Optimal-Round Preprocessing-MPC of Polynomials over Non-Zero Inputs via Distributed Random Matrix

DOR BITAN, Ben-Gurion University of the Negev, Israel
SHLOMI DOLEV, Ben-Gurion University of the Negev, Israel

The preprocessing model enables perfectly secure MPC in the presence of a dishonest majority, in doing so, circumventing a known no-go result regarding plain-model MPC. It was recently shown that this can even be done in an optimal number of rounds of communication, namely, two rounds. However, when the function to be evaluated is a polynomial with a possibly high degree over inputs taken from a large domain, in order to maintain perfect security against honest majority, existing solutions require either amount of memory exponential in the size of the domain or more than two rounds of communication.

We present here the first preprocessing-MPC schemes for high-degree polynomials over non-zero inputs, which have optimal round complexity, perfect security against coalitions of up to $N-1$ out of N parties, communication and space complexities that grow linearly with the number of monomials in the polynomial (independent of its degree), while using function-independent correlated randomness. We extend our results to the client-server model and present a scheme that enables a user to outsource the storage of non-zero secrets to N distrusted servers and have the servers obliviously evaluate polynomials over the secrets in a single round of communication, perfectly secure against coalitions of up to $N-1$ servers. Our schemes are based on a novel secret sharing scheme, *Distributed Random Matrix* (DRM), which we present here. The DRM secret sharing scheme supports homomorphic multiplications, and, after a single round of communication, supports homomorphic additions.

ACM Reference Format:
Dor Bitan and Shlomi Dolev. 2020. Optimal-Round Preprocessing-MPC of Polynomials over Non-Zero Inputs via Distributed Random Matrix. 1, 1 (November 2020), 20 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Keywords: MPC with preprocessing, Optimal round complexity, Homomorphic secret sharing, Two-party computation, Perfect security.

Mathematics Subject Classification: 68P25, 94A60, 94A62.

1 INTRODUCTION

Secure multiparty computation (MPC) is an extensively studied field in cryptography which discusses the following problem. N participants, $\mathcal{P}_1, \ldots, \mathcal{P}_N$, are holding secret inputs, $s_1, \ldots, s_N$, and wish to evaluate a function $y = f(x_1, \ldots, x_N)$ over their secret inputs while not revealing any

Research partially supported by the Lynne and William Frankel Center for Computer Science, the Rita Altura Trust Chair in Computer Science and also supported by a grant from the Ministry of Science, Technology and Space, Infrastructure Research in the Field of Advanced Computing and Cyber Security, the Israel & the Japan Science and Technology Agency (JST), and the German Research Funding Organization (DFG, Grant#8767581199).

Authors' addresses: Dor Bitan, Ben-Gurion University of the Negev, Israel, dorbi@post.bgu.ac.il; Shlomi Dolev, Ben-Gurion University of the Negev, Israel, dolev@cs.bgu.ac.il.

XXXX-XXXX/2020/11-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

information regarding their secret inputs to each other (except for what may be deduced from the output). A vast number of papers were written on that topic in the past four decades, e.g., [Yao82, GMW87, BOGW88, CCD88, BMR90, Riv99, DN03, ABT18], suggesting solutions to that problem based on different approaches. These solutions differ in their security and efficiency levels and their assumptions regarding the behavior of the parties and the communication setting.

Regarding the security level, MPC schemes may be either *information-theoretically secure* (IT-secure) or *computationally secure*. The security of computationally secure schemes is based on unproven computational hardness assumptions and assumes limitations on the computing power of possible adversaries. The security of IT-secure schemes is derived from information theory and is free of unproven assumptions. An IT-secure scheme which leaks some information is *statistically secure*. Otherwise, it is *perfectly secure*. The scope of this work is perfectly secure schemes.

In this work, we assume that parties are *honest-but-curious* (a.k.a. semi-honest). That is, parties follow the protocol, yet, they may attempt to use the information they receive throughout the execution of the protocol to gain information regarding the secret inputs of other parties. Furthermore, a subset of *corrupted* parties may form a coalition, joining the information they hold in an adversarial attempt to reveal the secret inputs of the other *honest* parties. The term *passive security* is used as short for security against honest-but-curious parties (or coalitions thereof).[1] Given an MPC protocol π, the maximal size t of an adversarial coalition of corrupted parties under which the privacy of the inputs of honest parties is maintained is the *threshold* of π. If $t < N/2$ (resp. $t \geq 2$) we say that π assumes *honest majority* (resp. *dishonest majority*).

In their seminal work from 1988, Ben-Or et al. [BOGW88] showed that, in the plain model, every function of N inputs can be efficiently computed with perfect passive security by N parties if and only if an honest majority is assumed.

One may circumvent this no-go result by switching to *the preprocessing model*, first suggested in [Bea97]. That model enables achieving perfect passive security against dishonest majority by enabling the parties to engage in an offline preprocessing phase before the secret inputs are known. At the end of that offline phase, the parties obtain *correlated randomness* (CR) - random coins to be used in the online phase of the protocol. That is, each party obtains a binary string of r bits such that the rN-string composed of the concatenation of the r-strings is a $\mathcal{D}$-distributed element of $\{0,1\}^{rN}$, where $\mathcal{D}$ is some predefined public distribution over $\{0,1\}^{rN}$. Preferably, $\mathcal{D}$ is independent of f. Given a P-MPC protocol, *the space complexity* of the scheme indicates how r grows with respect to other parameters (number of parties, size of the input, etc.). It was shown that MPC protocols in the preprocessing model (hereafter, we refer to such protocols as P-MPC protocols) can achieve goals which are known to be unachievable in the plain model. E.g., perfectly secure commitment schemes, perfectly secure oblivious transfer [Riv99], perfectly secure MPC with dishonest majority [IKM$^+$13].

Another measure of efficiency of MPC schemes is the *round complexity*. An MPC protocol is composed of *rounds of communication*. A round of communication (or round, in short) is a phase in which each party may do some or all of the following: (a) send at most one message to each of the other parties, (b) perform arbitrary computations, (c) receive at most one message from each of the other parties [KN06]. It is usually assumed that the parties are connected via authenticated point-to-point channels, and that f is given as an arithmetic or Boolean circuit. Primary solutions used rounds of communication to reduce the degree of the polynomial that encrypts the data after each multiplication during the computation. E.g., the methods of [BOGW88] are based on Shamir's

[1] The term *active security* refers to security against *malicious* parties. These parties might deviate from the protocol in an attempt to sabotage the computation process. Handling malicious parties is out of the scope of this work.

secret sharing scheme [Sha79], and the number of rounds of their protocols is proportional to the depth of the arithmetic circuit representing f.

Substantial efforts have been spent on finding the minimal number of rounds of communication required for perfectly secure MPC, both theoretically and practically. Bar-Ilan and Beaver [BIB89] were the first to suggest a way to evaluate functions in a constant number of rounds, followed by further works that attempt to lower bound that constant number. Theoretically, two rounds of communication are now known to be optimal for MPC — in the plain or preprocessing model [PR18, DLN19]. Recent works by [ABT18, GIS18, ACGJ18] present plain-model protocols which enable MPC in two rounds of communication and which have perfect passive security against honest majority. Ishai et al. suggested in [IKM+13] two-round P-MPC protocols with perfect passive security against dishonest majority, followed by several improvements suggested in [DNNR17, Cou19].

Though the problem of finding perfectly secure MPC schemes with optimal rounds and optimal threshold was resolved both in the plain and preprocessing models, all known solutions require using amounts of either time, memory or communication exponential in some of the parameters: depth or size of the circuit, size of the domain, number of parties. Particularly, in the plain model, the total communication complexity of two-round perfectly secure schemes is exponential in the depth of the arithmetic circuit. Achieving sub-exponential communication in the plain model requires increasing the number of rounds (or making other compromises, e.g., security-wise). In the preprocessing model, the space complexity of known solutions is (believed to be inherently[2]) exponential in the size of the input and N. Similarly, reducing the space complexity of these schemes (even for specific cases) costs in increasing round complexity (or other compromises).

In particular, consider the following scenario. N parties, holding non-zero inputs, want to jointly evaluate $f : \mathbb{F}_p^N \to \mathbb{F}_p$, where f is a polynomial of a possibly high degree and a small number of monomials, and $\mathbb{F}$ is a field of a potentially very large cardinality. The parties wish to carry that task while maintaining perfect security against dishonest majority and using two rounds of communication and function-independent correlated randomness. To the best of our knowledge, no known P-MPC scheme handles efficiently with this scenario. This scenario is hereafter referred to as *the target scenario.*

1.1 Our results

We construct the first N-party P-MPC schemes for polynomials over non-zero inputs which have communication and space complexities linear in the number of monomials, two-rounds of communication, perfect security against dishonest majority, and f-independent CR. Our *Distributed Random Matrix (DRM) two-round one-time secrets (OTS) scheme* presented here assumes that $f : \mathbb{F}_p^N \to \mathbb{F}_p$ is a polynomial with k monomials, and that all secret inputs are non-zero elements of $\mathbb{F}_p$. This scheme achieves all of the following properties.

- Perfect passive security against coalitions of up to $N - 1$ parties.
- Space complexity $O(kNn)$, where $n = \lceil \log p \rceil$ is the size of the input.
- Total communication complexity $O(kN^2n)$.
- Function-independent correlated randomness.
- Optimal round complexity. I.e., two rounds of communication.

[2] If every two-party functionality had a protocol with polynomial space complexity, this would imply an unexpected answer to a longstanding open question on the complexity of information-theoretic private information retrieval. See [IKM+13, CGKS95].

Our schemes are based on the *DRM secret sharing scheme*, our new homomorphic secret sharing scheme presented here. When we say that the CR required for our scheme is independent of f we mean that each unit of CR is f-independent. However, the amount of CR units required is linear in the number of monomials of f. In our schemes, these CR units can be generated and distributed by Ted and stored for future use by the parties in advance.

The client-server model. We extend our results to handle the following scenario. Assume $N \geq 2$ honest-but-curious servers and $m \geq 1$ users, with a fully connected network of servers and a connection channel between each user to each server. Let each of the users hold an arbitrary number of non-zero secret inputs in $\mathbb{F}_p$. The *DRM single-round client-server scheme* enables the users to securely outsource the storage of their private inputs to the servers and have the servers evaluate polynomials over the entire collection of users-inputs and obtain the result after a single round of communication between the servers. The DRM client-server scheme is perfectly secure against coalitions of up to $N - 1$ honest-but-curious servers. The users do not communicate with each other during the scheme, and each of them only distributes secret-shares of her inputs to the servers and receives the output from the servers. Hence, the users learn no information regarding the secret inputs of other users other than what may be deduced from the output. The servers may hold secret inputs as well.

Polynomials instead circuits and non-zero inputs. Our approach deviates from standard conventions of MPC. First, we evaluate each monomial of the polynomial independently, without converting it into an arithmetic circuit. On first sight, this choice may be unclear as arithmetic circuits have the benefit of enabling re-use of mid-values that were already computed. However, our approach enables handling high degree polynomials without being concerned with the depth of the arithmetic circuit, which is one of the main complexity bottlenecks in MPC. The communication and space complexities of our schemes are independent of the degree of f.

Second, we address the particular case of evaluating polynomials over non-zero inputs. A large part of the literature of MPC discusses the possibility of realizing *families of functions* (e.g., NC^1), but conditions over the inputs are hardly ever discussed. However, we inspect that non-zero inputs case and obtain a scheme which solves it (for polynomials) with remarkable performances — two rounds of communication, perfect passive security, dishonest majority, f-independent correlated randomness, $O(Nkn)$ space complexity, and $O(N^2kn)$ communication complexity.

Polynomial functions. The computations of polynomials (over a large field) using arithmetic operations are of course as natural as the computation of Boolean functions via logical gates, and capture many natural important tasks [Wig17]. These tasks include Fourier transforms, linear algebra, matrix computations, Reed-Muller encodings and more generally symbolic algebraic computations arising in many settings. E.g., the determinant of a square matrix of order N over a finite field is a polynomial of degree N, whose variables are the entries of that matrix. The entries of the product of two matrices are quadratic polynomials whose variables are the entries of these matrices. Similarly, the entries of the product of N square matrices are polynomials of degree N, whose variables are the entries of these matrices.

1.2 Related work

Ishai et al. [IKM+13] suggested two-round P-MPC schemes which are based on One-Time Truth Tables (OTTT) and have perfect passive security against dishonest majority. The communication complexity of their schemes grows linearly with the input size and the number of parties and is independent of the function. The space complexity of their schemes, however, is exponential in the number of parties and the size of the input, *regardless of* f, which makes their schemes impractical for large inputs or a large number of parties, even when considering simple functions. The CR used in their schemes is an additive secret sharing of the truth-table representation of the function, and hence dependent on it. They also suggested schemes with reduced space complexity, which assume a circuit representation of the function, but these schemes require rounds of communication proportional to the depth of the circuit. They also discuss the case of malicious parties and obtain statistically secure schemes for this case. In particular, when considering the target scenario, the schemes suggested in [IKM$^+$13] require either an amount of (function-dependent) CR exponential in the cardinality of $\mathbb{F}_p$, or a number of rounds proportional to the depth of an arithmetic circuit representation of the polynomial.

Damgård et al. [DZ13] presented MiniMac, a P-MPC protocol for *well-formed* Boolean circuits. That is, a circuit in which every layer is $\Omega(k)$ gates wide, and the number of bits that are output from layer i in the circuit and used in layer j is either 0 or $\Omega(k)$ for all $i < j$ (where k is the security parameter, and a constant number of exceptions are allowed). Their schemes have statistical active security against dishonest majority, negligible error probability, constant computational overhead, and communication complexity linear in the size of the circuit and the number of parties. Similar performances for general circuits are achieved by the TinyTable protocol, suggested in [DNNR17]. Recent work by Couteau [Cou19] presents P-MPC schemes for *layered Boolean circuits* — circuits whose nodes can be arranged into layers so that any edge connects adjacent layers. Their schemes have perfect passive security against dishonest majority, and their communication complexity is sublinear in the circuit size. However, the round complexity of their schemes is $O(d/\log\log s)$, where d and s are the depth and size, respectively, of the layered Boolean circuit. However, none of the works of [DZ13, DNNR17, Cou19] suggests a solution to the target scenario, as their protocols have round complexity proportional to the depth of the circuit, or have space complexity exponential in the cardinality of the field.

Ghodosi et al. discuss the problem of evaluating polynomials over non-zero inputs [GPS12]. They present a P-MPC protocol for this task. Their protocol is similar to our DRM-DBO scheme. However, their scheme is not optimal-round as it requires a secret sharing phase invoked offline. To the best of our knowledge, there is no known P-MPC protocol which enables evaluation of high-degree polynomials over non-zero inputs in two rounds of communication with perfect passive security against dishonest majority and with communication and space complexities that grow linearly with the number of monomials of f. The DRM two-round OTS scheme presented here is the first to solve the target scenario with all of these attributes.

1.3 Paper organization

The rest of the paper is organized as follows. Preliminaries appear in Section 2. In Section 3, we present the DRM secret sharing scheme and discuss its homomorphic properties. The DRM P-MPC schemes are presented in Section 4. Section 5 discusses the client-server model. Conclusions and appear in Section 6. Extensions of our schemes for handling general scenarios may be found in Appendix.

2 PRELIMINARIES

We recall some linear algebra and MPC notations and definitions and define several terms and concepts used throughout the paper. We use $\mathbb{F}_p$ to denote the finite field containing p elements (where p is prime), $\mathbb{F}_p^k$ to denote the k-dimensional vector space over $\mathbb{F}_p$, and $\mathbb{F}_p^\times$ to denote the multiplicative group of $\mathbb{F}_p$. $(\mathbb{F}_p^\times)^k$ is the set of k-tuples over $\mathbb{F}_p^\times$, with the operations '$+$' and '$\cdot$' for $\mathbb{F}_p$ entry-wise addition and scalar multiplication, respectively. The notation '$*$' stands for entrywise multiplication, and '$||$' for concatenation. We use $M_n(\mathbb{F}_p)$ to denote the set of square matrices of order n over $\mathbb{F}_p$. $\mathbb{N}$ is the set of natural numbers. For $n \in \mathbb{N}$, $[n] = \{1, \ldots, n\}$. If α is a k-tuple then $(\alpha)_i$ is the i'th entry of α. If C is a matrix then $[C]_i$ is the i'th column of C. We denote by $x \overset{R}{\leftarrow} A$ the process of assigning to the variable x a uniformly random element of the set A.

Security of MPC schemes. The security of an MPC protocol is formalized and proved through the *Ideal world vs. Real world* paradigm. We briefly overview the general idea. Let $\mathcal{P} = \{\mathcal{P}_j\}_{j=1}^{N}$ a set of N parties and assume that each party $\mathcal{P}_j$ is holding a secret value s_j in some domain $\mathcal{R}_j$. Assume that the parties wish to find $f(s_1, \ldots, s_n)$, where $f : \mathcal{R}_1 \times \cdots \times \mathcal{R}_N \to \mathcal{R}$, while not revealing to each other any information regarding their secret inputs, except for what may be deduced from $f(s_1, \ldots, s_N)$. In an ideal world, the parties could have found a trusted entity, *Ted*, to whom they will all tell their private inputs and from whom they will receive the output. Ted will perform the computation on their behalf and promise to keep their secrets safe.

In the real world, such a trusted entity is hard to find, and hence, the parties may attempt to perform the computation themselves by following an MPC protocol π. Informally, to consider π secure for computing f, it should have the property that by following it, the parties gain no information regarding the secret inputs of other parties that they could not have learned by following the ideal world solution. We also consider the case in which a subset $\mathcal{T} \subseteq \mathcal{P}$ of the parties join forces in an adversarial attempt to gain information regarding the secret inputs of parties in $\overline{\mathcal{T}} = \mathcal{P} - \mathcal{T}$. Informally, we would say that π is secure for computing f *with threshold* t if it holds that, for every $\mathcal{T} \subseteq \mathcal{P}$ with $|\mathcal{T}| \leq t$, the parties in $\mathcal{T}$ gain no more information regarding $\{s_i\}_{\mathcal{P}_i \in \overline{\mathcal{T}}}$ from π than they would have got from an ideal world solution. π leaks no more information than Ted if all the information obtained from π may be computed from the information received from Ted only. To formalize that, we define $view_j$ to be the random variable indicating s_j and all the messages that $\mathcal{P}_j$ receives through the execution of π, including the results of random choices that $\mathcal{P}_j$ makes. Recall that all parties are honest-but-curious, and hence, they follow the instructions of π. That leads us to

Definition 2.1. **Perfect correctness and perfect passive security of π.** Let f a function, $t \in \mathbb{N}$, and π an N-party protocol for computing f. We say that π realizes f with perfect correctness and perfect passive security with threshold t if (a) by executing π, all parties learn $y = f(s_1, \ldots, s_N)$, and (b) for every adversarial coalition of honest-but-curious parties $\mathcal{T} \subseteq \mathcal{P}$ with $|\mathcal{T}| \leq t$ there exists a simulator — a probabilistic algorithm $\mathtt{Sim}$ — which on inputs y and $\{s_j\}_{\mathcal{P}_j \subseteq \mathcal{T}}$, its output is identically distributed to $view_T = \{view_j\}_{\mathcal{P}_j \in \mathcal{T}}$.

Correlated Randomness (CR). In the preprocessing model, we assume that π may include an offline preprocessing phase in which the parties obtain correlated randomness. That is, each party $\mathcal{P}_j$ obtains a secret random binary string string $R_j \in \{0,1\}^r$ such that $R = R_1 || \ldots || R_N$ is a $\mathcal{D}$-distributed element of $\{0,1\}^{rN}$, where $\mathcal{D}$ is some predefined public distribution over $\{0,1\}^{rN}$ independent of the inputs. If $\mathcal{D}$ is also independent of f, we say that π uses f-independent CR.

We consider two ways of obtaining CR. The first involves a trusted initializer which provides the parties with the random strings R_j. We stress that there is a fundamental difference between the trusted entity Ted mentioned earlier in the ideal world solution to the trusted initializer considered now. While Ted receives the actual secret inputs from the parties and performs the computation in their behalf, the trusted initializer remains utterly oblivious to the secret inputs as the CR phase takes place before the secret inputs are known. Considering the presence of a trusted initializer seems quite natural because when we engage in digital communication, we very often use the services of a trusted server which provides authentication for the communicating parties. That server might as well provide the parties with CR. The second way of obtaining CR requires the parties running some offline protocol to generate and store correlated random strings.

3 DISTRIBUTED RANDOM MATRIX SECRET SHARING SCHEME

In this section, we describe the basic tool of this work, the *Distributed Random Matrix* procedure, DRM, and discuss some of its properties. DRM employs two other basic procedures — Mult.split (which will also be used by parties in our schemes to secret-share their private inputs) and Add.split. We now describe these schemes and discuss some of their properties.

Multiplicative secret sharing procedure. The following procedure is invoked by party $\mathcal{P}_i$ to split $s_i \in \mathbb{F}_p^\times$ into N multiplicative secret shares. Given a prime p, an element $s \in \mathbb{F}_p$, a natural number $N \in \mathbb{N}$, and $1 \le i \le N$, the procedure Mult.split is as follows. Pick $N-1$ uniformly random non-zero elements m_j $(1 \le j \le N, j \neq i)$ from $\mathbb{F}_p$ and set $m_i \in \mathbb{F}_p$ such that $s = \prod_{j=1}^{N} m_j$. Output $(m_1, \ldots, m_N)$, a sequence of *multiplicative shares* of s.

```
Procedure 1: Mult.split$(p, s, N, i)$
Input: a prime number $p$, $s \in \mathbb{F}_p$, $N \in \mathbb{N}$, and $1 \le i \le N$
for $1 \le j \le N, j \neq i$ do
    $m_j \xleftarrow{R} \mathbb{F}_p^\times$
end for
$\delta \leftarrow \prod_{j=1, j \neq i}^{N} m_j$
$m_i \leftarrow \frac{s}{\delta}$
return $(m_1, \ldots, m_N)$
```

Observe that, the assignment $m_i \leftarrow \frac{s}{\delta}$ in Mult.split implies that, if $s = 0$ then $m_i = 0$ and if $s \neq 0$ then $m_i \neq 0$. All other entries m_j of the output (with $j \neq i$) are uniformly random non-zero elements of $\mathbb{F}_p^\times$.

Lemma 3.1. *Procedure* Mult.split *is a perfectly secure secret sharing scheme for $\mathbb{F}_p^\times$ elements with a threshold of $N-1$ and which supports homomorphic multiplications.*

Proof. Assume that a user, holding a non-zero element $s \in \mathbb{F}_p^\times$, distributes the output of Mult.split(p, s, N, i) (for some i) to a set of parties $\{\mathcal{P}_j\}_{j=1}^{N}$. Any coalition of $N-1$ parties gains absolutely no information regarding s since given their shares, for every element $s' \in \mathbb{F}_p^\times$ there exists a single possible N'th share for which the product of the N shares equals s'. Now, assume that s_1 and s_2 are two non-zero elements of $\mathbb{F}_p$ that were secret-shared by a user using Mult.split. By construction, it immediately follows that

$$\prod_{j=1}^{N} \Big(\texttt{Mult.split}(p, s_1, N, i) * \texttt{Mult.split}(p, s_2, N, i') \Big)_j = s_1 \cdot s_2 .$$

The same holds for any number of shared secrets. If a user secret shares d elements $s_1, \ldots, s_d$ of $\mathbb{F}_p^\times$, then, having each of the parties compute the product of her shares (locally), each party obtains a multiplicative share of $\prod_{i=1}^{d} s_i$. □

Additive secret sharing procedure. Similarly to the previous procedure, given a prime p, an element $s \in \mathbb{F}_p$, and a natural number $N \in \mathbb{N}$, the procedure Add.split(p, s, N) is as follows. Pick $N-1$ uniformly random elements $a_1, \ldots, a_{N-1}$ from $\mathbb{F}_p$ and set $a_N = s - \sum_{i=1}^{N-1} a_i$. Output

$(a_1, \ldots, a_N)$, a sequence of *additive shares* of s. While `Mult.split` takes an input $i \in [N]$, `Add.split` takes no such input.

Lemma 3.2. *The procedure* `Add.split` *is a perfectly secure secret sharing scheme for* $\mathbb{F}_p$ *elements with threshold* $N-1$ *and which supports homomorphic additions.*

Proof. Assume that a user, holding an element $s \in \mathbb{F}_p$, distributes the a_j's to a set of parties $\{\mathcal{P}_j\}_{j=1}^N$. Any coalition of $N-1$ parties gains absolutely no information regarding s since given their shares, for every element $s' \in \mathbb{F}_p$ there exists a single possible N'th share for which the sum of the N shares equals s'. Now, assume that s_1 and s_2 are two elements of $\mathbb{F}_p$ that were `Add.split`-secret-shared. By construction it immediately follows that

$$\sum_{j=1}^{N} \big(\texttt{Mult.split}(p, s_1, N, i) + \texttt{Mult.split}(p, s_2, N, i')\big)_j = s_1 + s_2.$$

The same holds for any number of shared secrets. If a user secret shares d elements $s_1, \ldots, s_d$ of $\mathbb{F}_p$, then, having each of the parties locally compute the sum of the shares received, each party obtains an additive share of $\sum_{i=1}^d s_i$. □

Distributed Random Matrix (DRM) secret sharing procedure. We now define the procedure DRM. Given a prime p, an element $x \in \mathbb{F}_p$ and a natural number $N \in \mathbb{N}$, the procedure DRM outputs (the columns of) a matrix C, a *matrix-random-split of* x. C is generated by `Add.split`-secret sharing x, followed by `Mult.split`-secret sharing each of the additive shares. Formally, we have the following procedure.

```
Procedure 2: DRM$(p, s, N)$
Input: a prime number $p$, $s \in \mathbb{F}_p$, and $N \in \mathbb{N}$
$(\gamma_1, \ldots, \gamma_N) \leftarrow$ Add.split$(p, s, N)$
for $1 \le i \le N$ do
    $(c_{i1}, c_{i2}, \ldots, c_{iN}) \leftarrow$ Mult.split$(p, \gamma_i, N, i)$
end for
$C \leftarrow (c_{ij}) \in M_N(\mathbb{F}_p)$
return $\big([C]_1, \ldots, [C]_N\big)$
```

Observe that, Since the i'th row of C is an output of `Mult.split`(p, γ_i, N, i), the matrix C may contain zeroes only on its main diagonal, if any. That is, $c_{ij} = 0 \Longrightarrow i = j$.

Reconstruction of a DRM-secret-shared element x from N shares may be performed by multiplying all the elements in each row of C and summing the products. Namely,

$$\sum_{i=1}^{N} \prod_{j=1}^{N} c_{ij} = x.$$

To formalize that, we define the reconstruction procedure.

One may readily verify that, for a prime p, a natural number $N \in \mathbb{N}$ and an element $s \in \mathbb{F}_p$, it holds that `Reconstruct`$\big(p, \mathsf{DRM}(p, s, N), N\big) = s$.

We claim that the DRM secret sharing scheme supports an arbitrary number of homomorphic multiplications by `Mult.split`-secret-shared non-zero elements, and after a single round of communication, it supports an arbitrary number of homomorphic additions. To make this claim precise, we define the procedure M2A. The procedure is invoked by N parties, $\mathcal{P}_1, \ldots, \mathcal{P}_N$. Let $s \in \mathbb{F}_p$ and $\big([C]_1, \ldots, [C]_N\big) = \mathsf{DRM}(p, s, N)$. Assuming each party $\mathcal{P}_j$ is holding $[C]_j$, the following procedure enables the parties transforming from multiplicative shares of s to additive shares of it.

```
                              M2A
Communication round:
  For $1 \le i, j \le N$: $\mathcal{P}_j$ sends the $i$'t entry of $[C]_j$ to $\mathcal{P}_i$.
Output computation:
  For $1 \le i \le N$: $\mathcal{P}_i$ computes $\gamma_i = \prod_{j=1}^{N} c_{ij}$.
```

One may readily verify that, following M2A, each party obtains an additive share of s. Namely, $\sum_{i=1}^{N} \gamma_i = s$. We are now ready to state and prove the main theorem of this section.

Theorem 3.3 (DRM secret-sharing). *The procedure* DRM *is an N-party secret sharing scheme for $\mathbb{F}_p$ elements which has perfect passive security and threshold $t = N - 1$.* DRM *supports homomorphic multiplications by* Mult.split*-secret-shared non-zero elements. Have N parties hold* DRM*-shares of $s \in \mathbb{F}_p$, executing* M2A*, the parties obtain additive shares of s. These additive shares of s enable homomorphic additions with an arbitrary number of* Add.split*-secret-shared elements.*

Proof. To show that a secret sharing scheme is perfectly secure with threshold t we should show that for every coalition of t parties, the shares held by that coalition are independent of the secret. Let $x, x', x'' \in \mathbb{F}_p$, $x' \neq 0$. Assume that a user distributes the output $([C]_1, \ldots, [C]_N)$ of $\mathsf{DRM}(p, x, N)$ to a set $\mathcal{P} = \{\mathcal{P}_j\}_{j=1}^{N}$ of N parties, such that each $\mathcal{P}_j$ receives the DRM-share $[C]_j$ (we denote by c_{ij} the i'th entry of $[C]_j$). Let $h \in [N]$ and denote by $\mathcal{T}_h = \mathcal{P} - \{\mathcal{P}_h\}$ the adversarial coalition of size $N - 1$ which contains all parties except for $\mathcal{P}_h$, a single honest party. Our first goal is to show that the information held by $\mathcal{T}_h$ is independent of the x. Denote by $\Omega_{\mathcal{T}_h}$ the set of $(N-1)$-tuples of possible DRM-shares for parties in $\mathcal{T}_h$. $\Omega_{\mathcal{T}_h} \subseteq \big(\mathbb{F}_p^N\big)^{N-1}$ is the set of $(N-1)$-tuples with entries in $\mathbb{F}_p^N$ such that for every element $(v_1, \ldots, v_{h-1}, v_{h+1}, \ldots, v_N)$ of $\Omega_{\mathcal{T}_h}$ it holds that for every $j \in [N]$ $(j \neq h)$, the only entry of v_j that may be zero is its j'th entry. Formally,

$$\Omega_{\mathcal{T}_h} = \Big\{(v_1, \ldots, v_{h-1}, v_{h+1}, \ldots, v_N) \in \big(\mathbb{F}_p^N\big)^{N-1} \Big| (v_j)_i = 0 \Longrightarrow i = j\Big\}.$$

Now, to show that DRM is a perfectly secure secret sharing scheme with threshold $t = N - 1$, we need to show that the collection of $N - 1$ DRM-shares of x held by $\mathcal{T}_h$ is uniformly distributed over $\Omega_{\mathcal{T}_h}$ (independently of x). The key observation from which this follows is that given all entries of the matrix C except for any single entry c_{ii} on the main diagonal of C $(i \in [N])$, that last entry c_{ii} of C is uniquely determined by x. W.l.o.g., assume that $h = N$ (other cases may be treated similarly). The collection $([C]_1, \ldots, [C]_{N-1})$ of $N - 1$ DRM-shares of x held by $\mathcal{T}_N$ is uniformly distributed over $\Omega_{\mathcal{T}_N}$ independently of x since, given these shares, for every element $\beta \in \mathbb{F}_p$, there are $(p-1)^{N-1}$ different ways to choose a valid N'th DRM-share $[C]'_N$ of β. That is to say, $[C]'_N$ is an $\mathbb{F}_p^N$ column vector (whose only entry that may be zero is its N'th entry), and the order-N square matrix whose columns are the vectors $([C]_1, \ldots, [C]_{N-1}, [C]'_N)$ is a matrix-random-split of β. Indeed, uniformly randomly choose $N - 1$ non-zero elements $c'_{1,N}, \ldots, c'_{N-1,N}$ of $\mathbb{F}_p$ and set

$$c'_{NN} = \frac{\beta - \sum_{i=1}^{N-1}\Big(c'_{iN} \cdot \prod_{j=1}^{N-1} c_{ij}\Big)}{\prod_{j=1}^{N-1} c_{jN}}. \tag{1}$$

One may readily verify that the column vector $[C]'_N = (c'_{1N}, \dots, c'_{NN})$ is a valid N'th DRM-share of β. Hence, we conclude that the shares held by $\mathcal{T}_N$ are uniformly distributed over $\Omega_{\mathcal{T}_N}$ independently of x. This shows that DRM is a perfectly secure secret sharing scheme with threshold $t = N-1$.

Now, assume that the user distributes the Mult.split-shares $(m_1, \dots, m_N)$ of x' to the parties such that each party $\mathcal{P}_j$ obtains m_j. To show that DRM supports perfectly secure homomorphic multiplications by Mult.split-secret-shared non-zero elements, we need to show that:

- *Privacy.* The collection of $N-1$ DRM-secret-shares of x held by $\mathcal{T}_h$, and the $N-1$ Mult.split-secret-shares of x' held by $\mathcal{T}_h$, is uniformly distributed over $\Omega_{\mathcal{T}_h} \times (\mathbb{F}_p^{\times})^{N-1}$ (independently of x and x').
- *Correctness.* For $j \in [N]$, have $\mathcal{P}_j$ locally compute the product $m_j[C]_j$. Doing so, each party obtains a DRM-share of $x \cdot x'$. In other words, the entrywise product of the N-tuple of DRM-shares of x and the N-tuple of Mult.split-shares of x' is an N-tuple of DRM-shares of the product $x \cdot x'$.

The privacy immediately follows from Lemma 3.1 and the first part of this proof. In the first part of this proof we have shown that the DRM-shares of x are uniformly distributed over $\Omega_{\mathcal{T}_h}$. By Lemma 3.1, the Mult.split-shares of x' are uniformly distributed over $(\mathbb{F}_p^{\times})^{N-1}$. Hence, the collection of shares of x and x' is uniformly distributed over $\Omega_{\mathcal{T}_h} \times (\mathbb{F}_p^{\times})^{N-1}$. For correctness, observe that

$$\begin{aligned} \texttt{Reconstruct}\Big(p, \big(m_1[C]_1, \dots, m_N[C]_N\big), p\Big) &= \sum_{i=1}^{N} \prod_{j=1}^{N} \Big(m_j[C]_j\Big)_i \\ &= \sum_{i=1}^{N} \Bigg(\Big(\prod_{j=1}^{N} m_j\Big) \cdot \Big(\prod_{j=1}^{N} c_{ij}\Big)\Bigg) = x' \cdot \sum_{i=1}^{N} \prod_{j=1}^{N} c_{ij} = x \cdot x'. \end{aligned} \tag{2}$$

Similarly, the user may perform d such homomorphic multiplications of x with d Mult.split-shared non-zero elements $x'_1, \dots, x'_d$ (for an arbitrary d) and have the parties obtain a DRM-shares of the product $x \cdot \prod_{i=1}^{d} x'_i$. We also note that, even if x is made public, homomorphically multiplying the DRM-shared element x with the secret Mult.split-shared element x' leaks no information regarding x'.

Regarding the transformation from supporting homomorphic multiplications to supporting homomorphic additions, assumes that the parties execute M2A (using the DRM-shares of x) to obtain $(\gamma_1, \dots, \gamma_N) \in \mathbb{F}_p^N$ and receive Add.split-shares $(a_1, \dots, a_N)$ of x'' from the user. Our goal now is to show that the execution of M2A keeps x perfectly secure and enables perfectly secure homomorphic additions with x''. Explicitly, we need to show:

- *Correctness.* The sum of $(\gamma_1, \dots, \gamma_N)$ and $(a_1, \dots, a_N)$ is an N-tuple of additive shares of $x + x''$.
- *Privacy.* The collection of $N-1$ DRM-shares of x and the information received by $\mathcal{T}_h$ in the execution of M2A is uniformly distributed over $\Omega_{\mathcal{T}_h} \times (\mathbb{F}_p^{\times})^{N-1}$ (independently of x).

For correctness, observe that

$$\sum_{i=1}^{N} \big((\gamma_1, \dots, \gamma_N) + (a_1, \dots, a_N)\big)_i = \sum_{i=1}^{N} \gamma_i + \sum_{i=1}^{N} a_i = \sum_{i=1}^{N} \prod_{j=1}^{N} c_{ij} + x'' = x + x''.$$

For privacy, observe that all the new information received by $\mathcal{T}_h$ in the execution of M2A is the $N-1$ elements of $[C]_h$ which are not on the main diagonal of C. W.l.o.g. let $h = N$ (other cases follow similarly). Similarly to (1), given the new elements $c_{1N}, \dots, c_{N-1,N}$, for every $\beta \in \mathbb{F}_p$, there exists a single element c'_{NN} for which the matrix C', obtained from C by replacing the element c_{NN} with c'_{NN}, is a matrix-random-split of β. Namely,

$$c'_{NN} = \frac{\beta - \sum_{i=1}^{N-1} \prod_{j=1}^{N} c_{ij}}{\prod_{j=1}^{N-1} c_{jN}}.$$

We conclude that the collection of $N-1$ DRM-shares of x and the information received by $\mathcal{T}_h$ in the execution of M2A is uniformly distributed over $\Omega_{\mathcal{T}_h} \times (\mathbb{F}_p^{\times})^{N-1}$ (independently of x).

□

4 THE DRM P-MPC SCHEMES

In this section, we present perfectly secure P-MPC schemes for polynomials over non-zero inputs, based on the DRM secret sharing scheme. We assume that $\mathcal{P} = \{\mathcal{P}_j\}_{j=1}^{N}$ is a set of $N \geq 2$ honest-but-curious parties which are connected via point-to-point authenticated secure channels. For ease of presentation, we assume that each party $\mathcal{P}_j$ holds a single input $s_j \in \mathbb{F}_p$. In general, each party may hold an arbitrary number of secrets. Let $s = (s_1, \ldots, s_N)$. The function to be evaluated is $f : \mathbb{F}_p^N \to \mathbb{F}_p$, where:

$$f(x_1, \ldots, x_N) = \sum_{l=(l_1,\ldots,l_N)\in\mathcal{L}} a_l \cdot x_1^{l_1} \ldots x_N^{l_N},$$

where $\mathcal{L} = \{0, \ldots, p-1\}^N$ and $a_l \in \mathbb{F}_p$. For $l \in \mathcal{L}$, let $A_l = x_1^{l_1} \ldots x_N^{l_N}$. The l'th monomial of f is $a_l A_l$. The size of the polynomial (i.e., the number of monomials with $a_l \neq 0$) is k, and the size of the input is $n = \lceil \log p \rceil$.

We begin with the *database-oriented* (DBO) version of the DRM P-MPC scheme, and then present the *one-time secrets* (OTS) version. The DBO version includes a secret sharing round as a part of the preprocessing phase. This round creates a virtual database shared among the parties and enables any polynomial to be evaluated over that database. The OTS version of the scheme includes no secret sharing stage and hence solves the target scenario with optimal round complexity.

4.1 The DBO version

The general idea behind the scheme is as follows. At the preprocessing phase, the parties obtain k units of correlated randomness (CR). Each unit of CR is a matrix-random-split of $1 \in \mathbb{F}_p$, split between the parties such that each party obtains a single column of the matrix. These matrices are independent of f, the secrets, and one another. At the first round of the scheme, each party $\mathcal{P}_j$ shares s_j between all parties using the Mult.split secret sharing scheme. Next, f is evaluated using the homomorphic properties of the DRM secret sharing scheme. We assume that the parties agreed on an ordering of the monomials of f, and evaluate each monomial by performing multiplications of the corresponding column with the appropriate powers of shares of the secrets. In the second round of the scheme, a simultaneous M2A stage is performed, at the end of which, each party obtains an additive share of the output. In the third round of the scheme, each party reveals her share of the output. The sum of these shares equals $f(s_1, \ldots, s_N)$.

Theorem 4.1 (DBO). *The DRM three-round DBO scheme is a three-round N-party P-MPC scheme for polynomials over non-zero inputs which has perfect correctness, perfect passive security, threshold $N-1$, communication complexity $O(N^2nk)$, space complexity $O(Nnk)$, and f-independent CR.*

The DRM three-round DBO scheme

Preprocessing phase.

Correlated randomness. For each non-zero monomial $a_l A_l$ of f, each party $\mathcal{P}_j$ obtains a DRM-share $[C^{(l)}]_j$ of $1 \in \mathbb{F}_p$. Each $C^{(l)}$ is a matrix-random-split of $1 \in \mathbb{F}_p$.

Secret sharing. Each party $\mathcal{P}_i$ secret-shares s_i using Mult.split. The shares $s_{i1}, \dots, s_{iN}$ of s_i are distributed such that $\mathcal{P}_j$ receives s_{ij}.

Online phase.

Eval. 1. For each monomial $a_l A_l$ of f, each party $\mathcal{P}_j$ computes:

$$\alpha_j^{(l)} = \prod_{i=1}^{N} s_{ij}^{l_i} \cdot [C^{(l)}]_j.$$

Com. 1. For $i, j \in [N]$, $\mathcal{P}_j$ sends the i'th entry of each $\alpha_j^{(l)}$ to $\mathcal{P}_i$.

Eval. 2. For each monomial $a_l A_l$ of f, each party $\mathcal{P}_i$ computes:

$$U_i^{(l)} = a_l \prod_{j=1}^{N} \left(\alpha_j^{(l)}\right)_i.$$

Com. 2. Each party $\mathcal{P}_i$ sends $y_i = \sum_l U_i^{(l)}$ to all other parties.

Output reconstruction. Each party computes $\sum_{i=1}^{N} y_i$.

Proof. *Security.* Let $h \in [N]$ and denote by $\mathcal{T}_h = \mathcal{P} - \{\mathcal{P}_h\}$ the adversarial coalition of size $N-1$ which contains all parties except for $\mathcal{P}_h$. W.l.o.g, we assume that $h = N$. We construct Sim as follows. Given $\{s_j\}_{j \in [N-1]}$ and $y = f(s_1, \dots, s_N)$, the simulator chooses a uniformly random element from the set of possible values of s_N. Formally, let

$$E = \left\{a \middle| f(s_1, \dots, s_{N-1}, a) = y\right\} \subseteq \mathbb{F}_p,$$

and pick a uniformly randomly element s'_N of E. Then, Sim simulates the actions of all N parties according to the instructions of DBO for N parties with secrets $s_1, \dots, s_{N-1}, s'_N$, and outputs the simulated view of the first $N-1$ parties. By Lemma 3.1, for every secret, any subset of $N-1$ shares is uniformly distributed over $\mathbb{F}_p^{N-1}$, and hence, so is the part of the view of $\mathcal{T}_h$ that is viewed at the secret sharing stage. In *Com. 1*, each message received by $\mathcal{T}_h$ is some non-zero element $\prod_{m=1}^{N} s_{m,j}^{l_m}$, multiplied by some c_{ij}. Since, those c_{ij}'s are uniformly random non-zero elements, and multiplication by non-zero elements in $\mathbb{F}_p$ is a bijection, the part of the view of $\mathcal{T}_h$ that is viewed at *Com. 1* is uniformly distributed. The last message obtained by $\mathcal{T}_h$ from $\mathcal{P}_N$ at *Com. 2* is a function of the input y and the additive shares held by $\mathcal{T}_h$. We conclude that, given $s_1, \dots, s_{N-1}$ and y, the view of $\mathcal{T}_h$ is uniformly distributed over the domain of possible views. Now, since the parties are honest-but-curious, they follow the instructions of DBO, and hence, the simulated view output by Sim is identically distributed to $view_{\mathcal{T}_h}$.

Correctness. The correctness of the scheme follows from the fact that

$$\sum_{i=1}^{N} y_i = \sum_{l} \sum_{i=1}^{N} U_i^{(l)} = \sum_{l} \sum_{i=1}^{N} a_l \prod_{j=1}^{N} (\alpha_j)_i = \sum_{l} a_l \sum_{i=1}^{N} \prod_{j=1}^{N} \left(s_{1j}^{l_1} \dots s_{Nj}^{l_N} \cdot [C^{(l)}]_j\right)_i$$

$$= \sum_{l} a_l \sum_{i=1}^{N} \left(s_1^{l_1} \dots s_N^{l_N} \cdot \prod_{j=1}^{N} c_{ij}^{(l)}\right) = \sum_{l} a_l \cdot s_1^{l_1} \dots s_N^{l_N} \cdot \sum_{i=1}^{N} \gamma_i^{(l)},$$

where $\gamma_i^{(l)}$ denotes the product $c_{i1}^{(l)} \dots c_{in}^{(l)}$. Since each $C^{(l)}$ is a matrix-random-split of 1, we have $\sum_{i=1}^{N} \gamma_i^{(l)} = 1$ and hence, $\sum_{i=1}^{N} y_i = f(s_1, \dots, s_N)$.

Communication complexity. By construction, the scheme requires three rounds. The total number of bits communicated in the online phase, is $O(N^2kn)$. Indeed, in *Com. 1*, each of the N parties sends N messages, where each message is a k-tuple, and each entry of that k-tuple is an $\mathbb{F}_p$ element. That is a total of N^2kn bits. In *Com. 2*, each party sends a single $\mathbb{F}_p$ element to all other parties, i.e., a total of Nn bits. All in all, $N^2kn + Nn$ bits are communicated.

Space complexity. How many bits of CR are required? At the preprocessing phase, for each monomial of f, each party obtains a single column of an order-N $\mathbb{F}_p$-valued square matrix. That is a total of kNn bits obtained by each party. Hence, the space complexity of the scheme is $O(kNn)$. □

We note that DBO assumes that each party holds a single input. This assumption is not obligatory and was made only for the ease of presentation. In general, each party may have an arbitrary number of secrets. In that case, at the secret sharing stage, each party should secret-share each secret she holds independently. The rest follows trivially.

4.2 The OTS version

Our novel OTS version of the DRM two-round MPC scheme includes no secret sharing stage and solves the target scenario with optimal round complexity. It does include a preprocessing stage at which f-independent CR is obtained by the parties. The scheme is presented below.

The DRM two-round OTS scheme

Preprocessing phase.

Correlated randomness. For each non-zero monomial a_lA_l of f, each party $\mathcal{P}_j$ obtains a DRM-share $[C^{(l)}]_j$ of $1 \in \mathbb{F}_p$. Each $C^{(l)}$ is a matrix-random-split of $1 \in \mathbb{F}_p$.

Online phase.

Eval. 1. For each monomial of f, each party $\mathcal{P}_j$ computes $\alpha_j^{(l)} = s_j^{l_j}[C^{(l)}]_j$.

Com. 1. For $i, j \in [N]$, $\mathcal{P}_j$ sends the i'th entry of each $\alpha_j^{(l)}$ to $\mathcal{P}_i$.

Eval. 2. For each monomial a_lA_l of f, each party $\mathcal{P}_i$ computes:

$$U_i^{(l)} = a_l \prod_{j=1}^{N} \left(\alpha_j^{(l)}\right)_i.$$

Com. 2. Each party $\mathcal{P}_i$ sends $y_i = \sum_l U_i^{(l)}$ to all other parties.

Output reconstruction. Each party computes $\sum_{i=1}^{N} y_i$.

Theorem 4.2 (OTS). *The DRM two-round OTS scheme is a two-round N-party P-MPC scheme for polynomials over non-zero inputs which has perfect correctness, perfect passive security, threshold $N-1$, communication complexity $O(N^2nk)$, space complexity $O(Nnk)$, and f-independent CR.*

Proof. The correctness and security of OTS follows from similar arguments to those of DBO. For correctness, observe that,

$$\sum_{i=1}^{N} \sum_{l} U_i^{(l)} = \sum_{l} \sum_{i=1}^{N} U_i^{(l)} = \sum_{l} \sum_{i=1}^{N} a_l \prod_{j=1}^{N} (\alpha_j)_i = \sum_{l} a_l \sum_{i=1}^{N} \prod_{j=1}^{N} \left(s_j^{l_j}[C^{(l)}]_j\right)_i$$

$$= \sum_{l} a_l \sum_{i=1}^{N} \left(s_1^{l_1} \ldots s_N^{l_N} \cdot \prod_{j=1}^{N} c_{ij}^{(l)} \right) = \sum_{l} a_l \cdot s_1^{l_1} \ldots s_N^{l_N} \cdot \sum_{i=1}^{N} \gamma_i^{(l)} = f(s_1, \ldots, s_N).$$

This scheme has the same security properties as DBO, i.e., perfect passive security and threshold $N-1$. This claim may be proved by arguments similar to those used at the security proof of DBO. The key observation is that for every l and j, if $i \neq j$, then the i'th entry of $[C^{(l)}]_j$ is a uniformly random non-zero element of $\mathbb{F}_p$, and hence, so is $\alpha_j^{(l)}$. By construction, the communication and space complexities of OTS are the same as those of the DBO. □

5 THE CLIENT-SERVER MODEL

In the last years, there is ongoing growth in the popularity of cloud services. More and more companies offer on-demand storage devices and computing engines. While users of cloud services enjoy the benefits of fast and cheap data processing, they are required to send their information to a possibly untrusted cloud. A problem arises when that information is private. One possible solution to that problem may be found at the scope of Fully Homomorphic Encryption (FHE). Unfortunately, all known FHE schemes are only computationally secure and are currently time-wise inefficient. A different approach comes from the scope of MPC — have the user secret-share her data among several servers, and have the servers engage in an MPC scheme to jointly process computations over that data.

The DRM P-MPC schemes suggested above may be used by a set of $m \geq 1$ users to securely outsource the storage of their private information to a set of $N \geq 2$ honest-but-curious servers, and have the servers perform computations over that information in a single round of communication.

We now present the *DRM one-round client-server scheme*. For ease of presentation, we assume $m = 1$. Assume that a user has a private connection channel with N honest-but-curious servers, denoted $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_N$. The scheme we suggest now enables a user to secret-share M non-zero elements, $s_1, \ldots, s_M \in \mathbb{F}_p$, amongst the servers in a way that allows the user to evaluate $f(s_1, \ldots, s_M)$ using computing engines provided by the servers, where $f : \mathbb{F}_p^M \to \mathbb{F}_p$ is a polynomial with monomials $a_l A_l$ as above. The general idea is as follows. The user uses `Mult.split` to secret-share her secret inputs among the servers, and sends the servers a sufficient amount of CR units for future use. Whenever the user wishes to evaluate a polynomial f over the secrets, she sends a query to the servers with f. Then, the servers homomorphically evaluate f over their shares in a single round of communication and send the additive shares of the result to the user.

Correctness and perfect passive security of the client-server scheme against coalitions of up to $N-1$ honest-but-curious servers follow from Theorem (3.3).

More users. If $m > 1$, then let M denote the total number of secrets held by all the users. Then, to enable the set of m users outsource their privet information to the servers and have the servers process computations over the set of all the secrets, we modify the client-server scheme as follows.

- At *Secret sharing*, each user secret-shares her inputs (using `Mult.split`).
- At *Query*, the users divide the work of generating the required CR between them, such that each user generates $\approx k/m$ matrix-random-splits of 1 and distributes them to the servers.
- At the last two stages of the scheme, the servers send the additive shares of the result to each of the users, which in turn compute the output locally.

In fact, the servers may have secrets of their own as well, and f may be a function of those secrets too. In that case, the users learn no more information regarding the secret inputs of the servers other than what may be learned from the output.

The DRM one-round client-server scheme

Secret sharing. For $i \in [M]$, the user secret-shares each s_i using `Mult.split`. The shares $s_{i1}, \ldots, s_{iN}$ of s_i are distributed among the servers such that $\mathcal{P}_j$ receives s_{ij}.

Query. The users sends f to the servers. For each monomial $a_l A_l$ of f, the user distributes `DRM`-shares of $1 \in \mathbb{F}_p$. Each server $\mathcal{P}_j$ obtains $[C^{(l)}]_j$, where $C^{(l)}$ is a matrix-random-split of $1 \in \mathbb{F}_p$. (This stage may be done in advance before f is known.)

Eval. 1. For each monomial $a_l A_l$ of f, each server $\mathcal{P}_j$ computes:

$$\alpha_j^{(l)} = \prod_{i=1}^{M} s_{ij}^{l_i} \cdot [C^{(l)}]_j.$$

Communication. For $i, j \in [N]$, $\mathcal{P}_j$ sends the i'th entry of each $\alpha_j^{(l)}$ to $\mathcal{P}_i$.

Eval. 2. For each monomial $a_l A_l$ of f, $\mathcal{P}_i$ computes:

$$U_i^{(l)} = a_l \prod_{j=1}^{N} \left(\alpha_j^{(l)}\right)_i.$$

Retrieving additive shares. Each server $\mathcal{P}_i$ sends $y_i = \sum_l U_i^{(l)}$ to the user.

Output reconstruction. The user computes $\sum_{i=1}^{N} y_i$.

6 CONCLUSIONS

In this paper, we have suggested schemes for perfectly secure multiparty computation with optimal round complexity of polynomials over non-zero inputs, both in the preprocessing model and the client-server model. We began with the construction of an N-party perfectly secure secret sharing scheme which supports multiplications with non-zero elements with threshold $N - 1$ against honest-but-curious adversary. We showed how the parties may efficiently generate additive shares of the secret from the multiplicative shares of it in a single round of communication, thus enabling homomorphic additions with further secrets. This secret sharing scheme was then used to construct a perfectly secure two-round P-MPC scheme for polynomials over non-zero inputs in $\mathbb{F}_p$. We have extended our scheme to the client-server model.

We note that our schemes, based on evaluating each monomial independently, induce considerable computational overhead compared to schemes that evaluate the polynomial using an arithmetic circuit representation. Circuit representations enable re-use of mid-values that were already computed. How big is that overhead? In order to (asymptotically) compare the computational complexity of our schemes to that of standard schemes, one should write the number of monomials k of f in terms of the size s and depth d of the arithmetic circuit. Finding the relation between the number of monomials of a general function and the size and depth of a circuit which computes the same function has roots in the algebraic analog of the $P \overset{?}{=} NP$ problem (suggested by Valiant in [Val79]) and is beyond the scope of this paper.

However, the round complexity of our OTS scheme is optimal, i.e., two-rounds. To emphasize the importance of round-efficiency, we note that, while processing information becomes faster as technology improves, the time that it takes to transmit information between two distant places is strongly limited by the speed of light. One may consider a future need to perform MPC over inputs held by parties which reside in distant places, perhaps outside of earth. If T is the time it takes to process the computations needed for evaluation of f using our schemes, then if the distance between parties is such that sending messages between parties takes more time than T, optimal-round schemes outperform any scheme with non-optimal round complexity.

The case of malicious parties (that may deviate from the protocol) was not discussed in this paper. As a future direction, it would be interesting to investigate ways to equip our schemes with mechanisms that will guarantee (to some extent) security and correctness in the face of active adversaries.

Lastly, we believe that our new approach and techniques may be used to securely outsource computations in a reduced cost of communication, and may be found to have further uses in many other scopes.

7 ACKNOWLEDGMENTS.

We thank Dani Berend for being involved during the entire research providing original ideas throughout. We thank the participants of the Crypto seminar, held at Tel Aviv University at July 18th 2019 — Iftach Haitner, Ran Canetti, and students and researchers from the computer science department — for useful comments.

REFERENCES

[ABT18] Benny Applebaum, Zvika Brakerski, and Rotem Tsabary. Perfect secure computation in two rounds. In *Theory of Cryptography Conference*, pages 152–174. Springer, 2018.

[ACGJ18] Prabhanjan Ananth, Arka Rai Choudhuri, Aarushi Goel, and Abhishek Jain. Round-optimal secure multiparty computation with honest majority. In *Annual International Cryptology Conference*, pages 395–424. Springer, 2018.

[Bea97] Donald Beaver. Commodity-based cryptography. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 446–455. ACM, 1997.

[BIB89] Judit Bar-Ilan and Donald Beaver. Non-cryptographic fault-tolerant computing in constant number of rounds of interaction. In *Proceedings of the eighth annual ACM Symposium on Principles of distributed computing*, pages 201–209. ACM, 1989.

[BMR90] Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols. In *Proceedings of the twenty-second annual ACM Symposium on Theory of Computing*, pages 503–513. ACM, 1990.

[BOGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 1–10. ACM, 1988.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgard. Multiparty unconditionally secure protocols. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 11–19. ACM, 1988.

[CGKS95] Benny Chor, Oded Goldreich, Eyal Kushilevitz, and Madhu Sudan. Private information retrieval. In *Proceedings of IEEE 36th Annual Foundations of Computer Science*, pages 41–50. IEEE, 1995.

[Cou19] Geoffroy Couteau. A note on the communication complexity of multiparty computation in the correlated randomness model. In *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, 2019, Proceedings, Part II*, pages 473–503, 2019.

[DLN19] Ivan Damgård, Kasper Green Larsen, and Jesper Buus Nielsen. Communication lower bounds for statistically secure mpc, with or without preprocessing. *IACR Cryptology ePrint Archive*, 2019:220, 2019.

[DN03] Ivan Damgård and Jesper Buus Nielsen. Universally composable efficient multiparty computation from threshold homomorphic encryption. In *Annual International Cryptology Conference*, pages 247–264. Springer, 2003.

[DNNR17] Ivan Damgård, Jesper Buus Nielsen, Michael Nielsen, and Samuel Ranellucci. The tinytable protocol for 2-party secure computation, or: Gate-scrambling revisited. In *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, 2017, Proceedings, Part I*, pages 167–187, 2017.

[DZ13] Ivan Damgård and Sarah Zakarias. Constant-overhead secure computation of boolean circuits using preprocessing. In *Proceedings of Theory of Cryptography 2013 - The 10th Theory of Cryptography Conference TCC.*, pages 621–641, 2013.

[GIS18] Sanjam Garg, Yuval Ishai, and Akshayaram Srinivasan. Two-round mpc: information-theoretic and black-box. In *Theory of Cryptography Conference*, pages 123–151. Springer, 2018.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 218–229. ACM, 1987.

[GPS12] Hossein Ghodosi, Josef Pieprzyk, and Ron Steinfeld. Multi-party computation with conversion of secret sharing. *Designs, Codes and Cryptography*, 62(3):259–272, 2012.

[IKM+13] Yuval Ishai, Eyal Kushilevitz, Sigurd Meldgaard, Claudio Orlandi, and Anat Paskin-Cherniavsky. On the power of correlated randomness in secure computation. In *Theory of Cryptography Conference*, pages 600–620. Springer, 2013.

[KN06] Eyal Kushilevitz and Noam Nisan. *Communication Complexity.* Cambridge University Press, United Kingdom, 2006.

[PR18] Arpita Patra and Divya Ravi. On the exact round complexity of secure three-party computation. In *Annual International Cryptology Conference*, pages 425–458. Springer, 2018.

[Riv99] Ronald Rivest. Unconditionally secure commitment and oblivious transfer schemes using private channels and a trusted initializer. *Unpublished manuscript*, 1999.

[Sha79] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[Val79] Leslie G Valiant. Completeness classes in algebra. In *Proceedings of the eleventh annual ACM symposium on Theory of computing*, pages 249–261. ACM, 1979.

[Wig17] Avi Wigderson. Technical perspective: Low-depth arithmetic circuits. *Communications of the ACM*, 60(6):91–91, 2017.

[Yao82] Andrew Chi-Chih Yao. Protocols for secure computations. In *FOCS*, volume 82, pages 160–164, 1982.

A NON-ZERO INPUTS AND GENERAL FUNCTIONS

We now show some simple ways in which our schemes may be adjusted to support evaluation of arbitrary functions over possibly-zero inputs. We stress that the performances of the resulting schemes is often inferior to those of existing solutions. We present these ways mainly to show that, theoretically, our approach is capable of supporting general scenarios. [3]

First, we note that any function $f : \mathbb{F}_p^N \longrightarrow \mathbb{F}_p$ can be represented as a multivariate polynomial. This representation may be obtained, for example, by solving a system of linear equations, or using other polynomial interpolation methods. Due to Fermat's little theorem, which states that $x^p \equiv x \pmod p$, f can be written as a polynomial of degree at most $N(p-1)$ and with at most p^N monomials. Namely, one may write

$$f(x_1,\dots,x_N) = \sum_{l=(l_1,\dots,l_N)\in\mathcal{L}} a_l \cdot x_1^{l_1}\dots x_N^{l_N}, \tag{3}$$

where $\mathcal{L} = \{0,\dots,p-1\}^N$, for appropriate elements $a_l \in \mathbb{F}_p$.

Next, if f is a Boolean function, our scheme may be used to support MPC of f by working in $\mathbb{F}_2$. A *True* Boolean value is $1 \in \mathbb{F}_2$ and a *False* Boolean value is $0 \in \mathbb{F}_2$. Boolean operations may be identified with field operations in the following way. The $\wedge$ operation is identified with $\mathbb{F}_2$ multiplication, the $\oplus$ operation with $\mathbb{F}_2$ addition, and the $\neg$ operation with adding 1 in $\mathbb{F}_2$. The $\vee$ operation of two literals is identified with $x + y + xy$, where x and y are the elements of $\mathbb{F}_2$ corresponding to the literals. Then, given a Boolean formula φ over Boolean literals $b_1,\dots,b_M \in \{True, False\}$, one can take the $\mathbb{F}_2$ correspondents $s_1,\dots,s_M \in \mathbb{F}_2$ of $b_1,\dots,b_M$, and evaluate the Boolean formula $\varphi : \{True, False\}^M \longrightarrow \{True, False\}$ using the polynomial $\tilde{\varphi} : \mathbb{F}_2^M \longrightarrow \mathbb{F}_2$, obtained by replacing Boolean operations and literals with their $\mathbb{F}_2$ correspondents.

Now, the communication and space complexities of our schemes are analyzed with respect to the number k of monomials in the polynomial representation of the function. Most of the known MPC schemes assume that f is given as an arithmetic circuit, and their communication and space complexities are analyzed with respect to the size s and depth d of the circuit. In order to (asymptotically) compare the performances of our schemes to those of a different scheme which assumes f is given as a circuit with size s and depth d, one must write k in terms of s and d. For an arbitrary

[3] Note that [GPS12] suggests a specific example for a specific technique to cope with possible zeros, which is a special case for our techniques.

f, that task might be hard. In general, consider the following question.

What is the relation between the size and depth of a circuit which computes an arbitrary function f and the size of a polynomial which computes f?

This question is an open problem, rooted in the algebraic analog of the $P \overset{?}{=} NP$ problem, suggested by Valiant in [Val79], and is out of the scope of this paper.

So far, we have shown how MPC of Boolean or arithmetic functions reduces to MPC of polynomials over finite fields. However, our schemes assumed that the inputs are non-zero elements of the field. Next, we suggest several ways of handling possibly-zero inputs.

First, we suggest the *q-bounded* approach. Let $s = (s_1, \ldots, s_N) \in \mathbb{F}_p^N$. One can compute $f(s)$ by performing operations in $\mathbb{F}_p$ according to the representation of f as a multivariate polynomial. The same result is obtained if one computes $f(s)$ over the positive integers and then takes the result modulo p. Formally, for each entry s_j of s let a_j denote the minimal positive integer such that $a_j \equiv s_j \pmod p$. Then, performing the computation over the a_j's using integer operations one obtains an integer result $f(s)_{\mathbb{N}}$, such that $f(s)_{\mathbb{N}} \equiv f(s) \pmod p$. If q is a prime number such that for every $s \in \mathbb{F}_p^N$, computation of $f(s)$ over the integers yields an integer result, $f(s)_{\mathbb{N}}$, which is smaller than q, then f is *q-bounded.* Since there are at most p^N monomials in f, for every $s \in \mathbb{F}_p^N$ it holds that $f(s)_{\mathbb{N}} < p^N \cdot (p-1) \cdot (p^{p-1})^N = p^N \cdot (p-1) \cdot p^{Np-N} = (p-1) \cdot p^{Np}$. Hence, f is q bounded for a prime q larger than $(p-1) \cdot p^{Np}$. In practice, one may find a smaller prime q' for which f is q'-bounded.

Now, we can use this fact to evaluate $\mathbb{F}_p$-polynomials over possibly-zero inputs by working in $\mathbb{F}_q$ for large enough q. To this end we present *the DRM three-round DBO-qB scheme.* DBO-qB supports evaluating polynomials over possibly-zero inputs by embedding $\mathbb{F}_p$ in a larger field, $\mathbb{F}_q$, and using DBO as a subroutine. The larger field $\mathbb{F}_q$ is chosen to satisfy the condition that f is q-bounded. The embedding is performed as follows. For $s_j \in \mathbb{F}_p$, let σ_j denote the minimal positive integer such that $\sigma_j \equiv s_j \pmod p$. Let $\tilde{s}_j \equiv \sigma_j \pmod q$ the $\mathbb{F}_q$ correspondent of s_j in the q world. Let $\tilde{s} = (\tilde{s}_1, \ldots, \tilde{s}_N)$. Now, let $\tilde{f} : \mathbb{F}_q^N \longrightarrow \mathbb{F}_q$ denote the function corresponding to f in the q-world. That is, $\tilde{f}$ is obtained from f by replacing the leading coefficients of the monomials with their q-world correspondents. DBO-qB may be invoked by the parties to find $f(s)$.

The DRM three-round DBO-qB scheme

Calling DBO. Use DBO to find $\tilde{y} = \tilde{f}(\tilde{s}) \in \mathbb{F}_q$.

Computing p-world output. Let σ denote the minimal positive integer such that $\sigma \equiv \tilde{y} \pmod q$, and let $y \equiv \sigma \pmod p$. Output y.

Theorem A.1. *DBO-qB is a three-round N-party P-MPC scheme for arithmetic functions which has perfect correctness, perfect passive security, threshold $N-1$, communication complexity $O(kN^3 n2^n)$, space complexity $O(kN^2 n2^n)$, and f-independent CR.*

Proof. Since all q-world inputs are non-zero elements of $\mathbb{F}_q$, security of DBO-qB follows from that of DBO. Correctness follows from that of DBO and the fact that f is q-bounded. By construction, the round complexity is two. Now, in DBO-qB, all the messages and the CR are $\mathbb{F}_q$ elements. At the worst case, $q \approx p^{pN}$, and hence, the number of bits required for each element is $\lceil \log q \rceil \approx \log(p^{pN}) = pN \log p = 2^{\log p} N \log p = nN2^n$. Since the number of messages and amount of CR remains unchanged, the exact space and communication complexities of DBO-qB are obtained from those of DBO by replacing n with $nN2^n$. □

DBO-qB solves the general case by replacing $\mathbb{F}_p$ elements with $\mathbb{F}_q$ elements, hence the factor 2^n. DBO-IS and DBO-IS$_2$, which we now present, avoid the 2^n factor. Instead, these schemes replace f with a K-monomials version of it.

The DRM three-round DBO-IS scheme solves the general case by splitting each input to a sum of two non-zero elements, and replacing f with *the split-inputs version* of f. That is, for a polynomial f, let $\varphi : \mathbb{F}_p^{2N} \longrightarrow \mathbb{F}_p$ denote the function obtained from f by replacing each variable x_i of f with the sum of two variables, z_i and w_i, as follows:

$$\begin{aligned}\varphi(z_1 \ldots, z_N, w_1, \ldots, w_N) &= \sum_{l \in \mathcal{L}} a_l \cdot (z_1 + w_1)^{l_1} \cdots (z_N + w_N)^{l_N} \\ &= \sum_{\lambda \in \Lambda} b_\lambda \cdot z_1^{\lambda_1} \cdots z_N^{\lambda_N} \cdot w_1^{\lambda_{N+1}} \cdots w_N^{\lambda_{2N}},\end{aligned} \tag{4}$$

where $\lambda = (\lambda_1, \ldots, \lambda_{2N})$, $\Lambda = \{0, 1, \ldots, p-1\}^{2N}$, and $b_\lambda \in \mathbb{F}_p$. φ is the split-inputs version of f. We note that, since f is a polynomial of N variables, $k \leq p^N$, and since φ is a polynomial of $2N$ variables, $K \leq p^{2N} = (p^N)^2$.

Now, if $p \neq 2$, DBO-IS may be invoked by the parties to find $f(s)$.

The DRM three-round DBO-IS scheme

Each party $\mathcal{P}_j$ arbitrarily picks $\alpha_j, \beta_j \in \mathbb{F}_p^\times$ such that $s_j = \alpha_j + \beta_j$.
Use DBO to find $y = \varphi(\alpha_1, \ldots, \alpha_N, \beta_1, \ldots, \beta_N)$. Output y.

Theorem A.2. *DBO-IS is a three-round N-party P-MPC scheme for $\mathbb{F}_p$ functions ($p \neq 2$) which has perfect correctness, perfect passive security, threshold $N-1$, f-independent CR, communication complexity $O(N^2nK)$, and space complexity $O(NnK)$, where K is the number of monomials of the split-inputs version of f.*

Proof. Correctness follows from that of DBO and from the observation that if $\alpha_i, \beta_i \in \mathbb{F}_p$ $(1 \leq i \leq N)$ are such that for every $i \in [N]$ it holds that $s_i = \alpha_i + \beta_i$, then $f(s_1, \ldots, s_N) = \varphi(\alpha_1, \ldots, \alpha_N, \beta_1, \ldots, \beta_N)$. The security and complexity properties follow immediately from those of DBO. □

Now, in $\mathbb{F}_2$, 1 cannot be written as a sum of two non-zero elements. This is the reason for the requirement $p \neq 2$ in Theorem A.2. DBO-IS$_2$ solves the case $p = 2$ by embedding $\mathbb{F}_2$ in $\mathbb{F}_3$ and using DBO-IS as a subroutine. The embedding is performed as follows. The elements $0, 1 \in \mathbb{F}_2$ are identified with $0, 1 \in \mathbb{F}_3$. For $s_j \in \mathbb{F}_2$ let $\overline{s_j}$ denote the $\mathbb{F}_3$ correspondent of s_j. $\mathbb{F}_2$ operations are identified with $\mathbb{F}_3$ operations as follows. $\mathbb{F}_2$-multiplication is identified with $\mathbb{F}_3$-multiplication, and $\mathbb{F}_2$-addition is identified with $Add : \mathbb{F}_3^2 \longrightarrow \mathbb{F}_3$, $Add(x, y) = x + y + xy$. Let $\overline{f} : \mathbb{F}_3^N \longrightarrow \mathbb{F}_3$ denote the $\mathbb{F}_3$ correspondent of f. That is, $\overline{f}$ is obtained from f by replacing the $\mathbb{F}_2$-operations '$\cdot$' and '+' of f with the $\mathbb{F}_3$-operations '$\cdot$' and '*Add*'. The parties may invoke DBO-IS$_2$ to find $f(s)$.

The DRM three-round DBO-IS$_2$ scheme

Use DBO-IS to find $y = \overline{f}(\overline{s_1}, \ldots, \overline{s_N})$. Output y.

Theorem A.3. *DBO-IS$_2$ is a three-round N-party P-MPC scheme for arithmetic functions over $\mathbb{F}_2$ which has perfect correctness, perfect passive security, threshold $N-1$, f-independent CR, communication complexity $O(N^2nK)$, and space complexity $O(NnK)$, where K is the number of monomials of the split-input version of the $\mathbb{F}_3$ correspondent of f.*

The proof follows immediately from construction. The schemes suggested above use DBO as a subroutine to solve the case of evaluating arbitrary functions over possibly-zero inputs. We construct OTS-qB, OTS-IS, and OTS-IS$_2$ schemes by using OTS as a subroutine in a similar fashion. These schemes handle the target scenario with optimal round complexity.

Theorem A.4. *OTS-qB is a two-round N-party P-MPC scheme for arithmetic functions which has perfect correctness, perfect passive security, threshold $N-1$, communication complexity $O(kN^3n2^n)$, space complexity $O(kN^2n2^n)$, and f-independent CR.*

Theorem A.5. *OTS-IS is a two-round N-party P-MPC scheme for $\mathbb{F}_p$ functions ($p \neq 2$) which has perfect correctness, perfect passive security, threshold $N-1$, f-independent CR, communication complexity $O(N^2nK)$, and space complexity $O(NnK)$, where K is the number of monomials of the split-inputs version of f.*

Theorem A.6. *OTS-IS$_2$ is a two-round N-party P-MPC scheme for arithmetic functions over $\mathbb{F}_2$ which has perfect correctness, perfect passive security, threshold $N-1$, f-independent CR, communication complexity $O(N^2nK)$, and space complexity $O(NnK)$, where K is the number of monomials of the split-input version of the $\mathbb{F}_3$ correspondent of f.*