Equivalence of Polynomial Identity Testing and Deterministic Multivariate Polynomial Factorization

Swastik Kopparty[*] Shubhangi Saraf[†] Amir Shpilka[‡]

Abstract

In this paper we show that the problem of deterministically factoring multivariate polynomials reduces to the problem of deterministic polynomial identity testing. Specifically, we show that given an arithmetic circuit (either explicitly or via black-box access) that computes a multivariate polynomial f, the task of computing arithmetic circuits for the factors of f can be solved deterministically, given a deterministic algorithm for the polynomial identity testing problem (we require either a white-box or a black-box algorithm, depending on the representation of f).

Together with the easy observation that deterministic factoring implies a deterministic algorithm for polynomial identity testing, this establishes an equivalence between these two central derandomization problems of arithmetic complexity.

Previously, such an equivalence was known only for multilinear circuits [SV10].

[*] Department of Mathematics & Department of Computer Science, Rutgers University. swastik.kopparty@gmail.com. Research supported in part by a Sloan Fellowship and NSF CCF-1253886.

[†] Department of Mathematics & Department of Computer Science, Rutgers University. shubhangi.saraf@gmail.com. Research supported by NSF grant CCF-1350572.

[‡] Faculty of Computer Science, Technion – Israel Institute of Technology, Haifa, Israel, shpilka@cs.technion.ac.il. The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 257575.

1 Introduction

In this paper we study the relation between two fundamental algebraic problems, polynomial identity testing (PIT for short) and polynomial factorization. We show that the tasks of giving deterministic algorithms for polynomial identity testing and for polynomial factorization are, essentially, equivalent. We first give some background on both problems and then discuss our results in detail.

Polynomial Factorization. The problem of polynomial factorization for multivariate polynomials asks the following: Given a polynomial $f(X_1, \ldots, X_n)$, compute each of the irreducible factors of f. From the arithmetic complexity point of view, it is most natural to have f be presented as an arithmetic circuit, and ask that the algorithm returns irreducible factors of f in the form of arithmetic circuits (this is called the white-box model). Alternatively, we could assume that we have black-box access to f, and ask that the factorization algorithm outputs a black-box for each of the irreducible factors. One can also consider a (simpler) decision version of this question: given an arithmetic circuit computing a multivariate polynomial, decide whether the polynomial is irreducible or not. In the decision version the algorithm just has to answer 'yes' or 'no' and it is not required to find the factorization.

A surprising and fundamental result in arithmetic complexity states that factoring of multivariate polynomials can be done *efficiently* in both the black-box and white-box settings [Kal89, KT90]. This implies the amazing fact that if f has a circuit of size s and degree d in n variables, then the irreducible factors of f have arithmetic circuits of size $\text{poly}(s, d, n)$. Both these factorization algorithms are randomized, and just as in the case of polynomial identity testing (that we discuss below), it is a long standing open question whether there is an efficient *deterministic* algorithm for factoring multivariate polynomials (see [GG99, Kay07]). Moreover, there is no known deterministic algorithm even for (1) the decision problem of irreducibility testing, and (2) the problem of factoring *multilinear* polynomials.

An important tool in designing randomized factorization algorithms is Hilbert's irreducibility theorem, which in one formulation says that restricting an irreducible polynomial to a random two-dimensional subspace keeps the polynomial irreducible with high probability. In other words, restricting a polynomial f to a random two-dimensional subspace does not change the number of irreducible factors of f. Given this, multivariate factoring algorithms proceed as follows. First restrict the polynomial to a randomly chosen two-dimensional space. Then, perform bi-variate factorization of the restricted polynomial. Finally, "lift" each factor to the whole space. The study of factorization of multivariate polynomials has led to significant advances in our understanding of the quantitative aspects of Hilbert's irreducibility theorem [Kal95].

In [SV10], Shpilka and Volkovich proved that factoring multilinear polynomials reduces to polynomial identity testing for multilinear polynomials. Their algorithm relies heavily on the fact that factors of multilinear polynomials must be supported on disjoint sets of variables, and in particular it does not follow the usual methodology outlined above for factoring polynomials. This result has two interesting aspects. First, it gives a close connection between these two basic problems. Second, it shows that deterministic factorization algorithms may be hard to find, as they are equivalent to the PIT problem for multilinear circuits, which, if found, would yield an explicit multilinear polynomial with exponential multilinear circuit complexity. In their work Shpilka and Volkovich left open the question of whether the same fundamental relation holds for the general case (i.e.

for general (non-multilinear) polynomials). In this paper we resolve this problem affirmatively by giving a reduction from multivariate polynomial factorization to PIT for general circuits. This highlights the significance of the polynomial identity testing problem as the problem closest to being "complete" for the algebraic version of the class RP.

For more on polynomial factorization we refer the reader to the surveys [Kal90, Kal92, Kal03, Gat06] as well as to the lecture notes of Sudan [Sud99]. For more on algebra in computation we refer to the excellent book [GG99].

Polynomial Identity Testing. Let $\mathcal{C}$ be a class of arithmetic circuits defined over some field $\mathbb{F}$. The polynomial identity testing problem (PIT for short) for $\mathcal{C}$ is the question of deciding whether a given circuit from $\mathcal{C}$ computes the identically zero polynomial. This question can be considered both in the black-box model, in which we can only access the polynomial computed by the circuit using queries, or in the white-box model where the circuit is given to us. The importance of this fundamental problem stems from its many applications. For example, the deterministic primality testing algorithm of [AKS04] and the fast parallel algorithm for perfect matching of [MVV87] are based on solving PIT problems.

In this paper, we consider PIT for the class of poly(n)-size, poly(n)-degree arithmetic circuits in n variables. PIT has a well known randomized algorithm [Sch80, Zip79, DL78], but no sub-exponential time deterministic algorithm is known in the general case. This question received a lot of attention recently and several deterministic black-box algorithms were devised for restricted classes of arithmetic circuits, but the solution for the general model remains elusive. The works of [HS80, KI04, Agr05, DSY09] proved that derandomizing PIT, either in the white-box setting or in the black-box setting, implies lower bounds for arithmetic circuits. The work of Kabanets and Impagliazzo (and [DSY09] for small depth circuits) also proved the reverse direction, namely, that using a hard problem one can devise a *hitting set* for arithmetic circuits. I.e., given a hard function one can use it to construct a black-box algorithm for PIT. It is interesting to note that the PIT problem becomes very difficult already for depth 3 circuits. Indeed, [GKKS13] proved that a polynomial time black-box PIT algorithm for depth 3 circuits (of unbounded degree) implies an exponential lower bound for general arithmetic circuits (and hence using the ideas of [KI04] a quasi-polynomial time PIT for general circuits). For more on PIT see the survey [SY10].

In this work we show that the two derandomization problems are (essentially) equivalent. Namely, we show that a polynomial time deterministic PIT algorithm exists if and only if there is a deterministic polynomial time factorization algorithm. The result holds both in the black-box and the white-box models. That is, if the PIT algorithm is in the black-box setting then deterministic black-box factorization is possible and vice versa, and similarly for the white-box case. The black-box case essentially follows by carefully inspecting the proofs of the original randomized factoring algorithms. It is the whitebox case that is the core of the technical contribution of this work.

1.1 Our Results

Before stating our main result we first define the model of arithmetic circuits.

An *arithmetic circuit* in the variables $\boldsymbol{X} = (X_1, \ldots, X_n)$, over the field $\mathbb{F}$, is a labelled directed acyclic graph. The inputs (nodes of in-degree zero) are labelled by variables from $\boldsymbol{X}$ or by constants from the field. The internal nodes are labelled by + or ×, computing the sum and product, respectively, of the polynomials on the tails of incoming edges (subtraction is obtained using the constant -1). Outputs are nodes of out-degree zero.

The *size* of a circuit (or formula) is the number of gates in it. The *depth* of the circuit is the length of a longest path between an output node and an input node.

When we say degree of a multivariate polynomial, we mean its *total degree.* An arithmetic circuit C has degree bounded by d if all the gates computed by the circuit have degree bounded by d. Finally, we shall say that C is an (n, s, d)-circuit if it is an n-variate arithmetic circuit of size s with degree bounded by d.

Our main result states that if PIT can be solved deterministically in polynomial time, then given as input a size-s n-variate arithmetic circuit over $\mathbb{F}_{p^\ell}$ or $\mathbb{Q}$ computing a degree d polynomial f, one can deterministically in time $\mathrm{poly}(n, s, d, p^\ell)$ (in the case of $\mathbb{F}_{p^\ell}$) or $\mathrm{poly}(n, s, d, t)$ (where t is the bit-complexity of the constants in the circuit, in the case of $\mathbb{Q}$), compute the factors of f. We now formally state our main result.

Theorem 1 (Main) *Let $\mathbb{F}$ be either the finite field $\mathbb{F}_{p^\ell}$ (with characteristic p) or the rationals $\mathbb{Q}$.*

Suppose white-box (black-box) polynomial identity testing for size s, degree d, n-variable arithmetic circuits over $\mathbb{F}$ can be solved deterministically in time $\mathrm{poly}(n, s, d)$.

Suppose we are given white-box (black-box respectively) access to a polynomial $f(X_1, \ldots, X_n) \in \mathbb{F}[X_1, \ldots, X_n]$ computed by an arithmetic circuit of size at most s and degree at most d. Let

$$f = \prod_{i=1}^{k} g_i^{p^{e_i} \cdot j_i}$$

be the factorization of f, where the g_i are irreducible and $p \nmid j_i$ for each i.

Then we can compute, for each $i \in [k]$: (i) e_i, (ii) j_i, and (iii) an arithmetic circuit (black-box respectively) for the factor $g_i^{p^{e_i}}$ (the factor g_i respectively) in deterministic time $\mathrm{poly}(n, s, d, t)$, *where:*

1. *$t = \ell \cdot p$, if $\mathbb{F} = \mathbb{F}_{p^\ell}$ is a field of characteristic p.*
2. *$t =$ maximum bit-complexity of the constants used in the circuit, if $\mathbb{F} = \mathbb{Q}$.*

The main new technical content of this theorem is for the white-box case. In the process, we also give a new (and possibly simpler) proof of Kaltofen's result [Kal89] showing that factors of polynomials computed by small circuits have small circuits. This new proof has the advantage of being constructive in a certain precise sense, and this plays an important role in our main result.

The fact that we only compute arithmetic circuits for $g_i^{p^{e_i}}$ in the white-box setting can be interpreted as follows: we produce an "augmented" arithmetic circuit for each g_i, which is in the form of an arithmetic circuit, followed by a unary gate which computes the p^{e_i}th root. Even the randomized white-box factorization algorithm of Kaltofen [Kal89] only achieves this kind of factorization.

Over finite fields, note that dependence on the field in the running time of the above multivariate factoring algorithm is polynomial in $p \cdot \ell$, while in principle it could be simply polynomial in $\ell \cdot \log p$.

This dependence on the characteristic is a well known fundamental problem: today it is not even known how to deterministically factor univariate polynomials of degree d in time $\mathrm{poly}(d, \ell, \log p)$ (this is unknown even for $d = 2$!). This is the only bottleneck for our multivariate factoring algorithm: if one could factor univariate polynomials of degree d over $\mathbb{F}_{p^\ell}$ deterministically in time $\mathrm{poly}(d, \ell, \log p)$, then the running time in our main theorem can be improved to depend polynomially on $\ell \cdot \log p$ (instead of depending polynomially on $\ell \cdot p$).

We note that the other direction in the equivalence of PIT and factorization was observed in [SV10], namely, that deterministic factorization (even for the decision problem of irreducibility testing) implies deterministic PIT.

Observation 1 (Observation 1 in [SV10]) *Let $\mathcal{C}$ be a class of arithmetic circuits. Assume that there is a deterministic algorithm for the decision version of the polynomial factorization problem. That is, an algorithm that when given access (explicit or via a black-box) to an (n, s, d) circuit $C \in \mathcal{C}$ runs in time $T(s, d)$ and outputs "true" iff the polynomial computed by C is irreducible. Then, there is a deterministic algorithm that runs in time $O(T(s + 2, d))$ and solves the PIT problem for size s circuits from $\mathcal{C}$.*

1.2 Proof technique

Our algorithms are closely related to the randomized black-box factorization algorithms of [Kal89, KT90], and so we start by first giving a high level view of those algorithms. In the explanation below we adopt the terminology of lecture 9 in [Sud99]. The initialization step in all factoring algorithms is to massage the polynomial f to be factored to the case where it is squarefree, monic in some variable X and satisfies $\frac{\partial f}{\partial X} \neq 0$. The next step of the randomized algorithms is to restrict f to a randomly chosen two-dimensional subspace composed of all points $\{(X, \alpha_1 T + \beta_1, \ldots, \alpha_n T + \beta_n)\}$ (where $\alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_n$ are chosen at random). The idea is that for such a random subspace, all irreducible factors of f remain irreducible with high probability. This is an "effective" version of Hilbert's irreducibility theorem that was proved by Kaltofen in [Kal95].

Theorem 1.1 (Effective Hilbert irreducibility) *Let $S \subseteq \mathbb{F}$ be a finite set and $g(X, A_1, \ldots, A_n)$ a monic polynomial in X of total degree at most d. If g is irreducible then it holds that*

$$\Pr_{\boldsymbol{\alpha},\boldsymbol{\beta}}[g(X, \alpha_1 T + \beta_1, \ldots, \alpha_n T + \beta_n) \text{ is not irreducible }] < O(d^5/|S|),$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are chosen uniformly and independently from S^n.

Note that the lemma guarantees that if we pick the set S to be large enough, then with high probability all irreducible factors of f remain irreducible when restricted to the chosen subspace. The randomized algorithm then proceeds by factoring f over the subspace $\{(X, \alpha_1 T + \beta_1, \ldots, \alpha_n T + \beta_n)\}$. At this stage we have a restriction of each factor to the chosen two-dimensional space. The final step of the factorization algorithms is to use these restrictions, either via the Hensel lifting lemma [Kal89] (in the white-box case) or via trivariate factorizations [KT90] (in the black-box case), to find a global factor over the entire space.

We now explain how we derandomize these algorithms using PIT in the black-box case and in the white-box case. Perhaps surprisingly, the proof is simpler in the black box case. This can be explained by the fact that our assumption is also stronger - in this case we assume that we have a black-box PIT algorithm, which is a stronger assumption than a white-box PIT algorithm.

1.2.1 The black-box case

We would like to simulate the above randomized algorithm in the black-box case. Using blackbox PIT, one can easily do the preprocessing to make f monic and squarefree. All that remains is to find a good two-dimensional subspace to restrict to. The main observation here is that Theorem 1.1 can be strengthened in the following way.

Theorem 1.2 (Effective and efficient Hilbert irreducibility) *Let $g(X, A_1, \ldots, A_n)$ be a monic polynomial in X of total degree at most d, that is computed by an arithmetic circuit of size s. If g is irreducible then there is a circuit of size $\mathrm{poly}(s, d)$ in $2n$ variables computing a polynomial $h(Z_1, \ldots, Z_n, Y_1, \ldots, Y_n)$ of degree $O(d^5)$ so that if $h(\boldsymbol{\alpha}, \boldsymbol{\beta}) \neq 0$ then $g(X, \alpha_1 T + \beta_1, \ldots, \alpha_n T + \beta_n)$ is irreducible. Furthermore, this circuit can be computed in a black-box manner from the circuit for g.*

We do not prove this theorem here, but it is implicit in the proof of Theorem 1.1 in [Kal95] (for an excellent exposition, see Theorem 1 of Lecture 9 of [Sud99]).

We would like to use our black-box PIT algorithm on the polynomial h given in Theorem 1.2 to find a two-dimensional subspace that will preserve irreducibility for all irreducible factors of f (i.e. we want to find a nonzero simultaneously for all h's corresponding to factors of f). For this we need to know that h has a small circuit, and this in turns requires us to know that each irreducible factor g of f indeed has a small circuit (so that we can apply Theorem 1.2 with reasonable parameters). Luckily, Kaltofen [Kal89] proved that if f can be computed by an arithmetic circuit of size s then each of its factors can be computed by arithmetic circuits of size $\mathrm{poly}(s, d, n)$. We conclude that there exists a circuit of degree $O(d^6)$ and size $\mathrm{poly}(s, d, n)$ in $2n$ variables such that any non-zero assignment to it gives a "good" two-dimensional space. We can thus use the assumed black-box PIT for such circuits to claim that the black-box PIT algorithm finds such a good pair $(\boldsymbol{\alpha}, \boldsymbol{\beta})$. Notice that we do not need to know the circuits for the factors nor the circuit of Theorem 1.2, but rather it is enough to have a bound on its complexity to be able to use the black-box PIT algorithm to find a good subspace. This is the main difference from the white-box model (we shall elaborate on this point more later).

Having found a good two-dimensional subspace using the PIT algorithm, we can proceed with the proof as in the black-box randomized case [KT90], noting that all remaining steps can be performed in the black-box model deterministically. Putting all these steps together, the deterministic black-box factoring algorithm follows in a straightforward way[1]. The rest of the paper is devoted to the proof of the whitebox case.

[1] This was also independently observed by Dvir and Mendes de Oliveira (personal communication).

1.2.2 The white-box case

In the white-box case things are trickier. We first explain why we cannot adopt the same strategy as in the black-box case. The natural thing would be to try and compute the circuit guaranteed by Theorem 1.2 for each of the irreducible factors of f, and then use the white-box PIT algorithm to find a good subspace. However, the theorem only tells us that to compute the polynomial h (associated to an irreducible factor g of f), we need to start from a circuit computing g. But getting a circuit for g is the original problem we are trying to solve! Rephrasing, this strategy says that in order to factor f, we should first compute small circuits for each of its irreducible factors, which can then help us find a subspace that will help us compute circuits for its irreducible factors. This circularity prevents the above approach from being feasible in the white-box case.

A central piece of the deterministic black-box factorization algorithm (using PIT) is Kaltofen's theorem on the existence of small arithmetic circuits for the factors. Perhaps the proof of that theorem will suggest a way to construct the small circuits for the factors deterministically (using white-box PIT). Unfortunately, the main step in the proof of that theorem is *Hilbert's irreducibility theorem*, Theorem 1.1! One first restricts to a random 2-dimensional subspace, which with high probability preserves the factorization pattern of the original polynomial f. The proof merely uses the existence of such a 2-dimensional subspace, and the proof does not require nor show either how to construct such a subspace, or even how to construct a circuit detecting such a subspace.

The above explains why the white-box case requires a new constructive ingredient over the black-box case. The main technical contribution of this work is to show that instead of finding a good two-dimensional subspace, one can work with a "formal" two-dimensional subspace. Concretely, we work over the field of fractions $\mathbb{F}(A_1, \ldots, A_n)$ (in the formal variables $A_1, \ldots, A_n$; we may think of $(A_1, \ldots, A_n)$ as defining the "direction" of the formal subpace). That is, we define a new polynomial $\bar{f}(X, T, A_1, \ldots, A_n) \overset{\text{def}}{=} f(X, TA_1, \ldots, TA_n)$ and view $\bar{f}$ as a polynomial in $\mathbb{K}[X, T]$ over the field of rational functions $\mathbb{K} = \mathbb{F}(A_1, \ldots, A_n)$. It is not hard to prove that if f is monic in X then irreducible factors of f map to irreducible factors of $\bar{f}$ and vice versa. Thus, in some sense we have found a way to reduce the problem to the two-dimensional case. The main issue now is that the field is much more complicated and instead of working with constants from $\mathbb{F}$ we have to work with constants from $\mathbb{K}$.

The standard way to factorize a general bivariate polynomial over a field $\mathbb{K}$ would (1) perform a univariate polynomial factorization over the field $\mathbb{K}$ (which is nearly as hard as the original problem since $\mathbb{K}$ is so complicated), (2) perform Hensel lifting, (3) solve a linear system, (4) compute a GCD. However the polynomial $\bar{f}$ was chosen to have a special form, which allows the univariate polynomial factorization over $\mathbb{K}$ (which is what we need) to reduce to a univariate factorization over the small field $\mathbb{F}$! We then show that the remaining steps of the bivariate factorization can be done despite the complexity of the field $\mathbb{K}$. This is where the white-box PIT comes into play; it enables us to perform basic tasks over the field $\mathbb{K}$, such as linear algebra and working with polynomials in $\mathbb{K}[X, T]$, deterministically. Here we need to verify that all "constants" from $\mathbb{K}$ appearing in intermediate computations can be computed by small circuits (in the variables $A_1, \ldots, A_n$). In other words, while the field $\mathbb{K} = \mathbb{F}(A_1, \ldots, A_m)$ is quite complex, the elements from $\mathbb{K}$ that we will be using can all be computed by small circuits, and we will be able to compute and work with those circuits efficiently using white-box PIT.

1.3 Notation

We shall use capital letters X, T, A_i to denote variables. Lower case Greek letters α, β, c will be used to denote assignments to the variables or, more generally, constants from the field. Bold face letters $\boldsymbol{\alpha}, \boldsymbol{\beta}$ will denote vectors. Polynomials will be denoted by lower case letters a, b, f, g, h, and vectors of polynomials by bold face lower case $\boldsymbol{v}$.

1.4 Organization

In the next section we recall some algebraic tools and algebraic algorithms that will be useful for us. We then give our factorization algorithm. We conclude with some open questions.

2 Algorithmic and Algebraic tool kit

In this section we set up our notation and give some known facts about circuits and known facts from algebra.

2.1 Known facts about arithmetic circuits

The following well known lemma states that given access to a circuit computing a polynomial f we can construct (in a black-box manner) a circuit for each of the homogeneous components of f. For a proof see e.g. [SY10]. We denote the homogeneous components of f by $H^0(f), \ldots, H^d(f)$, where $H^i(f)$ is the sum of all monomials in f of degree exactly i.

Lemma 2.1 *Given an arithmetic circuit $C(X, Y_1, \ldots, Y_n)$, of size s, that computes a polynomial $f(X, Y_1, \ldots, Y_n) \in \mathbb{F}[X, Y_1, \ldots, Y_n]$ of degree d, we can construct arithmetic circuits for the homogeneous components of f in time $\mathrm{poly}(n, d, s)$. Furthermore, the circuit for each $H^i(f)$ is of size $O(ds)$. Similarly, if we let $f(X, Y_1, \ldots, Y_n) = \sum_{i=0}^{d} X^i f_i(Y_1, \ldots, Y_n)$ then we can in time $\mathrm{poly}(n, d, s)$ compute arithmetic circuits for the polynomials f_i. Furthermore, the circuit for each f_i is of size $O(ds)$.*

Another useful tool is that given a white-box PIT algorithm we can use it to find a non-zero assignment for a given non-zero circuit.

Lemma 2.2 (Decision to search reduction for white-box PIT) *Given an arithmetic circuit C computing a non-zero n-variate polynomial $f(Y_1, \ldots, Y_n)$ of degree d, and a white-box PIT algorithm that runs in time polynomial in the size of C, n and d, we can find a point $\mathbf{a} \in \mathbb{F}^n$ such that $f(\mathbf{a}) \neq 0$ in time $\mathrm{poly}(|C|, n, d)$.*

Proof Let $S = \{\alpha_0, \ldots, \alpha_d\}$ be a set of $d+1$ distinct values from $\mathbb{F}$. Notice that we can check, using the PIT algorithm, whether the restriction $Y_1 = \alpha_i \in S$ makes f vanish. Since the degree of f is d and $f \not\equiv 0$, there exists a value of $\alpha_i \in S$ such that $f(\alpha_i, Y_2, \ldots, Y_n) \not\equiv 0$. Hence, by a linear scan over S we can find such an index $0 \leq i \leq d$ such that $f(\alpha_i, Y_2, \ldots, Y_n) \not\equiv 0$. Fix $Y_1 = \alpha_i$ and repeat this procedure with the other variables $\{Y_2, \ldots, Y_n\}$. The running time is clearly bounded by nd times the running time of the PIT algorithm. ■

2.2 Algebraic tool kit

Lemma 2.3 (Gauss' Lemma) *Let $g(X, Y_1, \ldots, Y_n) \in \mathbb{F}(Y_1, \ldots, Y_n)[X]$ be a monic (in X) factor of $f(X, Y_1, \ldots, Y_n) \in \mathbb{F}[X, Y_1, \ldots, Y_n]$. Then $g(X, Y_1, \ldots, Y_n) \in \mathbb{F}[X, Y_1, \ldots, Y_n]$.*

For the proof see, e.g., Chapter 6.2 in [GG99].

The Resultant Let $f = \sum_{i=0}^{d} X^i a_i(Y_1, \ldots, Y_n)$ and $g = \sum_{i=0}^{e} X^i b_i(Y_1, \ldots, Y_n)$ be polynomials of X-degree exactly d and e respectively. Consider the $(d+e) \times (d+e)$ Sylvester matrix whose first e rows contain e shifts of the vector of coefficients $(a_d, \ldots, a_0, 0, \ldots, 0)$. Namely, the kth row begins with $k-1$ zeros and ends with $e-k$ zeros. E.g., the (e)th row is $(0, \ldots, 0, a_d, \ldots, a_0)$. The next d rows contain shifts of the vector of coefficients $(b_e, \ldots, b_0, 0, 0, \ldots, 0)$. Thus, the $(e+1)$st row in the matrix equals $(b_e, \ldots, b_0, 0, \ldots, 0)$ and the $(d+e)$th row contains the vector $(0, \ldots, 0, b_e, \ldots, b_0)$. The resultant of the polynomials $f(X, Y_1, \ldots, Y_n)$ and $g(X, Y_1, \ldots, Y_n)$ with respect to the variable X is defined to be the determinant of the Sylvester matrix defined above.

If we know that each a_i is a polynomial of degree at most d and each b_j is a polynomial of degree at most e, a crude upper bound on the degree of the resultant as a polynomial in the Y_is is $2de$.

We have the following basic properties of the resultant (see [Sud99]).

Lemma 2.4 (Resultant facts) *Let $d, e \geq 1$. Let $f(X, Y_1, \ldots, Y_n)$ have X-degree exactly d and let $g(X, Y_1, \ldots, Y_n)$ have X-degree exactly e. Let $u(Y_1, \ldots, Y_n)$ be their resultant with respect to the variable X. Then:*

1. *$u = 0$ if and only if $f(X, Y_1, \ldots, Y_n)$ and $g(X, Y_1, \ldots, Y_n)$ have a nontrivial GCD in the ring $\mathbb{F}(Y_1, \ldots, Y_n)[X]$.*
2. *there exist polynomials $v(X, Y_1, \ldots, Y_n)$ and $w(X, Y_1, \ldots, Y_n)$ such that:*

$$f \cdot v + g \cdot w = u.$$

The Discriminant $D_f(Y_1, \ldots, Y_n)$. Let $f = \sum_{i=0}^{d} X^i a_i(Y_1, \ldots, Y_n)$ be a degree d polynomial. Let $\frac{\partial f}{\partial X} = \sum_{i=0}^{d} i \cdot X^{i-1} a_i(Y_1, \ldots, Y_n)$ be its derivative with respect to X.

The discriminant of a polynomial $f(X, Y_1, \ldots, Y_n)$ with respect to the variable X, denoted $D_f(Y_1, \ldots, Y_n)$, is defined to be the resultant of the polynomials f and $\frac{\partial f}{\partial X}$. Since each a_i is a polynomial of degree at most d, a crude upper bound on the degree of $D_f(Y_1, \ldots, Y_n)$, as a polynomial in the Y_is is $2d^2$.

Lemma 2.5 (Small circuit for the Discriminant) *Let $f(X, Y_1, \ldots, Y_n)$ be a degree d polynomial computed by an arithmetic circuit of size s. Given this arithmetic circuit, we can find in deterministic time* $\mathrm{poly}(s, n, d)$ *an arithmetic circuit of size* $\mathrm{poly}(s, n, d)$ *that computes the discriminant $D_f(Y_1, \ldots, Y_n)$.*

Proof The proof follows from the following two simple facts: a small arithmetic circuit computing the coefficients of the X^i in f can be found using the arithmetic circuit for f, and, the Determinant has small arithmetic circuits. ■

The main property of the discriminant that we need is that if $f(X, Y_1, \ldots, Y_n)$ is squarefree, then $D_f(Y_1, \ldots, Y_n)$ is nonzero. Furthermore, if $D_f(\alpha_1, \ldots, \alpha_n) \neq 0$, then $f(X, \alpha_1, \ldots, \alpha_n)$ is a squarefree univariate polynomial.

2.3 Linear Algebra using PIT

In this section we explain how to perform linear algebra when coefficients of vectors are given as circuits.

Lemma 2.6 (Solving linear systems) *Let $M = (M_{i,j})$ be a $k \times n$ matrix, with each entry being a degree $\leq \Delta$ polynomial in $\mathbb{F}[A_1, \ldots, A_n]$. Suppose that we have an arithmetic circuit C of size at most s computing M. Then, given access to a PIT oracle, we can either:*

1. *find an arithmetic circuit of size at most $\mathrm{poly}(n, k, s, \Delta)$ computing a nonzero vector $\boldsymbol{v} \in (\mathbb{F}[A_1, \ldots, A_n])^n$ such that $M\boldsymbol{v} = 0$, or*
2. *declare that there are no nonzero vectors $\boldsymbol{v} \in (\mathbb{F}[A_1, \ldots, A_n])^n$ such that $M\boldsymbol{v} = 0$,*

deterministically in time $\mathrm{poly}(k, n, s, \Delta)$.

Proof The idea of the proof is the following. Iteratively, for every $j = 1, \ldots, n$ we shall find an $j \times j$ minor contained in the first j columns that is full rank. We will continue doing so until we either reach $j = n$ in which case it means that the matrix has full column rank and hence the only solution is $\boldsymbol{v} = \mathbf{0}$, or we get stuck at some value $j = j_0$. Using the fact that we cannot increase j further we will use this minor to construct the required vector $\boldsymbol{v}$.

We now explain the process. Using PIT we look for some non-zero entry in the first column. If no such entry is found we can simply take $\boldsymbol{v} = (1, 0, \ldots, 0)$. So assume that such a non-zero entry is found. After permuting the rows we can assume wlog that this is $M_{1,1}$. Thus, we have found a 1×1 minor satisfying the requirements. Assume that we have found an $j \times j$ full rank minor that, wlog, is composed of the first j rows and columns. Denote this minor with M_j. Now for every $(j+1) \times (j+1)$ submatrix of M contained in the first $j+1$ columns and containing M_j, we use our PIT oracle to check if its determinant is nonzero. If any of these submatrices have nonzero determinant, then we pick one of them and call it M_{j+1}. Otherwise, we have that the first $j+1$ columns of M are linearly dependent. In this case we can use Kramer's formula to find the unique (up to multiplication by elements of the field $\mathbb{F}(A_1, \ldots, A_n)$) vector $\boldsymbol{u} = (u_1, \ldots, u_r) \in \mathbb{F}(A_1, \ldots, A_n)^j$ such that $M_j \cdot \boldsymbol{u} = (M_{1,j+1}, \ldots, M_{j,j+1})$. Notice that each entry of $\boldsymbol{u}$ is of the form $\frac{\det(M_j^{(i)})}{\det(M_j)}$, where $M_j^{(i)}$ is the matrix obtained from replacing the ith column of M_j with the vector $(M_{1,j+1}, \ldots, M_{j,j+1})$.

Thus the vector $(\det(M_j^{(1)}), \ldots, \det(M_j^{(j)}), -\det(M_j), 0, 0, \ldots, 0)$ is the desired vector $\boldsymbol{v}$. Observe that $\boldsymbol{v}$ can be computed by a circuit of size $s + \mathrm{poly}(n, k, \Delta)$.

■

2.4 Computing division with remainder and GCD

In this subsection we explain how to compute division with remainder and GCD (greatest common divisor) for univariate polynomials in X, whose coefficients are given by arithmetic circuits in variables $A_1, \ldots, A_n$. We rely on the description of the algorithms in Chapter 9 of [GG99]. In what follows, each time we discuss arithmetic circuits that compute a ratio of polynomials, we can think of the circuit as computing two outputs, one for the numerator and one for the denominator.

Lemma 2.7 *Let $f \in \mathbb{F}(A_1, \ldots, A_n)[X]$ be a polynomial of degree d such that $f(0) = 1$. Assume there is an arithmetic circuit of size s computing all of f's coefficients (possibly as a ratio of two polynomials), of size s. Then, for every m, one can add $\mathrm{poly}(d, m)$ many gates to the circuit to obtain a circuit computing all coefficients of a polynomial g (as well as the coefficient of f) such that $fg \equiv 1 \mod X^m$. Moreover, this new circuit can be computed in a black-box fashion, namely, we only add gates to the circuit of f and connect them either to other new gates or to the outputs of the circuit for f.*

The upper bound that we give on the size of the circuit is very crude and it can be greatly improved, but for sake of simplicity we give the crude bound.

Proof The proof basically follows from Algorithm 9.3 of [GG99]. In that algorithm we define $g_0 = 1$ and recursively compute $g_{i+1} \overset{\text{def}}{=} (2g_i - fg_i^2) \mod X^{2^i}$. It is shown in Theorem 9.2 there that $fg_i \equiv 1 \mod X^{2^i}$. Thus, we only have to compute g_i for $2^i > m$. By induction it is not hard to see that if there is a circuit of size s_i that outputs all the coefficients of f and of g_i (recall that g_i has degree smaller than 2^i in X) then there is a circuit of size $s_i + \mathrm{poly}(d, m)$ computing the coefficients of f and g_{i+1}. Indeed, each coefficient of g_{i+1} is a sum of $\mathrm{poly}(d, 2^i)$ terms, each of which is a product of a coefficient of f with two coefficients of g_i. As the size of the circuit grows additively at each step, after $O(\log m)$ steps we get a circuit of size $s + \mathrm{poly}(d, m)$. Thus, there is a circuit of size $s + \mathrm{poly}(d, m)$ computing $g_{\lceil \log m \rceil}$, as required. ■

Lemma 2.8 (Division with remainder) *Let $f, g \in \mathbb{F}(A_1, \ldots, A_n)[X]$, where g is a monic polynomial in X. Assume there is a circuit of size s computing all the coefficients of f and g with respect to X (possibly each coefficient is a ratio of two polynomials). Let $\deg(f), \deg(g) \leq d$. Then one can add to this circuit $\mathrm{poly}(d)$ many gates to obtain a circuit computing all coefficients of the polynomials h, r (as well as those of f, g) such that $f = hg + r$ with $\deg(r) < \deg(g)$. Moreover, this new circuit can be computed in a black-box fashion, namely, we only add gates to the circuit of f, g and connect them either to other new gates or to the outputs of the circuit for f, g.*

Proof The proof follows from Theorem 9.6 of [GG99] and uses Lemma 2.7. ■

As we can compute division with remainder we can also compute GCD's efficiently, using PIT.

Lemma 2.9 (GCD) *Suppose we have access to a PIT oracle. Let f and g be univariate polynomials of degree at most Δ in $\mathbb{F}(T, Y_1, Y_2, \ldots, Y_n)[X]$. Assume there is a size s arithmetic circuit computing all coefficients of f and g (possibly as ratios of polynomials). Then one can compute in time $s + \mathrm{poly}(\Delta)$ a circuit that outputs the coefficients of f, g and the (monic) GCD(f, g) in $\mathbb{F}(T, Y_1, Y_2, \ldots, Y_n)[X]$.*

Proof We note that in order to follow Euclid's algorithm it is enough to be able to compute division with remainder, and to detect when to stop. Computing division with remainder can be done via Lemma 2.8, and detecting when to stop can be done using the PIT oracle. Hence, the usual execution of Euclid's algorithm gives the required circuit. Note that in Lemmas 2.7 and 2.8 we do not need to compute a new circuit at any step, but rather we add polynomially many new gates to the circuit at hand. Thus the upper bound on size follows. ■

As noted above, all the bounds that we obtain in these lemmas are far from being optimal. One can go more carefully over the usual algorithms for GCD, division with remainder etc. to obtain circuits of size $\tilde{O}(s)$.

3 White-box factorization algorithm

In this section, we give our deterministic white-box factorization algorithm (assuming a determinstic white-box PIT algorithm).

We give the basic outline of our algorithm below. We will elaborate on the various steps of the algorithm in the later sections.

Input: An arithmetic circuit for the polynomial $f(X, Y_1 \dots, Y_n)$

1. If the characteristic of $\mathbb{F}$ is $p > 0$ then make f not a p'th power, if the characteristic is 0 then do nothing (See Section 3.1).
2. Make f monic in X. (See Section 3.2.)
3. Reduce to the case where f is squarefree. (See Section 3.3.)
4. Reduce to the case of bivariate factoring over a large field. (See Section 3.4.)
   (a) Define $\bar{f}(X, T, A_1, A_2, \dots, A_n) \overset{\text{def}}{=} f(X, TA_1, TA_2, \dots, TA_n)$.
   (b) Show that factors of $\bar{f}$ in $\mathbb{F}(A_1, A_2, \dots, A_n)[X, T]$ correspond to factors of f in $\mathbb{F}[X, Y_1, Y_2, \dots, Y_n]$
5. Univariate factorization. (See Section 3.5.)
   (a) Note that $\bar{f}(X, 0, A_1, A_2, \dots, A_n) \in \mathbb{F}[X]$.
   (b) Via univariate polynomial factorization, find an irreducible factor $g_0(X) \in \mathbb{F}[X]$ of $\bar{f}(X, 0,\ A_1, A_2, \dots, A_n)$.
   (c) Write
   $$\bar{f}(X, T, A_1, A_2, \dots, A_n) = g_0(X, T, A_1, A_2, \dots, A_n) \cdot h_0(X, T, A_1, A_2, \dots, A_n) \mod T.$$
   Now view this as an equation in $\mathbb{K}[X, T]$.
6. Hensel Lifting. (See Section 3.6.)
   For $k = O(\log d)$, Hensel lift k times to get
   $$\bar{f}(X, T, A_1, A_2, \dots, A_n) = g_k(X, T, A_1, A_2, \dots, A_n) \cdot h_k(X, T, A_1, A_2, \dots, A_n) \mod T^{2^k}.$$

7. Solve a linear system (See Section 3.7.)

   (a) Suppose $g_k(X, T, A_1, \ldots, A_n) = \sum_{i \leq d, j \leq D} c_{ij}(A_1, \ldots, A_n) X^i T^j$. Consider the following homogeneous system of linear equations over the field $\mathbb{F}(A_1, \ldots, A_n)$ in the variables R_{ij}, S_{ij} :

   $$\sum_{i<d, j \leq d} R_{ij} X^i T^j = \left(\sum_{i \leq D, j \leq D} c_{ij} X^i T^j \right) \left(\sum_{i \leq D, j \leq D} S_{ij} X^i T^j \right) \mod T^{2^k}.$$

   This is a system of $O(D^2)$ homogeneous linear equations in $O(D^2)$ unknowns. Solve this system to get a nontrivial solution (if any).

   (b) If there is no solution, then $\bar{f}$ is irreducible.

   (c) Otherwise, if there is a solution, we find a polynomial with nontrivial GCD with $\bar{f}$.

8. Compute the GCD (See Section 3.8.)
   Use it to obtain a nontrivial factor of f.

9. Continue by recursion to factor f completely (See Section 3.9).

3.1 Making f not a pth power

Suppose $\mathbb{F} = \mathbb{F}_{p^\ell}$ is a field of characteristic p. The case where f is a perfect pth power causes problems for the derivative-based methods that will be used. Here we see how to reduce to the case where f is not a pth power.

We first describe how to find the largest e such that f is a perfect p^eth power. It is easy to see that f is a perfect p^eth power (but not a p^{e+1}th power) if and only if:

1. for every variable X_i, the coefficient of the monomial X_i^j in $f(X_1, \ldots, X_n, X_{n+1})$ is zero whenever $p^e \nmid j$,

2. there is some some variable X_i, and some monomial M containing X_i^j (with $p^{e+1} \nmid j$) such that the coefficient of M in $f(X_1, \ldots, X_n, X_{n+1})$ is nonzero.

This can be easily checked by making $\text{poly}(n, \deg(f))$ calls to the given PIT algorithm (via Lemma 2.1).

Now suppose f is a perfect p^eth power, but not a p^{e+1}th power. Renaming the variables $X, Y_1, \ldots, Y_n$, we may assume the variable X has a monomial X^j, with $p^{e+1} \nmid j$, that appears in f with a nonzero coefficient (possibly as part of a larger monomial). Let us write

$$f(X, Y_1, \ldots, Y_n) = \sum_{i=0}^{d} a_i(Y_1, \ldots, Y_n) X^i.$$

Thus the polynomial $a_i(Y_1, \ldots, Y_n) \equiv 0$ whenever $p^e \nmid i$.

Then $f(X, Y_1, \ldots, Y_n)$ can be written as $f^*(X^{p^e}, Y_1, \ldots, Y_n)$. Notice that $f^*(Z, Y_1, \ldots, Y_n)$ is not a pth power.

We now show that the irreducible factors of f^* are in $1-1$ correspondence with the irreducible factors of f and that if h^* divides f^* then the corresponding h divides f, and viceversa.

Suppose we find a factor $h^*(Z, Y_1, \ldots, Y_n)$ of $f^*(Z, Y_1, \ldots, Y_n)$. Then $h^*(X^{p^e}, Y_1, \ldots, Y_n)$ divides $f^*(X^{p^e}, Y_1, \ldots, Y_n) = f(X, Y_1, \ldots, Y_n)$.

Conversely, if $h(X, Y_1, \ldots, Y_n)$ is an irreducible factor of $f(X, Y_1, \ldots, Y_n)$. Since f is a perfect p^eth power, and h is irreducible, we have that h^{p^e} must divide $f(X, Y_1, \ldots, Y_n)$ too. Note that $h^{p^e}(X, Y_1, \ldots, Y_n)$ is of the form $h^*(X^{p^e}, Y_1, \ldots, Y_n)$, and since $h^*(X^{p^e}, Y_1, \ldots, Y_n)$ divides $f(X, Y_1, \ldots, Y_n) = f^*(X^{p^e}, Y_1, \ldots, Y_n)$, we have that $h^*(Z, Y_1, \ldots, Y_n)$ divides $f^*(Z, Y_1, \ldots, Y_n)$.

We note that if f is computed by a circuit of size s then f^* can be computed by a circuit of size $\text{poly}(s, d)$. Indeed, Lemma 2.1 implies there is a circuit of size $\text{poly}(s, d)$ computing all coefficients $a_i(Y_1, \ldots, Y_n)$. All that is left to do now is to multiply each a_i with X^{i/p^e} (recall that $p^e | i$ when $a_i \not\equiv 0$).

Thus we have reduced to the case of factoring f^*, which is not a perfect pth power. However, we can now only produce arithmetic circuits for the p^eth powers of the irreducible factors of f, not the irreducible factors themselves.

Note that when $\mathbb{F} = \mathbb{Q}$ then we do not have to do anything in this step.

3.2 Making f monic

Let $f(X, Y_1, \ldots, X_n) \in \mathbb{F}[X, Y_1, \ldots, Y_n]$ be a polynomial of total degree d. We would like to find an invertible linear transformation $\mathbf{M} \in \mathbb{F}^{(n+1)\times(n+1)}$ of the variables that makes f monic in X. We would also like to ensure that the derivative $\frac{\partial f}{\partial X}$ is a nonzero polynomial (which is a condition that will be useful in Section 3.3). It is easy to convert a factorization of $f(\mathbf{M} \cdot (X, Y_1, \ldots, Y_n))$ into a factorization of $f(X, Y_1, \ldots, Y_n)$.

Consider the polynomial

$$g(\langle Z_{i,j} \rangle_{i,j\in[n+1]}, X, Y_1, \ldots, Y_n) = f(\mathbf{Z} \cdot (X, Y_1, \ldots, Y_n)),$$

where $\mathbf{Z}$ is an $(n+1) \times (n+1)$ matrix consisting of the formal variables $Z_{i,j}$, and $\cdot$ represents matrix-vector multiplication. Observe that we can construct an arithmetic circuit for g using the given arithmetic circuit for f.

Thus (by Lemma 2.1), for each $i \in \{0, 1, \ldots, d\}$ we can construct an arithmetic circuit for the coefficient $c_i(\mathbf{Z}, Y_1, \ldots, Y_n)$ of X^i in $g(\mathbf{Z}, X, Y_1, \ldots, Y_n)$.

If d is the total degree of f, it is easy to see that the polynomial $c_d(\mathbf{Z}, Y_1, \ldots, Y_n)$ is nonzero. Since f is not a pth power, there is some i with $p \nmid i$ for which either X^i or some Y_j^i appears within a monomial with a nonzero coefficient in the polynomial $f(X, Y_1, \ldots, Y_n)$. In particular by substituting $\mathbf{Z}$ to be a suitable permutation matrix, we see that the polynomial $c_i(\mathbf{Z}, Y_1, \ldots, Y_n)$ is not identically 0. We can find one such i with $p \nmid i$ using our given white-box PIT algorithm polynomially many times.

Now consider the nonzero polynomial

$$c_d(\mathbf{Z}, Y_1, \ldots, Y_n) \cdot c_i(\mathbf{Z}, Y_1, \ldots, Y_n) \cdot \det(\mathbf{Z}),$$

(for which we have explicit circuits). Using the white-box PIT algorithm (via Lemma 2.2), we can find a setting $\mathbf{M}$ of the variables $\mathbf{Z}$ such that:

$$c_d(\mathbf{M}, Y_1, \ldots, Y_n) \cdot c_i(\mathbf{M}, Y_1, \ldots, Y_n) \cdot \det(\mathbf{M})$$

is a nonzero polynomial in $Y_1, \ldots, Y_n$.

We then define $\tilde{f}(X, Y_1, \ldots, Y_n)$ to be the polynomial:

$$f(\mathbf{M} \cdot (X, Y_1, \ldots, Y_n)).$$

We know that $\mathbf{M}$ is invertible since $\det(\mathbf{M})$ is nonzero. If we write:

$$\tilde{f}(X, Y_1, \ldots, Y_n) = \sum_{i=0}^{d} \tilde{c}_i(Y_1, \ldots, Y_n) X^i,$$

then we have (by construction) that: $\tilde{c}_d(Y_1, \ldots, Y_n)$ is a nonzero polynomial (which must in fact be a nonzero constant in $\mathbb{F}$ since the total degree of $\tilde{f}$ is d), and $\tilde{c}_i(Y_1, \ldots, Y_n)$ is a nonzero polynomial.

We can compute this constant $\tilde{c}_d$ explicitly by picking an arbitrary $(y_1, y_2, \ldots, y_n) \in \mathbb{F}^n$ and substituting it into the arithmetic circuit we have for $\tilde{c}_d(Y_1, \ldots, Y_n)$.

Dividing $\tilde{f}$ by $\tilde{c}_d$, we get the desired $\tilde{f}$ that is an invertible linear transformation of f, monic in X, for which $\frac{\partial \tilde{f}}{\partial X}$ is a nonzero polynomial. Indeed the coefficient of X^{i-1} in $\frac{\partial \tilde{f}}{\partial X}$ is $i \cdot \tilde{c}_i(Y_1, \ldots, Y_n)$, which is not zero since $p \nmid i$ and $\tilde{c}_i(Y_1, \ldots, Y_n) \not\equiv 0$. Finally, redefine f to equal $\tilde{f}$, and note that it suffices to factor the new f.

3.3 Reduction to the squarefree case

We now assume that we have a monic $f(X, Y_1, \ldots, Y_n) \in \mathbb{F}[X, Y_1, \ldots, Y_n]$ (which is not a pth power in the event that $\mathbb{F}$ has characteristic $p > 0$), and we wish to reduce to the case that it is squarefree, namely, that no irreducible factor of f has multiplicity larger than one. Let $f = \sum_{i=0}^{d} X^i f_i(Y_1, \ldots, Y_n)$ and further assume that we are given a circuit computing each f_i. We can assume this wlog (see Lemma 2.1).

Observe that if f is not squarefree, i.e $f = g^2 h$ where $\deg(g) \geq 1$, then $\frac{\partial f}{\partial X} = 2\frac{\partial g}{\partial X} \cdot gh + g^2 \frac{\partial h}{\partial X} = g \cdot \tilde{h}$, for $\tilde{h} = 2\frac{\partial g}{\partial X} h + g \frac{\partial h}{\partial X}$. Note that by adding d additional gates to the circuit such that the ith new gate computes $i \cdot f_i$ we get a circuit of size $s + O(d)$ computing all coefficients of f and of $\frac{\partial f}{\partial X}$. Crucially, the polynomial $\frac{\partial f}{\partial X}$ is not the zero polynomial (because of the nonzero coefficient of X^i for some $p \nmid i$). Our initial work ensuring that f was not a pth power was in order to ensure this.

Now, using the GCD algorithm of Lemma 2.9 we get a circuit of size $s + \mathrm{poly}(d)$ computing all coefficients of $f, \frac{\partial f}{\partial X}$ and $\mathrm{GCD}(f, \frac{\partial f}{\partial X})$. We now observe some facts about this GCD. First, $g | \mathrm{GCD}(f, \frac{\partial f}{\partial X})$. Secondly, the degree of the GCD is smaller than the degree of f. Thirdly, using Lemma 2.8 we can find a circuit of size $s + \mathrm{poly}(d)$ computing all coefficients of $f, \frac{\partial f}{\partial X}, \mathrm{GCD}(f, \frac{\partial f}{\partial X})$ and q_1, where q_1 is such that $f = q_1 \cdot \mathrm{GCD}(f, \frac{\partial f}{\partial X})$. Thus, we managed to express f as a product of two distinct polynomials each having a small circuit, and each having degree at least 1.

We can continue in this fashion to get a circuit of size $s + \text{poly}(d)$ (the process can run for at most d steps each step adding $\text{poly}(d)$ many new gates (by Lemma 2.8)) that computes all coefficients of polynomials $q_1, \ldots, q_\ell$ such that $f = q_1 \cdots q_\ell$ where each q_i is nonconstant and squarefree.

We note that in the process above we may encounter factors of f that are perfect p powers, although f is not and whose partial derivative wrt X is zero. Thus, we repeat the earlier steps to make each factor have the desired properties. It is clear that this process takes polynomial time and incurs a one time blowup to the size of the circuit.

Thus, from now on we assume wlog that we have a circuit of size s computing a squarefree monic polynomial $f(X, Y_1, \ldots, Y_n) \in \mathbb{F}[X, Y_1, \ldots, Y_n]$.

3.4 Reducing to the bivariate case

As described earlier, the first step in our proof, after the preprocessing steps described above, is translating $f(X, Y_1, \ldots, Y_n)$ to a bivariate polynomial.

Let $f(X, Y_1, \ldots, Y_n) \in \mathbb{F}[X, Y_1, \ldots, Y_n]$ be a polynomial of total degree d, which is squarefree and monic in X. We know that the discriminant $D_f(Y_1, \ldots, Y_n)$ of f (w.r.t the variable X) is a nonzero polynomial, and we have an arithmetic circuit for it. Thus, using our white-box PIT algorithm and Lemma 2.2, we can find an $\mathbf{a} \in \mathbb{F}^n$ such that $D_f(\mathbf{a}) \neq 0$. By translating the origin, we assume $D_f(0, \ldots, 0) \neq 0$. Define $\bar{f}(X, T, A_1, \ldots, A_n) \in \mathbb{F}[X, T, A_1, \ldots, A_n]$ by:

$$\bar{f}(X, T, A_1, \ldots, A_n) = f(X, A_1 T, A_2 T, \ldots, A_n T).$$

To ease notations, we shall denote $\mathbb{K} = \mathbb{F}(A_1, \ldots, A_n)$.

We first prove that irreducible factors of $\bar{f}$ are in $1-1$ correspondence with those of f.

Lemma 3.1 *Let f and $\bar{f}$ be as above. Let $\bar{h}(X, T, A_1, \ldots, A_n) \in \mathbb{F}[X, T, A_1, \ldots, A_n]$ be a factor of $\bar{f}(X, T, A_1, \ldots, A_n)$. Then there exists $h(X, Y_1, \ldots, Y_n) \in \mathbb{F}[X, Y_1, \ldots, Y_n]$ such that:*

$$\bar{h}(X, T, A_1, \ldots, A_n) = h(X, A_1 T, A_2 T, \ldots, A_n T).$$

$$h(X, Y_1, \ldots, Y_n) \mid f(X, Y_1, \ldots, Y_n) \quad \textit{in } \mathbb{F}[X, Y_1, \ldots, Y_n].$$

Proof Denote $\bar{h}(X, T, A_1, \ldots, A_n) = \sum_{i,j} X^i T^j \bar{h}_{i,j}(A_1, \ldots, A_n)$. Observe that it must be the case that $\bar{h}_{i,j}$ is a homogeneous polynomial in $A_1, \ldots, A_n$ of degree j (since $\bar{f}$ is such). Let $h(X, Y_1, \ldots, Y_n) = \bar{h}(X, 1, A_1, \ldots, A_n)$. I.e., $h = \sum_{i,j} X^i \bar{h}_{i,j}(A_1, \ldots, A_n)$. It is straightforward to verify that h has the required properties. ■

Combining Gauss' lemma (Lemma 2.3) with Lemma 3.1, and noting that $\bar{f}$ is monic in X, we get:

Corollary 3.2 *Let f and $\bar{f}$ be as above. Let $\bar{h}(X, T, A_1, \ldots, A_n) \in \mathbb{K}[X, T]$ be a monic (in X) factor of $\bar{f}(X, T, A_1, \ldots, A_n)$. Then there exists $h(X, Y_1, \ldots, Y_n) \in \mathbb{F}[X, Y_1, \ldots, Y_n]$ such that:*

$$\bar{h}(X, T, A_1, \ldots, A_n) = h(X, A_1 T, A_2 T, \ldots, A_n T).$$

$$h(X, Y_1, \ldots, Y_n) \mid f(X, Y_1, \ldots, Y_n) \quad \textit{in } \mathbb{F}[X, Y_1, \ldots, Y_n].$$

Furthermore, $h(X, Y_1, \ldots, Y_n)$ simply equals $\bar{h}(X, 1, Y_1, \ldots, Y_n)$.

Thus, in order to factor $f(X, Y_1, \ldots, Y_n)$, it suffices to factor $\bar{f}(X, T, A_1, \ldots, A_n)$ into its factors in $\mathbb{K}[X, T] = \mathbb{F}(A_1, \ldots, A_n)[X, T]$. These factors in $\mathbb{K}[X, T]$, when made monic in X will, by the above fact, give us the factors of $f(X, Y_1, \ldots, Y_n)$ in $\mathbb{F}[X, Y_1, \ldots, Y_n]$. Conversely it is easy to see that every factor of $f(X, Y_1, \ldots, Y_n)$ in $\mathbb{F}[X, Y_1, \ldots, Y_n]$ can be obtained in this way.

3.5 Factoring $\bar{f}$ over $\mathbb{K}[X, T]$ 1: Univariate factorization

Consider the univariate polynomial $f(X, 0, 0, \ldots,) \in \mathbb{F}[X]$. We know that it is a nonzero polynomial of degree d (by the monicness of f in X), and that it is squarefree (by the nonvanishing of $D_f(0, 0, \ldots, 0)$).

The next step of the algorithm is to factorize $f(X, 0, \ldots, 0)$ over $\mathbb{F}[X]$ using known algorithms for univariate polynomial factorization. By the well known algorithms of Lenstra-Lenstra-Lovasz [LLL82] and Berlekamp [Ber70], this can be performed in time $\mathrm{poly}(t, d)$ (where t is as in Theorem 1).

Theorem 3.3 ([Ber70, LLL82]) *Let $f \in \mathbb{F}[X]$ be a monic polynomial of degree d then there is a deterministic algorithm computing all factors of f that runs in time* $\mathrm{poly}(n, s, d, t)$*, where:*

1. *$t = \ell \cdot p$, if $\mathbb{F} = \mathbb{F}_{p^\ell}$ is a field of characteristic p.*
2. *$t =$ maximum bit-complexity of the constants used in the circuit, if $\mathbb{F} = \mathbb{Q}$.*

Let $g_0(X)$ be an irreducible monic factor of $f(X, 0, \ldots, 0)$ in $\mathbb{F}[X]$. Let $h_0(X)$ be such that:

$$g_0(X) \cdot h_0(X) = f(X, 0, \ldots, 0).$$

By the squarefreeness of $f(X, 0, \ldots, 0)$, we have that $g_0(X)$ and $h_0(X)$ are relatively prime in $\mathbb{F}[X]$.

By definition of $\bar{f}$:

$$g_0(X) \cdot h_0(X) = \bar{f}(X, 0, A_1, \ldots, A_n).$$

Therefore, we have the following identity in the ring $\mathbb{K}[X, T]$:

$$g_0(X) \cdot h_0(X) = \bar{f}(X, T, A_1, \ldots, A_n) \mod T.$$

3.6 Factoring $\bar{f}$ over $\mathbb{K}[X, T]$ 2: Hensel lifting

We now perform Hensel lifting, as in Lecture 7 of [Sud99]. Define k to be an integer such that

$$k > 2 \log d + 1.$$

Let $a_0(X), b_0(X)$ be univariate polynomials in $\mathbb{F}[X]$ such that $a_0 g_0 + b_0 h_0 = 1$. Such $a_0(X), b_0(X)$ can be efficiently found using Euclid's algorithm. We will now show how to use Hensel lifting to lift the solution

$$g_0(X) \cdot h_0(X) = \bar{f}(X, T, A_1, \ldots, A_n) \mod T,$$

to a solution

$$g_k(X) \cdot h_k(X) = \bar{f}(X, T, A_1, \ldots, A_n) \mod T^{2^k}.$$

Lemma 3.4 (Lifting the solution) *Given polynomials* $f, g_i, h_i, a_i, b_i \in \mathbb{K}[X,T]$ *such that (a)* $f = g_i h_i \mod T^{2^i}$, *(b)* $a_i g_i + b_i h_i = 1 \mod T^{2^i}$ *and (c)* g_i *is monic in* X, *consider the following process:*

1. $m_{i+1} = f - g_i \cdot h_i$
2. $\hat{g}_{i+1} = g_i + b_i \cdot m_{i+1}$
3. $\hat{h}_{i+1} = h_i + a_i \cdot m_{i+1}$
4. $v_{i+1} = (\hat{g}_{i+1} - g_i)/T^{2^i}$
5. *Write* $v_{i+1} = g_i q_{i+1} + r_{i+1}$, *with* $\deg_X(r_{i+1}) < \deg_X(g_i)$, *(by the division alg. Lemma 2.8)*
6. $g_{i+1} \overset{\text{def}}{=} g_i + T^{2^i} \cdot r_{i+1}$
7. $h_{i+1} \overset{\text{def}}{=} \hat{h}_{i+1} \cdot (1 + T^{2^i} q_{i+1})$
8. $w_{i+1} = a_i g_{i+1} + b_i h_{i+1} - 1$
9. $a_{i+1} \overset{\text{def}}{=} a_i - a_i w_{i+1}$
10. $b_{i+1} \overset{\text{def}}{=} b_i - b_i w_{i+1}$

Then (a) $f = g_{i+i} h_{i+1} \mod T^{2^{i+1}}$, *(b)* $a_{i+1} g_{i+1} + b_{i+1} h_{i+1} = 1 \mod T^{2^{i+1}}$, *(c)* g_{i+1} *is monic in* X. *Also (d)* $g_{i+1} = g_i \mod T^{2^i}$, *and (e)* $h_{i+1} = h_i \mod T^{2^i}$. *Moreover* g_{i+i} *and* h_{i+1} *are the unique polynomials (mod* $T^{2^{i+1}}$*) satisfying properties* (a), (c), (d) *and* (e).

Proof Straightforward calculations show $m_{i+1} = 0 \mod T^{2^i}$, $\hat{g}_{i+1} \cdot \hat{h}_{i+1} = f \mod T^{2^{i+1}}$, $\hat{g}_{i+1} = g_i \mod T^{2^i}$, and $h_{i+1} = h_i \mod T^{2^i}$. However $\hat{g}_{i+1}$ may not be monic in X. Steps 4-7 show how to construct g_{i+1} and h_{i+1} with the same properties as $\hat{g}_{i+1}$ and $\hat{h}_{i+1}$, but now ensuring that the property of being monic in X also holds.

Observe that since g_i is monic in X and $\deg_X(r_{i+1}) < \deg_X(g_i)$, then by the definition of g_{i+1} we get that g_{i+1} is also monic in X. Moreover,

$$\begin{aligned}
g_{i+1} &= g_i + T^{2^i} \cdot r_{i+1} \\
&= g_i + T^{2^i} \cdot (v_{i+1} - g_i q_{i+1}) \\
&= g_i + T^{2^i} \cdot \Big((\hat{g}_{i+1} - g_i)/T^{2^i} - g_i q_{i+1} \Big) \\
&= \hat{g}_{i+1} - T^{2^i} \cdot g_i q_{i+1} \\
&= \hat{g}_{i+1} - T^{2^i} \cdot \hat{g}_{i+1} q_{i+1} \mod T^{2^{i+1}} \\
&= \hat{g}_{i+1} \cdot (1 - T^{2^i} q_{i+1}) \mod T^{2^{i+1}}
\end{aligned}$$

From this and the definition of h_{i+1}, it is easy to see that $g_{i+1} \cdot h_{i+1} = \hat{g}_{i+1} \cdot \hat{h}_{i+1} = f \mod T^{2^{i+1}}$. Moreover $g_{i+1} = g_i \mod T^{2^i}$ and $h_{i+1} = h_i \mod T^{2^i}$. From the definitions of a_{i+1} and b_{i+1}, it is again easy to verify that $a_{i+1}g_{i+1} + b_{i+1}h_{i+1} = 1 \mod T^{2^{i+1}}$. Also g_{i+1} is monic in X, and $deg_X(g_{i+1})$ equals $deg_X(g_i)$.

All it remains is to show that g_{i+1} and h_{i+1} are the unique polynomials (mod $T^{2^{i+1}}$) satisfying properties (a), (c), (d) and (e).

If possible, let g'_{i+1} and h'_{i+1} be polynomials such that

1. $g'_{i+1} \cdot h'_{i+1} = f \mod T^{2^{i+1}}$
2. $g'_{i+1} = g_i \mod T^{2^i}$ and $h'_{i+1} = h_i \mod T^{2^i}$
3. g'_{i+1} is monic in X

Let $\tilde{g}_{i+1} = g'_{i+1} - g_{i+1}$ and let $\tilde{h}_{i+1} = h'_{i+1} - h_{i+1}$. Observe that $\tilde{g}_{i+1} = \tilde{h}_{i+1} = 0 \mod T^{2^i}$. Let $u = a_{i+1}\tilde{g}_{i+1} - b_{i+1}\tilde{h}_{i+1}$. Then,

$$\begin{aligned}
g_{i+1}(1+u) &= g_{i+1}(1 + a_{i+1}\tilde{g}_{i+1} - b_{i+1}\tilde{h}_{i+1}) \\
&= g_{i+1} + a_{i+1}g_{i+1}\tilde{g}_{i+1} - b_{i+1}g_{i+1}\tilde{h}_{i+1} \\
&= g_{i+1} + (1 - h_{i+1}b_{i+1})\tilde{g}_{i+1} - b_{i+1}g_{i+1}\tilde{h}_{i+1} \mod T^{2^{i+1}} \\
&= g_{i+1} + \tilde{g}_{i+1} - b_{i+1}(h_{i+1}\tilde{g}_{i+1} + g_{i+1}\tilde{h}_{i+1}) \mod T^{2^{i+1}} \\
&= g'_{i+1} - b_{i+1}(h_{i+1}\tilde{g}_{i+1} + g'_{i+1}\tilde{h}_{i+1}) \mod T^{2^{i+1}} \quad \text{since } \tilde{h}_{i+1}(g_{i+1} - g'_{i+1}) = 0 \mod T^{2^{i+1}} \\
&= g'_{i+1} - b_{i+1}(h_{i+1}(g'_{i+1} - g_{i+1}) + g'_{i+1}(h'_{i+1} - h_{i+1})) \mod T^{2^{i+1}} \\
&= g'_{i+1} - b_{i+1}(f - f) \mod T^{2^{i+1}} \quad \text{by expanding and cancelling terms} \\
&= g'_{i+1} \mod T^{2^{i+1}}
\end{aligned}$$

Moreover since g_{i+1} and g'_{i+1} are both monic in X, and since $g_{i+1} = g'_{i+1} = g_i \mod T^{2^i}$, this implies that the X-degree of g'_{i+1}, g_{i+1} and g_i are all the same. Hence, considering the coefficient of the leading monomial in X we obtain $u = 0 \mod T^{2^{i+1}}$. Thus $g_{i+1}(1+u) = g'_{i+1} \mod T^{2^{i+1}}$ implies that $g_{i+1} = g'_{i+1} \mod T^{2^{i+1}}$. ■

We are given g_0, h_0 in $\mathbb{F}[X]$. Since $f(X, 0, 0, \ldots,)$ is squarefree, observe that g_0 and h_0 are relatively prime. Thus we can obtain $a_0(X), b_0(X)$, univariate polynomials in X, such that $a_0g_0 + b_0h_0 = 1$. We view g_0, h_0, a_0, b_0 as elements of $\mathbb{F}(A_1, \ldots, A_n)[X, T]$.

We can iterate the above lemma for $i = 0, 1, \ldots, k-1$, to obtain $g_k, h_k \in \mathbb{F}(A_1, \ldots, A_n)[X, T]$, such that $f = g_kh_k \mod T^{2^k}$, and such that g_k is monic in X, and $g_k = g_0 \mod T$.

Claim 3.5 *The pair g_k, h_k obtained above is the unique pair of polynomials (mod T^{2^k}) such that (a) $f = g_kh_k \mod T^{2^k}$, (b) g_k is monic in X, and (c) $g_k = g_0 \mod T$.*

Proof If possible, let g'_k, h'_k be another distinct pair of polynomials (mod T^{2^k}) satisfying (a), (b) and (c). Notice that (a), (b) and (c) imply that it also must hold that $h'_k = h_k = h_0 \mod T$.

For $0 \leq i \leq k$, let $\tilde{g}_i = g_k \mod T^{2^i}$, and $\tilde{h}_i = h_k \mod T^{2^i}$, $\tilde{g}'_i = g'_k \mod T^{2^i}$, and $\tilde{h}'_i = h'_k \mod T^{2^i}$. Consider the first i for which the pair $\tilde{g}_i$, $\tilde{h}_i$ is distinct from $\tilde{g}'_i$, $\tilde{h}'_i$. This pair would contradict the uniqueness part of Lemma 3.4.

■

Lemma 3.6 (Small circuits) *The polynomials g_k, h_k lie in $\mathbb{F}[X, T, A_1, \ldots, A_n]$. Let $D = \max\{d, 2^k\}$, then we can express g_k and h_k as the following: $g_k(X, T, A_1, \ldots, A_n) = \sum_{j,j' \leq D} c_{jj'}(A_1, \ldots, A_n) X^j T^{j'}$ and $h_k(X, T, A_1, \ldots, A_n) = \sum_{j,j' \leq D} d_{jj'}(A_1, \ldots, A_n) X^j T^{j'}$. Furthermore, there is a single arithmetic circuit C in the variables $A_1, \ldots, A_n$ where C computes all the coefficients $c_{jj'}$ of g_k and $d_{jj'}$ of h_k. C has size at most size$(f) + \mathrm{poly}(n, k, d)$, degree at most $d2^k$, and can be computed in time at most $\mathrm{poly}(n, k, d)$.*

Proof Observe that none of the steps in Lemma 3.4 require division in $\mathbb{K}$ (not even step 5 which uses the division algorithm as described in Lemma 2.8). Thus, each $m_i, \hat{g}_i, \hat{h}_i, g_i, h_i, q_i, a_i, b_i, v_i, q_i, r_i, w_i$ actually lies in $\mathbb{F}[X, T, A_1, \ldots, A_n]$.

Let C_i be a circuit in the input variables $A_1, \ldots, A_n$ that outputs each of the coefficients of $X^j T^{j'}$ for each of the polynomials $m_i, \hat{g}_i, \hat{h}_i, g_i, h_i, q_i, a_i, b_i, v_i, q_i, r_i, w_i$ that are computed at stage i. Let s_i be the size of C_i, and d_i be the degree of C_i. Then, applying Lemmas 2.1 and 2.8 when needed, we obtain a circuit C_{i+1} that outputs each of the coefficients of $X^j T^{j'}$ for each of the polynomials $m_{i+1}, \hat{g}_{i+1}, \hat{h}_{i+1}, g_{i+1}, h_{i+1}, q_{i+1}, a_{i+1}, b_{i+1}, v_{i+1}, q_{i+1}, r_{i+1}, w_{i+1}$ that are computed at stage $i + 1$, where the size of C_{i+1}, s_{i+1}, is at most $s_i + \mathrm{poly}(d)$, and the degree of C_{i+1}, d_{i+1}, is at most $2d_i$. Moreover, given C_i, C_{i+1} can be computed in time $\mathrm{poly}(d, n)$.

Thus by a simple induction argument we obtain the bounds in the lemma.

■

3.7 Factoring $\bar{f}$ over $\mathbb{K}[X, T]$ 3: Solving a linear system

We have that $\bar{f} = g_k \cdot h_k \mod T^{2^k}$, where g_k is monic, and $k > 2\log d + 1$. Moreover $g_k \in \mathbb{K}[X, T]$ can be expressed as $g_k = \sum_{i \leq D, j \leq D} c_{ij}(A_1, \ldots, A_n) X^i T^j$, where $D = \max\{d, 2^k\}$, and there is a polynomial sized arithmetic circuit (in the input variables $A_1, A_2, \ldots, A_n$) computing the various c_{ij}.

Now consider the following homogeneous system of linear equations over the field $\mathbb{F}(A_1, \ldots, A_n)$ in the variables R_{ij}, S_{ij} :

$$\sum_{i<d, j\leq d} R_{ij} X^i T^j = \left(\sum_{i\leq D, j\leq D} c_{ij} X^i T^j \right) \left(\sum_{i\leq D, j\leq D} S_{ij} X^i T^j \right) \mod T^{2^k}. \tag{1}$$

This is a system of $O(D^2)$ homogeneous linear equations in $O(D^2)$ unknowns.

If this system of linear equations has a nontrivial solution, then one such solution can be obtained by Lemma 2.6. We will soon show how to use such a solution to obtain a factor of $\bar{f}$.

Claim 3.7 *If $\bar{f}(X, T, A_1, \ldots, A_n)$ is reducible in $\mathbb{F}(A_1, \ldots, A_n)[X, T]$, then the linear system* (1) *has a nontrivial solution.*

Proof Suppose $\bar{f}$ is reducible.

Recall that we have that $\bar{f}(X, 0, \ldots, 0)$ is squarefree (see Section 3.4), that

$$\bar{f}(X, 0, A_1, \ldots, A_n) = g_0(X, 0, A_1, \ldots, A_n) \cdot h_0(X, 0, A_1, \ldots, A_n),$$

that $g_0(X, 0, A_1, \ldots, A_n) \in \mathbb{F}[X]$ is irreducible, and that $g_0(X, 0, A_1, \ldots, A_n)$ and $h_0(X, 0, A_1, \ldots, A_n)$ are relatively prime. Let $c(X, T, A_1, \ldots, A_n) \in \mathbb{F}(A_1, \ldots, A_n)[X, T]$ be the unique irreducible factor of $\bar{f}(X, T, A_1, \ldots, A_n)$ for which $g_0(X, 0, A_1, \ldots, A_n)$ divides $c(X, 0, A_1, \ldots, A_n)$. Uniqueness of c follows from the fact that $\bar{f}(X, 0, \ldots, 0)$ is squarefree. Let $c'(X, T, A_1, \ldots, A_n)$ be such that $c \cdot c' = \bar{f}$. Note that since $\bar{f}$ is monic, thus by Lemma 2.3, so are c and c'.

Let $t_0(X, 0, A_1, \ldots, A_n)$ be such that

$$g_0(X, 0, A_1, \ldots, A_n) \cdot t_0(X, 0, A_1, \ldots, A_n) = c(X, 0, A_1, \ldots, A_n).$$

We want to apply Hensel lifting to this situation. Note that $g_0(X, 0, A_1, \ldots, A_n)$ and $t_0(X, 0, A_1, \ldots, A_n)$ are relatively prime in $\mathbb{K}[X]$ (again, this follows since $\bar{f}(X, 0, A_1, \ldots, A_n)$, and hence $c(X, 0, A_1, \ldots, A_n)$, is squarefree). Thus there exists $\tilde{a}_0, \tilde{b}_0 \in \mathbb{K}[X]$ such that $g_0 \cdot \tilde{a}_0 + t_0 \tilde{b}_0 = 1 \mod T$. Thus we can perform the lifting, as given in Lemma 3.4. After k steps of lifting, we get $g_k^*, t_k, \tilde{a}_k, \tilde{b}_k \in \mathbb{K}[X, T]$ such that g_k^* is monic in X, and:

$$g_k^* t_k = c \mod T^{2^k}.$$

Thus:

$$\bar{f} = g_k \cdot h_k = g_k^* \cdot t_k \cdot c' \mod T^{2^k}. \qquad (2)$$

Since g_k, g_k^* are both monic in X, and $g_k = g_k^* = g_0 \mod T$, we conclude by Claim 3.5 that $g_k = g_k^* \mod T^{2^k}$. Hence, Equation (2) is equivalent to

$$c \cdot c' = \bar{f} = g_k \cdot h_k = g_k \cdot t_k \cdot c' \mod T^{2^k}.$$

In other words,

$$c'(c - t_k \cdot g_k) = 0 \mod T^{2^k}.$$

As c' is monic in X it follows that

$$(c - t_k \cdot g_k) = 0 \mod T^{2^k}.$$

Therefore,

$$\sum_{i<d, j\leq d} R_{ij} X^i T^j = c,$$

and

$$\sum_{i\leq D, j\leq D} S_{ij}X^iT^j = t_k$$

gives a nontrivial solution to the linear system, as desired. ■

We now see how to extract a factor of $\bar{f}$ using any nontrivial solution to the linear system (1). Consider a nontrivial solution and define the polynomials:

$$r(X,T,A_1,\ldots,A_n) = \sum_{i<d, j\leq d} R_{ij}X^iT^j,$$

$$s(X,T,A_1,\ldots,A_n) = \sum_{i\leq D, j\leq D} S_{ij}X^iT^j.$$

Claim 3.8 *In the ring $\mathbb{F}(T,A_1,\ldots,A_n)[X]$, $r(X,T,A_1,\ldots,A_n)$ and $\bar{f}(X,T,A_1,\ldots,A_n)$ have nontrivial GCD.*

Proof Let $u(X,T,A_1,\ldots,A_n) \in \mathbb{F}[A_1,\ldots,A_n,T]$ be the resultant of $r(X)$ and $\bar{f}(X)$. Then, the T-degree of u is at most $2d^2$. Moreover, by Lemma 2.4, there exist $v(X,T,A_1,\ldots,A_n)$, $w(X,T,A_1,\ldots,A_n)$ such that:

$$v \cdot r + w \cdot \bar{f} = u.$$

Thus:

$$\begin{aligned} v \cdot r + w \cdot \bar{f} &= u \mod T^{2^k} \\ v \cdot g_k \cdot s + w \cdot g_k \cdot h_k &= u \mod T^{2^k} \\ g_k \cdot (v \cdot s + w \cdot h_k) &= u \mod T^{2^k}. \end{aligned}$$

Recall that the right hand side is a polynomial in $\mathbb{F}[A_1,\ldots,A_n,T]$, and thus does not depend on the variable X. However the polynomial g_k appearing on the left hand side is monic in the variable X. The only way this equation can hold is if $v \cdot s + w \cdot h_k$ equals $0 \mod T^{2^k}$, and thus $u = 0 \mod T^{2^k}$. By our bound on the T-degree of u, and since $2d^2 < 2^k$,[2] we get that u is identically 0. Thus, by Lemma 2.4, $r(X)$ and $\bar{f}(X)$ have a nontrivial GCD in $\mathbb{F}(T,A_1,\ldots,A_n)[X]$. ■

3.8 Factoring $\bar{f}$ over $\mathbb{K}[X,T]$ 4: Computing the GCD

Let $\bar{h}(X,T,A_1,\ldots,A_n)$ be the monic GCD of $r(X)$ and $\bar{f}(X)$ in $\mathbb{F}(T,A_1,\ldots,A_n)[X]$. A small arithmetic circuit (i.e. of size size(f) + poly(d,n)) for the coefficient of each monomial X^i in $\bar{h}$ can be computed using Lemmas 2.1 and 2.9. By Gauss' Lemma (Lemma 2.3), $\bar{h}(X,T,A_1,\ldots,A_n)$ lies in $\mathbb{K}[X,T]$. Thus by Corollary 3.2, for $h(X,Y_1,\ldots,Y_n) = \bar{h}(X,1,Y_1,\ldots,Y_n)$, we have that

$$h(X,Y_1,\ldots,Y_n) \in \mathbb{F}[X,Y_1,\ldots,Y_n],$$

and

$$h(X,Y_1,\ldots,Y_n) \mid f(X,Y_1,\ldots,Y_n) \quad \text{in } \mathbb{F}[X,Y_1,\ldots,Y_n].$$

[2] It is here where we use our choice $k > 2\log d + 1$.

3.9 Obtaining a complete factorization

So far we only found a non-trivial factor h of f. To obtain a complete the factorization we compute a circuit for $f, h, f/h$. By our result on the complexity of h and Lemma 2.8 we can achieve this with a circuit of size $\mathrm{size}(f) + \mathrm{poly}(d, n)$. Furthermore, both h and f/h are monic and squarefree. Thus, we can continue to factor them until we are left with irreducible factors. At each step of the factorization we can check whether we have found a trivial factor by running the PIT algorithm with the original polynomial. For example, checking whether $f = h$ will tell us whether f is irreducible or not, etc.

Combining all steps above we obtain a proof of Theorem 1.

4 Open Questions

We conclude by listing some open problems.

1. If $\mathbb{F}_{p^\ell}$ has characteristic p, and $g(X_1, \ldots, X_n) \in \mathbb{F}_{p^\ell}[X_1, \ldots, X_n]$ is such that g^p has a small arithmetic circuit, then does g have a small arithmetic circuit?
2. One can consider the problem of PIT for polynomial size circuits without a polynomial bound on the degree of the circuits. How does this problem relate to the problem of PIT with a polynomial bound on the degree of the circuits? What can be said about the problem of factorization of polynomials computed by polynomial size circuits (without a polynomial bound on their degree)?
3. Suppose a multivariate polynomial f can be computed by a small formula/algebraic branching program. Does it follow that all the factors of f can be computed by small formulas/algebraic branching programs? What if f is computed by a small depth circuit?
4. Can factorization of univariate polynomials of degree d over the finite field $\mathbb{F}_{p^\ell}$ be done deterministically in time $\mathrm{poly}(d, \ell \log p)$?

References

[Agr05] M. Agrawal. Proving lower bounds via pseudo-random generators. In *Proceedings of the 25th FSTTCS*, volume 3821 of *LNCS*, pages 92–105, 2005.

[AKS04] M. Agrawal, N. Kayal, and N. Saxena. Primes is in P. *Annals of Mathematics*, 160(2):781–793, 2004.

[Ber70] E. R. Berlekamp. Factoring polynomials over large finite fields. *Mathematics of Computation*, 24(111):713–735, 1970.

[DL78] R. A. DeMillo and R. J. Lipton. A probabilistic remark on algebraic program testing. *Inf. Process. Lett.*, 7(4):193–195, 1978.

[DSY09] Z. Dvir, A. Shpilka, and A. Yehudayoff. Hardness-randomness tradeoffs for bounded depth arithmetic circuits. *SIAM J. on Computing*, 39(4):1279–1293, 2009.

[Gat06] J. von zur Gathen. Who was who in polynomial factorization:. In *ISSAC*, page 2, 2006.

[GG99] J. von zur Gathen and J. Gerhard. *Modern computer algebra.* Cambridge University Press, 1999.

[GKKS13] A. Gupta, P. Kamath, N. Kayal, and R. Saptharishi. Arithmetic circuits: A chasm at depth three. In *54th Annual FOCS*, pages 578–587, 2013.

[HS80] J. Heintz and C. P. Schnorr. Testing polynomials which are easy to compute (extended abstract). In *Proceedings of the 12th annual STOC*, pages 262–272, 1980.

[Kal89] E. Kaltofen. Factorization of polynomials given by straight-line programs. In S. Micali, editor, *Randomness in Computation*, volume 5 of *Advances in Computing Research*, pages 375–412. 1989.

[Kal90] E. Kaltofen. Polynomial factorization 1982–1986. In DV Chudnovsky and RD Jenks, editors, *Computers in Mathematics.* Marcel Dekker, New York, 1990.

[Kal92] E. Kaltofen. Polynomial factorization 1987–1991. In *LATIN*, pages 294–313. Springer, 1992.

[Kal95] E. Kaltofen. Effective Noether irreducibility forms and applications. *J. of Computer and System Sciences*, 50(2):274–295, 1995.

[Kal03] E. Kaltofen. Polynomial factorization: a success story. In *ISSAC*, pages 3–4, 2003.

[Kay07] N. Kayal. *Derandomizing some number-theoretic and algebraic algorithms.* PhD thesis, Indian Institute of Technology, Kanpur, India, 2007.

[KI04] V. Kabanets and R. Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004.

[KT90] E. Kaltofen and B. M. Trager. Computing with polynomials given by black boxes for their evaluations: Greatest common divisors, factorization, separation of numerators and denominators. *J. of Symbolic Computation*, 9(3):301–320, 1990.

[LLL82] A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

[MVV87] K. Mulmuley, U. Vazirani, and V. Vazirani. Matching is as easy as matrix inversion. *Combinatorica*, 7(1):105–113, 1987.

[Sch80] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980.

[Sud99] M. Sudan. Algebra and computation. http://people.csail.mit.edu/madhu/FT98/course.html, 1999. Lecture notes.

[SV10] A. Shpilka and I. Volkovich. On the relation between polynomial identity testing and finding variable disjoint factors. In *ICALP (1)*, pages 408–419, 2010.

[SY10] A. Shpilka and A. Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends in Theoretical Computer Science*, 5(3-4):207–388, 2010.

[Zip79] R. Zippel. Probabilistic algorithms for sparse polynomials. In *EUROSAM*, pages 216–226, 1979.