Private Multiparty Sampling and Approximation of Vector Combinations

Yuval Ishai[1], Tal Malkin[2], Martin J. Strauss[3], and Rebecca N. Wright[4]

[1] Computer Science Dept., Technion, Haifa 32000 Israel.
[2] Dept. of Computer Science, Columbia University, New York, NY 10025 USA.
[3] Depts. of Math and EECS, University of Michigan, Ann Arbor, MI 48109 USA.
[4] Computer Science Dept., Stevens Institute of Technology, Hoboken, NJ 07030 USA.

Abstract. We consider the problem of private efficient data mining of vertically-partitioned databases. Each of several parties holds a column of a data matrix (a vector) and the parties want to investigate the componentwise combination of their vectors. The parties want to minimize communication and local computation while guaranteeing privacy in the sense that no party learns more than necessary. Sublinear-communication private protocols have been primarily been studied only in the two-party case. We give efficient multiparty protocols for sampling a row of the data matrix and for computing arbitrary functions of a row, where the row index is additively shared among two or more parties. We also give protocols for approximating the componentwise sum, minimum, or maximum of the columns in which the communication and the number of public-key operations are at most polynomial in the size of the small approximation and polylogarithmic in the number of rows.

1 Introduction

There are many real-life scenarios in which several mutually distrusting entities (e.g., credit agencies, hospitals, or network carriers) have a common interest in obtaining useful summaries of their combined data. For instance, the parties may want to learn basic statistics on the combined data, measure the amount of similarity between their inputs, or detect irregularities or fraud by means of identifying major discrepancies in common entries.

In our setting, each of M parties $P_1, \ldots, P_M$ has a length-N (column) vector, denoted x^m for P_m as a private input. For some M-ary function f, the parties want to compute a length-N vector y whose n'th component y_n is given by $f(x_n^1, x_n^2, \ldots, x_n^M)$. We write this as $y = f(\mathbf{x})$ and call y a *combination* of the parties' inputs. Examples of combination functions are the identity function (where an M-ary identity function simply returns its M inputs as outputs), or the sum function (that returns the sum of its M inputs).

If N is small, general secure multiparty computation [17, 18] can be used efficiently. We provide solutions that are efficient even when N (and therefore y) is very large. We aim for solutions with local computation at most polynomial in N and M and communication at most polynomial in M and $\log(N)$. Towards

In *Proceedings of the 34th International Colloquium on Automata, Languages and Programming* (ICALP 2007), July 9–13, 2007.

this end, we provide solutions in which the parties do not compute y, but rather some moderately sized "approximation" or "summary" of it. In the non-private setting, there is a rich body of work demonstrating that communication complexity can be dramatically improved by using an approximate solution that allows a small error probability (e.g., [1, 22, 16, 4, 8]). Generally, however, these existing approximations are not private, meaning that some of the parties may learn more than what follows from their inputs and outputs.

Useful approximations that the parties may wish to compute include a sample of components in y or a known transform of y such as the Fourier transform; statistical summaries of y, such as a norm of y; an approximate reconstruction of y, such as a piecewise-constant approximation with error nearly optimal according to some norm; and a succinct data structure to which one can pose queries about y. As a concrete class of examples, we focus on the quantity $\|y\|_a = (\sum_n y_n^a)^{1/a}$, which we call a *norm* of y and denote by ℓ^a. (Technically, it is only a norm for certain values of a.) One can regard a norm of y as an approximation to the vector y. A useful special case is the problem of multiparty set intersection size. The parties have subsets $A_1, A_2, \ldots, A_M$ of a known universe of size N and we want to compute $|\bigcap_m A_m|$. (In this case the vector combination function f is the bitwise-AND, namely $y = f(x^1, \ldots, x^M) = \bigwedge_m x^m$, and the output the parties seek is $\|y\|_1 = \sum_n y_n$.) Even in the two-party case and without any privacy requirements, it is impossible to achieve a constant multiplicative approximation with a sublinear amount of communication. We thus settle for an additive approximation, up to an error of $\pm\epsilon N$.

Our Results. We present communication-efficient solutions to a large class of useful special cases of efficient private distributed computation. Our results also have useful consequences for the general theory of secure multiparty computation with sublinear communication and highlight some qualitatively interesting differences between the two-party and the multiparty case.

Specifically, we show solutions to two multiparty problems: private multiparty sampling (Section 3) and private approximation of vector combinations (Section 4). Our private multiparty sampling solution uses two-party private information retrieval (PIR) as a building block. Private multiparty sampling itself is a useful tool in a wide range of private approximation scenarios, such as communication-efficient multiparty approximations of set intersection and of the ℓ^2-norm of the sum of M input vectors. For private approximation of vector combinations, we consider approximations to the componentwise sum, minimum, or maximum over M vectors of integers. In a private computation setting, this problem is usually not interesting in the two-party case, as the input vector of one party together with the output vector allows her to determine most (if not all) of the other party's input. However, when there is a larger number of parties, this problem becomes natural.

In the full version of the paper, we also discuss some interesting consequences of our results to the general problem of reducing sublinear-communication secure multiparty computation to two-party PIR.

Related Work. The approach of constructing secure sublinear-communication protocols was initiated in the context of private information retrieval [6] and further studied both in other specific contexts (e.g., [24]) and in more general settings [27]. Freedman et al. [13] give an efficient two-party protocol for approximating the size of an intersection of sets from a universe of size N with additive error small compared with N. That is, they compute the AND of x_n^1 and x_n^2 at some random position n unknown to the parties. Our results in Section 3 can be regarded as a generalization of this result to more than two parties and to functions other than the AND of bits. Indyk and Woodruff [20] give a two-party, polylog-communication private protocol for approximating the ℓ^2-norm of the difference (or sum) of vector inputs. Our results of Section 3 can be used to extend their result to more than two parties.

Naor and Nissim [27] present a general compiler of any two-party protocol into a private protocol which preserves the communication, up to polynomial factors. This compiler, however, generally requires an exponential amount of local computation and thus it is not directly useful for the approximation problems we consider. Nevertheless, for the classes of functions for which their compilation technique efficiently applies, our results of Section 3 can be used to efficiently generalize their protocols from two parties to more than two parties providing security against any subset of the parties.

2 Background

2.1 Privacy

When mutually suspicious parties conduct a computation on their joint data, they want to guarantee that the privacy of their inputs is protected, in the sense that the protocol leaks nothing more than necessary about their inputs. In our context, where the computed output of the parties is an approximation $g(y)$ of a combination vector $y = f(\mathbf{x})$, we consider two types of privacy guarantee.

Privacy with respect to the output. This is the traditional privacy guarantee for secure multiparty computation [5, 17] of output $g(f(x))$ from inputs x. Given a functionality mapping the parties' inputs to (possibly randomized) outputs, the requirement is that no set of parties can learn anything about the inputs of other parties from protocol messages beyond their own inputs and outputs. In our setting, this privacy guarantee is the desired one in applications where the parties are willing to disclose the result $g(f(x))$ but do not want to reveal any other information.

Privacy with respect to the combination vector. This kind of guarantee, introduced in [12], is "privacy of approximations." A protocol is a private approximation protocol for f if its output is (with high probability) a good approximation[1] for the exact output of f, and moreover each set of parties learn

[1] In this paper, we do not insist on a specific notion of approximation, such as additive or multiplicative one. Instead, we accept whatever function g the parties want to

nothing additional from protocol messages (including the actual output of the protocol) beyond their own inputs and the ideal output, $f(x)$. Stated in our context, while the output of the protocol is $g(f(x))$, the privacy guarantee is that nothing is leaked that doesn't follow from $f(x)$. This is a weaker privacy guarantee (which admits much more efficient results in some cases). This is appropriate in applications where the parties do not mind disclosing $f(x)$, but do not want any further information leaked.

Adversary model. All of our protocols are computationally private against a non-adaptive, semi-honest (passive) adversary corrupting an arbitrary subset of the M parties. Naor and Nissim [27] showed how to upgrade security in the semi-honest model into security in the malicious model with a low communication overhead. Thus, from a theoretical point of view, our solutions can be modified to provide security in the malicious model while remaining communication-efficient. From a more practical point of view, most of our protocols provide reasonable security guarantees against malicious adversaries even without modification. In particular, the highly efficient protocols in Section 4 are fully private against a malicious adversary in the sense that it cannot learn more about the inputs of uncorrupted parties than is allowed in an ideal function evaluation.

2.2 PIR and oblivious transfer

We make use of private information retrieval (PIR) and oblivious transfer (OT). A PIR protocol allows a receiver to retrieve an item from a large database held by a sender without revealing which item she is after, and while using only a small amount of communication [6, 23]. A *symmetric* PIR (SPIR) protocol [15, 28], or sublinear-communication oblivious transfer [30, 11], further guarantees that the receiver cannot learn more than a single entry of the database. Any PIR protocol can be used (in a black-box way) to obtain an OT protocol with similar communication complexity [28, 9]. The communication complexity of PIR and OT on a database containing N short entries (say, bits) can made as small as $O(N^\epsilon)$ for an arbitrary constant $\epsilon > 0$, assuming that a homomorphic encryption scheme exists [23, 31, 26], or even polylogarithmic in N under more specific assumptions [3, 25, 14]. In the following, when referring to OT (and its variants) we always assume the communication to be sublinear in N.

3 Private Multiparty Sampling

In this section, we consider the challenge of extending a private sampling technique that lies in the core of previous two-party private approximation protocols [12, 13, 20], to the multiparty setting. Private multiparty sampling allows M parties, each holding a database x^m, to privately obtain $f(x_r^1, \ldots, x_r^M)$ where f

compute as a useful approximation of f, and focus on guaranteeing privacy of the protocol. For example, statistical summaries such as the norm of the vector are often used as an approximation of a vector.

is some fixed M-argument functionality (say, exclusive-or) and r is an index of a random entry that should remain secret.

When there is no restriction on communication complexity, a private multiparty sampling protocol can be constructed by making a black-box use of an arbitrary OT protocol, as follows from general techniques for secure multiparty computation [18, 19, 21]. Interestingly, such a construction becomes more challenging to obtain in the domain of sublinear-communication protocols. Further, this difficulty does not arise in the two-party setting and only seems to crop up when there are three or more parties. Indeed, a simple black-box construction of two-party private sampling from an arbitrary OT protocol (alternatively, PIR) is given in [12]. This construction maintains the communication complexity of the underlying OT protocol.[2] Thus, sublinear-communication OT (alternatively, PIR) can be used as a black box to realize sublinear-communication two-party private sampling. We do not know whether the same is true in the multiparty setting; this is an interesting question left open by our work.

Instead, we present a private multiparty sampling protocol that makes black-box use of PIR but it relies on the assumption that the underlying PIR protocol has only a single round of interaction (which is the case for almost all PIR protocols from the literature). The round complexity of our protocol is linear in the number of parties. In the full version of this paper, we also show how to implement private multiparty sampling with non-black-box use of an underlying PIR primitive. Although this is less efficient, it can be based on an arbitrary PIR protocol (even one using multiple rounds) and can yield a constant-round protocol (assuming that the PIR protocol is).

Private multiparty sampling can be used as a building block in a wide range of private approximation scenarios. For instance, it can be used in a straightforward way to obtain communication-efficient approximations for multiparty set intersection. Generalizing a two-party protocol of [13], if we let f be the bitwise AND function, the intersection size can be efficiently approximated (up to a small additive error) by making multiple invocations of the sampling primitive and outputting the fraction of 1's in the outputs. Private sampling can also be used, following the two-party techniques of [20], to obtain polylog-communication private approximation of the ℓ^2-norm of the sum of the M inputs.

3.1 Oblivious Transfer with Distributed Receiver

Towards implementing private multiparty sampling, we introduce and study a distributed variant of oblivious transfer which is of independent interest. (This primitive can be used as a basic building block for sublinear-communication multiparty protocols, generalizing the protocol compiler from [27] to the multiparty case. Details are omitted due to space.)

[2] The protocol from [12] is described for the special case where f is the exclusive-or function but can be generalized (as was done in [13, 20]) to arbitrary functions f. Our discussion is quite insensitive to the particular choice of f and in fact applies also to the simpler "distributed OT" primitive defined in Section 3.1.

In distributed OT, the role of the receiver is distributed between M parties. (A very different variant of distributed OT was considered in [29].) Specifically, a large database x of N entries is held by a distinguished party, say P_1, who functions as a sender. The entries of the database are indexed by some finite group of size N, which is taken to be $\mathbb{Z}_N$ by default. The index n of the entry to be retrieved is distributed between the M parties in an additive way. That is, $n = \sum_{m=1}^{M} n_m$, where each n_m is a local input of P_m and addition is taken over the underlying group. At the end of the protocol, some distinguished party, say P_M, should learn the selected entry x_n. More formally, we define *distributed-receiver oblivious transfer* (or distributed OT for short) as an $(M-1)$-private protocol for the following M-party functionality.

Definition 1 (Functionality DistOT_G). *Let G be a finite group of size N. The functionality DistOT_G is defined as follows:*

- *Inputs: Each party m, $1 \leq m \leq M$, holds a group element $n_m \in G$. The first party P_1 additionally holds a database $x = (x_n)_{n \in G}$.*
- *The last party P_M outputs $x_{n_1+\ldots+n_M}$. Other parties have no output.*

3.2 Private Multiparty Sampling from Distributed OT

We now present efficient black-box construction of private multiparty sampling protocols from distributed OT. We start by formally defining the sampling functionality induced by f.

Definition 2 (Functionality Sample-f). *Let f be an M-party functionality. (The functionality f is a deterministic or randomized mapping from M inputs to M outputs.) The randomized functionality* Sample-f *is defined as follows:*

- *Inputs: Each party m, $1 \leq m \leq M$, holds a database $x^m = (x^m_n)_{n \in [N]}$.*
- *The functionality picks a secret, uniformly random index $r \in [N]$ and outputs $f(x^1_r, x^2_r, \ldots, x^M_r)$.*

We start by handling the easier case where f is the identity function, outputting the concatenation of its M inputs. We denote the resulting sampling functionality by Sample-ID. In this case, we can use the following reduction to DistOT_G where G is an arbitrary group of size N. In the following, we arbitrarily identify elements of G with indices in $[N]$.

Reducing Sample-ID to DistOT

1. Each party P_m picks a random group element $r_m \in_R G$.
2. In parallel, the parties make M calls to DistOT, where in call i party P_i acts as sender with database x^i and every party P_m (including P_i) lets $n_m = r_m$. As a result, party P_M obtains the values $x^m_{r_1+\ldots+r_M}$ for $1 \leq m \leq M$ and sends them to all parties.
3. Each party outputs the M values received from P_M.

The correctness and privacy of the above reduction are straightforward to verify.

We now turn to the question of obtaining Sample-f from DistOT, for a general functionality f. We start by observing that Sample-f can be efficiently reduced to a simpler (randomized) functionality Sample-AS, where AS (for "additive sharing") outputs an M-tuple of strings that are random subject to the restriction that their exclusive-or is the concatenation of the M inputs.

Proposition 1. *For any polynomial-time computable M-argument function f there is a constant-round black-box $(M-1)$-private reduction of* Sample-f *to* Sample-AS *and 1-out-of-2 OT.*

Proof (sketch): The reduction proceeds by first invoking Sample-AS to obtain an additively shared representation of the inputs to f, and then running a general-purpose constant-round protocol (based on OT) to compute f from these shares. For the latter one can use the protocol of Beaver et al. [2]. ∎

In the above reduction, the OT primitive could be dispensed with, as it can be implemented from Sample-AS (using [9]).

Given the above, it suffices to reduce Sample-AS to Sample-ID. For simplicity, we restrict the attention to the case where each entry of a database x^m is a single bit. (The general case of ℓ-bit entries can be handled analogously.) A natural approach that comes to mind is to let each party P_m mask every bit of x^m with a random bit b_m, invoke Sample-ID on the resulting masked databases, and then use a private computation of a (randomized) linear function to convert the masked entries $x^m_r \oplus b_m$ together with the masks b_m into the required additive sharing. This approach fails for the following reason: an adversary corrupting P_m can learn both $x^m_r \oplus b_m$ (from the output of Sample-ID) and the mask b_m, which together reveal x^m_r and thus (together with x^m) give partial information about r. This is not allowed by the ideal functionality Sample-AS. Other variants of this approach fail for similar reasons.

To avoid the above problem, we must generate the masks in a completely distributed way. We achieve this by using DistOT over the group $G' = G \times Z_2$:

Reducing Sample-AS to DistOT

1. Each party P_m prepares an extended database $(x')^m$ of size $2N$ such that for each $n' = (n, b) \in G'$ we have $(x')^m_{n'} = x^m_n \oplus b$. In addition, each P_m picks a random group element $r_m \in_R G$ and M random bits $b_{m,m'}$, $1 \le m' \le M$.
2. In parallel, the parties make M calls to $\mathsf{DistOT}_{G'}$. In call i party P_i acts as sender with database $(x')^i$ and every party P_m (including P_i) lets $n'_m = (r_m, b_{m,i})$. As a result, party P_M obtains the values $(x')^m_{(r_1,b_{1,m})+\ldots+(r_M,b_{M,m})} = x^m_{r_1+\ldots+r_M} \oplus (b_{1,m} \oplus b_{2,m} \oplus \cdots \oplus b_{M,m})$ for $1 \le m \le M$.
3. Each party P_m, $m < M$, outputs the M-tuple $(b_{m,1}, b_{m,2}, \ldots, b_{m,M})$. For $m = M$, party P_M outputs the exclusive-or of this M-tuple with the M-tuple obtained in Step 2 above.

Proposition 2. *The reduction described above is an $(M-1)$-private black-box reduction from* Sample-AS *to* DistOT*. Moreover, the reduction is totally non-interactive.*

Combining Propositions 1 and 2 yields an efficient black-box reduction from Sample-f to DistOT.

3.3 Implementing Distributed OT

It remains to implement DistOT. We mainly focus on general constructions based on 1-out-of-n OT (equivalently, PIR [28, 9]). For $n \in G$, we use $x \ll_G n$ to denote the database x' obtained from x by applying the permutation induced by adding n to each index. That is, $x'_{n'} = x_{n'+n}$, where addition is in the group G. Note that in the default case where $G = Z_N$, the notation "$\ll$" corresponds to the usual notation of a cyclic shift to the left. When there is no ambiguity or when the choice of the group does not matter, we omit the group subscript.

A black-box construction of DistOT using one-round OT

A one-round OT can be specified by a randomized query algorithm $Q(n, \rho)$ (where ρ is the receiver's secret randomness), an answering algorithm $A(x, q)$ and a reconstruction algorithm $R(a, \rho)$. (The security parameter k is implicit in this notation.) The reduction proceeds as follows. Each party P_m sends an OT query pointing to its input n_m to the sender P_1. Each such query can be used to "obliviously shift" the database x by the amount n_m; more precisely, the n-th entry of a shifted database y is simply the answer to the OT query on $y \ll n$. The result of each such oblivious shift may be viewed as being encrypted using the key owned by the originator of the OT query. At the end of the $M-1$ oblivious shifts, the sender holds an $(M-1)$-iterated encryption of $x \ll (\sum_{m=1}^{M} n_m) - n_1$. The (n_1)-th entry $x_{n_1+\ldots+n_M}$ can be recovered by passing its iterated encryption between the parties, letting each peel off its own layer of encryption using the OT reconstruction function. (See Figure 1.)

1. Each party P_m, $m > 1$, picks an OT query $q_m = Q(n_m, \rho_m)$, and sends it to P_1.
2. P_1 initializes $a^{M+1} := x$.
3. For $i = M$ downto 2, party P_1 lets a^i be a database of N entries defined by $a^i_n = A(a^{i+1} \ll_G n_i, q_i)$. It then sends $b_1 = a^2_{n_1}$ to P_2.
4. For $i = 2$ to $M-1$, party P_i lets $b_i = R(b^{i-1}, \rho_i)$ and sends b_i to P_{i+1}.
5. Party P_M outputs $b_M = R(b^{M-1}, \rho_M)$.

Fig. 1. A black-box reduction of DistOT to one-round OT.

Proposition 3. *The reduction described in Figure 1 is $(M-1)$-private.*

Proof (sketch): Correctness is easy to verify. For privacy, we briefly sketch a formal construction of a simulator. The simulator is given the inputs of corrupted parties and, possibly, the output $x_{n_1+\ldots+n_M}$. It simulates the message sequence b_i

by iteratively applying the OT simulator starting from either the actual output (if P_M is corrupted) or a default output otherwise. The output of each iteration is used for the next iteration, along with either an actual input n_i (if P_i is corrupted) or a default input (if P_i is uncorrupted). This simulation process produces all messages b_i along with the local inputs ρ_i, q_i in reverse order. ∎

The complexity of the above reduction depends on the number of parties M and the relation between the size ℓ' of the answers of the OT protocol to the length of the database entries ℓ. Ideally, we have $\ell' = \ell + k \cdot \mathit{polylog}(N)$, where k is the security parameter. (Such an OT protocol can be based on the Damgård-Jurik encryption scheme [10, 25].) In this case, the complexity of the resulting DistOT protocol (on top of the length of the OT queries) is $\ell + Mk \cdot \mathit{polylog}(N)$. Thus, the protocol can be applied also for non-constant M. When the number of parties M is viewed as constant, the DistOT protocol can be made communication-efficient even if, say, $\ell' = \mathit{poly}(\ell, k) \cdot N^{0.9}$. (By "communication-efficient," we mean that the dependence on N can be reduced to $\mathit{poly}(\ell, k) \cdot N^{\epsilon}$ for an arbitrary $\epsilon > 0$.) Such an OT protocol can be based on an arbitrary homomorphic encryption scheme [23, 31].

4 Private Approximation of Vector Combinations

In this section, towards further reducing the communication complexity, we consider as approximation functions any of a wide array of natural "summary" functions of the combined data vector y (e.g., the ℓ^2-norm or an approximate t-term Fourier representation). Specifically, we consider $M \geq 3$ parties holding length-N vectors $x^1, \ldots, x^m$ who, ideally, want to compute (and are willing to disclose to the other parties) the componentwise sum $y = \sum_m x^m$ of their vectors or the componentwise minimum $y = \bigwedge_m x^m$ of their vectors. The actual output computed is an approximate size-t summary Y for y (e.g., the ℓ^2-norm, or an approximate t-term Fourier representation).

We exploit the fact that the entire (long) vector y may be leaked in order to obtain simple private protocols for the above problems, in which the communication and the number of public-key operations are at most polynomial in the size of the small approximation and polylogarithmic in N. In contrast, most previous protocols for sublinear-communication secure approximation (including the results of Section 3) require roughly as many public-key operations as the size of the entire database (given the current state of the art of PIR).

4.1 Vector Sums

Proposition 4. *Suppose parties $P_1, P_2, \ldots, P_M$ hold length-N vectors x^1, x^2, $\ldots, x^M$. Let $y = \sum_m x^m$ be the componentwise sum. There is a protocol for generating a Gaussian random variable with mean zero and variance $\sum_n y_n^2$ that leaks (to any subset of the parties) no more than y, requires local computation $NM^{O(1)}$, communication $M^{O(1)}$, and $O(1)$ rounds of interaction.*

Proof: If each component r_n of a vector r is a unit normal random variable, then $\sum_n r_n y_n$ is a Gaussian random variable Y with mean zero and variance equal to our desired value, $\sum_n y_n^2$. In particular, Y together with r leak nothing else about the inputs x^m beyond what is implied by their sum y. The sum $\sum_n r_n y_n$ can in turn be computed by first letting each party compute a local sum $s^m = \sum_n r_n x_n^m$ and then using a standard $(M-1)$-private protocol for adding up the M (short) integers s^m. The protocol is described in Figure 2.

The communication of the protocol in Figure 2 is as claimed, because the secure-sum protocol is applied to M numbers s^m, and not length-N vectors. ∎

Sketch Sum

1. The parties agree on pseudorandom Gaussian random vector r in the clear.
2. Party m receives vector x^m as input.
3. The parties individually compute sketches $s^m = \sum_n r_n x_n^m$.
4. The parties use a secure-sum sub-protocol to compute $\sum_m s^m = \sum_m \sum_n r_n x_n^m = \sum_n r_n \sum_m x_n^m = \sum_n r_n y_n$, where $y = \sum_m x^m$ is the componentwise sum of the parties' input vectors.

Fig. 2. A protocol for computing an additive sketch.

Sketch Min

1. The parties agree on pseudorandom exponential random vector r in the clear.
2. Party m receives vector x^m as input.
3. The parties individually compute sketches $s^m = \bigwedge_n r_n x_n^m$.
4. The parties use a secure sub-protocol to compute $\bigwedge_m s^m = \bigwedge_m \bigwedge_n r_n x_n^m = \bigwedge_n r_n \bigwedge_m x_n^m = \bigwedge_n r_n y_n$, where $y = \bigwedge_m x^m$ is the componentwise minimum of the parties' input vectors.

Fig. 3. A protocol for computing a minimum sketch.

4.2 Vector Minima

We generalize the above protocol to the componentwise minimum instead of sum. Also, instead of approximating the quantity $\sum_n y_n^2$, we approximate the harmonic mean, or its inverse, $\sum_n y_n^{-1}$. See, e.g., [7] for example uses in algorithms of estimating the parameter of an exponential random variable.

Proposition 5. *Suppose parties $P_1, P_2, \ldots, P_M$ hold length-N positive-valued vectors $x^1, x^2, \ldots, x^M$. Let $y = \bigwedge_m x^m$ be the componentwise minimum. There is*

a protocol for generating an exponential random variable with parameter $\left(\sum_n y_n^{-1}\right)$ *that leaks (to any subset of the parties) no more than* y*, requires local computation* $NM^{O(1)}$*, communication* $M^{O(1)}$*, and* $O(1)$ *rounds of interaction.*

Proof: It is known that, if each component r_n of a vector r is a unit exponential random variable, then $\sum_n r_n y_n$ is an exponential random variable Y with parameter equal to our desired value of $\sum_n y_n^{-1}$. In particular, Y together with r leak no more than y. The parties use the protocol of Figure 3. The subproblem for which a secure protocol is needed is computing the minimum of M short integers, for which efficient $(M-1)$-private protocols exist (e.g., using the general-purpose constant-round protocol of [2]). ∎

Acknowledgments

Yuval Ishai was supported by grants 36/03 and 1310/06 from the Israel Science Foundation; Tal Malkin by NSF grant CCF-0347839; Martin Strauss by NSF grants DMS-0510203 and DMS-0354600; and Rebecca Wright by NSF grant CCR-0331584. We thank Stillian Stoev for helpful discussions.

References

1. N. Alon, P. Gibbons, Y. Matias, and M. Szegedy. Tracking join and self-join sizes in limited storage. In *Proc. Eighteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 10–20, 1999.
2. D. Beaver, S. Micali, and P. Rogaway. The round complexity of secure protocols. In *Proc. 22th ACM STOC*, pages 503–513, 1990.
3. C. Cachin, S. Micali, and M. Stadler. Computationally private information retrieval with polylogarithmic communication. In *Advances in Cryptology — EUROCRYPT '99*, LNCS 1592, pages 404–414. Springer-Verlag, 1999.
4. E. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.
5. R. Canetti. Security and composition of multiparty cryptographic protocols. *J. Cryptology*, 13(1):143–202, 2000.
6. B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. In *Proc. 36th IEEE FOCS*, pages 41–50, 1995.
7. E. Cohen. Size-estimation framework with applications to transitive closure and reachability. *J. Computer and System Sciences*, 55(3):441–453, 1997.
8. G. Cormode and S. Muthukrishnan. Estimating dominance norms of multiple data streams. In *Proc. 11'th European Symposium on Algorithms*, pages 148–160, 2003.
9. G. Di Crescenzo, T. Malkin, and R. Ostrovsky. Single database private information retrieval implies oblivious transfer. In *Advances in Cryptology — EUROCRYPT '00*, pages 122–138, 2000.
10. I. Damgard and M. Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. *Public Key Cryptography*, pages 119–136, 2001.
11. S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. *Communications of the ACM*, 28:637–647, 1985.

12. J. Feigenbaum, Y. Ishai, T. Malkin, K. Nissim, M. Strauss, and R. Wright. Secure multiparty computation of approximations. *ACM Transactions on Algorithms*, 2(3):435–472, 2005. An earlier version of this paper appeared in ICALP 2001.
13. M. Freedman, K. Nissim, and B. Pinkas. Efficient private matching and set intersection. In *Advances in Cryptology — EUROCRYPT '04*, LNCS 3027, pages 1–19. Springer-Verlag, 2004.
14. C. Gentry and Z. Ramzan. Single-database private information retrieval with constant communication rate. In *Proc. 32nd ICALP*, pages 803–815, 2005.
15. Y. Gertner, Y. Ishai, E. Kushilevitz, and T. Malkin. Protecting data privacy in private information retrieval schemes. *J. Computer and System Sciences*, 60(3):592–692, 2000. A preliminary version appeared in 30th STOC, 1998.
16. A. Gilbert, S. Guha, P. Indyk, Y. Kotidis, S. Muthukrishnan, and M. Strauss. Fast, small-space algorithms for approximate histogram maintenance. In *Proc. 34th ACM STOC*, pages 389–398, 2002.
17. O. Goldreich. Secure multi-party computation (working draft, version 1.1). available at `http://philby.ucsd.edu/cryptolib/BOOKS/oded-sc.html`, 1998.
18. O. Goldreich, S. Micali, and A. Wigderson. How to play ANY mental game. In *Proc. 19th ACM STOC*, pages 218–229. ACM Press, 1987.
19. O. Goldreich and R. Vainish. How to solve any protocol problem—an efficiency improvement. In *Advances in Cryptology — CRYPTO '87*, pages 73–86, 1987.
20. P. Indyk and D. Woodruff. Private polylogarithmic approximations and efficient matching. In *Proc. 3rd Theory of Cryptography Conference*, LNCS 3876, pages 245–264, 2006.
21. Joe Killian. Founding cryptography on oblivious transfer. In *Proc. 20th ACM STOC*, pages 20–31, 1988.
22. E. Kushilevitz and Y. Mansour. Learning decision trees using the fourier sprectrum. In *Proc. 23th ACM STOC*, pages 455–464, 1991.
23. E. Kushilevitz and R. Ostrovsky. Replication is NOT needed: SINGLE database, computationally-private information retrieval. In *Proc. 38th IEEE FOCS*, pages 364–373, 1997.
24. Y. Lindell and B. Pinkas. Privacy preserving data mining. *J. Cryptology*, 15(3):177–206, 2002. An earlier version appeared in *Proc. Crypto 2000*.
25. H. Lipmaa. An oblivious transfer protocol with log-squared communication. In *the 8th Information Security Conference (ISC'05)*, volume 3650 of *LNCS*, pages 314–328. Springer-Verlag, 2005.
26. E. Mann. Private access to distributed information. Master's thesis, Technion - Israel Institute of Technology, Haifa,, 1998.
27. M. Naor and K. Nissim. Communication preserving protocols for secure function evaluation. In *Proc. 33th ACM STOC*, pages 590–599, 2001.
28. M. Naor and B. Pinkas. Oblivious transfer and polynomial evaluation. In *Proc. 31st ACM STOC*, pages 245–254. ACM Press, 1999.
29. M. Naor and B. Pinkas. Distributed oblivious transfer. In *Proc. ASIACRYPT*, 2000.
30. M. O. Rabin. How to exchange secrets by oblivious transfer. Technical Report TR-81, Aiken Computation Laboratory, Harvard University, 1981.
31. J. P. Stern. A new and efficient all-or-nothing disclosure of secrets protocol. In *Advances in Cryptology — ASIACRYPT '98*, LNCS 1514, pages 357–371. Springer-Verlag, 1998.