Homomorphic Secret Sharing for Multipartite and General Adversary Structures Supporting Parallel Evaluation of Low-degree Polynomials

Reo Eriguchi[1,3] and Koji Nuida[2,3]

[1] The University of Tokyo, Tokyo, Japan
`reo-eriguchi@g.ecc.u-tokyo.ac.jp`
[2] Kyushu University, Fukuoka, Japan
`nuida@imi.kyushu-u.ac.jp`
[3] National Institute of Advanced Industrial Science and Technology, Tokyo, Japan

Abstract. Homomorphic secret sharing (HSS) for a function f allows input parties to distribute shares for their private inputs and then locally compute output shares from which the value of f is recovered. HSS can be directly used to obtain a two-round multiparty computation (MPC) protocol for possibly non-threshold adversary structures whose communication complexity is independent of the size of f. In this paper, we propose two constructions of HSS schemes supporting parallel evaluation of a single low-degree polynomial and tolerating multipartite and general adversary structures. Our multipartite scheme tolerates a wider class of adversary structures than the previous multipartite one in the particular case of a single evaluation and has exponentially smaller share size than the general construction. While restricting the range of tolerable adversary structures (but still applicable to non-threshold ones), our schemes perform ℓ parallel evaluations with communication complexity approximately $\ell/\log\ell$ times smaller than simply using ℓ independent instances. We also formalize two classes of adversary structures taking into account real-world situations to which the previous threshold schemes are inapplicable. Our schemes then perform $O(m)$ parallel evaluations with almost the same communication cost as a single evaluation, where m is the number of parties.

Keywords: Homomorphic secret sharing · General adversary structure · Parallel evaluation.

1 Introduction

This paper concerns a large-scale multiparty computation (MPC), in which the number m of parties are considerably large, e.g., $m = 1000$ [25]. We also aim at realizing parallel evaluation of a single low-degree polynomial (or SIMD operations) [9, 19]. That setting practically appears in privacy-preserving statistics and machine learning. A fundamental analysis such as linear regression can be expressed as an m-variate polynomial of constant degree [11, 24]. We should involve a large number of inputs to make analysis results useful. Furthermore, amortized communication cost is important if the same analysis is performed on different data sets.

The privacy requirement is specified by an adversary structure $\varDelta$, a family of all possible corrupted subsets of the whole set P. Although $\varDelta$ may contain all strict subsets, the all-but-one corruption is an unrealistically strong setting. Moreover, the only possible solution in that setting would be using fully homomorphic encryption (FHE) [21, 32] or general-purpose MPC (e.g., [3, 13, 23]). The existing FHE schemes are based on a narrow class of assumptions and their concrete efficiency leaves much to be desired. General-purpose MPC results in considerably large communication complexity proportional to the function description size, which is $O(m^d)$ for a degree-d polynomial. It is then important to improve performance by focusing on practical adversary structures. For example, many real-world situations are expressed by multipartite adversary structures [18], in which P is divided into L parts P_j and whether each subset X is in $\varDelta$ is determined by $(|X \cap P_1|, \ldots, |X \cap P_L|)$.

Homomorphic secret sharing (HSS) [6] allows m parties to distribute shares for their inputs and then locally compute output shares from which the output of a function is recovered. HSS offers protection against bounded collusion and can be constructed from weak assumptions, such as the intractability of the Diffie-Hellman problem [14], or even information-theoretically. It is directly used to obtain a two-round MPC protocol whose point-to-point communication cost is linear in its share size, which is independent of the function description size.

There are several constructions of HSS schemes. The packed secret sharing scheme [19] is an information-theoretic scheme that generalizes Shamir's scheme [31] to support parallel evaluation, the scheme of [17] is an information-theoretic one for multipartite adversary structures, and the schemes of [26, 30] are computational variants of Woodruff and Yekhanin's scheme [34] and the CNF scheme [2], respectively, which tolerate wider classes of adversary structures.

However, they do not give a satisfactory solution to practical non-threshold adversary structures $\varDelta$. The schemes [19, 26] need to set a corruption threshold to the *maximum* size of $X \in \varDelta$ and then are inapplicable if $\varDelta$ contains *at least one* set of size exceeding their tolerable thresholds. The construction [30] is applicable to any adversary structure but results in exponentially large share size for multipartite ones. In addition, $\varDelta$ is practically much smaller than maximally tolerable adversary structures of [17, 26, 30][4]. Then, the previous schemes satisfy unnecessarily strong privacy requirements and would limit the possibility of parallel evaluation. In summary, we need to construct HSS schemes tailored to given non-threshold adversary structures to tolerate corruptions in real-world situations and also to improve the amortized communication cost.

1.1 Our Results

We propose HSS schemes for multipartite and general adversary structures assuming k-HE, homomorphic encryption for polynomials of degree $k = O(1)$. Our

[4] Here, a maximally tolerable adversary structure of an HSS scheme means $\varDelta$ such that the scheme cannot tolerate $\varDelta \cup \{X\}$ for any $X \notin \varDelta$.

technical novelty is applying the packing technique of [19] to the non-threshold schemes [17, 30]. It is especially not straightforward to apply it to [30], which does not involve polynomial interpolation.

HSS for Multipartite Adversary Structures. Let $\varDelta$ be an L-partite adversary structure and N be the number of all vectors associated with maximal sets of $\varDelta$ (Table 1). When performing ℓ parallel evaluations of a degree-d polynomial, the input and output share sizes of our scheme are $O(N \log(m+\ell))$ and $O(\log(m+\ell))$, respectively, omitting polynomial factors of the security parameter λ. Note that $N = O(m^L)$ and L is practically small, e.g., $L = 2$ [18]. As a result, we obtain an MPC protocol for $\varDelta$ whose communication complexity is logarithmic in ℓ and is approximately $\ell/\log \ell$ times smaller than [17] while the range of tolerable $\varDelta$ degrades as ℓ increases (Fig. 1 (left)). Our scheme for $\ell = 1$ tolerates a wider class of adversary structures than [17]. By setting $L = 1$, we obtain a threshold HSS scheme (Table 2). Given k, d, and m, its tolerable threshold is strictly larger than [19]. Compared to [26], there is a trade-off between thresholds and share sizes. If a threshold t is smaller than $t^* = \lceil (k+1)m/d - 1 \rceil$, our scheme can perform $\ell \le t^* - t$ parallel evaluations with communication cost $\ell/\log \ell$ times smaller than [26].

Table 1. Comparison of HSS schemes for an L-partite adversary structure $\varDelta$ supporting ℓ parallel evaluations of a degree-d polynomial. Let $\varPi = (P_j)_{j \in [L]}$ be an L-partition of P, $\{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_N\}$ be the collection of $(|X \cap P_j|)_{j \in [L]}$ for all maximal sets X, and $\varPhi^{\varPi}(P) = (|P_j|)_{j \in [L]}$ (see Section 3.3). Define $\epsilon_{k,d}(\varDelta) = \min\{\left|(k+1)\varPhi^{\varPi}(P) - (\boldsymbol{a}_{i_1} + \cdots + \boldsymbol{a}_{i_d})\right|_\infty : 1 \le i_1 \le \ldots \le i_d \le N\}$, where $|\boldsymbol{v}|_\infty = \max_{j \in [L]}\{\max\{v_j, 0\}\}$ for $\boldsymbol{v} = (v_j)_{j \in [L]} \in \mathbb{Z}^L$.

Reference	[17]	Ours (Corollary 2)
Condition on $\varDelta$	$\epsilon_{0,d}(\varDelta) > 0$	$\epsilon_{k,d}(\varDelta) > d(\ell - 1)$
Input share size	$O(\ell N \log m)$	$O(N \log(m+\ell))$
Output share size	$O(\ell \log m)$	$O(\log(m+\ell))$
Assumption	–	k-HE

Table 2. Comparison of HSS schemes for a threshold adversary structure supporting ℓ parallel evaluations of a degree-d polynomial.

Reference	[19]	[26]	Ours (Corollary 1)
Threshold	$m/d - \ell$	$(k+1)m/d - 1$	$(k+1)m/d - \ell$
Input share size	$O(\log(m+\ell))$	$O(\ell \log m)$	$O(\log(m+\ell))$
Output share size	$O(\log(m+\ell))$	$O(\ell \log m)$	$O(\log(m+\ell))$
Assumption	–	k-HE	k-HE

HSS for General Adversary Structures. Let Δ be an adversary structure and N be the number of all maximal sets of Δ (Table 3). For ℓ parallel evaluations, the input and output share sizes of our scheme are $O(N \log(m+\ell))$ and $O(\log(m+\ell))$, respectively, omitting $\mathsf{poly}(\lambda)$ factors. Note that our scheme is applicable only to Δ such that $N = \mathsf{poly}(m)$ to guarantee polynomial computational complexity while $N = O(2^m)$ in the worst case. Nevertheless, as shown below, there is an interesting class of adversary structures even if $N = O(m)$. We obtain an MPC protocol for Δ whose communication complexity is logarithmic in ℓ and is $\ell/\log \ell$ times smaller than [30] while the range of tolerable Δ degrades as ℓ increases (Fig. 1 (right)).

Table 3. Comparison of HSS schemes for a general adversary structure Δ supporting ℓ parallel evaluations of a degree-d polynomial. Let N be the number of all the maximal sets $\{B_1, \ldots, B_N\}$ and $\mathbf{1}_m$ is the all-ones vector. Let $\boldsymbol{a}_i \in \mathbb{Z}^m$ be such that the j-th entry is 1 if $j \in B_i$ and otherwise 0. Define $\delta_{k,d}(\Delta) = \min\{|(k+1)\mathbf{1}_m - (\boldsymbol{a}_{i_1} + \cdots + \boldsymbol{a}_{i_d})|_+ : 1 \le i_1 \le \ldots \le i_d \le N\}$, where $|\boldsymbol{v}|_+ = \sum_{j \in [m]} \max\{v_j, 0\}$ for $\boldsymbol{v} = (v_j)_{j \in [m]} \in \mathbb{Z}^m$.

Reference	[30]	Ours (Corollary 3)
Condition on Δ	$\delta_{k,d}(\Delta) > 0$	$\delta_{k,d}(\Delta) > (d+1)(\ell-1)$
Input share size	$O(\ell N)$	$O(N \log(m+\ell))$
Output share size	$O(\ell)$	$O(\log(m+\ell))$
Assumption	k-HE	k-HE

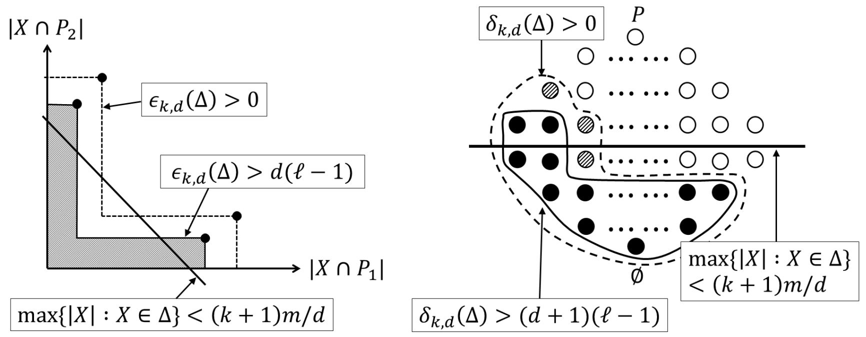
$|X \cap P_2|$
$\epsilon_{k,d}(\Delta) > 0$
$\epsilon_{k,d}(\Delta) > d(\ell-1)$
$|X \cap P_1|$
$\max\{|X| : X \in \Delta\} < (k+1)m/d$
$\delta_{k,d}(\Delta) > 0$
P
$\emptyset$
$\max\{|X| : X \in \Delta\}$
$< (k+1)m/d$
$\delta_{k,d}(\Delta) > (d+1)(\ell-1)$

Fig. 1. Conceptual comparison between the threshold constraint, $\epsilon_{k,d}(\Delta) > 0$, and $\epsilon_{k,d}(\Delta) > d(\ell-1)$ for a 2-partite adversary structure (left) and between the threshold constraint, $\delta_{k,d}(\Delta) > 0$, and $\delta_{k,d}(\Delta) > (d+1)(\ell-1)$ for a general adversary structure (right). The circles to the right show the subsets of P ordered with respect to inclusion.

Formalization of Practical Adversary Structures. We formalize two classes of adversary structures and show advantages of our schemes over [17, 19, 26, 30].

– Unbalanced 2-partite adversary structure: For parameters τ, σ and a 2-partition $\Pi = (P_1, P_2)$, we define a Π-partite adversary structure which contains all subsets X of size at most τm satisfying either $|X \cap P_1| \leq \sigma m$ or $|X \cap P_2| \leq \sigma m$. We suppose a situation in which an adversary belongs to either of the two parts and corrupts at most τm parties as in the threshold adversary structure but he is only able to corrupt σm parties from the other part. Our multipartite scheme can be applied to parameters τ, σ to which the threshold schemes [19, 26] cannot. It supports $\ell = O(m)$ parallel evaluations and decreases communication cost by $O(m)$ times compared to [17].
– Adversary structure induced by a random graph: For a graph on m parties (vertices), we define an adversary structure which contains the neighborhood of j for all vertices j, i.e., a subset of all vertices adjacent to j including j itself. That captures a situation in which an adversary is one of parties and corrupts all adjacent parties in the graph. We show that for a random graph whose edges occur independently with probability p, our HSS scheme for general adversary structures can be applied with high probability. Under certain parameter settings, our scheme supports $\ell = O(m)$ parallel evaluations and decreases communication cost by $O(m/\log m)$ times compared to [30]. The threshold schemes [19, 26] cannot be applied with high probability.

1.2 Related Work

There are packing methods of homomorphic encryption which packs many elements into a single ciphertext [12, 32] and it may be used to reduce the share size of [26, 30]. However, packing methods asymptotically reducing ciphertext size are based on the LWE assumption or the one that implies FHE.

Under the LWE assumption, spooky encryption [15] provides an HSS scheme for the adversary structure containing all strict subsets. The schemes [5–7] assume the client-server model, in which input parties have to trust and give sensitive information related to their inputs to a small number of external servers. They are only applied to $m = O(1)$ and not suited to our setting.

The authors of [26] show an MPC protocol with preprocessing whose online communication complexity is independent of the function description size. In the plain model, however, it ends up communication complexity linear in the description size since the preprocessing phase is jointly executed by input parties.

2 Technical Overview

In this section, we provide an overview of our constructions of HSS schemes. We give more detailed descriptions and security proofs in the following sections.

2.1 Adjusting the ILM Compiler [26] to Parallel Evaluation

The authors of [26] propose a compiler, which we call the ILM compiler, that converts an information-theoretic HSS (IT-HSS) scheme with *recovery information* to an HSS scheme using k-HE. In an IT-HSS scheme with recovery information, the decoding algorithm Dec needs to use auxiliary information to compute an output of a function. Our technical contributions are constructions of IT-HSS schemes supporting parallel evaluation, from which our HSS schemes are obtained by the ILM compiler. However, the original compiler assumes a single evaluation and a naive generalization ends up in output shares whose size is proportional to the number of evaluations. We need to consider in more detail an algebraic property of the IT-HSS scheme that the compiler assumes.

Specifically, to apply the ILM compiler, the output of Dec has to be expressed as $\boldsymbol{a}^\top \boldsymbol{y}$, where $\boldsymbol{y}$ is a vector representing an output share of the IT-HSS scheme and $\boldsymbol{a}$ is a vector whose entries are degree-k polynomials of recovery information. This is why the ILM compiler successfully works by letting a new sharing algorithm output ciphertexts $\widetilde{\boldsymbol{a}}$ for $\boldsymbol{a}$ and letting a new evaluation algorithm compute a ciphertext for $\boldsymbol{a}^\top \boldsymbol{y}$ from $\boldsymbol{y}$ and $\widetilde{\boldsymbol{a}}$. For ℓ parallel evaluations, the output of Dec is expressed as $\boldsymbol{A}\boldsymbol{y}$ using a matrix $\boldsymbol{A} = (\boldsymbol{a}_1, \dots, \boldsymbol{a}_\ell)^\top$ whose entries are degree-k polynomials of recovery information. However, if the IT-HSS scheme only satisfies this algebraic property, an output share consists of ℓ ciphertexts for $\boldsymbol{a}_i^\top \boldsymbol{y}$, whose size is proportional to ℓ. We require a more involved property that the output of Dec is expressed as $\boldsymbol{C}\boldsymbol{A}\boldsymbol{y}$ using $\boldsymbol{A} = (\boldsymbol{a}_1, \dots, \boldsymbol{a}_h)^\top$ whose entries are degree-k polynomials of recovery information and an ℓ-by-h *constant* matrix $\boldsymbol{C}$. Then, an output share consists of h ciphertexts for $\boldsymbol{a}_i^\top \boldsymbol{y}$ and the decoding procedures can be done by only using public information $\boldsymbol{C}$. An important feature is that the output share size is constant (ignoring $\mathsf{poly}(\lambda)$ factors) if we can choose h independent of ℓ.

2.2 HSS for Multipartite Adversary Structures

As a warm-up, we construct an HSS scheme for threshold adversary structures by applying the ILM compiler to a variant of the packed secret sharing scheme [19] whose recovery information is derivatives of an interpolating polynomial. That construction is a natural generalization of the IT-HSS scheme [26, 34], which is a variant of Shamir's scheme using derivatives as recovery information. Specifically, we fix $m + \ell$ points $\zeta_1, \dots, \zeta_m, \eta_1, \dots, \eta_\ell$. For a threshold t and a vector of ℓ inputs $\boldsymbol{x}$, the sharing algorithm chooses a random polynomial φ of degree $t + \ell - 1$ such that $(\varphi(\eta_i))_{i\in[\ell]} = \boldsymbol{x}$, and set input shares as $(\varphi(\zeta_j))_{j\in[m]}$ and recovery information as the derivatives of φ of the up to k-th order on the ζ_j's. Since the degree of the interpolating polynomial is at most $d(t+\ell-1)$ after evaluating a degree-d polynomial, the decoding succeeds if $d(t+\ell-1) < (k+1)m$ due to Hermite interpolation [33].

To deal with an L-partite adversary structure, we recall the HSS scheme [17], which decompose it to many threshold sub-structures and combines Shamir's schemes realizing them. We replace Shamir's schemes used there with the above variant of the packed scheme. In our resulting scheme, an input vector $\boldsymbol{x}$ can be additively split into $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N$, where N is the number of the sub-structures, and each $\boldsymbol{x}_u$ is shared by using the above threshold scheme. We can evaluate a degree-d polynomial if each monomial $\boldsymbol{x}_{u_1} * \cdots * \boldsymbol{x}_{u_d}$ is recovered by parties in some part of the L parts, where $*$ denotes the element-wise product. This is actually the case if the number of parties in some part exceeds a certain threshold relating to the degree of the interpolating polynomial for $\boldsymbol{x}_{u_1} * \cdots * \boldsymbol{x}_{u_d}$, which is formalized as the condition in Table 1.

Then, we can apply the ILM compiler to that IT-HSS scheme with recovery information. However, there still remains a problem of *context hiding*, which guarantees that output shares reveal nothing beyond the output value and is necessary for an application to MPC. Using a technique similar to [27], an HSS scheme for a single evaluation can be made context-hiding by modifying output shares so that they add up to the single output value and re-randomizing them by an additive sharing of zero. To output ℓ values, however, that solution results in communicating ℓ output shares each of which is an additive share of each output value. We observe that the set of all possible consistent output shares is a translation of a subspace W in a linear space V by an element $v \in V$ relating to the output values. Thus, we can re-randomize output shares by adding a random element from W, which can be predistributed in the sharing phase without increasing the share size asymptotically.

2.3 HSS for General Adversary Structures

Our starting point is the HSS scheme [30], which is the result of applying the ILM compiler to the CNF scheme [2]. In their scheme for Δ, a scalar input x is additively split into N elements x_u, where N is the number of all maximal sets of Δ and an index u corresponds to the u-th maximal set B_u. Then, an input share for the j-th party consists of x_u for all u with $j \notin B_u$ and his recovery information consists of x_u for all u with $j \in B_u$. To deal with a vector $\boldsymbol{x}$, we use the packing technique of [19], which is not straightforward since the above procedures do not involve polynomial interpolation. Specifically, we additively split $\boldsymbol{x}$ into N vectors $\boldsymbol{x}_u$ and distribute shares for $\boldsymbol{x}_u$ using the packed secret sharing among parties not in B_u instead of simply sending a copy of $\boldsymbol{x}_u$. That is, for each $u \in [N]$, the sharing algorithm chooses a random polynomial φ_u of degree $\ell - 1$ such that $(\varphi_u(\eta_i))_{i\in[\ell]} = \boldsymbol{x}_u$. It sets an input share for the j-th party as $\varphi_u(\zeta_j)$ for all u with $j \notin B_u$ and his recovery information as $\varphi_u(\zeta_j)$ for all u with $j \in B_u$ and some derivatives of φ_u on ζ_j for all $u \in [N]$. The input share size increases logarithmically in ℓ since each party receives elements from a field of size at least $m + \ell$.

The above scheme can evaluate a degree-d polynomial if for $\boldsymbol{u} = (u_1, \ldots, u_d) \in [N]^d$, sufficiently many values and derivatives of $\varphi_{u_1} \cdots \varphi_{u_d}$ is computed from input shares and degree-k polynomials of recovery information due to Hermite interpolation. However, this solution falls short of preventing the output share size from increasing linearly in ℓ since the coefficients in Hermite interpolation formula to compute $(\varphi_{u_1} \cdots \varphi_{u_d})(\eta_i)$ depend on $i \in [\ell]$ and hence parties need to send output shares separately for each $i \in [\ell]$. Instead, we introduce public polynomials $p_{\boldsymbol{u}}$ for all $\boldsymbol{u} \in [N]^d$ such that $p_{\boldsymbol{u}}(\eta_i) = 1$ for all $i \in [\ell]$ and that the values and derivatives of $g = \sum_{\boldsymbol{u}\in[N]^d} p_{\boldsymbol{u}} \varphi_{u_1} \cdots \varphi_{u_d}$ is computed from input shares and degree-k polynomials of recovery information. The condition in Table 3 comes from a sufficient condition for such polynomials to exist. An important feature is that the number of the values and derivatives of g to be communicated is independent of ℓ and that the decoding algorithm locally recovers g (and hence $g(\eta_i)$ for all $i \in [\ell]$) from them. Now, that scheme can perform ℓ parallel evaluations while the share size is logarithmic in ℓ. On the other hand, it has to pay the cost of degrading the range of tolerable adversary structures as ℓ increases since the degree of the interpolating polynomial g becomes higher.

3 Preliminaries

Notations. Let $\mathbb{Z}_+$ and $\mathbb{R}_+$ denote the sets of all non-negative integers and all non-negative real numbers, respectively. For $\ell \in \mathbb{N}$, define $[\ell] = \{1, \dots, \ell\}$ and $[0..\ell] = \{0\} \cup [\ell]$. The power set of a set X is denoted by 2^X and X^m is the Cartesian product of m copies of X. Let $\mathbb{F}$ be a finite field or the ring of integers $\mathbb{Z}$. The vector $\mathbf{1}_m \in \mathbb{F}^m$ is the one whose entries are all one. The i-th component of $\boldsymbol{v} \in \mathbb{F}^m$ is denoted by $\boldsymbol{v}(i)$. If $\mathbb{F} = \mathbb{Z}$, define $|\boldsymbol{v}|_+ = \sum_{i \in [m]} \max\{\boldsymbol{v}(i), 0\}$ and $|\boldsymbol{v}|_\infty = \max_{i \in [m]}\{\max\{\boldsymbol{v}(i), 0\}\}$. For $\boldsymbol{v}, \boldsymbol{w} \in \mathbb{Z}^m$, we write $\boldsymbol{v} \preceq \boldsymbol{w}$ if $\boldsymbol{v}(i) \leq \boldsymbol{w}(i)$ for any $i \in [m]$ and $\boldsymbol{v} \prec \boldsymbol{w}$ if $\boldsymbol{v} \preceq \boldsymbol{w}$ and $\boldsymbol{v} \neq \boldsymbol{w}$. Let $f \in \mathbb{F}[X_1, \dots, X_n]$ be an n-variate polynomial over $\mathbb{F}$. We say that f is a degree-k polynomial if its total degree is at most k. For $\ell \in \mathbb{N}$ and $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_n) \in (\mathbb{F}^\ell)^n$, we define $\mathcal{P}_\ell(f, (\boldsymbol{x}_1, \dots, \boldsymbol{x}_n)) = (f(\boldsymbol{x}_1(j), \dots, \boldsymbol{x}_n(j)))_{j \in [\ell]}$. Define the differential operator $\mathcal{D}$ as $(\mathcal{D}\varphi)(X) = \sum_i i c_i X^{i-1}$ for $\varphi = \sum_i c_i X^i \in \mathbb{F}[X]$. We assume $\mathcal{D}^0 \varphi = \varphi$. For two random variables R_1 and R_2 over a set $\mathcal{U}$, we write $R_1 \approx R_2$ if the statistical distance $\mathbb{SD}(R_1, R_2) = (1/2) \sum_{u \in \mathcal{U}} |\Pr[R_1 = u] - \Pr[R_2 = u]|$ is negligible in a security parameter. We write $u \leftarrow_{\$} \mathcal{U}$ if u is randomly chosen from $\mathcal{U}$.

3.1 Hermite Interpolation

We recall Hermite interpolation [33], which generalizes Lagrange interpolation in that it recovers a polynomial using its derivatives and values on given points.

Proposition 1 ([33]). *Let $m, k \in \mathbb{N}$ and $r_j \in [0..k]$ for $j \in [m]$. Let $\mathbb{F}$ be a prime field such that $|\mathbb{F}| \geq \max\{m, k+1\}$. Let $\zeta_1, \dots, \zeta_m$ be m distinct elements of $\mathbb{F}$. Let $y_{j,w} \in \mathbb{F}$ for each $j \in [m]$ and $w \in [0..r_j]$. Then, there exists a unique polynomial $g \in \mathbb{F}[X]$ such that $\deg g < \sum_{j \in [m]}(r_j + 1)$ and $\mathcal{D}^w g(\zeta_j) = y_{j,w}$ for all $j \in [m]$ and $w \in [0..r_j]$. Furthermore, g can be written as $g(X) = \sum_{j=1}^{m} \sum_{w=0}^{r_j} A_{j,w}(X) y_{j,w}$ using some polynomials $A_{j,w}$ of degree less than $\sum_{j=1}^{m}(r_j + 1)$ whose coefficients are independent of the $y_{j,w}$'s.*

We require that $\mathbb{F}$ is a prime field such that $|\mathbb{F}| \geq k + 1$ to guarantee that $w!$ for all $w \in [k]$, which appear as denominators in Hermite interpolation formula, have inverses in $\mathbb{F}$. Let $\ell \in \mathbb{N}$ and assume that $|\mathbb{F}| \geq \max\{m + \ell, k + 1\}$. Let $\zeta_1, \dots, \zeta_m, \eta_1, \dots, \eta_\ell$ be $m + \ell$ distinct elements of $\mathbb{F}$, $\boldsymbol{\zeta} = (\zeta_1, \dots, \zeta_m) \in \mathbb{F}^m$, and $\boldsymbol{\eta} = (\eta_1, \dots, \eta_\ell) \in \mathbb{F}^\ell$. Applying Proposition 1 to the case of $r_1 = \cdots = r_m = k$ implies an $\mathbb{F}$-linear map $\mathsf{Hermite}_{\boldsymbol{\zeta},\boldsymbol{\eta}} : \mathbb{F}^{(k+1)m} \to \mathbb{F}^\ell$ such that for any polynomial $g \in \mathbb{F}[X]$ of degree less than $(k + 1)m$, it holds that $\mathsf{Hermite}_{\boldsymbol{\zeta},\boldsymbol{\eta}}((\boldsymbol{z}_j)_{j \in [m]}) = (g(\eta_i))_{i \in [\ell]}$, where $\boldsymbol{z}_j = (\mathcal{D}^w g(\zeta_j))_{w \in [0..k]}$ for $j \in [m]$.

3.2 Homomorphic Encryption

We recall the notion of homomorphic encryption for degree-k polynomials.

Definition 1 (Homomorphic Encryption (HE)). *A homomorphic encryption scheme* $\mathsf{HE} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *for degree-k polynomials over* $\mathbb{F}$*, k-HE for short, consists of the following PPT algorithms:*

- $\mathsf{KGen}(1^\lambda)$*: Given the security parameter* 1^λ*, the key generation algorithm outputs a public key* pk *and a secret key* sk*.*
- $\mathsf{Enc}(\mathsf{pk}, \boldsymbol{x})$*: Given the public key* pk *and a message* $\boldsymbol{x} \in \mathbb{F}^n$ *for some* $n = \mathsf{poly}(\lambda)$*, the encryption algorithm outputs a ciphertext* $\boldsymbol{c} \in \mathcal{C}^n$ *in some ciphertext space* $\mathcal{C}$*.*
- $\mathsf{Eval}(\mathsf{pk}, f, \boldsymbol{c})$*: Given the public key* pk*, (a description of) a degree-k polynomial* $f \in \mathbb{F}[X_1, \dots, X_n]$*, and a ciphertext* $\boldsymbol{c} \in \mathcal{C}^n$ *for some* $n = \mathsf{poly}(\lambda)$*, the evaluation algorithm outputs a ciphertext* $c' \in \mathcal{C}$*.*
- $\mathsf{Dec}(\mathsf{sk}, \boldsymbol{c})$*: Given the secret key* sk *and a ciphertext* $\boldsymbol{c} \in \mathcal{C}^n$ *for some* $n = \mathsf{poly}(\lambda)$*, the decryption algorithm outputs a plaintext* $\boldsymbol{x} \in \mathbb{F}^n$*.*

As in [27, 30], we consider the standard notions of compactness, correctness, IND-CPA security, circuit privacy [8], and their multi-key variants. We provide the formal definitions in the full version. We focus on small $k = O(1)$, for which it is not known how k-HE can be bootstrapped [21] into FHE. For $k \in \{1, 2\}$, there exist efficient k-HE schemes [1, 4, 11, 16, 20, 29]. For general $k = O(1)$, k-HE schemes can be constructed from the LWE assumption with smaller parameters than those which imply FHE, and therefore concretely efficient. There exist several multi-key HE schemes in the literature [10, 28].

3.3 Adversary Structures

Let $P = [m]$ for $m \in \mathbb{N}$. A family $\varDelta$ of subsets of P is monotonically decreasing if $A \in \varDelta$ and $A \supseteq B$ imply $B \in \varDelta$ for any $A, B \subseteq P$. We call a monotonically decreasing family of subsets of P an adversary structure on P. A set $A \in \varDelta$ is called maximal if $A \subsetneq B$ implies $B \notin \varDelta$ for any $B \subseteq P$. Let $\varPi = (P_1, \dots, P_L)$ be an L-partition of P, i.e., $P_i \cap P_j = \emptyset$ for $i \neq j$ and $P = \bigcup_{j \in [L]} P_j$. A permutation τ on P is called a $\varPi$-permutation if $\tau(P_j) = P_j$ for any $j \in [L]$. An adversary structure $\varDelta$ is called $\varPi$-partite [18] if $\tau(B) \in \varDelta$ for any $B \in \varDelta$ and any $\varPi$-permutation τ. There is a geometric representation of $\varPi$-partite adversary structures. Let $\varPhi^\varPi : 2^P \to \mathbb{R}^L$ be a map defined by $\varPhi^\varPi(X) = (|X \cap P_j|)_{j \in [L]}$. A $\varPi$-partite adversary structure $\varDelta$ is uniquely determined by $\varPhi^\varPi(\varDelta)$. Note that, if $\boldsymbol{a} \in \varPhi^\varPi(\varDelta)$ and $\boldsymbol{a} \succeq \boldsymbol{b}$, then $\boldsymbol{b} \in \varPhi^\varPi(\varDelta)$ for any $\boldsymbol{a}, \boldsymbol{b} \in \varPhi^\varPi(2^P)$. Thus, any $\varPi$-partite adversary structure $\varDelta$ is uniquely determined only by specifying $\max \varPhi^\varPi(\varDelta) := \{\boldsymbol{a} \in \varPhi^\varPi(\varDelta) : \boldsymbol{a} \prec \boldsymbol{b} \preceq \varPhi^\varPi(P) \Rightarrow \boldsymbol{b} \notin \varPhi^\varPi(\varDelta)\}$. For a real number τ with $0 \leq \tau \leq 1$, the threshold adversary structure $\mathcal{T}_\tau^m$ is defined by $\mathcal{T}_\tau^m = \{A \subseteq P : |A| \leq \tau m\}$. Note that $\mathcal{T}_\tau^m$ is $\varPi$-partite for any partition $\varPi$.

3.4 Homomorphic Secret Sharing

We provide the definition of homomorphic secret sharing in the public-key setup model [27]. Homomorphic secret sharing can be defined in the multi-key setting and then lifted to the plain model [6] as described below. In addition, we here distinguish between clients holding inputs and servers performing evaluation. However, this is only for simplicity of the syntax and descriptions of schemes. In its application to MPC, we suppose that clients themselves do evaluation of a function as well as sharing of their inputs.

Definition 2 (Homomorphic Secret Sharing (HSS)). *Suppose that there are n inputs and m servers. A homomorphic secret sharing scheme consists of four PPT algorithms* $\mathsf{HSS} = (\mathsf{KGen}, \mathsf{Share}, \mathsf{Eval}, \mathsf{Dec})$*:*

- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$*: Given the security parameter 1^λ, the key generation algorithm outputs a public key* pk *and a secret key* sk*.*
- $(\mathsf{in}_j^{\langle i \rangle})_{j \in [m]} \leftarrow \mathsf{Share}(\mathsf{pk}, i, \boldsymbol{x}^{\langle i \rangle})$*: Given the public key* pk*, an index $i \in [n]$, and an input $\boldsymbol{x}^{\langle i \rangle} \in \mathcal{X}$, the sharing algorithm outputs input shares* $(\mathsf{in}_j^{\langle i \rangle})_{j \in [m]}$*.*
- $\mathsf{out}_j \leftarrow \mathsf{Eval}(\mathsf{pk}, j, f, (\mathsf{in}_j^{\langle i \rangle})_{i \in [n]})$*: Given the public key* pk*, an index $j \in [m]$, (a description of) a function $f : \mathcal{X}^n \to \mathcal{Y}$, and the shares of the j-th server* $(\mathsf{in}_j^{\langle i \rangle})_{i \in [n]}$*, the evaluation algorithm outputs an output share* out_j*.*
- $y \leftarrow \mathsf{Dec}(\mathsf{sk}, (\mathsf{out}_j)_{j \in [m]})$*: Given the secret key* sk *and output shares* $(\mathsf{out}_j)_{j \in [m]}$ *from all servers, the decoding algorithm outputs y.*

The efficiency measures $\alpha = \alpha(n, m, \log |\mathcal{X}|)$ and $\beta = \beta(n, m, \log |\mathcal{Y}|)$ are the lengths of input and output shares, respectively (omitting $\mathsf{poly}(\lambda)$ *factors). That is,* $\mathsf{in}_j^{\langle i \rangle} \in \{0,1\}^{\alpha \cdot \mathsf{poly}(\lambda)}$ *and* $\mathsf{out}_j \in \{0,1\}^{\beta \cdot \mathsf{poly}(\lambda)}$ *for any $i \in [n]$ and $j \in [m]$.*

Definition 3 (Correctness). *Let $\mathbb{F}$ be a finite field. An n-input m-server HSS scheme* HSS *is said to support ℓ parallel evaluations of degree-d polynomials over $\mathbb{F}$ if each input is a vector $\boldsymbol{x}^{\langle i \rangle} \in \mathbb{F}^\ell$ and the following holds: for any $\lambda \in \mathbb{N}$, any $n, m \in$* $\mathsf{poly}(\lambda)$*, any* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$*, any degree-$d$ polynomial $f \in \mathbb{F}[X_1, \ldots, X_n]$, any n-tuple of inputs $(\boldsymbol{x}^{\langle i \rangle})_{i \in [n]} \in (\mathbb{F}^\ell)^n$, any shares* $(\mathsf{in}_j^{\langle i \rangle})_{j \in [m]} \leftarrow \mathsf{Share}(\mathsf{pk}, i, \boldsymbol{x}^{\langle i \rangle})$ *for $i \in [n]$, and any* $\mathsf{out}_j \leftarrow \mathsf{Eval}(\mathsf{pk}, j, f, (\mathsf{in}_j^{\langle i \rangle})_{i \in [n]})$ *for $j \in [m]$, it holds that* $\Pr\left[\mathsf{Dec}(\mathsf{sk}, (\mathsf{out}_j)_{j \in [m]}) = \mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1 \rangle}, \ldots, \boldsymbol{x}^{\langle n \rangle}))\right] \geq 1 - \mathsf{negl}(\lambda)$*.*

It is sufficient for the HSS scheme to evaluate degree-d polynomials that is *homogeneous*, i.e., polynomials written as the sum of degree-d monomials. This is because we can pad any monomial of degree $d' < d$ with $d - d'$ copies of a dummy variable X_0. We set the corresponding input to $\boldsymbol{x}^{\langle 0 \rangle} = (1, \ldots, 1) \in \mathbb{F}^\ell$ and the shares of $\boldsymbol{x}^{\langle 0 \rangle}$ to some predetermined ones. For that reason, we assume in the following that a polynomial to be evaluated is homogeneous.

The security of an HSS scheme guarantees that collusion of servers in $B \in \Delta$ cannot guess inputs of clients from their input shares.

Definition 4 (Security). *Let Δ be an adversary structure on the set of servers. An n-input m-server HSS scheme* HSS *is said to satisfy Δ-privacy if for any PPT* *algorithm $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$, there exists a negligible function $\mathsf{negl}(\lambda)$ such that for any $\lambda \in \mathbb{N}$, $|\Pr[\mathsf{Security}^0_{\mathcal{A},\mathsf{HSS}} = 1] - \Pr[\mathsf{Security}^1_{\mathcal{A},\mathsf{HSS}} = 1]| < \mathsf{negl}(\lambda)$, where $\mathsf{Security}^b_{\mathcal{A},\mathsf{HSS}}$ is defined in Fig. 2 (left) for $b \in \{0,1\}$.*

We use the shorthand (n, m, ℓ, d, Δ)-HSS to refer to n-input m-server homomorphic secret sharing that supports ℓ parallel evaluations of degree-d polynomials and satisfies Δ-privacy. Note that an $(n, m, 1, d, \Delta)$-HSS scheme HSS_1 with efficiency measures α_1 and β_1 can be trivially extended to an (n, m, ℓ, d, Δ)-HSS scheme HSS_ℓ with efficiency measures $\alpha = \ell\alpha_1$ and $\beta = \ell\beta_1$ by simply running ℓ independent instances of HSS_1.

The notion of context hiding [1] assures that the output client learns nothing beyond the output of the computation.

Definition 5 (Context Hiding). *An (n, m, ℓ, d, Δ)-HSS scheme HSS is said to be context-hiding if for any PPT algorithm $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$, there exists a PPT algorithm S_{HSS} and a negligible function $\mathsf{negl}(\lambda)$ such that for any $\lambda \in \mathbb{N}$, $|\Pr[\mathsf{ContextHiding}^0_{\mathcal{A},S_{\mathsf{HSS}},\mathsf{HSS}} = 1] - \Pr[\mathsf{ContextHiding}^1_{\mathcal{A},S_{\mathsf{HSS}},\mathsf{HSS}} = 1]| < \mathsf{negl}(\lambda)$, where $\mathsf{ContextHiding}^b_{\mathcal{A},S_{\mathsf{HSS}},\mathsf{HSS}}$ is defined in Fig. 2 (right) for $b \in \{0,1\}$.*

```
$\mathsf{Security}^b_{\mathcal{A},\mathsf{HSS}}(1^\lambda)$:
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
$(\boldsymbol{x}_0, \boldsymbol{x}_1, B \in \Delta, \mathsf{state}) \leftarrow \mathcal{A}_0(\mathsf{pk})$
$(\mathsf{in}_1, \ldots, \mathsf{in}_m) \leftarrow \mathsf{Share}(\mathsf{pk}, \boldsymbol{x}_b)$
$b' \leftarrow \mathcal{A}_1(\mathsf{state}, (\mathsf{in}_j)_{j \in B})$
return $b'$
```

```
$\mathsf{ContextHiding}^b_{\mathcal{A},S_{\mathsf{HSS}},\mathsf{HSS}}(1^\lambda)$:
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
$(f, \boldsymbol{x}^{\langle 1 \rangle}, \ldots, \boldsymbol{x}^{\langle n \rangle}, \mathsf{state}) \leftarrow \mathcal{A}_0(\mathsf{pk})$
if $b = 0$ then
    $(\mathsf{in}^{\langle i \rangle}_1, \ldots, \mathsf{in}^{\langle i \rangle}_m) \leftarrow \mathsf{Share}(\mathsf{pk}, i, \boldsymbol{x}^{\langle i \rangle}),\ \forall i \in [n]$
    $y_j \leftarrow \mathsf{Eval}(\mathsf{pk}, j, f, (\mathsf{in}^{\langle i \rangle}_j)_{i \in [n]}),\ \forall j \in [m]$
else
    $(y_1, \ldots, y_m) \leftarrow S_{\mathsf{HSS}}(1^\lambda, \mathsf{pk}, f(\boldsymbol{x}^{\langle 1 \rangle}, \ldots, \boldsymbol{x}^{\langle n \rangle}))$
endif
$b' \leftarrow \mathcal{A}_1(\mathsf{state}, (y_1, \ldots, y_m))$
return $b'$
```

Fig. 2. Security and context hiding experiments for HSS

The authors of [26] introduce the notion of information-theoretic HSS with recovery information, which is an intermediate primitive for constructing HSS in the sense of Definition 2. That variant guarantees that a secret is protected against unbounded adversaries but the decoding algorithm requires auxiliary information to compute the output of a function.

Definition 6 (Information-Theoretic HSS with Recovery Information). *An (n, m, ℓ, d, Δ)-IT-HSS scheme with recovery information consists of three algorithms $\mathsf{HSS} = (\mathsf{Share}, \mathsf{Eval}, \mathsf{Dec})$:*

– $(\mathsf{in}_j^{\langle i\rangle}, \mathsf{rec}_j^{\langle i\rangle})_{j\in[m]} \leftarrow \mathsf{Share}(i, \boldsymbol{x}^{\langle i\rangle})$*: Given an input index* $i \in [n]$ *and an input* $\boldsymbol{x}^{\langle i\rangle}$*, the sharing algorithm outputs a set of input shares* $(\mathsf{in}_j^{\langle i\rangle})_{j\in[m]}$ *and recovery information* $(\mathsf{rec}_j^{\langle i\rangle})_{j\in[m]}$.

– $\mathsf{out}_j \leftarrow \mathsf{Eval}(j, f, (\mathsf{in}_j^{\langle i\rangle})_{i\in[n]})$*: Given an index* $j \in [m]$*, (a description of) a degree-*d *polynomial* f*, and the shares of the* j*-th server* $(\mathsf{in}_j^{\langle i\rangle})_{i\in[n]}$*, the evaluation algorithm outputs an output share* out_j.

– $y \leftarrow \mathsf{Dec}((\mathsf{out}_j)_{j\in[m]}, (\mathsf{rec}_j^{\langle i\rangle})_{i\in[n],j\in[m]})$*: Given output shares* $(\mathsf{out}_j)_{j\in[m]}$ *and recovery information* $(\mathsf{rec}_j^{\langle i\rangle})_{i\in[n],j\in[m]}$*, the decoding algorithm outputs* y.

The definition of correctness is the same as Definition 3 except that Dec *is allowed to take the recovery information as input. The definition of security is the same as Definition 4 except that the adversary* $\mathcal{A}$ *is unbounded.*

Application to MPC. Any (m, m, ℓ, d, Δ)-HSS scheme has a direct application to m-input MPC for an adversary structure Δ. Assume that there are m input parties each holding their private inputs $\boldsymbol{x}^{\langle i\rangle}$ and a distinguished output party. For an m-variate degree-d polynomial f, consider the following protocol:

1. The output party generates $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$ and publishes pk.
2. For $i \in [m]$, the i-th input party computes $(\mathsf{in}_j^{\langle i\rangle})_{j\in[m]} \leftarrow \mathsf{Share}(\mathsf{pk}, i, \boldsymbol{x}^{\langle i\rangle})$ and sends $\mathsf{in}_j^{\langle i\rangle}$ to the j-th input party for all $j \in [m]$.
3. For $j \in [m]$, the j-th input party computes $\mathsf{out}_j \leftarrow \mathsf{Eval}(\mathsf{pk}, j, f, (\mathsf{in}_j^{\langle i\rangle})_{i\in[n]})$ and sends it to the output party.
4. The output party outputs $y \leftarrow \mathsf{Dec}(\mathsf{sk}, (\mathsf{out}_j)_{j\in[m]})$.

The Δ-privacy of the HSS scheme guarantees that an adversary corrupting a subset X of input parties such that $X \in \Delta$ cannot guess the inputs of parties not in X. In addition, if the scheme is context-hiding, the output party obtains nothing beyond $\mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1\rangle}, \ldots, \boldsymbol{x}^{\langle n\rangle}))$. The individual communication complexity between input parties themselves is $\alpha \cdot \mathsf{poly}(\lambda)$ and each input party sends an output share of length $\beta \cdot \mathsf{poly}(\lambda)$ to the output party. Hence, the total communication complexity is at most $(\alpha m^2 + \beta m) \cdot \mathsf{poly}(\lambda)$.

Multi-Key HSS. As shown in [27], the syntax of HSS can be easily generalized to the multi-key settings by extending the evaluation algorithm so that all servers take as input all public keys of the participating input clients, the decoding algorithm takes as input all the corresponding secret keys. While the definition of security is unchanged and is required to hold for each secret key, the definitions of correctness and context hiding are extended accordingly. According to [27], multi-key HSS can be generically lifted to the plain model [6] requiring no public-key setup by simply letting input clients locally generate key pairs, adding their public keys and m-out-of-m additive shares of their secret keys to input shares, and letting servers relay the additive shares to an output client. Therefore, by instantiating it with a multi-key HSS scheme, we can lift the above MPC protocol to the plain model without increasing the order of communication complexity.

4 Adjusting the ILM Compiler to Parallel Evaluation

The ILM compiler [26] converts an IT-HSS scheme with recovery information to an HSS scheme based on HE. Although it originally assumes a single evaluation of polynomials, we can generalize the ILM compiler so that it is applicable to IT-HSS schemes supporting ℓ parallel evaluations. As shown in Section 2.1, that generalization is not trivial in that we need to consider in more detail an algebraic expression of the decoding algorithm to obtain an HSS scheme whose output share size is not proportional to ℓ.

Proposition 2. *Let* HSS_0 *be an* (n, m, ℓ, d, Δ)*-IT-HSS scheme and* HE *be an IND-CPA secure k-HE scheme. Assume that* HSS_0 *satisfies the following:*

- *An input share* $\mathsf{in}_j^{\langle i\rangle}$ *is a vector* $\boldsymbol{s}_j^{\langle i\rangle} \in \mathbb{F}^{\alpha}$.
- *Each piece of recovery information* $\mathsf{rec}_j^{\langle i\rangle}$ *is a vector* $\boldsymbol{r}_j^{\langle i\rangle} \in \mathbb{F}^{\rho}$.
- *An output share* out_j *is a vector* $\boldsymbol{y}_j \in \mathbb{F}^{\beta}$.
- $\mathsf{HSS}_0.\mathsf{Dec}((\mathsf{out}_j)_{j\in[m]}, (\mathsf{rec}_j^{\langle i\rangle})_{i\in[n],j\in[m]})$ *outputs* $\sum_{j\in[m]} \boldsymbol{C}_j(\boldsymbol{A}_j\boldsymbol{y}_j+\boldsymbol{b}_j)$, *where* $\boldsymbol{C}_j$ *is a constant* ℓ*-by-*h *matrix over* $\mathbb{F}$ *for some* h, $\boldsymbol{A}_j$ *is an* h*-by-*β *matrix whose entries are degree-k polynomials of* $(\boldsymbol{r}_j^{\langle i\rangle})_{i\in[n]}$ *over* $\mathbb{F}$, *and* $\boldsymbol{b}_j$ *is an* h*-dimensional vector whose entries are degree-k polynomials of* $(\boldsymbol{r}_j^{\langle i\rangle})_{i\in[n]}$.

Then, there exists an (n, m, ℓ, d, Δ)*-HSS scheme* HSS *with efficiency measures* $\alpha' = \alpha \log|\mathbb{F}| + \rho$ *and* $\beta' = h$.

Proof. For $j \in [m]$, write $\boldsymbol{A}_j = (\boldsymbol{p}_{j,1}, \ldots, \boldsymbol{p}_{j,h})^\top, \boldsymbol{b}_j = (q_{j,1}, \ldots, q_{j,h})^\top$, where $\boldsymbol{p}_{j,w}$ is a β-dimensional vectors of polynomials of $(\boldsymbol{r}_j^{\langle i\rangle})_{i\in[n]}$ and $q_{j,w}$ is a polynomial of $(\boldsymbol{r}_j^{\langle i\rangle})_{i\in[n]}$. In Fig. 3, we define HSS following the framework in [26].

```
$\mathsf{HSS.KGen}(1^\lambda)$:
    $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
    **return** $(\mathsf{pk}, \mathsf{sk})$

$\mathsf{HSS.Share}(\mathsf{pk}, i, \boldsymbol{x}^{\langle i\rangle})$:
    $(\boldsymbol{s}_j^{\langle i\rangle}, \boldsymbol{r}_j^{\langle i\rangle})_{j\in[m]} \leftarrow \mathsf{HSS}_0.\mathsf{Share}(i, \boldsymbol{x}^{\langle i\rangle})$
    $\boldsymbol{u}_j^{\langle i\rangle} \leftarrow \mathsf{HE.Enc}(\mathsf{pk}, \boldsymbol{r}_j^{\langle i\rangle}), \forall j \in [m]$
    $\mathsf{in}_j^{\langle i\rangle} = (\boldsymbol{s}_j^{\langle i\rangle}, \boldsymbol{u}_j^{\langle i\rangle}), \forall j \in [m]$
    **return** $(\mathsf{in}_j^{\langle i\rangle})_{j\in[m]}$

$\mathsf{HSS.Eval}(\mathsf{pk}, j, f, (\mathsf{in}_j^{\langle i\rangle})_{i\in[n]})$:
    $\boldsymbol{y}_j \leftarrow \mathsf{HSS}_0.\mathsf{Eval}(j, f, (\boldsymbol{s}_j^{\langle i\rangle})_{i\in[n]})$
    $\nu_{j,w} = \boldsymbol{p}_{j,w}^\top \boldsymbol{y}_j + q_{j,w}, \forall w \in [h]$
    $d_{j,w} \leftarrow \mathsf{HE.Eval}(\mathsf{pk}, \nu_{j,w}, (\boldsymbol{u}_j^{\langle i\rangle})_{i\in[n]}), \forall w \in [h]$
    **return** $\mathsf{out}_j = (d_{j,w})_{w\in[h]}$

$\mathsf{HSS.Dec}(\mathsf{sk}, (\mathsf{out}_j)_{j\in[m]})$:
    $\boldsymbol{d}_j = \mathsf{HE.Dec}(\mathsf{sk}, \mathsf{out}_j), \forall j \in [m]$
    **return** $\sum_{j\in[m]} \boldsymbol{C}_j \boldsymbol{d}_j$
```

Fig. 3. Compiler from HSS_0 to HSS based on HE

To see correctness, observe that $d_{j,w}$ in the evaluation algorithm of HSS is a ciphertext that decrypts to $\boldsymbol{p}_{j,w}(\boldsymbol{r}_j^{\langle 1\rangle}, \ldots, \boldsymbol{r}_j^{\langle n\rangle})^\top \boldsymbol{y}_j + q_{j,w}(\boldsymbol{r}_j^{\langle 1\rangle}, \ldots, \boldsymbol{r}_j^{\langle n\rangle})$

since each entry of $\boldsymbol{p}_{j,w}$ and $q_{j,w}$ are degree-k polynomials. Hence, it holds that $\boldsymbol{d}_j = \boldsymbol{A}_j \boldsymbol{y}_j + \boldsymbol{b}_j$ and the correctness of HSS_0 implies that $\sum_{j\in[m]} \boldsymbol{C}_j \boldsymbol{d}_j = \mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1\rangle}, \ldots, \boldsymbol{x}^{\langle n\rangle}))$. As in [26], the security of HSS follows from the information-theoretic security of HSS_0 and the IND-CPA security of HE. An input share $\mathsf{in}_j^{\langle i\rangle}$ consists of $\boldsymbol{s}_j^{\langle i\rangle} \in \mathbb{F}^\alpha$ and a ciphertext for $\boldsymbol{r}_j^{\langle i\rangle} \in \mathbb{F}^\rho$. An output share out_j consists of h ciphertexts. Therefore, the correctness of α' and β' follows. $\square$

The computational complexity of $\mathsf{HSS.Share}$ is that of $\mathsf{HSS}_0\mathsf{.Share}$ plus $m\rho \cdot \mathsf{poly}(\lambda)$. The computational complexity of $\mathsf{HSS.Eval}$ is that of $\mathsf{HSS}_0\mathsf{.Eval}$ plus $\beta(n\rho)^k h \cdot \mathsf{poly}(\lambda)$. Here, we assume that $\mathsf{HE.Eval}(f, \cdot)$ can be computed in time $|f| \cdot \mathsf{poly}(\lambda)$ for a function f whose description size is $|f|$. The computational complexity of $\mathsf{HSS.Dec}$ is $m\ell h \cdot \mathsf{poly}(\lambda)$.

Note that we can obtain a multi-key HSS scheme by replacing the HE scheme with the corresponding multi-key variant.

5 Warm-up: HSS for Threshold Adversary Structures

As a warm-up, we construct an IT-HSS scheme with recovery information for a threshold adversary structure $\mathcal{T}_\tau^m$. Let $n, m, \ell, d, k \in \mathbb{N}, \tau \in \mathbb{R}_+$, and set $t = \lfloor \tau m \rfloor \in \mathbb{Z}_+$. We fix $m+\ell$ distinct elements $\zeta_1, \ldots, \zeta_m, \eta_1, \ldots, \eta_\ell$ of $\mathbb{F}$ and set $\boldsymbol{\zeta} = (\zeta_j)_{j\in[m]}$ and $\boldsymbol{\eta} = (\eta_j)_{j\in[\ell]}$. The sharing algorithm on input $\boldsymbol{x}^{\langle i\rangle} \in \mathbb{F}^\ell$ chooses a random polynomial $\varphi^{\langle i\rangle}$ such that $\boldsymbol{x}^{\langle i\rangle} = (\varphi^{\langle i\rangle}(\eta_j))_{j\in[\ell]}$ and $\deg \varphi^{\langle i\rangle} \le t+\ell-1$. It sets input shares as $\mathsf{in}_j^{\langle i\rangle} = \varphi^{\langle i\rangle}(\zeta_j)$ and recovery information as $\mathsf{rec}_j^{\langle i\rangle} = ((\mathcal{D}^w \varphi^{\langle i\rangle})(\zeta_j))_{w\in[k]}$. The $\mathcal{T}_\tau^m$-privacy is straightforward since input shares are equivalent to shares of the packed secret sharing scheme [19], which reveal nothing about $\boldsymbol{x} \in \mathbb{F}^\ell$ as long as the number of shares is at most t.

Consider the simplest case of $f = X_1 \cdots X_d$. Then, $\mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1\rangle}, \ldots, \boldsymbol{x}^{\langle n\rangle})) = \boldsymbol{x}^{\langle 1\rangle} * \cdots * \boldsymbol{x}^{\langle d\rangle} = (g(\eta_j))_{j\in[\ell]}$, where $*$ denotes the element-wise product and $g = \varphi^{\langle 1\rangle} \cdots \varphi^{\langle d\rangle}$. For $j \in [m]$ and $w \in [0..k]$, the chain rule implies

$$\mathcal{D}^w g(\zeta_j) = \sum_{\substack{\boldsymbol{e}\in\mathbb{Z}_+^d:\\ e_1+\cdots+e_d=w}} \frac{w!}{e_1!\cdots e_d!} \prod_{\substack{\kappa\in[d]:\\ e_\kappa=0}} \varphi^{\langle\kappa\rangle}(\zeta_j) \prod_{\substack{\kappa\in[d]:\\ e_\kappa>0}} (\mathcal{D}^{e_\kappa}\varphi^{\langle\kappa\rangle})(\zeta_j). \tag{1}$$

The j-th server computes and outputs $\prod_{\kappa:e_\kappa=0} \varphi^{\langle\kappa\rangle}(\zeta_j)$ for all $w, \boldsymbol{e}$ with $\sum e_\kappa = w$ from its input shares. Then, the output client computes $\prod_{\kappa:e_\kappa>0}(\mathcal{D}^{e_\kappa}\varphi^{\langle\kappa\rangle})(\zeta_j)$ for all $j, w, \boldsymbol{e}$ from recovery information. Combining them with output shares, he obtains $\boldsymbol{z}_j = (\mathcal{D}^w g(\zeta_j))_{w\in[0..k]}$ and computes $\mathsf{Hermite}_{\boldsymbol{\zeta},\boldsymbol{\eta}}((\boldsymbol{z}_j)_{j\in[m]}) = (g(\eta_j))_{j\in[\ell]}$ if $d(t+\ell-1) < (k+1)m$.

To apply the ILM compiler, we have to check whether an output of the decoding algorithm has the form specified in Proposition 2. Indeed, each piece of recovery information is a vector $\boldsymbol{r}_j^{\langle i\rangle} \in \mathbb{F}^\rho$ for $\rho = k$ and an output share is a vector $\boldsymbol{y}_j$ whose dimension β is the number of all the pairs $(w, \boldsymbol{e})$ such that $w \in [0..k]$ and $\sum_\kappa e_\kappa = w$, i.e., $\beta \le k2^{k+d}$. In view of Eq. (1), $\mathcal{D}^w g(\zeta_j)$

is a degree-1 polynomial of $\prod_{\kappa:e_\kappa=0}\varphi^{\langle\kappa\rangle}(\zeta_j)$ for all $\boldsymbol{e}$ such that $\sum_\kappa e_\kappa = w$. Furthermore, the coefficient of each $\prod_{\kappa:e_\kappa=0}\varphi^{\langle\kappa\rangle}(\zeta_j)$ is a degree-k polynomial of $(\boldsymbol{r}_j^{\langle i\rangle})_{i\in[n]}$ since $|\{\kappa\in[d] : e_\kappa > 0\}| \leq w \leq k$. Therefore, $\boldsymbol{z}_j$ can be expressed as $\boldsymbol{A}_j\boldsymbol{y}_j + \boldsymbol{b}_j$ using a $(k+1)$-by-β matrix $\boldsymbol{A}_j$ and $(k+1)$-dimensional vector $\boldsymbol{b}_j$ whose entries are degree-k polynomials of $(\boldsymbol{r}_j^{\langle i\rangle})_{i\in[n]}$. Since $\mathsf{Hermite}_{\boldsymbol{\zeta},\boldsymbol{\eta}}((\boldsymbol{z}_j)_{j\in[m]})$ is linear in $(\boldsymbol{z}_j)_{j\in[m]}$, the output of Dec can be expressed as $\sum_{j\in[m]}\boldsymbol{C}_j(\boldsymbol{A}_j\boldsymbol{y}_j+\boldsymbol{b}_j)$ using some constant ℓ-by-$(k+1)$ matrices $\boldsymbol{C}_j$. In summary, we have the following theorem. Since it is a special case of Theorem 2, we omit the formal proof here.

Theorem 1. *Let $n, m, \ell, d, k \in \mathbb{N}$. Let $\tau \in \mathbb{R}_+$ and $t = \lfloor \tau m \rfloor \in \mathbb{Z}_+$. Let $\mathbb{F}$ be a prime field such that $|\mathbb{F}| \geq \max\{m+\ell, k+1\}$. Assume that $(k+1)m - dt > d(\ell-1)$. Then, there exists an $(n, m, \ell, d, \mathcal{T}_\tau^m)$-IT-HSS scheme HSS_0 with recovery information that satisfies the properties in Proposition 2 for $\alpha = 1$, $\rho = k$, $\beta = n^d k 2^{k+d}$, and $h = k+1$.*

By applying the ILM compiler in Proposition 2, assuming a k-HE scheme HE, we obtain an $(n, m, \ell, d, \mathcal{T}_\tau^m)$-HSS scheme HSS with efficiency measures $\alpha = \log|\mathbb{F}| + k$ and $\beta = k+1$. To make computational complexity polynomial in λ, we assume that $n, m \in \mathsf{poly}(\lambda)$ and $k, d \in O(1)$ since the most costly part is the evaluation algorithm, whose time complexity is $n^{d+k}k^{k+1}2^{k+d}\cdot\mathsf{poly}(\lambda)$.

Since the ILM compiler does not provide context hiding in general, we add re-randomizing procedures to the sharing and evaluation algorithms of the compiled scheme HSS. Specifically, an output share of HSS computed by the j-th server consists of $k+1$ ciphertexts $(d_{j,w})_{w\in[0..k]}$ of HE each of which decrypts to $\mathcal{D}^w g(\zeta_j)$ for a polynomial g of degree less than $(k+1)m$ such that $(g(\eta_j))_{j\in[\ell]} = \mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1\rangle}, \ldots, \boldsymbol{x}^{\langle n\rangle}))$. We modify the sharing algorithm so that the first client (on behalf of all clients) additionally generates a polynomial θ such that $\deg(\theta) < (k+1)m$ and $(\theta(\eta_j))_{j\in[\ell]} = \boldsymbol{0}$, and sends $(\mathcal{D}^w\theta(\zeta_j))_{w\in[0..k]}$ to the j-th server for each $j\in[m]$. Accordingly, the j-th server executes the evaluation algorithm of HSS to obtain $(d_{j,w})_{w\in[0..k]}$ and then using $(\mathcal{D}^w\theta(\zeta_j))_{w\in[0..k]}$, outputs $k+1$ ciphertexts $(d'_{j,w})_{w\in[0..k]}$ each of which decrypts to $(\mathcal{D}^w g'(\zeta_j))_{w\in[0..k]}$, where $g' = g + \theta$. In this modified scheme HSS', the output shares are ciphertexts of values and derivatives of g', which is uniformly distributed over the set $\{\widetilde{g} \in \mathbb{F}[X] : \deg(\widetilde{g}) < (k+1)m \wedge (\widetilde{g}(\eta_j))_{j\in[\ell]} = \mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1\rangle}, \ldots, \boldsymbol{x}^{\langle n\rangle}))\}$. Combined with the circuit-privacy of HE, the output shares can be simulated only from $\mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1\rangle}, \ldots, \boldsymbol{x}^{\langle n\rangle}))$. In summary, we have the following result. Again, we omit the formal proof here since Corollary 1 is a special case of Corollary 2.

Corollary 1. *Using the notations in Theorem 1, assume that $n, m \in \mathsf{poly}(\lambda)$ and $k, d \in O(1)$. Assuming a k-HE scheme HE, there exists an $(n, m, \ell, d, \mathcal{T}_\tau^m)$-HSS scheme HSS with efficiency measures $\alpha = \log|\mathbb{F}|+k$ and $\beta = k+1$. Furthermore, if HE is circuit-private, there exists a context-hiding $(n, m, \ell, d, \mathcal{T}_\tau^m)$-HSS scheme HSS' with efficiency measures $\alpha = (k+2)\log|\mathbb{F}| + k = O(\log(m+\ell))$ and $\beta = k+1 = O(1)$.*

To lift the above result to the plain model, it is sufficient to replace the HE scheme with the corresponding multi-key variant. We can instantiate the m-

input MPC protocol in Section 3.4 with the threshold HSS scheme in Corollary 1 and then the communication complexity is $O(m^2 \log(m+\ell))$.

6 HSS for Multipartite Adversary Structures

We show a construction of IT-HSS schemes with recovery information for multipartite adversary structures.

Theorem 2. *Let $n, m, \ell, d, k, L \in \mathbb{N}$. Let Π be an L-partition of the set P of m servers. Let Δ be a Π-partite adversary structure on P and $\max \Phi^{\Pi}(\Delta) = \{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_N\}$. Let $\mathbb{F}$ be a prime field such that $|\mathbb{F}| \geq \max\{m+\ell, k+1\}$. Assume that Δ satisfies the condition*

$$\forall (u_1, \ldots, u_d) \in [N]^d : |(k+1)\Phi^{\Pi}(P) - (\boldsymbol{a}_{u_1} + \cdots + \boldsymbol{a}_{u_d})|_\infty > d(\ell - 1). \quad (2)$$

Then, the scheme HSS_0 described in Fig. 4 is an (n, m, ℓ, d, Δ)-IT-HSS scheme with recovery information that satisfies the properties in Proposition 2 for $\alpha = N$, $\rho = Nk$, $\beta = n^d N^d k 2^{k+d}$, and $h = k+1$.

Proof. Since $|(k+1)\Phi^{\Pi}(P) - (\boldsymbol{a}_{u_1} + \cdots + \boldsymbol{a}_{u_d})|_\infty > d(\ell-1)$ for any $(u_1, \ldots, u_d) \in [N]^d$, there is a map $\psi : [N]^d \to [L]$ such that $(k+1)|P_v| > \boldsymbol{a}_{u_1}(v) + \cdots + \boldsymbol{a}_{u_d}(v) + d(\ell-1)$ for any $(u_1, \ldots, u_d) \in [N]^d$ and $v = \psi(u_1, \ldots, u_d)$.

To see correctness, let $f(X_1, \ldots, X_n) = \sum_{\boldsymbol{i} \in [n]^d} c_{\boldsymbol{i}} X_{i_1} \cdots X_{i_d}$. Using the notations in Fig. 4, for any input $\boldsymbol{x}^{\langle 1 \rangle}, \ldots, \boldsymbol{x}^{\langle n \rangle} \in \mathbb{F}^\ell$, it holds that

$$\begin{aligned}
\mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1 \rangle}, \ldots, \boldsymbol{x}^{\langle n \rangle})) &= \sum_{\boldsymbol{u} \in [N]^d} \sum_{\boldsymbol{i} \in [n]^d} c_{\boldsymbol{i}} \boldsymbol{x}_{u_1}^{\langle i_1 \rangle} * \cdots * \boldsymbol{x}_{u_d}^{\langle i_d \rangle} \\
&= \sum_{v=1}^{L} \sum_{\boldsymbol{u} \in \psi^{-1}(v)} \sum_{\boldsymbol{i} \in [n]^d} c_{\boldsymbol{i}} \boldsymbol{x}_{u_1}^{\langle i_1 \rangle} * \cdots * \boldsymbol{x}_{u_d}^{\langle i_d \rangle} \\
&= \sum_{v=1}^{L} (g_v(\eta_j))_{j \in [\ell]},
\end{aligned}$$

where $*$ denotes the element-wise product and $g_v = \sum_{\boldsymbol{u} \in \psi^{-1}(v)} \sum_{\boldsymbol{i} \in [n]^d} c_{\boldsymbol{i}} \varphi_{u_1,v}^{\langle i_1 \rangle} \cdots \varphi_{u_d,v}^{\langle i_d \rangle}$. It holds that

$$\mathcal{D}^w g_v = \sum_{\boldsymbol{u} \in \psi^{-1}(v)} \sum_{\boldsymbol{i} \in [n]^d} \sum_{\substack{\boldsymbol{e} \in \mathbb{Z}_+^d: \\ e_1 + \cdots + e_d = w}} \frac{w!}{e_1! \cdots e_d!} c_{\boldsymbol{i}} (\mathcal{D}^{e_1} \varphi_{u_1,v}^{\langle i_1 \rangle}) \cdots (\mathcal{D}^{e_d} \varphi_{u_d,v}^{\langle i_d \rangle})$$

and hence $y_{j,w} = (\mathcal{D}^w g_{v_j})(\zeta_j)$ for any $j \in [m]$ and $w \in [0..k]$. Since the degree of g_v is at most $\max_{\boldsymbol{u} \in \psi^{-1}(v)} \{\boldsymbol{a}_{u_1}(v) + \cdots + \boldsymbol{a}_{u_d}(v) + d(\ell-1)\} < (k+1)|P_v|$, we have $\mathsf{Hermite}_{\boldsymbol{\zeta}_v, \boldsymbol{\eta}}((\boldsymbol{z}_j)_{j \in P_v}) = (g_v(\eta_j))_{j \in [\ell]}$. Therefore, $\sum_{v \in [L]} \mathsf{Hermite}_{\boldsymbol{\zeta}_v, \boldsymbol{\eta}}((\boldsymbol{z}_j)_{j \in P_v}) = \mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1 \rangle}, \ldots, \boldsymbol{x}^{\langle n \rangle}))$.

Notations.

- Parameters $n, m, \ell, d, k, L \in \mathbb{N}$.
- An L-partition $\Pi = (P_1, \ldots, P_L)$ of the set of m servers.
- A Π-partite adversary structure Δ and $\max \Phi^{\Pi}(\Delta) = \{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_N\}$.
- A prime field $\mathbb{F}$ such that $|\mathbb{F}| \geq \max\{m + \ell, k + 1\}$.
- $m + \ell$ distinct elements $\zeta_1, \ldots, \zeta_m, \eta_1, \ldots, \eta_\ell \in \mathbb{F}$, $\boldsymbol{\zeta}_v = (\zeta_j)_{j \in P_v}$ for $v \in [L]$, and $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_\ell)$.
- A map $\psi : [N]^d \to [L]$ such that $(k+1)|P_v| > \boldsymbol{a}_{u_1}(v) + \cdots + \boldsymbol{a}_{u_d}(v) + d(\ell - 1)$ for any $(u_1, \ldots, u_d) \in [N]^d$ and $v = \psi(u_1, \ldots, u_d)$.
- For $j \in [m]$, a set

$$S_j = \left\{ (\boldsymbol{i}, \boldsymbol{u}, w, \boldsymbol{e}) : \begin{array}{c} \boldsymbol{i} = (i_1, \ldots, i_d) \in [n]^d, \ \boldsymbol{u} = (u_1, \ldots, u_d) \in \psi^{-1}(v_j), \\ w \in [0..k], \ \boldsymbol{e} = (e_1, \ldots, e_d) \in \mathbb{Z}_+^d \text{ s.t. } e_1 + \cdots + e_d = w \end{array} \right\}.$$

$\mathsf{Share}(i, \boldsymbol{x}^{\langle i \rangle})$**.** Given an index $i \in [n]$ and an input $\boldsymbol{x}^{\langle i \rangle} \in \mathbb{F}^\ell$:

1. Choose $\boldsymbol{x}_u^{\langle i \rangle} \in \mathbb{F}^\ell$, $u \in [N]$ at random such that $\boldsymbol{x}^{\langle i \rangle} = \sum_{u \in [N]} \boldsymbol{x}_u^{\langle i \rangle}$.
2. For each $u \in [N]$ and $v \in [L]$, choose $\varphi_{u,v}^{\langle i \rangle} \in \mathbb{F}[X]$ at random such that $\boldsymbol{x}_u^{\langle i \rangle} = (\varphi_{u,v}^{\langle i \rangle}(\eta_j))_{j \in [\ell]}$ and $\deg(\varphi_{u,v}^{\langle i \rangle}) \leq \boldsymbol{a}_u(v) + \ell - 1$.
3. For each $j \in [m]$, set
$$\mathsf{in}_j^{\langle i \rangle} = (\varphi_{u,v_j}^{\langle i \rangle}(\zeta_j))_{u \in [N]} \text{ and } \mathsf{rec}_j^{\langle i \rangle} = ((\mathcal{D}\varphi_{u,v_j}^{\langle i \rangle})(\zeta_j), \ldots, (\mathcal{D}^k \varphi_{u,v_j}^{\langle i \rangle})(\zeta_j))_{u \in [N]},$$
where $v_j \in [L]$ is the unique index such that $j \in P_{v_j}$.
4. Output $(\mathsf{in}_j^{\langle i \rangle}, \mathsf{rec}_j^{\langle i \rangle})_{j \in [m]}$.

$\mathsf{Eval}(j, f, (\mathsf{in}_j^{\langle i \rangle})_{i \in [n]})$**.** Given an index $j \in [m]$, a polynomial $f = \sum_{\boldsymbol{i} \in [n]^d} c_{\boldsymbol{i}} X_{i_1} \cdots X_{i_d}$ where $c_{\boldsymbol{i}} \in \mathbb{F}$, and input shares $(\mathsf{in}_j^{\langle i \rangle})_{i \in [n]}$, output

$$\mathsf{out}_j = \left(c_{\boldsymbol{i}} \prod_{\kappa : e_\kappa = 0} \varphi_{u_\kappa, v_j}^{\langle i_\kappa \rangle}(\zeta_j) : (\boldsymbol{i}, \boldsymbol{u}, w, \boldsymbol{e}) \in S_j \right).$$

$\mathsf{Dec}((\mathsf{out}_j)_{j \in [m]}, (\mathsf{rec}_j^{\langle i \rangle})_{i \in [n], j \in [m]})$**.** Given output shares $(\mathsf{out}_j)_{j \in [m]}$ and recovery information $(\mathsf{rec}_j^{\langle i \rangle})_{i \in [n], j \in [m]}$:

1. For each $j \in [m]$ and $w \in [0..k]$, compute
$$y_{j,w} := \sum_{\substack{\boldsymbol{i}, \boldsymbol{u}, \boldsymbol{e}: \\ (\boldsymbol{i}, \boldsymbol{u}, w, \boldsymbol{e}) \in S_j}} \frac{w!}{e_1! \cdots e_d!} \left(c_{\boldsymbol{i}} \prod_{\kappa : e_\kappa = 0} \varphi_{u_\kappa, v_j}^{\langle i_\kappa \rangle}(\zeta_j) \right) \left(\prod_{\kappa : e_\kappa > 0} (\mathcal{D}^{e_\kappa} \varphi_{u_\kappa, v_j}^{\langle i_\kappa \rangle})(\zeta_j) \right).$$
2. Letting $\boldsymbol{z}_j = (y_{j,w})_{w \in [0..k]}$, output $\sum_{v \in [L]} \mathsf{Hermite}_{\boldsymbol{\zeta}_v, \boldsymbol{\eta}}((\boldsymbol{z}_j)_{j \in P_v})$.

Fig. 4. An (n, m, ℓ, d, Δ)-IT-HSS scheme with recovery information for an L-partite adversary structure Δ

An input share is an N-dimensional vector, each piece of recovery information is an Nk-dimensional vector $\boldsymbol{r}_j^{\langle i\rangle}$, and an output share out_j is an $|S_j|$-dimensional vector $\boldsymbol{y}_j$. It follows from the definition of S_j that $|S_j| \le n^d \cdot N^d \cdot k \cdot 2^{k+d} =: \beta$. Each $y_{j,w}$ computed by Dec is a degree-1 polynomial with respect to out_j. The degree of $y_{j,w}$ with respect to $(\mathsf{rec}_j^{\langle i\rangle})_{i\in[n]}$ is at most the maximum of $|\{\kappa \in [d] : e_\kappa > 0\}|$ over all $\boldsymbol{e} = (e_1, \dots, e_d) \in \mathbb{Z}_+^d$ such that $e_1 + \cdots + e_d = w$ and hence is at most $w \le k$. Therefore, $\boldsymbol{z}_j = (y_{j,w})_{w\in[0..k]}$ can be expressed as $\boldsymbol{A}_j\boldsymbol{y}_j + \boldsymbol{b}_j$ using a $(k+1)$-by-β matrix $\boldsymbol{A}_j$ and $(k+1)$-dimensional vector $\boldsymbol{b}_j$ whose entries are degree-k polynomials of $(\boldsymbol{r}_j^{\langle i\rangle})_{i\in[n]}$. Since $\sum_{v\in[L]} \mathsf{Hermite}_{\boldsymbol{\zeta}_v,\boldsymbol{\eta}}((\boldsymbol{z}_j)_{j\in P_v})$ is linear with respect to $(\boldsymbol{z}_j)_{j\in[m]}$, the output of Dec can be expressed as $\sum_{j\in[m]} \boldsymbol{C}_j(\boldsymbol{A}_j\boldsymbol{y}_j + \boldsymbol{b}_j)$ using some constant ℓ-by-$(k+1)$ matrices $\boldsymbol{C}_j$.

To see Δ-privacy, we show that for any $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{F}^\ell$ and $B \in \Delta$, the distributions of input shares $(\mathsf{in}_j)_{j\in B}$ for $\boldsymbol{x}$ and $(\mathsf{in}'_j)_{j\in B}$ for $\boldsymbol{x}'$ are identical. Since $\Phi^\Pi(B) \preceq \boldsymbol{a}_i$ for some $i \in [N]$, there exist L polynomials $\theta_1, \dots, \theta_L \in \mathbb{F}[X]$ such that for all $v \in [L]$, $\deg(\theta_v) \le \boldsymbol{a}_i(v) + \ell - 1$, $(\theta_v(\eta_j))_{j\in[\ell]} = \boldsymbol{x}' - \boldsymbol{x}$, and $\theta_v(\zeta_j) = 0$ for any $j \in P_v \cap B$. We have a bijection between randomness used by Share on input $\boldsymbol{x}$ and randomness used by Share on input $\boldsymbol{x}'$ such that the shares of B are the same under this bijection. Indeed, we map any random polynomials $(\varphi_{u,v})_{u\in[N],v\in[L]}$ generated by Share on input $\boldsymbol{x}$ to $(\varphi'_{u,v})_{u\in[N],v\in[L]}$ where $\varphi'_{u,v} = \varphi_{u,v} + \theta_v$ if $u = i$ and $v \in [L]$, and otherwise $\varphi'_{u,v} = \varphi_{u,v}$. Then, $(\varphi'_{u,v})_{u\in[N],v\in[L]}$ provide consistent shares for $\boldsymbol{x}'$ and the shares of B resulting from $(\varphi'_{u,v})_{u\in[N],v\in[L]}$ are the same as the ones resulting from $(\varphi_{u,v})_{u\in[N],v\in[L]}$. □

By applying the ILM compiler in Proposition 2, we obtain an HSS scheme for multipartite adversary structures. To guarantee context hiding, we have to add certain re-randomizing procedures.

Corollary 2. *Using the notations in Theorem 2, assume that $n, m \in \mathsf{poly}(\lambda)$, $k, d \in O(1)$, and $N \in \mathsf{poly}(\lambda)$. Assuming a (resp. multi-key) k-HE scheme HE, there exists an (n, m, ℓ, d, Δ)-HSS scheme HSS (resp. in the plain model) with efficiency measures $\alpha = N \log|\mathbb{F}| + Nk$ and $\beta = k + 1$. Furthermore, if HE satisfies circuit privacy, there exists a context-hiding (n, m, ℓ, d, Δ)-HSS scheme HSS' (resp. in the plain model) with efficiency measures $\alpha = (N+k+1)\log|\mathbb{F}| + Nk = O(N\log(m+\ell))$ and $\beta = k + 1 = O(1)$.*

Proof. In view of Proposition 2 and Theorem 2, we obtain an (n, m, ℓ, d, Δ)-HSS scheme HSS with efficiency measures $\alpha = N\log|\mathbb{F}| + Nk$ and $\beta = k + 1$.

We make HSS context-hiding by adding re-randomizing procedures to the sharing and evaluation algorithms. Using the notations in the proof of Theorem 2, we can see that in HSS, the sharing algorithm executed by the i-th client outputs ciphertexts $\boldsymbol{c}_j^{\langle i\rangle} = (\mathsf{HE.Enc}(\mathsf{pk}, \mathcal{D}^w\varphi_{u,v_j}^{\langle i\rangle}(\zeta_j)))_{u\in[N],w\in[0..k]}$ for $j \in [m]$. The evaluation algorithm executed by the j-th server outputs $k+1$ ciphertexts $(d_{j,w})_{w\in[0..k]}$ each of which decrypts to $y_{j,w} = (\mathcal{D}^w g_{v_j})(\zeta_j)$ for $w \in [0..k]$, where g_v is a polynomial of degree less than $(k+1)|P_v|$. Note that there are degree-

k polynomials $\nu_{j,w}$ (whose coefficients depend on $(\varphi^{\langle i\rangle}_{u,v_j}(\zeta_j))_{u\in[N]}$) such that $d_{j,w} = \mathsf{HE.Eval}(\mathsf{pk}, \nu_{j,w}, (\boldsymbol{c}^{\langle 1\rangle}_j, \ldots, \boldsymbol{c}^{\langle n\rangle}_j))$ (see Fig. 3).

We fix any client, say, $i = 1$. If the input index is $i = 1$, we let the sharing algorithm additionally generate random polynomials $(\theta_v)_{v\in[L]}$ such that $\deg(\theta_v) < (k+1)|P_v|$ for all $v \in [L]$ and $\sum_{v\in[L]}(\theta_v(\eta_j))_{j\in[\ell]} = \mathbf{0}$. Then, it sends $k+1$ field elements $y'_{j,w} = (\mathcal{D}^w\theta_{v_j})(\zeta_j)$ for $w \in [0..k]$ to the j-th server in addition to an input share $\mathsf{in}^{\langle 1\rangle}_j$ of HSS. We do not modify the procedures for the other clients. Accordingly, the size of input shares are now $\alpha = (N+k+1)\log|\mathbb{F}| + Nk$. When executing the evaluation algorithm of HSS, the j-th server outputs $k+1$ ciphertexts $(d'_{j,w})_{w\in[0..k]}$ each of which decrypts to $y_{j,w} + y'_{j,w} = (\mathcal{D}^w g'_{v_j})(\zeta_j)$ where $g'_v = g_v + \theta_v$. More precisely, the j-th server computes $d'_{j,w} \leftarrow \mathsf{HE.Eval}(\mathsf{pk}, \nu_{j,w}(\cdot) + y'_{j,w}, (\boldsymbol{c}^{\langle 1\rangle}_j, \ldots, \boldsymbol{c}^{\langle n\rangle}_j))$ for all $w \in [0..k]$.

In this modified HSS scheme HSS', the output client receives $k+1$ ciphertexts $(d'_{j,w})_{w\in[0..k]}$ from the j-th server. Now, $d'_{j,w}$ decrypts to $(\mathcal{D}^w g'_{v_j})(\zeta_j)$ and the polynomials g'_v are uniformly random under the constraints $\sum_{v\in[L]}(g'_v(\eta_j))_{j\in[\ell]} = \mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1\rangle}, \ldots, \boldsymbol{x}^{\langle n\rangle}))$ and $\deg g'_v < (k+1)|P_v|$. Due to the circuit privacy of HE, the output shares can be simulated from $\mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1\rangle}, \ldots, \boldsymbol{x}^{\langle n\rangle}))$. Precisely, let S_{HE} be the simulator for the circuit privacy of HE. Given $\mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1\rangle}, \ldots, \boldsymbol{x}^{\langle n\rangle}))$, the simulator $S_{\mathsf{HSS}'}$ randomly chooses polynomials $\widetilde{g}_v$ for $v \in [L]$ such that $\sum_{v\in[L]}(\widetilde{g}'_v(\eta_j))_{j\in[\ell]} = \mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1\rangle}, \ldots, \boldsymbol{x}^{\langle n\rangle}))$ and $\deg\widetilde{g}_v < (k+1)|P_v|$ for all $v \in [L]$, and then computes $\widetilde{d}_{j,w} \leftarrow S_{\mathsf{HE}}(1^\lambda, \mathsf{pk}, (\mathcal{D}^w\widetilde{g}_{v_j})(\zeta_j))$ for all $j \in [L]$ and $w \in [0..k]$. It finally outputs $((\widetilde{d}_{1,w})_{w\in[0..k]}, \ldots, (\widetilde{d}_{m,w})_{w\in[0..k]})$.

We analyze the distribution of the output of $S_{\mathsf{HSS}'}$. Let $j \in [m]$, $w \in [0..k]$, and $r_{j,w}$ be any fixed field element. The circuit privacy of HE implies that

$$\mathsf{HE.Eval}(\mathsf{pk}, \nu_{j,w}(\cdot) + r_{j,w}, (\boldsymbol{c}^{\langle 1\rangle}_j, \ldots, \boldsymbol{c}^{\langle n\rangle}_j)) \approx S_{\mathsf{HE}}(1^\lambda, \mathsf{pk}, y_{j,w} + r_{j,w}),$$

where $\approx$ denotes statistical indistinguishability. Then, we have that

$$\begin{aligned}&(\mathsf{HE.Eval}(\mathsf{pk}, \nu_{j,w}(\cdot) + r_{j,w}, (\boldsymbol{c}^{\langle 1\rangle}_j, \ldots, \boldsymbol{c}^{\langle n\rangle}_j)))_{w\in[0..k],j\in[m]}\\ &\quad\approx (S_{\mathsf{HE}}(1^\lambda, \mathsf{pk}, y_{j,w} + r_{j,w}))_{w\in[0..k],j\in[m]}.\end{aligned} \tag{3}$$

Let V be the set from which $(\theta_v)_{v\in[L]}$ is sampled. Note that V is a linear space over $\mathbb{F}$. Since Eq. (3) holds for any fixed elements $r_{j,w}$, we can apply it to $r_{j,w} = y'_{j,w} = \mathcal{D}^w\theta_{v_j}(\zeta_j)$ where $(\theta_v)_{v\in[L]} \leftarrow_\$ V$, and obtain that

$$\begin{aligned}(d'_{j,w})_{w\in[0..k],j\in[m]} &= (\mathsf{HE.Eval}(\mathsf{pk}, \nu_{j,w}(\cdot) + y'_{j,w}, (\boldsymbol{c}^{\langle 1\rangle}_j, \ldots, \boldsymbol{c}^{\langle n\rangle}_j)))_{w\in[0..k],j\in[m]}\\ &\approx (S_{\mathsf{HE}}(1^\lambda, \mathsf{pk}, y_{j,w} + y'_{j,w}))_{w\in[0..k],j\in[m]}.\end{aligned} \tag{4}$$

The distribution of $(g'_v)_{v\in[L]}$ is the uniform distribution over an affine space $(g_v)_{v\in[L]}+V := \{(g_v+\theta_v)_{v\in[L]} : (\theta_v)_{v\in[L]} \in V\}$, which is the same as the distribution of $(\widetilde{g}_v)_{v\in[L]}$. Furthermore, the differential operator and the evaluation map are both linear maps over $\mathbb{F}$. Therefore, the distribution of $(\mathcal{D}^w g'_{v_j}(\zeta_j))_{w\in[0..k],j\in[m]}$

induced by $(\theta_v)_{v\in[L]} \leftarrow_{\$} V$ is identical to that of $(\mathcal{D}^w \widetilde{g}_{v_j}(\zeta_j))_{w\in[0..k],j\in[m]}$ induced by $(\widetilde{g}_v)_{v\in[L]} \leftarrow_{\$} (g_1,\dots,g_L)+V$. Combined with Eq. (4), we obtain that

$$\begin{aligned}(d'_{j,w})_{w\in[0..k],j\in[m]} &\approx (S_{\mathsf{HE}}(1^\lambda,\mathsf{pk},\mathcal{D}^w g'_{v_j}(\zeta_j)))_{w\in[0..k],j\in[m]}\\ &= (S_{\mathsf{HE}}(1^\lambda,\mathsf{pk},\mathcal{D}^w \widetilde{g}_{v_j}(\zeta_j)))_{w\in[0..k],j\in[m]}\\ &= S_{\mathsf{HSS}'}(1^\lambda,\mathsf{pk},\mathcal{P}_\ell(f,(\boldsymbol{x}^{\langle 1\rangle},\dots,\boldsymbol{x}^{\langle n\rangle}))).\end{aligned}$$

We show that the computational complexity of HSS' is polynomial in the security parameter λ. The most costly step in the sharing algorithm of HSS_0 is sampling random polynomials $\varphi^{\langle i\rangle}_{u,v}$, which can be done in polynomial time in m, N, ℓ by pre-computing and publishing a basis for the linear space

$$\mathcal{L}_{u,v} := \{\varphi_{u,v} \in \mathbb{F}[X] : \deg(\varphi_{u,v}) \le \boldsymbol{a}_u(v)+\ell-1,\ (\varphi_{u,v}(\eta_j))_{j\in[\ell]} = \mathbf{0}\}$$

for each $u\in[N]$ and $v\in[L]$. We have that $\ell = O(m)$ due to the condition (2) and hence the time complexity of that step is polynomial in λ. Since the additional re-randomizing steps of HSS' can also be done in time $\mathsf{poly}(m,N,\ell,\lambda)$, the time complexity of $\mathsf{HSS}'.\mathsf{Share}$ is still polynomial in λ. Next, $\mathsf{HSS}_0.\mathsf{Eval}$ computes $\beta = O(n^d N^d)$ products and $\mathsf{HE.Eval}$ can be executed by $\mathsf{HSS}'.\mathsf{Eval}$ in time $\beta(n\rho)^k \cdot \mathsf{poly}(\lambda) = O(n^{d+k}N^{d+k})\cdot\mathsf{poly}(\lambda)$. The time complexity of $\mathsf{HSS}'.\mathsf{Eval}$ is therefore also polynomial in λ. Finally, in view of Proposition 2, $\mathsf{HSS}'.\mathsf{Dec}$ can be done in time $m\ell(k+1)\cdot\mathsf{poly}(\lambda) = \mathsf{poly}(\lambda)$.

We can lift the above scheme to the plain model by replacing the HE scheme with the corresponding multi-key variant. $\square$

Theorem 1 and Corollary 1 can be recovered by setting $L=1$ and $\mathit{\Delta} = \mathcal{T}^m_\tau$, in which $\max \mathit{\Phi}^{\mathit{\Pi}}(\mathit{\Delta}) = \{\lfloor \tau m\rfloor\} \subseteq \mathbb{Z}_+$.

In Table 1 in Section 1.1, we have defined $\epsilon_{k,d}(\mathit{\Delta}) := \min\{|(k+1)\mathit{\Phi}^{\mathit{\Pi}}(P) - (\boldsymbol{a}_{u_1}+\cdots+\boldsymbol{a}_{u_d})|_\infty : \forall (u_1,\dots,u_d)\in[N]^d\}$. It can be seen that the condition $\epsilon_{0,d}(\mathit{\Delta}) > 0$ is equivalent to the Q_d property [2], namely there are no d sets in $\mathit{\Delta}$ whose union covers the entire set P. The authors of [17] construct an information-theoretic HSS scheme (without recovery information) tolerating a multipartite Q_d-adversary structure. Since $\epsilon_{k,d}(\mathit{\Delta}) > \epsilon_{0,d}(\mathit{\Delta})$ if $k\ge 1$, our scheme in the particular case of $\ell = 1$ can be viewed as a computational variant of the scheme [17] that tolerates a wider class of Q_d-adversary structures.

We can instantiate the m-input MPC protocol in Section 3.4 with the HSS scheme in Corollary 2 and then the communication complexity is $O(Nm^2\log(m+\ell))$, where ℓ is the number of parallel evaluations. Since $N = O(m^L)$ and L is practically chosen as $L\in\{2,3\}$ [18], it is much smaller than the description size $O(m^d)$ of a polynomial to compute. Furthermore, as shown in Section 8, there is still an interesting class of adversary structures even in the case of $N = O(1)$.

7 HSS for General Adversary Structures

We show a construction of IT-HSS schemes with recovery information for general adversary structures.

Theorem 3. *Let $n, m, \ell, d, k \in \mathbb{N}$. Let Δ be an adversary structure on the set P of m servers and $\{B_1, \ldots, B_N\}$ be all maximal sets of Δ. Let $\mathbb{F}$ be a prime field such that $|\mathbb{F}| \geq \max\{m+\ell, k+1\}$. Assume that Δ satisfies the condition*

$$\forall (u_1, \ldots, u_d) \in [N]^d : |(k+1)\mathbf{1}_m - (\boldsymbol{a}_{u_1} + \cdots + \boldsymbol{a}_{u_d})|_+ > (d+1)(\ell - 1), \quad (5)$$

where $\boldsymbol{a}_u \in \mathbb{Z}^m$ is a vector in which $\boldsymbol{a}_u(j) = 1$ if $j \in B_u$ and $\boldsymbol{a}_u(j) = 0$ otherwise. Then, the scheme HSS_0 described in Fig. 5 and 6 is an (n, m, ℓ, d, Δ)-IT-HSS scheme with recovery information that satisfies the properties in Proposition 2 for $\alpha = O(N)$, $\rho = O(Nk)$, $\beta = n^d N^d k 2^{k+d}$, and $h = k+1$.

Proof. To see correctness, let $f(X_1, \ldots, X_n) = \sum_{\boldsymbol{i} \in [n]^d} c_{\boldsymbol{i}} X_{i_1} \cdots X_{i_d}$. Using the notations in Fig. 5 and 6, it holds that $\mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1 \rangle}, \ldots, \boldsymbol{x}^{\langle n \rangle})) = (g(\eta_j))_{j \in [\ell]}$ for $h_{\boldsymbol{u}} = \sum_{\boldsymbol{i} \in [n]^d} c_{\boldsymbol{i}} \varphi_{u_1}^{\langle i_1 \rangle} \cdots \varphi_{u_d}^{\langle i_d \rangle}$ and $g = \sum_{\boldsymbol{u} \in [N]^d} p_{\boldsymbol{u}} h_{\boldsymbol{u}}$. For $j \in [m]$, $w \in [0..k]$, and $\boldsymbol{u} \in [N]^d$, the chain rule implies $\gamma_{j,\boldsymbol{u},w} = (\mathcal{D}^w h_{\boldsymbol{u}})(\zeta_j)$ if $\mu_{j,\boldsymbol{u}} \leq k - w$, and

$$\begin{aligned}
(\mathcal{D}^w g)(\zeta_j) &= \sum_{\boldsymbol{u} \in [N]^d} \sum_{v=0}^{w} \frac{w!}{v!(w-v)!} (\mathcal{D}^{w-v} p_{\boldsymbol{u}})(\zeta_j) \cdot (\mathcal{D}^v h_{\boldsymbol{u}})(\zeta_j) \\
&= \sum_{\substack{\boldsymbol{u} \in [N]^d: \\ \mu_{j,\boldsymbol{u}} \leq k-w}} \sum_{v=0}^{w} \frac{w!}{v!(w-v)!} (\mathcal{D}^{w-v} p_{\boldsymbol{u}})(\zeta_j) \cdot (\mathcal{D}^v h_{\boldsymbol{u}})(\zeta_j) \\
&\quad + \sum_{v'=1}^{w} \sum_{\substack{\boldsymbol{u} \in [N]^d: \\ \mu_{j,\boldsymbol{u}} = k-w+v'}} \sum_{v=0}^{w-v'} \frac{w!}{v!(w-v)!} (\mathcal{D}^{w-v} p_{\boldsymbol{u}})(\zeta_j) \cdot (\mathcal{D}^v h_{\boldsymbol{u}})(\zeta_j) \\
&\quad + \sum_{v'=1}^{w} \sum_{\substack{\boldsymbol{u} \in [N]^d: \\ \mu_{j,\boldsymbol{u}} = k-w+v'}} \sum_{v=0}^{v'-1} \frac{w!}{(w-v)!v!} (\mathcal{D}^{v} p_{\boldsymbol{u}})(\zeta_j) \cdot (\mathcal{D}^{w-v} h_{\boldsymbol{u}})(\zeta_j) \\
&\quad + \sum_{\substack{\boldsymbol{u} \in [N]^d: \\ \mu_{j,\boldsymbol{u}} > k}} \sum_{v=0}^{w} \frac{w!}{(w-v)!v!} (\mathcal{D}^{v} p_{\boldsymbol{u}})(\zeta_j) \cdot (\mathcal{D}^{w-v} h_{\boldsymbol{u}})(\zeta_j).
\end{aligned}$$

In the first and second terms of the last equation, $\gamma_{j,\boldsymbol{u},v} = \mathcal{D}^v h_{\boldsymbol{u}}(\zeta_j)$ since $\mu_{j,\boldsymbol{u}} \leq k - w \leq k - v$ and $\mu_{j,\boldsymbol{u}} \leq k - (w - v') \leq k - v$, respectively. Furthermore, in the third and fourth terms, we have $\mathcal{D}^v p_{\boldsymbol{u}}(\zeta_j) = 0$ since $\mu_{j,\boldsymbol{u}} = k - w + v' \geq v' > v$ and $\mu_{j,\boldsymbol{u}} > k \geq w \geq v$, respectively. It then holds that

$$\begin{aligned}
(\mathcal{D}^w g)(\zeta_j) = &\sum_{\substack{\boldsymbol{u} \in [N]^d: \\ \mu_{j,\boldsymbol{u}} \leq k-w}} \sum_{v=0}^{w} \frac{w!}{v!(w-v)!} (\mathcal{D}^{w-v} p_{\boldsymbol{u}})(\zeta_j) \cdot \gamma_{j,\boldsymbol{u},v} \\
&+ \sum_{v'=1}^{w} \sum_{\substack{\boldsymbol{u} \in [N]^d: \\ \mu_{j,\boldsymbol{u}} = k-w+v'}} \sum_{v=0}^{w-v'} \frac{w!}{v!(w-v)!} (\mathcal{D}^{w-v} p_{\boldsymbol{u}})(\zeta_j) \cdot \gamma_{j,\boldsymbol{u},v},
\end{aligned}$$

Notations.

- Parameters $n, m, \ell, d, k \in \mathbb{N}$.
- The family $\{B_1, \ldots, B_N\}$ of all maximal sets of $\varDelta$.
- A prime field $\mathbb{F}$ such that $|\mathbb{F}| \geq \max\{m + \ell, k + 1\}$.
- $m + \ell$ distinct elements $\zeta_1, \ldots, \zeta_m, \eta_1, \ldots, \eta_\ell \in \mathbb{F}$, $\boldsymbol{\zeta} = (\zeta_j)_{j \in [m]}$, and $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_\ell)$.
- For $j \in [m]$ and $\boldsymbol{u} = (u_1, \ldots, u_d) \in [N]^d$, a set $M_{j,\boldsymbol{u}} = \{\kappa \in [d] : j \in B_\kappa\}$ and $\mu_{j,\boldsymbol{u}} = |M_{j,\boldsymbol{u}}|$.
- For $j \in [m]$, a set

$$S_j = \left\{ (\boldsymbol{i}, \boldsymbol{u}, w, \boldsymbol{e}) : \begin{array}{c} \boldsymbol{i} = (i_1, \ldots, i_d) \in [n]^d, \boldsymbol{u} = (u_1, \ldots, u_d) \in [N]^d, \\ w \in [0..k] \text{ s.t. } w \leq k - \mu_{j,\boldsymbol{u}}, \\ \boldsymbol{e} = (e_1, \ldots, e_d) \in \mathbb{Z}_+^d \text{ s.t. } e_1 + \cdots + e_d = w \end{array} \right\}.$$

$\mathsf{Share}(i, \boldsymbol{x}^{\langle i \rangle})$**.** Given an index $i \in [n]$ and an input $\boldsymbol{x}^{\langle i \rangle} \in \mathbb{F}^\ell$:

1. Choose $\boldsymbol{x}_u^{\langle i \rangle} \in \mathbb{F}^\ell$, $u \in [N]$ at random such that $\boldsymbol{x}^{\langle i \rangle} = \sum_{u \in [N]} \boldsymbol{x}_u^{\langle i \rangle}$.
2. For each $u \in [N]$, choose $\varphi_u^{\langle i \rangle} \in \mathbb{F}[X]$ at random such that $\boldsymbol{x}_u^{\langle i \rangle} = (\varphi_u^{\langle i \rangle}(\eta_j))_{j \in [\ell]}$ and $\deg(\varphi_u^{\langle i \rangle}) \leq \ell - 1$.
3. For each $j \in [m]$, set

$$\begin{aligned} \mathsf{in}_j^{\langle i \rangle} &= (\varphi_u^{\langle i \rangle}(\zeta_j))_{u \in [N] : j \notin B_u} \text{ and} \\ \mathsf{rec}_j^{\langle i \rangle} &= ((\varphi_u^{\langle i \rangle}(\zeta_j))_{u \in [N] : j \in B_u}, ((\mathcal{D}\varphi_u^{\langle i \rangle})(\zeta_j), \ldots, (\mathcal{D}^k \varphi_u^{\langle i \rangle})(\zeta_j))_{u \in [N]}). \end{aligned}$$

4. Output $(\mathsf{in}_j^{\langle i \rangle}, \mathsf{rec}_j^{\langle i \rangle})_{j \in [m]}$.

$\mathsf{Eval}(j, f, (\mathsf{in}_j^{\langle i \rangle})_{i \in [n]})$**.** Given an index $j \in [m]$, a polynomial $f = \sum_{\boldsymbol{i} \in [n]^d} c_{\boldsymbol{i}} X_{i_1} \cdots X_{i_d}$ where $c_{\boldsymbol{i}} \in \mathbb{F}$, and input shares $(\mathsf{in}_j^{\langle i \rangle})_{i \in [n]}$, output

$$\mathsf{out}_j = \left(c_{\boldsymbol{i}} \prod_{\substack{\kappa: \\ \kappa \notin M_{j,\boldsymbol{u}}, e_\kappa = 0}} \varphi_{u_\kappa}^{\langle i_\kappa \rangle}(\zeta_j) : (\boldsymbol{i}, \boldsymbol{u}, w, \boldsymbol{e}) \in S_j \right).$$

Fig. 5. The sharing and evaluation algorithms of an $(n, m, \ell, d, \varDelta)$-IT-HSS scheme with recovery information for an adversary structure $\varDelta$

Notations. Using the notations in Fig. 5,

- For $\boldsymbol{u} \in [N]^d$, $p_{\boldsymbol{u}} \in \mathbb{F}[X]$ is the polynomial of minimum degree such that $p_{\boldsymbol{u}}(\eta_i) = 1$ for any $i \in [\ell]$ and $(\mathcal{D}^w p_{\boldsymbol{u}})(\zeta_j) = 0$ for any $w \in [0..k]$ and any $j \in [m]$ with $\mu_{j,\boldsymbol{u}} > w$.

$\mathsf{Dec}((\mathsf{out}_j)_{j\in[m]}, (\mathsf{rec}_j^{\langle i\rangle})_{i\in[n],j\in[m]})$**.** Given output shares $(\mathsf{out}_j)_{j\in[m]}$ and recovery information $(\mathsf{rec}_j^{\langle i\rangle})_{i\in[n],j\in[m]}$:

1. For each $j \in [m]$ and $w \in [0..k]$, compute

$$\gamma_{j,\boldsymbol{u},w} := \sum_{\substack{\boldsymbol{i},\boldsymbol{e}:\\ (\boldsymbol{i},\boldsymbol{u},w,\boldsymbol{e})\in S_j}} \frac{w!}{e_1! \cdots e_d!} \left(c_{\boldsymbol{i}} \prod_{\substack{\kappa:\\ \kappa\notin M_{j,\boldsymbol{u}}, e_\kappa=0}} \varphi_{u_\kappa}^{\langle i_\kappa\rangle}(\zeta_j) \right) \times \left(\prod_{\substack{\kappa:\\ \kappa\in M_{j,\boldsymbol{u}}, e_\kappa=0}} \varphi_{u_\kappa}^{\langle i_\kappa\rangle}(\zeta_j) \right) \left(\prod_{\kappa:e_\kappa>0} (\mathcal{D}^{e_\kappa}\varphi_{u_\kappa,v_j}^{\langle i_\kappa\rangle})(\zeta_j) \right).$$

for all $\boldsymbol{u} \in [N]^d$ such that $\mu_{j,\boldsymbol{u}} \le k - w$ and compute

$$y_{j,w} := \sum_{\substack{\boldsymbol{u}\in[N]^d:\\ \mu_{j,\boldsymbol{u}}\le k-w}} \sum_{v=0}^{w} \frac{w!}{v!(w-v)!} (\mathcal{D}^{w-v} p_{\boldsymbol{u}})(\zeta_j)\gamma_{j,\boldsymbol{u},v} + \sum_{v'=1}^{w} \sum_{\substack{\boldsymbol{u}\in[N]^d:\\ \mu_{j,\boldsymbol{u}}=k-w+v'}} \sum_{v=0}^{w-v'} \frac{w!}{v!(w-v)!} (\mathcal{D}^{w-v} p_{\boldsymbol{u}})(\zeta_j)\gamma_{j,\boldsymbol{u},v}.$$

2. Letting $\boldsymbol{z}_j = (y_{j,w})_{w\in[0..k]}$, output $\mathsf{Hermite}_{\boldsymbol{\zeta},\boldsymbol{\eta}}((\boldsymbol{z}_j)_{j\in[m]})$.

Fig. 6. The decoding algorithm of an (n, m, ℓ, d, Δ)-IT-HSS scheme with recovery information for an adversary structure Δ

which is equal to $y_{j,w}$. For each $j \in [m]$, $w \in [0..k]$, and $\boldsymbol{u} \in [N]^d$, define $c_{w,\boldsymbol{u}} = |\{j \in [m] : \mu_{j,\boldsymbol{u}} > w\}|$, and $\delta_{j,w,\boldsymbol{u}} = 1$ if $\mu_{j,\boldsymbol{u}} \leq w$ and otherwise 0. Proposition 1 implies that $\deg p_{\boldsymbol{u}} \leq \ell - 1 + \sum_{j \in [m]} \mu_{j,\boldsymbol{u}}$. We also have that

$$\begin{aligned}
\sum_{j \in [m]} \mu_{j,\boldsymbol{u}} &= \sum_{w \in [0..k]} c_{w,\boldsymbol{u}} \\
&= (k+1)m - \sum_{j=1}^{m} \sum_{w=0}^{k} \delta_{j,w,\boldsymbol{u}} \\
&= (k+1)m - \sum_{j=1}^{m} \max\{k+1-\mu_{j,\boldsymbol{u}}, 0\} \\
&= (k+1)m - |(k+1)\mathbf{1}_m - (\boldsymbol{a}_{u_1}(j) + \cdots + \boldsymbol{a}_{u_d}(j))|_+.
\end{aligned}$$

Hence, the condition (5) implies that $\deg p_{\boldsymbol{u}} < (k+1)m - d(\ell-1)$. Then, we have $\deg g \leq \max_{\boldsymbol{u} \in [N]^d}\{p_{\boldsymbol{u}} h_{\boldsymbol{u}}\} < (k+1)m$ and $\mathsf{Hermite}_{\boldsymbol{\zeta},\boldsymbol{\eta}}((\boldsymbol{z}_j)_{j \in [m]}) = (g(\eta_j))_{j \in [\ell]}$.

An input share $\mathsf{in}_j^{\langle i \rangle}$ is a vector of dimension $N_j := |\{u \in [N] : j \notin B_u\}|$ and each piece of recovery information $\mathsf{rec}_j^{\langle i \rangle}$ is a vector of dimension $N - N_j + Nk$. Since $|S_j| \leq n^d \cdot N^d \cdot k \cdot 2^{k+d} =: \beta$, an output share out_j is a vector $\boldsymbol{y}_j \in \mathbb{F}^{\beta}$. Each $y_{j,w}$ computed by Dec is linear in the $\gamma_{j,\boldsymbol{u},v}$'s, each of which is in turn a degree-1 polynomial of out_j. The degree of $y_{j,w}$ with respect to $(\mathsf{rec}_j^{\langle i \rangle})_{i \in [n]}$ is at most the maximum of $\mu_{j,\boldsymbol{u}} + |\{\kappa \in [d] : e_\kappa > 0\}|$ over all $\boldsymbol{u} \in [N]^d$ and all $\boldsymbol{e} \in \mathbb{Z}_+^d$ such that $\mu_{j,\boldsymbol{u}} \leq k - w$ and $\sum_\kappa e_\kappa = w$, which is at most $(k-w) + w \leq k$. Therefore, $\boldsymbol{z}_j = (y_{j,w})_{w \in [0..k]}$ can be expressed as $\boldsymbol{A}_j \boldsymbol{y}_j + \boldsymbol{b}_j$ using a $(k+1)$-by-β matrix $\boldsymbol{A}_j$ and $(k+1)$-dimensional vector $\boldsymbol{b}_j$ whose entries are degree-k polynomials of $(\boldsymbol{r}_j^{\langle i \rangle})_{i \in [n]}$. Since $\mathsf{Hermite}_{\boldsymbol{\zeta},\boldsymbol{\eta}}((\boldsymbol{z}_j)_{j \in [m]})$ is linear in $(\boldsymbol{z}_j)_{j \in [m]}$, the output of Dec can be expressed as $\sum_{j \in [m]} \boldsymbol{C}_j(\boldsymbol{A}_j \boldsymbol{y}_j + \boldsymbol{b}_j)$ using $\boldsymbol{C}_j \in \mathbb{F}^{\ell \times (k+1)}$.

To see Δ-privacy, we show that for any $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{F}^\ell$ and $B \in \Delta$, the distributions of $(\mathsf{in}_j)_{j \in B}$ for $\boldsymbol{x}$ and $(\mathsf{in}'_j)_{j \in B}$ for $\boldsymbol{x}'$ are identical. We may assume that $B = B_i$ for some $i \in [N]$. There exists a polynomial θ such that $\deg(\theta) \leq \ell - 1$ and $(\theta(\eta_j))_{j \in [\ell]} = \boldsymbol{x}' - \boldsymbol{x}$. We then have a bijection between the random strings used by Share on input $\boldsymbol{x}$ and those used by Share on input $\boldsymbol{x}'$ such that the shares of B are the same under this bijection. Indeed, we map any polynomials $(\varphi_u)_{u \in [N]}$ generated by Share on input $\boldsymbol{x}$ to $(\varphi'_u)_{u \in [N]}$ where $\varphi'_u = \varphi_u + \theta$ if $u = i$ and otherwise $\varphi'_u = \varphi_u$. Then, $(\varphi'_u)_{u \in [N]}$ provide consistent shares for $\boldsymbol{x}'$ and the shares of B resulting from $(\varphi'_u)_{u \in [N]}$ are the same as the ones for $(\varphi_u)_{u \in [N]}$ since the j-th share does not contain $\varphi'_i(\zeta_j)$ or $\varphi_i(\zeta_j)$ if $j \in B_i$. □

By applying the ILM compiler in Proposition 2, we obtain an HSS scheme for a general adversary structure. Again, as in Corollary 2, we have to add similar re-randomizing procedures to guarantee context hiding.

Corollary 3. *Using the notations in Theorem 3, assume that $n, m \in \mathsf{poly}(\lambda)$, $k, d \in O(1)$, and $N \in \mathsf{poly}(\lambda)$. Assuming a (resp. multi-key) k-HE scheme* HE*, there exists an (n, m, ℓ, d, Δ)-HSS scheme* HSS *(resp. in the plain model) with* *efficiency measures* $\alpha = N \log |\mathbb{F}| + Nk$ *and* $\beta = k + 1$*. Furthermore, if* HE *satisfies circuit privacy, there exists a context-hiding* (n, m, ℓ, d, Δ)*-HSS scheme* HSS' *(resp. in the plain model) with efficiency measures* $\alpha = (N+k+1) \log |\mathbb{F}| + Nk = O(N \log(m + \ell))$ *and* $\beta = k + 1 = O(1)$*.*

Proof. In view of Proposition 2 and Theorem 3, we obtain an (n, m, ℓ, d, Δ)-HSS scheme HSS with efficiency measures $\alpha = N \log |\mathbb{F}| + Nk$ and $\beta = k + 1$.

We make HSS context-hiding by adding re-randomizing procedures to the sharing and evaluation algorithms. Using the notations in the proof of Theorem 3, the output of the sharing algorithm of HSS on input $\boldsymbol{x}^{\langle i \rangle}$ includes

$$\boldsymbol{c}_j^{\langle i \rangle} = ((\mathsf{HE.Enc}(\mathsf{pk}, \varphi_u^{\langle i \rangle}(\zeta_j)))_{u \in [N]: j \in B_u}, (\mathsf{HE.Enc}(\mathsf{pk}, \mathcal{D}^w \varphi_u^{\langle i \rangle}(\zeta_j)))_{u \in [N], w \in [0..k]})$$

for $j \in [m]$. The evaluation algorithm executed by the j-th server outputs $k + 1$ ciphertexts $(d_{j,w})_{w \in [0..k]}$ each of which decrypts to $y_{j,w} = \mathcal{D}^w g(\zeta_j)$ for $w \in [0..k]$. Note that there are degree-k polynomials $\nu_{j,w}$ (whose coefficients depend on $(\varphi_u^{\langle i \rangle}(\zeta_j))_{u \in [N]: j \notin B_u}$) such that $d_{j,w} = \mathsf{HE.Eval}(\mathsf{pk}, \nu_{j,w}, (\boldsymbol{c}_j^{\langle 1 \rangle}, \ldots, \boldsymbol{c}_j^{\langle n \rangle}))$. We fix any client, say, $i = 1$. If the input index is $i = 1$, we let the sharing algorithm generate a random polynomial θ such that $\deg(\theta) < (k+1)m$ and $(\theta(\eta_j))_{j \in [\ell]} = \mathbf{0}$. Then, it sends $k + 1$ field elements $y'_{j,w} = \mathcal{D}^w \theta(\zeta_j)$ for $w \in [0..k]$ to the j-th server in addition to an input share $\mathsf{in}_j^{\langle 1 \rangle}$ of HSS. We do not modify the procedures for the other clients. The size of input shares are now $\alpha = (N+k+1) \log |\mathbb{F}| + Nk$. When executing the evaluation algorithm of HSS, the j-th server outputs $k + 1$ ciphertexts $(d'_{j,w})_{w \in [0..k]}$ each of which decrypts to $y_{j,w} + y'_{j,w} = (\mathcal{D}^w g')(\zeta_j)$ where $g' = g + \theta$. More precisely, the j-th server computes $d'_{j,w} \leftarrow \mathsf{HE.Eval}(\mathsf{pk}, \nu_{j,w}(\cdot) + y'_{j,w}, (\boldsymbol{c}_j^{\langle 1 \rangle}, \ldots, \boldsymbol{c}_j^{\langle n \rangle}))$ for all $w \in [0..k]$.

Given $\mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1 \rangle}, \ldots, \boldsymbol{x}^{\langle n \rangle}))$, the simulator $S_{\mathsf{HSS}'}$ randomly chooses a polynomial $\widetilde{g}$ such that $(\widetilde{g}(\eta_j))_{j \in [\ell]} = \mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1 \rangle}, \ldots, \boldsymbol{x}^{\langle n \rangle}))$ and $\deg \widetilde{g} < (k+1)m$, and then computes $\widetilde{d}_{j,w} \leftarrow S_{\mathsf{HE}}(1^\lambda, \mathsf{pk}, (\mathcal{D}^w \widetilde{g})(\zeta_j))$ for all $j \in [m]$ and $w \in [0..k]$, where S_{HE} is the simulator for the circuit privacy of HE. It finally outputs $((\widetilde{d}_{1,w})_{w \in [0..k]}, \ldots, (\widetilde{d}_{m,w})_{w \in [0..k]})$. We analyze the distribution of the output of $S_{\mathsf{HSS}'}$. Let $j \in [m]$, $w \in [0..k]$, and $r_{j,w}$ be any fixed field element. The circuit privacy of HE implies that $\mathsf{HE.Eval}(\mathsf{pk}, \nu_{j,w}(\cdot) + r_{j,w}, (\boldsymbol{c}_j^{\langle 1 \rangle}, \ldots, \boldsymbol{c}_j^{\langle n \rangle})) \approx S_{\mathsf{HE}}(1^\lambda, \mathsf{pk}, y_{j,w} + r_{j,w})$, where $\approx$ denotes statistical indistinguishability. Then, we have that

$$\begin{aligned} &(\mathsf{HE.Eval}(\mathsf{pk}, \nu_{j,w}(\cdot) + r_{j,w}, (\boldsymbol{c}_j^{\langle 1 \rangle}, \ldots, \boldsymbol{c}_j^{\langle n \rangle})))_{w \in [0..k], j \in [m]} \\ &\quad \approx (S_{\mathsf{HE}}(1^\lambda, \mathsf{pk}, y_{j,w} + r_{j,w}))_{w \in [0..k], j \in [m]}. \end{aligned} \tag{6}$$

Let V be the set from which θ is sampled. Note that V is a linear space over $\mathbb{F}$. Since Eq. (6) holds for any fixed elements $r_{j,w}$, we can apply it to $r_{j,w} = y'_{j,w} = \mathcal{D}^w \theta(\zeta_j)$ where $\theta \leftarrow_\$ V$, and obtain that

$$\begin{aligned} (d'_{j,w})_{w \in [0..k], j \in [m]} &= (\mathsf{HE.Eval}(\mathsf{pk}, \nu_{j,w}(\cdot) + y'_{j,w}, (\boldsymbol{c}_j^{\langle 1 \rangle}, \ldots, \boldsymbol{c}_j^{\langle n \rangle})))_{w \in [0..k], j \in [m]} \\ &\approx (S_{\mathsf{HE}}(1^\lambda, \mathsf{pk}, y_{j,w} + y'_{j,w}))_{w \in [0..k], j \in [m]}. \end{aligned} \tag{7}$$

The distribution of g' is the uniform distribution over an affine space $g + V := \{g+\theta : \theta \in V\}$, which is the same as that of $\widetilde{g}$. Taking derivatives and substitution are both linear maps over $\mathbb{F}$. Hence, the distribution of $(\mathcal{D}^w g'(\zeta_j))_{w\in[0..k],j\in[m]}$ induced by $\theta \leftarrow_{\$} V$ is identical to that of $(\mathcal{D}^w \widetilde{g}(\zeta_j))_{w\in[0..k],j\in[m]}$ induced by $\widetilde{g} \leftarrow_{\$} g + V$. Combined with Eq. (7), we obtain that

$$\begin{aligned}(d'_{j,w})_{w\in[0..k],j\in[m]} &\approx (S_{\mathsf{HE}}(1^\lambda, \mathsf{pk}, \mathcal{D}^w g'(\zeta_j)))_{w\in[0..k],j\in[m]} \\ &= S_{\mathsf{HSS}'}(1^\lambda, \mathsf{pk}, \mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1\rangle}, \ldots, \boldsymbol{x}^{\langle n\rangle}))).\end{aligned}$$

We show that the computational complexity of HSS' is polynomial in the security parameter λ. The most costly step in the sharing algorithm of HSS_0 is sampling random polynomials $\varphi_u^{\langle i\rangle}$, which can be done in polynomial time in N, ℓ by pre-computing and publishing a basis for the linear space $\mathcal{L}_u := \{\varphi_u \in \mathbb{F}[X] : \deg(\varphi_u) \le \ell - 1,\ (\varphi_u(\eta_j))_{j\in[\ell]} = \mathbf{0}\}$ for each $u \in [N]$. We have that $\ell = O(m)$ due to the condition (5) and hence the time complexity of that step is polynomial in λ. Since the additional steps of $\mathsf{HSS}'.\mathsf{Share}$ can also be done in time $\mathsf{poly}(m, N, \ell, \lambda)$, the time complexity of $\mathsf{HSS}'.\mathsf{Share}$ is polynomial in λ. Next, $\mathsf{HSS}_0.\mathsf{Eval}$ computes $\beta = O(n^d N^d)$ products and $\mathsf{HE}.\mathsf{Eval}$ can be executed by $\mathsf{HSS}'.\mathsf{Eval}$ in time $\beta(n\rho)^k \cdot \mathsf{poly}(\lambda) = O(n^{d+k} N^{d+k}) \cdot \mathsf{poly}(\lambda)$. The time complexity of $\mathsf{HSS}'.\mathsf{Eval}$ is therefore also polynomial in λ. Finally, in view of Proposition 2, $\mathsf{HSS}'.\mathsf{Dec}$ can be done in time $m\ell(k+1) \cdot \mathsf{poly}(\lambda) = \mathsf{poly}(\lambda)$.

We can lift the above scheme to the plain model by replacing the HE scheme with the corresponding multi-key variant. $\square$

Since $N = \mathsf{poly}(\lambda)$ is required, Corollary 3 cannot be applied to adversary structures whose number of all maximal sets is exponential in $m = \mathsf{poly}(\lambda)$, which actually occurs in the worst case. Therefore, Corollary 3 is especially important for the case of $N = \mathsf{poly}(m)$. As shown in Section 8, there is still an interesting class of adversary structures even if $N = O(m)$.

We can instantiate the m-input MPC protocol in Section 3.4 with the HSS scheme in Corollary 3 and then the communication complexity is $O(Nm^2 \log(m+\ell))$, where ℓ is the number of parallel evaluations.

The scheme [30] can be recovered by setting $\ell = 1$. Indeed, at Step 2 of the sharing algorithm in Fig. 5, a possible polynomial $\varphi_u^{\langle i\rangle}$ is only the constant polynomial $\boldsymbol{x}_u^{\langle i\rangle}$ of degree 0. Therefore, $\mathsf{in}_j^{\langle i\rangle}$ at Step 3 consists of $(\boldsymbol{x}_u^{\langle i\rangle})_{u\in[N]:j\notin B_u}$. We can set $\mathsf{rec}_j^{\langle i\rangle}$ as $(\boldsymbol{x}_u^{\langle i\rangle})_{u\in[N]:j\in B_u}$ by removing the other entries since $\mathcal{D}^w\varphi_u^{\langle i\rangle}$ is the zero polynomial for $w \ge 1$. This is exactly the same as the one in [30].

8 Formalization of Practical Adversary Structures

We formalize two classes of non-threshold adversary structures and show advantages of our schemes over the previous schemes [19, 17, 30, 26] in the application to m-input MPC shown in Section 3.4. That is, the i-th input party has ℓ kinds of data $\boldsymbol{x}^{\langle i\rangle} = (\boldsymbol{x}^{\langle i\rangle}(1), \ldots, \boldsymbol{x}^{\langle i\rangle}(\ell)) \in \mathbb{F}^\ell$ and an output party wants

$\mathcal{P}_\ell(f, (\boldsymbol{x}^{\langle 1 \rangle}, \ldots, \boldsymbol{x}^{m}))$. We suppose that the degree d is independent of the number of data m. For example, d depends only on the order of approximation for Fisher's linear discriminant analysis [24].

8.1 Unbalanced 2-Partite Adversary Structure

Let $\Pi = (P_1, P_2)$ be a 2-partition and $\tau, \sigma \in \mathbb{R}_+$. Define a Π-partite adversary structure $\mathcal{B}^{\Pi}_{\tau,\sigma} = \{X \subseteq [m] : |X| \leq \tau m \wedge (|X \cap P_1| \leq \sigma m \vee |X \cap P_2| \leq \sigma m)\}$. The motivation behind $\mathcal{B}^{\Pi}_{\tau,\sigma}$ is modification of $\mathcal{T}^{m}_{\tau}$ so that it takes into account a real-world situation. Suppose that input parties are classified to two organizations P_1, P_2 and an adversary $\mathcal{A}$ is one of the parties. If $\mathcal{A}$ belongs to P_1, then it would be more difficult for $\mathcal{A}$ to corrupt parties in the other organization P_2 than parties in P_1. We therefore add the constraint $|X \cap P_2| \leq \sigma m$ to the threshold constraint $|X| \leq \tau m$. Similarly, we require $|X \cap P_1| \leq \sigma m$. We show that an HSS scheme for $\mathcal{B}^{\Pi}_{\tau,\sigma}$ is obtained from Corollary 2 under a certain parameter setting. The proof is given in the full version.

Proposition 3. *Let $m \in \mathbb{N}$ be an even number and $\Pi = (P_1, P_2)$ be a 2-partition such that $|P_1| = |P_2| = m/2$. Let $k, d \in \mathbb{N}$ be constants and assume that d is an odd number. Let $\epsilon, \tau, \sigma \in \mathbb{R}_+$ be such that*

$$\frac{d-1}{2d}\tau + \frac{d+1}{2d}\sigma + \epsilon \leq \frac{k+1}{2d} \text{ and } \sigma \leq \tau. \tag{8}$$

and set $\ell = \epsilon m$. Assuming a circuit-private k-HE scheme and a prime field $\mathbb{F}$ with $|\mathbb{F}| \geq \max\{m + \ell, k + 1\}$, there exists a context-hiding $(m, m, \ell, d, \mathcal{B}^{\Pi}_{\tau,\sigma})$-HSS scheme with efficiency measures $\alpha = (k+3)\log|\mathbb{F}| + 2k$ and $\beta = k + 1$.

The threshold schemes [19, 26] only tolerate $\mathcal{T}^{m}_{\tau}$ and hence inapplicable to $\mathcal{B}^{\Pi}_{\tau,\sigma}$ for $\tau \geq (k+1)/d$. In the scheme [30], the input share size α is exponential in m since the number of all maximal sets of $\mathcal{B}^{\Pi}_{\tau,\sigma}$ is larger than $\binom{m/2}{\sigma m}\binom{m/2}{(\tau-\sigma)m}$. Our scheme can be applied to $\mathcal{B}^{\Pi}_{\tau,\sigma}$ even for τ such that $(k+1)/d \leq \tau < (k+1)/(d-1)$ and the input and output share sizes are only constant numbers of field elements and ciphertexts. Furthermore, if there are sufficiently many input parties, it can support parallel evaluations without increasing communication complexity.

To be more concrete, suppose that there are $m = 1000$ input parties and that a polynomial of degree $d = 5$ is computed on $\ell = 10$ data sets. Assuming a 1-HE scheme (i.e., $k = 1$), we can choose $\epsilon = 0.01$, $\tau = 0.45$, and $\sigma = 0.01$, which means that our scheme can tolerate a collusion X of at most 450 input parties such that $|X \cap P_1| \leq 10$ or $|X \cap P_2| \leq 10$. The point-to-point communication complexity between input parties is 4 field elements plus 2 ciphertexts and each input party sends 2 ciphertexts to an output party. Hence, the total communication complexity is $4m^2$ field elements plus $2m^2 + 2m$ ciphertexts. The schemes [19, 26] cannot be applied to this setting since $\tau > 0.4 = (k+1)/d$. Since the scheme of [17] does not support parallel evaluation, the communication complexity increases $\ell = \epsilon m$ times, that is, $4\epsilon m^3$ field elements plus $2\epsilon m^3 + 2\epsilon m^2$ ciphertexts.

8.2 Adversary Structure Induced by a Random Graph

Let $G = ([m], E)$ be a graph on the set of input parties (vertices) and let A_j be the set of all adjacent vertices of $j \in [m]$. We define an adversary structure $\Delta_G = \{X \subseteq [m] : X \subseteq A_j \cup \{j\} \text{ for some } j \in [m]\}$. Note that the number N of all the maximal sets of Δ_G is at most m. The motivation behind Δ_G is a real-world scenario in which an adversary is one of input parties and colludes with all adjacent parties in G. For p with $0 < p < 1$, we consider the probability distribution $\mathcal{G}(m, p)$ [22] over the set of all the graphs on m vertices, in which a random graph is obtained by starting with a set of m isolated vertices and adding every possible edge independently with probability p. We show that for sufficiently large m and G sampled from $\mathcal{G}(m, p)$, an HSS scheme for Δ_G can be obtained from Corollary 3 with high probability. The proof is given in the full version.

Proposition 4. *Let $m, d, k \in \mathbb{N}$ and p be a real number with $0 < p < 1$. Let $\ell \in \mathbb{N}$ be such that $\ell \leq (d+1)^{-1}k(1-p)^d(m-d)$ and $q \in \mathbb{R}_+$ be such that*

$$q \geq 1 - m^d \exp\left(-\frac{2(1-p)^{2d}(m-d)}{(k+1)^2}\right).$$

Assume a circuit-private k-HE scheme and a prime field $\mathbb{F}$ with $|\mathbb{F}| \geq \max\{m + \ell, k+1\}$. If G is sampled from $\mathcal{G}(m, p)$, then with probability at least q, there exists a context-hiding $(m, m, \ell, d, \Delta_G)$-HSS scheme with efficiency measures $\alpha = (m + k + 1)\log|\mathbb{F}| + 2k$ and $\beta = k + 1$.

We demonstrate concrete parameters. Assume a 1-HE scheme, i.e., $k = 1$. Suppose that we want to compute a single polynomial of degree $d = 5$ and that every possible edge occurs with probability $p = 0.45$. Then, if there are $m \geq 50000$ input parties, we can obtain, with probability at least 0.99, an HSS scheme for Δ_G computing the polynomial on $\ell \gtrsim m/200 \geq 250$ different data sets. The threshold schemes [19, 26] cannot be applied to this setting since $\Delta_G \nsubseteq \mathcal{T}_\tau^m$ with high probability if $p > \tau$. Precisely, we also show in the full version that if $p = \tau + \epsilon$, the probability that $\Delta_G \subseteq \mathcal{T}_\tau^m$ occurs is at most $\exp(-2(m\epsilon - p + 1)^2/(m-1))$, which converges exponentially to 0 for $m \to \infty$. For the above parameters, if $m \geq 1000$, $\Delta_G \nsubseteq \mathcal{T}_{0.4}^m$ occurs with probability at least 0.99. The scheme [30] can tolerate Δ_G and does not require $|\mathbb{F}| \geq m + \ell$. However, their scheme cannot support parallel evaluation and hence the efficiency measures $\alpha = O(\ell m \log|\mathbb{F}|)$ and $\beta = O(\ell)$ are ℓ times larger than our scheme. As for the assumption on the field size, since statistical analysis and machine learning typically deal with numerical data, we end up to choose a field of size much larger than $m + \ell$ due to another requirement of sufficiently approximating the data.

Acknowledgements

This research was partially supported by JSPS KAKENHI Grant Numbers JP20J20797 and 19H01109, JST CREST JPMJCR19F6 and JPMJCR14D6, and Ministry of Internal Affairs and Communications SCOPE Grant Number 182103105.

References

1. Attrapadung, N., Hanaoka, G., Mitsunari, S., Sakai, Y., Shimizu, K., Teruya, T.: Efficient two-level homomorphic encryption in prime-order bilinear groups and a fast implementation in webassembly. In: Proceedings of the 2018 on Asia Conference on Computer and Communications Security. pp. 685–697. ASIACCS '18 (2018)
2. Barkol, O., Ishai, Y., Weinreb, E.: On d-multiplicative secret sharing. Journal of Cryptology **23**(4), 580–593 (2010)
3. Beaver, D., Micali, S., Rogaway, P.: The round complexity of secure protocols. In: Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing. pp. 503–513. STOC '90 (1990)
4. Boneh, D., Goh, E.J., Nissim, K.: Evaluating 2-DNF formulas on ciphertexts. In: Theory of Cryptography. pp. 325–341 (2005)
5. Boyle, E., Couteau, G., Gilboa, N., Ishai, Y., Kohl, L., Scholl, P.: Efficient pseudorandom correlation generators: Silent OT extension and more. In: Advances in Cryptology – CRYPTO 2019, Part III. pp. 489–518 (2019)
6. Boyle, E., Gilboa, N., Ishai, Y.: Breaking the circuit size barrier for secure computation under DDH. In: Advances in Cryptology – CRYPTO 2016, Part I. pp. 509–539 (2016)
7. Boyle, E., Kohl, L., Scholl, P.: Homomorphic secret sharing from lattices without FHE. In: Advances in Cryptology – EUROCRYPT 2019, Part II. pp. 3–33 (2019)
8. Cachin, C., Camenisch, J., Kilian, J., Müller, J.: One-round secure computation and secure autonomous mobile agents. In: Automata, Languages and Programming. pp. 512–523 (2000)
9. Cascudo, I., Cramer, R., Xing, C., Yuan, C.: Amortized complexity of information-theoretically secure mpc revisited. In: Advances in Cryptology – CRYPTO 2018, Part III. pp. 395–426 (2018)
10. Castagnos, G., Laguillaumie, F.: Linearly homomorphic encryption from DDH. In: Topics in Cryptology – CT-RSA 2015. pp. 487–505 (2015)
11. Catalano, D., Fiore, D.: Using linearly-homomorphic encryption to evaluate degree-2 functions on encrypted data. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. pp. 1518–1529. CCS '15 (2015)
12. Cheon, J.H., Coron, J.S., Kim, J., Lee, M.S., Lepoint, T., Tibouchi, M., Yun, A.: Batch fully homomorphic encryption over the integers. In: Advances in Cryptology – EUROCRYPT 2013. pp. 315–335 (2013)
13. Damgård, I., Pastro, V., Smart, N., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: Advances in Cryptology – CRYPTO 2012. pp. 643–662 (2012)
14. Diffie, W., Hellman, M.: New directions in cryptography. IEEE transactions on Information Theory **22**(6), 644–654 (1976)
15. Dodis, Y., Halevi, S., Rothblum, R.D., Wichs, D.: Spooky encryption and its applications. In: Advances in Cryptology – CRYPTO 2016, Part III. pp. 93–122 (2016)
16. Elgamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory **31**(4), 469–472 (1985)

17. Eriguchi, R., Kunihiro, N.: d-Multiplicative secret sharing for multipartite adversary structures. In: 1st Conference on Information-Theoretic Cryptography (ITC 2020). Leibniz International Proceedings in Informatics (LIPIcs), vol. 163, pp. 2:1–2:16 (2020)
18. Farràs, O., Padró, C.: Ideal secret sharing schemes for useful multipartite access structures. In: Coding and Cryptology. pp. 99–108 (2011)
19. Franklin, M., Yung, M.: Communication complexity of secure computation (extended abstract). In: Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing. pp. 699–710. STOC '92 (1992)
20. Freeman, D.M.: Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In: Advances in Cryptology – EUROCRYPT 2010. pp. 44–61 (2010)
21. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing. pp. 169–0178. STOC '09 (2009)
22. Gilbert, E.N.: Random Graphs. The Annals of Mathematical Statistics **30**(4), 1141–1144 (1959)
23. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game. In: Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing. pp. 218–229. STOC '87 (1987)
24. Graepel, T., Lauter, K., Naehrig, M.: ML confidential: Machine learning on encrypted data. In: Information Security and Cryptology – ICISC 2012. pp. 1–21 (2013)
25. Hazay, C., Orsini, E., Scholl, P., Soria-Vazquez, E.: Tinykeys: A new approach to efficient multi-party computation. In: Advances in Cryptology – CRYPTO 2018, Part III. pp. 3–33 (2018)
26. Ishai, Y., Lai, R.W.F., Malavolta, G.: A geometric approach to homomorphic secret sharing. In: Public-Key Cryptography – PKC 2021. pp. 92–119 (2021)
27. Lai, R.W.F., Malavolta, G., Schröder, D.: Homomorphic secret sharing for low degree polynomials. In: Advances in Cryptology – ASIACRYPT 2018, Part III. pp. 279–309 (2018)
28. López-Alt, A., Tromer, E., Vaikuntanathan, V.: On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: Proceedings of the Forty-Fourth Annual ACM Symposium on Theory of Computing. pp. 1219–1234. STOC '12 (2012)
29. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Advances in Cryptology — EUROCRYPT '99. pp. 223–238 (1999)
30. Phalakarn, K., Suppakitpaisarn, V., Attrapadung, N., Matsuura, K.: Constructive t-secure homomorphic secret sharing for low degree polynomials. In: Progress in Cryptology – INDOCRYPT 2020. pp. 763–785 (2020)
31. Shamir, A.: How to share a secret. Communications of the ACM **22**(11), 612–613 (1979)
32. Smart, N.P., Vercauteren, F.: Fully homomorphic SIMD operations. Designs, Codes and Cryptography **71**(1), 57–81 (2014)
33. Spitzbart, A.: A generalization of Hermite's interpolation formula. The American Mathematical Monthly **67**(1), 42–46 (1960)
34. Woodruff, D., Yekhanin, S.: A geometric approach to information-theoretic private information retrieval. In: 20th Annual IEEE Conference on Computational Complexity (CCC'05). pp. 275–284 (2005)