Boolean Circuit Complexity of Algebraic Interpolation Problems

Marek Karpinski[1]

TR-89-027

May, 1989

Abstract

We present here some recent results on fast parallel interpolation of multivariate polynomials over finite fields. Some applications towards the general conversion algorithms for boolean functions are also formulated.

1 Dept. of Computer Science, University of Bonn and International Computer Science Institute, Berkeley, California. Supported in part by Leibniz Center for Research in Computer Science, by the DFG Grant KA 673/2-1, and by the SERC Grant GR-E 68297.

Introduction

We consider the general problem of interpolation of multivariate polynomials over finite fields given by black boxes (input oracles). In this setting we are given a polynomial f over $\mathrm{GF}[q]$, as a black box, and an information about its sparsity t (the bound on the number of nonzero coefficients). Given this, we must determine an extension $\mathrm{GF}[q^s]$ of $\mathrm{GF}[q]$, s as small as possible, and an efficient (deterministic boolean NC-algorithm, cf. [Co 85], [KR 88]) interpolation algorithm working over $\mathrm{GF}[q^s]$ to determine all coefficients of f in $\mathrm{GF}[q]$. Such a general problem arises in a number of applications, e.g., in design of efficient algorithms in algebra, coding theory and combinatorial optimization (cf. [Ga 83], [Ga 84], [Ka 85], [KT 88], [MS 72], [GK 87], [BT 88], [GKS 88]). The interest in the parallel (boolean circuit) complexity of this problem has arisen recently in connection with the design of fast parallel algorithms for the perfect matching problem [GK 87]. [GK 87] gave the first deterministic algorithm for sparse interpolation of determinants over fields of characteristic 0, and [BT 88] extended it to the case of arbitrary sparse polynomials over fields of characteristic 0.

Following [GK 87], and [GKS 88] we shall use uniform boolean circuits in our analysis. Given a (fixed) finite field $\mathrm{GF}[q]$. We say that the *black box* Interpolation Problem (over a finite field extension $\mathrm{GF}[q^s]$) is in NC^k (cf. [Co 85], [KR 88]), if there exists a class of uniform $(ntq)^{O(1)}$-size and $O(\log^k(ntq))$-depth boolean circuits with oracle nodes S (*returning* values of a black box over the field extension $\mathrm{GF}[q^s]$) computing for an arbitrary n-variate polynomial $f \in \mathrm{GF}[q][x_1, \ldots, x_n]$ all the nonzero coefficients and monomial vectors of f. The oracle S_f^s is defined by $S_f^s(x_1, \ldots, x_n, y)$ iff $f(x_1, \ldots, x_n) = y$ over $\mathrm{GF}[q^s]$. If the *lifting* of a black box (given explicitly by a straight-line program, determinant, boolean circuit, etc.) from the field $\mathrm{GF}[q]$ to the extension $\mathrm{GF}[q^s]$, and the computation of the value $f(x_1, \ldots, x_n)$ over $\mathrm{GF}[q^s]$ by a black box, are both in boolean NC (in P), then the explicit Interpolation Problem lies also in boolean NC (in P).

The reader is referred to [LN 83], [MS 72] for the basic algorithms for finite fields, and to [Co 85], [KR 88] for the basic models of parallel computation.

1. Lower Bounds

We shall state first a result on the number of queries necessary to interpolate a sparse polynomial $f \in \mathrm{GF}[q][x_1, \ldots, x_n]$ over $\mathrm{GF}[q]$ (i.e., for the case of $s = 1$).

Theorem 1. **([CDGK 88])** *Given an arbitrary finite field* $\mathrm{GF}[q]$ *and a* t*-sparse polynomial* $f \in \mathrm{GF}[q][x_1, \ldots, x_n]$ *given by a black box input oracle, any algorithm for testing whether* $f \equiv 0$ *requires* $\Omega(n^{\log t})$ *queries to the input oracle.*

PROOF. Define the following *search space* $X(n,t)$ for the class of t-sparse polynomials $f \in \mathrm{GF}[2][x_1, \ldots, x_n]$,

$$\begin{aligned} X(n,t) = \{\, A \mid {} & A \subseteq \{0,1\}^n\,, \\ & (\forall f \in \mathrm{GF}[q][x_1, \ldots, x_n],\ f \text{ is } t\text{-sparse},\ f \not\equiv 0)\,(\exists a \in A)\,[f(a) \neq 0]\,\}. \end{aligned}$$

Denote the minimal cardinality of the set A in $X(n,t)$ by $d(n,t)$,

$$d(n,t) = \min\{\#A \mid A \in X(n,t)\}.$$

Please note that $d(n,t)$ is the minimal number of queries to the input oracle needed by any adaptive or nonadaptive algorithm for testing whether $f/equiv0$.

For a 0-1 vector $x = (x_1, \ldots, x_n) \in \{0,1\}^n$, define the set $x^{-1}(0) = \{\, i \mid x_i = 0\,\}$. Now for every subset of indices $T \subseteq \{1, \ldots, n\}$ such that $\#T \leq \lfloor \log t \rfloor$ define a polynomial

$$f_T = \prod_{i \in T} (x_i \oplus 1) \prod_{j \notin T} x_j.$$

Denote this class of polynomials by $\mathcal{F}$. The polynomials $f_T \in \mathcal{F}$ have the following properties: (i) f_T's are t-sparse, and (ii) $f_T(x) \neq 0$ iff $x^{-1}(0) = T$.

Because of the properties (i) and (ii) above, an arbitrary search set $A \in X(n,t)$ distinguishing all f-sparse polynomials $f \in \mathrm{GF}[q][x_1, \ldots, x_n]$ from zero, must contain all vectors $x \in \{0,1\}^n$ such that $x^{-1}(0) = T$ for all sets $T \subseteq \{1, \ldots, n\}$ with cardinality $\#T \leq \lfloor \log t \rfloor$. Thus the cardinality of such a set is bounded from below by $\sum_{i=1}^{\lfloor \log t \rfloor} \binom{n}{i}$, and consequently $d(n,t) = \Omega(n^{\log t})$, which proves our theorem. $\square$

For the important case of boolean functions (GF[2]) we are able to prove the tight lower and upper bounds $\Theta(n^{\log t})$ for the number of queries necessary to determine identity to zero of t-sparse polynomials $f \in \mathrm{GF}[2][x_1, \ldots, x_n]$.

An Algorithm for GF[2] ([CDGK 88])

Input: t-sparse polynomial $f \in \mathrm{GF}[2][x_1, \ldots, x_n]$. given by a black box input oracle;

Output: **Yes**, if $f \equiv 0$; **No**, if $f \not\equiv 0$.

Step 1: **For all** n-bit vectors $v \in \{0,1\}^n$, having at most $\lfloor \log_2 t \rfloor$ zeros compute the values $\alpha_v = f(v)$.

Step 2: Output **Yes** iff $\forall v \, [\, \alpha_v = 0 \,]$.

The correctness of the above algorithm was proven in [CDGK 88]. We do not know whether the result generalizes for the arbitrary finite field GF[q]. We note that we deal here not only with interpolation of polynomials but arbitrary functions in their RSE-representation ([We 87]).

For arbitrary boolean functions $f : \{0,1\}^n \to \{0,1\}$ there exists exactly one {0,1}-vector $S = (S_A)_{A \subseteq \{1,\ldots,n\}}$ such that

$$f(x) = \bigoplus_{A \subseteq \{1,\ldots,n\}} S_A \wedge \bigwedge_{i \in A} x_i$$

for $\bigoplus$ boolean *XOR* and $\bigwedge$ boolean *AND*.

The size of the vector S is referred to as the size (RSE(f)) of f in its RSE-representation (cf. [We 87]) (and is in our framework *exactly* its sparsity over $\mathrm{GF}[2][x_1, \ldots, x_n]$).

Theorem 2. **([CDGK 88])** *Given an arbitrary boolean function f by the black box input oracle, there exists an algorithm for deciding over* GF[2] *whether $f \equiv 0$ using $O(n^{\log(RSE(f))})$ queries to the oracle. The algorithm is optimal with respect to the number of queries to the oracle over* GF[2] *taken by any (adaptive or non-adaptive) algorithm for this problem.*

The lower bounds of this Section proves the impossibility of polynomial time (and NC-) algorithms for the general sparse polynomial interpolation with input oracles over finite fields without proper field extensions. So the intriguing question arises whether we can do interpolation over finite fields at all - without going to the 'impossible' field extension $\mathrm{GF}[q^n]$ (where there are no effective deterministic procedures known even for finding primitive elements!).

In the next section we shall present surprising upper bounds on the Interpolation Problem using only '*slight*' (logarithmic in nt) extensions of a ground field.

2. Upper Bounds

We formulate now our main Interpolation result on the *slight* field extensions.

Theorem 3. **([GKS 88])** *Given any t-sparse polynomial $f \in \mathrm{GF}[q][x_1, \ldots, x_n]$ by the black box input oracle. There exists a deterministic parallel algorithm* (NC^3) *for interpolating f over a slight field extension $\mathrm{GF}[q^{\lceil 2\log_q(nt)+3\rceil}]$ working in $O(\log^3(ntq))$ parallel boolean time and $O(n^2t^6\log^2(ntq) + q^{2.5}\log^2 q)$ processors. For the fixed field $\mathrm{GF}[q]$, the algorithm works in $O(\log^3(nt))$ parallel boolean time and $O(n^2t^6\log^2(nt))$ processors.*

The algorithm discovered in [GKS 88] involves two major computational steps: (1) breaking the zero identity problem of polynomials over a slight field extension $\mathrm{GF}[q^{\lceil 2\log_q(nt)+3\rceil}]$, and (2) inductive enumeration of all partial solutions for terms and coefficient vectors over $\mathrm{GF}[q]$ by means of recursion using (1).

We develop here a new general method involving Cauchy ([C]) matrices to break zero-identity problem in Step 1, and combine it with the new parallel enumeration method based on [GK 87] to solve Step 2. The number of queries to the input oracle over the slight field extension $\mathrm{GF}[q^{\lceil 2\log_q(nt)+3\rceil}]$ is bounded by $t(1+(n-1)\binom{t}{2})$ $(= O(nt^3))$.

We shall investigate here in more detail the problem of checking identity to zero (Step 1) in order to compare it with the results of Section 1. (The method of Cauchy matrices applied here could be also of independent interest.)

Definition. (Cauchy matrix) ([C])
An $(N \times N)$ matrix $C = [c_{ij}]$ over the field $\mathrm{GF}[q]$ is called a Cauchy matrix, if

$$c_{ij} = \frac{1}{x_i + y_j}$$

for the fixed values $x_i, y_j \in \mathrm{GF}[q]$, $1 \le i, j \le N$.

Lemma. (cf., e.g. [MS 72]). *Let C be a Cauchy matrix, then the determinant*

$$Det(C) = \frac{\prod_{1 \le i < j \le N}(x_j - x_i)(y_j - y_i)}{\prod_{1 \le i,j \le N}(x_i + y_j)}$$

For any of its minors $\neq 0$ a similar formula holds. Therefore any minor of any size is nonsingular.

In our algorithm we construct the Cauchy matrix $C = [c_{ij}]$ by

$$c_{ij} = \frac{1}{i+j} \bmod p$$

where p is a prime.

An Algorithm for (the slight field extension) $\mathrm{GF}[q^{O(\log(nt))}]$ ([GKS 88])

Input: t-sparse polynomial $f \in \mathrm{GF}[q][x_1, \ldots, x_n]$ given by a black box input oracle;

Output: **Yes**, if $f \equiv 0$; **No**, if $f \not\equiv 0$.

Step 1: Determine a minimal s satisfying

$$q^s - 1 > 4nq(n-1)\binom{t}{2}.$$

So take $s = \lceil \log_q(nt) + 3 \rceil$.

Step 2: Construct the field $\mathrm{GF}[q^s]$ and a primitive element ω in $\mathrm{GF}[q^s]$ with the help of Berlekamp Algorithm [Be 70].

Step 3: Let $N = \frac{[q^s-1]}{4nq}$. Use the sieve of Erastosthenes to find a prime p with $2N < p \leq 4N$.

Step 4: Construct an $N \times N$ Cauchy matrix $C = [c_{ij}]$ by $c_{ij} = \frac{1}{i+j} \bmod p$, $1 \leq i, j \leq N$ by means of the Euclidean algorithm.

Step 5: Construct an arbitrary submatrix $\bar{C} = [\bar{c}_{ij}]$ of C of size $N \times n$.

Step 6: Query in parallel the black box for any row $\bar{c}_i = (\bar{c}_{ij})$, $1 \leq j \leq n$, of the matrix $\bar{C}$, and for each l, $0 \leq l < t$, at the points

$$\alpha_{l_i} = \omega^{l \cdot \bar{c}_i} = (\omega^{l \cdot \bar{c}_{i1}}, \omega^{l \cdot \bar{c}_{i2}}, \ldots, \omega^{l \cdot \bar{c}_{in}})$$

and at the zero point $\alpha_{0_0} = (0, \ldots, 0)$.

Step 7: Output **Yes** $(f \equiv 0)$ iff $\forall 0 \leq l < t, 0 \leq i \leq N\, [\, \alpha_{l_i} = 0\,]$.

The correctness proof of this algorithm is given below. The main reason for the algorithm to work is the strong *term separation* property of a Cauchy matrix constructed in Step 3 (the existence of a row $\bar{c}_i$ of a Cauchy matrix C separating arbitrary two monomials of f under $\omega^{l \cdot \bar{c}_i}$ substitution to the black box oracle). The Cauchy matrix design can be also viewed as a algebraic *deterministic coin flipping* method for constructing a witness value for the fact $f \not\equiv 0$ without actually using randomness.

We prove now the correctness of the Algorithm.

Theorem 4. *Given any fixed finite field* $\mathrm{GF}[q]$ *and a* t*-sparse polynomial* $f \in \mathrm{GF}[q][x_1, \ldots, x_n]$ *by the black box input oracle, there exists a deterministic parallel algorithm* (NC^2) *for interpolating* f *over* $\mathrm{GF}[q^{\lceil 2\log_q(nt)+3 \rceil}]$ *working in* $O(\log^2(nt))$ *parallel boolean time and* $O(n^2 t^3 \log^2(nt))$ *processors, and making* $O(nt^3)$ *queries to the black box input oracle.*

PROOF. Suppose $f(0,\ldots,0)=0$, otherwise $f \not\equiv 0$. We are going to prove that $f \not\equiv 0$ iff there exists l and i, $0 \le l < t$, $1 \le i \le N$, such that $f(\omega^{l\cdot\bar{c}_i}) \neq 0$. Suppose $f = \sum f_{\underline{k}} x^{\underline{k}}$, for $\underline{k} = (k_1,\ldots,k_n) \in \{0,\ldots,q-1\}^n$ the multi-indices. We show that for a certain vector $\bar{c}_i$, $1 \le i \le N$, after substituting $\omega^{\bar{c}_{ij}}$ for x_j's in monomials $x^{\underline{k}}$, any two monomials $x^{\underline{k}_1}$, $x^{\underline{k}_2}$, would yield two different elements of GF$[q]$.

Suppose that for a pair $\underline{k}_1 = (k_1^{(1)},\ldots,k_n^{(1)})$, $\underline{k}_2 = (k_1^{(2)},\ldots,k_n^{(2)})$, and $\bar{c}_i$ we have $\omega^{\bar{c}_i\cdot\underline{k}_1} = \omega^{\bar{c}_i\cdot\underline{k}_2}$. This means that $\sum k_j^{(1)}\bar{c}_{ij} \equiv \sum k_j^{(2)}\bar{c}_{ij} \pmod{q^s-1}$ and so $\sum(k_j^{(1)}-k_j^{(2)})\bar{c}_{ij} \equiv 0 \pmod{q^s-1}$. Since $|k_j^{(1)}-k_j^{(2)}| \le q-1$, and $\bar{c}_{ij} < 4N$, we have $|\sum_{1\le j\le n}(k_j^{(1)}-k_j^{(2)})\bar{c}_{ij}| < (q-1)n4N < (q^s-1)$. Therefore we have $\sum(k_j^{(1)}-k_j^{(2)})\bar{c}_{ij} = 0$.

Now for any pair of monomials $x^{\underline{k}_1}$, $x^{\underline{k}_2}$, we consider "bad" vectors $\bar{c}_i$, $1 \le i \le N$, such that $\sum_{1\le j\le n}(k_j^{(1)}-k_j^{(2)})\bar{c}_{ij} = 0$. There cannot be more than $(n-1)$ "bad" vectors for this pair. Suppose there exist n "bad" vectors $\bar{c}_{i_1},\ldots,\bar{c}_{i_n}$. Then the $n\times n$ submatrix of $\bar{C}$ build from the vectors $\bar{c}_{i_1},\ldots,\bar{c}_{i_n}$ would have determinant zero, since the following equality

$$\underbrace{\begin{pmatrix} \bar{c}_{i_1,1} & \cdots & \bar{c}_{i_1,n} \\ \vdots & & \vdots \\ \bar{c}_{i_n,1} & \cdots & \bar{c}_{i_n,n} \end{pmatrix}}_{:=C} \cdot \begin{pmatrix} k_1^{(1)}-k_1^{(2)} \\ \vdots \\ k_n^{(1)}-k_n^{(2)} \end{pmatrix} = 0$$

enforces singularity of C.

Since f is t-sparse there are at most $\binom{t}{2}$ pairs of monomials. Thus there are at most $(n-1)\binom{t}{2}$ "bad" vectors for all pairs of monomials $x^{\underline{k}_1}$, $x^{\underline{k}_2}$. Because $(n-1)\binom{t}{2} = \frac{4nq(n-1)\binom{t}{2}}{4nq} < \frac{[q^s-1]}{4nq} = N$, there must exist a (witness) vector $\bar{c}_{i_0}$ which is not bad for any pair of monomials $x^{\underline{k}_1}$, $x^{\underline{k}_2}$ (any two monomials $x^{\underline{k}_1}$, $x^{\underline{k}_2}$ yield distinct elements of GF$[q^s]$ after substituting $\omega^{\bar{c}_{i_0}}$. We show now that this implies $f(\omega^{l\bar{c}_{i_0}}) \neq 0$ for some $0 \le l < t$. Suppose the contrary, i.e. $f(\omega^{l\bar{c}_{i_0}}) = 0$ for all l, $0 \le l < t$. Then $XV = 0$, where $X = (f_{\underline{k}})_{\underline{k}}$ and $V = (\omega^{l\bar{c}_{i_0}\underline{k}})$ is the $t \times t$ matrix whose rows are indexed by l, $0 \le l < t$, and columns are indexed by the

$\underline{k}$ that appear as exponents in f.

Note now that

$$\det(V) = \prod_{\underline{k}_1 \neq \underline{k}_2} (\omega^{\sum_j \bar{c}_{i_0 j} k_j^{(1)}} - \omega^{\sum_j \bar{c}_{i_0 j} k_j^{(2)}}) \neq 0$$

(V is a Vandermonde Matrix), which leads to the contradiction, proving correctness of the Algorithm.

Complexity Analysis. We have chosen s to be minimal s satisfying $q^s - 1 > 4nq(n-1)\binom{t}{2}$. We have also chosen $N = \frac{[q^s-1]}{4nq}$, so we get the following upper bound for N, $N < qnt^2$. Furthermore, $|\mathrm{GF}[q^s] \leq Nnq$. The number of querries to the oracle made by the Algorithm for a fixed $\mathrm{GF}[q]$ is $O(nt^3)$.

The construction of the field $\mathrm{GF}[q^s]$ from $\mathrm{GF}[q]$ and s can be done by the Berlekamp Algorithm [Be 70] in $O(N \log^2(Nnq))$ processors and $O(\log^2 N)$ parallel time. Next we find a primitive element ω of $\mathrm{GF}[q^s]$ in the following way.

Factorize $q^s - 1$ into prime factors,

$$q^s - 1 = \prod p_i^{n_i}, \quad p_i \text{ prime.}$$

For any $a \in \mathrm{GF}[q^s]$, compute $a^{\frac{q^s-1}{p_i}}$ for each i (using the binary expansion of the exponent and the techniques from [Lo 83]). An element $a \in \mathrm{GF}[q^s]$ is a primitive element iff all these powers are distinct from 1. This all can be done in the same bounds as needed by the Berlekamp Algorithm.

The remaining part is to compute powers $\omega^{l \cdot \bar{c}_{ij}}$ for $0 \leq l < t$, $1 \leq i \leq N$, and $1 \leq j \leq n$ in $\mathrm{GF}[q^s]$. Using binary representation of the exponents this can be done in $O(Nnt \log^2(Nnq))$ processors and $O(\log^2 N)$ parallel time. Since $N < qnt^2$ for the fixed field $\mathrm{GF}[q]$ we get the upper bound of $O(n^2t^3 \log^2(nt))$ processors and $O(\log^2(nt))$ parallel time. This proves Theorem 4. □

Theorems 3 and 4 can be generalized to work over arbitrary fields of positive characteristic, by applying our method to the *slight* extensions of their primitive subfields of the same characteristic.

3. Some Consequences for Boolean Function

We shall derive some interesting consequences of Theorem 3, and 4 for the case of boolean functions (GF[2]). Although we formulate them for GF[2] only, same results holds for arbitrary 'small' (or fixed) finite fields (in this case instead of boolean circuits we use straight-line programs!).

The boolean RSE-Conversion Problem is the problem of converting a boolean function f (given by the input oracle), and such that RSE(f) $\leq t$ into the equivalent RSE-formula.

A SPARSE$_{\oplus}$–SAT problem is the problem of checking whether f (given as above) has a satisfying assignment.

Theorem 3 entails directly the following.

Corollary 1. *The boolean* RSE-Conversion Problem *is in* NC^3. *The algorithm uses $O(\log^3(nt))$ parallel boolean time and $O(n^2t^6\log^2(nt))$ processors.*

It is interesting to note that SPARSE$_{\oplus}$–SAT problem was not known before to be in P. Theorem 1 says that there is no polynomial time algorithm without using proper field extensions. Corollary 1 puts this problem in P and deterministic boolean NC^3, and Theorem 4 yields even better $O(\log^2 n)$ parallel time bound.

Corollary 2. SPARSE$_{\oplus}$–SAT *is in* NC^2. *The algorithm uses $O(\log^2(nt))$ parallel boolean time and $O(n^2t^3)$ processors.*

4. Further Research

The research on parallel complexity of multivariate polynomial interpolation was spurred by its application towards the parallel matching algorithms (cf. [GK 87]), and resulted already in several applications in problems like sparse factorization and polynomial GCD (cf. [KT 88], [BT 88]). The good bit-complexity algorithms require however computations over finite fields rather than $\mathbb{Z}$. In this connection an important problem arises to improve on the number of processors of the algorithms of Theorem 3, and 4.

It is also another interesting aspect of this work. Our method of Cauchy matrices provides a deterministic 'random-like' method for catching separator values of polynomial terms. Can it be also applied to other problems with known randomized solutions to yield the efficient deterministic solutions?

References

[AL 86] Adleman, L.M., Lenstra, H.K., *Finding Irreducible Polynomials over Finite Fields*, Proc. 18th ACM STOC (1986), pp. 350–355.

[B 81] Ben-Or, M., *Probabilistic Algorithms in Finite Fields*, Proc. 22th IEEE FOCS (1981), pp. 394–398.

[BT 88] Ben-Or, M., Tiwari, P., *A Deterministic Algorithm for Sparse Multivariate Polynomial Interpolation*, Proc. 20th ACM STOC (1988), pp. 301–309.

[Be 70] Berlekamp, E.R., *Factoring Polynomials over Large Finite Fields*, Math. Comp. 24 (1970), pp. 713–753.

[C] Cauchy, A.L., *Exercises d'Analyse et de Phys. Math.*, Vol 2, Paris, Bachelier (1841), pp.151–159.

[CDGK 88] Clausen, M., Dress, A., Grabmeier, J., Karpinski, M., *On Zero-Testing and Interpolation of k-Sparse Multivariate Polynomials over Finite Fields*, Research Report No.8522-CS, University of Bonn (1988); to appear in Theoretical Computer Science (1989).

[Co 85] Cook, S.A., *A Taxonomy of Problems with Fast Parallel Algorithms*, Information and Control 64 (1985), pp. 2–22.

[Ga 83] von zur Gathen, J., *Factoring Sparse Multivariate Polynomials*, Proc. 24th IEEE FOCS (1983), pp. 172–179.

[Ga 84] von zur Gathen, J., *Parallel Algorithm for Algebraic Problems*, SIAM J. Comput., 13 (1984), 808–824.

[GK 87] Grigoriev, D.Y., Karpinski, M., *The Matching Problem for Bipartite Graphs with Polynomially Bounded Permanents is in NC*, Proc. 28th IEEE FOCS (1987), Los Angeles, 1987, pp. 166–172.

[GKS 88] Grigoriev, D.Y., Karpinski, M., Singer, M.F., *Fast Parallel Algorithms for Sparse Multivariate Polynomial Interpolation over Finite Fields*, Research Report No.8523-CS, University of Bonn (1988); submitted to SIAM J. Comput., 1988.

[Ka 85] Kaltofen, E., *Computing with Polynomials Given by Straight-Line Programs I, Greatest Common Divisors*, Proc. 17th ACM STOC (1985), pp. 131–142.

[KT 88] Kaltofen, E., Trager, B., *Computing with Polynomials Given by Black Boxes for their Evaluations: Greatest Common Divisor, Factorization, Separation of Numerators and Denominators*, Proc. 29th IEEE FOCS (1988), pp. 296–305.

[KR 88] Karp, R.M., Remachandran,V., *A Survey of Parallel Algorithms for Shared-Memory Machines*, Research Report No.UCB/CSD 88/407, University of California, Berkeley (1988); to appear in Handbook of Theoretical Computer Science, North Holland (1989).

[LN 83] Lidl, H., Niederreiter, H., *Finite Fields, Encyclopedia of Mathematics and its Applications*, Vol.10, Cambridge University Press (1983).

[Lo 83] Loos, R., *Computing Rational Zeros of Integer Polynomials by p-adic Expansions*, SIAM J. Comput. 12 (1983), pp. 262–192.

[MS 72] MacWilliams, F.J., Sloane, N.J.A., *The Theory of Error Correcting Codes*, North Holland (1972).

[We 87] Wegener, I., *The Complexity of Boolean Functions*, Teubner (1987).