Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Private multiparty sampling and approximation of vector combinations[☆]

Yuval Ishai[a], Tal Malkin[b,*], Martin J. Strauss[c], Rebecca N. Wright[d]

[a] Computer Science Department, Technion, Haifa 32000, Israel
[b] Department of Computer Science, Columbia University, New York, NY 10025, USA
[c] Departments of Math and EECS, University of Michigan, Ann Arbor, MI 48109, USA
[d] Computer Science Department and DIMACS, Rutgers University, Piscataway, NJ 08854, USA

ARTICLE INFO

Keywords:
Secure multiparty computation
Sublinear-communication algorithms
Approximation algorithms

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 primarily been studied only in the two-party case. In contrast, this work focuses on multi-party settings.

First, we give efficient private multiparty protocols for sampling a row of the data matrix and for computing arbitrary functions of a random row, where the row index is additively shared among two or more parties. These results can be used to obtain private approximation protocols for several useful combination functionalities. Moreover, these results have some interesting consequences for the general problem of reducing sublinear-communication secure *multiparty* computation to two-party private information retrieval (PIR).

Second, we give protocols for computing approximations (summaries) of the componentwise sum, minimum, and maximum of the columns. Here, while providing a weaker privacy guarantee (where the approximation may leak up to the entire output vector), our protocols are extremely efficient. In particular, the required cryptographic overhead (compared to non-private solutions) is polylogarithmic in the number of rows.

1. Introduction

1.1. Overview

Our problem. We are interested in the following problem. A collection of M parties $P_1, \ldots, P_M$ have length-N (column) vectors $x^1, \ldots, x^M$, respectively, as private inputs. For some M-ary function f (a commonly-known parameter to our problem), the parties want to compute a length-N vector y whose nth component y_n is given by $f(x_n^1, x_n^2, \ldots, x_n^M)$. We write this as $y = f(\mathbf{x})$. We say that the vector y is a *combination* of the parties' inputs and that each party holds a *piece* of the database. 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).

[☆] A preliminary version of this paper appeared in [Y. Ishai, T. Malkin, M.J. Strauss, R.N. Wright, Private multiparty sampling and approximation of vector combinations, in: Proc. 34th International Colloquium on Automata, Languages and Programming — ICALP'07, 2007, pp. 243–254].

[*] Corresponding author.
E-mail address: tal@cs.columbia.edu (T. Malkin).

doi:10.1016/j.tcs.2008.12.062

We are interested in the case where N is large, so that y is a large object, while M is considered small. Thus the parties want to keep costs sublinear in N, where possible. We consider the following general resource bounds:

- Local computation at most polynomial in N and M[1];
- Communication at most polynomial in M and $\log(N)$. Less desirable, but sometimes acceptable, is communication which is sublinear in N (preferably with dependence on N that can be made N^{ϵ} for arbitrary $\epsilon > 0$).

Because computing y exactly typically requires communication at least linear in N, the parties do not compute $y = f(x)$, but rather some "approximation" or "summary" $g(y)$ which is moderately sized. Examples of useful approximations $g(y)$ that the parties may wish to compute are:

- A sample of components in y. (More generally, a sample of a known transform of y, such as the Fourier transform of y.)
- Statistical summaries of y, such as (an approximation of) a norm of y.
- An approximate reconstruction of y, such as a piecewise-constant approximation with error nearly optimal according to some norm.
- A succinct data structure to which one can pose queries about y.

Such approximations are abundant in the literature (see below). Generally, however, they are not *private*, meaning that some of the parties may learn more than what follows from their inputs and intended outputs. In this paper, we present private approximations for various functions f and g (including the general case of an arbitrary f), in two different frameworks of privacy (see details below).

Motivating examples. The above problems are motivated by 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.

An exact solution to most problems of this type provably requires an amount of communication that is linear in the size of the data (even if there are no privacy requirements). It turns out, however, that the communication complexity can be dramatically improved by settling for an approximate solution and allowing a small error probability. This has been the theme of a very rich and productive recent body of work (c.f., [1,36,43,25,24,8,15,12] and references therein).

As one concrete class of examples, we focus on the quantity $\|y\|_a = \left(\sum_n y_n^a\right)^{1/a}$, 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 $\left|\bigcap_m A_m\right|$. (In this case the vector combination function f is the bitwise-AND,[2] namely $y = f(x^1, \ldots, x^M) = \bigwedge_m x^m$, and the output the parties are interested in 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 $\pm\epsilon N$.

1.2. Privacy

When mutually-suspicious parties conduct joint data mining, they want to guarantee that the privacy of their inputs is protected, in the sense that the protocol leaks nothing more than their inputs and their outputs. In our context, where the actual output of the parties is not the combination vector $y = f(x)$ but some type of an approximation of it $g(y)$, we consider two types of privacy guarantees.

Privacy with respect to the output. This is the traditional privacy guarantee for inputs x and output $g(f(x))$, following standard definitions of secure computation from the literature [9,26]. 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. (By default, we assume that all parties receive an identical output.) Thus, this privacy guarantee is the desired one in applications where the parties are willing to disclose the approximation of f, namely $g(f(x))$, but do not want to reveal any other information.

Privacy with respect to the combination vector. The second kind of guarantee, introduced in [17], is "privacy of approximations". A protocol is a private approximation protocol for f if its output is (with high probability) a good approximation[3] for the exact output of f, and moreover each set of parties learns nothing additional from protocol messages (including the *actual* output of the protocol) beyond their own inputs and the *ideal* output, $f(x)$. See [17] for motivation. Stated in our context, while the output of the protocol is $g(f(x))$, the privacy guarantee is that nothing is leaked that does not follow from $f(x)$. This is a weaker privacy guarantee (which allows us to achieve much more efficient results in some cases).

[1] Since N is large, it is also desirable to have parties read the data as a data stream, using only a small amount of local storage. While most of the protocols we present satisfy this extra requirement, we do not consider it explicitly.

[2] Throughout this paper we use $\bigwedge$ to denote the component-wise minimum. In the case of binary values, like in this example, this is the same as the bitwise-AND.

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

This guarantee is appropriate in applications where the parties do not mind disclosing $f(x)$ (they compute $g(f(x))$ only for efficiency reasons), but do not want any further information leaked.

Adversary model. Below, we consider the semi-honest model, in which parties follow the protocol but infer whatever they can from the information they receive. In [45], the authors showed how to upgrade security in the semi-honest model into security in the malicious model with a low communication and computation overhead. Specifically, assuming the existence of strong collision-resistant hash functions, this transformation incurs a polylogarithmic *additive* overhead in communication and a polylogarithmic multiplicative overhead in computation. (While [45] explicitly consider the two-party setting, their approach applies to the multiparty setting as well.) Thus, from a theoretical point of view it suffices to focus on privacy in the semi-honest model, as we do in this work. From a more practical point of view, most of our protocols provide reasonable security guarantees even against malicious adversaries. In particular, the highly efficient protocols from 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. While in these protocols a malicious adversary can violate the correctness, e.g. by forcing the output to be random, this is not very significant for the type of functionalities we consider. These functionalities typically have the property that even in an ideal implementation, a malicious adversary can render the computation essentially useless by adversarially choosing its inputs.

All of the protocols we present in this paper are computationally private against a non-adaptive, semi-honest (passive) adversary corrupting an *arbitrary subset* of the M parties. Thus, we cannot make use of Multi-Party Computation (MPC) techniques that apply to the case of an honest majority.

1.3. Our results

We present communication-efficient solutions to a large class of useful special cases of the general problem described above. 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. We describe our results in two parts, corresponding to the two privacy frameworks discussed above (privacy with respect to the output, and with respect to the combination vector).

1.3.1. Private multiparty sampling

In Section 3 we consider the challenge of extending a private sampling technique, which lies in the core of previous two-party protocols for private approximations, to the multiparty setting. Specifically, we provide protocols for sampling a row of the data matrix and for computing arbitrary functions of a random row, where the row index is additively shared among two or more parties. This corresponds to the case where g is the fixed randomized function selecting a random (and secret) entry of y, and f can be an arbitrary function.

We show how to implement this primitive using any (two-party) private information retrieval (PIR) protocol [10,37] as a building block. This general transformation makes a non-black-box[4] use of the underlying PIR protocol, which can be quite costly (though still satisfying our requirements such as sublinear communication).

In the case where the underlying PIR protocol only has a single round of interaction, we show a more efficient variant of this transformation which makes a black-box use of the PIR query generation and answering algorithms, but still requires a non-black-box use of the PIR reconstruction algorithm.[5]

Our multiparty sampling protocols can be used as building blocks in a wide range of private approximation scenarios. For instance, they can be used in a straightforward way to obtain communication-efficient approximations for multiparty set intersection. Indeed, letting 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. This generalizes a (two-party) protocol of [20]. The sampling protocols can also be used, following the technique of [32], to obtain polylog-communication private approximation of the ℓ^2 norm of the sum of the M inputs. In Section 1.4 we briefly describe how our results can be used as a generalization of these and other previous works in the two-party setting, into the multi-party setting.

Applications to general secure computation. Our private sampling protocols of Section 3 are based on a natural generalization of PIR and sublinear-communication OT (SPIR) to the multiparty setting. The latter primitive distributes the role of the receiver between the M parties, taking the actual selection as the sum (modulo N) of M selection indices held by the different parties. Our efficient implementations of this primitive can be used as a basis for general sublinear-communication protocols for secure multiparty computation, following the approach of Naor and Nissim [45] in the two-party case. In contrast to the two-party case, where an arbitrary PIR protocol can be used as a *black box* as a basis for general sublinear-communication secure computation, we do not know whether this is the case also in the multiparty setting. The possibility of making a black-box use of PIR in the context of sublinear-communication multiparty computation remains an intriguing open question.

[4] By this we mean that the protocol cannot use R as an oracle, and must depend on its implementation; see [51] for formal definitions and discussion of black-box reductions in cryptography.

[5] This corrects an error in the conference version of this paper [33], see Section 3.3.

1.3.2. Private approximation of vector combinations

In Section 4 we focus on the case where f is either the (componentwise) sum over the integers or the minimum or the maximum of M vectors, and the goal is to approximate $f(x^1, \dots, x^M)$, while leaking nothing more than this combination vector. This problem is usually not very well motivated in the two-party case, as the input vector of one party together with the output vector allows him or her to determine most (if not all) of the other party's input, and thus privacy does not mean much. However, when there is a larger number of parties this problem becomes natural.

Towards reducing the communication complexity, we consider approximation functions g which may be any of a wide array of natural "summary" functions of the combined data vector y (e.g., an approximation of the ℓ^2-norm, or an approximate t-term Fourier representation). We show how to 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 cryptographic overhead in computation is at most polynomial in the size of the small approximation and polylogarithmic in the number of rows. This should be contrasted with most previous protocols for sublinear-communication secure approximation (as well as the results of Section 3), which, given state-of-the-art PIR, require a cryptographic computational overhead which is linear in the number of rows (the size of the entire data).

Going back to the example of multiparty set intersection, the protocol obtained from the results of Section 4 is much more efficient and practical than the one obtained from the private sampling primitive of Section 3, but provides a weaker privacy guarantee.

The results of Section 4 are based on the observation that most (non-private) approximations from the literature can be derived from short local randomized sketches that can be efficiently combined to yield the desired approximation. In previous uses of such sketches in the context of private approximations, it was necessary to hide the randomness that was used for generating the samples. Since in our setting we allow leaking the entire vector y, we can use *public* (pseudo)randomness for locally producing the sketches, and then obtain the output via secure computation of simple functions (addition, maximum or minimum) on the short sketches.

1.4. Related work

The approach of constructing secure sublinear-communication protocols was initiated in the context of private information retrieval [10] and further studied both in other specific contexts (e.g., [40]) and in more general settings [45].

The notion of secure approximations was introduced by Feigenbaum et al. [17], who provide sublinear-communication protocols for Hamming distance among two parties. As one of their tools, they show how to compute the XOR of x_n^1 and x_n^2 at a random position n unknown to the parties. Our results in Section 3 can be regarded as a generalization of this primitive to more than two parties and to functions other than the XOR of bits. They also provide a secure approximation protocol for the permanent and related #P-hard problems.

Halevi et al. [30] consider secure approximations of NP-hard functions and show some negative results. Specifically, they show there exist natural NP-hard functions, such as the size of the minimum vertex cover in a graph, which do not admit non-trivial secure approximation, although they do admit good approximation algorithms without the security restriction. They also point out that this phenomenon does not hold for all NP-hard functions, by presenting an artificial NP-hard function that can be securely approximated.

Definitions and protocols for privately approximating *search* problems (for which many exact solutions may exist) were given in [32,4–6]. In this work we are only concerned with the private approximation of functions with a unique exact output.

Freedman et al. [20] give an efficient two-party protocol for approximating the size of an intersection of sets from a universe of size N with additive error that is small compared with N. To do this, they compute the AND of x_n^1 and x_n^2 at some random position n unknown to the parties. Again, 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 [32] 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 this result to more than two parties.

Naor and Nissim [45] present a general compiler of any two-party protocol into a private protocol that 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 the compilation technique of [45] efficiently applies, our results of Section 3 can be used to efficiently generalize the protocols of [45] from two parties to more than two parties, providing security against any strict subset of the parties. (See more details in Section 3.1.1.)

In recent and independent work, Franklin et al. [19] consider a problem similar to the distributed oblivious transfer problem we consider in Section 3.1. They present a sublinear-communication protocol for this problem that extends *concrete* (two-party) PIR protocols from the literature to the multi-party setting by relying on a *threshold* homomorphic encryption scheme, rather than the standard homomorphic encryption scheme used in the original PIR protocols. In contrast, we focus on the goal of extending an *arbitrary* PIR protocol to the multi-party setting while keeping the communication complexity sublinear in N. Combined with concrete PIR protocols, our general non-black-box construction has a similar asymptotic communication, computation, and round complexity as the construction from [19], though the more specialized approach of [19] can typically be tailored to yield better dependence on the number of parties and better concrete efficiency.

2. Background and preliminaries

In this section, we present background, notations and definitions that we use in this paper. Throughout this paper, N serves as an input length parameter and M serves as the number of parties. As mentioned in Section 1, we consider N to be a large polynomial (so linear communication in N is prohibitive), while M is reasonably small. Accordingly, we seek efficient protocols, where the local computation of each party is polynomial in N and M, while the communication is polynomial in M and sublinear in N, preferably polynomial in $\log(N)$.

A function $f : \mathbb{N} \to [0, 1]$ is *negligible* if it is asymptotically smaller than any inverse polynomial, i.e., $f(n) \in n^{-\omega(1)}$. The function f is *overwhelming* if $1 - f$ is negligible. A *distribution ensemble* $D = \{D_x\}_{x\in X}$ is a family of probability distributions indexed by some infinite set X of binary strings. We sometimes take $X = \{1^n : n \in \mathbb{N}\}$, in which case the indices in X are viewed as natural numbers.

Definition 1. Two distribution ensembles $D = \{D_x\}_{x\in X}$ and $D' = \{D'_x\}_{x\in X}$ are *statistically indistinguishable*, (written $D \overset{s}{\equiv} D'$) if there is a negligible function $\mu(\cdot)$ such that, for every $x \in X$,

$$\mathrm{SD}(D_x, D'_x) < \mu(|x|),$$

where SD denotes statistical distance defined by $\mathrm{SD}(Z, Z') = \frac{1}{2}\sum_a |\Pr(Z = a) - \Pr(Z' = a)|$.

Ensembles D and D' are *computationally indistinguishable*, (written $D \overset{c}{\equiv} D'$), if for every family $\{C_n\}$ of polynomial-size circuits there exists a negligible function $\mu(\cdot)$ such that for every $x \in X$ of length n,

$$|\Pr(C_n(D_x) = 1) - \Pr(C_n(D'_x) = 1)| < \mu(n).$$

2.1. Secure multiparty computation

Secure multiparty computation allows two or more parties to evaluate a specified function of their inputs while hiding their inputs from each other. When formally defining security, it is convenient to think of an adversary that tries to gain as much advantage as it can by corrupting at most t parties during the execution of the protocol. Security (specifically, t-security for an adversary who may corrupt up to t parties) is then defined by requiring that whatever the adversary achieves in a real-life execution of the protocol, it can efficiently simulate in an *ideal process*, in which a trusted party is being used to evaluate the function. Thus, the protocol prevents the adversary from gaining an extra advantage over what it could have gained in an ideal solution.

There are several notions of security and various degrees of strength (e.g., [26,9,2,44]). In this work, we mostly deal with the special case of secure computation in the semi-honest model, or private computation. In this model it is assumed that the adversary is *passive* (or *honest-but-curious*) and cannot modify the behavior of corrupted parties. In particular, private computation is only concerned with the information learned by the adversary, and not with the effect misbehavior may have on the protocol's correctness. In [45], the authors showed how to convert any private protocol into a secure protocol, roughly preserving the computation *and* communication cost. (As noted in Section 1.2, this approach applies to the multiparty setting as well.)

Another distinction between different notions of security is the extent to which the transcript produced by the ideal-process adversary should resemble the one produced by the real-life execution of the protocol. The three standard variants are *perfect*, *statistical*, and *computational* indistinguishability. These naturally define corresponding notions of perfect, statistical, and computational security. Finally, there is a distinction between a *non-adaptive* adversary, who corrupts a number of parties before the computation starts and cannot corrupt any more parties later on, and an *adaptive* adversary, who may corrupt parties during the execution of the protocol, according to what has transpired so far.

In this work we focus on computational privacy against a non-adaptive passive adversary, who may corrupt an arbitrary subset of the M parties (namely, protocols achieving $(M - 1)$-privacy).[6] When we specify an M-party functionality g that should be privately computed, we will either denote the output of g by an M-tuple $(y_1, \ldots, y_M)$, in which case y_i is the output of party P_i, or by a single value, in which case this value should be known by default to all parties. (We also consider single-output functionalities in which the output is given to only one party.)

The formal definitions we use follow the standard ones of, e.g., [9,26]. We also refer to the (standard) notion of privacy discussed above as *privacy with respect to the output*. (In our context, the output is of the form $g(f(x))$ for some combination vector $f(x)$, namely the protocols are private computations of gf.) This is to differentiate this notion from the weaker one of *privacy with respect to the combination vector* (discussed in Section 1 and again below) that we use in Section 4 for private approximations of some combination functions.

The first general feasibility results for secure computation were obtained by Yao [54] and by Goldreich, Micali, and Wigderson [27]. Yao's constant-round two-party protocol was generalized to the multiparty case by Beaver et al. [3]. The following theorem relates the complexity of privately computing a functionality g to the circuit size of g.

[6] For our (non-adaptive) model, this is equivalent to M-privacy, namely the adversary may corrupt as many parties as he wants, up to all M parties. We chose to state our theorems with $(M - 1)$-privacy rather than M-privacy, as privacy for an adversary who corrupted all M parties is meaningless in this setting.

Theorem 2.1 (*[54,27,3]*). *Let $C = \{C_n\}$ be a uniform family of (deterministic or probabilistic*[7]*) Boolean circuits of size $s(n)$, where the input to C_n is viewed as an M-tuple of n-bit strings and its output as an M-tuple of strings. Let g denote the functionality computed by the family C. Then g can be privately computed in a constant number of rounds with $\tilde{O}(M^2 \cdot s(n))$ bits of communication by making (a black-box) use of an arbitrary oblivious transfer protocol.*[8]

A similar statement holds also when the adversary is active [39,34,49].

The asymptotic complexity notation $\tilde{O}(c(n))$ above should be read as $O(c(n) \cdot n^{\epsilon})$ for an arbitrarily small constant $\epsilon > 0$. However, $\tilde{O}(c(n))$ can be read as $O(c(n) \cdot \log^{O(1)} n)$ if stronger cryptographic assumptions are made. Specifically, in Theorem 2.1 it suffices to assume the underlying oblivious transfer primitive to be secure against sub-exponential adversaries.

Notice that any protocol that uses Theorem 2.1 to privately compute g is necessarily non-black-box with respect to g (as the specific circuit for g is needed, and not just oracle access to it). Indeed, this is what makes our reductions in Section 3.3 non-black-box (at least with respect to the reconstruction algorithm of PIR in the case of our "almost black-box" construction).

2.2. PIR and oblivious transfer

There has been a large body of work on obtaining sublinear-communication protocols for private information retrieval (PIR) and oblivious transfer (OT). A PIR protocol [10,37] allows a receiver to retrieve a selected item from a large database held by a sender without revealing which item he or she is after, and while using only a small amount of communication (sublinear in the size of the database). A *symmetric* PIR (SPIR) protocol [23], or 1-out-of-N oblivious transfer [50,16,26], further guarantees that the receiver cannot learn more than a single entry of the database. Specifically, OT is defined as follows.

Definition 2 (*Oblivious Transfer*). An *n-choose-1 oblivious transfer* protocol (with security against a passive adversary), abbreviated as $\binom{n}{1}$-OT or just OT, is a private protocol for the following deterministic functionality between two parties called a *sender* and a *receiver*. The sender's input is an n-bit string x and the receiver's input is an index $i \in [n]$. The receiver outputs the bit x_i, and the sender has no output.

By Theorem 2.1, $\binom{n}{1}$-OT can be implemented with nearly linear communication under standard assumptions. However, the $\binom{n}{1}$-OT functionality also admits much more efficient solutions:

Theorem 2.2 (*[37,23,52,42,46,13]*). *Assuming the existence of a homomorphic encryption scheme,*[9] *there is a 2-round $\binom{n}{1}$-OT protocol with $\tilde{O}(1)$ bits of communication.*

As in Theorem 2.1, $\tilde{O}(1)$ above should be read as $O(N^{\epsilon})$ for an arbitrarily small constant $\epsilon > 0$, but can be read as $O(\log^{O(1)} N)$ if stronger cryptographic assumptions are made, in this case specific number-theoretic assumptions from [7,41,22]. We refer the reader to the above papers for a more concrete analysis of the complexity tradeoffs and to [21] for a general survey of PIR protocols.

Theorem 2.2 uses the fact that any PIR protocol can be used in a black-box way to obtain an OT protocol with similar communication complexity [46,13], and that PIR on a database containing N short entries (say, bits) can be achieved with communication complexity $\tilde{O}(1)$, based on the corresponding assumptions. In the following, when referring to OT (and its variants) we always assume the communication to be sublinear in N.

We refer the reader to the papers referenced above for the formal definition of PIR, but note that some of the PIR definitions (e.g., [7,41]) also require that the computational complexity of the receiver be sublinear. This property is needed by our constructions in Section 3.3, and is achieved by all known PIR protocols. In fact, in almost all PIR protocols from the literature the receiver's computational complexity is $\tilde{O}(1)$. (The only exception we are aware of is the protocol from [22], in which the receiver needs to spend $\tilde{O}(n^{1/2})$ time to compute discrete logarithms.)

2.3. Secure computation of approximations and privacy with respect to the combination vector

The notion of secure computation of approximations was introduced in [17], which gives two definitions and several protocols (that lay the basis for some of the applications of our private computation results of Section 3). Secure computation of an approximation allows two or more parties to evaluate a specified approximation g applied to a function f of their inputs x, where the parties do not want to leak any more information than given by $f(x)$. The motivation is to enjoy the advantages of approximations (where exact computation is prohibitively expensive) in a setting where the parties want to maintain their privacy. See [17] for more discussion and motivation.

[7] A probabilistic circuit includes, in addition to the standard inputs, a polynomial number of random inputs.

[8] Oblivious transfer is a standard cryptographic primitive defined in the next section. It is implied by standard cryptographic assumptions, such as the existence of (enhanced) trapdoor permutations. See [26].

[9] Loosely speaking, a semantically-secure encryption scheme [29] is said to be *homomorphic* if: (1) The plaintexts are taken from some group $(H, +)$, and (2) From encryptions of group elements h_1, h_2 it is possible to *efficiently* compute a *random* encryption of $h_1 + h_2$. Homomorphic encryption can be based on a variety of intractability assumptions, including the Quadratic Residuosity assumption and the Decisional Diffie–Hellman assumption.

In Section 4 we use a definition given in [17] (Definition 7 there),[10] and call it *privacy with respect to the combination vector*. The notion of privacy with respect to the combination vector requires that the parties correctly compute the approximation $g(f(x))$, and that an ideal-world adversary (simulator), given the inputs of the corrupted parties, the approximate output $g(f(x))$, and *the ideal output (exact combination function)* $f(x)$, can simulate the view of the real world adversary. This privacy guarantee is appropriate in applications where the parties do not mind disclosing $f(x)$ (they compute $g(f(x))$ only for efficiency reasons), but do not want any further information leaked.

3. Private multiparty sampling

In this section we study the complexity of a *private multiparty sampling* primitive, which serves as a useful building block for private approximation protocols. This primitive allows M parties, each holding a database x^m, to privately obtain $f(x_r^1, \ldots, x_r^M)$ where f is some fixed M-argument functionality (say, exclusive-or) and r is an index of a *random* entry that should remain secret. (This is an instance of the general scenario described in Section 1 where g is a randomized projection function. Recall that in this scenario we assume M is much smaller than N.) A similar primitive was previously used in a two-party setting in the context of private approximations [17,20,32].

More formally, the sampling functionality induced by f is defined below. We define *private multi-party sampling of* f as an $(M-1)$-private protocol for this functionality.

Definition 3 (*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 $x^m = (x_n^m)_{n \in [N]}$.
- The functionality picks a secret, uniformly random index $r \in [N]$ and outputs $f(x_r^1, x_r^2, \ldots, x_r^M)$, where the ith output of f is given to party P_i.

When the only restriction on the communication complexity is to be polynomial, 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 [27,28,35]. Interestingly, such a construction becomes more challenging to obtain in the domain of sublinear-communication protocols. Even more interestingly, 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 is given in [17]. This construction maintains the communication complexity of the underlying OT protocol.[11] 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 show the following constructions of communication-efficient, multiparty private sampling protocols from PIR:

1. An "almost black-box" construction (in a sense to be explained below) of private sampling from a *one-round* (two-message) PIR protocol.
2. A *non-black-box* construction of private sampling from an *arbitrary* PIR protocol.[12]

The rest of this section is organized as follows. In Section 3.1 we define a distributed-OT primitive (and show an interesting application of it). In Section 3.2 we reduce general private multi-party sampling to distributed OT, and in Section 3.3 we reduce distributed OT to PIR (or sublinear OT).

3.1. Oblivious transfer with distributed receiver

Towards implementing the private multiparty sampling primitive, we introduce and study a distributed variant of oblivious transfer which is of independent interest. (We describe one application Section 3.1.1.)

In this distributed OT primitive the role of the receiver is distributed between the M parties.[13] 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. (Our approach can be applied to non-Abelian groups as well; however, since we only need to

[10] We note that [17] only gave the definition for a function g that satisfied some specific approximation requirement with respect to f, e.g., ϵ-approximation. In this paper we accept as an approximation whatever function g the parties had agreed in advance to compute as a useful summary of f, and focus on guaranteeing privacy of the protocol. Also, [17] gives the definition only for two parties, but the extension to multiple parties is straightforward.

[11] The protocol from [17] is described for the special case where f is the exclusive-or function but can be generalized (as was done in [20,32]) to arbitrary functions f. The following 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.

[12] To guarantee that the communication complexity of the sampling protocol is sublinear, we must assume that the receiver's *computational* complexity in the PIR protocol is sublinear. This is required by some definitions of PIR (e.g., [7,41]) and is achieved by all known constructions.

[13] A very different variant of distributed OT was considered in [47]. The protocols from [47] distribute the role of the sender (rather than the receiver) and do not address the question of obtaining sublinear communication complexity.

employ Abelian groups, we use additive notation for convenience.) 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 4 (*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 $x_{n_1+\cdots+n_M}$. Other parties have no output.

We will sometimes assign the identities of the "sender" and "receiver" (the party obtaining the output) to other pairs of parties (rather than P_1 and P_M), or even assign both roles to the same party.

3.1.1. An application to sublinear-communication multi-party protocols

We now interrupt the presentation of the steps leading to our main result, in order to describe an application of DistOT which is of independent interest. The reader who is eager to continue may skip ahead to Section 3.2, where we show how to use DistOT for implementing private multiparty sampling (Sample-f).

The DistOT primitive we defined above can also be used as a basic building block for sublinear-communication multiparty protocols. Specifically, we describe here how it can be used to provide a communication-preserving multiparty general transformation of non-private to private protocols, generalizing the protocol compiler from [45] to the multiparty case.

The protocol compiler from [45] transforms a non-private two-party protocol π into a private two-party protocol π' with roughly the same communication complexity. Our multi-party variant does the same in the case of M parties, where π' is private against arbitrary collusions. In the following we allow the amount of local computation in π' to be exponential in the communication; as in [45], the computation can be made polynomial in the communication for a restricted class of protocols π.

For simplicity, suppose that in each round j of π a prescribed party P_{i_j} sends the same message m_j to all other parties, where this message depends on the input x_{i_j} of P_{i_j} and the transcript c_{j-1} of the communication exchanged in the first $j-1$ rounds. The idea is to maintain the invariant that all parties have a share of the entire history at every step. Now, for round j of π, rather than sending m_j in the clear, the protocol π' additively shares m_j among the parties. This is done by using DistOT with party P_{i_j} holding a database of all possible next messages m_j indexed by the possible histories c_{j-1}, and all parties use their shares as the shares of the index. We describe this in more detail below.

The protocol π' has a phase for each round of π. In the beginning of phase j, the transcript c_{j-1} is additively shared between the M parties. The goal of phase j is to additively share m_j between the parties without revealing any additional information. This is done using DistOT in the following way. Let DistOT′ be a variant of DistOT in which the receiver's output is additively shared between the M parties. (A simple reduction from DistOT′ to DistOT is to let the sender mask all entries of x with the same random group element, and additively share the mask between all parties other than the receiver.)

Phase j of π' proceeds as follows. Party i_j, playing the role of the sender in DistOT′, prepares a database with $N = 2^{|c_{j-1}|}$ entries, where entry n contains the message m_j that P_{i_j} would send given its input and assuming that the shares of c_{j-1} held by all $M-1$ other parties add up to n. Now the parties invoke DistOT′, where P_{i_j} uses 0 as its share of the selection index and all other parties use their shares of c_{j-1}. As a result, the next message m_j is additively shared between all parties, as required.

In [45], the authors point out that certain special protocols can be made into private protocols with less computational cost than the general result. This is the case, for example, if only a limited number of transcripts are possible at some stage in the protocol. We note that our generalization to more than two parties captures this efficiency as well.

3.2. Private multiparty sampling from distributed OT

We now present an efficient black-box construction of private multiparty sampling protocols from distributed OT. We start by observing that Sample-f can be efficiently reduced to a simpler (randomized) functionality Sample-AS, and then complete the proof by showing how to reduce Sample-AS to DistOT.

Consider the randomized functionality AS (for "additive sharing"), which on M inputs, outputs an M-tuple of strings that are random subject to the restriction that their exclusive-or is the concatenation of the M inputs. Now, consider the randomized sampling functionality Sample-AS (namely Definition 3 with the functionality AS).

Proposition 3.1. *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.

Proof. We start by noting that 1-out-of-2 OT can be implemented from Sample-AS, using a result in [13]. Thus, the reduction may use 1-out-of-2 OT in addition to Sample-AS.

The reduction proceeds by first invoking Sample-AS to obtain an additively shared representation of the M inputs to f, and then running the general-purpose constant-round protocol of Theorem 2.1 (based on 1-out-of-2 OT) to compute f from these shares. □

We now turn to reducing Sample-AS to DistOT. To gain intuition, we first give a reduction of the simpler Sample-ID to DistOT, where ID is the identity function (namely each output is a concatenation of all M inputs directly — not additively shared as in Sample-AS). Once we present this reduction, we show why a naive approach to completing the proof by reducing

Sample-AS to Sample-ID does not work. However, this approach (together with the Sample-ID reduction) is the one that inspires and helps understand our final construction, directly reducing Sample-AS to DistOT.

The following is a reduction of Sample-ID to DistOT_G where G is an arbitrary group of size N. In the following, we arbitrarily identify elements of G with indices in $[N]$.

Proposition 3.2. *There is a constant-round black-box* $(M-1)$*-private reduction of* Sample-ID *to* DistOT. *The reduction makes* M *calls to* DistOT.

Proof. A simple reduction as required proceeds as follows.

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 the 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 M values $x^m_{r_1+\cdots+r_M}$ for $1 \le m \le M$ and sends them to all parties.
3. Each party outputs the M values obtained in the previous step.

The correctness of the above reduction is straightforward to verify. Privacy follows from the fact that a coalition involving a strict subset of the parties can learn just a strict subset of the random choices r_i, which are jointly independent of the final selection index $r_1 + \cdots + r_M$. □

Given this reduction, it would suffice 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 learns 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 get around the above problem, we need to generate the masks in a completely distributed way. We achieve this by using DistOT over the group $G' = G \times \mathbb{Z}_2$. The reduction proceeds as follows.

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 M values $(x')^m_{(r_1,b_{1,m})+\cdots+(r_M,b_{M,m})} = x^m_{r_1+\cdots+r_M} \oplus (b_{1,m} \oplus b_{2,m} \oplus \cdots \oplus b_{M,m})$ for $1 \le m \le M$.
3. Each party P_m, $m < M$, outputs the M-tuple $(b_{m,1}, b_{m,2}, \ldots, b_{m,M})$. Party P_M outputs the exclusive-or of $(b_{M,1}, b_{M,2}, \ldots, b_{M,M})$ with the M-tuple obtained in Step 2 above.

Note that this reduction is totally non-interactive, namely there are no messages sent between the parties outside of the black-box invocation of DistOT.

Proposition 3.3. *The reduction described above is a non-interactive,* $(M-1)$*-private black-box reduction from* Sample-AS *to* DistOT.

Proof. The correctness and privacy of the above reduction are similar to the reduction of Sample-ID to DistOT given in Proposition 3.2.

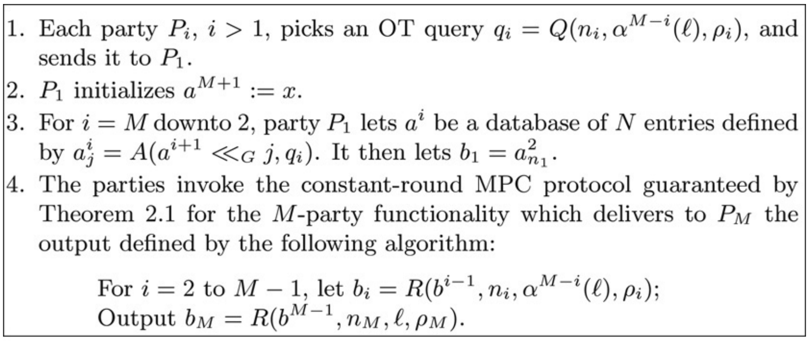
For correctness, consider the jth output bit of the M parties for some $1 \le j \le M$. The first $M-1$ of these output bits are random and independent bits $b_{m,j}$, $1 \le m \le M-1$, whereas the last bit is $x^j_{r_1+\cdots+r_M} \oplus (b_{1,j} \oplus b_{2,j} \oplus \cdots \oplus b_{M,j}) \oplus b_{M,j} = x^j_{r_1+\cdots+r_M} \oplus (b_{1,j} \oplus b_{2,j} \oplus \cdots \oplus b_{M-1,j})$. Thus, the parties hold random additive shares of $x^j_{r_1+\cdots+r_M}$ as required.

Privacy follows from the fact that any strict subset of the parties learns no information about any of the x^j or the selection index $r_1 \oplus \cdots \oplus r_M$. Thus, the view of any strict subset of the parties can be simulated by picking their random inputs $(r_m, b_{m,j})$ as well as the M outputs of DistOT, if P_M is corrupted, uniformly at random. □

Putting the reductions together, we obtain the following.

Theorem 3.1. *For any polynomial-time computable* M*-argument function* f*, there is an efficient, constant-round, black-box* $(M-1)$*-private reduction of* Sample-f *to* DistOT.

Proof. The theorem follows directly by combining Propositions 3.1 and 3.3. □



1. Each party P_i, $i > 1$, picks an OT query $q_i = Q(n_i, \alpha^{M-i}(\ell), \rho_i)$, 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 $a^i_j = A(a^{i+1} \ll_G j, q_i)$. It then lets $b_1 = a^2_{n_1}$.
4. The parties invoke the constant-round MPC protocol guaranteed by Theorem 2.1 for the M-party functionality which delivers to P_M the output defined by the following algorithm:

   For $i = 2$ to $M - 1$, let $b_i = R(b^{i-1}, n_i, \alpha^{M-i}(\ell), \rho_i)$;
   Output $b_M = R(b^{M-1}, n_M, \ell, \rho_M)$.

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

3.3. Implementing distributed OT from PIR

It remains to implement DistOT. In this section we show a general construction based on (sublinear 1-out-of-N) OT (equivalently, PIR).

We use the following notation. 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 $G = \mathbb{Z}_N$ 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 simplify notation by omitting the group subscript.

3.3.1. An "almost black-box" construction of DistOT *using one-round PIR.*

A one-round OT for a database x with N entries, each of length ℓ, with receiver interested in entry n, can be specified by three algorithms: a randomized query algorithm $Q(n, \ell, \rho)$ (where ρ is the receiver's secret randomness), an answering algorithm $A(x, q)$ and a reconstruction algorithm $R(a, n, \ell, \rho)$. We denote the length of the answers of A as $\alpha(\ell)$. (We make the security parameter k and the size N of the database implicit in the above notation.) We note that the general transformation from [46,13] gives a black-box construction of one-round sublinear-communication OT from one-round PIR.

In the following we present an efficient construction of a DistOT protocol from any one-round OT protocol. This construction makes a black-box use of the query algorithm Q and the answering algorithm A, but requires a non-black-box use of the reconstruction algorithm R. In particular, to obtain DistOT protocols with sublinear communication the *computational complexity* of R should be sublinear; as mentioned before, this is required by some definitions of PIR (e.g., [7,41]) and is achieved by all known constructions.

The DistOT protocol 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 jth entry of a shifted database y is simply the answer to the OT query on $y \ll j$. 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 $x \ll (\sum_{m=2}^{M} n_m)$, where each entry is of size $\alpha^{M-2}(\ell)$ (where $\alpha^i(\cdot)$ is the function defined by i iterations of α). The (n_1)th entry of this array contains an iterated encryption of the output $x_{n_1+\cdots+n_M}$. This output is privately computed and delivered to P_M using a general-purpose MPC protocol. This step requires a non-black-box use of the reconstruction algorithm R. The reduction is formally described in Fig. 1.

The correctness of the reduction is easy to verify. We now argue its privacy.

Proposition 3.4. *The reduction described in Fig. 1 is $(M - 1)$-private.*

Proof. The intuition for the privacy was already given above. Formally, we construct a simulator whose input consists of the inputs of corrupted parties and, if P_M is corrupted, the output $x_{n_1+\cdots+n_M}$. By a composition theorem for secure computation [9,26], it suffices to describe the simulator in a hybrid model in which Step 4 is replaced by an ideal function evaluation process that directly delivers the output to P_M. In this model simulating the view of any strict subset of parties is straightforward. The OT random inputs ρ_i are picked uniformly at random. If P_M is corrupted, then the message it receives in the ideal function evaluation process is simply the output of DistOT. Finally, if P_1 is corrupted then the OT-query received from any uncorrupted party P_i is simulated as $q_i = Q(0, \alpha^{M-i}(\ell), \rho_i)$, where ρ_i is picked at random. Note that the only difference between the simulation and the real protocol execution is that in the latter q_i is generated by using the actual input n_i instead of 0. The security of the PIR protocol (together with a standard hybrid argument) imply that the simulated view is indistinguishable from the real view. □

The complexity of the above reduction depends on the number M of parties, the function α (namely the relation between the size of the answers of the OT protocol and the length of the database entries ℓ), and the computational complexity of the OT reconstruction algorithm R. Ideally, we have $\alpha(\ell) = \ell + k \cdot polylog(N)$, where k is the security parameter. (Indeed, such an OT protocol can be based on the Damgård–Jurik encryption scheme [14,41].) In this case, the length of the input to the MPC protocol run in Step 4 is $\ell + O(M)k \cdot polylog(N)$. Thus, the protocol is efficient also for non-constant M. When the number M of parties is viewed as constant, the DistOT protocol can be made communication-efficient even if, say, $\alpha(\ell) = \mathsf{poly}(\ell, k) \cdot N^{\epsilon}$ for a small $\epsilon > 0$. Such an OT protocol can be based on an arbitrary homomorphic encryption scheme [37,52].

The above leads to the following theorem.

Theorem 3.2. *There is a constant-round $(M-1)$-private reduction of* DistOT *to any one-round OT protocol (Q, A, R). The reduction makes a black-box use of the query generation algorithm Q and the answering algorithm A. The resulting* DistOT *protocol has sublinear communication if one of the following two properties holds:*

- *The communication complexity and the circuit size of R are $\ell + k \cdot polylog(N)$, or*
- *The number of parties M is constant, and the above complexities are $\mathsf{poly}(\ell, k) \cdot N^{o(1)}$.*

Remark 1. In the conference version of this paper [33], it was suggested to implement the final step of the above DistOT protocol in a black-box way by passing the iterated encryption of the output between the parties, letting each party in its turn peel off its own layer of encryption using the OT reconstruction algorithm R. The underlying intuition is that it is safe to pass these encryptions between the parties because they contain no more information than the output. However, this intuition is flawed (and the protocol is insecure) for the following reason: Since the intermediate encryptions revealed during the peeling process were all generated by P_1, it is easy for P_1 to collude with, say, P_3 and completely break the privacy of P_2.

3.3.2. A non-black-box construction of DistOT *from arbitrary OT.*

The previous construction was restricted to OT (or PIR) protocols with only one-round. We now sketch a simple non-black-box approach for obtaining DistOT from an arbitrary OT protocol, which may involve multiple rounds of interaction. Combined with PIR-based OT protocols from the literature, this approach yields constant-round DistOT protocols in which the communication complexity is polylogarithmic in N and the computational complexity is quasilinear in N. However, the protocols obtained via the previous "almost black-box" approach have a better concrete efficiency.

For general R-round OT, we use general-purpose constant-round MPC for distributing all the secret information held by the OT receiver. More concretely, the reduction proceeds as follows.

Reducing DistOT to OT (non-black-box reduction)

1. Each party P_m, $m \leq M$, picks a random additive share r_m for the receiver's randomness in the OT protocol. Party P_1 additionally picks randomness ρ for the sender in the OT protocol. Parties P_1 and P_M initialize empty transcripts T_1 and T_M, respectively.
2. For $1 \leq h \leq R$ the parties do the following:
   - The parties use the constant-round OT-based $(M-1)$-private MPC protocol guaranteed by Theorem 2.1 to compute the message sent by the OT receiver in Round h given input $n \sum_{j \leq M} n_j$, randomness $r = \sum_{j \leq M} r_j$ and the transcript T_M of messages received so far. (The OT calls made by the MPC protocol are emulated by invoking the given OT primitive.) The output Q_h is revealed to P_1, who appends it to the transcript T_1.
   - Party P_1 *locally* computes its hth OT answer A_h based on its input x, its private randomness ρ, and the queries in T_1 received in previous iterations. The answer A_h is sent to P_M, who appends it to the transcript T_M.
3. The parties use the constant-round $(M-1)$-private MPC protocol guaranteed by Theorem 2.1 to compute the output of the OT receiver given its input $n = \sum_{j \leq M} n_j$, randomness $r = \sum_{j \leq M} r_j$, and transcript T_h of R messages received from the OT sender. The output is revealed to P_M.

The correctness of the above DistOT protocol follows in a straightforward way from that of the OT protocol and the MPC protocol. We now argue its privacy.

Proposition 3.5. *The above protocol is an $(M-1)$-private realization of* DistOT.

Proof. Similarly to proof of Proposition 3.4, it suffices to show simulation in a hybrid model in which the invocations of the MPC protocol are replaced by ideal oracle calls to the corresponding functionality. Intuitively, privacy in this model follows from the fact that a strict subset of the parties cannot learn any information about the effective randomness r of the distributed receiver.

More precisely, let $T \subset [M]$ be the set of corrupted parties. The simulator picks the random strings r_i, $i \in T$, at random. Suppose first that P_1 is uncorrupted. If P_M is also uncorrupted, then there is nothing else to simulate (since only P_1 and P_M receive any messages). If P_M is corrupted with output y, the simulator runs the OT simulator, feeding it with some default value for the receiver's (unknown) input and with y as the receiver's output. The simulator returns the sequence T_M of messages from the OT sender to the OT receiver produced by the OT simulator, along with y as the final message received by P_M in the DistOT protocol. Note that the only difference between the simulation and the real protocol (in the hybrid model discussed above) is that the receiver's selection in the OT protocol is fixed to some default value instead of the correct value i. The privacy of the OT protocol guarantees that the messages T_M sent by the sender in the two cases are computationally indistinguishable.

Suppose now that P_1 is corrupted. If P_M is uncorrupted, the simulator runs the sender's OT simulator (with x as sender's input) and outputs the sender's randomness ρ and sender's incoming messages T_1 produced by the simulator. If P_M is also corrupted, the simulator can simply run the OT protocol (picking the sender and receiver's randomness) and obtain ρ, T_1, T_M from the output of this execution. Here too, the correctness of the simulator follows from the privacy of the OT protocol and the independence between the randomness r_i of parties in T and the receiver's randomness r. □

The round complexity of DistOT is quadratic in the round complexity R of OT, since there are R iterations in which a constant-round MPC protocol is invoked, and in each round of the latter protocol OT may be invoked. The communication complexity of DistOT is dominated by the circuit size of the functionalities being computed via general-purpose MPC, which is bounded by the total computational complexity of the OT receiver (measured in terms of circuit size). Note that we only emulated the role of the *receiver* in a distributed way; thus, the communication complexity is not affected by the computational complexity of the OT sender. We conclude that the following theorem holds.

Theorem 3.3. *There is an $(M-1)$-private, non-black-box reduction of M-party* DistOT *to (two-party) OT satisfying the following efficiency requirements. If OT has round complexity R and receiver computational complexity C, then the resulting* DistOT *protocol has round complexity $O(R^2)$ and communication complexity $\mathsf{poly}(C, M)$.*

4. Private approximation of vector combinations

In this section, we consider $M \geq 3$ parties holding length-N vectors $\mathbf{x} = x^1, \ldots, x^M$ who, ideally, want to compute (and are willing to disclose to the other parties) $y = f(\mathbf{x})$ for a combination function f being either the componentwise sum (denoted $y = \sum_m x^m$), componentwise minimum (denoted $y = \bigwedge_m x^m$), or componentwise maximum (denoted $y = \bigvee_m x^m$) of their vectors. Because the length N of all vectors is presumed to be large, the parties settle for an approximate size-t summary $Y = g(y)$ for y (e.g., an estimate of the ℓ^2-norm, or an approximate t-term Fourier representation), but the parties want very efficient protocols.

We show how to exploit the fact that the entire (long) vector $y = f(\mathbf{x})$ may be leaked in order to obtain simple private protocols for the above problems, in which the cryptographic overhead is at most polynomial in the size of the small approximation and polylogarithmic in N. This should be contrasted with most previous protocols for sublinear-communication private computation of approximations (as well as the results of Section 3), which given state-of-the-art PIR require a cryptographic computational overhead which is linear in N. Typically, the linear overhead is unavoidable for the stricter notion of privacy (private computation of an approximate function $g(f(\mathbf{x}))$). As discussed, we avoid it by using the more liberal definition of private approximation (namely computing $g(f(\mathbf{x}))$ with privacy with respect to $f(\mathbf{x})$). We note that while our result for component-wise sum works for vectors over any domain (as long as the elements are of polynomial length), the main result for component-wise minimum works only for positive valued vectors, and we show how to extend it to work for domains including 0, such as Boolean values.

Technical outline. The results of this section are based on the observation that most (non-private) approximations from the literature can be derived from short local randomized sketches that can be efficiently combined to yield the desired approximation. In previous uses of such sketches in the context of private approximations, it was necessary to hide the randomness that was used for generating the samples. Since in our setting we allow leaking the entire vector y, we can use *public* (pseudo-)randomness for locally producing the sketches, and then obtain the output via generic secure computation of simple functions (addition, maximum or minimum) on the short sketches.

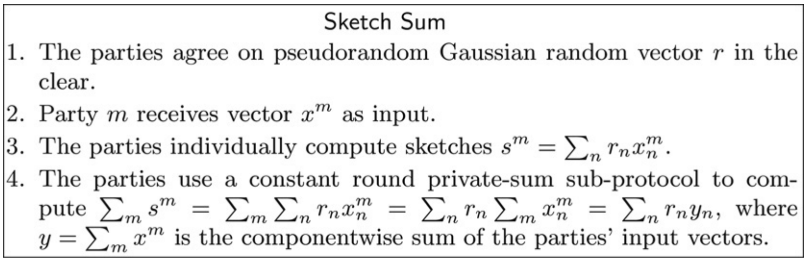
All results in this section use the above outline to privately approximate different norms of the desired combination vector, which are then used as in previous work to estimate the desired outputs. Specifically, we first show how to privately sample a Gaussian random variable with parameter the ℓ^2-norm of the sum: the sketch uses a pseudorandom Gaussian vector r, local computation of an inner product $\sum_n r_n x_n^m$ (over long vectors), followed by a generic secure computation of a sum function of the short sketches. This can then be used to estimate the ℓ^2-norm of the sum, and consequently produce several summaries of vector sum, exactly in the same way as in the non-private literature.

We then use similar techniques to privately sample an exponential random variable with parameter the harmonic mean (-1-norm) of the vector minimum; here, the sketch uses a pseudorandom exponential vector r, local computation of $\bigwedge_n r_n x_n^m$, and secure computation of a minimum function over the short sketches. This can then be used as in the non-private literature to estimate the -1-norm of the vector minimum, and, consequently, the vector minimum itself. We also show how to extend the above protocol for privately estimating the p-norm of the vector minimum for any $p < 0$. Finally, we use an analogous technique to estimate the p-norm of the vector maximum for any $p > 0$ (in particular, estimating the vector maximum). The same technique holds for other estimators built from *linear* projections of the data, including [31,38].

4.1. Vector sums

Following the approach outlined above, we start by showing a protocol for sampling a Gaussian random variable with variance the ℓ^2-norm of the vector sum.

Here and in the following, when we refer to a Gaussian or exponential random variable we actually refer to discrete approximations of these variables, since we are limited to finite precision. In standard algorithmic contexts this distinction is not important, but in the context of private approximations this might be a source of trouble—for example, the sum of two *discretized* Gaussian random variables is typically distinguishable from a single *discretized* Gaussian random variable. Below



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 constant round private-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.

we give an efficient protocol that is perfectly private in the model of computation over the reals. In [17], we showed how to convert such protocols into statistically private protocols in the model of computation over finite-precision arithmetic.

Proposition 4.1. *There is an $(M-1)$-private approximation of the sum function $y = f(\mathbf{x}) = \sum_m x^m$, which generates a Gaussian random variable with mean zero and variance $\sum_n y_n^2$, satisfying the following properties. The protocol uses two-party OT in a black-box manner, has local computation complexity of $NM^{O(1)}$, communication $M^{O(1)}$, and round complexity $O(1)$.*

Proof. It is known [18] that, 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, $\sum_n y_n^2$. This was exploited in [31]. 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 an $(M-1)$-private constant-round protocol (with black-box use of two-party OT) for adding up the M (short) integers s^m (e.g., the general-purpose private protocol of [3]). The protocol is described in Fig. 2.

It is straightforward to confirm that the communication of the protocol in Fig. 2 is as claimed, because the secure-sum protocol is applied to M *numbers* s^m, and not length-N vectors. It is also straightforward to confirm that the protocol does not leak anything beyond y to any subset of parties, as long as the underlying general-purpose private sum protocol on the short sketches is indeed private. Specifically, a simulator on input y samples r, computes $\sum_n r_n y_n$, and then invokes the simulator of the underlying private computation protocol to simulate the transcript. □

4.1.1. Vector sum summaries

From a small number of samples of a Gaussian random variable Y with mean zero, the variance $\sum_n y_n^2$ can be estimated with high accuracy. This is because, if Z is a sample of Y^2, then $E[Z] = \sum_m y_n^2$ and $\mathrm{var}(Z) \le O(E^2[Z])$. Thus the average W of $O(1/\epsilon^2)$ independent copies of Z is, with probability at least 7/8, in the range $(1 \pm \epsilon) \sum_n y_n^2$. If we take a median of $O(\log(1/\delta))$ independent copies of W, the result is in the range $(1 \pm \epsilon) \sum_n y_n^2$ with probability at least $1-\delta$.

Note that $\left(\sum_n y_n^2\right)^{1/2}$ is itself a useful summary of y. It is also possible to use these and similar sketches of y to estimate other quantities, such as individual components y_n, t-term heavy-hitter, Fourier, wavelet, piecewise-constant, etc., approximation; ℓ^2 error within $(1+\epsilon)$ of optimal [1,36,43,25,24,8,15]. For example, y_i can be approximated [1] as $Y_i = r_i \sum_n r_n y_n$ since $E[Y_i] = y_i$ and $E[Y_i^2] \le O\left(\sum_n y_n^2\right)$; as above, taking a median of $O(\log(1/\delta))$ means of $O(1/\epsilon^2)$ independent copies of Y_i yields an estimate in the range $y_i \pm \epsilon\sqrt{\sum_n y_n^2}$ with probability at least $1-\delta$. As for privacy, we note that any approximation built from a sketch of y leaks at most y and does not otherwise depend on the individual vector contributions x^m. Since our protocol in Fig. 2 is private with respect to y, it follows that any other summary produced from the output of the protocol (independent of the inputs) is also private with respect to y.

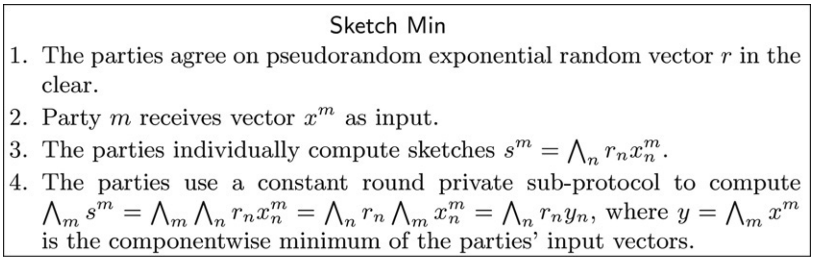
4.2. Vector minima

In this section, we generalize the above protocol to the componentwise *minimum* instead of the componentwise *sum*. Here, 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., [11] for example uses in algorithms of estimating the parameter of an exponential random variable. (We also give a toy example below.)

Recall that $\bigwedge$ denotes the minimum. We have:

Proposition 4.2. *There is an $(M-1)$-private approximation of the minima function $y = f(\mathbf{x}) = \bigwedge_m x^m$ over vectors of positive integers, which generates an exponential random variable with parameter $\left(\sum_n y_n^{-1}\right)$, satisfying the following properties. The protocol uses two-party OT in a black-box manner, has local computation complexity of $NM^{O(1)}$, communication $M^{O(1)}$, and round complexity $O(1)$.*

Proof. It is known [18] that, if each component r_n of a vector r is a unit exponential random variable, then $\bigwedge_n r_n y_n$ 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.



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 constant round private 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.

The parties use the protocol of Fig. 3. The sub-problem 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 Theorem 2.1). Otherwise, it is straightforward to check, in a way completely analogous to the protocol in Fig. 2, that all the desired properties of this protocol hold. □

As above, from $O(\log(1/\delta)/\epsilon^2)$ of samples of an exponential random variable X, the parameter can be estimated accurately. This is because the parameter is $E[X]$ and $\text{var}(X) \le O(E^2[X])$.

4.3. Extensions to vector maxima and other approximations

Note that the harmonic mean of y is the -1st p-norm of y. We can use these techniques instead to estimate other p-norms, for $p < 0$, as follows. Party m holds x^m and takes the $-p$th power of each component, getting $(x^m)^{-p}$. Then the parties proceed as above to estimate the -1st norm of $\bigwedge_m (x^m)^{-p}$, i.e., $\sum_n \bigwedge_m (x_n^m)^p = \sum_n \left(\bigwedge_m x_n^m\right)^p = \sum_n y_n^p$. Finally, the parties take the pth root, getting $\left(\sum_n y_n^p\right)^{1/p}$. Here we have exploited the fact that, for $p < 0$ and any two positive numbers a and b, we have $(a \wedge b)^{-p} = a^{-p} \wedge b^{-p}$. Similarly, for $p > 0$, we can estimate the pth norm of the componentwise *maximum* of the x^m, by observing that, for $p > 0$, we have $(a \vee b)^{-p} = a^{-p} \wedge b^{-p}$, where $\vee$ denotes the maximum. This was presented in [53]. In [53], it is also shown how to use (several independent copies of) an exponential random variable Y with parameter $\left(\sum_n y_n^{-1}\right)$ to estimate other quantities, such as individual components y_i for i chosen *after* the creation of Y.

Additional advantages of the min-sketch over the sum-sketch can be found in the Appendix. Specifically, we show that rounding can be done on min-sketches without destroying privacy, whereas rounding destroys privacy in general for sum-sketches. Furthermore, min-sketches fit nicely into the order- and duplication-insensitive framework of [48]; i.e., when combining the min-sketches of several parties by taking a minimum, it does not matter if some party's contribution is taken more than once. By contrast, a sum-sketch is ruined if a party contributes twice. (The order in which the contributions are taken is immaterial for both min- and sum- sketches.)

As an illustration, we briefly sketch an application of these techniques to the private approximation of set intersection size.[14] There are M parties and the mth party holds the 0/1-valued, length-N vector z^m, regarded as the characteristic vector of a set over a universe of size N. We assume each set has size $N/4$. The goal is to estimate the size of the intersection of the sets, i.e., $\sum_n \bigwedge_m z_n^m$. To proceed, the parties each form a vector x^m with $x_n^m = z_n^m + 1$, so x^m takes values in $\{1, 2\}$. Let y denote $\bigwedge_m x^m$. Then $y_n = 2$ if all sets contain n, and $y_n = 1$ otherwise, so that $\sum_n y_n^{-1} = N - |I|/2$, where I is the set intersection. The parties can use the above protocol with cost polynomial in $1/\epsilon$ to estimate $|I|$ up to ϵN additively, leaking only I itself.

Acknowledgments

We thank Stillian A. Stoev for helpful discussions and the anonymous referees for useful comments.

The first author was supported in part by ISF grant 1310/06 and BSF grant 2004361; the work was partially completed while he was visiting UCLA. The second author was supported in part by NSF grant CCF-0347839 and NSF grant CNS-0716245. The third author was supported in part by NSF grant DMS-0510203 and NSF grant DMS-0354600. The fourth author was supported in part by NSF grant CNS-0822269.

Appendix. More simple sketch-based data mining

A.1. Rounding

We now address a particular efficiency available to min-sketches. Some protocols for computing the minimum of input numbers have cost that depends on the number of bits in the input. In our approximate setting, it is often possible to round

[14] There are other solutions to this problem that are more straightforward. For example, one party broadcasts in the clear a random index into the vectors and then the parties do a private AND on just those M bits; the parties can repeat the protocol to drive down distortion and failure probability.

input numbers z to $\lceil z \rceil$, a power of $(1+\epsilon)$ for some parameter ϵ. In the case of sum-sketches, this kind of rounding destroys privacy [17]. For min-sketches, however, privacy is preserved.

For example, consider exponential random variables, which are $(\bigwedge, \ell^{-1})$-stable. Without the rounding, our sketch is $\bigwedge_m \bigwedge_n x_n^m r_n$, an exponential random variable, where r_n are exponential random variables. Note that the process of rounding up to a power of $(1+\epsilon)$ (denoted here by $\lceil \cdot \rceil$) distributes over minimization, so it follows that $\bigwedge_m \left\lceil \bigwedge_n x_n^m r_n \right\rceil = \left\lceil \bigwedge_m \bigwedge_n x_n^m r_n \right\rceil = \left\lceil \bigwedge_n r_n \bigwedge_m x_n^m \right\rceil = \left\lceil \bigwedge_n r_n y_n \right\rceil$, a value that depends only on y, which is allowed to leak.

A.2. Loosely connected network

In the non-private situation, min-sketches are advantageous in loose networks because they are (order- and) *duplication-insensitive* (ODI) as first studied for sensor networks in [48]. That is, suppose M parties are in a network with less-than-full connectivity, such as a sensor network. To compute the minimum of M numbers, parties can simply broadcast their number to whichever other parties are in earshot. When a party hears a broadcast, the party replaces a currently stored value with the minimum of that value and the broadcast value. Such a protocol is correct provided all the relevant parties are connected and the number of rounds is at least twice the diameter of the graph. Precise connectivity information is not needed by the algorithm or its analysis, since it does not matter if a party's contribution counts more than once toward the final sketch. Note that the sum of contributions cannot be computed in the straightforward way in this model, since we do not want a contribution to count more than once. Nevertheless, an approximation algorithm for the sum of non-negative numbers was presented in [48] based on sketches that are ODI.

In our context, we note that there are privacy advantages to ODI sketches for loosely networked parties. Each party only needs to know about its neighbors and the diameter of the network; global connectivity information is not needed and does not leak in the protocol.

Finally, we note that some security guarantees must necessarily be weakened in the loose-connectivity model. For example, if the removal of a small set of one or more vertices disconnects the graph, then the corresponding set of parties can mount a denial of service attack, at the very least. This will be the subject of future work. In this paper, our goal is to reduce the problem of private approximate computation on large data vectors to the problem of private exact computation on short sketches; this reduction holds in an ODI-preserving way in the loose connectivity model for min-sketch-based approximation, but not immediately for sum-sketch-based approximation.

References

[1] N. Alon, P. Gibbons, Y. Matias, M. Szegedy, Tracking join and self-join sizes in limited storage, J. Comput. System Sci. 64 (3) (2002) 719–747, Earlier version in Proc. PODS'99.

[2] D. Beaver, Foundations of secure interactive computing, in: Advances in Cryptology — CRYPTO'91, 1991, pp. 377–391.

[3] D. Beaver, S. Micali, P. Rogaway, The round complexity of secure protocols, in: Proc. 22th Annual ACM Symposium on the Theory of Computing — STOC'90, 1990, pp. 503–513.

[4] A. Beimel, P. Carmi, K. Nissim, E. Weinreb, Private approximation of search problems, in: Proc. 38th Annual ACM Symposium on the Theory of Computing — STOC'06, 2006, pp. 119–128.

[5] A. Beimel, R. Hallak, K. Nissim, Private approximation of clustering and vertex cover, in: Proc. 4th Theory of Cryptography Conference — TCC'07, 2007, pp. 383–403.

[6] A. Beimel, T. Malkin, K. Nissim, E. Weinreb, How should we solve search problems privately? in: Advances in Cryptology — CRYPTO'07, 2007, pp. 31–49.

[7] C. Cachin, S. Micali, M. Stadler, Computationally private information retrieval with polylogarithmic communication, in: Advances in Cryptology — EUROCRYPT'99, 1999, pp. 404–414.

[8] E.J. Candès, J.K. Romberg, T. Tao, Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information, IEEE Trans. Inform. Theory 52 (2) (2006) 489–509.

[9] R. Canetti, Security and composition of multiparty cryptographic protocols, J. Cryptology 13 (1) (2000) 143–202.

[10] B. Chor, O. Goldreich, E. Kushilevitz, M. Sudan, Private information retrieval, J. ACM 45 (6) (1998) 965–981, Earlier version in Proc. FOCS'05.

[11] E. Cohen, Size-estimation framework with applications to transitive closure and reachability, J. Comput. System Sci. 55 (3) (1997) 441–453.

[12] G. Cormode, S. Muthukrishnan, Estimating dominance norms of multiple data streams, in: Proc. 11th European Symposium on Algorithms — ESA'03, 2003, pp. 148–160.

[13] G. Di Crescenzo, T. Malkin, R. Ostrovsky, Single database private information retrieval implies oblivious transfer, in: Advances in Cryptology — EUROCRYPT'00, 2000, pp. 122–138.

[14] I. Damgård, M. Jurik, A generalisation, a simplification and some applications of paillier's probabilistic public-key system, in: Proc. 4th International Workshop on Practice and Theory in Public Key Cryptosystems — PKC'01, 2001, pp. 119–136.

[15] D.L. Donoho, Compressed sensing, Unpublished manuscript, Oct. 2004.

[16] S. Even, O. Goldreich, A. Lempel, A randomized protocol for signing contracts, Commun. ACM 28 (1985) 637–647.

[17] J. Feigenbaum, Y. Ishai, T. Malkin, K. Nissim, M.J. Strauss, R.N. Wright, Secure multiparty computation of approximations, ACM Trans. Algorithms 2 (3) (2006) 435–472, Earlier version in Proc. ICALP'01.

[18] W. Feller, An Introduction to Probability Theory and Its Applications, vol. 1, Wiley, 1968.

[19] M.K. Franklin, M. Gondree, P. Mohassel, Multi-party indirect indexing and applications, in: Advances in Cryptology — ASIACRYPT'07, 2007, pp. 283–297.

[20] M. Freedman, K. Nissim, B. Pinkas, Efficient private matching and set intersection, in: Advances in Cryptology — EUROCRYPT'04, 2004, pp. 1–19.

[21] W.I. Gasarch, A survey on private information retrieval (column: Computational complexity), Bull. EATCS 82 (2004) 72–107.

[22] C. Gentry, Z. Ramzan, Single-database private information retrieval with constant communication rate, in: Proc. 32nd International Colloquium on Automata, Languages and Programming — ICALP'05, 2005, pp. 803–815.

[23] Y. Gertner, Y. Ishai, E. Kushilevitz, T. Malkin, Protecting data privacy in private information retrieval schemes, J. Comput. System Sci. 60 (3) (2000) 592–692, Earlier version in Proc. STOC'98.

[24] A. Gilbert, S. Guha, P. Indyk, Y. Kotidis, S. Muthukrishnan, M. Strauss, Fast, small-space algorithms for approximate histogram maintenance, in: Proc. 34th Annual ACM Symposium on the Theory of Computing — STOC'02, 2002, pp. 389–398.

[25] A. Gilbert, S. Guha, P. Indyk, S. Muthukrishnan, M. Strauss, Near-optimal sparse Fourier representations via sampling, in: Proc. 34th Annual ACM Symposium on the Theory of Computing — STOC'02, 2002, pp. 152–161.

[26] O. Goldreich, Foundations of Cryptography: Basic Applications, Cambridge University Press, 2004.

[27] O. Goldreich, S. Micali, A. Wigderson, How to play any mental game, in: Proc. 19th Annual ACM Symposium on the Theory of Computing — STOC'87, 1987, pp. 218–229.

[28] O. Goldreich, R. Vainish, How to solve any protocol problem—An efficiency improvement, in: Advances in Cryptology — CRYPTO'87, 1987, pp. 73–86.

[29] S. Goldwasser, S. Micali, Probabilistic encryption, J. Comput. System Sci. 28 (1984) 270–299.

[30] S. Halevi, E. Kushilevitz, R. Krauthgamer, K. Nissim, Private approximations of NP-hard functions, in: Proc. 33th Annual ACM Symposium on the Theory of Computing — STOC'01, 2001, pp. 550–559.

[31] P. Indyk, Stable distributions, pseudorandom generators, embeddings, and data stream computation, J. ACM 53 (3) (2006) 307–323.

[32] P. Indyk, D.P. Woodruff, Polylogarithmic private approximations and efficient matching, in: Proc. 3rd Theory of Cryptography Conference – TCC'06, 2006, pp. 245–264.

[33] Y. Ishai, T. Malkin, M.J. Strauss, R.N. Wright, Private multiparty sampling and approximation of vector combinations, in: Proc. 34th International Colloquium on Automata, Languages and Programming — ICALP'07, 2007, pp. 243–254.

[34] Jonathan Katz, Rafail Ostrovsky, Adam Smith, Round efficiency of multi-party computation with a dishonest majority, in: Advances in Cryptology — EUROCRYPT'03, 2003, pp. 578–595.

[35] J. Kilian, Founding cryptography on oblivious transfer, in: Proc. 20th Annual ACM Symposium on the Theory of Computing — STOC'88, 1988, pp. 20–31.

[36] E. Kushilevitz, Y. Mansour, Learning decision trees using the Fourier spectrum, in: Proc. 23th Annual ACM Symposium on the Theory of Computing — STOC'91, 1991, pp. 455–464.

[37] E. Kushilevitz, R. Ostrovsky, Replication is not needed: Single database, computationally-private information retrieval, in: Proc. 38th IEEE Symposium on Foundations of Computer Science — FOCS'97, 1997, pp. 364–373.

[38] P. Li, Estimators and tail bounds for dimension reduction in l_α $(0 < \alpha \leq 2)$ using stable random projections, in: 19th Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms — SODA'08, 2008, pp. 10–19.

[39] Y. Lindell, Parallel coin-tossing and constant-round secure two-party computation, in: Advances in Cryptology — CRYPTO'01, 2001, pp. 171–189.

[40] Y. Lindell, B. Pinkas, Privacy preserving data mining, J. Cryptology 15 (3) (2002) 177–206, Earlier version in Proc. Crypto'00.

[41] H. Lipmaa, An oblivious transfer protocol with log-squared communication, in: Proc. 8th Information Security Conference — ISC'05, 2005, pp. 314–328.

[42] E. Mann, Private access to distributed information, Master's Thesis, Technion - Israel Institute of Technology, Haifa, 1998.

[43] Y. Mansour, Randomized interpolation and approximation of sparse polynomials, in: Proc. 19th International Colloquium on Automata, Languages and Programming — ICALP'92, 1992, pp. 261–272.

[44] S. Micali, P. Rogaway, Secure computation, in: Advances in Cryptology — CRYPTO'91, 1991, pp. 392–404.

[45] M. Naor, K. Nissim, Communication preserving protocols for secure function evaluation, in: Proc. 33th Annual ACM Symposium on the Theory of Computing — STOC'01, 2001, pp. 590–599.

[46] M. Naor, B. Pinkas, Oblivious transfer and polynomial evaluation, in: Proc. 31st Annual ACM Symposium on the Theory of Computing — STOC'99, ACM Press, 1999, pp. 245–254.

[47] M. Naor, B. Pinkas, Distributed oblivious transfer, in: Advances in Cryptology — ASIACRYPT'00, 2000.

[48] S. Nath, P.B. Gibbons, S. Seshan, Z.R. Anderson, Synopsis diffusion for robust aggregation in sensor networks, in: Proc. 2nd International Conference on Embedded Networked Sensor Systems, 2004, pp. 250–262.

[49] R. Pass, Bounded-concurrent secure multi-party computation with a dishonest majority, in: Proc. 36th Annual ACM Symposium on the Theory of Computing — STOC'04, 2004, pp. 232–241.

[50] M.O. Rabin, How to exchange secrets by oblivious transfer, Technical Report TR-81, Aiken Computation Laboratory, Harvard University, 1981.

[51] O. Reingold, L. Trevisan, S.P. Vadhan, Notions of reducibility between cryptographic primitives, in: Proc. 1st Theory of Cryptography Conference — TCC'04, 2004, pp. 1–20.

[52] J.P. Stern, A new and efficient all-or-nothing disclosure of secrets protocol, in: Advances in Cryptology — ASIACRYPT'98, 1998, pp. 357–371.

[53] S. Stoev, M. Hadjieleftheriou, G. Kollios, M.S. Taqqu, Norm, point, and distance estimation over multiple signals using max–stable distributions, in: Proc. 23rd International Conference on Data Engineering — ICDE'07, 2007, pp. 1006–1015.

[54] A. Yao, Protocols for secure computation, in: Proc. 23rd IEEE Symposium on Foundations of Computer Science — FOCS'82, 1982, pp. 160–164.