Matrix Computational Assumptions in Multilinear Groups* **

Paz Morillo[1], Carla Ràfols[2], and Jorge L. Villar[1]

[1] Universitat Politècnica de Catalunya, Spain
{paz.morillo,jorge.villar}@upc.edu
[2] Universitat Pompeu Fabra, Spain
carla.rafols@upf.edu

Abstract. We put forward a new family of computational assumptions, the Kernel Matrix Diffie-Hellman Assumption. Given some matrix $\mathbf{A}$ sampled from some distribution $\mathcal{D}$, the kernel assumption says that it is hard to find "in the exponent" a nonzero vector in the kernel of $\mathbf{A}^\top$. This family is the natural computational analogue of the Matrix Decisional Diffie-Hellman Assumption (MDDH), proposed by Escala *et al.* As such it allows to extend the advantages of their algebraic framework to computational assumptions.
The k-Decisional Linear Assumption is an example of a family of decisional assumptions of strictly increasing hardness when k grows. We show that for any such family of MDDH assumptions, the corresponding Kernel assumptions are also strictly increasingly weaker. This requires ruling out the existence of some black-box reductions between flexible problems (*i.e.*, computational problems with a non unique solution).

Keywords: Matrix Assumptions, Computational Problems, Black-Box Reductions, Structure Preserving Cryptography

1 Introduction

It is commonly understood that cryptographic assumptions play a crucial role in the development of secure, efficient protocols with strong functionalities. For instance, upon referring to the rapid development of pairing-based cryptography, X. Boyen [8] says that "it has been supported, in no small part, by a dizzying array of tailor-made cryptographic assumptions". Although this may be a reasonable price to pay for constructing new primitives or improve their efficiency, one should not lose sight of the ideal of using standard and simple assumptions. This is an important aspect of provable security. Indeed, Goldreich [16], for instance, cites "having clear definitions of one's assumptions" as one of the three main ingredients of good cryptographic practice.

There are many aspects to this goal. Not only it is important to use clearly defined assumptions, but also to understand the relations between them: to see, for example, if two assumptions are equivalent or one is weaker than the other. Additionally, the definitions should allow to make accurate security claims. For instance, although technically it is correct to say that unforgeability of the Waters' signature scheme [41] is implied by the DDH Assumption, defining the CDH Assumption allows to make a much more precise security claim.

A notable effort in reducing the "dizzying array" of cryptographic assumptions is the work of Escala *et al.* [11]. They put forward a new family of decisional assumptions in a prime order group $\mathbb{G}$, the *Matrix Diffie-Hellman* Assumption ($\mathcal{D}_{\ell,k}$-MDDH). It says that, given some matrix $\mathbf{A} \in \mathbb{Z}_q^{\ell \times k}$ sampled from some distribution $\mathcal{D}_{\ell,k}$, it is hard to decide membership in $\mathrm{Im}\,\mathbf{A}$, the subspace spanned by the columns of $\mathbf{A}$, in the exponent. Rather than as new assumption, it should be seen as an algebraic framework for decisional assumptions which includes as a special case the widely used k-Lin family.

* Work supported by the Spanish research project MTM2013-41426-R and by a Sofja Kovalevskaya Award of the Alexander von Humboldt Foundation and the German Federal Ministry for Education and Research.

** A preliminary version of this paper appears in *Advances in Cryptology - ASIACRYPT 2016, 22nd International Conference on the Theory and Application of Cryptology and Information Security,* Jung Hee Cheon, Tsuyoshi Takagi ed., LNCS, Springer, 2016. This is the full version.

This framework has some obvious conceptual advantages. For instance, it allows to explain all the members of the k-Lin assumption family (and also others, like the uniform assumption, appeared previously in [13,14,40]) as a single assumption and unify different constructions of the same primitive in the literature (e.g., the Naor-Reingold PRF [35] and the Lewko-Waters PRF [29] are special cases of the same construction instantiated with the 1-Lin and the 2-Lin Assumption, respectively). Another of its advantages is that it avoids arbitrary choices and instead points out to a trade-off between efficiency and security (a scheme based on any $\mathcal{D}_{\ell,k}$-MDDH Assumption can be instantiated with many different assumptions, some leading to stronger security guarantees and others leading to more efficient schemes). But follow-up work has also illustrated other possibly less obvious advantages. For instance, Herold *et al.* [21] have used the Matrix Diffie-Hellman abstraction to extend the model of composite-order to prime-order transformation of Freeman [13] and to derive efficiency improvements which were proven to be impossible in the original model.[3] We believe this illustrates that the benefits of conceptual clarity can translate into concrete improvements as well.

The security notions for cryptographic protocols can be classified mainly in hiding and unforgeability ones. The former typically appear in encryption schemes and commitments and the latter in signature schemes and soundness in zero-knowledge proofs. Although it is theoretically possible to base the hiding property on computational problems, most of the practical schemes achieve this notion either information theoretically or based on decisional assumptions, at least in the standard model. Likewise, unforgeability naturally comes from computational assumptions (typically implied by stronger, decisional assumptions). Thus, a natural question is if one can find a computational analogue of their MDDH Assumption which can be used in "unforgeability type" of security notions.

Most computational problems considered in the literature are search problems with a unique solution like the discrete logarithm or CDH. But unforgeability actually means the inability to produce one among many solutions to a given problem (*e.g.*, in many signature schemes or zero knowledge proofs). Thus, unforgeability is more naturally captured by a *flexible computational problem*, namely, a problem which admits several solutions[4]. This maybe explains why several new flexible assumptions have appeared recently when considering "unforgeability-type" security notions in structure-preserving cryptography [2]. Thus a useful computational analogue of the MDDH Assumption should not only consider problems with a unique solution but also flexible problems which can naturally capture this type of security notions.

1.1 Our Results

In the following $\mathcal{G} = (\mathbb{G}, q, \mathcal{P})$, being $\mathbb{G}$ some group in additive notation of prime order q generated by $\mathcal{P}$, that is, the elements of $\mathbb{G}$ are $\mathcal{Q} = a\mathcal{P}$ where $a \in \mathbb{Z}_q$. They will be denoted as $[a] := a\mathcal{P}$. This notation naturally extends to vectors and matrices as $[\boldsymbol{v}] = (v_1\mathcal{P}, \ldots, v_n\mathcal{P})$ and $[\mathbf{A}] = (A_{ij}\mathcal{P})$.

Computational Matrix Assumptions. In our first attempt to design a computational analogue of the MDDH Assumption, we introduce the *Matrix Computational DH Assumption*, (MCDH) which says that, given a uniform vector $[\boldsymbol{v}] \in \mathbb{G}^k$ and some matrix $[\mathbf{A}]$, $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}$ for $\ell > k$, it is hard to extend $[\boldsymbol{v}]$ to a vector in $\mathbb{G}^\ell$ in the image of $[\mathbf{A}]$, $\mathrm{Im}[\mathbf{A}]$. Although this assumption is natural and is weaker than the MDDH one, we argue that it is equivalent to CDH.

We then propose the *Kernel Matrix DH Assumption* ($\mathcal{D}_{\ell,k}$-KerMDH). This new flexible assumption states that, given some matrix $[\mathbf{A}]$, $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}$ for some $\ell > k$, it is hard to find a vector $[\boldsymbol{v}] \in \mathbb{G}^\ell$ in the kernel of $\mathbf{A}^\top$. We observe that for some special instances of $\mathcal{D}_{\ell,k}$, this assumption has appeared in the literature in [2,18,19,27,32] under different names, like *Simultaneous Pairing, Simultaneous Double Pairing (SDP in the following), Simultaneous Triple Pairing, 1-Flexible CDH, 1-Flexible Square CDH.* Thus, the new KerMDH Assumption allows us to organize and give a unified view on several useful assumptions.

[3] More specifically, we are referring to the lower bounds on the image size of a projecting bilinear map of [38] which were obtained in Freeman model [13]. The results of [21] by-passed this lower bounds allowing to save on pairing operations for projecting maps in prime order groups.

[4] In the cryptographic literature we sometimes find the term "strong" as an alternative to "flexible", like the Strong RSA or the Strong DDH.

This suggests that the KerMDH Assumption (and not the MCDH one) is the right computational analogue of the MDDH framework. Indeed, for any matrix distribution the $\mathcal{D}_{\ell,k}$-MDDH Assumption implies the corresponding $\mathcal{D}_{\ell,k}$-KerMDH Assumption. As a unifying algebraic framework, it offers the advantages mentioned above: it highlights the algebraic structure of any construction based on it, and it allows writing many instantiations of a given scheme in a compact way.

The power of Kernel Assumptions. At Eurocrypt 2015, our KerMDH Assumptions were applied to design simpler QA-NIZK proofs of membership in linear spaces [26]. They have also been used to give more efficient constructions of structure preserving signatures [25], to generalize and simplify the results on quasi-adaptive aggregation of Groth-Sahai proofs [17] (given originally in [24]) and to construct a tightly secure QA-NIZK argument for linear subspaces with unbounded simulation soundness in [15]. The power of a KerMDH Assumption is that it allows to guarantee uniqueness. This has been used by Kiltz and Wee [26], for instance, to compile some secret key primitives to the public key setting. Indeed, Kiltz and Wee [26] modify a hash proof system (which is only designated verifier) to allow public verification (a QA-NIZK proof of membership). In a hash proof system for membership in some linear subspace of $\mathbb{G}^n$ spanned by the columns of some matrix $[\mathbf{M}]$, the public information is $[\mathbf{M}^\top\mathbf{K}]$, for some secret matrix $\mathbf{K}$, and given the proof $[\boldsymbol{\pi}]$ that $[\boldsymbol{y}]$ is in the subspace, verification tests if $[\boldsymbol{\pi}] \stackrel{?}{=} [\mathbf{y}^\top\mathbf{K}]$.

The core argument to compile this to a public key primitive is that given $([\mathbf{A}], [\mathbf{KA}])$, $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}$ and any pair $[\boldsymbol{y}], [\boldsymbol{\pi}]$, the previous test is equivalent to $e([\boldsymbol{\pi}^\top], [\mathbf{A}]) = e([\boldsymbol{y}^\top], [\mathbf{KA}])$, under the $\mathcal{D}_{\ell,k}$-KerMDH Assumption. Indeed,

$$e([\boldsymbol{\pi}^\top], [\mathbf{A}]) = e([\boldsymbol{y}^\top], [\mathbf{KA}]) \iff e([\boldsymbol{\pi}^\top - \boldsymbol{y}^\top\mathbf{K}], [\mathbf{A}]) = [\mathbf{0}] \stackrel{\mathcal{D}_{\ell,k}\text{-KerMDH}}{\Longrightarrow} [\boldsymbol{\pi}] = [\boldsymbol{y}^\top\mathbf{K}]. \tag{1}$$

That is, although potentially there are many possible proofs which satisfy the public verification equation (left hand side of Equation (1)), the $\mathcal{D}_{\ell,k}$-KerMDH Assumption guarantees that only one of them is efficiently computable, so verification gives the same guarantees as in the private key setting (right hand side of Equation (1)). This property is also used in a very similar way in [15] and also in the context of structure preserving signatures in [25]. In Section 6 we use it to argue that, of all the possible openings of a commitment, only one is efficiently computable, *i.e.* to prove computational soundness of a commitment scheme. Moreover, some previous works, notably in the design of structure preserving cryptographic primitives [1,2,3,31], implicitly used this property for one specific KerMDH Assumption: the Simultaneous (Double) Pairing Assumption.

On the other hand, we have already discussed the importance of having a precise and clear language when talking about cryptographic assumptions. This justifies the introduction of a framework specific to computational assumptions, because one should properly refer to the assumption on which security is actually based, rather than just saying "security is based on an assumption weaker than $\mathcal{D}_{\ell,k}$-MDDH". A part from being imprecise, a problem with such a statement is that might lead to arbitrary, not optimal choices. For example, the signature scheme of [30] is based on the SDP Assumption but a slight modification of it can be based on the $\mathcal{L}_2$-KerMDH Assumption. If the security guarantee is "the assumption is weaker than 2-Lin" then the modified scheme achieves shorter public key and more efficient verification with no loss in security. Further, the claim that security is based on the MDDH decisional assumptions when only computational ones are necessary might give the impression that a certain tradeoff is in place when this is not known to be the case. For instance, Jutla and Roy [24] construct constant-size QA-NIZK arguments of membership in linear spaces under what they call the "Switching Lemma", which is proven under a certain $\mathcal{D}_{k+1,k}$-MDDH Assumption. However, a close look at the proof reveals that in fact it is based on the corresponding $\mathcal{D}_{k+1,k}$-KerMDH Assumption[5]. For these assumptions, prior to our work, it was unclear whether the choice of larger k gives any additional guarantees.

[5] To see this, note that in the proof of their "Switching Lemma" on which soundness is based, they use the output of the adversary to decide if $\boldsymbol{f} \stackrel{?}{\in} \operatorname{Im}\mathbf{A}$, $\mathbf{A} \leftarrow \mathcal{RL}_k$, by checking whether $[\boldsymbol{f}]$ is orthogonal to the adversary's output (equation (1), proof of Lemma 1, [24], full version), and where $\mathcal{RL}_k$ is the matrix distribution of Section 3.5.

Strictly Increasing Families of Kernel Assumptions. An important problem is that it is not clear whether there are increasingly weaker families of KerMDH Assumptions. That is, some decisional assumptions families parameterized by k like the k-Lin Assumption are known to be strictly increasingly weaker. The proof of increasing hardness is more or less immediate and the term *strictly* follows from the fact that every two $\mathcal{D}_{\ell,k}$-MDDH and $\mathcal{D}_{\widetilde{\ell},\widetilde{k}}$-MDDH problems with $\widetilde{k} < k$ are separated by an oracle computing a k-linear map. For the computational case, increasing hardness is also not too difficult, but nothing is known about *strictly* increasing hardness (see Fig. 1). This means that, as opposed to the decisional case, prior to our work, for protocols based on KerMDH Assumptions there was no-known tradeoff between larger k (less efficiency) and security.

In this paper, we prove that the families of matrix distributions in [11], $\mathcal{U}_{\ell,k}$, $\mathcal{L}_k$, $\mathcal{SC}_k$, $\mathcal{C}_k$ and $\mathcal{RL}_k$, as well as a new distribution we propose in Section 7, the *circulant* family $\mathcal{CI}_{k,d}$, define families of kernel problems with increasing hardness. For this we show a tight reduction from the smaller to the larger problems in each family. Our main result (Theorem 2) is to prove that the hardness of these problems is *strictly* increasing. For this, we prove that there is no black-box reduction from the larger to the smaller problems in the multilinear generic group model. These new results correspond to the dotted arrows in Fig. 1.

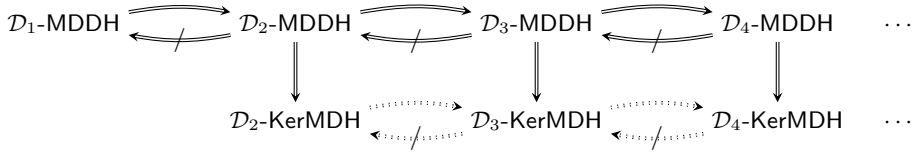

Fig. 1. Implication and separation results between Matrix Assumptions (dotted arrows correspond to the new results).

Having in mind that the computational problems we study in the paper are defined in a generic way, that is without specifying any particular group, the generic group approach arises naturally as the setting for the analysis of their hardness and reducibility relations. Otherwise, we would have to rely on specific properties of the representation of the elements of particular group families, not captured by the generic model.

The proof of Theorem 2 requires dealing with the notion of black-box reduction between flexible problems. A black-box reduction must work for any possible behavior of the oracle, but, contrary to the normal (unique answer) black-box reductions, here the oracle has to choose among the set of valid answers in every call. Ruling out the existence of a reduction implies that for any reduction there is an oracle behavior for which the reduction fails. This is specially subtle when dealing with multiple oracle calls. We think that the proof technique we introduce to deal with these issues can be considered as a contribution in itself and can potentially be used in future work.

Combining the black-box techniques and the generic group model is not new in the literature. For instance Dodis et al. [10] combine the black-box reductions and a generic model for the group $\mathbb{Z}_n^*$ to show some uninstantiability results for FDH-RSA signatures.

Theorem 2 supports the intuition that there is a tradeoff between the size of the matrix — which typically results in less efficiency — and the hardness of the KerMDH Problems, and justifies the generalization of several protocols to different choices of k given in [17,24,25,26].

Applications. The discussion of our results given so far should already highlight some of the advantages of using the new Kernel family of assumptions and the power of these new assumptions, which have already been used in compelling applications in follow-up work in [17,25,26]. To further illustrate the usefulness of the new framework, we apply it to the study of trapdoor commitments. First, we revisit the Pedersen commitment [37] to vectors of scalars and its extension to vectors of group elements of Abe *et al.* [2] in

bilinear groups. We unify these two constructions and we generalize to commit vectors of elements at each level $\mathbb{G}_r$, for any $0 \leq r \leq m$ under the extension of KerMDH Assumptions to the ideal m-graded encodings setting. In particular, when $m = 2$ we recover in a single construction as a special case both the original Pedersen and Abe *et al.* commitments.

The (generalized) Pedersen commitment maps vectors in $\mathbb{G}_r$ to vectors in $\mathbb{G}_{r+1}$, is perfectly hiding and computationally binding under any Kernel Assumption. In Section 6.3 we use it as a building block to construct a "group-to-group" commitment, which maps vectors in $\mathbb{G}_r$ to vectors in the same group $\mathbb{G}_r$. These commitments were defined in [3] because they are a good match to Groth-Sahai proofs. In [3], two constructions were given, one in asymmetric and the other in symmetric bilinear groups. Both are optimal in terms of commitment size and number of verification equations. Rather surprisingly, we show that both constructions in [3] are special instances of our group-to-group commitment for some specific matrix distributions.

A New Family of MDDH Assumptions of Optimal Representation Size. We also propose a new interesting family of Matrix distributions, the circulant matrix distribution, $\mathcal{CI}_{k,d}$, which defines new MDDH and KerMDH assumptions. This family generalizes the Symmetric Cascade Distribution ($\mathcal{SC}_k$) defined in [11] to matrices of size $\ell \times k$, $\ell = k + d > k + 1$. We prove that it has optimal representation size d independent of k among all matrix distributions of the same size. The case $\ell > k + 1$ typically arises when one considers commitments/encryption in which the message is a vector of group elements instead of a single group element and the representation size typically affects the size of the public parameters.

We prove the hardness of the $\mathcal{CI}_{k,d}$-KerMDH Problem, by proving that the $\mathcal{CI}_{k,d}$-MDDH Problem is generically hard in k-linear groups. Analyzing the hardness of a family of decisional problems (depending on a parameter k) can be rather involved, specially when an efficient k-linear map is supposed to exist. This is why in [11], the authors gave a practical criterion for generic hardness when $\ell = k + 1$ in terms of irreducibility of some polynomials involved in the description of the problem. This criterion was used then to prove the generic hardness of several families of MDDH Problems. To analyze the generic hardness of the $\mathcal{CI}_{k,d}$-MDDH Problem for any d, the techniques in [11] are not practical enough, and we need some extensions of these techniques for the case $\ell > k + 1$, recently introduced in [20]. However, we could not avoid the explicit computation of a large (but well-structured) Gröbner basis of an ideal associated to the matrix distribution. The new assumption can be used to instantiate the commitment schemes of Section 6 with shorter public parameters and improved efficiency.

2 Preliminaries

For $\lambda \in \mathbb{N}$, we write 1^λ for the string of λ ones. For a set S, $s \leftarrow S$ denotes the process of sampling an element s from S uniformly at random. For an algorithm $\mathcal{A}$, we write $z \leftarrow \mathcal{A}(x, y, \ldots)$ to indicate that $\mathcal{A}$ is a (probabilistic) algorithm that outputs z on input $(x, y, \ldots)$. For any two computational problems $\mathbb{P}_1$ and $\mathbb{P}_2$ we recall that $\mathbb{P}_1 \Rightarrow \mathbb{P}_2$ denotes the fact that $\mathbb{P}_1$ reduces to $\mathbb{P}_2$, and then '$\mathbb{P}_1$ is hard' $\Rightarrow$ '$\mathbb{P}_2$ is hard'. Thus, we will use '$\Rightarrow$' both for computational problems and for the corresponding hardness assumptions.

Let Gen denote a cyclic group instance generator, that is a probabilistic polynomial time (PPT) algorithm that on input 1^λ returns a description $\mathcal{G} = (\mathbb{G}, q, \mathcal{P})$ of a cyclic group $\mathbb{G}$ of order q for a λ-bit prime q and a generator $\mathcal{P}$ of $\mathbb{G}$. We use additive notation for $\mathbb{G}$ and its elements are $a\mathcal{P}$, for $a \in \mathbb{Z}_q$ and will be denoted as $[a] := a\mathcal{P}$. The notation extends to vectors and matrices in the natural way as $[\boldsymbol{v}] = (v_1\mathcal{P}, \ldots, v_n\mathcal{P})$ and $[\mathbf{A}] = (A_{ij}\mathcal{P})$. For a matrix $\mathbf{A} \in \mathbb{Z}_q^{\ell \times k}$, $\operatorname{Im} \mathbf{A}$ denotes the subspace of $\mathbb{Z}_q^\ell$ spanned by the columns of $\mathbf{A}$. Thus, $\operatorname{Im}[\mathbf{A}]$ is the corresponding subspace of $\mathbb{G}^\ell$.

2.1 Multilinear Maps

In the case of groups with a bilinear map, or more generally with a k-linear map for $k \geq 2$, we consider a generator producing the tuple $(e_k, \mathbb{G}_1, \mathbb{G}_k, q, \mathcal{P}_1, \mathcal{P}_k)$, where $\mathbb{G}_1, \mathbb{G}_k$ are cyclic groups of prime-order q, $\mathcal{P}_i$ is a generator of $\mathbb{G}_i$ and e_k is a non-degenerate efficiently computable k-linear map $e_k : \mathbb{G}_1^k \to \mathbb{G}_k$, such that $e_k(\mathcal{P}_1, \ldots, \mathcal{P}_1) = \mathcal{P}_k$. We actually consider graded encodings which offer a richer structure.

For any fixed $k \geq 1$, let MGen_k be a PPT algorithm that on input 1^λ returns a description of a graded encoding $\mathcal{MG}_k = (e, \mathbb{G}_1, \ldots, \mathbb{G}_k, q, \mathcal{P}_1, \ldots, \mathcal{P}_k)$, where $\mathbb{G}_1, \ldots, \mathbb{G}_k$ are cyclic groups of prime-order q, $\mathcal{P}_i$ is a generator of $\mathbb{G}_i$ and e is a collection of non-degenerate efficiently computable bilinear maps $e_{i,j} : \mathbb{G}_i \times \mathbb{G}_j \to \mathbb{G}_{i+j}$, for $i + j \leq k$, such that $e(\mathcal{P}_i, \mathcal{P}_j) = \mathcal{P}_{i+j}$. For simplicity we will omit the subindexes of e when they become clear from the context. Sometimes $\mathbb{G}_0$ is used to refer to $\mathbb{Z}_q$. For group elements we use the following implicit notation: for all $i = 1, \ldots, k$, $[a]_i := a\mathcal{P}_i$. The notation extends in a natural way to vectors and matrices and to linear algebra operations. We sometimes drop the index when referring to elements in $\mathbb{G}_1$, *i.e.*, $[a] := [a]_1 = a\mathcal{P}_1$. In particular, it holds that $e([a]_i, [b]_j) = [ab]_{i+j}$.

Additionally, for the asymmetric case, let AGen_2 be a PPT algorithm that on input 1^λ returns a description of an asymmetric bilinear group $\mathcal{AG}_2 = (e, \mathbb{G}, \mathbb{H}, \mathbb{T}, q, \mathcal{P}, \mathcal{Q})$, where $\mathbb{G}, \mathbb{H}, \mathbb{T}$ are cyclic groups of prime-order q, $\mathcal{P}$ is a generator of $\mathbb{G}$, $\mathcal{Q}$ is a generator of $\mathbb{H}$ and $e : \mathbb{G} \times \mathbb{H} \to \mathbb{T}$ is a non-degenerate, efficiently computable bilinear map. In this case we refer to group elements as: $[a]_G := a\mathcal{P}$, $[a]_H := a\mathcal{Q}$ and $[a]_T := ae(\mathcal{P}, \mathcal{Q})$.

3 A Generic Model For Groups With Graded Encodings

In this section we describe a (purely algebraic) generic model for the graded encodings in order to obtain meaningful results about the hardness and separations of computational problems. The model is an adaptation of Maurer's generic group model [33,34] including the k-graded encodings, but in a completely algebraic formulation that follows the ideas in [5,12,20]. Since the k-graded encodings functionality implies the k-linear group functionality, the former gives more power to the adversaries or reductions working within the corresponding generic model. This in particular means that non-existential results proven in the richer k-graded encodings generic model also imply the same results in the k-linear generic group model. Therefore, here we consider the former model. However, for the sake of simplicity, in other sections of this paper we will use the more common name 'multilinear generic group model' to refer to the k-graded encodings generic model.

In a first approach we consider Maurer's model adapted to the graded encodings functionality, but still not phrased in a purely algebraic language. In this model, an algorithm $\mathcal{A}^{\text{gen}}$ does not deal with proper group elements $[y]_a \in \mathbb{G}_a$, but only with labels (Y, a), and it has access to an additional oracle $\mathcal{O}^{\text{gen}}$ internally performing the group operations, so that $\mathcal{A}^{\text{gen}}$ cannot benefit from the particular way the group elements are represented.

As we need to handle elements in different groups, we will use the shorter vector notation $[\boldsymbol{x}]_{\boldsymbol{a}} = ([x_1]_{a_1}, \ldots, [x_\alpha]_{a_\alpha}) = (x_1\mathcal{P}_{a_1}, \ldots, x_\alpha\mathcal{P}_{a_\alpha}) \in \mathbb{G}_{a_1} \times \cdots \times \mathbb{G}_{a_\alpha}$. Note that the length of a vector of indices $\boldsymbol{a}$ is denoted by a corresponding Greek letter α. We will also use a tilde to denote variables containing only non-group elements (*i.e.*, elements not belonging to any of $\mathbb{G}_1, \ldots, \mathbb{G}_k$).

3.1 The Generic Model for Algorithms Without Additional Oracle Access

We first address the simple case of algorithms without access to additional oracles (i.e., only the oracle associated to the generic model will be considered).

Definition 1 (Generic Algorithm). *An algorithm $\mathcal{A}^{gen}$ is called generic (in the sense of the generic model for k-graded encodings) if, on start it receives as input some labels $(X_1, a_1), \ldots, (X_\alpha, a_\alpha)$ representing the vector of group elements $[\boldsymbol{x}]_{\boldsymbol{a}}$. The stateful oracle $\mathcal{O}^{gen}$ implementing the k-graded encodings initially holds a table $\mathcal{T}$ containing these group elements along with their corresponding labels. That is, every entry in $\mathcal{T}$ has the form $([x_i]_{a_i}, (X_i, a_i))$. Additionally, $\mathcal{A}^{gen}$ can receive some extra inputs corresponding to non-group elements, including for instance some size parameters. We denote these extra inputs as $\widetilde{x}$, which can be considered as a bit string. For each group $\mathbb{G}_a$, $a = 1, \ldots, k$, two additional labels $(0, a), (1, a)$, corresponding to the neutral element and the generator, are implicitly given to $\mathcal{A}^{gen}$. The size of the groups q is also implicitly given to $\mathcal{A}^{gen}$.*

During its execution $\mathcal{A}^{gen}$ can adaptively make the following queries to the oracle $\mathcal{O}^{gen}$:

- $\mathsf{GroupOp}((Y_1, a), (Y_2, a))$*: group operation in* $\mathbb{G}_a$ *for two previously issued labels in* $\mathbb{G}_a$ *resulting in a new label* (Y_3, a) *in* $\mathbb{G}_a$*. In addition,* $\mathcal{O}^{gen}$ *performs the real operation* $[y_3]_a = [y_1]_a + [y_2]_a$ *(using the group elements stored in* $\mathcal{T}$*) and adds the new entry* $([y_3]_a, (Y_3, a))$ *to* $\mathcal{T}$*.*
- $\mathsf{GroupInv}((Y, a))$*: idem. for group inversion in* $\mathbb{G}_a$*.*
- $\mathsf{GroupPair}((Y_1, a), (Y_2, b))$*: bilinear map for two previously issued labels in* $\mathbb{G}_a$ *and* $\mathbb{G}_b$*,* $a + b \leq k$*, resulting in a new label* $(Y_3, a + b)$ *in* $\mathbb{G}_{a+b}$*. The corresponding new entry is added to* $\mathcal{T}$*.*
- $\mathsf{GroupEqTest}((Y_1, a), (Y_2, a))$*: test two previously issued labels in* $\mathbb{G}_a$ *for equality of the corresponding group elements, resulting in a bit (*$1 =$ *equality).*

Every badly formed query (for instance, containing a label not previously issued by the oracle or as an input to $\mathcal{A}^{gen}$*) is answered with a special rejection symbol* $\perp$*.*

In the end, $\mathcal{A}^{gen}$ *outputs another set of labels* $(Y_1, b_1), \ldots, (Y_\beta, b_\beta)$ *(given at some time by the generic model oracle* $\mathcal{O}^{gen}$*) corresponding to group elements* $[\boldsymbol{y}]_{\boldsymbol{b}} = ([y_1]_{b_1}, \ldots, [y_\beta]_{b_\beta})$*, along with some non-group elements, denoted as* $\widetilde{y}$*, which is also a bit string. The final state of the oracle* $\mathcal{O}^{gen}$ *is defined to be the part of the internal table* $\mathcal{T}$ *consisting of the labels* $(Y_1, b_1), \ldots, (Y_\beta, b_\beta)$ *and their corresponding group elements* $[\boldsymbol{y}]_{\boldsymbol{b}}$*.*

Observe that all valid queries to $\mathcal{O}^{\text{gen}}$ can be properly managed by it with the help of the k-graded encodings. Indeed, all the group indices in the labels are forced to belong to the set $\{1, \ldots, k\}$.

The generic algorithm $\mathcal{A}^{\text{gen}}$ described above is not a proper algorithm, because it actually does not perform any computation with group elements but it only instructs the oracle $\mathcal{O}^{\text{gen}}$ to carry out the computations. However, one can obtain a real algorithm $\mathcal{A}$ from $\mathcal{A}^{\text{gen}}$ and a front-end in the following way:

1. The front-end intercepts $\mathcal{A}$'s input, replaces the group elements by labels and internally stores these group elements.
2. $\mathcal{A}^{\text{gen}}$ starts its execution and the front-end emulates the oracle $\mathcal{O}^{\text{gen}}$ for it, thus performing the actual computations with the internally stored group elements.
3. The front-end collects the output given by $\mathcal{A}^{\text{gen}}$ at the end of the execution and replaces the labels by the corresponding group elements to obtain the intended output for $\mathcal{A}$. If the front-end detects an invalid label (not previously issued by $\mathcal{O}^{\text{gen}}$ as the response of some oracle query), then the execution is aborted.

There is a technical problem with the non-group elements, since the value of $\widetilde{x}$ could give some unintended information to $\mathcal{A}^{\text{gen}}$ about $x_1, \ldots, x_\alpha$. For instance, we could set $\widetilde{x} = x_1$. In order to obtain meaningful results, in the following definition we capture some extra requirements for the generation of the inputs of $\mathcal{A}^{\text{gen}}$.

Definition 2 (Bounded Polynomial Source). *We call* D *a bounded polynomial source of degree* d *and with* δ *parameters if there exists a polynomial map* $f : \mathbb{Z}_q^\delta \to \mathbb{Z}_q^\alpha$ *of total degree* d *such that* D *outputs the evaluation of* f *at a uniformly distributed random point* $(t_1, \ldots, t_\delta) \in \mathbb{Z}_q^\delta$*.*[6]

Additionally, we consider the case that D *depends on an auxiliary input* $\tau \in \{0,1\}^*$*. In this case we require that for every value of* τ*,* D_τ *is a bounded polynomial source of degree* $\leq d$ *and for a fixed* δ *(i.e.,* d *and* δ *are independent of* τ*). When needed, we will refer to the polynomial map instance corresponding to* D_τ *explicitly as* f_τ*. However, for simplicity we will use* f *if it causes no ambiguity.*

We will consider from now on that the input of a generic algorithm comes from a bounded polynomial source D (i.e., the output $\boldsymbol{x}$ of D is conveniently encoded as group elements $[\boldsymbol{x}]_{\boldsymbol{a}}$), where the auxiliary input consists of the non-group elements $\widetilde{x}$. This extra requirement could be seen as a serious limitation of the generic model. However, the way the inputs are sampled is typically part of the specification of the problem to be solved by $\mathcal{A}^{\text{gen}}$. In particular, kernel problems dealt with in this paper are particular examples of bounded polynomial sources, where the auxiliary information consists of the size parameters k and ℓ of the associated matrix distribution.

[6] For simplicity, we assume that the coefficients of the polynomial map do not depend on q.

Following the usual step in generic group model proofs (see for instance [5,11,20]), we use polynomials as labels to group elements. Namely, labels in $\mathbb{G}_a$ are polynomials in $\mathbb{Z}_q[\boldsymbol{X}]$, where the algebraic variables $\boldsymbol{X} = (X_1, \ldots, X_\alpha)$ are just formal representations of the group elements in the input of $\mathcal{A}^{\text{gen}}$. We emphasize that the degree of these input labels (as polynomials in $\mathbb{Z}_q[\boldsymbol{X}]$) is one. Now in the oracle side, group operations are replaced by polynomial operations in the labels. Indeed, $\mathsf{GroupOp}((Y_1, a), (Y_2, a)) = (Y_1 + Y_2, a)$, $\mathsf{GroupInv}((Y, a)) = (-Y, a)$ and $\mathsf{GroupPair}((Y_1, a), (Y_2, b)) = (Y_1 Y_2, a + b)$. It is easy to see that for any valid label (Y, a), $\deg Y \leq a$. It clearly holds for the input group elements (since $\deg X_1 = \cdots = \deg X_\alpha = 1$ and all $a_1, \ldots, a_\alpha \geq 1$), and the inequality is preserved by $\mathsf{GroupOp}$, $\mathsf{GroupInv}$ and $\mathsf{GroupPair}$. In particular, the oracle call $\mathsf{GroupPair}((Y_1, a), (Y_2, b))$ results in the label $(Y_1 Y_2, a + b)$. Then the input inequalities $\deg Y_1 \leq a$ and $\deg Y_2 \leq b$ imply the output inequality $\deg(Y_1 Y_2) = \deg Y_1 + \deg Y_2 \leq a + b$. On the other hand, $\mathsf{GroupOp}((Y_1, a), (Y_2, a))$ outputs $(Y_1 + Y_2, a)$ and $\deg(Y_1 + Y_2) \leq \max(\deg Y_1, \deg Y_2) \leq a$. The same argument applies to $\mathsf{GroupInv}$.

This polynomial-based labelling is well-defined, since the group element $[y]_a$ corresponding to a label (Y, a) is exactly $[Y(x_1, \ldots, x_\alpha)]_a$. Therefore different group elements always receive different labels. On the other hand, two different labels (Y_1, a) and (Y_2, a) can be associated to the same group element $[y]_a$, thus implying that the polynomial $Y_1 - Y_2 \in \mathbb{Z}_q[\boldsymbol{X}]$ vanishes at the point $(x_1, \ldots, x_\alpha)$. These label collisions can occur either predictably, because of the way the point $(x_1, \ldots, x_\alpha)$ is sampled, or by the unexpected ability of $\mathcal{A}^{\text{gen}}$ to find them. An example of predictable collisions occurs when the source D produces $x_1 = t$ and $x_2 = -t$. Then, the oracle call $\mathsf{GroupEqTest}((X_1, a), \mathsf{GroupInv}((X_2, a)))$ always results in 1. This is something predictable by $\mathcal{A}^{\text{gen}}$ itself, assuming that it knows the map f. Notice that $\mathcal{A}^{\text{gen}}$ can predict all answers given by the oracle $\mathcal{O}^{\text{gen}}$ except for some $\mathsf{GroupEqTest}$ queries, that we call nontrivial.

Definition 3 (Nontrivial Equality Test Queries). *Given a bounded polynomial source D with polynomial map $f : \mathbb{Z}_q^\delta \to \mathbb{Z}_q^\alpha$ of total degree d and auxiliary information $\widetilde{\tau} \in \{0,1\}^*$, and a generic algorithm $\mathcal{A}^{gen}$, the oracle call $\mathsf{GroupEqTest}((Y_1, a), (Y_2, a))$ is called nontrivial (for some specific value of $\widetilde{\tau}$) if and only if $Y_1 \circ f_{\widetilde{\tau}} \neq Y_2 \circ f_{\widetilde{\tau}}$, as polynomials in $\mathbb{Z}_q[\boldsymbol{T}]$, where $\boldsymbol{T} = (T_1, \ldots, T_\delta)$ are the parameters of the bounded polynomial source D.*

Here we will assume that the description of the map f is fixed, and it can be used by $\mathcal{A}^{\text{gen}}$, and also the auxiliary input $\widetilde{\tau}$ is given to $\mathcal{A}^{\text{gen}}$. Clearly this would not be the case when dealing with decision problems, where a distinguisher aims at telling apart two possible problem instance distributions (i.e., there are two candidates for f). Nevertheless here we restrict ourselves to the study of search problems. At this point all the information $\mathcal{A}^{\text{gen}}$ can obtain from the oracle $\mathcal{O}^{\text{gen}}$ is via the nontrivial $\mathsf{GroupEqTest}$ queries. Indeed, $\mathcal{A}^{\text{gen}}$ can perform itself the same polynomial operations in the labels as $\mathcal{O}^{\text{gen}}$, and it can use the map f to predict the answers to the trivial $\mathsf{GroupEqTest}$ queries.

We now introduce a "purely algebraic" version of the generic model by slightly modifying the oracle $\mathcal{O}^{\text{gen}}$. To distinguish the two models, we denote the modified oracle as $\mathcal{O}^{\text{pa-gen}}$. In this new model we need to specify a bounded polynomial source which provides the input for $\mathcal{A}^{\text{gen}}$.

Definition 4 (Purely-Algebraic Generic Algorithm). [7] *Given a bounded polynomial source D with auxiliary input $\widetilde{x} \in \{0,1\}^*$ and polynomial map $f : \mathbb{Z}_q^\delta \to \mathbb{Z}_q^\alpha$ of total degree d, an algorithm $\mathcal{A}^{gen}$ is called generic (in the sense of the purely-algebraic generic model for k-graded encodings) with respect to D if it follows Definition 1, except that it receives as input $\widetilde{x}$ and the output of D (this was not required before) and it has access to the oracle $\mathcal{O}^{pa\text{-}gen}$, which differs from $\mathcal{O}^{gen}$ in that:*

- *The internal table $\mathcal{T}$ now contains entries $(X_i \circ f_{\widetilde{x}}, (X_i, a_i))$, where the group element has been replaced by the "internal" polynomial $\widehat{X}_i = X_i \circ f_{\widetilde{x}} \in \mathbb{Z}_q[\boldsymbol{T}]$.*
- *For valid labels $\mathsf{GroupEqTest}((Y_1, a), (Y_2, a))$ outputs 1 if and only if the corresponding internal polynomials are equal, that is, $\widehat{Y}_1 = \widehat{Y}_2$, as polynomials in $\mathbb{Z}_q[\boldsymbol{T}]$.*
- *$\mathcal{O}^{pa\text{-}gen}$ does not store any group element and it does not perform any real group operation. Instead, once $\mathcal{A}^{gen}$ has finished its execution by outputting the labels $(Y_1, b_1), \ldots, (Y_\beta, b_\beta)$, $\mathcal{O}^{pa\text{-}gen}$ samples a*

[7] We are actually defining a new generic model. Any particular generic algorithm can work in both models.

random point $\boldsymbol{t} \in \mathbb{Z}_q^\delta$ (in the same way as D would did it) and then it evaluates the corresponding internal polynomials $\widehat{Y}_1, \ldots, \widehat{Y}_\beta$ extracted from $\mathcal{T}$ at the sampled point $\boldsymbol{t} \in \mathbb{Z}_q^\delta$, that is $\boldsymbol{y} = \widehat{\boldsymbol{Y}}(\boldsymbol{t}) = \boldsymbol{Y} \circ f_{\widetilde{x}}(\boldsymbol{t})$. Finally, $\mathcal{O}^{pa\text{-}gen}$ computes the output group elements $[\boldsymbol{y}]_{\boldsymbol{b}}$ from their discrete logarithms $\boldsymbol{y}$.

Notice that in the plain (non-algebraic) generic model $\boldsymbol{t}$ is also sampled from D, but at the very beginning of the game. Therefore, it is self-evident that the representation of the group elements is irrelevant for $\mathcal{A}^{\text{gen}}$ in the purely-algebraic model, since the group elements themselves are never created. Is can be proved that, with this change in the generic model, the behavior of $\mathcal{A}^{\text{gen}}$ can only change negligibly, assuming that $\boldsymbol{x}$ comes from a bounded polynomial source and q is exponentially large. This is just a standard argument used in proofs in the generic group model. The difference between the original model and its purely algebraic reformulation is typically upper-bounded by using Schwartz-Zippel Lemma and the union bound, as shown for instance in [5,12,20]. More precisely,

Lemma 1. *Let $\mathcal{A}^{gen}$ be a generic algorithm receiving inputs from a bounded polynomial source D with auxiliary input $\widetilde{x} \in \{0,1\}^*$ and polynomial map $f : \mathbb{Z}_q^\delta \to \mathbb{Z}_q^\alpha$ of total degree d. Then, for exponentially large q, the difference between the success probabilities of $\mathcal{A}^{gen}$ in the generic model for k-graded encodings interacting with either $\mathcal{O}^{gen}$ or $\mathcal{O}^{pa\text{-}gen}$ is negligible. This holds even when $\widetilde{x}$ is correlated with the random tape of $\mathcal{A}^{gen}$.*

Proof. The only way in which the plain generic model and its purely algebraic counterpart can differ is when $\mathcal{A}^{\text{gen}}$ makes a nontrivial $\mathsf{GroupEqTest}$ query. Namely, for specific values of $\widetilde{x}$ and the random tape of $\mathcal{A}^{\text{gen}}$, the algorithm is able to find two valid labels (Y_1, a) and (Y_2, a) such that their corresponding internal polynomials are different, that is $Y_1 \circ f_{\widetilde{x}} \neq Y_2 \circ f_{\widetilde{x}}$ as polynomials in $\mathbb{Z}_q[\boldsymbol{T}]$, while the answer of $\mathsf{GroupEqTest}((Y_1, a), (Y_2, a))$ is 1 in the plain (non-algebraic) generic model. Equivalently, $\mathcal{A}^{\text{gen}}$ is able to find a nonzero polynomial $Z = Y_1 \circ f_{\widetilde{x}} - Y_2 \circ f_{\widetilde{x}} = (Y_1 - Y_2) \circ f_{\widetilde{x}} \in \mathbb{Z}_q[\boldsymbol{T}]$ that vanishes at the random point $\boldsymbol{t} = (t_1, \ldots, t_\delta)$ internally sampled by D.

For every individual choice of $\widetilde{x}$ and the random tape of $\mathcal{A}^{\text{gen}}$, the probability that this event occurs is at most $\frac{dk}{q}$ by Schwartz-Zippel Lemma, because $\deg((Y_1 - Y_2) \circ f_{\widetilde{x}}) \leq \deg(Y_1 - Y_2) \deg f_{\widetilde{x}} \leq kd$. Finally, if Q_{gen} is the number of queries to $\mathcal{O}^{\text{gen}}$ made by $\mathcal{A}^{\text{gen}}$ other than $\mathsf{GroupEqTest}$ queries, then $\mathcal{A}^{\text{gen}}$ receives at most $Q_{\text{gen}} + \alpha + 2k$ different labels (including the input labels and the implicitly given ones). By a straightforward hybrid argument (or simply using the union bound) the difference between the success probabilities of $\mathcal{A}^{\text{gen}}$ in the two models is upper bounded by

$$\Pr[\mathsf{Bad}] \leq \binom{Q_{\text{gen}} + \alpha + 2k}{2} \frac{dk}{q},$$

which is negligible if q is exponentially large. Notice that this bound applies to every choice of $\widetilde{x}$ and the random tape of $\mathcal{A}^{\text{gen}}$, which means in particular that the result holds even if these two objects are not independent random variables. □

We stress that this bound is computed for any value of $\widetilde{x}$ and, by Definition 2, the parameters $(t_1, \ldots, t_\delta)$ are sampled independently of $\widetilde{x}$. This captures the fact that $\mathcal{A}^{\text{gen}}$ cannot receive unintended information about $x_1, \ldots, x_\alpha$ via the non-group element inputs.

Once we have proved that generic algorithms receiving inputs from a bounded polynomial source performs similarly in both the original generic model and the purely-algebraic one, we now assume that both models are the same, and choose the simplest one: the purely-algebraic generic model. The simplicity of the model comes from the fact that $\mathcal{A}^{\text{gen}}$ receives absolutely no information about the group element inputs $(x_1, \ldots, x_\alpha)$. Actually $\mathcal{A}^{\text{gen}}$ does not perform the computations but it just instructs $\mathcal{O}^{\text{pa-gen}}$ to do so. Namely, from the particular values of $\widetilde{x}$ and a random tape, $\mathcal{A}^{\text{gen}}$ computes the polynomials $Y_1, \ldots, Y_\beta$ and gives them to the oracle (incrementally, via the oracle calls). Then $\mathcal{O}^{\text{pa-gen}}$ samples $(x_1, \ldots, x_\alpha)$ from D given $\widetilde{x}$ and evaluates the polynomials at the sampled point. Nevertheless, $\mathcal{A}^{\text{gen}}$ still computes the non-group elements in the output $\widetilde{y}$. But then $\widetilde{y}$ can only depend on $\widetilde{x}$ and the random tape of $\mathcal{A}^{\text{gen}}$, which are the only inputs received by $\mathcal{A}^{\text{gen}}$ (in the purely algebraic model the oracle $\mathcal{O}^{\text{pa-gen}}$ gives no information to $\mathcal{A}^{\text{gen}}$). Then, for a fixed value of $\widetilde{x}$ and for every choice of the random tape of $\mathcal{A}^{\text{gen}}$, $\mathcal{A}^{\text{gen}}$ runs deterministically, and the polynomials $Y_1, \ldots, Y_\beta$ are used for all possible choices of $(x_1, \ldots, x_\alpha)$ (drawn from $D_{\widetilde{x}}$).

3.2 Algebraic Algorithms Without Oracle Access

We now capture the previous considerations into a formal definition of algebraic algorithm (no to be confused with a generic algorithm).

Definition 5 (Algebraic Algorithm). *An algorithm $\mathcal{A}^{alg}$ is called algebraic (in the sense of the generic model for k-graded encodings) if on start it receives as input the group elements $[x_1]_{a_1}, \ldots, [x_\alpha]_{a_\alpha}$ and an auxiliary input $\widetilde{x}$ (the size of the groups q and the group generators $[1]_1, \ldots [1]_k$ are also given implicitly to $\mathcal{A}^{alg}$), and its output $([y_1]_{b_1}, \ldots, [y_\beta]_{b_\beta}, \widetilde{y})$ is computed by the following two stages:*

1. *From $\widetilde{x}$ and the random tape, $\mathcal{A}^{alg}$ computes polynomials $Y_1, \ldots, Y_\beta \in \mathbb{Z}_q[\boldsymbol{X}]$ such that $\deg Y_i \leq b_i$. In addition $\mathcal{A}^{alg}$ computes the auxiliary output $\widetilde{y}$.*
2. *$\mathcal{A}^{alg}$ uses the computed polynomials and the k-graded encodings to compute $[y_1]_{b_1}, \ldots, [y_\beta]_{b_\beta}$ such that $y_j = Y_j(x_1, \ldots, x_\alpha)$, for $j = 1, \ldots, \beta$.*

Observe that $\mathcal{A}^{\text{alg}}$ is an actual algorithm with a very restricted structure, and it do not require the assistance of any oracle, as it makes use of the real k-graded encodings. However algebraic algorithms are in some sense equivalent to generic algorithms as stated below.

Lemma 2. *Given a generic algorithm $\mathcal{A}^{gen}$ as described in Definition 1 with inputs $(X_1, a_1), \ldots, (X_\alpha, a_\alpha)$ and $\widetilde{x}$, and a bounded polynomial source D, as described in Definition 2, with auxiliary input $\widetilde{x} \in \{0,1\}^*$ and polynomial map $f : \mathbb{Z}_q^\delta \to \mathbb{Z}_q^\alpha$ of total degree d, there exists an algebraic algorithm $\mathcal{A}^{alg}$ that produces the same outputs as $\mathcal{A}^{gen}$ in the purely-algebraic k-graded encoding generic model (i.e., interacting with the oracle $\mathcal{O}^{pa\text{-}gen}$) with input group elements sampled from D.*

Proof. $\mathcal{A}^{\text{alg}}$ receives the inputs $[x_1]_{a_1}, \ldots, [x_\alpha]_{a_\alpha}$ and an auxiliary input $\widetilde{x}$, where $(x_1, \ldots, x_\alpha)$ are sampled from $D_{\widetilde{x}}$. Then it simulates the oracle $\mathcal{O}^{\text{pa-gen}}$ for $\mathcal{A}^{\text{gen}}$ following Definition 4. Namely, $\mathcal{A}^{\text{alg}}$ starts the execution of $\mathcal{A}^{\text{gen}}$ with the inputs $((X_1, a_1), \ldots, (X_\alpha, a_\alpha), \widetilde{x})$, where $a_1, \ldots, a_\alpha$ are the values in the definition of $\mathcal{A}^{\text{gen}}$ and $\boldsymbol{X} = (X_1, \ldots, X_\alpha)$ are algebraic variables. Then all the oracle queries made by $\mathcal{A}^{\text{gen}}$ are simulated in a straightforward way, by performing the corresponding polynomial operations in the labels, except for the $\mathsf{GroupEqTest}$ queries. A valid query $\mathsf{GroupEqTest}((Y_1, a), (Y_2, a))$ is answered with 1 if and only if the corresponding internal polynomials are equal, namely $Y_1 \circ f_{\widetilde{x}} = Y_2 \circ f_{\widetilde{x}}$ as polynomials in $\mathbb{Z}_q[\boldsymbol{T}]$. This first stage of $\mathcal{A}^{\text{alg}}$ ends up by collecting the output polynomials $Y_1, \ldots, Y_\beta \in \mathbb{Z}_q[\boldsymbol{X}]$ and the auxiliary output $\widetilde{y}$ given by $\mathcal{A}^{\text{gen}}$. The second stage follows exactly Definition 5.

The simulation performed by $\mathcal{A}^{\text{alg}}$ is perfect and it fulfils the requirements given in Definition 5, since the values of $[x_1]_{a_1}, \ldots, [x_\alpha]_{a_\alpha}$ are only used in the second stage. The degree requirement $\deg Y_i \leq b_i$ as polynomials in $\mathbb{Z}_q[\boldsymbol{X}]$ directly comes from the above discussion about the degrees of the polynomial labels across the oracle calls. Recall that $\deg X_i = 1$ by definition, as the labels are polynomials in $\mathbb{Z}_q[\boldsymbol{X}]$. □

As a consequence, for any generic algorithm $\mathcal{A}^{\text{gen}}$ in the original (i.e. non-algebraic) generic model for k-graded encodings in exponentially large groups receiving inputs from a bounded polynomial source, there exists an algebraic algorithm $\mathcal{A}^{\text{alg}}$ with identical performance except for a negligible probability. Moreover, the very restricted structure of an algebraic algorithm allows us to state the following technical result.

Lemma 3. *Let $\mathcal{A}^{alg}$ be an algebraic algorithm. Let $([\boldsymbol{x}]_{\boldsymbol{a}}, \widetilde{x})$ and $([\boldsymbol{y}]_{\boldsymbol{b}}, \widetilde{y})$ respectively be the input and output of $\mathcal{A}^{alg}$. Then, for every choice of $\widetilde{x} \in \{0,1\}^*$ and any choice of the random tape of $\mathcal{A}^{alg}$, there exist polynomials $Y_1, \ldots, Y_\beta \in \mathbb{Z}_q[\boldsymbol{X}]$ of degree upper bounded by $b_1, \ldots, b_\beta$ such that $\boldsymbol{y} = \boldsymbol{Y}(\boldsymbol{x})$, for all $\boldsymbol{x} \in \mathbb{Z}_q^\alpha$, where $\boldsymbol{Y} = (Y_1, \ldots, Y_\beta)$. Moreover, $\widetilde{y}$ only depends on $\widetilde{x}$ and the random tape of $\mathcal{A}^{alg}$ (i.e., it does not depend on $\boldsymbol{x}$).*

Proof. The proof of the lemma comes straightforward from the structure given in Definition 5. Indeed, the polynomials $Y_1, \ldots, Y_\beta$ and the auxiliary output $\widetilde{y}$ are exactly the ones computed by $\mathcal{A}^{\text{alg}}$ in the first stage of its execution. Thus, they directly fulfil the three requirements: $\deg Y_i \leq b_i$, $y_i = Y_i(x_1, \ldots, x_\alpha)$ and $\widetilde{y}$ only can depend on $\widetilde{x}$ and the random tape of $\mathcal{A}^{\text{alg}}$. □

In addition, algebraic algorithms and bounded polynomial sources can be related in the following way:

Lemma 4. *The composition of an algebraic algorithm, that only outputs group elements, with a bounded polynomial source is another bounded polynomial source.*

Proof. Let $\mathcal{A}^{\mathrm{alg}}$ be an algebraic algorithm with input $([\boldsymbol{x}]_{\boldsymbol{a}}, \widetilde{x})$ and output $([\boldsymbol{y}]_{\boldsymbol{b}})$. Let us assume that $\boldsymbol{x}$ is produced by a bounded polynomial source D with auxiliary input τ and polynomial map $f_\tau : \mathbb{Z}_q^\delta \to \mathbb{Z}_q^\alpha$ of total degree d. For any choice of τ, $\widetilde{x}$ and the random tape of $\mathcal{A}^{\mathrm{alg}}$, according to Lemma 3 there exists a polynomial map $\boldsymbol{Y} = (Y_1, \ldots, Y_\beta)$ of degree at most k (since for all $j = 1, \ldots, \beta$, $\deg Y_j \leq b_j \leq k$) such that $\boldsymbol{y} = \boldsymbol{Y}(\boldsymbol{x})$, for all $\boldsymbol{x} \in \mathbb{Z}_q^\alpha$. Therefore, $\boldsymbol{y} = \boldsymbol{Y} \circ f_\tau(\boldsymbol{t})$ for all $\boldsymbol{t} \in \mathbb{Z}_q^\delta$. This actually shows that $\boldsymbol{y}$ behaves exactly as the output of a bounded polynomial source with auxiliary input $\tau' = (\tau, \widetilde{x}, \$)$, where $\$$ denotes the random tape of $\mathcal{A}^{\mathrm{alg}}$, and polynomial map $f'_{\tau'} : \mathbb{Z}_q^\delta \to \mathbb{Z}_q^\beta$ such that $f' = Y \circ f$. The degree of f' is upper bounded by $\deg Y \circ f \leq \deg Y \deg f \leq kd$. □

3.3 A Comment on Generic Hardness

As usual, the proposed generic model reduces the analysis of the hardness of some problems to solving a merely algebraic problem related to polynomials. As an example, consider a computational problem $\mathcal{P}$ which instances are entirely described by some group elements in the base group $\mathbb{G}_1$, $[\boldsymbol{x}] \leftarrow \mathcal{P}.\mathsf{InstGen}(1^\lambda)$, and its solutions are also described by some group elements $[\boldsymbol{y}]_{\boldsymbol{b}} \in \mathcal{P}.\mathsf{Sol}([\boldsymbol{x}])$. We also assume that $\mathcal{P}.\mathsf{InstGen}$ just samples $\boldsymbol{x}$ by evaluating polynomial functions of constant degree at a random point. Then, $\mathcal{P}$ is generically hard if and only if for all (randomized) polynomials $Y_1, \ldots, Y_\beta \in \mathbb{Z}_q[\boldsymbol{X}]$ of degrees upper bounded by $b_1, \ldots, b_\beta$ respectively,

$$\Pr([\boldsymbol{y}]_{\boldsymbol{b}} \in \mathcal{P}.\mathsf{Sol}([\boldsymbol{x}]) : \ [\boldsymbol{x}] \leftarrow \mathcal{P}.\mathsf{InstGen}(1^\lambda), \ \boldsymbol{y} = \boldsymbol{Y}(\boldsymbol{x})) \in \mathit{negl}(\lambda)$$

where $\boldsymbol{Y} = (Y_1, \ldots, Y_m)$ and the probability is computed with respect the random coins of the instance generator and the randomized polynomials.[8] In a few words, this means that the set $\mathcal{P}.\mathsf{Sol}([\boldsymbol{x}])$ cannot be hit by polynomials of the given degree evaluated at $\boldsymbol{x}$.

3.4 Adding Oracles to the Model

The previous model extends naturally to algorithms with oracle access (*e.g.*, black-box reductions) but only when the oracles fit well into the generic model. Let us introduce the notion of an algebraic algorithm $\mathcal{A}^{\mathcal{O}}$, with oracle access to $\mathcal{O}$.

Definition 6 (Algebraic Oracle Algorithm). *An oracle algorithm $\mathcal{A}^{\mathcal{O}}$ is called algebraic (in the sense of the purely-algebraic generic model for k-graded encodings) if on start it receives as input the group elements $[\boldsymbol{x}]_{\boldsymbol{a}} = ([x_1]_{a_1}, \ldots, [x_\alpha]_{a_\alpha})$ and an auxiliary input $\widetilde{x}$. Next, it sets $[\boldsymbol{z}]_{\boldsymbol{c}} = [\boldsymbol{x}]_{\boldsymbol{a}}$ and $\widetilde{z} = \widetilde{x}$ as inputs for the first iteration and proceeds as follows:*

1. *From $\widetilde{z}$ and the random tape, it computes polynomials $Y_1, \ldots, Y_\beta, U_1, \ldots, U_\delta \in \mathbb{Z}_q[\boldsymbol{Z}]$ such that $\deg Y_i \leq b_i$ and $\deg U_i \leq d_i$, for some fixed vectors of indices $(b_1, \ldots, b_\beta)$ and $(d_1, \ldots, d_\delta)$.[9] In addition it computes some non-group elements $(\widetilde{y}, \widetilde{u}, \widetilde{stop})$.*
2. *The algorithm uses the polynomials $Y_1, \ldots, Y_\beta, U_1, \ldots, U_\delta$ and the k-graded encodings to compute $[y_1]_{b_1}, \ldots, [y_\beta]_{b_\beta}, [u_1]_{d_1}, \ldots, [u_\delta]_{d_\delta}$ such that $y_i = Y_i(\boldsymbol{z})$, for $i = 1, \ldots, \beta$ and $u_j = U_j(\boldsymbol{z})$, for $j = 1, \ldots, \delta$.*[10]

[8] We can similarly deal with problems with non-group elements both in the instance description and the solution, but this would require a more sophisticated formalization, in which both the polynomials and the non-group elements in the solution could depend on the non-group elements in the instance, but in an efficient way.

[9] All the index vectors except for $\boldsymbol{d}$ and $\boldsymbol{e}$ can indeed change in each iteration.

[10] These computations must be always possible, which imposses some limitations in the degree and the particular form of the polynomials, depending on the group indices $c_1, \ldots, c_\gamma$. For instance, if $c_1 = k$ then Z_1 can only appear in a linear terms in the polynomials. These limitations can be easily modelled by assigning to each formal variable Z_i a degree c_i and then allowing only a polynomial Y_j of total degree at most d_j for a group element $[y_j]_{d_j}$. However, this notion of degree is not the one that we need in Schwartz-Zippel lemma. Thus, we dropped this discussion in the exposition for the sake of readability.

3. *If* $\widetilde{stop} = 1$ *then the algorithm finishes its execution by outputting* $([y_1]_{b_1}, \ldots, [y_\beta]_{b_\beta}, \widetilde{y})$.[11]
4. *Otherwise, the oracle* $\mathcal{O}$ *is queried with* $([u_1]_{d_1}, \ldots, [u_\delta]_{d_\delta}, \widetilde{u})$. *Let the answer given by the oracle be* $([v_1]_{e_1}, \ldots, [v_\epsilon]_{e_\epsilon}, \widetilde{v})$, *for some fixed vector of indices* $(e_1, \ldots, e_\epsilon)$.
5. *The algorithm sets* $[\boldsymbol{z}]_{\boldsymbol{c}} = ([\boldsymbol{y}]_{\boldsymbol{b}}, [\boldsymbol{v}]_{\boldsymbol{e}})$ *and* $\widetilde{z} = (\widetilde{y}, \widetilde{v})$ *as inputs for the next iteration, and continues.*

Observe that the algebraic oracle algorithm is split into sections between oracle calls. All state information between sections is contained in the iteration input/output variables $[\boldsymbol{z}]_{\boldsymbol{c}}$ and $\widetilde{z}$. Each section is required to behave as a standalone algebraic algorithm, but no special requirement is imposed to the oracle $\mathcal{O}$. However, to make this definition useful, the oracle must preserve the algebraic properties of the different sections, like properly separating group elements from non-group elements. Indeed, a completely arbitrary oracle (specified in the plain model) could have access to the internal representation of the group elements, and then it could leak some information about the group elements that is outside the generic model. Thus, we will impose the very limiting constraint that the oracles are also "algebraic", meaning that the oracle's input/output behavior respects the one-wayness of the graded encodings,[12] and they only perform "polynomial operations" on the group elements.

Definition 7 (Algebraic Oracle). *Let* $([\boldsymbol{u}]_{\boldsymbol{d}}, \widetilde{u})$ *and* $([\boldsymbol{v}]_{\boldsymbol{e}}, \widetilde{v})$ *respectively be a query to an oracle* $\mathcal{O}$ *and its corresponding answer, where* $\widetilde{u}$ *and* $\widetilde{v}$ *contain the respective non-group elements. The oracle* $\mathcal{O}$ *is called algebraic if for any choice of* $\widetilde{u}$ *there exist polynomials* $V_1, \ldots, V_\epsilon \in \mathbb{Z}_q[\boldsymbol{U}, \boldsymbol{R}]$, $\boldsymbol{R} = (R_1, \ldots, R_\rho)$, *of constant degree (i.e., not depending on* $\widetilde{u}$, q *or any security parameter) such that*

- *for the specific choice of* $\widetilde{u}$, $v_i = V_i(\boldsymbol{u}, \boldsymbol{r})$, $i = 1, \ldots, \epsilon$, *for all* $\boldsymbol{u} \in \mathbb{Z}_q^\epsilon$ *and* $\boldsymbol{r} \in \mathbb{Z}_q^\rho$, *where* $\boldsymbol{r} = (r_1, \ldots, r_\rho)$ *are random parameters defined and uniformly sampled by the oracle,*
- $\widetilde{v}$ *does not depend on* $\boldsymbol{u}, \boldsymbol{r}$ *(thus,* $\boldsymbol{r}$ *can only have influence in the group elements in the answer),*
- *the polynomial* V_j *does not depend on any variable* U_i *such that* $e_j < d_i$ *(in order to preserve the one-wayness of the graded encodings).*

The parameters $\boldsymbol{r}$ capture the behavior of an oracle solving a problem with many solutions (called here a "flexible" problem), and they are independent across different oracle calls, meaning that the oracle is stateless. Observe that the first two requirements in the definition mean that $\boldsymbol{v}$ depends algebraically on $\boldsymbol{u}, \boldsymbol{r}$ and no extra information about $\boldsymbol{u}, \boldsymbol{r}$ can be leaked through $\widetilde{v}$. Removing any of these requirements from the definition results in that a generic algorithm using such an oracle will no longer be algebraically generic. Also notice that after a call to an algebraic oracle, there is no guarantee that labels (Y, a) fulfil the bound $\deg Y \leq a$.

Although the notion of algebraic oracle looks very limiting, it is general enough for our purposes. Notice for example that the notion of algebraic oracle excludes a Discrete Logarithm oracle, as it destroys the one-wayness property of the graded encodings, but on the other hand, oracles solving CDH or the Computational Bilinear Diffie-Hellman problem fit well in the definition. As an example, consider an oracle solving CDH in $\mathbb{G}_1$. The input here is $([\boldsymbol{u}]_{\boldsymbol{d}}, \widetilde{u}) = ([u_1]_1, [u_2]_1)$ while the output is $([\boldsymbol{v}]_{\boldsymbol{e}}, \widetilde{v}) = ([u_1 u_2]_1)$. This oracle needs no non-group elements and there are also no internal parameters involved, since CDH is not a flexible problem. Observe that the oracle can be modelled by a single polynomial $V_1(\boldsymbol{U}, \boldsymbol{R}) = V_1(U_1, U_2) = U_1 U_2$. Moreover, $\deg V_1 = 2 > b_1 = 1$, which is something that is beyond the limitations of the generic model for k-linear encodings. But this just means that the CDH oracle is performing a nontrivial task.

Regarding the way an algebraic oracle chooses the internal parameters, we restrict ourselves to the random uniform case, in which $\boldsymbol{r}$ is sampled uniformly at random independently in every query to the oracle. Notice that other interesting probability distributions can be easily captured by properly choosing the polynomials $V_1, \ldots, V_\epsilon$. Algebraic oracles can also be related to bounded polynomial sources:

Lemma 5. *The composition of an algebraic oracle with a bounded polynomial source is another bounded polynomial source.*

[11] If $\widetilde{\text{stop}} = 1$, normally $\widetilde{u}$ and $\boldsymbol{u}$ are empty, or they contain dummy values.

[12] This property is not strictly necessary and could be removed without affecting the results in the paper. However, it seems a very natural property and it avoids giving the algorithm the capability to compute unbounded multilinear maps.

Proof. Let $\mathcal{O}^{\mathrm{alg}}$ be an algebraic oracle with input $([\boldsymbol{u}]_{\boldsymbol{a}}, \widetilde{u})$ and output $([\boldsymbol{v}]_{\boldsymbol{b}})$. Let us assume that $\boldsymbol{u}$ is produced by a bounded polynomial source D with auxiliary input τ and polynomial map $f_\tau : \mathbb{Z}_q^\delta \to \mathbb{Z}_q^\alpha$ of degree d. For any choice of τ and $\widetilde{u}$, according to Definition 7 there exists a polynomial map $\boldsymbol{V} = (V_1, \ldots, V_\beta)$ of constant degree $d_{\mathcal{O}}$ such that $\boldsymbol{v} = \boldsymbol{V}(\boldsymbol{u}, \boldsymbol{t})$, for all $\boldsymbol{u} \in \mathbb{Z}_q^\alpha$ and $\boldsymbol{r} \in \mathbb{Z}_q^\rho$, where $\boldsymbol{r}$ denotes the internal parameters of the oracle. Therefore, $\boldsymbol{v} = \boldsymbol{V}(f_\tau(\boldsymbol{t}), \boldsymbol{r})$ for all $\boldsymbol{t} \in \mathbb{Z}_q^\delta$ and $\boldsymbol{r} \in \mathbb{Z}_q^\rho$. This shows that $\boldsymbol{v}$ behaves exactly as the output of a bounded polynomial source with auxiliary input $\tau' = (\tau, \widetilde{u})$ and polynomial map $f'_{\tau'} : \mathbb{Z}_q^\delta \times \mathbb{Z}_q^\rho \to \mathbb{Z}_q^\beta$ such that $f'(\boldsymbol{t}, \boldsymbol{r}) = \boldsymbol{V}(f_\tau(\boldsymbol{t}), \boldsymbol{r})$. The degree of f' is upper bounded by $\deg V \circ f \leq \deg V \deg f \leq d_{\mathcal{O}} d$. □

Now we can state the generalization of Lemma 3 for oracle algorithms. It is actually a 'composition theorem' meaning that combining an algebraic (oracle) algorithm with an algebraic oracle produces another (weak) algebraic algorithm. We said 'weak' because the degrees of the corresponding input-output polynomials no longer satisfy the constraint $\deg Y \leq a$ (since the algebraic oracle typically performs nontrivial tasks, like in the CDH example). We actually can only guarantee that the degrees of the polynomials are upper bounded by constants (i.e., the bounds are independent of the security parameter). Furthermore, we need to impose that the number of oracle calls is upper bounded by a constant Q. Observe that the degree of the polynomials could grow exponentially in the number of oracle calls (e.g., using the answer of the previous call in the next oracle query for the oracle $\mathcal{O}$ computing $[v]_1 = [u^2]_1$ yields polynomials of degree 2^Q).

Lemma 6. *Let $\mathcal{A}^{\mathcal{O}}$ be an algebraic oracle algorithm (as given in Definition 7), making a constant*[13] *bounded number of calls Q to an algebraic oracle $\mathcal{O}$. Let $([\boldsymbol{x}]_{\boldsymbol{a}}, \widetilde{x})$ and $([\boldsymbol{y}]_{\boldsymbol{b}}, \widetilde{y})$ respectively be the input and output of $\mathcal{A}$. Then, for every choice of $\widetilde{x}$ and the random tape, there exist polynomials of constant degree $Y_1, \ldots, Y_\beta \in \mathbb{Z}_q[\boldsymbol{X}, \boldsymbol{R}_1, \ldots, \boldsymbol{R}_Q]$, such that $\boldsymbol{y} = \boldsymbol{Y}(\boldsymbol{x}, \boldsymbol{r}_1, \ldots, \boldsymbol{r}_Q)$, for all inputs, where $\boldsymbol{Y} = (Y_1, \ldots, Y_\beta)$, and $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_Q$ are the parameters introduced in Definition 7 for the Q queries. Moreover, $\widetilde{y}$ does not depend on $\boldsymbol{x}$ or $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_Q$.*

Proof. We proceed by induction in Q. The first induction step, $Q = 0$, follows immediately from Lemma 3, because $\mathcal{A}^{\mathcal{O}}$ is just an algebraic algorithm (without oracle access). For $Q \geq 1$, we split $\mathcal{A}^{\mathcal{O}}$ into two sections $\mathcal{A}_0^{\mathcal{O}}$ and $\mathcal{A}_1$, separated exactly at the last query point (see Figure 2). Let $([\boldsymbol{s}]_{\boldsymbol{c}}, \widetilde{s})$ be the state information (group and non-group elements) that $\mathcal{A}_0^{\mathcal{O}}$ passes to $\mathcal{A}_1$, $([\boldsymbol{u}]_{\boldsymbol{d}}, \widetilde{u})$ be the Q-th query to $\mathcal{O}$, and $([\boldsymbol{v}]_{\boldsymbol{d}}, \widetilde{v})$ be its corresponding answer. We assume that $\mathcal{A}_0^{\mathcal{O}}$ and $\mathcal{A}_1$ receive the same random tape, \$, (perhaps introducing some redundant computations in $\mathcal{A}_1$). Observe that the output of $\mathcal{A}_0^{\mathcal{O}}$ consists of $([\boldsymbol{s}]_{\boldsymbol{c}}, \widetilde{s})$ and $([\boldsymbol{u}]_{\boldsymbol{c}}, \widetilde{u})$.

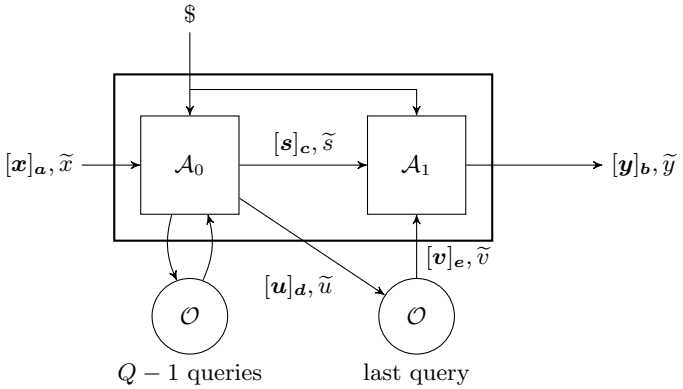

Fig. 2. Splitting of the oracle algorithm in Lemma 6.

[13] Here, 'constant' means independent of the security parameter.

By the induction assumption, for any choice of $\widetilde{x}$ and \$ there exist polynomials of constant degree $S_1, \ldots, S_\gamma, U_1, \ldots, U_\delta \in \mathbb{Z}_q[\boldsymbol{X}, \boldsymbol{R}_1, \ldots, \boldsymbol{R}_{Q-1}]$ such that for all $\boldsymbol{x} \in \mathbb{Z}_q^\alpha$ and $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{Q-1} \in \mathbb{Z}_q^\rho$, $\boldsymbol{s} = \boldsymbol{S}(\boldsymbol{x}, \boldsymbol{r}_1, \ldots, \boldsymbol{r}_{Q-1})$ and $\boldsymbol{u} = \boldsymbol{U}(\boldsymbol{x}, \boldsymbol{r}_1, \ldots, \boldsymbol{r}_{Q-1})$, where $\boldsymbol{S} = (S_1, \ldots, S_\gamma)$ and $\boldsymbol{U} = (U_1, \ldots, U_\delta)$. Moreover, $\widetilde{s}$ and $\widetilde{u}$ only depend on $\widetilde{x}$ and \$.

Now, the algorithm $\mathcal{A}_1$ receives as input $([\boldsymbol{v}]_{\boldsymbol{e}}, [\boldsymbol{s}]_{\boldsymbol{c}}, \widetilde{v}, \widetilde{s})$. By Definition 7, $\boldsymbol{v}$ also depend polynomially on $\boldsymbol{u}$ and $\boldsymbol{r}_Q$. Namely, for every choice of $\widetilde{u}$, there exist polynomials of constant degree $V_1, \ldots, V_\epsilon \in \mathbb{Z}_q[\boldsymbol{U}, \boldsymbol{R}_Q]$ such that $\boldsymbol{v} = \boldsymbol{V}(\boldsymbol{u}, \boldsymbol{r}_Q)$, where $\boldsymbol{V} = (V_1, \ldots, V_\epsilon)$, while $\widetilde{v}$ depends only on $\widetilde{u}$.

Since $\mathcal{A}_1$ is just an algebraic algorithm without oracle access, by Lemma 3, for any choice of $\widetilde{v}$, $\widetilde{s}$ and \$, there exist polynomials of constant degree $Y_1, \ldots, Y_\beta \in \mathbb{Z}_q[\boldsymbol{V}, \boldsymbol{S}]$ such that $\boldsymbol{y} = \boldsymbol{Y}(\boldsymbol{v}, \boldsymbol{s})$, where $\boldsymbol{Y} = (Y_1, \ldots, Y_\beta)$, for all $\boldsymbol{v} \in \mathbb{Z}_q^\epsilon$ and $\boldsymbol{s} \in \mathbb{Z}_q^\gamma$, while $\widetilde{y}$ depends only on $\widetilde{v}$, $\widetilde{s}$ and \$. By composition of all the previous polynomials, we show that $\boldsymbol{y}$ depend polynomially on $\boldsymbol{x}$ and $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_Q$, where the polynomials depend only on \$ and $\widetilde{x}$. Indeed

$$\boldsymbol{y} = \boldsymbol{Y}(\boldsymbol{V}(\boldsymbol{U}(\boldsymbol{x}, \boldsymbol{r}_1, \ldots, \boldsymbol{r}_{Q-1}), \boldsymbol{r}_Q), \boldsymbol{S}(\boldsymbol{x}, \boldsymbol{r}_1, \ldots, \boldsymbol{r}_{Q-1}))$$

and all the polynomials involved depend only on $\widetilde{x}$, $\widetilde{s}$, $\widetilde{u}$, $\widetilde{v}$ and \$, but all in turn only depend on $\widetilde{x}$ and \$. In addition, for the same reason, $\widetilde{y}$ only can depend on $\widetilde{x}$ and \$, which concludes the proof. □

The previous lemma shows that the combination of an algebraic oracle algorithm and an algebraic oracle essentially preforms polynomial operations in the group elements while preserving some independence with the non-group elements, as happens to the plain algebraic algorithms.

However, at this point we do not know how to deal with generic oracle algorithms. The problem is that a real oracle $\mathcal{O}$, algebraic or not, deals with group elements rather than labels. Then a generic algorithm $\mathcal{A}^{\text{gen}}$ cannot directly query the oracle $\mathcal{O}$, and it cannot directly receive the oracle responses. On the other hand, the oracle $\mathcal{O}^{\text{gen}}$ associated to the generic model cannot transform the group elements contained in $\mathcal{O}$'s answers into labels in a consistent way for an arbitrary oracle $\mathcal{O}$. Indeed, if $\mathcal{O}^{\text{gen}}$ just assigns fresh unrelated labels (i.e., new formal variables) to the group elements provided by $\mathcal{O}$, a lot of nontrivial GroupEqTest queries would appear, and the purely algebraic version of the generic model is no longer equivalent to the plain (non-algebraic) generic model.

Again, the definition of algebraic oracle is very helpful in the context of generic algorithms. We show below that the procedure used in the study of algorithms without oracles can also be successfully applied to the case with algebraic oracles. Firstly, we update the definitions of generic algorithm (in both generic models) in order to make the generic model oracles $\mathcal{O}^{\text{gen}}$ and $\mathcal{O}^{\text{pa-gen}}$ intercept the queries to the algebraic oracle $\mathcal{O}$ made by the algorithm $\mathcal{A}^{\text{gen}}$.

Definition 8 (Generic Algorithm With Extra Oracle Access). *An algorithm $\mathcal{A}^{gen}$ is called generic with extra oracle access to an algebraic oracle $\mathcal{O}$ if it fits into Definition 1 but with an additional query type to $\mathcal{O}^{gen}$:*

- GroupOracleQuery$((U_1, d_1), \ldots, (U_\delta, d_\delta), \widetilde{u})$*: queries the oracle $\mathcal{O}$ with the corresponding group elements $[u_1]_{d_1}, \ldots, [u_\delta]_{d_\delta}$ and the non-group elements $\widetilde{u}$. Let $([v_1]_{e_1}, \ldots, [v_\epsilon]_{e_\epsilon}, \widetilde{v})$ be the answer given by $\mathcal{O}$. Then $\mathcal{O}^{gen}$ assigns fresh labels $(V_1, e_1), \ldots, (V_\epsilon, e_\epsilon)$ to the group elements $[v_1]_{e_1}, \ldots, [v_\epsilon]_{e_\epsilon}$ returned by $\mathcal{O}$. Then $\mathcal{O}^{gen}$ stores the group element/label pairs them in its internal table $\mathcal{T}$, and sends back the new labels and the non-group elements $\widetilde{v}$ to $\mathcal{A}^{gen}$. After each successful* GroupOracleQuery *query the new formal variables $V_1, \ldots, V_\epsilon$ are added to the polynomial label system. Recall that* GroupEqTest *queries are answered by $\mathcal{O}^{gen}$ based on the actual group elements, and not on the labels.*

Similarly, in the purely-algebraic generic model $\mathcal{O}^{gen}$ is replaced by $\mathcal{O}^{pa\text{-}gen}$, with the following differences:

- *$\mathcal{A}^{gen}$ receives its input from a bounded polynomial source D.*
- *Following Definition 4, $\mathcal{O}^{pa\text{-}gen}$ maintains a table $\mathcal{T}$ consisting of pairs of internal polynomials and labels $(\widehat{X}_i, (X_i, a_i))$. But now, each internal polynomial $\widehat{X}_i$ depends on the internal parameters $\boldsymbol{T}$ of the source D and the internal parameters $\boldsymbol{R}_1, \ldots, \boldsymbol{R}_{Q_i}$ generated by $\mathcal{O}$ in the Q_i queries performed before the label (X_i, a_i) was created.*

– *During* GroupOracleQuery, $\mathcal{O}^{pa\text{-}gen}$ *queries the oracle* $\mathcal{O}$ *on* $([0]_{d_1}, \ldots, [0]_{d_\delta}, \widetilde{u})$*, that is, using dummy group elements, and receives its answer* $([v_1]_{e_1}, \ldots, [v_\epsilon]_{e_\epsilon}, \widetilde{v})$*. Then,* $\mathcal{O}^{pa\text{-}gen}$ *ignores the group elements* $([v_1]_{e_1}, \ldots, [v_\epsilon]_{e_\epsilon})$ *replacing them by fresh labels* $(V_1, e_1), \ldots, (V_\epsilon, e_\epsilon)$*, and it sends back to* $\mathcal{A}^{gen}$ *these labels along with* $\widetilde{v}$*. In addition,* $\mathcal{O}^{pa\text{-}gen}$ *computes the corresponding internal polynomials* $(\widehat{V}_1, \ldots, \widehat{V}_\epsilon)$ *by composing the polynomials* $V_i(U_1, \ldots, U_\delta, R_1, \ldots, R_\rho)$ *given in the definition of the algebraic oracle* $\mathcal{O}$ *with the internal polynomials* $\widehat{U}_1, \ldots, \widehat{U}_\delta$*, retrieved from* $\mathcal{T}$*. As mentioned above, the new formal parameters* $R_1, \ldots, R_\rho$ *defined by the oracle* $\mathcal{O}$ *are added to the label system, that is, now internal polynomials also depend on these new parameters. Notice that every oracle call adds new parameters to the system, since different calls to* $\mathcal{O}$ *are independent.*
– *For valid labels* GroupEqTest$((Y_1, a), (Y_2, a))$ *outputs* 1 *if and only if the corresponding internal polynomials are equal, that is* $\widehat{Y}_1 = \widehat{Y}_2$*, as polynomials in* $\mathbb{Z}_q[\boldsymbol{T}, \boldsymbol{R}_1, \ldots, \boldsymbol{R}_{Q'}]$*, where* Q' *is the number of queries already issued to* $\mathcal{O}$*.*
– $\mathcal{O}^{pa\text{-}gen}$ *does not store any group element and it does not perform any real group operation until* $\mathcal{A}^{gen}$ *finishes its execution. Instead* $\mathcal{O}^{pa\text{-}gen}$ *obtains its final state by sampling* $(t_1, \ldots, t_\alpha)$ *a posteriori form the bounded polynomial source* D *and also the parameters* $\boldsymbol{R}_1, \ldots, \boldsymbol{R}_Q$ *uniformly at random. Then* $\mathcal{O}^{pa\text{-}gen}$ *evaluates the internal polynomials* $\widehat{Y}_1, \ldots, \widehat{Y}_\beta$ *at the sampled values, obtaining* $\boldsymbol{y} = \widehat{\boldsymbol{Y}}(\boldsymbol{t}, \boldsymbol{r}_1, \ldots, \boldsymbol{r}_Q)$*. It finally computes the corresponding group elements* $[\boldsymbol{y}]_{\boldsymbol{b}}$ *from the computed discrete logarithms* $\boldsymbol{y}$*.*

Notice that all the internal polynomials in the table $\mathcal{T}$ are now polynomials in the source parameters and in all the internal parameters defined in the queries to the algebraic oracle.

We end this section with two lemmas that generalize the results in the previous section to the case of oracle algorithms. Essentially, we claim that generic algorithms perform as well as algebraic algorithms do, even when they have access to an extra algebraic oracle, provided that their inputs are produced by a bounded polynomial source and the groups are exponentially large.

Lemma 7. *Given a generic algorithm* $\mathcal{A}^{gen}$ *with extra oracle access to an algebraic oracle* $\mathcal{O}$ *as described in Definition 8, with inputs* $(X_1, a_1), \ldots, (X_\alpha, a_\alpha)$ *and* $\widetilde{x}$*, and a bounded polynomial source* D*, as described in Definition 2, with auxiliary input* $\widetilde{x} \in \{0,1\}^*$ *and polynomial map* $f : \mathbb{Z}_q^\delta \to \mathbb{Z}_q^\alpha$ *of total degree* d*, there exists an algebraic oracle algorithm* $\mathcal{A}^{alg}$ *with access to* $\mathcal{O}$ *that produces the same outputs as* $\mathcal{A}^{gen}$ *in the purely-algebraic* k*-graded encoding generic model (i.e., interacting with the oracle* $\mathcal{O}^{pa\text{-}gen}$*) with input group elements sampled from* D*.*

Proof. $\mathcal{A}^{\text{alg}}$ receives the inputs $[x_1]_{a_1}, \ldots, [x_\alpha]_{a_\alpha}$ and an auxiliary input $\widetilde{x}$, where $(x_1, \ldots, x_\alpha)$ are sampled from $D_{\widetilde{x}}$. Then it simulates the oracle $\mathcal{O}^{\text{pa-gen}}$ for $\mathcal{A}^{\text{gen}}$ following Definition 8. Namely, on start $\mathcal{A}^{\text{alg}}$ prepares its first iteration by starting the execution of $\mathcal{A}^{\text{gen}}$ with the inputs $((X_1, a_1), \ldots, (X_\alpha, a_\alpha), \widetilde{x})$, and setting $[\boldsymbol{z}]_{\boldsymbol{c}} = [\boldsymbol{x}]_{\boldsymbol{a}}$ and $\widetilde{z} = \widetilde{x}$ as inputs for its first iteration.

During each iteration, $\mathcal{A}^{\text{alg}}$ maintains an internal table $\mathcal{T}$ exactly as $\mathcal{O}^{\text{pa-gen}}$ would do, and it answers all oracle queries made by $\mathcal{A}^{\text{gen}}$ in a straightforward way, by performing the corresponding polynomial operations in the labels, except for the GroupEqTest, which is answered based on the internal polynomials stored in $\mathcal{T}$, until $\mathcal{A}^{\text{gen}}$ queries GroupOracleQuery or it finishes its execution.

Whenever $\mathcal{A}^{\text{gen}}$ queries GroupOracleQuery on $((U_1, d_1), \ldots, (U_\delta, d_\delta), \widetilde{u})$, $\mathcal{A}^{\text{alg}}$ uses the polynomials $U_1, \ldots, U_\delta \in \mathbb{Z}_q[\boldsymbol{Z}]$ and the k-graded encodings to compute the group elements $([u_1]_{d_1}, \ldots, [u_\delta]_{d_\delta})$ from the stored elements $[\boldsymbol{z}]_{\boldsymbol{c}}$. Formally (compared to Definition 6) the vector $\boldsymbol{y}$ is empty (and so is $\boldsymbol{Y}$) and $\widetilde{\text{stop}} = 0$. Moreover, the internal state of $\mathcal{A}^{\text{gen}}$ and the table $\mathcal{T}$ are the non-group elements stored at this point by $\mathcal{A}^{\text{alg}}$.

Now, $\mathcal{A}^{\text{alg}}$ queries $\mathcal{O}$ with $([u_1]_{d_1}, \ldots, [u_\delta]_{d_\delta}, \widetilde{u})$, receiving the answer $([v_1]_{e_1}, \ldots, [v_\epsilon]_{e_\epsilon}, \widetilde{v})$. Then, it adds the new entries $(\widehat{V}_i, (V_i, e_i))$ for $i = 1, \ldots, \epsilon$ to the table $\mathcal{T}$ exactly in the same way as $\mathcal{O}^{\text{pa-gen}}$ does in Definition 8, that is, ignoring the group elements $([v_1]_{e_1}, \ldots, [v_\epsilon]_{e_\epsilon})$ and generating fresh labels, but using the polynomials defining the algebraic oracle to compute the internal polynomials $\widehat{V}_1, \ldots, \widehat{V}_\epsilon$.

Next, $\mathcal{A}^{\text{alg}}$ sends $((V_1, e_1), \ldots, (V_\epsilon, e_\epsilon), \widetilde{v})$ to $\mathcal{A}^{\text{gen}}$, and appends $[\boldsymbol{v}]_{\boldsymbol{e}}$ to $[\boldsymbol{z}]_{\boldsymbol{c}}$ to obtain the input for its next iteration, and similarly it appends $\widetilde{v}$ to $\widetilde{z}$, and continues.

If $\mathcal{A}^{\text{gen}}$ eventually finishes its execution by giving the output $((Y_1, b_1) \ldots, (Y_\beta, b_\beta), \widetilde{y})$ to $\mathcal{A}^{\text{alg}}$, then it formally sets $\widetilde{\text{stop}} = 0$, and again it uses the stored elements $[\boldsymbol{z}]_{\boldsymbol{c}}$, the polynomials $Y_1, \ldots, Y_\beta \in \mathbb{Z}_q[\boldsymbol{Z}]$ and the k-graded encodings to compute the group elements $([y_1]_{b_1}, \ldots, [y_\beta]_{b_\beta})$. Finally, it outputs $([y_1]_{b_1}, \ldots, [y_\beta]_{b_\beta}, \widetilde{y})$.

The simulation performed by $\mathcal{A}^{\text{alg}}$ is perfect and it fulfils the requirements given in Definition 6. Indeed, the only difference in the simulation with respect to the definition of $\mathcal{O}^{\text{pa-gen}}$ is that the oracle $\mathcal{O}$ is called with the "actual" group elements $[\boldsymbol{u}]_{\boldsymbol{d}}$ instead of using dummy ones. However, since the output group elements given by $\mathcal{O}$ are ignored, and the non-group elements $\widetilde{v}$ cannot depend on $[\boldsymbol{u}]_{\boldsymbol{d}}$ (according to Definition 7), the view of $\mathcal{A}^{\text{gen}}$ is not affected but such difference. □

Lemma 8. *Let $\mathcal{A}^{gen}$ be a generic algorithm that makes a constant number Q of oracle calls to an algebraic oracle $\mathcal{O}$, and it receives its input from a bounded polynomial source D with auxiliary input $\widetilde{x} \in \{0,1\}^*$ and polynomial map $f : \mathbb{Z}_q^\delta \to \mathbb{Z}_q^\alpha$ of total degree d. Then, for exponentially large q, the difference between the success probabilities of $\mathcal{A}^{gen}$ in the generic model for k-linear encodings interacting with either $\mathcal{O}^{gen}$ or $\mathcal{O}^{pa\text{-}gen}$ is negligible.*

Proof. If $Q = 0$, this lemma is a particular case of Lemma 1. Thus, we only consider the case $Q \leq 1$. We will use a hybrid argument to gradually transform the plain generic model to the purely-algebraic one. We define a sequence of oracles $\mathcal{O}^{\text{hy}_\ell\text{-gen}}$ for $\ell = 1, \ldots, Q$. The oracle $\mathcal{O}^{\text{hy}_\ell\text{-gen}}$ starts behaving as $\mathcal{O}^{\text{pa-gen}}$ until the ℓ-th GroupOracleQuery oracle call is performed (including it). After this call, $\mathcal{O}^{\text{hy}_\ell\text{-gen}}$ behaves as $\mathcal{O}^{\text{gen}}$. By convention we define $\mathcal{O}^{\text{hy}_0\text{-gen}} = \mathcal{O}^{\text{gen}}$ and $\mathcal{O}^{\text{hy}_{Q+1}\text{-gen}} = \mathcal{O}^{\text{pa-gen}}$. Let us denote by Succ_ℓ the success probability of $\mathcal{A}^{\text{gen}}$ when it interacts with the oracle $\mathcal{O}^{\text{hy}_\ell\text{-gen}}$, where 'success' means here any fixed predicate of the input and output values of $\mathcal{A}^{\text{gen}}$.

Observe that conceptually, the difference between $\mathcal{O}^{\text{gen}}$ and $\mathcal{O}^{\text{hy}_\ell\text{-gen}}$ is that the parameters $\boldsymbol{t}$ of the bounded polynomial source and the internal parameters of the first ℓ GroupOracleQuery calls are sampled in the latter after the oracle $\mathcal{O}$ answers the ℓ-th query. Therefore, $\mathcal{O}^{\text{hy}_\ell\text{-gen}}$ does not deal with real group elements until this point. In the special case of $\ell = Q + 1$, the parameter sampling is performed after $\mathcal{A}^{\text{gen}}$ finishes its execution. The main idea in the proof is using Lemma 1 for each section of $\mathcal{A}^{\text{gen}}$ between consecutive calls to GroupOracleQuery, and also for the initial and final sections. In addition, the way GroupOracleQuery calls the real oracle $\mathcal{O}$ changes in $\mathcal{O}^{\text{hy}_\ell\text{-gen}}$, because $\mathcal{O}$ is fed with dummy group elements in the first ℓ calls. But this is just a technical difference, because the real action of the algebraic oracle $\mathcal{O}$ on group elements is performed by $\mathcal{O}^{\text{hy}_\ell\text{-gen}}$ by means of the polynomials defining $\mathcal{O}$ after the parameters are sampled.

We show that $|\mathsf{Succ}_{\ell+1} - \mathsf{Succ}_\ell|$ is negligible, under the conditions stated in the lemma. Indeed, we can split $\mathcal{A}^{\text{gen}}$ into two sections $\mathcal{A}_\ell^-$ and $\mathcal{A}_\ell^+$ separated by the ℓ-th GroupOracleQuery call, as shown in Figure 3.

Observe that in either of the two scenarios, that is using $\mathcal{O}^{\text{hy}_\ell\text{-gen}}$ or $\mathcal{O}^{\text{hy}_{\ell+1}\text{-gen}}$, $\mathcal{A}_\ell^-$ is in itself a generic oracle algorithm interacting with $\mathcal{O}^{\text{pa-gen}}$. Indeed, we can replace the bounded polynomial source D, the generic algoritm section $\mathcal{A}_\ell^-$ and the first ℓ queries to GroupOracleQuery by another equivalent bounded polynomial source, D_ℓ, with parameters $(\boldsymbol{t}, \boldsymbol{r}_1, \ldots, \boldsymbol{r}_\ell)$ and polynomial map extracted from lemmas 6 and 7.[14] This reduces the problem to the case $\ell = 0$, but for $D' = D_\ell$, $\mathcal{A}'^{\text{gen}} = \mathcal{A}_\ell^+$, $Q' = Q - \ell$, $\ell' = 0$ and $([\boldsymbol{x'}]_{\boldsymbol{a'}}, \widetilde{x'}) = [\boldsymbol{s}]_{\boldsymbol{c}}, [\boldsymbol{v}]_{\boldsymbol{e}}, \widetilde{x}, \widetilde{v})$. However, the non-group elements $\widetilde{x'} = (\widetilde{s}, \widetilde{v})$ in the input of $\mathcal{A}'^{\text{gen}}$ now could be correlated with its random tape. Notice that the case $\ell = Q$ becomes $Q' = 0$ which is again a particular case of Lemma 1. We focus now on $\ell < Q$ (that is, $\mathcal{A}^+$ makes at least one GroupOracleQuery call).

We analyze the differences between the two games in which $\mathcal{A}'^{\text{gen}}$ interacts with either $\mathcal{O}^{\text{hy}_0\text{-gen}} = \mathcal{O}^{\text{gen}}$ or $\mathcal{O}^{\text{hy}_1\text{-gen}}$ for $Q' \geq 1$. We consider again the splitting of $\mathcal{A}'^{\text{gen}}$ into $\mathcal{A}'^-_1$ and $\mathcal{A}'^+_1$, shown in Figure 4. In both games, $\mathcal{A}'^+_1$ is interacting with $\mathcal{O}^{\text{gen}}$. Hence, the only differences between the games can be produced by $\mathcal{A}'^-_1$ via a nontrivial GroupEqTest query, since $\mathcal{A}'^-_1$ is making no GroupOracleQuery calls. According to Lemma 1, these nontrivial queries occur only with a negligible probability (even when the non-group

[14] Actually, for fixed $\widetilde{x}$ and the random tape, Lemma 6 gives polynomials $S_1, \ldots, S_\gamma, V_1, \ldots, V_\epsilon \in \mathbb{Z}_q[\boldsymbol{X}, \boldsymbol{R}_1, \ldots, \boldsymbol{R}_\ell]$ such that $\boldsymbol{s} = \boldsymbol{S}(\boldsymbol{x}, \boldsymbol{r}_1, \ldots, \boldsymbol{r}_\ell)$ and $\boldsymbol{v} = \boldsymbol{V}(\boldsymbol{x}, \boldsymbol{r}_1, \ldots, \boldsymbol{r}_\ell)$. Then the polynomial map of D_ℓ is the composition of $(\boldsymbol{S}, \boldsymbol{V})$ with the polynomial map f of D.

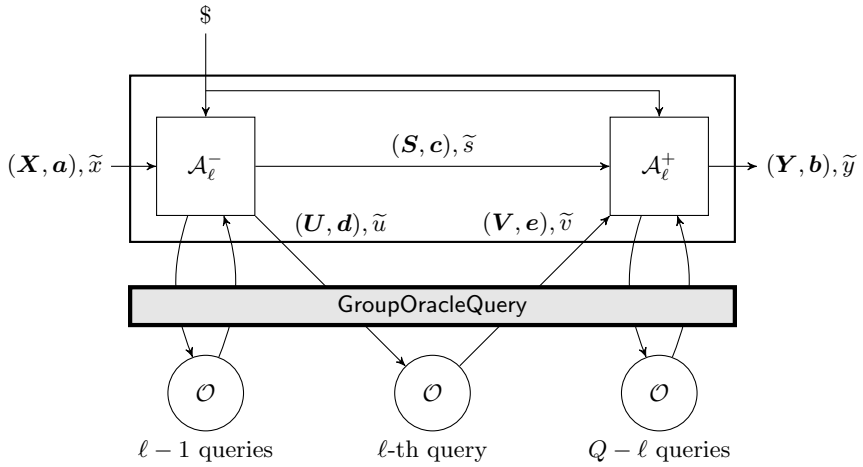

Fig. 3. Splitting of the generic algorithm in Lemma 8, for $1 \leq \ell \leq Q$.

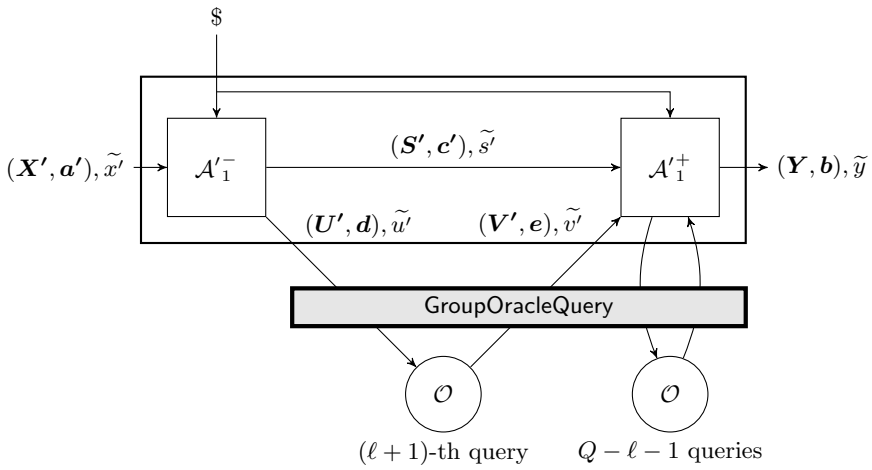

Fig. 4. Equivalent splitting in Lemma 8 after the replacement.

element in the input are correlated to the random tape). Therefore, $\left|\mathsf{Succ}'_1 - \mathsf{Succ}'_0\right| \in \mathit{negl}$, and undoing the previous replacement, this implies that $|\mathsf{Succ}_{\ell+1} - \mathsf{Succ}_\ell| \in \mathit{negl}$ also for all $\ell = 1, \ldots, Q$. Finally,

$$|\mathsf{Succ}_{Q+1} - \mathsf{Succ}_0| \leq \sum_{\ell=0}^{Q} |\mathsf{Succ}_{\ell+1} - \mathsf{Succ}_\ell| \in \mathit{negl}$$

which concludes the proof. □

The previous results in particular imply that for any generic oracle algorithm $\mathcal{A}^{\mathrm{gen}}$ in the plain model, receiving inputs form a bounded polynomial source and with access to an algebraic oracle, there exists an algebraic oracle algorithm with an indistinguishable behavior. Therefore, we can extract from $\mathcal{A}^{\mathrm{gen}}$ the polynomials $Y_1, \ldots, Y_\beta \in \mathbb{Z}_q[\boldsymbol{X}, \boldsymbol{R}_1, \ldots, \boldsymbol{R}_Q]$ described in Lemma 6.

3.5 The Matrix Decisional Diffie-Hellman Assumption

We recall here the definition of the decisional assumptions introduced in [11], which are the starting point of our flexible computational matrix problems.

Definition 9. *[11], Let $\ell, k \in \mathbb{N}$ with $\ell > k$. We call $\mathcal{D}_{\ell,k}$ a matrix distribution if it outputs (in polynomial time, with overwhelming probability) matrices in $\mathbb{Z}_q^{\ell \times k}$ of full rank k. We denote $\mathcal{D}_k := \mathcal{D}_{k+1,k}$.*

Definition 10 ($\mathcal{D}_{\ell,k}$-MDDH Assumption). *[11] Let $\mathcal{D}_{\ell,k}$ be a matrix distribution. The $\mathcal{D}_{\ell,k}$-Matrix Diffie-Hellman ($\mathcal{D}_{\ell,k}$-MDDH) Problem is telling apart the two probability distributions $(\mathbb{G}, q, \mathcal{P}, [\mathbf{A}], [\mathbf{A}\boldsymbol{w}])$ and $(\mathbb{G}, q, \mathcal{P}, [\mathbf{A}], [\boldsymbol{z}])$, where $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}, \boldsymbol{w} \leftarrow \mathbb{Z}_q^k, \boldsymbol{z} \leftarrow \mathbb{Z}_q^\ell$.*

We say that the $\mathcal{D}_{\ell,k}$-Matrix Diffie-Hellman ($\mathcal{D}_{\ell,k}$-MDDH) Assumption holds relative to Gen if the corresponding problem is hard, that is, if for all PPT adversaries $\mathcal{A}$, the advantage

$$\mathbf{Adv}_{\mathcal{D}_{\ell,k},\mathsf{Gen}}(\mathcal{A}) = \Pr[\mathcal{A}(\mathcal{G}, [\mathbf{A}], [\mathbf{A}\boldsymbol{w}]) = 1] - \Pr[\mathcal{A}(\mathcal{G}, [\mathbf{A}], [\boldsymbol{z}]) = 1] \in \mathit{negl}(\lambda),$$

where the probability is taken over $\mathcal{G} = (\mathbb{G}, q, \mathcal{P}) \leftarrow \mathsf{Gen}(1^\lambda)$, $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}, \boldsymbol{w} \leftarrow \mathbb{Z}_q^k, \boldsymbol{z} \leftarrow \mathbb{Z}_q^\ell$ and the coin tosses of adversary $\mathcal{A}$.

In the case of symmetric k-linear groups, we similarly say that the $\mathcal{D}_{\ell,k}$-MDDH Assumption holds relative to MGen_k when

$$\mathbf{Adv}_{\mathcal{D}_{\ell,k},\mathsf{MGen}_k}(\mathcal{A}) = \Pr[\mathcal{A}(\mathcal{MG}_k, [\mathbf{A}]_1, [\mathbf{A}\boldsymbol{w}]_1) = 1] - \Pr[\mathcal{A}(\mathcal{MG}_k, [\mathbf{A}]_1, [\boldsymbol{z}]_1) = 1] \in \mathit{negl}(\lambda),$$

where the probability is taken over $\mathcal{MG}_k = (e, \mathbb{G}_1, \ldots, \mathbb{G}_k, q, \mathcal{P}_1, \ldots, \mathcal{P}_k) \leftarrow \mathsf{MGen}_k(1^\lambda)$, $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}, \boldsymbol{w} \leftarrow \mathbb{Z}_q^k, \boldsymbol{z} \leftarrow \mathbb{Z}_q^\ell$ and the coin tosses of adversary $\mathcal{A}$. The asymmetric case is dealt with in the same way. We will say that the $\mathcal{D}_{\ell,k}$-MDDH Assumption holds relative to AGen_2 in the left (resp. in the right) when $\mathcal{A}$ is given as input the output $\mathcal{AG}_2 = (e, \mathbb{G}, \mathbb{H}, \mathbb{T}, q, \mathcal{P}, \mathcal{Q})$ of AGen_2 along with the matrix $\mathbf{A}$ and the vector $\mathbf{A}\boldsymbol{w}$ or $\boldsymbol{z}$, both encoded in $\mathbb{G}$ (resp. in $\mathbb{H}$).

The following definition just aims to simplifying some of the statements in the paper.

Definition 11. *A matrix distribution $\mathcal{D}_{\ell,k}$ is hard if the corresponding $\mathcal{D}_{\ell,k}$-MDDH problem is hard in the generic k-linear group model.*

Some particular families of matrix distributions were presented in [11]. Namely,

$$\mathcal{SC}_k : \mathbf{A} = \begin{pmatrix} a & & 0 \\ 1 & \ddots & \\ & \ddots & a \\ 0 & & 1 \end{pmatrix} \quad \mathcal{C}_k : \mathbf{A} = \begin{pmatrix} a_1 & & 0 \\ 1 & \ddots & \\ & \ddots & a_k \\ 0 & & 1 \end{pmatrix} \quad \mathcal{L}_k : \mathbf{A} = \begin{pmatrix} a_1 & & 0 \\ & \ddots & \\ 0 & & a_k \\ 1 & \cdots & 1 \end{pmatrix},$$

where $a, a_i \leftarrow \mathbb{Z}_p$, and $\mathcal{U}_{\ell,k}$ which is simply the uniform distribution in $\mathbb{Z}_p^{\ell \times k}$. The $\mathcal{SC}_k$-MDDH Assumption is the Symmetric Cascade Assumption, the $\mathcal{C}_k$-MDDH Assumption is the Cascade Assumption, which were

proposed for the first time. $\mathcal{U}_{\ell,k}$-MDDH is the Uniform Assumption, which appeared under other names in [7,36]. $\mathcal{L}_k$-MDDH is the Decisional Linear Assumption [6,22,39]. For instance, we can consider the case $k = 2$, in which the $\mathcal{L}_2$-MDDH problem is given $([1], [a_1], [a_2])$, tell apart the two distributions $([1], [a_1], [a_2], [w_1 a_1], [w_2 a_2], [w_1 + w_2])$ and $([1], [a_1], [a_2], [z_1], [z_2], [z_3])$, where $a_1, a_2, w_1, w_2, z_1, z_2, z_3$ are random. This is exactly the 2-Lin Problem, since we can always set $z_1 = w_1 a_1$ and $z_2 = w_2 a_2$. We also give examples of matrix distributions which did not appear in [11] but that are implicitly used in the problems 2 and 4 in Section 4.2. The Randomized Linear and the Square Polynomial distributions are respectively given by the matrices

$$\mathcal{RL}_k : \mathbf{A} = \begin{pmatrix} a_1 & & 0 \\ & \ddots & \\ 0 & & a_k \\ b_1 & \cdots & b_k \end{pmatrix} \qquad \mathcal{P}_{\ell,2} : \mathbf{A} = \begin{pmatrix} a_1 & a_1^2 \\ a_2 & a_2^2 \\ \vdots & \vdots \\ a_\ell & a_\ell^2 \end{pmatrix}$$

where $a_i \leftarrow \mathbb{Z}_q$ and $b_i \leftarrow \mathbb{Z}_q^\times$. Jutla and Roy [24] referred to $\mathcal{RL}_k$-MDDH Assumption as the k-lifted Assumption.

4 The Matrix Diffie-Hellman Computational Problems

In this section we introduce two families of search problems naturally related to the Matrix Decisional Diffie-Hellman problems. In the first family, given a matrix $[\mathbf{A}]$, where $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}$, and the first k components of a vector $[\boldsymbol{z}]$, the problem is completing it so that $\boldsymbol{z} \in \operatorname{Im} \mathbf{A}$.

Definition 12 ($\mathcal{D}_{\ell,k}$-MCDH). *Given a matrix distribution $\mathcal{D}_{\ell,k}$, such that the upper $k \times k$ submatrix of $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}$ has full rank with overwhelming probability, the computational matrix Diffie-Hellman Problem is given $([\mathbf{A}], [\boldsymbol{z}_0])$, with $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}$, $\boldsymbol{z}_0 \leftarrow \mathbb{Z}_q^k$, compute $[\boldsymbol{z}_1] \in \mathbb{G}^{\ell-k}$ such that $(\boldsymbol{z}_0 \| \boldsymbol{z}_1) \in Im\, \mathbf{A}$.*

The full-rank condition ensures the existence of solutions to the $\mathcal{D}_{\ell,k}$-MCDH problem instance. Thus, we tolerate the existence of a negligible fraction of unsolvable problem instances. Indeed, all known interesting matrix distributions fulfil this requirement. Notice that CDH and the computational k-Lin problems are particular examples of MCDH problems. Namely, CDH is exactly $\mathcal{L}_1$-MCDH and the computational k-Lin problem is $\mathcal{L}_k$-MCDH. Indeed, the $\mathcal{L}_1$-MCDH problem is given $[1], [a], [z_1]$, compute $[z_2]$ such that (z_1, z_2) is collinear with $(1, a)$, or equivalently, $z_2 = z_1 a$, which is solving the CDH problem. All MCDH problems have a unique solution and they appear naturally in some scenarios using MDDH problems. For instance, the one-wayness of the encryption scheme in [11] is equivalent to the corresponding MCDH assumption.

There is an immediate relation between any MCDH problem and its decisional counterpart. Not surprisingly, for any matrix distribution $\mathcal{D}_{\ell,k}$, $\mathcal{D}_{\ell,k}$-MDDH $\Rightarrow$ $\mathcal{D}_{\ell,k}$-MCDH.

We are not going to study the possible reductions between MCDH problems, due to the fact that, essentially, any MCDH problem amounts to computing some polynomial on the elements of $\mathbf{A}$, and it is then equivalent to CDH ([4,23]), although the tightness of the reduction depends on the degree of the polynomial.

In the second family of computational problems, given a matrix $[\mathbf{A}]$, where $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}$, the problem is finding $[\boldsymbol{x}]$ such that $\boldsymbol{x} \in \ker \mathbf{A}^\top \setminus \{\mathbf{0}\}$. It is notable that some computational problems in the literature are particular cases of this second family.

Definition 13 ($\mathcal{D}_{\ell,k}$-KerMDH). *Given a matrix distribution $\mathcal{D}_{\ell,k}$, the Kernel Diffie-Hellman Problem is given $[\mathbf{A}]$, with $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}$, find a nonzero vector $[\boldsymbol{x}] \in \mathbb{G}^\ell$ such that $\boldsymbol{x}$ is orthogonal to $Im\, \mathbf{A}$, that is, $\boldsymbol{x} \in \ker \mathbf{A}^\top \setminus \{\mathbf{0}\}$.*

Definition 13 naturally extends to asymmetric bilinear groups. There, given $[\mathbf{A}]_H$, the problem is to find $[\boldsymbol{x}]_G$ such that $\boldsymbol{x} \in \ker \mathbf{A}^\top \setminus \{\mathbf{0}\}$. A solution can be obviously verified by checking if $e([\boldsymbol{x}^\top]_G, [\mathbf{A}]_H) = [\mathbf{0}]_T$. We can also consider an extension of this problem in which the goal is to solve the same problem but giving the solution in a different group $\mathbb{G}_r$, in some ideal graded encoding $\mathcal{MG}_m$, for some $0 \leq r \leq \min(m, k-1)$. The case $r = 1$ corresponds to the previous problem defined in a m-linear group.

Definition 14 (($r, m, \mathcal{D}_{\ell,k}$)-KerMDH). *Given a matrix distribution $\mathcal{D}_{\ell,k}$ over a m-linear group $\mathcal{MG}_m$ and r an integer $0 \leq r \leq \min(m, k-1)$, the $(r, m, \mathcal{D}_{\ell,k})$-KerMDH Problem is to find $[\boldsymbol{x}]_r \in \mathbb{G}_r^\ell$ such that $\boldsymbol{x} \in \ker \mathbf{A}^\top \setminus \{\mathbf{0}\}$.*

When the precise degree of multilinearity m is not an issue, we will write $(r, \mathcal{D}_{\ell,k})$-KerMDH instead of $(r, m, \mathcal{D}_{\ell,k})$-KerMDH, for any $m \geq r$. We excluded the case $r \geq k$ because the problem is easy.

Lemma 9. *For all integers $k \leq r \leq m$ and for all matrix distributions $\mathcal{D}_{\ell,k}$, the $(r, m, \mathcal{D}_{\ell,k})$-KerMDH Problem is easy.*

Proof. If $\ell = k+1$, a solution to the problem is the vector $[(A_1, -A_2, \ldots, (-1)^k A_{k+1})]_r$ where A_i is the minor of $\mathbf{A}$ obtained by deleting the i-th row, computed by means of the m-linear map, as $m \geq r$. In the case $\ell > k+1$ the solution can be obtained with a similar trick applied to any full-rank $(k+1) \times k$ submatrix of $\mathbf{A}$. □

The kernel problem is also harder than the corresponding decisional problem, in multilinear groups.

Lemma 10. *In a m-linear group, $\mathcal{D}_{\ell,k}$-MDDH $\Rightarrow$ $(r, m, \mathcal{D}_{\ell,k})$-KerMDH for any matrix distribution $\mathcal{D}_{\ell,k}$ and for any $0 \leq r \leq m-1$. In particular, for $m \geq 2$, $\mathcal{D}_{\ell,k}$-MDDH $\Rightarrow$ $\mathcal{D}_{\ell,k}$-KerMDH.*

Proof. Given an instance of the $\mathcal{D}_{\ell,k}$-MDDH problem $([\mathbf{A}], [\boldsymbol{z}])$, a solution to test membership in $\operatorname{Im} \mathbf{A}$ is simply checking whether $e([\boldsymbol{x}^\top]_r, [\boldsymbol{z}]) = [\boldsymbol{x}^\top \boldsymbol{z}]_{r+1} \stackrel{?}{=} [0]_{r+1}$, where $[\boldsymbol{x}]_r$ is the output of the $(r, m, \mathcal{D}_{\ell,k})$-KerMDH solver on input $[\mathbf{A}]$. For a real instance of $\mathcal{D}_{\ell,k}$-MDDH (*i.e.*, $\boldsymbol{z} \in \operatorname{Im} \mathbf{A}$) the solver gives always the correct answer. Furthermore, if the instance is random (*i.e.*, $\boldsymbol{z}$ is a random vector) then $\boldsymbol{x}^\top \boldsymbol{z} = 0$ occurs only with a negligible probability $1/q$. Therefore, the reduction works fine with overwhelming probability. □

4.1 The Kernel DH Assumptions in the Multilinear Maps Candidates

We have shown that for any hard matrix distribution $\mathcal{D}_{\ell,k}$ the $\mathcal{D}_{\ell,k}$-KerMDH problem is generically hard in m-linear groups. We emphasize that all our results refer to generic, ideal multilinear maps (in fact, to graded encodings, which have more functionality). Our aim is only to give necessary condition for the assumptions to hold in candidate multilinear maps. The status of current candidate multilinear maps is rather uncertain, e.g. it is described in [28] as "break-and-repair mode". Thus, it is hard to argue if our assumptions hold in any concrete instantiation and we leave this as an open question for further investigation.

4.2 A Unifying View on Computational Matrix Problems

In this section we recall some computational problems in the cryptographic literature that we unify as particular instances of KerMDH problems. These problems are listed below, as they appear in the cited references. In the following, all parameters a_i and b_i are assumed to be randomly chosen in $\mathbb{Z}_q$.

1. Find-Rep [9]: Given $([a_1], \ldots, [a_\ell])$, find a nonzero tuple $(x_1, \ldots, x_\ell)$ such that $x_1 a_1 + \ldots + a_\ell x_\ell = 0$.
2. Simultaneous Double Pairing (SDP) [2]: Given the two tuples, $([a_1], [b_1])$ and $([a_2], [b_2])$, find a nonzero tuple $([x_1], [x_2], [x_3])$ such that $x_1 b_1 + x_2 a_1 = 0$, $x_1 b_2 + x_3 a_2 = 0$.
3. Simultaneous Triple Pairing [18]: Given the two tuples, $([a_1], [a_2], [a_3])$ and $([b_1], [b_2], [b_3])$, find a nonzero tuple $([x_1], [x_2], [x_3])$ such that $x_1 a_1 + x_2 a_2 + x_3 a_3 = 0$, $x_1 b_1 + x_2 b_2 + x_3 b_3 = 0$.
4. Simultaneous Pairing [19]: Given $([a_1], [a_2], \ldots, [a_\ell])$ and $([a_1^2], [a_2^2], \ldots, [a_\ell^2])$, find a nonzero tuple $([x_1], \ldots, [x_\ell])$ such that $\sum_{i=1}^\ell x_i a_i = 0$, $\sum_{i=1}^\ell x_i a_i^2 = 0$.
5. 1-Flexible Diffie-Hellman (1-FlexDH) [32]: Given $([1], [a], [b])$, find a triple $([r], [ra], [rab])$ with $r \neq 0$.
6. 1-Flexible Square Diffie-Hellman (1-FlexSDH) [27]: Given $([1], [a])$, find a triple $([r], [ra], [ra^2])$ with $r \neq 0$.
7. ℓ-Flexible Diffie-Hellman (ℓ-FlexDH) [32]: Given $([1], [a], [b])$, find a $(2\ell+1)$-tuple $([r_1], \ldots, [r_\ell], [r_1 a], [r_1 r_2 a], \ldots, [(\prod_{i=1}^\ell r_i)a], [(\prod_{i=1}^\ell r_i)ab])$ such that $r_j \neq 0$ for all $j = 1, \ldots, \ell$.

8. Double Pairing (DP) [18]: In an asymmetric group $(\mathbb{G}, \mathbb{H}, \mathbb{T})$, given a pair of random elements $([a_1]_H, [a_2]_H) \in \mathbb{H}^2$, find a nonzero tuple $([x_1]_G, [x_2]_G)$ such that $[x_1 a_1 + x_2 a_2]_T = [0]_T$.

Notice that Find-Rep is just $(0, \mathcal{U}_{\ell,1})$-KerMDH, SDP is $\mathcal{RL}_2$-KerMDH, the Simultaneous Triple Pairing problem is $\mathcal{U}_2$-KerMDH, the Simultaneous Pairing problem is $\mathcal{P}_{\ell,2}$-KerMDH. DP corresponds to $\mathcal{U}_1$-KerMDH in an asymmetric bilinear setting. On the other hand, 1-FlexDH is $\mathcal{C}_2$-KerMDH, 1-FlexSDH problem is $\mathcal{SC}_2$-KerMDH and ℓ-FlexDH for $\ell > 1$ is the only one which is not in the KerMDH problem family. However, ℓ-FlexDH $\Rightarrow \mathcal{C}_{\ell+1}$-KerMDH. Getting the last three results require a bit more work, as we show in the next two lemmas.

Lemma 11. *1-FlexDH* $= \mathcal{C}_2$-KerMDH *and 1-FlexSDH* $= \mathcal{SC}_2$-KerMDH.

Proof. The proof of the first statement is obvious from the fact that the solutions $([r], [ra], [rab])$ of the 1-FlexDH problem instance $([1], [a], [b])$ correspond exactly to the nonzero vectors in $\ker \mathbf{A}^\top$ for $\mathbf{A} = \begin{pmatrix} -a & 0 \\ 1 & -b \\ 0 & 1 \end{pmatrix}$. The second statement is proven in a similar way. □

Lemma 12. *ℓ-FlexDH $\Rightarrow \mathcal{C}_{\ell+1}$-KerMDH.*

Proof. Given a ℓ-FlexDH problem instance $([1], [a], [b])$, pick random $r_2, \ldots, r_\ell \in \mathbb{Z}_q^*$ and compute the matrix $[\mathbf{A}]$ where

$$\mathbf{A} = \begin{pmatrix} -a & & & & 0 \\ 1 & -r_2 & & & \\ & \ddots & \ddots & & \\ & & 1 & -r_\ell & \\ & & & 1 & -b \\ 0 & & & & 1 \end{pmatrix}$$

Then, run the $\mathcal{C}_{\ell+1}$-KerMDH solver on $[\mathbf{A}]$, obtaining the vector $([r_1], [r_1 a], [r_1 r_2 a], \ldots, [r_1 \cdots r_\ell a], [r_1 \cdots r_\ell ab])$ for some $r_1 \in \mathbb{Z}_q^*$, which along with $([r_2], \ldots, [r_\ell])$ solves the ℓ-FlexDH problem. □

5 Reduction and Separation of Kernel Diffie-Hellman Problems

In this section we prove that the most important matrix distribution families $\mathcal{U}_{\ell,k}$, $\mathcal{L}_k$, $\mathcal{SC}_k$, $\mathcal{C}_k$ and $\mathcal{RL}_k$ define families of KerMDH problems with **strictly** increasing hardness, as we precisely state in Theorem 2, at the end of the section. By 'strictly increasing' we mean that

1. there are known reductions of the smaller problems to the larger problems (in terms of k) within each family, and
2. there are no black-box reductions in the other way in the multilinear generic group model.

This result shows the necessity of using $\mathcal{D}_{\ell,k}$-KerMDH Assumptions for $k > 2$. A similar result is known for the corresponding $\mathcal{D}_{\ell,k}$-MDDH problems. Indeed, one can easily prove a separation between large and small decisional problems. Observe that any efficient m-linear map can efficiently solve any $\mathcal{D}_{\ell,k}$-MDDH problem with $k \leq m - 1$, and therefore every two $\mathcal{D}_{\ell,k}$-MDDH and $\mathcal{D}_{\widetilde{\ell},\widetilde{k}}$-MDDH problems with $\widetilde{k} < k$ are separated by an oracle computing a k-linear map.

However, when dealing with the computational $\mathcal{D}_{\ell,k}$-KerMDH family, no such a trivial argument is known to exist. Actually, a m-linear map does not seem to help to solve any $\mathcal{D}_{\ell,k}$-KerMDH problem with $k > 1$. Furthermore, the m-linear map seems to be useless for any (reasonable) reduction between KerMDH problems defined over the same group. Indeed, all group elements involved in the problem instances and their solutions belong to the base group $\mathbb{G}$, and the result of computing any m-linear map is an element in $\mathbb{G}_m$, where no efficient map from $\mathbb{G}_m$ back to $\mathbb{G}$ is supposed to exist.

5.1 Separation

In this section we firstly show the non-existential part of Theorem 2. Namely, we show that there is no black-box reduction in the multilinear generic group model (described in Section 3) from $\mathcal{D}_{\ell,k}$-KerMDH to $\mathcal{D}_{\widetilde{\ell},\widetilde{k}}$-KerMDH for $k > \widetilde{k}$, assuming that the two matrix distributions $\mathcal{D}_{\ell,k}$ and $\mathcal{D}_{\widetilde{\ell},\widetilde{k}}$ are hard (see Definition 11). Before proving the main result we need some technical lemmas and also a new geometrical notion defined on a family of subspaces of a vector space, named *t-Elusiveness*.

In the first lemma we show that the natural (black-box, algebraic) reductions between KerMDH problems have a very special form. Observe that a black-box reduction to a flexible problem must work for any adversary solving it. In particular, the reduction should work for **any** solution given by this adversary, or for **any** probability distribution of the solutions given by it. Informally, the lemma states that the output of a successful reduction can always be computed in essentially two ways:

1. By just applying a (randomized) linear map to the answer given by the adversary in the last call. Therefore, all possibly existing previous calls to the adversary are just used to prepare the last one.
2. By just ignoring the last call to the adversary and using only the information gathered in the previous ones.

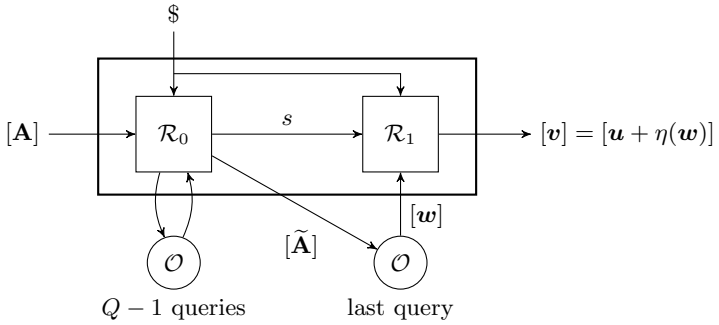

Fig. 5. Splitting of the black-box reduction.

Let $\mathcal{R}^{\mathcal{O}}$ be a black-box reduction of $\mathcal{D}_{\ell,k}$-KerMDH to $\mathcal{D}_{\widetilde{\ell},\widetilde{k}}$-KerMDH, in the purely algebraic multilinear generic group model, discussed in Section 3, for some matrix distributions $\mathcal{D}_{\ell,k}$ and $\mathcal{D}_{\widetilde{\ell},\widetilde{k}}$. Namely, $\mathcal{R}^{\mathcal{O}}$ solves $\mathcal{D}_{\ell,k}$-KerMDH with a non-negligible probability by making $Q \geq 1$ queries to an oracle $\mathcal{O}$ solving $\mathcal{D}_{\widetilde{\ell},\widetilde{k}}$-KerMDH with probability one. As we aim at ruling out the existence of some reductions, we just consider the best possible case any black-box reduction must be able to handle. Now we split the reduction as $\mathcal{R}^{\mathcal{O}} = (\mathcal{R}_0^{\mathcal{O}}, \mathcal{R}_1)$, where the splitting point is the last oracle call, as shown in Figure 5. We actually use the same splitting in the proof of Lemma 6. More formally, on the input of $[\mathbf{A}]$, for $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}$, and after making $Q-1$ oracle calls, $\mathcal{R}_0^{\mathcal{O}}$ stops by outputting the last query to $\mathcal{O}$, that is a matrix $[\widetilde{\mathbf{A}}]$, where $\widetilde{\mathbf{A}} \in \mathcal{D}_{\widetilde{\ell},\widetilde{k}}$, together with some state information s for $\mathcal{R}_1$. Next, $\mathcal{R}_1$ resumes the execution from s and the answer $[\boldsymbol{w}] \in \mathbb{G}^{\widetilde{\ell}}$ given by the oracle, and finally outputs $[\boldsymbol{v}] \in \mathbb{G}^{\ell}$. Without loss of generality, we assume that both stages $\mathcal{R}_0^{\mathcal{O}}$ and $\mathcal{R}_1$ receive the same random tape, \$ ($\mathcal{R}_1$ can redo the computations performed by $\mathcal{R}_0^{\mathcal{O}}$).

Lemma 13. *There exists an algebraic oracle $\mathcal{O}$ (in the sense of Definition 7), that solves the $\mathcal{D}_{\ell,k}$-KerMDH Problem with probability one.*

Proof. Observe that $\mathcal{D}_{\ell,k}$ only uses group elements both in the instance description and in the solution to the problem. In addition, the problem (input/output relation) can be described by a polynomial map. Indeed, one can use the k-minors of $\mathbf{A}$, which are just polynomials of degree k, to obtain a basis of $\ker \mathbf{A}^{\top}$.

Then the oracle can use parameters $r_1, \ldots, r_{\ell-k}$ as the coefficients of an arbitrary linear combination of the basis vectors. Sampling these parameters uniformly results in an oracle answer uniformly distributed in $\ker \mathbf{A}^\top$. □

Lemma 6 applied to $\mathcal{R}_0^{\mathcal{O}}$ (and using also Lemma 13) implies that only the group elements in s can depend on $\mathbf{A}$. Indeed, the non-group elements in s can only depend on the random tape \$. Now, from Lemma 3 applied to $\mathcal{R}_1$, we know that its output $[\boldsymbol{v}]$ is determined by a polynomial map of total degree at most one in the input group elements (*i.e.*, $\widetilde{\mathbf{A}}$ and the group elements in s), and the coefficients of this polynomial can only depend on \$, and the non-group elements in s, which in turn only depend on \$. Therefore, splitting the polynomial map into two parts, for every fixed \$ and every fixed oracle behavior in the first $Q-1$ oracle calls there exists a vector $u \in \mathbb{Z}_q^\ell$ and a linear map $\eta : \mathbb{Z}_q^{\widetilde{\ell}} \to \mathbb{Z}_q^\ell$ such that we can write $\boldsymbol{v} = \boldsymbol{u} + \eta(\boldsymbol{w})$, where $\boldsymbol{u}$ actually depends on the group elements in s. The important fact here is that η can only depend on \$, but not on $\mathbf{A}$.

Lemma 14. *Let $\mathcal{R}^{\mathcal{O}} = (\mathcal{R}_0^{\mathcal{O}}, \mathcal{R}_1)$ be a black-box reduction from $\mathcal{D}_{\ell,k}$-KerMDH to $\mathcal{D}_{\widetilde{\ell},\widetilde{k}}$-KerMDH, in the purely algebraic multilinear generic group model, making $Q \geq 1$ calls to an oracle $\mathcal{O}$ solving the latter with probability one. If $\mathcal{R}^{\mathcal{O}}$ succeeds with a non negligible probability ε then, for every possible behavior of the oracle, either $\Pr(\eta(\boldsymbol{w}) \in S') > negl$ or $\Pr(\boldsymbol{u} \in S') > negl$, where $S' = \ker \mathbf{A}^\top \setminus \{\mathbf{0}\}$, $[\mathbf{A}]$ is the input of $\mathcal{R}^{\mathcal{O}}$, and its output is written as $[\boldsymbol{u} + \eta(\boldsymbol{w})]$, for some $\boldsymbol{u}$ only depending on the state output by $\mathcal{R}_0^{\mathcal{O}}$, $[\boldsymbol{w}]$ is the answer to the Q-th oracle query, and $\eta : \mathbb{Z}_q^{\widetilde{l}} \to \mathbb{Z}_q^l$ is a (randomized) linear map that only depends on the random tape of $\mathcal{R}^{\mathcal{O}}$.*

Proof. Let us denote $S = \ker \mathbf{A}^\top$, where $[\mathbf{A}]$ is the input to $\mathcal{R}^{\mathcal{O}}$, and $S' = S \setminus \{\mathbf{0}\}$. Analogously, $\widetilde{S} = \ker \widetilde{\mathbf{A}}^\top$, where $[\widetilde{\mathbf{A}}]$ is the Q-th oracle query, and $\widetilde{S}' = \widetilde{S} \setminus \{\mathbf{0}\}$. From the discussion preceding the lemma, we know that $\boldsymbol{u}$ and η are well-defined and fulfil the required properties. In particular, η depends only on the random tape, \$, of $\mathcal{R}^{\mathcal{O}}$.

As a black-box reduction, $\mathcal{R}^{\mathcal{O}}$ is successful means that it is successful for every possible behavior of the oracle in its Q queries, with a success probability at least ε. We arbitrarily fix its behavior in the first $Q-1$ queries. Concerning the last one, for all $\boldsymbol{w} \in \widetilde{S}'$, $\Pr(\boldsymbol{u} + \eta(\boldsymbol{w}) \in S') > \varepsilon$, where the probability is computed with respect to \$ and the randomness of $[\mathbf{A}]$. Now, defining

$$p_{\boldsymbol{w}} = \Pr(\boldsymbol{u} \in S \wedge \boldsymbol{u} + \eta(\boldsymbol{w}) \in S')$$

$$r_{\boldsymbol{w}} = \Pr(\boldsymbol{u} \notin S \wedge \boldsymbol{u} + \eta(\boldsymbol{w}) \in S')$$

we have $p_{\boldsymbol{w}} + r_{\boldsymbol{w}} > \varepsilon$. But not all $r_{\boldsymbol{w}}$ can be non-negligible since the corresponding events are disjoint. Indeed, for any vector $\boldsymbol{w} \neq \mathbf{0}$ and any different $\alpha_1, \alpha_2 \in \mathbb{Z}_q^\times$,

$$\boldsymbol{u} + \eta(\alpha_1 \boldsymbol{w}) \in S,\ \boldsymbol{u} + \eta(\alpha_2 \boldsymbol{w}) \in S \quad \Rightarrow \quad (\alpha_2 - \alpha_1)\boldsymbol{u} \in S \quad \Rightarrow \quad \boldsymbol{u} \in S$$

and then $\sum_{\alpha \in \mathbb{Z}_q^\times} r_{\alpha \boldsymbol{w}} \leq 1$. Thus, there exists α_m such that $r_{\alpha_m \boldsymbol{w}} \leq \frac{1}{q-1}$, which implies $p_{\alpha_m \boldsymbol{w}} > \varepsilon - \frac{1}{q-1}$. Now, we split $p_{\alpha_m \boldsymbol{w}}$, depending on whether $\boldsymbol{u} \in S'$ or $\boldsymbol{u} = \mathbf{0}$,

$$\begin{aligned} p_{\alpha_m \boldsymbol{w}} &= \Pr(\boldsymbol{u} = \mathbf{0} \wedge \eta(\boldsymbol{w}) \in S') + \Pr(\boldsymbol{u} \in S' \wedge \boldsymbol{u} + \eta(\alpha_m \boldsymbol{w}) \in S') \leq \\ &\leq \Pr(\eta(\boldsymbol{w}) \in S') + \Pr(\boldsymbol{u} \in S') \end{aligned}$$

and conclude that either $\Pr(\boldsymbol{u} \in S') > negl$ or for all nonzero $\boldsymbol{w} \in \widetilde{S}'$, $\Pr(\eta(\boldsymbol{w}) \in S') > negl$. However, which one is true could depend on the particular behavior of the oracle in the first $Q-1$ calls. □

The following property of the hard matrix distributions allows us to prove that indeed in the last lemma $\Pr(\eta(\boldsymbol{w}) \in S \setminus \{\mathbf{0}\}) \in negl$.

Definition 15 (t-Elusiveness). *A family of subspaces $\mathcal{S}$ of a vector space X over the finite field $\mathbb{Z}_q$ is called t-elusive for some $t < \dim X$ if for all t-dimensional subspaces $F \subset X$, $\Pr(F \cap S \neq \{\mathbf{0}\}) \in negl$, where the probability is computed with respect to the choice of $S \in \mathcal{S}$. A matrix distribution $\mathcal{D}_{\ell,k}$ is called t-elusive if the family $\{\ker \mathbf{A}^\top\}_{\mathbf{A} \in \mathcal{D}_{\ell,k}}$ is t-elusive.*

Lemma 15. *If a matrix distribution $\mathcal{D}_{\ell,k}$ is hard (as given in Definition 11) then $\mathcal{D}_{\ell,k}$ is k-elusive.*

Proof. By definition, given a non-k-elusive matrix distribution $\mathcal{D}_{\ell,k}$, there exists a k-dimensional vector subspace $F \subset \mathbb{Z}_q^\ell$ such that $\Pr_{\mathbf{A}\leftarrow\mathcal{D}_{\ell,k}}(F \cap \ker \mathbf{A}^\top \neq \{\mathbf{0}\}) = \varepsilon > negl$. F can be efficiently computed from the description of $\mathcal{D}_{\ell,k}$ with standard tools from linear algebra.

Let $\mathbf{M} \in \mathbb{Z}_q^{k\times\ell}$ be a maximal rank matrix such that $\operatorname{Im}\mathbf{M}^\top = F$. Then, $\dim(F \cap \ker \mathbf{A}^\top) = \dim(\operatorname{Im}\mathbf{M}^\top \cap \ker \mathbf{A}^\top) \leq \dim\ker(\mathbf{A}^\top\mathbf{M}^\top) = \dim\ker(\mathbf{MA})^\top = \dim\ker(\mathbf{MA})$, as $\mathbf{MA}$ is a $k \times k$ square matrix. Thus, we know that

$$\Pr_{\mathbf{A}\leftarrow\mathcal{D}_{\ell,k}}(\operatorname{rank}(\mathbf{MA}) < k) \geq \varepsilon$$

Now we show how to solve the $\mathcal{D}_{\ell,k}$-MDDH problem with advantage almost ε on some k-linear group $\mathbb{G}$, by means of a k-linear map. Let $[(\mathbf{A}\|\boldsymbol{z})]$ be an instance of the $\mathcal{D}_{\ell,k}$-MDDH problem. In a 'real' instance $\boldsymbol{z} = \mathbf{A}\boldsymbol{x}$ for a uniformly distributed vector $\boldsymbol{x} \in \mathbb{Z}_q^k$, while in a 'random' instance, $\boldsymbol{z}$ is uniformly distributed $\mathbb{Z}_q^\ell$. A distinguisher can efficiently compute $[\mathbf{MA}]$ and $[\mathbf{M}\boldsymbol{z}]$. Observe that in a 'real' instance $\operatorname{rank}(\mathbf{MA}\|\mathbf{M}\boldsymbol{z}) = \operatorname{rank}(\mathbf{MA}\|\mathbf{MA}\boldsymbol{x}) = \operatorname{rank}(\mathbf{MA})$, while in a 'random' instance $\mathbf{M}\boldsymbol{z}$ is uniformly distributed in $\mathbb{Z}_q^k$. Therefore, for a 'random' instance there is a non-negligible probability, greater than $\varepsilon - \frac{1}{q}$, that $\operatorname{rank}(\mathbf{MA}) < k$ and $\operatorname{rank}(\mathbf{MA}\|\mathbf{M}\boldsymbol{z}) = \operatorname{rank}(\mathbf{MA}) + 1$, because $\mathbf{M}\boldsymbol{z} \in \operatorname{Im}(\mathbf{MA})$ occurs only with a negligible probability $< \frac{1}{q}$. Then, the distinguisher can efficiently tell apart the two cases because with a k-linear map at hand computing the rank of a $k \times k$ or a $k \times k + 1$ matrix can be done efficiently. □

In the next theorem we use the k-elusiveness to prove that $\Pr(\boldsymbol{u} \in \ker \mathbf{A}^\top \setminus \{\mathbf{0}\}) > negl$ for all possible behaviors of the oracle in the first $Q-1$ calls. This actually implies that the reduction can directly output $\boldsymbol{u}$, and only $Q-1$ oracle calls are actually needed. Therefore, by the descent method we show that no successful reduction exists unless $\mathcal{D}_{\ell,k}$-KerMDH is easy. However, we still need an additional technical lemma.

Lemma 16. *Consider integers $l = k+d$, $\widetilde{l} = \widetilde{k}+\widetilde{d}$ such that $k, d, \widetilde{k}, \widetilde{d} > 0$ and $k > \widetilde{k}$. Let $\eta : \mathbb{Z}_q^{\widetilde{l}} \to \mathbb{Z}_q^l$ be a linear map. Then, there exists a subspace F of $Im\,\eta$ of dimension at most k such that for all $\widetilde{d}$-dimensional subspaces $\widetilde{S}$ of $\mathbb{Z}_q^{\widetilde{l}}$, either $\widetilde{S} \subset \ker\eta$ or $\dim F \cap \eta(\widetilde{S}) \geq 1$.*

Proof. If $\operatorname{rank}\eta \leq k$ it suffices to take $F = \operatorname{Im}\eta$. Indeed, if $\widetilde{S} \not\subset \ker\eta$, *i.e.*, $\eta(\widetilde{S}) \neq \{\mathbf{0}\}$, then $\dim F \cap \eta(\widetilde{S}) = \dim\eta(\widetilde{S}) \geq 1$. Otherwise, $\operatorname{rank}\eta > k$, let F a subspace of $\operatorname{Im}\eta$ of dimension k, using the Grassman's formula,

$$\begin{aligned}\dim F \cap \eta(\widetilde{S}) &= \dim F + \dim\eta(\widetilde{S}) - \dim(F + \eta(\widetilde{S})) \geq k + \dim\eta(\widetilde{S}) - \operatorname{rank}\eta \geq \\ &\geq k + \dim\widetilde{S} - \dim\ker\eta - \operatorname{rank}\eta = k + \widetilde{d} - \widetilde{l} = k - \widetilde{k} \geq 1\end{aligned}$$

□

Theorem 1. *Let $\mathcal{D}_{\ell,k}$ be k-elusive. If there exists a black-box reduction in the purely algebraic multilinear generic group model from $\mathcal{D}_{\ell,k}$-KerMDH to another problem $\mathcal{D}_{\widetilde{\ell},\widetilde{k}}$-KerMDH with $\widetilde{k} < k$, then $\mathcal{D}_{\ell,k}$-KerMDH is easy.*

Proof. Let us assume the existence of the claimed reduction, $\mathcal{R}^{\mathcal{O}} = (\mathcal{R}_0^{\mathcal{O}}, \mathcal{R}_1)$, making $Q \geq 1$ oracle queries, where Q is minimal, and with a success probability ε. Then, by Lemma 14, its output can be written as $[\boldsymbol{u} + \eta(\boldsymbol{w})]$, where $\eta : \mathbb{Z}_q^{\widetilde{l}} \to \mathbb{Z}_q^l$ is a (randomized) linear map that does not depend on the particular choice of the matrix $\mathbf{A}$ in the $\mathcal{D}_{\ell,k}$-KerMDH input instance, but only on the random tape of the reduction. Let us denote as above $S = \ker\mathbf{A}^\top$, and $S' = S \setminus \{\mathbf{0}\}$. Analogously, $\widetilde{S} = \ker\widetilde{\mathbf{A}}^\top$, where $\widetilde{\mathbf{A}} \leftarrow \mathcal{D}_{\widetilde{\ell},\widetilde{k}}$ and $\widetilde{S}' = \widetilde{S} \setminus \{\mathbf{0}\}$.

We now prove that in Lemma 14, for any possible behavior of the oracle in the first $Q-1$ calls, there exists a particular behavior in the last call such that $\Pr(\eta(\boldsymbol{w}) \in S')$ is negligible. Namely, the Q-th query

is answered by $\mathcal{O}$ by choosing a uniformly distributed $\boldsymbol{w} \in \widetilde{S}'$ (as required to be algebraic, according to Definition 7). Indeed, $\Pr(\eta(\boldsymbol{w}) \in S') = \Pr(\eta(\boldsymbol{w}) \in S) - \Pr(\eta(\boldsymbol{w}) = 0)$. Now, developing the second term,

$$\begin{aligned}
\Pr(\eta(\boldsymbol{w}) = 0) &= \Pr(\eta(\boldsymbol{w}) = 0 \mid \widetilde{S} \subset \ker \eta) \Pr(\widetilde{S} \subset \ker \eta) + \\
&\quad + \Pr(\eta(\boldsymbol{w}) = 0 \mid \widetilde{S} \not\subset \ker \eta) \Pr(\widetilde{S} \not\subset \ker \eta) = \\
&= \Pr(\widetilde{S} \subset \ker \eta) + \Pr(\boldsymbol{w} \in \widetilde{S} \cap \ker \eta \mid \widetilde{S} \not\subset \ker \eta) \Pr(\widetilde{S} \not\subset \ker \eta) = \\
&= \Pr(\widetilde{S} \subset \ker \eta) + \mathit{negl}
\end{aligned}$$

where the last equality uses that the probability that a vector uniformly distributed in $\widetilde{S}'$ belongs to a proper subspace of $\widetilde{S}'$ is negligible. Analogously,

$$\begin{aligned}
\Pr(\eta(\boldsymbol{w}) \in S) &= \Pr(\eta(\boldsymbol{w}) \in S \mid \eta(\widetilde{S}) \subset S) \Pr(\eta(\widetilde{S}) \subset S) + \\
&\quad + \Pr(\eta(\boldsymbol{w}) \in S \mid \eta(\widetilde{S}) \not\subset S) \Pr(\eta(\widetilde{S}) \not\subset S) = \\
&= \Pr(\eta(\widetilde{S}) \subset S) + \Pr(\boldsymbol{w} \in \widetilde{S} \cap \eta^{-1}(S) \mid \eta(\widetilde{S}) \not\subset S) \Pr(\eta(\widetilde{S}) \not\subset S) = \\
&= \Pr(\eta(\widetilde{S}) \subset S) + \mathit{negl}
\end{aligned}$$

Thus, $\Pr(\eta(\boldsymbol{w}) \in S') = \Pr(\eta(\widetilde{S}) \subset S) - \Pr(\widetilde{S} \subset \ker \eta) + \mathit{negl}$. Now, using Lemma 16, we know that there exists a subspace F of dimension at most k such that if $\widetilde{S} \not\subset \ker \eta$, then $\dim F \cap \eta(\widetilde{S}) \geq 1$. Therefore $\Pr(\eta(\widetilde{S}) \subset S) - \Pr(\widetilde{S} \subset \ker \eta) \leq \Pr(\eta(\widetilde{S}) \subset S \wedge \dim F \cap \eta(\widetilde{S}) \geq 1) \leq \Pr(\dim F \cap S \geq 1)$. Due to the k-elusiveness of $\mathcal{D}_{\ell,k}$, from Lemma 15, the last probability is negligible. Namely, it is upper bounded by $\mathbf{Adv}_{\mathcal{D}_{\ell,k}\text{-MDDH}} + \frac{1}{q}$, where $\mathbf{Adv}_{\mathcal{D}_{\ell,k}\text{-MDDH}}$ denotes the advantage of a distinguisher for the $\mathcal{D}_{\ell,k}$-MDDH problem. By Lemma 14,

$$\Pr(\boldsymbol{u} \in S \setminus \{\mathbf{0}\}) > \varepsilon - \frac{1}{q-1} - \mathbf{Adv}_{\mathcal{D}_{\ell,k}\text{-MDDH}} - \frac{1}{q},$$

for any possible behavior of the oracle in the first $Q-1$ calls. Therefore, we can modify the reduction $\mathcal{R}$ to output $\boldsymbol{u}$, without making the Q-th oracle call. The modified reduction is also successful, essentially with the same probability ε, with only $Q-1$ oracle calls, which contradicts the assumption that Q is minimal. In summary, if the claimed reduction exists then there also exists an algorithm (a "reduction with $Q = 0$") directly solving $\mathcal{D}_{\ell,k}$-KerMDH without the help of any oracle and with the same success probability. □

Now we consider the contrapositive statement, that directly applies to the known families of hard matrix distributions.

Corollary 1. *If a matrix distribution family $\{\mathcal{D}_{\ell,k}\}$ is hard then for any $\mathcal{D}_{\ell,k}$ and $\mathcal{D}_{\widetilde{\ell},\widetilde{k}}$ in the family with $k > \widetilde{k}$ there is no black-box reduction in the multilinear generic group model from $\mathcal{D}_{\ell,k}$-KerMDH to $\mathcal{D}_{\widetilde{\ell},\widetilde{k}}$-KerMDH.*

Proof. Since all $\mathcal{D}_{\ell,k}$-MDDH problems in the family are generically hard on a k-linear group, we know that $\mathcal{D}_{\ell,k}$ is k-elusive by Lemma 15, and also $\mathcal{D}_{\ell,k}$-KerMDH is hard in that group (otherwise, any solution to $\mathcal{D}_{\ell,k}$-KerMDH can be used to solve $\mathcal{D}_{\ell,k}$-MDDH). By the above theorem, no black-box reduction in the multilinear generic group model from $\mathcal{D}_{\ell,k}$-KerMDH to $\mathcal{D}_{\widetilde{\ell},\widetilde{k}}$-KerMDH can exist for $k > \widetilde{k}$. □

5.2 Algebraic Reductions

In contrast to the previous negative results we now show how to build some natural reductions among different families of KerMDH problems of the same size. Thanks to the algebraic nature of matrix distributions it is easy to find some generic reductions among the corresponding problems.

Definition 16. *We say that $\mathcal{D}^1_{\ell,k}$ is algebraically reducible to $\mathcal{D}^2_{\ell,k}$ if there exist two efficiently samplable matrix distributions, $\mathcal{L}$ which outputs a matrix $\mathbf{L} \in \mathbb{Z}_q^{\ell\times\ell}$ matrix and $\mathcal{R}$ which outputs a matrix $\mathbf{R} \in \mathbb{Z}_q^{k\times k}$, such that given $\mathbf{A} \leftarrow \mathcal{D}^1_{\ell,k}$ the distribution of the matrix $\mathbf{LAR}$ is negligibly close to $\mathcal{D}^2_{\ell,k}$. In this case we write $\mathcal{D}^1_{\ell,k} \overset{a}{\Rightarrow} \mathcal{D}^2_{\ell,k}$.*

We note that since we assume that the matrices output by either of the distributions $\mathcal{D}^1_{\ell,k}$, $\mathcal{D}^2_{\ell,k}$ have full rank with overwhelming probability, the distributions $\mathcal{L}, \mathcal{R}$ must output full rank matrices also with overwhelming probability. We provide two examples of algebraic reductions: Taking random $\mathbf{L}$ and $\mathbf{R}$ gives the reduction $\mathcal{D}_{\ell,k} \overset{a}{\Rightarrow} \mathcal{U}_{\ell,k}$ for any matrix distribution $\mathcal{D}_{\ell,k}$, and considering $\mathbf{L}$ the identity matrix and $\mathbf{R}$ a random invertible diagonal matrix, we obtain $\mathcal{L}_k \overset{a}{\Rightarrow} \mathcal{RL}_k$.

The notion of algebraic reducibility is useful to find reductions among the MDDH problems and also the Kernel problems.

Lemma 17. *$\mathcal{D}^1_{\ell,k} \overset{a}{\Rightarrow} \mathcal{D}^2_{\ell,k}$ implies both $\mathcal{D}^1_{\ell,k}$-MDDH $\Rightarrow \mathcal{D}^2_{\ell,k}$-MDDH and $\mathcal{D}^1_{\ell,k}$-KerMDH $\Rightarrow \mathcal{D}^2_{\ell,k}$-KerMDH.*

Proof. Given instance of the $\mathcal{D}^1_{\ell,k}$-MDDH problem, $([\mathbf{A}], [\boldsymbol{z}])$, the tuple $([\mathbf{LAR}], [\mathbf{L}\boldsymbol{z}])$, with $\mathbf{L} \leftarrow \mathcal{L}$, $\mathbf{R} \leftarrow \mathcal{R}$ is a properly distributed instance of the $\mathcal{D}^2_{\ell,k}$-MDDH problem. Indeed, it is easy to see that 'real' instances are transformed into 'real' instances, and 'random' instances into 'random' ones.

On the other hand, we show that given an algorithm $\mathcal{A}$ which solves $\mathcal{D}^2_{\ell,k}$-KerMDH there exists another algorithm which solves $\mathcal{D}^1_{\ell,k}$-KerMDH with the same probability. Given $\mathbf{A} \leftarrow \mathcal{D}^1_{\ell,k}$-KerMDH problem, sample two matrices $\mathbf{L} \leftarrow \mathcal{L}$, $\mathbf{R} \leftarrow \mathcal{R}$ and construct an instance $\mathbf{LAR}$ of $\mathcal{D}^2_{\ell,k}$. Let $[\boldsymbol{x}]$ be the output of $\mathcal{A}$ on input $[\mathbf{LAR}]$, that is, $\boldsymbol{x}$ is a nonzero vector such that $\boldsymbol{x}^\top \mathbf{LAR} = \mathbf{0}^\top$. Since $\mathbf{L}$ and $\mathbf{R}$ are invertible with overwhelming probability, $\boldsymbol{x}^\top \mathbf{LA} = \mathbf{0}^\top$ also holds. Then output the nonzero vector $[\mathbf{L}^\top \boldsymbol{x}]$ as a solution to the $\mathcal{D}^1_{\ell,k}$-KerMDH problem. □

From the above results, it is straightforward that $\mathcal{D}_{\ell,k}$-KerMDH $\Rightarrow \mathcal{U}_{\ell,k}$-KerMDH, for any matrix distribution $\mathcal{D}_{\ell,k}$, and $\mathcal{L}_k$-KerMDH $\Rightarrow \mathcal{RL}_k$-KerMDH.

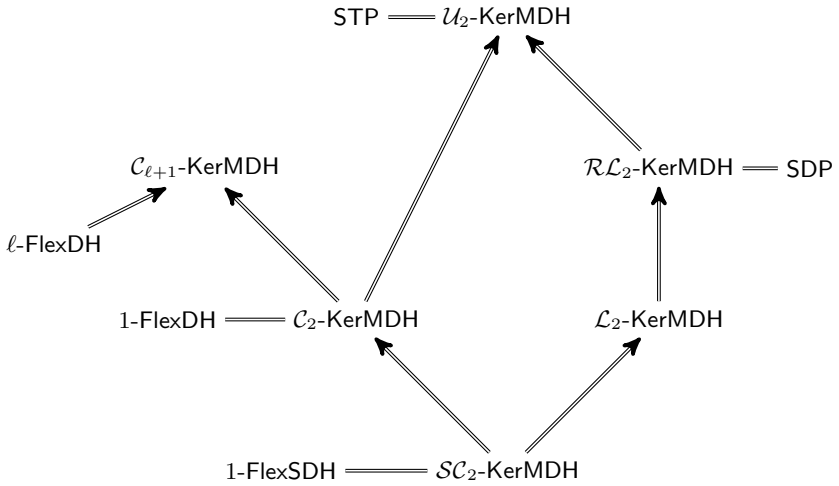

Fig. 6. Equivalences and reductions between some small KerMDH problems.

5.3 Increasing Families of KerMDH Problems

Most matrix distributions, like $\mathcal{U}_{\ell,k}$, $\mathcal{L}_k$, $\mathcal{SC}_k$, $\mathcal{C}_k$ and $\mathcal{RL}_k$, are indeed families parameterized by their size k. The negative results in Corollary 1 prevent us from finding reductions from larger to smaller

KerMDH problems. Nevertheless, we provide here some examples of (tight) reductions going in the other way, within each of the previous families.

Lemma 18. *$\mathcal{U}_{\widetilde{\ell},\widetilde{k}}$-KerMDH $\Rightarrow$ $\mathcal{U}_{\ell,k}$-KerMDH for any $\widetilde{k} \leq k$, $\widetilde{\ell} > \widetilde{k}$ and $\ell > k$.*

Proof. We divide the proof into two steps: Firstly, assume that $\widetilde{\ell} = \widetilde{k} + 1$, $k \geq \widetilde{k}$, $\ell \geq k + 1$. Given an instance $[\widetilde{\mathbf{A}}]$, with $\widetilde{\mathbf{A}} \leftarrow \mathcal{U}_{\widetilde{k}+1,\widetilde{k}}$, we choose a full-rank matrix $\mathbf{L} \in \mathbb{Z}_q^{\ell \times (k+1)}$ and compute $[\mathbf{A}] = \mathbf{L}([\widetilde{\mathbf{A}}] \oplus [\mathbf{I}])$, where $\mathbf{I}$ is the identity matrix of size $(k - \widetilde{k}) \times (k - \widetilde{k})$ and $\oplus$ operation denotes diagonal block matrix concatenation. That is

$$U \oplus V = \begin{pmatrix} U & 0 \\ 0 & V \end{pmatrix}.$$

Clearly, the probability distribution of the new matrix is statistically close to the uniform distribution in $\mathbb{Z}_q^{\ell \times k}$. Any vector $[\boldsymbol{x}]$, obtained from a solver of $\mathcal{U}_{\ell,k}$-KerMDH, such that $\boldsymbol{x} \in \ker \mathbf{A}^\top \setminus \{\mathbf{0}\}$ can be transformed into $[\widetilde{\boldsymbol{x}}]$ such that $\widetilde{\boldsymbol{x}} \in \ker \widetilde{\mathbf{A}}^\top \setminus \{\mathbf{0}\}$ with overwhelming probability,[15] by just letting $[\widetilde{\boldsymbol{x}}]$ to be the first $\widetilde{k} + 1$ components of $\mathbf{L}^\top[\boldsymbol{x}]$. Thus, we have built a tight reduction $\mathcal{U}_{\widetilde{k}+1,\widetilde{k}}$-KerMDH $\Rightarrow$ $\mathcal{U}_{\ell,k}$-KerMDH.

The second step, $k = \widetilde{k}$, $\widetilde{\ell} > \ell = \widetilde{k} + 1$, is simpler. Given an instance $[\widetilde{\mathbf{A}}]$, with $\widetilde{\mathbf{A}} \leftarrow \mathcal{U}_{\widetilde{\ell},\widetilde{k}}$, define the matrix $[\mathbf{A}]$ to be the upper $\widetilde{k} + 1$ rows of $[\widetilde{\mathbf{A}}]$. Clearly $\mathbf{A}$ follows the uniform distribution in $\mathbb{Z}_q^{(\widetilde{k}+1)\times\widetilde{k}}$. Now, any vector $[\boldsymbol{x}]$ such that $\boldsymbol{x} \in \ker \mathbf{A}^\top \setminus \{\mathbf{0}\}$ can be transformed into $[\widetilde{\boldsymbol{x}}]$ such that $\widetilde{\boldsymbol{x}} \in \ker \widetilde{\mathbf{A}}^\top \setminus \{\mathbf{0}\}$, by just padding $\boldsymbol{x}$ with $\widetilde{\ell} - \widetilde{k} - 1$ zeros. Thus, $\mathcal{U}_{\widetilde{\ell},\widetilde{k}}$-KerMDH $\Rightarrow$ $\mathcal{U}_{\widetilde{k}+1,\widetilde{k}}$-KerMDH. By concatenating the two tight reductions we obtain the general case. □

Lemma 19. *For $\mathcal{D}_k = \mathcal{L}_k$, $\mathcal{SC}_k$, $\mathcal{C}_k$ and $\mathcal{RL}_k$, $\mathcal{D}_k$-KerMDH $\Rightarrow$ $\mathcal{D}_{k+1}$-KerMDH.*

Proof. We start with the case $\mathcal{D}_k = \mathcal{L}_k$. Observe that given a matrix $\widetilde{\mathbf{A}} \leftarrow \mathcal{L}_k$, with parameters $a_1, \ldots, a_k$, we can build a matrix $\mathbf{A}$ following the distribution $\mathcal{L}_{k+1}$, by adding an extra row and column to $\widetilde{\mathbf{A}}$ corresponding to new random parameter $a_{k+1} \in \mathbb{Z}_q$. Moreover, given $\boldsymbol{x} = (x_1, \ldots, x_{k+2}) \in \ker \mathbf{A}^\top \setminus \{\mathbf{0}\}$, the vector $\widetilde{\boldsymbol{x}} = (x_1, \ldots, x_k, x_{k+2})$ is in $\ker \widetilde{\mathbf{A}}^\top \setminus \{\mathbf{0}\}$ (except for a negligible probability due to the possibility that $a_{k+1} = 0$ and $\widetilde{\boldsymbol{x}} = \mathbf{0}$, while $\boldsymbol{x} \neq \mathbf{0}$). The reduction consists of choosing a random a_{k+1}, then building $[\mathbf{A}]$ from $[\widetilde{\mathbf{A}}]$ as above, and finally obtaining $[\widetilde{\boldsymbol{x}}]$ from $[\boldsymbol{x}]$ by deleting the $(k+1)$-th coordinate.

Similarly, from a matrix $\widetilde{\mathbf{A}} \leftarrow \mathcal{SC}_k$, with parameter a, we can obtain a matrix $\mathbf{A}$ following $\mathcal{SC}_{k+1}$ by adding a new row and column to $\widetilde{\mathbf{A}}$. Now given $\boldsymbol{x} = (x_1, \ldots, x_{k+2}) \in \ker \mathbf{A}^\top \setminus \{\mathbf{0}\}$, it is easy to see that the vector $\widetilde{\boldsymbol{x}} = (x_1, \ldots, x_{k+1})$ is always in $\ker \widetilde{\mathbf{A}}^\top \setminus \{\mathbf{0}\}$.

The other implications $\mathcal{C}_k$-KerMDH $\Rightarrow$ $\mathcal{C}_{k+1}$-KerMDH and $\mathcal{RL}_k$-KerMDH $\Rightarrow$ $\mathcal{RL}_{k+1}$-KerMDH are proven by following the same ideas. □

By combining Corollary 1 with the explicit reductions given above, we can now state our main result in this section.

Theorem 2. *The matrix distribution families $\{\mathcal{U}_{\ell,k}\}$, $\{\mathcal{L}_k\}$, $\{\mathcal{SC}_k\}$, $\{\mathcal{C}_k\}$ and $\{\mathcal{RL}_k\}$ define families of* KerMDH *problems with* ***strictly*** *increasing hardness. Namely, for any $\mathcal{D}_{\ell,k}$ and $\mathcal{D}_{\widetilde{\ell},\widetilde{k}}$ belonging to one of the previous families, such that $\widetilde{k} < k$,*

1. *there exists a tight reduction, $\mathcal{D}_{\widetilde{\ell},\widetilde{k}}$-KerMDH $\Rightarrow$ $\mathcal{D}_{\ell,k}$-KerMDH,*
2. *there is no black-box reduction in the multilinear generic group model in the opposite direction.*

6 Applications

We have already mentioned that the Kernel Matrix Diffie-Hellman Assumptions have already found applications in follow-up work, more concretely:

[15] Actually, $\widetilde{\boldsymbol{x}} = \mathbf{0}$ depends on the $(\widetilde{k}+1)$-th column of $\mathbf{L}$ which is independent of $\mathbf{A}$.

1. to generalize and improve previous constructions of QA-NIZK proofs for linear spaces [26],
2. to construct more efficient structure preserving signatures starting from affine algebraic MACS [25],
3. to improve and generalize aggregation of Groth-Sahai proofs [17] or
4. to construct a tightly secure QA-NIZK argument for linear subspaces with unbounded simulation soundness [15].

As a new application, we use our new framework to abstract two constructions of trapdoor commitments. See Section 7 for a discussion on the advantages of instantiating these commitments with the new circulant matrix distribution.

6.1 Definition of a Trapdoor Commitment

As a concrete application, we study how to abstract two constructions of trapdoor commitments in the literature to any Kernel Assumption. We recall the definition of a trapdoor commitment scheme.

Definition 17. *A commitment scheme is a tuple of three algorithms* $(\mathsf{K}, \mathsf{Comm}, \mathsf{Vrfy})$ *such that:*

- K *is a randomized algorithm, which on input the security parameter* 1^λ *outputs a commitment key* ck*,*
- Comm *is a randomized algorithm which, on input the commitment key* ck *and a message* m *in the message space* $\mathcal{M}_{ck}$ *outputs a commitment* c *and an opening* Op*,*
- Vrfy *is a deterministic algorithm which, on input the commitment key* ck*, a message* m *in the message space* $\mathcal{M}_{ck}$ *and an opening* Op*, outputs* 1 *if* Op *is a valid opening of* c *to the message* m *and* 0 *otherwise.*

Correctness requires that

$$\Pr\left[1 \leftarrow \mathsf{Vrfy}(ck, c, m, Op) : ck \leftarrow \mathsf{K}(1^\lambda), m \leftarrow \mathcal{M}_{ck}, (c, Op) \leftarrow \mathsf{Comm}(ck, m)\right] = 1.$$

Definition 18. *A commitment scheme is binding if, for any polynomial-time adversary* $\mathcal{A}$*,*

$$\Pr\left[1 \leftarrow \mathsf{Vrfy}(ck, c, m, Op) \cap 1 \leftarrow \mathsf{Vrfy}(ck, c, m', Op') : ck \leftarrow \mathsf{K}(1^\lambda), (c, m, Op, m', Op') \leftarrow \mathcal{A}(ck)\right]$$

is negligible. It is hiding if, for any polynomial-time adversary $\mathcal{A}$*,*

$$|\Pr\left[b' = b : ck \leftarrow \mathsf{K}(1^\lambda), (m_0, m_1, st) \leftarrow \mathcal{A}(ck), b \leftarrow \{0,1\}, (c, Op) \leftarrow \mathsf{Comm}(ck, m_b), b' \leftarrow \mathcal{A}(st, c)\right] - \frac{1}{2}|$$

is negligible.

Definition 19. *A commitment scheme is trapdoor if* K *additionally outputs a trapdoor key* tk *and there is an efficient algorithm* $\mathsf{TrapdoorEquiv}$ *which, on input* (ck, tk, c, m, Op, m') *outputs* Op' *such that* $1 \leftarrow \mathsf{Vrfy}(ck, c, m', Op')$*. Further, for any pair of valid messages* m*,* m' *and legitimately generated* ck, tk*, it holds that the distributions* (ck, c, Op') *when* $(c, Op) \leftarrow \mathsf{Comm}(ck, m)$*,* $Op' \leftarrow \mathsf{TrapdoorEquiv}(ck, tk, c, m, Op, m')$ *or when* $(c, Op') \leftarrow \mathsf{Comm}(ck, m')$ *are indistinguishable.*

6.2 Generalized Pedersen Commments in Multilinear Groups

In a group $(\mathbb{G}, q, \mathcal{P})$ where the discrete logarithm is hard, the Pedersen commitment is a statistically hiding and computationally binding commitment to a scalar. It can be naturally generalized to several scalars. Abe *et al.* [2] show how to do similar Pedersen type commitments to vectors of group elements. With our new assumption family we can write both the Pedersen commitment and the commitment of [2] as a single construction and generalize it to (ideal) graded encodings.

- $\mathsf{K}(1^\lambda, d, m)$: Let $\mathcal{MG}_m = (e, \mathbb{G}_1, \mathbb{G}_2, \ldots, \mathbb{G}_m, q, \mathcal{P}_1, \ldots, \mathcal{P}_m) \leftarrow \mathsf{MGen}_m(1^\lambda)$. Sample $\mathbf{A} \leftarrow \mathcal{D}_{k+d,k}$. Let $\overline{\mathbf{A}}$ be the first k rows of $\mathbf{A}$ and $\underline{\mathbf{A}}$ the remaining d rows and $\mathbf{T} := \underline{\mathbf{A}}\overline{\mathbf{A}}^{-1}$ (w.l.o.g. we can assume $\overline{\mathbf{A}}$ is invertible). Output $ck := (\mathcal{MG}_m, [\mathbf{A}]_1)$, $tk := (\mathbf{T})$.

- $\mathsf{Comm}(ck, [\boldsymbol{v}]_r)$: To commit to a vector $[\boldsymbol{v}]_r \in \mathbb{G}_r^d$, for any $r < m$, pick $\boldsymbol{s} \leftarrow \mathbb{Z}_q^k$, and output $[\boldsymbol{c}]_{r+1} := e([(\boldsymbol{s}^\top \,||\, \boldsymbol{v}^\top)]_r, [\mathbf{A}]_1) = [(\boldsymbol{s}^\top \,||\, \boldsymbol{v}^\top)\,\mathbf{A}]_{r+1} \in \mathbb{G}_{r+1}^k$, and the opening $Op = ([\boldsymbol{s}]_r)$.
- $\mathsf{Vrfy}(ck, [\boldsymbol{v}]_r, Op)$: Given a message $[\boldsymbol{v}]_r$ and opening $Op = ([\boldsymbol{s}]_r)$, this algorithm outputs 1 if $[\boldsymbol{c}]_{r+1} = e([(\boldsymbol{s}^\top \,||\, \boldsymbol{v}^\top)]_r, [\mathbf{A}]_1)$.
- $\mathsf{TrapdoorEquiv}(ck, tk, [\boldsymbol{c}]_{r+1}, [\boldsymbol{v}]_r, Op, [\boldsymbol{v}']_r)$: On a commitment $[\boldsymbol{c}]_{r+1} \in \mathbb{G}_{r+1}^k$ to message $[\boldsymbol{v}]_r$ with opening $Op = ([\boldsymbol{s}]_r)$, compute: $[\boldsymbol{s}']_r := [\boldsymbol{s}]_r + \mathbf{T}^\top[(\boldsymbol{v} - \boldsymbol{v}')]_r \in \mathbb{G}_r^k$. Output $Op' = ([\boldsymbol{s}']_r)$ as the opening of $[\boldsymbol{c}]_{r+1}$ to $[\boldsymbol{v}']_r$.

The analysis is almost identical to [2]. The correctness of the trapdoor opening is straightforward. The hiding property of the commitment is unconditional, while the soundness (at level r) is based on the $(r, m, \mathcal{D}_{\ell,k})$-$\mathsf{KerMDH}$ Assumption. Indeed, given two messages $[\boldsymbol{v}]_r, [\boldsymbol{v}']_r$ with respective openings $[\boldsymbol{s}]_r, [\boldsymbol{s}']_r$, it obviously follows that $[\boldsymbol{w}] := [((\boldsymbol{s} - \boldsymbol{s}')^\top \,||\, (\boldsymbol{v} - \boldsymbol{v}')^\top)]_r$ is a nonzero element in the kernel (in $\mathbb{G}_r$) of $\mathbf{A}^\top$, *i.e.* $e([\boldsymbol{w}^\top]_r, [\mathbf{A}]_1) = [\mathbf{0}]_{r+1}$.

Notice that the Pedersen commitment (to multiple elements) is for messages in $\mathbb{G}_0$ and $\mathbf{A} \leftarrow \mathcal{U}_{d+1,1}$ and soundness is based on the $(0, m, \mathcal{U}_{d+1,1})$-$\mathsf{KerMDH}$. The construction proposed in [2] is for an asymmetric bilinear group $\mathcal{AG}_2 = (e, \mathbb{G}, \mathbb{H}, \mathbb{T}, q, \mathcal{P}, \mathcal{Q})$, and in this case messages are vectors in the group $\mathbb{H}$ and the commitment key consists of elements in $\mathbb{G}$, *i.e.* $ck = (\mathcal{AG}_2, [\mathbf{A}]_G)$, $\mathbf{A} \leftarrow \mathcal{U}_{d+1,1}$. Further, a previous version of the commitment scheme of [2] in symmetric bilinear groups (in [18]) corresponds to our construction with $\mathbf{A} \leftarrow \mathcal{U}_{2+d,2}$.

6.3 Group-to-Group Commitments

The commitments of the previous section are "shrinking" because they map a vector of length d in the group $\mathbb{G}_r$ to a vector of length k, for some k independent of and typically smaller than d. Abe *et al.* [3] noted that in some applications it is useful to have "group-to-group" commitments, *i.e.* commitments which are defined in the same group as the vector message. The motivation for doing so in the bilinear case is that these commitments are better compatible with Groth-Sahai proofs.

There is a natural construction of group-to-group commitments which uses the generalized Pedersen commitment of section 6.2, which is denoted as $\mathsf{Ped.C} = (\widetilde{\mathsf{K}}, \widetilde{\mathsf{Comm}}, \widetilde{\mathsf{Vrfy}}, \widetilde{\mathsf{TrapdoorEquiv}})$ in the following.

- $\mathsf{K}(1^\lambda, d, m)$: Run $(\widetilde{ck}, \widetilde{tk}) \leftarrow \widetilde{\mathsf{K}}(1^\lambda, m, d)$, output $ck = \widetilde{ck}$ and $tk = \widetilde{tk}$.
- $\mathsf{Comm}(ck, [\boldsymbol{v}]_r)$: To commit to a vector $[\boldsymbol{v}]_r \in \mathbb{G}_r^d$, for any $0 < r < m$, pick $[\boldsymbol{t}]_{r-1} \leftarrow [\mathbb{G}]_{r-1}^k$. Let $([\tilde{\boldsymbol{c}}]_r, \widetilde{Op} = ([\boldsymbol{s}]_{r-1})) \leftarrow \widetilde{\mathsf{Comm}}(ck, [\boldsymbol{t}]_{r-1})$ and output $c := ([\boldsymbol{t} + \boldsymbol{v}]_r, [\tilde{\boldsymbol{c}}]_r)$ and the opening $Op = ([\boldsymbol{s}]_r)$.
- $\mathsf{Vrfy}(ck, c, [\boldsymbol{v}]_r, Op)$: On input $c = ([\boldsymbol{y}]_r, [\widetilde{\boldsymbol{c}}]_r)$, this algorithm computes $[\widetilde{\boldsymbol{c}}]_{r+1}$ and outputs 1 if $[\boldsymbol{t}]_r := [\boldsymbol{y} - \boldsymbol{v}]_r$ satisfies that $1 \leftarrow \widetilde{\mathsf{Vrfy}}(ck, [\widetilde{\boldsymbol{c}}]_{r+1}, [\boldsymbol{t}]_r, [\boldsymbol{s}]_r)$, else it outputs 0.
- $\mathsf{TrapdoorEquiv}(ck, tk, c, [\boldsymbol{v}]_r, Op, [\boldsymbol{v}']_r)$: On a commitment $c = ([\boldsymbol{y}]_r, [\widetilde{\boldsymbol{c}}]_r)$ with opening $Op = ([\boldsymbol{s}]_r)$, if $[\boldsymbol{t}]_r := [\boldsymbol{y} - \boldsymbol{v}]_r$ and $[\boldsymbol{t}']_r := [\boldsymbol{y} - \boldsymbol{v}']_r$, this algorithm computes $[\widetilde{\boldsymbol{c}}]_{r+1}$ and runs the algorithm $\widetilde{Op} \leftarrow \widetilde{\mathsf{TrapdoorEquiv}}(ck, tk, [\widetilde{\boldsymbol{c}}]_{r+1}, [\boldsymbol{t}]_r, [\boldsymbol{s}]_r, [\boldsymbol{t}']_r)$, and outputs $\widetilde{Op}$.

A commitment is a vector of size $k + d$ and an opening is of size k. The required security properties follow easily from the properties of the generalized Pedersen commitment.

Theorem 3. *C is a perfectly hiding, computationally binding commitment.*

Proof. Since the generalized Pedersen commitment is perfectly hiding, then $([\boldsymbol{t} + \boldsymbol{v}]_r, \widetilde{\mathsf{Comm}}(\widetilde{ck}, [\boldsymbol{t}]_{r-1}))$ perfectly hides $[\boldsymbol{v}]_r$ because $[\boldsymbol{t}]_r$ acts as a one-time pad. Similarly, it is straightforward to see that the computationally binding property of C follows from the computationally binding property of the generalized Pedersen commitment. □

Interestingly, this construction explains the two instantiations of "group-to-group" commitments given in [3]. Indeed, the generalization of the scheme described above to the asymmetric bilinear case (under the $\mathcal{U}_{1+d,1}$-KerMDH Assumption) matches the construction in [3], Sect. 4.1. When $\mathbf{A}$ is sampled from the distribution which results from sampling a matrix from the $\mathcal{RL}_2$ distribution and appending $d - 1$

additional random rows, it matches the construction in [3], Sect. 4.2. For illustration, we discuss this last example in some more detail.

In [3], Sect. 4.2., (see Fig.2), the commitment to a message $(M_1, \ldots, M_d) \in \mathbb{G}$ is a tuple $(C_1, \ldots, C_{d+2}) \in \mathbb{G}^{d+2}$ where the elements in each of the groups are written in multiplicative notation, $G, H, \{G_j, F_j\}_{j=0,\ldots,d}$ are random elements in $\mathbb{G}$ and is defined by the equations:

$$C_i = M_i H^{\tau_i} \qquad C_{d+1} = G_0^{\tau_0} \prod_{j=1}^{d} G_j^{\tau_j} \qquad C_{d+2} = F_0^{\mu_0} \prod_{j=1}^{d} F_j^{\tau_j}.$$

Let $\begin{pmatrix} G_0 & 1_{\mathbb{G}} \\ 1_{\mathbb{G}} & F_0 \\ G_1 & F_1 \\ \vdots & \vdots \\ G_d & F_d \end{pmatrix} \in \mathbb{G}^{(d+2)\times 2}$ and define $\mathbf{A}$ as the corresponding matrix of discrete logarithms in base H.
To see that this construction if of the appropriate form, it suffices to note that the pair (C_{d+1}, C_{d+2}) is the Pedersen commitment to $(\tau_1, \ldots, \tau_n)$ with randomness τ_0, μ_0 and commitment key $\mathbf{A}$.

7 A New Matrix Distribution and Its Applications

Both of our commitment schemes of Section 6 base security on some $\mathcal{D}_{k+d,k}$-KerMDH assumptions, where d is the length of the committed vector. When $d > 1$, the only example of $\mathcal{D}_{k+d,k}$-MDDH Assumption considered in [11] is the one corresponding to the uniform matrix distribution $\mathcal{U}_{k+d,k}$, which is the weakest MDDH Assumption of size $(k+d) \times k$. Another natural assumption for $d > 1$ is the one associated to the matrix distribution resulting from sampling from an arbitrary hard distribution $\mathcal{D}_{k+1,k}$ (*e.g.*, $\mathcal{L}_k$) and adding $d-1$ new random rows. Following the same ideas in the proof of Lemma 18, it is easy to see that the resulting $\mathcal{D}_{k+d,k}$-MDDH assumption is equivalent to the original $\mathcal{D}_{k+1,k}$-MDDH assumption. However, for efficiency reasons, we would like to have a matrix distributions with an even smaller representation size. This motivates us to introduce a new family of matrix distributions, the $\mathcal{CI}_{k,d}$ family.

Definition 20 (Circulant Matrix Distribution). *We define $\mathcal{CI}_{k,d}$ as*

$$\mathbf{A} = \begin{pmatrix} a_1 & & & 0 \\ \vdots & a_1 & & \\ a_d & \vdots & \ddots & \\ 1 & a_d & & a_1 \\ & 1 & \ddots & \vdots \\ & & \ddots & a_d \\ 0 & & & 1 \end{pmatrix} \in \mathbb{Z}_q^{(k+d)\times k}, \qquad \textit{where } a_i \leftarrow \mathbb{Z}_q$$

Matrix $\mathbf{A}$ is such that each column can be obtained by rotating one position the previous column, which explains the name. Notice that when $d = 1$, $\mathcal{CI}_{k,d}$ is exactly the symmetric cascade distribution $\mathcal{SC}_k$, introduced in [11]. It can be shown that the representation size of $\mathcal{CI}_{k,d}$, which is the number of parameters d, is the optimal among all hard matrix distributions $\mathcal{D}_{k+d,k}$ defined by linear polynomials in the parameters. A similar argument shows that the circulant assumption is also optimal in the sense that it has a minimal number of nonzero entries among all hard matrix distributions $\mathcal{D}_{k+d,k}$. It can also be proven that $\mathcal{CI}_{k,d}$-MDDH holds generically in k-linear groups, which implies the hardness of the corresponding KerMDH problem. To prove the generic hardness of the assumption, we turn to a result of Herold [20, Thm. 5.15 and corollaries]. It states that if all matrices produced by the matrix distribution are full-rank, $\mathcal{CI}_{k,d}$ is a hard matrix distribution. Indeed, an algorithm solving the $\mathcal{CI}_{k,d}$-MDDH problem in the k-linear generic group model must be able to compute a polynomial in the ideal $\mathfrak{H} \subset \mathbb{Z}_q[a_1, \ldots, a_d, z_1, \ldots, z_{k+d}]$ generated by all the $(k+1)$-minors of $\mathbf{A}\|\boldsymbol{z}$ as polynomials in $a_1, \ldots, a_d, z_1, \ldots, z_{k+d}$. Although

this ideal can actually be generated using only a few of the minors, we need to build a Gröbner basis of $\mathfrak{H}$ to reason about the minimum degree a nonzero polynomial in $\mathfrak{H}$ can have. We show that, carefully selecting a monomial order, the set of all $(k+1)$-minors of $\mathbf{A}\|\boldsymbol{z}$ form a Gröbner basis, and all these minors have total degree exactly $k+1$. Therefore, all nonzero polynomials in $\mathfrak{H}$ have degree at least $k+1$, and then they cannot be evaluated by any algorithm in the k-linear generic group model. The full proof of both properties of $\mathcal{CI}_{k,d}$ can be found in Appendix A.1.

As for other matrix distribution families, we can combine Corollary 1 and the techniques used in Lemma 19 to show that for any fixed $d \geq 1$ the $\mathcal{CI}_{k,d}$-KerMDH problem family has strictly increasing hardness.

Theorem 4. *For any $d \geq 1$ and for any $k, \widetilde{k}$ such that $\widetilde{k} < k$*

1. *there exists a tight reduction, $\mathcal{CI}_{\widetilde{k},d}$-$\mathsf{KerMDH} \Rightarrow \mathcal{CI}_{k,d}$-$\mathsf{KerMDH}$,*
2. *there is no black-box reduction in the multilinear generic group model in the opposite direction.*

The new assumption gives new instantiations of the commitment schemes of Section 6 with public parameters of size d, independent of k. Further, because the matrix $\mathbf{A} \leftarrow \mathcal{CI}_{k,d}$ has a many zero entries, the number of exponentiations computed by the Commit algorithm, and the number of pairings of the verification algorithm is kd, as opposed to $k(k+d)$ for the uniform assumption. This seems to be optimal, but we do not prove this formally.

8 Acknowledgements

The authors thank E. Kiltz and G. Herold for improving this work through very fruitful discussions. Also, G. Herold gave us the insight and guidelines to prove the hardness of the circulant matrix distribution. Part of this research have been done during a fruitful stage at HGI Bochum.

References

1. M. Abe, M. Chase, B. David, M. Kohlweiss, R. Nishimaki, and M. Ohkubo. Constant-size structure-preserving signatures: Generic constructions and simple assumptions. In X. Wang and K. Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 4–24, Beijing, China, Dec. 2–6, 2012. Springer, Heidelberg, Germany. 3
2. M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, and M. Ohkubo. Structure-preserving signatures and commitments to group elements. In T. Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 209–236, Santa Barbara, CA, USA, Aug. 15–19, 2010. Springer, Heidelberg, Germany. 2, 3, 4, 20, 28, 29
3. M. Abe, K. Haralambiev, and M. Ohkubo. Group to group commitments do not shrink. In D. Pointcheval and T. Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 301–317, Cambridge, UK, Apr. 15–19, 2012. Springer, Heidelberg, Germany. 3, 5, 29, 30
4. F. Bao, R. H. Deng, and H. Zhu. Variations of Diffie-Hellman problem. In S. Qing, D. Gollmann, and J. Zhou, editors, *ICICS 03*, volume 2836 of *LNCS*, pages 301–312, Huhehaote, China, Oct. 10–13, 2003. Springer, Heidelberg, Germany. 19
5. G. Barthe, E. Fagerholm, D. Fiore, J. C. Mitchell, A. Scedrov, and B. Schmidt. Automated analysis of cryptographic assumptions in generic group models. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 95–112, Santa Barbara, CA, USA, Aug. 17–21, 2014. Springer, Heidelberg, Germany. 6, 8, 9
6. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55, Santa Barbara, CA, USA, Aug. 15–19, 2004. Springer, Heidelberg, Germany. 19
7. D. Boneh, S. Halevi, M. Hamburg, and R. Ostrovsky. Circular-secure encryption from decision Diffie-Hellman. In D. Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 108–125, Santa Barbara, CA, USA, Aug. 17–21, 2008. Springer, Heidelberg, Germany. 19
8. X. Boyen. The uber-assumption family (invited talk). In S. D. Galbraith and K. G. Paterson, editors, *PAIRING 2008*, volume 5209 of *LNCS*, pages 39–56, Egham, UK, Sept. 1–3, 2008. Springer, Heidelberg, Germany. 1

9. S. Brands. Untraceable off-line cash in wallets with observers (extended abstract). In D. R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 302–318, Santa Barbara, CA, USA, Aug. 22–26, 1994. Springer, Heidelberg, Germany. 20
10. Y. Dodis, I. Haitner, and A. Tentes. On the instantiability of hash-and-sign RSA signatures. In R. Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 112–132, Taormina, Sicily, Italy, Mar. 19–21, 2012. Springer, Heidelberg, Germany. 4
11. A. Escala, G. Herold, E. Kiltz, C. Ràfols, and J. Villar. An algebraic framework for Diffie-Hellman assumptions. In R. Canetti and J. A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 129–147, Santa Barbara, CA, USA, Aug. 18–22, 2013. Springer, Heidelberg, Germany. 1, 4, 5, 8, 18, 19, 30, 34
12. A. Escala, G. Herold, E. Kiltz, C. Ràfols, and J. Villar. An Algebraic Framework for Diffie–Hellman Assumptions. *Journal of Cryptology*, pages 1–47, 2015. 6, 9
13. D. M. Freeman. Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In H. Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 44–61, French Riviera, May 30 – June 3, 2010. Springer, Heidelberg, Germany. 2
14. D. Galindo, J. Herranz, and J. L. Villar. Identity-based encryption with master key-dependent message security and leakage-resilience. In S. Foresti, M. Yung, and F. Martinelli, editors, *ESORICS 2012*, volume 7459 of *LNCS*, pages 627–642, Pisa, Italy, Sept. 10–12, 2012. Springer, Heidelberg, Germany. 2
15. R. Gay, D. Hofheinz, E. Kiltz, and H. Wee. Tightly CCA-secure encryption without pairings. In M. Fischlin and J.-S. Coron, editors, *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 1–27, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany. 3, 28
16. O. Goldreich. On post-modern cryptography. Cryptology ePrint Archive, Report 2006/461, 2006. `http://eprint.iacr.org/2006/461`. 1
17. A. González, A. Hevia, and C. Ràfols. QA-NIZK arguments in asymmetric groups: New tools and new constructions. In T. Iwata and J. H. Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 605–629, Auckland, New Zealand, Nov. 30 – Dec. 3, 2015. Springer, Heidelberg, Germany. 3, 4, 28
18. J. Groth. Homomorphic trapdoor commitments to group elements. Cryptology ePrint Archive, Report 2009/007, 2009. `http://eprint.iacr.org/2009/007`. 2, 20, 21, 29
19. J. Groth and S. Lu. A non-interactive shuffle with pairing based verifiability. In K. Kurosawa, editor, *ASIACRYPT 2007*, volume 4833 of *LNCS*, pages 51–67, Kuching, Malaysia, Dec. 2–6, 2007. Springer, Heidelberg, Germany. 2, 20
20. G. Herold. Applications of classical algebraic geometry to cryptography. *PhD Thesis, Ruhr-Universität Bochum*, 2014. 5, 6, 8, 9, 30, 34
21. G. Herold, J. Hesse, D. Hofheinz, C. Ràfols, and A. Rupp. Polynomial spaces: A new framework for composite-to-prime-order transformations. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 261–279, Santa Barbara, CA, USA, Aug. 17–21, 2014. Springer, Heidelberg, Germany. 2
22. D. Hofheinz and E. Kiltz. Secure hybrid encryption from weakened key encapsulation. In A. Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 553–571, Santa Barbara, CA, USA, Aug. 19–23, 2007. Springer, Heidelberg, Germany. 19
23. A. Joux and A. Rojat. Security ranking among assumptions within the uber assumption framework. Cryptology ePrint Archive, Report 2013/291, 2013. `http://eprint.iacr.org/2013/291`. 19
24. C. S. Jutla and A. Roy. Switching lemma for bilinear tests and constant-size NIZK proofs for linear subspaces. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 295–312, Santa Barbara, CA, USA, Aug. 17–21, 2014. Springer, Heidelberg, Germany. 3, 4, 19
25. E. Kiltz, J. Pan, and H. Wee. Structure-preserving signatures from standard assumptions, revisited. In R. Gennaro and M. J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 275–295, Santa Barbara, CA, USA, Aug. 16–20, 2015. Springer, Heidelberg, Germany. 3, 4, 28
26. E. Kiltz and H. Wee. Quasi-adaptive NIZK for linear subspaces revisited. In E. Oswald and M. Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 101–128, Sofia, Bulgaria, Apr. 26–30, 2015. Springer, Heidelberg, Germany. 3, 4, 28
27. F. Laguillaumie, P. Paillier, and D. Vergnaud. Universally convertible directed signatures. In B. K. Roy, editor, *ASIACRYPT 2005*, volume 3788 of *LNCS*, pages 682–701, Chennai, India, Dec. 4–8, 2005. Springer, Heidelberg, Germany. 2, 20
28. T. Lepoint. Zeroizing attacks on multilinear maps. ECRYPT-CSA Workshop on Tools for Asymmetric Cryptanalysis, 2015. Slides available at `http://cryptool.hgi.rub.de/program.html`. 20
29. A. B. Lewko and B. Waters. Efficient pseudorandom functions from the decisional linear assumption and weaker variants. In E. Al-Shaer, S. Jha, and A. D. Keromytis, editors, *ACM CCS 09*, pages 112–120, Chicago, Illinois, USA, Nov. 9–13, 2009. ACM Press. 2

30. B. Libert, T. Peters, M. Joye, and M. Yung. Linearly homomorphic structure-preserving signatures and their applications. In R. Canetti and J. A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 289–307, Santa Barbara, CA, USA, Aug. 18–22, 2013. Springer, Heidelberg, Germany. 3
31. B. Libert, T. Peters, M. Joye, and M. Yung. Non-malleability from malleability: Simulation-sound quasi-adaptive NIZK proofs and CCA2-secure encryption from homomorphic signatures. In P. Q. Nguyen and E. Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 514–532, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany. 3
32. B. Libert and D. Vergnaud. Multi-use unidirectional proxy re-signatures. In P. Ning, P. F. Syverson, and S. Jha, editors, *ACM CCS 08*, pages 511–520, Alexandria, Virginia, USA, Oct. 27–31, 2008. ACM Press. 2, 20
33. U. M. Maurer. Towards the equivalence of breaking the Diffie-Hellman protocol and computing discrete algorithms. In Y. Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 271–281, Santa Barbara, CA, USA, Aug. 21–25, 1994. Springer, Heidelberg, Germany. 6
34. U. M. Maurer. Abstract models of computation in cryptography (invited paper). In N. P. Smart, editor, *10th IMA International Conference on Cryptography and Coding*, volume 3796 of *LNCS*, pages 1–12, Cirencester, UK, Dec. 19–21, 2005. Springer, Heidelberg, Germany. 6
35. M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *38th FOCS*, pages 458–467, Miami Beach, Florida, Oct. 19–22, 1997. IEEE Computer Society Press. 2
36. M. Naor and G. Segev. Public-key cryptosystems resilient to key leakage. In S. Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 18–35, Santa Barbara, CA, USA, Aug. 16–20, 2009. Springer, Heidelberg, Germany. 19
37. T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 129–140, Santa Barbara, CA, USA, Aug. 11–15, 1992. Springer, Heidelberg, Germany. 4
38. J. H. Seo. On the (im)possibility of projecting property in prime-order setting. In X. Wang and K. Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 61–79, Beijing, China, Dec. 2–6, 2012. Springer, Heidelberg, Germany. 2
39. H. Shacham. A cramer-shoup encryption scheme from the linear assumption and from progressively weaker linear variants. Cryptology ePrint Archive, Report 2007/074, 2007. `http://eprint.iacr.org/2007/074`. 19
40. J. L. Villar. Optimal reductions of some decisional problems to the rank problem. In X. Wang and K. Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 80–97, Beijing, China, Dec. 2–6, 2012. Springer, Heidelberg, Germany. 2
41. B. R. Waters. Efficient identity-based encryption without random oracles. In R. Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 114–127, Aarhus, Denmark, May 22–26, 2005. Springer, Heidelberg, Germany. 1

A More Details About the Circulant Matrix Distribution

In this appendix we give more details about both the hardness and the optimality of the representation size of the circulant matrix distribution.

A.1 Optimality of the Representation Size

Lemma 20. *A matrix distribution $\mathcal{D}_{\ell,k}$ defined by linear polynomials in the parameters, $\mathbf{A}(t_1,\ldots,t_d) = \mathbf{A}_0 + \mathbf{A}_1 t_1 + \ldots + \mathbf{A}_d t_d$, where $\mathbf{A}_1,\ldots,\mathbf{A}_d$ are linearly independent matrices, can only be hard if the number of parameters d is at least $\ell - k$.*

Proof. Assume for contradiction that $d < \ell - k$. Then, by gaussian elimination in $\mathbf{A}$ we can transform the matrix into another one, $\mathbf{B}(t_1,\ldots,t_d) = \mathbf{L}\mathbf{A}(t_1,\ldots,t_d)$ for a constant invertible matrix $\mathbf{L}$, such that the first column of $\mathbf{B}$ has zeroes in the lower $\ell - d - 1 \geq k$ positions (as at most $d+1$ entries can be linearly independent as polynomials in $t_1,\ldots,t_d$). Now, it is straightforward to see that this can be used to solve the associated $\mathcal{D}_{\ell,k}$-MDDH problem. Indeed, if $\widehat{\mathbf{L}}$ denotes the lowest k rows of $\mathbf{L}$, given an instance $([\mathbf{A}],[\boldsymbol{z}])$, we transform it into $([\widehat{\mathbf{L}}\mathbf{A}],[\widehat{\mathbf{L}}\boldsymbol{z}])$ and then compare the ranks of $\widehat{\mathbf{L}}\mathbf{A}$ and $\widehat{\mathbf{L}}\mathbf{A}\|\widehat{\mathbf{L}}\boldsymbol{z}$ (computed by means of the k-linear map). The ranks are equal for 'real' instances, while they are different with overwhelming probability for 'random' instances of the $\mathcal{D}_{\ell,k}$-MDDH problem, which contradicts the hardness of $\mathcal{D}_{\ell,k}$. □

The linear independency requirement in the lemma just means that there is no redundancy among the d parameters (that is, the map $(t_1, \ldots, t_d) \mapsto \mathbf{A}(t_1, \ldots, t_d)$ is injective). The representation of $\mathbf{A}$ must contain at least d group elements, due to the previous injectivity. Therefore, $\mathcal{CI}_{k,d}$ has optimal representation size.

A.2 Hardness

Here we prove that $\mathcal{CI}_{k,d}$ is a hard matrix distribution (*i.e.*, the $\mathcal{CI}_{k,d}$-MDDH problem is generically hard in k-linear groups), using Theorem 5.15 and specially its Corollary 5.16 in [20] in the linear polynomial case, and Gröbner basis computations in some polynomial ideal.

Intuitively, an algorithm solving $\mathcal{CI}_{k,d}$-MDDH problem in the k-linear generic group model must know some nonzero polynomial in $t_1, \ldots, t_d, z_1, \ldots, z_{k+d}$ vanishing whenever $\boldsymbol{z} \in \operatorname{Im} \mathbf{A}(\boldsymbol{t})$. But this can only happen if such polynomial belongs to the ideal $\mathfrak{H} \in \overline{\mathbb{Z}}_q[t_1, \ldots, t_d, z_1, \ldots, z_{k+d}]$[16] generated by the relations between $t_1, \ldots, t_d, z_1, \ldots, z_{k+d}$ obtained by elimination of the variables $w_1, \ldots, w_k$ in the equation $\boldsymbol{z} = \mathbf{A}(\boldsymbol{t})\boldsymbol{w}$.

$\mathcal{CI}_{k,d}$ has some interesting properties that makes possible the generic hardness proof. Namely, it is defined by linear polynomials (*i.e.*, $\mathbf{A}(\boldsymbol{t})$ is made of polynomials of degree one in the parameters $t_1, \ldots, t_d$), and $\operatorname{rank} \mathbf{A}(\boldsymbol{t}) = k$ for all possible choices of $t_1, \ldots, t_d \in \overline{\mathbb{Z}}_q$ (*i.e.*, in the algebraic closure of $\mathbb{Z}_q$). This second property comes from the fact that the lowest k-minor of $\mathbf{A}(\boldsymbol{t})$ is constant and equal to 1. With these two properties, Theorem 5.15 and its Corollary 5.16 in [20] essentially state that $\mathfrak{H}$ is precisely the ideal generated by all the $(k+1)$-minors of $\mathbf{A}(\boldsymbol{t})\|\boldsymbol{z}$ as polynomials in $t_1, \ldots, t_d, z_1, \ldots, z_{k+d}$. More precisely,

Theorem 5 (from Theorem 5.15 and its Corollary 5.16 in [20]). *Let $\mathcal{D}_{\ell,k} = \{\mathbf{A}(\boldsymbol{t}) \mid \boldsymbol{t} \leftarrow \mathbb{Z}_q^d\}$ be a polynomial matrix distribution of degree one such that the matrices $\mathbf{A}(\boldsymbol{t})$ in the distribution have always full rank, for all choices of the parameters $t_1, \ldots, t_d$ in the algebraic closure of $\mathbb{Z}_q$. Let*

$$\mathfrak{H} = I(\{(\boldsymbol{t}, \mathbf{A}(\boldsymbol{t})\boldsymbol{w}) \mid \boldsymbol{t} \in \mathbb{Z}_q^d, \boldsymbol{w} \in \mathbb{Z}_q^k\})$$

that is the ideal of the polynomials in $\overline{\mathbb{Z}}_q[\boldsymbol{t}, \boldsymbol{z}]$ vanishing at all (rational) points such that $\boldsymbol{z} = \mathbf{A}(\boldsymbol{t})\boldsymbol{w}$, for some $\boldsymbol{w} \in \mathbb{Z}_q^k$, and

$$\mathfrak{D} = (\{\det{}_{i_1,\ldots,i_{k+1}}(\mathbf{A}(\boldsymbol{t})\|\boldsymbol{z}) \mid 1 \le i_1 < i_2 < \cdots < i_{k+1} \le \ell\})$$

the ideal generated by all $(k+1)$-minors of $\mathbf{A}(\boldsymbol{t})\|\boldsymbol{z}$. Then $\mathfrak{H} = \mathfrak{D}$ and it is a prime ideal.

At this point, proving the generic hardness of $\mathcal{CI}_{k,d}$ amounts to proving that there is no nonzero polynomial of total degree less that $k+1$ in $\mathfrak{H}$. Indeed, this means that the only way to generically solve the $\mathcal{CI}_{k,d}$-MDDH problem is computing a polynomial of degree strictly greater than k, which is not feasible in k-linear groups. Notice that in the general case $d \ge 1$, finding a lower bound for the total degree in $\mathfrak{H}$ is a nontrivial task, while in the case $\ell = k+1$ or $d = 1$, as seen in [11], it is as easy as computing the degree of the determinant polynomial $\det(\mathbf{A}(\boldsymbol{t})\|\boldsymbol{z})$. The main reason for that difficulty is the fact that the ideal $\mathfrak{H}$ is not principal. Therefore, we need to compute a Gröbner basis of $\mathfrak{H}$, and show that all the polynomials in it have total degree at least $k+1$.

Gröbner bases can be computed quite easily for specific ideals by means of a computer, but here we will build bases for an infinite collection of ideals, that is for arbitrary values of the size parameters k and d. Thus, we have to compute them by hand. Fortunately, we manage to show that the set of all $(k+1)$-minors of $\mathbf{A}(\boldsymbol{t})\|\boldsymbol{z}$ as polynomials in $t_1, \ldots, t_d, z_1, \ldots, z_{k+d}$, where $\mathbf{A} \leftarrow \mathcal{CI}_{k,d}$, is a Gröbner basis of $\mathfrak{H}$.

We recall some basic notions related to ideals and Gröbner basis. An admissible monomial order $\prec$ in the polynomial ring $\mathbb{Z}_q[t_1, \ldots, t_d, z_1, \ldots, z_{k+d}]$ is a total order among the monomials in it such that for any monomials $\mathfrak{m}_1, \mathfrak{m}_2, \mathfrak{m}_3$

[16] $\overline{\mathbb{Z}}_q$ denotes the algebraic closure of the field $\mathbb{Z}_q$. We define the ideal in the algebraic closure for technical reasons, although the polynomial used by the algorithm will necessarily have its coefficients in $\mathbb{Z}_q$.

1. $\mathfrak{m}_1 \neq 1 \;\Rightarrow\; 1 \prec \mathfrak{m}_1$
2. $\mathfrak{m}_1 \prec \mathfrak{m}_2 \;\Rightarrow\; \mathfrak{m}_1\mathfrak{m}_3 \prec \mathfrak{m}_2\mathfrak{m}_3$

The leading monomial of a polynomial $\mathfrak{p}$, denoted by $\mathrm{LM}(\mathfrak{p})$ is defined as the greatest of its monomials (without the coefficient)[17] with respect of the monomial order. We recall that a Gröbner basis of $\mathfrak{H} \subset \mathbb{Z}_q[t_1, \ldots, t_d, z_1, \ldots, z_{k+d}]$ with respect of an admissible monomial order is a set of nonzero polynomials $G = \{\mathfrak{g}_1, \ldots, \mathfrak{g}_s\} \subset \mathfrak{H}$ with the following properties

1. for every $\mathfrak{f} \in \mathfrak{H}$ there exist $\mathfrak{c}_1, \ldots, \mathfrak{c}_s \in \mathbb{Z}_q[t_1, \ldots, t_d, z_1, \ldots, z_{k+d}]$ such that $\mathfrak{f} = \mathfrak{c}_1\mathfrak{g}_1 + \ldots + \mathfrak{c}_s\mathfrak{g}_s$,
2. for every $\mathfrak{f} \in \mathfrak{H}$, $\mathrm{LM}(\mathfrak{f})$ is divisible by $\mathrm{LM}(\mathfrak{g})$ for some $\mathfrak{g} \in G$.

The following lemma comes in a straightforward way from the previous definition

Lemma 21. *The minimal degree of nonzero polynomials in an ideal $\mathfrak{H}$ is the minimal degree of the polynomials in any Gröbner basis of $\mathfrak{H}$ with respect to any admissible monomial order compatible with the total degree.*

From now on we fix the following admissible monomial order:

1. $\deg \mathfrak{m}_1 < \deg \mathfrak{m}_2 \;\Rightarrow\; \mathfrak{m}_1 \prec \mathfrak{m}_2$, where deg denotes the total degree of the monomial,
2. $z_1 \prec \ldots \prec z_{k+d} \prec t_1 \prec \ldots \prec t_d$,
3. if $\deg \mathfrak{m}_1 = \deg \mathfrak{m}_2$, $\prec$ is the lexicographical order. That is, we write $\mathfrak{m}_1 = z_1^{\alpha_1} \cdots z_{k+d}^{\alpha_{k+d}} t_1^{\alpha_{k+d+1}} \cdots t_d^{\alpha_{k+2d}}$ and $\mathfrak{m}_2 = z_1^{\beta_1} \cdots z_{k+d}^{\beta_{k+d}} t_1^{\beta_{k+d+1}} \cdots t_d^{\beta_{k+2d}}$. Then, for the same total degree, $\mathfrak{m}_1 \prec \mathfrak{m}_2$ if and only if the first nonzero difference $\beta_i - \alpha_i$ is positive.

Given $\mathbf{A} \leftarrow \mathcal{CI}_{k,d}$ we denote by $\Delta(\boldsymbol{i}) = \Delta(i_1, \ldots, i_{k+1}) = \det_{i_1,\ldots,i_{k+1}}(\mathbf{A}(\boldsymbol{t})\|\boldsymbol{z})$ the $(k+1)$-minor of $\mathbf{A}(\boldsymbol{t})\|\boldsymbol{z}$ defined by the rows $1 \leq i_1 < i_2 < \cdots < i_{k+1} \leq k+d$. From now on we will use $\boldsymbol{i} \in \binom{k+d}{k+1}$ as a shorthand for the previous inequalities. These determinants have very special properties. Indeed, we show that with respect to the previous monomial order the main diagonal defines their leading monomial.

Lemma 22. *For all $\boldsymbol{i} \in \binom{k+d}{k+1}$, $LM(\Delta(\boldsymbol{i})) = z_{i_{k+1}} t_{i_1} t_{i_2-1} \cdots t_{i_k-k+1}$.*

Proof. In order to simplify the notation, we can always write the $(k+1)$-minors of $\mathbf{A}(\boldsymbol{t})\|\boldsymbol{z}$ as

$$\Delta(\boldsymbol{i}) = \begin{vmatrix} t_{i_1} & t_{i_1-1} & \cdots & t_{i_1-k+1} & z_{i_1} \\ t_{i_2} & t_{i_2-1} & \cdots & t_{i_2-k+1} & z_{i_2} \\ \vdots & \vdots & \vdots\vdots\vdots & \vdots & \vdots \\ t_{i_k} & t_{i_k-1} & \cdots & t_{i_k-k+1} & z_{i_k} \\ t_{i_{k+1}} & t_{i_{k+1}-1} & \cdots & t_{i_{k+1}-k+1} & z_{i_{k+1}} \end{vmatrix}$$

assuming that $t_{d+1} = 1$, and $t_i = 0$ for any i outside the range $i = 1, \ldots, d+1$. Then we show that in the development of the determinant, the monomial corresponding to the main diagonal $\mathfrak{m}_{\mathrm{diag}} = z_{i_{k+1}} t_{i_1} t_{i_2-1} \cdots t_{i_k-k+1}$ is the leading monomial. Firstly, notice that all the terms occurring in the main diagonal are proper terms (*i.e.*, neither 0 nor 1). Indeed, $1 \leq i_1 \leq i_2 - 1 \leq \cdots \leq i_k - k + 1 \leq i_{k+1} - k \leq d$, which is something that deeply depends on the circulant structure of the matrix $\mathbf{A}$.

Now, assume by contradiction that there is another term in the development of the determinant, $\mathfrak{m} = z_{i_{\sigma(k+1)}} t_{i_{\sigma(1)}} t_{i_{\sigma(2)}-1} \cdots t_{i_{\sigma(k)}-k+1}$ (written in a possibly unsorted way), for a permutation $\sigma \in S_{k+1}$, such that $\mathfrak{m}_{\mathrm{diag}} \prec \mathfrak{m}$. Due to the monomial order, the last column must contribute to this term with the variable z_{k+1} (that is $\sigma(k+1) = k+1$). Otherwise, $z_{i_{\sigma(k+1)}} \prec z_{i_{k+1}}$, which contradicts $\mathfrak{m}_{\mathrm{diag}} \prec \mathfrak{m}$.

We can rewrite $\mathfrak{m}$ in terms of the inverse permutation $\pi = \sigma^{-1}$ as

$$\mathfrak{m} = z_{i_{k+1}} t_{i_1-\pi(1)+1} t_{i_2-\pi(2)+1} \cdots t_{i_k-\pi(k)+1}$$

[17] We call *leading term* to the leading monomial multiplied by the corresponding coefficient.

Then, $\pi(1) \neq 1$ would imply $a_{i_1-\pi(1)+1} \prec a_{i_1}$, which is also in contradiction with $\mathfrak{m}_{\text{diag}} \prec \mathfrak{m}$. Therefore, $\sigma(1) = 1$. Observe that it could happen that $i_1 - \pi(1) + 1 \leq 0$, which means that $t_{i_1-\pi(1)+1}$ is actually 0, which also contradicts $\mathfrak{m}_{\text{diag}} \prec \mathfrak{m}$.

Proceeding similarly with subsequent indexes (rows) in increasing order, we easily show that σ can only be the identity permutation, which concludes the proof. □

Now we prove that the set of all $(k+1)$-minors of $\mathbf{A}\|\boldsymbol{z}$ is a Gröbner basis. The proof of this result is rather technical and deeply relies on the properties of determinants.

Theorem 6. *The set $G = \{\Delta(\boldsymbol{i}) \mid \boldsymbol{i} \in \binom{k+d}{k+1}\}$ is a Gröbner basis of $\mathfrak{H}$ with respect to the monomial order $\prec$.*

Proof. The usual way to prove that a set $G = \{\mathfrak{g}_1, \ldots, \mathfrak{g}_s\}$ is a Gröbner basis is by means of the so-called S-polynomials. The S-polynomial of a pair $\mathfrak{g}_i, \mathfrak{g}_j \in G$ for $i \neq j$ is defined by

$$\mathfrak{s}_{i,j} = \text{SPOL}(\mathfrak{g}_i, \mathfrak{g}_j) = \frac{\mathfrak{m}_{i,j}}{\mathfrak{m}_i}\mathfrak{g}_i - \frac{\mathfrak{m}_{i,j}}{\mathfrak{m}_j}\mathfrak{g}_j \tag{2}$$

where $\mathfrak{m}_i = \text{LM}(\mathfrak{g}_i)$, $\mathfrak{m}_j = \text{LM}(\mathfrak{g}_j)$, and $\mathfrak{m}_{i,j}$ denotes the least common multiple of $\mathfrak{m}_i$ and $\mathfrak{m}_j$. Then, G is a Gröbner basis if and only if for all $i \neq j$, $\text{RED}_G(\mathfrak{s}_{i,j}) = 0$, where $\text{RED}_G(\mathfrak{p})$ denotes the reduction of a polynomial $\mathfrak{p}$ by repeatedly taking the remainder of the division by elements in G until no further division can be properly performed[18]. Indeed, the famous Buchberger's algorithm for computing Gröbner basis iteratively uses the previous computation to either verify that a pair $\mathfrak{g}_i, \mathfrak{g}_j \in G$ passes the check, or to add its reduced S-polynomial to G.

Observe that any expression of the form $\mathfrak{p} = \mathfrak{c}_1\mathfrak{g}_1 + \ldots + \mathfrak{c}_s\mathfrak{g}_s$, where all the leading monomials $\mathfrak{n}_j = \text{LM}(\mathfrak{c}_j\mathfrak{g}_j)$, $j = 1, \ldots, s$, are different, shows that $\text{RED}_G(\mathfrak{p}) = 0$. Indeed, without loss off generality we can sort the previous terms to ensure that $\mathfrak{n}_1 \prec \ldots \prec \mathfrak{n}_s$. Clearly, $\text{LM}(\mathfrak{p}) = \mathfrak{n}_s$ and reducing $\mathfrak{p}$ by $\mathfrak{g}_s$ gives the remainder $\mathfrak{c}_1\mathfrak{g}_1 + \ldots + \mathfrak{c}_{s-1}\mathfrak{g}_{s-1}$. Therefore, by induction, we get $\text{RED}_G(\mathfrak{p}) = 0$.

Buchberger also provided two optimization rules to speed up the algorithm. In particular, the second one (Buchberger's chain criterion) says that if for some indexes i, j, v, $\text{RED}_G(\mathfrak{s}_{i,v}) = \text{RED}_G(\mathfrak{s}_{j,v}) = 0$ and $\mathfrak{m}_v$ divides $\mathfrak{m}_{i,j}$, then also $\text{RED}_G(\mathfrak{s}_{i,j}) = 0$. We use this criterion to inductively show that in our case, we only need to deal with pairs of $(k+1)$-minors differing only in one row.

We now split the proof into several technical claims. The idea is drawing a path between any two $(k+1)$-minors in which at each step we only change one row of the minor.

The first claim says that from any two $(k+1)$-minors with the same upper $\alpha - 1$ rows but that differ in the α-th row, we can build a third "hybrid" minor by moving the α-th row from one determinant to the other, and the corresponding three leading monomials are related in a suitable way for the chain criterion described above.

Claim 1. For any two sequences $\boldsymbol{i}, \boldsymbol{i^*} \in \binom{k+d}{k+1}$ such that the first difference occurs at position α, for $1 \leq \alpha \leq k$, that is $i_j = i^*_j$ if $j < \alpha$ and $i_\alpha < i^*_\alpha$,

$$\text{LM}(\Delta(i_1, \ldots, i_\alpha, i^*_{\alpha+1} \ldots, i^*_{k+1})) \quad \text{divides} \quad \text{lcm}(\text{LM}(\Delta(\boldsymbol{i})), \text{LM}(\Delta(\boldsymbol{i^*})))$$

Proof (of Claim 1). According to Lemma 22,

$$\text{LM}(\Delta(i_1, \ldots, i_\alpha, i^*_{\alpha+1} \ldots, i^*_{k+1})) = \begin{cases} z_{i^*_{k+1}} t_{i_1} \cdots t_{i_\alpha-\alpha+1} t_{i^*_{\alpha+1}-\alpha} \cdots t_{i^*_k-k+1} & \text{if } \alpha < k \\ z_{i^*_{k+1}} t_{i_1} \cdots t_{i_k-k+1} & \text{if } \alpha = k \end{cases}$$

Then the claim directly comes from the fact that this monomial can be split into two coprime factors $t_{i_1} \cdots t_{i_\alpha-\alpha+1}$ and $z_{i^*_{k+1}} t_{i^*_{\alpha+1}-\alpha} \cdots t_{i^*_k-k+1}$, each dividing $\text{LM}(\Delta(\boldsymbol{i}))$ and $\text{LM}(\Delta(\boldsymbol{i^*}))$, respectively. Indeed, both factors are coprime because $i_1 \leq \cdots \leq i_\alpha - \alpha + 1 < i^*_\alpha - \alpha + 1 \leq i^*_{\alpha+1} - \alpha \leq \cdots \leq i^*_{k+1} - k$. □

[18] The reduction algorithm repeatedly takes a polynomial $\mathfrak{p}$ and checks whether some $\text{LM}(\mathfrak{g}_i)$ divides $\text{LM}(\mathfrak{p})$. If so, the algorithm cancels out the leading term of $\mathfrak{p}$ by subtracting from it the appropriate multiple of $\mathfrak{g}_i$, and repeats the procedure with the resulting polynomial. Otherwise, the algorithm adds the leading term of $\mathfrak{p}$ to the output polynomial and proceeds with the remaining terms of $\mathfrak{p}$, until they are exhausted. The output has the property that none of its monomials is divisible by any $\text{LM}(\mathfrak{g}_i)$.

We call *adjacent pair* to any pair of minors $\Delta(\boldsymbol{i})$ and $\Delta(\boldsymbol{i^*})$ such that $\boldsymbol{i}$ and $\boldsymbol{i^*}$ differ only at position α, for some $1 \leq \alpha \leq k+1$, that is $i_j = i^*_j$ if $j \neq \alpha$ and $i_\alpha < i^*_\alpha$. This pair can be actually described by an increasing sequence of length $k+2$, $1 \leq i_1 < \cdots < i_\alpha < i^*_\alpha < \cdots < i_{k+1} \leq k+d \in \binom{k+d}{k+2}$ and the index α. The previous claim allows us to build a path connecting any two $(k+1)$-minors such that every consecutive pairs of minors in the path is an adjacent pair. An example for $k=5$ is depicted below. The numbers in every column in the table correspond to the indices of the rows in every minor in the path connecting $\Delta(1,5,6,9,12,13)$ and $\Delta(2,4,7,8,11,14)$.

i_1	1	1	1	1	1	1	2
i_2	5	4	4	4	4	4	4
i_3	6	6	6	6	6	7	7
i_4	9	9	8	8	8	8	8
i_5	12	12	12	11	11	11	11
i_6	13	13	13	13	14	14	14

Using the above path we show that, according to Buchberger's chain criterion, to prove that G is a Gröbner basis it suffices to check only the S-polynomials of adjacent pairs.

Claim 2. If the S-polynomial of every adjacent pair of $(k+1)$-minors is reducible to 0, then so are all the other S-polynomials (and therefore G is a Gröbner basis of $\mathfrak{H}$).

Proof (of Claim 2). We prove the claim by (descending) induction in α, the index of the first row-difference between two $(k+1)$-minors. For $\alpha = k+1$ (*i.e.*, the minors only differ in the last row) the statement is obviously true, as the two minors form an adjacent pair. Now, let us assume that the statement is true for $\alpha = \alpha_0$, where $1 < \alpha_0 \leq k+1$. Then for any two minors with the first row-difference occurring at row $\alpha_0 - 1$, say $\mathfrak{g} = \Delta(i_1, \ldots, i_{\alpha_0-2}, i_{\alpha_0-1}, \ldots, i_{k+1})$ and $\mathfrak{g}^* = \Delta(i_1, \ldots, i_{\alpha_0-2}, i^*_{\alpha_0-1}, \ldots, i^*_{k+1})$ with $i_{\alpha_0-1} < i^*_{\alpha_0-1}$, we define as in the first claim the "hybrid" minor $\mathfrak{h} = \Delta(i_1, \ldots, i_{\alpha_0-1}, i^*_{\alpha_0}, \ldots, i^*_{k+1})$. Then, by Claim 1 we know that $\mathrm{LM}(\mathfrak{h})$ divides $\mathrm{lcm}(\mathrm{LM}(\mathfrak{g}), \mathrm{LM}(\mathfrak{g}^*))$. On the other hand, the induction assumption implies $\mathrm{RED}_G(\mathrm{SPOL}(\mathfrak{h}, \mathfrak{g})) = 0$, since the first row-difference between $\mathfrak{g}$ and $\mathfrak{h}$ occurs at row α_0. Moreover, $\mathrm{RED}_G(\mathrm{SPOL}(\mathfrak{h}, \mathfrak{g}^*)) = 0$ because $(\mathfrak{h}, \mathfrak{g}^*)$ is an adjacent pair. Thus, by Buchberger's chain criterion, $\mathrm{RED}_G(\mathrm{SPOL}(\mathfrak{g}, \mathfrak{g}^*)) = 0$, concluding the proof of the second claim. □

The last step in the proof of the theorem is showing that the S-polynomial of every adjacent pair of $(k+1)$-minors is reducible to 0. Actually, the S-polynomial of an adjacent pair of minors can be embedded into the determinant of a $(k+2) \times (k+2)$ matrix. This matrix gives us a syzygy (*i.e.*, a linear relation with polynomial coefficients among elements in G) that allows to manually reduce the S-polynomial to 0.

Claim 3. For any adjacent pair of $(k+1)$-minors given by the sequence $\boldsymbol{i} \in \binom{k+d}{k+2}$ and the index $1 \leq \alpha \leq k+1$, that is $\mathfrak{g} = \Delta(i_1, \ldots, i_\alpha, i_{\alpha+2} \ldots, i_{k+2})$ and $\mathfrak{g}^* = \Delta(i_1, \ldots, i_{\alpha-1}, i_{\alpha+1} \ldots, i_{k+2})$, $\mathrm{RED}_G(\mathrm{SPOL}(\mathfrak{g}, \mathfrak{g}^*)) = 0$.

Proof (of Claim 3). Let us consider for the case $\alpha \leq k$ the extended matrix

$$\mathbf{B} = \begin{pmatrix} t_{i_1} & t_{i_1-1} & \cdots & t_{i_1-k+1} & z_{i_1} & t_{i_1-\alpha+1} \\ \vdots & \vdots & \vdots\vdots & \vdots & \vdots & \vdots \\ t_{i_\alpha} & t_{i_\alpha-1} & \cdots & t_{i_\alpha-k+1} & z_{i_\alpha} & t_{i_\alpha-\alpha+1} \\ t_{i_{\alpha+1}} & t_{i_{\alpha+1}-1} & \cdots & t_{i_{\alpha+1}-k+1} & z_{i_{\alpha+1}} & t_{i_{\alpha-1}-\alpha+1} \\ \vdots & \vdots & \vdots\vdots & \vdots & \vdots & \vdots \\ t_{i_{k+2}} & t_{i_{k+2}-1} & \cdots & t_{i_{k+2}-k+1} & z_{i_{k+2}} & t_{i_{k+2}-\alpha+1} \end{pmatrix}$$

which is a $(k+2)\times(k+2)$ matrix[19] with two repeated columns: the α-th and the last columns. Thus, $\det \mathbf{B} = 0$, and using Laplace expansion of the determinant along the last column we obtain the following syzygy:

$$\sum_{\substack{j=1\\ j\neq\alpha,\alpha+1}}^{k+2} (-1)^{k+j} t_{i_j-\alpha+1}\mathfrak{h}_j + (-1)^{k+\alpha}\left(t_{i_\alpha-\alpha+1}\mathfrak{g}^* - t_{i_{\alpha+1}-\alpha+1}\mathfrak{g}\right) = 0 \tag{3}$$

where $\mathfrak{h}_j = \Delta(i_1,\ldots,i_{j-1},i_{j+1},\ldots,i_{k+2})$. Actually, $\mathfrak{g} = \mathfrak{h}_\alpha$ and $\mathfrak{g}^* = \mathfrak{h}_{\alpha+1}$.

Notice that the S-polynomial $\mathfrak{s} = \mathrm{SPOL}(\mathfrak{g},\mathfrak{g}^*)$, as given by Equation 2, is exactly $t_{i_{\alpha+1}-\alpha+1}\mathfrak{g} - t_{i_\alpha-\alpha+1}\mathfrak{g}^*$, since

$$\mathrm{LM}(\mathfrak{g}) = z_{i_{k+2}} t_{i_1}\cdots t_{i_\alpha-\alpha+1} t_{i_{\alpha+2}-\alpha}\cdots t_{i_{k+1}-k+1}$$

and

$$\mathrm{LM}(\mathfrak{g}^*) = z_{i_{k+2}} t_{i_1}\cdots t_{i_{\alpha-1}-\alpha+2} t_{i_{\alpha+1}-\alpha+1}\cdots t_{i_{k+1}-k+1}$$

As a consequence, Equation 3 gives an explicit reduction of $\mathfrak{s}$ to 0. Namely,

$$\mathfrak{s} = \sum_{\substack{j=1\\ j\neq\alpha,\alpha+1}}^{k+2} (-1)^{\alpha+j} t_{i_j-\alpha+1}\mathfrak{h}_j \tag{4}$$

Actually, to see that Equation 4 implies $\mathrm{RED}_G(\mathfrak{s}) = 0$, we only need to show that all leading monomials $\mathfrak{n}_{\alpha,j} = \mathrm{LM}(t_{i_j-\alpha+1}\mathfrak{h}_j) = t_{i_j-\alpha+1}\mathrm{LM}(\mathfrak{h}_j)$, for $j = 1,\ldots,k+2$, $j\neq\alpha,\alpha+1$, corresponding to nonzero terms[20] are different. We now compute all leading monomials $\mathfrak{n}_{\alpha,j}$ for any $j\neq\alpha,\alpha+1$ (written as possibly unsorted products):

$$\mathfrak{n}_{\alpha,j} = \begin{cases} z_{i_{k+2}} t_{i_1-\alpha+1} t_{i_2}\cdots t_{i_{k+1}-k+1} & \text{for } j=1\\ z_{i_{k+2}} t_{i_1}\cdots t_{i_{j-1}-j+2} t_{i_j-\alpha+1} t_{i_{j+1}-j+1}\cdots t_{i_{k+1}-k+1} & \text{for } 1<j<k+1\\ z_{i_{k+2}} t_{i_1}\cdots t_{i_k-k+1} t_{i_{k+1}-\alpha+1} & \text{for } j=k+1\\ z_{i_{k+1}} t_{i_1}\cdots t_{i_k-k+1} t_{i_{k+2}-\alpha+1} & \text{for } j=k+2 \end{cases}$$

Clearly, $\mathfrak{n}_{\alpha,k+2}$ is different to the others, and the only coincidence can be $\mathfrak{n}_{\alpha,j} = \mathfrak{n}_{\alpha,j^*}$ for some $1\leq j < j^*\leq k+1$. But this would imply (removing all common terms)

$$t_{i_j-\alpha+1} t_{i_{j+1}-j+1}\cdots t_{i_{j^*-1}-j^*+3} t_{i_j^*-j^*+2} = t_{i_j-j+1} t_{i_{j+1}-j}\cdots t_{i_{j^*-1}-j^*+2} t_{i_j^*-\alpha+1}$$

But due to the inequalities of the indices, the least index in the righthand side, $i_j - j + 1$, can only be canceled out with the term $i_j - \alpha + 1$ in the left hand side, thus implying $j = \alpha$. Similarly, $i_j^* - j^* + 2$ on the left must be the same as $i_j^* - \alpha + 1$ on the right, and then $j^* = \alpha + 1$. But these values of j, j^* are out of the correct range, what shows that no collision among the $\mathfrak{n}_{\alpha,j}$ is actually possible.

For the remaining case, $\alpha = k+1$, we proceed similarly, defining

$$\mathbf{B} = \begin{pmatrix} t_{i_1} & t_{i_1-1} & \cdots & t_{i_1-k+1} & z_{i_1} & z_{i_1}\\ \vdots & \vdots & \vdots\vdots\vdots & \vdots & \vdots & \vdots\\ t_{i_{k+1}} & t_{i_{k+1}-1} & \cdots & t_{i_{k+1}-k+1} & z_{i_{k+1}} & z_{i_{k+1}}\\ t_{i_{k+2}} & t_{i_{k+2}-1} & \cdots & t_{i_{k+2}-k+1} & z_{i_{k+2}} & z_{i_{k+2}} \end{pmatrix}$$

Now, the syzygy is

$$\sum_{j=1}^{k} (-1)^{k+j} z_{i_j}\mathfrak{h}_j - z_{i_{k+1}}\mathfrak{g}^* + z_{i_{k+2}}\mathfrak{g} = 0$$

[19] Recall that we are using the notational convention introduced in Lemma 22. Thus, some entries in the matrix can be equal to 0 or 1.

[20] Some terms in Equation 4 can be zero due to $t_{i_j-\alpha+1} = 0$, what happens exactly when $i_j - \alpha + 1 \leq 0$ or $i_j - \alpha + 1 \geq d + 2$

where $\mathfrak{h}_j = \Delta(i_1, \ldots, i_{j-1}, i_{j+1}, \ldots, i_{k+2})$. Then

$$\mathfrak{s} = \mathrm{SPOL}(\mathfrak{g}, \mathfrak{g}^*) = z_{i_{k+2}}\mathfrak{g} - z_{i_{k+1}}\mathfrak{g}^* = -\sum_{j=1}^{k}(-1)^{k+j} z_{i_j}\mathfrak{h}_j \tag{5}$$

and for any $j < k+1$

$$\mathfrak{n}_{k+1,j} = \mathrm{LM}(z_{i_j}\mathfrak{h}_j) = z_{i_j}\mathrm{LM}(\mathfrak{h}_j) = \begin{cases} z_{i_1} z_{i_{k+2}} t_{i_2} t_{i_3 - 1} \cdots t_{i_{k+1} - k + 1} & \text{for } j = 1 \\ z_{i_j} z_{i_{k+2}} t_{i_1} \cdots t_{i_{j-1} - j + 2} t_{i_{j+1} - j + 1} \cdots t_{i_{k+1} - k + 1} & \text{for } 1 < j < k+1 \end{cases}$$

which are clearly different. This shows that again Equation 5 is a reduction to 0 of $\mathrm{SPOL}(\mathfrak{g}, \mathfrak{g}^*)$, which covers all the remaining adjacent pairs of $(k+1)$-minors. □

Now, the theorem statement is a direct consequence of Claims 2 and 3. □

The next corollary finally proves that $\mathcal{CI}_{k,d}$ is a hard matrix distribution, that is, the $\mathcal{CI}_{k,d}$-MDDH problem is generically hard in k-linear groups.

Corollary 2. *All nonzero polynomials in $\mathfrak{H}$ have degree at least $k+1$.*

Proof. According to Lemma 22 the degree of all polynomials $\Delta(\boldsymbol{i})$ for $\boldsymbol{i} \in \binom{k+d}{k+1}$ is exactly $k+1$. Thus, by Lemma 21 and Theorem 6 the statement follows directly. □