Proc. of the 24th International Colloquium on Automata, Languages and Programming*, volume 1256 of *Lecture Notes in Computer Science*, pages 401–407, 1997.

[2] A. Ambainis and S. Lokam. Improved upper bounds on the simultaneous messages complexity of the generalized addressing function. In *LATIN 2000*, volume 1776 of *Lecture Notes in Computer Science*, pages 207 – 216. Springer, 2000.

[3] L. Babai, P. G. Kimmel, and S. V. Lokam. Simultaneous messages vs. communication. In *12th Annual Symposium on Theoretical Aspects of Computer Science*, volume 900 of *Lecture Notes in Computer Science*, pages 361–372. Springer, 1995.

[4] D. Beaver and J. Feigenbaum. Hiding instances in multioracle queries. In C. Choffrut and T. Lengauer, editors, *STACS '90, 7th Annu. Symp. on Theoretical Aspects of Computer Science*, volume 415 of *Lecture Notes in Computer Science*, pages 37–48. Springer-Verlag, 1990.

[5] D. Beaver, J. Feigenbaum, J. Kilian, and P. Rogaway. Locally random reductions: Improvements and applications. *J. of Cryptology*, 10(1):17–36, 1997. Early version: Security with small communication overhead, CRYPTO '90, volume 537 of *Lecture Notes in Computer Science*, pages 62-76. Springer-Verlag, 1991.

[6] A. Beimel, Y. Ishai, and T. Malkin. Reducing the servers' computation in private information retrieval: PIR with preprocessing. Manuscript, 2001. Preliminary version: M. Bellare, editor, *Advances in Cryptology – CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 56–74. Springer, 2000, 2001.

[7] J. Benaloh and J. Leichter. Generalized secret sharing and monotone functions. In S. Goldwasser, editor, *Advances in Cryptology – CRYPTO '88*, volume 403 of *Lecture Notes in Computer Science*, pages 27–35. Springer-Verlag, 1990.

[8] C. Cachin, S. Micali, and M. Stadler. Computationally private information retrieval with polylogarithmic communication. In J. Stern, editor, *Advances in Cryptology – EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 402–414. Springer, 1999.

[9] B. Chor and N. Gilboa. Computationally private information retrieval. In *Proc. of the 29th Annu. ACM Symp. on the Theory of Computing*, pages 304–313, 1997.

[10] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. In *Proc. of the 36th Annu. IEEE Symp. on Foundations of Computer Science*, pages 41–51, 1995. Journal version: *J. of the ACM*, 45:965–981, 1998.

[11] G. Di-Crescenzo, Y. Ishai, and R. Ostrovsky. Universal service-providers for private information retrieval. *J. of Cryptology*, 14(1):37–74, 2001. Preliminary version in *Proc. of the 17th Annu. ACM Symp. on Principles of Distributed Computing*, pages 91–100, 1998.

[12] Y. Gertner, Y. Ishai, E. Kushilevitz, and T. Malkin. Protecting data privacy in private information retrieval schemes. In *Proc. of the 30th Annu. ACM Symp. on the Theory of Computing*, pages 151–160, 1998. Journal version: *J. of Computer and System Sciences*, 60(3):592–629, 2000.

[13] N. Gilboa and Y. Ishai. Compressing cryptographic resources. In *Advances in Cryptology – CRYPTO '99*, pages 591–608. Springer-Verlag, 1999.

[14] O. Goldreich and L. Trevisan. On the length of linear error-correcting codes having a 2-query local decoding procedure. Manuscript, 2000.

[15] Y. Ishai and E. Kushilevitz. Improved upper bounds on information theoretic private information retrieval. In *Proc. of the 31th Annu. ACM Symp. on the Theory of Computing*, pages 79 – 88, 1999.

[16] Y. Ishai and X. Sun. Personal communication, 2000.

[17] M. Ito, A. Saito, and T. Nishizeki. Secret sharing schemes realizing general access structure. In *Proc. of the IEEE Global Telecommunication Conf., Globecom 87*, pages 99–102, 1987. Journal version: Multiple Assignment Scheme for Sharing Secret. *J. of Cryptology*, 6(1):15-20, 1993.

[18] H. Karloff and L. Schulmann. Manuscript, 2000.

[19] J. Katz and L. Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proc. of the 32th Annu. ACM Symp. on the Theory of Computing*, pages 80–86, 2000.

[20] E. Kushilevitz and R. Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In *Proc. of the 38th Annu. IEEE Symp. on Foundations of Computer Science*, pages 364–373, 1997.

[21] J. H. van Lint. *Introduction to Coding Theory*. Springer-Verlag, 1982.

[22] E. Mann. Private access to distributed information. Master's thesis, Technion – Israel Institute of Technology, Haifa, 1998.

[23] A. Shamir. How to share a secret. *Communications of the ACM*, 22:612–613, 1979.

A Approximations of $\Lambda(m, d)$

We next want to give bounds on $\Lambda(m, d)$. For an integer d we can use the following approximation:

$$\Lambda(m, d) > \frac{(m-d)^d}{d!} \tag{1}$$

In particular, for $\Lambda(m, d) \geq n$ to hold, it is sufficient to let $m = (d!n)^{1/d} + d = \alpha_d d n^{1/d} + d$, where α_d is a constant depending on d. It holds that $\alpha_3 = (6)^{1/3}/3 \approx 0.61$, $\alpha_5 = (120)^{1/5}/5 \approx 0.52$, and $\alpha_d < 0.5$ for $d \geq 7$ (by the Stirling approximation $\lim_{d \to \infty} \alpha_d = 1/e$).

If $d = \lfloor \theta m \rfloor$ for some constant $\theta \leq 0.5$ we will use the following approximation.

$$2^{(H(\theta) - o(1))m} \leq \Lambda(m, \lfloor \theta m \rfloor) \leq 2^{H(\theta)m} \tag{2}$$

(cf. [21, Theorem 1.4.5]). In particular, for $\Lambda(m, \lfloor \theta m \rfloor) \geq n$ to hold, it is sufficient to let $m = (1/H(\theta) + o(1)) \log n$.

B Low-Degree Encoding

In this section we prove the validity of the low-degree encodings **E1** defined in Section 4.1. That is, for n distinct vectors $v^1, \ldots, v^n$ in GF(2)^m with Hamming weight at most d, we define $\mathbf{E1}(i) = v^i$. We need to show the existence of degree-d polynomials p_i such that $p_i(v^j)$ is 1 if $i = j$ and is zero otherwise. Assume, without loss of generality, that if $j < i$ then the weight of v^j is greater or equal to the weight of v^i. Denote by S_i the subset of $[m]$ containing the positions in which v^i is 1. We define the polynomials p_i one after the other, starting with p_1 and ending at p_n: Let

$$p_i \overset{\text{def}}{=} \prod_{h \in S_i} Y_h - \sum_{j : S_i \subset S_j} p_j . \tag{3}$$

If $S_i \subset S_j$ then the weight of v^j is greater than the weight of v^i and thus all the polynomials in the right hand side of (3) are already defined. Clearly, the degree of the polynomials p_i is at most degree d. We prove by induction on i that $p_i(v^j)$ equals 1 iff $i = j$. First, by the induction hypothesis, $p_i(v^i) = \prod_{h \in S_i} v^i_h = 1$. Second, if $j \neq i$ then $p_i(v^j) = \prod_{h \in S_i} v^j_h$ if $S_i \not\subset S_j$ and $p_i(v^j) = \prod_{h \in S_i} v^j_h - p_j(v^j)$ otherwise (again, by the induction hypothesis). In both cases $p_i(v^j) = 0$.