FACTORIZATION AND DECOMPOSITION OF POLYNOMIALS

Article for the Handbook of the Core of Algebra

Joachim von zur Gathen

June 15, 2000

Carl Friedrich Gauß proved that (multivariate) polynomials over a field or over the integers form a Unique Factorization Domain. The computational version of this fundamental result asks for an algorithm which, given a polynomial as input, finds its irreducible factors. And the complexity-theoretic version asks to do this as efficiently as possible. This question and its answers form one of the successful areas of *computer algebra*.

An easy kind of factorization is to calculate the greatest common divisor (gcd) of two polynomials. Euclid's algorithm—*the granddaddy of all algorithms* (Knuth (1981), p. 318)—does this for univariate polynomials over a field. It has numerous applications, for example in the Chinese Remainder Algorithm, for modular arithmetic and computation in algebraic extensions, in coding theory (Berlekamp-Massey algorithm), and for fast linear algebra on "sparse" matrices. The theory of subresultants gives important structural insights and leads to efficient algorithms, for example the modular methods for integer and for multivariate polynomials.

If $f_1, \ldots, f_r$ are the distinct irreducible monic factors of $f = f_1^{e_1} \cdots f_r^{e_r} \in F[x]$ over a field F, then $f_1 \cdots f_r$ is the *squarefree part* of f. It can be computed as $f/\gcd(f, f')$ in characteristic zero, while in characteristic $p > 0$, also pth roots have to be extracted. This is easy over all fields of practical interest, such as finite fields or function fields over them, but there are sufficiently bizarre (but still "computable") fields over which squarefreeness is undecidable, in the sense of Turing. The squarefree part is useful in the symbolic integration of rational functions.

Algorithms for the factorization of polynomials are built in a hierarchical way: One starts with univariate polynomials over finite fields, then over $\mathbb{Q}$, and then over algebraic extensions and of multivariate polynomials.

The first modern factorization methods for $f \in \mathbb{F}_q[x]$ of degree n, where $\mathbb{F}_q$ is a finite field with q elements, are due to Berlekamp. His motivation came from coding theory, and he found an algorithm based on linear algebra that uses $O^{\sim}(n^3 + nq)$ operations in $\mathbb{F}_q$. Here, the "soft Oh" notation $O^{\sim}$ means that we ignore factors of $\log n$. This is ok for small q, such as $q = 2$, which is particularly important in coding theory. But for large q, this is not polynomial in the "input length" $n \log q$ of f. A milestone was Berlekamp's invention of a polynomial time algorithm, using $O^{\sim}(n^3 + n \log q)$ operations. This is a probabilistic algorithm of Las Vegas type, whose output is always correct but whose running time is a random variable whose mean is given above.

A decade later, Cantor and Zassenhaus presented an algorithm which proceeds in two stages. In the *distinct-degree factorization* stage, the (squarefree) input is factored into a product of polynomials each of whose irreducible factors has the same degree. This is achieved by taking the gcd with $g_i = x^{q^i} - x$ for $i = 1, 2, \ldots$, and based on the fact that g_i is the product of all monic irreducible polynomials in $\mathbb{F}_q[x]$ whose degree divides i. This fact, and the squarefree and the distinct-degree factorization are in Gauß' *Nachlaß* (posthumous works).

The second *equal-degree factorization* stage factors a polynomial all of whose irreducible factors have the same degree. This is done by a probabilistic algorithm, the rudiments of which can already be found in Legendre's work. It remains an open question for the theory whether this can be achieved deterministically in polynomial time.

These algorithms have been improved in the 1990's and there is now a variety of algorithms which are optimal in a specific range of the proportion of degree n to field length $\log q$. The corresponding software can attack enormously large problems; in 2000, polynomials of degree one million (over $\mathbb{F}_2$) can be factored.

The next task is to factor integer polynomials. Gauß' Lemma reduces this to factoring in $\mathbb{Q}[x]$ and in $\mathbb{Z}$. The latter seems computationally hard (at our state of knowledge). Zassenhaus' algorithm for factoring in $\mathbb{Q}[x]$ works by first factoring modulo a (small) prime p, then applying *Hensel lifting* to get a factorization modulo a sufficiently large power of p, and finally trying out all combinations of the modulo factors to find the true factors. This works well for small inputs but uses exponential time. Lenstra, Lenstra and Lovász gave an efficient algorithm to find "short" vectors in integer lattices. Among many other computational applications, this also provides a polynomial-time algorithm for factoring in $\mathbb{Q}[x]$.

The next task are bivariate polynomials. Again, a judicious application of modular factorization, Hensel lifting, and an (efficient) factor combination yields an efficient factorization algorithm. Factorization over algebraic extensions can be handled in a similar way.

For polynomials in more than two variables, one may apply the same technology. However, the input length of the *dense representation* of polynomials, where the coefficient of each term up to the degree has to be specified, grows too quickly in size. It is more desirable to use concise forms such as the *sparse representation*, where only the nonzero coefficients are given. It is a remarkable achievement, mainly due to Kaltofen, to factor polynomials probabilistically in time polynomial in the input length for even more concise representations, namely by *arithmetic circuits* (a.k.a. straight-line programs) or by *black boxes*. The main ingredient are efficient versions of Hilbert's irreducibility theorem, as proved by Kaltofen and this author.

Factorizaton in $\mathbb{Z}_m[x]$, for $m \in \mathbb{N}$, exhibits a number of unusual properties. For example, there are polynomials with exponentially many factorizations into irreducible polynomials, but these can all be represented by a polynomial-sized data structure.

Detailed discussions, references, and reports on implementations can be found in von zur Gathen & Gerhard (1999).

The *composition* of two univariate polynomials $g, h \in F[x]$ is $g \circ = g(h) \in F[x]$. In the *decomposition problem*, we are given $f \in F[x]$ and ask whether there exist $g, h \in F[x]$ so that $f = g \circ h$ and $2 \leq \deg g < \deg f$. The first polynomial-time algorithm was presented by Kozen und Landau in 1986, and the fastest one, by this author, uses $O(n \log^2 n \log\log n)$ operations in F, where $n = \deg f$ and $r = \deg g$ is prescribed, with $\operatorname{char} F \nmid r$ (the so-called *tame case*). Ritt showed in 1922 that such decompositions are essentially unique. Many generalizations have been studied: rational functions by Zippel, algebraic functions by Kozen, Landau, and Zippel, and multivariate decompositions by several authors. The problem has important applications in the simplification of parametrizations of algebraic curves and of the inverse kinematic equations in robotics.

References

JOACHIM VON ZUR GATHEN & JÜRGEN GERHARD (1999). *Modern Computer Algebra*. Cambridge, UK.

DONALD E. KNUTH (1981). *The Art of Computer Programming, vol.2, Seminumerical Algorithms*. Reading MA, 2nd edition.

Joachim von zur Gathen
Fachbereich Mathematik-Informatik
Universität Paderborn
33095 Paderborn, Germany
gathen@upb.de
http://www-math.upb.de/~gathen/