FEIG, EPHRAIM

SOME RESULTS IN ALGEBRAIC COMPLEXITY THEORY

City University of New York PH.D. 1980

SOME RESULTS IN ALGEBRAIC COMPLEXITY THEORY

by

EPHRAIM FEIG

A dissertation submitted to the Graduate Faculty in Mathematics in partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York.

1980

This manuscript has been read and accepted for the Graduate Faculty in Mathematics in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

6/5/80
date

Chairman of Examining Committee

6/5/80
date

Executive Officer

Professor Burton Randol

Dr. Shmuel Winograd, IBM

Supervisory Committee

The City University of New York

Dedicated to Linda Maher with much love

Acknowledgement

I would like to thank Prof. L. Auslander for so many reasons which made the preparation of this thesis an exciting and joyous experience.

I would also like to thank Dr. S. Winograd for his kind help and encouragement.

Table of Contents

I. Introduction

This paper is an account of some of my work in algebraic complexity theory during the past year and a half. The work was motivated by recent results on the structure and multiplicative complexity of the Discrete Fourier Transform (DFT). The basic idea behind these results is to block-diagonalize the matrix of the DFT using left and right multiplications by rational matrices. Block-diagonalizing the matrix of $DFT(Z_p)$, the 1-dimensional DFT on p points, where p is a prime, is already discussed in [11]. Winograd looks at the "core" of the $DFT(Z_p)$ matrix, that is, the $DFT(Z_p)$ matrix without its first row and column (whose entries are all 1). After permuting the rows and columns of the "core," the resulting matrix is circulant [7]. This matrix can therefore be viewed as the image of some element in a polynomial quotient ring under the regular representation, and can be block-diagonalized using the Chinese Remainder Theorem (CRT). This will be quickly reviewed in section II-1.

In the spring of last year, Prof. Auslander

suggested to me that I study $DFT(Z_p \oplus \cdots \oplus Z_p)$, the k-dimensional DFT on p^k points, where p is a prime, by investigating the group of additive characters on the field with p^k elements. In section II-5 we show that the core of the DFT matrix in this case can be block-diagonalized by permutation matrices to a direct sum of $p^k-1/p-1$ copies of the core of the $DFT(Z_p)$ matrix. This yields a method for computing the $DFT(Z_p \oplus \cdots \oplus Z_p)$ by performing $p^k-1/p-1$ different $DFT(Z_p)$ computations. When this was discovered, it was firmly believed that this scheme yielded minimal algorithms for the $DFT(Z_p \oplus \cdots \oplus Z_p)$ as direct sums of minimal algorithms for $DFT(Z_p)$. In [11] Winograd constructs minimal algorithms for $DFT(Z_p)$ as direct sums of minimal algorithms for products in finite algebraic extension fields.

The investigation branched into two directions, the complexity issue and the classification problem for minimal algorithms. In [11] Winograd has already proved that the multiplicative complexity of $DFT(Z_p)$ is $2p-\psi(p-1)-3$, where $\psi(p-1)$ equals the number of divisors of p-1. In section III-2 we prove that every bilinear algorithm for computing the N products $ab_1, \ldots, ab_N$ in any quadratic

extension field uses at least $3N$ multiplications. This yields methods for determining the multiplicative complexity of $DFT(Z_5 \oplus \ldots \oplus Z_5)$ and $DFT(Z_7 \oplus \ldots \oplus Z_7)$ for the class of bilinear algorithms. A recent result of Auslander and Winograd [3] enables us to compute the multiplicative complexity of $DFT(G)$ for most finite abelian groups G. Although the results in III-2 are dwarfed by the cited result, we present it because it is the proof which we find interesting. It establishes a non-trivial lower bound on the multiplicative complexity of a system of bilinear forms without using either a substitution or a dimension argument.

The major part of this paper deals with the classification problem. In [10] Winograd classified all minimal bilinear algorithms for computing products in finite algebraic extension fields. He showed that they all essentially use Toom's minimal algorithm for computing the product of polynomials [8], and these are known to be impractical for polynomials of high degrees [13]. The main result of chapter IV is that every minimal-division-free algorithm for computing products in a finite algebraic extension field is bilinear. Our main tool is the method of substitutions (see also [3], [5], [9], [12]). We study how various systems of bilinear forms and

minimal algorithms wich compute them are transformed by various projection operators. We also rely heavily on Winograd's structure theorem [12] which asserts that every minimal algorithm for computing a system of bilinear forms induces a minimal quadratic algorithm for computing this system.

Our results in chapter IV apply to other systems of bilinear forms, including systems for computing products in various finite dimensional division rings over fields. As a consequence, we prove that the multiplicative complexity of the quaternion product over a real field is 8.

II. Our motivating example

II-1. Introduction

We start by describing the Discrete Fourier Transform on a finite abelian group G, which we shall denote by $DFT(G)$. Let $\hat{G}$ be the dual group of G, that is, the group of homomorphisms of G into $\mathbb{C}$, the complex numbers. G may be written uniquely in the form $G = \mathbb{Z}/n_1\mathbb{Z} \oplus \cdots \oplus \mathbb{Z}/n_k\mathbb{Z}$, the direct sum of k cyclic groups of order n_i, where $n_i \mid n_{i+1}$ for all $i = 1, \ldots, k-1$. Let $\epsilon_j = e^{(2\pi i/n_j)}$. The characters in $\hat{G}$ map G onto the multiplicative subgroup of $\mathbb{C}$ generated by ϵ_k. There exists an isomorphism $\lambda : G \longrightarrow \hat{G}$ given by

$$\langle \lambda(a_1, \ldots, a_k), (x_1, \ldots, x_k) \rangle = \prod_{j=1}^{k} (\epsilon_j)^{a_j x_j}.$$

We will write $\hat{g} = \lambda(g)$, for every $g \in G$.

Let $L^2(G)$ be the vector space of functions from G to $\mathbb{C}$, and define $L^2(\hat{G})$ similarly. If the order of the group $|G| = n$, then $L^2(G) \cong L^2(\hat{G}) \cong \mathbb{C}^n$. $DFT(G)$ is the invertible linear operator

$$\mathcal{F} : L^2(G) \longrightarrow L^2(\hat{G})$$

defined by

$$\mathcal{F} f(\hat{g}) = \sum_{h \in G} \langle \hat{g}, h \rangle f(h)$$

We also write $\hat{f} = \mathcal{F} f$. We may identify G with $\hat{G}$ via our isomorphism λ and we will write $\hat{f}(g) = \mathcal{F} f(\hat{g})$.

If G is cyclic of order n, we may write $G = \{0, 1, \ldots, n-1\}$ with group operation addition modulo n. Let $\epsilon = e^{2\pi i/n}$. Then

$$f(g) = \sum_{h=0}^{n-1} \epsilon^{gh} f(h) .$$

Define the vectors $(\underline{f}) = (f(0) \quad f(1) \quad \cdots \quad f(n-1))^t$ and $(\hat{\underline{f}}) = (\hat{f}(0) \quad \hat{f}(1) \quad \cdots \quad \hat{f}(n-1))^t$. Let $(F) = (\epsilon^{gh})$ be the $n \times n$ matrix whose $(g+1, h+1)$ entry is ϵ^{gh}, for $0 \leq g,h \leq n-1$. Then we have $(\hat{\underline{f}}) = (F)(\underline{f})$; this is the way we usually encounter the Discrete Fourier Transform.

In the general case where $G = \mathbb{Z}/n_1\mathbb{Z} \oplus \cdots \oplus \mathbb{Z}/n_k\mathbb{Z}$ we may write every $g \in G$ in the form $g = (a_1, \ldots a_k)$ with $0 \leq a_j \leq n_j - 1$. For $j = 1, \ldots, k$ let $p_j : G \longrightarrow \mathbb{Z}/n_j\mathbb{Z}$ be the natural projection maps. Then

$$f(g) = \sum_{h \in G} \prod_{j=1}^{k} \epsilon_j^{p_j(g)p_j(h)} f(h).$$

Let us choose the usual lexicographic ordering for the elements in both $L^2(G)$ and $L^2(\hat{G})$; for example

for $f \in L^2(G)$ we will have $(\underline{f}) = (f(0,\ldots,0)$ $f(0,\ldots,1)$ $\cdots\cdots$ $f(n_1-1, \ldots, n_k-1))^t$. Let (F) be the matrix of $DFT(G)$ relative to this ordering, and let (F_j) for $j = 1,\ldots,k$ be the matrices corresponding to $DFT(\mathbb{Z}/n_j\mathbb{Z})$ also relative to the lexicographic order, then

$$(F) = (F_1) \otimes \cdots \otimes (F_k) \quad .$$

II-2. DFT($\mathbb{Z}_p$), p prime

Let $G = \mathbb{Z}_p$ with p prime. We may identify G with the field with p elements; that is, we fix an isomorphism of G into the additive group of the field with p elements. The following lemma describes the additive characters of a finite field. We prove it for arbitrary finite fields because we will need this generality in the multi-dimensional case.

Lemma II-1: Let G be a field with p^k elements, and let $\hat{G}$ be the dual to the additive group. Let $w \in G$ be a generator of the cyclic multiplicative group of order p^k-1, let $\lambda \in \hat{G}$ be a non-trivial character, and let $1 \in \hat{G}$ be the trivial character. For $n = 0, 1, \ldots, p^k-2$, define $\lambda_n : G \longrightarrow \mathbb{C}$ by setting $\langle \lambda_n, w^j \rangle = \langle \lambda, w^{j+n} \rangle$ and $\langle \lambda_n, 0 \rangle = 1$.

Then $G = \{\lambda_n\}_{n = 0,\ldots,p^k-2} \cup 1$.

Proof: To see that each λ_n defines a character, we note that $\langle \lambda_n, w^{a+w^b} \rangle = \langle \lambda, w^{a+n+w^{b+n}} \rangle = \langle \lambda, w^{a+n} \rangle \langle \lambda, w^{b+n} \rangle = \langle \lambda_n, w^a \rangle \langle \lambda_n, w^b \rangle$, and by hypotheses, $\langle \lambda_n, 0 \rangle = 1$.

To see that these are all distinct characters, we suppose the contrary, that for some integers $m \neq n \mod p^k-1$, we have $\lambda_m = \lambda_n$. Then for any integer s, $\langle \lambda_m, w^s \rangle = \langle \lambda_n, w^s \rangle$. This implies that $\langle \lambda_s, w^m \rangle = \langle \lambda_s, w^n \rangle$, and so for any s, $\langle \lambda_s, w^m - w^n \rangle = \langle \lambda_s, w^m \rangle \langle \lambda_s, w^n \rangle^{-1} = 1$. This implies that $w^m = w^n$, and we get a contradiction.

We now return to our discussion of $DFT(Z_p)$. The above lemma points out a certain cyclic property of the non-trivial characters, and this property is a consequence of the multiplicative structure in the field Z_p. Let us arrange the entries in our input vector $\underline{f}$ and our output vector $\hat{\underline{f}}$ so as to highlight this structure. First put $N = p-2$, and let $(\tilde{\underline{f}}) = (f(0) \quad f(1) \quad f(w) \quad f(w^2) \quad \cdots \quad f(w^N))^t$. Let $B: G \longrightarrow \hat{G}$ be the bijection given by $B(w^j) = \lambda_j$, $j = 0, \ldots, N$, and $B(0) = 1$, where λ_j and 1 are as defined in the lemma above. Let $(\tilde{\hat{\underline{f}}}) = (\hat{f}(B(0)) \quad \hat{f}(B(1)) \quad \hat{f}(B(w)) \quad \hat{f}(B(w^2)) \quad \cdots \quad \hat{f}(B(w^N)))^t$. Our lemma says that $(\tilde{\hat{\underline{f}}}) = (\tilde{C})(\tilde{\underline{f}})$, where the matrix

(C) is

$$\begin{pmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \langle\lambda,1\rangle & \langle\lambda,w\rangle & \langle\lambda,w^2\rangle & & \langle\lambda,w^N\rangle \\ 1 & \langle\lambda_1,1\rangle & \langle\lambda_1,w\rangle & \langle\lambda_1,w^2\rangle & & \langle\lambda_1,w^N\rangle \\ 1 & \langle\lambda_2,1\rangle & \langle\lambda_2,w\rangle & \langle\lambda_2,w^2\rangle & & \langle\lambda_2,w^N\rangle \\ \vdots & & & & & \vdots \\ 1 & \langle\lambda_N,1\rangle & \langle\lambda_N,w\rangle & \langle\lambda_N,w^2\rangle & \cdots & \langle\lambda_N,w^N\rangle \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \langle\lambda,1\rangle & \langle\lambda,w\rangle & \langle\lambda,w^2\rangle & & \langle\lambda,w^N\rangle \\ 1 & \langle\lambda,w\rangle & \langle\lambda,w^2\rangle & \langle\lambda,w^3\rangle & & \langle\lambda,1\rangle \\ 1 & \langle\lambda,w^2\rangle & \langle\lambda,w^3\rangle & \langle\lambda,w^4\rangle & & \langle\lambda,w\rangle \\ \vdots & & & & & \vdots \\ 1 & \langle\lambda,w^N\rangle & \langle\lambda,1\rangle & \langle\lambda,w\rangle & \cdots & \langle\lambda,w^{N-1}\rangle \end{pmatrix}$$

We will ultimately be concerned with constructing algorithms for $DFT(\mathbb{Z}_p)$. We may restrict our considerations to what we call the "core" of the matrix $(\tilde{C})$. This is the $(p-1)\times(p-1)$ minor of $(\tilde{C})$ obtained from it by dropping the first row and the first column. We will label this core $(\hat{C})$.

We recognize that $(\hat{C})$ is a circulant matrix. Let A be the companion matrix of $u^{p-1}-1$. Also, let (C) be the matrix obtained from $(\hat{C})$ by leaving the first column fixed and reversing the order of the last N columns. A direct calculation shows

that $$(C) = \sum_{j=0}^{N} \langle \lambda, w^j \rangle A^j \qquad (1)$$

Let $(\underline{f}) = (f(1) \quad f(w^N) \quad f(w^{N-1}) \quad \cdots \quad f(w))^t$ and $(\hat{\underline{f}}) = (\hat{f}(F(1)) \quad \hat{f}(F(w^N)) \quad \hat{f}(F(w^{N-1})) \cdots \hat{f}(F(w)))^t$. We view the entries in $(\underline{f})$ as distinct indeterminants over G (this is because we want to use our algorithms on arbitrary input vectors). Let H be the field extension of G obtained by adjoining to it ϵ and the indeterminants in $(\underline{f})$. We claim that we can view the action $(C)(\underline{f})$ as a product in $H[u] / \langle u^{p-1}-1 \rangle$ This is essentially the Theorem of the Rational Canonical Form, and it enables us to block-diagonalize (C) using the Chinese Remainder Theorem. We now digress momentarily to discuss these two theorems in our setting.

II-3. The Rational Canonical Form and the Chinese Remainder Theorems

Let F be a field, u an indeterminant over F,

and $g(u) \in F[u]$ a monic polynomial. We can write $g(u) = \prod_{i=1}^{M} g_i(u)$, where $g_i(u)$ are all relatively prime. The Chinese Remainder Theorem (CRT) asserts that

$$F[u] / \langle g(u) \rangle \cong \bigoplus \sum_{i=1}^{M} F[u] / \langle g_i(u) \rangle .$$

Let A be the companion matrix of $g(u)$; that is, if $g(u) = a_0 + a_1 u + \cdots + u^n$, then

$$A = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & -a_0 \\ 1 & 0 & 0 & & 0 & -a_1 \\ 0 & 1 & 0 & & 0 & -a_2 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & -a_{n-1} \end{pmatrix}$$

Let A_i be the companion matrices of $g_i(u)$ for $i = 1,\ldots,M$. For any F-matrix D, let $\mathcal{A}(D)$ be the algebra generated by D over F. We know that if A_h is the companion matrix of any monic polynomial $h(u)$, then the minimum polynomial of A_h equals the characteristic polynomial of A_h and is in fact $h(u)$. Therefore there exist isomorphisms $I: F[u] / \langle g(u) \rangle \longrightarrow \mathcal{A}(A)$ given by $I(u) = A$ and $I_i: F[u] / \langle g_i(u) \rangle \longrightarrow \mathcal{A}(A_i)$ given

by $I_i(u) = A_i$.

The CRT asserts that there exists an F-matrix N such that

$$NAN^{-1} = \begin{pmatrix} A_1 & & 0 \\ & A_2 & \\ & & \ddots \\ 0 & & A_M \end{pmatrix}$$

Let H be a field extension of F, and for any F-matrix D, let $\mathcal{A}_H(D)$ be the algebra genereated by D over H. Then

$$\mathcal{A}_H(D) \cong \mathcal{A}(D) \otimes_F H .$$

Let J be the isomorphism given by the CRT. Then J extends to an isomorphism

$$J \otimes 1 : H[u] / \langle g(u) \rangle \quad \oplus\sum \quad H[u] / \langle g_i(u) \rangle .$$

In the language of matrices, if $K = \sum_{i=0}^{n-1} f_i A^i$, $f_i \in H$, then

$$NKN^{-1} = \begin{pmatrix} K_1 & & 0 \\ & K_2 & \\ & & \ddots \\ 0 & & K_M \end{pmatrix}$$

and K_i can be viewed as elements in $\mathcal{A}_H(A_i)$.

In the language of complexity theory, if

$$r(u) = \sum_{i=0}^{n-1} r_i u^i \quad \text{and} \quad s(u) = \sum_{i=0}^{n-1} s_i u^i \ ,$$

$r_i, s_i \in H$, then computing the coefficients of $r(u)s(u) \mod g(u)$ is equivalent over F to computing the coefficients of $r(u)s(u) \mod g_i(u)$ for all $i = 1, \ldots, M$.

II-4. DFT(Z_p), continued

We have just seen that the action $(C)(\underline{f})$ corresponds to a multiplication in the polynomial quotient ring $H[u] / \langle u^{p-1}-1 \rangle$. To be precise, if we let

$$X(u) = \sum_{i=0}^{p-2} \langle \lambda, w^i \rangle u^i \quad \text{and} \quad Y(u) = \sum_{i=0}^{p-2} y_i u^i$$

where $y_o = f(1)$ and for $i=1, \ldots, p-2$, $y_i = f(w^{p-1-i})$, then the action $(C)(\underline{f})$ corresponds to the multiplication $X(u)Y(u) \mod u^{p-1}-1$.

We know that $u^{p-1}-1 = \prod_{i=1}^{\psi(p-1)} \varphi_i(u)$, where $\varphi_i(u)$ are distinct irreducible factors of $u^{p-1}-1$ (the cyclotomic polynomials) and $\psi(p-1)$ = the number of divisors of $p-1$. Let $\varphi_1(u) = u-1$.

Let $p_i : H[u] / \langle u^{p-1}-1 \rangle \longrightarrow H[u] / \langle \varphi_i(u) \rangle$ be the natural projections. Then

$$p_i X(u) = \sum_{i=0}^{p-2} \langle \lambda, w^i \rangle = -1$$

because for $i = 0, \ldots, p-2$, $\langle \lambda, w^i \rangle$ are all distinct p^{th} roots of unity not equal to 1. For $p \geq 3$, let $R = p-1/2$, $S(u) = u^R-1$ and $T(u) = u^R+1$. Then we have the following commuting diagram of isomorphisms

$$\begin{array}{ccc} H[u] / \langle u^{p-1}-1 \rangle & \longrightarrow & \oplus \sum_{i=1}^{(p-1)} H[u] / \langle \varphi_i(u) \rangle \\ & \searrow \quad \nearrow & \\ & H[u] / \langle S(u) \rangle \oplus H[u] / \langle T(u) \rangle & \end{array}$$

Because $w^R = -1$, we have $w^{i+R} + w^i = w^i(w^R + 1) = 0$. This implies that $\langle \lambda, w^i \rangle \langle \lambda, w^{i+R} \rangle = 1$ and therefore $\langle \lambda, w^i \rangle$ and $\langle \lambda, w^{i+R} \rangle$ are complex conjugate. Let p_S and p_T be the natural projections of $H[u] / \langle u^{p-1}-1 \rangle$ onto $H[u] / \langle S(u) \rangle$ and $H[u] / \langle T(u) \rangle$, respectively. Then we see that the coefficients of $p_S(X(u))$ are all real and the coefficients of $p_T(X(u))$ are all pure imaginary. Our commuting diagram shows that the projections of $X(u)$ onto each $H[u] / \langle \varphi_i(u) \rangle$ are polynomials whose coefficients are either all real or all pure imaginary.

We have shown that one can compute $DFT(Z_p)$ by first computing the coefficients of the products $X(u)Y(u) \mod \varphi_i(u)$, $i = 1, \ldots, (p-1)$. Then, rational linear combinations of these coefficients together with some rational linear combinations of the entries in our input vector $(\underline{f})$ will give us the entries of the DFT output, $(\hat{\underline{f}})$. We saw that computing the product $X(u)Y(u) \mod \varphi_1(u)$ requires no essential multiplications. For $i = 2, \ldots, \psi(p-1)$, we can compute the products $\mod \varphi_i(u)$ using the scheme described in [11]. This scheme is based on minimal algorithms for computing arbitrary products in finite algebraic extension fields [13]. These in turn all use some variation of Toom's algorithm for computing the product of polynomials [8], [10]. In [11] Winograd proves that the algorithms derived using this scheme are minimal with respect to non-rational (essential) multiplications. Now, each product $\mod \varphi_i(u)$, $i = 2, \ldots, \psi(p-1)$ can be computed with $2d_i - 1$ essential multiplications, where d_i = degree of $\varphi_i(u)$. We can therefore compute $DFT(Z_p)$ with

$$\sum_{i=2}^{\psi(p-1)} (2d_i - 1) = 2(p-2) - (\psi(p-1) - 1)$$

$$= 2p - \psi(p-1) - 3$$

essential multiplications. Let us denote by $\mu DFT(G)$

the multiplicative complexity of $DFT(G)$ over the rationals. We may summarize our discussion in the last three sections as follows:

Theorem II-1: $\mu\, DFT(Z_p) = 2p - \psi(p-1) - 3$, where $\psi(p-1)$ = the number of divisors of $p-1$. Furthermore, there exists minimal algorithms for $DFT(Z_p)$ all of whose essential multiplications have factors which are either real or pure imaginary.

II-5: $DFT(Z_p \oplus \cdots \oplus Z_p)$, p prime

Let $G = Z_p \oplus \cdots \oplus Z_p$, the direct sum of k copies of the cyclic group of order p, where p is prime. Our goal in this section is to reduce the computation of $DFT(G)$ to what we call a direct sum computation of $p^k-1/p-1$ copies of $DFT(Z_p)$. We identify G with the field with p^k elements. Lemma II-1 described the additive character of this field, and $DFT(G)$ is built out of these. Using the notation of lemma II-1, for any $f \in L^2(G)$ we

have $\hat{f}(0) = f(0) + \sum_{i=0}^{N} f(w^i)$, where $N = p^k-2$,

$$\hat{f}(w^j) = f(0) + \sum_{i=0}^{N} \langle \lambda_j, w^i \rangle f(w^i), \quad j = 1, \ldots, p^k-1.$$

Let $(\underline{\hat{f}}) = (f(0)\ f(1)\ f(w)\ f(w^2) \cdots f(w^N))^t$, and let

$b: G \longrightarrow \hat{G}$ be the bijection given by $b(w^j) = \lambda_j$ for $j = 0, 1, \ldots, N$ and $b(0) = 1$, where λ_j and 1 are as defined in the previous lemma. Observe that this bijection is not a group homomorphism. Let us order the entries in the output vector as follows: $(\underline{\hat{f}}) = (\hat{f}(0) \quad \hat{f}(1) \quad \hat{f}(\lambda_1) \quad \hat{f}(\lambda_2) \quad \ldots\ldots \quad \hat{f}(\lambda_N))^t$. Relative to this choice of bases, the matrix of DFT(G) has the form

$$\begin{pmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \langle\lambda,1\rangle & \langle\lambda,w\rangle & \langle\lambda,w^2\rangle & & \langle\lambda,w^N\rangle \\ 1 & \langle\lambda_1,1\rangle & \langle\lambda_1,w\rangle & \langle\lambda_1,w^2\rangle & & \langle\lambda_1,w^N\rangle \\ 1 & \langle\lambda_2,1\rangle & \langle\lambda_2,w\rangle & \langle\lambda_2,w^2\rangle & & \langle\lambda_2,w^N\rangle \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 1 & \langle\lambda_N,1\rangle & \langle\lambda_N,w\rangle & \langle\lambda_N,w^2\rangle & \cdots & \langle\lambda_N,w^N\rangle \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \langle\lambda,1\rangle & \langle\lambda,w\rangle & \langle\lambda,w^2\rangle & & \langle\lambda,w^N\rangle \\ 1 & \langle\lambda,w\rangle & \langle\lambda,w^2\rangle & \langle\lambda,w^3\rangle & & \langle\lambda,1\rangle \\ 1 & \langle\lambda,w^2\rangle & \langle\lambda,w^3\rangle & \langle\lambda,w^4\rangle & & \langle\lambda,w\rangle \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 1 & \langle\lambda,w^N\rangle & \langle\lambda,1\rangle & \langle\lambda,w\rangle & \cdots & \langle\lambda,w^{N-1}\rangle \end{pmatrix}$$

Let C be the core of the above matrix, that is, the matrix derived from the above one by dropping

the first row and the first column, and for $j = 0, 1, \ldots, N$ let $\tilde{f}(\lambda_j) = \hat{f}(\lambda_j) - \hat{f}(1)$. Also let $(\underline{\tilde{f}}) = (\tilde{f}(1) \ \tilde{f}(\lambda) \ \tilde{f}(\lambda_1) \ \ldots \ \tilde{f}(\lambda_N))^t$, and let $(\underline{f})$ be the vector derived from $(\underline{\tilde{f}})$ by dropping the top entry $f(0)$. Then $(\underline{\tilde{f}}) = C\,(\underline{f})$. We recognize this system as the one corresponding to a well known cyclic convolution. More specifically, $\tilde{f}(\lambda_j)$ is the coefficient of the u^j term in the expression for

$$\left(\sum_{j=0}^{N} \langle \lambda, w^j \rangle u^j \right) \left(f(1) + \sum_{j=1}^{N} f(w^{N-j+1})\, u^j \right)$$

mod $u^{N+1}-1$. To simplify our notation, we let $y_0 = f(1)$ and for $j = 1,\ldots,N$ we let $y_j = f(w^{N-j+1})$. The above cyclic convolution now has the form

$$\left(\sum_{j=0}^{N} \langle \lambda, w^j \rangle u^j \right) \left(\sum_{j=0}^{N} y_j u^j \right) \quad \text{mod} \quad u^{N+1}-1 .$$

To further simplify our notation, we will let $X(u) = \sum_{j=0}^{N} \langle \lambda, w^j \rangle u^j$ and $Y(u) = \sum_{j=0}^{N} y_j u^j$.

Let $a = p^k - 1 = N+1$ and $b = p^k - 1/p - 1$. For $i = 1, \ldots, p-1$, let ϵ_i be the set of primitive p-th roots of unity. Define for $i = 1,\ldots,p-1$ $B_i = \{ j \mid \langle \lambda, w^j \rangle = \epsilon_i, \ 0 \ \ j \ \ a-1 \}$ and define $B_0 = \{ j \mid \langle \lambda, w^j \rangle = 1, \ 0 \ \ j \ \ a-1 \}$.

In G is a subfield H which is isomorphic to

to Z/pZ. The elements in H are $\{0, w^b, w^{2b}, \ldots, w^{(p-1)b=1}\}$, and G is an algebraic extension of H of degree k. For every $w^j \in G$ and for $n = 0, \ldots, p-2$, $w^j w^{nb}$ equals the sum of w^j $\eta(n)$ times, and $\eta : \{0, \ldots, p-2\} \longrightarrow \{1, \ldots, p-1\}$ is a bijection. We have

$$(1) \qquad \langle \lambda, w^{j+nb} \rangle = \langle \lambda, w^j \rangle \eta(n) .$$

Choose ϵ_i and an integer r, $0 \leq r \leq p-2$, such that $\langle \lambda, w^r \rangle = \epsilon_i$. Then $\langle \lambda, w^r \rangle \eta(n)$ runs over all primitive p-th roots of unity as n runs over $0, \ldots, p-2$. Let us define for $s = 0, \ldots, p-1$ $P_s(u) = \sum_{j \in E_s} u^j$. **Equation** (1) implies that $P_s(u) = u^{(s)b} P_i(u) \mod u^a-1$. For ease of notation, let us write $\epsilon_i = \epsilon$ and $P_i(u) = P(u)$. We may assume that we chose our non-trivial character so that $\langle \lambda, w \rangle = \epsilon$. Also let $S(u) = \epsilon + \epsilon^{\eta(1)} u^b + \epsilon^{\eta(2)} u^{2b} + \ldots + \epsilon^{\eta(p-2)} u^{(p-2)b}$ Then $X(u) = S(u)P(u) + P_0(u) \mod u^a-1$. Our discussion shows that computing DFT(G) is equivalent to computing the coefficients of

$$(2) \qquad (S(u)P(u) + P_0(u))(Y(u)) \mod u^a-1.$$

The coefficients of $P(u)$ and $P_0(u)$ are all 1 or 0; the coefficients of $Y(u)$ may be viewed as distinct indeterminants over the rationals.

Let us define v_i for $i = 0, \ldots, a-1$ by the formula $\sum_{i=0}^{N} v_i u^i = F(u)Y(u) \mod u^a-1$. Computing the v_i requires no essential multiplications. Let $V(u) = \sum_{i=0}^{N} v_i u^i$. We see that we can compute DFT(G) by first computing the coefficients of $S(u)V(u) \mod u^a-1$. Define for $j = 0, \ldots, b-1$, the polynomials

$$V_j(u) = \sum_{i=0}^{p-2} v_{j+ib} u^{j+ib} .$$

Then

$$V(u) = \sum_{j=0}^{b-1} V_j(u)$$

and

$$S(u)V(u) = \sum_{i=0}^{b-1} S(u)V_j(u) \mod u^a-1.$$

Furthermore, if for any integer r the coefficient of u^r in $S(u)V_j(u)$ is not 0, then for all $i \neq j$, the coefficient of u^r in $S(u)V_i(u)$ is 0. We see that one way of computing DFT(G) is by first performing b computations for $S(u)V_j(u) \mod u^a-1$ and then completing the algorithm without any more essential multiplications. The change of variables $v \longrightarrow u^b$ gives that the DFT(G)

can be computed by first computing the coefficients of the b products $\overline{S}(v)\overline{V}_j(v)$ mod $v^{p-1}-1$, where $\overline{S}(v) = \epsilon + \epsilon^{\eta(1)}v + \epsilon^{\eta(2)}v^2 + \ldots + \epsilon^{\eta(p-2)}v^{p-2}$ and $\overline{V}_j(v) = \sum_{i=0}^{p-2} v_{j+ib}v^i$. We recognize these as the computations in Winograd's algorithm for $DFT(Z_p)$.

Let us restate our results in the language of matrices. First, for any matrix A and any integer m define the matrix

$$mA = \begin{pmatrix} A & & 0 \\ & A & \\ 0 & & \ddots \\ & & & A \end{pmatrix}$$

the m-fold direct sum. We have shown that there exist permutation matrices P_1 and P_2 such that $P_1(\underline{F})P_2 = bC$, where C is a circulant $(p-1)\times(p-1)$ matrix. C in fact is the matrix corresponding to $DFT(Z_p)$. Now by the CRT, there exist rational matrices R_1 and R_2 such that

$$R_1CR_2 = \begin{pmatrix} A_1 & & 0 \\ & A_2 & \\ 0 & & \ddots \\ & & & A_{\psi(p-1)} \end{pmatrix}$$

where $A_i = bC_i$, and the C_i may be viewed as the images of elements in the algebraic extension field $F[u]/\langle\varphi_i(u)\rangle$ under the regular representation.

We can summarize by saying that there exist rational matrices R_l and R_r such that

$$R_l(\underline{F})R_r = \begin{pmatrix} bA_1 & & & 0 \\ & bA_2 & & \\ & & \ddots & \\ 0 & & & bA_{\Psi(p-1)} \end{pmatrix}$$

where $(\underline{F})$ is the core of the DFT(G) matrix. We claim that R_l and R_r are non-singular. To prove our assertion it suffices to show that $P(u)$ and u^a-1 are relatively prime. The proof of this fact presented here is due to A. Vasquez.

Lemma II-2: $P(u)$ and u^a-1 are relatively prime.

Proof: Suppose the contrary. Then there is some a-th root of unity α such that $P(\alpha) = 0$. Now G^x, the multiplicative group of invertible elements in G, is cyclic. Let w generate G^x; then as χ ranges over all multiplicative characters of G^x, $\chi(w)$ ranges over all a-th roots of unity. Therefore our assumption implies that there exists a multiplicative character χ such that $P(\chi(w)) = P(\alpha) = 0$.

We will prove the lemma by studying the Gauss sum associated with χ and the non-trivial additive character λ which we used to define $P(u)$. The classical results which we will cite can be found in Lang's Cyclotomic Fields.

The Gauss sum is defined as

$$\tau_\lambda(\chi) = \sum_{j=0}^{a} \chi(w^j)\lambda(w^j) .$$

We break up this sum as we did in our construction above. We have $\tau_\lambda(\chi)$

$$= \sum_{j \in P_0} \chi(w^j) + \sum_{i=0}^{p-2} \left(\sum_{j \in P_i} \chi(w^j)\lambda(w^j) \right)$$

$$= \sum_{j \in P_0} \chi(w^j) + \sum_{j \in P_i} \left(\sum_{i=0}^{p-2} \chi(w^{j+ib})\lambda(w^{j+ib}) \right)$$

$$= \sum_{j \in P_0} \chi(w^j) + \sum_{j \in P_i} \left(\sum_{i=0}^{p-2} \chi(w^j)\chi(w^{ib})\lambda(w^j)\eta(i) \right)$$

$$= \sum_{j \in P_0} \chi(w^j) + \left(\sum_{j \in P_i} \chi(w)^j \right) \left(\sum_{i=0}^{p-2} \chi(w^{ib})\lambda(w^j)\eta(i) \right)$$

$$= \sum_{j \in P_0} \chi(w^j) + P(\chi(w)) \left(\sum_{i=0}^{p-2} \chi(w^{ib})\lambda(w^j)\eta(i) \right)$$

$$= \sum_{j \in P_0} \chi(w^j).$$

Now χ cannot be the trivial character, since otherwise $P(\chi(w)) = P(1) = \sum_{j \in P_i} (1)^j \neq 0$. If χ is not trivial, then we know that $\tau_\lambda(\chi) \neq 0$.

However, $\sum_{j=0}^{a-1} \chi(w^j) = 0$, and we can manipulate this sum, just as we did above, to get

$$\sum_{j \in P_0} \chi(w^j) = \sum_{j=0}^{a-1} \chi(w^j) = 0.$$

This implies the contradiction $\tau_\lambda(X) = 0$, and so we have proved the lemma.

The lemma implies that the two systems DFT(G) and the direct sum system we have constructed are computationally equivalent.

Remark: Winograd has communicated to us an elementary argument which proves the lemma.

II-6: DFT($\mathbb{Z}_5 \oplus \mathbb{Z}_5$)

We will construct a minimal algorithm for DFT($\mathbb{Z}_5 \oplus \mathbb{Z}_5$). Given the 25 distinct indeterminants $f(a,b)$ with $0 \leq a,b \leq 4$, we would like to compute the 25 values

$$\hat{f}(a,b) = \sum_{(x,y)} \epsilon^{ax+by} f(x,y)$$

where $\epsilon = e^{2\pi i/5}$. The polynomial u^2+2 is irreducible over $\mathbb{Z}_5$, so we can look at the field $H = \mathbb{Z}_5[u] / \langle u^2+2 \rangle$. H is a vector space of dimension 2 over $\mathbb{Z}_5$ with basis $\{1, u\}$. We will write $a+bu = (a,b)$ for all $a+bu \in H$. $\omega = (1,1)$ generates a cyclic multiplicative group of order 24. We have a bijection $b: H \longrightarrow H$ given by $b(0) = 0$ and the

the following table:

$j \longrightarrow w^j$		$j \longrightarrow w^j$		$j \longrightarrow w^j$		$j \longrightarrow w^j$	
0	(1,0)	6	(3,0)	12	(4,0)	18	(2,0)
1	(1,1)	7	(3,3)	13	(4,4)	19	(2,2)
2	(4,2)	8	(2,1)	14	(1,3)	20	(3,4)
3	(0,1)	9	(0,3)	15	(0,4)	21	(0,2)
4	(3,1)	10	(4,3)	16	(2,4)	22	(1,2)
5	(1,4)	11	(3,2)	17	(4,1)	23	(2,3)

Accordingly, we relabel our input data as follows: $f(0,0) = y_{-1}$, $f(1,0) = y_0$, and for $j = 1, \ldots, 23$, $f(w^{24-j}) = y_j$.

Let us choose the non-trivial character λ defined by $\langle \lambda, (a,b) \rangle = \epsilon^a$. Define

$$X(u) = \sum_{j=0}^{23} \langle \lambda, w^j \rangle u^j \quad ,$$

$$P(u) = 1 + u + u^5 + u^{14} + u^{22} \quad ,$$

$$S(u) = \alpha_1 + \alpha_2 u^6 + \alpha_4 u^{12} + \alpha_3 u^{18} \quad ,$$

where $\alpha_i = \epsilon^i - 1$,

$$Y(u) = \sum_{j=0}^{23} y_j u^j \quad .$$

Then $X(u) = S(u)P(u) \mod u^{24}-1$. (Notice the modification from the construction in the previous section. There we were concerned with the complexity question and we did not worry about computing $P_0(u)Y(u) \mod u^a-1$, which requires no essential steps. Here we are trying to minimize additions as well. This modification holds for all $DFT(Z_p \oplus \ldots \ldots \oplus Z_p)$.)

We would like to compute the coefficients of $X(u)Y(u) = S(u)P(u)Y(u) \mod u^{24}-1$. We can first compute $P(u)Y(u) \mod u^{24}-1$ with only additions.

Define $V(u) = \sum_{i=0}^{23} v_i u^i = P(u)Y(u) \mod u^{24}-1$,

then $v_i = y_i + y_{i+1} + y_{i+5} + y_{i+14} + y_{i+22}$, where the indexing numbers are taken mod 24. It is clear how to compute each v_i with 4 additions.

Next, to compute the coefficients of $S(u)V(u) \mod u^{24}-1$, we perfom the following 6 computations: for $m = 0, \ldots, 5$, we compute

$$\begin{pmatrix} \tilde{f}_m \\ \tilde{f}_{6+m} \\ \tilde{f}_{12+m} \\ \tilde{f}_{18+m} \end{pmatrix} = \begin{pmatrix} \alpha_1 & \alpha_3 & \alpha_4 & \alpha_2 \\ \alpha_2 & \alpha_1 & \alpha_3 & \alpha_4 \\ \alpha_4 & \alpha_2 & \alpha_1 & \alpha_3 \\ \alpha_3 & \alpha_4 & \alpha_2 & \alpha_1 \end{pmatrix} \begin{pmatrix} v_m \\ v_{6+m} \\ v_{12+m} \\ v_{18+m} \end{pmatrix}$$

Now we use Winograd's algorithm 9 to compute the 6 cyclic convolutions. We first compute the following 24 products: for $m = 0, \ldots, 5$

$$p_{1,m} = \int_1 (v_m - v_{6+m} + v_{12+m} - v_{18+m})$$
$$p_{2,m} = \int_2 (v_m + v_{6+m} - v_{12+m} - v_{18+m})$$
$$p_{3,m} = \int_3 (v_{6+m} - v_{18+m})$$
$$p_{4,m} = \int_4 (v_m - v_{12+m})$$

where

$$\int_1 = \tfrac{1}{4}(\epsilon - \epsilon^2 + \epsilon^4 - \epsilon^3)$$
$$\int_2 = \tfrac{1}{2}(\epsilon - \epsilon^4)$$
$$\int_3 = \tfrac{1}{4}(\epsilon + \epsilon^2 - \epsilon^4 - \epsilon^3)$$
$$\int_4 = \epsilon - \epsilon^2 - \epsilon^4 + \epsilon^3 .$$

Observe that $\int_1$ is real and $\int_2$, $\int_3$, and $\int_4$ are pure imaginary. We may assume that the $\int_i$ have been computed once and for all and are stored in memory. We can compute the 6 cyclic convolutions with 36 additions and 24 multiplications.

We also compute the 6 non-essential multiplications $p_{0,m} = (-5/4)(v_m + v_{6+m} + v_{12+m} + v_{18+m})$ for $m = 0, \ldots, 5$. This requires 18 additions and 6 multiplications by $(-5/4)$.

Then,

$$\begin{pmatrix} \tilde{f}_m \\ \tilde{f}_{6+m} \\ \tilde{f}_{12+m} \\ \tilde{f}_{18+m} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & -1 & 0 \\ 1 & -1 & 1 & 0 & -1 \\ 1 & 1 & -1 & 1 & 0 \\ 1 & -1 & -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} p_{0,m} \\ p_{1,m} \\ p_{2,m} \\ p_{3,m} \\ p_{4,m} \end{pmatrix}$$

These we can compute with 60 additions. Also, for $j = 0, \ldots, 23$, the $\tilde{f}_j$ are precisely $\hat{f}(a,b)-\hat{f}(0,0)$ permuted around. We next describe this permutation. Recall that $\langle \lambda, (x,y) \rangle = \epsilon^x$. Using the notation of Lemma II-1, $\langle \lambda_{(a,b)}, (x,y) \rangle = \langle \lambda, (a,b)(x,y) \rangle = \langle \lambda, (ax-2yb, ay+bx) \rangle = \epsilon^{ax-2by}$. But for some integers $0 \leq r,s \leq 5$, $\langle \lambda_{(a,b)}, (x,y) \rangle = \epsilon^{rx+sy}$. We must have $a = r \bmod 5$ and $b = 2s \bmod 5$. This says that $\hat{f}(r,s) = \tilde{f}_{r,2s} + \hat{f}(0,0)$, where the subscripts are taken mod 5, for all $(r,s) \neq 0$, $(r,s) \neq 0$. Computing these 24 coefficients require 24 more additions. Finally, we compute $\hat{f}(0,0)$ with 24 additions. Altogether, we have computed $DFT(Z_5 \oplus Z_5)$ with 24 essential multiplications, 6 multiplications by (-5/4), and 258 additions.

Remark: The modification in this construction is due to Winograd.

III. A Direct Sum Theorem

III-1. Introduction

In the previous section we saw that if G is a finite direct sum of $\mathbb{Z}/p\mathbb{Z}$ where p is prime, then we can compute $DFT(G)$ by first computing what we call a direct sum system of semilinear forms. In a subsequent paper we will show that this scheme can be used for the DFT of any finite abelian group. We also saw earlier how we can use algorithms for systems of bilinear forms to yield algorithms for these DFTs. We are therefore motivated to explore the multiplicative complexity of certain direct sum systems of bilinear forms.

Let G be a field, and let $(\underline{x}) = (x_1 \cdots x_c)^t$ and $(\underline{y}) = (y_1 \cdots y_b)^t$ be vectors of distinct indeterminants over G. For $k = 1, \ldots, a$, let $b_k = \sum_{i,j} f_{ijk} x_i y_j$ be a collection of bilinear forms, with $f_{ijk} \in G$. We may write

$$\begin{pmatrix} b_1 \\ \vdots \\ b_a \end{pmatrix} = \begin{pmatrix} L_{11} & \cdots & L_{1b} \\ \vdots & & \vdots \\ L_{a1} & \cdots & L_{ab} \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_b \end{pmatrix}$$

where $L_{kj} = \sum_i f_{ijk} x_i$. Let us also write

$$A(\underline{x}) = \begin{pmatrix} L_{11} & \cdots & L_{1b} \\ \vdots & & \vdots \\ L_{a1} & \cdots & L_{ab} \end{pmatrix}$$

We call $A(\underline{x})(\underline{y})$ a system of bilinear forms.

Definition III-1: An algorithm $\mathcal{A}$ is a finite sequence of elements $h_j \in F = G(\underline{x},\underline{y})$, the purely transcendental extension of G obtained by adjoining to it the indeterminants in $(\underline{x})$ and $(\underline{y})$, where either

(i) $h_j \in G \cup \{x_1, \ldots, x_c, y_1, \ldots, y_b\}$ or

(ii) $h_j = h_p * h_q$, $p < j$, $q < j$, and $*$ is one of the field operations in F.

Definition III-2: The algorithm $\mathcal{A}$ is said to compute the system $A(\underline{x})(\underline{y})$ if for every $k = 1,\ldots,a$ there exists j such that $b_k = h_j$, where as above, b_k is one of the bilinear forms in the system and h_j is a step in .

Remark: The algorithm in our definition has no branching; for a justification see ([9] , section IIIa). Also compare our definition with the ones in the cited reference; here we fix our base set to be $G \cup \{x_1, \ldots, x_c, y_1, \ldots, y_b\}$.

Definition III-3: A step h_j of $\mathcal{A}$ is called essential of $h_j = h_p * h_q$ and either

(i) $*$ is a multiplication, $h_p \notin G$, and $h_q \notin G$; or

(ii) $*$ is a division, and the divisor $h_q \notin G$.

The following presentation theorem is a direct consequence of our definitions. For a proof, see [6].

Theorem III-1: Let $\mathcal{A}$ be an algorithm computing $\Lambda(\underline{x})(\underline{y})$ and let $h(1), \ldots, h(t)$ be the essential steps of $\mathcal{A}$. Then there exists an $a \times t$ matrix M with entries from G, and an a-dimensional vector $L(\underline{x},\underline{y})$ of linear forms (not necessarily homogeneous) in the indeterminants of $(\underline{x})$ and $(\underline{y})$ over G, such that

$$\Lambda(\underline{x})(\underline{y}) = M \left(h(1) \ \ldots\ h(t) \right)^t + L(\underline{x},\underline{y}) .$$

Furthermore, for $k = 1, \ldots, t$, $h(k) =$

$$\left(\sum_{i<k} a_{ik} h(i) + \underline{x}_k + \underline{y}_k + f_k \right) * \left(\sum_{i<k} a'_{ik} h(i) + \underline{x}'_k + \underline{y}'_k + f'_k \right) ,$$

where a_{ik}, a'_{ik}, f_k, $f'_k \in G$, $\underline{x}_k$ and $\underline{x}'_k$ are homogeneous linear forms in $(\underline{x})$ over G, $\underline{y}_k$ and $\underline{y}'_k$ are homogeneous linear forms in $(\underline{y})$ over G, and $*$ is either a multiplication or a division.

Definition III-4: We define the degree of the algorithm $\mathcal{A}$ to be the number of essential steps it contains, and we denote it by $\mu(\mathcal{A})$.

Definition III-5: We define $\mu\Lambda(\underline{x}) = \inf \{ t \mid t = \mu(\mathcal{A})$ and $\mathcal{A}$ is some algorithm computing

$A(\underline{x})(\underline{y})\}$. We call $\mu A(\underline{x})$ the complexity of the system $A(\underline{x})(\underline{y})$.

Definition III-6: If $\mu A(\underline{x}) = t$ and $\mathcal{A}$ is an algorithm of degree t which computes $A(\underline{x})(\underline{y})$, then we call $\mathcal{A}$ a minimal algorithm for $A(\underline{x})(\underline{y})$.

We next list a hierarchy of classes of algorithms for computing systems of bilinear forms with which we shall be concerned for the rest of this paper.

Definition III-7: $\mathcal{A}$ is called a division-free algorithm if all of the essential steps of $\mathcal{A}$ are multiplications.

Definition III-8: If each of the essential steps of $\mathcal{A}$ is a multiplication of the form $(\underline{x}_k + \underline{y}_k)(\underline{x}'_k + \underline{y}'_k)$ where $\underline{x}_k$ and $\underline{x}'_k$ are homogeneous linear forms in the indeterminants of $(\underline{x})$ and $\underline{y}_k$ and $\underline{y}'_k$ are homogeneous linear forms in the indeterminants of $(\underline{y})$, the $\mathcal{A}$ is called a quadratic algorithm.

Corresponding to theorem III-1, we have the following presentation theorem for quadratic algorithms.

Theorem III-1': Let $\mathcal{A}$ be a quadratic algorithm computing $A(\underline{x})(\underline{y})$ and let $h(1), \ldots, h(t)$ be the essential steps of $\mathcal{A}$. Then there exists a matrix M with entries from G such that

$$A(\underline{x})(\underline{y}) = M\left(h(1) \ldots h(t)\right)^t.$$

Definition III-9: If each essential step of a quadratic algorithm $\mathcal{A}$ has the form

$$h(k) = \left(\sum_i a_{ik}x_i\right)\left(\sum_i b_{ik}y_i\right)$$

and $a_{ik}, b_{ik} \in G$, then $\mathcal{A}$ is called a bilinear (or non-commutative) algorithm.

Definition III-10: For a bilinear algorithm $\mathcal{A}$, we define $\bar{\mu}\mathcal{A}$ to equal the number of essential steps in $\mathcal{A}$. We define $\bar{\mu}A(\underline{x}) = \inf\{t \mid t = \bar{\mu}\mathcal{A}$ and $\mathcal{A}$ is some bilinear algorithm computing $A(\underline{x})(\underline{y})$. We call $\bar{\mu}\mathcal{A}$ (resp. $\bar{\mu}A(\underline{x})$) the bilinear complexity of $\mathcal{A}$ (resp. $A(\underline{x})(\underline{y})$).

Remark: Notice that we did not make a special definition for quadratic complexity analogous to our definition III-10 of the bilinear complexity. This is because in [12] Winograd proved that given an algorithm $\mathcal{A}$ of degree t for computing $A(\underline{x})(\underline{y})$, we can induce from it a quadratic algorithm $\tilde{\mathcal{A}}$ of degree at most t which also computes $A(\underline{x})(\underline{y})$. In particular, if $\mathcal{A}$ is minimal, then the induced algorithm $\tilde{\mathcal{A}}$ must also be minimal and, therefore, of the same degree.

The proof of Winograd's theorem cited in the previous remark is constructive; one actually constructs the quadratic algorithm from the given one.

We also refer the reader to [9] for a nice treatment of this inducing process. In the following chapter we shall only need the following consequence of the proof.

Theorem III-2 (Winograd): Let $\mathcal{A}$ be a minimal algorithm computing $A(\underline{x})(\underline{y})$. Suppose that $\mu(\mathcal{A}) \geq 2$ and that the first two essential steps of $\mathcal{A}$ are the multiplications

$$h(1) = (\underline{x}_1 + \underline{y}_1 + f_1)(\underline{x}'_1 + \underline{y}'_1 + \underline{f}'_1) \quad \text{and}$$

$$h(2) = (ah(1) + \underline{x}_2 + \underline{y}_2 + f_2)(a'h(1) + \underline{x}'_2 + \underline{y}'_2 + \underline{f}'_2).$$

Let $L = f_1(\underline{x}'_1 + \underline{y}'_1) + f'_1(\underline{x}_1 + \underline{y}_1)$. Then there exists a minimal quadratic algorithm $\tilde{\mathcal{A}}$ which computes $A(\underline{x})(\underline{y})$ and which has as its first two essential steps the multiplications

$$\tilde{h}(1) = (\underline{x}_1 + \underline{y}_1)(\underline{x}'_1 + \underline{y}'_1) \quad \text{and}$$

$$\tilde{h}(2) = (aL + \underline{x}_2 + \underline{y}_2)(a'L + \underline{x}'_2 + \underline{y}'_2) .$$

III-2. A direct sum theorem for quadratic extensions

Let $x_0, x_1, y_{0,1}, y_{1,1}, \cdots\cdots, y_{0,b}, y_{1,b}$ be distinct indeterminants over G. Let C be the companion matrix of an irreducible quadratic polynomial. Let $(\underline{y}) = (y_{0,1}\ y_{1,1}\ \cdots\ y_{0,b}\ y_{1,b})^t$ and let $C(\underline{x}) = x_0 I + x_1 C$, where I denotes the 2×2 identity matrix. In this section we will determine the bilinear complexity of the system

$$bC(\underline{x})(\underline{y}) = \begin{pmatrix} C(\underline{x}) & & 0 \\ & C(\underline{x}) & \\ 0 & & C(\underline{x}) \end{pmatrix} \begin{pmatrix} \\ \underline{y} \\ \\ \end{pmatrix}$$

Lemma III-1: The determinant $|C(\underline{x})|$ does not split.

Proof: Suppose $C = \begin{pmatrix} 0 & a \\ 1 & b \end{pmatrix}$. Then

$$|C(\underline{x})| = \begin{vmatrix} x_0 & ax_1 \\ x_1 & x_0+bx_1 \end{vmatrix} = x_0^2 + bx_0x_1 - ax_1^2.$$ The minimum polynomial of C is $x^2 - bx - a$, and by assumption is irreducible.

Definition III-11: Let $A(\underline{x})$ be a matrix of linear forms. We define $d_rA(\underline{x})$ to be the maximum number of rows $r_1, \ldots, r_k$ of $A(\underline{x})$ such that if $\sum_{j=1}^{k} a_{ij}r_j \in G$ and $a_{ij} \in G$, then $a_{ij} = 0$ for all $j = 1, \ldots, k$. $d_rA(\underline{x})$ is called the G-row rank of $A(\underline{x})$. We similarly define the G-column rank of $A(\underline{x})$ and denote it be $d_cA(\underline{x})$.

Definition III-12: If either $A(\underline{x})$ or $A^t(\underline{x})$ has the form $(a_1m \cdots a_sm)$, where $a_j \in G$ and m is a linear form in $(\underline{x})$ over G, then $A(\underline{x})$ is called a rank-1 vector.

Theorem III-3: Let $A(\underline{x})$ be an $a \times b$ matrix of homogeneous linear forms in $(\underline{x})$ over G. Suppose that $d_r A(\underline{x}) = a$, $d_c A(\underline{x}) = b$, and $\bar{\mu} A(\underline{x}) = t$. Then there exists a $t \times t$ matrix

$$B(\underline{x}) = \left(\begin{array}{c|c} A(\underline{x}) & * \\ \hline * & * \end{array}\right)$$

such that the determinant $|B(\underline{x})|$ is not zero and splits over G into linear factors.

Proof: Our assumption $\bar{\mu} A(\underline{x}) = t$ implies that there exists a bilinear algorithm of the form

$$A(\underline{x})(\underline{y}) = M \begin{pmatrix} L_1 \cdot L_1' \\ \vdots \\ L_t \cdot L_t' \end{pmatrix}$$

$$= M \begin{pmatrix} a_{11}L_1 & \cdots & a_{1b}L_1 \\ & & \\ a_{t1}L_t & \cdots & a_{tb}L_t \end{pmatrix} \begin{pmatrix} y_1 \\ \\ y_b \end{pmatrix}$$

$$= M\, \alpha(\underline{y}),$$

where $L_i = \sum_{j=1}^{a} f_{ij}x_j$, $L_i' = \sum_{j=1}^{b} a_{ij}y_j$,

and a_{ij}, $f_{ij} \in G$. The rows of α are linearly

independent because $\bar{\mu} A(\underline{x}) = t$. Let $k = t-a$. We may assume that the first k rows of α and the rows of $A(\underline{x})$ are all linearly independent. Let

$$\tilde{A}(\underline{x}) = \begin{pmatrix} A(\underline{x}) \\ a_{11}L_1 \cdots\cdots a_{1b}L_1 \\ \vdots \qquad\qquad \vdots \\ a_{k1}L_k \cdots\cdots a_{kb}L_k \end{pmatrix}$$

By the duality theorem for the transposes of systems of bilinear forms (see [9] , theorem III-4) $\bar{\mu} \tilde{A}(\underline{x}) = \bar{\mu} \tilde{A}^t(\underline{x})$, so there exists an algorithm of the form

$$\tilde{A}^t(\underline{x})(\underline{z}) = N \begin{pmatrix} M_1 \cdot M_1' \\ \vdots \\ M_t \cdot M_t' \end{pmatrix}$$

$$= N \begin{pmatrix} b_{11}M_1 \cdots\cdots b_{1t}M_1 \\ \vdots \qquad\qquad \vdots \\ b_{t1}M_t \cdots\cdots b_{tt}M_t \end{pmatrix} (\underline{z})$$

$$= N \beta(\underline{z}) ,$$

where M_i (resp. M_i') are homogeneous linear forms in $(\underline{x})$ (resp. $(\underline{y})$) and $(\underline{z})$ is a vector of t distinct indeterminants.

Let $l = t-b$. We may assume that the first l rows of β and the rows of $A^t(\underline{x})$ are linearly

independent. Let

$$B^t(\underline{x}) = \begin{pmatrix} \tilde{\Lambda}^t(\underline{x}) \\ b_{11}M_1 \cdots\cdots b_{1t}M_1 \\ \\ b_{l1}M_1 \cdots\cdots b_{lt}M_t \end{pmatrix}$$

Then $d_r B^t(\underline{x}) = d_c B^t(\underline{x}) = \bar{\mu}\, B^t(\underline{x}) = t$. Finally, because $\bar{\mu}\, B^t(\underline{x}) = \bar{\mu}\, B(\underline{x})$, the G-row span of $B(\underline{x})$ has a basis consisting of all rank-1 vectors. Thus there exists a non-singular matrix C over G such that

$$CB(\underline{x}) = \begin{pmatrix} c_{11}N_1 \cdots\cdots c_{1t}N_t \\ \vdots \qquad\qquad \vdots \\ c_{t1}N_t \cdots\cdots c_{tt}N_t \end{pmatrix}$$

$$\begin{pmatrix} N_1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & N_t \end{pmatrix} \begin{pmatrix} c_{11} \cdots\cdots c_{1t} \\ \vdots \qquad\qquad \vdots \\ c_{t1} \cdots\cdots c_{tt} \end{pmatrix}.$$

Let $(N) = \begin{pmatrix} N_1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & N_t \end{pmatrix}$ and $\gamma = \begin{pmatrix} c_{11} \cdots\cdots c_{1t} \\ \vdots \qquad\qquad \vdots \\ c_{t1} \cdots\cdots c_{tt} \end{pmatrix}$.

(Notice, N_i are linear forms and $c_{ij} \in G$.) We claim that γ is non-singular. Otherwise there exists a vector $v \neq 0$ such that $CB(\underline{x})v = (N)\gamma\, v = 0$, from which it follows that $B(\underline{x})v = 0$, contradicting

our previous result that $d_c P(\underline{x}) = t$. The matrix $B(\underline{x})$ clearly satisfies the conditions of our theorem.

<u>Theorem</u> III-4: Let $A(\underline{x}) = \begin{pmatrix} L_1 & L_2 \\ L_3 & L_4 \end{pmatrix}$, where L_i are homogeneous linear forms in $(\underline{x})$ over G. Suppose that $|A(\underline{x})| \neq 0$ and does not split over G. Then $\bar{\mu}\, kA(\underline{x}) \geq 3k$ (Recall, $kA(\underline{x})$ means the k-fold direct sum of $A(\underline{x})$).

<u>Proof</u>: Let $\bar{\mu}\, kA(\underline{x}) = t$, and let $s = t-2k$. Clearly $d_r kA(\underline{x}) = d_c kA(\underline{x}) = 2k$, so by the previous theorem there exists a $t \times t$ matrix of the form

$$B(\underline{x}) = \left(\begin{array}{c|c} kA(\underline{x}) & D(\underline{x}) \\ \hline \multicolumn{2}{c}{E(\underline{x})} \end{array} \right)$$

such that $|B(\underline{x})| \neq 0$ and splits into linear factors over G. We can block-expand for the determinant as follows: $B(\underline{x}) = \sum_i a_i |E_i||A_i|$, where $a_i \in G$, E_i range over all $s \times s$ minors of $E(\underline{x})$, and A_i are the complementary $2k$ columns of $\left(kA(\underline{x}) \mid D(\underline{x})\right)$. Now if $t \leq 3k-1$, each A_i contains at least $k+1$ columns of $kA(\underline{x})$. Hence $|A(\underline{x})|$ is a factor in each summand of this block expansion for the determinant. Thus, because of unique factorization in polynomial rings, $|B(\underline{x})|$ cannot split into linear factors over G.

Corollary: Let Q be a quadratic extension field of G and let r, s_1, ..., s_k be arbitrary elements in Q. Any bilinear algorithm for computing the products rs_1, ..., rs_k uses at least 3k multiplications.

Proof: This is an immediate consequence of Lemma III-1 and theorem III-4.

IV. Classification of Minimal algorithms

IV-1. Introduction

In chapter II we saw that in certain cases the DFT is computationally equivalent to a direct-sum system of semilinear forms whose direct summands can be viewed as products in finite algebraic extension fields. We also saw how minimal algorithms for systems of bilinear forms for computing products in algebraic extension fields yielded algorithms for these direct summands. In chapter I we pointed out that Auslander and Winograd in [3] have proved that direct sums of minimal algorithms so derived yield minimal algorithm for DFT computations in many cases. In this chapter we study the classification problem for minimal algorithms for systems of bilinear forms for computing products in finite algebraic extension fields.

In a recent paper [10] Winograd classified all minimal bilinear algorithms for computing such products. Furthermore, he showed that every minmal quadratic algorithm for computing products in an algebraic extension field of degree 2 is necessarily bilinear. What we have tried to do is prove that all minimal algorithms for computing products in finite algebraic extension fields are bilinear. We have not succeeded

in this task, but we did make considerable progress towards this goal.

In this chapter we prove that for a certain class of systems of bilinear forms all minimal division-free algorithms are bilinear. This class includes systems for computing products in algebraic extension fields of arbitrary finite degree. It also includes systems for computing products of Toeplitz matrices with vectors and systems for computing products of Hankel (FIR filter) matrices with vectors. Finally, an interesting corollary of our results is that the multiplicative complexity of the quaternion product ofver a real field is 8.

Let us quickly outline the ideas in this chapter. Let F be an algebraic extension field of degree n of a field G, and let $F(\underline{x})(\underline{y})$ be a system of bilinear forms for computing the product in F. $F(\underline{x})$ is an $n \times n$ matrix of homogeneous linear forms in n indeterminants over G. The system $F(\underline{x})(\underline{y})$ has three important properties. First, if $f,g \in G^n$ and $f^t F(\underline{x})g = 0$, then either $f = 0$ or $g = 0$. Second, the entries of $F(\underline{x})$ span an n-dimensional vector space over G. And third, the multiplicative complexity of the system $F(\underline{x})(\underline{y})$ is $2n-1$.

This motivates our studying the following general

situation: $A(\underline{x})(\underline{y})$ is a system of bilinear forms, where $A(\underline{x})$ is an $a \times b$ matrix of homogeneous linear forms over a ground field G, and the system satisfies the following:

(i) if $f \in G^a$ and $g \in G^b$ and $f^t A(\underline{x}) g = 0$, then either $f = 0$ or $g = 0$;

(ii) the entries of $A(\underline{x})$ span an a-dimensional vector space over G:

(iii) the multiplicative complexity of the system $A(\underline{x})(\underline{y})$ is $a+b-1$.

We look closely at how these systems and minimal quadratic algorithms for computing them are affected by various projections. This is the technique of substitutions which has been successfully used by Auslander [2], [3], Winograd [3], [11], [13], and various other authors in studying the complexity of certain systems of bilinear and semilinear forms. The properties of $A(\underline{x})(\underline{y})$ which we have listed above guarrantee the existence of the various substitutions that we need, and also allows us to draw much informations from the systems and algorithms induced by these substitutions. We are then able to show that all minimal quadratic algorithms for computing $A(\underline{x})(\underline{y})$ are bilinear.

To pass from minimal quadratic algorithms to minimal division-free algorithms, we use the

presentation theorem of Winograd [12] cited in the previous chapter (theorem III-2). This theorem told us how, given any minimal algorithm for computing $\Lambda(\underline{x})(\underline{y})$, we can construct from it a minimal quadratic algorithm for computing this system. Finally, when the original algorithm is division-free, we are able to use an induction argument and, putting all these ingredients together, prove that the algorithm must be bilinear.

IV-2. Multiplication in Division Rings

In this chapter we shall be interested in the structure of minimal division-free algorithms for computing products in certain division rings (for example, finite algebraic extension fields). In this section we describe what systems of bilinear forms for computing products in division-rings look like, and we give two important examples.

Let D be a division ring of dimension n over a fields G, and let us fix a vector space isomorphism $i: D \longrightarrow G^n$. Because multiplication by an element in D is a linear operation, we induce the left-regular representation $L: D \longrightarrow M_n(G)$ defined as follows: for $d_1, d_2 \in D$, $L(d_1)$ is the matrix such that $L(d_1)i(d_2) = i(d_1 d_2)$.

Let $(\underline{x}) = (x_1 \ldots x_n)^t$ and $(\underline{y}) = (y_1 \ldots y_n)^t$

be vectors of distinct indeterminants over G, and for $j = 1, \ldots, n$, let $e_j \in G^n$ be the vector $(0 \ldots 1 \ldots 0)^t$, the single non-zero entry is 1, and it appears in the j-th row. Let $d_j = i^{-1}(e_j)$ and let $D(\underline{x}) = \sum_{j=1}^{n} x_j L(d_j)$. $D(\underline{x})$ is an $n \times n$ matrix of homogeneous linear forms in the indeterminants of $(\underline{x})$ over G. The vector $D(\underline{x})(\underline{y})$ is called a system of bilinear forms for computing the product in D.

Theorem IV-1: Let $f, g \in G^n$ and suppose that $f^t D(\underline{x}) g = 0$. Then either $f=0$ or $g=0$.

Proof: Choose a non-zero $d \in D$ and define the linear transformation $L_d: D \longrightarrow G^n$ by $L_d(h) = L(h)i(d) = i(hd)$ for every $h \in D$. Because D contains no zero-divisors, the kernel of L_d is trivial. Therefore, for every non-zero $d \in D$, $L(D)d = L_d(D) = G^n$.

Now suppose that $f^t D(\underline{x}) g = 0$. Then

$$f^t \left(\sum_{j=1}^{n} x_j L(d_j) \right) g = \sum_{j=1}^{n} x_j f^t L(d_j) g = 0.$$

Because $x_1, \ldots, x_n$ are distinct indeterminants, we must have $f^t L(d_j) g = 0$ for each $j = 1, \ldots, n$. Now $d_1, \ldots, d_n$ generate D as a vector space over G,

and therefore $f^t L(D)g = 0$. If $g \neq 0$, then by the previous paragraph $L(D)g = G^n$. We must therefore have that $f^t G^n = 0$, which implies that $f = 0$, and we have proven the theorem.

Example IV-1: Let $G[u]$ be the polynomial ring in the one variable u over G, and let $g(u) \in G[u]$ be a monic irreducible polynomial of degree n. Let C_g be the companion matrix of $g(u)$, and let $\mathcal{A}(C_g)$ be the algebra generated by C_g over G. Then an isomorphism

$$I : G[u] / \langle g(u) \rangle \longrightarrow \mathcal{A}(C_g)$$

is given by $I(u) = C_g$. Fix the vector space isomorphism $i: G[u] / \langle g(u) \rangle \longrightarrow G^n$ defined by $i(u^{j-1}) = e_j$, for $j = 1, \ldots, n$. Relative to this choice of basis, the regular representation has the form

$$L\left(\sum_{j=0}^{n-1} a_j u^j\right) = \sum_{j=0}^{n-1} a_j C_g{}^j \quad , \quad a_j \in G$$

and $\left(\sum_{j=0}^{n-1} x_j C_g{}^j\right)(\underline{y})$ is a system of bilinear forms for computing the product in the extension field $G[u] / \langle g(u) \rangle$.

Example IV-2: The quaternions K over a real field G are a four dimensional ring over G with generators 1, i, j, k, and with the relations $i^2 = j^2 = k^2 = -1$, $ij = k$, $jk = i$, and $ki = j$. Let us fix the vector space isomorphism $\underline{i}: K \longrightarrow G^4$ given by $\underline{i}(1) = e_1$, $\underline{i}(i) = e_2$, $\underline{i}(j) = e_3$, and $\underline{i}(k) = e_4$, where $e_j \in G^4$ are as defined in the beginning of this section. Then computing the product of the quaternions is the same as computing the following system of bilinear forms:

$$\begin{pmatrix} x_1 & -x_2 & -x_3 & -x_4 \\ x_2 & x_1 & x_4 & -x_3 \\ x_3 & -x_4 & x_1 & x_2 \\ x_4 & x_3 & -x_2 & x_1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix}$$

IV-3. Definite Matrices

Motivated by the statement of theorem IV-1, we now give the following definition.

Definition IV-1: Let $(\underline{x}) = (x_1, \ldots, x_c)$ be a vector of distinct indeterminants over a field G, and let $A(\underline{x})$ be an $a \times b$ matrix whose entries are homogeneous linear forms in the indeterminants of $(\underline{x})$ over G. We will call $A(\underline{x})$ definite of it satisfies the following condition: for every $f \in G^a$ and $g \in G^b$, if $f^t A(\underline{x}) g = 0$ then either $f=0$ or $g=0$.

Definition IV-2: Let $A(\underline{x})$ be as above (not necessarily definite). We define the G-rank of $A(\underline{x})$ to be the dimension of the vector space generated by all the entries in $A(\underline{x})$ over G, and we denote it by $r_G(A(\underline{x}))$.

Example IV-3: If $D(\underline{x})(\underline{y})$ is a system of bilinear forms for computing the product in a division ring D of dimension n over a field G, then $r_G(D(\underline{x})) = n$ and by theorem IV-1 $D(\underline{x})$ is definite.

Example IV-4: An $n \times n$ matrix T is called a Toeplitz matrix if for all $2 \leq i,j \leq n$ we have $T(i,j) = T(i-1,j-1)$, where by $T(i,j)$ we mean the i,j^{th} entry of T. The system of bilinear forms for computing the product of an $n \times n$ Toeplitz matrix and a vector is $T(\underline{x})(\underline{y})$, where

$$T(\underline{x}) = \begin{pmatrix} x_0 & x_1 & x_{n+1} & \cdots & x_{2n-2} \\ x_1 & x_0 & x_n & & x_{2n-3} \\ x_2 & x_1 & x_0 & & x_{2n-4} \\ \vdots & \vdots & \vdots & & \vdots \\ x_{n-1} & x_{n-2} & x_{n-3} & \cdots & x_0 \end{pmatrix}$$

We leave it to the reader to verify that $T(\underline{x})$ is definite over any field. We will, however, present the following proof that $T(\underline{x})$ is definite over the rationals, Q, because we will use this argument again later in the paper. Suppose that

$f, g \in Q^n$ and $f^t T(\underline{x})g = 0$. Let $a \in Q$ be a non-zero square-free integer not equal to ± 1. Then by Eisenstein's criterion, $u^n - a$ is irreducible over Q. Let $\tilde{T}(\underline{x})$ be the matrix obtained from $T(\underline{x})$ by substituting ax_{2n-1-j} for x_j for all $j = n, \ldots, 2n-2$. Then $f^t \tilde{T}(\underline{x})g = 0$. But $\tilde{T}(\underline{x})(\underline{y})$ is a system of bilinear forms for computing products in the extension field $Q[u] / \langle u^n - a \rangle$. Therefore either $f = 0$ or $g = 0$, and we have shown that $T(\underline{x})$ is definite.

Example IV-5: Let $A(\underline{x}) = \begin{pmatrix} x_0 & -x_1 \\ x_1 & x_0 \end{pmatrix}$ and $(\underline{y}) = (y_0 \quad y_1)^t$. $A(\underline{x})(\underline{y})$ is the system of bilinear forms for computing the product of two arbitrary complex numbers. The matrix $A(\underline{x})$ is definite over the rationals Q but not over $Q(i)$, the Gaussian rationals. This is not surprising because u^2+1 is irreducible over Q but not over $Q(i)$. Similarly, the matrix given in example IV-2 is definite over Q but not over $Q(i)$. Of course, the quaternions as a division ring over $Q(i)$ is a vector space of dimension 2, and the matrix of the system of bilinear forms for computing the quaternion product over $Q(i)$ is $\begin{pmatrix} x_1 & -\bar{x}_2 \\ x_2 & \bar{x}_1 \end{pmatrix}$, where by $\bar{x}$ we mean the complex conjugate of x. This last matrix is

definite over $Q(i)$ by theorem IV-1.

Lemma IV-1: Let $A(\underline{x})$ be a definite $a \times b$ matrix. Then

(i) Every non-zero element in the G-row span of $A(\underline{x})$ has G-rank b; every non-zero element in the G-column span of $A(\underline{x})$ has G-rank a.

(ii) If $c \leq a$, $d \leq b$, M is a $c \times a$ G-matrix of rank c, and N is a $b \times d$ G-matrix of rank d, then $MA(\underline{x})N$ is definite.

Proof: (i) Let $v = f^t A(\underline{x})$, where $f \in G^a$ and $f \neq 0$. Clearly $r_G(v) \leq b$. Suppose $r_G(v) \leq b-1$. Then there exists a non-zero $g \in G^b$ such that $f^t A(\underline{x})g = v \cdot g = 0$. But this cannot be because we assumed that $A(\underline{x})$ is definite. This proves the first part of (i); the second part is proved similarly.

(ii) Our assumptions on M and N imply that if $f \in G^c$ and $g \in G^d$ are both non-zero, then also $f^t M \in G^a$ and $Ng \in G^b$ are both non-zero. Thus, if $f^t MA(\underline{x})Ng = 0$, then either $f^t M = 0$ or $Ng = 0$, which implies that either $f = 0$ or $g = 0$.

Corollary: If $A(\underline{x})$ is a definite $a \times b$ matrix, then (i) every entry of $A(\underline{x})$ is non-zero, and (ii) if $g \in G^b$ and $g \neq 0$, then $r_G(A(\underline{x})g) = a$.

For the rest of this chapter we will be interested in the following situation: $A(\underline{x})$ is a definite $a \times b$

matrix and $\mu A(\underline{x}) = a+b-1$. We should remind the reader of the well known rank theorem [1]:

Theorem IV-2 (Fiduccia-Malcstein): If $A(\underline{x})$ is a definite $a \times b$ matrix, then $\mu A(\underline{x}) \geq a+b-1$.

Lemma IV-2: Let $A(\underline{x})$ be a definite $a \times b$ matrix and let $M(h(1) \cdots h(a+b-1))^t$ be a quadratic algorithm computing $A(\underline{x})(\underline{y})$ (cf theorem III-1'). Then every a columns of M are linearly independent.

Proof: Let $M\ (c_1 | \cdots\cdots | c_{a+b-1})$, where c_i are the columns of M, and suppose, say, that $c_1, \ldots, c_a$ are linearly dependent. Then there is a non-zero $f \in G^a$ such that $f^t M = (0 \cdots 0\ d_{a+1} \cdots \cdots d_{a+b-1})$, where $d_j \in G$ for $j = a+1, \ldots, a+b-1$.

Now $f^t A(\underline{x})(\underline{y}) = \sum_{j=a+1}^{a+b-1} d_j h(j)$, so that $\mu f^t A(\underline{x})$ $\leq b-1$. But by lemma IV-1, $r_G(f^t A(\underline{x})) = b$, and so by theorem IV-2, $\mu f^t A(\underline{x}) \geq b$. We have reached a contradiction. Of course we could have argued similarly for any a columns of M, and so we have proved the lemma.

Remark: Let $A(\underline{x})$ and M be as in lemma IV-2, and let $W = (c_1 | \cdots | c_a)$. Lemma IV-2 asserts that W is invertible, and by lemma IV-1, $W^{-1}A(\underline{x})$ is definite. Clearly if every minimal algorithm computing

$N^{-1}A(\underline{x})(\underline{y})$ is bilinear, then the same holds for all minimal quadratic algorithms computing $A(\underline{x})(\underline{y})$, and vice versa. Therefore, in any argument in which we only impose on $A(\underline{x})$ the condition of definiteness, we may assume that $N = (\, I \mid \tilde{N} \,)$, where I is the $a \times a$ identity matrix. Also, an argument similar to the one in the proof of lemma IV-2 shows that none of the entries of $\tilde{N}$ are 0.

IV-4. Substitutions

Denote by $L_G(\underline{x},\underline{y})$ the G-linear span of the indeterminants in $(\underline{x})$ and $(\underline{y})$, and denote by $G[\underline{x},\underline{y}]$ the polynomial ring $G[x_1,\cdots,x_c,y_1,\cdots y_b]$.

Definition IV-3: Given a mapping $s: \{x_1,\cdots,x_c,y_1,\cdots,y_b\} \longrightarrow L_G(\underline{x},\underline{y})$, s can be extended uniquely to a homomorphism $\bar{s}: G[\underline{x},\underline{y}] \longrightarrow G[\underline{x},\underline{y}]$ which is the identity on G. We shall call such a homomorphism $\bar{s}$ a substitution.

Remark: Compare with the definition of substitution in [3] and [11]. Here we allow substitutions for indeterminants in both $(\underline{x})$ and $(\underline{y})$. Also, because in this chapter we will only discuss division-free algorithms, we will not worry about extending substitutions to the field of fractions $G(\underline{x},\underline{y})$.

In the remainder of this section we describe how

certain substitutions affect systems of bilinear forms and various algorithms which compute them.

A: Let $A(\underline{x})$ be a definite $a \times b$ matrix, and let $A(\underline{x})(\underline{y}) = M\,(h(1) \cdots h(a+b-1))^t$ be a minimal quadratic algorithm, where $h(i) = (\underline{x}_i+\underline{y}_i)(\underline{x}'_i+\underline{y}'_i)$ as in definition III-8. Let v be any non-zero element in the G-span of the entries in $A(\underline{x})(\underline{y})$, and write $v = \sum_i k_i(\underline{x})y_i$, where $k_i(\underline{x})$ are linear forms in $(\underline{x})$ over G. By lemma IV-1, v is definite, and by the corollary to the same lemma, $k_i(\underline{x}) \neq 0$ for all $i = 1, \ldots, b$. We see, in particular, that v is not in the purely transcendental extension field $G(x_1,\ldots,x_c,y_1,\ldots,y_{b-1})$. Therefore the indeterminant y_b must appear somewhere in the algorithm. We may assume, after perhaps permuting the columns of m and correspondingly, the $h(i)$, that

$$\underline{y}_1 = ay_b + \sum_{i=1}^{b-1} b_i y_i \ , \quad \text{with } a,\ b \in G \text{ and } a \neq 0.$$

Let $a_i = b_i/a$. If, for every y_b appearing in our algorithm we substitute $-\sum_{i=1}^{b-1} a_i y_i - \underline{x}_1$, we get a new algorithm of the form

$\tilde{M}(\tilde{h}(2) \cdots \tilde{h}(a+b-1))^t$, where $\tilde{M}$ is the matrix of the last $a+b-2$ columns of M, and $\tilde{h}(i)$ are the projections of the original $h(i)$ induced by the substitution $(\underline{x}_1 + \underline{y}_1) \longrightarrow 0$. For $i = 1, \ldots, b$, let c_i be the columns of $A(\underline{x})$. The new algorith computes $C(\underline{x})(\tilde{\underline{y}}) + \underline{x}_1 c_b$, where $C(\underline{x}) = (c_1 + a_1 c_b \mid \cdots \mid c_{b-1} + a_{b-1} c_b)$ and $(\tilde{\underline{y}}) = (y_1, \cdots \cdots y_{b-1})^t$. $C(\underline{x})$ is also definite because if $f \in G^a$ and $g \in G^{b-1}$ and $f^t C(\underline{x}) g = 0$, then, letting $p = (a_1 \cdots a_{b-1}) \cdot g$ and $q = (g \mid p)^t$, we have $f^t A(\underline{x}) q = 0$.

We can repeat this process of substitution b times. If $k \leq b-1$, then after the k^{th} substitution we are computing a system of the form $\hat{A}(\underline{x})(\hat{\underline{y}}) + C(\underline{x})$, where $\hat{A}(\underline{x})$ is a definite $a \times (b-k)$ matrix, $C(\underline{x})$ is a column vector of quadratic forms in $(\underline{x})$, and we may assume that $(\hat{\underline{y}}) = (y_1 \cdots y_{b-k})^t$. Also, the induced algorithm is minimal, of degree $a+b-k-1$, and each essential multiplication is the projection of one of the original $h(i)$ induced by the substitutions we used.

Remark 1: In the first substitution above we replaced y_b by a linear form in the indeterminants $x_1, \ldots, x_c, y_1, \ldots, y_{b-1}$. This was equivalent to projecting along $(\underline{x}_1 + \underline{y}_1) = 0$. Of course we could have instead substituted for any of the indeterminants

appearing in $\underline{y}_1$. For the rest of the paper, whenever we do substitutions, we will not have to worry about which specific indeterminant we are eliminating. For example, in describing the above substitution, we shall have simply said: let us substitute $(\underline{x}_1 + \underline{y}_1) = 0$.

Remark 2: The above argument shows that we may assume that $\underline{y}_1, \ldots, \underline{y}_b$ are linearly independent. Notice, in fact, that a slight modification of that argument shows the following: let $A(\underline{x})$ be an $a \times b$ matrix of homogeneous linear forms, and suppose that the columns of $A(\underline{x})$ are linearly independent (we are not assuming definiteness). Suppose that $A(\underline{x})(\underline{y}) = M(h(1) \cdots h(t))^t$ is a quadratic algorithm (not necessarily minimal) with, as usual $h(i) = (\underline{x}_i+\underline{y}_i)(\underline{x}'_i+\underline{y}'_i)$. Then we may assume that $\underline{y}_1, \ldots, \underline{y}_b$ are linearly independent. Notice that by theorem IV-2, we have $b \leq t$. The modified argument starts by observing that because the columns of $A(\underline{x})$ are linearly independent, at least one entry in the b^{th} column of $A(\underline{x})$ is not 0. The proof continues almost word for word as the one above, except that after each substitution, (instead of claiming that the induced matrix is definite) we claim that the columns of the induced matrix are again linearly independent.

b: Our second type of substitution is essentially the same as the first, except that we write things differently. Again, let $M(h(1) \cdots h(t))^t$ be a quadratic algorithm computing $A(\underline{x})(\underline{y})$, and assume that the columns of $A(\underline{x})$ are linearly independent. Write as usual $h(i) = (\underline{x}_i+\underline{y}_i)(\underline{x}'_i+\underline{y}'_i)$. By remark 2, we may assume that $\underline{y}_1, \cdots, \underline{y}_b$ are linearly independent. Write $\underline{y}_j = \underline{b}_j \cdot (\underline{y})$ and $\underline{x}_j = \underline{c}_j \cdot (\underline{x})$, where $\underline{b}_j \in G^b$ and $\underline{c}_j \in G^c$. Let (A) be the matrix

$$\begin{pmatrix} \underline{c}_1 \\ \hline \vdots \\ \hline \underline{c}_a \end{pmatrix}$$ and let $(B) = \begin{pmatrix} \underline{b}_1 \\ \hline \vdots \\ \hline \underline{b}_b \end{pmatrix}$. Then

substituting $(\underline{x}_1+\underline{y}_1) = \cdots = (\underline{x}_b+\underline{y}_b) = 0$ is the same as setting $(A)(\underline{x}) + (B)(\underline{y}) = 0$. Because $\underline{y}_1, \ldots, \underline{y}_b$ are linearly independent, (B) is non-singular. Let $(\underline{u}) = -(B)^{-1}A(\underline{x})$. Then $r_G(\underline{u}) = r_G(\underline{x}_1 \cdots \underline{x}_b) = \text{rank } (A)$. Also $A(\underline{x})(\underline{u}) = \tilde{M}(\tilde{h}(b+1) \cdots \tilde{h}(a+b-1))^t$, where $\tilde{M}$ is the matrix composed of the last $a-1$ columns of M and $\tilde{h}(i)$ are the projections of the original $h(i)$ induced by the identifications $(A)(\underline{x}) + (B)(\underline{v}) = 0$.

Remark: In the special case that the original quadratic algorithm is minimal we can say considerably more about the structure of the induced system and

and algorithms. Specifically, if we write $\tilde{h}(i) = k_i k_i'$ for $i = b+1, \ldots, a+b-1$, where k_i and k_i' are linear forms, then we assert that all the k_i and k_i' and the entries of $(\underline{u})$ are in the G-linear span of the entries of $A(\underline{x})$. Recall that $(\underline{y}) = (y_1 \ \ldots \ y_b)^t$, so after b substitutions of the form $(\underline{x}_1+\underline{y}_1) = \ldots = (\underline{x}_b+\underline{y}_b) = 0$ with $\underline{y}_1, \ldots, \underline{y}_b$ linearly independent, the indeterminants $y_1, \ldots, y_b$ vanish. Then either our assertion is true or the induced algorithm contains extraneous variables. The second possibility is excluded by the following lemma.

Lemma IV-3: Let $A(\underline{x})(\underline{y}) = M\,(h(1) \cdots h(t))^t$ be a presentation of a minimal quadratic algorithm. Then every ideterminant appearing in the algorithm must already appear in $A(\underline{x})(\underline{y})$.

Proof: Suppose, on the contrary, that the extraneous indeterminant w appears in the algorithm but not in $A(\underline{x})(\underline{y})$. Let $h(k)$ be any essential step of the algorithm which contains w. Then we can write $h(k) = (aw + L)(a'w + L')$, where L and L' are linear forms, $a, a' \in G$, and we may assume that $a \neq 0$. If we now substitute $-a^{-1}L$ for every w appearing in both $A(\underline{x})(\underline{y})$ and the algorithm we get an induced algorithm of degree at most t-1 which also computes $A(\underline{x})(\underline{y})$. This contradicts our assumption that the original algorithm is minimal.

C: Let $k_1, \ldots, k_d$ be linearly independent homogeneous linear forms in the indeterminants of $(\underline{x})$ over G. We can substitute $k_i = f_i \in G$ for $i = 1, \ldots, d$.

Definition IV-4: If the linearly independent linear forms $k_1, \ldots, k_d$ are in the G-linear span of the entries in $A(\underline{x})$, then the substitutions $k_i = f_i \in G$ are said to be effective on $A(\underline{x})$.

First let us see what this type of substitution does in the following special case. As usual, $(\underline{x}) = (x_1 \cdots x_c)^t$, and let $u = R(\underline{x})$, where R is some $a \times c$ G-matrix. If we substitute

$$x_c - \sum_{i=0}^{c-1} f_i x_i = 0, \quad \text{where } f_i \in G, \quad \text{then we}$$

induce from u the vector

$$\tilde{u} = R \left(\begin{array}{c|c} I & \begin{matrix} f_1 \\ \vdots \\ f_{c-1} \end{matrix} \\ \hline 0 \quad \cdots \quad 0 & \end{array} \right) (\underline{x}) \text{ , where I}$$

is the $(c-1) \times (c-1)$ identity matrix. In particular, we see that this type of substitution may reduce the G-rank of u by at most 1.

Next, if $d \leq c$ and $k_1, \ldots, k_d$ are linearly independent homogeneous linear forms, we may substitute $k_1 = \cdots = k_d = 0$. This will induce, after

perhaps permuting the entries in $(\underline{x})$ and relabeling them,

$$\tilde{u} = R \left(\begin{array}{c|c} I_{c-d} & * \\ \hline \cdots\cdots\ 0 \ \cdots\cdots & \end{array} \right) (\underline{x}) \quad , \text{ where } I_{c-d} \text{ is}$$

the $(c-d) \times (c-d)$ identity matrix. In particular, the G-rank of u has been reduced by at most d. If the substitutions were effective on u then the G-rank of u has been reduced by exactly d.

Remark: If $r_G(\underline{u}) = r$, we can always find r effective substitutions which will induce the 0 vector from $(\underline{u})$. By the above discussion, each of these substitutions will, in fact, lower the G-rank of $(\underline{u})$ by 1. In particular, there exist $r-1$ substitutions which induce from $(\underline{u})$ a vector $(\tilde{\underline{u}}) = vg$, where v is a linear form in $(\underline{x})$ and $g \in G^a$, $g \neq 0$.

Lemma IV-4: Let $A(\underline{x})$ be a definite $a \times b$ matrix, with $r_G(A(\underline{x})) = a$. Let $(\underline{u})$ be a b-dimensional column vector whose entries are all in the G-linear span of the entries of $A(\underline{x})$. Let $n \leq a$. After n substitutions which are effective on $A(\underline{x})$, either the induced vector $(\tilde{\underline{u}}) = 0$ or the induced system $\tilde{A}(\underline{x})(\tilde{\underline{u}}) \neq 0$.

Proof: Suppose that $r_G(\underline{u}) = r \geq 1$. By the

previous remark, there exist $r-1$ substitutions which induce from $\tilde{A}(\underline{x})(\underline{\tilde{u}})$ a system of the form $\hat{A}(\underline{x})(\underline{\hat{u}}) = v\hat{A}(\underline{x})g$, where $v \notin G$ is a linear form and $0 \neq g \in G^a$. These substitutions are effective on $\tilde{A}(\underline{x})$, and so $r_G\hat{A}(\underline{x}) = a-n-(r-1)$. Also $1 = r_G(\underline{\hat{u}}) \leq r_G A(\underline{x})$. Therefore, $a-n-(r-1) \geq 1$. Now $r_G(A(\underline{x})) = a$, and because each substitution reduces the G-rank of $A(\underline{x})g$ by at most 1, we have $r_G(A(\underline{x})g) \geq a-n-(r-1)$. We have shown that $r_G(A(\underline{x})g) \geq 1$, which implies that $v\hat{A}(\underline{x})g \neq 0$, and therefore $\tilde{A}(\underline{x})(\underline{\tilde{u}}) \neq 0$.

Remark: An examination of the above proof shows that the lemma is valid for $n = 0$.

IV-5. More preliminaries

Definition IV-5: An algorithm computing $A(\underline{x})(\underline{y})$ will be called semi-bilinear if each of its essential multiplications has the form $(\underline{x}_i+\underline{y}_i)(\underline{y}'_i)$.

Throughout this section, $A(\underline{x})$ is a definite $a \times b$ matrix with $\mu A(\underline{x}) = a+b-1$.

Lemma IV-5: Let $A(\underline{x})(\underline{y}) = E\,(h(1) \cdots h(t))^t$ be a minimal quadratic algorithm. Suppose that $b-1$ of the essential steps are of the form $(\underline{x}_i+\underline{y}_i)(\underline{y}'_i)$. Then the algorithm is semi-bilinear.

Proof: We may assume that for $i = 1, \ldots, b-1$, $h(i) = (\underline{x}_i+\underline{y}_i)(\underline{y}'_i)$. For $j = 1, \ldots, a$, let r_j be

the j^{th} row of $A(\underline{x})$. Suppose that

$$\left(\sum_{j=1}^{a} a_j r_j \right) (\underline{y}) + \sum_{i=1}^{b-1} b_i h(i) = 0, \quad \text{where}$$

$a_j, b_j \in G$. Let $v = \sum_{j=1}^{a} a_j r_j$. If $v \neq 0$, then by lemma IV-1, $r_G(v) = b$, and by theorem IV-2, $\mu(v) \geq b$. But we can compute $v(\underline{y})$ using the b-1 essential multiplications $h(1), \ldots, h(b-1)$. Therefore $v = 0$, which implies that for all $j = 1, \ldots, a$ we must have $a_j = 0$. Also, then, for $i = 1, \ldots, b-1$ we must have $b_i = 0$, because the essential steps of any minimal algorithm are always G-linearly independent. We have shown that $r_1, \ldots, r_a, h(1), \ldots, h(b-1)$ are G-linearly independent, and a simple dimension argument shows that these form a basis for the vector space spanned by $h(1), \ldots, h(a+b-1)$. This shows that the algorithm is semi-bilinear.

Lemma IV-6: Let $A(\underline{x})(\underline{y}) = M\,(h(1) \cdots h(a+b-1))^t$ be a minimal quadratic algorithm, and suppose that b-1 of the essential steps are multiplications of the form $(\underline{x}_i)(\underline{y}_i)$. Then the algorithm is bilinear.

Proof: The proof is essentially the same as the proof of the previous lemma, and is omitted.

Lemma IV-7: Let $A(\underline{x})(\underline{y}) = M\,(h(1) \cdots h(a+b-1))^t$

be a minimal semi-bilinear algorithm, with $h(i) = (\underline{x}_i+\underline{y}_i)(\underline{y}'_i)$. Then any b of the $\underline{y}_i$ are linearly independent.

Proof: We will only show that $\underline{y}'_1, \ldots, \underline{y}'_b$ are linearly independent. Suppose, on the contrary, that $r_G(\underline{y}'_1, \ldots, \underline{y}'_b) = d \leq b-1$. Then after d substitutions (of the type in IV-4-A), say, $\underline{y}'_1 = \cdots = \underline{y}'_d = 0$, we are computing a system $\tilde{A}(\underline{x})(\tilde{\underline{y}})$, where $\tilde{A}(\underline{x})$ is a definite $a \times (b-d)$ matrix, and our induced algorithm uses at most $a-1$ essential multiplications. But by theorem IV-2, $\mu A(\underline{x}) \geq a$. This contradiction proves the lemma.

Lemma IV-8: Suppose that $a \geq 2$. Then every minimal semibilinear algorithm computing $A(\underline{x})(\underline{y})$ is bilinear.

Proof: We prove the lemma by induction on b. The lemma is obvious for $b=1$. So let us assume that $b \geq 2$. We may substitute $\underline{y}'_1 = 0$ and induce a system $\tilde{A}(\underline{x})(\tilde{\underline{y}})$, where $\tilde{A}(\underline{x})$ is a definite $a \times (b-1)$ matrix. Also, the induced algorithm is both semi-bilinear and minimal. By induction, the induced algorithm is bilinear. Therefore, for $i = 2, \ldots, a+b-1$ we must have $\underline{y}_i = g_i\underline{y}'_1$, where $g_i \in G$. We can argue similarly by substituting $\underline{y}'_2 = 0$ in the original algorithm that, for $i = 1, 3, \ldots, a+b-1$, $\underline{y}_i = h_i\underline{y}'_2$, where $h_i \in G$. By the previous lemma,

$\underline{y}'_1$ and $\underline{y}'_2$ are linearly independent, and so for $i = 3, \ldots, a+b-1$, we must have $\underline{y}_i = 0$. Now because we have assumed that $a \geq 2$, the algorithm satisfies the hypothesis of lemma IV-6 and is therefore bilinear.

Combining the results of the four lemmas in this section, we get the following:

Lemma IV-9: Let $a \geq 2$, and let $A(\underline{x})$ be a definite $a \times b$ matrix with $\mu A(\underline{x}) = a+b-1$. Let $M\ (h(1) \cdots h(a+b-1))^t$ be a quadratic algorithm computing $A(\underline{x})(\underline{y})$. Suppose that $b-1$ of the essential steps of the algorithm are of the form $h(i) = (\underline{x}_i+\underline{y}_i)(\underline{y}'_i)$. Then the algorithm is bilinear.

Remark: We need $a \geq 2$ in the statement of the previous lemma. For example, $(x_1 \ \ x_2)(y_1 \ \ y_2)^t = (x_1+y_2)(y_1) + (x_2-y_1)(y_2)$ is a minimal semibilinear algorithm satisfying all the hypothesis of lemma IV-9 except $a \geq 2$, and this algorithm is not bilinear.

IV-6: On Minimal Quadratic Algorithms

Theorem IV-3: Let $A(\underline{x})$ be a definite $a \times b$ matrix with $a \geq 2$, $r_G(A(\underline{x})) = a$, and $\mu A(\underline{x}) = a+b-1$. Then every minimal quadratic algorithm computing $A(\underline{x})(\underline{y})$ is bilinear.

Proof: Let $A(\underline{x})(\underline{y}) = M\ (h(1) \cdots h(a+b-1))^t$ be a minimal quadratic algorithm with, as usual,

$h(i) = (\underline{x}_i+\underline{y}_i)(\underline{x}'_i+\underline{y}'_i)$. By remark 2 in IV-4-A, we may assume that $\underline{y}_1, \ldots, \underline{y}_b$ are linearly independent. Let us make a subsitution of the type in IV-4-B, and set $\underline{x}_1+\underline{y}_1 = \cdots = \underline{x}_b+\underline{y}_b = 0$. We induce $A(\underline{x})(\underline{u}) = \tilde{M}\ (\tilde{h}(b+1) \ldots \tilde{h}(a+b-1))^t$, and $r_G(\underline{u}) = r_G(\underline{x}_1 \cdots \underline{x}_b)$. If $r_G(u) = 0$, then we are done by lemma IV-9.

Suppose that $r_G(\underline{u}) = r \geq 1$. For $i = b+1, \ldots, a+b-1$, let us write $\tilde{h}(i) = k_i k'_i$, where k_i and k'_i are linear forms. By the remark in IV-4-B, all the k_i, k'_i and the entries in $(\underline{u})$ are in the G-linear span of the entries of $A(\underline{x})$. By the remark after lemma IV-4, $(\tilde{h}(b+1) \cdots \tilde{h}(a+b-1)) \neq 0$, so we may assume that $k_{b+1} \neq 0$. Substituting $k_{b+1} = 0$ is effective on $A(\underline{x})$, and by lemma IV-4, either the induced $(\tilde{\underline{u}}) = 0$ or the induced system $\tilde{A}(\underline{x})(\tilde{\underline{u}}) \neq 0$. If the second possibility occurs, then we can repeat this procedure; we may assume that the induced $k_{b+2} \neq 0$, and substitute $k_{b+2} = 0$. Again this substitution is effective on the induced $\tilde{A}(\underline{x})$. by the discussion in IV-4-C, each such substitution reduces the G-rank of $(\underline{u})$ by at most 1. Since after sufficiently many (at most a-1) substitutions of the form $k_i = 0$ we can annihilate $(\underline{u})$, we see that we can make d substitutions of the form $k_i = 0$, with $r-1 \leq d \leq a-1$, and such that the G-rank of the vector induced form $(\underline{u})$ by these substitutions is 1.

Let us denote the system induced from $A(\underline{x})(\underline{y})$ by these d substitutions by $\hat{A}(\underline{x})(\hat{\underline{u}})$. We may assume that these substitutions where $k_{b+1} = \cdots = k_{b+d} = 0$. Again by lemma IV-4, we have

$$(*) \quad \hat{A}(\underline{x})(\hat{\underline{u}}) = \hat{M}(\hat{h}(b+d+1) \cdots \hat{h}(a+b-1))^t \neq 0.$$

As in the proof of lemma IV-4, we can write $(\underline{u}) = v\underline{g}$, where $v \notin G$ is a linear form in $(\underline{x})$ and $0 \neq \underline{g} \in G^a$. Then $\hat{A}(\underline{x})(\hat{\underline{u}}) = v\hat{A}(\underline{x})\underline{g}$. Also, by the corollary to lemma IV-1, $r_G(A(\underline{x})\underline{g}) = a$, and therefore $r_G(\hat{A}(\underline{x})\underline{g}) \geq a-d$. Therefore the vector $\hat{A}(\underline{x})(\hat{\underline{u}}) = v\hat{A}(\underline{x})\underline{g}$ contains as entries at least a-d linearly independent quadratic forms. But equation (*) above shows that the entries of $\hat{A}(\underline{x})(\hat{\underline{u}})$ are contained in the G-linear span of at most a-d-1 quadratic forms. This contradicts our assumption that $r_G(\underline{u}) \geq 1$, and so we have proven the theorem.

IV-7: On Minimal Division-Free Algorithms

In the previous section we showed that for a certain class of systems of bilinear forms, all minimal quadratic algorithms are bilinear. Using this result and some of the ideas discussed in III-1, we can now show that for this class of systems, all minimal division-free algorithms are essentially bilinear. First we would like to make this last

phrase precise.

Definition IV-5: An essential step of an algorithm is said to be essentially bilinear if it is of the form $(\underline{x}_i+f_i)(\underline{y}_i+f_i')$, where $f_i, f_i' \in G$. An algorithm is said to be essentially bilinear if each of its essential steps is essentially bilinear.

Theorem IV-4: Let $A(\underline{x})$ be a definite $a \times b$ matrix, with $a \geq 2$, $r_G(A(\underline{x})) = a$, and $\mu A(\underline{x}) = a+b-1$. Then every minimal division-free algorithm for computing $A(\underline{x})(\underline{y})$ is essentially bilinear.

Proof: The proof is by induction on b; the theorem is obvious when $b = 1$. Before we can proceed with the induction argument, however, we must first show that the first two essential steps of the algorithm are essentially bilinear.

Let $A(\underline{x})(\underline{y}) = M\,(h(1) \cdots h(a+b-1))^t + L(\underline{x},\underline{y})$ be a presentation of a minimal division-free algorithm. Let the first two essential steps of the algorithm be the multiplications $h(1) = (\underline{x}_1+\underline{y}_1+f_1)(\underline{x}_1'+\underline{y}_1'+f_1')$ and $h(2) = (g h(1)+\underline{x}_2+\underline{y}_2+f_2)(g' h(1)+\underline{x}_2'+\underline{y}_2'+f_2')$. By theorem III-2, there is a minimal quadratic algorithm computing $A(\underline{x})(\underline{y})$ which has as its first essential multiplication $\tilde{h}(1) = (\underline{x}_1+\underline{y}_1)(\underline{x}_1'+\underline{y}_1')$. By theorem IV-3, this induced quadratic algorithm must be bilinear. We may therefore assume that $\underline{y}_1 = \underline{x}_1' = 0$, and that

$h(1) = (\underline{x}_1+f_1)(\underline{y}_1'+f_1')$. Also by theorem III-2, the second essential multiplication of the induced quadratic algorithm has the form $\tilde{h}(2) = \left[(gf_1'\underline{x}_1+\underline{x}_2) + (gf_1\underline{y}_1'+\underline{y}_2)\right]\left[(g'f_1'\underline{x}_1+\underline{x}_2') + (g'f_1\underline{y}_1'+\underline{y}_2')\right]$. And again by theorem IV-3, this induced algorithm is bilinear, and so either $gf_1'\underline{x}_1+\underline{x}_2 = 0$ or $gf_1\underline{y}_1'+\underline{y}_2 = 0$.

Suppose that $gf_1'\underline{x}_1+\underline{x}_2 = 0$. Then $gf_1\underline{y}_1'+\underline{y}_2 \neq 0$. Let us substitute in the original algorithm $\underline{y}_1'+q$ for $\underline{y}_1'$, where $q \in G$, $q \neq 0$. This substitution induces $A(\underline{x})(\underline{y}) + N(\underline{x}) = \tilde{M}\,(\tilde{h}(1) \cdots \tilde{h}(a+b-1))^t + L(\underline{x},\underline{y})$, where $N(\underline{x})$ is a vector of linear forms in $(\underline{x})$. We rewrite this as $A(\underline{x})(\underline{y}) = \tilde{M}\,(\tilde{h}(1) \cdots \cdots \tilde{h}(a+b-1))^t + \tilde{L}(\underline{x},\underline{y})$. By our construction, the first two essential steps of this new algorithm are $\tilde{h}(1) = (\underline{x}_1+f_1)(\underline{y}_1'+f_1'+q)$ and $h(2) = (gh(1)+\underline{x}_2+\underline{y}_2+f_2)(g'h(1)+\underline{x}_2'+\underline{y}_2'+f_2')$. From this new algorithm we induce a minimal quadratic algorithm in which, by theorem III-2, the second essential step has the form $\left[g(f_1'+q)\underline{x}_1+\underline{x}_2+(gf_1\underline{y}_1'+\underline{y}_2)\right]\left[g'(f_1'+g)\underline{x}_1+\underline{x}_2'+(g'f_1\underline{y}_1'+\underline{y}_2')\right]$. Because $gf_1\underline{y}_1'+\underline{y}_2 \neq 0$, we must have $g(f_1'+q)\underline{x}_1+\underline{x}_2 = 0$. But we also assumed that $gf_1'\underline{x}_1+\underline{x}_2 = 0$. This implies that $gq\underline{x}_1 = 0$. Finally, because $\underline{x}_1 \neq 0$ and $q \neq 0$, we must have $g = 0$, and therefore also $\underline{x}_2 = 0$.

Similarly, if we had assumed that $gf_1\underline{y}_1'+\underline{y}_2 = 0$, we could have argued that $g = \underline{y}_2 = 0$. Also similarly we can argue that $g' = 0$ and either $\underline{x}_2' = 0$ or $\underline{y}_2' = 0$. Hence we have that in our original algorithm $h(2)$ is essentially bilinear. We may assume, then, that $h(2) = (\underline{x}_2+f_2)(\underline{y}_2'+f_2')$. We can now proceed with our induction argument.

As we said earlier, the theorem is obvious for $b = 1$. Suppose then that $b \geq 2$. Let $h(k)$ be the first essential step of the algorithm which is not essentially bilinear. Now $h(k) =$

$$\left(\sum_{i<k} a_{ik}h(i) + \underline{x}_k+\underline{y}_k+f_k \right)\left(\sum_{i<k} a_{ik}'h(i) + \underline{x}_k'+\underline{y}_k'+f_k' \right).$$

Substitute $\underline{y}_1'+f_1' = 0$. This induces a system $\tilde{A}(\underline{x})(\underline{\tilde{y}}) + N(\underline{x})$, where $\tilde{A}(\underline{x})$ is a definite $a \times (b-1)$ matrix and $N(\underline{x})$ is a vector of linear forms. Also the induced algorithm is minimal, division-free, and of degree $a+(b-1)-1$. Therefore by induction this induced algorithm is essentially bilinear. Now because the induced steps $h(2)$,··, $h(a+b-1)$ are all essential and linearly independent, we must have that for $i = 2, \ldots, k-1$, $a_{ik} = a_{ik}' = 0$.

Similarly, if we substitute $\underline{y}_2'+f_2' = 0$ in the original algorithm, we can argue that for $i = 1, 3, \ldots, k-1$, $a_{ik} = a_{ik}' = 0$. We have therefore shown that $h(k) = (\underline{x}_k+\underline{y}_k+f_k)(\underline{x}_k'+\underline{y}_k'+f_k')$.

Clearly, because we have assumed that $h(1)$, ..., $h(k-1)$ are essentially bilinear and because we know the special form of $h(k)$, we can rearrange the order of the first k essential steps of our algorithm to obtain a new minimal algorithm computing $A(\underline{x})(y)$ which uses the same essential steps as our original algorithm, but whose first essential step is $h(k) = (\underline{x}_k+\underline{y}_k)(\underline{x}'_k+\underline{y}'_k)$. Now by theorem IV-3, this quadratic algorithm is bilinear. We have therefore shown that $h(k)$ is essentially bilinear. But this contradicts our assumptions on $h(k)$, and proves the theorem.

IV-8. Corollaries

Theorem IV-5: All minimal division-free algorithms for computing products in finite algebraic extension fields are essentially bilinear.

Proof: Let $A(\underline{x})(\underline{y})$ be a system of bilinear forms for computing products in an algebraic extension field of degree n. The discussion in IV-2 showed that $A(\underline{x})$ is a definite $n \times n$ matrix with $r_Q(A(\underline{x})) = n$. Also, Winograd has shown in [13] that $\mu A(\underline{x}) = 2n-1$. Our theorem now follows directly from theorem IV-4.

In example IV-4 we described what an $n \times n$ Toeplitz matrix looks like. A very similar matrix is an $n \times n$ Hankel matrix H in which $H(i,j) =$

$H(i-1,j+1)$, where by $H(i,j)$ we mean the i,j^{th} entry of H. Let $T(\underline{x})(\underline{y})$ be the system for computing the product of an arbitrary $n \times n$ Toeplitz matrix and an arbitrary vector, and let $H(\underline{x})(\underline{y})$ be the system for computing the product of an arbitrary $n \times n$ Hankel matrix with an arbitrary vector. Let J be the $n \times n$ permutation matrix in which $J(n-i-1,i) = 1$ for $i = 1, \ldots, n$. Then $T(\underline{x})J = H(\underline{x})$, and so the two systems $T(\underline{x})(\underline{y})$ and $H(\underline{x})(\underline{y})$ are computationally equivalent; given any algorithm for computing one of them, we can derive from it an algorithm for computing the other by simply relabeling the indeterminants in $(\underline{y})$. The matrices $T(\underline{x})$ and $H(\underline{x})$ are definite, and by theorem IV-2, $\mu T(\underline{x}) = \mu H(\underline{x}) \geq 2n-1$. In [9] Winograd shows how to construct algorithms for $H(\underline{x})(\underline{y})$ which use $2n-1$ essential bilinear multiplication. The basic idea there is that one of the transposes of the system $H(\underline{x})(\underline{y})$ is the system for computing the product of two arbitrary polynomials of degree $n-1$. We can now use an argument similar to the one in example IV-4 to prove:

Theorem IV-6: Over the rationals, all minimal division-free algorithms for computing products of arbitrary $n \times n$ Toeplitz (Hankel) matrices and arbitrary vectors are essentially bilinear.

Proof: Let $T(\underline{x})(\underline{y})$ be the system in question, and suppose that contrary to our assertion, there is a minimal algorithm for $T(\underline{x})(\underline{y})$ which is not essentially bilinear. Because the rationals contain infinitely many square-free integers not equal to ± 1, we can find such an integer a so that after substituting ax_{2n-1-j} for x_j for all $j = n, \ldots, 2n-2$, the induced algorithm is also not essentially bilinear. But this algorithm computes the product in the algebraic extension field $Q[u] / \langle u^n - a \rangle$. By theorem IV-5, this algorithm must be essentially bilinear. This contradiction implies our theorem.

Remark: We refer the reader to [9] for a discussion on transposes of systems of bilinear forms and algorithms for computing these transposes. Theorem IV-6 shows that all minimal division-free algorithms for computing $T(\underline{x})(\underline{y})$ are essentially derivable, via the transpose method, from minimal algorithms for computing products of arbitrary polynomials. The latter have already been classified by Winograd in [13].

Theorem IV-9: The multiplicative complexity of the quaternion product over a real field is 8.

Proof: Denote by $K(\underline{x})$ the matrix of the system of bilinear forms for computing the quaternion product (see example IV-2). By theorems IV-1 and IV-2, $\mu K(\underline{x}) \geq 7$. Suppose that $\mu K(\underline{x}) = 7$.

By Winograd's presentation theorem [12], there exists a quadratic algorithm of degree 7 which computes $K(\underline{x})(\underline{y})$. By theorem IV-3, this algorithm must be bilinear. but Howell has shown that $\bar{\mu} K(\underline{x}) = 8$. This contradiction implies that $\mu K(\underline{x}) \neq 7$. Finally, $\bar{\mu} K(\underline{x}) = 8$ implies $\mu K(\underline{x}) = 8$.

Remark: Porodin exhibits a bilinear algorithm of degree 8 for computing the quaternion product [4].

Bibliography

1. A. Aho, J. Hopcroft, and J. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, Mass. 1974.

2. L. Auslander and R. Tolimieri, Is computing with the finite Fourier Transform pure or applied mathematics? Bulletin (New series) of the American Mathematical Society, Vol. 1, No. 6, November, 1979.

3. L. Auslander and S. Winograd, The multiplicative complexity of certain sets of semilinear forms defined by polynomials, to be published.

4. L. Dolkin, On the complexity of a class of arithmetic computations, Ph.D. thesis, Harvard Univ., 1973.

5. C. M. Fiduccia and Y. Zalcstein, Algebras having linear multiplicative complexities, Journal of the Association for Computing Machinery, Vol. 24, No. 2, April, 1977.

6. T. D. Howell and J. C. Lafon, the complexity of the quaternion product, Rep. TR 75-245, Cornell University, Dept. of Computer Science, 1974.

7. C. M. Rader, Discrete Fourier Transform when the number of data samples is prime, Proc. of IEEE, Vol. 5, No. 6, 1968.

8. A. L. Toom, The complexity of a scheme of functional elements, Soviet Mathematics, Translations of Doklady Akademie, Nauk. S.S.S.R. 4, June, 1963.

9. S. Winograd, Arithmetic complexity of computations, Conference Board of Mathematical Sciences Regional Series in Mathematics, 1978.

10. S. Winograd, On multiplication in algebraic extension fields, Theoretical Computer Science, No. 8, 1979.

11. S. Winograd, On the multiplicative complexity of the Discrete Fourier Transform, Advances in Mathematics, Vol. 32, No. 2, May, 1979.

12. S. Winograd, On the number of multiplications necessary to compute certain functions, Comm. Pure and Applied Mathematics, 23, 1970.

13. S. Winograd, Some bilinear forms whose multiplicative complexity depends on the field of constants, Math. Systems Theory, 10, 1977.