Modular Representations of Polynomials: Hyperdense Coding and Fast Matrix Multiplication

Vince Grolmusz
Department of Computer Science
Eötvös University, Budapest

***Abstract*—A certain modular representation of multilinear polynomials is considered. The modulo 6 representation of polynomial f is just any polynomial $f+6g$. The 1-a-strong representation of f modulo 6 is polynomial $f+2g+3h$, where no two of g,f and h have common monomials.**

Using this representation, some surprising applications are described: it is shown that n homogeneous linear polynomials $x_1,x_2,\ldots,x_n$ can be linearly transformed to $n^{o(1)}$ linear polynomials, such that from these linear polynomials one can get back the 1-a-strong representations of the original ones, also with linear transformations. Probabilistic Memory Cells (PMC's) are also defined here, and it is shown that one can encode n bits into n PMC's, transform n PMC's to $n^{o(1)}$ PMC's (we call this Hyperdense Coding), and one can transform back these $n^{o(1)}$ PMC's to n PMC's, and from these how one can get back the original bits, while from the hyperdense form one could have got back only $n^{o(1)}$ bits. A method is given for converting $n\times n$ matrices to $n^{o(1)}\times n^{o(1)}$ matrices and from these tiny matrices one can retrieve 1-a-strong representations of the original ones, also with linear transformations. Applying PMC's to this case will return the original matrix, and not only the representation.

I. Introduction

Let f be an n-variable, multi-linear polynomial (that is, every variable appears on the power of 0 or 1) with integer coefficients, for example $f(x_1,x_2,x_3)=34x_1x_2+23x_1x_2x_3$. For any positive integer $m>1$, we say that multi-linear polynomial f_1 is a mod m representation of polynomial f, if the corresponding coefficients of the two polynomials are congruent modulo m; for example, $f_1(x_1,x_2,x_3) = 4x_1x_2+3x_1x_2x_3+5x_2$ is a mod 5 representation of the f in the previous example. If we choose a non-prime-power, composite modulus, say $m=6$, then the modulo 6 representation of polynomial f is also a modulo 3 and modulo 2 representation at the same time. This means, that if we examine the properties of the modulo 6 representations of multi-linear polynomials (or, equivalently, multi-linear polynomials over ring Z_6), it is not probable that we get more interesting properties over Z_6 than over fields F_2 or F_3.

Over composite, non-prime-power moduli (say 6), however, we can consider different representations as well. We will define 1-a-strong representations of polynomials formally in the next section, but now it is enough to say that the 1-a-strong representation of multi-linear polynomial f modulo 6 is a polynomial $f+2g+3h$, where no two of f,g and h have common monomials. The last restriction is necessary, since otherwise the constant polynomial 0 would be the 1-a-strong representation modulo 6 of an arbitrary polynomial f, simply because $0=f+3f-4f$.

A. On the motivation

The motivation of the somewhat strange notion of the 1-a-strong representation modulo 2 and 3 comes from the following results:

In 1992 Barrington, Beigel and Rudich [1] showed a degree-$O(\sqrt{n})$ polynomial representing the n-variable OR function modulo 6, while it was known that for all prime moduli that representation needs linear degree in n.

Using this BBR-polynomial, we gave a surprising construction for certain set-systems, falsifying conjectures on their non-existence in [2], and gave explicit Ramsey-graph constructions from these set systems in [2] and in [3].

Building onto these results, in [4] we proved, that a representation modulo 6 (what we call a 0-a-strong representation in the next section) of the elementary symmetric polynomials can be computed dramatically faster than over any prime moduli. This result plays a main rôle in the proofs of the present work.

The motivation of defining the Probabilistic Memory Cells is clearly to give a realistic model of data storage where our hyperdense coding works.

B. Our Results:

Let m be a non-prime power composite constant (that is, it is constant in n, e.g., $m=6$).

Address: Pázmány P. stny. 1/C, H-1117 Budapest, Hungary; E-mail: grolmusz@cs.elte.hu, http://www.cs.elte.hu/∼grolmusz

(a) From the n variables $x_1, x_2, \ldots, x_n$, (each seen as a 1-variable linear function,) we compute $t = n^{o(1)}$ linear functions $z_1, z_2, \ldots, z_t$, and from these t linear functions again n linear functions $x'_1, x'_2, \ldots, x'_n$, such that x'_i is a 1-a-strong representation of linear function (i.e., variable) x_i, for $i = 1, 2, \ldots, n$. Both computations are linear transformations.

(b) We define Probabilistic Memory Cells (PMC's). By an observation of a PMC one can get a constant amount of information. We encode n bits into n PMC's: one bit into one PMC, and we use the first linear transformation in (a) to transform the n PMC's to $t = n^{o(1)}$ PMC's (observing these t PMC's would yield only $O(n^{o(1)})$ bits of information), and then we transform these t PMC's back to n PMC's, also with a linear transformation, and the observation of the resulting n PMC's will yield the original n bits. We call this phenomenon hyperdense coding modulo m.

(c) For any $n \times n$ matrix X with elements from set $\mathbb{Z}_m$, we compute an $n^{o(1)} \times n^{o(1)}$ matrix Z with elements from set $\mathbb{Z}_m$, such that from Z, one can retrieve the 1-a-strong representation of the $n \times n$ matrix X; here both operations (the computing and the retrieval) are simple linear transformations.

(d) For $n \times n$ matrices X and Y, with elements from set $\mathbb{Z}_m$, we compute the 1-a-strong representation of the product matrix XY, with only $n^{o(1)}$ multiplications, significantly improving our earlier result of computing the 1-a-strong representation of the matrix-product with $n^{2+o(1)}$ multiplications [5].

It is clear that by using Probabilistic Memory Cells for storing each entry of the binary matrix X in (c), matrix Z can be stored with $n^{o(1)}$ PMC's, from which we can compute the original $n \times n$ matrix X, by using the second linear transform of (c) and observations of the resulting n^2 PMC's. We call this phenomenon the *dimension defying* property of the 1-a-strong representation.

II. Preliminaries

A. A-strong representations

In [4] we gave the definition of the *a-strong (i.e., alternative-strong) representation* of polynomials. Here we define the *alternative*, and the *0-a-strong* and the *1-a-strong* representations of polynomials. Note that the 0-a-strong representation, defined here, coincides with the a-strong representation of the paper [4].

Note also, that for prime or prime-power moduli, polynomials and their representations (defined below), coincide. This fact also motivates the examination of such representations.

Definition 1 ([5]): Let m be a composite number with prime-factorization $m = p_1^{e_1} p_2^{e_2} \cdots p_\ell^{e_\ell}$. Let $\mathbb{Z}_m$ denote the ring of integers modulo m. Let f be a multi-linear polynomial of n variables over $\mathbb{Z}_m$:

$$f(x_1, x_2, \ldots, x_n) = \sum_{\alpha \in \{0,1,2,\ldots,d\}^n} a_\alpha x^\alpha,$$

where $a_\alpha \in \mathbb{Z}_m$, $x^\alpha = \prod_{i=1}^n x_i^{\alpha_i}$. Then we say that

$$g(x_1, x_2, \ldots, x_n) = \sum_{\alpha \in \{0,1,2,\ldots,d\}^n} b_\alpha x^\alpha, ,$$

is an

- *alternative representation* of f modulo m, if
$$\forall \alpha \in \{0, 1, 2, \ldots, d\}^n \quad \exists j \in \{1, 2, \ldots, \ell\} :$$
$$a_\alpha \equiv b_\alpha \pmod{p_j^{e_j}};$$
- *0-a-strong representation* of f modulo m, if it is an alternative representation, and, furthermore, if for some i, $a_\alpha \not\equiv b_\alpha \pmod{p_i^{e_i}}$, then $b_\alpha \equiv 0 \pmod{p_i^{e_i}}$;
- *1-a-strong representation* of f modulo m, if it is an alternative representation, and, furthermore, if for some i, $a_\alpha \not\equiv b_\alpha \pmod{p_i^{e_i}}$, then $a_\alpha \equiv 0 \pmod{m}$;

In other words, for modulus 6, in the alternative representation, each coefficient is correct either modulo 2 or modulo 3, but not necessarily both.

In the 0-a-strong representation, the 0 coefficients are always correct both modulo 2 and 3, the non-zeroes are allowed to be correct either modulo 2 or 3, and if they are not correct modulo one of them, say 2, then they should be 0 mod 2. That is, coefficient 1 can be represented by 1, 3 or 4, and nothing else.

In the 1-a-strong representation, the non-zero coefficients of f are correct for both moduli in g, but the zero coefficients of f can be non-zero either modulo 2 or modulo 3 in g, but not both.

Remark 2: The 1-a-strong representations of polynomial f can be written in the form:

$$f + p_1^{e_1} g_1 + p_2^{e_2} g_2 + \cdots + p_\ell^{e_\ell} g_\ell,$$

where the g_i have no monomials in common with each other, nor with f.

Example 3: Let $m = 6$, and let $f(x_1, x_2, x_3) = x_1x_2 + x_2x_3 + x_1x_3$, then $g(x_1, x_2, , x_3) = 3x_1x_2 + 4x_2x_3 + x_1x_3$ is a 0-a-strong representation of f modulo 6; $g(x_1, x_2, , x_3) = x_1x_2 + x_2x_3 + x_1x_3 + 3x_1^2 + 4x_2$ is a 1-a-strong representation of f modulo 6; $g(x_1, x_2, , x_3) = 3x_1x_2 + 4x_2x_3 + x_1x_3 + 3x_1^2 + 4x_2$ is an alternative representation modulo 6.

Example 4: Let $m = 6$. Then $0 = xy - 3xy + 2xy$ is **not** a 1-a-strong representation of xy.

B. Previous results for a-strong representations

In [4] we considered elementary symmetric polynomials

$$S_n^k = \sum_{\substack{I\subset\{1,2,\ldots,n\}\\ |I|=k}} \prod_{i\in I} x_i,$$

and proved that for constant k, 0-a-strong representations of elementary symmetric polynomials S_n^k can be computed dramatically faster over non-prime-power composites than over primes.

In [4], the following theorem was proved:

Theorem 5 ([4]): Let the prime factorization of positive integer m be $m = p_1^{e_1} p_2^{e_2} \cdots p_\ell^{e_\ell}$, where $\ell > 1$. Then a degree-2 0-a-strong representation of

$$S_n^2(x,y) = \sum_{\substack{i,j\in\{1,2,\ldots,n\}\\ i\neq j}} x_i y_j, \tag{1}$$

modulo m:

$$\sum_{\substack{i,j\in\{1,2,\ldots,n\}\\ i\neq j}} a_{ij} x_i y_j \tag{2}$$

can be computed as the following product:

$$\sum_{j=1}^{t-1} \left(\sum_{i=1}^n b'_{ij} x_i\right)\left(\sum_{i=1}^n c'_{ij} y_i\right)$$

where $t = \exp(O(\sqrt[\ell]{\log n (\log\log n)^{\ell-1}})) = n^{o(1)}$. Moreover, this representation satisfies that $\forall i \neq j : a_{ij} = a_{ji}$. $\square$

The following result is the basis of our theorems in the present paper.

Theorem 6 ([5]): Let $m = p_1^{e_1} p_2^{e_2} \cdots p_\ell^{e_\ell}$, where $\ell > 1$, and $p_1, p_2, \ldots, p_\ell$ are primes. Then a degree-2, 1-a-strong representation of the dot-product $f(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n) = \sum_{i=1}^n x_i y_i$ can be computed with $t = \exp(O(\sqrt[\ell]{\log n (\log\log n)^{\ell-1}})) = n^{o(1)}$ multiplications of the form

$$\sum_{j=1}^{t} \left(\sum_{i=1}^n b_{ij} x_i\right)\left(\sum_{i=1}^n c_{ij} y_i\right) \tag{3}$$

Proof: Let $g(x,y) = g(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n)$ be the degree-2 polynomial from Theorem 5 which is a 0-a-strong representation of $S_n^2(x,y)$. Then consider polynomial

$$h(x,y) = (x_1+x_2+\ldots+x_n)(y_1+y_2+\ldots+y_n) - g(x,y). \tag{4}$$

In $h(x,y)$, the coefficients of monomials $x_i y_i$ are all 1 modulo m, and the coefficients of monomials $x_i y_j$, for $i \neq j$ are 0 at least for one prime-power divisor of m, and if it is not 0 for some prime divisor, then it is 1. Consequently, by Definition 1, $h(x,y)$ is a 1-a-strong representation of the dot-product $f(x,y)$. ∎

III. Dimension-Defying: Linear Functions

For simplicity, let $m = 6$.

By Theorem 6, a 1-a-strong representation of the dot-product $\sum_{i=1}^n x_i y_i$ can be computed as

$$\sum_{i=1}^n x_i y_i + 3g(x,y) + 2h(x,y) =$$
$$= \sum_{j=1}^{t} \left(\sum_{i=1}^n b_{ij} x_i\right)\left(\sum_{i=1}^n c_{ij} y_i\right) \tag{5}$$

where $b_{ij}, c_{ij} \in \{0,1\}$ and where both g and h have the following form: $\sum_{i\neq j} a_{ij} x_i y_j$, and no term $x_i y_j$ appears in both h and g; and, finally $t = \exp(O(\sqrt{\log n \log\log n})) = n^{o(1)}$. Note that every monomial $x_i y_j$, $i \neq j$ has really a coefficient which is a multiple of 3 or 2, since 1-4=3 modulo 6 and 1-4=3 modulo 6.

Now, let us observe that for each $j = 1, 2, \ldots, t$,

$$z_j = \sum_{i=1}^n b_{ij} x_i \tag{6}$$

is a linear combination of variables x_i.

Let these $t = n^{o(1)}$ linear forms be the encoding of the n 0-1 variables x_i. The decoding is done also from (5): the 1-a-strong representation of x_i can be computed by plugging in

$$y^i = (0, 0, \ldots, \overbrace{1}^{i}, 0, \ldots, 0).$$

Obviously, on the LHS of (5) we get the 1-a-strong representation of x_i, and on the RHS we get a linear combination of the z_j of (6).

By matrix-notation, if x is a length-n vector, and $B = \{b_{ij}\}$ is an $n \times t$ matrix with b_{ij}'s given in (5), and $C = \{c_{ij}\}$ is an $n \times t$ matrix with c_{ij}'s given in (5), then we can write that

$$z = xB, \text{ and } x' = zC^T = xBC^T.$$

Consequently, $x' = xBC^T$ is a length-n vector, such that for $i = 1, 2, \ldots, n$, $x'_i = x_i + 3g_i(x) + 4h_i(x)$ where $g(x)$ and $h(x)$ are integer linear combinations (that is, homogeneous linear functions) of the coordinates of x such that none of them contains x_i and they do not contain the same x_j with non-zero coefficients. The proof of this fact is obvious from (5). It is easy to see that we proved the following Theorem (stating for general m this time):

Theorem 7: For any non-prime-power positive integer m, and positive integer n, there exist effectively computable constant $n \times t$ matrices B and C over $\mathbb{Z}_m$, with $t = n^{o(1)}$, such that for any vector $x = (x_1, x_2, \ldots, x_n)$ with variables as coordinates, the coordinate i of the length-n vector xBC is a 1-a-strong representation of polynomial x_i modulo m, for $i = 1, 2, \ldots, n$.

□

Note, that xB has t coordinates (linear functions), while xBC^T has again n coordinates (linear functions). Note that similar representation is *impossible* with m prime and $t < n$.

For an application of this striking observation we need the definition of Probabilistic Memory Cells.

IV. Probabilistic Memory

The words "probabilistic" and "memory" are rarely mixed well: a probabilistically behaving memory element – typically – is not desirable in any computer. Here we consider 1-0 step functions on the real interval $[0, 1]$, describing some physical object changing its state from 1 to 0 in a random point of the interval $[0, 1]$. We assume that the distribution of this point is uniform in the the real interval $[0, 1]$. We also assume that the distribution of these random points are independent. The randomness will assure us that with probability 1, no two different functions have the state-change at the same moment. We intend to use integer linear combinations of these functions for dense data storage. The formal definition is as follows:

Definition 8: An m-Probabilistic Memory Cell (m-PMC for short) is a step-function $\rho : [0, 1] \to \mathbb{Z}_m$, such that $\rho(i)^{-1}$, for $i = 0, 1, \ldots, m - 1$, is a finite union of subintervals of the interval $[0, 1]$. $a \in [0, 1]$ is a step-point of ρ if $\lim_{+a} \rho \neq \lim_{-a} \rho$. The step-value in step-point a is equal to $\lim_{+a} \rho - \lim_{-a} \rho$ modulo m. An m-PMC is simple, if there exists an $a \in [0, 1]$ such that $\rho^{-1}(1) = [0, a]$, and $\rho^{-1}(0) = (a, 1]$. A collection of m-PMC's $\rho_1, \rho_2, \ldots, \rho_n$ is called a proper-(n, m)-PMC, if

- every ρ_i is a simple m-PMC, and
- for all $i \neq j$, the step-points of ρ_i and ρ_j differ.

The observation operator $\mathcal{O}(\rho)$ returns the (un-ordered) set of step-values, modulo m, in all the step-points of m-PMC ρ, that is, $\mathcal{O}(\rho) \subset \{0, 1, \ldots, m - 1\}$, for any m-PMC ρ.

Note, that the set of the m-PMC's form a module over the integer ring $\mathbb{Z}_m$. Note also, that the set of step-points of an integer linear combination of several m-PMC's is a subset of the union of the step-points of the individual PMC's.

Fact. If the step-points are distributed uniformly and independently in each of the n simple m-PMC's, then their collection will form a proper-(n, m)-PMC with probability 1.

This is the reason that the word "Probabilistic" appears in Definition 8.

Example 9: In Figure 1, the linear combination of simple PMC's ρ and ξ, $2\rho + 3\xi$ is also a PMC, and $\mathcal{O}(2\rho + 3\xi) = \{-2, -3\} = \{4, 3\}$, with $m = 6$.

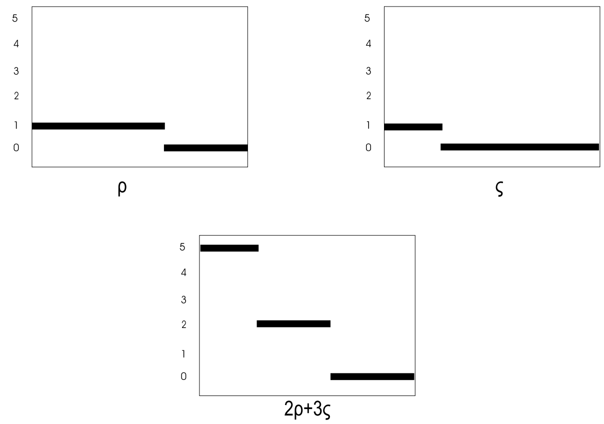

Fig. 1. Linear combination of PMC's ρ and ξ.

Example 10: The sum of the members of the proper-$(n, 6)$-PMC $\rho_1, \rho_2, \ldots, \rho_n$ is also a 6-PMC $\xi = \sum_{i=1}^{n} \rho_i$, and clearly, $\mathcal{O}(\xi) = \{5\}$.

Motivation: We defined PMC's in order to get applications for our main results in this work. The notion is somewhat analogous with quantum computation: we imagine a PMC as a physical object with m inner states changing in time (where the time corresponds to the interval [0,1]), but we can observe only the change of that inner states (and not the identities of the states). The motivation of the *observation operator* is also comes from the quantum computations. For example, we can observe the wave-lengths (or spectrum) of the photons emitted by that physical object in the state-change. Note, that during an observation we are not measuring the multiplicity, the timing, or any pattern of the change, just the set of differences of the states, modulo m. Consequently, observing any PMC returns a subset of set $\{0, 1, \ldots, m - 1\}$, that is, for constant m we get information, encodeable with exactly m bits.

V. Hyperdense Coding

Let $h_1, h_2, \ldots, h_n$ be n bits. Let $\rho_1, \rho_2, \ldots, \rho_n$ be a proper $(n, 6$-PMC. Now define $x_i = h_i \rho_i$, for $i = 1, 2, \ldots, n$, and let $x = (x_1, x_2, \ldots, x_n)$. Clearly, the x_i's are also PMC's. Now, let us use matrices B and C from Theorem 7. Let $z = xB$ be a vector, and each of the $t = n^{o(1)}$ coordinates of it is a PMC. Note, that observing any coordinate of z yields only $O(1)$ bits of information, $O(n^{o(1)})$ in total. However, if we do not observe the coordinates of z, but instead of that we apply the linear transform C^T to it, then we would get back the 1-a-strong representation of polynomials x_i in each coordinate of $zC^T = xBC^T$ in case of variables as $x_i's$, that is: $x_i' = x_i + 3g_i(x) + 2h_i(x)$. But now we have PMC's instead of linear functions.

What happens if we observe x_i'? Clearly, for $m = 6$,

$$h_i = 1 \iff 5 \in \mathcal{O}(x_i'),$$

since in case of $h_i = 0$ every step-value is a multiple of 2 or 3. That means that by observing the n PMC's in the coordinates of $zC^T = xBC^T$, we get back the n bits of $h_1, h_2, \ldots, h_n$.

Note, that the t coordinates of z also contained the information on the n input-bits, but with observations we were not able to recover it. We call z the hyperdense coding of bits $h_1, h_2, \ldots, h_n$. Consequently, we have proved (again stating for general m):

Theorem 11: For any non-prime-power positive integer m, and positive integer n, there exist effectively computable constant $n \times t$ matrices B and C over $\mathbb{Z}_m$, with $t = n^{o(1)}$, such that for any bit-sequence $h_1, h_2, \ldots, h_n$ can be encoded into n m-PMC's $x = (x_1, x_2, \ldots, x_n)$, and these m-PMC's can be linearly transformed into t m-PMC's $z = xB$, and these PMC's can be linearly transformed to n PMC's $x' = zC^T = xBC^T$, such that the observation of the PMC's in the coordinates of x' yields the original values of $h_1, h_2, \ldots, h_n$.
□

Note, that in a completely different quantum-mechanical model, Bennett and Wiesner [6], using Einstein-Podolski-Rosen entangled pairs, showed that n classic bits can be encoded by $\lceil n/2 \rceil$ quantum bits. They called their result superdense coding. Since our method yields significantly more dense coding (although in a different model), that is the reason of calling it "hyperdense coding".

VI. Matrix Compression

Definition 12: Let $X = \{x_{ij}\}$ be an $n \times n$ matrix with one-variable homogeneous linear functions (that is, $x_{ij}'s$) as entries. Then $Y = \{y_{ij}\}$ is a 1-a-strong representation of the matrix X modulo m if for $1 \le i, j \le n$, the polynomial y_{ij} of n^2 variables $\{x_{uv}\}$ is a 1-a-strong representation of polynomial x_{ij} modulo m.

If we plug in column-vectors instead of just variables in the homogeneous linear forms of Theorem 7, then we will get linear combinations of the column-vectors. Consequently, we proved the following implication of Theorem 6:

Theorem 13: For any non-prime-power positive integer m, and positive integer n, there exist effectively computable constant $n \times t$ matrices B and C, such that for any $n \times n$ matrix $X = \{x_{ij}\}$, XBC^T is a 1-a-strong representation of matrix X modulo m, where $t = n^{o(1)}$.

The dimension-defying implication of Theorem 13 is that X is an $n \times n$ matrix, XB is an $n \times n^{o(1)}$ matrix, and XBC^T is again an $n \times n$ matrix.

Transposing the matrices in Theorem 13, we get:

Corollary 14: With the notations of Theorem 13, CB^TX is a 1-a-strong representation of matrix X modulo m, where $t = n^{o(1)}$.

Our main result in this section is the following implication of Corollary 14 and Theorem 13:

Theorem 15: For any non-prime-power $m > 1$, there exist effectively computable constant $n \times t$ matrices B and C, such that for any matrix $X = \{x_{ij}\}$, B^TXB is a $t \times t$ matrix, where $t = n^{o(1)}$, and matrix CB^TXBC^T is a 1-a-strong representation of matrix X modulo m.

The dimension-defying implication of Theorem 15 is that from the $n \times n$ matrix X with simple linear transformations we make the tiny $n^{o(1)} \times n^{o(1)}$ matrix B^TXB, and from this, again with simple linear transformations, $n \times n$ matrix CB^TXBC^T, where it is a 1-a-strong representation of matrix X modulo m.

Similarly as in Section V, where we changed our result from linear functions to numbers with using PMC's, now we repeat the same method in the following Theorem:

Theorem 16: For any non-prime-power $m > 1$, and for any positive integer n, there exist effectively computable constant $n \times t$ matrices B and C, such that any $H = \{h_{ij}\}$ a 0-1 $n \times n$ matrix can be encoded into an $n \times n$ matrix $X = \{x_{ij}\}$ with n^2 PMC's as entries, applying two linear transforms to this matrix we get an $t \times t$ matrix B^TXB which contains t^2 m-PMC's, and applying two further linear transforms, we get the $n \times n$ matrix CB^TXBC^T, with n^2 PMC's as entries, whose observation returns the original 0-1 values of the matrix H.

Proof: Let $\rho_{11}, \rho_{12}, \ldots, \rho_{nn}$ be a proper (n^2, m)-PMC, and let us define the $x_{ij} = h_{ij}\rho_{ij}$. Clearly, the entries of CB^TXBC^T are 1-a-strong representations of x_{ij}'s, so by observing its (i, j) entry, x_{ij}' the following holds:

$$h_{ij} = 1 \iff m - 1 \in \mathcal{O}(x_{ij}').$$

■

VII. Matrix multiplication

The matrix multiplication is a basic operation in mathematics in applications in almost every branch of mathematics itself, and also in the science and engineering in general. An important problem is finding algorithms for fast matrix multiplication. The natural algorithm for computing the product of two $n \times n$ matrices uses n^3 multiplications. The first, surprising algorithm for fast matrix multiplication was the recursive method of Strassen [7], with $O(n^{2.81})$ multiplications. After a long line of results, the best known algorithm today was given by Coppersmith and Winograd [8], requiring only $O(n^{2.376})$ multiplications. Some of these methods can be applied successfully in practice for the multiplication of large matrices [9]. (For the introduction in algebraic complexity see the book by Bürgisser, Clausen and Shokrollahi [10]).

The best lower bounds for the number of needed multiplications are between $2.5n^2$ and $3n^2$, depending on the underlying fields (see [11], [12], [13]). A result of Raz [14] gives an $\Omega(n^2 \log n)$ lower bound for the number of multiplications, if only bounded scalar multipliers can be used in the algorithm.

In [5] we gave an algorithm with $n^{2+o(1)}$ multiplications for computing the 1-a-strong representation of the matrix product modulo non-prime power composite numbers (e.g., 6). The algorithm was an application of a method of computing a representation of the dot-product of two length-n vectors with only $n^{o(1)}$ multiplications.

In the present work, we significantly improve the results of [5], we give an algorithm for computing the 1-a-strong representation of the product of two $n \times n$ matrices with only $n^{o(1)}$ multiplications.

Definition 17: Let $X = \{x_{ij}\}$ and $Y = \{y_{ij}\}$ be two $n \times n$ matrices with $2n^2$-variable homogeneous linear functions (that is, $x'_{ij}s$ and $y'_{ij}s$ as entries). We say that matrix $V = \{v_{ij}\}$ is a 1-a-strong representation of the product-matrix XY, if for $1 \le i, j \le n$, v_{ij}, as a $2n^2$-variable polynomial, is a 1-a-strong representation of polynomial $\sum_{k=1}^n x_{ik}y_{kj}$ modulo m.

We need to define a sort of generalization of the matrix-product:

Definition 18: $f : R^{2n} \to R$ is a homogeneous bilinear function over ring R if

$$f(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n) = \sum_{1 \le i,j \le n} a_{ij}x_iy_j$$

for some $a_{ij} \in R$. Let $U = \{u_{ij}\}$ be an $u \times n$ matrix over ring R, and let $V = \{v_{k\ell}\}$ be an $n \times v$ matrix over R. Then $U(f)V$ denotes the $u \times v$ matrix over R with entries $w_{i\ell}$, where

$$w_{i\ell} = f(u_{i1}, u_{i2}, \ldots, u_{in}, v_{1\ell}, v_{2\ell}, \ldots, v_{n\ell}).$$

Note, that if f is the dot-product, then $U(f)V$ is just the simple matrix-product.

First we need a simple lemma, stating that the associativity of the matrix multiplication is satisfied also for the "strange" matrix-multiplication defined in Definition 18:

Lemma 19: Let

$$f(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n) = \sum_{1 \le i,j \le n} a_{ij}x_iy_j$$

and let

$$g(x_1, x_2, \ldots, x_v, y_1, y_2, \ldots, y_v) = \sum_{1 \le i,j \le v} b_{ij}x_iy_j$$

be homogeneous bilinear functions over the ring R. Let $U = \{u_{ij}\}$ be an $u \times n$ matrix, and let $V = \{v_{k\ell}\}$ be an $n \times v$ matrix, and $W = \{w_{ij}\}$ be a $v \times w$ matrix over R, where u, n, w are positive integers. Then $(U(f)V)\,(g)W = U(f)\,(V(g)W)$, that is, the "strange" matrix-multiplication, given in Definition 18, is associative.

Proof 1: The proof is obvious from the homogeneous bilinearity of f and g.

Proof 2: We also give a more detailed proof for the lemma. The entry of row i and column k of matrix $U(f)V$ can be written as

$$\sum_{z,t} a_{zt}u_{iz}v_{tk}.$$

Consequently, the entry in row i and column r of $(U(f)V)(g)W$ is

$$\sum_{k,\ell} b_{k\ell} \left(\sum_{z,t} a_{zt}u_{iz}v_{tk} \right) w_{\ell r}.$$

On the other hand, entry (t, r) in $V(g)W$ is

$$\sum_{k,\ell} b_{k\ell}v_{tk}w_{\ell r},$$

and entry (i, r) in $U(f)(V(g)W)$ is

$$\sum_{z,t} a_{zt}u_{iz} \sum_{k,\ell} b_{k\ell}v_{tk}w_{\ell r},$$

and this proves our statement.□

Now we are in the position of stating and proving our main theorem for matrix multiplications:

Theorem 20: Let X and Y two $n \times n$ matrices, and let $m > 1$ be a non-prime-power integer. Then the 1-a-strong representation of the matrix-product XY can be computed with $t^3 = n^{o(1)}$ non-scalar multiplications.

Proof: We use Theorem 13 and Corollary 14. Let us consider $t \times n$ matrix B^TX and $t \times n$ matrix YB; these matrices can be computed without any multiplications from X and Y (we do not count multiplications by constants). Let $h(x, y)$ be the homogeneous bi-linear function (4). Then $B^TX(h)YB$ can be computed with $n^{o(1)}$ multiplications (Note, that because of Lemma 19, the associativity holds). Now compute matrix $CB^TX(f)YBC^T = (CB^TX)(f)(YBC^T)$ without any further (non-constant) multiplication. By Theorem 13 and Corollary 14, CB^TX and YBC^T is a 1-a-strong representations of X and Y respectively, and they are the linear combinations of the rows of X and columns of Y, respectively. Consequently, using Theorem 6, $CB^TX(f)YBC^T$ is a 1-a-strong representation of XY. ■

VIII. Open Problems

It is a great challenge to prove or disprove the computability of the matrix product with only $n^{2+o(1)}$ multiplication. We pose here the following problem:

By using our computation of the 1-a-strong representation of the matrix product upto $O(n^2)$ times (even for different matrices), compute the (exact, not a representation) matrix product of two $n \times n$ matrices.

Solution for this open problem would yield a matrix-multiplication algorithm with only $O(n^{2+o(1)})$ multiplications.

It is also a challenge to give a non-trivial lower bound to the number t in Theorem 5.

Acknowledgement. The author acknowledges the partial support of OTKA grant NK67867.

Note. A Maple(tm) worksheet with numerical examples of matrices B and C can be downloaded from the address:
http://www.cs.elte.hu/∼grolmusz/supporting.mws

REFERENCES

[1] D. A. M. Barrington, R. Beigel, and S. Rudich, "Representing Boolean functions as polynomials modulo composite numbers," *Comput. Complexity*, vol. 4, pp. 367–382, 1994, appeared also in *Proc. 24th Ann. ACM Symp. Theor. Comput.*, 1992.

[2] V. Grolmusz, "On set systems with restricted intersections modulo a composite number," in *Lecture Notes in Computer Science Vol. 1276*, August 1997, pp. 82–90, journal version to appear in Combinatorica.

[3] ——, "Superpolynomial size set-systems with restricted intersections mod 6 and explicit Ramsey graphs," *Combinatorica*, vol. 20, pp. 73–88, 2000.

[4] ——, "Computing elementary symmetric polynomials with a subpolynomial number of multiplications," *SIAM Journal on Computing*, vol. 32, no. 6, pp. 1475–1487, 2003.

[5] ——, "Near quadratic matrix multiplication modulo composites," ECCC, Tech. Rep. TR03-001, 2003, http://eccc.hpi-web.de/eccc-reports/2003/TR03-001/index.html#C01.

[6] C. Bennett and S. Wiesner, "Communication via one- and two particle operators on Einstein-Podolski-Rosen states," *Phys. Rev. Lett.*, vol. 69, pp. 2881–2884, 1992.

[7] V. Strassen, "Gaussian elimination is not optimal," *Numerische Mathematik*, vol. 13, pp. 354–356, 1969.

[8] D. Coppersmith and S. Winograd, "Matrix multiplication via arithmetic progressions," *J. Symbolic Comput.*, vol. 9, no. 3, pp. 251–280, 1990.

[9] D. H. Bailey, "Extra high speed matrix multiplication on the Cray-2," *SIAM J. Sci. Statist. Comput.*, vol. 9, no. 3, pp. 603–607, 1988.

[10] P. Bürgisser, M. Clausen, and M. Shokrollahi, *Algebraic Complexity Theory*. Springer, 1997.

[11] M. Bläser, "A $\frac{5}{2}n^2$-lower bound for the rank of $n \times n$-matrix multiplication over arbitrary fields," in *40th Annual Symposium on Foundations of Computer Science (New York, 1999)*. IEEE Computer Soc., Los Alamitos, CA, 1999, pp. 45–50.

[12] N. H. Bshouty, "A lower bound for matrix multiplication," *SIAM J. Comput.*, vol. 18, no. 4, pp. 759–765, 1989.

[13] A. Shpilka, "Lower bounds for matrix product," in *IEEE Symposium on Foundations of Computer Science*, 2001, pp. 358–367. [Online]. Available: citeseer.nj.nec.com/482013.html

[14] R. Raz, "On the complexity of matrix product," in *Proceedings of the thirty-fourth annual ACM symposium on Theory of computing*. ACM Press, 2002.