Co-Orthogonal Codes

Extended Abstract

Vince Grolmusz

Department of Computer Science, Eötvös University, Budapest, Address: Pázmány P. stny. 1/C, H-1117, Budapest, HUNGARY; E-mail: grolmusz@cs.elte.hu

Abstract. We define, construct and sketch possible applications of a new class of non-linear codes: co-orthogonal codes. The advantages of these codes are twofold: first, it is easy to decide whether two codewords form a unique pair (this can be used in decoding information or identifying users of some not-publicly-available or non-free service on the Internet or elsewhere), and the identification process of the unique pair can be distributed between entities, who perform easy tasks, and only the information, gathered from all of them would lead to the result of the identifying process: the entities, taking part in the process will not have enough information to decide or just to conjecture the outcome of the identification process.
Moreover, we describe a fast (and general) method for generating (non-linear) codes with prescribed dot-products with the help of multi-linear polynomials.

Keywords: non-linear codes, co-orthogonal codes, codes and set-systems, multi-linear polynomials, composite modulus

1 Introduction

In the present paper, we define, construct and sketch possible applications of a new class of non-linear codes: co-orthogonal codes. The advantages of these codes are twofold: first, it is easy to decide whether two codewords form a unique pair (this can be used in decoding information or identifying users of some not-publicly-available or non-free service on the Internet or elsewhere). Second, the identification process of the unique pair can be distributed between entities, who perform easy tasks, and only the information, gathered from all of them would lead to the result of the identifying process: the entities, taking part in the process will not have enough information to decide or just to conjecture the outcome of the identification process.

The identification is done by the following procedure: two sequences, $a = (a_1, a_2, \ldots, a_n)$ and $b = (b_1, b_2, \ldots, b_n)$ forms a *pair*, say, modulo 6, if 6 is a divisor of the sum

$$\sum_{i=1}^{n} a_i b_i.$$

If, for every a there is exactly one b which forms a pair with a (and this fact can be decided easily), then, for example, keeping b secret would identify a. Our main result is that these codes (co-orthogonal codes) contain much more code-words, than the orthogonal codes, where the pairs are identified by the fact that in the sum above 6 is not a divisor of the sum. We also note, that taking 6 (or a non-prime power other small integer) is an important point here: we will get more code-words than in the case of primes!

1.1 Orthogonal codes

Definition 1. *Let $n > 0$ and $r > 1$ be integers, and let Z_r denote modulo-r ring of integers. We call $A \subset Z_r^n$ an* orthogonal code*, if we can list the elements of A as $A = \{a^1, a^2, \ldots, a^\ell\} \cup \{b^1, b^2, \ldots, b^\ell\}$, such that for all $a^i \in A$: $a^i \cdot b^i \not\equiv 0 \pmod r$, but for all $i \neq j$: $a^i \cdot b^j \equiv 0 \pmod r$.*

In the definition above we allow $a^i = b^i$, and $u \cdot v$ denotes the dot-product (or scalar-product) of vectors u and v.

Example 1. Let $n = 2, r = 5$, and $A = \{a^1, a^2, b^1, b^2\}$, where $a^1 = (1,1), b^1 = (4,2), a^2 = (4,2), b^2 = (3,2)$. Then the pairwise dot-products can be given as a 2×2 matrix:

$$\begin{array}{c} \\ a^1 \\ a^2 \end{array} \begin{array}{c} \begin{array}{cc} b^1 & b^2 \end{array} \\ \begin{pmatrix} a^1 \cdot b^1 & a^1 \cdot b^2 \\ a^2 \cdot b^1 & a^2 \cdot b^2 \end{pmatrix} \end{array} = \begin{pmatrix} 6 & 5 \\ 20 & 16 \end{pmatrix} \equiv \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \pmod 5.$$

The advantages of orthogonal codes The orthogonal codes have very attractive properties:

- For each $a^i \in A$ it is easy to verify that a given vector $u \in A$ would serve as b^i: if $a^i \cdot u \not\equiv 0 \pmod r$ then $u = b^i$.
- The dot-product can be computed in a small memory: only modulo r computations (multiplications and additions) are needed.
- Suppose, that we have three players P_1, P_2, P_3, and the coordinates of the code-words of a^i and u are partitioned between them: suppose, that the first $n/3$ coordinates of the vectors are known for P_1, the second $n/3$ to P_2 and the third $n/3$ to P_3 (assume, that n is a multiple of 3). Then they can verify collectively whether $u = b^i$: for $j = 1, 2, 3$, P_j compute the dot-product of their known coordinates, and the mod r sum of their result gives the answer: if the value is not 0 modulo r, then $u = b^i$. Let us note, that the players *alone* tipically will not know the answer.
- An easy and fast parallel algorithm computes the dot-product of the two vectors, with n processors for the length-n vectors it can be done in $c \log n$ time.

Let us remark, that a trivial code with the highest possible rate has almost all properties mentioned above: indeed, consider code $B = \{0,1\}^n$, and let $a, b \in B$ form a pair if $a = b$. Since $a = b$ can be verified bitwise, it follows that the pair-verification can be done in parallel. However, as was noticed by Király [6], if $a \neq b$ then it will be witnessed by one or more processors or players, who knows only the bits (or sub-sequences) of a and b, so if a and b does not form a pair then it will be known for a player. This property can be fatally bad if our goal is to hide the outcome of the verification process.

The disadvantage of orthogonal codes It is easy to see, that if r is a prime, then the maximum size of an orthogonal code (that is, the cardinality of A, $|A|$) is at most $2n$, if the length of the code-words are n: (simply the matrix in Example 1 has full rank, because of the orthogonality property). That shows, that the rate of orthogonal codes are *extremely low* for r primes. It is not difficult to show, that the situation is not much better if r is a composite number: if r has ℓ prime divisors, then $|A|$ is at most $2\ell n$, which is still very small.

1.2 Co-orthogonal codes

Definition 2. *We call $A \subset Z_r^n$ a* co-orthogonal code*, if we can list the elements of A as $A = \{a^1, a^2, \ldots, a^\ell\} \cup \{b^1, b^2, \ldots, b^\ell\}$, such that for all $a^i \in A$: $a^i \cdot b^i \equiv 0 \pmod r$, but for all $i \neq j$: $a^i \cdot b^j \not\equiv 0 \pmod r$.*

Example 2. Let $n = 2, r = 6$, and $A = \{a^1, a^2, a^3, a^4, a^5, a^6, b^1, b^2, b^3, b^4, b^5, b^6\}$, where $a^1 = (2,1), a^2 = (5,1), a^3 = (2,3), a^4 = (2,2), a^5 = (1,2), a^6 = (3,5)$, and $b^1 = (5,2), b^2 = (1,1), b^3 = (3,2), b^4 = (1,5), b^5 = (2,5), b^6 = (1,3)$. Then the pairwise dot-products can be given as a 6×6 matrix:

$$\begin{array}{c} \\ a^1 \\ a^2 \\ a^3 \\ a^4 \\ a^5 \\ a^6 \end{array} \begin{array}{c} \begin{array}{cccccc} b^1 & b^2 & b^3 & b^4 & b^5 & b^6 \end{array} \\ \begin{pmatrix} 12 & 3 & 8 & 7 & 9 & 5 \\ 27 & 6 & 17 & 10 & 15 & 8 \\ 16 & 5 & 12 & 17 & 19 & 11 \\ 14 & 4 & 10 & 12 & 14 & 8 \\ 9 & 3 & 7 & 11 & 12 & 7 \\ 25 & 8 & 19 & 28 & 21 & 18 \end{pmatrix} \end{array} \equiv \begin{pmatrix} 0 & 3 & 2 & 1 & 3 & 5 \\ 3 & 0 & 5 & 4 & 3 & 2 \\ 4 & 5 & 0 & 5 & 1 & 5 \\ 2 & 4 & 4 & 0 & 2 & 2 \\ 3 & 3 & 1 & 5 & 0 & 1 \\ 1 & 2 & 1 & 4 & 3 & 0 \end{pmatrix} \pmod 6.$$

Note, that for code-length is 2 again, but we have $|A| = 12$.

We have shown in [2], that if r is a prime, then the rate of co-orthogonal codes are not much larger than the rate of orthogonal codes: there exist at most $O(n^{r-1})$ co-orthogonal code-words for any n . However, quite surprisingly, for non-prime-power, composite r's (e.g., $r = 6$), there are co-orthogonal codes of much larger rate (see Theorem 1).

The advantages of co-orthogonal codes It is obvious, that the co-orthogonal codes have the same advantages as the orthogonal codes: It is very easy to identify the matching code-word pairs; the computations can be done modulo a small number (in our example this number is 6); the identification process can be shared between players, not knowing the outcome of the identification. However, we can show, that the serious problem of the orthogonal codes, i.e., their very low rate does not appear here. We show this in the next section, with algorithmically fast constructions.

2 Constructing Co-orthogonal codes modulo a composite number

We give here some constructions for co-orthogonal codes. These constructions use techniques from papers [3], [2], and especially from [4]. The existence of (special) mod 6 co-orthogonal codes with high rate falsified old conjectures in extremal set theory (see [3] for details). This high rate of our codes facilitates the possible application of co-orthogonal codes. We note, that our codes here are binary (i.e., only 0 and 1 will appear in the code-words), but r, the modulus is a non-prime-power composite number.

Theorem 1. *Let m be a positive integer, and suppose that m has $r > 1$ different prime divisors: $m = p_1^{\alpha_1} p_2^{\alpha_2} ... p_r^{\alpha_r}$. Then there exists $c = c(m) > 0$, such that for every integer $N > 0$, there exists an explicitly constructible binary co-orthogonal code H modulo m, of length N , such that $|H| \geq \exp\left(c\frac{(\log N)^r}{(\log\log N)^{r-1}}\right)$. Moreover, the minimum Hamming-distance between any two codewords of H is*

$$\frac{c' N \log\log N}{\log N}$$

for a positive c'.

Note, that $|H|$ grows faster even for $m = 6$ than any polynomial in n. Note also, that the code-generation is a fast polynomial algorithm (see Section 3.3).

2.1 k-wise Co-Orthogonal Codes

A generalization of the co-orthogonal codes is the *k-wise co-orthogonal codes.* While co-orthogonal codes are useful since their easy pair-identification property, k-wise co-orthogonal codes can be used for group-identification. Since dot-product (or scalar-product) can be defined only between only two vectors, we need a natural generalization here.

Definition 3. *Let $A = \{a_{ij}\}$ and $B = \{b_{ij}\}$ two $u \times v$ matrices over a ring R with unit element* 1. *Their* Hadamard-product *is an $u \times v$ matrix $C = \{c_{ij}\}$, denoted by $A \odot B$, and is defined as $c_{ij} = a_{ij} b_{ij}$, for $1 \leq i \leq u$, $1 \leq j \leq v$. Let*

$k \geq 2$. The k-wise dot product of vectors of length n, $a^1, a^2, \ldots, a^k$ is computed as

$$(a^1 \odot a^2 \odot \cdots \odot a^k) \cdot \mathbf{1},$$

where $\mathbf{1}$ denotes the length n all-1 vector.

Note, that if a^i is a characteristic vector of a subset A_i of an n-element ground-set (for $i = 1, 2, \ldots, k$), then $a^1 \odot a^2 \odot \cdots \odot a^k$ is a characteristic vector of $\bigcap_{i=1}^{k} A_i$, and the k-wise dot-product of $a^1, a^2, \ldots, a^k$ gives the *size* of this intersection.

Definition 4. *For a $k \geq 2$ we call $A \subset Z_r^n$ a* k-wise co-orthogonal code *modulo m, if the following holds:*

- *$\forall a^1 \in H$ there exist $a^2, a^3, \ldots, a^k \in H$ such that $(a^1 \odot a^2 \odot \cdots \odot a^k) \cdot \mathbf{1} \equiv 0 \pmod{m}$,*
- *If $\{a^2, a^3, \ldots, a^k\} \neq \{b^2, b^3, \ldots, b^k\}$, $b^i \in H$, $i = 2, 3, \ldots, k$, then*

$$(a^1 \odot b^2 \odot b^3 \odot \cdots \odot b^k) \cdot \mathbf{1} \not\equiv 0 \pmod{m}.$$

Now we can formulate the following result. The construction is – surprisingly – exactly the same as the construction for proving Theorem 1, and it is an improvement of a construction appeared in [5] for set-systems.

Theorem 2. *Let $n, t \geq 2$ integers, and let $p_1, p_2, \ldots, p_r$ be pairwise different primes, and let $m = p_1 p_2 \cdots p_r$. Then there exists a $c_m > 0$ and an explicitly constructible code H of length N, such that $|H| \geq \exp\left(\frac{c_m (\log N)^r}{(\log \log N)^{r-1}}\right)$, and H is k-wise co-orthogonal for* any *$k \geq 2$.*

The identification of a group is done with the easy computation of the k-wise dot-product of the codes of the members of the group. If this number is 0 modulo m, then the group is passed the identification, otherwise it failed.

3 Constructing code $f(A)$ from f and A

Our construction is based on a method given in [4] and the BBR-polynomial [1]. In paper [4], we gave a general construction for hypergraphs with prescribed intersection sizes. Our construction described here can also be applied for constructing codes with prescribed dot-product matrices. For a detailed discussion of this problem, see [4].

Here we re-formulate this method in a form which is more suitable for our purposes in the present work.

Definition 5. *Let $f(x_1, x_2, \ldots, x_n) = \sum_{I \subset \{1,2,\ldots,n\}} \alpha_I x_I$ be a multi-linear polynomial, where $x_I = \prod_{i \in I} x_i$. Let $w(f) = |\{\alpha_I : \alpha_I \neq 0\}|$ and let $\mathrm{L}_1(f) = \sum_{I \subset \{1,2,\ldots,n\}} |\alpha_I|$.*

Definition 6. *Let $A = \{a^1, a^2, \ldots, a^\ell\} \subset \{0,1\}^n$ be a binary code. Then the matrix of A, denoted by $M(A)$, is an $n \times \ell$ 0-1 matrix, with column j equal to the code-word a^j, for $j = 1, 2, \ldots, \ell$.*

Definition 7. *Let $A = \{a^1, a^2, \ldots, a^\ell\} \subset \{0,1\}^n$ be a binary code, and let f be an n-variable multi-linear polynomial with positive integer coefficients. Then binary code $f(A) = \{c^1, c^2, \ldots, c^\ell\} \subset \{0,1\}^{\mathrm{L}_1(f)}$ is defined as follows: The rows of $M(f(A))$ correspond to monomials x_I's of f; there are α_I identical rows of $M(f(A))$, corresponding to the same x_I. The row, corresponding to x_I, is defined as the Hadamard-product of those rows i of $M(A)$, with $i \in I$.*

Example 3. Let $f(x_1, x_2, x_3, x_4) = x_1 + x_2 + 2x_3x_4$, and let the matrix $M(A)$ of code $A = \{a^1, a^2, a^3\}$ be

$$M(A) = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{ccc} a^1 & a^2 & a^3 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} \end{array}.$$

Then the matrix of code $f(A)$ is

$$M(f(A)) = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3x_4 \\ x_3x_4 \end{array} \begin{array}{c} \begin{array}{ccc} c^1 & c^2 & c^3 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} \end{array}.$$

The most important property of code $f(A)$ is given in the following Theorem:

Theorem 3. *Let $t \geq 2$ and let $a^{i_\ell} \in A$ for $\ell = 1, 2, \ldots, t$. Then*

$$f(a^{i_1} \odot a^{i_2} \odot \cdots \odot a^{i_t}) = (c^{i_1} \odot c^{i_2} \odot \cdots \odot c^{i_t}) \cdot \mathbf{1}. \tag{1}$$

An analogous theorem for set-systems appeared in [4]. We reproduce here its short proof.

Proof. Consider a monomial x_I of f, for some $I \subset \{1, 2, \ldots, n\}$. Let us observe, that monomial x_I contributes one to the left hand side of equation (1) exactly when for all $j \in I$, the j^{th} coordinate of every code-word $a^{i_1}, a^{i_2}, \ldots, a^{i_t}$ are equal to 1. This happens exactly if the coordinate of $c^{i_1} \odot c^{i_2} \odot \cdots \odot c^{i_t}$, corresponding to monomial x_I, is 1, that means, that one is contributed to the right hand side of 3. □

3.1 Our main construction

Our binary co-orthogonal code will be constructed as $f(A)$, from a well-chosen polynomial f and a dense code A. For simplicity, in this preliminary version of this work, we prove Theorems 1 and 2 only for modulus $m = 6$.

Our f will be the BBR-polynomial. *Barrington, Beigel* and *Rudich* [1] showed, that for integers α and β, there exists an explicitly constructible, symmetric, n-variable, degree-$O(\min(2^\alpha, 3^\beta))$ polynomial f, (the BBR-polynomial), satisfying over $x = (x_1, x_2, \ldots, x_n) \in \{0,1\}^n$:

$$f(x) \equiv 0 \pmod 6 \iff \sum_{i=1}^{n} x_i \equiv 0 \pmod{2^\alpha 3^\beta}. \tag{2}$$

Our A is defined as follows. Let A_0 denote all the weight-$2^\alpha 3^\beta - 1$ binary code-words of length $n-1$. Code A is got from code A_0 if we add a leading 1 for all codewords in A_0, consequently, each word in A is binary, has weight $2^\alpha 3^\beta$, and the dot-product of any two different words in A is non-zero, but less than $2^\alpha 3^\beta$. However, the dot-product of any $a \in A$ with itself is equal to $2^\alpha 3^\beta$.

Let us remark, that A itself is a co-orthogonal code modulo $2^\alpha 3^\beta$.

Moreover, for any $k \geq 2$, the dot-product of any k words (containing at least two different words) in A (see Definition 3) is also non-zero, and less than $2^\alpha 3^\beta$.

Now let α be the smallest integer that $n^{1/3} < 2^\alpha$, and let β be the smallest integer such that $n^{1/3} < 3^\beta$. Then the degree of f is $O(n^{1/3})$.

Let us consider now code $H = f(A)$. It contains at least $\binom{n}{n^{2/3}}$ code-words of length $N = \mathrm{L}_1(f) = O(n^{n^{1/3}})$, so

$$|H| = |f(A)| = \exp\left(c\frac{(\log N)^2}{(\log\log N)}\right).$$

And, from Theorem 3, for any $a \in H$ a forms a pair only with $b = a$, for any $b \in H$, modulo 6, this proves the first part of Theorem 1. The k-wise co-orthogonality follows also from Theorem 3.

For computing the minimum-distance of the code (for proving Theorem 1), we should note, that any two elements a^i and a^j of A differs in at least one bit. The weight of a^i is $2^\alpha 3^\beta = \Theta(n^{2/3})$, that means — because polynomial f is symmetric – that the corresponding code-word of $f(A)$, c^i has weight at least

$$\sum_{k=1}^{n^{1/3}} \binom{2^\alpha 3^\beta}{k} > \binom{n^{2/3}}{n^{1/3}}.$$

If we flip one 1-bit to zero, then at least

$$\binom{n^{2/3}}{n^{1/3}} - \binom{n^{2/3}-1}{n^{1/3}}$$

monomials of f become zero, that is, at least that many coordinates of c^i become zero by flipping any bit. Now the result follows.

3.2 Alternative constructions

We would get more dense codes if we had a BBR polynomial with smaller degree, but, unfortunately, it is not known whether there exists such polynomial with lower degree. (For other applications of the BBR polynomial see [3] and [2]).

Alternatively, we can choose different codes A for the construction. For example, consider the following code A. Let vectors a^i be all the weight-$2^\alpha 3^\beta$ code-words of length n, and let b^i be the complement of a^i, for $i = 1, 2, \ldots, \binom{n}{2^\alpha 3^\beta}$. Then it is easy to see, that A is co-orthogonal code modulo $2^\alpha 3^\beta$. Then, from Theorem 3, with the BBR-polynomial f, code $f(A)$ is also a co-orthogonal code, but now modulo 6.

We list some further variants of codes A in Section 4.

3.3 Algorithmic complexity of computing $f(A)$

Let a^i be a code-word of A and let c^i the corresponding code-word of $f(A)$. Then the coordinates of c^i is computed as the values of monomials of f with substituting a^i: $f(a^i)$. The value of a degree-d monomial can be computed in $O(d)$ steps; so the length-N c^i can be computed in $N \log N$ steps.

4 Cryptographic applications

Perhaps the most straightforward application is the following one: keep secret vector a^i, and accept vector x only if $a^i \cdot x \equiv 0 \pmod{m}$. From the co-orthogonal code, only $x = b^i$ satisfy this relation, but outside that code, many more x's may satisfy it; for example, $x = 0$ always satisfies this requirement.

Consequently, for any cryptographic application first we should verify whether x is in the code or not. We call this phase *membership testing*. If x fails the membership test, reject it. If x passes the membership test, then compute $y = a^i \cdot x \bmod m$ (even in a distributed way), and accept x iff y is 0, modulo m, and reject it otherwise.

Another problem with our main construction (Section 3.1) is the following: The pair of any $a \in f(A)$ is the same a itself!. That means, that if the verification process is distributed among players, any player who finds that a coordinate is different in a and b will know that they are not pairs. This problem can be avoided by applying some linear transformations for the codes, as described in Section 4.2.

4.1 Membership testing

It is easy to see, that in any polynomial f, satisfying property (2), must contain monomials of degree one x_i $(i = 1, 2, \ldots, \ell)$ with a non-zero coefficient. That is also true for the BBR-polynomial f used in our construction.

Now, suppose that we need to verify whether $c \in f(A)$. We know, that which coordinates of a code-word c should correspond to monomials x_i $(i = 1, 2, \ldots, n)$, then, first we read only these (at most n) coordinates of c. The values of x_i should be equal to the coordinates of some $a^j \in A$: $x_i = a^j_i$, for $i = 1, 2, \ldots, n$. If this will not be satisfied, discard c, it is not in our code. Otherwise, compute from a^j and from f the code-word c^j. If $c = c^j$, accept c, otherwise reject.

This algorithm can be performed in $O(|c| \log(|c|)$ steps by the most straightforward implementation.

4.2 Non-binary, non-self-paired codes

We describe a quite general method to get non-binary from the binary co-orthogonal codes generated in Section 3.1. Note, that if the pair of a was a itself in the original construction, this property will disappear in the modified one:

Our idea comes from the following well-known identity:

$$x'A \cdot y = x' \cdot yA^T,$$

where x', y are length-n q-ary code-words, and A is an $n \times n$ matrix.

So, if m is a prime we can consider the m-element-field, and transform our code C into code C_A for any non-singular matrix A over the field as follows:

$$C_A = \{xA^{-1} : x \in C\} \cup \{yA^T : y \in C\}.$$

Clearly, if $x'A = x$, then $x \cdot y = x'A \cdot y = x' \cdot yA^T$, so if C was a co-orthogonal code, then C_A is also a co-orthogonal code over the field, and the pair of $x' = xA^{-1}$ is $xA^T = x'AA^T$, which tipically differs from x' (we should avoid unitary A's).

However, in our main construction m is composite, say $m = 6$. So, we should choose a non-singular $n \times n$ matrix A over the 2 element field, and another one, B, over the 3 element field. Suppose now, that C is a co-orthogonal code modulo 6. Then certainly

$$C_{A,B} = \{3xA^{-1} + 2xB^{-1} : x \in C\} \cup \{3yA^T + 2yB^T : y \in C\}$$

is a co-orthogonal code mod 6, and the pair of $3xA^{-1} + 2xB^{-1}$ is exactly $3xA^T + 2xB^T$ (Note that A^{-1} is the inverse of A over GF_2, and B^{-1} is the inverse of B over the three element field.)

Acknowledgment. We are grateful to Zoltán Király and to Lajos Rónyai for discussions on this topic. The author acknowledges the partial support of János Bolyai Fellowship and research grants EU FP5 IST FET No. IST-2001-32012, OTKA T030059 and an ETIK grant.

References

1. David A. Mix Barrington, Richard Beigel, and Steven Rudich. Representing Boolean functions as polynomials modulo composite numbers. *Comput. Complexity*, 4:367–382, 1994. Appeared also in *Proc. 24th Ann. ACM Symp. Theor. Comput.*, 1992.
2. Vince Grolmusz. Low-rank co-diagonal matrices and Ramsey graphs. *The Electronic Journal of Combinatorics*, 7:R15, 2000. www.combinatorics.org.
3. Vince Grolmusz. Superpolynomial size set-systems with restricted intersections mod 6 and explicit Ramsey graphs. *Combinatorica*, 20:73–88, 2000.
4. Vince Grolmusz. Constructing set-systems with prescribed intersection sizes. Technical Report DIMACS TR 2001-03, DIMACS, January 2001. ftp://dimacs.rutgers.edu/pub/dimacs/TechnicalReports/TechReports/2001/2001-03.ps.gz.

5. Vince Grolmusz. Set-systems with restricted multiple intersections and explicit Ramsey hypergraphs. Technical Report DIMACS TR 2001-04, DIMACS, January 2001. ftp://dimacs.rutgers.edu/pub/dimacs/TechnicalReports/TechReports/2001/2001-04.ps.gz.
6. Zoltán Király. personal communication.