Ann. Rev. Comput. Sci. 1988. 3 : 317–47

ALGEBRAIC COMPLEXITY THEORY

Joachim von zur Gathen

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

1. INTRODUCTION

Algebraic complexity theory investigates the computational cost of solving problems with an algebraic flavor. Several cost measures are of interest. We consider arithmetic circuits, which can perform the (exact) arithmetic operations $+$, $-$, $*$, $/$ at unit cost, and take their size (=sequential time) or their depth (=parallel time) as cost functions. This is a natural "structured" model of computation for the computation of rational functions over any ground field. If inputs can be represented by strings over a finite alphabet—as is the case for polynomials over $\mathbb{Q}$—we can also use a "general" model such as Turing machines or Boolean circuits. The *complexity* of a problem is the minimal cost (in the measure under consideration) sufficient to solve it. Its investigation splits into two tasks, which require very different methodologies. The first task is the design of good algorithms, proving upper bounds on the complexity. The second, usually more difficult task, is the discovery of intrinsic properties ("invariants") of problems, and estimation of the progress that an algorithm can make, say step by step, in terms of these invariants, thus proving lower bounds on the cost of any conceivable algorithm.

Within the wider field of complexity theory, few areas have had similar success in establishing matching upper and lower bounds on the complexity of many natural problems. Our subject takes its questions from computer science, mainly numerical and symbolic computation. The approach is mathematical, and some problems, by their nature, require fairly sophisticated methods.

Classifying our problems under the perspective of *polynomial time*, they fall into three categories. In the first category (Sections 2 and 3

8756–7016/88/1115–0317$02.00

on polynomial arithmetic, and 4 on bilinear functions), polynomial-time algorithms are easy, e.g. for interpolation by a polynomial. For this example, a nontrivial algorithm reduces the typical running time of $O(n^2)$ to essentially linear, e.g. to $O(n \log n)$ (depending on the model used). Then, in the powerful "nonscalar model," the lower bound $\Omega(n \log n)$ shows this algorithm to be asymptotically optimal.

In the second category (Sections 5 on the Valiant Hypothesis, 7 on factorization of polynomials, and 9 on permutation groups), algorithms clearly exist, but polynomial time is a nontrivial question. An example is the factorization of univariate polynomials over finite fields, where a trivial exhaustive search works in exponential time; polynomial-time polynomial factorization is not obvious. Valiant's arithmetic analogue of the Boolean question "$P \stackrel{?}{\neq} NP$" gives us specific polynomials (e.g. the permanent) to focus on for "polynomial time vs exponential time"; this theory has no definite answers yet.

For problems of the third category, no algorithm is obvious at all (Section 8 on arithmetic theories). Indeed, the question whether a system of polynomials has a solution is undecidable over $\mathbb{Q}$. The nontrivial decision procedures over $\mathbb{R}$ or $\mathbb{C}$ use doubly exponential running time, and there are matching lower bounds.

A different criterion of quality is introduced in Section 6: fast parallel algorithms (of poly-logarithmic depth) with a small (i.e. polynomial) number of processors. This point of view is also mentioned in Sections 7, 8, and 9.

Many of the methods and algorithms have found practical application in the highly successful Computer Algebra Systems, "one of our society's technical wonders" (Caviness), for which Buchberger et al (1983), Caviness (1986), and Kaltofen (1987) give surveys. This exposé excludes the connections to this and other areas of application, such as numerical computation, signal processing, and fixed-connection parallelism.

Throughout the paper, we use a "worst-case" approach; the "average-case complexity of problems occurring in practice" is even hard to formalize, and meaningful results in this direction would be very interesting.

Writing this article has been greatly facilitated by the excellent survey of Strassen (1984). Several of our topics are treated in depth in textbooks, quoted near the beginning of the respective section. In most sections, a particular algorithm or proof is meant to convey the flavor of the subject, and I have tried to identify one central "open question"; of course, these choices are quite subjective.

Most areas had established their problems, paradigms, and fundamental methods by the mid-1970s. The exceptions are permutation groups and

factorization of polynomials, where most of the polynomial-time results date from this decade, and the parallel algorithms.

2. POLYNOMIAL ARITHMETIC

The birthday of "algebraic complexity theory" is often considered to be in the year 1954. In a pioneering paper, Ostrowski (1954) asked: "Is Horner's rule optimal for the evaluation of a polynomial?" He proposed the model of *nonscalar complexity*, formalized by Strassen (1972b, 1973a): In order to compute a polynomial or a rational function $f \in F(x_1, \ldots, x_n)$ over a field F, an *arithmetic circuit* (or straight-line program) α over $F \cup \{x_1, \ldots, x_n\}$ is allowed to fetch the inputs $x_1, \ldots, x_n$, arbitrary constants from F, and to perform an arithmetic operation $+, -, *, /$ on previously computed results. Thus at each node of α, some rational function is computed. (Division by the rational function zero is ruled out.)

Most of the results mentioned in this and the next two sections can be found in Borodin & Munro (1975); some of the algorithms also appear in other texts—e.g. Aho et al (1974), Knuth (1981), and Buchberger et al (1983).

A circuit α *computes* f if f is one of the results in α. The *scalar operations* are fetching inputs or constants, $+$ and $-$, and also $u * v$ or u/v if $v \in F$ is a scalar. The number of nonscalar operations in α is the *nonscalar cost* of α, and the *nonscalar complexity* $L_{ns}^F(f)$ is the minimal nonscalar cost of circuits computing f. (Counting also $+$, $-$, and scalar $*$ and $/$, we obtain the *total complexity* $L_{tot}^F(f)$.)

As an example, for $f = x_1^2 + x_2^2$ we obviously have $L_{ns}^F(f) \leq 2$ and $L_{tot}^F(f) \leq 3$. If $i = \sqrt{-1} \in F$, then $L_{ns}^F(f) = L_{ns}^F((x_1 + i * x_2) * (x_1 - i * x_2)) \leq 1$. In fact, we have equality in the last two bounds, and $L_{ns}^F(f) = 2$ if $i \notin F$. From now on, we usually leave out the superscript F. Ostrowski's question refers to the evaluation of a general polynomial $f = \sum_{0 \leq j \leq n} a_j x^j \in F[x, a_0, \ldots, a_n]$, where both x and the coefficients $a_0, \ldots, a_n$ are treated as inputs. Horner's rule shows $L_{ns}(f) \leq n$ (over any F), and the problem was: Does equality hold?

Pan (1966) answered this in the affirmative: $L_{ns}(f) = n$, where $f = \sum_{i \leq n} a_i x^i$. His method, later generalized by Winograd (1970), Strassen (1972a, 1973a), and Hartmann & Schuster (1980), introduces a linear substitution of the variables $a_1, \ldots, a_n$, which makes the first nonscalar operation of the hypothetical circuit trivial and reduces the complexity by at most one. Pan then verifies that at least n such substitutions are required to make f linear.

The central problem *multiplication of polynomials* is the task of computing $\{c_0, \ldots, c_{2n}\}$ where $c_k = \sum_{i+j=k} a_i b_j \in F[a_0, \ldots, a_n, b_0, \ldots, b_n]$ is a coefficient

of the product of the polynomials a and b with coefficients a_i and b_j. Denoting by $M_{\#}(n) = L_{\#}(\{c_0, \ldots, c_{2n}\})$ its complexity (with $\#$ either "*ns*" or "*tot*", and an obvious extension of L to sets of functions), the school method shows $M_{tot}(n) = O(n^2)$.

In the nonscalar model, we may simply evaluate a and b at $2n+1$ points in F, multiply the $2n+1$ pairs of values, and interpolate. Evaluation and interpolation have cost zero, so that $M_{ns}(n) \leq 2n+1$. (This assumes that F has more than $2n$ elements.) The algorithm providing a substantial improvement for M_{tot} is fundamental in several areas of computer science: the *Discrete Fourier Transform*, introduced by Cooley & Tukey (1965) into computer science. It follows the same approach as above, only that the special m points $\omega^0, \omega^1, \omega^2, \ldots, \omega^{m-1}$ are chosen, where $\omega \in F$ is a primitive mth root of unity and m the smallest power of 2 greater than $2n$. The kick is that now evaluation and interpolation can be performed with a total of $O(n \log n)$ operations, using a divide-and-conquer technique. Thus $M_{tot}(n) = O(n \log n)$. Schönhage & Strassen (1971) show that this approach also works when such an ω does not exist in F, proving $M_{tot}(n) = O(n \log n \log \log n)$ for any field F (see Schönhage 1977; Cantor & Kaltofen 1987). Their method also provides the fastest known Boolean circuits for integer multiplication. This "Fast Fourier Transform" has many generalizations and applications in signal transmission and processing (see Beth 1984).

The problems of *squaring*, *inversion* mod x^n (given $a \in F[x]$ with $a(0) = 1$, compute $b \in f[x]$ so that $ab \equiv 1 \bmod x^n$), and *division with remainder* (given $a, b \in F[x]$ with $b \neq 0$, compute $q, r \in F[x]$ so that $a = qb + r$ and $\deg r < \deg b$) are all equivalent to multiplication (Sieveking 1972; Strassen 1973a; Borodin & Moenck 1974; Kung 1974). This means that if one of these four problems can be solved with $s(n)$ operations, they all can be performed with $O(s(n))$ operations (provided s is "reasonable"). In particular, L_{ns} is $O(n)$, and L_{tot} is $O(n \log n \log \log n)$ for these four problems.

Evaluation at many points computes the values of a polynomial $a \in F[x]$ of degree less than n at n points. The inputs to the problem are the coefficients of a and the points $x_1, \ldots, x_n$. Borodin & Moenck (1974) use a divide-and-conquer technique to compute the polynomials

$$p_1 = \prod_{1 \leq i \leq n/2} (x - x_i), \quad p_2 = \prod_{n/2 < i \leq n} (x - x_i)$$

with a binary tree of multiplications. Two divisions with remainder yield

$$a = q_k p_k + a_k, \quad \deg a_k < \deg p_k \quad \text{for} \quad k = 1, 2.$$

Now recursive evaluation of a_1 at the first half of the points, and a_2 at the second half, provides the required values, since e.g.

$$a(x_1) = q_1(x_1)p_1(x_1)+a_1(x_1) = a_1(x_1).$$

Leaving out the subscript "*ns*" or "*tot*" and assuming that $M(n)$ is "reasonable" (which the mentioned bounds are), the time estimate $O(M(n)\log n)$ follows (see also Borodin & Munro 1971; Fiduccia 1972; Strassen 1973a).

The same bound holds for *interpolation* (Horowitz 1972), more generally the *Chinese remainder problem* (which includes Hermite interpolation) (Borodin & Moenck 1974), computing the *elementary symmetric functions* and the *greatest common divisor*, more generally the *Euclidean representation*, which consists of all the quotients computed in the Euclidean algorithm (Knuth 1970; Brown 1971; Schönhage 1971; Moenck 1973; Strassen 1983).

Open question Is M_{tot} nonlinear? Is the total complexity of computing the Discrete Fourier Transform nonlinear? Is L_{tot} larger than L_{ns} by more than a constant factor for evaluation at many points, and the problems of the last paragraph?

The computation of a power x^n by multiplications only corresponds to the computation of n by *addition chains*. Trivially, the minimal length of such chains lies between $\log n$ and $2\log n$. The study of these chains was proposed by Scholz (1937), and in fact, $\log n + o(\log n)$ multiplications are sufficient (Brauer 1939; Erdös 1960; Schönhage 1975; Knuth 1981).

The computation of $x^{31} = x^{2^5}/x$ is somewhat easier with divisions than without. However, Strassen's (1973b) result on avoiding divisions shows that they are of limited help: If a polynomial of degree d can be computed with s nonscalar operations, it can be computed with a total number of $s\cdot d(d-1)/2$ operations, and also with $7ds$ operations. This result has been extended by Kaltofen (1986) to rational functions, computing the numerator and denominator polynomials separately.

As an aside, we note that—as in any circuit-based complexity theory—for the asymptotic complexity $L_{ns}(f_n)$ of families $f = (f_n)_{n\in\mathbb{N}}$ with $f_n \in F(x_1, \ldots, x_n)$ we have to consider families $\alpha = (\alpha_n)_{n\in\mathbb{N}}$ of arithmetic circuits, with varying input length. How can we compute a "description" of α_n, given n, say on a Turing machine? Fortunately, we can avoid this issue of *uniformity* (see Borodin 1977; Ruzzo 1981 for Boolean circuits), since all upper bounds (i.e. algorithms) quoted in this survey are uniform, and the lower bounds do not assume any uniformity.

3. NONLINEAR LOWER BOUNDS

Once all the amazingly fast $O(n\log^k n)$ algorithms of Section 2 were found, an obvious question was: Can we improve these upper bounds further? Or are there matching lower bounds? The analogous question is ubiquitous in

many areas of theoretical computer science and has found few satisfactory answers. One of the beauties of our subject is that we have a complete answer to some of these fundamental questions.

In a seminal paper, Strassen (1973a) uses the degree $\deg X$ of an algebraic variety $X \subseteq F^m$, a notion from algebraic geometry. (Formally, one should assume that F is algebraically closed and work in projective space, but the final complexity results apply to any infinite field.)

$$X = \{a \in F^m : f_1(a) = \cdots = f_k(a) = 0\}$$

is the set of zeroes of some polynomials $f_1, \ldots, f_k \in F[x_1, \ldots, x_m]$, and $\deg X$ is the maximal size of a finite intersection $X \cap L$ with any affine linear subspace L of F^m. If $X = \{a \in F^m : f(a) = 0\}$ is a hypersurface, with $f \in F[x_1, \ldots, x_m]$ squarefree, then $\deg X = \deg f$; thus the degree of polynomials is generalized by this notion of degree. Consider an arbitrary circuit α with inputs $x_1, \ldots, x_n$ and of nonscalar cost s, the functions $u_1, \ldots, u_s \in F(x_1, \ldots, x_n)$ computed at the nonscalar nodes of α, and the graph X_α of α:

$$X_\alpha = \{(a_1, \ldots, a_n, a_{n+1}, \ldots, a_{n+s}) \in F^{n+s} : a_{n+i} = u_i(a_1, \ldots, a_n) \quad \text{for} \quad 1 \leq i \leq s\}.$$

For simplicity, we assume that only multiplications occur; the following argument goes through with minor modifications if divisions are allowed. For each $i \leq s$, there exist $\lambda_{i0}, \lambda_{i1}, \ldots, \lambda_{i,n+i-1}, \mu_{i0}, \mu_{i1}, \ldots, \mu_{i,n+i-1} \in F$ such that

$$g_i = a_{n+i} - \left(\lambda_{i0} + \sum_{1 \leq j \leq n} \lambda_{ij} x_j + \sum_{1 \leq j < i} \lambda_{i,n+j} a_{n+j}\right) * \left(\mu_{i0} + \sum_{1 \leq j \leq n} \mu_{ij} x_j + \sum_{1 \leq j < i} \mu_{i,n+j} a_{n+j}\right) = 0$$

on X_α. Thus

$$X_\alpha = \{a \in F^{n+s} : g_1(a) = \cdots = g_s(a) = 0\}$$

is the intersection of the s quadratic hypersurfaces $Y_i = \{a : g_i(a) = 0\}$. Bezout's theorem states that

$$\deg(X \cap Y) \leq \deg X \cdot \deg Y$$

for varieties $X, Y \subseteq F^m$. In fact, equality holds if multiplicities are properly accounted for. This was stated by Bezout in the early 19th century for plane curves, and the first rigorous proof is due to van der Waerden (1928). In our case, it follows that

$$\deg X_\alpha = \deg \bigcap_{1 \le i \le s} Y_i \le 2^s.$$

Now we consider the graph G of evaluation at many points:

$$G = \left\{ (a_0, \ldots, a_{n-1}, x_1, \ldots, x_n, y_1, \ldots, y_n) \in F^{3n} : \forall i \le n \, y_i = \sum_{0 \le j < n} a_j x_i^j \right\}.$$

We determine a linear space $L = \{a\} \times F^n \times \{y\} \subseteq F^{3n}$ of dimension n, by fixing some $a = (a_0, \ldots, a_{n-1})$ and $y = (y_1, \ldots, y_n)$ such that each polynomial $\sum a_j x^j - y_i \in F[x]$ has exactly $n-1$ distinct roots in F; such a and y clearly exist. By definition, it follows that $\deg G \ge (n-1)^n$ (in fact, equality holds).

Now assume that α is a program of nonscalar cost s that computes evaluation at many points. Then G is a projection of X_α, and a general theorem about the degree of projections implies that $\deg X_\alpha \ge \deg G$. Together we find the lower bound

$$s \ge \log \deg X_\alpha \ge \log \deg G \ge n \log (n-1).$$

Strassen's (1973a) general theorem states that $L_{ns}(f_1, \ldots, f_r) \ge \log \deg G$, where $G \subseteq F^{n+r}$ is the graph of $f_1, \ldots, f_r$. He obtains the same asymptotically optimal $\Omega(n \log n)$ bound for the nonscalar complexity of interpolation, the computation of all power sums $\sum_{1 \le i \le n} x_i^j$ $(1 \le j \le n)$, and computing all elementary symmetric functions (in n variables). More elementary proofs concerning Strassen's approach are in Schönhage (1976), Schnorr (1981), and Heintz (1983).

Open question What is the complexity of these problems over finite fields?

Strassen (1976) adapts his lower bounds for evaluation at many points and interpolation to finite fields. However, Mihaĭljuk (1979) computes the elementary symmetric functions in linear time.

The nonscalar operation count is, admittedly, a somewhat optimistic model of computation. It might be realistic when the multiplication of data-dependent objects is much more expensive that their addition or multiplication by fixed scalars, e.g. when the inputs are multiple-precision real numbers, rational functions, or matrices. However, the main justification—to the author, at least—is the fact that the asymptotically matching upper and lower bounds tell us that we have the ultimate algorithms, in the sense of this model. These lower bounds apply trivially to the total operations count, but unfortunately no better lower bounds are known in this more realistic model, and the upper bounds are higher by a factor of $\log n$ (or $\log n \log \log n$).

For about a decade, no matching nonlinear upper and lower bounds were known for single funtions. Schnorr (1981) proves that

$L_{ns}(\sum_{1\le i\le k} x_i^n y^i) \ge \frac{1}{4}k\log n$ if $k \le n^{1/4}$. Baur & Strassen (1982) show that if $f \in F[x_1, \ldots, x_n]$ has nonscalar complexity s, then the set of all partial derivatives $\Delta f = \{\partial f/\partial x_1, \ldots, \partial f/\partial x_n\}$ has complexity at most $3s$. (Also, $L_{tot}(\Delta f) \le 5L_{tot}(f)$.) They use this general result to show that the "middle" elementary symmetric function alone has nonscalar complexity $\Omega(n\log n)$.

If a polynomial of degree at most n is given by its values at $n+1$ points, then calculating the value at a new (indeterminate) point requires $\Omega(n\log n)$ nonscalar operations (Stoß 1985). There are several ways of representing polynomials, including the coefficient sequence or a list of values. No representation is known for which the basic operations like addition, multiplication, evaluation (at many points), and gcd can be performed in linear time.

In Section 2, we noted that Horner's rule is optimal for evaluating a polynomial with indeterminate coefficients. What about specific polynomials $f = \sum_{0\le i\le n} a_i x^i \in F[x]$, where only x is treated as an indeterminate? Paterson & Stockmeyer (1973) split f into about $m = \lceil\sqrt{n}\rceil$ chunks of degree less than m each:

$$f = \sum_{0\le i\le m} f_i \cdot (x^m)^i, \quad f_i = \sum_{0\le j<m} a_{im+j} x^j.$$

Calculating first $x^2, x^3, \ldots, x^m$ and then using Horner's rule, this shows that any f can be evaluated with $2\sqrt{n}+1$ nonscalar multiplications. In fact, they show that $\sqrt{2n}+O(\log n)$ is sufficient. They also prove that "almost all" polynomials in $F[x]$ require at least $\sqrt{n}$ nonscalar operations. Here, "almost all" is in the strong sense that there exists a nonzero test polynomial $\tau \in \mathbb{Z}[A_0, \ldots, A_n]$ such that $\tau(a_0, \ldots, a_n) = 0$ for any exceptional polynomial $f = \sum a_i x^i$; in particular, the set of exceptions has measure zero (say, over $\mathbb{C}$). Any polynomial $f \in \mathbb{C}[x]$ whose coefficients are algebraically independent over $\mathbb{Q}$ has $L_{ns}^{\mathbb{C}}(f) \ge \sqrt{n}$, and there are "0-1-polynomials" f—i.e. with coefficients only 0 or 1—with $L_{ns}^{\mathbb{C}}(f) = \Omega((n/\log n)^{1/2})$ (Lipton 1975; Schnorr 1978; Schnorr & Van de Wiele 1980; Heintz & Schnorr 1982). Strassen (1974) and Schnorr (1978) push this further by exhibiting *specific* polynomials that are hard to compute. For example:

$$L_{ns}^{\mathbb{C}}\left(\sum_{0\le j\le n} 2^{2^j} x^j\right) = \Omega((n/\log n)^{1/2}),$$

$$L_{ns}^{\mathbb{C}}(\varepsilon_n) = \Omega((n/\log n)^{1/2}) \quad \text{for} \quad \varepsilon_n = \sum_{0\le j\le n} \exp(2\pi i/2^j) x^j.$$

The elegant method of Heintz & Sieveking (1980) deals with algebraic coefficients, and shows, for example

$$L_{ns}^{\mathbb{C}}\left(\sum_{0\le j\le n} j^r x^j\right) = \Omega(n^{1/2}/\log n) \quad \text{for} \quad r\in\mathbb{Q}\backslash\mathbb{Z},$$

$$L_{ns}^{\mathbb{C}}\left(\sum_{0\le j\le n} a^{1/j}x^j\right) = \Omega((n/\log n)^{1/2}) \quad \text{for} \quad a\in\mathbb{R},\quad a>0,\quad a\neq 1$$

(see von zur Gathen & Strassen 1980).

Similar results about evaluation of general and specific polynomials are available for counting all multiplications, where the complexity is $\lceil (n+1)/2\rceil$ for a general polynomial (Motzkin 1955; Belaga 1958; Winograd 1970) and $\Omega(n/\log n)$ for ε_n. For the total complexity L_{tot}, all 0-1-polynomials of degree n have cost $O(n/\log n)$ (Savage 1974), and most have complexity $\Omega(n/\log n)$ (Schnorr & Van de Wiele 1980). For the *additive complexity* L_+, one counts only $+$ and $-$. Then, for example, $L_+^{\mathbb{C}}(\varepsilon_n) = \Omega(n/\log n)$, and

$$\Omega((n/\log n)^{1/2}) \le L_+(f) \le O(n/\log n)$$

for most 0-1-polynomials f (Belaga 1958, 1961; Pan 1966; Borodin & Cook 1976; Schnorr & Van de Wiele 1980; Risler 1985). Stoß (1986) gives a unified treatment of many of these results, including the case of positive characteristic.

Strassen (1983) extends the model to *algebraic decision trees*, where branching according to tests "$a \stackrel{?}{=} 0$" is allowed. He proves an amazing "uniform optimality" result for a fast version of Euclid's algorithm for polynomials. Ben-Or (1983) works over $\mathbb{R}$, allows tests "$a \stackrel{?}{\ge} 0$", replaces algebraic geometry by real cohomology, and proves for example that $\Omega(n\log n)$ arithmetic operations and tests are required to decide whether inputs $x_1,\ldots,x_n$ are pairwise distinct.

4. BILINEAR PROBLEMS

The bulk of computer time used up in scientific computing is spent on problems from linear algebra, such as calculating matrix-vector and matrix-matrix products. How fast can we calculate such products?

$$f = \sum_{\substack{1\le i\le m\\ 1\le j\le n}} a_{ij}x_iy_j \in F[x_1,\ldots,x_m,y_1,\ldots,y_n], \quad \text{with each } a_{ij}\in F,$$

is called a *bilinear form* in the x_i, y_j. An an example, $N\times N$-matrix multiplication is the task of computing the bilinear forms $f_{ik} = \Sigma_j x_{ij}y_{jk}$ in $2N^2$ variables x_{ij}, y_{jk} $(1\le i,j,k\le N)$. De Groote (1987) gives a thorough and

precise introduction to the field of bilinear complexity, Winograd (1980) concentrates on results applicable in the area of signal processing, and Heintz (1985) provides an exposé of the central results.

Consider the problems of matrix multiplication, matrix inversion, computing the determinant or the characteristic polynomial, calculating the LR-decomposition of a matrix, and, for $F = \mathbb{C}$, the QR-decomposition and unitary transformation to upper Hessenberg form. The standard algorithms for these problems use $O(n^3)$ operations, and indeed Gaussian elimination is optimal if only row and column operations are used. It turns out that all problems have the same asymptotic complexity (up to constant factors), so that a fast algorithm for one of them immediately gives fast algorithms for all of them (van der Waerden 1938; Strassen 1969, 1973b; Schönhage 1973; Bunch & Hopcroft 1974; Baur & Strassen 1982; Keller-Gehrig 1985).

Strassen (1969) surprised the world by showing that Gaussian elimination is not optimal. He devised a clever scheme for 2×2-matrix multiplication, using only 7 multiplications (and 18 additions). Recursive application to $n \times n$-matrices yields an algorithm of total cost $O(n^\tau)$ with $\tau = \log_2 7 < 2.808 < 3$.

That paper started the area of *bilinear complexity*. By Strassen's (1973b) technique for avoiding divisions, we may assume that an optimal algorithm uses $+$ and $*$ only. Furthermore, the nonscalar and the total complexities are asymptotically equal (up to constant factors).

The *bilinear complexity* $R(\phi)$ of a set $\phi = \{f_1, \ldots, f_p\} \subseteq F[x_1, \ldots, x_m, y_1, \ldots, y_n]$ of bilinear forms is the smallest number of "bilinear multiplications"

$$(\Sigma \lambda_i x_i) * (\Sigma \mu_j y_j) \quad \text{with} \quad \lambda_i, \mu_j \in F$$

sufficient to compute ϕ (i.e. multiplications of x_i by x_j are disallowed, and we allow linear combinations of these products for free to obtain ϕ). Then

$$L_{ns}(\phi) \leq R(\phi) \leq 2L_{ns}(\phi)$$

(Strassen 1973b). If $f_k = \Sigma_{i,j} a_{ijk} x_i y_j$, we can identify ϕ with the *tensor*

$$t = \sum_{i,j,k} a_{ijk} x_i \otimes y_j \otimes z_k \in F^m \otimes F^n \otimes F^p,$$

where the x_i, y_j, z_k form a basis of F^m, F^n, F^p, respectively. A *simple* $n \times m \times p$-*tensor* is the form $u \otimes v \otimes w$, and the *rank* of t is the smallest number of simple tensors sufficient to express t:

$$\operatorname{rank}(t) = \min\left\{r \in \mathbb{N} : \exists u_1, \ldots, u_r \in F^m, \right.$$

$$\left. v_1, \ldots, v_r \in F^n, w_1, \ldots, w_r \in F^p : t = \sum_{1 \le i \le r} u_i \otimes v_i \otimes w_i \right\}$$

(Gastinel 1971 implicitly; Strassen 1972a, 1973b; Fiduccia 1972). This notion generalizes the rank of matrices (where one has $p = 1$), is independent of the bases chosen, and

$$R(\phi) = \operatorname{rank}(t).$$

Thus the algorithmic notion of bilinear complexity equals the algebraic notion of rank. Some bounds on the bilinear complexity are:

$R(2 \times 2 \text{ times } 2 \times n\text{-matrix multiplication}) \geq 7n/2$ (Hopcroft & Kerr 1971),
$R(n \times n\text{-matrix multiplication}) \geq 2n^2 - 1$ (Brockett & Dobkin 1978),
$R(\text{complex multiplication}) = 3$ (Winograd 1970),
$R(\text{quaternion multiplication}) = 8$ (de Groote 1975; van Leeuwen & van Emde Boas 1978), and
$R(\text{multiplication in } F[x]/(f)) = 2 \deg f - r$ (Fiduccia & Zalcstein 1977; Winograd 1977),

where $f \in F[x]$ has r distinct irreducible factors. In particular, multiplication in an extension field of degree n over F has complexity $2n-1$. From the Kronecker normal form one can read off the rank of $m \times n \times 2$-tensors (Grigoryev 1978; Ja'ja' 1979).

The first of these results implies that Strassen's (1969) algorithm is optimal for 2×2-matrices. In fact, de Groote (1978) shows that all optimal algorithms for 2×2-matrices are "equivalent" to Strassen's. All these lower bounds on the bilinear complexity $R(A)$ of multiplication in a finite-dimensional (associative) algebra A (except the quaternions) are subsumed in the elegant result of Alder & Strassen (1981):

$$R(A) \geq 2 \dim_F A - r,$$

where r is the number of maximal two-sided ideals of A.

Strassen (1973b) pointed out the importance of the "direct sum conjecture": $R(A \oplus B) = R(A) + R(B)$; after Schönhage's (1981) negative result on a related conjecture, this is also believed to be false (Strassen 1987).

The bilinear complexity of multiplying two polynomials in $F[x]$ of degree at most n is $2n+1$ if F has at least $2n$ elements, and it is at most slightly more than linear over small fields (Grigoryev 1978; Lempel et al 1983). In 1973, Brockett & Dobkin (1978) discovered a connection between bilinear complexity and error-correcting codes (see also Lempel & Winograd 1977),

which leads over $F = \mathbb{Z}_2$ to the lower bound $3.52n$ for polynomial multiplication (Brown & Dobkin 1980), and of $2.5n^2 - o(n^2)$ for matrix multiplication (Bshouty 1987).

For almost a decade after Strassen (1969), no further insight was obtained on the exponent $\omega = \omega_F \geq 2$ of matrix multiplication, defined as

$$\omega = \inf\{\tau \in \mathbb{R} : R(n \times n\text{-matrix multiplication}) = O(n^\tau)\}.$$

Then Pan (1978) got the ball rolling again, Bini et al (1979) introduced the powerful new technique of "approximation algorithms," and Schönhage (1981) found his famous "τ-theorem," vastly generalizing the result of Hopcroft & Musinski (1973) that if $R(m \times n \text{ times } n \times p - \text{matrix multiplication}) \leq r$, then $(mnp)^{\omega/3} \leq r$. Some of these developments took place at a memorable Oberwolfach meeting on complexity theory in October 1979, and the end result was the estimate $\omega < 2.496$ (Coppersmith & Winograd 1982). In Pan (1984), one of the participants gives his account.

Yet another breakthrough by Strassen (1987) led to the current world record $\omega < 2.376$ (Coppersmith & Winograd 1987).

Open question Decide whether $\omega = 2$.

The exponent η for solving systems of linear equations satisfies $\eta \leq \omega$. It is not known whether $\eta = \omega$.

Figure 1 shows some upper bounds on ω with the approximate time they were announced. Between b and f, at least seven other short-lived world records are not shown. Pan (1984) gives a more complete figure.

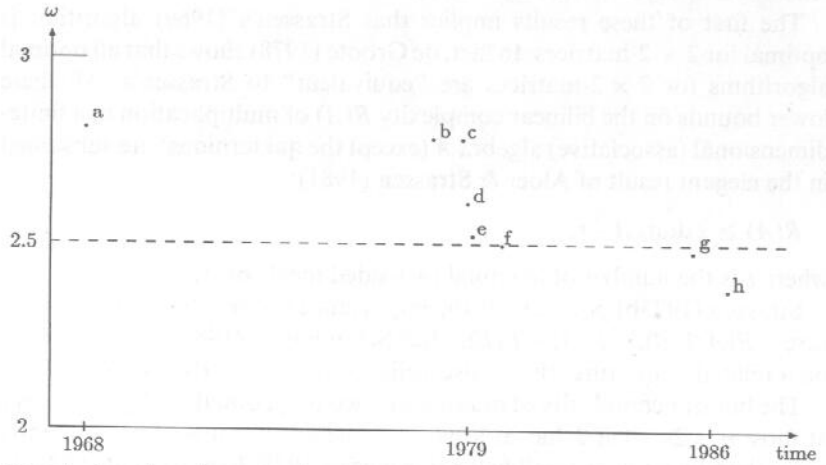

Figure 1 Some bounds on ω and announcement dates. a: Strassen 1968, 2.808. b: Pan 1978, 2.781. c: Bini et al 1979, 2.780. d: Schönhage 1979, 2.609. e: Pan 1979, 2.522. f: Coppersmith & Winograd 1980, 2.498. g: Strassen 1986, 2.4785. h: Coppersmith & Winograd 1986, 2.376.

5. THE VALIANT HYPOTHESIS

The most important development of the 1970s in computer science was the theory of P vs NP (Cook 1971; Karp 1972; see also Levin 1973). This figured prominently among the achievements for which Cook (in 1982) and Karp (in 1985) received the ACM Turing Award, the highest research prize in computer science. The central unsolved problem here is the *Cook Hypothesis*: "$P \neq NP$". Valiant (1979a, 1982) translates this approach into the setting of algebraic complexity theory. This is one of the fundamental contributions for which Valiant received in 1986 the Nevanlinna medal (see Strassen 1986), with which the mathematical community honors once every four years outstanding achievements in theoretical computer science.

Valiant's analog of P is the set P_F of *p-computable* families $f = (f_n)_{n\in\mathbb{N}}$ of polynomials $f_n \in F[x_1, \ldots, x_n]$, with $L_{tot}(f_n)$ and $\deg f_n$ polynomially bounded. By Strassen (1973b), $L_{tot}(f_n) = O((\deg f_n \cdot L_{ns}(f_n))^2)$, so that for considerations of polynomial time the total and nonscalar costs are equally informative. The Boolean polynomial-time reductions are replaced by "*p*-projections" $f \leq g$, where f is obtained from g by substituting variables and constants:

$$f_n = g_{m(n)}(a_1, \ldots, a_{m(n)}),$$

with $a_1, \ldots, a_{m(n)} \in F \cup \{x_1, \ldots, x_n\}$ and $m(n) = n^{O(1)}$. Consider the following characterization of NP: A language A is in NP if and only if there exists a language $B \in P$ and a polynomially bounded $t: \mathbb{N} \to \mathbb{N}$ such that

$$\forall n \in \mathbb{N}\ \forall x \in \Sigma^n (x \in A \Leftrightarrow \exists e \in \Sigma^{t(n)}\ x \star e \in B),$$

or, equivalently expressed with characteristic functions:

$$\forall n \in \mathbb{N}\ \forall x \in \Sigma^n\ \chi_A(x) = \bigvee_{e \in \Sigma^{t(n)}} \chi_B(x \star e).$$

Here, Σ is the alphabet, and $\star$ a new symbol. Replacing the disjunction over e by a sum, Valiant (1979a) calls *p-definable* any family f of polynomials for which there exists a family $g \in P_F$ and a polynomially bounded $t: \mathbb{N} \to \mathbb{N}$ such that

$$\forall n \in \mathbb{N}\ f_n(x_1, \ldots, x_n) = \sum_{e_{n+1}, \ldots, e_{t(n)} \in \{0,1\}} g_{t(n)}(x_1, \ldots, x_n, e_{n+1}, \ldots, e_{t(n)}).$$

These p-definable polynomials form the analog of NP.

The central conjecture in the theory now is the *Valiant Hypothesis*: "There exist p-definable families of polynomials which are not p-computable."

Open question Is the Valiant Hypothesis true?

For an $n \times n$-matrix x with entries $x_{ij} (1 \leq i, j \leq n)$, the permanent

$$\operatorname{per}(x) = \sum_{\sigma \in S_n} x_{1,\sigma 1} \cdots x_{n,\sigma n}$$

differs from the determinant only in that each coefficient in the sum equals 1. From now on we assume that the characteristic of F is not two, since otherwise the permanent equals the determinant. Valiant shows that the permanent is p-complete (for p-definable polynomials under p-projections), so that the Valiant Hypothesis is equivalent to the conjecture that the permanent is not p-computable. Valiant (1979b) introduces the Boolean complexity class $\#P \supseteq NP$ of "counting problems," and shows that the Boolean problem of computing the number $HC(G)$ of Hamiltonian cycles is a graph G is $\#P$-complete, and that the corresponding family of polynomials is p-complete. Already the decision problem "$HC(G) \stackrel{?}{\geq} 1$" *is* NP-complete. In contrast, the existence of a perfect matching can be decided in polynomial time (Hall 1956). Surprisingly, Valiant also shows that computing the number $\operatorname{per}(M(G))$ of perfect matchings is complete for $\#P$, where $M(G)$ is the incidence matrix of G.

The theory simplifies if we generously replace "polynomial time" $n^{O(1)}$ by "quasipolynomial time" $2^{\log n^{O(1)}}$. Valiant's (1979a) "universality of the determinant" states that every qp-computable family of polynomials (with polynomially bounded degree) is a qp-projection of the determinant. This translates the *Extended Valiant Hypothesis*: "There exist p-definable polynomials which are not qp-computable" from complexity theory into the purely algebraic conjecture: "The permanent is not a qp-projection of the determinant." Presumably one of the hopes in this approach is that the more structured setting of algebraic complexity, where for example powerful tools from algebraic geometry are available, might allow us to solve analogs of Boolean problems that have defied intense efforts for decades (see Borodin 1982). Unfortunately, the area has not attracted much interest so far; a recent survey (von zur Gathen 1987c) hopes to make the subject more visible.

The Extended Valiant Hypothesis conjectures that if the $n \times n$-permanent is a projection of the $m \times m$-determinant (i.e. obtained by substituting constants and indeterminates), then $m = 2^{(\log n)^{\omega(1)}}$. The first progress on this question is the bound $m \geq \sqrt{2}n - 6\sqrt{n}$ (von zur Gathen 1987d, with the help of Babai and Seress).

6. PARALLEL COMPUTATIONS

The realization that physics places fundamental limitations on further advances for sequential "von Neumann" computers has made "parallel

computation" a hot research area, also in Boolean complexity theory. For algebraic computing, we define—in analogy with the established Boolean complexity class NC (Pippenger 1979; see Cook 1985)—for $k \in \mathbb{N}$ the complexity class NC_F^k of polynomial families $f = (f_n)$ with degree $n^{O(1)}$ which can be computed by arithmetic circuits of depth $O((\log n)^k)$ and size $n^{O(1)}$. (The depth of a circuit α is the longest length of paths within α.)

The elementary problems of linear algebra are fundamental. Csanky (1976) showed that the determinant is in NC_F^2, if char $(F) = 0$. A systematic study was initiated around 1982, when Csanky's result was generalized to arbitrary fields (Borodin et al 1982; Berkowitz 1984; Chistov 1985). Chistov's algorithm to compute the characteristic polynomial $\chi(A) = \det(tI_n - A) \in F[t]$ of a matrix $A = (a_{ij})_{1 \le i,j \le n} \in F^{n \times n}$ considers for $1 \le r \le n$ the lower right submatrix $A_r = (a_{ij})_{r \le i,j \le n} \in F^{r' \times r'}$, where $r' = n - r + 1$, and $d_r = \det(I_{r'} - tA_r) \in F[t]$. Thus $\chi(A) = t^n d_1(t^{-1})$. Writing

$$(b_{ij}^{(r)})_{r \le i,j \le n} = (I_{r'} - tA_r)^{-1} = I_{r'} + \sum_{1 \le k} t^k A_r^k \in F[[t]]^{r' \times r'},$$

$b_r = b_{rr}^{(r)}$, and $d_{n+1} = 1$, we have $b_r = d_{r+1}/d_r$ by Cramer's rule, and

$$\prod_{1 \le r \le n} b_r = \frac{1}{d_1}.$$

Thus one calculates A_r^i for $1 \le i, r \le n$, then $d_1^{-1} \bmod t^{n+1}$, and finally $d_1 \bmod t^{n+1}$ by a Newton iteration. All this can be done in depth $O(\log^2 n)$ and size $n^{O(1)}$.

For problems with a "combinatorial" output—e.g. computing the rank of matrices—we have to extend the arithmetic circuits to "arithmetic Boolean circuits." These also have test gates "$a \stackrel{?}{=} 0$" for an arithmetic result $a \in F$, Boolean operations on the resulting Boolean values, and selection gates whose output is one of the two arithmetic inputs, depending on the value of a third Boolean input. With this natural extension, the rank is in NC_F^2 (Borodin et al 1982, probabilistically; Mulmuley 1987, deterministically), and also problems combining arithmetic and combinatorial aspects like solving (possibly singular) systems of linear equations.

There is a natural notion of *NC_F^1-reduction* $f \le g$, where one has a circuit computing f in depth $O(\log n)$, with oracle gates for g. (A little care is required in defining the depth and size of oracle gates.) Let us call *DET* and *RANK* the classes of all problems NC_F^1-reducible to the determinant and rank, respectively. Then

$$NC_F^1 \subseteq RANK \subseteq DET \subseteq NC_F^2.$$

Many elementary problems from linear algebra are *NC_F^1-complete* for one of these classes. Examples are: marking a set of rows and columns of a matrix that form a maximal nonsingular minor, determining solvability of a system of linear equations, and a basis for the image of a linear mapping are complete for *RANK*; the characteristic polynomial, the first n powers of an $n \times n$-matrix, the inverse of a matrix, solving a system of linear equations, and computing a basis for the nullspace of a matrix are complete for *DET* (see von zur Gathen 1986b).

Open question Is one of the inclusions $NC_F^1 \subseteq DET \subseteq NC_F^2$ proper?

Most problems from polynomial arithmetic (see Section 2) wind up in *DET*, such as the gcd, the entries of the Extended Euclidean Scheme, interpolation (including rational and Hermite interpolation), and Padé approximation (Borodin et al 1982; von zur Gathen 1986a). Using Strassen's (1973b) technique of "avoiding divisions," Valiant et al (1983) and Miller et al (1988) show that any rational function in P_F—i.e. p-computable—is in NC_F^2. This is quite surprising, since the Boolean analog "$P = NC^2$" is conjectured to be false. Their result hinges on the restriction of polynomial degrees in our complexity classes; for classes $\widetilde{NC}_F^k$ defined without this restriction, the inequalities

$$\widetilde{NC}_F^1 \subsetneq \widetilde{NC}_F^2 \subsetneq \cdots \subsetneq \widetilde{NC}_F \subsetneq \tilde{P}_F$$

are trivial.

One of the most instructive problems in parallel arithmetic is that of computing a large power. The algorithm of "repeated squaring" (see Knuth 1981) proves that the minimal depth d_F^b of arithmetic circuits computing x^b satisfies $d_F^b \leq \lceil \log b \rceil$, and a degree argument shows equality: $d_F^b = \lceil \log b \rceil$ over an infinite field F (Kung 1976).

Over a finite field F with q elements, Fermat's little theorem $a^{q-1} = 1$ may help (for nonzero a), and indeed the bound becomes

$$m \leq d_F^b \leq m+1,$$

where $m = \min\{\lceil \log b \rceil, \lceil \log (q-b) \rceil\}$, and the circuit is not required to answer correctly for $a = 0$ (von zur Gathen 1987a). Thus if $q \geq 2^{n+1}$ and $2^{n-1} < b \leq 2^n$, then the disappointing linear lower bound $d_F^b \geq n$ follows. Quite surprisingly, Fich & Tompa (1988) show that if F has exactly 2^{n+1} elements (or, more generally, has small characteristic $p \leq n$ and $2^{O(n)}$ elements), then x^b can be computed on arithmetic circuits over the prime field $\mathbb{Z}_p$ of depth $O(\log^2 n)$. Now field elements are represented by their coordinates in some basis of F over $\mathbb{Z}_p$. Von zur Gathen & Seroussi (1986) note that also the trace and square roots have this behavior. Arithmetic circuits (over F) have been taken for granted as the appropriate model for

computing polynomials over F, and all results mentioned so far in this survey have used this model. Contrary to our intuition, for parallel powering this model is exponentially weaker than another reasonable model, namely arithmetic circuits over the prime field $\mathbb{Z}_p$ of F.

For the integer power problem of computing $a^b \bmod m$, where $a, b, m \in \mathbb{N}$ are n-bit numbers, we have to use Boolean circuits. If $m > 2b$ is a prime and only arithmetic mod m is used, we have seen that the depth is $\lceil \log_2 b \rceil$—i.e. $\Omega(n)$. However, if m has only small prime factors $\leq n$, then the problem can be solved in optimal depth $O(\log n)$ (von zur Gathen 1987a).

7. FACTORING POLYNOMIALS

Starting in this section, we change focus in two ways. So far, our problems trivially had polynomial-time solutions (except in Section 5); now we look at tasks for which polynomial-time algorithms are nontrivial. Second, our problems are not rational any more, and it is appropriate to switch to Boolean circuits (or Turing machines, or RAMs) as the model of computation.

Over any field F, a polynomial $f \in F[x_1, \ldots, x_n]$ has a factorization $f = f_1 \ldots f_r$ into irreducible polynomials $f_1, \ldots, f_r$, unique up to scalar multiples and permutations. How can we calculate this factorization? Landau (1987) gives a more comprehensive survey of the results to be discussed.

A first observation is that over very general "computable" fields, even irreducibility of univariate polynomials is undecidable (Fröhlich & Shepherdson 1955). In a remarkable preview of things to come, van der Waerden (1930) showed—before the advent of computability theory—that the existence of an undecidable subset of $\mathbb{N}$ (in today's terminology; an "ignorabimus"[1] in his words) implies that irreducibility is undecidable. In the same volume, Hilbert (1930) proposed that "*in der Mathematik gibt es kein Ignorabimus . . .*" (apparently intending a different meaning of the Latin word). However, if the field F is finitely generated over its prime field ($\mathbb{Q}$ or $\mathbb{Z}_p$), then polynomials over F can be factored effectively (Hermann 1926).

For efficient algorithms, we observe as a simplification that $\bar{f} = f/\gcd(f, \partial f/\partial x)$ is the *squarefree part* of $f \in F[x]$, if $\operatorname{char}(F) = 0$; i.e. each irreducible factor of f occurs with multiplicity 1 in $\bar{f}$. A modification (see Knuth 1981) allows us to assume also over finite fields that the input polynomial is squarefree.

The first polynomial-time factorization result, over a finite field $F = GF(q)$ with q elements, is due to Berlekamp (1967). If $f = f_1 \ldots f_r$ is a

[1] We will never know.

factorization into distinct monic irreducible polynomials $f_1, \ldots, f_r \in F[x]$, then

$$R = F[x]/(f) \overset{\psi}{\cong} F[x]/(f_1) \times \cdots \times F[x]/(f_r)$$

is the Wedderburn decomposition of R as a product of the fields $F[x]/(f_i) \supseteq F$; the F-linear isomorphism ψ^{-1} is given by the Chinese remainder algorithm. The Frobenius mapping

$$\phi : R \to R;\ \phi(a) = a^q$$

is an F-linear map, and its fixed point set is the diagonal set

$$T = \ker(\phi - id) = \psi^{-1}(F \times \cdots \times F),$$

by Fermat's little theorem:

$$\forall b \in F[x]/(f_i)\ (b^q = b \Leftrightarrow b \in F).$$

If $T = F$, then f is irreducible. Otherwise, for any $a \in T \backslash F$ there exist $b \in F$ and $1 \leq i, j \leq r$ such that

$$(\psi(a-b))_i = 0, \quad (\psi(a-b))_j \neq 0.$$

Then $g = \gcd(f, a-b)$ is a nontrivial divisor of f with $f_i | g$ and $f_j \nmid g$ (identifying $a = c \bmod f \in R$ with a corresponding polynomial $c \in F[x]$ of degree less than $\deg f$). Since the matrix of ϕ (in the natural basis) and a basis for the linear space T are easy to compute, Berlekamp's factorization algorithm uses $(dq)^{O(1)}$ operations in F, where $d = \deg f$.

Berlekamp's interest came from applications in the theory of error-correcting codes (Berlekamp 1968), where the case $q = 2$ is particularly important. Indeed, if q is small, the algorithm runs in polynomial time.

However, for large q only time polynomial in the input size $d \log q$ is regarded as feasible. An ingenious idea of Berlekamp (1970) is to introduce probabilistic choice: For randomly chosen a in T, $\gcd(f, a^{(q-1)/2} - 1)$ is a nontrivial factor of f with high probability. (For even q, one uses a variant of this algorithm.) This leads to a probabilistic algorithm of time $(d \log q)^{O(1)}$. It is of "Las Vegas" type—i.e. either returns correctly the complete factorization of f or else "failure", the latter with arbitrarily small probability $\varepsilon > 0$ and running time proportional to $\log \varepsilon^{-1}$). Today we recognize this as a fundamental contribution: the first probabilistic polynomial-time solution to a problem for which no deterministic polynomial-time algorithm is known.

Several authors have given analyses and variants of the algorithm (Rabin 1980; Cantor & Zassenhaus 1981; Camion 1983; von zur Gathen 1984a). The best time bound is by Ben-Or (1981): $O(d^2 \log d \log \log d \cdot$

$\log q$) operations in F. Before Berlekamp, Butler (1954) had used the fixed points of the Frobenius mapping for an efficient irreducibility test, and Schwarz (1956) to determine the degrees of irreducible factors.

Open question Can one factor a polynomial $f \in GF(q)[x]$ of degree d deterministically in time $(d \log q)^{O(1)}$?

Berlekamp (1970) shows that for this question one may assume that f is a product of linear factors, and that q is prime. For several special cases, an affirmative answer is in Schoof (1985) (for special square roots), Huang (1985) (for cyclotomic f), Rónyai (1987b) (for a bounded number of factors), and Moenck (1977) and von zur Gathen (1987b) (if all prime factors of $q-1$ are small); all results except Schoof's assume the Extended Riemann Hypothesis.

The next case of interest is $\mathbb{Q}[x]$. Zassenhaus (1969) recognized the importance of p-adic *Hensel lifting*. Algorithms—such as Wang's (1978) multivariate factorizer—based on his ideas work well in practice, but consume exponential time in the worst case. This is due to (exceptional) irreducible polynomials in $\mathbb{Z}[x]$ which have many factors modulo each prime p (Berlekamp 1970; Kaltofen et al 1983).

After more than a decade, a breakthrough by Lenstra et al (1982) put this problem into polynomial time. The core of their method is a fast computation of short vectors in integer lattices, which has since found many applications in other areas (see Kannan 1987). Also in 1982, Kaltofen (1985a) reduced factoring in $\mathbb{Q}[x_1, \ldots, x_n]$ to $\mathbb{Q}[x]$, so that this problem was also solved in polynomial time. These methods then yielded in 1982–1983 a flurry of polynomial-time factorization algorithms for (multivariate) polynomials over algebraic number fields and—probabilistically—over finite fields (Chistov & Grigoryev 1982; Lenstra 1983, 1984, 1985, 1987; von zur Gathen 1984b; Grigoryev & Chistov 1984; Kannan et al 1984; Schönhage 1984; von zur Gathen & Kaltofen 1985; van der Hulst & Lenstra 1985; Landau 1985).

These multivariate factorizers work in time polynomial in the length $(d+1)^n$ of a dense representation, to which each monomial of degree up to d contributes. The next problem is to deal with the important sparse representation—to whose length only monomials with nonzero coefficients contribute—and the even more concise representation by an arithmetic circuit. Zippel (1979, 1981) contributes the important idea that nonsparseness is probably preserved under randomly chosen substitutions. The central technical tools are efficient versions of Hilbert's (1892) irreducibility theorem which show that under certain random substitutions irreducible multivariate polynomials remain irreducible (Heintz & Sieveking 1981; Kaltofen 1985b; von zur Gáthen 1985). This leads to a probabilistic

factorization algorithm with cost polynomial in the degree and the length of an arithmetic circuit describing the input polynomial (Kaltofen 1986). The degree enters the running time polynomially, since otherwise even a simple problem like testing whether the gcd of two univariate polynomials is 1 is *NP*-hard (Plaisted 1984). Freeman et al (1986) have implemented these ideas in a computer algebra system (on top of MACSYMA).

An important noncommutative generalization of the factorization problem is the Wedderburn decomposition of finite-dimensional associative algebras (Friedl & Rónyai 1985; Rónyai 1987a).

The binary representation of integers suggests a formal analogy with polynomials in $\mathbb{Z}_2[x]$. Indeed, some fundamental computational problems (such as multiplication, gcd, Chinese remainder algorithm) can be solved with essentially the same method for both domains (see Lipson 1981). Factorization seems to be different: All known integer factorization algorithms require exponential time (in the binary length of the input), and much in the exciting new fields of public-key cryptosystems (Rivest et al 1978) and zero-knowledge proofs (Goldwasser et al 1988) is based on the assumption that the problem is computationally not feasible. In contrast, an integer can be tested for primality probabilistically in polynomial time (Solovay & Strassen 1977), or deterministically in quasi-polynomial time (Adleman et al 1983) or in polynomial time assuming the Extended Riemann Hypothesis (Miller 1975); the surveys by Dixon (1984) and Lenstra & Lenstra (1987) give further references. Deciding whether a quadratic equation in two variables has an integer solution is *NP*-complete (Manders & Adleman 1978).

8. ALGEBRAIC THEORIES

The tenth problem in Hilbert's (1900) famous list asks (in today's interpretation) for computer programs to determine solvability by integers of an integral polynomial equation, long before the advent of programmable computers. In the 1930s, the work of Gödel, Church, Kleene, Turing, and Rosser showed that a negative answer to such a question is possible—contrary to Hilbert's hopes. Indeed, Matyasevich (1970) proved that Hilbert's tenth problem is unsolvable, based on the work of J. Robinson and Davis, Putnam & Robinson; see Davis (1973) for an overview.

An (arithmetic) *expression* over a field is built up from constants and indeterminates, using $+$, $-$, $*$, and $/$. A (first-order) *formula* is made up from atomic formulas "expression $= 0$" (and "expression > 0" for real fields) with the logical operators $\neg$, $\wedge$, $\vee$, $\exists$, and $\forall$ (with quantifiers ranging over field elements). The *theory* of a field (or a set of fields) consists

of all true formulas. Assuming that the *length* of constants is defined (e.g. unit length, or binary length if only integer constants occur), there is a natural notion of length $\ell(\phi)$ of a formula ϕ (using the dense encoding for polynomials).

The theory of $\mathbb{Q}$ is undecidable; similarly the theory of all fields (Robinson 1949). Special subproblems may be solvable: Polynomials over $\mathbb{Q}$ can be factored in polynomial time (see Section 7), and one can decide (in doubly exponential time) the *ideal membership problem*: Is a given multivariate polynomial over $\mathbb{Q}$ in an ideal given by generators (Cardoza et al 1976; Mayr & Meyer 1982, based on Hermann 1926)? In fact, Mayr & Meyer show that any algorithm uses at least exponential space, say on a Turing machine. Hermann's result leads to an algorithm for ideal membership over any field, with a doubly exponential number of arithmetic operations.

The theory of a single finite field is trivial to decide; the theories of all finite fields, and of all finite fields of fixed characteristic are also decidable (Ax 1968). A decision procedure for p-adic fields was given by Ax & Kochen (1965) and Ershov et al (1965); Cohen (1969) presents a primitive recursive quantifier elimination algorithm. Of particular interest are the theories of real closed and of algebraically closed fields. The latter includes the question of whether a system of polynomial equations over an arbitrary field has any solution (in an algebraic extension field).

The theory of $\mathbb{R}$ (or of real closed fields) encompasses the questions of elementary geometry (compass and ruler constructions), and has recently seen interesting applications in robotics. We first discuss the upper bounds (=algorithms), then some lower bounds. Tarski (1948) has shown that the theory is decidable. His algorithm runs in time $2^{2^{\cdot^{\cdot^{2}}}}$, where the height of the tower of 2s equals the length ℓ of the input formula. Seidenberg (1954) and Cohen (1969) made conceptual simplifications, but Collins's (1975) method of "cylindrical algebraic decomposition" brought a dramatic improvement to $2^{2^{O(\ell)}}$. He implemented his ideas in the computer algebra system SAC-2. Wüthrich (1976) simplified this approach, based on ideas of Monk and Solovay; these algorithms actually provide quantifier elimination. Risler (1988) gives a survey of these methods.

The most precise result is by Grigoryev (1988), whose decision procedure uses $M(kd)^{(O(n))^{4r-2}}$ steps. Here the input formula contains k atomic formulas $f_i \geq 0$, each polynomial $f_i \in \mathbb{Z}[x_1, \ldots, x_n]$ has degree at most d and each coefficient bounded by 2^M in absolute value, and r is the number of alternations of $\forall$ and $\exists$ quantifiers, assuming that the formula is in prenex form.

Ben-Or et al (1986) give a parallel decision procedure, using exponential depth (or equivalently, work space) and a doubly exponential number of

processors. Their algorithm shows that for fixed n the decision problem of the theory of $\mathbb{R}$ is in *NC*.

Many questions in robotics, such as the "piano mover's problem," can be phrased as first-order formulas over $\mathbb{R}$. Reif (1979) indicates a general formulation of this problem which is complete for the complexity class of polynomial space, and outlines polynomial-time algorithms for moving a rigid polyhedron in 2 or 3 dimensions. The substantial work of Schwartz & Sharir (1983a,b) continues along these lines. Canny's (1987) methods work in simply exponential time under assumptions that seem reasonable for the robotics problems.

The special case where all polynomials are linear has many applications, for example in economics; Megiddo (1987) presents a survey of linear programming.

Drexler (1978) and Garcia & Zangwill (1979) propose the "homotopy method" for solving a system $\phi(x) = 0$ of polynomial equations. This has been successfully applied to certain problems in physics, where one knows a priori that the number of solutions is finite (Li 1987). One considers a "homotopy" $\psi(x, t)$ in variables t and $x = (x_1, \ldots, x_n)$, where $\psi(x, 1) = \phi(x)$, and $\psi(x, 0)$ is an "easy" system, say of the form $x_i^{e_i} = b_i$. One traces, with numerical methods, these easy solutions as the parameter t moves from 0 to 1. It would be interesting to establish the range of applicability and the computational cost of this method. Different heuristic methods, implemented in the Maple Computer Algebra System (Char et al 1986), rescued the author in a problem that defied his pencil and paper attempts (see von zur Gathen 1987d).

Quantifier elimination procedures for the theory of algebraically closed fields follow in characteristic zero from the real methods; Heintz & Wüthrich (1975), Heintz (1983), and Chistov & Grigoryev (1984) give a solution in arbitrary characteristic. They use doubly exponential time. Fitchas et al (1987) give a parallel quantifier elimination method, both for real closed and algebraically closed fields, attaining the same bounds as Ben-Or et al (1986) for the real decision problem.

Algorithms with doubly exponential cost are not feasible except for very small inputs; these methods are polynomial-time if the number of quantified variables is bounded. However, in the general case one cannot do much better: Fischer & Rabin (1974) prove an exponential time lower bound for deciding the theory of real closed fields. (This large lower bound is one of Rabin's many contributions to complexity theory, for which he received the ACM Turing Award in 1976, with Scott.) Heintz (1983), Davenport & Heintz (1988), and Fitchas et al (1987) show that the output of quantifier elimination may have doubly exponential size, both for algebraically closed fields and for real closed fields. Thus the fast parallel methods cannot be essentially improved.

Buchberger introduced in 1965 the powerful general method of Gröbner bases to compute with polynomial ideals; see Buchberger (1985) for an overview. In general, the cost of the method is not well understood [at least doubly exponential (Möller & Mora 1984)], but an approach of Bayer & Stillman (1985, unpublished) raises the hope that for many natural geometric problems the running time may be more reasonable.

9. PERMUTATION GROUPS

A permutation group G on $\{1,\dots,n\}$ is a subgroup of S_n, the group of all $n!$ permutations of $\{1,\dots,n\}$. Such groups arise naturally in many areas, such as combinatorics, physics, and chemistry. The size of such groups may be exponential in n. A concise representation of G is by *generators* $g_1,\dots,g_k \in S_n$, so that $G = \langle g_1,\dots,g_k\rangle$ consists of all products that can be formed using $g_1,\dots,g_k$. As an example, let $g_1 = (1,2,\dots,n)$ be the cyclic shift, and $g_2 = (1,2)$ be a transposition. The $G = \langle g_1\rangle \cong \mathbb{Z}_n$ is cyclic, and $G = \langle g_1, g_2\rangle = S_n$.

For the following computational problems, generators $g_1,\dots,g_k \in S_n$ of G are part of the input.

1. Given $g \in S_n$, is $g \in G$ (*membership*)?
2. Determine $\#G$ (*order*).
3. Given $I \subseteq \{1,\dots,n\}$, determine generators for $G_I = \{g \in G : \forall i \in I\ g(i) = i\}$ (*point-wise set stabilizer*).
4. Given $I \subseteq \{1,\dots,n\}$, determine generators for $\mathrm{Stab}_I(G) = \{g \in G : g(I) = I\}$ (*set stabilizer*).
5. Determine generators for normal subgroups $\{1\} = G_0 \lhd G_1 \lhd \cdots \lhd G_t = G$ such that each G_i/G_{i-1} is simple (*composition series*), and generators for each G_i/G_{i-1} (*composition factors*).

For the last problem, we have to exhibit a faithful permutation representation of G_i/G_{i-1} on some set with $m_i \le n$ elements, and then the generators are in S_{m_i}. The output tells us, in particular, whether G is solvable.

The seminal work of Sims (1970) constructs from the input $g_1,\dots,g_k$ a system of *strong generators* for G:

$$\begin{pmatrix} h_{11} & h_{12} & \dots & h_{1,n-1} & h_{1n} \\ 1 & h_{22} & \dots & h_{2,n-1} & h_{2n} \\ \vdots & \vdots & & \vdots & \vdots \\ 1 & 1 & \dots & 1 & h_{nn} \end{pmatrix}.$$

with the identity below the main diagonal, where $h_{ii}, \ldots, h_{in}$ are right coset representatives for G_i mod G_{i-1}, $G_i = G_{\{1,\ldots,i-1\}}$, and each h_{ij} is either 1 or satisfies $h_{ij}(i) = j$. Then any $g \in G$ has a unique representation $g = g_n \cdot g_{n-1} \cdots g_1$, where g_i is from the ith row.

Sims's "sifting" algorithm—slightly modified by Furst et al (1980)—computes strong generators by starting with a trivial table and inductively inserting one given generator after the other. If g is to be inserted, it is sifted down the rows $i = 1, 2, \ldots, n$ of the table as follows. At level i, if there is a table entry h_{ij} with $g(i) = h_{ij}(i)$, replace g by $h_{ij}^{-1} g$. (Note that $h_{ij}^{-1} g \in G_i$.) Otherwise insert g in position $(i, g(i))$. After sifting the generators, also each product of two table entries has to be sifted. Furst et al prove that the whole process runs in time $O(kn^2 + n^6)$; this can be improved to $O(kn^2 + n^5)$ (Babai 1986). It shows that Problems 1, 2, and 3 are in P, and Luks (1987) extends this to Problem 5.

Open question Can set stabilizers (Problem 4) be computed in polynomial time?

A central combinatorial question is the status of *graph isomorphism*: Given two graphs, determine whether they are isomorphic. This problem is in NP, is not known to be in P, and there is some evidence that it is probably not NP-complete (see Goldwasser et al 1988). Major progress on this question resulted from the group-theoretic advances: Babai (1979) suggested a connection between the problems, and graph isomorphism is in P for graphs of bounded degree (Luks 1982), of bounded genus (Filotti & Mayer 1980), or of bounded eigenvalue multiplicity (Babai et al 1982); Miller (1983a,b) extends these results. Graph isomorphism is polynomial-time reducible to Problem 4 (Luks 1982), so that a positive answer to the open question would also settle this combinatorial problem.

As soon as all problems were solved in polynomial time, "someone changed the rules" (in Luks's words) and the hunt for fast parallel algorithms started, with parallel time $(\log n)^{O(1)}$ and $n^{O(1)}$ processors (i.e. the complexity class NC), for permutation groups $G \leq S_n$. The first step in 1983 was an algorithm by McKenzie & Cook (1987) for Abelian permutation groups. After further results by Luks & McKenzie and Luks, finally Babai et al (1987) found parallel algorithms of small depth for all our problems (except Problem 4, of course). In their words, "most striking is the depth of group-theoretic machinery that is required for parallelizing even the rudimentary task of membership testing"; their proof relies on the classification of finite simple groups. They also find pointwise stabilizers of sets, and then put the problem of determining isomorphism of graphs with bounded multiplicity of eigenvalues into NC.

SUMMARY

Algebraic complexity theory has seen significant research for less than 20 years. There have been spectacular successes, in particular the matching upper and lower bounds of Sections 2 and 3, also the polynomial-time and fast parallel algorithms of Sections 6, 7, and 9. We are now aware of what the central problems are, such as the complexity of the Discrete Fourier Transformation, of matrix multiplication, and of the permanent. Apart from combinatorial or ad hoc techniques, central tools have been the methods of algebraic geometry, and of Diophantine geometry in Section 7.

ACKNOWLEDGMENT

This work was supported by National Science and Engineering Council of Canada, grant 3-650-126-40.

Literature Cited

Adleman, L. M., Pomerance, C., Rumely, R. S. 1983. On distinguishing prime numbers from composite numbers. *Ann. Math.* 117: 173–206

Aho, A. V., Hopcroft, J. E., Ullman, J. D. 1974. *The Design and Analysis of Computer Algorithms*. Reading, Mass: Addison-Wesley. $x+470$ pp.

Alder, A., Strassen, V. 1981. On the algorithmic complexity of associative algebras. *Theor. Comput. Sci.* 15: 201–11

Ax, J. 1968. The elementary theory of finite fields. *Ann. Math.* 88: 239–71

Ax, J., Kochen, S. 1965. Diophantine problems over local fields. II. A complete set of axioms for p-adic number theory. *Am. J. Math.* 87: 631–48

Babai, L. 1979. *Monte Carlo algorithms in graph isomorphism testing*. Tech. Rep. 79-10, Dép. Math. Stat., Univ. Montréal

Babai, L. 1986. On the length of subgroup chains in the symmetric group. *Commun. Algebra* 14: 1729–36

Babai, L., Grigoryev, D. Yu., Mount, D. M. 1982. Isomorphism of graphs with bounded eigenvalue multiplicity. *Proc. 14th Ann. ACM Symp. Theory Comput., San Francisco*, pp. 310–24

Babai, L., Luks, E. M., Seress, Á. 1987. Permutation groups in NC. *Proc. 19th Ann. ACM Symp. Theory Comput., New York*, pp. 409–20

Baur, W., Strassen, V. 1982. The complexity of partial derivatives. *Theor. Comput. Sci.* 22: 317–30

Belaga, E. G. 1958. Some problems involved in the computation of polynomials. *Dokl. Akad. Nauk. SSSR* 123: 775–77

Belaga, E. G. 1961. Evaluation of polynomials of one variable with preliminary processing of the coefficients. *Probl. Kibern.* 5: 7–15

Ben-Or, M. 1981. Probabilistic algorithms in finite fields. *Proc. 22nd IEEE Symp. Found. Comput. Sci.*, pp. 394–98

Ben-Or, M. 1983. Lower bounds for algebraic computation trees. *Proc. 15th Ann. ACM Symp. Theory Comput., Boston*, pp. 80–86

Ben-Or, M., Kozen, D., Reif, J. 1986. The complexity of elementary algebra and geometry. *J. Comput. Syst. Sci.* 32: 251–64

Berkowitz, S. J. 1984. On computing the determinant in small parallel time using a small number of processors. *Inf. Process. Lett.* 18: 147–50

Berlekamp, E. R. 1967. Factoring polynomials over finite fields. *Bell Syst. Tech. J.* 46: 1853–59

Berlekamp, E. R. 1968. *Algebraic Coding Theory*. New York: McGraw-Hill

Berlekamp, E. R. 1970. Factoring polynomials over large finite fields. *Math. Comput.* 24: 713–35

Beth, T. 1984. *Verfahren der schnellen Fourier-Transformation*. Stuttgart. Teubner. 316 pp.

Bini, D., Capovani, M., Lotti, G., Romani, F. 1979. $O(n^{2.7799})$ complexity for matrix

multiplication. *Inf. Process. Lett.* 8: 234–35

Borodin, A. 1977. On relating time and space to size and depth. *SIAM J. Comput.* 6: 733–44

Borodin, A. 1982. Structured vs. general models in computational complexity. In *Logic and Algorithmic*, Symposium in honour of Ernst Specker. *Enseign. Math.* 30: 47–65

Borodin, A., Cook, S. 1976. On the number of additions to compute specific polynomials. *SIAM J. Comput.* 5: 146–57

Borodin, A., von zur Gathen, J., Hopcroft, J. 1982. Fast parallel matrix and GCD computations. *Inf. Control* 52: 241–56

Borodin, A., Moenck, R. 1974. Fast modular transforms. *J. Comput. Syst. Sci.* 8: 366–86

Borodin, A., Munro, I. 1971. Evaluating polynomials at many points. *Inf. Process. Lett.* 1: 66–68

Borodin, A., Munro, I. 1975. *The Computational Complexity of Algebraic and Numeric Problems*. New York: American Elsevier. x + 174 pp.

Brauer, A. 1939. On addition chains. *Bull. Am. Math. Soc.* 45: 736–39

Brockett, R. W., Dobkin, D. 1978. On the optimal evaluation of a set of bilinear forms. *Lin. Algebra Appl.* 19: 207–35

Brown, W. S. 1971. On Euclid's algorithm and the computation of polynomial greatest common divisors. *J. Assoc. Comput. Mach.* 18: 478–504

Brown, M. R., Dobkin, D. 1980. An improved lower bound on polynomial multiplication. *IEEE Trans. Comput.* 29: 337–40

Bshouty, N. 1988. *Lower bound for matrix multiplication. Proc. 29th IEEE Symp. Found. Comput. Sci.*, White Plains NY. In press

Buchberger, B. 1985. Gröbner bases: an algorithmic method in polynomial ideal theory. In *Multidimensional Systems Theory*, ed. N. K. Bose, pp. 184–232. Dordrecht: Reidel

Buchberger, B., Collins, G. E., Loos, R. 1983. *Computer Algebra. Symbolic and Algebraic Computation*. Wien/New York: Springer-Verlag. 2nd ed. 283 pp.

Bunch, J., Hopcroft, J. 1974. Triangular factorization and inversion by fast matrix multiplication. *Math. Comput.* 28: 231–36

Butler, M. C. R. 1954. On the reducibility of polynomials over a finite field. *Q. J. Math. Oxford* 5: 102–7

Camion, P. 1983. A deterministic algorithm for factorizing polynomials of $F_q[x]$. *Ann. Discr. Math.* 17: 149–57

Canny, J. 1987. A new algebraic method for robot motion planning and real geometry. *Proc. 28th IEEE Symp. Found. Comput. Sci., Los Angeles*, pp. 39–48

Cantor, D. G., Kaltofen, E. 1987. *Fast multiplication of polynomials over arbitrary rings*. Prelim. rep., 12 pp. Rensselaer Polytechnic Inst., Troy NY. *Acta Inf.* In press

Cantor, D. G., Zassenhaus, H. 1981. On algorithms for factoring polynomials over finite fields. *Math. Comput.* 36: 587–92

Cardoza, E., Lipton, R., Meyer, A. R. 1976. Exponential space complete problems for Petri nets and commutative semigroups: preliminary report. *Proc. 8th Ann. ACM Symp. Theory Comput.*, pp. 50–54

Caviness, B. F. 1986. Computer algebra: past and future. *J. Symb. Comput.* 2: 217–36

Char, B. W., Fee, G. J., Geddes, K. O., Gonnet, G. H., Monagan, M. B. 1986. A tutorial introduction to Maple. *J. Symb. Comput.* 2: 179–200

Chistov, A. L. 1985. Fast parallel calculation of the rank of matrices over a field of arbitrary characteristic. *Proc. Int. Conf. Found. Comput. Theory. Springer Lect. Notes Comput. Sci.* 199: 63–69

Chistov, A. L., Grigoryev, D. Yu. 1982. *Polynomial-time factoring of the multivariable polynomials over a global field.* LOMI preprint E-5-82. Leningrad: USSR Akad. Sci.

Chistov, A. L., Grigoryev, D. Yu. 1984. Complexity of quantifier elimination in the theory of algebraically closed fields. *Proc. 11th Symp. Math. Found. Comput. Sci., Springer Lect. Notes Comput. Sci.* 176: 17–31

Cohen, P. J. 1969. Decision procedures for real and p-adic fields. *Commun. Pure Appl. Math.* 22: 131–51

Collins, G. E. 1975. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. *Springer Lect. Notes Comput. Sci* 33: 134–83

Cook, S. A. 1971. The complexity of theorem proving procedures. *Proc. 3rd Ann. ACM Symp. Theory Comput.*, pp. 151–58

Cook, S. A. 1985. A taxonomy of problems with fast parallel algorithms. *Inf. Control* 64: 2–22

Cooley, J. W., Tukey, J. W. 1965. An algorithm for the machine calculation of complex Fourier series. *Math. Comput.* 19: 297–301

Coppersmith, D., Winograd, S. 1982. On the asymptotic complexity of matrix multiplication. *SIAM J. Comput.* 11: 472–92

Coppersmith, D., Winograd, S. 1987. Matrix multiplication via arithmetic progressions. *Proc. 19th Ann. ACM Symp. Theory Comput., New York*, pp. 1–6

Csanky, L. 1976. Fast parallel matrix inversion algorithms. *SIAM J. Comput.* 5: 618–23

Davenport, J. H., Heintz, J. 1988. Real quantifier elimination is doubly exponential. *J. Symb. Comput.* 5: 29–35

Davis, M. 1973. Hilbert's tenth problem is unsolvable. *Am. Math. Mon.* 80: 233–69

Dixon, J. D. 1984. Factorization and primality tests. *Am. Math. Mon.* 91: 333–52

Drexler, F. J. 1978. A homotopy-method for the calculation of all zero-dimensional polynomial ideals. In *Continuation Methods*, ed. H. Wacker, pp. 69–93. New York: Academic

Erdös, P. 1960. Remarks on number theory. III. On addition chains. *Acta Arith.* 6: 77–81

Ershov, Yu. L., Lavrov, I. A., Taimanov, A. D., Taitslin, M. A. 1965. Elementary theories. *Russ. Math. Surv.* 20: 35–105

Fich, F., Tompa, M. 1988. The parallel complexity of exponentiating polynomials over finite fields. *J. Assoc. Comput. Mach.* 35. In press

Fiduccia, C. 1972. Polynomial evaluation via the division algorithm: the fast Fourier transform revisited. *Proc. 4th Ann. ACM Symp. Theory Comput., Denver*, pp. 88–93

Fiduccia, C., Zalcstein, Y. 1977. Algebras having linear multiplicative complexity. *J. Assoc. Comput. Mach.* 24: 311–31

Filotti, I. S., Mayer, J. N. 1980. A polynomial time algorithm for determining isomorphism of graphs of fixed genus. *12th Ann. ACM Symp. Theory Comput., Los Angeles*, pp. 236–43

Fischer, M. J., Rabin, M. O. 1974. Super-exponential complexity of Presburger arithmetic. In *Complexity of Computation*, ed. R. M. Karp, pp. 27–41. Am. Math. Soc., Providence, RI

Fitchas, N., Galligo, A., Morgenstern, J. 1987. *Algorithmes rapides en séquentiel et en parallèle pour l'élimination de quantificateurs en géometrie élémentaire.* A apparaître dans: Séminaire Structures Algébriques Ordonnées, UER de Mathématiques, Université de Paris VII

Freeman, T. S., Imirzian, G. M., Kaltofen, E. 1986. *A system for manipulating polynomials given by straight-line programs.* Tech. Rep. 86-15, Dept. Comput. Sci. Rensselaer Polytechnic Inst., Troy, NY. 20 pp.

Friedl, K., Rónyai, L. 1985. Polynomial time solutions to some problems in computational algebra. *Proc. 17th Ann. ACM Symp. Theory of Comput.*, Providence, pp. 153–62

Fröhlich, A., Shepherdson, J. C. 1955. Effective procedures in field theory. *Philos. Trans. R. Soc. London Ser. A* 248: 407–32

Furst, M., Hopcroft, J., Luks, E. 1980. Polynomial-time algorithms for permutation groups. *Proc. 21st Ann. IEEE Symp. Found. Comput. Sci., Syracuse, NY*, pp. 36–41

Garcia, C. B., Zangwill, W. I. 1979. Finding all solutions to polynomial systems and other systems of equations. *Math. Program.* 16: 159–76

Gastinel, N. 1971. Sur le calcul des produits de matrices. *Numer. Math.* 17: 222–29

von zur Gathen, J. 1984a. Parallel algorithms for algebraic problems. *SIAM J. Comput.* 13: 802–24

von zur Gathen, J. 1984b. Hensel and Newton methods in valuation rings. *Math. Comput.* 42: 637–61

von zur Gathen, J. 1985. Irreducibility of multivariate polynomials. *J. Comput. Syst. Sci.* 31: 225–64

von zur Gathen, J. 1986a. Representations and parallel computations for rational functions. *SIAM J. Comput.* 15: 432–52

von zur Gathen, J. 1986b. Parallel arithmetic computations: a survey. *Proc. 12th Int. Symp. Math. Found. Comput. Sci., Bratislava. Springer Lect. Notes in Comput. Sci.* 233: 93–112

von zur Gathen, J. 1987a. Computing powers in parallel. *SIAM J. Comput.* 16: 930–45

von zur Gathen, J. 1987b. Factoring polynomials and primitive elements for special primes. *Theor. Comput. Sci.* 52: 77–89

von zur Gathen, J. 1987c. Feasible arithmetic computations: Valiant's hypothesis. *J. Symb. Comput.* 4: 137–72

von zur Gathen, J. 1987d. Permanent and determinant. *Lin. Algebra Appl.* 96: 87–100

von zur Gathen, J., Kaltofen, E. 1985. Factorization of multivariate polynomials over finite fields. *Math. Comput.* 45: 251–61

von zur Gathen, J., Seroussi, G. 1986. Boolean circuits versus arithmetic circuits. *Proc. 6th Int. Conf. Comput. Sci., Santiago, Chile*, pp. 171–84

von zur Gathen, J., Strassen, V. 1980. Some polynomials that are hard to compute. *Theor. Comput. Sci.* 11: 331–35

Goldwasser, S., Micali, S., Rackoff, C. 1988. The knowledge complexity of interactive proof systems. *SIAM J. Comput.* In press

Grigoryev, D. Yu. 1978. *Some new bounds on tensor rank.* LOMI preprint E-2-78. Leningrad: USSR Akad. Sci. 13 pp.

Grigoryev, D. Yu. 1988. Complexity of deciding Tarski algebra. *J. Symb. Comput.* 5: 65–108

Grigoryev, D. Yu., Chistov, A. L. 1984. Fast decomposition of polynomials into irreducible ones and the solution of systems of algebraic equations. *Sov. Math. Dokl.* 29: 380–83

de Groote, H. F. 1975. On the complexity of quaternion multiplication. *Inf. Process. Lett.* 3: 177–79

de Groote, H. F. 1978. On varieties of optimal algorithms for the computation of bilinear mappings. II. Optimal algorithms for 2×2-matrix multiplication. *Theor. Comput. Sci.* 7: 127–48

de Groote, H. F. 1987. *Lectures on the complexity of bilinear problems. Springer Lect. Notes Comput. Sci.* 245. 135 pp.

Hall, M., Jr. 1956. An algorithm for distinct representatives. *Am. Math. Mon.* 63: 716–717

Hartmann, W., Schuster, P. 1980. Multiplicative complexity of some rational functions. *Theor. Comput. Sci.* 10: 53–61

Heintz, J. 1983. Definability and fast quantifier elimination in algebraically closed fields. *Theor. Comput. Sci.* 24: 239–77

Heintz, J. 1985. *Zur Berechnungskomplexität von Polynomen und bilinearen Abbildungen. Ein Exposé.* Habilitationsschrift, Fachbereich Mathematik, Universität Frankfurt. 45 pp.

Heintz, J., Schnorr, C. P. 1982. Testing polynomials which are hard to compute. In *Logic and Algorithmic*, Symposium in honour of Ernst Specker. *Enseign. Math.* 30: 237–54

Heintz, J., Sieveking, M. 1980. Lower bounds for polynomials with algebraic coefficients. *Theor. Comput. Sci.* 11: 321–30

Heintz, J., Sieveking, M. 1981. Absolute primality of polynomials is decidable in random polynomial time in the number of variables. *Springer Lect. Notes Comput. Sci.* 115: 16–28

Heintz, J., Wüthrich, R. 1975. An efficient quantifier elimination algorithm for algebraically closed fields. *SIGSAM Bull.* 9: 11

Hermann, G. 1926. Die Frage der endlich vielen Schritte in der Theorie der Polynomideale. *Math. Ann.* 95: 736–88

Hilbert, D. 1892. Ueber die Irreduzibilität ganzer rationaler Funktionen mit ganzzahligen Koeffizienten. *J. Reine Angew. Math.* 110: 104–29

Hilbert, D. 1900. Mathematische Probleme. *Nachr. Akad. Wiss. Göttingen* 253–97

Hilbert, D. 1930. Probleme der Grundlegung der Mathematik. *Math. Ann.* 102: 1–9

Hopcroft, J. E., Kerr, L. R. 1971. On minimizing the number of multiplications necessary for matrix multiplication. *SIAM J. Appl. Math.* 20: 30–36

Hopcroft, J. E., Musinski, J. 1973. Duality applied to the complexity of matrix multiplication and other bilinear forms. *SIAM J. Comput.* 2: 159–73

Horowitz, E. 1972. A fast method for interpolation using preconditioning. *Inf. Process. Lett.* 1: 157–63

Huang, M. A. 1985. Riemann Hypothesis and finding roots over finite fields. *Proc. 17th Ann. ACM Symp. Theory Comput.*, Providence, RI, pp. 121–30

van der Hulst, M.-P., Lenstra, A. K. 1985. Factorization of polynomials by transcendental evaluation. *Proc. EUROCAL 85*, Vol. 2. *Springer Lect. Notes Comput. Sci.* 204: 138–45

Ja'ja', J. 1979. Optimal evaluation of pairs of bilinear forms. *SIAM J. Comput.* 8: 443–62

Kaltofen, E. 1985a. Polynomial-time reductions from multivariate to bi- and univariate integral polynomial factorization. *SIAM J. Comput.* 14: 469–89

Kaltofen, E. 1985b. Effective Hilbert irreducibility. *J. Comput. Syst. Sci.* 66: 123–37

Kaltofen, E. 1986. Uniform closure properties of p-computable functions. *Proc. 18th Ann. ACM Symp. Theory Comput., Berkeley*, pp. 330–37

Kaltofen, E. 1987. Computer algebra algorithms. *Ann. Rev. Comput. Sci.* 2: 91–118

Kaltofen, E., Musser, D. R., Saunders, B. D. 1983. A generalized class of polynomials that are hard to factor. *SIAM J. Comput.* 12: 473–83

Kannan, R. 1987. Algorithmic geometry of numbers. *Ann. Rev. Comput. Sci.* 2: 231–67

Kannan, R., Lenstra, A. K., Lovász, L. 1984. Polynomial factorization and nonrandomness of bits of algebraic and some transcendental numbers. *Proc. 16th Ann. ACM Symp. Theory Comput., Washington, DC*, pp. 191–200

Karp, R. M. 1972. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, ed. R. E. Miller, J. W. Thatcher, J. D. Bonlinger, pp. 85–104. New York: Plenum

Keller-Gehrig, W. 1985. Fast algorithms for the characteristic polynomial. *Theor. Comput. Sci.* 36: 309–17

Knuth, D. E. 1970. The analysis of algorithms. *Proc. Int. Congr. Math., Nice*, 3: 269–74

Knuth, D. E. 1981. *The Art of Computer Programming*, Vol. 2. *Seminumerical Algorithms.* Reading, Mass: Addison-Wesley. 2nd ed. xiv + 688 pp.

Kung, H. T. 1974. On computing reciprocals of power series. *Numer. Math.* 22: 341–48

Kung, H. T. 1976. New algorithms and lower bounds for the parallel evaluation of certain rational expressions and recurrences. *J. Assoc. Comput. Mach.* 23: 252–61

Landau, S. 1985. Factoring polynomials over algebraic number fields. *SIAM J. Comput.* 14: 184–95

Landau, S. 1987. Factoring polynomials quickly. *Notices Am. Math. Soc.* 34: 3–8
van Leeuwen, J., van Emde Boas, P. 1978. Elementary proofs of lower bounds in complexity theory. *Lin. Algebra Appl.* 19: 63–80
Lempel, A., Seroussi, G., Winograd, S. 1983. On the complexity of multiplication in finite fields. *Theor. Comput. Sci.* 22: 285–96
Lempel, A., Winograd, S. 1977. A new approach to error-correcting codes. *IEEE Trans. Inf. Theory* 23: 503–8
Lenstra, A. K. 1983. Factoring polynomials over algebraic number fields. *Proc. EUROCAL. Springer Lect. Notes Comput. Sci.* 162: 245–54
Lenstra, A. K. 1984. Factoring multivariate integral polynomials. *Theor. Comput. Sci.* 34: 207–13
Lenstra, A. K. 1985. Factoring multivariate polynomials over finite fields. *J. Comput. System Sci.* 30: 235–48
Lenstra, A. K. 1987. Factoring multivariate polynomials over algebraic number fields. *SIAM J. Comput.* 16: 591–98
Lenstra, A. K., Lenstra, H. W. Jr. 1987. *Algorithms in number theory*. Tech. Rep. 87-008, Dept. Comput. Sci., Univ. Chicago. To appear in *Handbook of Theoretical Computer Science*, ed. J. van Leeuwen, A. Meyer, M. Nivat, M. Paterson, D. Perrin. North-Holland: Amsterdam
Lenstra, A. K., Lenstra, H. W., Lovász, L. 1982. Factoring polynomials with rational coefficients. *Math. Ann.* 261: 515–34
Levin, L. A. 1973. Universal search problems. *Problemy Peredachy Informacii* 9: 115–16. Engl. transl. in *Probl. Inf. Transm.* 9: 265–66
Li, T.-Y. 1987. Solving polynomial systems. *Math. Intell.* 9: 33–39
Lipson, J. D. 1981. *Elements of Algebra and Algebraic Computing*. Reading, Mass: Addison-Wesley
Lipton, R. J. 1975. Polynomials with 0-1 coefficients that are hard to evaluate. *Proc. 16th Ann. IEEE Symp. Found. Comput. Sci., Berkeley*, pp. 6–10
Luks, E. M. 1982. Isomorphism of graphs of bounded valence can be tested in polynomial time. *J. Comput. Syst. Sci.* 25: 42–65
Luks, E. M. 1987. Computing the composition factors of a permutation group in polynomial time. *Combinatorica* 7: 87–99
Manders, K. L., Adleman, L. 1978. NP-complete decision problems for binary quadratics. *J. Comput. Syst. Sci.* 16: 168–84
Matyasevich, Yu. V. 1970. Enumerable sets are diophantine. *Dokl. Akad. Nauk SSSR* 191: 279–82. Engl. transl. in *Sov. Math. Dokl.* 11: 354–58
Mayr, E. W., Meyer, A. R. 1982. The complexity of the word problems for commutative semigroups and polynomial ideals. *Adv. Math.* 46: 305–29
McKenzie, P., Cook, S. A. 1987. The parallel complexity of Abelian permutation group problems. *SIAM J. Comput.* 16: 880–909
Megiddo, N. 1987. Linear programming (1986). *Ann. Rev. Comput. Sci.* 2: 119–45
Mihailjuk, M. V. 1979. On the complexity of calculating the elementary symmetric functions in finite fields. *Sov. Math. Dokl.* 20: 170–74
Miller, G. L. 1975. Riemann's hypothesis and tests for primality. *Proc 7th Ann. ACM Symp. Theory Comput., Albuquerque, NM*, pp. 234–39
Miller, G. L. 1983a. Isomorphism of k-contractible graphs. A generalization of bounded valence and bounded genus. *Inf. Control* 56: 1–20
Miller, G. L. 1983b. Isomorphism of graphs which are pairwise k-separable. *Inf. Control* 56: 21–33
Miller, G. L., Ramachandran, V., Kaltofen, E. 1988. Efficient parallel evaluation of straight-line code and arithmetic circuits. *SIAM J. Comput.* 17. In press
Moenck, R. 1973. Fast computation of GCD's. *Proc. 5th Ann. ACM Symp. Theory Comput.*, pp. 142–51
Moenck, R. T. 1977. On the efficiency of algorithms for polynomial factoring. *Math. Comput.* 31: 235–50
Möller, H. M., Mora, F. 1984. Upper and lower bounds for the degree of Gröbner bases. *Proc. EUROSAM 84. Springer Lect. Notes Comput. Sci.* 174, pp. 172–83
Motzkin, T. S. 1955. Evaluation of polynomials. *Bull. Am. Math. Soc.* 61: 163
Mulmuley, K. 1987. A fast parallel algorithm to compute the rank of a matrix over an arbitrary field. *Combinatorica* 7: 101–4
Ostrowski, A. M. 1954. On two problems in abstract algebra connected with Horner's rule. In *Studies in Mathematics and Mechanics*, presented to Richard von Mises, pp. 40–48. New York: Academic
Pan, V. Ya. 1966. Methods of computing values of polynomials. *Russ. Math. Surv.* 21: 105–36
Pan, V. Ya. 1978. Strassen's algorithm is not optimal. *Proc. 19th Ann. IEEE Symp. Found. Comput. Sci.*, pp. 166–76
Pan, V. 1984. *How to Multiply Matrices Faster. Springer Lect. Notes Comput. Sci.* 179. 212 pp.
Paterson, M. S., Stockmeyer, L. J. 1973. On the number of nonscalar multiplications necessary to evaluate polynomials. *SIAM J. Comput.* 2: 60–66

Pippenger, N. 1979. On simultaneous resource bounds. *Proc. 20th Ann. IEEE Symp. Foundat. Comput. Sci.*, pp. 307–11

Plaisted, D. A. 1984. New NP-hard and NP-complete polynomial and integer divisibility problems. *Theor. Comput. Sci.* 31: 125–38

Rabin, M. O. 1980. Probabilistic algorithms in finite fields. *SIAM J. Comput.* 9: 273–80

Reif, J. H. 1979. Complexity of the mover's problem and generalizations. *Proc. 20th Ann. IEEE Symp. Found. Comput. Sci.*, San Juan, Puerto Rico, pp. 421–27

Risler, J. J. 1985. Additive complexity and zeros of real polynomials. *SIAM J. Comput.* 14: 178–83

Risler, J.-J. 1988. Some aspects of complexity in real algebraic geometry. *J. Symb. Comput.* 5: 109–19

Rivest, R. L., Shamir, A., Adleman, L. 1978. A method for obtaining digital signatures and public-key cryptosystems. *Commun. Assoc. Comput. Mach.* 21: 120–26

Robinson, J. 1949. Definability and decision problems in arithmetic. *J. Symb. Logic* 14: 98–114

Rónyai, L. 1987a. Simple algebras are difficult. *Proc. 19th Ann. ACM Symp. Theory of Comput.*, New York, pp. 398–408. *J. Algorithms*. In press

Rónyai, L. 1987b. Factoring polynomials over finite fields. *Proc. 28th IEEE Symp. Found. Comput. Sci., Los Angeles*, pp. 132–37

Ruzzo, W. L. 1981. On uniform circuit complexity. *J. Comput. Syst. Sci.* 22: 365–83

Savage, J. E. 1974. An algorithm for the computation of linear forms. *SIAM J. Comput.* 3: 150–58

Schnorr, C. P. 1978. Improved lower bounds on the number of multiplications/divisions which are necessary to evaluate polynomials. *Theor. Comput. Sci.* 7: 251–61

Schnorr, C. P. 1981. An extension of Strassen's degree bound. *SIAM J. Comput.* 10: 371–82

Schnorr, C. P., Van de Wiele, J. P. 1980. On the additive complexity of polynomials. *Theor. Comput. Sci.* 10: 1–18

Scholz, A. 1937. Aufgabe 253. *Jahresber. DMV* 47: 41–42

Schönhage, A. 1971. Schnelle Berechnung von Kettenbruchentwicklungen. *Acta Inf.* 1: 139–44

Schönhage, A. 1973. Unitäre Transformationen großer Matrizen. *Numer. Math.* 20: 409–17

Schönhage, A. 1975. A lower bound for the length of addition chains. *Theor. Comput. Sci.* 1: 1–12

Schönhage, A. 1976. An elementary proof for Strassen's degree bound. *Theor. Comput. Sci.* 3: 267–72

Schönhage, A. 1977. Schnelle Multiplikation von Polynomen über Körpern der Charakteristik 2. *Acta Inf.* 7: 395–98

Schönhage, A. 1981. Partial and total matrix multiplication. *SIAM J. Comput.* 10: 434–55

Schönhage, A. 1984. Factorization of univariate integer polynomials by diophantine approximation and an improved basis reduction algorithm. *Proc. ICALP. Springer Lect. Notes Comput. Sci.* 172, pp. 436–47

Schönhage, A., Strassen, V. 1971. Schnelle Multiplikation großer Zahlen. *Computing* 7: 281–92

Schoof, R. J. 1985. Elliptic curves over finite fields and the computation of square roots mod *p*. *Math. Comput.* 44: 483–94

Schwartz, J. T., Sharir, M. 1983a. On the "piano movers'" problem. I. The case of a two-dimensional rigid polygonal body moving amidst polygonal barriers. *Commun. Pure Appl. Math.* 36: 345–98

Schwartz, J. T., Sharir, M. 1983b. On the "piano movers" problem. II. General techniques for computing topological properties of real algebraic manifolds. *Adv. Appl. Math.* 4: 298–351

Schwarz, S. 1956. On the irreducibility of polynomials over a finite field. *Q. J. Math. Oxford* 7: 110–24

Seidenberg, A. 1954. A new decision method for elementary algebra. *Ann. Math.* 60: 365–74

Sieveking, M. 1972. An algorithm for division of powerseries. *Computing* 10: 153–56

Sims, C. C. 1970. Computational methods in the study of permutation groups. In *Computational Problems in Abstract Algebra*, ed. J. Leech, pp. 169–83. Oxford: Pergamon

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

Stoß, H.-J. 1985. The complexity of evaluating interpolation polynomials. *Theor. Comput. Sci.* 41: 319–23

Stoß, H.-J. 1986. Lower bounds on the complexity of rational functions. Preprint, Universität Konstanz. 39 pp.

Strassen, V. 1969. Gaussian elimination is not optimal. *Numer. Math.* 13: 354–56

Strassen, V. 1972a. Evaluation of rational functions. In *Complexity of Computer Computations*, ed. R. E. Miller, J. W. Thatcher, J. D. Bonlinger. New York: Plenum

Strassen, V. 1972b. Berechnung und Programm. I. *Acta Inf.* 1: 320–35

Strassen, V. 1973a. Die Berechnungskomplexität von elementarsymmetrischen Funktionen und von Interpolationskoeffizienten. *Numer. Math.* 20: 238–51

Strassen, V. 1973b. Vermeidung von Divisionen. *J. Reine Angew. Math.* 264: 182–202

Strassen, V. 1974. Polynomials with rational coefficients which are hard to compute. *SIAM J. Comput.* 3: 128–49

Strassen, V. 1976. Computational complexity over finite fields. *SIAM J. Comput.* 5: 324–31

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

Strassen, V. 1984. Algebraische Berechnungskomplexität. In *Perspectives in Mathematics*. Basel: Birkhäuser Verlag

Strassen, V. 1986. The work of L. G. Valiant. *Proc. Int. Cong. Math., Berkeley*

Strassen, V. 1987. Relative bilinear complexity and matrix multiplication. *J. Reine Angew. Math.* 375: 406–43

Tarski, A. 1948. *A Decision Method for Elementary Algebra and Geometry*. Berkeley: Univ. Calif. Press. 63 pp.

Valiant, L. G. 1979a. Completeness classes in algebra. *Proc. 11th Ann. ACM Symp. Theory Comput., Atlanta*, pp. 249–61

Valiant, L. G. 1979b. The complexity of computing the permanent. *Theor. Comput. Sci.* 8: 189–201

Valiant, L. G. 1982. Reducibility by algebraic projections. In *Logic and Algorithmic*, Symposium in honour of Ernst Specker. *Enseign. Math.* 30: 365–80

Valiant, L., Skyum, S., Berkowitz, S., Rackoff, C. 1983. Fast parallel computation of polynomials using few processors. *SIAM J. Comput.* 12: 641–44

van der Waerden, B. L. 1928. Eine Verallgemeinerung des Bézoutschen Theorems. *Math. Ann.* 99: 497–541

van der Waerden, B. L. 1930. Eine Bemerkung über die Unzerlegbarkeit von Polynomen. *Math. Ann.* 102: 738–39

van der Waerden, B. L. 1938. Eine Bemerkung zur numerischen Berechnung von Determinanten und Inversen von Matrizen. *Jahresber. DMV* 48: 29–30

Wang, P. S. 1978. An improved multivariate polynomial factoring algorithm. *Math. Comput.* 32: 1215–31

Winograd, S. 1970. On the number of multiplications necessary to compute certain functions. *Commun. Pure Appl. Math.* 23: 165–79

Winograd, S. 1977. Some bilinear forms whose multiplicative complexity depends on the field of constants. *Math. Syst. Theory* 10: 169–80

Winograd, S. 1980. *Arithmetic Complexity of Computations*. SIAM Reg. Conf. Ser. Appl. Math. 33. Philadelphia: SIAM. 93 pp.

Wüthrich, H. R. 1976. Ein Entscheidungsverfahren für die Theorie der reell-abgeschlossenen Körper. In *Komplexität von Entscheidungsproblemen. Springer Lect. Notes Comput. Sci.*, ed. E. Specker, V. Strassen, 43: 138–62

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

Zippel, R. 1979. Probabilistic algorithms for sparse polynomials. *Proc. EUROSAM*, Marseille, pp. 216–26

Zippel, R. 1981. Newton's iteration and the sparse Hensel algorithm. *Proc. 1981 ACM Symp. Symb. Algebraic Comput., Snowbird, Utah*, pp. 68–72