SINGULARITY OF SPARSE CIRCULANT MATRICES IS NP-COMPLETE

ILIA TOLI

ABSTRACT. It is shown by Karp reduction that deciding the singularity of $(2^n - 1) \times (2^n - 1)$ sparse circulant matrices (SC problem) is NP-complete. We can write them only implicitly, by indicating values of the $2 + n(n+1)/2$ eventually nonzero entries of the first row and can make all matrix operations with them. The positions are $0, 1, 2^i + 2^j$. The complexity parameter is $n$. Mulmuley’s work on the rank of matrices [5] makes SC stand alone in a list of 3,000 and growing NP-complete problems.

1. INTRODUCTION

It is proved that calculating the singularity of sparse circulant matrices (SC) is NP-complete. ”Sparse“ in our context means that only entries $0, 1, 2^i + 2^j$ of the first row are eventually nonzero. The other rows are defined from the first one. In the next section we define 3 problems MQ, PQ, SC. MQ is a well established one. Our polynomial time reductions are $MQ \Rightarrow PQ \Rightarrow SC$. The reduction $MQ \Rightarrow PQ$ has already been established by Kipnis and Shamir ([4], Section 3.) We give another proof in Appendix [B]. In the main body of this paper we deal with $PQ \Rightarrow SC$.

Most of the known 3000 and counting NP-complete problems are rewording of each other, iterative, or just variations of exhaustive search. Mulmuley [5] has proven that calculating the rank of a matrix is none of the above, so SC holds a special place among all NP-complete problems. We can make nearly any (counterexamples exist) assumption on the shape of $n$ and the problem remains NP-complete. The commutative ring $(M_k, +, \cdot)$ of $k \times k$ matrices over a finite field is mathematically exceptionally rich. Matrices are very sparse, circulant, symmetric. We believe that SC has the potential to simplify the Complexity Zoo.

To the best of our knowledge, no previous work has been done in this direction. There exist very few NP-problems mentioning matrices, like MinRank, MaxRank, Sing [1] but they are of a completely different nature. In MinRank e.g., we are given a matrix with part of the entries from a ring and part variables that assume values in that ring. The question is to find the values of the variables for which the matrix has

Key words and phrases. NP-complete, circulant, matrix, singular, determinant, polynomial, equation.
the least rank. Here we are given all the entries of the matrix and ask whether the determinant is 0.

In the third section we prove that SC is NP-hard. In the fourth we provide SC with a certificate for yes answers to any instance. This completes the NP-completeness proof. In fifth and sixth sections we give considerations on the present state of things and possible paths to follow. In appendix A we give an illustration of the algorithm of the fourth section. In appendix B we give another proof of \( MQ \Rightarrow PQ \).

2. Statement of the Problems

Let’s start by stating the two problems, one novel and one classic. We solve the classic NP-complete problem in polynomial time using as a subroutine the novel problem, and this proves the NP-hardness of the novel problem (Karp reduction.)

2.1. MQ (Multivariate Quadratic).

Definition 2.1. A term is a monic monomial.

2.1.1. Instance. Polynomials \( P_i[x_1, x_2, \ldots, x_n] \), \( 1 \leq i \leq m \), from the ring \( \mathbb{F}_2[x_1, x_2, \ldots, x_n] \) of multivariate polynomials with coefficient in the field \( \mathbb{Z}/2\mathbb{Z} \).

2.1.2. Question. Do there exist \( u_1, u_2, \ldots, u_n \in \{0,1\} \) such that, for \( 1 \leq i \leq m \), \( P_i[u_1, u_2, \ldots, u_n] = 0 \)?

2.1.3. Comment. Remains NP-complete even if none of the polynomials has a term involving more than 2 variables, that is, all polynomials are quadratic. Easily solved in polynomial time if no term involves more than 1 variable, that is, all polynomials are linear, or if there is just one polynomial. Variant in which the \( u_i \) are allowed to range over the algebraic closure of \( \mathbb{F}_2 \) is NP-hard, even if no term involves more than 2 variables. It is easy to prove that the problem remains NP-complete for all characteristics and if \( m, n \) are polynomially related. By this case we can always assume that \( m = n \). If \( m > n \), we can assume that \( (m - n) \) variables are there and just don’t appear. If \( n > m \) we can count the last equation \( (n - m) \) times.

2.2. PQ (Pseudoquadratic).

Definition 2.2. A polynomial \( P(x) \) as below with coefficients from some field \( \mathbb{F}_q \) is called pseudoquadratic.

\[
P(x) = \sum_{i,j=0}^{n-1} a_{ij}x^{2^i+2^j} + \sum_{i=0}^{n-1} b_ix^{2^i} + c
\]

In characteristic 2, \( P(x) \) has \( 2 + n(n + 1)/2 \) terms. It is rather sparse and writable in \( O(n^2) \) space. The degree of \( P(x) \) is at most, and generally equal to, \( (2^{n-1} + 2^{n-2}) \).
2.2.1. Instance. ([4], Section 3) A polynomial \( P(x) \in \mathbb{F}_{2^n}[x] \) over the field \( \mathbb{F}_{2^n} \).

2.2.2. Question. Does there exist \( u \in \mathbb{F}_{2^n} \) such that \( P(u) = 0 \)?

2.2.3. Comments. Transformation from MQ ([4], Section 3). Remains NP-complete for a variety of generalizations that fall out of the scope of this paper. Not known to be NP-hard or NP-complete for \( P(x) \) having just 3 terms. Trivially solved in polynomial time if it has 2 terms. It’s trivial to bring MQ to PQ to MQ in polynomial time [4].

2.3. SC (Sparse Circulant).

Definition 2.3. A \( n \times n \) circulant matrix is a matrix whose row number \( k \) is obtained by left shifting the zeroth row \( (a_{00}, a_{01}, a_{02}, \ldots, a_{0,(n-1)}) \) by \( k \) positions. That is, \( a_{i,j} = a_{0,(i+j) \mod n} \).

The other equivalent definition is by shifting right. This one has the nice property that the matrix is symmetric. Two more equivalent definitions by columns.

Definition 2.4. Here a sparse circulant matrix is a \( (2^n - 1) \times (2^n - 1) \) circulant matrix with \( 2 + n(n + 1)/2 \) nonzero entries in the first row, located in the positions 0, 1, \( 2^i + 2^j \).

2.3.1. Instance. A \( (2^n - 1) \times (2^n - 1) \) sparse circulant matrix with entries from the finite field \( \mathbb{F}_{2^n} \).

2.3.2. Question. Is the matrix singular?

2.3.3. Comment. Evidently, it takes exponential space in \( n \) (the complexity parameter) to write the matrix explicitly. We assume writing it implicitly, by giving the nonzero entries of the first row. This is no hassle for performing the necessary matrix operations.

It is not known whether the problem remains NP-complete if the nonzero entries are just 3. Easily solved in polynomial time if the nonzero entries are 2.

3. PQ implies SC, or SC is NP-hard

We take it from Kipnis and Shamir ([4], Section 3) that PQ is NP-complete. We bring PQ into SC in polynomial time by Karp reduction, that is, we’ll solve PQ in polynomial time using SC as a subroutine.

We have

\[
\{ P(x) \text{ has roots in } \mathbb{F}_{2^n} \} \iff \{ \text{Res}(P(x), x^{2^n} - x) = 0 \}
\]

The Sylvester matrix associated with resultant \( \text{Res}(P(x), x^{2^n} - x) \) is not circulant at all, we transform it into one with the same singularity.
3.1. From Sylvester to circulant matrix.

**Definition 3.1.** Let \( \mathbf{c} = (c_0, c_1, c_2, \ldots, c_{m-1})^t \) be a column vector and \( f \neq 0 \) a scalar. \( Z_{f,m,n}(c) = (z_{ij}) \) is called an \( f \)-circulant \( m \times n \) matrix if 
\[
z_{ij} = c_{i-j \mod m} \quad \text{for} \quad i \geq j \quad \text{and} \quad z_{ij} = f c_{i-j \mod m} \quad \text{for} \quad i < j.
\]

For \( f = 1 \) it is circulant and for \( f = -1 \) anticirculant.

For illustration of our steps, take \( n = 3, \mathbb{F}_{2^n} = \mathbb{F}_2 \) and a polynomial of degree \( k < 2^n \) with coefficients in it, \( p(x) = a_5 x^5 + a_4 x^4 + a_3 x^3 + a_2 x^2 + a_1 x + a_0 \), not necessarily sparse. \( \text{Res}(x^{2^n} - x, p(x)) \) is the determinant of the \( (2^n + k) \times (2^n + k) \) Sylvester matrix in (4).

\[
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
\end{pmatrix}
\]

The case \( a_0 = 0 \) is trivial, assume \( a_0 \neq 0 \). The last column has only one nonzero entry. The singularity does not change after erasing the last column and row.

In (3) add column \( i \) to column \( (2^n + i) \) for \( 1 \leq i \leq k \), and get (4).

\[
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\end{pmatrix}
\]

In the first \( k \) rows of (4), there is only one nonzero entry per row. Singularity does not change after erasing the first \( k \) rows and columns. We’re left to find the singularity of the \((2^n - 1) \times (2^n - 1)\) circulant...
matrix in the right down square of (4). This proves that SC is NP-hard. In order to finish the proof we must prove that given a yes instance of SC (the determinant is 0) we can provide a proof, verifiable in polynomial time.

4. The yes Answer is Verifiable in Polynomial Time, or SC is NP-complete

To a system $S$ of $n$ quadratic equations in $n$ variables we have associated the corresponding pseudoquadratic polynomial $P$, and to it the corresponding sparse circulant matrix $M$ (4). We’ll write $P_S(x), M_P$ and so on for the associated objects.

$$\text{(5) } (\det M = 0) \iff (P_M \text{ has solutions}) \iff (S_M \text{ has solutions})$$

If we have a subroutine that finds out that $\det M = 0$, we are able to recover a solution (all solutions if their number is polynomially bounded) of the associated polynomial $P_M(x)$ and of the associated system $S_M$. Each of these solutions works as a certificate: "$\det M = 0$ because $P_M(x_0) = 0$". A solution to $P_M(x)$ can be verified in polynomial time. Same is true for a solution of $S_M$. We’ll use $S_M$ in the generation of the certificate because it’s technically easier.

It seems a little bit odd to work with $M$ and then for the proof of singularity regress to the problems we started from, but why not? It might be easier to guess the singularity of $M$ rather than guess whether $S_M$ has solutions, and then return to $S_M$ for the certificate. Next we give an algorithm for recovering all solutions of $S_M$ in polynomial time using as a subroutine the singularity of $M$ and ignoring the complexity of $M$.

4.1. Producing a certificate. All we’re left to do is produce a solution to $S_M$ given $\det M = 0$. Recall from (5) that $S_M$ has a solution iff $\det M = 0$.

By an argument of padding with zeros accordingly as in appendix B, we can assume to have as many equations as variables in all systems we’ll deal with, though the number of variables decreases by 1 per iteration and the number of equations typically remains constant.

We build a sparse binary tree whose nodes are systems derived from $S_M$. In this section drop the index $M$ for simplicity. The empty string is $\lambda$. The $k$-bit string $i = i_1i_2i_3\ldots i_k$ is the index of the system $S^k_i$ obtained from $S$ by substituting $x_{i_j} = i_j$. The children of $S^{k-1}_i$ are $S^k_{i0}$ and $S^k_{i1}$. The depth of $S^k_i$ is $k$. The $ij$ means "juxtapose strings $i$ and $j".$

Every yes node in Algorithm (4.1) will lead to at least one solution. It might bifurcate but cannot extinguish. If there are $s$ solutions, in any depth $k$ we can’t have more than $s$ yes-es. In the depth $(k + 1)$ we
Algorithm 1 Solving MQ using SC as a subroutine

\[ k = 0; S_0^0 = S; \] check whether \( |M_{S_0^0}| \) is 0, yes or no ;

while \( k < n \) do
  for every yes node \( S^k \) do
    write \( S^{k+1}_i \), \( S^{k+1}_{i+1} \), comment: construct the two children of \( S^k \)
    check whether \( |M_{S^{k+1}_i}| \) and \( |M_{S^{k+1}_{i+1}}| \) are 0, yes or no ;
  od;
  \( k += 1; \)
od:
return the indices of the yes nodes ;

can’t have more than 2s nodes altogether. This yields the non-sharp upper bound for the number of nodes, 2ns.

Let’s assume that writing down \( S \) takes \( O(n^3) \) space. So, making one substitution is \( O(n^3) \). We need \( 2ns \) substitutions and the complexity becomes \( O(sn^4) \). At every substitution we have to check the singularity of the associated matrix. If \( M \) is the complexity of checking once, \( O(sn^4 + snM) \) is the overall complexity of the algorithm. We found all the solutions in polynomial time. This ends the proof of NP-completeness.

5. Conclusions

By the argument of padding with zeros as in appendix B we can make nearly every assumption about the shape of \( n \) and the problem remains NP-complete. Experimentally our determinants do behave differently for different shapes of \( n \) and this might be relevant. We’ve not yet been able to take advantage of it.

There are many ways to decide the singularity of sparse circulant matrices, the most promising of them all being Mulmuly’s [5]. Or we can check whether are there any zero eigenvalues. Calculating only one of them is rather inexpensive. There are \((2^n - 1)\) eigenvalues, checking them all is once exponential in \( n \). We may check the number of linearly independent rows or columns. In general it comes down to Gaussian elimination, which here is exponential in \( n \). We haven’t yet taken any advantage of the special sparseness of our matrices and of other structure in them. Because \((2^n - 1)\) is odd, there always exist the \((2^n - 1)\)th primitive roots of unity. Fast Fourier Transform methods apply. All these constructions work for all characteristics, with few adaptations.

It was proved that the problem of deciding the singularity of sparse circulant matrices is NP-complete. This is the first NP-complete problem of the kind. An even more interesting feature is their potential for further improvements of complexity.
None of the algorithms mentioned here for calculating the singularity
- is novel or
- takes advantage of
  - the sparsity,
  - the being circulant and symmetric,
  - the fact that the ring of circulant matrices is commutative,
  - the shape of $n$,
  - deciding the singularity, as opposed to the determinant, rank or eigenvalues.

Further exploration of the topic is needed to clarify how much the complexity can be reduced with specially designed algorithms that take advantage from the very special structure, and whether that has any theoretical impact.

**References**

[1] Jonathan F. Buss, Gudmund S. Frandsen, and Jeffrey O. Shallit. The computational complexity of some problems of linear algebra, 1997.

[2] A. S. Fraenkel and Y. Yesha. Complexity of solving algebraic equations. *Inf. Process. Lett.*, 10(4/5):178–179, 1980.

[3] G.-M. Greuel, G. Pfister, and H. Schönemann. SINGULAR 3.1.0 — A computer algebra system for polynomial computations, 2009. [http://www.singular.uni-kl.de](http://www.singular.uni-kl.de)

[4] Aviad Kipnis and Adi Shamir. Cryptanalysis of the HFE public key cryptosystem. [http://www.springerlink.com/content/hemlkn5e1wmch4b6/](http://www.springerlink.com/content/hemlkn5e1wmch4b6/)

[5] Ketan Mulmuley. A fast parallel algorithm to compute the rank of a matrix over an arbitrary field. *Combinatorica*, 7(1):101–104, 1987.

**Appendix A. Example of Solving MQ in Polynomial Time**

**Given SC**

For illustration of the Algorithm 4.1 only, let’s have the trivial system $S$ of 26 equations in 7 variables, which was generated in Singular [3].

\[
\begin{align*}
    t_6 t_7 & \quad t_2 t_6 + t_6 + t_2 + 1 \\
    t_5 t_7 + t_7 & \quad t_1 t_6 + t_6 + t_1 + 1 \\
    t_4 t_7 & \quad t_5 + t_6 + 1 \\
    t_3 t_7 + t_7 & \quad t_4 t_5 \\
    t_2 t_7 + t_7 & \quad t_3 t_5 + t_5 \\
    t_1 t_7 + t_7 & \quad t_2 t_5 + t_6 + 1 \\
    t_5 t_6 + t_6 + t_5 + 1 & \quad t_1 t_5 + t_6 + 1 \\
    t_4 t_6 + t_4 & \quad t_4 \\
    t_3 t_6 + t_3 + t_6 + 1 & \quad t_3 t_4 + t_4
\end{align*}
\]

The execution of Algorithm 4.1 gives the tree in Figure ???. The yes nodes are marked with * and are easy to guess from the graph anyway. We find the solutions 0010010, 1110100 and 1110101 over $\mathbb{F}_2$. 

Some solutions are multiple but it doesn’t matter in this argument. The complete tree has $2^7 = 128$ leaves, here we have only 14.

APPENDIX B. THE TRANSFORMATION FROM MQ TO PQ

Let’s have a system $S$ of $m$ equations in $n$ variables. By giving to all variables $x_1, x_2, x_3, \ldots x_n$ all possible values we define a function $(\mathbb{F}_2)^n \rightarrow (\mathbb{F}_2)^m$. After eventually padding in the end either one of the strings $n, m$ bit long, we can assume that $m = n$. The old elements of $(\mathbb{F}_2)^m$ identify canonically with the ones that end with $(n - m)$ zeros. If $n > m$, we can keep the function as it is, sending nothing to the new elements. If $n < m$ we add $(m - n)$ variables and no equations for them. Canonically the new elements go to the $(\mathbb{F}_2)^m$ element where goes the element to whom they’re extensions.

So, the system $S$ defines a function $(\mathbb{F}_2)^n \rightarrow (\mathbb{F}_2)^n$ which induces a function $\mathbb{F}_{2^n} \rightarrow \mathbb{F}_{2^n}$. Every function in a finite field $\mathbb{F}_{2^n}$ is alternatively defined by a polynomial from $\mathbb{F}_{2^n}[x]$. If it has $t$ monomials to which we know the terms but not the coefficients and $t$ is small, we need only $t$ evaluations to get it by interpolation.

We have put in one-to-one correspondence each system of quadratic equations with a univariate polynomial with the property that every string that is a solution to the system is also a solution to the univariate polynomial, first considered as element of $(\mathbb{F}_2)^n$ and then as element of $\mathbb{F}_{2^n}$.

In this shift from multivariate systems to univariate polynomials, to the quadratic systems of equations correspond pseudoquadratic polynomials ([4], Section 3), or polynomials of the form (\ref{1}).

An alternative and constructive proof of this fact comes from the process of expanding a given univariate polynomial into a system of $n$ equations in $n$ variables. It’s obvious that the correspondence between univariate polynomials $P(x)$ over $\mathbb{F}_{2^n}$ and systems $S_P$ of $n$ equations in $n$ variables over $\mathbb{F}_2$ is one-to-one. when expanding a term $x^d$ of $P(x)$ we get as many linear factors as the Hamming weight of $d$. The biggest Hamming weight will be the degree of the system. (This argument holds only for base field $\mathbb{F}_2$.) On pseudoquadratic polynomials it is by definition 2.

Northeastern University 360 Huntington Ave, Dept of Maths, 540 Lake Hall, Boston, MA 02115-5000. ilia.toli@gmail.com.