Proceedings of the International Congress of Mathematicians
Vancouver, 1974

Some Results in Algebraic Complexity Theory

Volker Strassen

1. Summary. The minimal number of multiplications/divisions involved in various problems of symbolic manipulation of polynomials and rational functions is investigated.

2. Definitions. A finite set of rational functions can always be computed (i.e., evaluated) without loss in efficiency on most inputs by a program not containing any branching instructions. The sequence of intermediate results produced by implementing such a program on an idealized computer is called a computation. All elements of a computation are rational functions in the input variables. Formally, we have

DEFINITION 1. Let k be an infinite field, $x_1, \cdots, x_m$ indeterminates over k. A finite sequence β from $k(x_1, \cdots, x_m)$ is called a computation in $k(x_1, \cdots, x_m)$ iff each element of β is either an indeterminate, or an element of k, or is obtained from two previous elements by applying addition, subtraction, multiplication or division. β computes a finite set $\{f_1, \cdots, f_r\}$ of rational functions iff each f_i occurs in β.

The indeterminates are interpreted as inputs; the elements of k appearing in β are thought of as being stored in the program.

The running time of a program will depend on how long it takes the computer to perform the various arithmetic operations. For mathematical convenience we will assume that k-linear operations are instantaneous. Thus we define the running time or length of a computation β as the number of elements of β which are neither indeterminates nor are k-linearly dependent on the set of previous elements. If k has characteristic 0, most of the results below remain correct when all multiplications and divisions are counted.

DEFINITION 2 (OSTROWSKI [15]). Let $f_1, \cdots, f_r \in k(x_1, \cdots, x_m)$. $L(f_1, \cdots, f_r) :=$ minimal length of computations in $k(x_1, \cdots, x_m)$ which compute $\{f_1, \cdots, f_r\}$ is called the complexity of $\{f_1, \cdots, f_r\}$.

Upper bounds for the complexity are usually proved by exhibiting an algorithm. Thus Horner's rule implies

$$L(a_0x^n + \cdots + a_n) \leqq n, \tag{1}$$

considered in $k(a_0, \cdots, a_n, x)$.

3. Pan's method. In the above-mentioned paper Ostrowski conjectured that we have equality in (1). This was proved 12 years later by Pan **[16]** by an elementary but ingenious method, which consists in substituting for one indeterminate a linear combination of the others and looking at the effect of this substitution on the first nonlinear operation of a given computation. Pan's method can be successfully applied to a surprising number of simple computational problems "with general coefficients", such as the evaluation of several polynomials, of a polynomial in several indeterminates, of a homogeneous polynomial, of the product of a vector by a matrix, or of a continued fraction (see Winograd **[24]**, **[25]**, Borodin-Munro **[2]**, Strassen **[20]**). Also an arbitrary single quadratic form can be treated, as long as char $k \neq 2$. Unfortunately, the lower bounds derived by Pan's method cannot exceed the number of inputs to the problem.

4. Nonlinear lower bounds. In the sequel, $f \sim g$, $f \asymp g$, and $f \prec g$ mean respectively that f and g are asymptotically equal, f and g have the same order of magnitude, $f = O(g)$. All logarithms are to the base 2. The proofs of the lower bounds in this and the next section use some algebraic geometry. They can be found in Strassen **[21]**, **[22]** and **[23]**.

Let us first consider the problem of computing the set of elementary symmetric functions in n variables:

$$\sigma_1 := \textstyle\sum x_i, \quad \sigma_2 := \sum_{i<j} x_i x_j, \quad \cdots, \quad \sigma_n := x_1 \cdots x_n.$$

Clearly $L(\sigma_1) = 0$. Pan's method yields $L(\sigma_2) = n - 1$ for $k = \boldsymbol{R}$ and $L(\sigma_n) = n - 1$. Also $L(\sigma_1, \cdots, \sigma_n) \leqq n \log n$ (Horowitz **[8]**).

THEOREM 1. $L(\sigma_1, \cdots, \sigma_n) \sim n \log n$.

More generally one has

THEOREM 2. *Let F be a finite set of symmetric rational functions of transcendency degree t over k. Then $L(F) \geqq t \log(t/e)$. E.g., if* char $k = 0$ *and $s_\rho := \sum_i x_i^\rho$ then $L(s_1, \cdots, s_n) \sim n \log n$.*

Horner's rule is optimal for evaluating a general polynomial at one point. Is it also optimal for evaluating such a polynomial at many (say $n + 1$) general points? In other words, what is the complexity of $y_0, \cdots, y_n$ in $k(a_0, \cdots, a_n, x_0, \cdots, x_n)$, where

$$y_0 = a_0x_0^n + \cdots + a_n, \quad \cdots, \quad y_n = a_0x_n^n + \cdots + a_n? \tag{2}$$

Surprisingly, separate evaluation using Horner's rule is not optimal (Borodin and

Munro [1]). One even has the following drastic result

$$L(y_0, \cdots, y_n) \prec n \log n$$

(Fiduccia [7], Moenck and Borodin [14], amended by Sieveking [19], Strassen [21]; see also Kung [11] and the result of S. Cook in Knuth [9, p. 275]).

THEOREM 3. *$L(y_0,\cdots, y_n) \geqq (n + 1) \log n$, and therefore $L(y_0,\cdots,y_n) \asymp n \log n$.*

In contrast to Pan's result Theorem 3 remains true if $a_0,\cdots, a_n$ are replaced by arbitrary elements $\alpha_0,\cdots, \alpha_n \in k$, as long as $\alpha_0 \neq 0$. So, e.g., $L(x_0^n,\cdots,x_n^n) \sim n \log n$.

The inverse problem to evaluation is interpolation. Here inputs $x_0,\cdots, x_n$, $y_0, \cdots, y_n$ are given and the coefficients $a_0,\cdots, a_n$ of the unique polynomial of degree n that interpolates y_i at x_i are to be computed. Equivalently, $a_0,\cdots, a_n$ can be defined by (2), where now $x_0, \cdots, x_n, y_0,\cdots, y_n$ are interpreted as indeterminates. Again one has $L(a_0,\cdots, a_n) \prec n \log n$ (Horowitz [8], Moenck and Borodin [14]; see also Strassen [21]).

THEOREM 4. *$L(a_0,\cdots, a_n) \geqq (n + 1) \log n$, and therefore $L(a_0,\cdots, a_n) \asymp n \log n$.*

As it happens, several of the previous results are concerned with the computational complexity of going from one representation of a univariate polynomial to another: Computing the elementary symmetric functions means computing the coefficients from the roots; evaluation and interpolation relate the coefficient representation to the representation by a list of values (at $n + 1$ points). Our methods apply to several similar problems. Going from the set of roots to a list of values, going from one list of values to a new one, differentiating or integrating a polynomial given by a list of values all have a complexity of order of magnitude $n \log n$. On the other hand, one can expand a polynomial at a new point in linear time (Shaw and Traub [18]).

The problems discussed here belong to the field of symbolic manipulation (Collins [6]). Because of the constant use of modular algorithms in this area, evaluation and interpolation are of special importance. Usually one is interested in the case $k = Z_p$, since one has already applied modular reductions to integer coefficients (see Brown [4] for a typical situation). In many cases neither the base points for evaluation and interpolation nor the primes p to be used are known in advance. Thus apart from treating the base points as inputs (as we do in this paper) one has to look for algorithms that work over any Z_p (or at least over any Z_p with p not too small). Now it is easy to see that, roughly speaking, such algorithms are equivalent to algorithms over $\boldsymbol{Q}$. Since $\boldsymbol{Q}$ is an infinite field, the results of this paper apply (see Strassen [23] for a detailed discussion).

5. A problem involving branching. Let A_0, A_1 be univariate polynomials over a field k such that $n := \deg A_0 \geqq \deg A_1 \geqq 0$. For simplicity assume char $k = 0$ (but the remarks at the end of the last section apply here too). Euclid's algorithm

$$A_0 = Q_1A_1 + A_2, \quad A_1 = Q_2A_2 + A_3, \quad \cdots, \quad A_{t-1} = Q_tA_t,$$

with $\deg A_i > \deg A_{i+1}$ for $i \geqq 1$ yields the Euclidean representation $(Q_1,\cdots, Q_t, A_t)$

of the pair (A_0, A_1). From this representation one can read off several important items: the continued fraction of A_0/A_1, the greatest common divisor of A_0 and A_1, the resultant of A_0 and A_1 (Collins [5]), the discriminant of A_0 if $A_1 = A_0'$, the number of zeroes of A_0 in an arbitrary interval if $A_1 = A_0'$ and if k is the field of real numbers (Sturm). Improving the work of Lehmer [12] and Knuth [10], Schönhage [17] computes the coefficients of $Q_1, \cdots, Q_t, A_t$ from the coefficients of A_0, A_1 with $\prec n \log n$ multiplications and divisions (actually these papers are concerned with the analogous problem in number theory; the translation to polynomials is due to Moenck [13]).

Size and shape of the output $(Q_1, \cdots, Q_t, A_t)$ is determined by its sequence of degrees $\boldsymbol{d} := (d_1, \cdots, d_t, d_{t+1})$. Since $\boldsymbol{d}$ depends on the input polynomials A_0, A_1, every algorithm for computing the Euclidean representation has to use branching instructions, say of the form "if $f = 0$ then go to i else go to j", where f has been previously computed. Let $M_{\boldsymbol{d}}$ be the set of inputs for which the output has shape $\boldsymbol{d}$ and let $H(\boldsymbol{d})$ be the entropy of the probability vector that is obtained from $\boldsymbol{d}$ by normalization.

THEOREM 5. *There are constants* $0 < c < c'$ *with the following properties:*

(1) *For all* $\boldsymbol{d}$ *Schönhage's algorithm takes* $< c'n(H(\boldsymbol{d}) + 1)$ *multiplications and divisions on* $M_{\boldsymbol{d}}$.

(2) *For all* $\boldsymbol{d}$ *any algorithm that computes the Euclidean representation takes* $>$ $cn(H(\boldsymbol{d}) + 1)$ *multiplications and divisions on some input of* $M_{\boldsymbol{d}}$.

Thus, roughly speaking, Schönhage's algorithm is uniformly optimal. We remark that although branching instructions themselves are not counted, every multiplication and division is counted, even if it serves only to prepare a branching instruction.

To a reader, who is interested in a detailed treatment of algebraic complexity theory, we suggest the book by Borodin and Munro [3].

References

1. A. Borodin and I. Munro, Information Processing Lett. **1** (1971), 66–68.

2. ———, J. Comput. System Sci. **6** (1972).

3. ———, *Computational complexity of algebraic and numeric problems*, American Elsevier, New York, 1975.

4. W. S. Brown, *On Euclid's algorithm and the computation of polynomial greatest common divisors*, J. Assoc. Comput. Mach. **18** (1971), 478–504. MR **46** #6570.

5. G. E. Collins, *Subresultants and reduced polynomial remainder sequences*, J. Assoc. Comput. Mach. **14** (1967), 128–142. MR **35** #6352.

6. ———, *Computer algebra of polynomials and rational functions*, Amer. Math. Monthly **80** (1973), 725–755. MR **48** #2106.

7. C. Fiduccia, Proc. Fourth Sympos. on the Theory of Computing, ACM, New York, 1972, pp. 88–93.

8. Ellis Horowitz, *A fast method for interpolation using preconditioning*, Information Processing Lett. **1** (1971/72), 157–163. MR **47** #4413.

9. D. E. Knuth, *The art of computer programming*. Vol. II: *Seminumerical algorithms*, Addison-Wesley, Reading, Mass., 1969. MR **44** #3531.

10. D. E. Knuth, *The analysis of algorithms*, Proc. Internat. Congress Math. (Nice, 1970), vol. 3, Gauthier-Villars, Paris, 1971, pp. 269–274.

11. H. T. Kung, *On computing reciprocals of power series*, Dept. of Comput. Sci., Carnegie-Mellon University, Sept. 1973.

12. D. H. Lehmer, Amer. Math. Monthly **45** (1937), 227–233.

13. R. Moenck, Proc. Fifth Sympos. on Theory of Computing, ACM, New York, 1973, pp. 142–151.

14. R. Moenck and A. Borodin, Proc. Thirteenth IEEE Sympos. on Switching and Automata Theory, Oct. 1972, pp. 90–96.

15. A. M. Ostrowski, *On two problems in abstract algebra connected with Horner's rule*, Studies in Math. Mech., Presented to Richard von Mises, Academic Press, New York, 1954, pp. 40–48. MR **16**, 523.

16. V. Ja. Pan, *On means of calculating values of polynomials*, Uspehi Mat. Nauk **21** (1966), no. 1 (127), 103–134 = Russian Math. Surveys **21** (1966), no. 1, 105–136.

17. A. Schoenhage, Acta Informatica **1** (1971), no. 1, 139–144.

18. M. Shaw and J. F. Traub, Proc. Thirteenth IEEE Sympos. on Switching and Automata Theory, Oct. 1972, pp. 105–107.

19. M. Sieveking, *An algorithm for division of power series*, Computing (Arch. Elektron. Rechnen) **10** (1972), 153–156. MR **47** #1257.

20. V. Strassen, *Complexity of computer computations*, Plenum Press, New York, 1972, pp. 1–10.

21. ———, Numer. Math. **20** (1973), 238–251.

22. ———, *Die Berechnungskomplexität der symbolischen Differentiation von Interpolationspolynomen*, Theoretical Computer Science, 1975.

23. ———, *The computational complexity of continued fractions* (in preparation).

24. S. Winograd, *On the number of multiplications necessary to compute certain functions*, Comm. Pure Appl. Math. **23** (1970), 160–170. MR **41** #4778.

25. ———, Proc. Internat. Congress Math. (Nice, 1970), vol. 3, Gauthier-Villars, Paris, 1971, pp. 269–274.

UNIVERSITY OF ZURICH
ZURICH, SWITZERLAND