New Multiple Covering Codes by Tabu Search

Patric R. J. Östergård*
Department of Computer Science
Helsinki University of Technology
FI–02150 Espoo, FINLAND

Abstract

The problem of finding good multiple coverings (with or without repeated words) of the Hamming space F_2^n is considered. Recently, extensive tables of upper and lower bounds for such codes were presented (Hämäläinen *et al.*, Bounds for binary multiple covering codes, *Des. Codes Cryptogr.* **3** (1993), 251–275). The new codes found in this work improve on 27 upper bounds in those tables. The codes were found using tabu search. The implementation of this method is discussed, and it is shown how it also can be used to search for large codes.

1 Introduction and Definitions

The problem of finding good coverings of Hamming spaces has attracted a lot of attention during the last decade. In this paper binary codes will be discussed. However, many of the results can be generalized to codes over other (even mixed) alphabets. We consider codes over F_2^n, where $F_2 = \{0, 1\}$ is the two-element Galois field. A code is a nonempty set $C \subseteq F_2^n$. In some particular cases we allow C to be a multiset. The Hamming distance $d(x, y)$ between two words $x, y \in F_2^n$ is the number of coordinates in which they differ. The Hamming distance between a word $x \in F_2^n$ and a code $C \subseteq F_2^n$ is $d(x, C) = \min_{c \in C} d(x, c)$.

A code C is said to be an $(n, |C|, r, \mu)$ *multiple covering* (MC) if for all $x \in F_2^n$ there is a set of codewords $C' \subseteq C$, such that $|C'| \geq \mu$ and $d(x, c) \leq r$ for all $c \in C'$. Furthermore, if we allow C and C' to be multisets we call the code a *multiple covering with repeated codewords* (MCR). We are now interested in the functions

$$\begin{aligned} K(n, r, \mu) &= \min\{M \mid \text{there is an } (n, M, r, \mu) \text{ MC}\} \text{ and} \\ \overline{K}(n, r, \mu) &= \min\{M \mid \text{there is an } (n, M, r, \mu) \text{ MCR}\}. \end{aligned}$$

It is in practice impossible to determine exact values of these functions in the general case, so effort has been put into obtaining upper and lower bounds. Upper bounds are constructive: they are proved by finding a corresponding code. If $\mu = 1$

*This research was supported by the Academy of Finland.

we are considering traditional covering codes, which have been extensively studied; the most recent tables of upper and lower bounds on binary covering codes can be found in [15] and [12], respectively.

Earlier results on multiple coverings include those by Clayton [3] and Van Wee *et al.* [16]. Recently, Hämäläinen *et al.* [6] collected bounds on $K(n, r, \mu)$ and $\overline{K}(n, r, \mu)$ for $n \leq 16$, $r \leq 4$, $\mu \leq 4$. At the end of their introduction they mention some reasons why the codes in the paper can be considered reasonably good. Anyhow, in this paper it is shown how efficient and extensive computer searches have led to as many as 27 improvements on their upper bounds. New results on lower bounds for multiple covering codes can be found in [2].

In Section 2 the optimization method used in the search, tabu search, is briefly discussed. A matrix method that can be used to find large codes is presented, and data structures employed to make the search more efficient are explained. It is also mentioned how these can be slightly modified and used in search for other types of coverings. In Section 3 the new upper bounds are tabulated and compared with the best known old results. Codes corresponding to new bounds are listed in the Appendix.

2 Computer Search for Coverings

The outlines of our search for codes (and in most other works in the same area) are as follows: The size of the code, $|C|$, is fixed (to a value slightly better than the best known upper bound) for given values of n, r, and μ. Thereafter, starting from an initial code, which is usually chosen at random, we try to perturb the codewords to obtain a desired covering. Before going into the methods used in the search for coverings, we display the data structures used.

2.1 Data Structures

The structures are displayed in Figure 1. The values in the tables are sampled from a search for a $(4, 8, 1, 2)$ MC (which is known to exist). The two-dimensional array `table[][]` has one column for every word in F_2^n (in lexicographical order). The first element in each column indicates how many times the word is covered. The other elements contain pointers to the $\sum_{i=0}^{r} \binom{n}{i}$ words that are within Hamming distance r from the word. The codewords are saved in a single-dimension array `code[]` as pointers that point to the corresponding entries in table `table[][]`. To help understanding our example, we have used indices instead of pointers in the figure.

Let $C_i = \{c \in C \mid d(c, w_i) \leq r\}$, where $w_i \in F_2^n$ is the binary word that is i in decimal form. Now consider the function

$$f(C) = \sum_{i=0}^{2^n-1} \max\{0, \mu - |C_i|\}. \tag{1}$$

`code[]`

0	2	3	5	6	8	11	15

`table[][]`

3	3	4	3	3	1	2	4	2	2	3	3	1	2	2	2

0	0	0	1	0	1	2	3	0	1	2	3	4	5	6	7
1	1	2	2	4	4	4	5	8	8	8	9	8	9	10	11
2	3	3	3	5	5	6	6	9	9	10	10	12	12	12	13
4	5	6	7	6	7	7	7	10	11	11	11	13	13	14	14
8	9	10	11	12	13	14	15	12	13	14	15	14	15	15	15

Figure 1: Data structures.

The code C is a required covering if $|C_i| \geq \mu$ for all words in F_2^n, that is, if $f(C) = 0$. The problem of finding a multiple covering can now be seen as a combinatorial optimization problem, where the cost function $f(C)$ has to be minimized. The example in Figure 1 has cost $f(C) = 2$.

In practically all previously published results on computer searches for covering codes, the optimization method used has been *simulated annealing*, see, for example, [11, 13, 17]. However, the author recently discovered [14] that *tabu search* [5] adapted in a proper way to covering problems outperforms simulated annealing. That conclusion is confirmed by the results in this paper.

2.2 Tabu Search

Tabu search is a *local search method.* A key concept of such methods is that of *neighborhood.* The neighborhood of a solution is a set of solutions that are obtained by changing (usually slightly) the current solution. In local search, the optimization process gives a series of solutions, where a solution always is a neighbor of the previous solution. A well-known such method is *steepest descent*, where we look at all neighbors with lower cost and choose the one with least cost (or randomly one of these if there are many).

In proceeding towards a minimum, tabu search obeys the steepest descent heuristic. The process continues until a *local minimum* is reached. No neighbor of such a solution has lower cost. To get out of this minimum we have to accept solutions that do not improve on the present value of the cost function. The neighbor with least cost is still chosen. Now, in the next step there is a risk that we will get back into the minimum from which we are trying to escape, which would lead to a loop in the optimization process. To avoid this, a list of prohibited changes, a tabu list, is created. This list contains information about the L most recent changes, the inverses of which are prohibited. The changes are usually not saved as such in the tabu list, but in an encoded form. See [5] for further details.

In this way, tabu search goes through local minima in the part of the search space where the costs are low. A global minimum will hopefully be found in this search. The cost of a global minimum for the problem discussed here is 0, and the search can be terminated when such a solution is encountered.

2.3 Searching for Coverings

The optimization process proceeds as follows: Go through the integers in $\{0, 1, \ldots, 2^n - 1\}$ cyclically until an i such that $|C_i| < \mu$ is encountered. Now the neighborhood of the present solution consists of all solutions that are obtained by replacing one codeword $c \in C$ with a codeword c' such that $d(c', w_i) \leq r$. For MCs, we must also check that $c' \notin C$.

Let us again take a look at the example in Figure 1. We go through the first elements of the columns of `table[][]` and find out that $|C_5| = 1 < 2 = \mu$. The costs of the solutions in the neighborhood is now displayed in Figure 2. The columns are the old codewords (c), and the rows are the possible new codewords (c').

$c' \backslash c$	0	2	3	5	6	8	11	15
1	2	2	1	3	3	3	2	4
4	1	0	0	2	1	3	2	3
5	*	*	*	*	*	*	*	*
7	2	1	1	3	2	4	2	3
13	1	1	0	1	2	2	0	1

Figure 2: Costs of solutions in neighborhood.

If we were searching for an MCR, we would also take into account the starred moves in Figure 2. As can be seen in the figure, four of the moves lead to least cost. One of these is chosen at random. Since the cost is now 0, the search is stopped and the code is saved. If the cost had been positive, we would have continued the search and added *the position in* `code[]` *of the word that was changed* to the tabu list. A good rule of thumb is to choose a list length close to $L = |C|/10$.

2.4 A Matrix Method

With growing n, $|C|$, and/or r, there comes a point when a direct search does not work satisfactorily any more. This is due to two reasons. Firstly, the sizes of the tables in Figure 1 must not exceed the size of the computer memory, and, secondly, with many codewords and a large space (F_2^n) the algorithm does not find good codes very easily. One possible solution to this problem is to reduce the search space by giving the code a structure. This can be accomplished by the following *matrix method.*

The method was presented for coverings with $\mu = 1$, $r = 1$ by Blokhuis and Lam [1]. It turns out that it is straightforward to generalize it to many other types of coverings, including multiple coverings [6]. Let $A = [a_1\ a_2\ \cdots\ a_n]$ be a $k \times n$ matrix over F_2. A set $S \subseteq F_2^k$ is said to form a μ-fold r-covering (or simply covering if the value of r is understood) of F_2^k using A if every $x \in F_2^k$ can be expressed in at least μ ways as a sum of one element of S and at most r columns of A.

Theorem 1 ([6]) *If S forms a μ-fold r-covering of F_k^2 using a $k \times n$ matrix A, then $K(n, r, \mu) \leq |S| \cdot 2^{n-k}$.*

A similar theorem can be stated for MCRs, see [6, Theorem 9]. Now, when we try to apply this result (for given n, r, and μ), we first have to choose an appropriate k. If $k = n$ and $A = I$ is the $n \times n$ identity matrix, we are faced with the direct approach. The smaller the value of k is, the stronger the structure of the final code is. In [6, Theorem 9] the rank of A is assumed to be k; however, even if the rank of A is less than k, the approach in [10, Corollary 1] can be used to prove Theorem 1.

After fixing k, we try to find coverings for many different matrices A. We consider matrices of form $A = [I\ M]$, and adopt the same approach as in [13]; the reader is referred to that paper for further details. It should, however, be noted that for multiple coverings also matrices A with all-zero and repeated columns should be considered. In [13] 1-fold coverings were considered; in that case such columns are superfluous.

The data structures presented in Figure 1 can also be used with this approach. The pointers are set to point to the words obtained when any linear combination of at most r columns of A are added to the original word.

We conclude the discussion of the matrix method by stating a theorem that is a straight-forward generalization of a result in [15]. The elements $s_1, s_2, \ldots, s_t$ are not necessarily distinct.

Theorem 2 *If $t > 1$ and $s_1, s_2, \ldots, s_t$ form a μ-fold 1-covering of F_2^k using a $k \times n$ matrix A, then $\overline{K}(t-1, 1, \mu) \leq (n+1) \cdot 2^{t-k-1}$. If A contains no all-zero and no repeated columns, then $K(t-1, 1, \mu) \leq (n+1) \cdot 2^{t-k-1}$.*

Proof. W.l.o.g, assume that s_1 is the all-zero vector (we can add any vector to all elements s_i). Now let $s'_1 = s_1, s'_2 = a_1, s'_3 = a_2, \ldots, s'_{n+1} = a_n$ and $a'_1 = s_2, a'_2 = s_3, \ldots, a'_{t-1} = s_t$. It is a straight-forward task to verify that the elements $s'_1, s'_2, \ldots, s'_{n+1}$ form a μ-fold 1-covering of F_2^k using the $k \times (t-1)$ matrix $A' = [a'_1\ a'_2\ \cdots\ a'_{t-1}]$. Furthermore, if A contains no all-zero and no repeated columns, the new words s'_i are all different, so the new code is a MC. □

Corollary 1 $K(\overline{K}(n, 1, \mu) - 1, 1) \leq (n+1) \cdot 2^{\overline{K}(n,1,\mu)-n-1}$.

These results can be used to get good bounds especially for large n. However, also some best known codes of length at most 16 can be explained by these. From $\overline{K}(4, 1, 2) = 7$ [6] we get $K(6, 1, 2) \leq 20$. That bound was proved in [6] using the matrix method, with $t = 10$ and a 5×6 matrix A that has no all-zero and no repeated columns. Using that construction and Theorem 2 we then obtain $K(9, 1, 2) \leq 112$, another best known upper bound.

2.5 Other Covering Problems

In the cost function (1) we can, of course, set $\mu = 1$, so our approach works as such also for the traditional covering problem.

In [7] the problem of finding good codes that are multiple coverings of the farthest-off points is considered. We have such a covering if for all $w_i \in F_2^n$, $d(w_i, C) \leq r$, and $|C_i| \geq \mu$ whenever $d(w_i, C) = r$ (C_i and w_i are defined as in Section 2.1). Now, we divide the pointers in `table[][]` in Section 2.1 into two parts. For each word (column), the pointers to words within distance at most $r-1$ are separated from the pointers to words at distance r. In the optimization process, words that belong to the former group are considered as if they were covered μ times instead of once. Cost function (1) can then be used as in the search for MCs.

Other types of coverings, for example, nonbinary codes and weighted coverings [4], can be considered by similar modifications. The interested reader is also referred to [9].

3 New Binary Multiple Covering Codes

The new bounds that we have found and that improve on those in [6] are displayed in Table I. Corresponding codes can be found in Table II in the Appendix or can be derived from other codes as explained in this section. We will now briefly discuss some of the new codes.

Table I. New upper bounds on $K(n, r, \mu)$.

n	r	μ	Lower bound [6]	Upper bound [6]	New upper bound
8	1	4	114	125	124
9	1	4	206	220	216
10	1	2	187	220	216
10	1	3	289	320	316
10	1	4	374	416	408
11	1	2	342	380	368
12	1	2	631	752	704
12	1	4	1262	1376	1344
13	1	3	1756	1984	1920
13	1	4	2342	2560	2528
14	1	3	3356	3776	3712
14	1	4	4370	4992	4864
8	2	3	22	26	24
11	2	3	92	104	100
13	2	2	190	256	240
13	2	3	268	352	336
10	3	2	13	19	18
10	3	3	18	26	24
10	3	4	24	32	30
11	3	2	18	30	24
11	3	3	27	40	36
14	3	2	74	128	120
12	4	2	11	19	18
12	4	4	22	32	30
13	4	2	16	30	26
13	4	3	23	40	36

In [6] only two cases are given where the best known upper bounds for $K(n,r,\mu)$ and $\overline{K}(n,r,\mu)$ differ. The first entry in Table I removes one of these (the other one is $7 = \overline{K}(4,1,2) < K(4,1,2) = 8$). We have been able to find one new MCR that improves on the best known MC. The code is listed in Table II, and it proves $\overline{K}(10,1,3) \leq 312$ ($K(10,1,3) \leq 316$).

For definitions of the ADS construction and related concepts—such as normality and subnormality—used in the next two paragraphs, the reader is referred to [6, 8].

The $(8,24,2,3)$ code in Table II (listed in the Appendix) is normal, so the ADS construction (with the code $\{000,111\}$) can be used to get $(8+2i,24,2+i,3)$ ($i \geq 1$) MCs. Other families of new MCs obtained in the same way are $(10+2i,18,3+i,2)$, $(10+2i,30,3+i,4)$, and $(11+2i,36,3+i,3)$. The normality of the codes acting as seeds for these families has been checked. Care must be taken when the ADS construction is applied, since not all coordinates of these codes are acceptable.

Some best known codes are abnormal. An abnormal $(13,112,3,4)$ MC can be found in [6]. The $(11,24,3,2)$ MC in Table II is also abnormal. This is interesting, since the code is very good and might even be optimal. It is not even subnormal. This was verified using a computer program that checked all $2^{24} = 16777216$ possible partitions of the 24 codewords into two subcodes. If there is a normal (or subnormal) MC with the same parameters, then a record-breaking $(13,24,4,2)$ MC can be constructed.

The following old bounds can be explained in an easy way using the results of this paper. The bound $K(12,4,3) \leq 24$ follows from the results in the previous paragraphs. The bound $K(12,3,2) \leq 48$ is obtained using $K(n+1,r,\mu) \leq 2K(n,r,\mu)$. By taking the union of the new $(11,24,3,2)$ MC and one of its translates (this is possible without getting multiple occurrences of codewords, since the code has minimum distance 3), we get $K(11,3,4) \leq 48$.

The new bounds $K(11,1,2) \leq 368$ and $K(11,3,2) \leq 24$ are best known upper bounds even if we only require the words $w \in F_2^n$ with $d(w,C) = r$ to be covered $\mu = 2$ times [7].

The search required about 4 CPU-months of computation time on SUN workstations. More effort would certainly have given more and better improvements; however, a trade-off always has to be made between computation time and quality of results in studies where probabilistic methods are used.

Appendix

New multiple coverings are here listed in hexadecimal form. All but one of these codes were found by the matrix method. The columns of M and the words of S are listed in Table II, separated by a semi-colon. The number of rows in the matrix (k) is also given for each code. All the words of a $(9,216,1,4)$ code are listed. All codes are MCs, except for the $(10,312,1,3)$ MCR.

Table II. New multiple coverings.

Code	k	Words of M and S
$(8, 124, 1, 4)$	7	7C; 0, 1, 2, 3, 4, 7, 8, B, C, D, E, F, 12, 14, 17, 1D, 21, 27, 28, 2E, 31, 32, 34, 37, 38, 39, 3A, 3B, 3D, 3E, 42, 48, 4B, 4D, 50, 51, 53, 54, 55, 56, 57, 58, 59, 5A, 5B, 5C, 5E, 5F, 61, 62, 64, 65, 66, 67, 68, 6B, 6D, 6E, 72, 74, 7B, 7D.
$(9, 216, 1, 4)$		3, 4, 5, 6, 9, D, 10, 12, 17, 19, 1A, 1C, 1D, 20, 23, 24, 28, 2A, 2E, 31, 33, 37, 3A, 3D, 3E, 3F, 43, 46, 47, 48, 49, 4B, 4C, 4F, 50, 54, 56, 57, 59, 5A, 60, 61, 62, 65, 6B, 6C, 6E, 6F, 72, 74, 75, 79, 7A, 7D, 81, 82, 83, 89, 8A, 8E, 8F, 90, 95, 97, 9B, 9C, 9E, A4, A6, A7, A8, A9, AD, AF, B0, B1, B4, B6, B8, BB, C2, C4, C5, C8, CA, CC, D0, D1, D3, D5, DB, DD, DE, DF, E1, E2, E5, E7, EB, EC, F2, F3, F6, F8, FB, FC, FD, FE, 100, 105, 106, 108, 10A, 10B, 10E, 10F, 111, 112, 115, 11B, 11C, 11F, 123, 125, 126, 127, 128, 129, 12B, 12D, 131, 132, 134, 136, 138, 13C, 140, 141, 143, 145, 14A, 14D, 151, 153, 154, 15A, 15C, 15E, 15F, 160, 162, 166, 16C, 16D, 16E, 173, 174, 177, 178, 179, 17B, 17F, 180, 181, 183, 184, 18C, 18F, 192, 195, 196, 197, 198, 199, 19B, 19C, 1A2, 1A3, 1A4, 1AA, 1AC, 1AD, 1B1, 1B2, 1B5, 1BA, 1BB, 1BD, 1BE, 1BF, 1C4, 1C6, 1C7, 1C9, 1CA, 1CD, 1CF, 1D2, 1D3, 1D6, 1D8, 1D9, 1DD, 1E0, 1E1, 1E7, 1E9, 1EA, 1EB, 1EE, 1F0, 1F4, 1F5, 1F7, 1F8, 1FE.
$(10, 216, 1, 2)$	9	100; 6, C, 10, 11, 15, 1C, 21, 26, 2C, 31, 33, 3A, 3F, 45, 47, 48, 4C, 52, 5A, 5F, 62, 69, 6F, 74, 79, 7F, 80, 82, 8D, 97, 99, 9E, A7, A8, AA, AD, B4, BA, BB, C1, C3, CE, CF, D3, D4, D9, E3, E4, EA, F0, F4, F5, F6, FC, 103, 109, 10A, 10B, 10F, 114, 116, 11B, 120, 125, 12B, 136, 138, 13D, 140, 14B, 14C, 151, 156, 15D, 160, 162, 165, 16E, 173, 177, 178, 17E, 184, 185, 18B, 192, 197, 198, 19E, 1A1, 1A7, 1AC, 1AE, 1B1, 1B2, 1BD, 1C6, 1C7, 1C8, 1D1, 1D8, 1DA, 1DD, 1E2, 1E9, 1ED, 1FB, 1FF.
$(10, 312, 1, 3)$	7	70, E, F; 0, 1, 3, 4, 8, 11, 18, 1A, 1C, 1F, 20, 25, 25, 28, 2D, 37, 39, 3A, 3C, 3E, 44, 46, 49, 4D, 4E, 53, 56, 5B, 5E, 62, 65, 66, 6D, 6F, 70, 72, 73, 77, 7B.
$(10, 316, 1, 3)$	8	E0, 1F; 0, 4, B, D, 12, 13, 14, 17, 18, 19, 21, 25, 26, 28, 2A, 2B, 2C, 39, 3D, 3E, 3F, 42, 43, 48, 49, 4D, 4E, 51, 57, 5A, 5E, 63, 64, 67, 70, 72, 73, 74, 7B, 7D, 80, 81, 89, 8A, 8E, 8F, 95, 96, 98, 9C, A2, A6, AC, AF, B0, B1, B5, B6, BA, BB, BC, BF, C4, C5, C7, D0, D2, D3, D4, DB, DD, E1, E7, E8, E9, EA, EE, FD, FE.
$(10, 408, 1, 4)$	7	60, 50, 38; 2, 3, 5, 6, 7, A, B, E, F, 11, 12, 14, 15, 19, 1D, 20, 22, 27, 2C, 2D, 30, 31, 34, 3A, 3B, 3E, 40, 42, 49, 4B, 4C, 4E, 52, 55, 57, 58, 5C, 60, 61, 64, 67, 68, 69, 73, 76, 77, 78, 7B, 7D, 7E, 7F.

Table II. (Cont.)

$(11, 368, 1, 2)$	9	180, 17F; 7, C, 12, 19, 1F, 20, 21, 2A, 34, 35, 3F, 43, 44, 48, 4D, 4E, 51, 5C, 63, 6C, 76, 7A, 7B, 80, 81, 8B, 94, 95, 9E, A6, AD, B3, B8, BE, C2, CD, D7, DA, DB, E2, E5, E9, EC, EF, F0, FD, 106, 107, 10A, 118, 119, 11F, 120, 12A, 12D, 133, 134, 143, 14D, 150, 151, 156, 165, 169, 16C, 16F, 176, 17A, 17D, 181, 18B, 18C, 192, 195, 1A6, 1A7, 1AB, 1B8, 1B9, 1BE, 1C4, 1C8, 1CD, 1CE, 1D7, 1DB, 1DC, 1E2, 1EC, 1F0, 1F1, 1F7.
$(12, 704, 1, 2)$	8	80, C0, A0, 7F; 9, A, 11, 17, 1E, 29, 2F, 31, 32, 3E, 43, 4A, 57, 58, 5D, 63, 6F, 72, 78, 7D, 80, 84, 85, 86, 8C, 94, 9B, A3, AA, B7, B8, BD, C9, CF, D1, D2, DE, E0, E4, E5, E6, EC, F4, FB.
$(12, 1344, 1, 4)$	7	60, 50, 28, 18, 47; 4, 7, 9, A, C, F, 10, 15, 16, 1B, 21, 22, 29, 2A, 2C, 30, 33, 35, 36, 38, 3D, 3E, 40, 41, 42, 44, 49, 4A, 4F, 53, 5C, 5D, 5E, 60, 67, 6F, 73, 75, 76, 78, 7B, 7C.
$(13, 1920, 1, 3)$	8	E0, D0, B0, 70, CF; 1, A, 10, 17, 18, 19, 1A, 1C, 21, 22, 26, 2D, 2F, 39, 3E, 43, 44, 46, 4B, 4D, 56, 58, 60, 67, 6D, 71, 72, 75, 7E, 7F, 81, 82, 85, 8E, 8F, 92, 9C, A4, A5, AB, B3, B4, B6, BB, BD, C3, C8, D0, D7, D8, D9, DA, DC, E3, E4, E5, EB, EE, F8, FF.
$(13, 2528, 1, 4)$	8	C0, A0, 60, E0, 9F; 1, 3, 5, 6, 7, 8, B, D, F, 16, 18, 19, 22, 24, 29, 2C, 31, 32, 35, 3A, 3F, 43, 47, 4C, 4E, 50, 53, 57, 58, 5C, 5D, 62, 67, 69, 6A, 70, 72, 7C, 7D, 7F, 81, 85, 86, 88, 89, 8A, 8B, 8D, 96, 97, 99, A0, A7, A9, AE, B0, B3, B4, BB, BC, C2, C4, C9, CF, D1, D4, D5, DA, DE, DF, E0, E4, E9, EE, F3, F5, FA, FB, FE.
$(14, 3712, 1, 3)$	8	C0, A0, 90, 70, F0, 6C; 0, 2, 3, 4, 7, 1A, 1D, 20, 2E, 2F, 31, 35, 38, 3B, 3C, 3F, 49, 4B, 51, 55, 5A, 5E, 60, 63, 64, 65, 67, 7A, 7D, 80, 83, 8D, 8E, 91, 92, 9C, A4, A9, B2, B6, B9, BD, C5, C6, CE, D2, D6, D8, DB, DC, DF, E4, E7, E9, EA, F3, F7, F8.
$(14, 4864, 1, 4)$	7	40, 20, 60, 10, 70, 6E, 11; 0, 3, 8, D, 15, 16, 1B, 1E, 24, 2A, 31, 34, 35, 37, 39, 3A, 3B, 3F, 41, 42, 43, 47, 49, 4C, 4D, 4F, 52, 5C, 63, 66, 68, 6D, 6E, 70, 75, 78, 7B, 7E.
$(8, 24, 2, 3)$	7	7F; 11, 1F, 20, 2E, 42, 44, 48, 67, 6B, 6D, 70, 7E.
$(11, 100, 2, 3)$	9	1C0, 3E; 44, 48, 4F, 57, 6A, 7F, 85, 89, 8E, 96, AB, B0, BE, 113, 118, 122, 124, 131, 13D, 1D2, 1D9, 1E3, 1E5, 1F0, 1FC.
$(13, 240, 2, 2)$	10	3C0, 338, 307; A, D, 3E, 4D, 79, 7E, BD, CE, 116, 126, 155, 165, 193, 1A0, 1D0, 1E3, 216, 265, 293, 2A0, 2D0, 2E3, 309, 33C, 33F, 34C, 34F, 37A, 38E, 3FD.
$(13, 336, 2, 3)$	9	1C0, 1B0, 168, 154; 3, 6, 19, 31, 3D, C7, CA, D1, ED, EF, F6, 148, 14B, 160, 176, 17A, 194, 19A, 1A4, 1B7, 1BC.
$(10, 18, 3, 2)$	9	1F0; 28, 91, F1, 153, 155, 1B6, 1BE, 1BF, 1C8.
$(10, 30, 3, 4)$	9	1FC; 18, 1B, 23, 29, 31, 46, 7E, C2, FA, 100, 149, 151, 161, 186, 1BE.

Table II. (Cont.)

$(11, 24, 3, 2)$	10	3FF; 5, 42, 57, CF, 12B, 1A6, 1B3, 1E1, 27C, 2A3, 347, 398.
$(11, 36, 3, 3)$	10	3F0; 3, 50, 86, BC, 139, 163, 1D5, 1DA, 1E0, 22F, 273, 2CC, 2F9, 315, 31A, 346, 39F, 3AC.
$(14, 120, 3, 2)$	11	7C0, 73C, 6B3; DA, DC, E7, E8, 15E, 161, 17E, 287, 298, 2A7, 2B7, 2B8, 301, 32E, 331.
$(13, 26, 4, 2)$	12	FFF; C, F6, 210, 25C, 2A6, 2EA, 31E, 3E4, 562, 582, 598, 63C, 6C6.

References

[1] A. Blokhuis and C. W. H. Lam, More coverings by rook domains, *J. Combin. Theory Ser. A* **36** (1984), 240–244.

[2] W. Chen and D. Li, Lower bounds for multiple covering codes, preprint.

[3] R. F. Clayton, "Multiple Packings and Coverings in Algebraic Coding Theory," Ph.D. dissertation, University of California, Los Angeles, 1987.

[4] G. D. Cohen, I. S. Honkala, S. N. Litsyn, and H. F. Mattson, Jr., Weighted coverings and packings, *IEEE Trans. Inform. Theory*, submitted for publication.

[5] F. Glover, Tabu search—Part I, *ORSA J. Comput.* **1** (1989), 190–206.

[6] H. O. Hämäläinen, I. S. Honkala, M. K. Kaikkonen, and S. N. Litsyn, Bounds for binary multiple covering codes, *Des. Codes Cryptogr.* **3** (1993), 251–275.

[7] H. O. Hämäläinen, I. S. Honkala, S. N. Litsyn, and P. R. J. Östergård, Bounds for binary codes that are multiple coverings of the farthest-off points, *SIAM J. Discrete Math.* **8** (1995), to appear.

[8] I. S. Honkala, On the normality of multiple covering codes, *Discrete Math.* **125** (1994), 229–239.

[9] I. S. Honkala and P. R. J. Östergård, Applications in code design, in E. H. L. Aarts and J. K. Lenstra (eds.), "Local Search Algorithms," Wiley, New York, to appear.

[10] G. A. Kabatyanskii and V. I. Panchenko, Unit sphere packings and coverings of the Hamming space, (in Russian), *Probl. Peredach. Inform.* **24** (1988), 3–16. English translation in *Probl. Inform. Trans.* **24** (1988), 261–272.

[11] P. J. M. van Laarhoven, E. H. L. Aarts, J. H. van Lint and L. T. Wille, New upper bounds for the football pool problem for 6, 7 and 8 matches, *J. Combin. Theory Ser. A* **52** (1989), 304–312.

[12] D. Li and W. Chen, New lower bounds for binary covering codes, *IEEE Trans. Inform. Theory* **40** (1994), 1122–1129.

[13] P. R. J. Östergård, New upper bounds for the football pool problem for 11 and 12 matches, *J. Combin. Theory Ser. A* **67** (1994), 161–168.

[14] P. R. J. Östergård, Constructing covering codes by tabu search, *Discrete Appl. Math.*, submitted for publication.

[15] P. R. J. Östergård and M. K. Kaikkonen, A table of binary covering codes, *IEEE Trans. Inform. Theory*, submitted for publication.

[16] G. J. M. van Wee, G. D. Cohen, and S. N. Litsyn, A note on perfect multiple coverings of the Hamming space, *IEEE Trans. Inform. Theory* **37** (1991), 678–682.

[17] L. T. Wille, The football pool problem for 6 matches: a new upper bound obtained by simulated annealing, *J. Combin. Theory Ser. A* **45** (1987), 171–177.

(Received 21/12/94)