Reprinted from JOURNAL OF COMPUTER AND SYSTEM SCIENCES
All Rights Reserved by Academic Press, New York and London

Vol. 31, No. 2, October 1985
Printed in Belgium

Irreducibility of Multivariate Polynomials*

JOACHIM VON ZUR GATHEN

Department of Computer Science, University of Toronto, Toronto, M5S 1A4 Canada

This paper deals with the problem of computing the degrees and multiplicities of the irreducible factors of a given multivariate polynomial. This includes the important question of testing for irreducibility. A probabilistic reduction from multivariate to bivariate polynomials is given, over an arbitrary (effectively computable) field. It uses an expected number of field operations (and certain random choices) that is polynomial in the length of a computation by which the input polynomial is presented, and the degree of the polynomial. Over algebraic number fields and over finite fields, we obtain polynomial-time probabilistic algorithms. They are based on an effective version of Hilbert's irreducibility theorem. © 1985 Academic Press, Inc.

1. INTRODUCTION

The problem of factoring polynomials has a venerable history going back to the last century. The first polynomial-time algorithms are Berlekamp's [2, 3] (probabilistic) methods over finite fields. Zassenhaus [52] proposed a Hensel lifting method for integral polynomials, but no polynomial-time algorithm was known for more than a decade. Lenstra, Lenstra and Lovász [32] provided polynomial-time factorization for univariate polynomials over the rational numbers, and Kaltofen [18, 19, 22] for multivariate polynomials. The subsequent results by Chistov and Grigoryev [6], von zur Gathen and Kaltofen [11], Landau [26], Lenstra [29–31] show that multivariate polynomials over algebraic number fields or finite fields can be factored in polynomial time.

All factoring algorithms rely on a modular approach, which eventually reduces the given problem to that of univariate polynomials over finite fields, which is then solved by some variant of Berlekamp's algorithm. An unpleasant phenomenon is that irreducible polynomials may have reducible modular images; the older algorithms used trial combinations of these factors, and incurred exponential cost in the worst case [3, 24]. In practice, however, this phenomenon seems to occur so rarely that for implementations it is not a real problem. (Weinberger [49] proved existence of a polynomial-time algorithm to compute the number of factors of a univariate polynomial with rational coefficients assuming the extended Riemann hypothesis.)

* An Extended Abstract of this work appeared in "Proc. 24th Annu. IEEE Sympos. Foundations of Computer Science," Tucson, AZ, 1983, pp. 172–179. This work was partially supported by NSERC Grant 3-650-126-40 and Schweizerischer Nationalfonds Grant 2175-0.83.

0022-0000/85 $3.00

As an explanation of the above empirical observation, sometimes Hilbert's irreducibility theorem was cited. It states that under some (and in fact, most) substitutions an irreducible multivariate polynomial with rational coefficients remains irreducible. However, the usual versions of this theorem are ineffective and do not provide an algorithmic approach. Heintz and Sieveking [15] and Kaltofen [18, 19] have established polynomial-time algorithms with the help of certain variants of Hilbert's irreducibility theorem. The central result of this paper is a probabilistic effective version of Hilbert's irreducibility Theorem for polynomials over arbitrary fields.

The polynomial-time algorithms mentioned above use a number of operations which is polynomial in some "size" $s(f)$ of an input polynomial $f \in F[x_1, ..., x_n]$. We will count the arithmetic operations in F; if the elements of F are represented over some finite alphabet and we can estimate the size of intermediate results, then we get a bound on the number of "bit operations."

Disregarding the question of representation of field elements, there are (at least) four different ways of representing a polynomial $f \in F[x_1, ..., x_n]$ of degree d, each giving a notion of "size."

The first is the dense representation, where the coefficient in f of each monomial in $x_1, ..., x_n$ of degree at most d is given. Thus there are $\beta_{d,n} = \binom{d+n}{d}$ coefficients to be specified, and since a monomial of degree d can be represented by its coefficient and d factors, the "dense size" of f is $s_{\text{dense}}(f) \leqslant (d+1)\,\beta_{d,n}$. (We neglect the $O(\log n)$ cost of encoding the index i of x_i.)

The second one is the sparse representation, where a sequence of pairs

$$(\text{monomial } x_1^{e_1} \cdots x_n^{e_n}, \text{coefficient } f_e \in F)$$

is given, with $f = \sum_{e \in \mathbb{N}^n} f_e x_1^{e_1} \cdots x_n^{e_n}$. If we consider $e_1 + \cdots + e_n + 1$ as the size of such a pair, then we have for the "sparse size" $s_{\text{spar}}(f)$

$$d + k \leqslant s_{\text{spar}}(f) \leqslant (d+1)\,k,$$

if f has k nonzero coefficients. The dense representation is of course a special case of the sparse one.

The third representation is by a formula (or "arithmetic expression" or "term") involving the operations $x_1, ..., x_n$, constants from F, and $+, -, *, /$. The size of a formula is the number of operations used, and the "formula size" $s_{\text{form}}(f)$ is the size of a smallest formula for f. (It is common to count only the operations $+, -, *, /$ for the formula size; including inputs and constants in the count changes it at most by a factor of 3, and here is more consistent with the computation model discussed below.) The sparse representation is the special case of a formula which is a sum of products of one constant and variables.

The fourth representation is by a computation using the operations $x_1, ..., x_n$, constants from F, and $+, -, *, /$. The size of computation is the number of operations used, and the "computation size" $s_{\text{comp}}(f)$ is the size of a smallest computation for f. A formula is a special computation, with fan-out at most 1.

We clearly have

$$s_{\text{dense}} \geqslant s_{\text{spar}} \geqslant s_{\text{form}} \geqslant s_{\text{comp}},$$

and for each inequality, there are examples where the gap is exponential. We remark that if the number n of variables is constant and the degree d of f is polynomial in $s_{\text{comp}}(f)$, then all four sizes are polynomially related, since $s_{\text{dense}}(f) \leqslant (d+n)^{n+2}$. (In view of the symmetry of $\beta_{d,n}$, one may interchange the roles of d and n in this remark.)

The multivariate factoring algorithms mentioned above have running time polynomial in s_{dense}. In this paper we consider the problem of finding the "factorization pattern" of a polynomial, i.e., the degrees and multiplicities of its irreducible factors. This subsumes of course the problem of testing for irreducibility. We give a probabilistic reduction for this problem from multivariate to bivariate polynomials, for which the number of arithmetic steps used is polynomial in the size s of a computation by which the input polynomial f is presented, and the degree d of f. (We cannot say "polynomial in $s_{\text{comp}}(f) + d$," since a given computation may have length more than polynomial in $s_{\text{comp}}(f)$, and it seems difficult to then find a computation of short length for f; see Strassen [45, Problem 1.2].) It is clear that d may be exponential in s, and already very simple questions, e.g., whether the gcd of two univariate polynomials is nontrivial, are *NP*-hard if s is the only parameter describing the input size [34].

The reduction for the factorization pattern is based on Theorem 4.5, which gives a probabilitstic effective version of Hilbert's irreducibility theorem. It states that over an arbitrary field for certain random substitutions, which reduce multivariate to bivariate polynomials, the factorization pattern remains unchanged with high probability. The proof of the effective irreducibility theorem uses methods of algebraic geometry. We quote a Bertini theorem from Lang's textbook [27] that asserts that a general hyperplane section of an irreducible variety is irreducible. Apart from this theorem, only basic notions from the first chapter of Shafarevich's textbook [39] are used.

Using the results mentioned above for bivariate polynomials, we obtain probabilistic polynomial-time bit computations for the factorization pattern of multivariate polynomials over two types of fields. The first type, the algebraic number fields, is discussed in Section 6 and includes of course the important case of the rational numbers. Here a problem is to control the size of intermediate results when computations are evaluated for specific inputs. We represent a probabilistic simulation of a computation in a number of bit operations which is polynomial in the input plus output size. The second type are the finite fields, considered in Section 7. Now the field may not have enough elements to make the probabilistic algorithms work, and we extend the field. In general, when one makes algebraic extensions of fields, polynomials have a tendency to split. We prove that for certain extensions—easy to describe and arbitrarily large—this does not happen.

Heintz and Sieveking [15] have given a test for absolute irreducibility (i.e., irreducibility in $\mathbb{C}[x_1,..., x_n]$) of integral polynomials. This has been improved by Kaltofen [21] to random polynomial-time, also allowing a fast parallel version.

A more difficult problem than testing for irreducibility is to actually factor a given multivariate polynomial. A heuristic approach was given in Zippel [53]; solutions are given in [10, 23]. The expected running time of those algorithms is polynomial in $s_{\text{spar}}(f)$ and $\deg f$; [10] assumes that the number of factors is bounded. It remains a challenge to see whether the cost for factoring can be made polynomial in the size of a computation for f and $\deg f$.

2. The Bertini Theorem

The theorems going back to Bertini [4] come in several flavors. They usually assert that if an algebraic variety (embedded in some affine or projective space) has a certain property, then—under suitable conditions— also the intersection with a general hyperplane has this property. Properties considered include smoothness, normality and the case of interest to us: irreducibility. The first rigorous proofs of this case seem due to van der Waerden [48] and Zariski [50]; see Jouanolou [17] for a modern approach.

In the context of algebraic computations, Bertini's theorem has been used by Heintz and Sieveking [15] for testing whether integer polynomials are irreducible over $\mathbb{C}$. In this section, we put Bertini's theorem in the form that we need. It then asserts that for an irreducible polynomial over an algebraically closed field in n variables there exists a linear substitution for $n-2$ of the variables such that the resulting bivariate polynomial is irreducible. We use this to show in Lemma 4.3 that "almost all" substitutions have this property.

We will use substitutions by linear functions of two variables throughout the paper, and it is convenient to have a notation for them.

Definition 2.1. If F is a field, $n \geqslant 2$, $f \in F[x_1,..., x_n]$ and

$$t = (u, v, w) = (u_3,..., u_n, v_3,..., v_n, w_3,..., w_n) \in F^{3(n-2)},$$

then we define $f\{t\}$ as

$$f\{t\} = f(x_1, x_2, u_3 x_1 + v_3 x_2 + w_3,..., \\ u_n x_1 + v_n x_2 + w_n) \in F[x_1, x_2].$$

Theorem 2.2 (Bertini). *Let K be an algebraically closed field, $n \geqslant 2$ and $f \in K[x_1,..., x_n]$ irreducible. Then there exists an algebraically closed field L containing K and $t \in L^{3(n-2)}$ such that $f\{t\} \in L[x_1, x_2]$ is irreducible.*

Proof. We prove the theorem for all algebraically closed fields K and all polynomials by induction on n. We can assume that $n \geqslant 3$. Let $y_1,..., y_{n+1}$ be indeterminates over $K(x_1,..., x_n)$, F an algebraically closed field containing $K(y_1,..., y_{n+1})$, and

$$y = y_1x_1 + \cdots + y_{n-1}x_{n-1} - y_{n+1} \in F[x_1,..., x_n].$$

By the Bertini theorem in Lang [27, Chap. VIII, Proposition 12], the ideal

$$I = (f, y + y_nx_n) \subseteq F[x_1,..., x_n]$$

is prime. If we consider the embedding

$$\phi: F[x_1,..., x_{n-1}] \to F[x_1,..., x_n]$$

which is the identity on $\{x_1,..., x_{n-1}\}$, and

$$g = f(x_1,..., x_{n-1}, -y/y_n) \in F[x_1,..., x_{n-1}],$$

then $\phi^{-1}(I)$ is prime, and $g \in \phi^{-1}(I)$. Considering the section

$$\psi: F[x_1,..., x_n] \to F[x_1,..., x_{n-1}]$$

of ϕ with $\psi(x_n) = -y/y_n$, one sees that $\phi^{-1}(I) \subseteq (g)$. It follows that g is irreducible. Applying the induction hypothesis to g, we find an algebraically closed field $L \supseteq F \supseteq K$ and $\bar{t} = (u, v, w) \in L^{3(n-3)}$ such that

$$g\{\bar{t}\} = f(x_1, x_2, u_3x_1 + v_3x_2 + w_3,..., u_{n-1}x_1 + v_{n-1}x_2 + w_{n-1},$$
$$\frac{-1}{y_n}(y_1x_1 + y_2x_2 + \sum_{3 \leqslant j < n} y_j(u_jx_1 + v_jx_2 + w_j) - y_{n+1})) \in L[x_1, x_2]$$

is irreducible. Then

$$t = \left(u_3,..., u_{n-1}, \frac{-1}{y_n}\left(y_1 + \sum_{3 \leqslant j < n} y_ju_j\right),\right.$$
$$v_3,..., v_{n-1}, \frac{-1}{y_n}\left(y_2 + \sum_{3 \leqslant j < n} y_jv_j\right),$$
$$\left. w_3,..., w_{n-1}, \frac{-1}{y_n}\left(\sum_{3 \leqslant j < n} y_jw_j - y_{n+1}\right)\right) \in L^{3(n-2)}$$

satisfies the claim of the theorem. ▮

The following question comes up naturally: does Theorem 2.2 also hold for simple substitutions $x_i = w_i \in L$? We briefly discuss this question and give a criterion.

The answer to the question is negative in general: $f = (x_1 + x_2)^2 - x_3 \in K[x_1, x_2, x_3]$ is irreducible, but for any algebraically closed $L \supseteq K$ and $w \in L$ the polynomial $(x_1 + x_2)^2 - w \in L[x_1, x_2]$ is reducible.

For the criterion below we first note that if x_1, say, does not occur in f, then $f(x_1, x_2, w_3, ..., w_n) \in L[x_2]$ is univariate, hence reducible for algebraically closed L (assuming degree at least 2). We can therefore assume that both x_1 and x_2 occur in f, and then the composition

$$K[x_3, ..., x_n] \to K[x_1, ..., x_n] \to K[x_1, ..., x_n]/(f) = R$$

is injective, so that we get an embedding from $K(x_3, ..., x_n)$ into the quotient field Q of R.

THEOREM 2.3. *Let K be an algebraically closed field, $n \geqslant 2$, and $f \in K[x_1, ..., x_n]$ irreducible, with x_1 and x_2 occurring in f. The following are equivalent:*

(i) *$f(x_1, x_2, w_3, ..., w_n)$ is irreducible for some $w \in K^{n-2}$.*

(ii) *$f(x_1, x_2, w_3, ..., w_n)$ is irreducible for "almost all" $w \in K^{n-2}$.*

(iii) *$K(x_3, ..., x_n)$ is algebraically closed in Q.*

(iv) *f is irreducible in $L[x_1, x_2]$, where L is an algebraic closure of $K(x_3, ..., x_n)$.*

The result will not be needed in the rest of the paper, and we forego a proof.

3. CONES

In this section we prove that mappings given by polynomials of small degree can be separated from points outside (the closure of) their image by test polynomials of small degree (Lemma 3.3). This will be used in the next section to separate the reducible polynomials from some irreducible ones.

The first proof of this lemma uses only cones and other elementary notions from algebraic geometry, as, e.g., in Shafarevich [39, Chap. I]. We assume this material throughout the section. The reader more familiar with algebraic geometry may skip to the end of this section for a second, more concise proof.

We recall the standard definition of a cone. If $X \subseteq F^m$ is a closed irreducible subvariety of dimension n, and $L \subseteq F^m \setminus X$ an affine linear space of dimension $i \leqslant m-n-2$, then

$$\begin{aligned} C(X, L) &= \{(1-c)\,x + cl \in F^m : x \in X, l \in L, c \in F\} \\ &= \{y \in F^m : \exists x \in X \;\; \exists l \in L \text{ such that } y \text{ lies on the line} \\ &\qquad \text{through } x \text{ and } l\} \subseteq F^m \end{aligned}$$

is the cone over X with vertex L. If $L = \{a\}$ consists of a single point, we write

$C(X, a)$ for $C(X, \{a\})$; similarly for $X = \{x\}$. If furthermore $\varphi: F^n \to F^m$ and $\lambda: F^l \to F^m$ are mappings with im $\varphi = X$, im $\lambda = L$, and λ linear, then

$$\begin{aligned} C(\varphi, \lambda) \colon F^n \times F^l \times F &\to F^m \\ (a, b, c) &\mapsto (1-c)\,\varphi(a) + c\lambda(b) \end{aligned}$$

is a mapping with image $C(X, L)$.

Lemma 3.1. *Let F be an algebraically closed field, $X \subseteq F^m$ a closed irreducible variety of dimension $n \leqslant m-2$, and $h \in F^m \setminus X$. Then there exists an affine linear space $L \subseteq F^m \setminus X$ of dimension $m-n-2$ such that the cone $Y = C(X, L)$ has dimension $m-1$, and $h \notin \bar{Y}$, where $\bar{Y}$ is the closure of Y in F^m.*

Proof. We show by induction on i for $0 \leqslant i \leqslant m-n-2$ that there exists a linear space $L_i \subseteq F^m \setminus X$ of dimension i such that the cone $Y_i = C(X, L_i)$ has dimension $n+i+1$, and $h \notin \bar{Y}_i$.

For the case $i = 0$, we consider an embedding $F^m \subseteq \mathbb{P}^m$ of F^m into projective space, the closure $\bar{X}$ of X in $\mathbb{P}^m$, and the triples of collinear points

$$\begin{aligned} T = \{(x, y, z) \in \mathbb{P}^m \times \mathbb{P}^m \times \mathbb{P}^m \colon & x = y \text{ or } x = z \\ & \text{or } y = z \text{ or } x, y, z \text{ lie on one line}\}. \end{aligned}$$

T is a closed subset of $(\mathbb{P}^m)^3$, since x, y, z are collinear if and only if the $3 \times (m+1)$ matrix given by the projective coordinates of x, y, z has rank at most 2. Let $\pi_2 \colon T \to \mathbb{P}^m$ be induced by the projection onto the second factor. For $a \in \mathbb{P}^m \setminus \bar{X}$, consider the projective cone over $\bar{X}$ with vertex a:

$$\begin{aligned} C_a &= \{y \in \mathbb{P}^m \colon \exists x \in \bar{X} \text{ such that } y \text{ lies on the line through } x \text{ and } a\} \\ &= \pi_2(T \cap (\bar{X} \times \mathbb{P}^m \times \{a\})). \end{aligned}$$

The fibers of the projection of $T \cap (\bar{X} \times \mathbb{P}^m \times \{a\})$ onto $\bar{X}$ are all isomorphic to $\mathbb{P}^1$, and therefore this intersection is irreducible of dimension $n+1$. Therefore C_a is closed and irreducible, and of dimension at most $n+1$. It contains $\bar{X}$ properly, and therefore $\dim C_a = n+1 < m$. Also,

$$C(X, a) \subseteq C_a,$$

and also the closure in F^m of $C(X, a)$ is contained in C_a. For $a, b \in \mathbb{P}^m \setminus \bar{X}$, we have

$$b \in C_a \Leftrightarrow a \in C_b.$$

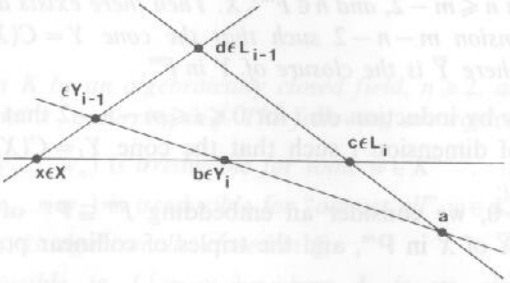
For any $a \in (\mathbb{P}^m \setminus C_h) \cap F^m$, we have

$$h \notin C_a \supseteq \overline{C(X, a)}.$$

The case $i = 0$ is proven.

For $i > 0$, by the induction hypothesis there exists a linear space L_{i-1} of dimension $i-1$ such that $Y_{i-1} = C(X, L_{i-1})$ has dimension $n+i \leqslant m-2$, and $h \notin \bar{Y}_{i-1}$.

Now we apply the case $i=0$, and find $a \in F^m \setminus \bar{Y}_{i-1}$ such that $Z = C(\bar{Y}_{i-1}, a)$ has dimension $n+i+1$ and $h \notin \bar{Z}$. Let $L_i = C(L_{i-1}, a)$ be the linear space spanned by L_{i-1} and a. Then $Y_i = C(X, L_i)$ has dimension $n+i+1$, and we now show that $\bar{Y}_i \subseteq \bar{Z}$. For any $b \in Y_i$, there exist $x \in X$ and $c \in L_i$ such that $b \in C(x, c)$, and $d \in L_{i-1}$ such that $c \in C(d, a)$. Then $C(x, d) \subseteq Y_{i-1}$, and the plane spanned by a, d, and x is contained in $\bar{Z}$. (The figure illustrates the case where $C(a, b)$ and $C(x, d)$ are not parallel, and thus $b \in Z$.) In particular, $b \in \bar{Z}$ and thus $\bar{Y}_i \subseteq \bar{Z}$, and $h \notin \bar{Y}_i$. (In fact, $\bar{Y}_i = \bar{Z}$.) ▌



The following is a variant of Lemma 2.3 in Strassen [45], appropriate to our context.

LEMMA 3.2. *Let F be a field, $m, n, s, t \in \mathbb{N}$, and the mapping*

$$\varphi = (\varphi_1, ..., \varphi_m) \colon F^n \to F^m$$

be given by polynomials $\varphi_1, ..., \varphi_m \in F[x_1, ..., x_n]$ of total degree at most t. If

$$\binom{s+m}{m} > \binom{st+n}{n},$$

then there exists $\tau \in F[y_1, ..., y_m] \setminus \{0\}$ with $\deg \tau \leqslant s$ and $\tau(\varphi_1, ..., \varphi_m) = 0$.

Proof. Consider the F-vector space

$$A = \{\tau \in F[y_1, ..., y_m] \colon \deg \tau \leqslant s\},$$

and the F-linear mapping

$$\varphi^* \colon A \to F[x_1, ..., x_n]$$
$$\tau \mapsto \tau(\varphi_1, ..., \varphi_m).$$

For $\tau \in A$ we have $\deg(\varphi^*(\tau)) \leqslant st$, and the image of φ^* is contained in the vector space

$$B = \{\sigma \in F[x_1, ..., x_n] \colon \deg \sigma \leqslant st\}.$$

Since

$$\binom{s+m}{m} = \dim A > \dim B = \binom{st+n}{n}$$

we have $\ker(\varphi^*) \neq \{0\}$. Any $\tau \in \ker(\varphi^*) \setminus \{0\}$ is sufficient. ∎

Lemma 3.3. *Let m, n, t, $\varphi\colon F^n \to F^m$ be as in Lemma 3.2, $h_1, \ldots, h_r \in F^m$ for some $r \geq 1$, assume that F is algebraically closed, that there exists $u \in F^n$ with $\varphi^{-1}(\varphi(u))$ finite, and that there exists τ with*

$$\begin{aligned} &\tau \in F[y_1, \ldots, y_m], \\ &\tau(\varphi_1, \ldots, \varphi_m) = 0, \\ \forall j \leq r, \quad &\tau(h_j) \neq 0. \end{aligned} \tag{$*$}$$

Then there exists τ with $(*)$ and $\deg \tau \leq m(t+1)^{m-1}$.

First proof. Let $X = \overline{\operatorname{im} \varphi}$ be the closure of $\operatorname{im} \varphi$. Since φ has a finite fiber, $\dim X = n$. Also,

$$\exists \tau \quad \text{with} \quad (*) \Leftrightarrow h_1, \ldots, h_r \notin X.$$

We first consider the case $r = 1$, and let $h = h_1$. Let $L \subseteq F^m \setminus X$ be a linear space of dimension $m - n - 2$ as in Lemma 3.1, with $Y = C(X, L)$ of dimension $m - 1$ and $h \notin \bar{Y}$. Furthermore, let $\lambda\colon F^{m-n-2} \to F^m$ be a linear map with image L, and

$$\psi = C(\varphi, \lambda)\colon F^{m-1} \to F^m.$$

ψ is given by polynomials $\psi_1, \ldots, \psi_m \in F[x_1, \ldots, x_{m-1}]$ of degree at most $t+1$, and there exists $\tau \in F[y_1, \ldots, y_m]$ such that

$$\tau(\psi_1, \ldots, \psi_m) = 0, \quad \tau(h) \neq 0.$$

Now let $s = m(t+1)^{m-1}$. Then

$$\begin{aligned} \binom{s+m}{m} \binom{s(t+1)+m-1}{m-1}^{-1} &= \frac{(s+1) \cdots (s+m)}{(s(t+1)+1) \cdots (s(t+1)+m-1) \cdot m} \\ &> \frac{s}{m} \left(\frac{1}{t+1} \right)^{m-1} = 1, \end{aligned}$$

since

$$\frac{s+j}{s(t+1)+j} \geq \frac{1}{t+1}$$

for all $j \geq 1$. Therefore

$$\binom{s+m}{m} > \binom{s(t+1)+m-1}{m-1},$$

and by Lemma 3.2 there exists $\tau_0 \in F[y_1,..., y_m] \setminus \{0\}$ with $\deg \tau_0 \leqslant s$ and $\tau_0(\psi_1,..., \psi_m) = 0$. Since $\bar{Y} = \overline{\operatorname{im} \psi}$ is an irreducible hypersurface and $h \notin \bar{Y}$, there exists some irreducible factor τ of τ_0 with $\tau(\psi_1,..., \psi_m) = 0$, and then $\tau(h) \neq 0$. This proves the case $r = 1$. The lemma now follows by induction on $r \geqslant 1$. For the inductive step, we may assume $\tau_1, \tau_2 \in F[y_1,..., y_m]$ such that $\tau_i(\varphi_1,..., \varphi_m) = 0$ and $\deg \tau_i \leqslant m(t+1)^{m-1}$ for $i = 1, 2$, and

$$\forall j < r, \quad \tau_1(h_j) \neq 0,$$
$$\tau_2(h_r) \neq 0.$$

Then there exists $u \in F$ such that $\tau = \tau_1 + u\tau_2$ satisfies $(*)$.

Second proof (with a slightly different bound). The graph of φ

$$G = \{(a, b) \in F^n \times F^m : \forall i \leqslant m, b_i - \varphi_i(a) = 0\} \subseteq F^n \times F^m$$

is closed and irreducible, and $\dim G = n$. By Strassen [46, Lemma 6.5], $\deg G \leqslant t^m$. We now use that taking projections or cones of varieties does not increase the degree [13, Chap. 1, Sect. 3, pp. 172–173]. Therefore $X = \overline{\operatorname{im} \varphi} = \overline{\operatorname{proj}_2(G)}$ has degree at most t^m, and X is the set of zeroes of some polynomials of degree at most t^m ([14, Proposition 3]; this fact can also be proved using cones as above). Some polynomial of degree at most t^m then vanishes on X but not at any of $h_1,..., h_r$. ∎

4. An Effective Hilbert Irreducibility Theorem

Hilbert's [16] irreducibility theorem asserts that for an irreducible polynomial $f \in \mathbb{Q}[x_1,..., x_n]$ there exists a substitution by integers for all but one variable such that the resulting univariate polynomial is irreducible. In this section, we prove an effective version of this theorem, and at the end of the section compare with previous results.

The approach is as follows: First we consider algebraically closed fields, so that we can apply Lemma 3.3. We prove existence of a "test polynomial" τ of small degree which separates the reducible bivariate polynomials (of degree at most d) from given irreducible polynomials $h_1,..., h_r$. Thus the vector of indeterminates of τ corresponds to the vector of coefficients of a bivariate polynomial with degree at most d, $\tau(h_j) \neq 0$ for $1 \leqslant j \leqslant r$, and $\tau(g) = 0$ for every reducible polynomial g. (Such a separation is in general only possible if the ground field is algebraically closed, and then clearly only makes sense for polynomials in at least two variables.)

Given a polynomial f in many variables of total degree at most d, we consider the substitution $x_i = u_i x_1 + v_i x_2 + w_i$ (for $i \geqslant 3$) as a mapping from the set of (u_i, v_i, w_i)'s to bivariate polynomials. If f is irreducible, then Bertini's theorem guarantees that some irreducible h is in the image of this mapping. Then τ as above separates the "unlucky" substitutions, under which f becomes reducible, from the lucky ones. In particular, f remains irreducible under "almost all" such substitutions. This is the required irreducibility theorem for algebraically closed fields. It is then easy to extend it to general fields. Actually, rather than just irreducibility, even the "factorization pattern" of a general polynomial is preserved under almost all substitutions of the above type. Let

$$\beta_d = \binom{d+2}{2}$$

and

$$X_d = \{g \in F[x, y]\colon \deg g \leqslant d\},$$
$$Y_d = \{g \in X_d\colon g \text{ is reducible or } g \in F\}.$$

We fix some isomorphism $X_d \to F^{\beta_d}$, so that $\tau(g) \in F$ for $\tau \in F[y_1, ..., y_{\beta_d}]$ and $g \in X_d$.

Lemma 4.1. *Let F be an algebraically closed field, $d, r \in \mathbb{N}$, and $h_1, ..., h_r \in X_d \setminus Y_d$. Then there exists $\tau \in F[y_1, ..., y_{\beta_d}]$ such that*

$$\forall g \in Y_d, \qquad \tau(g) = 0,$$
$$\forall j \leqslant r, \qquad \tau(h_j) \neq 0,$$
$$\deg \tau \leqslant \frac{d}{6} \beta_d 3^{\beta_d}.$$

Proof. Let $k = \lfloor d/2 \rfloor$, and for $1 \leqslant i \leqslant k$ let

$$\mu_i\colon X_i \times X_{d-i} \to X_d$$

be given by the multiplication of polynomials. Let $Z_i = \operatorname{im} \mu_i$. Then $Y_d = \bigcup_{1 \leqslant i \leqslant k} Z_i$. We first prove that each Z_i is closed. Consider the projectivization $\mathbb{P}X_d$ of X_d, which is a projective space of dimension $\beta_d - 1$, and similarly $\mathbb{P}X_i$ and $\mathbb{P}X_{d-i}$. $\mathbb{P}X_d$ can be viewed as the set of equivalence classes of nonzero homogeneous polynomials in $F[x, y, z]$ of degree d, where two polynomials are equivalent if one is a scalar multiple of the other. The mapping

$$\phi\colon X_d \setminus \{0\} \to \mathbb{P}X_d$$
$$f \mapsto \text{class of } z^d f\left(\frac{x}{z}, \frac{y}{z}\right)$$

is a bundle with fiber $F \setminus \{0\}$. Multiplication again gives a mapping

$$\mathbb{P}X_i \times \mathbb{P}X_{d-i} \to \mathbb{P}X_d.$$

The image Z of this mapping is closed [39, Chap. I, Sect. 5, Proposition] and therefore $Z_i \setminus \{0\} = \phi^{-1}(Z)$ is closed in $X_d \setminus \{0\}$. Since $0 \in Z_i$, Z_i is closed in X_d.

μ_i is given by quadratic forms $\mu_{i1}, ..., \mu_{i,\beta_d}$ in $\beta_i + \beta_{d-i}$ variables. Since $Z_i \subseteq Y_d$ is closed in X_d and $h_1, ..., h_r \notin Z_i$, there exists $\sigma_i \in F[y_1, ..., y_{\beta_d}]$ such that

$\sigma_i(\mu_{i1},\ldots,\mu_{i,\beta_d})=0$ and $\sigma_i(h_j)\neq 0$ for $1\leqslant j\leqslant r$. In order to apply Lemma 3.3, it remains to find a finite fiber.

Of course μ_i has no finite fiber, since $\mu_i(cf_1,(1/c)f_2)=\mu_i(f_1,f_2)$ for all $c\in F\setminus\{0\}$ and $(f_1,f_2)\in X_i\times X_{d-i}$. So we consider

$$X_i^*=\{f\in X_i\colon \text{the coefficient of } x^i \text{ in } f \text{ is } 1\},$$
$$\mu_i^*=\mu_i\restriction(X_i^*\times X_{d-i}).$$

$X_i^*\subseteq X_i$ is an affine linear subspace of dimension β_i-1. If $f_1\in X_i^*$ and $f_2\in X_{d-i}$ are irreducible and $\gcd(f_1,f_2)=1$ (such f_1,f_2 exist!), then

$$(\mu_i^*)^{-1}(\mu_i^*(f_1,f_2))=(\mu_i^*)^{-1}(f_1\cdot f_2)=\{(f_1,f_2)\},$$

and thus μ_i^* has some finite fibre. Clearly $\overline{\operatorname{im}\mu_i^*}\subset\overline{\operatorname{im}\mu_i}=\operatorname{im}\mu_i=Z_i$. In order to prove $\overline{\operatorname{im}\mu_i^*}=Z_i$, let $(f_1,f_2)\in X_i\times X_{d-i}$. If the coefficient c of x^i in f_1 is nonzero, then $(1/c)f_1\in X_i^*$ and $\mu_i^*((1/c)f_1,cf_2)=\mu_i(f_1,f_2)\in\operatorname{im}\mu_i^*$. If $c=0$, then consider for $u\in F\setminus\{0\}$,

$$g_1(u)=x^i+\frac{1}{u}f_1\in X_i^*,\qquad g_2(u)=uf_2\in X_{d-i},$$

and for $u\in F$,

$$g_0(u)=ux^if_2+f_1f_2.$$

If $u\neq 0$, then $g_0(u)=\mu_i^*(g_1(u),g_2(u))\in\operatorname{im}\mu_i^*$, and $W=\{g_0(u)\colon u\in F\setminus\{0\}\}\subseteq\operatorname{im}\mu_i^*$. Therefore

$$\mu_i(f_1,f_2)=g_0(0)\in\bar{W}\subseteq\overline{\operatorname{im}\mu_i^*}.$$

We now apply Lemma 3.3 to μ_i^*. Note that σ_i as above separates $\overline{\operatorname{im}\mu_i^*}$ from $h_1,\ldots,h_r$. It follows that there exists $\tau_i\in F[y_1,\ldots,y_{\beta_d}]$ of degree at most $\beta_d 3^{\beta_d-1}$ such that

$$\tau_i(\mu_{i1}^*,\ldots,\mu_{i,\beta_d}^*)=0,$$
$$\forall j\leqslant r,\qquad \tau_i(h_j)\neq 0.$$

Now $\tau=\tau_1\cdots\tau_k$ is sufficient. ▮

LEMMA 4.2. *Let $F\subseteq K$ be fields, $f\in K[x_1,\ldots,x_n]\setminus F[x_1,\ldots,x_n]$ of total degree d. Then there exists $\rho\in F[U_3,\ldots,U_n,V_3,\ldots,V_n,W_3,\ldots,W_n]\setminus\{0\}$ of degree at most d such that for all $t\in F^{3(n-2)}$ we have*

$$\rho(t)\neq 0\Rightarrow f\{t\}\notin F[x_1,x_2].$$

Proof. Write

$$f=\sum_{i\in\mathbb{N}^n} f_i x_1^{i_1}\cdots x_n^{i_n}$$

with $f_i \in K$. By assumption, $f_i \notin F$ for some $i \in \mathbb{N}^n$. Let

$$m = \max\{i_1 + \cdots + i_n : i \in \mathbb{N}^n \text{ and } f_i \notin F\},$$

$$m_1 = \min\{i_1 : \exists i_2, \ldots, i_n \in \mathbb{N} \text{ such that } i_1 + \cdots i_n = m \text{ and } f_i \notin F\},$$

$$I = \{i \in \mathbb{N}^n : i_1 + \cdots + i_n = m, i_1 = m_1, \text{ and } f_i \notin F\}.$$

For any $t \in F^{3(n-2)}$, the coefficient of $x_1^{m_1} x_2^{m-m_1}$ in

$$f\{t\} = \sum_{i \in \mathbb{N}^n} f_i x_1^{i_1} x_2^{i_2} (u_3 x_1 + v_3 x_2 + w_3)^{i_3} \cdots (u_n x_1 + v_n x_2 + w_n)^{i_n}$$

is

$$a(t) + \sum_{i \in I} f_i v_3^{i_3} \cdots v_n^{i_n}$$

for some $a(t) \in F$. We now consider K as a vector space over F. Let $j_1, \ldots, j_s \in I$ be such that

$$(f_{j_0}, f_{j_1}, \ldots, f_{j_s})$$

with $f_{j_0} = 1$ is a basis for the vector space

$$F + \sum_{i \in I} f_i F \subseteq K$$

over F. Note that $s \geqslant 1$, and, e.g., $f_{j_1} \notin F$. Then there exist $b_{ik} \in F$, for $i \in I$ and $0 \leqslant k \leqslant s$, such that

$$f_i = \sum_{0 \leqslant k \leqslant s} b_{ik} f_{j_k}$$

for all $i \in I$. Set

$$\rho = \sum_{i \in I} b_{i1} V_3^{i_3} \cdots V_n^{i_n} \in F[V_3, \ldots, V_n].$$

Then $\rho \neq 0$, and for any $t \in F^{3(n-2)}$ we have

$$\begin{aligned} f\{t\} \in F[x_1, x_2] &\Rightarrow \sum_{0 \leqslant k \leqslant s} \left(\sum_{i \in I} b_{ik} v_3^{i_3} \cdots v_n^{i_n} \right) f_{j_k} \\ &= \sum_{i \in I} \left(\sum_{0 \leqslant k \leqslant s} b_{ik} f_{j_k} \right) v_3^{i_3} \cdots v_n^{i_n} = \sum_{i \in I} f_i v_3^{i_3} \cdots v_n^{i_n} \in F \\ &\Rightarrow \forall k, 1 \leqslant k \leqslant s, \sum_{i \in I} b_{ik} v_3^{i_3} \cdots v_n^{i_n} = 0 \\ &\Rightarrow \rho(t) = 0. \quad ▮ \end{aligned}$$

LEMMA 4.3. *Let F be an arbitrary field, $n \geqslant 1$, and $f \in F[x_1,..., x_n]$ irreducible of total degree d. Then there exists a field K containing F and a nonzero polynomial*

$$\sigma \in K[U_3,..., U_n, V_3,..., V_n, W_3,..., W_n]$$

of total degree less than $(d^3/6)\,\beta_d\, 3^{\beta_d} + d2^d$ such that for all $t = (u, v, w) \in F^{3(n-2)}$ with $\sigma(t) \neq 0$ the polynomial

$$f\{t\} = f(x_1, x_2, u_3x_1 + v_3x_2 + w_3,..., u_nx_1 + v_nx_2 + w_n) \in F[x_1, x_2]$$

is irreducible.

Proof. Let $\bar{F}$ be an algebraic closure of F, and $f = f_1 \cdots f_r$ the absolute factorization of f, with $f_i \in \bar{F}[x_1,..., x_n]$ irreducible. (The irreducible polynomials f_i need not be pairwise distinct; if char $F = p > 0$, then f_i may occur p^e times among $f_1,..., f_r$ for some $e \geqslant 0$.)

First note that if L is any field containing $\bar{F}$ such that $x_1,..., x_n$ are algebraically independent over L, then each f_i is irreducible in $L[x_1,..., x_n]$. Applying Theorem 2.2 repeatedly, we find algebraically closed fields $\bar{F} \subseteq L_1 \subseteq L_2 \subseteq \cdots \subseteq L_r$ and $t_i \in L_i^{3(n-2)}$ such that $f_i\{t_i\} \in L_i[x_1, x_2]$ is irreducible. Let $K = L_r$, fix some i, $1 \leqslant i \leqslant r$, and consider the mapping

$$\phi_i\colon K^{3(n-2)} \to X_d$$
$$t = (u, v, w) \mapsto f_i\{t\}$$

where $X_d \cong K^{\beta_d}$. By the above, $\phi_i(t_i) \notin Y_d \subseteq X_d$. By Lemma 4.1, there exists a polynomial τ_i in β_d variables of total degree at most $(d/6)\,\beta_d 3^{\beta_d}$ such that

$$\forall g \in Y_d \quad \tau_i(g) = 0, \qquad \text{and} \qquad \tau_i(\phi_i(t_i)) \neq 0 \qquad \text{for} \quad 1 \leqslant i \leqslant r.$$

Since ϕ_i is given by polynomials of total degree at most d in the u_j, v_j, w_j, we have that

$$\pi_i = \tau_i \circ \phi_i \in K[U_3,..., U_n, V_3,..., V_n, W_3,..., W_n] = R$$

is a polynomial of degree at most $(d^2/6)\,\beta_d 3^{\beta_d}$, and $\pi_i(t_i) \neq 0$ for all i. With $\pi = \pi_1 \cdots \pi_r$,

$$\forall t \in K^{3(n-2)} \forall i \quad \pi(t) \neq 0 \Rightarrow f_i\{t\} \text{ is irreducible.}$$

If $r = 1$, then $\sigma = \pi \in R$ is sufficient for the lemma. Otherwise, for each $I \subseteq \{1,..., r\} = S$ with $I \notin \{\varnothing, S\}$ note that $f_I = \prod_{i \in I} f_i \in K[x_1,..., x_n] \setminus F[x_1,..., x_n]$. For any such I, let $\rho_I \in R$ be the polynomial from Lemma 4.2 for f_I, and

$$\sigma = \pi \cdot \prod_{\substack{I \subseteq S \\ I \neq \varnothing, S}} \rho_I \in R.$$

Let $t \in F^{3(n-2)}$ be a substitution such that $\sigma(t) \neq 0$. Then each $f_i\{t\}$ is irreducible, and

$$f\{t\} = f_1\{t\} \cdots f_r\{t\}$$

is an irreducible factorization of $f\{t\}$ in $K[x_1, x_2]$. Any factor $g \in K[x_1, x_2]$ of $f\{t\}$ is therefore (up to a scalar multiple) of the form

$$g = \prod_{i \in I} f_i\{t\}$$

for some $I \subseteq \{1, ..., r\}$. If g is a nontrivial factor, then $\rho_I(t) \neq 0$ and hence $g \notin F[x_1, x_2]$. Thus $f\{t\}$ is irreducible. The degree of σ is less than $(d^3/6)\, \beta_d 3^{\beta_d} + d2^d$. ∎

Definition 4.4. Let F be a field, $n, r \geqslant 1$, $f, f_1, ..., f_r \in F[x_1, ..., x_n]$, and $e_1, ..., e_r \geqslant 1$ such that f_i is irreducible of total degree d_i for each i, and

$$f = f_1^{e_1} \cdots f_r^{e_r},$$

$$\gcd(f_i, f_j) = 1 \quad \text{for} \quad 1 \leqslant i < j \leqslant r.$$

Then $(d_1, e_1; ...; d_r, e_r)$ is called a *factorization pattern* for f.

The factorization pattern is unique up to certain permutations. We can make it unique by stipulating, e.g., that for all i, j, $1 \leqslant i < j \leqslant r$,

$$d_i \leqslant d_j \quad \text{and} \quad (d_i = d_j \Rightarrow e_i \leqslant e_j).$$

We will implicitly assume some such normalization and speak of *the* factorization pattern of f. It is now easy to show that rather than just irreducibility, the complete factorization pattern is preserved under random substitutions.

Theorem 4.5. *Let F be an arbitrary field, $n \geqslant 1$ and $f \in F[x_1, ..., x_n]$ of total degree d. Then there exists a field K containing F and a nonzero polynomial*

$$\tau \in K[U_3, ..., U_n, V_3, ..., V_n, W_3, ..., W_n] = R$$

of total degree at most 9^{d^2} such that for all $t = (u, v, w) \in F^{3(n-2)}$ with $\tau(t) \neq 0$ the polynomial

$$f\{t\} = f(x_1, x_2, u_3 x_1 + v_3 x_2 + w_3, ..., u_n x_1 + v_n x_2 + w_n) \in F[x_1, x_2]$$

has the same factorization pattern as f.

Proof. Let

$$f = f_1^{e_1} \cdots f_r^{e_r}$$

be the irreducible factorization of f in $F[x_1,..., x_n]$, with $\gcd(f_i, f_j)=1$ for $i \neq j$. (This is unique up to permutations and scalar multiples.) For $1 \leqslant i \leqslant r$, let K_i and σ_i be as in Lemma 4.3 (with f_i for f). We can assume that $K_i \subseteq K$ for some field K and all i, and then $\sigma_i \in R$. Let $\sigma = \prod_{1 \leqslant i \leqslant r} \sigma_i$. Then for any $t \in F^{3(n-2)}$ with $\sigma(t) \neq 0$ each $f_i\{t\}$ is irreducible, and the degree of σ is less than $(d^4/6)\, \beta_d 3^{\beta_d} + d^2 2^d$.

For any i, $1 \leqslant i \leqslant r$, let $h_{i0} = F[x_1,..., x_n]$ be the homogeneous part of f_i of highest degree $d_i = \deg f_i$. Then

$$\begin{aligned} h_{i1} &= h_{i0}(x_1, x_2, U_3 x_1 + V_3 x_2,..., U_n x_1 + V_n x_2) \\ &\in F[x_1, x_2][U_3,..., U_n, V_3,..., V_n] \end{aligned}$$

is a nonzero polynomial of degree at most d in $U_3,..., V_n$ over $F[x_1, x_2]$. Let $\pi_i \in R$ be the coefficient of the lexicographically highest term of h_{i1} in x_1 and x_2. Then

$$\forall i\, \forall t = (u, v, w) \in F^{3(n-2)} \quad \pi_i(u, v) \neq 0 \Rightarrow \deg(f_i\{t\}) = d_i.$$

Set $\pi = \pi_1 \cdots \pi_r$. Now for any i, $1 \leqslant i \leqslant r$, write

$$\begin{aligned} f_i &= \sum_{e \in \mathbb{N}^2} f_{ie} x_1^{e_1} x_2^{e_2}, \\ f_i^* &= f_i(x_1, x_2, U_3 x_1 + V_3 x_2 + W_3,..., U_n x_1 + V_n x_2 + W_n) \\ &= \sum_{e \in \mathbb{N}^2} f_{ie}^* x_1^{e_1} x_2^{e_2} \in R[x_1, x_2] \end{aligned}$$

with $f_{ie} \in F[x_3,..., x_n]$, and $f_{ie}^* \in R$.

Fix some i, j with $1 \leqslant i < j \leqslant r$. We now provide a condition that guarantees $\gcd(f_i\{t\}, f_j\{t\}) = 1$. We know that $\gcd(f_i, f_j) = 1$, or, equivalently, f_i is not a scalar multiple of f_j. This implies that there exist $a, b \in \mathbb{N}^2$ such that

$$\delta = \det \begin{pmatrix} f_{ia} & f_{ib} \\ f_{ja} & f_{jb} \end{pmatrix} \neq 0.$$

Set

$$\delta^* = \det \begin{pmatrix} f_{ia}^* & f_{ib}^* \\ f_{ja}^* & f_{jb}^* \end{pmatrix} \in R.$$

Since each f_{ie}^*, f_{je}^* has total degree at most d, δ^* has total degree at most $2d$. Under the substitution ψ with $\psi(U_i) = \psi(V_i) = 0$ and $\psi(W_i) = x_i$ for all i, $3 \leqslant i \leqslant n$, we have $\psi(\delta^*) = \delta$. It follows that δ^* is nonzero. For any $t \in F^{3(n-2)}$ with $\delta^*(t) \neq 0, f_i\{t\}$ is not a scalar multiple of $f_j\{t\}$. Thus if they are both irreducible, $\gcd(f_i\{t\}, f_j\{t\}) = 1$.

Now for each $i < j$ as above, we take the $\delta^* \in R$ as constructed, and let $\rho \in R$ be the product of all the δ^*. Since $r \leqslant d$, we have at most $\binom{d}{2} \leqslant d^2/2$ such δ^*, and ρ has degree at most d^3.

With $\tau = \sigma\pi\rho \in R$, the condition $\tau(t) \neq 0$ guarantees that $f\{t\}$ has the same factorization pattern as f. Also,

$$\deg \tau = \deg \sigma + \deg \pi + \deg \rho \leqslant \frac{d^4}{6}\, \beta_d 3^{\beta_d} + d^2 2^d + d + d^3 < 9^{d^2}. \quad \blacksquare$$

Remark 4.6. We want to compare Lemma 4.3 with the number-theoretic "Hilbert irreducibility theorems" to be found in the literature. A rather general version is given in Lang [28]; it states that over certain fields F, for any irreducible polynomial $f \in F[x_1,..., x_n]$ and almost all $a_2,..., a_n \in F$, the polynomial $f(x_1, a_2,..., a_n) \in F[x_1]$ is irreducible. The fields F for which this holds are called "Hilbertian fields," and include, e.g., all algebraic number fields, but exclude—by obvious counterexamples—the finite fields and algebraically closed fields. "Almost all" then is in the sense of a Lebesgue measure.

This is a much weaker sense than the algebro-geometric "almost all" that we use throughout this paper. It means that there exists a nonzero test polynomial τ such that the required property (here: preserving irreducibility) holds for any argument t whenever $\tau(t) \neq 0$. Given a bound on the degree of τ, we obtain a probabilistic algorithm via Fact 4.7 below.

The previous Hilbert irreducibility theorems did not lead to algorithms, since they failed to provide effectively (deterministically or probabilistically) irreducibility-preserving substitutions. Zippel's [53] sparse factoring algorithm was based on the unproved assumption that an effective Hilbert irreducibility theorem holds over $\mathbb{Q}$ for simple substitutions.

However, a number-theoretic result by Sprindzhuk [42] may lead to an effective version. The ultimate goal here would be a deterministic polynomial-time factorization procedure for sparse multivariate polynomials. Although only valid over $\mathbb{Q}$ (or more generally, Hilbertian fields), the number-theoretic irreducibility theorems have the two advantages of only using simple substitutions of constants for variables, and of reducing to univariate polynomials. Any method valid also over algebraically closed fields cannot have either of these advantages (see end of Sect. 2).

In retrospect, the results of Heintz and Sieveking [15] and Kaltofen [18, 19] can be used to obtain effective Hilbert irreducibility theorems. Kaltofen [20] exhibits an elementary proof for a result similar to the present one. It essentially replaces the 9^{d^2} in Theorem 4.5 by 2^d, and is valid at least in characteristic zero and over finite fields. Theorem 4.5 leads to probabilistic algorithms via the following fact.

FACT 4.7 [38, Corollary 1]. *Let $\tau \in F[y_1,..., y_m]$ have total degree at most k, and $A \subseteq F$ finite with a elements. Then*

$$\#\{u \in A^m : \tau(u) = 0\} \leqslant k a^{m-1}.$$

For randomly chosen $u \in A^m$ (with respect to the uniform distribution) we have $\operatorname{Prob}(\tau(u) = 0) \leqslant k/a$.

5. Reduction to Bivariate Polynomials

In this section, we present the model of computation to be used, and phrase Strassen's method [44] of avoiding divisions so that we obtain an effective version, suitable for our framework.

As mentioned in the Introduction, we use the notion of a computation (or "straight-line program") over $F \cup \{x_1, ..., x_n\} \cup \{+, -, *, /\}$, which is formally defined in Strassen [43]. Such a computation is a sequence $((\tau_1, \lambda_1), ..., (\tau_s, \lambda_s))$. Each τ_i is an operation, either $\tau_i \in \{+, -, *, /\}$ and then $\lambda_i = (k, l)$ with $1 \leqslant k, l < i$, or $\tau_i \in F \cup \{x_1, ..., x_n\}$ and then $\lambda_i = \varnothing$. We call s the size of α, and there also a natural notion of depth (= parallel time) of α. There are rational functions $f_1, ..., f_s \in F(x_1, ..., x_n)$ associated in a natural way with α, and α computes f_s; in fact, any subset of $\{f_1, ..., f_s\}$. We assume that no division by (the rational function) zero is attempted. Throughout this paper, we assume that $n \leqslant s$; this is satisfied, e.g., if all variables occur in α.

Each such α can be encoded by an $\bar{\alpha} = (\gamma, \beta) \in F^* \times \{0, 1\}^*$ as follows: $\beta = (\beta_1, ..., \beta_t)$ encodes s, n, each τ_i (with a special symbol for those $\tau_i \in F$) and λ_i. We can achieve this with $t = O(s \log s)$. The vector $\gamma = (\gamma_1, ..., \gamma_s) \in F^s$ of constants has

$$\gamma_i = \begin{cases} \tau_i & \text{if } \tau_i \in F, \\ 0 & \text{otherwise.} \end{cases}$$

Based on the results of the previous section, we now present a probabilistic polynomial-time reduction for computing the factorization pattern from multivariate to bivariate polynomials. The only restriction—that the ground field be large enough—will be removed in Section 7. Since the input is a computation we can view the algorithm as a "compilation" which produces another computation, namely for a bivariate polynomial. Apart from arithmetic operations in F, the algorithm uses tests "$a = 0$?" in F, random choices from a finite subset of F, and Boolean operations.

Algorithm FACTORIZATION PATTERN.

Input: An encoding $\bar{\alpha}$ of a computation as above of a polynomial $f \in F[x_1, ..., x_n]$, and a finite set $A \subseteq F$.

Output: Either the encoding $\overline{\alpha\{t\}}$ of a computation $\alpha\{t\}$ for a bivariate polynomial $g = f\{t\} \in F[x_1, x_2]$, or "failure."

1. Choose $t = (u, v, w) \in A^{3(n-2)}$ at random.

2. Compute the description $\overline{\alpha\{t\}}$ of a computation $\alpha\{t\}$ as follows. The first $7n$ steps are such that $f_1 = 0$ and $f_{6n+j} = u_j x_1 + v_j x_2 + w_j$ for $3 \leqslant j \leqslant n$ and the intermediate results f_i. For $1 \leqslant i \leqslant s$, we have the step $(\tau^*_{7n+i}, \lambda^*_{7n+i})$ with

$$\tau^*_{7n+i} = \begin{cases} + & \text{if } \tau_i \in \{x_3, \dots, x_n\}, \\ \tau_i & \text{otherwise,} \end{cases}$$

$$\lambda^*_{7n+i} = \begin{cases} (7n+k, 7n+l) & \text{if } \lambda_i = (k, l) \neq \varnothing, \\ (6n+j, 1) & \text{if } \tau_i = x_j, 3 \leqslant j \leqslant n, \\ \lambda_i & \text{otherwise.} \end{cases}$$

3. Choose $b = (b_1, b_2) \in A^2$ at random, and execute $\alpha\{t\}$ with input b_i for x_i. If a division by zero occurs, then return "failure".

THEOREM 5.1. *Let α be a computation of size s for a polynomial $f \in F[x_1, \dots, x_n]$ of degree d, and $A \subseteq F$ finite with $a = \# A$. On input α and A, FACTORIZATION PATTERN returns in $O(s \log s)$ steps either "failure" or a description $\alpha\{t\}$ for a computation of size at most $8s$ for a bivariate polynomial $g \in F[x_1, x_2]$ of degree at most d. $\alpha\{t\}$ uses only constants from $\gamma \cup A$, where γ is the set of constants used by α. With probability greater than $1 - (9^{d^2} + 2^s)/a$, "failure" does not occur and g and f have the same factorization pattern. In particular, with this probability g is irreducible if and only if f is irreducible.*

Proof. By Theorem 4.5 and Fact 4.7, f and g have different factorization pattern with probability at most $9^{d^2}/a$. We estimate the probability of failure in step 3. The purpose of this step is to ensure that $\alpha\{t\}$ really is a computation in our sense, i.e., that no division $f_i = f_k/f_l$ with $f_l\{t\} = 0$ occurs. If failure occurs at a division step $f_i = f_k/f_l$, then

$$f_l\{t\}(b) = 0.$$

One easily proves by induction on i that there exist polynomials $p_i, q_i \in F[x_1, \dots, x_n]$ of degree at most 2^{i-1} such that $f_i = p_i/q_i$ [25]. Then $p_l \neq 0$, since α is a computation, and we can assume that $q_l\{t\}(b) \neq 0$, since otherwise failure has occurred at an earlier step. Then

$$\begin{aligned} \operatorname{Prob}(f_l\{t\}(b) = 0) &= \operatorname{Prob}(p_l\{t\} = 0) + \operatorname{Prob}(p_l\{t\} \neq 0 \text{ and } p_l\{t\}(b) = 0) \\ &\leqslant \deg p_l/a + \deg p_l\{t\}/a \leqslant 2^l/a. \end{aligned}$$

(We use the fact that $p_l \in F[x_1, \dots, x_n]$ gives rise to a nonzero polynomial in $F[x_1, x_2, U_3, \dots, W_n]$.) Therefore, the probability that failure occurs in step 3 or the factorization pattern of g is different from that of f is less than

$$\left(9^{d^2} + \sum_{1 \leqslant l < s} \cdot\, 2^l\right) \Big/ a < (9^{d^2} + 2^s)/a. \quad \blacksquare$$

Thus, e.g., if $a \geqslant 2^d/(9^{d^2} + 2^s)$, then we have a probabilistic polynomial-time reduction from multivariate to bivariate factorization pattern, with error probability less than 2^{-d}. Producing a random element from A may seem a powerful step, and one might want to assume that such a random generation takes $O(\log a)$ "random bit choices," so that $O(d^2+s)$ such basic choices are required. For convenience, we say that A is part of the input. In fact, we only need a procedure to generate random elements of A. In characteristic zero, we will often have $A=\{1,..., a\}$, so that the binary representation of a suffices to specify A. In the sequel, we always consider A to contribute $\log(\# A)$ to the input size.

The algorithm FACTORIZATION PATTERN is the basic computational result of this paper. In the remainder, we discuss improvements and applications to special cases. We first describe a variant—to be used below—of the algorithm that employs Strassen's [44] method of avoiding division. The algorithm as above returns an encoding of a computation $\alpha\{t\}$ for $g=f\{t\}\in F[x_1, x_2]$, on which we would then run a bivariate algorithm for factorization pattern. If we want to supply the bivariate algorithm with a list of coefficients of g—which is usually required in such algorithms—rather than just a computation, then there are (at least) three probabilistic ways of achieving this:

1. Run $\alpha\{t\}$ on d^2 appropriately chosen values (a_1, a_2), and use interpolation. Here $d=\deg f$, and it is assumed that no division by zero occurs in $\alpha\{t\}$ for these values.

2. Use Strassen's method to make $\alpha\{t\}$ division-free, and then compute all homogeneous parts (or even coefficients) of the bivariate intermediate results separately.

3. Use Strassen's method to make α division-free, and then compute all homogeneous parts of the intermediate results separately.

The last possibility gives rise to the following algorithm; see also [5, Remark 1].

Algorithm DIVISION-FREE CONVERSION.

Input: A computation α for a polynomial $f\in F[x_1,..., x_n]$ of degree d, and a finite set $A\subseteq F$.

Output: Either "failure," or another computation α^* for f.

1. Choose $b=(b_1,..., b_n)\in A^n$ at random.

2. Execute α on input b. If a division by 0 occurs, return "failure" and stop.

3. (Comment: Now for every division $f_i=f_k/f_l$ in α, we have $c_l=f_l(b_1,..., b_n)\neq 0$, and

$$\begin{aligned} f_i&=f_k/(c_l(1-g)) \\ &\equiv (f_k/c_l)(1+g+g^2+\cdots+g^d) \bmod (x_1-b_1,..., x_n-b_n)^{d+1}, \end{aligned}$$

where $g = 1 - f_l/c_l \in F[x_1,..., x_n]$ with $g(b_1,..., b_n) = 0$. Each intermediate result can be written as

$$f_i = \sum_{0 \leqslant j} \Bigg(\sum_{\substack{e \in \mathbb{N}^n \\ e_1 + \cdots + e_n = j}} f_{ije}(x_1 - b_1)^{e_1} \cdots (x_n - b_n)^{e_n} \Bigg),$$

with $f_{ije} \in F$. The jth summand is the homogeneous part of degree j of f_i with respect to $x_1 - b_1,..., x_n - b_n$; i.e., in the Taylor expansion of f_i around b.)

We have a computation α' for f consisting of three phases. The first is the calculation of $c_l^{-1} = f_l(b)^{-1} \in F$ for each divisor f_l. For the second phase, we replace each operation in α by computations for each homogeneous part of degree at most d with respect to $x_1 - b_1,..., x_n - b_n$. This is clear for constants, and +, *. An input x_i has two homogeneous parts b_i and $x_i - b_i$ of degree 0 and 1, respectively. For a division, we use the formula above, calculating each homogeneous part of each g^j separately. In the third phase, we add the $d+1$ homogeneous parts for the final result.

4. (We now have a division-free computation α' which is homogeneous with respect to the $x_i - b_i$.) Transform α' into a division-free computation α^* in which the first two phases are homogeneous with respect to $x_1,..., x_n$.

PROPOSITION 5.2 (Strassen). *Consider a description $\bar{\alpha} = (\gamma, \beta)$ of a computation α of size s for a polynomial $f \in F[x_1,..., x_n]$ of degree d, and a finite set $A \subseteq F$ with $a = \# A$ as input to DIVISION-FREE CONVERSION. As output, the algorithm produces either "failure" or the description $\overline{\alpha^*} = (\gamma^*, \beta^*)$ of another computation α^* of size $O(sd \log^2 d)$ for f. α^* has three phases: the first phase involves only constants from $\{\gamma_1,..., \gamma_s\} \cup A$ (and no inputs x_i), the second phase is a homogeneous division-free computation, and the third phase consists of $d+1$ additions. The conversion procedure can be performed with $O(sd \log(sd) \log^2 d)$ bit operations and $O(n \log a)$ random bit choices. The probability of failure is at most $2^s/a$.*

The following algorithm is now obvious.

ALGORITHM DIVISION-FREE FACTORIZATION PATTERN.

Input: A computation α for a polynomial $f \in F[x_1,..., x_n]$ of degree d, where α is division-free and consists of three phases as α^* in Proposition 5.2, and a finite set $A \subseteq F$ with $a = \# A$.

Output: Either "failure," or computations for each coefficient of a bivariate polynomial $g \in F[x_1, x_2]$.

1. Choose $t = (u, v, w) \in A^{3(n-2)}$ at random.

2. Replace x_j by $u_j x_1 + v_j x_2 + w_j$ for $3 \leqslant j \leqslant n$. Replace each (homogeneous) operation in phase two of α by operations computing each of the coefficients of the corresponding bivariate polynomial. Skip phase three.

3. Return the computations for the coefficients of $g = f\{t\} \in F[x_1, x_2]$.

Proposition 5.3. *If α has size s, then DIVISION-FREE FACTORIZATION PATTERN can be executed in $O(sd^2 \log^4 d \log(sd))$ bit operations and $O(n \log a)$ random bit choices. The computation for each coefficient of g has size $O(sd^2 \log^4 d)$. With probability greater than $1 - 9^{d^2}/a$, f and g have the same factorization pattern.*

Remark 5.4. The conversion to a division-free computation assumes that we have an upper bound d on the degree of the result f. We can obtain a probabilistic estimate d^* for d as follows. We choose $u_0, u_1, \ldots, u_n$ randomly from a large finite subset of F, substitute $u_i x_1$ for x_i $(i \geq 2)$, and compute the result $r \in F$ for $x_1 = u_0$. For $e = 1, 2, \ldots$, we perform DIVISION-FREE CONVERSION for this computation α^*, expanding around $x_1 - u_1$ and truncating modulo $(x_1 - u_1)^{e+1}$. We then execute the resulting univariate computation for $x_1 = u_0$ to obtain a result $r_e \in F$, and let d^* be the first value of e for which $r_e = r$. Then $d^* \leq d$. Furthermore, $d = d^*$ with probability at least $1 - 3 \cdot 2^s/a$. (The reduction to a univariate computation is actually not necessary, but simplifies the procedure.)

Can we also obtain a "fast parallel version" of the reduction? "Fast parallel" should mean depth (= parallel time) polynomial in log(input size), with simultaneously polynomial size. Unfortunately, this seems in general impossible for both the conversion algorithm and the converted computation $\alpha\{t\}$. If z is an indeterminate over an infinite field F, then any computation of z^{2^m}, given z as input, takes parallel time $\Omega(m)$ [25]. Thus, e.g., if $\mathbb{Q} \subseteq F$ and $z \in F$ is transcendental over $\mathbb{Q}$, α might have z as input and compute z^{2^m} for $m = s/2$. In this case, the validity test in step 3 of FACTORIZATION PATTERN, the converted algorithm $\alpha\{t\}$ and the constant phase of α^* (as in Proposition 5.2) all take parallel time $\Omega(s)$.

However, it is quite reasonable to consider computations that use polynomial time to compile their constants, and then poly-log depth to perform the computation depending on the inputs. By the general parallelization method of [47], the second and third phase of α^* can be performed in depth $O(\log s \log(sd))$ and size $O((sd \log^2 d)^3)$.

Proposition 5.5. *Let α be a division-free computation of size s and depth r for a polynomial $f \in F[x_1, \ldots, x_n]$ of degree d. There is a probabilistic algorithm that outputs either "failure" or a list of the coefficients of a bivariate polynomial g. With probability greater than $1 - 2^{-sd}$, "failure" does not occur and g and f have the same factorization pattern. The algorithm can be performed in depth $O(r \log^4 d)$ and size $O(s^2 d^6)$.*

Remark 5.6. The present method also allows us to obtain a different type of "factorization pattern"

$$(d_{11}, \ldots, d_{1n}, e_1; \ldots; d_{r1}, \ldots, d_{rn}, e_r),$$

where $f = f_1^{e_1} \cdots f_r^{e_r}$ is as in Definition 4.4, and d_{ij} is the degree $\deg_{x_j} f_i$ of f_i in x_j. Kaltofen [23] first showed how to compute this pattern.

To apply our methods, we let $n \geqslant 4$, $A \subseteq F$ be finite and large, $t \in A^{3(n-2)}$ be a substitution, and we assume that $g = f\{t\}$ has the same factorization pattern (Definition 4.4) as f. We compute a factorization $g = g_1^{e_1} \cdots g_r^{e_r}$ of g. We consider the unique factorization of f as above such that $f_i\{t\}$ is a scalar multiple of g_i. Also, let $z_j = u_j x_1 + v_j x_2 + w_j \in F[x_1, x_2]$. For $3 \leqslant j \leqslant n$, we compute a factorization of

$$h_j = f(x_1, x_2, z_3, ..., z_{j-1}, x_j, z_{j+1}, ..., z_n) \in F[x_1, x_2, x_j].$$

Since $h_j(x_1, x_2, z_j) = g$, each irreducible factor of h_j becomes a scalar multiple of a unique g_i under the substitution $x_j = z_j$. This sets up a bijection between the irreducible factors $\{h_{1j}, ..., h_{rj}\}$ of h_j and $\{g_1, ..., g_r\}$ with $h_{ij}(x_1, x_2, z_j) = g_i$, and with high probability we have

$$d_{ij} = \deg_{x_j} f_i = \deg_{x_j} h_{ij}.$$

In order to calculate d_{i1}, we assume that

$$M = \begin{pmatrix} u_3 & v_3 \\ u_4 & v_4 \end{pmatrix} \in F^{2 \times 2}$$

is invertible. For randomly chosen u_j and v_j, this happens with probability at least $1 - 2/a$ by Fact 4.7, where $a = \# A$. Let

$$y_j = w_j + (u_j, v_j) \cdot M^{-1} \cdot \begin{pmatrix} x_3 - w_3 \\ x_4 - w_4 \end{pmatrix} \in F[x_3, x_4]$$

for $5 \leqslant j \leqslant n$, and

$$y_2 = (0, 1) \cdot M^{-1} \cdot \begin{pmatrix} x_3 - w_3 \\ x_4 - w_4 \end{pmatrix}.$$

We compute a factorization of

$$h_1 = f(x_1, y_2, x_3, x_4, y_5, ..., y_n) \in F[x_1, x_3, x_4].$$

Then $h_1(x_1, z_3, z_4) = g$, and again this substitution provides a bijection between the irreducible factors of h_1 and those of g. If h_{i1} corresponds to g_i under this bijection, then with high probability

$$d_{i1} = \deg_{x_1} f_i = \deg_{x_1} h_{i1}.$$

Similarly, we obtain $d_{12}, ..., d_{r2}$.

We note a difference between the algorithm DIVISION-FREE FACTORIZATION PATTERN and the one of this remark. The former is a probabilistic reduction from multivariate to bivariate polynomials for the problem of computing the factorization pattern, but for the latter, we actually have to factor trivariate polynomials. Given the factorization of g, it is easy to compute the factorizations of the trivariate polynomials by a (dense) Hensel lifting. On the other hand, the known methods even for testing bivariate polynomials for irreducibility, say over $\mathbb{Q}$ or a finite field, all require to factor some (at least univariate) polynomial.

6. Algebraic Number Fields

Proposition 5.3 provides an efficient random computation for the factorization pattern of multivariate polynomials over those fields where the factorization pattern of bivariate polynomials can be computed in polynomial time. Such computations usually make use of a factorization procedure, at least for univariate polynomials. The prime examples are the prime fields $\mathbb{Q}$ [32, 22] and $\mathbb{Z}_p$ [3, 11, 29]; [6] deals with the general case of fields finitely generated over their prime fields. We now consider algebraic number fields—where the most interesting case is the field of the rational numbers—and defer the case of finite fields to the next section. As an auxiliary result, Corollary 6.9 presents a probabilistic polynomial-time simulation of computations by Boolean circuits.

So let F be a number field, presented as $F=\mathbb{Q}[z]/(h)$ with $h\in\mathbb{Q}[z]$ irreducible of degree m. We assume throughout this section that h has integral coefficients and is monic. It is easy to convert the general case to this special situation. A *standard representation* of an element $b\in F$ (with respect to the given minimal polynomial h) consists of the binary representations of $r_0,..., r_m\in\mathbb{Z}$, where

$$b=\frac{1}{r_m}\sum_{0\leqslant j<m} r_j y^j,$$

$r_m\neq 0$, and $y=z \bmod h$ is a generator for F over $\mathbb{Q}$.

Now let α be a computation over $F\cup\{x_1,..., x_n\}$, computing a polynomial $f\in F[x_1,..., x_n]$ of degree d. FACTORIZATION PATTERN applies to F in a straightforward way. However, we want to apply a bivariate factorization algorithm to the resulting $g\in F[x_1, x_2]$. Such algorithms require g to be given by a list of coefficients, so that we have to perform DIVISION-FREE CONVERSION. Step 2 of that algorithm evaluates α at a specific input. We first have to present a probabilistic polynomial-time algorithm for this evaluation. Rather than counting arithmetic operations in F, it is now more relevant to count bit operations.

In terms of arithmetic operations, the (sequential) complexity of evaluating a polynomial is well studied. When we count bit operations (say, on a Turing machine or a Boolean circuit), it is surprising that no polynomial-time algorithm is known to evaluate a polynomial over $\mathbb{Q}$ given by a computation. This problem is non-trivial even for specific polynomials like the determinant, if we consider it as given by a program for Gaussian elimination; Edmonds [7] gave a solution in this case.

We want the number of bit operations to be polynomial in the input plus output size, i.e., the lengths of representations of the computations, the input values, and the output value. The problem is that intermediate results may have more than polynomial length. This is illustrated by the trivial example of a computation of length s, using the constant $f_1 = 2$, computing $f_{s-1} = 2^{2^{s-1}}$ and having $f_s = f_{s-1}/f_{s-1} = 1$ as output, independent of the input. We will use the rather obvious approach of computing modulo a prime p. Now the problem is that one might have a similar example of a computation as above, but with f_{s-1} being divisible by "all small primes," so that the last division step fails modulo p. (See [40] for a computation of $(2^s)!$ in size $O(s)$, using division with remainder.) However, we obtain a probabilistic polynomial-time algorithm for evaluating a computation for a polynomial, by computing modulo a randomly chosen large prime.

Definition 6.1. If α is a computation over F, then in a *standard representation* $\bar{\alpha} = (\gamma, \beta)$ of α every constant $\gamma_i \in F$ has to be given in standard representation. The length $l(\bar{\alpha})$ is the maximal binary length of the integers occurring in $\bar{\alpha}$.

In particular, for

$$h = z^m + h_{m-1}z^{m-1} + \cdots + h_0 \in \mathbb{Z}[z] \setminus \{0\}$$

we have

$$l(h) \leqslant \log \max_{0 \leqslant j < m} |h_j| + 1,$$

and for $b = 1/r_m \sum_{0 \leqslant j < m} r_j y^j \in F$ with $r_j \in \mathbb{Z}$, $r_m \neq 0$,

$$l(b) \leqslant \log \max_{0 \leqslant j \leqslant m} |r_j| + 1.$$

Also, when $b = (b_1, ..., b_n) \in F^n$, we use $l(b) = \max_{1 \leqslant i \leqslant n} l(b_i)$. Even when $m = 1$, we do not require in a standard representation two integers r_0, r_1 (representing $r_0/r_1 \in \mathbb{Q}$) to be relatively prime, and therefore $l(\bar{\alpha})$ depends not only on α, but on the particular representation given. In our algorithms, we assume inputs α, h, b to be given in a standard representation, and then write $l(\alpha)$, $l(h)$, $l(b)$ referring to the length given by that particular representation.

Definition 6.2. Let α be a computation over $F \cup \{x_1, ..., x_n\}$, and $b \in F^n$. If on execution of α on input b no division by zero occurs, then we say that α is *defined at* b.

Remark 6.3. In Proposition 5.2, we have noted that for a random $b \in A^n$ with $\# A = a$, α is defined at b with probability at least $1 - 2^s/a$.

Note that all intermediate results $f_i \in F(x_1, \ldots, x_n)$ of α may be defined at b, but yet α is not defined at b. An example is $b=0$ and $\alpha = ((x_1, \varnothing), (/, (1, 1)))$, so that $f_1 = x_1, f_2 = x_1/x_1 = 1$.

Given integers p, r, r', q, q' with $|r'|$, $|q'| \leqslant p/2$, q, $q' \neq 0$, $r' \equiv r \bmod p$, and $q' \equiv q \bmod p$, we call (r', q') a mod-p-representation of $r/q \in \mathbb{Q}$. For the following algorithm, we use some Monte Carlo test for compositeness of numbers, e.g., Solovay and Strassen [41]. On input an integer p and a confidence parameter $\gamma > 0$, it can be performed in $O((\log p)^{2+\varepsilon} \log(1/\gamma))$ bit operations for any $\varepsilon > 0$. If p is prime, it returns "p is prime." If p is composite, it returns either "p is composite" or "p is prime"; the latter with probability at most γ. The exponent ε really only hides logarithmic factors (in $\log p$). To simplify notation in the sequel, we introduce the following abbreviation.

Definition 6.4. Let $s, t \colon \mathbb{N} \to \mathbb{R}_{\geqslant 0}$ be functions. We write $s = O^*(t)$ if and only if there exist $k, m \in \mathbb{N}$ such that

$$\forall n \geqslant m, \qquad s(n) \leqslant t(n) \cdot (\log(2 + t(n)))^k.$$

The purpose of the summand 2 is to make the logarithm always at least 1, so that, e.g., $s = O^*(t)$ for the constant functions $s(n) = 5$, $t(n) = 1$.

Algorithm POLYNOMIAL TEST.

Input: The coefficients of an irreducible monic polynomial $h \in \mathbb{Z}[z]$ of degree m such that $F = \mathbb{Q}[z]/(h)$, a computation α for a polynomial $f \in F[x_1, \ldots, x_n]$, and $b = (b_1, \ldots, b_n) \in F^n$, with $b_1, \ldots, b_n$ and the constants of α in a standard representation, and a confidence parameter δ, $0 < \delta < 1$.

Output: Either "failure," or "α is defined at b."

1. Set $R = 1 + l(h) + l(\alpha) + l(b) + \log m$, $T = (3/\delta)\, s^2 m^s R$, $N = 2T \log T$, and $t = \lceil 2 \cdot \log N \cdot \log(3/\delta) \rceil$. Choose independently integers $p_1, \ldots, p_t$ with $1 \leqslant p_i \leqslant N$ at random, and run a Monte Carlo compositeness test on them, with confidence parameter $\delta/3$. Let p be the first p_i for which "p_i is prime" is returned. If always "p_i is composite" is returned, then output "failure" and stop.

2. Execute α on input b. Maintain a mod-p-representation for the intermediate results. If a division by zero occurs, then return "failure" and stop.

3. Return "α is defined at b."

Proposition 6.5. *Let h, α, b, δ be an input for POLYNOMIAL TEST, s the size of α, $m = \deg h$, and $k = \max\{s, m, l(h), l(\alpha), l(b), \log(1/\delta)\}$. Then the algorithm can be performed in $O^*(k^5)$ bit operations. If α is not defined at b, then "failure" is returned. If α is defined at b, then "α is defined at b" is returned with probability at least $1 - \delta$.*

Proof. Denote by $f_1, \ldots, f_s \in F(x_1, \ldots, x_n)$ the intermediate results of α. We consider a standard representation $(r_{i0}, \ldots, r_{im})$ (with respect to h) for each intermediate result $f_i(b)$, for which no division by zero has occurred, where each $r_{ij} \in \mathbb{Z}$, and

$$f_i(b) = \frac{1}{r_{im}} \sum_{0 \leqslant j < m} r_{ij} y^j \in F.$$

(In step 2 of the algorithm, we compute representatives r'_{ij} of $r_{ij} \bmod p$, with $r'_{ij} \in \mathbb{Z}$ and $|r'_{ij}| \leqslant p/2$.) We can find these standard representations for $f_i(b)$ along the algorithm in the obvious way. If, e.g., $f_i = f_k * f_l$ is a multiplication step in α, then $r_{i0}, \ldots, r_{i,m-1} \in \mathbb{Z}$ are obtained by dividing

$$\Bigl(\sum_{0 \leqslant j < m} r_{kj} z^j\Bigr)\Bigl(\sum_{0 \leqslant j < m} r_{lj} z^j\Bigr)$$

by h with remainder. Since h is integral and monic, this remainder is integral. Also $r_{im} = r_{km} \cdot r_{lm}$. An addition step is treated similarly, and inputs and constants are trivial. If $f_i = f_k / f_l$ is a division and $f_i(b) \neq 0$, then we use an inverse of $\sum r_{lj} z^j$ modulo h, as calculated, e.g., in the extended Euclidean algorithm of [1, Chap. 8], and remove common factors from r_{lm} and the coefficients of this inverse. To get a bound on the size of the intermediate results, we consider the system of $2m-1$ linear equations corresponding to

$$\sum_{0 \leqslant j < m} y_{ij} z^j \cdot \sum_{0 \leqslant j < m} r_{lj} z^j = \sum_{0 \leqslant j < m} r_{kj} z^j + \sum_{0 \leqslant j \leqslant m-2} q_j z^j \cdot h$$

in $2m-1$ unknowns y_{ij}, q_j. This system has a unique solution, and we let r_i^* be the determinant of the coefficient matrix. We set

$$r_{ij}^* = (y_{ij} \cdot r_i^*) \cdot r_{lm} \quad \text{for} \quad 0 \leqslant j < m,$$

and $r_{im}^* = r_i^* r_{km}$. Then, if $(r_{i0}, \ldots, r_{im})$ is the representation for $f_i(b)$ computed above, we have $r_{ij} \leqslant r_{ij}^*$ for all j. Now let

$$\begin{aligned} M_1 &= \max\{2^{l(\alpha)}, 2^{l(b)}\}, \\ M_i &= (m 2^{1+l(h)})^{m(i-1)} M_1^{m^{i-1}} \qquad \text{for} \quad 2 \leqslant i \leqslant s. \end{aligned}$$

We first prove that $|r_{ij}| \leqslant M_i$ for $1 \leqslant i \leqslant s$ and $0 \leqslant j \leqslant m$, by induction on i. The claim is clear for $i = 1$. For the induction, we consider a division step $f_i = f_k / f_l$; the other operations are checked similarly. We have to solve a $(2m-1) \times (2m-1)$-system of linear equations, and know that

$$|r_{lj}|, |r_{kj}| \leqslant M_{i-1}.$$

Cramer's rule and Hadamard's inequality imply that

$$|r_{ij}| \leqslant |r_{ij}^*| \leqslant (2m)^{m/2} M_{i-1}^m (2m)^{(m-1)/2} 2^{l(h) \cdot (m-1)} \leqslant (2m 2^{l(h)} M_{i-1})^m = M_i.$$

Before we estimate the failure probability, we have to specify how to "keep all intermediate results in mod-p-representation" in step 2. With $r_{ij} \in \mathbb{Z}$ as above, we simply maintain representatives $r'_{ij} \in \mathbb{Z}$ of $r_{ij} \bmod p$ with $|r'_{ij}| \leqslant p/2$, and calculate r'_{ij} from r'_{kj}, r'_{lj} as indicated above. We assume now that α is defined at b, and consider the first failing division step $f_i = f_k/f_l$. Then $f_l(b) \neq 0$, hence $(r_{l0}, \ldots, r_{l,m-1}) \neq (0, \ldots, 0)$, and $f_l(b) \equiv 0 \bmod p$, so that p divides each of $r_{l0}, \ldots, r_{l,m-1}$. Let

$$P = \{p \in \mathbb{N} : 2 \leqslant p \leqslant N \text{ and } p \text{ is prime}\},$$

so that

$$\# P = \pi(N) \geqslant \frac{N}{\log_e N} \geqslant \frac{N}{\log N}$$

for $N \geqslant 17$, by the prime number theorem of Rosser and Schoenfeld [36]. ("log" without subscript always stands for "$\log_2$.") Let

$$Q = \{p \in P : r_{l0} \equiv \cdots \equiv r_{l,m-1} \equiv 0 \bmod p \text{ for some division step } f_i = f_k/f_l\}.$$

For each divisor f_l as above, there exists a j, $0 \leqslant j < m$ with $r_{lj} \neq 0$ and $|r_{lj}| \leqslant M_s$. Therefore

$$\# \{p \in P : p \mid r_{lj}\} \leqslant \log M_s,$$

$$\# Q \leqslant s \cdot \log M_s \leqslant s(ms(1 + l(h) + \log m) + m^{s-1} \log M_1) \leqslant s^2 m^s R.$$

Furthermore we have

$$\frac{N}{\log N} = \frac{2T \log T}{\log(2T \log T)} \geqslant T,$$

since $T \geqslant 2 \log T$ for $T \geqslant 4$. Together we obtain

$$\begin{aligned}
\text{Prob(failure)} &\leqslant \text{Prob}(p_1, \ldots, p_t \text{ are composite}) + \text{Prob}(p \text{ is not prime}) + \# Q/\# P \\
&\leqslant \left(\frac{N - \# P}{N}\right)^t + \frac{\delta}{3} + s^2 m^s R \frac{\log N}{N} \\
&\leqslant \left(1 - \frac{1}{\log N}\right)^t + \frac{\delta}{3} + \frac{\delta}{3} \\
&\leqslant e^{-t/\log N} + \frac{2\delta}{3} \leqslant \frac{\delta}{3} + \frac{2\delta}{3} = \delta.
\end{aligned}$$

For the timing estimate, we know that an operation (addition, multiplication, division with remainder, computing a modular inverse) on i-bit integers (resp.

polynomials with rational coefficients of degree at most i) can be performed with $O^*(i)$ bit operations (resp. operations in $\mathbb{Q}$) ([1, Chap. 8]. In step 2, each intermediate result is represented by integers of at most $\log N$ bits, and one operation $(+, -, *, /) \bmod h$ can be performed in

$$O^*(m \log N) = O^*\left(m \log \frac{sR}{\delta} + sm \log m\right) = O^*(k^2)$$

bit operations, giving total cost $O^*(k^3)$ for step 2. The cost for step 1 is $O^*((\log N)^3(\log(1/\delta))^2)$ or $O^*(k^5)$. ▮

Note that even if p is not a prime and "α is defined at b" is returned, then this is the correct output.

We now have the required random polynomial-time version of Strassen's division-free conversion.

ALGORITHM DIVISION-FREE CONVERSION OVER A NUMBER FIELD.

Input: The coefficients of a monic irreducible polynomial $h \in \mathbb{Z}[z]$ of degree m such that $F = \mathbb{Q}[z]/(h)$, the description $\bar{\alpha}$ of a computation α for a polynomial $f \in F[x_1, ..., x_n]$ of degree d, and a number $a \in \mathbb{N}$.

Output: Either "failure," or another computation α^* for f.

1. Set $A = \{1, ..., 2a\} \subseteq \mathbb{N} \subseteq F$, and choose $b \in A^n$ at random.
2. Call algorithm POLYNOMIAL TEST with input h, α, b, and $\delta = 2^{s-1}/a$.
3. Execute steps 3 and 4 of Algorithm DIVISION-FREE CONVERSION on input α, b, and output α^*.

PROPOSITION 6.6. *Let h, $\bar{\alpha} = (\gamma, \beta)$, d, a be an input for DIVISION-FREE CONVERSION OVER A NUMBER FIELD, and s the size of the computation α. The algorithm outputs either "failure" or a computation α^* for f. α^* has size $O(sd \log^2 d)$, and consists of three phases: the first phase involves only constants from $\{\gamma_1, ..., \gamma_s\} \cup \mathbb{N}$ (and no inputs x_i), the second phase is a homogeneous division-free computation, and the third phase consists of $d+1$ additions. Let*

$$k = \max\{s, d, m, \log a, l(h), l(\alpha)\}.$$

The conversion procedure can be performed with $O^*(k^5)$ bit operations and $O(n \log a)$ random bit choices. The failure probability is at most $2^s/a$.

Proof. The failure probability is $\leqslant 2^s/2a + \delta = 2^s/a$, using Remark 6.3 and Proposition 6.5. Since $l(b)$ and $\log(1/\delta)$ are $O(k)$, the number of bit operations is $O^*(k^5)$ in step 2, and $O^*(k^3)$ in step 3. ▮

Algorithm POLYNOMIAL TEST runs in random polynomial time (in the input size) and almost evaluates a computation, namely it computes the value modulo a large prime. We cannot expect such an algorithm for actual evaluation, because the output size may be more than polynomial in the input size. However, if we take the output size into account, we do get a random polynomial time procedure for evaluation. If

$$f=\frac{1}{f_m}\sum_{\substack{e\in\mathbb{N}^n\\0\leqslant j<m}} f_{ej}y^jx_1^{e_1}\cdots x_n^{e_n}$$

with $f_{ej}, f_m \in \mathbb{Z}$, then the maximal binary length $L(f)$ of any of the f_{ej}, f_m is called the length of (this representation of) f.

Remark 6.7. We only allow one denominator f_m, since otherwise for

$$f=\sum_{e\in\mathbb{N}^n}\frac{1}{p_e}x_1^{e_1}\cdots x_n^{e_n}\in\mathbb{Q}[x_1,\ldots,x_n]$$

of degree d, where p_e runs through the first $\binom{n+d}{n}$ primes, the length of $f(1,\ldots,1)$ might be exponential in the length of the representation. (The restriction may actually not be necessary, since it is not clear that computations of small size can compute such polynomials.)

ALGORITHM POLYNOMIAL EVALUATION.

Input: As for POLYNOMIAL TEST, and $L(f)$, and the degrees d and m of f and h resp.

Output: Either "failure," or $f(b)\in F$ (in standard representation).

1. Set $B=L(f)+2d(n+1)\,l(b)+2dm\log(dm)+dml(h)$.
2. Call Algorithm POLYNOMIAL TEST. In step 1 of that procedure, set $C=B+\lceil\log((6s\log^2 N)/\delta)\rceil$, and make the following changes: Use the value $t=\lceil 2\cdot(2C+1+\log N)\cdot\log(3/\delta)\rceil$, and choose independently integers $p_1,\ldots,p_t$ with
$$2^{2C+1}<p_i\leqslant N2^{2C+1}.$$
3. Return the computed mod-p-representation of $f(b)$.

PROPOSITION 6.8. *Let h, α, b, δ be an input for POLYNOMIAL EVALUATION, s the size of α, and $k=\max\{s, d, m, L(f), l(b), l(h), l(a), \log(1/\delta)\}$. Then the algorithm can be performed in $O^*(k^{11})$ bit operations, and $l(f(b))\leqslant B$. If α is not defined at b, then "failure" is returned. If α is defined at b, then the failure probability is at most δ.*

Proof. We first show that $l(f(b))\leqslant B\leqslant C$. Then, if (r, q) is the mod-p-representation of $f(b)$, in fact we have $f(b)=r/q$. Write

$$f = \frac{1}{f_m} \sum_{\substack{e \in \mathbb{N}^n \\ 0 \leqslant j < m}} f_{ej}\, y^j x_1^{e_1} \cdots x_n^{e_n},$$

$$b = (b_1, \ldots, b_n),$$

$$b_i = \frac{1}{b_{im}} \sum_{0 \leqslant j < m} b_{ij}\, y^j,$$

with $f_{ej}, f_m, b_{ij} \in \mathbb{Z}$. For any $e \in \mathbb{N}^n$ with $e_1 + \cdots + e_n \leqslant d$, define

$$u_e = \left(\sum_{0 \leqslant j < m} b_{1j} z^j \right)^{e_1} \cdots \left(\sum_{0 \leqslant j < m} b_{nj} z^j \right)^{e_n} \in \mathbb{Z}[z],$$

so that $u_e \bmod h = (b_{1m} b_1)^{e_1} \cdots (b_{nm} b_n)^{e_n}$. When the product for u_e is multiplied out, there are at most m^d summands c, each with $l(c) \leqslant d \cdot l(b)$. It follows that $l(u_e) \leqslant d \log m + dl(b)$. For any $e \in \mathbb{N}^n$, we next define

$$v_e = \sum_{0 \leqslant j < m} f_{ej} z^j b_{1m}^{d-e_1} \cdots b_{nm}^{d-e_n} u_e \in \mathbb{Z}[z],$$

$$v = \sum_{e \in \mathbb{N}^n} v_e \in \mathbb{Z}[z];$$

then

$$\begin{aligned} l(v_e) &\leqslant \log m + L(f) + dnl(b) + l(u_e) \\ &\leqslant L(f) + d(n+1)\, l(b) + (d+1) \log m, \\ l(v) &\leqslant \log \binom{n+d}{n} + \max_e l(v_e) \\ &\leqslant d \log(n+1) + L(f) + d(n+1)\, l(b) + (d+1) \log m, \end{aligned}$$

$$\deg v \leqslant (d+1)(m-1),$$

$$f(b) = \frac{1}{f_m b_{1m}^d \cdots b_{nm}^d} \cdot (v \bmod h) \in F.$$

The coefficients $w_0, \ldots, w_{m-1}$ of $v \bmod h = \sum w_j y^j \in F$ can be computed by solving a system of at most $(d+1)(m-1)$ linear equations, with each entry of the coefficient matrix having length at most $l(h)$, and each entry of the constant side of length at most $l(v)$. Using Cramer's rule and Hadamard's inequality we obtain

$$\begin{aligned} l(w_j) &\leqslant l(v) + dm \log(dm) + dm\, l(h) \\ &\leqslant L(f) + 2d(n+1)\, l(b) + 2\, dm \log(dm) + dm\, l(h) = B, \end{aligned}$$

and thus $l(f(b)) \leqslant B$.

For the timing estimate, we first note that $\log N$ is $O^*(k)$, B and C are $O(k^3)$, and t is $O(k^4)$. Set $D = 2^{2C+1}$. A random prime number as required can be chosen in $O(t \cdot \log(ND))$ or $O(k^7)$ random bit choices, and $O^*(t(\log(ND))^2 \log(3/\delta))$ or

$O^*(k^{11})$ bit operations. The cost for one operation mod h is now $O^*(m\log(ND))$ or $O^*(k^4)$, giving total cost $O^*(k^5)$ for step 2 of the call of POLYNOMIAL TEST.

For the estimate of the failure probability, let $\pi(x)$ be the number of prime numbers between 2 and x. By the prime number theorem in Rosser and Schoenfeld [36, Corollary 1], the fraction corresponding now to $(N-\#P)/N$ satisfies for $C \geqslant 1$ and $N \geqslant 15$ the following:

$$\frac{\pi(ND)-\pi(D)}{ND-D} > \frac{\dfrac{ND}{\log(ND)} - \dfrac{2D}{\log D}}{D(N-1)}$$
$$\geqslant (N-2\log N)/((N-1)\log(ND)) \geqslant (2\cdot(2C+1+\log N))^{-1}.$$

Thus again the probability that $p_1,..., p_t$ are composite is at most $\delta/3$.

It remains to estimate the failure probability under the assumption that α is defined at b and p is indeed prime. We can assume that $C \geqslant 12$, and first note that

$$\frac{s}{2^C} \leqslant \delta, \qquad \frac{\log^2 N}{2^C} \leqslant \frac{1}{6}, \qquad \frac{(2C+1)^2}{2^C} \leqslant \frac{1}{6},$$

$$E = \frac{N}{\log(ND)} - \frac{2}{\log D} \geqslant \frac{1}{\log(ND)}.$$

By the argument given above, $l(r_i) \leqslant C < \log(ND)$ for every intermediate result $r_i \in F$ of α on input b. Thus for

$$Q = \{p \colon D \leqslant p \leqslant ND,\ p \text{ is prime, some division in } \alpha \text{ on input } b \text{ fails modulo } p\},$$

we have

$$\#Q \leqslant s\log(ND).$$

Thus

$$\text{Prob(failure)} \leqslant \frac{s\log(ND)}{\pi(ND)-\pi(D)} \leqslant \frac{s\log(ND)}{DE}$$
$$\leqslant \frac{s}{2^C}\cdot\frac{\log^2(ND)}{2^{C+1}} \leqslant \delta\left(\frac{2\log^2 N}{2^{C+1}} + \frac{2(2C+1)^2}{2^{C+1}}\right)$$
$$\leqslant \delta\left(\frac{1}{6}+\frac{1}{6}\right) = \frac{\delta}{3}. \qquad \blacksquare$$

COROLLARY 6.9. *Computations over algebraic number fields can be simulated by probabilistic Boolean circuits with size polynomial in the input and output size.*

It is clear that also "probabilistic computations" over algebraic number fields can be simulated, e.g., Las Vegas computations that make probabilistic choices from a finite set $A \subseteq \mathbb{Z} \subseteq F$ and either return the correct function value or "failure." Then $\log(\max A)$ will enter the probabilistic Boolean simulation time.

The main result of this section is the following random polynomial-time algorithm for the factorization pattern over number fields.

ALGORITHM FACTORIZATION PATTERN OVER A NUMBER FIELD.

Input: The coefficients of an irreducible monic polynomial $h \in \mathbb{Z}[z]$ of degree m such that $F = \mathbb{Q}[z]/(h)$, a computation α of size s for a polynomial $f \in F[x_1,..., x_n]$ of degree d, and $L(f)$.

Output: Either "failure," or the factorization pattern of f.

1. Call procedure DIVISION-FREE CONVERSION OVER A NUMBER FIELD with input h, α, d, $a = 2^{sd}(9^{d^2} + 2^s)$, and output α^*.

2. Call procedure DIVISION-FREE FACTORIZATION PATTERN with input α^* and $A = \{1,..., a\} \subseteq \mathbb{N} \subseteq F$. Output is a computation α_i for each of the $\binom{d+2}{2}$ coefficients $g_i \in F$ of a polynomial $g \in F[x_1, x_2]$, with degree at most d.

3. Set $B = d \log(n+1) + L(f) + 2d(n+1) \log a + 2\, dm \log(dm) + dm\, l(h)$. For all i, call procedure POLYNOMIAL EVALUATION with input h, α_i for $g_i \in F$ with $n = 0$, $\delta = 2^{-sd}$, $L(g_i) = B$, d, m, to compute the standard representation of each coefficient g_i. (No input b is required.)

4. Apply a factorization algorithm for bivariate polynomials to g, and return the factorization pattern of g.

THEOREM 6.10. *Consider an input h, α, f for the algorithm. Let $m = \deg h$, and*

$$k = \max\{s, d, m, l(h), l(\alpha), L(f)\}.$$

Then the Algorithm FACTORIZATION PATTERN OVER A NUMBER FIELD can be executed with $O^*(k^{46})$ bit operations. With probability greater than $1 - 2^{-sd}$, it returns the correct factorization pattern of f. Furthermore, $L(g) \leqslant B$.

Proof. By definition, $L(g_i) \leqslant L(g)$, and we first estimate $L(g)$. We write f with integer coefficients f_m, f_{ej} as usual, and let

$$\begin{aligned} f^* &= f(x_1, x_2, U_3 x_1 + V_3 x_2 + W_3,..., U_n x_1 + V_n x_2 + W_n) \\ &= \frac{1}{f_m} \sum_{e,j} f_{ej}\, y^j x_1^{e_1} x_2^{e_2} (U_3 x_1 + V_3 x_2 + W_3)^{e_3} \cdots (U_n x_1 + V_n x_2 + W_n)^{e_n} \\ &= \frac{1}{f_m} \sum_{\substack{d \in \mathbb{N}^2 \times \mathbb{N}^{3(n-2)} \\ 0 \leqslant j < m}} g_{dj}\, y^j x_1^{d_1} x_2^{d_2} U_3^{d_{13}} \cdots U_n^{d_{1n}} V_3^{d_{23}} \cdots V_n^{d_{2n}} W_3^{d_{33}} \cdots W_n^{d_{3n}} \end{aligned}$$

with $g_{dj} \in \mathbb{Z}$. Each g_{dj} is the sum of at most $\binom{n+d}{d}$ summands f_{ej}, and therefore

$$L(f^*) \leqslant \log \binom{n+d}{d} + L(f) \leqslant d \log(n+1) + L(f).$$

Since $g = f^*(u_3, ..., u_n, v_3, ..., v_n, w_3, ..., w_n)$ and $l(u_3, ..., w_n) \leqslant \log a$, we have by Proposition 6.8 that

$$L(g) \leqslant B = O(k^3).$$

Step 1 can be executed with $O^*(k^{10})$ bit operations by Proposition 6.6, and step 2 with $O^*(k^4)$ operations; this step is formal and does not involve actual calculation with elements from F. For the output of step 1, both $l(\alpha^*)$ and the size of α^* are $O(k^2)$. If we denote by $s_i, d_i, ...$ the parameters in the ith call of POLYNOMIAL EVALUATION in step 3, then $s_i = O^*(k^4)$, $d_i = l(b_i) = 0$ (since each g_i is constant), $m_i = m$, $L(g_i) = O(k^3)$, $l(\alpha_i) = O^*(k^3)$, $h_i = h$, $\log(1/\delta) = O(k^2)$. Proposition 6.8 yields the estimate $O^*(d^2k^{44})$ for step 3. The estimate of Lenstra [30] gives a bound of $O^*(k^{31})$ bit operations for step 4.

The failure probability is less than $2^s/a$ in a step 1, and the correct factorization pattern is computed with probability at least $1 - 9^{d^2}/a$. ▮

Remark 6.11. A more careful look at the proof of Proposition 6.8 shows that steps 1, 2, 3 can be performed in $O^*(k^{15})$ bit operations.

We do not get a fast parallel algorithm over number fields, since even for univariate factoring (or irreducibility testing) over $\mathbb{Q}$ no fast parallel algorithm is known. (See [9] for a discussion.)

7. Field Extensions and Factorization

The factorization pattern algorithm for multivariate polynomials in Section 5 assumes that one can make random choices from a sufficiently large finite subset of the ground field. This may not be possible over a small finite field. In this section we prove that one can make arbitrarily large algebraic extensions of a field without changing the factorization of a given polynomial. This allows us to apply the algorithm also to small finite fields.

Theorem 7.1. *Let F be an arbitrary field, $f \in F[x_1, ..., x_n]$ of total degree d, and $F \subseteq K$ a finite algebraic extension of degree m such that* $\gcd(m, d) = 1$. *Then*

(i) *f irreducible in $F[x_1, ..., x_n] \Leftrightarrow f$ irreducible in $K[x_1, ..., x_n]$.*

(ii) *If each prime factor of m is greater than d, then f has the same factorization pattern over F and K.*

Proof. We will use a classical notion, the norm $N = N_{K/F}: K \rightarrow F$ of the given field extension. For any $n \geqslant 0$, we also have a mapping $N: K[x_1, ..., x_n] \rightarrow$

$F[x_1,..., x_n]$. If, e.g., $K=F(a)$ is separable over F, $m=[K:F]$, $L \supseteq K$ a splitting field of the minimal polynomial of a over F, and $a_1 = a, a_2,..., a_m \in L$ are the conjugates of a over F, then

$$N\left(\sum_{\substack{0 \leqslant i < m \\ e \in \mathbb{N}^n}} f_{ie} a^i \mathbf{x}^e\right) = \prod_{1 \leqslant k \leqslant m} \left(\sum_{\substack{0 \leqslant i < m \\ e \in \mathbb{N}^n}} f_{ie} a_k^i \mathbf{x}^e\right),$$

where $f_{ie} \in F$. (Actually, this N is nothing but $N_{K(x_1,...,x_n)/F(x_1,...,x_n)}$.) N is multiplicative, $\deg N(g) = m \deg g$, and if $f \in F[x_1,..., x_n]$, then $N(f)=f^m$. (See, e.g., [51, Chap. II, Sect. 10].)

In proving (i), the implication "$\Leftarrow$" is clear. So let f be irreducible in $F[x_1,..., x_n]$, $g, h \in K[x_1,..., x_n]$ with $f=gh$, and $k=\deg g$ the total degree of g. Then

$$f^m = N(f) = N(g)\, N(h)$$

implies that $N(g)=f^l$ for some l, $0 \leqslant l \leqslant m$, and

$$dl = \deg N(g) = km.$$

Now from $\gcd(m, d)=1$ it follows that m divides l. Thus either $l=k=0$, or else $l=m$, which implies $k=d$ and $f=g$. In either case, the factorization is trivial. (ii) follows immediately from (i). ∎

Using this theorem, it is now easy to put the reduction of Section 5 to work over a finite field F. In order to compute the factorization pattern of a bivariate polynomial g of total degree at most d—as output by DIVISION-FREE FACTORIZATION PATTERN—we use the probabilistic factoring algorithm BIVARIATE FACTORING from [11]. It either returns the correct factorization of g or else "failure." The latter happens with probability at most 2^{-d}.

Algorithm FACTORIZATION PATTERN OVER A FINITE FIELD.

Input: A computation α over a finite field F with q elements, computing a polynomial $f \in F[x_1,..., x_n]$ of total degree d.

Output: Either a factorization pattern, or "failure."

1. Set $a = 2^{2d}(9^{d^2} + 2^s)$. If $q \geqslant a$, then set $K=F$ and go to step 3. Else set $m = \max\{d, \log_q a\}$, and choose a prime number l with $m < l \leqslant 2m$.

2. Choose monic polynomials $h_1,..., h_{8ld} \in F[z]$ of degree l at random, and test them for irreducibility. If none is irreducible, return "failure" and stop. Otherwise, let h be the first irreducible h_i, and set $K=F[z]/(h)$.

3. Call Algorithm DIVISION-FREE CONVERSION, and then DIVISION-FREE FACTORIZATION PATTERN with input α computing $f \in K[x_1,..., x_n]$, and some $A \subseteq K$ with $\# A = a$. Evaluate the resulting computations for the coefficients of $g = f\{t\}$.

4. Run a bivariate factoring algorithm over K on the output g of step 3. Return the factorization pattern of g (or possibly "failure").

THEOREM 7.2. *Let F be a finite field with q elements, and α a computation of size s over F computing a polynomial $f \in F[x_1, ..., x_n]$ of total degree d. On input α, FACTORIZATION PATTERN OVER A FINITE FIELD outputs the correct factorization pattern of f with probability greater than 2^{-d}. Let $k = \max\{s, d, \log q\}$. The algorithm can be performed in $O(k^{14})$ bit operations, and $O(k^5)$ random bit choices.*

Proof. There are exactly $(q^l - q)/l$ irreducible monic polynomials of degree l in $F[x]$ (this is already in Schönemann [37, Sect. 46]), so that step 2 has failure probability at most

$$\left(1 - \frac{q^l - q}{lq^l}\right)^{8ld} \leqslant \left(1 - \frac{1}{2l}\right)^{8ld} \leqslant e^{-2d} < 2^{-2d}.$$

We run the bivariate factoring algorithm in step 4 of FACTORIZATION PATTERN twice to obtain failure probability at most 2^{-2d}. Using Propositions 5.2 and 5.3, it follows that the factorization pattern of f is computed with probability at least 2^{-d}.

For the timing estimate, first note that a prime number l as required exists by Bertrand's postulate [36, Corollary 3]. We can compute such an l deterministically in $O(m^{3/2} \log^2 m)$ bit operations. Step 2 uses $O(l^3 d \log^2 l \log \log l \log q)$ or $O^*(k^7)$ operations in F [35]. One arithmetic operation in K can be performed in $O^*(l)$ or $O^*(k^2)$ operations in F, and thus with $O^*(k^3)$ bit operations. The two procedures called in step 3 work in $O^*(k^3)$ operations in K, or $O^*(k^6)$ bit operations. The bivariate factoring algorithm over K works in $O(d^3 \log^2(q^l)(d^7 + \log d \log(q^l)))$ or $O(k^{14})$ bit operations and uses $O(d \log d \log(q^l))$ random bit choices. Step 2 uses $O(l^2 d \log q)$ random bit choices. ∎

We also obtain a parallel version of the algorithm. For general computations, we cannot expect fast parallel evaluation, only polynomial-time transformations (as in Proposition 5.5) that yield special computations which can be evaluated fast in parallel. Therefore we now assume that the input is a computation of depth r, and look for a factorization pattern algorithm with depth polynomial in r.

We use the parallel bivariate factoring algorithm from [11]. The algorithm has to be performed in the field K constructed in step 2 of FACTORIZATION PATTERN OVER A FINITE FIELD. For factoring $f\{t\}$, one may have to extract pth roots of elements of K, where $p = \text{char } K$. This can be performed by Boolean circuits of depth $O(\log^2 e + \log p)$ and size $(e \cdot \log p)^{O(1)}$, if $\# K = p^e$ [8, 12].

COROLLARY 7.3. *Let F, q, α, s, f, n, d, k be as in Theorem 7.2, $p = \text{char } F$, $q = p^e$, and r the depth of α. Then the factorization pattern of f can be computed with a Boolean circuit of depth $O(\log^6 k(r + \log p))$ and size $k^{O(1)}$.*

Proof. We only estimate the depth of the required Boolean circuits. Step 1 can be performed in depth $O(\log^3 m)$. For step 2, we use a deterministic version of the parallel irreducibility test in [9, Sect. 4], which works in depth $O(\log^2(el) \log p \log^2 \log p)$. The evaluation in step 3 of FACTORIZATION PATTERN OVER A FINITE FIELD can be performed in depth $O(r \log^2 d (\log \log q)^2)$. The bivariate factorization algorithm can be implemented on a Boolean circuit of depth $O(\log^2 d \log^2(del) \log p (\log \log q)^2)$. ▌

8. Formulas and Sparse Representations

The results of the previous sections simplify somewhat when we restrict ourselves to the cases where α is a formula or a sparse representation. As usual, α has size s and computes $f \in F[x_1, ..., x_n]$ of degree d, and $A \subseteq F$ has a elements.

Theorem 8.1. *There are two modifications of Algorithm FACTORIZATION PATTERN, which on input of a formula α, output either "failure" or a formula $\alpha\{t\}$ for a bivariate polynomial $g = f\{t\}$. With probability greater than $1 - (9^{d^2} + 2^s)/a$, "failure" does not occur and g and f have the same factorization pattern. In particular, with this probability g is irreducible if and only if f is irreducible.*

(i) *(Sequential version) The first modification can be performed with $O(s)$ steps, and $\alpha\{t\}$ has size at most 8s.*

(ii) *(Parallel version) The second modification can be performed with $O(s^{1.45})$ steps. The size of $\alpha\{t\}$ is $O(s)$, and the depth of $\alpha\{t\}$ is $O(\log s)$.*

Proof. (i) In α, the algorithm simply writes $u_j x_1 + v_j x_2 + w_j$ for each x_j with $3 \leqslant j \leqslant n$.

(ii) A construction for $\alpha\{t\}$ is given in [33] with the compilation time $O(s^{\xi})$, where $\xi = 1/\log_2 \delta = 1.44...$, and $\delta = (1 + \sqrt{5})/2$. ▌

Theorem 8.2. *There are two modifications of algorithm FACTORIZATION PATTERN, which on input of a sparse representation α, output either "failure" or a sparse representation $\alpha\{t\}$ for a bivariate polynomial $g = f\{t\}$. With probability greater than $1 - 9^{d^2}/a$, "failure" does not occur and g and f have the same factorization pattern. If f is reducible and "failure" does not occur, then g is reducible.*

(i) *(Sequential version) The first modification can be performed with $O(sd^3)$ steps, and $\alpha\{t\}$ has size $O(d^2)$.*

(ii) *(Parallel version) The second modification can be performed with $O(sd^4 \log^2 d)$ steps, and depth $O(\log^2 d + \log s)$. The size of $\alpha\{t\}$ is $O(d^2)$, and the depth of $\alpha\{t\}$ is $O(\log d)$.*

Proof. (i) Write $f=\sum_{e\in\mathbb{N}^n} f_e x_1^{e_1}\cdots x_n^{e_n}$ with $f_e\in F$. For each $e\in\mathbb{N}^n$ with $f_e\neq 0$, we have $e_1+\cdots+e_n\leqslant d$, and

$$f_e x_1^{e_1} x_2^{e_2}(u_3x_1+v_3x_2+w_3)^{e_3}\cdots(u_nx_1+v_nx_2+w_n)^{e_n}\in F[x_1,x_2]$$

is a polynomial of degree at most d and is (densely) represented by its $\beta_d=\binom{d+2}{2}$ coefficients. (For bivariate polynomials, the dense and sparse representations have polynomially related lengths, and we do not distinguish between the two.) Each coefficient can be computed with $6d\beta_d=O(d^3)$ operations in F. Thus the sparse representation of g can be computed with $7d\beta_d s=O(d^3s)$ operations in F. We can read $\deg g$ from this representation of g, and return "failure" if $\deg g<\deg f$. We are then guaranteed that g is reducible if f is. By Theorem 4.5 and Fact 4.7, the factorization patterns of f and g agree and $\deg f=\deg g$ with probability at least $1-9^{d^2}/a$.

(ii) The coefficients of the product of two bivariate polynomials of degree at most d can be computed in size $O(d^4)$ and depth $O(\log d)$. The claim now follows, using the algorithm from (i). It is clear how to evaluate a sparse representation fast in parallel. ▮

Acknowledgments

It is a pleasure to acknowledge the many conversations I had with Erich Kaltofen about the factoring problem, and the numerous improvements and corrections he contributed. Thanks also go to the referees for various helpful suggestions.

References

1. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, "The Design and Analysis of Computer Algorithms," Addison-Wesley, Reading, Mass., 1974.
2. E. R. Berlekamp, Factoring polynomials over finite fields, *Bell System Tech. J.* **46** (1967), 1853–1859.
3. E. R. Berlekamp, Factoring polynomials over large finite fields, *Math. Comp.* **24** (1970), 713–735.
4. E. Bertini, Sui sistemi lineari, *Rend. Reale Istit. Lombard.* **15** (1882), 24–28.
5. A. Borodin, J. von zur Gathen, and J. Hopcroft, Fast parallel matrix and GCD computations, *Inform. and Control* **52** (1982), 241–256.
6. A. L. Chistov and D. Yu. Grigoryev, "Polynomial-time Factoring of the Multivariable Polynomials over a Global Field," LOMI preprint E-5-82, Leningrad, 1982.
7. J. Edmonds, Systems of distinct representatives and linear algebra, *J. Res. Nat. Bur. Standards* **71B** (1967), 241–245.
8. F. Fich and M. Tompa, The parallel complexity of exponentiating polynomials over finite fields, *in* "Proc. 17th Annu. ACM Sympos. Theory of Computing," Providence, R.I., 1985, 38–47.
9. J. von zur Gathen, Parallel algorithms for algebraic problems, *SIAM J. Comput.* **13** (1984), 802–824.
10. J. von zur Gathen and E. Kaltofen, Factoring sparse multivariate polynomials, *J. Comput. System Sci.* **31** (1985).
11. J. von zur Gathen and E. Kaltofen, Factorization of multivariate polynomials over finite fields, *Math. Comp.* **45** (1985), 251–261.

12. J. von zur Gathen and G. Seroussi, Boolean circuits are exponentially more powerful than arithmetic circuits, manuscript, April 1985.

13. P. Griffiths and J. Harris, "Principles of algebraic geometry," Wiley, New York, 1978.

14. J. Heintz, "Definability and Fast Quantifier Elimination in Algebraically Closed Fields," Ph.D. thesis, Universität Zürich, 1982.

15. J. Heintz and M. Sieveking, Absolute primality of polynomials is decidable in random polynomial time in the number of variables, Lecture Notes in Comput. Sci. Vol. 115, pp.16–28, Springer-Verlag, Berlin/New York, 1981.

16. D. Hilbert, Ueber die Irreduzibilität ganzer rationaler Funktionen mit ganzzahligen Koeffizienten, *J. Reine Angew. Math.* **110** (1892), 104–129.

17. J.-P. Jouanolou, Théorèmes de Bertini et applications, preprint, Institut de Recherche Mathématique Avancée, Université Louis Pasteur, Strasbourg, 1982.

18. E. Kaltofen, A polynomial reduction from multivariate to bivariate integral polynomial factorization, *in* "Proc. 14th Annu. ACM Sympos. Theory of Computing," San Francisco, 1982, pp. 261–266.

19. E. Kaltofen, A polynomial-time reduction from bivariate to univariate integral polynomial factorization, *in* "Proc. 23rd Annu. IEEE Sympos. Foundations of Computer Science," Chicago, Ill, 1982, pp. 57–64.

20. E. Kaltofen, Effective Hilbert irreducibility, *in* "Proc. EUROSAM 1984," Cambridge, U.K., Lecture Notes in Computer Science Vol. 174, pp. 277–284, Springer-Verlag, New York/Berlin, 1984; *Inform. Contr.*, in press.

21. E. Kaltofen, Fast parallel absolute irreducibility testing, *J. Symbolic Comput.* **1** (1985), 57–67.

22. E. Kaltofen, Polynomial-time reductions from multivariate to bi- and univariate integral polynomial factorization, *SIAM J. Comput.* **14** (1985), 469–489.

23. E. Kaltofen, Computing with polynomials given by straightline programs II: Sparse factorization, *in* "Proc. 26th Annu. IEEE Sympos. Foundations of Computer Science," Portland, Ore., 1985, pp. 451–458.

24. E. Kaltofen, D. R. Musser, and B. D. Saunders, A generalized class of polynomials that are hard to factor, *SIAM J. Comput.* **12** (1983), 473–483.

25. H. T. Kung, New algorithms and lower bounds for the parallel evaluation of certain rational expressions and recurrences, *J. Assoc. Comput. Mach.* **23** (1976), 252–261.

26. S. Landau, Factoring polynomials over algebraic number fields, *SIAM J. Comput.* **14** (1985), 184–195.

27. S. Lang, "Introduction to Algebraic Geometry," Addison–Wesley, Reading, Mass., 1972.

28. S. Lang, "Fundamentals of Diophantine Geometry," Springer-Verlag, New York, 1983.

29. A. K. Lenstra, Factoring multivariate polynomials over finite fields, *J. Comput. System Sci.* **30** (1985), 235–248.

30. A. K. Lenstra, Factoring multivariate polynomials over algebraic number fields, *in* "Proc. Math. Foundations of Computer Science," Lecture Notes in Computer Science Vol. 176, pp. 389–396, Springer-Verlag, New York/Berlin, 1984.

31. A. K. Lenstra, Factoring multivariate integral polynomials, *Theoret. Comput. Sci.* **34** (1984), 207–213.

32. A. K. Lenstra, H. W. Lenstra, and L. Lovász, Factoring polynomials with rational coefficients, *Math. Ann.* **261** (1982), 515–534.

33. D. E. Muller and F. P. Preparata, Restructuring of arithmetic expressions for parallel evaluation, *J. Assoc. Comput. Mach.* **23** (1976), 534–543.

34. D. A. Plaisted, New *NP*-hard and *NP*-complete polynomial and integer divisibility problems, *Theoret. Comput. Sci.* **31** (1984), 125–138.

35. M. O. Rabin, Probabilistic algorithms in finite fields, *SIAM J. Comput.* **9** (1980), 273–280.

36. J. B. Rosser and L. Schoenfeld, Approximate formulas for some functions of prime numbers, *Illinois J. Math.* **6** (1962), 64–94.

37. T. Schönemann, Grundzüge einer allgemeinen Theorie der höheren Congruenzen, deren Modul eine reelle Primzahl ist, *J. Reine Angew. Math.* **31** (1846), 296–325.

38. J. T. Schwartz, Fast probabilistic algorithms for verification of polynomial identities, *J. Assoc. Comput. Mach.* **27** (1980), 701–717.

39. I. R. Shafarevich, Basic algebraic geometry, Grundlehren Band 213, Springer-Verlag, New York/Berlin, 1974.

40. A. Shamir, Factoring numbers in $O(\log n)$ arithmetic steps, *Inform. Process. Lett.* **8** (1979), 28–31.

41. R. Solovay and V. Strassen, A fast Monte-Carlo test for primality, *SIAM J. Comput.* **6** (1977), 84–85.

42. V. G. Sprindzhuk, Diophantine equations with unknown prime numbers, *Proc. Steklov Inst. Math.* **158** (1981), 180–196; English transl. *Proc. Steklov Inst. Math.* **158** (1983), 197–214.

43. V. Strassen, Berechnung und Programm, I, *Acta Inform.* **1** (1972), 320–335.

44. V. Strassen, Vermeidung von Divisionen, *J. Reine Angew. Math.* **264** (1973), 182–202.

45. V. Strassen, Polynomials with rational coefficients which are hard to compute, *SIAM J. Comput.* **3** (1974), 128–149.

46. V. Strassen, The computational complexity of continued fractions, *SIAM J. Comput.* **12** (1983), 1–27.

47. L. Valiant, S. Skyum, S. Berkowitz, and C. Rackoff, Fast parallel computation of polynomials using few processors, *SIAM J. Comput.* **12** (1983), 641–644.

48. B. L. van der Waerden, Zur algebraischen Geometrie. X. Ueber lineare Scharen von reduziblen Mannigfaltigkeiten, *Math. Ann.* **113** (1936), 705–712.

49. P. J. Weinberger, Finding the number of factors of a polynomial, *J. Algorithms* **5** (1984), 180–186.

50. O. Zariski, Pencils on an algebraic variety and a new proof of a theorem of Bertini, *Trans. Amer. Math. Soc.* **50** (1941), 48–70.

51. O. Zariski and P. Samuel, "Commutative Algebra," Vol. 1, Van Nostrand, Princeton, N. J., 1958.

52. H. Zassenhaus, On Hensel factorization, I, *J. Number Theory* **1** (1969), 291–311.

53. R. Zippel, Newton's iteration and the sparse Hensel algorithm, *in* "Proc. 1981 ACM Sympos. Symbolic Algebraic Computation," Utah, 1981, pp. 68–72.

Printed by the St. Catherine Press Ltd., Tempelhof 41, Bruges, Belgium