Factorization of polynomials, factoring polynomials – Since C.F. Gauss it is known that an arbitrary polynomial over a field or over the integers can be factored into irreducible factors, essentially uniquely (cf. also Factorial ring). For an efficient version of Gauss’ theorem, one asks to find these factors algorithmically, and to devise such algorithms with low cost.

Based on a precocious uncomputability result in [13], one can construct sufficiently bizarre (but still ‘computable’) fields over which even square-freeness of polynomials is undecidable in the sense of A.M. Turing (cf. also Undecidability; Turing machine). But for the fields of practical interest, there are algorithms that perform extremely well, both in theory and practice. Of course, factorization of integers and thus also in \( \mathbb{Z}[x] \) remains difficult; much of cryptography (cf. Cryptology) is based on the belief that it will remain so. The base case concerns factoring univariate polynomials over a finite field \( \mathbb{F}_q \) with \( q \) elements, where \( q \) is a prime power. A first step is to make the input polynomial \( f \in \mathbb{F}_q[x] \), of degree \( n \), square-free. This is easy to do by computing \( \gcd(f, \partial f / \partial x) \) and possibly extracting \( p \)th roots, where \( p = \text{char} \mathbb{F}_q \). The main tool of all algorithms is the Frobenius automorphism \( \sigma : R \to R \) on the \( \mathbb{F}_q \)-algebra \( R = \mathbb{F}_q[x]/(f) \). The pioneering algorithms are due to E.R. Berlekamp [1, 2], who represents \( \sigma \) by its matrix on the basis \( 1, x, x^2, \ldots, x^{n-1} \mod f \) of \( R \). A second approach, due to D.G. Cantor and H. Zassenhaus [3], is to compute \( \sigma \) by repeated squaring.

A third method (J. von zur Gathen and V. Shoup [7]) uses the so-called polynomial representation of \( \sigma \) as its basic tool. The last two algorithms are based on Gauss’ theorem that \( x^{q^i} - x \) is the product of all monic irreducible polynomials in \( \mathbb{F}_q[x] \) whose degree divides \( d \).

Thus, \( f_1 = \gcd(x^q - x, f) \) is the product of all linear factors of \( f \); next, \( f_2 = \gcd(x^{q^2} - x, f/f_1) \) consists of all quadratic factors, and so on. This yields the distinct-degree factorization \( (f_1, f_2, \ldots) \) of \( f \).

The equal-degree factorization step splits any of the resulting factors \( f_i \). This is only necessary if \( \deg f_i > i \). Since all irreducible factors of \( f_i \) have degree \( i \), the algebra \( R_i = \mathbb{F}_q[x]/(f_i) \) is a direct product of \( (at least two) \) copies of \( \mathbb{F}_q \). A random element \( a \) of \( R_i \) is likely to have Legendre symbol \( \pm 1 \) in some and \( -1 \) in other copies; then \( \gcd \left( a^{q^{(i-1)/2}} - 1, f_i \right) \) is a non-trivial factor of \( f_i \). To describe the cost of these methods, one uses fast arithmetic, so that polynomials over \( \mathbb{F}_q \) of degree up to \( n \) can be multiplied with \( O(n \log n \log \log n) \) operations in \( \mathbb{F}_q \), or \( O^*(n) \) for short, where the so-called ‘soft O’ hides factors that are logarithmic in \( n \). Furthermore, \( \omega \) is an exponent for matrix multiplication, with the current (in 2000) world record \( \omega < 2.376 \), from [4].

All algorithms first compute \( x^q \) modulo \( f \), with \( O^*(n \log q) \) operations in \( \mathbb{F}_q \). One can show that in an appropriate model, \( \Omega(\log q) \) operations are necessary, even for \( n = 2 \). The further cost is as follows:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>( \mathbb{F}_q[x]/(f) ) as</th>
<th>Cost in ( O^* )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Berlekamp</td>
<td>( \mathbb{F}_q )-vector space</td>
<td>( n^2 )</td>
</tr>
<tr>
<td>Cantor-Zassenhaus</td>
<td>multiplicative semi-group</td>
<td>( n^2 \log q )</td>
</tr>
<tr>
<td>von zur Gathen-Shoup</td>
<td>( \mathbb{F}_q )-algebra</td>
<td>( n^2 )</td>
</tr>
</tbody>
</table>

Table 1

For small fields, even better algorithms exist.

The next problem is factorization of some \( f \in \mathbb{Q}[x] \). The central tool is Hensel lifting, which lifts a factorization of \( f \) modulo an appropriate prime number \( p \) to...
one modulo a large power $p^k$ of $p$. Irreducible factors of $f$ will usually factor modulo $p$, according to the Chebotarev density theorem. One can then try various factor combinations of the irreducible factors modulo $p^k$ to recover a true factor in $\mathbb{Q}[x]$. This works quite well in practice, at least for polynomials of moderate degree, but uses exponential time on some inputs (for example, on the Swinnerton-Dyer polynomials). In a celebrated paper, A.K. Lenstra, H.W. Lenstra, Jr. and L. Lovász [11] introduced basis reduction of integer lattices (cf. LLL basis reduction method), and applied this to obtain a polynomial-time algorithm. Their reduction method has since found many applications, notably in cryptanalysis (cf. also Cryptology). A method in [9] promises an even faster factorizing method.

The next tasks are bivariate polynomials. It can be solved in a similar fashion, with Hensel lifting, say, modulo one variable, and an appropriate version of basis reduction, which is easy in this case. Algebraic extensions of the ground field are handled similarly.

Multivariate polynomials pose a new type of problem: how to represent them? The dense representation, where each term up to the degree is written out, is often too long. One would like to work with the sparse representation, using only the non-zero coefficients. The methods discussed above can be adapted and work reasonably well on many examples, but no guarantees of polynomial time are given. Two new ingredients are required. The first are efficient versions (due to E. Kaltofen and van zur Gathen) of Hilbert's irreducibility theorem (cf. also Hilbert theorem). These versions say that if one reduces many to two variables with a certain type of random linear substitution, then each irreducible factor is very likely to remain irreducible. The second ingredient is an even more concise representation, namely by a black box which returns the polynomial's value at any requested point. A highlight of this theory is the random polynomial-time factorization method in [10].

Each major computer algebra system has some variant of these methods implemented. Special-purpose software can factor huge polynomials, for example of degree more than one million over $\mathbb{F}_2$. Several textbooks describe the details of some of these methods, e.g. [8], [12], [5], [14].

Factorization of polynomials modulo a composite number presents some surprises, such as the possibility of exponentially many irreducible factors, which can nevertheless be determined in polynomial time, in an appropriate data structure.; see [6].

For a historical perspective, note that the basic idea of equal-degree factorization was known to A.M. Legendre, while Gauss had found, around 1798, the distinct-degree factorization algorithm and Hensel lifting. They were to form part of the eighth chapter of his 'Disquisitiones Arithmeticae', but only seven got published, due to lack of funding.

References


Joachim von zur Gathen
E-mail address: gathen@upb.de

MSC1991: 12D05