mathematics

Article
Maximal $(v, k, 2, 1)$ Optical Orthogonal Codes with $k = 6$ and 7 and Small Lengths

Tsonka Baicheva *,† and Svetlana Topalova †

Institute of Mathematics and Informatics, Bulgarian Academy of Sciences, 1113 Sofia, Bulgaria; svetlana@math.bas.bg
* Correspondence: tsonka@math.bas.bg
† These authors contributed equally to this work.

Abstract: Optical orthogonal codes (OOCs) are used in optical code division multiple access systems to allow a large number of users to communicate simultaneously with a low error probability. The number of simultaneous users is at most as big as the number of codewords of such a code. We consider $(v, k, 2, 1)$-OOCs, namely OOCs with length v, weight k, auto-correlation 2, and cross-correlation 1. An upper bound $B_0(v, k, 2, 1)$ on the maximal number of codewords of such an OOC was derived in 1995. The number of codes that meet this bound, however, is very small. For $k \leq 5$, the $(v, k, 2, 1)$-OOCs have already been thoroughly studied by many authors, and new upper bounds were derived for $(v, 4, 2, 1)$ in 2011, and for $(v, 5, 2, 1)$ in 2012. In the present paper, we determine constructively the maximal size of $(v, 6, 2, 1)$- and $(v, 7, 2, 1)$-OOCs for $v \leq 165$ and $v \leq 153$, respectively. Using the types of the possible codewords, we calculate an upper bound $B_1(v, k, 2, 1) \leq B_0(v, k, 2, 1)$ on the code size of $(v, 6, 2, 1)$- and $(v, 7, 2, 1)$-OOCs for each length $v \leq 720$ and $v \leq 340$, respectively.

Keywords: optical orthogonal code; construction; optimal; code division multiple access system

MSC: 94B60

Citation: Baicheva, T.; Topalova, S. Maximal $(v, k, 2, 1)$ Optical Orthogonal Codes with $k = 6$ and 7 and Small Lengths. *Mathematics* **2023**, *11*, 2457. https://doi.org/10.3390/math11112457

Academic Editor: Christos G. Massouros

Received: 10 April 2023
Revised: 11 May 2023
Accepted: 18 May 2023
Published: 26 May 2023

1. Introduction
1.1. Optical Orthogonal Codes

Optical orthogonal codes (OOCs) were proposed by Chung, Salehi, and Wei [1] as a multiple access technique for optical fibre networks. These codes can be used in a great variety of wide-band code division multiple access environments, enabling a large number of users to transmit information asynchronously, efficiently, and reliably. They can also have applications in mobile radio, frequency-hopping spread spectrum communications, radar, sonar signal design, etc. This has motivated the wide study of OOCs, and many constructions and bounds about OOCs with particular parameters are known.

OOCs with equal auto- and cross-correlation constraints were studied first [2–8]. Yang and Fuja showed in [9] that a significant increase in the maximal number of codewords (for the given parameters) is possible by letting the auto-correlation constraint exceed the cross-correlation constraint and that, for a given performance requirement, the OOC may be one with unequal constraints.

OOCs have multiple relations to other combinatorial structures, such as partial designs, difference families, and other types of codes [2,6,7,10,11]. The OOCs that are studied in the present paper can also be considered as constant-weight unequal error protection codes with two levels of protection [9,12].

1.2. Basic Definitions and Notations

For the basic concepts and notations concerning optical orthogonal codes, we followed [13,14]. We denote by Z_v the ring of integers modulo v. A $(v, k, \lambda_a, \lambda_c)$-optical orthogonal code (OOC) is a set $\mathbf{C} \subseteq \{0,1\}^v$ of binary vectors of length v called *codewords*, all of Hamming weight k (with k nonzero coordinates), such that two arbitrary cyclic shifts x', x'' of a codeword $x \in \mathbf{C}$ intersect in at most λ_a coordinates and two arbitrary cyclic shifts x', y' of any distinct codewords $x, y \in \mathbf{C}$ intersect in at most λ_c coordinates. For our purposes, however, it is much more convenient to consider the set of indexes of the nonzero coordinates of a codeword and the following definition of an OOC.

Definition 1 ([13])**.** *A $(v, k, \lambda_a, \lambda_c)$-OOC can be defined as a collection $\mathcal{C} = \{C_1, \ldots, C_s\}$ of k-subsets (codeword-sets) of Z_v such that any two distinct translates of a codeword-set share at most λ_a elements, while any two translates of two distinct codeword-sets share at most λ_c elements:*

$$|C_i \cap (C_i + t)| \leq \lambda_a, \;\; 1 \leq i \leq s, \;\; 1 \leq t \leq v-1 \qquad (1)$$

$$|C_i \cap (C_j + t)| \leq \lambda_c, \;\; 1 \leq i < j \leq s, \;\; 0 \leq t \leq v-1. \qquad (2)$$

Condition (1) is called the auto-correlation property and (2) the cross-correlation property. The integers v and k are called the *length* and the *weight* of the code. The *size* of $\mathcal{C}$ is the number s of its codeword-sets. A (v, k, λ, λ)-OOC is also denoted by (v, k, λ)-OOC.

Let us consider communication via an optical network with a code division multiple access system, where s users transmit information simultaneously. Each of the s codewords of the OOC is assigned to one user of the network. The correlation constraints make it possible for a user to start a successful transmission at any time. At the transmitting end, each information bit is encoded into a frame of v optical chips, and each user transmits data only to k chips (according to the nonzero coordinates of the assigned codeword).

Consider a codeword-set $C = \{c_1, c_2, \ldots, c_k\}$. Denote by ΔC the multiset of the values of the differences $c_i - c_j$, $i \neq j$, $i, j = 1, 2, \ldots, k$. The auto-correlation property means that at most λ_a differences are the same. Denote by $\Delta' C$ the underlying set of ΔC.

Definition 2 ([13])**.** *The type of C is the number of elements of $\Delta' C$, i.e., the number of different values of its differences. The type of a codeword is the type of the codeword-set corresponding to it.*

If $\lambda_c = 1$, the cross-correlation property means that $\Delta C_1 \bigcap \Delta C_2 = \emptyset$ for two distinct codeword-sets C_1 and C_2 of the $(v, k, \lambda_a, 1)$-OOC. When we construct OOCs with cross-correlation $\lambda_c = 1$, we choose the codewords in such a way that their difference sets do not intersect. That is why, if we are only interested in the OOC existence problem for some parameters, we can use the following definition.

Definition 3. *Two codeword-sets C_1 and C_2 (and their corresponding codewords) are equivalent if $\Delta'(C_1) = \Delta'(C_2)$.*

An example of a $(v, 6, 2, 1)$-OOC is presented in Figure 1.

$\mathcal{C}$	$\Delta' C$	Type
{0,1,7,11,32,36}	1 4 6 7 10 11 13 17 21 25 29 31 32 35 36 38 41	17
{0,3,8,22,27,30}	3 5 8 12 14 15 18 19 20 22 23 24 27 28 30 34 37 39	18

Figure 1. Example of a $(v, 6, 2, 1)$-OOC.

Definition 4 ([9])**.** *An OOC is optimal if its size reaches a parameter-dependent upper bound.*

Definition 5 ([15])**.** *Two $(v, k, \lambda_a, \lambda_c)$ optical orthogonal codes $\mathcal{C}$ and $\mathcal{C}'$ are multiplier equivalent if they can be obtained from one another by an automorphism of Z_v and the replacement of codeword-sets by some of their translates.*

OOCs with $\lambda_a \neq \lambda_c$ were first investigated in [9]. There are already several papers on $(v, 4, 2, 1)$- and $(v, 5, 2, 1)$-OOCs [13–18].

A $(v,k,2,1)$-OOC can have codewords of type $k(k-1)/2 \leq T \leq k(k-1)$. Each difference should appear in at most one codeword, and there are $v-1$ differences from Z_v. That is why a natural upper bound on the maximum size $M(v,k,2,1)$ of a $(v,k,2,1)$-OOC (first obtained in [9]) can be derived by supposing that all the codewords of the OOC are of the smallest possible type $k(k-1)/2$, namely:

$$M(v,k,2,1) \leq B_0(v,k,2,1) = \left\lfloor \frac{2(v-1)}{k(k-1)} \right\rfloor.$$

Further results, however, show that this bound is attained by a very small number of codes. There are almost no optimal codes with respect to it. That is why better bounds have been derived for $k=4$ [13] and $k=5$ [14]. The next two cases $k=6$ and $k=7$ are of practical importance as well, but have not yet been explicitly considered.

1.3. The Present Paper

In the present work, we studied the properties of $(v,k,2,1)$-OOCs with $k=6$ or 7. Our investigation was computer-aided. We used our own software written for this particular problem in C++. For the smallest lengths, we found all codes up to multiplier equivalence. For bigger lengths, we determined the number of codewords in a maximal code, and finally, for all lengths up to 720 for $k=6$ and 340 for $k=7$, we calculated an upper bound $B_1(v,k,2,1) \leq B_0(v,k,2,1)$ on the size of a maximal code.

Section 2 describes the methods that were used; the results are described in Section 3, and the conclusion and open problems are the subjects of Section 4.

2. Methods

2.1. The Main Tasks

The computer algorithms that we used were based on backtrack search (which is of exponential complexity) and cannot be used for very big lengths. That is why we applied different techniques for the study of the codes in different length ranges. We considered codes with at least two codewords. From the existing upper bound $B_0(v,k,2,1)$, you can see that a $(v,k,2,1)$-OOC of size two has a length at least $V_0 = 31$ for $k=6$ and at least $V_0 = 43$ for $k=7$. That is why we only considered lengths greater than V_0. Define $V_1 = 100$, $V_2 = 165$, $V_3 = 720$ for $k=6$, and $V_1 = 94$, $V_2 = 153$, $V_3 = 340$ for $k=7$. For $V_0 \leq v \leq V_1$, we constructed all (up to multiplier equivalence) OOCs with the maximal number of codewords (with the exception of six code lengths, for which we constructed part of the OOCs). For $V_1 < v \leq V_2$, we found the exact size of the maximal codes by constructing at least one OOC with these parameters. For all $v \leq V_3$, we calculated an upper bound $B_1(v,k,2,1)$ on the size of the maximal codes. We, first of all, found this upper bound, because it further helped us to construct the maximal codes for $V_0 < v \leq V_2$.

We explain here how $B_1(v,k,2,1)$ was calculated, how the exact size of the maximal codes was determined, and how the codes with the smallest lengths were classified. We have to start, however, with a very brief description of our algorithm for the classification of OOCs. The details on it can be found in [15]. We outline here only the main ideas in order to further show how we used it in the present investigation.

2.2. Classification Algorithm

2.2.1. Preliminaries

1. Lexicographic order:
   We assume that $c_1 < c_2 < \ldots < c_k$ for each codeword-set $C = \{c_1, c_2, \ldots, c_k\}$ and define a lexicographic order on the codeword-sets implying that: $C' = \{c'_1, c'_2, \ldots, c'_k\}$ is lexicographically smaller than $C'' = \{c''_1, c''_2, \ldots, c''_k\}$ if the type of C' is smaller than that of C'' or if the types of the two codewords are the same, and $c'_i = c''_i$ for $i < j$ and $c'_j < c''_j$ for some j.
2. Assume $c_1 = 0$:

If a codeword-set $C \in \mathcal{C}$ is replaced by a translate $C + t \in \mathcal{C}$, an equivalent OOC is obtained. That is why, without loss of generality, we assume that each codeword-set is lexicographically smaller than the codeword-sets of its translates. This means that $c_1 = 0$.

3. Array of possible codeword-sets:
Before the search starts, an array is constructed, which contains all possible codewords, namely all k-subsets of Z_v that satisfy the auto-correlation property and are smaller than all their translates. They are found in lexicographic order. The automorphisms of Z_v are applied to each constructed codeword-set. If some automorphism maps the current codeword-set to a smaller set, the current set is not added, because it is already somewhere in the array. If the current set is added to the array, the codeword-sets to which it is mapped by the automorphisms of Z_v are added right after that, and this makes the tests for the multiplier equivalence of partial solutions very fast.

2.2.2. Exhaustive Backtrack Search

After the construction of the array, a backtrack search is applied to choose the codeword-sets of the OOC among all these possibilities for them. The first codeword-set is chosen in all multiplier inequivalent ways, and for each of them, ChooseCodeword(2, *num* + 1) is called to add the next codeword-sets to the OOC in all possible ways. Here, *num* is the number of the first chosen codeword-set in the array of all possible codeword-sets. The code segment presented below shows how the r-th codeword of the OOC is chosen in all possible ways. In it, *ALL* is the number of all possible codeword-sets, and s the size of the constructed OOCs.

```
void ChooseCodeword(int r, int start)
{
    for(int i=start; i<=ALL; i++)
    {
        if( NotPossible(r, i) ) continue;
        if( TypeNotOK(r, i) ) continue;
        if( NotNew(r, i) ) continue;
        AddCodeword(r, i);
        if(r==s) WriteOOC();
        else ChooseCodeword(r+1, i);
        TakeCodeword(r, i);
    }
}
```

The function *NotPossible* returns true if the set of differences of the i-th codeword-set contains differences that are already covered by the previously added codewords, and *NotNew* returns true if the code of these r codewords can be mapped by some automorphisms of Z_v to a lexicographically smaller one. Let T_r be the type of the r-th chosen codeword-set, and let d_r be the number of distinct differences covered by the r sets. We only look for codes with s codeword-sets. Because of the lexicographic order, the type of the remaining codeword-sets (in the array we choose them from) is at least as big as that of the r-th chosen one. That is why $d_r + (s - r)T_r \leq v - 1$. If this does not hold, *TypeNotOK* returns true. The functions *AddCodeword* and *TakeCodeword* update the set of differences covered by the already chosen codeword-sets, and *WriteCodeword* saves the constructed OOCs.

2.3. *The Upper Bound* $B_1(v, k, 2, 1)$

There are only a small number of codewords of the three smallest types, namely with $k(k-1)/2$, $k(k-1)/2 + 1$, and $k(k-1)/2 + 2$ distinct differences. We constructed (for each considered length $V_0 \leq v \leq V_3$) all the codes that have only codewords of types smaller than $k(k-1)/2 + 3$. We did this using the algorithm for the classification of OOCs (Section 2.2), but with an array of possible codeword-sets only of the three smallest types. We established that such codes have at most three codewords for $V_0 \leq v \leq V_3$. Consider any $(v,k,2,1)$-OOC, and denote by m the maximum possible number of its codewords of the three smallest types and by $d_{m\,min}$ the minimum number of differences covered by m such codewords. Then,

$$M(v,k,2,1) \leq B_1(v,k,2,1) = m + \left\lfloor \frac{v-1-d_{m\,min}}{\frac{k(k-1)}{2}+3} \right\rfloor.$$

We further obtain an upper bound $T_{B_1\,max}(v,k,2,1)$ on the type of codewords in a code with size $B_1(v,k,2,1)$ by supposing that all but one of its codewords are of the smallest possible types. The number u of the differences that are not covered by these $B_1(v,k,2,1)-1$ codewords is

$$u = v - 1 - d_{m\,min} - (B_1(v,k,2,1) - m - 1)\left(\frac{k(k-1)}{2} + 3\right).$$

If $u \geq k(k-1)$, the last codeword can be of the greatest possible type and $T_{B_1\,max}(v,k,2,1) = k(k-1)$. If $u < k(k-1)$, then $T_{B_1\,max}(v,k,2,1) = u$. The value of $T_{B_1\,max}(v,k,2,1)$ is very important for the determination of $M(v,k,2,1)$.

2.4. *The Maximum Number of Codewords of a* $(v,k,2,1)$*-OOC*

To determine $M(v,k,2,1)$, we have to construct a code with $B_1(v,k,2,1)$ codewords or to prove by exhaustive search that such a code does not exist. If we have proven that an OOC with $B_1(v,k,2,1)$ codewords does not exist, we construct a code with $B_1(v,k,2,1)-1$ codewords. We used the algorithm for the classification of OOCs (Section 2.2), but with an array of possible codeword-sets that contains only sets that are mutually inequivalent by Definition 3 and have types less than $T_{B_1\,max}(v,k,2,1)$. The value of $T_{B_1\,max}(v,k,2,1)$ is usually relatively small when an OOC with $B_1(v,k,2,1)$ codewords does not exist, and this makes it possible to prove its nonexistence by exhaustive backtrack search.

2.5. *Parallel Implementation*

The most-difficult cases for proving nonexistence were run on the powerful multiprocessor computing system Avitohol of the Bulgarian Academy of Sciences (see the acknowledgement at the end of the paper). For that purpose, we developed a parallel implementation of the classification algorithm. Its main idea is that each process obtains all nonequivalent solutions for the first two codewords, but extends to codes only part of them. For that purpose, we assigned consecutive numbers to the solutions of size two and computed the residues R of these numbers modulo the number of processes. The process with number P extends only solutions with $R = P$. There is a great number of solutions for the first two codewords, and therefore, the computing times of the different processes did not differ very much.

3. Bound, Maximum Size, and Classification Results

The classification results about the maximal $(v,k,2,1)$-OOCs with $k = 6$ and 7 and small lengths can be used in direct practical applications, because the access to all multiplier inequivalent maximal codes for a given length and number of users allows easily choosing the most-appropriate OOC for a given application with no need for any additional, sometimes complicated, mathematical computations. The classification results for the smallest lengths are presented in Tables 1 and 2. Only OOCs with at least two codewords were included. For each length, we give the values of the previously known bound B_0, the bound B_1 that we obtained, the size M of the maximal codes, and the number of multiplier-inequivalent OOCs.

Table 1. Maximal $(v, 6, 2, 1)$-OOCs with at least two codewords and $v \leq 100$.

v	B_0	B_1	M	OOCs
40	2	2	2	1
42	2	2	2	1
44	2	2	2	1
45	2	2	2	2
46	3	2	2	10
47	3	2	2	7
48	3	2	2	58
49	3	2	2	33
50	3	2	2	165
51	3	2	2	200
52	3	2	2	506
53	3	2	2	433
54	3	3	2	2251
55	3	3	2	1967
56	3	3	2	6246
57	3	3	2	6944
58	3	3	2	15,874
59	3	3	2	12,861
60	3	3	3	1
61	4	3	3	2
62	4	3	3	9
63	4	3	3	10
64	4	3	3	52
65	4	3	3	42
66	4	3	3	313
67	4	3	3	186
68	4	3	3	987
69	4	3	3	1250
70	4	3	3	5654
71	4	3	3	3477
72	4	4	3	21,487
73	4	4	3	13,547
74	4	4	4	1
75	4	4	3	91,956
76	5	4	3	217,428
77	5	4	4	1
78	5	4	4	6
79	5	4	4	10
80	5	4	4	52
81	5	4	4	72
82	5	4	4	428
83	5	4	4	320
84	5	4	4	3734
85	5	4	4	2510
86	5	4	4	12,360
87	5	4	4	13,035
88	5	4	4	65,033
89	5	4	4	46,355
90	5	5	4	$\geq$20,925

Table 1. *Cont.*

v	B_0	B_1	M	OOCs
91	6	5	4	≥5442
92	6	5	4	≥26,215
93	6	5	5	3
94	6	5	5	12
95	6	5	5	18
96	6	5	5	106
97	6	5	5	95
98	6	5	5	1150
99	6	5	5	934
100	6	5	5	≥1165

Table 2. Maximal $(v, 7, 2, 1)$-OOCs with at least two codewords and $v \leq 94$.

v	B_0	B_1	M	OOCs
67	3	2	2	34
68	3	2	2	108
69	3	2	2	132
70	3	2	2	487
71	3	2	2	384
72	3	3	2	1497
73	3	3	2	1208
74	3	3	2	3735
75	3	3	2	6087
76	3	3	2	12,432
77	3	3	2	13,506
78	3	3	2	52,070
79	3	3	2	32,364
80	3	3	2	132,413
81	3	3	2	125,433
82	3	3	2	287,830
83	3	3	2	240,606
84	3	3	2	1,279,965
85	4	3	3	1
86	4	3	3	1
87	4	3	3	5
88	4	3	3	2
89	4	3	3	8
90	4	3	3	23
91	4	3	3	44
92	4	3	3	84
93	4	3	3	≥159
94	4	3	3	≥136

The codes and information on the different types of codes (with respect to the types of codewords) are given as the Supplementary Materials.

Example: There are five inequivalent $(63, 7, 2, 1)$-OOCs of three types, which are presented as:

Types of codes by differences
0)3: 24-2
1)1: 24-1 30-1
2)1: 24-1 32-1

This means that there are 3 codes of Type 0 with 2 codewords of Type 24, 1 code of Type 1 having one codeword of Type 24 and one of Type 30, and 1 code of Type 2 with one codeword of Type 24 and one of Type 32.

The size of the maximal $(v, 6, 2, 1)$ codes for $v \leq 165$ and $(v, 7, 2, 1)$ for $v \leq 153$ is presented in Tables 3 and 4.

Table 3. The size of maximal $(v, 6, 2, 1)$-OOCs with $101 \leq v \leq 165$.

v	B_0	B_1	M
101	6	5	5
102	6	5	5
103	6	5	5
104	6	5	5
105	6	5	5
106	7	5	5
107	7	5	5
108	7	6	5
109	7	6	6
110	7	6	6
111	7	6	6
112	7	6	6
113	7	6	6
114	7	6	6
115	7	6	6
116	7	6	6
117	7	6	6
118	7	6	6
119	7	6	6
120	7	6	6
121	8	6	6
122	8	6	6
123	8	6	6
124	8	6	6
125	8	6	6
126	8	7	7
127	8	7	7
128	8	7	7
129	8	7	7
130	8	7	7
131	8	7	7
132	8	7	7
133	8	7	7
134	8	7	7
135	8	7	7
136	9	7	7
137	9	7	7
138	9	7	7
139	9	7	7
140	9	7	7
141	9	7	7
142	9	7	7
143	9	7	7
144	9	8	8
145	9	8	8
146	9	8	8
147	9	8	8
148	9	8	8
149	9	8	8
150	9	8	8

Table 3. *Cont.*

v	B_0	B_1	M
151	10	8	8
152	10	8	8
153	10	8	8
154	10	8	8
155	10	8	8
156	10	8	8
157	10	8	8
158	10	8	8
159	10	8	8
160	10	9	9
161	10	8	8
162	10	9	9
163	10	9	9
164	10	9	9
165	10	9	9

Table 4. The size of maximal $(v, 7, 2, 1)$-OOCs with $95 \leq v \leq 153$.

v	B_0	B_1	M
95	4	3	3
96	4	4	3
97	4	4	3
98	4	4	3
99	4	4	3
100	4	4	3
101	4	4	3
102	4	4	3
103	4	4	3
104	4	4	3
105	4	4	3
106	5	4	3
107	5	4	3
108	5	4	3
109	5	4	4
110	5	4	4
111	5	4	4
112	5	4	4
113	5	4	4
114	5	4	4
115	5	4	4
116	5	4	4
117	5	4	4
118	5	4	4
119	5	4	4
120	5	5	4
121	5	5	4
122	5	5	4
123	5	5	4
124	5	5	4
125	5	5	4
126	5	5	4
127	6	5	4
128	6	5	4

Table 4. *Cont.*

v	B_0	B_1	M
129	6	5	4
130	6	5	4
131	6	5	4
132	6	5	4
133	6	5	5
134	6	5	5
135	6	5	5
136	6	5	5
137	6	5	5
138	6	5	5
139	6	5	5
140	6	5	5
141	6	5	5
142	6	5	5
143	6	5	5
144	6	6	5
145	6	6	5
146	6	6	5
147	6	6	5
148	7	6	5
149	7	6	5
150	7	6	5
151	7	6	5
152	7	6	5
153	7	6	5

From Tables 1–4, one can see that only several maximal OOCs attain the bound B_0, $M(v,6,2,1) = B_1(v,6,2,1)$ in 87% of the OOCs, and $M(v,7,2,1) = B_1(v,7,2,1)$ in 44% of the codes.

Tables 5 and 6 present the bounds B_0 and B_1 for all $(v,6,2,1)$ codes for $v \leq 720$ and $(v,7,2,1)$ codes for $v \leq 340$.

Table 5. Bounds on the maximal size of a $(v,6,2,1)$-OOC with $31 \leq v \leq 720$.

v	B_0	B_1
31–35	2	1
36–45	2	2
46–53	3	2
54–60	3	3
61–71	4	3
72–75	4	4
76–89	5	4
90–90	5	5
91–105	6	5
106–107	7	5
108–120	7	6
121–125	8	6
126–135	8	7
136–143	9	7
144–150	9	8
151–159	10	8
160–160	10	9
161–161	10	8
162–165	10	9
166–179	11	9

Table 5. *Cont.*

v	B_0	B_1
180–180	11	10
181—195	12	10
196–197	13	10
198–210	13	11
211–215	14	11
216–225	14	12
226–233	15	12
234–240	15	13
241–251	16	13
252–255	16	14
256–269	17	14
270–270	17	15
271–285	18	15
286–287	19	15
288–300	19	16
301–303	20	16
304–304	20	17
305–305	20	16
306–315	20	17
316–322	21	17
323–330	21	18
331–339	22	18
340–340	22	19
341–341	22	18
342–345	22	19
346–359	23	19
360–360	23	20
361–375	24	20
376–377	25	20
378–390	25	21
391–395	26	21
396–405	26	22
406–413	27	22
414–420	27	23
421–431	28	23
432–435	28	24
436–449	29	24
450–450	29	25
451–465	30	25
466–467	31	25
468–480	31	26
481–485	32	26
486–495	32	27
496–503	33	27
504–510	33	28
511–519	34	28
520–520	34	29
521–521	34	28
522–525	34	29
526–539	35	29
540–540	35	30
541–555	36	30
556–557	37	30
558–570	37	31
571–575	38	31
576–585	38	32
586–593	39	32
594–600	39	33
601–611	40	33
612–615	40	34
616–629	41	34
630–630	41	35

Table 5. *Cont.*

v	B_0	B_1
631–645	42	35
646–646	43	36
647–647	43	35
648–660	43	36
661–664	44	36
665–675	44	37
676–683	45	37
684–690	45	38
691–699	46	38
700–700	46	39
701–701	46	38
702–705	46	39
706–719	47	39
720–720	47	40

Table 6. Bounds on the maximal size of a $(v, 7, 2, 1)$-OOC with $43 \leq v \leq 340$.

v	B_0	B_1
43–47	2	1
48–63	2	2
64–71	3	2
72–84	3	3
85–95	4	3
96–105	4	4
106–119	5	4
120–126	5	5
127–143	6	5
144–147	6	6
148–167	7	6
168–168	7	7
169–189	8	7
190–191	9	7
192–210	9	8
211–215	10	8
216–231	10	9
232–239	11	9
240–252	11	10
253–263	12	10
264–273	12	11
274–287	13	11
288–294	13	12
295–311	14	12
312–315	14	13
316–335	15	13
336–336	15	14
337–340	16	14

4. Conclusions and Remarks

In the considered length range, we observed the following:

- The OOCs contain only a few codewords of the three smallest types. For very small lengths, they are an important part of all codewords, but for bigger lengths, they comprise a really small part of all codewords and their effect on the maximal code size becomes almost negligible.
- $M(v, 6, 2, 1) = B_0(v, 6, 2, 1)$ for only six values of $40 \leq v \leq 165$ (40, 42, 44, 45, 60, 74).
- $M(v, 7, 2, 1) < B_0(v, 7, 2, 1)$ for all $67 \leq v \leq 153$.
- $M(v, 6, 2, 1) = B_1(v, 6, 2, 1)$ for 110 values of $40 \leq v \leq 165$.
- $M(v, 7, 2, 1) = B_1(v, 7, 2, 1)$ for 38 values of $67 \leq v \leq 153$.

- $B_1(v,6,2,1) < B_0(v,6,2,1)$ for all $v \geq 91$.
- $B_1(v,7,2,1) < B_0(v,7,2,1)$ for all $v \geq 169$.
- The bound we calculated can be approximated in the covered length range with:

$$B_1(v,6,2,1) \leq \left\lfloor \frac{v}{18} \right\rfloor + f(v)$$

where $f(v) = 1$ for $v \equiv 16$ and $v \equiv 17 \pmod{18}$ and $f(v) = 0$ for all the other values of v.

$$B_1(v,7,2,1) = \left\lfloor \frac{v}{24} \right\rfloor.$$

Our results are consistent with the previous results that we know, namely:

- All values of $M(v,k,2,1)$ and $B_1(v,k,2,1)$ obtained by us are never greater than the upper bound $B_0(v,k,2,1)$ from [9].
- The values of $B_1(v,6,2,1)$ obtained by us coincide with the OOCs constructed in [9].

To determine the maximum number of codewords in $(v,6,2,1)$- and $(v,7,2,1)$-OOCs or to find a tight upper bound on it remains an open problem for lengths v outside those that were considered in the present paper. The study of $(v,k,2,1)$-OOCs with $k > 7$ is an open problem for which our computer-aided approach is presently difficult to apply because it is based on backtracking (the backtrack search is of exponential complexity and can only be used for relatively small parameters). Future computer-aided constructions for bigger parameters will, most probably, use suitable restrictions or new theoretical results.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/math11112457/s1, Examples of maximal $(v,6,2,1)$- and $(v,7,2,1)$-OOCs.

Author Contributions: Conceptualisation, S.T. and T.B.; methodology, S.T. and T.B.; software, S.T.; validation, S.T. and T.B.; investigation, S.T. and T.B.; data curation, T.B.; writing—original draft preparation, S.T. and T.B.; writing—review and editing, S.T. and T.B. All authors have read and agreed to the published version of the manuscript.

Funding: T.B. was partially supported by the Bulgarian National Science Fund under Contract No. DO1-168/28.07.2022, and S.T. was partially supported by the Bulgarian National Science Fund under Contract No. KP-06-N32/2-2019.

Data Availability Statement: The size of the maximal $(v,6,2,1)$ codes for $v \leq 165$ and $(v,7,2,1)$ for $v \leq 153$ and one OOC for each of these lengths are available as Supplementary Materials in STables.pdf. Information on the different types of codes (with respect to the types of codewords) is given in the files S$(v,6,2,1)$ooc.txt and S$(v,7,2,1)$ooc.txt there. All classified $(v,6,2,1)$ and $(v,7,2,1)$ codes are available in S$(v,6,2,1)$.rar and S$(v,7,2,1)$.rar.

Acknowledgments: The authors would like to thank Marco Buratti (Sapienza Universita di Roma) for the discussions in a personal communication on the importance of the codewords of the smallest types. The authors acknowledge the provided access to the e-infrastructure of the NCHDC—part of the Bulgarian National Roadmap on RIs, with the financial support from Grant No. DO1-168/28.07.2022.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviation is used in this manuscript:

OOC Optical orthogonal codes

References

1. Chung, F.R.K.; Salehi, J.A.; Wei, V.K. Optical orthogonal codes: Design, analysis, and applications. *IEEE Trans. Inf. Theory* **1989**, *35*, 595–604. [CrossRef]
2. Moreno, O.; Zhang, Z.; Kumar, P.V.; Zinoviev, V.A. New constructions of optimal cyclically permutable constant weight codes. *IEEE Trans. Inf. Theory* **1995**, *41*, 448–455. [CrossRef]
3. Yin, J. Some combinatorial constructions for optical orthogonal codes. *Discret. Math.* **1998**, *185*, 201–219. [CrossRef]

4. Fuji-hara, R.; Miao, Y. Optical orthogonal codes: Their bounds and new optimal constructions. *IEEE Trans. Inf. Theory* **2000**, *46*, 2396–2406.
5. Chu, W.; Colbourn, C.J. Optimal $(n, 4, 2)$- OOC of small order. *Discret. Math.* **2004**, *279*, 163–172. [CrossRef]
6. Wang, X.; Chang, W. Further results on $(v, 4, 1)$-perfect difference families. *Discret. Math.* **2010**, *310*, 1995–2006. [CrossRef]
7. Baicheva, T.; Topalova, S. Classification of optimal $(v, 4, 1)$ binary cyclically permutable constant weight codes and cyclic $S(2, 4, v)$ designs with $v \leq 76$. *Probl. Inf. Transm.* **2011**, *47*, 224–231. [CrossRef]
8. Chu, W.; Golomb, S.W. A new recursive construction for optical orthogonal codes. *IEEE Trans. Inf. Theory* **2003**, *49*, 3072–3076.
9. Yang, G.C.; Fuja, T.E. Optical orthogonal codes with unequal auto- and cross-correlation constraints. *IEEE Trans. Inf. Theory* **1995**, *41*, 96–106. [CrossRef]
10. Buratti, M.; Pasotti, A. Further progress on difference families with block size 4 or 5. *Des. Codes Cryptogr.* **2010**, *56*, 1–20. [CrossRef]
11. Momihara, K. On cyclic $2(k-1)$-support $(n, k)_{k-1}$ difference families. *Finite Fields Their Appl.* **2009**, *15*, 415–427. [CrossRef]
12. Dunning, L.A.; Robbins, W.E. Optimal encoding of linear block codes for unequal error protection. *Inf. Control* **1978**, *37*, 150–177. [CrossRef]
13. Buratti, M.; Momihara, K.; Pasotti, A. New results on optimal $(v, 4, 2, 1)$ optical orthogonal codes. *Des. Codes Cryptogr.* **2011**, *58*, 89–109. [CrossRef]
14. Buratti, M.; Pasotti, A.; Wu, D. On optimal $(v, 5, 2, 1)$ optical orthogonal codes. *Des. Codes Cryptogr.* **2013**, *68*, 349–371. [CrossRef]
15. Baicheva, T.; Topalova, S. Optimal $(v, 4, 2, 1)$ optical orthogonal codes with small parameters. *J. Comb. Des.* **2012**, *20*, 142–160. [CrossRef]
16. Baicheva, T.; Topalova, S. Optimal $(v, 5, 2, 1)$ optical orthogonal codes of small v. *Appl. Algebra Eng. Commun. Comput.* **2013**, *24*, 165–177. [CrossRef]
17. Momihara, K.; Buratti, M. Bounds and constructions of optimal$(n, 4, 2, 1)$ optical orthogonal codes. *IEEE Trans. Inf. Theory* **2009**, *55*, 514–523. [CrossRef]
18. Wang, X.; Chang, Y. Further results on optimal $(v, 4, 2, 1)$-OOCs. *Discret. Math.* **2012**, *312*, 331–340. [CrossRef]