БЪЛГАРСКА АКАДЕМИЯ НА НАУКИТЕ
ИНСТИТУТ ПО МАТЕМАТИКА И
ИНФОРМАТИКА

Цонка Стефанова Байчева

ОПТИМАЛНИ КОДОВЕ ЗА КОНТРОЛ НА ГРЕШКИ И ОПТИЧНИ КОМУНИКАЦИИ

дисертация

за придобиване на научната степен
„доктор на науките“
по научна специалност
01.01.12 - информатика

2015 г.

Съдържание

Увод **4**

0.1 **Обзор на някои известни резултати** 5
0.2 **Структура на изложението** 10
0.3 **Апробация на резултатите** 17
0.4 **Авторска справка** 19

1 Основни понятия и предварителни резултати **21**

1.1 **Основни понятия от Теория на кодирането** 21
1.2 **Линейни кодове** 21
1.3 **Циклични кодове** 27
1.4 **Скъсени циклични (CRC) кодове** 28
1.5 **Граници за радиус на покритие на код** 29
1.6 **Минимален радиус на покритие на $[n, k]$ кодове** 32

2 Програмни реализации на основните методи за пресмятане на параметрите, определящи възможностите за контрол на грешки, на линейни кодове **33**

2.1 **Параметри, определящи възможностите за контрол на грешки, на линейни кодове** 33
2.2 **Сложност на задачата за пресмятане на минимално разстояние, радиус на покритие и тегловни разпределения на код** 34
2.3 **Алгоритми за пресмятане на тегловни разпределения и на радиус на покритие на код** 37
2.3.1 **Алгоритми за пресмятане на тегловни разпределения на линеен код** 38
2.3.2 **Алгоритми за пресмятане на радиус на покритие на линеен код** 42

3 Радиус на покритие на класове линейни кодове **46**
3.1 **Радиус на покритие на троичните негациклични кодове с дължини до 26** 47
3.2 **Двоични и троични квази-съвършени кодове с малки размерности** 50
3.3 **Минимален радиус на покритие на двоични линейни кодове с малки дължини** 62
3.4 **Минимален радиус на покритие на двоичните линейни кодове с размерност 6** 68
3.5 **Минимален радиус на покритие на троичните линейни кодове с размерност 4** 74

4 Поведение на шумозащитни кодове при откриване и коригиране на грешки **80**
4.1 **Поведение при откриване на грешки на двоични и троични линейни кодове с дължини до 33** 81
4.2 **Пресмятане на тегловното разпределение на съседните класове на циклични кодове** 84
4.3 **Поведение на троичният [13,7,5] квадратично-остатъчен код** 86
4.4 **Сравнение на поведението за контрол на грешки на класове линейни кодове** 92

5 Скъсени циклични (CRC) кодове **101**
5.1 **Поведение при откриване на грешки на стандартизирани CRC кодове** 102
5.2 **CRC кодове с 8 проверочни бита** 111
5.3 **CRC кодове с 16 проверочни бита** 115
5.4 **CRC кодове с до 10 проверочни бита** 119

6 Някои характеристики на шумозащитни кодове свързани с техните възможности за контрол на грешки **128**
6.1 **Нютонов радиус на покритие** 128
6.2 **LUEP кодове** 132
6.3 **Нормализирани двоични линейни кодове** 137

7 Оптични ортогонални кодове и свързани с тях комбинаторни структури **142**

7.1 **Класификация на оптимални $(v, 4, 2, 1)$ и $(v, 5, 2, 1)$ оптични ортогонални кодове с малки дължини** 143
7.1.1 **Класификация на оптимални $(v, 4, 2, 1)$ оптични ортогонални кодове** 147
7.1.2 **Класификация на оптимални $(v, 5, 2, 1)$ оптични ортогонални кодове** 151
7.2 **Оптимални $(v, 3, 1)$ двоични циклично - пермутационни константно - тегловни кодове с дължини до 61** 154
7.3 **Класификация на $(v, k, 1)$ циклични разностни множества с малки параметри** 158

А Радиуси на покритие на троичните негациклични кодове с дължини до 26 **163**

Б Минимален радиус на покритие на троичните циклични кодове с дължини до 40 **166**

В Подходящи двоични и троичини циклични, троични негациклични и двоични кодове с максимално минимално разстояние **169**

Г Тегловни разпределения на лидерите на съседни класове на троичните циклични кодове с дължини до 14 **174**

Д CRC полиноми от 8-ма степен. **176**

Е Стандартизирани CRC кодове и кодове с най-малки стойности на функцията P_{ue} с 16 проверочни бита. **178**

Ж CRC полиноми с до 5 проверочни бита. **180**

З Оптимални $(v, 4, 2, 1)$ и $(v, 5, 2, 1)$ оптични ортогонални кодове **184**

Литература **196**

Публикации по дисертацията **213**

Списък на цитирания **216**

Увод

За рождена дата на теорията на кодирането може да се счита излизането на класическата статия [178] на Шенон през 1948 година. Шумозащитните кодове са били разработени за да откриват и коригират грешките, които се появяват при предаването на дискретна информация по комуникационни канали или при нейното съхраняване на различни носители. Целта на кодирането е да се добави допълнителна информация към данните така, че да има възможност да бъде възстановено оригиналното съобщение, ако са възникнали не повече от предварително очакван брой грешки. На съвременния етап, теорията на кодирането е самостоятелна дисциплина, която използва резултати от различни области на математиката като алгебра, геометрия, комбинаторика, дискретна математика. Това позволява създаването на кодове с добри шумозащитни характеристики.

От гледна точка на практическото приложение на шумозащитните кодове и на кодовете за оптични комуникации е необходимо не само да се покаже съществуването на код с определени характеристики, но и да се конструира самия код. Интерес представлява също пълната класификация на кодове със зададени параметри, както и определянето на техните основни характеристики. Така става възможен избора на най-подходящия за всяка конкретна ситуация код. Особено актуални и интересни са тези резултати в последните години, когато все по-често се правят софтуерни реализации на кодирането и декодирането и това позволява да бъде избран най-точно отговарящия на изискванията на конкретното приложение код.

В огромната част от случаите, използването на чисто математически подходи за решаването на тези задачи не може да доведе до търсения резултат. Една възможност за решаването на такъв тип задачи е използването на компютър, което е свързано с възможностите му да извършва аритметични операции или да съпоставя данни и в зависимост от получения резултат да предприема едно или друго продължение на работата, а също и да съхранява, обработва и предоставя за използуване в най-разнообразен вид огромни обеми от данни. Директната им атака с компютър обаче позволява да се получат решения само за ограничени по обем входни данни, за които е невъзможно те да бъдат получени ръчно. Причината за това е във факта, че експоненциално нарастващата им трудност бързо стопява предимството на компютъра да работи многократно по-бързо от човека и да помни големи количества информация. За по-големи входни данни се налага за всяка отделна задача да се провеждат предварителни математически изследвания, които макар да не решават директно въпроса, силно съкращават възможностите,

които трябва да се изследват с компютър. Друга възможност е разработването на все по-бързи и икономични на памет програми, което изисква отличното познаване и умелото използване на свойствата на изследваните математически обекти и възможностите на компютъра. Така се оформя една все по-успешна хибридна математико-компютърна стратегия, с помощта на която бяха решени някои трудни алгоритмично разрешими задачи в различни области на математиката. Ще спомена построяването на някои ортогонални латински квадрати, доказателството за несъществуване на крайна проективна равнина от ред 10, доказателството на теоремата за четирите цвята.

У нас също се използва подобна стратегия. В последните 30-ина години група български специалисти от различни научни институти и висши учебни заведения, обединени в семинара по „Алгебрична и комбинаторна теория на кодирането“, получиха важни резултати с помощта на математико-компютърни методи за решаване на задачи от областта на теорията на кодирането.

Целта на настоящата работа е изследването и представянето на класове шумозащитни кодове имащи добри характеристики по отношение на откриване и коригиране на грешки, както и на оптични ортогонални кодове и свързаните с тях комбинаторни структури. Да се направят класификации на някои класове от най-широко използваните в практиката кодове и да се определят техни основни характеристики, свързани с възможностите им за контрол на грешки. Да се разработи удобна за използването на тези данни процедура, която да позволи избора на най-подходящ и ефективен за всяко конкретно практическо приложение код. Наред с това, да бъдат решени отворени проблеми от теорията на кодирането.

0.1 Обзор на някои известни резултати

Множеството от редици с равни дължини (*кодови думи*) от букви на азбуката $L = \{a_1, a_2, \ldots, a_s\}$ се нарича *блоков код* или просто *код* над азбуката L, дължината n на кодовите думи - *блокова дължина*, а броя на ненулевите елементи в кодовата дума - *тегло* на кодовата дума. Когато всички кодови думи на един блоков код имат едно и също тегло, кодът се нарича *константно-тегловен*.

Разстояние по Хеминг между две думи с дължина n над азбуката L се дефинира като броя на позициите, в които тези две думи се различават, а под *разстояние на дума* с дължина n над азбуката L *до код* се разбира най-малкото Хемингово разстояние на думата до кодова дума. *Минимално разстояние на кода* се нарича минималното разстояние между две различни кодови думи.

Радиусът на покритие на код е най-малкото цяло число R такова, че кълбата с радиус R, описани около кодовите думи, покриват цялото n-мерно векторно пространство над $GF(q)$. *Радиусът на сферичната опаковка* е максималната стойност на радиуса e на кълбата, описани около кодовите думи така, че тези кълба да нямат общи точки. Една сравнително проста долна граница за радиуса на покритие R е следната: $R \geq e$. Кодовете, които удовлетворяват тази граница с равенство, се наричат *съвършени*, а тези, за които радиусът на покритие надвишава с 1 радиуса на сферичната опаковка - *квази-съвършени*.

Най-често за азбука L се вземат елементите на някое крайно поле $GF(q)$. Ако думите на един код образуват линейно пространство с размерност k над полето $GF(q)$, кодът се нарича *q-ичен линеен код с размерност k*. Линеен код с дължина n, размерност k и минимално разстояние d над поле с q елемента ще означаваме с $[n, k, d]_q$. Тези кодове, поради ясната си математическа структура, имат ефективни кодиращи и декодиращи алгоритми, което ги прави предпочитани за много практически приложения. От друга страна, тези кодове имат връзки с други класически математически обекти като групи, графи, комбинаторни дизайни, крайни геометрии и други, така че тяхното изследване представлява интерес и от чисто теоретична гледна точка.

Радиусът на покритие е основен параметър на кода. Ако кодът се използва за поправяне на грешки и се декодира по метода на максималното правдоподобие, радиусът му на покритие е максималното тегло на коригируем вектор-грешка, а ако се използва за компресиране на данни, е мярка за максималното им изкривяване. Радиусът на покритие ни дава мярка и за това дали един код е максимален (към него да не могат да се прибавят още кодови думи без да се намали минималното му разстояние), което е възможно само ако радиусът на покритие е строго по- малък от минималното разстояние на кода.

Интересът към радиуса на покритие нараства значително след 1980 година и вече има натрупани значително количество изследвания по този въпрос. В излязлата през 1997 година книга „Covering Codes“ [47] са цитирани 714 работи, а авторите поддържат постоянно допълващ се списък в Интернет. Въпреки това, е известно сравнително малко за точните стойности на радиусите на покритие на основни фамилии кодове и в [48] като отворен проблем е поставено определянето на точните стойности на радиусите на покритие на класове кодове.

Редица автори са използвали компютър за решаването на този проблем. Радиусите на покритие на двоичните циклични кодове с дължини, не надминаващи 31, са пресметнати от Downie и Sloane [75], а на тези с дължини 33, 35 и 39 от Великова и Манев [192]. По-късно Dougherty и Janwa [74] пресмятат радиусите на покритие

на всички циклични кодове с дължини, не надминаващи 64 и размерности до 28. В [193] е даден начин за пресмятане на радиуса на покритие на някои двоични самодопълнителни кодове и е получена една горна граница за такива кодове. В [194] са разгледани двоични циклични кодове с дължина $n = u.v$, които се получават при следното разлагане на полинома $x^{uv} - 1 = (x-1)\left(\frac{x^u-1}{x-1}\right)\left(\frac{x^v-1}{x-1}\right) f_1(x)$ и u и v са две взаимно прости нечетни числа. За два от тези кодове са дадени горни и долни граници, а за останалите са пресметнати точните стойности на радиусите им на покритие. Радиусите на покритие на двоични линейни кодове, достигащи границата на Griesmer, са изследвани от Busschbach, Gerretzen и Tilborg [37] и Додунеков [67].

За радиусите на покритие на не двоични линейни кодове е известно малко. Известни са радиусите на покритие на кодовете на Рид-Соломон и на някои БЧХ кодове с конструктивно разстояние най-много 3 [48]. Великова [2] е пресметнала радиусите на покритие на кодовете над $GF(4)$ с дължина не надминаваща 15.

Функцията $t_q[n,k]$ се дефинира като минималния радиус на покритие на линейния код C, когато C се изменя в множеството от всички $[n,k]$ кодове над $GF(q)$ за фиксирани $n,\ k$ и q.

Тази функция е широко изследвана за кодове над $GF(2)$ и са получени много резултати. Cohen, Karpovsky, Mattson, Jr. и Schatz [48] и Graham и Sloane [103] са определили с помощта на компютърни пресмятания стойностите на $t_2[n,1]$, $t_2[n,2]$, $t_2[n,3]$, $t_2[n,4]$, $t_2[n,5]$ за всяко n и са построили таблица с граници за $t_2[n,k]$ за $n \leq 32$ в първата работа и за $n \leq 64$ - във втората. Всички резултати за функцията $t_2[n,k]$, получени след тези две работи, са обобщени в Таблица 7.1 от [47].

Граници и точни стойности за функцията $t_3[n,k]$ са дадени в таблица II от [11], а за $t_4[n,2]$ в [195].

Основна мярка за възможностите за откриване и коригиране на грешки на един шумозащитен код е неговото минимално разстояние. От теорията на кодирането е известно, че код с минимално разстояние d открива до $d-1$ и коригира до $t = \left\lfloor \frac{d-1}{2} \right\rfloor$ грешки. Следователно при фиксирана дължина n, най-добър ще е кодът с максималното минимално разстояние d_{max}. Достатъчно ли е обаче да успеем да конструираме един код с дължина n и минимално разстояние d_{max} и да сме сигурни, че това ще е най-подходящия избор? Оказва се, че това не е така, защото има и други важни мерки за възможностите за контрол на грешки на един шумозащитен код. Това са вероятността за неоткрита грешка P_{ue} и вероятността за коректно декодиране P_{corr}.

Ясно е, че когато искаме да изберем $[n,k,d]$ код за конкретно приложение,

най-добрият избор ще бъде код с минимална вероятност за неоткрита грешка P_{ue} (оптимален код) или с максимална вероятност за коректно декодиране P_{corr}. Много често обаче вероятността за грешка ε на комуникационния канал не е фиксирана величина, т.е. тя може да се променя по време на предаването на данни. Тогава код, който е оптимален за $\varepsilon' \neq \varepsilon$, може да не е оптимален за p. В такъв случай е удачно да имаме критерий, според който да определяме доколко даден код е подходящ за откриване и коригиране на грешки.

Ще казваме, че един код C е *t-подходящ* (или само подходящ, когато кодът се използва само за откриване на грешки) ако функцията P_{ue} е монотонна в целия интервал $[0, \frac{q-1}{q}]$ [132]. Дискретни достатъчни условия един $[n, k, d]_q$ код да бъде *t-подходящ* са представени в работите на Додунекова, Додунеков и Николова [3, 70, 71].

Посочените критерии и вероятностите P_{ue} и P_{corr} зависят от тегловото разпределение на кодовите думи, на думите в съседните класове и на теглата на лидерите на съседни класове на кода. За съжаление, тези разпределения са определени само за много малък брой кодове. Известни са тегловните разпределения на кодовите думи на съвършените кодове [145] (кодовете с повторение и техните дуални, кодовете на Хеминг и симплекс кодовете, кодовете на Golay), на кодовете на Reed-Muller от първи и втори ред [145], на някои техни подкодове [123], на коригиращите до 3 грешки БЧХ кодове [141], на МДР кодовете [168], на МДМ кодовете [84, 85, 163], на почти МДР кодовете [61, 68, 84]. Тегловните разпределения на съответните двоични кодове на някои кодове на Reed -Solomon са изследвани в [116, 125, 135, 170, 173, 174]

През 1978 г. Berlecamp, McEliece and Van Tilborg [14] показват, че определянето на минималното разстояние на код е NP-трудна задача, от което следва, че определянето на тегловото разпределение на код е NP-труден проблем. През 1984 година McLoughlin [154] показва, че задачата за определяне на горна граница за радиуса на покритие на линеен код е $\prod_2$-пълна, а оттук следва, че и определянето на тегловото разпределение на думите в съседните класове и на лидерите на съседни класове са също изчислително трудни задачи. Следователно определянето на тези тегловни разпределения за конкретни класове кодове е интересна, както от практическа, така и от изследователска гледна точка, задача. Основните резултати от изследванията по този проблем са събрани в „Handbook on Coding Theory“ [169] и в монографиите „Covering Codes“ [47], „Error detecting codes“ [132] и „Codes for error detection“ [134].

Линейният $[n, k]$ код C се нарича *цикличен*, ако за всяка кодова дума $c = (c_0, c_1, \ldots, c_{n-1})$, която принадлежи на C, думата $c' = (c_{n-1}, c_0, \ldots, c_{n-2})$ също при-

надлежи на този код. Ако разгледаме кодовите думи от C, чиито първи j информационни координати са 0 и изтрием тези координати от кодовите думи, ще получим $[n-j, k-j]$ код. Тези кодове наричаме *скъсени циклични* (CRC) кодове. CRC кодовете не са циклични кодове в общия случай, но имат поне същите възможности за контрол на грешки, както и цикличния код, от който са получени. CRC кодовете откриват всяка единична грешка и всички групи от грешки с дължина до броя на проверочните им символи. Това, както и изключително бързите и лесни за имплементиране кодиращи и декодиращи алгоритми, правят CRC кодовете изключително популярни за практическо приложение. Това е причината и някои CRC кодове да бъдат стандартизирани. В момента са известни 27 стандартизирани кода с от 1 до 64 проверочни бита.

CRC кодовете са били обект на много изследвания: [39, 40, 92, 93, 128, 132, 134, 136, 155, 172, 202, 203, 204] и др. За съжаление, те показват, че голяма част от стандартизираните кодове са с по-лошо поведение в сравнение с някои не стандартизирани CRC кодове. Основната причина за това е, че няма публикувани достатъчно данни, за да може да се подбере най-подходящия за конкретно практическо приложение. Тази липса става особено осезаема в последните години, когато софтуерните имплементации на кодирането и декодирането са преобладаващи. И ако преди наличието на евтин чип, който да прави това, е бил един от мотивите да се избере стандартизиран код, сега картината е променена. А и все по-нарастващото добавяне на компютърно управление към най-разнообразни уреди и механизми е свързано и с прилагането на някакъв CRC код. Следователно пресмятането на теглovинте спектри на CRC кодовете е особено интересна задача, както с теоретично така и с практическо значение. Друг важен проблем е предлагането на процедура, която да позволи на потребителите ефективен избор на най-подходящия за конкретното приложение код.

Свойството нормализираност и смесена директна сума (amalgamated direct sum ADS) на двоични кодове са въведени в [103], за да помогнат при конструирането на „покриващи кодове“, т.е. кодове с малки радиуси на покритие в сравнение с другите със същата дължина и размерност. Подходът изисква участващите в смесената директна сума кодове да са нормализирани. Едно свойство на кода, което не се установява лесно. По тази причина са правени изследвания от различни автори [47, 48, 49, 50, 103, 115, 119, 130], за да се отговори на въпроса: кои кодове са нормализирани. До този момент е известно, че нормализирани са кодовете с $n \leq 15$ или $k \leq 5$ или $d \leq 4$ или $R \leq 3$ или $d \geq 2R-1$ или $n-k \leq 9$.

Една от основните техники в съвременните високоскоростни оптични комуникации, която позволява на много потребители едновременно да използват една

и съща преносна среда, е optical code division multiple accesses (OCDMA) техниката. OCDMA работи асинхронно, без централизиран контрол и при нея няма конфликт между предаваните пакети с информация. Имаме N на брой двойки, които си комуникират през оптичната мрежа. За да се изпрати информация от потребителя j до потребителя k, към информацията на j се добавя допълнителна кодова последователност, която позволява на k да отдели предназначената за него информация от целия информационен поток, пристигащ при приемника. За целта е необходимо да се разработят кодови последователности, които удовлетворяват следните две условия:

1) всяка последователност трябва да бъде лесно различима от всички други последователности, които се получават при нейното циклично завъртане;

2) всяка последователност трябва да бъде лесно различима от всички други последователности, които се получават при цикличното завъртане на останалите използвани последователности или от някое тяхно циклично завъртане.

Едно решение на този проблем са предложените през 1989 година от Salehi [175] оптични ортогонални кодове (Optical Orthogonal Codes - OOC). Оптичният ортогонален код е константно-тегловен блоков код, т.е. множество от двоични последователности (кодови думи) с дължина v и тегло k, които имат специални автокорелационни и кроскорелационни свойства. По-точно, ниските стойности на автокорелацията позволяват да се удовлетвори първото условие, а на кроскорелацията - второто. Оптичните ортогонални кодове имат също приложения в мобилните радио комуникации, комуникациите на базата на разпръснат спектър, дизайн на сигнали за радари и сонари, конструиране на последователности за активни M потребителя измежду T в канали с колизии и без обратна връзка [17, 46, 53]. Тези кодове са широко изследвани и са известни различни техни конструкции [34, 35, 45, 46, 89, 157, 206]. Всички изброени изследвания оставят отворен въпроса за съществуването на оптични ортогонални кодове за параметрите, за които няма конструкции и доказателства за несъществуването им, както и за получаването на класификационни резултати за тях.

0.2 Структура на изложението

Работата съдържа, освен настоящия увод, седем глави и приложение.

В глава 1 са въведени основните понятия и твърдения от теорията на кодирането, които са използвани по-нататък в изложението. Дадени са горни и долни граници за радиуса на покритие на код и за стойностите на функцията $t_q[n, k]$.

В глава 2 са разгледани параметрите, определящи възможностите за откриване и коригиране на грешки, на линеен код.

Първият раздел представя теоретични резултати, според които минималното разстояние, радиусът на покритие, тегловното разпределение на кода, на съседните му класове и на лидерите на съседни класове са параметрите определящи поведението на един линеен код при откриване и коригиране на грешки.

Раздел 2.2 е посветен на сложността на задачите за пресмятане на минимално разстояние, радиус на покритие и тегловно разпределение на код. Дефинирани са класовете на сложност P, NP както и йерархията на задачите решавани за полиномно време. Показано е, че задачите за определяне на параметрите на кодовете, обект на изследване в дисертацията, са NP-трудни.

Методите за пресмятане на тегловно разпределение и на радиус на покритие на линеен код, използвани в дисертацията, са разгледани в раздел 2.3. Описани са и особеностите на програмната им реализация, която е направена на езика C++. Представените в този раздел методи са общи и работят за всякакви линейни кодове. Специфични алгоритми и техните програмни реализации, използвани за решаването на конкретни задачи, са описани в следващите глави, където се разглеждат и решените с тяхна помощ задачи.

В 2.3.1 са дадени алгоритми за ефективно пресмятане на тегловното разпределение (което включва и определянето на минималното разстояние) на q-ичен линеен код базирани на кода на Grey. За разлика от други алгоритми (виж [105, 151, 177, 200]), разработени за циклични кодове, тези могат да се използват за всеки линеен код без допълнителни изисквания към структурата му освен линейност. Те пресмятат цялото тегловно разпределение на кода, а не само първите няколко тегла, както е в [12].

Трите основни метода за пресмятане на радиус на покритие на код, използвани в дисертацията, са дадени в 2.3.2. Те също се базират само на линейността на кодовете и не отчитат други специфични особености на структурата на кода. Описани са особеностите на техните компютърни имплементации и е пресметната сложността им по време и памет. Първият и вторият от тези методи могат да се използват и за пресмятане на тегловното разпределение на лидерите на съседни класове и на теглата на векторите в самите съседни класове.

В края на раздела са коментирани начините за проверка на коректността на получените, с разработените в дисертацията програмни средства, резултати.

В глава 3 са определени радиусите на покритие на някои класове линейни кодове.

В раздел 3.1 са пресметнати радиусите на покритие на всички троични не-

гациклични кодове с четни дължини до 26. Направена е класификация на тези кодове като са използвани някои техни алгебрични свойства и специално разработен за това софтуер. За 22 от кодовете са използвани различни математически съображения, за да бъдат определени радиусите им на покритие, а за останалите - компютърни пресмятания. Резултатите са представени в Таблица 3.1.1 в приложението. Пресметнати са и минималните разстояния и тегловните разпределения на тези кодове. Получените резултати водят до затваряне на 7 от отворените случаи за стойностите на функцията $t_3[n,k]$, а за други три са подобрени горните граници.

Систематично изследване на възможните параметри на двоични и троични квази-съвършени кодове с малки размерности е направено в раздел 3.2. Първо е представен списък от безкрайни фамилии от квази-съвършени кодове, който включва всички известни двоични, троични и четвъртични кодове. След това са дадени известните ни спорадични примери на двоични и троични квази-съвършени кодове. Класифицирани са всички двоични квази-съвършени кодове с размерности до 9 и всички троични с размерности до 6. Получени са и частични класификации за кодове с размерности до 14. Дадени са нови примери на квази-съвършени кодове с минимално разстояние по-голямо от 5 и така е даден отговор на поставения от Etzion и Mounits в [81] отворен въпрос да се намерят нови такива кодове или да се докаже, че известните до тогава са единствени.

Получените резултати показват, че за всяка размерност има само няколко възможни дължини, за които съществуват квази-съвършени кодове. За някои параметри са намерени стотици и дори хиляди квази-съвършени кодове, което показва, че условието радиусът на покритие да превишава с едно радиуса на сферичната опаковка не е толкова рестриктивна характеристика на кода. Изследването поставя и два отворени въпроса свързани със съществуването на други квази-съвършени кодове с минимално разстояние по-голямо от 8, освен двоичните кодове с повторение, и за горна граница за минималното разстояние на квази-съвършен код.

В следващите три раздела от дисертацията са определени някои стойности на функцията $t_2[n,k]$.

В раздел 3.3 е използвана класификация на кодове със зададени параметри за да се докаже несъществуването им. Първо са определени възможните стойности на дуалното разстояние на кодовете, които ще се класифицират. След това е използвано компютърно търсене за кодове с фиксирани дължина, размерност, радиус на покритие и минимално разстояние на дуалния код. Това е направено за кодове с параметри отговарящи на първите 6 отворени случая за стойности на

$t_2[n,k]$ и радиус на покритие равен на долната граница за тези стойности. Оказа се, че кодове с търсените параметри не съществуват и тъй като горната и долната граница за $t_2[n,k]$ в тези случаи се различават само с 1 бяха определени стойностите на $t_2[17,6] = 5$, $t_2[17,8] = 4$, $t_2[18,7] = 5$, $t_2[19,7] = 5$, $t_2[20,8] = 5$ и $t_2[21,7] = 6$. Като следствие бяха определени и 4 нови стойности на функцията $l(m,R)$ - минималната дължина на код с размерност m и радиус на покритие R. Показано е също така, че съществува единствен $[14,6]$ код с минимален радиус на покритие 3.

В раздел 3.4 са определени минималните радиуси на покритие на всички двоични линейни кодове с размерност 6 като е показано, че

$$t_2[n,6] = \left\lfloor \frac{n-8}{2} \right\rfloor, \ for\ n \geq 18.$$

Стойностите на функцията за дължини по-малки от 18 бяха известни от Таблица 7.1 от [47]. Класифицирани са също всички двоични кодове с размерност до 6 и дължина до 15 имащи минимален радиус на покритие. На основата на тази класификация е предложена конструкция позволяваща да се определи пораждаща матрица за всеки код с размерност до 6 и минимален радиус на покритие. Показани са и примери за конструиране на кодове с минимален радиус на покритие и размерности по-големи от 6.

В последния раздел от трета глава са изследвани троичните линейни кодове с размерност 4. Направена е класификация на всички троични линейни кодове с размерност 4 и на базата на тази класификация са определени някои неизвестни стойности на функцията $t_3[n,4]$.

В четвърта глава е изследвано поведението на шумозащитни кодове при откриване и коригиране на грешки.

В раздел 4.1 са пресметнати тегловните разпределения на кодовите думи, на съседните класове и на лидерите на съседни класове на двоичните циклични кодове с дължини до 33, на троичните циклични и негациклични кодове с дължини до 20 и на някои двоични линейни кодове с дължини до 33, имащи максимално минимално разстояние. С програма, написана на Maple, е проверена монотонността на функцията P_{ue} за краен брой точки от интервала $\varepsilon \in [0, \frac{q-1}{q}]$, като по този начин са определени кодовете, които не са подходящи за контрол на грешки. Резултатите са представени в Таблици 4.1.1 - 4.1.4 в приложението на дисертацията.

Цикличните кодове са важен клас на линейните кодове. Те са интересни от теоретична гледна точка заради богатата си алгебрична структура. Изследванията на тази структура са довели до разработването на методи за конструиране на такива кодове с голямо минимално разстояние каквито са БЧХ кодовете и

квадратично-остатъчните кодове, както и на ефективни методи за декодиране какъвто е Meggit декодера. В раздел 4.2 е предложен метод за пресмятане на тегловното разпределение на съседните класове на циклични кодове като е използвана алгебричната им структура. Като илюстрация на метода са пресметнати тегловните разпределения на лидерите на съседни класове на всички троични циклични кодове с дължини до 14.

В раздел 4.3 е разгледан троичният квадратично-остатъчен $[13, 7, 5]$ код и е пресметнато, че радиусът му на покритие е 3, т.е. кодът е квази-съвършен. Показано е, че кодът е *подходящ* за откриване и коригиране на грешки и, като е използвана допълнителна информация за структурата на кода, са предложени два декодиращи алгоритъма. Вторият алгоритъм е особено ефективен защото намирането на позициите на грешките и декодирането използват таблица само с 12 елемента от GF(27).

На базата на богата гама от прости примери, в раздел 4.4 са демонстрирани интересни факти, които са полезни при решаването на практически проблеми в съвременните комуникации. Показано е, че кодове с еднакви основни параметри като дължина, размерност, минимално разстояние и радиус на покритие могат да имат различно поведение при контрол на грешки. Решен е, поставеният в книгата на MacWilliams and Sloane [145], отворен проблем 5.1 (глава 5, стр. 132).

Глава 5 разглежда поведението при откриване и коригиране на грешки на CRC кодове.

В раздел 5.1 е направен обзор на поведението при откриване на грешки на стандартизирани CRC кодове. Представени са известните методи за пресмятане на тегловните разпределения на CRC кодовете. Сравнени са стандартизирани CRC кодове и е показано, че някои от тях са лош избор за почти всички дължини. Такъв пример е DARK, който е оптимален за дължина 8, но за дължини по-големи от 10 има вероятност за неоткрита грешка значително по-голяма от оптималната. Дори за дължините, за които е стандартизиран този код, той има поведение много по-лошо от други CRC кодове.

В раздел 5.2 са разгледани полиноми от 8-ма степен над GF(2), които са подходящи за пораждащи полиноми на CRC кодове. Техните минимални разстояния, вероятността за неоткрита грешка и свойството да са *подходящи* са сравнени с кода използван в ATM стандарта. Пресметнати са радиусите им на покритие и тегловното разпределение на лидерите на съседните им класове. Намерени са два по-добри от ATM стандарта кодове за дължина 40, на която стандартизирания код се използва.

CRC кодовете с пораждащ полином от 16 степен над GF(2), които могат да

се използват за откриване на грешки в комуникационни системи, са изследвани в раздел 5.3. Направена е пълна класификация на всички такива кодове. Както и в изследването на кодовете от предишния раздел, са пресметнати минималните им разстояния, вероятността за неоткрита грешка и свойството да са *подходящи* за дължини от 18 до 1024. Направени са сравнения и са определени оптималните, по отношение на вероятността за неоткрита грешка и минималното разстояние, кодове за всяка от тези дължини. Показано е, че с много малки изключения, стандартизираните кодове имат поведение по-лошо от оптималното.

В раздел 5.4 са продължени изследванията върху поведението на CRC кодове от предишните два раздела. В този раздел е направена пълна класификация на всички полиноми над GF(2) до 10-та степен, които могат да бъдат пораждащи полиноми на CRC кодове. Пресметнати са всички необходими данни за оценката на поведението им за откриване и коригиране на грешки. Данните са на разположение на всички, за които биха представлявали интерес, на `http://www.moi.math.bas.bg/~tsonka`. Предложена е бърза и лесна процедура за избор на оптимален за конкретно приложение код. Разгледани са също и кодове с дължини по-големи от реда на полинома, тъй като в практиката се използват и такива кодове. Изведена е формула за пресмятане на броя на кодовите им думи с минимално тегло. Този брой е важен при оценката на вероятността за неоткрита грешка на кода.

В глава 6 са изследвани някои характеристики на шумозащитни кодове свързани с техните възможности за контрол на грешки.

В раздел 6.1 са пресметнати нютоновите радиуси на покритие на всички двоични циклични кодове с дължини до 31, на двоичните кодове с максимално минимално разстояние с дължини до 33 и на всички троични циклични и негациклични кодове с дължини до 22. Този параметър на кода показва максималното тегло на коригируемите по единствен начин грешки. Той е полезен при разработване на декодиращи алгоритми, работещи над границата на сферичната опаковка.

Линейни кодове с неравномерна защита на символите са разгледани в раздел 6.2. С помощта на компютърно търсене, са определени възможностите за неравномерна защита на всички троични циклични и негациклични кодове с дължини до 26, които имат минимално разстояние поне 3.

В раздел 6.3 е доказано, че всички двоични кодове с дължини 16, 17 и 18, или ко-размерност 10 са нормализирани. Направена е класификация на всички тези кодове. Чрез прилагане на смесена директна сума към нормализирани кодове могат да бъдат получени нови кодове с по-големи дължини и размерности и малък радиус на покритие, като в много от случаите могат да бъдат получени и такива

с минималния възможен радиус на покритие.

В глава 7 са получени първите класификационни резултати за оптични ортогонални кодове с параметри, които са най-често използвани в практиката. Тъй като оптични ортогонални кодове с определени параметри отговарят и на други комбинаторни обекти, са получени нови класификационни резултати и за тях.

Броят на кодовите думи в един оптичен ортогонален код се нарича размер на кода. Оптимални оптични ортогонални кодове са тези, които имат най-големия възможен брой кодови думи за фиксирани параметри.

Класификации на оптимални оптични ортогонални кодове с тегла 4 и 5, автокорелация 2 и кроскорелация 1 са направени в раздел 7.1. Направена е пълна класификация с точност до изоморфизъм на оптималните оптични ортогонални кодове с тегло 4 и дължини до 75, без тези на дължина 71. Конструиран е и по един оптимален оптичен ортогонален код, ако съществува такъв, за дължини до 181. Оптималните оптични ортогонални кодове с тегло 5 и дължина до 114 са класифицирани с точност до мултипликативна еквивалентност и е конструиран по един код, ако съществува, за дължини до 155.

Оптичните ортогонални кодове с определени параметри се наричат също циклично-пермутационни константно-тегловни (cyclically permutable constant weight, CPCW) кодове. В раздел 7.2 са класифицирани оптималните циклично-пермутационни константно-тегловни кодове с тегло на кодовите думи 3 и дължини до 61. Съществуването на циклично-пермутационни константно-тегловни кодове с дължини $v = 6t + 2$, $t \equiv 2, 3 \ (mod\ 4)$ беше известно до нашата работа, но нямаше класификационни резултати за такива кодове. Тъй като изследваните в този раздел оптимални циклично-пермутационни константно-тегловни кодове отговарят на оптимални циклични двоични константно тегловни кодове с тегло 3 и минимално разстояние 4, на оптични ортогонални кодове и на разностни пакетирания (difference packings) със съответните параметри, са получени класификационни резултати и за тези комбинаторни обекти. Когато оптималните циклично-пермутационни константно-тегловни кодове са съвършени, те отговарят на циклични Щайнерови системи от тройки и на циклични разностни фамилии. Така получената класификация за циклично-пермутационни константно-тегловни кодове с дължина 61 е и нов класификационен резултат за циклични Щайнерови системи от тройки от ред 61 и за $(61, 3, 1)$ циклични разностни фамилии.

Връзката между съвършените оптични ортогонални кодове и цикличните разностни фамилии позволи получаването на класификационни резултати за циклични разностни фамилии с малки параметри, като се използва подход подобен на този от раздел 7.1. Резултатите са представени в раздел 7.3. Те повтарят из-

вестните вече от предишни публикации класификации, както и нови за по-големи дължини.

Класификационните резултати за различни класове кодове получени в настоящата работа са систематизирани и представени в таблици в приложенията към нея. Таблиците са отделени от основния текст, за да не се затруднява четенето му от една страна, и за да могат лесно да се правят справки за интересуващите читателя конкретни параметри, от друга.

0.3 Апробация на резултатите

В дисертацията са включени резултати получени в периода 1998 г. – 2013 г. Резултатите, представени в публикации [209, 211, 213, 221, 224, 225] и [227], са получени самостоятелно. Останалите резултати са получени в съавторство, както следва:

Додунеков, Kötter	[212]
Буюклиев	[215, 226]
Буюклиев, Додунеков, Williams	[217]
Великова	[218]
Буюклиев, Додунеков, Fack	[223]
Ваврек	[214]
Додунеков, Казаков	[208, 210]
Sallam	[219, 222, 220]
Ганчева	[216]
Топалова	[228, 229, 230, 231, 232]

Публикувани са в следните научни списания:

IEEE Trans. on Information Theory	[211], [212], [214], [223]
Advances in Mathematics of Communications	[226], [227]
Computer Communications	[208]
IEE Proc. Communications	[210]
IEEE Trans. on Communications	[224]
Journal of Combinatorial Designs	[228]
Applical Algebra in Eng., Communic. and Computing	[232]
Mathematics of Distances and Applications	[231]
Annuarie de L'Université de Sofia 'St. Kl. Ohridski'	[218]
Mathematica Balkanica	[216], [217], [222]
Serdica, Journal of Computing	[221]

Включените в дисертацията резултати са докладвани на:

Международна конференция *Algebraic and Combinatorial Coding Theory*, 1998 г., 2002 г., 2006 г., Русия и 2000 г., 2004 г., 2008 г., 2012 г., България;

Международна конференция *Optimal codes and Related Topics*, 1998 г., 2001 г., 2007 г., 2009 г., 2011 г., България;

IEEE International Symposium of Information Theory, 2000 г., Италия;

Конгрес на Математическата асоциация на Югоизточна Европа *MASSEE*, 2003 г., България, 2006 г., Кипър, 2009 г., Македония;

Юбилейна конференция на ВТУ *Mathematics, Informatics and Computer Sciences*, 2006 г., България;

Workshop on Combinatorial search, 2005 г., Унгария

NATO Advsnced Recearch Workshop *Enchancing cryptographic primitives with techniques from error correcting codes*, 2008 г., България;

Международна конференция *Applications of Computer Algebra*, 2006 и 2012 г., България;

Международна конференция *Mathematics in Industry*, 2010 г., България;

Международна конференция *Mathematics of Distances and Applications*, 2012 г., България;

Международна конференция *Mathematics Days in Sofia*, 2014 г., България.

Отделни резултати от дисертацията са докладвани пред:

Националния семинар по теория на кодирането;

Семинар на Великотърновски университет „Св. св. Кирил и Методий“ и секция МОИ;

Семинар в Технически университет в Улм, 2000 г., Германия;

Семинар в Института по математика на Унгарската Академия на науките 2004 г., 2005 г., 2008 г. Унгария;

Семинар в университета в Гент, 2006 г., 2008 г., 2010г., Белгия.

0.4 Авторска справка

По мнение на автора основните приноси на дисертанционния труд са:

- Определени са
  - Радиусите на покритие всички троични негациклични кодове с четни дължини до 26.
  - $t_2[17,6] = 5, t_2[17,8] = 4, t_2[18,7] = 5, t_2[19,7] = 5, t_2[20,8] = 5$ и $t_2[21,7] = 6$.
  - $t_3[20,10] = t_3[20,11] = t_3[21,11] = 4$, $t_3[24,12] = t_3[25,13] = 5$, $t_3[21,10] = t_3[22,11] = 5$, $t_3[16,8] = 4$, $t_3[18,8] = 5$, $t_3[24,11] = 6$, $t_3[28,14] = 6$, $t_3[31,16] = 6$, $t_3[33,20] = 5$, $t_3[34,17] = 7$, $t_3[36,18] = 7$, $t_3[40,20] = 8$.
  - Минималните радиуси на покритие на всички двоични линейни кодове с размерност 6.
  - Двоичните циклични кодове с дължини до 33, троичните циклични и негациклични кодове с дължини до 20 и някои двоични кодове с дължини до 33 и максимално минимално разстояние, които не са подходящи за контрол на грешки.
  - Нютоновите радиуси на покритие на всички двоични циклични кодове с дължини до 31, на двоични кодове с максимално минимално разстояние с дължини до 33 и на всички троични циклични и негацикличини кодове с дължини до 22.
  - Възможностите за неравномерна защита на символите на всички троични циклични и негациклични кодове с дължини до 26, които имат минимално разстояние поне 3.
- Класифицирани са
  - Двоичните квази-съвършени кодове с размерности до 9 и троичните квази-съвършени кодове с размерности до 6. Получени са частични класификации за двоични и троични квази-съвършени кодове с размерности до 14.
  - Всички двоични кодове с минимален радиус на покрите с размерност до 6 и дължина до 15.
  - Всички троични кодове с размерност до 4.

  - Скъсените цикличини кодове с до 10 и с 16 проверочни символа.
  - Оптималните $(v, 4, 2, 1)$ оптични ортогонални кодове с $v \leq 75$, $v \neq 71$.
  - Оптималните $(v, 5, 2, 1)$ оптични ортогонални кодове с $v \leq 114$.
  - $(v, k, 1)$ циклично-пермутационните константно-тегловни кодове с $v \leq 61$.
  - $(61, 3, 1)$, $(73, 4, 1)$, $(76, 4, 1)$, $(81, 5, 1)$ и $(85, 5, 1)$ цикличните разностни множества.

- Предложена е конструкция за определяне на пораждаща матрица на произволен двоичен код с размерност до 6 и минимален радиус на покритие.

- Доказано е, че всички двоични кодове с дължини 16, 17 и 18, или ко-размерност 10 са нормализирани.

- Предложена е ефективна процедура за избор на най-подходящия за съответното приложение скъсен цикличен код, която използва резултатите от класификацията на скъсените циклични кодове с до 10 и с 16 проверочни символа.

- Решен е поставен от Etzion и Mounits отворен въпрос като са намерени нови примери на квази-съвършени кодове с минимално разстояние по-голямо от 5.

- Решен е отворен проблем 5.1 (глава 5, стр. 132) поставен в книгата на MacWilliams and Sloane „The theory of Error-Correcting Codes“.

- Определени са оптималните по отношение на вероятността за неоткрита грешка и минимално разстояние скъсени циклични кодове с до 10 и с 16 проверочни символа. Някои от тях са намерили различни практически приложения в:

  - комуникационни системи за управление на жп транспорт и градски релсов транспорт в Италия (Ansaldo STS S.p.A.), Германия (Thales Rail Signalling Solutions GesmbH), Индия (Safety Critical Railway project);

  - телекомуникационни системи в Русия, Турция (UEKAE-TUBITAK);

  - затворени безжични системи разработвани в ТУ Берлин;

  - US патент 8341510 B2, 2012;

  - US патент 8327251 B2, 2012.

Глава 1

Основни понятия и предварителни резултати

Ще представим основни дефиниции и резултати от теория на кодирането, необходими по-нататък за изложението.

1.1 Основни понятия от Теория на кодирането

Нека $F_q = GF(q)$ е поле с q елемента, където $q = p^s$ и p е просто число, а F_q^n е n-мерното векторно пространство над F_q.

Дефиниция 1.1.1. *Непразно подмножество на F_q^n се нарича q-ичен код с дължина n.*

Векторите, принадлежащи на кода се наричат *кодови думи.*

1.2 Линейни кодове

Дефиниция 1.2.1. *Всяко k-мерно подпространство на n-мерното векторно пространство F_q^n се нарича линеен $[n, k]$ код над F_q с блокова дължина n и размерност k.*

Дефиниция 1.2.2. *Пораждаща матрица G на линеен код C над F_q с блокова дължина n и размерност k е $k \times n$ матрица, чиито редове образуват базис на кода като линейно пространство над F_q.*

Линейният $[n, k]$ код с пораждаща матрица G представлява съвкупността от вектори $\{u.G \mid u \in F_q^k\}$.

Дефиниция 1.2.3. *Под скаларно произведение на векторите* $u = (u_1, ..., u_n)$ *и* $v = (v_1, ..., v_n)$ *от* F_q^n *ще разбираме елемента от* F_q $u.v = u_1v_1 + u_2v_2 + ... + u_nv_n$.

Два вектора са *ортогонални*, ако скаларното им произведение е 0.

Дефиниция 1.2.4. *Дуален код* $C^\perp$ *на линейния код* C *ще наричаме ортогоналното допълнение на* C, *т.е.* $C^\perp = \{v \in F_q^n \mid v.c = 0, \forall c \in C\}$.

Ако C е $[n, k]$ код, то дуалният му $C^\perp$ е линеен $[n, n-k]$ код. Размерността $n-k$ на $C^\perp$ ще наричаме *ко-размерност* на C.

Дефиниция 1.2.5. *Проверочна матрица* H *на кода* C *наричаме всяка пораждаща матрица на дуалния му код* $C^\perp$.

Нека $(n-k) \times n$ матрицата H е проверочна матрица за кода C, т. е. $C = \{u \in F_q^n \mid Hu^t = 0\}$. За пораждащата матрица G е в сила, че $H.G^t = 0$.

Ако пораждащата матрица G на кода C е във вида $G = [I_k \mid A]$, където I_k е единичната $k \times k$ матрица, казваме, че C е в *систематичен вид*. Тогава $H = [-A^t \mid I_{n-k}]$.

Дефиниция 1.2.6. *Разстояние по Хеминг* $d(x, y)$ *между два вектора* $x = (x_1, x_2, \ldots, x_n)$ *и* $y = (y_1, y_2, \ldots, y_n)$ *от* F_q^n *се нарича броят на координатите, в които те се различават:*

$$d(x, y) = |\{i | x_i \neq y_i\}|.$$

Дефиниция 1.2.7. *Тегло по Хеминг* $wt(v)$ *на вектора* $v \in F_q^n$ *се нарича броят на ненулевите координати на вектора* v.

Дефиниция 1.2.8. *Минимално разстояние на кода* C *се дефинира като минималното измежду разстоянията между две различни кодови думи:* $d = d(C) = \min\{d(u, v) \mid u, v \in C, u \neq v\}$.

Дефиниция 1.2.9. *Съседен клас на кода* C, *определен от вектора* x, *ще наричаме множеството* $x + C = \{x + c \mid c \in C\}$. *Лидер на съседен клас наричаме вектор с минимално тегло в съседния клас, т.е. всеки вектор* $y \in x + C$ *може да е лидер на този съседен клас, ако* $wt(y) = \min\{wt(x + c) \mid c \in C\}$.

Всеки вектор от пространството F_q^n е в някой съседен клас на C. Всеки съседен клас съдържа точно q^k вектора и два съседни класа не се пресичат или съвпадат.

Дефиниция 1.2.10. *Разстоянието на вектор $x \in F_q^n$ до кода C се определя като минималното измежду разстоянията на вектора x до кодова дума, т.е.* $d(x, C) = \min\{d(x, c) \mid c \in C\}$.

Вижда се, че разстоянието от вектора x до кода C е равно на теглото на кой да е лидер на съседния клас $x + C$.

Дефиниция 1.2.11. *Радиус на покритие $R = R(C)$ на код C се нарича максималното измежду разстоянията на вектор от пространството F_q^n до кода C, т.е.* $R(C) = max\{d(x, C) \mid x \in F_q^n\}$.

Кълбо $K(c, R)$ с център c и радиус R е множеството от всички вектори, които са на разстояние по-малко или равно на R, от центъра c. Радиусът на покритие на линеен код е равен на теглото на най-тежкия лидер на съседен клас на кода и е най-малкото цяло число R, такова че кълбата с радиус R, описани около кодовите думи, покриват цялото пространство F_q^n.

Съседните класове, в които има единствен вектор с минимално тегло, представляват специален интерес. *Нютонов радиус ν* на код е теглото на най-тежкия единствен лидер на съседен клас. Този параметър определя максималното тегло на еднозначно коригируем вектор-грешка (виж [106]).

Линейните кодове с дължина n, размерност k, минимално разстояние d и радиус на покритие R над крайно поле с q елемента ще означаваме с $[n, k, d]_q R$. Когато някои от тези параметри не са известни или са ясни от контекста, те ще бъдат пропускани.

Лема 1.2.12. *[104] Нека C е линеен $[n, k]$ код над F_q с проверочна матрица $H = (h_1 h_2 \dots h_n)$, където $h_i, i = 1, \dots, n$ са стълбовете на матрицата H. Тогава:*

а) Минималното разстояние на кода C е равно на най-малкото естествено число d, такова че $0 = a_1 h_{t_1} + a_2 h_{t_2} + \dots + a_d h_{t_d}$ за някои $a_i \in F_q \backslash \{0\}$, $h_{t_i} \in \{h_1, h_2, \dots, h_n\}$, $1 \le i \le d$ и $t_i \ne t_j$.

б) Теглото на лидера на съседния клас $x + C$, е равно на най-малкото цяло число l, такова че $Hx^t = a_1 h_{t_1} + a_2 h_{t_2} + \dots + a_l h_{t_l}$ за някои $a_i \in F_q$ и $h_{t_i} \in \{h_1, h_2, \dots, h_n\}$ за $1 \le i \le l$.

в) Радиусът на покритие на кода C е най-малкото естествено число r, такова, че всеки ненулев вектор-стълб с $n - k$ координати е линейна комбинация на не повече от r стълба на H.

Два линейни кода се наричат *еквивалентни*, ако всички кодови думи на едния, могат да се получат от кодовите думи на другия, чрез прилагане на последователност от следните трансформации:

(1) пермутация на координатите;

(2) умножаване на фиксирана координата от всяка кодова дума с фиксиран ненулев елемент от полето;

(3) прилагане на автоморфизъм на полето към елементите във всички координатни позиции.

Пораждащите матрици на два еквивалентни кода могат да се преобразуват една в друга чрез елементарни преобразования по редове, пермутация на стълбовете и умножаване на стълб с фиксиран ненулев елемент на полето. Същото е в сила и за проверочните им матрици.

Основните параметри на един линеен код C са блокова дължина n, размерност k, минимално разстояние d и радиус на покритие R. Тези параметри съвпадат при еквивалентни кодове.

Автоморфизъм на линеен код C е крайна последователност от трансформации от тип (1) - (3), която изобразява всяка кодова дума от C в кодова дума от C. Множеството от всички автоморфизми на C е група, която наричаме *група от автоморфизми* на кода и бележим с $Aut(C)$.

За всеки линеен код C означаваме с A_i броя на кодовите думи с тегло i. Редицата $(A_0, A_1, \ldots, A_n)$ наричаме *тегловен спектър* или *тегловно разпределение* на C.

Ще означаваме с α_i за $i = 0, 1, \ldots, n$ броят на лидерите на съседни класове с тегло i, а редицата $(\alpha_0, \alpha_1, \ldots, \alpha_n)$ ще наричаме *тегловен спектър* или *спектър на лидерите на съседни класове* на кода.

За краткост ще означаваме спектъра на кода с $\bar{A}$, а спектъра на лидерите на съседни класове с $\bar{\alpha}$.

Ще казваме, че блоковият код C *открива* t *грешки*, ако всяка дума a', получена от кодовата дума a чрез промяна на $1, 2, \ldots, t$ нейни символа, не е кодова дума и *коригира* t *грешки*, ако разстоянието $d(a, a')$ е строго по-малко от разстоянието на всяка друга кодова дума до a'. Тогава всеки линеен $[n, k, d]_q$ код открива до $d - 1$ грешки и коригира до $t = \left\lfloor \dfrac{d-1}{2} \right\rfloor$ грешки.

Да разгледаме комуникационна система, в която кодова дума x от $[n, k, d]_q$ код C се предава през комуникационен канал. Комуникационният канал добавя към изпратената кодова дума *вектор-грешка* e и получателя получава думата y. Векторът грешка има същата дължина както кодовата дума x и получената дума y и ненулеви елементи в позициите където са възникнали грешки, т.е. $y = x + e$.

Комуникационният канал се описва от вероятността да се получи вектор y с дължина n когато е бил изпратен вектор x със същата дължина. Моделът на

канала, който ще използваме е q-ичен *симетричен канал без памет*. Това е дискретен канал с q-ичен вход и q-ичен изход и вероятност за грешка на канала ε, $0 \leq \varepsilon \leq \dfrac{q-1}{q}$. Всеки символ се получава коректно с вероятност $1-\varepsilon$ и се променя в някой от останалите $q-1$ символа с една и съща вероятност $\dfrac{\varepsilon}{(q-1)}$.

При приемника, получената кодова дума се декодира към най-близката по Хемингово разстояние кодова дума. Приемаме, че векторът-грешка винаги е някой от лидерите на съседен клас. Тогава:

1. Може да има единствена кодова дума x, такава че $d(x,y)$ е минимално. В този случай, декодираме коректно y като x. Вероятността за коректно декодиране е

$$P_{corr} = \sum_{i=0}^{n} \alpha_i \left(\frac{\varepsilon}{q-1}\right)^i (1-\varepsilon)^{n-i}, \tag{1.1}$$

а вероятността за грешка е

$$P_{err} = 1 - P_{corr} = 1 - \sum_{i=0}^{n} \alpha_i \left(\frac{\varepsilon}{q-1}\right)^i (1-\varepsilon)^{n-i}. \tag{1.2}$$

2. Ще открием грешка, ако има по-вече от една кодова дума, която е на минимално разстояние от y.

3. Ще декодираме грешно, ако шумът по канала е променил x така, че най-близката кодова дума до y е $x' \neq x$, т.е. имаме *неоткрита грешка*.

Тъй като кодът е линеен, неоткрити грешки възникват, тогава и само тогава когато векторът-грешка e е ненулева кодова дума. Ако са променени i позиции на кодовата дума, т.е. e има тегло i, то вероятността на тази грешка е $\left(\dfrac{\varepsilon}{q-1}\right)^i (1-\varepsilon)^{n-i}$ и вероятността за неоткрита грешка е

$$P_{ue}(C,\varepsilon) = \sum_{i=1}^{n} A_i \left(\frac{\varepsilon}{q-1}\right)^i (1-\varepsilon)^{n-i}. \tag{1.3}$$

Да означим с $P_{ue}^{(t)}(C,\varepsilon)$ вероятността за неоткрита грешка след коригиране на t грешки. Нека $Q_{h,l}$ да е броят на думите с тегло l в съседен клас с лидер с тегло h, като са изключени лидерите. Тогава (see [124, 144])

$$P_{ue}^{(t)}(C,\varepsilon) = \sum_{h=0}^{t} \sum_{l=0}^{n} Q_{h,l} \left(\frac{\varepsilon}{q-1}\right)^l (1-\varepsilon)^{n-l}. \tag{1.4}$$

Когато искаме да намерим $[n,k]$ код за откриване и коригиране на грешки за някое приложение, най-добрият избор би бил код, чиято стойност на вероятността за неоткрита грешка след коригиране на t грешки $P_{ue}^{(t)}(C,\varepsilon)$ е минимална (*оптимален код*), ако знаем ε. Отворен въпрос е за кои n и k съществуват $[n,k]$ кодове, които са оптимални за всяко $\varepsilon \in [0, \frac{q-1}{q}]$. Някои резултати в тази посока за

двоични кодове са представени в [132].

Теорема 1.2.13. *Ако C е $[n,k]$ код, който е оптимален за ε, то и $C^\perp$ е $[n,n-k]$ код оптимален за $\dfrac{1-2\varepsilon}{2-2\varepsilon}$. Освен това*

$$P_{ue}[C,\varepsilon] = 2^k(1-\varepsilon)^n P_{ue}\left[C^\perp, \frac{1-2\varepsilon}{2-2\varepsilon}\right] + 2^{k-n} - (1-\varepsilon)^n.$$

Следствие 1.2.14. *Ако C е код, който е оптимален за всяко $\varepsilon \in \left[0, \dfrac{1}{2}\right]$, то и $C^\perp$ е оптимален.*

Теорема 1.2.15. *Ако $1 \leq k \leq 4$ и $n \geq k$, то съществува $[n,k]$ код, който е оптимален за всяко $\varepsilon \in \left[0, \dfrac{1}{2}\right]$.*

За съжаление, доказателството на тази теорема не дава конструкции за такива кодове.

Когато разглеждаме задачата за намиране на оптимални кодове можем да отделим два проблема. Първо, може да не знаем стойността на ε или тя да се променя по време на комуникацията и тогава код, който е оптимален за $\varepsilon' \neq \varepsilon$, може да не е оптимален за ε. Второ, дори и да знаем стойността на εq не е известен общ метод за конструиране на оптимални кодове различен от пълното претърсване. Следователно полезно е да имаме критерии които да ни помагат да определим дали един код е добър за контрол на грешки.

Кодът C се нарича *t-подходящ* (или само *подходящ*, когато $t = 0$ и той се използва само за откриване на грешки) ако $P_{ue}^{(t)}(C,\varepsilon)$ е монотонна и *t-добър* ако $P_{ue}^{(t)}(C,\varepsilon) \leq P_{ue}^{(t)}(C, \frac{q-1}{q})$ за всички $\varepsilon \in [0, \frac{q-1}{q}]$. Ясно е, че ако един код е подходящ, то той е и добър.

За много от кодовете обаче стойностите на $\bar{A}$ и $Q_{h,l}$ не са известни и тези критерии е трудно да бъдат проверени.

Дискретни достатъчни условия за един линеен $[n,k,d]_q$ код да е подходящ и добър са изведени от Додунекова и Додунеков [70]

Нека да означим с $A_i^{(t)} = \sum_{h=0}^{t} Q_{h,i}$ за $i = t+1, \ldots, n$ тегловното разпределение на векторите в съседните класове с лидери с тегла до t като лидерите са изключени. Да означим с $V_q(t)$ обема на q-ична сфера с радиус t в n-мерното векторно пространство над $GF(q)$ и нека $m_{(i)} = m(m-1)\ldots(m-i+1)$.

Теорема 1.2.16. *Ако* $(q^{-(n-k)} - q^{-n})V_q(t) \geq q^{-l}\sum_{i=t+1}^{l} \frac{l_{(i)}}{n_{(i)}} A_i^{(t)}$
за $l = t+1, \ldots, n$, то C е t-добър за коригиране на грешки.

Теорема 1.2.17. *Ако* $\sum_{i=t+1}^{l} \frac{l_{(i)}}{n_{(i)}} A_i^{(t)} \geq q\sum_{i=t+1}^{l-1} \frac{(l-1)_{(i)}}{n_{(i)}} A_i^{(t)}$ *за $l = t+2, \ldots, n$, то C е t-подходящ за коригиране на грешки.*

Забележка. За $t = 0$ тези условия са достатъчни кодът C да е *добър* и *подходящ* за откриване на грешки.

1.3 Циклични кодове

Дефиниция 1.3.1. *Линейният код C с дължина n се нарича цикличен, ако за всяка кодова дума $c = (c_0, c_1, \ldots, c_{n-1})$, която принадлежи на C, думата $c' = (c_{n-1}, c_0, \ldots, c_{n-2})$ също принадлежи на този код.*

Нека $R_n = F_q[x]/(x^n - 1)$ е пръстена състоящ се от класовете остатъци на пръстена $F_q[x]$ по модул полинома $x^n - 1$. Във всеки клас по модул $x^n - 1$ се съдържа точно по един нормиран полином от степен, по-малка от n. Изображението $c = (c_0, c_1, \ldots, c_n) \to c_0 + c_1 x + \ldots + c_{n-1}x^{n-1}$ задава изоморфизъм между линейните пространства F_q^n и R_n. При това изображение цикличните кодове с дължина n се отъждествяват с идеалите в R_n. Всеки идеал I в пръстена R_n е главен и има единствен нормиран полином $g(x)$ с минимална степен, който поражда идеала I, т.е. $I = (g(x))$ и полинома $g(x)$ дели $x^n - 1$.

Дефиниция 1.3.2. *Нека C е цикличен код. Нормираният полином $g(x)$ с минимална степен, който поражда кода C, се нарича пораждащ полином за кода C, а полиномът $h(x) = \dfrac{(x^n - 1)}{g(x)}$ е проверочен полином за кода C.*

Цикличният код C с блокова дължина n и пораждащ полином $g(x)$ има размерност $k = deg\ h(x) = n - deg\ g(x)$. Ако пораждащият полином е $g(x) = g_0 + g_1 x + \ldots + g_m x^m$, където $m = n - k$, то една пораждаща матрица за C е:

$$G = \begin{bmatrix} g_0 & g_1 & . & . & . & g_m & 0 & . & . & . & 0 \\ 0 & g_0 & g_1 & . & . & . & g_m & . & . & . & 0 \\ . & . & . & . & . & . & . & . & . & . & . \\ 0 & . & . & . & 0 & g_0 & g_1 & . & . & . & g_m \end{bmatrix}.$$

Една проверочна матрица за кода C можем да получим, използвайки проверочния му полином $h(x) = h_0 + h_1 + \ldots + h_k x^k$:

$$H = \begin{bmatrix} 0 & . & . & . & 0 & h_k & . & . & . & h_1 & h_0 \\ 0 & . & . & . & h_k & . & . & . & h_1 & h_0 & 0 \\ . & . & . & . & . & . & . & . & . & . & . \\ h_k & . & . & . & h_1 & h_0 & 0 & . & . & . & 0 \end{bmatrix}.$$

Дуалният код $C^{\perp}$ на един цикличен код C е също цикличен и е еквивалентен на кода породен от $h(x)$. $C^{\perp}$ се поражда от полинома $x^k h(x^{-1}) = h_k + h_{k-1}x + \ldots + h_0 x^k$.

Нека $(q,n)=1$ и C е цикличен код над F_q с дължина n и пораждащ полином $g(x)$. Ако α е примитивен n-ти корен на единицата над F_q, то $\alpha \in F_{q^m}$, където m е показателят на q по модул n, т.е. m е минималното естествено число, такова че $q^m \equiv 1\ (mod\ n)$. Тъй като $g(x)$ дели x^n-1, то всеки корен на пораждащия полином е степен на α. По такъв начин, цикличният код C може да се определи със задаване на α и с множеството $P=\{p \mid \alpha^p$ е корен на $g(x)\}$. Множеството от корени на пораждащия полином (а понякога и множеството P) ще наричаме нули или корени на кода. Ако γ е корен на $g(x) \in F_q[x]$, то и γ^q - спрегнат на γ, е корен на $g(x)$. От това следва, че множеството P от корените на един цикличен код трябва да е затворено относно умножение с q по модул n.

Дефиниция 1.3.3. *Циклотомичен клас по модул n над F_q при $(q,n)=1$, ще наричаме множеството $C_s=\{s,qs,q^2s,\ldots,q^{m_s-1}s\}$, където m_s е минималното естествено число, за което $(q^{m_s}-1)s \equiv 0\ (mod\ n)$.*

Множеството $\mathbf{Z}_n$ от остатъците по модул n се разбива на обединение на циклотомични класове и множеството от корени на един цикличен код P е също обединение на циклотомични класове. Следователно множеството от корени на кода се определя точно, ако е зададен примитивен n-ти корен на единицата и по един елемент от всеки циклотомичен клас C_s, за който $C_s \subset P$.

Нека n е просто число. Цялото число i, взаимно просто с n, се нарича *квадратичен остатък* по модул n, ако съществува цяло число X за което $X^2 \equiv i\ (mod\ n)$. Съществуват точно $\dfrac{(n-1)}{2}$ квадратични остатъка по модул n. Останалите $\dfrac{(n-1)}{2}$ ненулеви остатъка по модул n се наричат *квадратични неостатъци*. Полиномът x^n-1 се разлага на множителите $(x-1)$, $g_m(x)$ и $g_n(x)$, където $M=\{k|\alpha^k$ - корен на $g_m(x)\}$ се състои от квадратичните остатъци по модул n, а $N=\{s|\alpha^s$ - корен на $g_n(x)\}$ - от квадратичните неостатъци по модул n. Цикличните кодове с пораждащи полиноми $g_m(x)$, $(x-1)g_m(x)$, $g_n(x)$ и $(x-1)g_n(x)$ се наричат *квадратично-остатъчни кодове*.

1.4 Скъсени циклични (CRC) кодове

Ако ни е даден един систематичен линеен $[n,k]$ код, можем да получим $[n-i,k-i]$ систематичен линеен код като изтрием първите i информационни символа от всички кодови думи. Когато това е цикличен код, в общия случай, полученият код не е цикличен, защото вече не всяко циклично завъртане на кодова дума ще доведе до получаване на кодова дума. Ще наричаме този код *скъсен цикличен код*.

Скъсеният цикличен код има поне същото минимално разстояние както и цикличният, от който е получен.

Дефиниция 1.4.1. *Псевдо-цикличен код е код, който е идеал по модул някакъв полином* $f(x)$.

Следващите две теореми показват, че псевдо-цикличните кодове и скъсените циклични кодове съвпадат.

Теорема 1.4.2. *[167, Теорема 8.5] Всеки псевдо-цикличен код с минимално тегло по-голямо от 2 е скъсен цикличен код.*

Теорема 1.4.3. *[167, Теорема 8.6] Всеки скъсен цикличен код е псевдо-цикличен код.*

Кодиращите и декодиращите процедури за CRC кодовете са същите както и за цикличните кодове.

1.5 Граници за радиус на покритие на код

Радиус на сферичната опаковка на един код е най-голямото цяло e, такова че кълбата с радиус e, описани около кодовите думи, не се пресичат. Тогава е в сила следната *граница на сферичната опаковка* за радиуса на покритие на код:

$$R \geq e = \left\lfloor \frac{d-1}{2} \right\rfloor. \tag{1.5}$$

Една долна граница (*граница на сферичното покритие*) [189] за радиус на покритие на кодове следва от факта, че кълбата с радиус R покриват цялото пространство F_q^n:

$$\sum_{i=0}^{R(C)} \binom{n}{i} (q-1)^i \geq q^{n-k}. \tag{1.6}$$

Една горна граница за радиуса на покритие е получена в [62]:

$$R(C) \leq n - k. \tag{1.7}$$

Нека C е линеен код, а с s' да бележим броят на различните ненулеви тегла в дуалния му код. Тогава в сила е *границата на Delsarte* [62]:

$$R(C) \leq s'. \tag{1.8}$$

Нека C_1 и C_2 са два линейни кода с равни дължини n и кодът C_2 съдържа кода C_1, т.е. $C_1 \subset C_2$. Дефинираме $m(C_2, C_1)$ като минимално тегло на дума от кода C_2, която не принадлежи на кода C_1, т.е. $m(C_2, C_1) = \min\{wt(x) \mid x \in C_2 \backslash C_1\}$

и $M(C_2, C_1)$ като максималното разстояние на дума от кода C_2 до кода C_1 , т.е. $M(C_2, C_1) = max\{d(x, C_1) \mid x \in C_2\}$. Вярна е следната

Лема 1.5.1. *[48] (Лема за подкод.) Нека C_1 и C_2 са линейни кодове, за които $C_1 \subset C_2$. Тогава:*

$$R(C_1) \geq M(C_2, C_1) \geq m(C_2, C_1) \geq d(C_2) \tag{1.9}$$

$$R(C_1) \leq R(C_2) + M(C_2, C_1). \tag{1.10}$$

Чрез използване на описаните по-долу конструкции, могат да се получат нови кодове, радиусите на покритие на които се пресмятат точно или удовлетворяват някои неравенства.

- *Външна директна сума на кодове* [48]. Нека C_1 е $[n_1, k_1]$ код и C_2 е $[n_2, k_2]$ код с пораждащи матрици съответно G_1 и G_2 и радиуси на покритие R_1 и R_2. Кодът C, който е директна сума на пространствата C_1 и C_2, е $[n_1+n_2, k_1+k_2, \min\{d_1, d_2\}]$ код и се поражда от матрицата

$$G = \begin{bmatrix} G_1 & 0 \\ 0 & G_2 \end{bmatrix}.$$

Кодът C има радиус на покритие

$$R = R_1 + R_2. \tag{1.11}$$

- *Удължаване на код* [48]. Нека C е линеен $[n, k]$ код с пораждаща матрица G. Прибавяме един стълб към пораждащата матрица G и получаваме матрицата G'. Разглеждаме $[n+1, k]$ кода C', породен от матрицата G', който се нарича удължен за кода C. Радиусът на покритие на кода C' е или същият като на кода C, или с едно по-голям от него. Ако се прибавя нулева координата, то радиусът на покритие на кода се увеличава с 1.

- *Скъсяване на код* [48]. Ако изрежем s координати от един код C, ще получим нов код C''. Радиусът на покритие на скъсения код C'' намалява с не повече от s спрямо радиусът на покритие на C.

- *Слепване на кодове* [48]. Нека C_1 е $[n_1, k_1]$ код и C_2 е $[n_2, k_2]$ код с пораждащи матрици съответно G_1 и G_2 и $k_1 \leq k_2$. Кодът C с пораждаща матрица $G = [G_1' \mid G_2]$, където G_1' е матрица получена от G_1 с допълване с $k_2 - k_1$ реда нули, е $[n_1 + n_2, k_2, d]$ код, където $d \geq \min\{d_1, d_2\}$. Тази конструкция на слепване може да дава различни кодове, ако се избират различни пораждащи матрици на кодовете C_1 и C_2. За радиуса на покритие на получения код е изпълнено следното неравенство:

$$R(C) \geq R(C_1) + R(C_2). \tag{1.12}$$

- Нека пораждащата матрица на един линеен код има следния вид:

$$G = \begin{bmatrix} G_1 & A \\ 0 & G_2 \end{bmatrix}.$$

Ако C_1 и C_2 са кодовете, породени от матриците G_1 и G_2, то тогава е в сила следната граница за радиуса на покритие [150]:

$$R(C) \leq R(C_1) + R(C_2). \tag{1.13}$$

Известна е [196] и граница за радиуса на покритие на кодове над произволни крайни полета без нулева координата:

$$R(C) \leq \left\lfloor \frac{n(q-1)}{q} \right\rfloor. \tag{1.14}$$

В тази граница се достига равенство, ако C е код с размерност 1 или ако кодът е q пъти повторен код с по-малка дължина. Като се използва този факт, може да бъде получен радиусът на покритие на $[sk, k]$ код, който представлява s пъти повторено пространството F_q^k. Такъв код има пораждаща матрица $G = [I_k I_k \ldots I_k]$, еквивалентна на

$$G = \begin{bmatrix} E & 0 & 0 & . & . & . & 0 \\ 0 & E & 0 & . & . & . & 0 \\ 0 & 0 & E & . & . & . & 0 \\ . & . & . & . & . & . & . \\ 0 & 0 & 0 & . & . & . & E \end{bmatrix}.$$

В тази матрица $E = (\underbrace{1 \ldots 1}_{k})$, а $0 = (\underbrace{0 \ldots 0}_{k})$. Радиусът му на покритие е

$$R = sR_1 = s \left\lfloor \frac{k(q-1)}{q} \right\rfloor, \tag{1.15}$$

където $R_1 = \left\lfloor \frac{k(q-1)}{q} \right\rfloor$ е радиусът на покритие на кода с пораждаща матрица E.

Твърдение 1.5.2. *[193] Нека C е цикличен $[n, k]$ код над F_q с пораждащ полином $g(x) = g_0 + g_1 x + \ldots + g_r x^r$, $r = n - k$. Тогава*

$$R(C) = R^{(k)} \leq R^{(k-1)} \leq \ldots \leq R^{(2)} \leq R^{(1)},$$

където $R^{(s)}$ е радиусът на покритие на $[r + s, s]$ кода $C^{(s)}$ с

$$G^{(s)} = \left.\begin{bmatrix} g_0 & g_1 & . & . & . & g_r & 0 & . & . & . & 0 \\ 0 & g_0 & g_1 & . & . & . & g_r & 0 & . & . & 0 \\ . & . & . & . & . & . & . & . & . & . & . \\ 0 & . & . & . & 0 & g_0 & g_1 & . & . & . & g_r \end{bmatrix}\right\} s$$

1.6 Минимален радиус на покритие на $[n, k]$ кодове

Един от проблемите на теорията на кодирането е свързан с намирането на такива кодове, при които кодовите думи са разположени така, че нито един вектор от пространството F_q^n да не е много далече от най-близката кодова дума. От математическа гледна точка това е свързано с определянето на стойността на функцията $t_q[n, k]$, която е най-малкият радиус на покритие на q-ичен $[n, k]$ код при фиксирани $q,\ n$ и k.

Като е използвана конструкцията външна директна сума на кодове, в [49] са получени следните граници за $t_q[n, k]$:

$$t_q[n_1 + n_2, k_1 + k_2] \leq t_q[n_1, k_1] + t_q[n_2, k_2] \tag{1.16}$$

$$t_q[n + 1, k] \leq t_q[n, k] + 1 \tag{1.17}$$

$$t_q[n + 1, k + 1] \leq t_q[n, k] \tag{1.18}$$

$$t_q[n, k] \leq t_q[n + 1, k]. \tag{1.19}$$

Глава 2

Програмни реализации на основните методи за пресмятане на параметрите, определящи възможностите за контрол на грешки, на линейни кодове

2.1 Параметри, определящи възможностите за контрол на грешки, на линейни кодове

Основните параметри на един линеен код са дължината n, размерността k, минималното разстояние d и радиусът на покритие R [62]. Те определят възможностите за откриване и коригиране на грешки на кода.

Дефиниция 2.1.1. *Блоковият код C открива t грешки, ако всяка дума a', получена чрез пермутация на $1, 2, \ldots, t$ от позициите на кодовата дума a, не е кодова дума.*

Теорема 2.1.2. *Кодът C открива t грешки, тогава и само тогава когато $d > t$.*

Дефиниция 2.1.3. *Блоковият код C коригира t грешки, ако за всяка кодова дума a и всяка дума a', получена чрез пермутация на $1, 2, \ldots, t$ от позициите на a, хеминговото разстояние $d(a, a')$ е строго по-малко от хеминговото разстояние на всяка друга кодова дума до a'.*

Теорема 2.1.4. *Кодът C коригира t грешки, тогава и само тогава когато $d > 2t$.*

С други думи, един линеен $[n, k, d]_q$ код открива до $d-1$ грешки и коригира до $t = \left\lfloor \frac{d-1}{2} \right\rfloor$ грешки.

От всичко казано до тук и от (1.1), (1.2), (1.3) и (1.4) можем да заключим, че параметрите определящи възможностите за контрол на грешки на линеен код са минималното разстояние d, радиусът на покритие R и тегловните спектри $\bar{A}$, $\bar{\alpha}$ и $Q_{h,l}$. Точните стойности на тези параметри от теоретични съображения са определени само за някои специални класове кодове. Другият подход, който се прилага, за да бъдат определени тези параметри, е използването на компютърни пресмятания. Каква е сложността на задачите за пресмятане на минимално разстояние, радиус на покритие и тегловни разпределения на код ще покажем в следващия раздел.

2.2 Сложност на задачата за пресмятане на минимално разстояние, радиус на покритие и тегловни разпределения на код

Болшинството дискретни и комбинаторни задачи допускат намиране на решение чрез пълно изчерпване на възможностите. За съжаление броят на стъпките при пълното изчерпване расте експоненциално с увеличаване на размера на задачата. Счита се, че задача за изчерпване се решава ефективно, ако съществува алгоритъм, който решава тази задача за време ограничено от полином зависещ от размера на задачата.

Под *задача* ще разбираме някакъв общ въпрос, на който трябва да се даде отговор. Обикновено задачата съдържа няколко *параметъра*, чиито конкретни значения са определени. Обикновено тя се определя от следната информация:

(1) *вход*, който е списък на параметрите на задачата;

(2) *свойство*, което трябва да удовлетворява отговора, т.е. решението на задачата.

Под *алгоритъм* ще разбираме обща, изпълнявана стъпка по стъпка, процедура за решаване на задача.

Дефиниция 2.2.1. *Класът P е множество от задачи, които магат да бъдат решени от алгоритъм, който със сигурност ще завърши работата си след брой стъпки, ограничени от полином, зависещ от дължината на входа.*

P отговаря на класа алгоритми работещи за *полиномно време*. Задачите от класа P се считат податливи на решаване, докато тези извън него - неподатливи на решаване.

Недетерминистичен алгоритъм е такъв алгоритъм, който когато трябва да избере измежду две алтернативи, може да създаде две свои копия, които да продължат да работят паралелно. Изпълнено многократно, това разделяне може да доведе до експоненциално нарастване на броя на копията. Казва се, че алгоритъмът решава задачата, ако всяко от неговите копия дава коректен резултат.

Дефиниция 2.2.2. *Класът NP е множество от задачи, които могат да бъдат решени от недетерминистичен алгоритъм, чието време за изпълнение е ограничено от полином зависещ от дължината на входа.*

С други думи, класът NP е множество от задачи, които се решават от алгоритми за търсене с връщане. Ясно е, че NP отговаря на класа от алгоритми работещи за *недетерминистично полиномно време*.

През 1971 Cook [9, Теорема 10.3] доказва, че една задача от NP, наречена *задача за удовлетворимост*, има следното интересно свойство. Всяка задача (p) от NP може да бъде сведена до задачата за удовлетворимост. Тогава, ако може да бъде намерен алгоритъм решаващ за полиномно време задачата за удовлетворимост, то този алгоритъм може да бъде модифициран до алгоритъм решаващ за полиномно време задачата (p). Учените от много време вече работят върху задачи от класа NP и не са успели да намерят алгоритъм с полиномна времева сложност за нито една от тях. Това води до допускането, че задачата за удовлетворимост не може да бъде решена за полиномно време и следователно $NP \neq P$.

По-късно Karp [121] обръща нещата и показва, че задачата за удовлетворимост от своя страна може да се сведе до други (по-точно до 21) задачи от NP. Следователно ако поне една от тези задачи може да бъде решена за полиномно време, то за такова време може да бъде решена всяка задача от NP, т.е. $NP=P$.

Една задача x е *NP-пълна* или *NP-C*, ако $x \in NP$ и всяка задача от NP може да бъде сведена до x за полиномно време. Една задача е *NP-трудна*, ако известна NP-пълна задача може да се сведе до нея за полиномно време. Книгата на Garey и Johnson [97] съдържа хиляди задачи, за които е известно че са NP-трудни. Тогава, ако много вероятното предположение, че $NP \neq P$ е истина, то нито една NP-пълна задача не може да бъде решена за полиномно време.

През 1976 г. Stockmeyer [185] въвежда *полиномна йерархия* на задачите от P и NP.

Дефиниция 2.2.3. *[185] Йерархията на задачите, решавани за полиномно вре-*

ме, е $\{\sum_k^p, \prod_k^p, \Delta_k^p : k \geq 0\}$, *където*

$$\sum_0^p = \prod_0^p = \Delta_0^p;$$

и за $k \geq 0$

$$\sum_{k+1}^p = NP(\sum_k^p),\ \prod_{k+1}^p = NP - C(\sum_k^p),\ \Delta_{k+1}^p = P(\sum_k^p).$$

В частност $\sum_1^p = NP$ и $\prod_1^p = NP$-C.

За определяне на параметрите, от които зависи поведението при контрол на грешки на произволен код, трябва да се решават задачи, които правят пълно изчерпване и следователно ще имат експоненциална сложност по отношение на входа. Оценки на сложността на алгоритмите за определяне на основните параметри на линейни кодове са направени в работите на няколко автора.

През 1978 г. Berlecamp, McEliece and van Tilborg [14] показват че следната задача е NP-пълна:

Вход. Двоична матрица A, двоичен вектор y и цяло число $w > 0$.

Свойство. Съществува двоичен вектор x с Хемингово тегло $wt(x) = w$, такъв че $xA = 0$.

Това означава, да се определи дали за зададена проверочна матрица A и тегло w съществува кодова дума с такова тегло. Авторите изказват предположение, че задачата остава NP-пълна и когато се замени $wt(x) = w$ с $0 < wt(x) \leq w$, т.е. задачата за проверка на горна граница за минималното тегло на код е също NP-пълна.

По-късно Ntafos и Hakimi [162] доказват, че следните задачи са NP-пълни:

1) Да се намери кодова дума с минимално тегло, което не се дели на k, за всяко $k \geq 2$. За $k = 2$ това е задачата за намиране на дума с минимално нечетно тегло.

2) Да се намери кодова дума с максимално тегло.

3) За зададени произволни цели числа w_1 и w_2, $0 < w_1 < w_2$, да се намери кодова дума x, такава че $w_1 \leq |x| \leq w_2$.

През 1997 г. Vardy [191] доказва, че следната задача $\prod$ е NP- пълна:

Вход. Двоична $m \times n$ матрица H, двоичен вектор y и цяло число $w > 0$.

Свойство. Съществува ненулев вектор $x \in F_2^n$ с тегло $\leq w$, такъв че $Hx^t = 0$.

Това означава, че пресмятането на минималното разстояние на двоичен линеен код е NP-трудна задача. Наистина, нека C е линеен код зададен с проверочната матрица H и d да е минималното разстояние на този код. Ако d е известно, то свойството на $\prod$ може да бъде проверено просто като се сравнят d и w. От друга страна, ако $\prod$ може да бъде решена, то и d може да бъде намерено чрез алгори-

тъма за $\prod$, изпълнен за $w = 1, 2, \ldots$ до тогава докато се намери първия вектор, който удовлетворява свойството.

От всичко казано до тук може да се заключи, че и определянето на тегловото разпределение на линеен код е NP-трудна задача.

В [154] McLoughlin разглежда следния проблем.

Вход. Двоична $m \times n$ матрица A и цяло число w.

Свойство. Съществува двоичен вектор x с Хемингово тегло $\leq w$, такъв че $xA = y$.

Това свойство отговаря на определянето на горна граница за радиуса на покритие на код, т.е. че радиусът на покритие е не по-голям от w.

McLoughlin показва, че тази задача е NP-трудна и по точно е от класа $\prod_2^p$ от полиномната йерархия, тъй като свойството съдържа последователността $\forall y\ \exists x$, която е характеристика за този клас. Като следствие е показано, че допълващата го задача за проверка на долна граница на радиуса на покритие е $\sum_2^p$-пълна. Тъй като всяка от тези две задачи може да бъде сведена до задачата за пресмятане на радиус на покритие, тази задача е едновременно $\prod_2^p$-трудна и $\sum_2^p$-трудна. Също така, тези две задачи могат да бъдат сведени и до задача за определяне на тегловото разпределение на лидерите на съседни класове $\bar{\alpha}$. Следователно и тази задача принадлежи на класа на NP-трудните задачи.

В [132] е доказана следната теорема.

Теорема 2.2.4. *Задачата за пресмятане на $P_{ue}(C, \varepsilon)$ като функция на някое рационално ε и на пораждаща матрица G е NP-трудна.*

Всички представени до тук резултати показват, че задачите за пресмятане на минимално разстояние, радиус на покритие, тегловни разпределения на кода, на съседните му класове и на лидерите на съседни класове (основните характеристики, определящи поведението при контрол на грешки на линеен код) имат експоненциална сложност.

2.3 Алгоритми за пресмятане на тегловни разпределения и на радиус на покритие на код

В теорията на кодирането не са известни резултати, които да определят радиуса на покритие или тегловните разпределения на код в общия случай. Известните многобройни горни и долни граници за тези параметри, често са трудно проверими и рядко дават възможност само с пресмятането им да се определи точната им

стойност. По тази причина, много широко използван подход от изследователите, работещите по проблематиката, е да се правят компютърни пресмятания. Както вече беше показано в предишния раздел, задачите свързани с определянето на точните стойности на R, $\bar{A}$, $\bar{\alpha}$, $Q_{h,l}$ са NP-трудни и следователно за решаването им няма известни алгоритми с линейна времева сложност. Тук ще представим използваните в дисертацията методи за пресмятане на R, $\bar{A}$, $\bar{\alpha}$, $Q_{h,l}$, както и особеностите на техните компютърни реализации.

2.3.1 Алгоритми за пресмятане на тегловни разпределения на линеен код

Два детерминистични алгоритъма за пресмятане на първите елементи от тегловните разпределения на циклични кодове са представени в [12]. Първият от тях е добър за кодове с малки размерности - до $n/2$, а вторият - за кодове с размерности по-големи от $n/2$. В [105] и [151] са дадени методи за пресмятане на тегловното разпределение на неразложими циклични кодове. Segal и Ward [177, 200] са пресметнали тегловните разпределения на някои неразложими циклични кодове.

Целта на тази дисертация - да се изследват възможностите за контрол на грешки на класове линейни кодове, предполага разработените програмни средства да могат да бъдат използвани за различни линейни кодове и да не са тясно свързани със свойствата на кодовете от определен клас. Това от една страна ги прави по-универсални, а от друга прилагането им за различни класове кодове предполага тяхното много по-добро тестване, а оттук и по-голямата им надеждност. В този раздел ще представим общите методи за пресмятане на d, R, $\bar{A}$, $\bar{\alpha}$ и $Q_{h,l}$ използвани в дисертацията, ще опишем алгоритмите, особеностите на техните програмни реализации и сложността им по време и памет. Всички техни програмни реализации са на езика C++.

Най-общият метод използван за решаване на задачата за пресмятането на тегловното разпределение на линеен $[n, k]$ код над $GF(q)$ е да се тестват всичките му q^k кодови думи. Тази задача включва и определянето на минималното разстояние на код. За целта се използва код на Grey, който е такова подреждане на последователностите с коефициенти на линейната комбинация, определяща от кои от редовете на пораждащата матрица ще се формира поредната кодова дума, че всеки следващ се получава само с добавяне на 1 към някоя от компонентите на предишния. Има много начини за подреждане на кодовите думи, така че да удовлетворяват това изискване, но най-често използвания е отразеният код на Grey. Такъв подход за пресмятане на минималното разстояние и на тегловното

разпределение на q-ични кодове е използван в [4, 5, 147].

В дисертацията се използва q-ичен (q е просто число) код на Grey $G(n)$ с дължина n, който се дефинира по следния начин:

$$G_q(n) = \begin{bmatrix} 0 & G_q(n-1) \\ 1 & G'_q(n-1) \\ 2 & G_q(n-1) \\ \vdots & G_q(n-1) \\ q-1 & \begin{bmatrix} G_q(n-1), q \equiv 0 \ mod(2) \\ G'_q(n-1), q \neq 0 \ mod(2) \end{bmatrix} \end{bmatrix}$$

$$G'_q(n) = \begin{bmatrix} q-1 & \begin{bmatrix} G_q(n-1), q \equiv 0 \ mod(2) \\ G'_q(n-1), q \neq 0 \ mod(2) \end{bmatrix} \\ \vdots & G_q(n-1) \\ 2 & G'_q(n-1) \\ 1 & G_q(n-1) \\ 0 & G'_q(n-1) \end{bmatrix}.$$

$G_n(0) = G'_n(0)$ са матрици с нулеви размерности. Термина отразен идва от свойството на кода $G'_n(i)$ да е отражение на $G_n(i)$. Отразеният код на Grey може да бъде конструиран с помощта на Алгоритъм 1.

Algorithm 1. q-ичен отразен код на Grey(int q,n)

```
int g[n];
Gray(n);
 procedure Gray(int n)
{
 int j;
 if( n<=0 ) break;
 else
 {
  for( j=0; j<q; j++ )
  {
   g[n]=j;
   if ( (j%2)==0 ) Gray(n-1);
   else Grayr(n-1);
  }
 }
}
```

```
procedure Grayr(int n)
{
 int j;
 if( n<=0 ) break;
 else
 {
  for( j=q-1; j>0; j– )
  {
   g[n]=j;
   if ( (j%2)==0 ) Grayr(n-1);
   else Gray(n-1);
  }
 }
}
```

Този алгоритъм може да има и имплементация със стек (виж например [4, 79]), но заради яснотата на изложението тук предлагаме рекурсивната.

Един удобен начин да се опише кодът на Grey е с *последователност на преходите* [127, 171], която показва коя позиция се променя при преминаването от текущата към следващата дума. Ако стълбовете на $G(n)$ са номерирани започвайки от 1 от дясно на ляво, то последователността на преходите за двоичния отразен код на Grey с дължина n е следната:

$T_n = t_1, t_2, \ldots, t_{2^n-1},$

като

$T_1 = 1,$

и

$T_{n+1} = T_n, n+1, -T_n^r.$

Ще отбележим, че това представяне се различава малко от дефиницията от [127, 171], като тук е добавен знак, за да отбележим увеличаването или намаляването на стойността в позицията.

T_n изглежда по подобен начин за q-ичния отразен код на Gray. Тъй като позицията може да приема всяка стойност от 0 до $q-1$, то трябва да отчитаме, че ще имаме последователност от увеличаващи се и намаляващи стойности във всяка позиция. Последователността на преходите за q-ичен отразен код на Grey с дължина n ще е следната:

$T_n = t_1, t_2, \ldots, t_{q^n-1},$

като

$T_1 = 1, 1, \ldots, 1,$

и имаме общо $q-1$ единици. Тогава T_{n+1} се определя по следния начин:

$$T_{n+1} = T_n, n+1, -T_n^r, n+1, T_n, \ldots, n+1, \begin{cases} -T_n^r & q \equiv 0 \ mod(2), \\ T_n & q \neq 0 \ mod(2), \end{cases}$$

където $-T_n^r$ е отражението на T_n взето с обратен знак. За илюстрация, нека да разгледаме троичен код на Grey за $n = 3$. Последователността на преходите за този код е:

$T_3 = 1, 1, 2, -1, -1, 2, 1, 1, 3, -1, -1, -2, 1, 1, -2, -1, -1, 3, 1, 1, 2, -1, -1, 2, 1, 1.$

Нека C да е линеен $[n, k]$ код над $GF(q)$ и q да е просто число. Тогава C съдържа q^k кодови думи и може да бъде представен като линейната обвивка на k базисни вектора, формиращи проверочната матрица G.

$$G = \begin{bmatrix} g_{0,0} & g_{0,1} & g_{0,2} & g_{0,3} & \cdots & g_{0,n-1} \\ g_{1,0} & g_{1,1} & g_{1,2} & g_{1,3} & \cdots & g_{1,n-1} \\ g_{2,0} & g_{2,1} & g_{2,2} & g_{2,3} & \cdots & g_{2,n-1} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ g_{k-1,0} & g_{k-1,1} & g_{k-1,2} & g_{k-1,3} & \cdots & g_{k-1,n-1} \end{bmatrix}.$$

Кодовите думи на C се получават като се генерират всички q^k комбинации от редовете на G. Тогава Алгоритъм 1 трябва да бъде модифициран така, че да прави тези линейни комбинации от редовете на G. Това става като се използва последователността на преходите на кода на Grey. Имплементацията е представена в Алгоритъм 2, който позволява определянето на тегловото разпределение на кода C.

Algorithm 2. Q-ARY WEIGHT DISTRIBUTION(int q,k,n,g[n],G[k,n])

```
int g[n];
Gray(k);

procedure Gray(int k)
{
 int i,j;
 if( k<=0 ) compute weight;
 else
 {
  Grey(k-1);
  for( j=0; j<q; j++ )
  {
   for( i=0; i<n; i++ )
```

```
        g[i]=g[i]+G[k][i] mod q;
    if ( (j%2)==0 ) Gray(k-1);
    else Grayr(k-1);
   }
  }
}

procedure Grayr(int k)
{
 int i,j;
 if( k<=0 ) compute weight;
 else
 {
  Greyr(k-1);      for( j=q-1, j>0; j– )
  {
   for( i=0; i<n; i++ )
        g[i]=g[i]-G[k][i] mod q;
   if ( (j%2)==0 ) Grayr(k-1);
   else Gray(k-1);
  }
 }
}
```

Алгоритъмът изисква nq^k събирания в $GF(q)$, за да генерира тегловото разпределение. В двоичния случай събирането по модул 2 е изключващо или и ако n-те бита се пакетират в една или няколко думи, броят на операциите ще се редуцира до sq^k, където s е броят на думите, в които са пакетирани тези n бита. За случаите когато $q > 2$ могат също да се правят пакетирания и да се използват таблици за имплементиране на събирането. Такива решения са предложени в [1, 25, 148]. Програмата, която реализира Алгоритъм 2 е наречена `QWD`.

2.3.2 Алгоритми за пресмятане на радиус на покритие на линеен код

Ако C е линеен $[n, k]$ код над F_q с радиус на покритие R, то основните методи, по които се пресмята радиусът на покритие са следните.

Метод 1: Този метод се основава на Лема 1.2.12. Ако H е една проверочна

матрица на линеен код, то радиусът на покритие на този код е най-малкото цяло число R, такова че всеки $(n-k)$-мерен не нулев вектор-стълб може да се получи като линейна комбинация на най-много R стълба на H.

Метод 2: Този метод се основава на факта, че радиусът на покритие на линеен код е равен на теглото на най-тежкия лидер на съседен клас. Ако кодът е в систематичен вид, по един представител от всеки съседен клас може да се получи, ако се разгледат всички думи от вида $(\underbrace{0,\ldots,0}_{k},a)$, където $a \in GF(q^{n-k})$.

Метод 3: [75] Основава се на твърдението, че ако R_0 е фиксирано цяло число, то $R \leq R_0$, тогава и само тогава когато всяка дума с тегло R_0+1 е на разстояние от кода, не надминаващо R_0.

Метод 1 и Метод 3 са използвани в работата на Downie и Sloane [75] за пресмятане на радиусите на покритие на двоичните циклични кодове с дължини до 31. Великова и Манев [192] са продължили изследването за двоичните циклични кодове с дължини 33, 35 и 39 като също са използвали тези два метода. По- късно Dougherty и Janwa [74] използват Метод 1, за да пресметнат радиусите на покритие на всички двоични циклични кодове с дължини до 64 и размерности до 28. За троични кодове тези методи са приложени в [10] и [11].

Сега ще опишем особеностите на компютърните реализации на тези методи.

Алгоритъм 3: При реализацията на Метод 1 се използва масивът el с m елемента, където $m = \left\lfloor \frac{u}{8} \right\rfloor + 1$, а u е броя на всички $(n-k)$-мерни вектори, т.е. за всяка $(n-k)$-орка е отделен по 1 бит. Стъпките на алгоритъма са следните.

1) Първоначално целият масив се инициализира с нули.

2) Отбелязват се битовете от el, съответстващи на стълбовете h_i $(1 \leq i \leq n)$ на проверочната матрица H, като им се присвоява стойност 1.

3) Генерират се последователно всички линейни комбинации от r =2, 3 и т.н. стълба на H.

4) За всеки вектор получен като резултат от тези комбинации, на съответния му бит от el се присвоява стойност 1, ако тя все още е 0.

Стъпки 3) и 4) се повтарят докато всички битове на el станат равни на 1 и тогава $R(C) = r$.

Едновременно с това, за всяка стойност на $r = 2, 3, \ldots$ определяме броя N_r на единиците добавени в el, а разликите $N_r - N_{r-1}$ за $r = 1, 2, \ldots, R(C)$ определят спектъра на лидерите на съседните класове на кода.

Генерирането на всички r-елементни подмножества, където $r = 2, 3, \ldots, R(C)$, на n-елементното множество от стълбовете на H е в лексикографски ред (според алгоритъма от [171]) и става за линейно време. Тогава общото време необходимо

за определяне на радиуса на покритие на кода зависи от броя на линейните комбинации, които трябва да се направят със стълбовете на H и е пропорционално на $\sum_{i=0}^{r} \binom{n}{i} 2^i$, а необходимата памет за съхраняване на масива el е $\left\lfloor \frac{q^{n-k}}{8} \right\rfloor + 1$. Вижда се, че приложението на метода е лимитирано от необходимата памет за разполагане на масива el и той, макар и сравнително бърз, е приложим за кодове с малки ко-размерности $(n-k)$. Програмата реализираща този алгоритъм е `CovRadM1`.

Алгоритъм 4: За да бъде приложен Метод 2, пораждащата матрица на кода трябва да е в систематичен вид.

1) Привежда се в систематичен вид пораждащата матрица на кода.

2) Генерират се и се запомнят в паметта всички кодови думи.

3) Разглеждат се всички вектори с дължина n от вида $(\underbrace{0,\ldots,0}_{k},a)$, $a \in F_q^{n-k}$ и се пресмята тяхното разстояние до кода. Най-голямото от тези разстояния е търсеният радиус на покритие.

Ако по време на изпълнението на стъпка 3) се намери вектор, който е на разстояние от кода равно на горна граница за радиуса му на покритие, то това е и точната му стойност и тестването на останалите вектори се прекратява.

В случаите когато броят на кодовите думи е голям, те не могат да се съберат в паметта и затова се налага кодът да се генерира за всеки вектор поотделно, т.е. q^{n-k} пъти.

За да определим спектъра на лидерите на съседни класове, трябва да се премахне проверката за достигната горна граница за радиуса на покритие и да се генерира по един представител от всеки съседен клас. Техните разстояния до кода са този спектър. Когато определяме тегловните разпределения в съседните класове, трябва да генерираме всеки от тях чрез прибавяне на всички вектори от вида $(\underbrace{0,\ldots,0}_{k},a)$ към всички кодови думи на кода.

За всеки от разглежданите q^{n-k} вектора се правят по q^k сравнения в най-лошия случай и следователно броят на пресмятанията е пропорционален на q^n. Тъй като генерирането на кода в случаите, когато той не се събира в паметта, става за линейно (спрямо броя на кодовите думи) време като се използва код на Grey, общата оценка за сложността по време на алгоритъма не се променя по същество. Необходимата памет за разполагане на кода е $q^k \left\lceil \frac{n}{s} \right\rceil$, където с s е означен броят на елементите на полето, записани в един байт. Програмната реализация на Алгоритъм 4 е `CovRadM2`.

Алгоритъм 5: При реализацията на метод 3 се разглеждат всички думи

с тегло $R_0 + 1$ и се пресмята разстоянието им до кода. Както и при предния алгоритъм, когато кодът може да се събере в паметта, той се генерира само веднъж в началото. В противен случай, това се прави за всяка разглеждана дума. Ако всички изследвани думи се окажат на разстояния от кода не надминаващи R_0, се прави извод, че $R(C) \leq R_0$. Иначе е изпълнено, че $R(C) > R_0$.

Броят на пресмятанията при този алгоритъм е приблизително равен на $\binom{n}{R_0+1} 2^{R_0+1} q^k$, а необходимата памет е както при Метод 2 - $q^k \left\lceil \frac{n}{s} \right\rceil$. Този метод е удачно да бъде използван в случаите, когато известните горна и долна граница за радиуса на покритие на кода имат близки стойности. Тогава, за да се определи точната му стойност, е достатъчно да се тестват само няколко значения на R_0. Алгоритъмът се реализира от програмата `CovRadM3`.

За решаването на конкретните задачи, резултатите от които са описани в следващите глави, освен тези основни програмни средства са разработвани и допълнителни или са правени модификации на основните програмни средства. Те ще бъдат представени в следващите глави в контекста на задачите, за чието решаване са използвани. За някои от задачите са разработени и допълнителни алгоритми и са направени техни програмни реализации. Те също ще бъдат представени по-нататък в изложението.

Един въпрос, който винаги възниква при представяне на резултати получени чрез компютърни пресмятания, е за коректността на тези резултати. Проверката на коректността на получените в дисертацията резултати е правена по два начина, извън стандартното тестване на програмите за предварително известни резултати, което всеки програмист прави. Единият използва предварително известни долни и горни граници за изследваните параметри и това, че получените резултати се намират в тези граници. Другият е свързан с пресмятането на някои от параметрите от две различни програми. За радиус на покритие например са използвани пресмятания по два от методите. За други задачи проверката е правена и като са сравнявани резултатите получени с разработените в тази дисертация програмни средства с тези получени с разработени от други колеги такива, каквато е например програмната система на Илия Буюклиев Q-EXTENSION [23]. В някои случаи резултатите са получени независимо от различни съавтори.

Глава 3

Радиус на покритие на класове линейни кодове

В тази глава са разгледани задачи за намиране на радиусите на покритие на някои класове троични линейни кодове.

Изследвани са троичните негациклични кодове с четни дължини до 26. Това изследване е продължение на предишна работа на автора [10], където са класифицирани всички троични циклични кодове с дължини до 25 и са пресметнати радиусите им на покритие. В изследването са разглеждани само негацикличните кодове с четни дължини, защото тези с нечетни дължини са еквивалентни на циклични кодове. Направена е класификация на всички троични негациклични кодове с четни дължини до 26 и са пресметнати минималните им разстояния, тегловните разпределения на кодовите им думи и радиусите им на покритие. Уточнени са 7 стойности на функцията $t_3[n,k]$, а за други три случая са подобрени горните граници.

Интересен клас линейни кодове са квази-съвършените кодове. В раздел 3.2 систематично са изследвани двоичните и троични линейни кодове и са направени класификации на квази-съвършени кодове с размерности до 14. Направен е и обзор на известните резултати за такива кодове над полета с два, три и четири елемента.

Ще отбележим, че като знаем радиуса на покритие на някои кодове от тях можем да получим нови като използваме някои от известните конструкции, представени в първа глава, като външна директна сума, удължаване и скъсяване на код, слепване на кодове. Новите кодове ще бъдат с точно известен радиус на покрите за първите две конструкции или ще знаем някаква граница за него за другите две.

3.1 Радиус на покритие на троичните негациклични кодове с дължини до 26

Констациклични кодове са линейни кодове, които са затворени относно констациклично завъртане на кодовите думи. Констацикличното завъртане на n-орката $(a_0, a_1, \ldots, a_{n-2}, a_{n-1})$ води до n-орката $(ca_{n-1}, a_0, a_1, \ldots, a_{n-2})$, където c е фиксиран не нулев елемент на полето. Констацикличните кодове имат много от добре известните алгебрични свойства на цикличните кодове [145, Глава 7]. Един начин да бъде описан констацикличния код C е като идеал в пръстена на полиномите $F[x]/(x^n - c)$, затворен относно събиране и умножение по модул $x^n - c$. Може да се докаже, че C е главен идеал и като такъв той съдържа единствен полином от минимална степен, означен с $g(x)$, който поражда C, т.е. $C = \langle g(x) \rangle$. Пораждащият полином $g(x)$ трябва да е делител на $x^n - c$, и степента r на $g(x)$ определя броя на проверочните символи на C, т.е. $\langle g(x) \rangle$ е $[n, k]$ код, където $n - r = k$.

Когато $c = -1$, кодовете се наричат *негациклични* [13, Глава 9]. В този случай, $g(x)$ е делител на $x^n + 1$. Полиномът $h(x) = (x^n + 1)/g(x)$ е проверочен полином на кода C.

Тъй като $x^n + 1 = (x^{2n} - 1)/(x^n - 1)$, корените на полинома $x^n + 1$ са тези корени на $x^{2n} - 1$, които не са корени на полинома $x^n - 1$. Ако α е примитивен корен на полинома $x^{2n} - 1$, неговите нечетни степени са корени на полинома $x^n + 1$, т.е. корените на полинома $x^n + 1$ са нечетните степени на примитивния $2n$-ти корен на единицата. Следователно можем да характеризираме кода C с неговото дефиниционно множество, което се състои от всички j, такива че α^j е корен на $g(x)$.

В [138] е показано, че всеки троичен негацикличен код с нечетна дължина е еквивалентен на цикличен код. Радиусите на покритие на всички троични циклични кодове с дължини до 25 са пресметнати в [10]. Затова по-нататък ще разглеждаме само троичните негациклични кодове с четни дължини.

Като източник за троичните негациклични кодове с дължина 26 е използвана Таблица II-A от [138]. В таблицата е дадено разлагането на множители на $x^n + 1$ над GF(3) за $n \leq 50$. На основата на това разлагане и като е използвана програмата GEN_NCYCL са генерирани всички възможни троични негациклични кодове с дължини до 26. Като вход на тази програма се подават полиномите, на които се разлага $x^n + 1$, а в резултат тя генерира пораждащите полиноми на всички циклични кодове с дължина n и ги записва във файл. Определени са и

дефиниционните множества на всеки от кодовете като е използвана Таблица IV от [82].

Според [13, Теорема 5.81], два кода са еквивалентни, ако техните дефиниционни множества са еквивалентни с точност до умножение с цяло число взаимно просто с тяхната дължина. Проверката на това условие е направена с програмата $EQUIV$ и така са отделени само не еквивалентните кодове. Вход на $EQUIV$ са дефиниционните множества на всички генерирани от GEN_NCYCL кодове, а в резултат се получава списък на не еквивалентните такива. За кодовете от този списък са пресметнати минималните им разстояния и тегловните разпределения на кодовите им думи като е използвана програмата QWD. Радиусите на покритие на част от кодовете са получени аналитично, за останалите са използвани компютърни пресмятания по Метод 1 и Метод 2.

Резултатите (списък на не еквивалентните троични негациклични кодове с дължина до 26, техните минимални разстояния, радиуси на покритие $R(C)$ и пораждащи полиноми) са дадени в Таблица 3.1.1 в приложението. Пораждащите полиноми са представени като последователност от коефициенти като на първо място е коефициента пред най-високата степен. Например последователността 10201 означава $x^4 + 2x^2 + 1$.

Ще представим накратко математическите съображения, които са използвани, за да се определи аналитично радиусът на покритие на част от изследваните в този раздел кодове. Нека да означим с C_i кодът с номер i в таблицата с резултатите (Таблица 3.1.1 от приложението).

1) Кодовете $C_3, C_8, C_{20}, C_{27}, C_{29}, C_{48}, C_{54}, C_{60}$ и C_{74} са $[sk, k]$ кодове имащи, с точност до еквивалентност, пораждащи матрици от следния вид:

$$G = \begin{bmatrix} E & 0 & 0 & \dots & 0 & 0 \\ 0 & E & 0 & \dots & 0 & 0 \\ 0 & 0 & E & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 0 & E \end{bmatrix},$$

където $E = (\underbrace{1, ..., 1}_{s})$ и $0 = (\underbrace{0, ..., 0}_{s})$. Всеки код от този вид е директна сума на кодове с пораждаща матрица E (виж [47, Теорема 3.2.1]), т.е. $R(C) = kR(E) = k\left\lfloor \frac{2s}{3} \right\rfloor$.

2) Кодът C_{10} има проверочна матрица $H = [I_4|I_4]$, където I_4 е единична 4×4 матрица. Според Метод 1 $R(C_{10}) = 4$. По същия начин са определени радиусите на покритие на кодовете C_2, C_5, $C_{17}, C_{22}, C_{24}, C_{31}, C_{51}, C_{56}$ и C_{63}.

3) Векторът $(0, 0, 0, 0, 0, 0, 0, 0, 2, 1, 2, 1, 2, 1, 2, 1, 1, 2, 0, \ 0, 2, 1, 0, 0)$ е на разстояние 12 от кода C_{61}, т.е. $R(C_{61}) \geq 12$. От друга страна $R(C_{61}) \leq R^{(3)} = R[19, 3] = 12$

(виж [193, Твърдение1.2.2]). Следователно $R(C_{61}) = 12$.

4) Нека пораждащата матрица на линейния код C да е от вида $G = \begin{bmatrix} G' & A \\ 0 & G'' \end{bmatrix}$. Ако C' и C'' са кодове с пораждащи матрици G' и G'', то $R(C) \leq R(C') + R(C'')$ [150]. За кодът C_{72}, C' е $[12, 6]$ код с $R(C') = 4$, а C'' е $[14, 2]$ код с $R(C'') = 7$. Следователно $R(C_{72}) \leq 11$. Векторът $(0,0,0,0,0,0,0,0,1,0,0,1,2,2,1,1,2,2,1,2,1,2,1,0,0,0)$ е на разстояние 11 от C_{72}, т.е. $R(C_{72}) \geq 11$. Следователно $R(C_{72}) = 11$.

Посъщия начин, за кодът C_{73}, C' е $[8, 4]$ код с $R(C') = 2$, а C'' е $[18, 2]$ код с $R(C'') = 12$. Следователно $R(C_{73}) \leq 14$. Векторът $(0,0,0,0,0,0,2,1,2,1,1,2,2,1,2,1,2,1,2,1,0,0,0,0,0,0)$ е на разстояние 14 от C_{73}, т.е. $R(C_{73}) \geq 14$. Следователно $R(C_{73}) = 14$.

Ще отбележим, че кодовете $C_4, C_6, C_{11}, C_{18}, C_{30}, C_{32}$ и C_{65} са квази-съвършени.

Граници и точни стойности за функцията $t_3[n, k]$ за троични линейни кодове с $n \leq 27$ са дадени в [11, Таблица II]. Като са използвани вече определените в този раздел радиуси на покритие на троичните негациклични кодове с дължини до 26 в следващото твърдение са получени някои точни стойности и нови горни граници за $t_3[n, k]$.

Твърдение 3.1.1. *1)* $t_3[20, 6] = 7 - 8$.

2) $t_3[20, 10] = t_3[20, 11] = t_3[21, 11] = 4$.

3) $t_3[24, 12] = t_3[25, 13] = 5$.

4) $t_3[21, 10] = t_3[22, 11] = 5$

5) $t_3[22, 10] = 5 - 6$.

6) $t_3[25, 12] = 5 - 6$.

Доказателство.

Според [11, Таблица II], $t_3[20, 6] = 7-9$, $t_3[20, 10] = t_3[20, 11] = t_3[21, 11] = 4-5$, $t_3[24, 12] = t_3[25, 13] = 5-6$, $t_3[21, 10] = t_3[22, 11] = 5-6$, $t_3[22, 10] = t_3[25, 12] = 5-7$.

Беше пресметнато, че $[20, 6]$ кодът C_{47} има радиус на покритие 8 и $[20, 10]$ кодът C_{41} има радиус на покритие 4. Следователно $t_3[20, 6] = 7 - 8$ и $t_3[20, 10] = 4$.

От границата (1.12) за $t_q[n, k]$, $t_3[24, 12] \leq t_3[20, 10] + t_3[4, 2] = 4 + 1 = 5$, от където получаваме $t_3[24, 12] = 5$.

Тъй като $t_3[21, 10] \leq t_3[20, 10] + 1 = 5$, $t_3[22, 10] \leq t_3[21, 10] + 1 = 6$, $t_3[25, 12] \leq t_3[24, 12] + 1 = 6$, от границата (1.13) следва, че $t_3[21, 10] = 5$, $t_3[22, 10] = 5 - 6$ и $t_3[25, 12] = 5 - 6$.

Като използваме (1.14) от $t_3[20, 11] \leq t_3[20, 10]$ следва, че $t_3[20, 11] = 4$.

От границата (1.15) $t_3[25, 13] \leq t_3[24, 12]$ и следователно $t_3[25, 13] = 5$ и $t_3[22, 11] \leq t_3[21, 10]$. Тогава $t_3[22, 11] = 5$ и $t_3[21, 11] \leq t_3[20, 10]$, от където следва, че $t_3[21, 11] = 4$. ◇

Резултатите от този раздел са получени самостоятелно, публикувани са в [211] и са представени на International Symposium on Infromation Theory в Сорето, Италия през 2000 г.

3.2 Двоични и троични квази-съвършени кодове с малки размерности

В първа глава беше дефиниран радиусът на сферичната опаковка

$$e(C) = \left\lfloor \frac{d-1}{2} \right\rfloor$$

и беше представена границата на сферичната опаковка (1.1)

$$R(C) \geq e(C).$$

Радиусът на сферичната опаковка ни дава теглото на най-тежкия еднозначно коригируем вектор грешка, докато радиусът на покрите на кода е мярката за максималното тегло на коригируем (еднозначно или не) вектор грешка. Специален случай в теорията на кодирането са съвършените кодовете, за които $R(C) = e(C)$. Проблемът за намиране на всички съвършени кодове е поставен от Golay през 1949 и е напълно решен през 1973 независимо от Tietäväinen [188] и от Зиновьев и Леонтьев [207]. Съществуват само следните съвършени кодове:

- $[n, n, 1]_q 0$ кодовете за всяко $n \geq 1$,
- $[2s+1, 1, 2s+1]_2 s$ кодовете с повторение за всяко $s \geq 1$,
- кодът с дължина n, състоящ се само от една кодова дума,
- q-ичните кодове с параметри на кодовете на Хеминг,
- $[23, 12, 7]_2 3$ двоичния код на Golay,
- $[11, 6, 5]_3 2$ троичния код на Golay.

Следващата стъпка в тази посока е да се разглеждат квази-съвършени кодове, т.е. кодове, за които радиусът на сферичната опаковка и радиусът на покритие се различават с 1. Минималното разстояние на такива кодове е $2e+1$ или $2e+2$. Един съвсем естествен въпрос е кои кодове са квази-съвършени? Ясно е, че всеки код с радиус на покритие 1 и минимално разстояние 1 или 2 е квази-съвършен. Следователно кодовете с радиус на покритие 1 не са интересни и нашето изследване ще се фокусира върху кодове с радиус на покрите поне 2.

Първо ще направим един кратък обзор на известните до момента резултати за съществуване на квази-съвършени кодове с радиус на покрите по-голям от 1.

Квази-съвършените кодове с радиуси на покритие 2 и 3 са изследвани от различни автори. В частност, кодовете с параметри $[n,k,d]_q2$ $d=3,4$ са квази-съвършени. Тези кодове са свързани с 1-saturating множества в прективните пространства $PG(n-k-1,q)$ и в литературата са описани много такива кодове [56]–[60],[83],[94],[102],[103],[108]. В следващата теорема показваме конструкция на последователност от квази-съвършени кодове.

Теорема 3.2.1. *Да приемем, че съществува $[n,k,d]_q2$ квази-съвършен код с $n \leq \frac{q^{n-k}-1}{q-1}-2$ и $3 \leq d \leq 4$. Тогава съществува и $[n+1,k+1,3]_q2$ квази-съвършен код.*

Доказателство. Нека да добавим стълб към проверочната матрица на $[n,k,d]_q2$ кода, за да получим нов $[n+1,k+1,d_{new}]_qR_{new}$ код. Според Лема 1.2.12 $R_{new} \leq 2$. Тъй като дължината на новия код е $n+1 \leq \frac{q^{n-k}-1}{q-1}-1$, то е невъзможно $R_{new}=1$. За всеки добавен стълб е изпълнено $d_{new} \leq 3$. Ако този стълб не е получен чрез умножение на вече наличен в проверочната матрица стълб с не нулев елемент на полето $GF(q)$, то $d_{new}=3$. Подходящ избор на нов стълб е възможен тъй като $n \leq \frac{q^{n-k}-1}{q-1}-2$. ◇

$[n,n-4,5]_q3$ квази-съвършените кодове съответстват на пълни арки в проективното пространство $PG(3,q)$ и са изследвани в [57] и [59]. Обзор, както и нови резултати за двоични квази-съвършени кодове могат да бъдат намерени в [81]. Оказва се, че има много голям брой квази-съвършени кодове с минимално разстояние до 5.

Значително по-малко е известно за q-ичните квази-съвършени кодове с $q>2$. Една безкрайна фамилия от троични квази-съвършени кодове е известна от работата на Гашков и Сидельников [98]. Членове на фамилията са $[(3^s+1)/2,(3^s+1)/2-2s,5]_33$ кодовете. Две фамилии от четвъртични кодове $[(4^s-1)/3,(4^s-1)/3-2s,5]_43$ и $[(2^{2s+1}+1)/3,(2^{2s+1}+1)/3-2s-1,5]_43$ са представени в [76] и [99], а от Додунеков е показано, че техните членове са квази-съвършени кодове [65, 66].

За пръв път компютър е бил използван при конструиране на квази-съвършени кодове от Wagner през 1966 [197]. Той предлага алгоритъм, който използва свойствата на проверочната матрица на двоичен линеен квази-съвършен код, за да построи такива кодове. Като предварително фиксира броя на проверочните символи и на грешките, които да се коригират от кода, алгоритъмът намира по един

квази-съвършен код за всяка тествана дължина, ако такъв код съществува. С помощта на програмата, реализираща този алгоритъм, са намерени 27 нови двоични линейни квази-съвършени кода [197, 198]. Кодовете са с дължини между 19 и 55 и всички имат радиус на покритие 3. По-късно Simonis [179] доказва, че един от кодовете на Wagner ($[23, 14, 5]$) е единствен.

Тегловните разпределения и поведението при откриване на грешки на троичния $[13, 7, 5]$ квадратично-остатъчен код са изследвани в [212]. Показано е, че радиусът на покритие на кода е три, т.е. той е квази-съвършен. По-късно Додунеков и Данев [54] доказват, че този код е първият член на фамилия от троични квази-съвършени кодове с параметри $[(3^s - 1)/2, (3^s - 1)/2 - 2s, 5]_3$ за всички нечетни $s \geq 3$.

Всички представени до тук резултати водят до следния въпрос: „Колко рестриктивно е свойството на един код да е квази-съвършен, т.е. има ли не еквивалентни квази-съвършени кодове, тъй като всички посочени примери са само за един код за съответните параметри?“ Отговорът на този въпрос ще дадем сега.

Подходът, който сме използвали, е да се класифицират кодове със зададени параметри. Първо фиксираме размерността на кода и за тази размерност определяме възможните дължини и минимални разстояния на кодовете, които биха могли да са квази-съвършени. След това класифицираме всички такива кодове и определяме радиусите им на покритие. Така намираме всички квази-съвършени кодове със зададените параметри. За да определим параметрите на възможните кандидати за квази-съвършени кодове с радиус на покритие $e + 1$, ние вземаме предвид, че минималното разстояние на тези кодове може да има само две стойности: $2e + 1$ или $2e + 2$. За да бъдат определени възможните дължини на квази-съвършените кодове при фиксирана размерност, са използвани таблиците на Brouwer [29] с граници за обема на код и таблиците за минималния радиус на покритие на двоичните [47] и троичните [11] линейни кодове. Например да разгледаме двоичните линейни кодове с размерност 7. Тъй като търсим кодове с радиус на покритие поне 2, то минималното им разстояние трябва да е по-голямо от 3. От таблицата на Brouwer определяме, че първата дължина, за която има кодове с минимално разстояние 3, е 11. От таблицата за $t_2[n, k]$ от [47] виждаме, че стойността на тази функция за $[11, 7]$ кодовете е 2, т.е. съществуват кодове с радиус на покритие 2. Следователно е възможно да съществува квази-съвършен двоичен линеен $[11, 7, 3]2$ код. По аналогичен начин установяваме, че е възможно съществуването на двоични $[12, 7, 3]2$ квази-съвършени кодове. За дължина $n = 13$ от таблицата на Brouwer получаваме, че максималното възможно минимално разстояние за $[13, 7]$ код е 4, а от таблицата за $t_2[n, k]$ се вижда, че минималният радиус

на покритие на $[13, 7]$ код е 2. Тогава квази-съвършени биха могли да бъдат както $[13, 7, 4]2$, така и $[13, 7, 3]2$ кодовете. Максималното минимално разстояние на $[14, 7]$ кодовете е 4, а $t_2[14, 7] = 3$. Следователно не е възможно да съществуват квази-съвършени $[14, 7]$ кодове. По аналогичен, с предните случаи, начин се вижда, че е възможно съществуването на $[15, 7, 5]3$ квази-съвършени кодове. И това е последната дължина, на която е възможно съществуването на квази-съвършени кодове с размерност 7, защото максималните минимални разстояния на кодовете със следващите дължини растат по-бавно, отколкото минималния им радиус на покритие. За илюстрация на казаното, за следващите няколко дължини имаме следните максимални минимални разстояния $[16, 7, d_{max} = 6]$, $[17, 7, d_{max} = 6]$, $[18, 7, d_{max} = 7]$, $[19, 7, d_{max} = 8]$, $[20, 7, d_{max} = 8]$, докато минималните радиуси на покрите за кодове с тези параметри са съответно $t_2[16, 7] = 4$, $t_2[17, 7] = 4$, $t_2[18, 7] = 5$, $t_2[19, 7] = 5$, $t_2[20, 7] = 5$.

Така бяха определени параметрите, за които е възможно да съществуват двоични квази-съвършени кодове с размерности до 14 и на троични квази-съвършени кодове с размерности до 13. Тези параметри са представени в таблиците по-долу.

Таблица 3.2.1. Възможни параметри за двоични квази-съвършени кодове с размерности до 14

k/n	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
2			?																				
3	?	?						?															
4			?	?		?																	
5				?	?																		
6					?	?			?														
7						?	?	?		?													
8							?	?	?		?			?									
9								?	?	?		?	?		?	?							
10									?	?	?			?		?	?	?					
11											?	?			?	?	?	?	?				
12												?	?	?		?	?		?	?			
13													?	?	?		?	?	?		?	?	
14														?	?	?		?	?	?		?	?

Таблица 3.2.2. Възможни параметри за троични квази-съвършени кодове с размерности до 13

k/n	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
2	?																				
3				?																	
4			?	?	?																
5				?	?	?	?														
6					?	?		?	?												
7						?	?	?	?	?											
8							?	?	?												
9								?	?	?	?	?									
10										?	?		?	?		?		?			
11											?	?		?	?		?		?		
12												?	?		?	?		?		?	?
13													?	?		?	?		?	?	

След като са определени параметрите, за които е възможно да съществуват квази-съвършени кодове с фиксирана размерност, всички такива кодове са класифицирани като е използван подходът разработен от Илия Буюклиев в [23].

Две основни техники се прилагат при класификацията. Първата се основава на съкращаване, а втората - на скъсяване на кодове. При съкращаване на кода, изтриваме една или няколко негови позиции от всяка кодова дума. При скъсяване на кода, фиксираме една негова позиция, избираме кодовите думи, които имат 0 в тази позиция, изтриваме тази позиция и получаваме скъсения $[n-1, k-1]$ код на $[n, k]$ кода.

При съкращаването, в голяма част от случайите, размерността на кода остава непроменена. Това не е така когато всички не нулеви позиции на кода са премахнати. Нека G да е пораждащата матрица на линейния $[n, k, d]_q$ код C. Тогава *остатъчния* код $\mathrm{Res}(C, \mathbf{c})$ на C по отношение на кодовата дума c е кодът генериран от рестрикцията на G по стълбовете, където c има нулева стойност. Една долна граница за минималното разстояние на остатъчния код е следната.

Лема 3.2.2. *[64] Нека C е $[n, k, d]_q$ код и нека $\mathbf{c} \in C$ е с тегло w, където $d > w(q-1)/q$. Тогава $Res(C, \mathbf{c})$ е $[n-w, k-1, d']_q$ код с минимално разстояние $d' \geq d - w + \lceil w/q \rceil$.*

Като използваме тази операция в обратна посока, ние търсим $[n, k, d]_q$ код на базата на неговия остатъчен $[n-w, k-1, d']_q$ код по отношение на кодова дума с тегло w или на базата на съкратения с i координати $[n-i, k, d']_q$ код. Можем да приложим тези операции рекурсивно за остатъчния или съкратения код, съответно. Повтаряме процедурата докато получим като стартов код такъв

с малки параметри, позволяващи всички кодове с тези параметри да бъдат лесно класифицирани. Например, ако използваме като стартов код $[3,2,2]_2$ кода, можем да получим от него всички $[8,3,5]_2$ кодове. Вземаме само не еквивалентните от тях и получаваме всички $[28,4,20]_2$ кодове и също отделяме само не еквивалентните. Така получаваме класификацията на всички $[28,4,20]_2$ кодове.

При другата използвана техника, увеличаваме едновременно дължината и размерността на кода, т.е. конструираме $[n,k,d]_q$ кодове като разширяваме $[n-i,k-i,d]_q$ или $[n-i-1,k-i,d]_q$ кодове. Следващата лема показва кога могат да се използват кодове от втория тип [145, p. 592].

Лема 3.2.3. *Нека C да е $[n,k,d]_q$ код. Ако съществува кодова дума $\mathbf{c} \in C^{\perp}$ с $wt(\mathbf{c}) = i$, то съществува и $[n-i,k-i+1,d]_q$ код.*

Ако G е пораждаща матрица на $[n-i,k-i,d]_q$ или на $[n-i-1,k-i,d]_q$ код, ние я разширяваме по всички възможни начини съответно до

$$\left(\begin{array}{c|c} * & \mathbf{I}_i \\ \hline \mathbf{G} & \mathbf{0} \end{array}\right) \tag{3.1}$$

или

$$\left(\begin{array}{c|c} * & \mathbf{1}\ \mathbf{I}_i \\ \hline \mathbf{G} & \mathbf{0} \end{array}\right). \tag{3.2}$$

В това представяне $\mathbf{I}_i$ е $i \times i$ единичната матрица, а $\mathbf{1}$ е вектор-стълб само от единици. Ако приемем, че матрицата G е в систематичен вид, можем да фиксираме още k стълба. Така получаваме

$$\left(\begin{array}{c|c|c} * & \mathbf{0} & \mathbf{I}_i \\ \hline \mathbf{G}_1 & \mathbf{I}_k & \mathbf{0} \end{array}\right) \tag{3.3}$$

или

$$\left(\begin{array}{c|c|c} * & \mathbf{0} & \mathbf{1}\ \mathbf{I}_i \\ \hline \mathbf{G}_1 & \mathbf{I}_k & \mathbf{0} \end{array}\right). \tag{3.4}$$

За да определим G_1 можем да приложим рекурсивно същия подход, докато стигнем до дъното на тази йерархия от разширявания, където стои тривиален $[k,k,1]_q$ код.

За класификацията на част от кодовете, обект на това изследване, използвахме и вече готови класификации. Както вече беше споменато, ние се интересуваме от кодове с радиус на покрите по-голям от 1. Следователно минималното им разстояние трябва да е поне 3. Тогава дуалните им кодове трябва да са проективни кодове, т.е. такива, чието дуално минимално разстояние е поне 3. Двоичните

проективни кодове с размерности до 6 са класифицирани от Илия Буюклиев в [22]. Те бяха използвани като основа за класифицирането на двоичните кодове с ко-размерности $n-k$ до 6. За да направим това, измежду класифицираните в [22] проективни кодове, разглеждахме само тези, които имат необходимото дуално разстояние. Например, за да класифицираме всички двоични $[8,2,5]$ кодове, ние разгледахме 14-те $[8,6]$ проективни кода. Само един от тях има дуално разстояние 5 и следователно имаме единствен $[8,2,5]$ код. По същия начин беше използвана класификацията на троичните проективни кодове с размерности до 4 [215], за да бъдат класифицирани троичните кодове с ко-размерности до 4, които биха могли да бъдат квази-съвършени.

След като беше завършена класификацията на кодовете, които биха могли да са квази-съвършени, бяха пресметнати радиусите им на покрите. За целта беше използван Алгоритъм 4. Според него, за да получим по един представител от всеки съседен клас, трябва да генерираме всички думи от вида $(\underbrace{0,\ldots,0}_{k},a)$, $a \in F_q^{n-k}$, като в нашия случай $q=2$ или 3. Тъй като всеки вектор с тегло по-малко или равно на e е единствен лидер на съседен клас, бяха тествани само думите от посочения по-горе вид с тегло по-голямо от e. Стъпките на модифицирания, за целта на това изследване, Алгоритъм 4 са следните.

1) Привежда се в систематичен вид пораждащата матрица на кода.

2) Генерират се и се запомнят в паметта всички кодови думи.

3) Последователно се разглеждат всички вектори с дължина n от вида $(\underbrace{0,\ldots,0}_{k},$ $a)$, $a \in F_q^{n-k}$ за $q=2$ или $q=3$ и с тегло $\geq e$. Пресмята се тяхното разстояние до кода. Ако се намери вектор, който е на разстояние по-голямо от $e+1$ от кода, търсенето се преустановява, защото радиусът на покритие на съответния код е поне $e+2$, а ние търсим кодове с радиус на покритие $e+1$.

При изпълнението на тази модификация на Алгоритъм 4, трябва да се тестват най-много $\sum_{i=e}^{n-k}\binom{n-k}{i+1}(q-1)^{i+1}$ думи и това става в случая, когато намираме квази-съвършен код. В останалите случаи, процедурата прекъсва при намирането на първия вектор с тегло по-голямо от $e+1$, което може да се случи в много начален етап от работата на алгоритъма.

Забележка. Тегловното разпределение $\bar{\alpha}$ на лидерите на съседни класове на квази-съвършените кодове е известно. Всички вектори с тегла по-малки или равни на e са единствени лидери на съседни класове, т.е. $\alpha_i = \binom{n}{i}(q-1)^i$ за $i=0,\ldots,e$. Тогава за α_{e+1} получаваме $\alpha_{e+1} = q^k - \sum_{i=0}^{e}\alpha_i$.

С помощта на представения подход, бяха класифицирани всички двоични и

троични квази-съвършени кодове с размерности съответно до 9 и 6. Бяха получени и частични резултати за двоичните квази-съвършени кодове с размерности до 14 и за троичните квази-съвършени кодове с размерности до 13. В таблиците по-долу са показани промените в Таблици 3.2.1 и 3.2.2, като с $\surd$ са отбелязани параметрите, за които са направени пълни класификации, с N тези, за които е показано несъществуване и с ? - отворените случаи.

Таблица 3.2.3. Двоични квази-съвършени кодове с размерности до 14

k/n	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
2			$\surd$																				
3	$\surd$	$\surd$						N															
4			$\surd$	$\surd$		$\surd$																	
5				$\surd$	$\surd$																		
6					$\surd$	$\surd$			$\surd$														
7						$\surd$	$\surd$	$\surd$		$\surd$													
8							$\surd$	$\surd$	$\surd$		N			N									
9								$\surd$	$\surd$	$\surd$		$\surd$	N		N	N							
10									$\surd$	$\surd$	$\surd$			$\surd$		N	N	?					
11											$\surd$	$\surd$			$\surd$	?	N	N	?				
12												$\surd$	$\surd$	?		$\surd$	$\surd$		$\surd$	$\surd$			
13													$\surd$	$\surd$	?		$\surd$	?	?		?	?	
14														$\surd$	$\surd$	$\surd$		$\surd$	$\surd$	?		?	?

Таблица 3.2.4. Троични квази-съвършени кодове с размерности до 13

k/n	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
2	$\surd$																				
3				N																	
4			$\surd$	$\surd$	N																
5				$\surd$	$\surd$	N	N														
6					$\surd$	$\surd$		$\surd$	N												
7						$\surd$	$\surd$	$\surd$	$\surd$	?											
8							$\surd$	$\surd$	?												
9								$\surd$	$\surd$	?	?	?									
10										$\surd$	?		?	?		?		?			
11											$\surd$	?		?	?		?		?		
12												$\surd$	?		?	?		?		?	?
13													$\surd$	?		?	?		?	?	

Параметрите на квази-съвършените кодове, за които са направени пълни класификации, както и съответните бройки кодове за тези параметри са обобщени в Таблица 3.2.5.

Таблица 3.2.5. Класификационни резултати за двоични и троични квази-съвършени кодове

Двоични квази-съвършенни кодове								
Код	Всички	КС	Код	Всички	КС	Код	Всички	КС
$[5,2,3]$	1	1	$[12,7,3]$	84	55	$[16,11,3]$	143	143
$[6,3,3]$	1	1	$[13,7,4]$*	45	1	$[17,11,4]$*	39	5
$[8,2,5]$*	1	1	$[13,7,3]$	1660	7	$[17,11,3]$	70416	12221
$[7,3,3]$	3	2	$[15,7,5]$	6	4	$[20,11,5]$	13924	565
$[8,4,4]$*	1	1	$[12,8,3]$	2	2	$[17,12,3]$	129	129
$[8,4,3]$	4	4	$[13,8,3]$	109	88	$[18,12,4]$*	33	1
$[9,4,4]$*	4	1	$[14,8,3]$	4419	65	$[21,12,5]$	2373	666
$[9,4,3]$	19	1	$[13,9,3]$	1	1	$[22,12,6]$	128	1
$[11,4,5]$	1	1	$[14,9,3]$	126	113	$[24,12,8]$	1	1
$[9,5,3]$	5	5	$[15,9,3]$	11464	380	$[24,12,7]$	11	11
$[10,5,4]$*	4	1	$[17,9,5]$	1	1	$[25,12,8]$	7	2
$[10,5,3]$	37	12	$[14,10,3]$	1	1	$[18,13,3]$	113	113
$[10,6,3]$	4	4	$[15,10,3]$	142	131	$[19,13,3]$	366064	185208
$[11,6,3]$	58	25	$[16,10,3]$	28900	2296	$[22,13,5]$	128	120
$[14,6,5]$	11	1	$[19,10,5]$	31237	13	$[19,14,3]$	91	91
$[11,7,3]$	3	3	$[16,11,4]$*	1	1	$[20,14,4]$*	24	1
Троични квази-съвършенни кодове								
Код	Всички	КС	Код	Всички	КС	Код	Всички	КС
$[5,2,3]$*	2	2	$[10,6,3]$	195	10	$[12,8,3]$	805	753
$[6,3,3]$	1	1	$[12,6,6]$	1	1	$[14,8,5]$	1	1
$[7,4,3]$	4	4	$[12,6,5]$	36	18	$[12,9,3]$	1	1
$[8,4,4]$*	3	2	$[10,7,3]$	2	2	$[13,9,3]$	1504	1479
$[8,4,3]$	37	5	$[11,7,3]$	339	319	$[14,10,3]$	2695	2659
$[8,5,3]$	3	3	$[12,7,3]$	60910	1	$[15,11,3]$	4304	4304
$[9,5,3]$	87	23	$[13,7,5]$	6	5	$[16,12,3]$	6472	6472
$[9,6,3]$	3	3	$[11,8,3]$	1	1	$[17,13,3]$	8846	8846
$[10,6,4]$*	1	1						

Някои от кодовете, включени в таблицата, не са нови. Те са били конструирани в предишни работи. Ще отбележим, че квази-съвършените кодове с минимално разстояние 3 или 4 и радиус на покритие 2 са свързани с 1-saturating множества в проективните пространства $PG(n-k-1,q)$ по следния начин: точките на 1-saturating n-множества могат да се разгледат като $(n-k)$-мерни стълбове на проверочната матрица на $[n,k]_q 2$ код. Също така, квази-съвършените кодове с минимално разстояние 4 са пълни шапки в $PG(n-k-1,q)$. Конструкции на минимални 1-saturating множества и пълни шапки в проективните пространства $PG(k-1,2)$ са описани в [56], [58]–[60], [83], [129]. Кодовете, които са конструирани в тези работи, са отбелязани със звезда в таблицата. Някои от отбелязаните кодове са получени също в [129], където са дадени рекурсивни конструкции на пълни шапки в $PG(n-k-1,2)$. Съществуването на кодове с параметри $[10,5,3]_2 2$,

$[14,6,5]_2 3$ и $[13,7,3]_2 2$ е показано в [103]. $[17,9,5]_2 3$ кодът е първият представител на безкрайната фамилия от $[2^{2s}+1, 2^{2s}+1-4, 5]_2$, $s \geq 2$ кодове на Zetterberg, за които Додунеков [65] е доказал, че са квази-съвършени. Кодовете $[19,10,5]_2 3$, $[20,11,5]_2 3$, $[23,14,5]_2 3$ и $[24,14,6]_2 3$ са сред квази-съвършените кодове, получени с компютърно търсене от Wagner. Той получава само по един представител за всеки от изброените параметри. Нашата класификация показва, че $[19,10,5]_2 3$ и $[20,11,5]_2 3$ квази-съвършените кодове не са единствени. Има още 12 $[19,10,5]_2$ и 564 $[20,11,5]_2$ квази-съвършени кода. $[24,12,8]_2$ е добре известният разширен код на Golay, който също е квази-съвършен.

За пълнота на класификационните резултати за квази-съвършени кодове, ще отбележим и още някои такива кодове получени в работите на други автори. В [57] е представен единственият $[6,1,5]_2 3$ код. Друг единствен $[5,1,5]_3 3$ квази-съвършен код е даден в [59]. В [83] са намерени квази-съвършените $[16,11,4]_3 2$, $[17,12,4]_3 2$ и $[18,13,4]_3 2$ кодове като пълни шапки в $PG(4,3)$. Квази-съвършеният код $[21,14,4]_2 2$ е намерен като пълна шапка в $PG(6,2)$ в [94]. В [129] е показано, че има 5 не еквивалентни $[21,14,4]_2 2$ кода.

Като приложим Теорема 3.2.1 към кодовете, получени в Таблица 3.2.5, получаваме следните вериги от параметри на квази-съвършени кодове.
$[5,2,3]_2 2 \to [6,3,3]_2 2$;
$[8,4,4]_2 2 \to \ldots \to [14,10,3]_2 2$;
$[9,4,4]_2 2 \to \ldots \to [30,25,3]_2 2$;
$[13,7,4]_2 2 \to \ldots \to \mathbf{[18,12,3]_2 2} \to [19,13,3]_2 2 \to \mathbf{[20,14,3]_2 2} \to \ldots \to \mathbf{[62,56,3]_2 2}$;
$[5,2,3]_3 2 \to [12,3,3]_3 2$;
$[8,4,4]_3 2 \to \ldots \to [17,13,3]_3 2 \to \mathbf{[18,14,3]_3 2} \to \ldots \to \mathbf{[40,36,3]_3 2}$;
$[12,7,3]_3 2 \to \mathbf{[13,8,3]_3 2} \to \ldots \to \mathbf{[121,116,3]_3 2}$.
Кодовете, които не са получени в настоящата класификация, са дадени в болд.

До нашата работа, единствените известни примери на кавзи-съвършени кодове с минимално разстояние по-голямо 5 бяха двоичните кодове с повторение, $[22,12,6]_2 3$ съкратеният код на Golay, $[7,1,7]_3 4$ и $[8,1,8]_2 4$ кодовете, класифицирани в [57]. Ние даваме примери на още такива кодове и по този начин отговаряме на отворения проблем, поставен в статията на Etzion и Mounits [81], където се предлага да се намерят нови или да се докаже несъществуването на квази-съвършени кодове с $d > 5$. Най-интересни са $[24,12,7]_2 4$ и $[25,12,8]_2 4$ кодовете, които са първите примери на квази-съвършени кодове с $R = 4$ извън $[24,12,8]_2 4$ разширения код на Golay, $[7,1,7]_3 4$ и $[8,1,8]_2 4$ кодовете с повторение. По-долу са представени пораждащите матрици на тези кодове. Кодовете са в систематичен вид с пораждащи матрици от вида $G = [I_k|A]$, като единичната матрица I_k е пропусната, за

да не се утежнява записа.

Пораждащи матрици на двоични квази-съвършени кодове с минимално разстояние 7 или 8

А. $[24, 12, 7]_2 4$ квази-съвършени кодове

$$A_1 = \begin{pmatrix} 010010110101 \\ 101000111001 \\ 111010010010 \\ 000110011011 \\ 010101010110 \\ 101100001110 \\ 111111011101 \\ 111100110100 \\ 101101010011 \\ 010101111001 \\ 011000011111 \\ 000011111110 \end{pmatrix} \quad A_2 = \begin{pmatrix} 110001001011 \\ 001011000111 \\ 111010010010 \\ 100101100101 \\ 010101010110 \\ 101100001110 \\ 011100100011 \\ 111100110100 \\ 001110101101 \\ 110110000111 \\ 111011100001 \\ 000011111110 \end{pmatrix} \quad A_3 = \begin{pmatrix} 010101001011 \\ 001100111001 \\ 011110010010 \\ 100010011011 \\ 110110101000 \\ 101100001110 \\ 111111011101 \\ 111011001010 \\ 101010101101 \\ 110110000111 \\ 011000011111 \\ 000011111110 \end{pmatrix}$$

$$A_4 = \begin{pmatrix} 100010111101 \\ 001011010011 \\ 101110001110 \\ 110011101010 \\ 110111010100 \\ 111101111111 \\ 111110100001 \\ 011011001101 \\ 111010110110 \\ 110110011011 \\ 001111111000 \\ 000111100111 \end{pmatrix} \quad A_5 = \begin{pmatrix} 000010111101 \\ 011100110100 \\ 011001101001 \\ 110100001101 \\ 110000110011 \\ 101101111111 \\ 101110100001 \\ 111100101010 \\ 111101010001 \\ 100110011011 \\ 001111111000 \\ 010111100111 \end{pmatrix} \quad A_6 = \begin{pmatrix} 000011011110 \\ 101010101011 \\ 101110010101 \\ 010011110001 \\ 010110101100 \\ 111101111111 \\ 111111000010 \\ 111011001101 \\ 111010110110 \\ 010110011011 \\ 001111111000 \\ 000111100111 \end{pmatrix}$$

$$A_7 = \begin{pmatrix} 110001010101 \\ 001011011001 \\ 111010010010 \\ 000110010111 \\ 110110100100 \\ 001111100010 \\ 111111001111 \\ 111100101010 \\ 001110101101 \\ 110110011001 \\ 111011100001 \\ 000011111110 \end{pmatrix} A_8 = \begin{pmatrix} 000011011011 \\ 001010101101 \\ 001110010110 \\ 100011100110 \\ 100110110001 \\ 101101111111 \\ 101111001000 \\ 101011010101 \\ 101010111010 \\ 100110001111 \\ 001111100011 \\ 010111111100 \end{pmatrix} A_9 = \begin{pmatrix} 010101011101 \\ 011011101100 \\ 001110001111 \\ 100001101011 \\ 110111000110 \\ 111100100001 \\ 101010110010 \\ 101101010111 \\ 111110011100 \\ 110011110101 \\ 011000111011 \\ 000111111010 \end{pmatrix}$$

$$A_{10} = \begin{pmatrix} 110001101110 \\ 111111011111 \\ 101010111100 \\ 010110101101 \\ 100100110011 \\ 001011100111 \\ 011101110100 \\ 111110100010 \\ 101101101001 \\ 010111000110 \\ 011000111011 \\ 000111111010 \end{pmatrix} A_{11} = \begin{pmatrix} 000111100110 \\ 101101010101 \\ 101100101011 \\ 110000110111 \\ 110011011100 \\ 111011100001 \\ 111110010010 \\ 011010101110 \\ 011001011011 \\ 010101101101 \\ 001111111000 \\ 000110011111 \end{pmatrix}$$

В. $[25, 12, 8]_2 4$ квази-съвършени кодове

$$A_1 = \begin{pmatrix} 1101101100100 \\ 1101000111001 \\ 1110100001101 \\ 1110110110000 \\ 1011001110010 \\ 1011010101100 \\ 1000100111110 \\ 0111100101010 \\ 0111101010001 \\ 0100110011011 \\ 0001111111000 \\ 0010111100111 \end{pmatrix} A_2 = \begin{pmatrix} 1101101110000 \\ 1101000100111 \\ 1100011101001 \\ 1100001011110 \\ 1001110010110 \\ 1001101001101 \\ 1000100111011 \\ 0101011010101 \\ 0101010111010 \\ 0100110001111 \\ 0001111100011 \\ 0010111111100 \end{pmatrix}$$

Получените в това изследване резултати показват, че има само по няколко дължини, за които е възможно да съществуват квази-съвършени кодове за всяка

размерност. За някои от параметрите бяха намерени стотици и хиляди квази-съвършени кодове, което показва че това не е толкова рестриктивно свойство за кода. Намерените квази-съвършени кодове с минимално разстояние по-голямо от 5 ни дава основание да считаме, че за по-големи размерности ще съществуват квази-съвършени кодове с по-големи радиуси на покритие. Следователно интересно би било да се отговори на следните въпроси:

• Съществуват ли други квази-съвършени кодове с минимално разстояние по-голямо от 8, с изключение на двоичните кодове с повторение?

• Има ли горна граница за минималното разстояние на квази-съвършен код?

В заключение ще отбележим, че направените изследвания водят до извода, че класификацията на всички възможни параметри на квази-съвършени кодове би била многократно по-тежка задача в сравнение с тази за съвършените кодове.

Резултатите от този раздел са съвместни с Илия Буюклиев, Стефан Додунеков и Veerle Fack и са публикувани в [223]. Представени са на International Workshop on Optimal Codes and Related Topics във Бялата Лагуна, България, през 2007 година и на лекция на семинара на групата CAAGT от унирситета в Гент, Белгия.

3.3 Минимален радиус на покритие на двоични линейни кодове с малки дължини

Интересен изследователски проблем, свързан с радиуса на покритие на код, е определянето на стойностите на функцията $t_q[n,k]$ за фиксирани дължини и размерности. Най-интензивно е изследвана функцията $t_2[n,k]$ и точни стойности или горни и долни граници за нея са получени в [8, 38, 48, 74, 103, 110, 180, 186, 187]. Резултатите от тези работи са обощени в Таблица 7.1 от [47], където са дадени точните стойности или горните и долните граници за $t_2[n,k]$ за кодове с дължини до 64.

В настоящата глава са разгледани първите шест отворени случая от Таблица 7.1 от [47] за $t_2[n,k]$. Подходът, който е използван, е да се конструират кодове с радиус на покритие равен на долната граница за $t_2[n,k]$. Оказа се, че за всички параметри не съществуват кодове с радиуси на покритие равни на долната граница и тъй като долната и горната граници се различават само с единица, бяха определени точните стойности на $t_2[n,k]$ за тези параметри.

Първо ще докажем три леми относно $t_2[n,k]$, които ще бъдат използвани по-късно.

Лема 3.3.1. *Радиусът на покритие на $[n,k]$ кода C с минимално разстояне 1 или 2 е поне $t_2[n-1,k-1]$.*

Доказателство. Проверочната матрица на $[n,k,1]$ код има стълб само от единици, а на $[n,k,2]$ код има еднакви стълбове. От Теорема 1.2.12-в) следва, че $R(C) \geq t_2[n-1,k-1]$. ◇

Лема 3.3.2. *Радиусът на покритие на $[n,k,d]$ кодът C с минимално разстояние на дуалния му код 1 е поне $t_2[n-1,k]+1$.*

Доказателство. Без загуба на общност може да считаме, че векторът $(1,0,\ldots,0)$ принадлежи на дуалния код на кода C. Следователно първата координата на C е нулева и $R(C) \geq t_2[n-1,k]+1$. ◇

Лема 3.3.3. *Радиусът на покритие на $[n,k,d\geq 3]$ код C с минимално разстояние на дуалния му код 2 е поне $t_2[n-2,k]+1$.*

Доказателство. Тъй като минималното разстояние на дуалният код $C^{\perp}$ на кода C е 2, можем да приемем без загуба на общност, че векторът $(1,1,0,\ldots,0)$ принадлежи на $C^{\perp}$. Тогава $c_1 = c_2$ за всяка кодова дума $c = (c_1, c_2, \ldots, c_n)$ от кода C и можем да приемем, че пораждащата му матрица е във вида:

$$G = \left[\begin{array}{cc|c} 1 & 1 & \\ 0 & 0 & \\ \vdots & \vdots & G' \\ 0 & 0 & \end{array}\right].$$

Кодът C има минимално разстояние поне 3 и следователно G' е пораждаща матрица на $[n-2,k]$ код C'. От границата за слепване на кодове $R(C) \geq R(C')+1 \geq t_2[n-2,k]+1$. □

В нашето изследваме искаме да построим базис G на кода C със зададен радиус на покрите $r = R(C)$ и фиксирано минимално разстояние Δ на дуалния му код. За целта, конструираме базиса H на дуалния на C код. Попълваме матрицата H последователно ред след ред като избираме всеки следващ ред така, че кодът, който тя ще поражда да има минимално разстояние Δ. За да ограничим възможните избори за редове, ще използваме още едно условие. Да приемем, че сме избрали първите i реда на H. Да означим с $T_i(m)$ $i = 1,2,\ldots,R$ броят на линейните комбинациите от i стълба на H, чийто префикс е $m \in F_2^i$. Според Теорема 1.2.12-в) трябва да бъде удовлетворено следното условие:

$$T_i(m) \geq 2^{n-k-i}. \tag{3.3.1}$$

Тогава на всяка стъпка, на която избираме нов ред, ще избираме само такива, които удовлетворяват условието (3.3.1).

Нека да пермутираме стълбовете на дуалния код по такъв начин, че единиците в думата с минимално тегло да са в първите Δ позиции, а нулите - в останалите. Тогава можем да считаме, че базисът G на C е в следния вид:

$$G = [\underbrace{G_0}_{\Delta} \mid \underbrace{G'}_{n-\Delta}],$$

където теглата на векторите в G_0 са четни. Ще означим с C_0 кодът породен от G_0, а с C' - кодът породен от G'.

Има два основни случая за кодовете, които изследваме.

Случай 1: $\Delta \in \{3, 4\}$ *и редовете на* G' *са линейно независими.* Ще конструираме базисът G в две стъпки. Първо ще конструираме всички не еквивалентни кодове C' с пораждащи матрици G' и радиус на покритие $R' = R - 1$ като конструираме техните проверочни матрици H'. Тъй като всеки код има проверочна матрица в систематичен вид, можем да фиксираме част от H', т.е.

$$H' = \begin{bmatrix} & 1 & \dots & 0 \\ D' & & \ddots & \\ & 0 & \dots & 1 \end{bmatrix}.$$

След това определяме всички не еквивалентни матрици D' като избираме само редове, които удовлетворяват условието (3.3.1). След като определим G' от H', трябва да попълним оставащите Δ стълба от G, т.е. да определим G_0. Аналогично на построяването на G', ще строим проверочната матрица H на кода C във вида:

$$H = \begin{bmatrix} & 1 \dots 0\ 0 & \dots 0 \\ D' & \ddots & \ddots \\ & 0 \dots 1\ 0 & \dots 0 \\ * \dots * & 0 \dots 0\ 1 & \dots 0 \\ \ddots & \ddots & \ddots \\ \underbrace{* \dots *}_{k} & 0 \dots 0\ 0 & \dots 1 \end{bmatrix} \leftarrow \underbrace{0 \dots 0}_{n-\Delta} \underbrace{1 \dots 1}_{\Delta}$$

На мястото на първия непопълнен ред на H може да бъде поставен този означен със стрелка, защото сумата от стълбовете на G_0 е 0. За останалите ще правим пълно претърсване, ограничено от изпълнението на условие (3.3.1).

Случай 2: $\Delta = 4$ *и има линейно зависими редове в* G', т.е. размерността на матрицата G' е с 1 по-малка от тази на G. Конструираме всички не еквивалентни $[n - \Delta, k - 1]$ кодове с радиус на покритие $R' = R - 1$ по същия начин както и в Случай 1. Матрицата G, в която все още не са попълнени първите 4 стълба, е в

следния вид:

$$\begin{bmatrix} * & \dots & * & & & \\ & \ddots & & & G' & \\ * & \dots & * & 0 & \dots & 0 \end{bmatrix}.$$

Тъй като последният ред не е изцяло нулев, можем да приемем, че имаме 1 в най-дясната позиция на непопълнената част, т.е. в четвърта позиция. Като прибавим последния ред към предишните редове, можем да получим нули в техните позиции номер 4. Тази операция няма да промени матрицата G'. След това пермутираме стълбовете така, че четвъртата единица да отиде на последно място в G. Ако приемем, че G' е в систематичен вид $([G'_x|I])$, получаваме G във вида:

$$\begin{bmatrix} * & \dots & * & & & & 1 & \dots & 0 \\ & \ddots & & & G'_x & & & \ddots & \\ * & \dots & * & 0 & \dots & 0 & 0 & \dots & 1 \end{bmatrix}.$$

Както и в предишния случай, за да консруираме G ние конструираме H във вида:

$$H = \begin{bmatrix} & & 0 & 1 & \dots & 0 \\ & {G'_x}^{tr} & \vdots & & & \\ & & 0 & & \ddots & \\ * & \dots & * & & & \\ & \ddots & & & & \\ * & \dots & * & 0 & \dots & 1 \end{bmatrix} \leftarrow 0 \dots 01 \overbrace{0 \dots 0 \underbrace{1 \dots 1}_{\Delta}}^{n-k}$$

Първите шест отворени случая за $t_2[n,k]$ в Таблица 7.1 от [47] са следните: $t_2[17,6] = 4-5$; $t_2[17,8] = 3-4$; $t_2[18,7] = 4-5$; $t_2[19,7] = 4-5$; $t_2[20,8] = 4-5$ и $t_2[21,7] = 5-6$. Изложената по-горе техника за класификация на двоични линейни кодове със зададен радиус на покритие беше използвана в търсенето на кодове с дължини и размерности като на кодовете от отворените случаи и радиуси на покритие равни на долната граница. С помощта на Леми 3.3.1, 3.3.2 и 3.3.3, първо показваме, че минималните разстояния на изследваните кодове и на техните дуални трябва да са по-големи от 2. След това използваме програма, която реализира техниката за класификация, за да намерим всички не еквивалентни кодове с търсените параметри.

Сега ще опишем накратко как това беше направено за всеки от изследваните шест случая. Максималните възможни стойности на минималните разстояния за изследваните кодове бяха взети от [29].

1. [17,6] кодове.

Твърдение 3.3.4. *Ако съществува* $[17,6]$ *код с радиус на покритие 4, то мини-*

малното разстояние на дуалния му код е $\Delta \in \{3, 4\}$.

Доказателство. От Лема 3.3.1 следва, че $t_2[17, 6, 1] \geq t_2[16, 5] = 5$ и $t_2[17, 6, 2] \geq t_2[16, 5] = 5$, от Лема 3.3.2, че за $[17, 6]$ кодовете с $\Delta = 1$, $t_2[17, 6] \geq t_2[16, 6] + 1 = 5$ и от Лема 3.3.3, че за $[17, 6]$ кодовете с $d \geq 3$ и $\Delta = 2$, $t_2[17, 6] \geq t_2[15, 6] + 1 = 5$. За да завършим доказателството, ще отбележим, че $\Delta \leq 4$ за всички $[17, 6]$ кодове. □

Случай 1: $\Delta \in \{3, 4\}$ и редовете на G' са линейно независими. Компютърното търсене показва, че има 15 не еквивалентни кода с $\Delta = 3$ и 87 не еквивалентни кода с $\Delta = 4$, но радиусите на покритие на всички тези кодове са по-големи от 4.

Случай 2: $\Delta = 4$ и размерността на матрицата G' е с 1 по-малка от тази на G, т.е. G' е 5×13 матрица. От границата за слепване на кодове $R(C) \geq R(C_0) + R(C') = 1 + R(C')$. Тъй като $t_2[13, 5] = 4$, имаме $R(C) \geq 5$. Следователно не можем да конструираме код с радиус на покритие 4 в този случай.

Следователно не съществуват $[17, 6]$ кодове с радиус на покритие 4 и $t_2[17, 6] = 5$.

2. [17,8] кодове.

Твърдение 3.3.5. *Ако съществува* $[17, 8]$ *код с радиус на покритие 3, то минималното разстояние на дуалния му код е* $\Delta \in \{3, 4, 5\}$.

Ще пропуснем доказателствата на това и на следващите 4 твърдения, тъй като те са подобни на доказателството на Твърдение 3.3.4.

За $\Delta \in \{3, 4\}$ бяха тествани случаи 1 и 2, но радиусите на покритие на всички 1004 кода, които бяха конструирани са по-големи от 4.

За $\Delta = 5$, бяха класифицирани всички $[17, 9, 5]$ кода. Оказа се, че има единствен $[17, 9, 5]$ код със следната пораждаща матрица

$$\begin{bmatrix} 11111000000000000 \\ 11000110100000000 \\ 00110001101000000 \\ 10100001010100000 \\ 00110100010010000 \\ 01100010010001000 \\ 00010011010000100 \\ 10000111010000010 \\ 10010011100000001 \end{bmatrix}.$$

Радиусът на покритие на дуалният му код е 5.

Следователно не съществуват $[17, 8]$ кодове с радиус на покритие 3 и $t_2[17, 8] = 4$.

3. [18,7] кодове.

Твърдение 3.3.6. *Ако съществува* $[18, 7]$ *код с радиус на покритие 4, то минималното разстояние на дуалния му код е* $\Delta \in \{3, 4\}$.

Бяха тествани случаи 1 и 2, но не бяха намерени кодове с радиуси на покритие 4 и следователно $t_2[18, 7] = 5$.

От този резултат непосредствено следва, че $t_2[19, 7] \geq 5$, т.е. $t_2[19, 7] = 5$.

4. [20,8] кодове.

Твърдение 3.3.7. *Ако съществува* $[20, 8]$ *код с радиус на покритие 4, то минималното разстояние на дуалния му код е* $\Delta \in \{3, 4\}$.

Случай 1: За $\Delta = 3$ от границата за слепване на кодове имаме $R(C) \geq 1 + t_2[17, 8] = 5$. Затова търсим само кодове с $\Delta = 4$. Радиусите на покритие на всички конструирани кодове са по-големи от 4.

Случай 2: Тъй като $t_2[16, 7] = 4$, получаваме $R(C) \geq 1 + t_2[16, 7] = 5$.

От случаи 1 и 2 следва, че $t_2[20, 8] = 5$.

5. [21,7] кодове.

Твърдение 3.3.8. *Ако съществува* $[21, 7]$ *код с радиус на покритие 5, то минималното разстояние на дуалния му код е* $\Delta \in \{3, 4\}$.

Случай 1: Търсим смо кодове с $\Delta = 4$, защото за кодовете с $\Delta = 3$ имаме $R(C) \geq 1 + t_2[18, 7] = 6$. Радиусите на покритие на всички конструирани кодове са по-големи от 5.

Случай 2: От $t_2[17, 6] = 5$ следва, че $R(C) \geq 1 + 5 = 6$.

Следователно $t_2[21, 7] = 6$.

Друга алтернатива, когато търсим кодове с най-добри покриващи свойства, е при зададени ко-размерност m и радиус на покритие R да определим функцията $l(m, R)$, която е най-малката възможна дължина на код със зададени m и R. Ще представим получените в този раздел резултати и в термините на тази функция.

Следствие 3.3.9. $l(9, 3) = 18$, $l(11, 4) = 19$, $21 \leq l(12, 4) \leq 23$ *и* $22 \leq l(14, 5) \leq 24$.

Доказателство. От Таблица 7.2 ([47], стр. 202-208) е известно, че $17 \leq l(9, 3) \leq 18$, $17 \leq l(11, 4) \leq 19$, $19 \leq l(12, 4) \leq 23$ и $21 \leq l(14, 5) \leq 24$. Резултата се получава непосредствено от Твърдения 3.3.4, 3.3.5, 3.3.6 и 3.3.7. □

В края на този раздел ще покажем единствеността на един код с минимален радиус на покритие. В [103] е показано, че $t_2[14, 6] = 3$ като е конструиран код

с тези параметри. Изказано е също и предположението, че този код е единствен. По-късно в [69] е показано, че съществува единствен $[14, 6, 5]$ код и радиусът му на покритие е 3. Минималното разстояние на всеки $[14, 6]$ код е по-малко или равно на 5, а на неговия дуален е по-малко или равно на 4. Следователно за да докажем хипотезата от [103], трябва да покажем, че радиусите на покритие на $[14, 6, d \leq 4]$ кодовете са по-големи от 3. От Леми 3.3.1, 3.3.2 и 3.3.3 следва следното твърдение.

Твърдение 3.3.10. *Ако съществува* $[14, 6, d \leq 4]$ *код с радиус на покритие 3, то минималното разстояние на дуалния му код е* $\Delta \in \{3, 4\}$.

Прилагаме същия подход както и при предишните кодове и тестваме случаи 1 и 2. Компютърното търсене показа, че всички такива кодове имат радиуси на покритие поне 4.

Резултатите от този раздел са получени съвместно с Веселин Ваврек и са публикувани в [214]. Представени са на EuroWorkshop on Optimal Codes and Related Topics в Слънчев бряг, България през 2001 г.

3.4 Минимален радиус на покритие на двоичните линейни кодове с размерност 6

В този раздел определяме минималните радиуси на покритие на всички двоични линейни кодове с размерност 6. Стойностите на функцията$t_2[n, k]$ за кодове с размерности до 5 са определени в [48] и [103]. В [103] са определени и горни граници за $t_2[n, k]$ за кодове с размерности 6 и 7. По-точно, за размерност 6 е доказано, че

$$t_2[n, 6] \leq \left\lfloor \frac{n-8}{2} \right\rfloor, \text{ за } n \geq 18 \tag{3.4.1}$$

За да определим стойностите на функцията $t_2[n, 6]$ за всяко n, използвахме класификацията на всички двоични проективни кодове с размерност 6 направена от Илия Буюклиев в [22]. След това, трябваше да проверим дали тези кодове имат определен радиус на покритие. От решаващо значение беше да имаме ефективен алгоритъм, който бързо да отхвърля кодовете, чиито радиуси на покритие са по-големи от предварително определения. Много подходяща за целта се оказа предложената от Илия Буюклиев модификация на Алгоритъм 4 за пресмятане на радиуса на покритие на линеен код. Алгоритъмът е евристичен и бързо определя долна граница на радиуса на покритие на линеен код.

Идеята на алгоритъма е подобна на използваната в раздел 3.2. Използваме Метод 2, за пресмятането на радиуса на покритие на кода. Пораждащата матрица

на кода е в систематичен вид и генерираме по един представител от всеки съседен клас като разглеждаме векторите от вида $(\underbrace{0,\ldots,0}_{k},a)$, $a \in F_2^{n-k}$. В раздел 3.2 систематично генерирахме всички такива вектори с тегла по-големи от $\left\lfloor \frac{d-1}{2} \right\rfloor$, тъй като за тях е известно, че са единствени лидери на съседни класове. В този раздел използваме евристична техника, с цел да намерим колкото е възможно по-бързо лидер на съседен клас с тегло по-голямо от някаква предварително зададена стойност R за радиуса на покритие на кода. Намирането на такъв лидер означава, че радиусът на покрите на кода е по-голям от R. Алгоритъмът започва с тестването на вектор c от съседния клас $K_c = \{c + C\}$. Ако теглото на лидера на този съседен клас е по-голямо от R, работата на алгоритъма приключва. В противен случай, формираме множествата от съседи на c - $N(c)$, които се различават от c в една позиция и използваме оценъчната функция $f(c)$, за да намерим текущото най-добро решение. Функцията $f(c) = wt(K_c)2^k - A(K_c)$ зависи от теглото на съседния клас $wt(K_c)$ и броя $A(K_c)$ на векторите с минимално тегло от K_c. Целта ни е да максимизираме $f(c)$, т.е. да минимизираме броя на векторите с минимално тегло в съседния клас. Търсенето продължава в множеството от съседи $N(c)$ докато намерим съседен клас с тегло по-голямо от R. Ако претърсването на това множество не доведе до очаквания резултат, добавяме шум към вектора c като инвертираме произволно избрани негови позиции и повтаряме процедурата. Схематично алгоритъмът е показан по-долу.

Алгоритъм 6. LowerBoundCoveringRadius(R_{min})

c, c': vector; br_0,br: integer;
{ $br_0 := 0$;
  while $br_0 < const_0$
  { $br_0 := br_0 + 1$; $br := 0$;
    Select a random vector $c \in F_2^n \setminus \{C\}$;
    while $br < const$
    { $br := br + 1$;
      while exists $c' \in N(c)$ such that $f(c') > f(c)$ do $c := c'$;
       if $wt(K_c) > R_{min}$ break; output $R > R_{min}$
      Add some noise to c;
    }
  }
}

Тъй като в настоящото изследване беше важно да покажем несъществуването на кодове с определен радиус на покритие, Алгоритъм 6 се оказа много ефективен. Той беше използван, за да се покаже несъществуването на кодове с дължини $22 \leq n \leq 54$, n четно и фиксиран радиус на покритие R. Бяха тествани 236779414 такива кода и предложения алгоритъм се оказа изключително ефективен, т.е. много бързо беше възможно да се намери лидер на съседен клас с тегло по-голямо от R.

Като основа за изследването на минималния радиус на покрите на двоичните кодове с размерност 6 използваме Таблица 7.1 от [47] като в нея отразяваме и новия резултат за $t_2[17, 6]$, получен в предишния раздел. Известните точни стойности и горни и долни граници за $t_2[n, 6]$ са обобщени в следващата таблица.

Таблица 3.4.1. Граници за $t_2[n, 6]$ за $n \leq 64$.

n	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
$\mathbf{t_2[n, 6]}$	1	1	1	2	2	3	3	3	4
n	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**
$\mathbf{t_2[n, 6]}$	4	5	5	5	6	6	6-7	7	7-8
n	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**
$\mathbf{t_2[n, 6]}$	7-8	8-9	8-9	9-10	9-10	9-11	10-11	10-12	11-12
n	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**
$\mathbf{t_2[n, 6]}$	11-13	11-13	12-14	12-14	13-15	13-15	14-16	14-16	14-17
n	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**
$\mathbf{t_2[n, 6]}$	15-17	15-18	16-18	16-19	17-19	17-20	17-20	18-21	18-21
n	**52**	**53**	**54**	**55**	**56**	**57**	**58**	**59**	**60**
$\mathbf{t_2[n, 6]}$	19-22	19-22	20-23	20-23	20-24	21-24	21-25	22-25	22-26
n	**61**	**62**	**63**	**64**					
$\mathbf{t_2[n, 6]}$	23-26	23-27	23-27	24-28					

Сега ще докажем, че за двоични кодове с размерност 6 и дължина по-голяма от 17, границата (3.4.1) е точна. По този начин определяме стойностите на функцията $t_2[n, 6]$ за всяко n. Подходът, който използваме се основава на определянето на радиусите на покритие на двоичните проективни кодове с размерност 6.

Теорема 3.4.1.

$$t_2[n, 6] = \left\lfloor \frac{n-8}{2} \right\rfloor, \ \textit{за } n \geq 18.$$

Доказателство. За кодовете с дължини 18 - 21, стойностите на $t_2[n, 6]$ са известни и те изпълняват условията на теоремата. За останалите кодове с дължини до 64, горните граници от Таблица 3.4.1 съвпадат със стойностите, дадени в теоремата. Остава да докажем, че тези горни граници са точни.

Нека да разгледаме първият отворен случай - $[22, 6]$ кодовете. Ако $[22, 6]$ кодът C има повтарящи се стълбове, то можем да го разглеждаме като слепване на два кода: $[20, 6]$ код с минимален радиус на покритие и $[2, 1]$ кодът с повторение с радиус на покритие 1. Следователно $R(C) \geq t_2[20, 6]+1 = 7$ и ако съществува $[22, 6]$ код с радиус на покритие 6, то той трябва да е проективен. От класификацията на двоичните проективни кодове с размерност 6 [22] е известно, че има 2852541 такива кода. С Алгоритъм 6 показваме, че всички тези кодове имат радиуси на покритие по-големи от 6. Следователно $t_2[22, 6] = 7$.

Нека сега C да е $[24, 6]$ код. Ако има $[24, 6]$ кодове с радиус на покрите 7, то те трябва да са проективни и с Алгоритъм 6 показваме, че всичките 8239576 такива кодове имат радиус на покритие по-голям от 7. Следователно $t_2[24, 6] = 8$. Тъй като $t_2[25, 6] \geq t_2[24, 6]$, получаваме и че $t_2[25, 6] = 8$.

Повтаряме тази процедура докато определим, че $t_2[54, 6] = t_2[55, 6] = 23$. От Таблица 3.4.1 се вижда, че за всеки две съседни дължини, границите за $t_2[n, 6]$ са еднакви. Тогава е достатъчно да разгледаме само първия код (този с четна дължина), а резултата за втория следва непосредствено от него.

Нека C да е $[56, 6]$ код. Ако той има повторени стълбове, то $R(C) \geq t_2[54, 6] + 1 = 23 + 1 = 24$. В противен случай, C може да се разглежда като скъсен с 7 позиции $[63, 6]$ Симпексен код, чийто радиус на покрите е 31. Тогава от границата за скъсяване на кодове следва, че $R(C) \geq 31 - 7 = 24$. Следователно $t_2[56, 6] = 24$ и $t_2[57, 6] = 24$. По аналогичен начин получаваме, че $t_2[58, 6] = t_2[59, 6] = 25$, $t_2[60, 6] = t_2[61, 6] = 26$ и $t_2[62, 6] = t_2[63, 6] = 27$.

За $n \geq 64$, всеки $[n, 6]$ код трябва да има повторени стълбове и $t_2[n, 6] \geq t_2[n-2, 6]+1$, от където получаваме $t_2[n, 6] \geq \lfloor (n-8)/2 \rfloor$ за всяко $n \geq 18$. Следователно границата е точна. □

От гледна точка на приложенията е много важно не само да знаем стойността на минималния радиус на покритие на код с фиксирани дължина и размерност, но и да имаме конструкции на такива кодове. Сега ще представим класификации и конструкции на двоични кодове с размерности до 6 и радиуси на покритие $R = t_2[n, k]$. При конструирането на кодовете важна роля играят понятията *нормализирани* линейни кодове и конструкцията *смесена директна сума* (amalgamated direct sum - ADS) за нормализирани кодове, въведена в [103].

Нека C да е двоичен код с дължина n, размерност k и радиус на покритие R, т.е. $[n, k]R$ код. Ако i е една от n-те му координати, нека да означим с $C_0^{(i)}$ множеството от кодови думи, при които i-тата координата е 0 и с $C_1^{(i)}$ множеството от кодови думи, при които е 1. Приемаме, че $C_1^{(i)}$ не е празно и тогава $C_0^{(i)}$ и $C_1^{(i)}$

съдържат по 2^{k-1} кодови думи. За произволна двоична n-орка, нека

$$f_0(x) = dist(x, C_0^{(i)}) = \min_{c \in C_0^{(i)}} dist(x, c),$$

$$f_1(x) = dist(x, C_1^{(i)}).$$

Тогава

$$N^{(i)} = \max_x \{f_0(x) + f_1(x)\}$$

се нарича *норма* на C по отношение на i-тата координата и

$$N = \min_i N^{(i)}$$

е нормата на C. Координатите i, за които $N = N^{(i)}$ се наричат *приемливи*. Накрая, кодът е *нормализиран* ако нормата му удовлетворява

$$N \leq 2R + 1.$$

Теорема 3.4.2. *[47] Да приемем, че A е нормализиран двоичен $[n_A, k_A]R_A$ код, чиято последна координата е приемлива и B е нормализиран двоичен $[n_B, k_B]R_B$ код, чиято първа координата е приемлива. Тогава тяхната смесена директна сума*

$$A \dot{\oplus} B = \{(a, 0, b) | (a, 0) \in A, (0, b) \in B\} \cup \{(a, 1, b) | (a, 1) \in A, (1, b) \in B\}$$

е $[n_A + n_B - 1, k_A + k_B - 1]R$ код с $R \leq R_A + R_B$. По-общо, ако нормата на A по отношение на последната координата е N_A и нормата на B по отношение на първата координата е N_B, то кодът $A \dot{\oplus} B$ има норма $N_A + N_B - 1$ и следователно радиус на покритие най-много $\frac{1}{2}(N_A + N_B - 1)$. В частност, ако радиусът на покритие на $A \dot{\oplus} B$ е равен на $R_A + R_B$, то $A \dot{\oplus} B$ е нормализиран и припокритата координата е приемлива.

Следният резултат от [103] е много полезен при конструирането на кодове с малки радиуси на покрите.

Теорема 3.4.3. *Ако C е $[n, k]R$ нормализиран код, то съществуват $[n+2i, k]R+i$ нормализирани кодове за всяко $i \geq 0$.*

Известно е, че всички двоични $[n, k]$ кодове с дължини до 15 са нормализирани. Ние класифицираме всички такива кодове като използваме програмата Q-EXTENSION [23] и след това пресмятаме радиусите им покритие. В Таблица 3.4.2 са дадени бройките не еквивалентни кодове с $R = t_2[n, k]$. Случаят $k = 1$ е тривиален и е включен в таблицата само за пълнота. Пораждащите матрици на класифицираните кодове могат да бъдат намерени на http //www.moi.math.bas.bg/∼tsonka.

Таблица 3.4.2. Двоични кодове с $R = t_2[n,k]$ за $n \leq 15$ и $k \leq 6$.

Код	Всички	Код	Всички	Код	Всички	Код	Всички	Код	Всички	Код	Всички
$[3,1]1$	1	$[3,2]1$	2								
$[4,1]2$	1	$[4,2]1$	2	$[4,3]1$	3						
$[5,1]2$	1	$[5,2]2$	4	$[5,3]1$	4	$[5,4]1$	4				
$[6,1]3$	1	$[6,2]2$	4	$[6,3]2$	11	$[6,4]1$	7	$[6,5]1$	5		
$[7,1]3$	1	$[7,2]3$	7	$[7,3]2$	12	$[7,4]1$	1	$[7,5]1$	11	$[7,6]1$	6
$[8,1]4$	1	$[8,2]3$	6	$[8,3]3$	31	$[8,4]2$	37	$[8,5]1$	2	$[8,6]1$	16
$[9,1]4$	1	$[9,2]4$	11	$[9,3]3$	32	$[9,4]2$	2	$[9,5]2$	102	$[9,6]1$	4
$[10,1]5$	1	$[10,2]4$	9	$[10,3]4$	76	$[10,4]3$	164	$[10,5]2$	19	$[10,6]2$	255
$[11,1]5$	1	$[11,2]5$	15	$[11,3]4$	72	$[11,4]3$	5	$[11,5]3$	841	$[11,6]2$	100
$[12,1]6$	1	$[12,2]5$	12	$[12,3]5$	162	$[12,4]4$	653	$[12,5]3$	129	$[12,6]3$	4126
$[13,1]6$	1	$[13,2]6$	20	$[13,3]5$	149	$[13,4]4$	11	$[13,5]4$	6527	$[13,6]3$	2101
$[14,1]7$	1	$[14,2]6$	16	$[14,3]6$	316	$[14,4]5$	2334	$[14,5]4$	889	$[14,6]3$	1
$[15,1]7$	1	$[15,2]7$	26	$[15,3]6$	286	$[15,4]5$	25	$[15,5]5$	47604	$[15,6]4$	47552

Като направим смесена директна сума на всеки от класифицираните кодове с размерности до 5 и на $[2s+1,1]s$ кода с повторение с необходимата дължина, според Теорема 3.4.3, можем да получим код с произволна дължина n по-голяма от 15, размерност k до 5 и радиус на покритие равен на $t_2[n,k]$.

В [103] са конструирани два нормализирани кода. Показано е, че всички координати на тези кодове са приемливи. Това са $[14,6,5]3$ и $[19,6,7]5$ кодовете. Единствеността на $[14,6,5]3$ кода беше показана в предишния раздел, а също така беше потвърдено и от направената по-горе класификация. $[19,6]5$ кодът трябва да е проективен ($t_2[19,5] \geq t_2[17,6]+1 = 6$) и като използваме класификацията от [22], определихме радиусите на покритие на всички 366089 проективни $[19,6]$ кодове. Оказа се, че има само един код с радиус на покритие 5 и следователно и $[19,6,7]5$ кодът е единствен. Тогава започвайки от $[14,6,5]3$ или $[19,6,7]5$ кодовете и прилагайки смесена директна сума с $[2s+1,1]s$ кода с повторение със съответната дължина, можем да получим всеки $[n,6]R = t_2[n,6]$ код с дължина по-голяма от 17. За да получим $[17,6]5$ код, може да приложим смесена директна сума между един от класифицираните $[15,6]4$ нормализирани кодове и $[3,1]1$ кода с повторение.

Нормализираните кодове, класифицирани в този раздел, могат също да бъдат използвани за конструиране на кодове с минимален радиус на покритие и за размерности по-големи от 6. Например известно е, че $t_2[17,8] = t_2[18,8] = 4$. Ако разгледаме смесена директна сума на $[8,4]2$ и $[10,5]2$ нормализирани кодове, според Теорема 3.4.2, получаваме $[17,8]4$ нормализиран код. По същия начин, смесената директна сума на $[9,4]2$ и $[10,5]2$ нормализирани кодове дава $[18,8]4$ нормализиран код.

Нека сега да разгледаме смесена директна сума на $[7,4]1$ и $[14,6]3$ нормализирани кодове. Резултатът е $[20,9]4$ нормализиран код с минимален радиус на

покритие. $[25, 9]6$ нормализиран код с минимален радиус на покритие може да бъде конструиран чрез смесена директна сума на $[7, 4]1$ код и на $[19, 6]5$ кода, конструиран в [103], за който доказахме по-горе, че е единствен.

Резултатите в този раздел са съвместни с Илия Буюклиев и са публикувани в [226]. Представени са на International Workshop on Algebraic and Combinatorial Coding Theory в Пампорово, България през 2008 и на семинар на групата CAAGT в университета в Гент, Белгия през същата година.

3.5 Минимален радиус на покритие на троичните линейни кодове с размерност 4

В този раздел са изследвани троичните линейни кодове с размерност 4. Направена е класификация на всички троични линейни кодове с размерност 4 и на базата на тази класификация са определени някои неизвестни стойности на функцията $t_3[n, 4]$.

Класификация на троичните проективни кодове с размерност 4 по отношение на минималното им разстояние, като са използвани проективни геометрии, е направена в [82]. В тази глава използваме подхода на McKay [153] за отхвърляне на изоморфни решения, за да направим същата класификация.

Нека да означим с M $q^m \times \dfrac{q^m - 1}{q - 1}$ матрицата от кодови думи на симплексния код. За да конструираме проективен код със зададена размерност, използваме факта, че той е скъсена версия на съответния симплексен код. С други думи, кодовите думи на всеки проективен код C се получават като се изберат определен брой стълбове от матрицата M. Ще казваме, че кодът C е дефиниран от тези стълбове на M.

Основната идея е да се конструират рекурсивно нови кодове, които в съответствие с терминологията на McKay ще наричаме *кодове-наследници*, от кодове, които ще наричаме *кодове-родители*. В нашия случай, ако кодът-родител е дефиниран от n стълба на матрицата M, то кодът-наследник ще бъде дефиниран от тези стълбове плюс още един нов, различен от тях стълб от M. Като кодове-наследници ще бъдат приемани само тези от така конструираните кодове, които отговарят на родителския тест и на теста за изоморфизъм.

При родителския тест използваме въведеното в [153] канонично подреждане на координатите на кодовете. Тогава родителският тест може да бъде преминат само от тези кодове-наследници, чийто последен добавен стълб е първи в канонич-

ното подреждане или е в една и съща орбита с първия в каноничното подреждане. От кодовете-наследници, които са преминали родителския тест, вземаме само по един представител от всеки клас на еквивалентност.

Алгоритъмът за конструиране на проективни кодове има следните стъпки.

- Започваме от празното множеството и рекурсивно правим следното.
- За даден код от дървото на търсене конструираме всички възможни кодове-наследници като добавяме към кода-родител един нов стълб, различен от тези които са вече в кода-родител.
- За всеки такъв код-наследник правим родителския тест и за кодовете-наследници, които успешно са го преминали, правим теста за изоморфизъм измежду кодовете, които са получени от един и същ родител.

За пресмятането на каноничното подреждане на стълбовете и реализирането на теста за изоморфизъм, използваме алгоритъма от [24]. Предимството на този алгоритъм за нашето изследване е, че трябва да правим теста за изоморфизъм само между кодовете-наследници на един код-родител.

С този алгоритъм и със съответния софтуер, разработен от Илия Буюклиев, бяха класифицирани всички троични проективин кодове с размерност 4. Те са представени по-долу като за всяка дължина е посочен общия брой на не еквивалентните кодове. Като степен е дадени броя на кодовете, имащи съответното минимално разстояние, записано като основа на израза.

По-късно този алгоритъм беше разширен, така че да бъдат класифицирани всички троични кодове с размерност 4 и дължини до $n = 20$. За целта разширяваме с необходимия брой повторени координати съответните проективни $[n - s, 4]$ кодове. След това използваме алгоритъма от [24], за да отделим не еквивалентните кодове. Програмната реализация на тази част от изследването е направена от Илия Буюклиев.

k= 4

$n = 4$	1	1^{1}
$n = 5$	3	1^{2}, 2^{1}
$n = 6$	8	1^{3}, 2^{5}
$n = 7$	19	1^{4}, 2^{11}, 3^{4}
$n = 8$	44	1^{4}, 2^{16}, 3^{21}, 4^{3}
$n = 9$	91	1^{3}, 2^{16}, 3^{45}, 4^{26}, 5^{1}
$n = 10$	199	1^{3}, 2^{13}, 3^{55}, 4^{112}, 5^{15}, 6^{1}
$n = 11$	401	1^{2}, 2^{10}, 3^{46}, 4^{174}, 5^{165}, 6^{4}
$n = 12$	806	1^{1}, 2^{6}, 3^{33}, 4^{154}, 5^{448}, 6^{164}
$n = 13$	1504	1^{1}, 2^{3}, 3^{19}, 4^{102}, 5^{478}, 6^{843}, 7^{58}
$n = 14$	2659	1^{1}, 2^{2}, 3^{9}, 4^{53}, 5^{314}, 6^{1318}, 7^{950}, 8^{12}
$n = 15$	4304	2^{1}, 3^{5}, 4^{22}, 5^{151}, 6^{941}, 7^{2559}, 8^{623}, 9^{2}
$n = 16$	6472	3^{2}, 4^{9}, 5^{57}, 6^{439}, 7^{2310}, 8^{3478}, 9^{177}
$n = 17$	8846	4^{3}, 5^{19}, 6^{153}, 7^{1099}, 8^{4617}, 9^{2937}, 10^{18}
$n = 18$	11127	5^{5}, 6^{45}, 7^{356}, 8^{2454}, 9^{6799}, 10^{1466}, 11^{2}
$n = 19$	12723	6^{10}, 7^{89}, 8^{782}, 9^{4582}, 10^{6935}, 11^{324}, 12^{1}
$n = 20$	13358	7^{16}, 8^{178}, 9^{1514}, 10^{6893}, 11^{4722}, 12^{35}
$n = 21$	12723	8^{28}, 9^{328}, 10^{2526}, 11^{7860}, 12^{1981}
$n = 22$	11127	9^{47}, 10^{528}, 11^{3587}, 12^{6530}, 13^{435}
$n = 23$	8846	10^{68}, 11^{763}, 12^{4220}, 13^{3747}, 14^{48}
$n = 24$	6472	11^{91}, 12^{977}, 13^{3913}, 14^{1484}, 15^{7}
$n = 25$	4304	12^{114}, 13^{1074}, 14^{2764}, 15^{351}, 16^{1}
$n = 26$	2659	13^{127}, 14^{1014}, 15^{1462}, 16^{55}, 17^{1}
$n = 27$	1505	14^{127}, 15^{801}, 16^{569}, 17^{7}, 18^{1}
$n = 28$	807	15^{113}, 16^{520}, 17^{171}, 18^{3}
$n = 29$	402	16^{90}, 17^{274}, 18^{38}
$n = 30$	201	17^{66}, 18^{127}, 19^{8}
$n = 31$	94	18^{45}, 19^{47}, 20^{2}
$n = 32$	47	19^{26}, 20^{20}, 21^{1}
$n = 33$	23	20^{15}, 21^{8}
$n = 34$	12	21^{9}, 22^{3}
$n = 35$	6	22^{5}, 23^{1}
$n = 36$	4	23^{3}, 24^{1}
$n = 37$	2	24^{2}
$n = 38$	1	25^{1}
$n = 39$	1	26^{1}
$n = 40$	1	27^{1}

Последната стъпка от това изследване е определянето на някои от неизвестните стойности на функцията $t_3[n,4]$. В [11] е представена таблица с граници и точни стойности на функцията $t_3[n,k]$ за кодове с дължини до 27. По-късно, в [164] са определени стойностите на $t_3[10,4]$ и $t_3[12,4]$. Ние разширяваме тази таблица за кодове с дължини до 40 (виж Таблица 3.5.1 от приложението). Долната граница е границата на сферичната опаковка (1.2), а за горна граница е използвана най-добрата от границите (1.12) – (1.15). Някои точни стойности са получени в работите на различни автори и са означени така: с c - получените в [10], с n - в [211], с D - в [55], с O_1 - в [164] и с O_2 - в [164]. След това е приложена границата (1.12) и са уточнени още следните 9 стойности на $t_3[n,k]$:
1 - $t_3[16,8] \leq t_3[8,4] + t[8,4] = 4 \Rightarrow t_3[16,8] = 4$,
2 - $t[18,8] \leq t[14,7] + t[4,1] = 5 \Rightarrow t_3[18,8] = 5$,
3 - $t_3[24,11] \leq t_3[20,10] + t_3[4,1] = 6 \Rightarrow t_3[24,11] = 6$,
4 - $t_3[28,14] \leq t_3[24,12] + t_3[4,2] = 6 \Rightarrow t_3[28,14] = 6$,
5 - $t_3[31,16] \leq t_3[20,10] + t[11,6] = 6 \Rightarrow t_3[31,16] = 6$,
6 - $t_3[33,20] \leq t_3[20,10] + t_3[13,10] = 5 \Rightarrow t_3[33,20] = 5$,
7 - $t_3[34,17] \leq t_3[20,10] + t_3[14,7] = 7 \Rightarrow t_3[34,17] = 7$,
8 - $t_3[36,18] \leq t_3[34,17] + t_3[2,1] = 7 \Rightarrow t_3[36,18] = 7$,
9 - $t_3[40,20] \leq t_3[20,10] + t_3[20,10] = 8 \Rightarrow t_3[40,20] = 8$.

При определянето в [11] на радиусите на покритие на троичните кодове с размерности 2 и 3 се оказа, че за всички дължини до $\dfrac{3^m-1}{3-1}$ за $m = 2,3$ има проективен код с минимален радиус на покритие. Затова при определянето на стойностите на $t_3[n,k]$, ние първо тествахме проективните кодове със съответната дължина. Ако не се намереха кодове с радиус на покритие равен на долната граница за $t_3[n,k]$, трябваше да се проверят и кодовете с повторени координати.

Нека C да е код с n_1 еквивалентни координати. Тогава можем да разглеждаме C като слепване на $[n_1,k_1]$ код C_1 и на $[n_2,k_2]$ код C_2, като $k_1 \leq k_2$ и пораждащите им матрици са съответно G_1 и G_2. Формираме пораждащата матрица G на кода C така $G = [G_1'|G_2]$, като G_1' е G_1 с добавени $k_2 - k_1$ нулеви реда. Тогава C е $[n_1+n_2,k_2]$ код, чийто радиус на покритие удовлетворява условието

$$R(C) \geq R(C_1) + R(C_2).$$

Ние използваме тази граница по следния начин. Нека $[n,4]$ кодът C да има повторени координати и нека те да са на първите s позиции в пораждащата му матрица. Тогава можем да разглеждаме C като слепване на кодовете C_1, който е $[s,1]$ код и на C_2, който е $[n-s,4]$ проективен код. Радиусът на покритие на C_1 е

$$R(C_1) = \left\lfloor \frac{2s}{3} \right\rfloor,$$

а за C_2 избираме код с минималния възможен радиус на покритие

$$R(C_2) = t_3[n-s, 4].$$

Така получаваме долна граница за радиуса на покритие на C

$$R(C) \geq \left\lfloor \frac{2s}{3} \right\rfloor + t_3[n-s, 4]. \tag{3.1}$$

За да определим радиусите на покритие на проективните кодове и на получените по гореописания начин кодове с повторени координати, използваме модификация на метод 2, като тестваме само думите от вида $(\underbrace{0, \ldots, 0}_{k}, a)$, $a \in F_3^{n-4}$ и $wt(a) \geq t_3[n, 4]$. Ако по време на търсенето намерим лидер на съседен клас с тегло равно на долната граница за $t_3[n, 4]$ го прекъсваме, защото е намерен код със зададения радиус на покритие. Ако намерим лидер на съседен клас с тегло по-голямо от $t_3[n, k]$, спираме търсенето, защото радиусът на покритие на кода е със сигурност по-голям от зададената стойност. Модифицираният по този начин алгоритъм дава много бързо резултат, когато радиусът на покритие на кода е по-голям от долната граница за $t_3[n, 4]$ и успяваме за кратко време да отхвърлим голям брой кодове, имащи по-голям от предварително зададения радиус на покритие.

Първият отворен случай за $t_3[n, 4]$ е на дължина $n = 13$. Тестваме всички 70440 $[13, 4]$ кодове, като търсим такъв с радиус на покритие 5. Оказва се, че няма такъв и следователно $t_3[13, 4] > 5$. Намираме проективни кодове с радиус на покритие 6 и така получаваме, че $t_3[13, 4] = 6$. За следващата дължина имаме $t_3[14, 4] \geq t_3[13, 4]$ и определяме, че $t_3[14, 4] = 6$.

Ако един $[15, 4]$ код има повторени координати можем да го разглеждаме като слепване на кодове. Тогава можем да приемем, че C_2 е $[13, 4]$ код с минималния възможен радиус на покритие, а C_1 е $[2, 1]$ код с радиус на покритие 1. Тогава съгласно (3.1) получаваме, че $R(C) \geq 6 + 1 = 7$ и следователно ако $[15, 4]$ код има радиус на покритие 6, то той трябва да е проективен. Тестваме всички класифицирани 1504 такива кода и получаваме, че те са с радиуси на покритие $R([15, 4]) \geq 7$.

По аналогичен начин разглеждаме всички случаи до дължина 23 и на дължина 25 и получаваме следните резултати $t_3[13, 4] = t_3[14, 4] = 6$, $t_3[15, 4] = 7$, $t_3[16, 4] = t_3[17, 4] = 8$, $t_3[18, 4] = t_3[19, 4] = 9$, $t_3[20, 4] = 10$, $t_3[21, 4] = t_3[22, 4] = 11$, $t_3[23, 4] = 12$ и $t_3[25, 4] = 13$.

Специално ще отбележим случаят $n = 19$. За него получаваме, че е възможно да имаме повторени координати в пораждащата му матрица. Затова тестваме дали всички 80233923 $[19, 4]$ кода имат радиус на покритие равен на долната граница, а именно $R = 9$. Оказа се, че съществуват само два троични $[19, 4]$ кода с минимален

радиус на покритие 9. Единият е проективен и има следният тегловни спектър $A_0 = 1, A_9 = 2, A_{11} = 12, A_{12} = 16, A_{13} = 30, A_{14} = 14, A_{15} = 2, A_{17} = 4$.

Този резултат беше полезен и при определянето на минималния радиус на покритие на $[21, 4]$ кодовете. За тях също беше възможно да има кодове с минимален радиус на покритие и с повторени координати. Тези кодове могат да се разглеждат като слепване на $[2, 1]$ код с радиус на покритие 1 и $[19, 4]$ код с минимален радиус на покритие 9. Тъй като съществуват само два $[19, 4]$ кода с радиус на покритие 9, ние ги разширихме по всевъзможните начини с 2 еднакви координати. Радиусите на покритие на всички кодове се оказаха по-големи от 10. Също тествахме и проективните $[21, 4]$ кодове и техните радиусите на покритие се оказаха по-големи от 10, а намерихме такива с радиус на покритие 11. Следователно $t_3[21, 4] = 11$.

За останалите кодове на размерност 4 получаваме подобрени стойности за долната граница за функцията $t_3[n, 4]$. Така четвъртият ред на таблицата за $t_3[n, k]$ се променя по следния начин.

n	5	6	7	8	9	10	11	12	13
	1	1	2	2	3	4	4	5	6
n	14	15	16	17	18	19	20	21	22
	6	7	8	8	9	9	10	11	11
n	23	24	25	26	27	28	29	30	31
	12	12-13	13	13-14	14-15	15-16	16-17	16-17	17-18
n	32	33	34	35	36	37	38	39	40
	17-19	18-19	18-20	19-21	19-21	20-22	20-23	21-23	21-24

Забележка 2. Във всички случаи, в които са определени стойностите на функцията $t_3[n, k]$ (т.е. $13 \leq n \leq 20$), има проективни кодове с минимален радиус на покритие.

Резултатите в този раздел са съвместни с Илия Буюклиев и са публикувани в [215]. Представени са на International Workshop on Algebraic and Combinatorial Coding Theory в Кранево, България през 2004.

Глава 4

Поведение на шумозащитни кодове при откриване и коригиране на грешки

В тази глава е изследвано поведението при контрол на грешки на класове линейни кодове. Отделено е специално внимание на троичния квадратично-остатъчен $[13, 7, 5]$ код. Показано е, че той е квази-съвършен и са предложени два ефективни декодиращи алгоритъма за него. Направено е сравнение на поведението при откриване и коригиране на грешки на класове линейни кодове. Решен е отворен проблем (5.1) от книгата на MacWilliams и Sloane [145].

Достатъчни условия един линеен код да е подходящ в целия интервал $\left[0, \frac{q-1}{q}\right]$ или в негови подинтервали, основаващи се на дължината и минималното разстояние на дуалния му код, са представени от Додунекова и Николова [3, 72, 73].

За да бъдат проверени тези достатъчни условия, трябва да се знаят тегловното разпределение $\bar{A}$ на кода C или минималното разстояние на дуалния му код $C^{\perp}$. Тези параметри, особено $\bar{A}$, не са известни за повечето кодове и определянето им е изчислително трудна задача, както беше показано в раздел 2.2. Следователно проверката дали един код е подходящ за контрол на грешки също е трудна задача и определянето на класове кодове, които са подходящи, представлява значителен интерес както от теоретична гледна точка, така и от гледна точка на приложенията. Освен това, ако имаме подходящи кодове, от тях могат да се конструират нови, също подходящи кодове. От монографията на Klöve и Korzhik [132] са известни следните конструкции на нови подходящи кодове от известни такива.

Нека S_k да е една пораждаща матрица на Симплекс кода, т.е. S_k е $k \times \frac{q^k - 1}{q - 1}$

матрица над GF(q), такава че всички нейни стълбове са не нулеви, различни и не пропорционални. Нека C да е $[n,k]$ код над GF(q) с пораждаща матрица G. Тогава ще означаваме с C^* слепването на C и Симплекс кода, т.е. $\left[n+\frac{q^k-1}{q-1},k\right]$ кода с пораждаща матрица $G^*=[G|S_k]$. Разглежда се също и итерация на *-операцията, като се дефинира C^{r*} по следния начин:

$C^{0*}=C$,

$C^{(r+1)*}=(C^{r*})^*$.

Теорема 4.0.1. *[132] Ако C е подходящ, то и C^* е подходящ.*

Теорема 4.0.2. *[132] Ако C е $[n,k]_q$ код и $r\geq\max\{0,n-2d\}$, то C^{r*} е подходящ.*

Теорема 4.0.3. *[132] Ако C е $[n,k]_q$ код и $r\geq n-2$, то C^{r*} е подходящ.*

4.1 Поведение при откриване на грешки на двоични и троични линейни кодове с дължини до 33

Класификации на двоични циклични кодове, на двоични кодове с максимално минимално разстояние и на троични циклични и негациклични кодове са направени съответно в [10], [75], [117], [211]. Ние пресмятаме $\bar{A}$, $\bar{\alpha}$ и $Q_{h,l}$ за всички тези кодове, като използваме описаните във втора глава алгоритми. За определянето на $\bar{\alpha}$ за кодове с големи размерности е използван Метод 1, като са генерирани всички комбинации от $\rho=t+1,\ldots,R(C)$ стълба на проверочната матрица H и за всяка стойност на ρ са преброени различните $n-k$-мерни вектори, получени от тези комбинации. Ще отбележим, че не е необходимо да се тестват стойностите на ρ от 1 до t, защото е известно, че всеки $n-k$-мерен вектор с тегло $1,\ldots,t$ е единствен лидер на съседен клас, т.е. стойностите на $\alpha_0,\alpha_1,\ldots,\alpha_t$ са съответно $1,\binom{n-k}{1},\ldots,\binom{n-k}{t}$. За пресмятането на $Q_{h,l}$ за кодове с малки размерности е използван Метод 2.

След като са пресметнати $\bar{A}$, $\bar{\alpha}$ и $Q_{h,l}2$, може да се провери с произволна прецизност условието кодът да е t-подходящ, както и дискретните достатъчни условия за това. Така различните кодове могат да бъдат сравнявани и да се определи най-добрия по отношение на вероятността за неоткрита грешка и за коректно предаване.

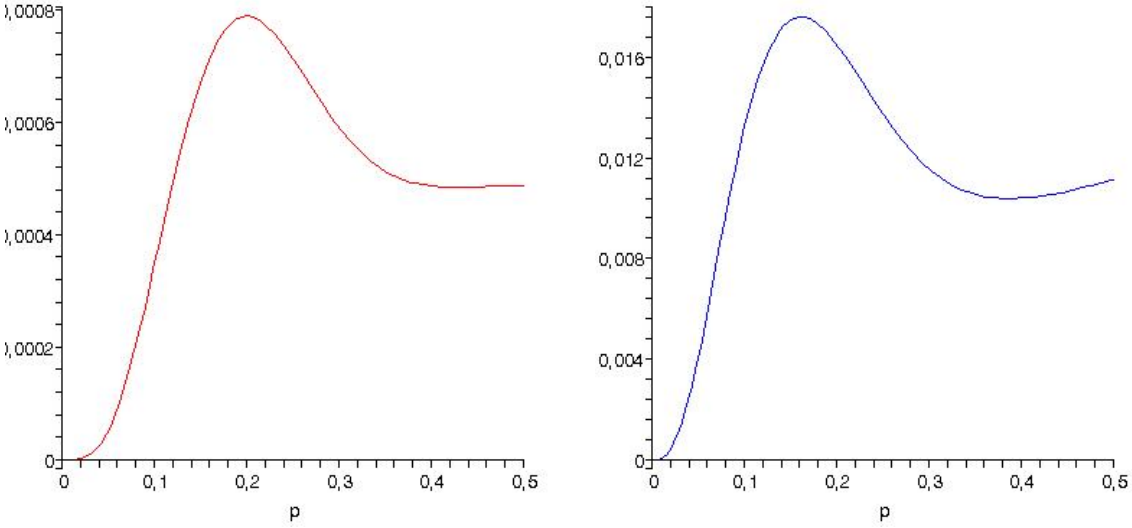
Ние използвахме програмата $ISMONOT$ написана на Maple, за да проверим монотонстта на функцията $P_{ue}^{(t)}(C,\varepsilon)$ за изследваните кодове за крайно множест-

во от стойности на ε в интервала $\left[0, \frac{q-1}{q}\right]$ със стъпка 10^{-5} и така определихме всички кодове, които не са t-подходящи. Пакетът Maple беше избран тъй като проверката на условието за монотонност става за линейно време, т.е. не ни е нужно бързодействие на програмата, а системата предлага възможности за работа с дроби. Така сравненията на стойностите на функцията $P_{ue}^{(t)}(C, \varepsilon)$ на всяка стъпка са максимално прецизни, тъй като сравняваме самите дроби, а не тяхно представяне с мантиса и порядък както е в C++. Също така пакетът дава възможност за построяване на графики на функции и представените по-долу графики са направени с негова помощ.

Резултатите са представени в Таблици 4.1.1–4.1.4 в приложението към дисертацията. В първите три таблица са дадени резултатите за цикличните и негацикличните кодове. Включени са само кодовете, които са подходящи за коригиране на грешки като са посочени техните дължина, размерност, минимално разстояние, пораждащ полином и стойностите на t, за които те са подходящи. Пораждащите полиноми са записани като последователност от коефициенти като този пред най-високата степен е на първо място. С ^{o} са отбелязани кодовете имащи максимално минимално разстояние, а с * тези, които имат най-малкия възможен радиус на покритие. В последната таблица са резултатите за двоичните линейни кодове с максимално минимално разстояние. Оказва се, че всички кодове с максимално минимално разстояние са t-подходящи. В допълнение са пресметнати и радиусите им на покритие.

Като илюстрация за това как могат да бъдат използвани тези данни, на фиг. 4.1 е представена графиката на вероятността за неоткрита грешка на двоичния цикличен $[21, 10, 4]$ код. Той има минимално разстояние 4 и може да коригира една грешка. Следователно $t = 0, 1$.

От графиката се вижда, че този код не е нито $t = 0$, нито $t = 1$ подходящ, ако искаме да го използваме в целия интервал $\varepsilon \in \left[0, \frac{1}{2}\right]$. Много често в практическите приложения се интересуваме от използването на кода в някакъв подинтервал на интервала от вероятности за грешка $\varepsilon \in \left[0, \frac{q-1}{q}\right]$. Като разполагаме с резултатите, получени в този раздел, можем да направим заключения за това дали кодът е подходящ в някой подинтервал на $\left[0, \frac{q-1}{q}\right]$. Ако се върнем на двоичния цикличен $[21, 10, 4]$ код можем да направим заключението, че той е подходящ в интервала $\varepsilon \in [0, 0.21]$ и 1-подходящ в интервала $\varepsilon \in [0, 0.16]$, т.е. може да се използва за откриване и съответно за коригиране на 1 грешка в тези интервали.

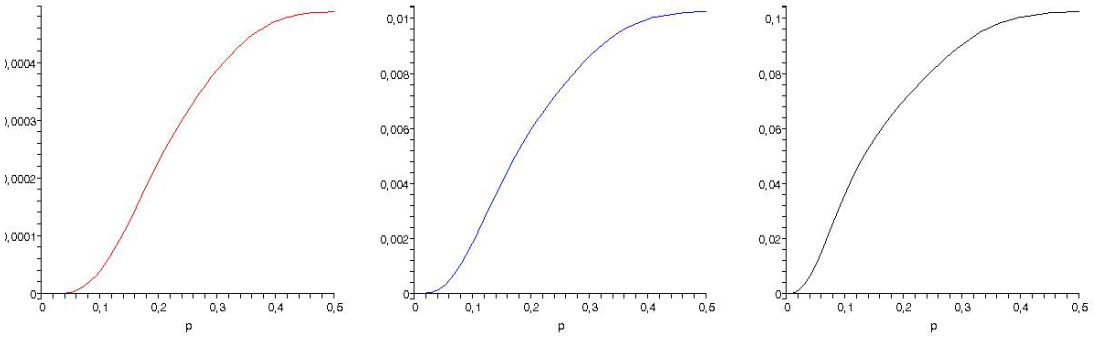


Фигура 4.1: P_{ue} **на двоичния цикличен** $[21, 10, 4]$ **код.**

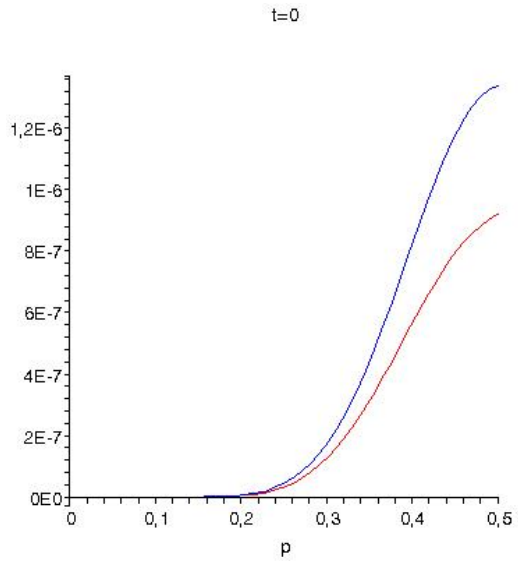
На фигура 4.2 е показан пример на $t = 0, 1, 2$ подходящ в целия интервал $\left[0, \frac{1}{2}\right]$ двоичен цикличен $[21, 10, 5]$ код.

Пример за сравнение на вероятността за коректно декодиране на два двоични с максимално минимално разстояние $[25, 5, 12]$ кода е представен на фигура 4.3.

Резултатите в този раздел са получени самостоятелно и са публикувани в [221]. Част от резултатите са представени на International Workshop on Algebraic and Combinatorial Coding Theory в Банско, България [209].



Фигура 4.2: P_{ue} **на двоичния цикличен** $[21, 10, 5]$ **код.**



Фигура 4.3: P_{corr} **на двоичните с максимално минимално разстояние** $[25, 5, 12]$ **кодове.**

4.2 Пресмятане на тегловното разпределение на съседните класове на циклични кодове

В този раздел предлагаме ефективен метод за пресмятане на тегловното разпределение на лидерите на съседни класове $\bar{\alpha}$ на циклични кодове.

Нека C е цикличен $[n, k]$ код над крайно поле с q елемента $F_q = GF(q)$ и нека пораждащия полином на C да е $g(x)$ от степен $deg(g(x)) = n - k$. С V означаваме n-мерното векторно пространство над F_q. Тогава изображението $\sigma : V \to V$ ще е цикличното завъртане на думите от V, т.е.

$$\sigma(a_0, a_1, a_2, \ldots, a_{n-1}) = (a_{n-1}, a_0, a_1, \ldots, a_{n-2}).$$

Теорема 4.2.1. *Нека C да е цикличен $[n, k]$ код с пораждащ полином $g(x) = x^{n-k} + g_{n-k-1}x^{n-k-1} + \ldots + g_1x + g_0$ и нека $a = (a_0, a_1, \ldots, a_{n-k-1}, 0, \ldots, 0)$ да е вектор от пространството V. Тогава следните два съседни класа съвпадат*

$$\sigma(a) + C = r + C,$$

където $r = (0, a_0, a_1, \ldots, a_{n-k-2}, 0, \ldots, 0) - a_{n-k-1}(g_0, g_1, \ldots, g_{n-k-1}, 0, \ldots, 0)$.

Доказателство. Нека да съпоставим на вектор от V полином от пръстена на

полиномите $F_q[x]$

$$v = (v_0, v_1, \ldots, v_{n-1}) \to v(x) = v_0 + v_1 x + \ldots + v_{n-1}x^{n-1}.$$

Ако C е цикличен код с пораждащ полином $g(x)$ от степен $m = n - k$, то е добре известно, че

$$b \in a + C \Leftrightarrow g(x)|(b(x) - a(x)).$$

Нека $a = (a_0, a_1, \ldots, a_{n-k-1}, 0, \ldots, 0)$ да е вектор от V. Тогава

$$b = \sigma(a) = (0, a_0, a_1, \ldots, a_{n-k-1}, 0, \ldots, 0)$$

и съответния му полином е

$$b(x) = \sigma(a)(x) = xa(x) = a_0 x + a_1 x^2 + \ldots + a_{n-k-1}x^{n-k}.$$

Остатъкът от делението на $b(x)$ на $g(x) = x^{n-k} + g_{n-k-1}x^{n-k-1} + \ldots + g_1 x + g_0$ е

$$r(x) = b(x) - a_{n-k-1}g(x) =$$

$$= a_0 x + a_1 x^2 + \ldots + a_{n-k-2}x^{n-k-1} - a_{n-k-1}(g_{n-k-1}x^{n-k-1} + \ldots + g_1 x + g_0)$$

и съответстващия му вектор е

$$r = (0, a_0, a_1, \ldots, a_{n-k-2}, 0, \ldots, 0) - a_{n-k-1}(g_0, g_1, \ldots, g_{n-k-1}, 0, \ldots, 0).$$

Ако два вектора a и b принадлежат на един и същи съседен клас на кода C, то съответстващите им полиноми имат един и същи остатък при деление на $g(x)$. Следователно можем да получим по един представител от всеки съседен клас, ако разглеждаме векторите от вида

$$a = (a_0, a_1, \ldots, a_{n-k-1}, 0, \ldots, 0).$$

Нека проверочната матрица на кода C да е от вида $H = [I_{n-k}|B]$. Ако $a = (a_0, a_1, \ldots, a_{n-k-1}, 0, \ldots, 0)$ е вектор от V, то неговия синдром е $s(a) = Ha^t = (a_0, a_1, \ldots, a_{n-k-1})^t$. Според Теорема 4.2.1 имаме $\sigma(a) + C = r + C$ и тогава

$$s(\sigma(a)) = (0, a_0, a_1, \ldots, a_{n-k-2}) - a_{n-k-1}(g_0, g_1, \ldots, g_{n-k-1}).$$

Следователно от синдрома на дума от V можем да пресметнем синдромите на всичките нейни циклични завъртания. ◇

Нека $G =< \sigma >$ да е циклична група породена от σ. Групата има n елемента.

Лема 4.2.2. *Нека C да е цикличен $[n, k]$ код и $a \in V$. Нека $B = \{\sigma(z)|z \in a + C\}$. Тогава B е съседен клас на кода C и $B = \sigma(a) + C$.*

Доказателство. $\sigma(a+c_1) - \sigma(a+c_2) = \sigma(a) + \sigma(c_1) - \sigma(a) - \sigma(c_2) = \sigma(c_1 - c_2) =$
$= \sigma(c_3) \in C.$ ◇

От тази лема следва, че можем да разглеждаме действието на G над множеството от всички съседни класове на кода C като $\sigma(a + C) = \sigma(a) + C$. Под това действие, множеството от всички съседни класове се разбива на не пресичащи се орбити $O(a + C) = \{\sigma^t(a) + C \mid t = 0, \ldots, n - 1\}$ и дължината на всяка орбита (т.е. броят на различните съседни класове) е делител на n. Всички съсед-

ни класове, които принадлежат на една и съща орбита, имат едно и също тегловно разпределение. Можем да получим по един представител на съседен клас от всяка орбита като вземем векторите $a = (a_0, a_1, \ldots, a_{n-k-1}, 0, \ldots, 0)$, $\phi(a) = (0, a_0, a_1, \ldots, a_{n-k-2}, 0, \ldots, 0) - a_{n-k-1}(g_0, g_1, \ldots, g_{n-k-1}, 0, \ldots, 0)$, $\phi^2(a), \ldots, \phi^{n-1}(a)$. Ако последните k координати на векторите a и b са нули, те принадлежат на съседни класове от една и съща орбита, тогава и само тогава когато съществува s, такова че $b = \phi^s(a)$.

Каква е ефективността на този подход? Ако правим пресмятанията по Метод 2, броят на стъпките на алгоритъма ще е пропорционален на $n3^n$. Ако тестваме само един съседен клас от всяка орбита, можем да намалим този брой. По-точно, ако имаме s различни орбити, броят на стъпките ще е пропорционален на $n3^{k+s}$ и този брой е по-малък от $n3^n$, защото $s < n - k$. Времевата сложност може да бъде намалена допълнително, ако се вземе под внимание факта, че всички вектори с тегла по-малки или равни на $t = \left\lfloor \frac{d-1}{2} \right\rfloor$ са единствени лидери на съседни класове. Следователно трябва да тестваме само векторите с тегла по-големи от t.

Като илюстрация на метода сме пресметнали тегловните разпределения на лидерите на съседни класове на някои троични циклични кодове с дължини до 14. Пораждащите матрици на кодовете са взети от класификацията направена в [10]. Получените резултати (списък на троичнити циклични кодове с дължини до 14, корените на пораждащите им полиноми и тегловното разпределение на лидерите на съседните им класове $\bar{\alpha}$) са представени в Таблица 4.2.1 в приложението.

Резултатите в този раздел са получени съвместно с Евгения Великова и са публикувани в [218].

4.3 Поведение на троичният [13,7,5] квадратично-остатъчен код

В този раздел е изследван троичният $[13, 7, 5]$ квадратично-остатъчен код, показано е, че е подходящ за откриване и коригиране на грешки и са представени два ефективни алгоритъма за декодирането му.

Нека да означим с C троичинят $[13, 7, 5]$ квадратично-остатъчен код. Той може да бъде описан по следния начин. Нека да приемем, че крайното поле $GF(3^3)$ се поражда от корена β на примитивния полином $x^3 - x + 1$. Тогава $\alpha = \beta^2$ е примитивен 13-ти корен на единицата в $GF(3^3)$. Квадратичните остатъци по модул

13 са

$$Q = \{1, 3, 4, 9, 10, 12\}.$$

Кодът C се дефинира като троичен линеен код с дължина 13 и пораждащ полином

$$g(x) = \prod_{j \in Q} (x - \alpha^j) = x^6 + 2x^4 + 2x^3 + 2x^2 + 1$$

(виж [145, Chapter 16]). Размерността на C е $dim(C) = 7$, а минималното му разстояние е 5.

Ще изследваме поведението за контрол на грешки на C, когато той се използва за коригиране на 2 или по-малко грешки и комуникацията се осъществява през троичен симетричен канал без памет.

За да анализираме поведението при откриване и коригиране на грешки на кода C, първо пресмятаме тегловното разпределение на съседните му класове. За целта използваме програмата CovRadM2. То е представено в следващата таблица.

Тегла / Брой със. кл.	0	1	2	3	4	5	6	7	8	9	10	11	12	13
1	1					78	182	286	390	520	442	234	26	28
26		1			15	57	167	295	432	500	429	213	62	16
78			1	4	11	54	164	304	435	494	427	212	67	14
78			1	2	16	57	146	319	429	509	409	215	72	12
78			1	2	14	64	144	296	469	486	407	222	70	12
78			1	2	14	62	151	294	446	526	384	220	77	10
78				5	15	50	163	294	446	520	381	233	68	12
78				4	16	56	154	288	461	514	390	212	84	8
78				3	20	55	138	311	463	495	386	238	66	12
26				2	23	54	131	328	438	512	397	210	84	8
52				4	18	51	149	313	444	497	415	207	79	10
78				4	16	58	147	290	484	474	413	214	77	10

От таблицата се вижда, че радиусът на покритие на кодът C е 3.

Нека да приемем, че кодът C се използва едновременно за откриване и за коригиране на грешки през троичен симетричен канал. Използваме Теорема 1.2.17 и програмата `CheckDiscreetSuffCond` написана на Maple, за да проверим дали се удовлетворяват дискретните достатъчните условия за един линеен код да е t-подходящ за контрол на грешки. Минималното разстояние на кода C е 5 и следователно проверяваме тези условия за $t = 0, 1, 2$. Тъй като вече имаме пресметнати тегловните разпределения на съседните класове, проверката става за линейно време. Тук отново използваме възможността на Maple да сравнява рационални дроби, така че проверката на достатъчните условия да е максимално прецизна. В резул-

тат получихме, че кодът C е t-подходящ за $t = 0, 1, 2$ (а следователно и t-добър) в целия интервал $\left[0, \frac{2}{3}\right]$, т.е. е подходящ както за откриване на грешки така и за коригиране на 1 или 2 грешки за произволен троичен симетричен канал.

Алгебричен декодиращ алгоритъм за троичния $[13, 7, 5]$ квадратично-остатъчен код е представен в [115]. Като използваме допълнителна информация за структурата на кода, ние предлагаме два нови декодиращи алгоритъма за този код, които са с по-малка сложност в сравнение с представения в [115] алгоритъм.

А. Първото предложение се основава на факта, че кодът C е БЧХ код.

Дефиниция 4.3.1. *Нека β да е пораждащ елемент на крайното поле $GF(q)$. Цикличният код с дължина n над $GF(q)$ е БЧХ код с конструктивно разстояние δ, ако пораждащия му полином $g(x)$ е полинома от най-ниска степен над $GF(q)$, който има за корени $\beta^b, \beta^{b+1}, \ldots, \beta^{b+\delta-2}$. Тогава c е кодова дума, тогава и само тогава когато $c(\beta^b) = c(\beta^{b+1}) = \ldots = c(\beta^{b+\delta-2}) = 0$. Минималното разстояние на БЧХ кода е по-голямо или равно на конструктивното му разстояние δ.*

Твърдение 4.3.2. *Кодът C е БЧХ код.*

Доказателство. Да разгледаме $\gamma = \alpha^8$ (или α^{11}), които са примитивни 13-ти корени на единицата. Тогава $\{\gamma^5, \gamma^6, \gamma^7, \gamma^8\} = \{\alpha, \alpha^9, \alpha^4, \alpha^{12}\}$ (или $= \{\alpha^3, \alpha, \alpha^{12}, \alpha^{10}\}$) и следователно пораждащия полином $g(x)$ на C има за корени множество от четири последователни нули.

Следователно C може да бъде декодиран като се използва всеки от стандартните БЧХ декодиращи алгоритми за коригиране на две грешки (виж [19, Chapter 7]). ◇

От това твърдение и от БЧХ границата става ясно, че кодът C има минимално разстояние 5. Това се вижда също и от таблицата с тегловните разпределения на C.

В. Вторият подход използва друго важно свойство на C. Корени на полинома $g(x)$ са α и α^{-1} едновременно, т.е. C е реверсивен (виж [145, p.206]). По-точно

$$g(x) = g_1(x)g_2(x) = (x^3 + x^2 + x + 2)(x^3 + 2x^2 + 2x + 2),$$

където $g_1(\alpha) = 0$, $g_2(\alpha^{-1}) = 0$. Ще отбележим, че реверсивността е свойство на всички квадратично-остатъчни кодове с дължини $n \equiv -1 \bmod 4$, т.е. когато -1 е квадратичен остатък $\bmod n$ (виж [143, p.87]).

Нека $r(x) = c(x) + e(x)$ да е получената дума, когато е била изпратена кодовата дума $c(x) \in C$ и $e(x)$ да е векторът-грешка. Ще използваме локаторите $1, \alpha, \alpha^2, \ldots, \alpha^{12}$. Нека да разгледаме синдромите

$$S_i = e(\alpha^i) \in GF(3^3).$$

Тъй като корените на $g(x)$ са

$$\{\alpha, \alpha^3, \alpha^9\} \cup \{\alpha^{12} = \alpha^{-1}, \alpha^{10} = \alpha^{-3}, \alpha^4 = \alpha^{-9}\}$$

можем да пресметнем синдромите S_1 и S_{-1} от $r(x)$. Тази информация се оказва достатъчна, за да изведем индикатор, който да ни показва дали са се появили 1, 2 или 3 грешки по време на комуникацията. Ще отбележим, че тъй като радиусът на покритие е три, можем да приемем, че броя на грешките е най-много три, т.е. полиномът $e(x)$ има най-много три не нулеви коефициента. Тъй като минималното разстояние на C е 5, може да се случи грешка с тегло две и друга грешка с тегло три да имат един и същи синдром. В такъв случай, според декодирането по максималното правдоподобие, решаваме че са се появили две грешки.

Ясно е, че $S_1 = 0$ и $S_{-1} = 0$, тогава и само тогава когато получената дума $r(x)$ е без грешки. Да приемем, че $S_1 \neq 0$ и за удобство да означим с $\nu = S_1 S_{-1}$.

Твърдение 4.3.3. *$\nu = 1$, тогава и само тогава когато се е появила точно една грешка в процеса на комуникация.*

Доказателство. Да допуснем, че $e(x) = ux^i$ за някое i, $0 \leq i \leq 12$, $u \in GF(3)^* = \{1, 2\}$. Тогава $S_1 = u\alpha^i$, $S_{-1} = u\alpha^{-i}$ и $\nu = u^2 = 1$.

Да допуснем сега обратното, т.е. $\nu = 1$. Да приемем първо, че имаме две грешки и те са в позиции i и j, т.е. $e(x) = ux^i + vx^j, u, v \in GF(3)^*, 0 \leq i \leq j \leq 12$. Тогава

$$S_1 = u\alpha^i + v\alpha^j,$$
$$S_{-1} = u\alpha^{-i} + v\alpha^{-j},$$

и

$$\nu = 1 = u^2 + v^2 + uv(\alpha^{i-j} + \alpha^{j-i}) = 2 + uv(\alpha^{i-j} + \alpha^{j-i}),$$

което е невъзможно, защото минималното тегло на C е 5 и α не може да е корен на реверсивен полином от степен по-малка от 13 с по-малко от 5 не нулеви троични коефициента.

Да приемем сега, че имаме три грешки, т.е. $e(x) = ux^i + vx^j + wx^k, u, v, w \in GF(3)^*, 0 \leq i \leq j \leq k \leq 12$. В този случай

$$S_1 = u\alpha^i + v\alpha^j + w\alpha^k,$$
$$S_{-1} = u\alpha^{-i} + v\alpha^{-j} + w\alpha^{-k},$$
$$\nu = 1 = u^2 + v^2 + w^2 + uv(\alpha^{i-j} + \alpha^{j-i}) + uw(\alpha^{i-k} + \alpha^{ki}) + vw(\alpha^{j-k} + \alpha^{k-j}).$$

Да положим $a = uv\alpha^{i-j}$, $b = vw\alpha^{j-k}$. Тогава от горното равенство получаваме

$$1 = a + \frac{1}{a} + b + \frac{1}{b} + ab + \frac{1}{ab}.$$

Или еквивалентното му

$$(1 + a + b + ab)(1 + \frac{1}{ab}) = 0.$$

Както и в предишния случай $1 + a + b + ab \neq 0$, защото в противен случай α би била корен на реверсивен троичен полином с тегло по-малко от 5. Ако

$$1 + \frac{1}{ab} = 0,$$

то $\alpha^{2(i-k)} = 1$, което води до $i = k$, и отново получаваме противоречие.

Следователно не са възможни три грешки и ако $\nu = 1$ имаме само една грешка. $\diamond$

Забележка. В случая когато $\nu = 1$, т.е. имаме една грешка, стойността на грешката и нейният локатор могат да бъдат пресметнати от S_1.

По-точно, ако $S_1 = u\alpha^i$, то

$$S_1^{13} = u^{13}\alpha^{13i} = u,$$

и

$$uS_1 = \alpha^i.$$

Да допуснем сега, че $\nu \neq 1$. Нека да означим с $Tr(.)$ следата от $GF(27)$ към $GF(3)$. Дефинираме μ по следния начин: $\mu = \frac{S_1}{S_{-1}}$.

Твърдение 4.3.4. *$Tr(\nu - \nu^4) = 1$, тогава и само тогава когато са възникнали две грешки. Позициите на тези грешки се представят от α^i и α^j и удовлетворяват равенствата $\alpha^i = \sqrt{\mu \cdot \zeta}$ и $\alpha^j = \sqrt{\frac{\mu}{\zeta}}$, където ζ е корен на полинома $1-(\nu+1)z+z^2$. Стойностите на грешките e_i и e_j, които са в позиции съответстващи на α^i и α^j, се пресмятат така:*

$$e_i = \frac{S_1\zeta}{\alpha^i(\zeta + 1)}, \quad e_j = \frac{S_1}{\alpha^j(\zeta + 1)}.$$

Доказателство. Нека $e(x)$ да е зададен по следния начин $e(x) = ux^i + vx^j$. Ще отбележим, че ν е равно на нула само ако са възникнали нула или три грешки. Това твърдение е очевидно за нула и една грешки. Да допуснем, че са възникнали две грешки и тогава имаме $S_1 = u\alpha^i + v\alpha^j$. Ако S_1 е равно на нула, то би било вярно $\alpha^{j-i} \in GF(3)$, което е противоречие. Подобно твърдение е вярно за $S_{12} = S_{-1}$ и можем да заключим, че и S_1 и S_{-1} не са равни на нула. Следователно μ и ν са определени и не са нули.

Да разгледаме матрицата

$$\mathcal{S} = \begin{pmatrix} S_{10} & S_{12} & S_1 \\ S_{12} & S_1 & S_3 \end{pmatrix}, \tag{4.1}$$

която може да бъде представена като

$$\mathcal{S} = \begin{pmatrix} \alpha^{10i} & \alpha^{10j} \\ \alpha^{12i} & \alpha^{12j} \end{pmatrix} \begin{pmatrix} u & 0 \\ 0 & v \end{pmatrix} \begin{pmatrix} 1 & \alpha^{2i} & \alpha^{4i} \\ 1 & \alpha^{2j} & \alpha^{4j} \end{pmatrix}. \tag{4.2}$$

От равенство (4.2) се вижда, че $\mathcal{S}$ има ранг две и всяко решение $(\sigma_0, \sigma_1, \sigma_2)$ на системата от уравнения $\mathcal{S}\sigma^T = 0$ трябва да съответства на решение на уравнението

$$\begin{pmatrix} 1 & \alpha^{2i} & \alpha^{4i} \\ 1 & \alpha^{2j} & \alpha^{4j} \end{pmatrix} \begin{pmatrix} \sigma_0 \\ \sigma_1 \\ \sigma_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Следователно решението трябва да е кратно на полинома

$$\sigma(x) = \sigma_0 + \sigma_1 x^2 + \sigma_2 x^4 = (x^2 - \alpha^{2i})(x^2 - \alpha^{2j}).$$

Като преобразуваме матрицата $\mathcal{S}$ във вида

$$\mathcal{S} = \begin{pmatrix} S_{-1}^3 & S_{-1} & S_1 \\ S_{-1} & S_1 & S_1^3 \end{pmatrix}, \tag{4.3}$$

може да се провери, че $\sigma(x)$ се представя като $\sigma(x) = \mu^2 - \mu(\nu+1)x^2 + x^4$. Полагаме $y = x^2/\mu$ и тогава $\sigma'(y) = \mu^2(1-(\nu+1)y+y^2)$. Следователно ако имаме две грешки, полиномът $1-(\nu+1)y+y^2$ трябва да е разложим над $GF(27)$ и двата му корена са α^{2i}/μ и α^{2j}/μ. Така доказваме втората част на твърдението. Може да се провери чрез непосредствено пресмятане, че полиномът $1 - (\nu + 1)y + y^2$ се разлага над $GF(27)$ ако ν е или 0 или 1 или е корен на полинома $1 - z - z^3 - z^9 + z^4 + z^{10} + z^{12}$, което за удобство може да се запише като $Tr(\nu - \nu^4) = 1$. Случаят $\nu = 0$ отговаря на наличието на нула или три грешки, а случаят $\nu = 1$ отговаря на една грешка. Ще докажем обратното чрез броене. Имаме 12 начина да изберем ν, такова че $Tr(\nu - \nu^4) = 1$ и 26 начина да изберем S_1 при зададено ν. Следователно има точно $12 \cdot 26 = 312$ възможни избора на синдроми S_1 и S_{-1}, такива че полиномът $1-(\nu+1)y+y^2$ да се разлага над $GF(27)$. От друга страна, има точно 312 възможни вектор-грешки с тегло 2, което доказва втората част на твърдението.

Накрая, за да определим стойностите на грешките, ще разгледаме следните равенства:

$$\begin{pmatrix} \alpha^i & \alpha^j \\ \alpha^{3i} & \alpha^{3j} \end{pmatrix} \begin{pmatrix} e_i \\ e_j \end{pmatrix} =$$

$$= \begin{pmatrix} 1 & 1 \\ \alpha^{2i} & \alpha^{2j} \end{pmatrix} \begin{pmatrix} \alpha^i & 0 \\ 0 & \alpha^j \end{pmatrix} \begin{pmatrix} e_i \\ e_j \end{pmatrix} = \begin{pmatrix} S_1 \\ S_1^3 \end{pmatrix}.$$

Като решим тази система от уравнения използвайки равенствата $\nu\mu = S_1^2$, $\alpha^{2i} = \mu\zeta$, $\alpha^{2j} = \mu/\zeta$ и $\nu + 1 = \zeta + 1/\zeta$, получаваме изразите за стойностите на грешките. $\diamond$

Забележка. Има 12 стойности на ν, които определят всички възможни грешки с тегло 2. Могат да се пресметнат предварително дванадесетте корена на поли-

нома $1-(\nu+1)y+y^2$ (по един за всяко ν) и така да се сведе до минимум сложността на процедурата за коригиране на две грешки. Тогава позициите на двете грешки се определят от Твърдение 4.3.4, както и от съответните стойности на грешките.

Като се основаваме на Твърдения 4.3.3 и 4.3.4, предлагаме следния алгебричен алгоритъм за декодиране на C.

Стъпка 1. Пресмятат се S_1 и S_{-1}.

Стъпка 2. Ако $S_1 = S_{-1} = 0$, не са възникнали грешки. В противен случай премини към стъпка 3.

Стъпка 3. Пресмята се $\nu = S_1 S_{-1}$. Ако $\nu = 1$, възникнала е една грешка. Пресмята се локаторът на грешката α^i и стойността на грешката както е в забележката. В противен случай премини към стъпка 4.

Стъпка 4. Ако $Tr(\nu - \nu^4) = 1$, са възникнали две грешки. Намира се коренът ζ на полинома $1-(\nu+1)y+y^2$. Позициите и стойностите на грешките се дават от Твърдение 4.3.4. В противен случай, са възникнали три грешки.

Този алгоритъм е особено ефективен заради начина на определяне на позициите на грешките като декодирането се свежда до търсене в таблица, съдържаща 12 елемента от $GF(27)$.

Резултатите в този раздел са получени съвместно със Стефан Додунеков и Ralf Kötter и са публикувани в [212]. Представени са на International Workshop on Algebraic and Combinatorial Coding Theory в Псков, Русия

4.4 Сравнение на поведението за контрол на грешки на класове линейни кодове

В този раздел формулираме въпроси свързани с поведението при откриване и коригиране на грешки на линейни кодове, чиито отговори ще дадем в изложението. На базата на конкретни примери, показваме различни аспекти от оценката на поведението на линейните кодове при контрол на грешки. Накрая, даваме отговор на един отворен проблем поставен в [145].

Основна задача в теорията на кодирането е да се оптимизира един от основните параметри n, k или d на линеен код, ако са фиксирани другите два. Така се оформят следните три оптимизационни задачи.

(1) Да се определи $N_q(k,d)$ - най-малката стойност на дължината n, за която съществува $[n,k,d]_q$ код.

(2) Да се определи $K_q(n,d)$ - най-голямата стойност на размерността k, за която съществува $[n,k,d]_q$ код.

(3) Да се определи $D_q(n,k)$ - най-голямата стойност на минималното разстояние d, за която съществува $[n,k,d]_q$ код.

Код с дължина $N_q(k,d)$, размерност k и минимално разстояние d се нарича оптимален по отношение на n. Аналогично, кодовете с параметри $[n,K_q(n,d),d]_q$ и $[n,k,D_q(n,k)]_q$ се наричат оптимални по отношение на k и d. Стойностите на $D_q(n,k)$ за малки q могат да се намерят в [29] както и на http://www.codetables.de/.

Интересен въпрос за линейни кодове с еднакви основни параметри (дължина, размерност, минимално разстояние и радиус на покритие) е доколко прецизно тези параметри определят поведението им при отриване и коригиране на грешки. За да илюстрираме проблема, ние разглеждаме три класа двоични линейни кодове и отговаряме на следните въпроси.

1. Ако кодът е оптимален по отношение на някоя от функциите $N_q(k,d)$, $K_q(n,d)$ и $D_q(n,k)$, дали той е оптимален по отношение и на другите две?

2. Дали получаваме отново не еквивалентни кодове, ако разширим с проверка по четност два не еквивалентни кода?

3. Съществуват ли кодове с еднакви основни параметри, но различни тегловни разпределения? Нещо по-вече, съществуват ли не еквивалентни кодове с еднакви основни параметри и с еднакви тегловни разпределения?

4. Съществуват ли кодове с еднакви тегловни разпределения на лидерите на съседни класове, но с различни тегловни разпределения на самите кодове?

5. Дали оптималните за откриване на грешки кодове са оптимални и по отношение на коригиране на $1,2,\ldots,t$ грешки?

6. Когато два кода имат еднакви радиус на покритие и Нютонов радиус, какви са бройките на единствените лидери на съседни класове за съседните класове с тегла по-големи от t?

7. (Изследователски проблем 5.1 от [145]). До каква степен спектърът на лидерите на съседни класове на кода $\bar{\alpha}$ определя стойностите на спектъра на лидерите на съседни класове на дуалния му код?

За да демонстрираме как различните параметри на един линеен код влияят върху поведението му при контрол на грешки, ще използваме три класа двоични линейни кодове с параметри $[15,3,7]$, $[15,3,8]$ и $[16,3,8]$. В [87] Fontaine и Peterson са представили два $[15,3,7]$ кода, като единият е оптимален по отношение на вероятността за коректно предаване P_{err}, ако вероятността за грешка на канала е малка ($\varepsilon \leq 0.307$), а другият е оптимален, ако тя е голяма ($\varepsilon \geq 0.307$). По тази причина ние решихме да изследваме целия клас от $[15,3,7]$ кодовете и свързаните с тях $[16,3,8]$ кодове, получени с добавяне към $[15,3,7]$ кодовете на проверка по четност. Третият клас, който избрахме са оптималните по отношение на ми-

нималното разстояние при дължина 15 и размерност 3 кодове, а именно $[15, 3, 8]$ кодовете.

Според [29] имаме $N_2(3, 7) = 13,\ N_2(3, 8) = 14,\ D_2(15, 3) = D_2(16, 3) = 8,$ $K_2(15, 7) = K_2(16, 8) = 5,\ K_2(15, 8) = 4$. Следователно

(i) $[15, 3, 7]$ кодовете не са оптимални по отношение на нито една от функциите $D_2(n, k), N_2(k, d)$ и $K_2(n, d)$.

(ii) $[15, 3, 8]$ и $[16, 3, 8]$ кодовете са оптимални по отношение на $D_2(n, k)$, но не са оптимални по отношение на другите две функции.

С помощта на софтуерния пакет Q-EXTENSION [23] бяха класифицирани кодовете с разглежданите параметри. Имаме 17 не еквивалентни $[15, 3, 7]$ кода, 12 не еквивалентни $[16, 3, 8]$ кода и 3 не еквивалентни $[15, 3, 8]$ кода. За всички тези кодове бяха пресметнати тегловните им разпределения $\bar{A}$, разпределенията на лидерите на съседни класове $\bar{\alpha}$, тегловните разпределения на самите съседни класове, групите им от автоморфизми (AUT), радиусите им на покритие и Нютоновите им радиуси. Резултатите са представени в таблиците по-долу. За да направим записа по-компактен, стълбовете на пораждащите матрици са записани като десетични числа, чието двоично представяне е съответния стълб. В третата колона са дадени групите от автоморфизми, а в четвъртата - тегловните спектри $\bar{A}$. В последната колона на Таблица 4.4.1 са записани номерата на съответните кодове от Таблица 4.4.2 тъй като кодовете в Таблица 4.2.2 са получени от тези в Таблица 4.4.1 чрез добавяне на проверка по четност.

Ще отбележим, че има не еквивалентни $[15, 3, 7]$ кодове, които се разширяват до един и същ $[16, 3, 8]$ код. Например кодовете с номера 1, 2 и 16 от Таблица 4.4.1 се разширяват до код номер 1 от Таблица 4.4.2.

Също така се вижда от таблиците, че почти всички кодове с еднакви дължини, размерности и минимални разстояния (т.е. кодовете от една таблица) имат различни тегловни спектри $\bar{A}$. Това означава, че те ще имат различни стойности на вероятността за неоткрита грешка $P_{ue}^{(t)}(C, \varepsilon)$ след коригиране на t грешки. Има обаче и не еквивалентни кодове с еднакви тегловни спектри. Такива са двойките кодове с номера 5,6; 8,9; 13,14 от Таблица 4.4.1 и 4,5; 9,10 от Таблица 4.4.2.

Таблица 4.4.1. $[15, 3, 7]$ кодове

№	Пораждаща матрица	AUT	Тегловен спектър	Екв. на
1	773333666622124	13824	$3z^7, 2z^8, 1z^{10}, 1z^{13}$	1
2	773333666222124	10368	$2z^7, 3z^8, 1z^9, 1z^{13}$	1
3	773333666224124	3456	$3z^7, 1z^8, 1z^9, 1z^{10}, 1z^{12}$	4
4	773333666624124	9216	$3z^7, 2z^8, 1z^{11}, 1z^{12}$	3
5	773333666224124	10368	$2z^7, 2z^8, 2z^9, 1z^{12}$	6
6	773335666222124	3456	$2z^7, 2z^8, 2z^9, 1z^{12}$	4
7	733335666224124	3456	$3z^7, 1z^8, 2z^{10}, 1z^{11}$	4
8	773331666224124	3456	$2z^7, 2z^8, 1z^9, 1z^{10}, 1z^{11}$	5
9	773335662224124	1152	$2z^7, 2z^8, 1z^9, 1z^{10}, 1z^{11}$	4
10	773331662224124	9216	$2z^7, 3z^8, 2z^{11}$	3
11	773335666224124	1728	$1z^7, 3z^8, 2z^9, 1z^{11}$	4
12	773331662244124	10368	$3z^7, 1z^9, 3z^{10}$	9
13	773335662244124	1728	$2z^7, 1z^8, 2z^9, 2z^{10}$	9
14	773351662224124	3072	$2z^7, 1z^8, 2z^9, 2z^{10}$	10
15	773351666224124	1152	$1z^7, 2z^8, 3z^9, 1z^{10}$	9
16	333311666644124	41472	$3z^7, 2z^8, 1z^9, 1z^{14}$	1
17	777711222444124	497664	$3z^7, 3z^8, 1z^{15}$	12

Таблица 4.4.2. $[16, 3, 8]$ кодове

№	Пораждаща матрица	AUT	Тегловен спектър
1	7773333666622124	82944	$5z^8, 1z^{10}, 1z^{14},$
2	7773335666222124	31104	$3z^8, 3z^9, 1z^{13},$
3	7733331666624124	73728	$5z^8, 2z^{12},$
4	7733335666224124	6912	$4z^8, 2z^{10}, 1z^{12},$
5	7773331666224124	41472	$4z^8, 2z^{10}, 1z^{12},$
6	7773335666224124	20736	$2z^8, 4z^9, 1z^{12},$
7	7733351666224124	3456	$2z^8, 3z^9, 1z^{10}, 1z^{11},$
8	7733355662224124	9216	$3z^8, 2z^9, 2z^{11},$
9	7733351662244124	10368	$3z^8, 4z^{10},$
10	7733551662224124	36864	$3z^8, 4z^{10},$
11	7733551666224124	9216	$1z^8, 4z^9, 2z^{10},$
12	7777111222444124	7962624	$6z^8, 1z^{16},$

Таблица 4.4.3 $[15, 3, 8]$ кодове

№	Пораждаща матрица	AUT	Тегловен спектър
1	777333566622124	31104	$6z^8, 1z^{12},$
2	773335166624124	4608	$5z^8, 2z^{10},$
3	773355166224124	9216	$3z^8, 4z^9,$

С програмата `ComparePue`, написа на MAPLE, ние сравнихме всички тези кодове по отношение на вероятността за неоткрита грешка след коригиране на t грешки. Тъй като кодовете са с минимални разстояния 7 или 8, то имаме $t = 0, 1, 2, 3$. Резултатите са следните:

$[15, 3, 7]_2$ кодове.
Най-добър е кодът с номер 15 за всяко $\varepsilon \in (0, 1/2]$ и $t = 0, 1, 2, 3$.

$[16, 3, 8]_2$ кодове.
Най-добър е кодът с номер 11 за всяко $\varepsilon \in (0, 1/2]$ и $t = 0, 1, 2, 3$.

$[15, 3, 8]_2$ кодове.
Най-добър е кодът с номер 3 за всяко $\varepsilon \in (0, 1/2]$ и $t = 0, 1, 2, 3$.

Интересно е също така да сравним кодовете от трите класа по отношение на вероятността за грешка. За да можем да направим това, пресметнахме тегловните разпределения на лидерите на съседни класове $\bar{\alpha}$. Получените резултати са представени в Таблица 4.4.4. Тъй като радиусът на покритие на кода е теглото на най-тежкия лидер на съседен клас, чрез това пресмятане, определяме и този параметър. Със звезда в Таблица 4.4.4 са отбелязани кодовете имащи минимален радиус на покритие. В последната колона на таблицата са дадени и Нютоновите радиуси на покритие.

Трябва да отбележим, че има кодове с различни тегловни спектри $\bar{A}$, но с еднакви спектри на лидерите на съседни класове $\bar{\alpha}$. Например такива са $[15, 3, 7]$ кодовете с номера 1, 2 и 16. Това означава, че тези кодове ще имат различни стойности на вероятността за неоткрита грешка $P_{ue}^{(t)}(C, \varepsilon)$, но ще имат еднакви стойности на вероятността за коректно предаване P_{err}. Тогава единственият параметър, който ще ги различава е $P_{ue}^{(t)}(C, \varepsilon)$ и най-добрият между тях ще е този с минимална стойност на вероятността за неоткрита грешка.

Таблица 4.4.4. Тегловни разпределения на лидерите на съседни класове и Нютонови радиуси.

Код	α_0	α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8	ν
$[15, 3, 7]$										
1^*	1	15	105	455	1192	1572	756			6
2^*	1	15	105	455	1192	1572	756			6
3^*	1	15	105	455	1226	1682	612			6
4	1	15	105	455	1192	1656	600	72		5
5^*	1	15	105	455	1225	1647	648			6
6^*	1	15	105	455	1226	1682	612			6
7^*	1	15	105	455	1226	1682	612			6
8^*	1	15	105	455	1225	1647	648			6
9^*	1	15	105	455	1226	1682	612			6
10^*	1	15	105	455	1192	1656	600	72		5
11^*	1	15	105	455	1226	1682	612			6
12^*	1	15	105	455	1260	1693	567			6
13^*	1	15	105	455	1260	1693	567			6
14	1	15	105	455	1261	1711	500	48		5
15^*	1	15	105	455	1260	1693	567			6
16^*	1	15	105	455	1192	1572	756			6
17	1	15	105	455	1158	1498	702	162		5
$[16, 3, 8]$										
1^*	1	16	120	560	1647	2764	2328	756		6
2^*	1	16	120	560	1716	3079	2430	270		6
3	1	16	120	560	1647	2848	2256	672	72	5
4^*	1	16	120	560	1681	2908	2294	612		6
5^*	1	16	120	560	1680	2872	2295	648		6
6^*	1	16	120	560	1750	3216	2286	243		6
7^*	1	16	120	560	1750	3231	2298	216		6
8^*	1	16	120	560	1716	3194	2345	240		6
9^*	1	16	120	560	1715	2953	2260	567		6
10	1	16	120	560	1716	2972	2211	548	48	5
11^*	1	16	120	560	1785	3298	2196	216		6
12	1	16	120	560	1613	2656	2200	864	162	5
$[15, 3, 8]$										
1	1	15	105	455	1159	1497	783	81		5
2	1	15	105	455	1192	1506	750	72		5
3	1	15	105	455	1261	1711	500	48		5

С програмата `ComparePcorr`, написана на MAPLE, сравняваме трите класа кодове по отношение на вероятността им за коректно предаване.

$[15, 3, 7]_2$ кодове.

Резултатите за вероятността за коректно предаване P_{err} за $[15, 3, 7]$ кодовете са вече известни от [87] и ние само ги проверихме. За $\varepsilon \leq 0.307$ най-добър е кодът с номер 14. За $\varepsilon > 0.307$ най-добри са кодовете с номера 12, 13 и 15, тъй като имат еднакви тегловни разпределения на лидерите на съседни класове $\bar{\alpha}$.

$[16, 3, 8]_2$ кодове.

Най-добър е кодът с номер 11 за всяко $\varepsilon \in (0, 1/2]$.

$[15, 3, 8]_2$ кодове.

Най-добър е кодът с номер 3 за всяко $\varepsilon \in (0, 1/2]$.

В своя работа [181] от 1956 година Slepian показва, че по отношение на вероятността за грешка, $[7, 3, 3]$ кодът е по-добър от $[7, 3, 4]$ Симплекс кода. Затова ние сравнихме най-добрите $[15, 3, 7]$ кодове с най-добрия $[15, 3, 8]$ код и получихме, че $[15, 3, 7]$ кодовете са по-добри при $\varepsilon > 0.307$. Следователно примерът на Slepian не е изолиран и не винаги кодовете с максимално минимално разстояние имат най-добро поведение по отношение на вероятността за грешка.

Когато разглеждаме резултатите за най-добри кодове по отношение на $P_{ue}^{(t)}(C, \varepsilon)$ и P_{err} виждаме, че и за трите класа изследвани кодове тези, които са най-добри по отношение на първата функция са също най-добри и по отношение на втората. Интересно е да се знае дали това е така винаги?

От Таблица 4.4.4 се вижда, че има кодове, които имат едни и същи радиуси на покритие и едни и същи Нютонови радиуси. Ако ги сравним по отношение на броя на единствените лидери (UCL) в съседните класове с тегла по-големи от t, виждаме, че те се различават. Най-добрите кодове в този смисъл са тези, които имат най-голям брой единствени лидери. Резултатите за трите класа кодове са следните.

$[15, 3, 7]_2$ кодове.

Кодове с $R = 6$ и $\nu = 6$. За съседните класове с тегло 4 - код номер N 12 (1260 UCL), за съседните класове с тегло 5 - код номер 5 (1416 UCL), за съседните класове с тегло 6 - код номер 16 (324 UCL). Този случай е интересен и с това, че радиусът на покритие и Нютоновият радиус на тези кодове са едни и същи.

Кодове с $R = 7$ и $\nu = 5$. За съседните класове с тегла 4 и 5 - код номер 14 (1227 и

1310 UCL съответно).

$[16, 3, 8]_2$ кодове.
Кодове с $R = 7$ и $\nu = 6$. За съседните класове с тегла 4 и 5 - код номер 11 (1750 и 2792 UCL съответно), за съседните класове с тегла 6 - код номер 2 (1080 UCL).
Кодове с $R = 8$ и $\nu = 5$. За съседните класове с тегла 4 и 5 - код номер 10 (1613 and 1848 UCL съответно).

$[15, 3, 8]_2$ кодове.
За съседните класове с тегла 4 и 5 - код номер 3 (1158 и 1152 UCL съответно).

Друго интересно наблюдение, направено в [15], се отнася и за изследваните от нас кодове. Има кодове с еднакви тегловни спектри, но различни спектри на лидерите на съседни класове. Такива кодове са $[15, 3, 7]$ кодовете с номера 5,6; 8,9; 13,14 и $[16, 3, 8]$ кодовете с номера 4,5; 9,10. Тегловният спектър на конкретния съседен клас показва доколко успешно всяка дума от този съседен клас се апроксимира от множеството на кодовите думи.

Ще завършим този раздел с един интересен пример, който ни дава възможност да решим изследователския проблем 5.1, поставен в книгата на MacWilliams и Sloane [145] (Глава 5, стр. 132). Проблемът е следния.

Нека C да е линеен код. Нека α_i да е броят на лидерите на съседни класове на C с тегло i и α'_i да е съответният брой за дуалния му код $C^{\perp}$. До каква степен числата $\{\alpha_i\}$ определят $\{\alpha'_i\}$?

Оказа се, че тегловното разпределение на лидерите на съседни класове на кода не определя това на дуалния му код. В таблица 4.4.5 на следващата страница са представени тегловните спектри на лидерите на съседни класове на дуалните на $[15, 3, 7]$ кодовете. Това са $[15, 12]$ кодове. Вижда се, че $[15, 3, 7]$ кодовете с номера 1, 2 и 16 имат еднакви спектри на лидерите на съседните им класове, но техните дуални - не. По-точно дуалните кодове на кодовете с номера 1 и 2 имат еднакви спектри, но код номер 16 има различен. Това е вярно също и за кодовете с номера 3,6,7,9,11 и 12,13,15.

Резултатите в този раздел са съвместни с Илия Буюклиев, Стефан Доддунеков и Wolfgang Williems и са публикувани в [217]. Представени са на Internationa Congress MASSEE в Боровец през 2003 г.

Таблица 4.4.5. Тегловни разпределения на лидерите на съседни класове на $[15, 12]$ кодовете

№	α_0	α_1	α_2	№	α_0	α_1	α_2	№	α_0	α_1	α_2	№	α_0	α_1	α_2
1	1	6	1	6	1	7		11	1	7		16	1	5	2
2	1	6	1	7	1	7		12	1	6	1	17	1	4	3
3	1	6	1	8	1	6	1	13	1	7					
4	1	6	1	9	1	7		14	1	7					
5	1	6	1	10	1	6	1	15	1	7					

Глава 5

Скъсени циклични (CRC) кодове

Ще припомним, че CRC кодовете са скъсени двоични циклични кодове. За да могат да се откриват грешки в информационна последователност от k бита $i = [i_0, i_1, \ldots, i_{k-1}]$, към тях се прибавя последователност $r = [r_0, r_1, \ldots, r_{p-1}]$ от p проверочни бита и така се формира кодовата дума $c = [i, r]$, която се състои от $n = k + p$ бита. Последователността r от проверочни битове се изчислява чрез i, така че

$$r(x) \equiv (x^p.i(x)) \ mod \ g(x),$$

където $i(x) = i_0 + i_1x + \ldots + i_{k-1}x^{k-1}$ и $r(x) = r_0 + r_1x + \ldots + r_{p-1}x^{p-1}$ са съответно информационните и проверочните битове представени като полиноми, а $g(x)$ е пораждащия полином на кода. Нека кодовата дума $c(x) = x^p i(x) + r(x) = q(x)p(x) + r(x) + r(x) = q(x)p(x)$ да е изпратена през комуникационен канал. Шумът в канала добавя съответната грешка $e(x)$ към $c(x)$ и на приемащия край се получава $v(x) = c(x) + e(x)$. Броят на не нулевите коефициенти в $e(x)$ е точно броя на сгрешените битове в $v(x)$, а техните степени са позициите, в които са възникнали грешките. Откриването на грешките става като се пресметне остатъка

$$r'(x) \equiv v(x) \ mod \ g(x).$$

Ако той е 0 се счита, че не са възникнали грешки при комуникацията.

Ще отбележим, че пораждащият полином $g(x)$ дели $x^{n_c} + 1$, където $n_c = min\{m | x^m \equiv 1 \bmod g(x)\}$. Числото n_c се нарича *ред на полинома* $g(x)$. Тогава ние разглеждаме двоичен цикличен $[n_c, n_c - p]$ код D с пораждащ полином $g(x)$ от степен p, където $n_c = x^m + 1$. На практика много често се използва $[n, n - p]$ подкод C на кода D получен чрез скъсяването на D в $n_c - n$ позиции. Всички тези кодове са CRC кодове.

Ще представим някои основни свойства на CRC кодовете.

- Тъй като пораждащият полином $g(x)$ дели $x^{n_c} - 1$, където $n_c = x^m - 1$ и

$m = min\{m|x^m \equiv 1 \ mod \ g(x)\}$, то $g(0) \neq 0$. Следователно $g(x)$ има поне два не нулеви коефициента: пред x^p и пред свободния член. Тогава нито една кодова дума не може да е равна на x^i. Следователно CRC кодът може да *открива всяка единична грешка* $e(x) = x^i, \ i = 0, 1, \ldots, n-1$.

- Ако CRC кодът се състои само от кодови думи с четно тегло, т.е. пораждащия полином е от вида $g(x) = g'(x)(x+1)$, то той *открива всички грешки с четно тегло.*

Група от грешки с дължина t е вектор, чиито не нулеви компоненти са разположени в t последователни позиции, първата и последната от които са различни от нула. Ако даден вектор е циклично еквивалентен на група от грешки с дължина t, казваме че той е *циклична група от грешки* с дължина t. Представените по-долу свойства относно поведението при откриване на групи от грешки на скъсените циклични кодове могат да бъдат намерени в много книги и статии. Ние ги представяме според [201].

Твърдение 5.0.1. *Скъсеният цикличен (CRC) код с пораждащ полином от степен* p *може да открива:*

- *всяка група от грешки с дължина до* p*;*
- $\left(1 - \frac{1}{2^{p-1}}\right)$ *от групите от грешки с дължина* $p+1$*;*
- $\left(1 - \frac{1}{2^{p}}\right)$ *от групите от грешки с дължина по-голяма от* $p+1$*.*

Твърдение 5.0.2. *Ако пораждащият полином* $g(x)$ *на скъсен цикличен* $[n, n-p]$ *код* C *е примитивен полином, то* C *може да открива всяка грешка с тегло 2.*

5.1 Поведение при откриване на грешки на стандартизирани CRC кодове

Изброените по-горе свойства правят CRC кодовете безспорни фаворити когато трябва да се избере код за откриване на грешки. Те се използват широко за откриване на грешки в цифровите комуникационни системи и при съхраняването на данни. По тази причина има стандартизирани десетки такива кодове с до 64 проверочни бита.

CRC кодовете са били обект на изследване в много работи (виж например [39], [40], [92], [93], [128], [132], [134], [136], [137], [155], [172], [176], [202]), но основно е изследвана трудността на определянето на стойността на вероятността за неоткрита грешка и са предложени методи за нейното оценяване.

Fujiwara, Kasami, Kitai и Lin [92] са разработили метод за оценка на вероятността за неоткрита грешка на скъсени кодове на Хеминг, които са CRC кодове с примитивен пораждащ полином. Аналогичен метод за скъсени кодове на Хеминг само с четни тегла е предложен от Wolf и Blakeney в [203]. Castagnoly, Bräuer и Herrmann [39] разширяват този метод, така че да могат да се разглеждат произволни скъсени циклични кодове. И трите работи пресмятат спектрите на дуалните на CRC кодовете и времевата сложност на разработените в тях методи е $O(2^p)$.

В нашите изследвания сме използвали Алгоритъм 2, за да пресметнем тегловните спектри на разглежданите кодове. Нека C да е $[n, n-p]$ скъсен цикличен код с пораждащ полином $g(x)$. За малки стойности на p, каквито са разглежданите от нас случаи, е по-ефективно вместо да се пресмята спектъра $\bar{A}$ на кода, да се пресметне този на неговия дуален $\bar{B}$, който съдържа 2^p кодови думи. Тогава времевата сложност на алгоритъма ще е както и на предложените по-горе алгоритми $O(2^p)$. За да приложим Алгоритъм 2, първо пресмятаме реда n_c на пораждащия полином $g(x)$ на кода C. След това пресмятаме и проверочния му полином $h(x) = (x^{n_c} - 1)/g(x)$, който е пораждащ за дуалния на C код $C^{\perp}$. Това се прави с програмата `GenerCRC` написана на C++. След това с програмата `QWD` се пресмята $\bar{B}$.

Спектърът на кода C се пресмята от този на дуалния му за линейно време като се използват трансформациите на MacWilliams, които дават връзка между спектъра на кода $\bar{A}$ и на неговия дуален $\bar{B}$. Те са валидни за кодове над произволно крайно поле, но тук ще представим техния вариант за двоични кодове, който сме използвали. Приемаме, че $\binom{n}{r} = 0$ за $r > n$.

Теорема 5.1.1. *[144] Нека C да е двоичен $[n, k]$ код. За $v = 0, \ldots, n$ е вярно следното*

$$\sum_{j=0}^{n} \binom{j}{n} A_j = 2^{k-v} \sum_{j=0}^{n} (-1)^j \binom{n-j}{n-v} B_j$$

и

$$\sum_{j=0}^{n} \binom{n-j}{n-v} B_j = 2^{v-k} \sum_{j=0}^{n} \binom{n-j}{v} A_j.$$

В случая, когато $v = 0$, *получаваме*

$$\sum_{j=0}^{n} A_j = 2^k B_0 = 2^k.$$

Програмата `MacWillTransform` написана на MAPLE реализира тези пресмятания.

В последните години все по-масово навлиза процесорното управление във всички уреди и устройства като се започне от бита и се стигне до приложения в автомобилостроенето, железопътния транспорт, при ескалаторите и управления на машини за различни отрасли на промишлеността. Всички тези разнообразни приложения са известни под името затворени системи за управление. Неотделима част от всяка една такава система е функцията за контрол на грешките на обменяните данни, която в болшинството от случаите се изпълнява от CRC код. Тъй като тези приложения имат специфични изисквания, е общоприето да се разработват индивидуални комуникационни протоколи за конкретното приложение или за конкретния производител. Това става възможно и заради заменянето на хардуерната имплементация на кодирането и декодирането със CRC кодове със софтуерна. Примери за комуникационни протоколи в затворени системи, в които се използват CRC кодове, за да осигурят целостта на данните, са Controller Area Network (CAN) [20], FlexRay [86] и TTP/C [190].

Широко разпространена практика при избора на CRC код за някое приложение е да се избере стандартизиран такъв, като се счита, че той ще е достатъчно добър. Оказва се обаче че това в голяма част от случаите не е така. Например някои от стандартизираните 16-битови CRC кодове имат поведение при контрол на грешки, което е по-лошо от това на други не стандартизирани 16-битови CRC кодове. Една от причините е, че в първите години са били използвани изключително хардуерни имплементации на кодирането и декодирането и използването на пораждащ полином с по-малко не нулеви коефициенти е водело до по-евтина имплементация. Друга причина е, че липсват достатъчно данни, позволяващи да се оцени поведението на стандартизираните кодове и да се сравни с това на други не стандартизирани такива.

В този раздел, на базата на някои от най-често използваните стандартизирани кодове, ще покажем, че подхода да се избира стандартизиран код или такъв с неразложим пораждащ полином, или пораждащ полином имащ множител $(x + 1)$ (т.е. код, който открива всички грешки с нечетно тегло) в много случаи е неудачен.

В Таблица 5.1.1 са дадени някои примери на CRC кодове, за които в [136] са пресметнати първите няколко елемента от тегловния им спектър. Това е направено за дължина 48, на която работят много от кодовете използвани в затворени

системи. Първият от тях е CCITT-16, който има само 4 не нулеви коефициента и това е било сериозно предимство при ранните хардуерни имплементации на този код. Както се вижда от таблицата, за дължина 48 той има минимално разстояние 4, докато друг стандартизиран полином, с един по-малко проверочни битове, CAN има минимално разстояние 6. Той има и по-малко кодови думи с тегло 6 в сравнение с CCITT-16, т.е. той няма да може да открива само 2191 от грешките с тегло 2, докато за CCITT-16 това ще е невъзможно за 2430 от тях. Друго интересно сравнение е между стандартизирания CRC-12 код и предложения в [136] код с 12 проверочни символа и пораждащ полином $x^{12}+x^8+x^7+x^6+x^5+x^4+1$. Той има минимално разстояние 5 и с по-вече от половината по-малко кодови думи с тегло 6 в сравнение с CRC-12. 8-битовият стандартизиран код CRC-8 също има минимално разстояние 4 на тази дължина. Добре е разработчиците на системи за запазване на интегритета на данните да използват него вместо друг стандартизиран 8-битов код DARK-8, който има минимално разстояние 2 на дължина 48. Минимално разстояние 4 на дължина 48 има и 7-битовият код с пораждащ полином $x^7+x^5+x^4+x^2+x+1$ предложен в [136], докато стандартизирания 7-битов код CRC-7 има минимално разстояние 3. Тези примери показват, че има полиноми с по-добро поведение от стандартизираните и съществува необходимост от указания кои полиноми на какви дължини са подходящи да бъдат използвани.

Таблица 5.1.1 Тегловни разпределения на CRC кодове на дължина 48

CRC полином	A_2	A_3	A_4	A_5	A_6
$x^{16}+x^{12}+x^5+1$ (CCITT-16)	0	0	84	0	2430
$x^{15}+x^{14}+x^{10}+x^8+x^7+x^4+x^3+1$ (CAN)	0	0	0	0	2191
$x^{12}+x^{11}+x^3+x^2+x+1$ (CRC-12)	0	0	575	0	28809
$x^{12}+x^8+x^7+x^6+x^5+x^4+1$	0	0	0	1452	13258
$x^8+x^5+x^4+x^3+1$ (DARK-8)	66	0	2039	13122	124248
$x^8+x^7+x^6+x^4+x^2+1$ (CRC-8)	0	0	2984	0	253084
x^7+x^3+1 (CRC-7)	0	216	2690	27051	226856
$x^7+x^5+x^4+x^2+x+1$	0	0	5589	0	451125

Ще представим и още някои конкретни примери в тази посока. Ще сравняваме вероятността за неоткрита грешка $P_{ud}(C,\varepsilon)$ на кодовете и ще казваме, че този, който има по-малка стойност на тази вероятност има по-добро поведение.

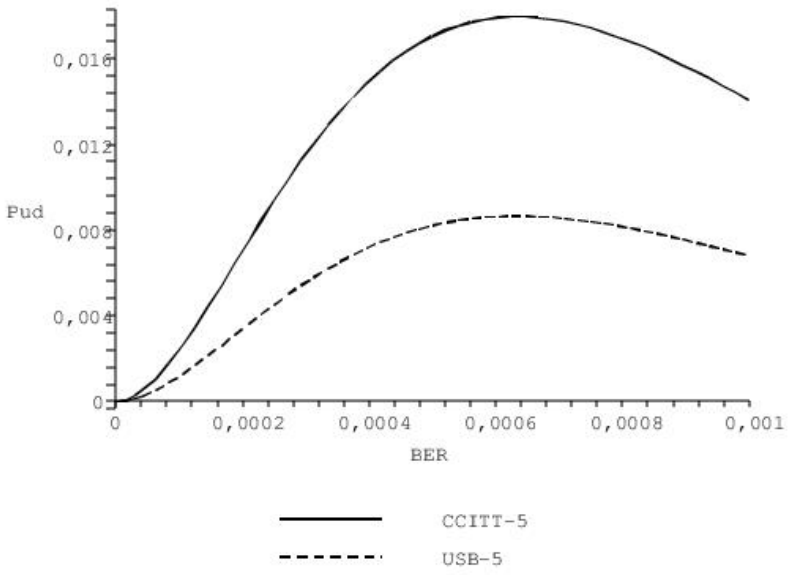
• 5-битови CRC кодове

5-битови CRC кодове се използват например за откриване на грешки в Universal Serial Bus (USB) спецификацията за осъществяване на комуникация между устройства и контролер (обикновено персонални компютри) и в ITU стандарта

за телекомуникационни системи. $x^5 + x^2 + 1$ е USB-5 полинома използван в USB спецификацията, а ITU G.704 използва 5-битовия CCITT-5 полином x^5+x^3+x+1.

USB-5 полинома се използва да защитава 11 информационни бита (т.е. работи на дължина 16) и е оптимален в този случай. Той има сравнително добро поведение и при по-големи дължини, но не е добър избор за по-малки дължини от 16. CCITT-5 е оптимален на дължина 15, има по-лошо поведение в сравнение с USB-5 за дължини от 16 до 31 и повече от два пъти по-лошо поведение при по-големи дължини.

CCITT-5 полинома се дели на $(x + 1)$, поради което може да открива всяка нечетна грешка. На дължина 3156 той е най-добър измежду 5-битовите полиноми, които се делят на $(x + 1)$. USB-5 не се дели на $(x + 1)$, но има по-добро поведение от CCITT-5 за всички дължини по-големи от 15. Според ITU G.704 стандарта, CCITT-5 се използва на дължина 3156, но както се вижда от фигура 5.1 поведението на USB-5 на тази дължина е 2.077 по-добро от това на CCITT-5.



Фигура 5.1: Сравнение на P_{ud} на CCITT-5 и USB-5 на дължина 3156.

На пръв поглед това поведение на CCITT-5 е странно, защото заради вида на пораждащия си полином той открива всяка нечетна грешка и би могло да се очаква, че ще има и достатъчно добро поведение. За стойност на вероятността за грешка на канала $\varepsilon = 10^{-6}$, каквато е тя в условията при които работи CCITT-5, голяма част от грешките имат тегло 1 и евентуално 2, докато по-тежки грешки

са малко вероятни. В Таблица 5.1.2 е посочен броя на кодовите думи с тегла до 5 в спектрите на двата кода. Както се вижда от нея, броят на кодовите думи с тегло 2 на USB-5 кода е два пъти по-малък отколкото на CCITT-5. В USB-5 има кодови думи с тегло 3, каквито в CCITT-5 няма, но броя на кодовите му думи с тегло 4 е също около 2 пъти по-малък от този на CCITT-5. Тъй като грешки с тегла по-големи от 2 са много малко вероятни, поведението на двата кода за тази вероятност за грешка на канала се определя основно от броя на кодовите думи с тегло 2.

Таблица 5.1.2. Брой на кодовите думи с тегло до 5 за CCITT-5 и USB-5 за дължина 3156.

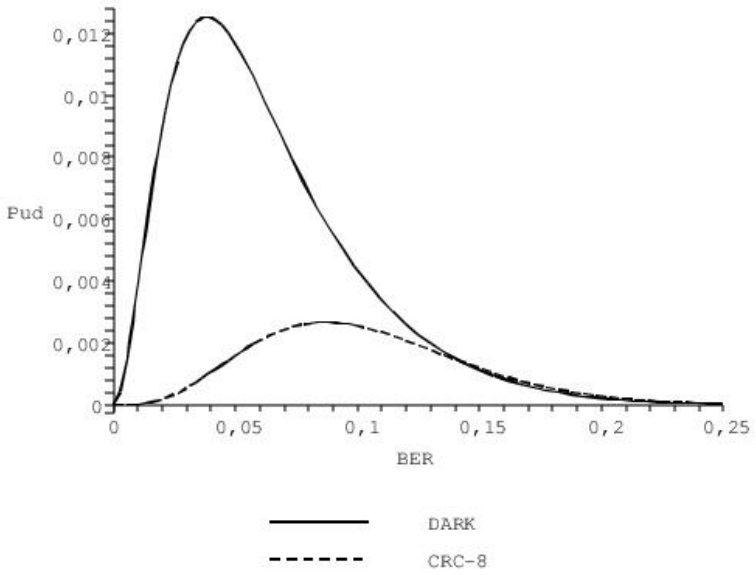
CRC Полином	**d**	**Брой кодови думи с тегла**			
		2	3	4	5
CCITT-5: $x^5 + x^3 + x + 1$	2	330435	0	257909068726	0
USB-5: $x^5 + x^2 + 1$	2	159075	163552409	128929654767	81278805135219

8, 16, и 32-битовите CRC кодове са най-широко използваните, защото тези дължини са кратни на дължината на думата в микроконтролерите. Нека да разгледаме поведението на някои такива стандартизирани кодове.

• 8-битови CRC кодове

Най-често използваният 8-битов полином е $x^8 + x^7 + x^6 + x^4 + x^2 + 1$ (CRC-8). Кодът, породен от този полином, има минимално разстояние 4 до дължина 85, но на дължини от 86 до 119 минималното му разстояние е 2, докато има други 8-битови CRC кодове, които имат минимално разстояние 4. Например друг стандартизиран полином ATM HEC $x^8 + x^2 + x + 1$ има минимално разстояние 4 до дължина 127.

Проблемът с избор на стандартизиран полином е, че той може да не е добър за почти всички кодови дължини. Стандартизираният DARK полином $x^8 + x^5 + x^4 + x^3 + 1$ поражда CRC код, който има максималното възможно минимално разстояние за дължина 16, но има минимално разстояние 2 за дължини над 18 и изключително лошо поведение по отношение на вероятността за неоткрита грешка. DARK се използва за откриване на грешки на дължини между 24 и 56, където поведението му е значително под най-доброто. Като илюстрация на това, на фигура 5.2 сме представили сравнение на вероятността за неоткрита грешка на DARK и на CRC-8 на дължина 56.



Фигура 5.2: Сравнение на P_{ud} на DARK и CRC-8 за дължина 56.

Два примера на реални протоколи, използващи CRC-8, са SMBus и XMODEM. SMBus е ниско скоростна комуникационна шина използвана при "умни"батерии и портативни електронни зарядни устройства. Версия 1.1 на SMBus използва CRC-8. Голямата част от обменяните съобщения са между 16 и 40 бита и са надеждно защитени от CRC-8, който на тези дължини има минимално разстояние 4. Има обаче и команда за пакетна комутация, която използва 35-байтови (280 бита) съобщения. На тази дължина минималното разстояние на CRC-8 е 2. Предложеният в [136] полином $x^8+x^6+x^3+x^2+1$ би бил по-добър избор за SMBus. CRC кодовете, породени от този полином, имат минимално разстояние $d = 3$ за дължини до 247 бита и по-добро поведение по отношение на вероятността за неоткрита грешка за дължини до 280.

XMODEM протокола се използва при предаване на пакети с дължина 128 байта (1024 бита) защитени от CRC-8. В този случай отново имаме по-добра алтернатива и това е използването на стандартизирания ATM HEC полином или предложения в [136] полином $x^8 + x^6 + x^3 + x^2 + 1$. И в двата случая кодовете породени от тези полиноми имат минимално разстояние $d = 2$ на тази дължина, но по-добро поведение в сравнение със CRC-8 по отношение на вероятността за неоткрита грешка.

• 16-битови CRC кодове

Няколко 16-битови CRC кода са стандартизирани:
$x^{16}+x^{14}+x^{12}+x^{11}+x^9+x^8+x^7+x^4+x+1$, IEC TC 57;
$x^{16}+x^{14}+x^{13}+x^{11}+x^{10}+x^9+x^8+x^6+x^5+x+1$, IEEE WG 77.1;
$x^{16}+x^{12}+x^5+1$ или $x^{16}+x^{14}+x+1$, CCITT X.25;
$x^{16}+x^{15}+x^2+1$, ANSI;
$x^{16}+x^{15}+x^{13}+x^7+x^4+x^2+x+1$ IBM-SDLC.

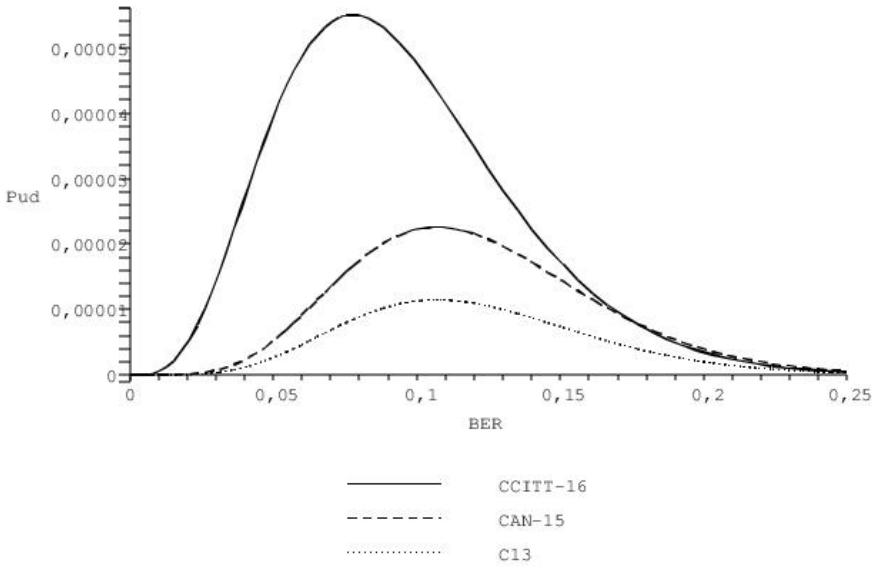
Измежду всички тези стандартизирани кодове, само IEEE WG 77.1 полинома поражда оптимални кодове и те са за дължини 254 и 255. Поведението на останалите стандартизирани кодове е далеко от оптималното за всички дължини между 18 и 1024.

В Таблица 5.3.1 са дадени бройките на кодовите думи с тегла от 1 до 6 за три 16-битови CRC кода.

Таблица 5.3.1 Брой на кодовите думи с тегла до 6 за CRC кодове, които защитават 48 информационни бита.

Пров. битове	CRC полином	d	Брой кодови думи с тегла				
			2	3	4	5	6
16	*CCITT-16* $x^{16}+x^{12}+x^5+1$	4	0	0	84	0	2430
16	$x^{16}+x^{15}+x^{12}+x^7+x^6+x^4+x^3+1$	6	0	0	0	0	2191
15	*CAN* $x^{15}+x^{14}+x^{10}+x^8+x^7+x^4+x^3+1$	6	0	0	0	0	4314

На първият ред от таблицата е даден широко използвания CCITT-16 полином, който има относително малък брой не нулеви коефициенти. CRC кодът с дължина 56, породен от този полином, открива всички грешки с тегло до 3. Минималното му разстояние е $d = 4$ и той не може да открива 84 от грешките с тегло 4. За сравнение (виж фиг. 5.3) кодът с пораждащ полином от същата степен $C_{13} = x^{16}+x^{15}+x^{12}+x^7+x^6+x^4+x^3+1$, който е даден на втория ред от таблицата, има минимално разстояние 6 на тази дължина. Следователно той открива всички грешки с тегло 4 и всички с тегло 5. На третият ред в таблицата е 15-битовия стандартизиран код CAN-15. На дължина 56 той също открива всички грешки с тегла 4 и 5 и има един проврочен бит по-малко в сравнение с предните два кода. Сравнение на вероятностите за неоткрита грешка на дължина 56 за тези три кода е представено на фигура 5.3. Най-добро поведение има не стандартизирания код, но и стандартизирания 15-битов код CAN-15 е в пъти по-добър от CCITT-16



Фигура 5.3: Сравнение на P_{ud} на CCITT-16, CAN-15 и C_{13} на дължина 56.

• 32-битови CRC кодове

IEEE 802.3 стандарта използва 32-битов CRC код със следния неразложим (но не примитивен) пораждащ полином $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$. Той има минимално разстояние $d \geq 8$ за дължини до 91, $d = 7$ за дължини до 171, $d = 6$ за дължини до 268, $d = 5$ за дължини до 2974, $d = 4$ за дължини до 91607 и $d = 3$ за дължини поне до 128K бита. В [39] е предложен полином, който има $d = 6$ за дължини почти до 32K бита. По-късно Koopman [137] предлага 32-битовия полином $x^{32}+x^{30}+x^{29}+x^{28}+x^{26}+x^{20}+x^{19}+x^{17}+x^{16}+x^{15}+x^{11}+x^{10}+x^7+x^6+x^4+x^2+x+1$, който има като множител $x+1$. CRC кодът, породен от този полином, има $d = 6$ за дължини до 16360 и $d = 4$ за дължини до 114663, което е по-вече от 9 пъти по-добро в сравнение с IEEE 802.3 стандарта.

В заключение ще отбележим, че всички посочени до тук примери показват, че обичайната практика да се избира стандартизиран CRC полином с презумпцията, че той е добър, е грешна, защото някои от тях имат поведение много далеч от оптималното. Още по-вече, че дори да е известно, че един полином е добър (има максималното възможно минимално разстояние), обикновено няма информация за какви дължини той е такъв и данни, които да позволят сравняването на поведе-

нието с това на други публикувани или стандартизирани полиноми. Следователно пълното изследване на всички полиноми от дадена степен би помогнало на разработчиците при избора на най-подходящия полином за конкретното приложение. Много полезно ще бъде и разработването на насоки, според които да се избира подходящ полином.

Решения на тези проблеми са предложени в следващите три раздела, в които са изследвани CRC кодове с до 16 проверочни бита. Предложени са добри 8 и 16-битови CRC кодове като са посочени дължините, за които те са оптимални. За CRC кодовете с до 10 проверочни символа са получени всички необходими количествени данни, които да позволят избора на оптимален за конкретно приложение код и е разработена процедура, която на базата на пресметнатите данни, да прави този избор.

Резултатите от този раздел са съвместни с Faiza Sallam и са публикувани в [219, 220, 222]. Представени са на семинара "Математика, Информатика и Компютърни Науки"във Велико Търново, България, 2006 г. и на International Workshop on Algebraic and Combinatorial Coding Theory в Звенигород, Русия, 2006 г.

5.2 CRC кодове с 8 проверочни бита

При стандартизираната технология за предаване на цифрови данни ATM (Asynchronous Transfer Mode) [160] с 8-битов CRC код се защитават хедърите, които носят информацията необходима за насочване на данните през мрежата. Хедърите са с дължина 40 бита, като последните 8 от тях са проверочните битове и се наричат Header Error Control (HEC). ATM стандартът използва CRC код със следния пораждащ полином $g(x) = x^8 + x^2 + x + 1$.

В този раздел разглеждаме двоични полиноми от 8 степен, които са подходящи да бъдат използвани като пораждащи полиноми на CRC кодове. Сравняваме минималните им разстояния, свойството им да са подходящи за контрол на грешки и вероятността им за неоткрита грешка със стандартизирания ATM полином. Пресметнати са също и радиусите на покритие и тегловните разпределения на лидерите на съседни класове на всички изследвани кодове.

Нека $g(x)$ да е пораждащ полином от осма степен, т.е. $g(x) = \sum_{i=0}^{8} a_i x^i$. Двоичната последователност, която ще се предава през комуникационния канал, може да има дължина от 9 (когато се предава 1 информационен бит) до реда на пораждащия полином. Моделът на канала, през който се извършва комуникацията, е двоичен симетричен канал (binary symmetric channel (BSC)) и поведението на

CRC кода ще се определя от стойността на вероятността за неоткрита грешка $P_{ue}(C, \varepsilon)$. При неголеми стойности на вероятността за грешка ε на двоичния симетричен канал много по-често се появяват грешки с малко тегло в сравнение с такива с голямо. В такъв случай е удачно да се използват CRC кодове с максимално минимално разстояние.

Тъй като цикличните кодове и всички техни скъсявания, които изследваме, имат ко-размерност 8, е по-удобно да работим вместо със самите кодове с техните дулни. Тогава подходяща за нашето изследване е следната формула от [202] за определяне на вероятността за неоткрита грешка

$$P_{ue}(C, \varepsilon) = 2^{-p}[1 + \sum_{i=d^{\perp}}^{n} B_i(1-2\varepsilon)^i] - (1-\varepsilon)^n, \tag{5.2.1}$$

където $d^{\perp}$ е минималното разстояние на дуалния код, а B_i са елементите на спектъра му.

В това изследване разглеждаме само полиноми $g(x)$, които пораждат не еквивалентни кодове, имащи поне два не нулеви коефициента (пред най-високата степен и свободния член) и от ред по-голям от 40. Последното условие е свързано с това, че ще правим сравнение с ATM стандарта, а той работи на дължина 40. Както ще бъде показано в раздел 5.4, кодовете, породени от полиноми с ред по-малък от 40, когато се използват на тази дължина, имат минимално разстояние 2. Затова ние изследваме следните четири класа полиноми.

1. Всички неразложими полиноми от 8-ма степен.

2. Всички неразложими полиноми от 7-ма степен умножени по $x + 1$. Полиномът използван в ATM стандарта принадлежи към този клас.

3. Всички неразложими полиноми от 6-та степен умножени по $(x + 1)^2$.

4. Всички неразложими полиноми от 6-та степен умножени по $x^2 + x + 1$.

Кодовете, генерирани от тези полиноми, са тествани за всички дължини $n \in [10, min(127, order(g))]$.

Първо пресмятаме тегловните разпределения на дуалните кодове (които имат 2^8 кодови думи) на всеки от изброените по-горе кодове. За целта използваме програмата `QWD`. След това, като използваме формулата (5.2.1), изследваме поведението на вероятността за неоткрита грешка $P_{ue}(C, \varepsilon)$. Това правим като сравняваме стойностите на функцията за фиксирани стойности на вероятността за грешка на канала. По-точно за всички стойности на $\varepsilon = e^{\frac{-i}{10}}$, където $31 \leq i \leq 300$. По този начин можем да определим само в кои интервали функцията $P_{ue}(C, \varepsilon)$ не е монотонно растяща. Също така бяха тествани дискретните достатъчни условия от теореми 1.2.16 и 1.2.17 един код да е добър и подходящ. Бяха пресметнати радиу-

сите на покритие и тегловните разпределения на лидерите на съседни класове на всички кодове. Тъй като дуалните кодове на всички изследвани кодове имат само 2^8 кодови думи, пресмятанията бяха направени по метод 1 с програмата `CovRadM1`. Като използвахме вече известните тегловни разпределения на лидерите на съседни класове, пресметнахме и стойностите на функцията P_{corr} за всички изследвани кодове за дължина 40.

Получените в това изследване резултати са представени в таблици 5.2.1 до 5.2.4 в приложение Г. В таблиците са включени само полиноми, които пораждат не еквивалентни кодове. В първата колона са дадени коефициентите на полиномите като се започва от коефициента пред най-високата степен. След това в таблица 5.2.1 е даден редът на всеки от полиномите. Редът на всички полиноми от таблица 5.2.2 е 127, а на тези от таблици 5.2.3 и 5.2.4 е 63 и по тази причина не е изрично посочван в отделна колона. В 3 последователни колони са дадени дължините на кодовете, за които вероятността за неоткрита грешка е монотонно растяща функция и са удовлетворени достатъчните условия кодът да е добър и подходящ. В последните две колони са минималните разстояния и радиусите на покритие на кодовете за съответните дължини. Те са представени по следния начин: първо минималното разстояние и след това за кои дължини кодовете имат това минимално разстояние. Например: 4,10..28 означава, че кодовете с дължини от 10 до 28 имат минимално разстояние (или съответно радиус на покритие) 4. Примитивните полиноми в таблица 5.2.1 са отбелязани със звезда. Всички неразложими множители от степени 6 и 7 в таблици 5.2.2 до 5.2.4 са примитивни. Всички кодове с радиус на покритие 2 в таблица 5.2.1 са квази-съвършени.

За всички разглеждани кодове характеристиките им монотонност на функцията на вероятността за неоткрита грешка, добър, подходящ, минимално разстояние и радиус на покритие са представени в таблиците, така че всеки два кода могат да бъдат лесно сравнявани директно. Това, което липсва в таблиците, са стойностите на вероятността за неоткрита грешка $P_{ue}(C, \varepsilon)$. Тези стойности, за всички изследвани полиноми, са пресметнати по формулата (5.2.1) за породените от тях кодовете на дължина 40 (дължината, на която работи ATM стандарта). Кодът C_1 от таблица 5.2.1 с пораждащ полином $g_1(x) = x^8+x^7+x^4+x^3+x^2+x+1$ има минимална стойност на вероятността за неоткрита грешка за $\varepsilon \in [0, 1/2]$ сред кодовете с неразложим пораждащ полином от 8-ма степен. Оказва се, че тази стойност е и най-добрата за дължина $n = 40$ за всички изследвани кодове за стойности на $\varepsilon > 0.022266$.

Кодът C_2 с пораждащ полином $g_2(x) = x^8 + x^5 + x^3 + x^2 + x + 1$ има минимална стойност на вероятността за неоткрита грешка за $\varepsilon \in [0, 1/2]$ сред кодовете

породени от полиномите от таблица 5.2.2. Този код има също най-малък брой кодови думи с минимално тегло. Пораждащият му полином обаче има по-голям брой не нулеви коефициенти в сравнение с ATM стандарта. Измежду полиномите с по-малък брой не нулеви коефициенти в таблица 5.2.2 има няколко, за които кодовете породени от тях имат по-малки стойности на вероятността за неоткрита грешка в сравнение с ATM стандарта. Най-добрият измежду тях е $g_3(x) = x^8 + x^5 + x^4 + 1$.

Кодът породен от полинома $g_4(x) = x^8 + x^7 + x^5 + x^4 + x^3 + x^2 + x + 1$ има минимална стойност на вероятността за неоткрита грешка за $\varepsilon \in [0, 1/2]$ сред кодовете породени от полиномите от таблици 5.2.3 и 5.2.4. Тази стойност обаче е по-голяма в сравнение с тези на най-добрите кодове от таблици 5.2.1 и 5.2.2.

Можем да обобщим тези резултати за CRC кодове на дължина 40 така.

• За вероятност за грешка на канала $\varepsilon \in [0, 0.022266]$, най-добър е кодът породен от полинома $g_2(x) = x^8 + x^5 + x^3 + x^2 + x + 1$.

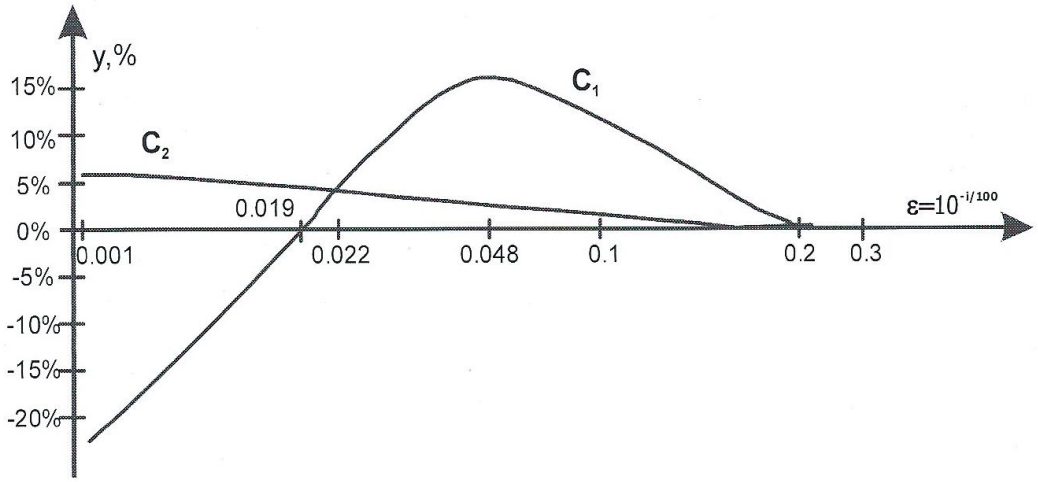
• За вероятност за грешка на канала $\varepsilon \in [0.022266, 0.5]$, най-добър е кодът породен от полинома $g_1(x) = x^8 + x^7 + x^4 + x^3 + x^2 + x + 1$.

Сега нека да сравним най-добрите по отношение на $P_{ue}(C, \varepsilon)$ кодове на дължина 40 - C_1 и C_2, и ATM кода. Тъй като тези разлики са по-забележими при малки стойности на ε, ние направихме сравнение за $\varepsilon = 10^{-i/100}$, където $31 \leq i \leq 200$. Това сравнение е представено на фигура 5.4. Вместо $P_{ue}(C, \varepsilon)$ е представено, изразеното в проценти, съотношение

$$y = \frac{P_{ue}(C_{ATM,\varepsilon})}{P_{ue}(C_j, \varepsilon)} - 1,$$

където с C_j е означен кодът с пораждащ полином $g_i(x)$ за $i = 1, 2$. Кодът от ATM стандарта е поставен на абсцисната ос. Вижда се, че при стойности на вероятността за неоткрита грешка между 0.019331 и 0.2, поведението на C_1 е с до 18% по-добро от това на ATM кода, като максимумът се достига при $\varepsilon = 0.048$. Поведението на кодът C_2 е с до 5% по-добро от това на ATM кода и има същото минимално разстояние $d = 4$ и радиус на покритие $R = 3$ като него. C_1 има с едно по-малко минимално разстояние $d = 3$ от тях и затова има значително по лошото поведение (до -25% за $\varepsilon = 0.01$) в сравнение с ATM кода за по-слабо зашумени канали.

Протоколът, с който работи ATM стандарта, включва режим на работа само за откриване на грешки, когато каналът е зашумен и за откриване и коригиране на единична грешка (минималното му разстояние е 4), когато не е. ATM кодът има радиус на покритие 3 и такъв или с едно по-малък имат всички останали изследвани кодове. Сравнение беше направено и по отношение на поведението спрямо вероятността за коректно предаване на всички изследвани кодове на дължина 40. За няколко кода беше получена най-ниска стойност на тази вероятност и всички



Фигура 5.4: Сравнение на най-добрите CRC-8 кодове и ATM стандарта.

те са от таблица 5.2.1. Кодът C_1 е измежду тях, но разликите в стойностите на тази функция за всички изследвани кодове са твърде незначителни и затова няма да бъдат коментирани. Можем да приемем, че на тази дължина, всички кодове имат еднакво поведение по отношение на вероятността за коректно предаване.

Резултатите от този раздел са съвместни със Стефан Додунеков и Петър Казаков и са публикувани в [208].

5.3 CRC кодове с 16 проверочни бита

В този раздел, подобно на предишния, се разглеждат всички полиноми от 16-та степен, които са подходящи да бъдат използвани като пораждащи полиноми на CRC кодове. Изследват се породените от тях кодове с дължини до 1024 и се сравняват с 16-битови стандартизирани CRC кодове. Има няколко стандартизирани такива кодове с 16 проверочни символа, а именно IEC TC57, CCITT X.25, ANSI, IBM-SDLC, IEEE WG77.1.

Скъсени циклични кодове с 16 проверочни бита са изследвани в няколко работи. Fujiwara, Kasami, Kitai и Lin [92] изследват поведението при откриване на грешки на CRC кодове получени при скъсяването на разширени кодове на Хеминг с минимално разстояние 4 и стандартизирани от International Telegraph and

Telephone Consultative Committee като CCITT X.25. Те показват, че скъсяването на кодовете променя тяхното поведение при откриване на грешки. Ако кодовете се скъсят твърде много, те вече не удовлетворяват условието да бъдат подходящи за откриване на грешки.

Сравнение на някои съществуващи стандартизирани CRC кодове е направено в [202]. Изследвани са 3 стандартизирани CRC кода - CRC-12, ANSI и CCITT X.25. Те са сравнени с други полиноми от същия вид. Показано е, че пораждащи полиноми с нисък ред пораждат кодове с лоши характеристики по отношение на вероятността за неоткрита грешка.

CRC кодове с 16 проверочни бита са изследвани от Castagnoly, Ganz и Graber в [40]. Те са търсили полиноми, които пораждат кодове с максимално минимално разстояние за дадена дължина. За породените от намерените полиноми кодове, са пресметнали стойностите на функцията $P_{ue}(C, \varepsilon)$ и са ги сравнили с тези на стандартизираните кодове. В резултат, те предлагат нови по-добри полиноми.

Funk [93] разглежда БЧХ кодове породени от полиноми с четни или нечетни тегла и имащи максимално минимално разстояние за фиксирани дължина и размерност. Той определя и представя в таблица кодове, които за фиксиран брой проверочни символи, имат минимална стойност на $P_{ue}(C, \varepsilon)$ при съответните скъсявания. Прави също и сравнение между тези кодове и стандартизирани такива.

В нашето изследване разглеждаме всички полиноми от 16-та степен, имащи поне два не нулеви коефициента, т.е. всички неразложими полиноми от 16-та степен всички неразложими полиноми от 15-та степен умножени по $(x + 1)$, всички неразложими полиноми от 14-та степен умножени по $(x^2 + 1)$ или по $(x^2 + x + 1)$ и т.н. като пропускаме реципрочните. Кодовете, породени от тези полиноми, са тествани за всички дължини от 18 до $\min[1024, n_c]$. Първо определяме тегловното разпределение на дуалните им кодове $\bar{B}$ с програмата `QWD`, а след това с `MacWillTransform` пресмятаме минималните разстояния на самите кодове. След това тестваме поведението на функцията $P_{ue}(C, \varepsilon)$ и определяме интервалите, в които тя не е монотонно растяща. Правим това не само за кодовете на пълната им дължина, но и за всички техни скъсявания, за седем фиксирани стойности на вероятността за грешка на канала $\varepsilon = 10^{-5}$, $\varepsilon = 10^{-4}$, $\varepsilon = 10^{-3}$, $\varepsilon = 10^{-2.5}$, $\varepsilon = 10^{-2}$, $\varepsilon = 10^{-1.5}$ и $\varepsilon = 10^{-1}$. Проверяваме и достатъчните условия кодовете да са добри и подходящи за откриване на грешки.

Предимството на този подход е, че изследваме всички кодове с дължини от 18 до $\min[1024, n_c]$ и така определяме най-добрите по отношение на вероятността за неоткрита грешка кодове за осемте фиксирани стойности на ε. Освен това имаме информация за реда на полинома, за минималното разстояние на породените от

него кодове и дали те удовлетворяват достатъчните условия за добър и подходящ за откриване на грешки код.

Резултатите от изследването за вероятност за грешка на канала $\varepsilon = 10^{-3}$ са представени в таблица 5.3.1 в приложението. Заради компактност на записа, коефициентите на полиномите са дадени в 16-тично представяне. Например полиномът $x^{16} + x^{15} + x^{13} + x^5 + 1$ е записан като 1A021. Посочен е редът на полинома, дължините, за които съответния код е най-добър (има минимална стойност на вероятността за неоткрита грешка), удовлетворява теста за монотонност на функцията $P_{ue}(C, \varepsilon)$, изпълнява достатъчните условия да е добър и подходящ за откриване на грешки. В последната колона е дадено минималното разстояние на кодовете по следния начин. Първо е посочено минималното разстояние, а след това и дължините, за които съответния код има такова минимално разстояние. На първите 10 реда от таблицата са 5-те стандартизирани полинома и 5-те най-добри полинома представени в работата на Castagnoly, Ganz и Graber [40]. Останалата част от таблицата включва опредените от нас най-добри полиноми.

Най-добрите полиноми за $\varepsilon = 10^{-5}$ съвпадат с тези от таблица 5.3.1 само с 5 изключения. Тези изключения са следните:

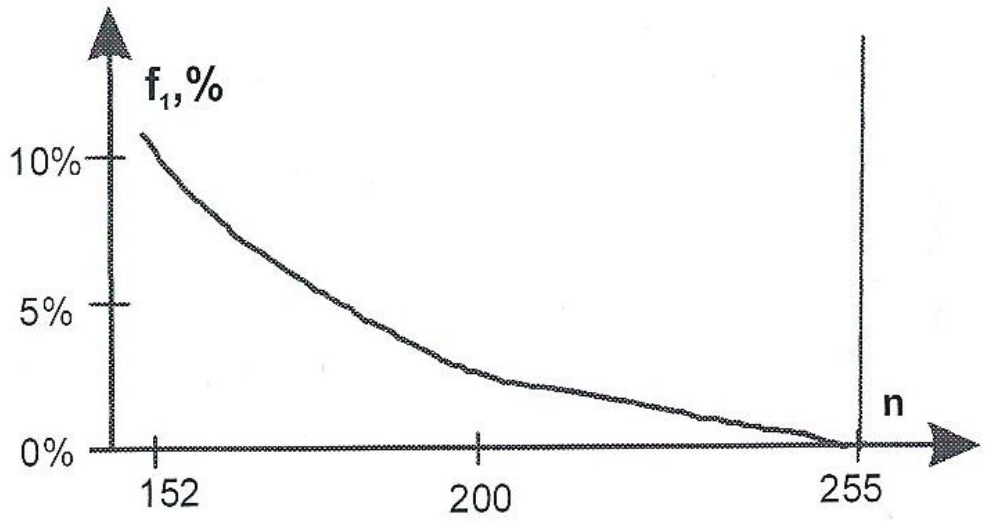
- за $n = 879 - 883$ най-добри са кодовете с пораждащ полином 10595;
- за $n = 720$ най-добър е кодът с пораждащ полином 1E667;
- за $n = 705$ и $n = 706$ най-добри са кодовете с пораждащ полином 1FC9F;
- за $n = 680$ и $n = 681$ най-добри са кодовете с пораждащ полином 19E91;
- за $n = 361$ и $n = 595$ най-добри са кодовете с пораждащ полином 1941F.

Ще направим сравнение на стандартизирани полиноми с тези с най-добро поведение. На фиг. 5.5 е представено поведението на кода породен от стандартизирания полином IEEE WG77.1 (16F63) и е сравнено с това на най-добрите за дължини от 152 до 255 кодове при $\varepsilon = 10^{-1}$. Тъй като за различните дължини имаме различни най-добри кодове, сме ги означили с BEST(n). На абсцисната ос са дължините на кодовете, а на ординатната - стойностите в проценти на

$$f_1(n) = \frac{P_{ue}(C_{16F63}, 0.001)}{P_{ue}(C_{BEST(n)}, 0.001)} - 1.$$

От графиката се вижда, че поведението на IEEE WG77.1 е близо да оптималното, с разлика до 5% от него, за дължини между $n = 181$ и $n = 255$. За дължини извън този интервал, поведението му е значително по-лошо.

Кодовете породени от стандартизирания полином IEC TC57 (15B93) на дължини от 94 до 128 имат стойности на вероятността за неоткрита грешка близка до оптималните (разлики от 1% до 5%) и достигат оптималната стойност на $P_{ue}(C, \varepsilon)$ на дължина $n = 19$. Поведението на останалите стандартизирани полиноми е не-



Фигура 5.5: Сравнение на най-добрите CRC-16 кодове и IEEE стандарта.

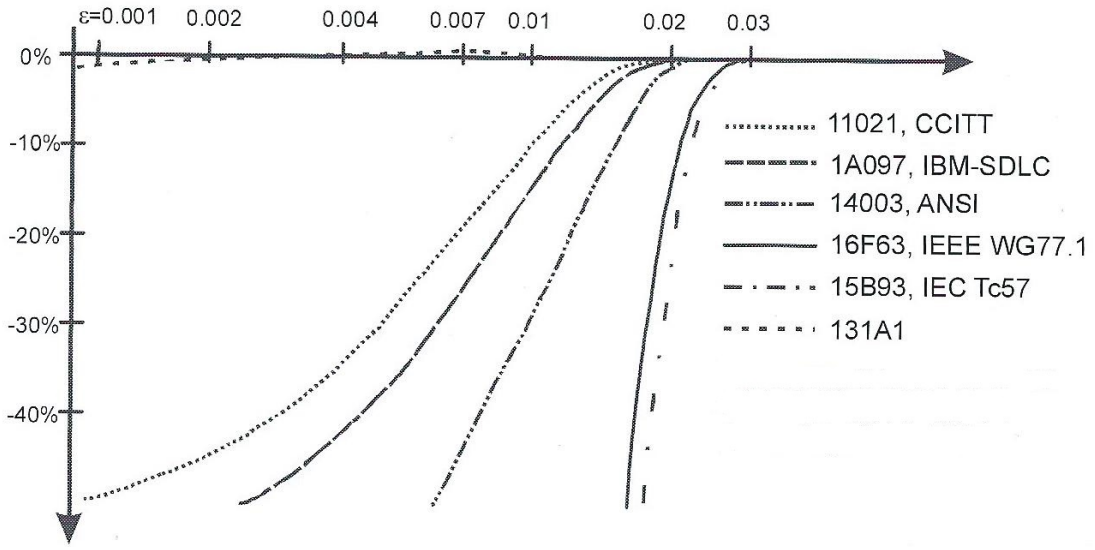
задоволително за всички изследвани дължини от $n = 18$ до $n = 1024$.

Кодовете породени от полиномите предложени в [40] имат най-доброто минимално разстояние и са добри и подходящи в някои интервали. Броят на кодовите им думи с минимално тегло обаче е по-голям от този на най-добрите намерени в това изследване. Следователно, особено за малки стойности на ε, те ще имат по-лошо поведение по отношение на вероятността за неоткрита грешка.

На фиг. 5.6 е представено сравнение между стандартизирани полиноми и полиномите 1941F и 131A1 за $n = 529$. На абсцисната ос са представени стойностите на ε, а на ординатната - стойностите на функцията f_2 в проценти, дефинирана по следния начин:

$$f_{2,i}(\varepsilon) = \frac{P_{ue}(C_{1941F}, \varepsilon)}{P_{ue}(C_i, \varepsilon)} - 1.$$

Със C_i са означени стандартизираните полиноми или полиномът 131A1. Най-добрият код за дължина $n = 529$ е породеният от полинома 1941F. Неговата графика съвпада с абсцисната ос. Вижда се, че единствено кодът породен от полинома 131A1, има малко по-добро поведение (с до 0.17%) в интервала $\varepsilon[0.003, 0.03]$ от кода с пораждащ полином 1941F, докато поведението на другите кодове е много



Фигура 5.6: Сравнение на най-добрите CRC-16 кодове и стандартизираните 16-битови кодове за $n = 529$.

по-лошо.

За слабо зашумени двоични симетрични канали, които много по-често генерират вектор-грешки с малко тегло отколкото с голямо, е удобно да се използват кодове с максимално минимално разстояние. Оказва се, че най-добрите кодове, посочени в таблица 5.3.1 и в допълнението към нея за $\varepsilon = 10^{-5}$, са най-добри и в това отношение, защото имат максималното възможно минимално разстояние измежду двоичните кодове с техните дължини и размерности. Има само едно изключение за дължина $n = 152$.

Резултатите от този раздел са съвместни със Стефан Додунеков и Петър Казаков и са публикувани в [210]. Предварителни резултати са представени на International Workshop on Algebraic and Combinatorial Coding Theory в Псков, Русия през 1998 г. и на семинар в университета в Улм, Германия.

5.4 CRC кодове с до 10 проверочни бита

Направените в предишните раздели изследвания, както и тези на други автори, безспорно показват, че някои приложения използват CRC кодове, които имат

поведение много по-лошо от най-добрите известни такива кодове. Всички тези изследвания обаче предлагат най-добри полиноми за фиксирани дължини и вероятности за грешка на канала, но не дават всички необходими данни, които да позволят на разработчиците на комуникационни системи сами да правят сравнение между полиномите и да избират най-подходящия за тяхното конкретно приложение. Например в предишния раздел са изследвани CRC полиноми от 16 степен за дължини от 18 до 1024 и са предложени най-добри полиноми за две конкретни стойности на вероятността за грешка на канала: за зашумен ($\varepsilon = 10^{-3}$) и слабо зашумен ($\varepsilon = 10^{-5}$) двоичен симетричен канал. Значително по-разширено изследване е направено в [136] където са тествани всички CRC полиноми от степени между 3 и 16 и породените от тях кодове с дължини до 2048. Посочени са най-добрите за затворени системи, при които вероятността за грешка на канала е $\varepsilon = 10^{-6}$. И в двете работи резултатите са получени за конкретни вероятности за грешки на канала и могат да не са валидни за други стойности на ε. Някои от приложенията могат също така да се нуждаят от код, който има добро поведение не само за определена дължина, а за интервал от кодови дължини или, заради особеностите на имплементацията на кодирането и декодирането, теглото на полинома може да е от значение. Освен това в [136] е пропусната една важна характеристика на CRC полиномите. Там не се взема под внимание редът на полинома, което, както ще покажем в този раздел, ни дава информация за минималното разстояние на кодовете породени от CRC полиномите. По-точно, ако кодът се използва на дължини по-големи от реда на пораждащия го полином, то неговото минимално разстояние е 2.

Традиционно CRC кодовете се използват предимно за откриване на грешки, но някои приложения използват комбинирани схеми с едновременно откриване и коригирани на грешки. В този случай трябва да може да се оцени поведението на кода и при коригиране на грешки, за да може да се избере най-добрия такъв. Следователно пълното изследване на всички CRC кодове с фиксиран брой проверочни символи и предоставянето на цялата информация от това изследване ще позволи на разработчиците да изберат най-ефективния за тяхното конкретно приложение код.

За да предоставим цялата нужна информация за сравняване на поведението при контрол на грешки на CRC кодове с до 10 проверочни символа, ние изследваме всички полиноми до 10 степен, които са подходящи да бъдат пораждащи полиноми на такива кодове. Първо правим списък на всички такива полиноми като изключваме от него реципрочните, тъй като пораждат еквивалентни кодове, и определяме реда на тези полиноми. След това пресмятаме минималните разс-

тояния, броя на кодовите думи с минимално тегло, тегловното разпределение на дуалните кодове и на лидерите на съседните класове на кодовете, породени от полиномите от списъка, и на всички техни скъсявания. Наличието на тези данни позволява пресмятането и сравняването на поведението на кодовете при откриване и коригиране на грешки да става за линейно време. За да избегнем сравняването на всички полиноми от зададена степен при избора на най-подходящ полином, предлагаме лесна процедура от 4 стъпки, която позволява да се сравняват само няколко измежду най-добрите.

Ще припомним, че двоичният полином трябва да има поне два не нулеви коефициента (пред x^p и 1), за да бъде подходящ за пораждащ полином на CRC код с p проверочни символа. Следователно имаме 2^{p-1} полинома за всяко p. Някои от тях са реципрочни, което означава, че пораждат еквивалентни кодове и всички скъсявания на тези кодове са също еквивалентни кодове. Затова ние не разглеждаме и двата полинома, а само единия от тях. Така формираме списъка от всички полиноми, които са подходящи да бъдат пораждащи полиноми на CRC кодове с до 10 проверочни символа. Този списък съдържа също разлагането на множители на тези полиноми и техния ред. Разлагането и пресмятането на реда на полиномите правим с програма написана на Maple. Тя се базира на следните два резултата от теорията на крайните полета.

Теорема 5.4.1 ([140], Corollary 3.4). *Ако $g \in GF(q)[x]$ е неразложим полином над $GF(q)$ от степен p, то $ord(g)$ дели $q^p - 1$.*

Теорема 5.4.2 ([140], Theorem 3.11). *Нека $GF(q)$ да е поле на Галоа с q елемента, q да е степен на простото число p и нека $f \in GF(q)[x]$ да е полином от положителна степен като $f(0) \neq 0$. Нека $f = af_1^{b_1} f_2^{b_2} \dots f_k^{b_k}$ и $f_1, \dots, f_k$ да са различни нормализирани неразложими полиноми в $GF(q)[x]$, където $a \in F_q$, а $b_1, \dots, b_k \in N$ е каноничното разлагане на f в $GF(q)[x]$. Тогава $ord(f) = ep^t$, където e е най-малкото общо кратно на $ord(f_1), \dots, ord(f_k)$ и t е най-малкото цяло число, такова че $p^t \geq max(b_1, \dots, b_k)$.*

Тъй като пълният списък с полиноми е твърде дълъг (съдържа 550 полинома), като пример представяме в таблицата по-долу списъка с полиноми със степен до 5-та. (Пълният списък, както и всички данни пресметнати в този раздел, могат да бъдат намерени на `http://www.math.bas.bg/~tsonka`). За компактност на записа, множителите на полиномите са представени чрез номерата, под които те се намират в списъка. Знакът $'$ след някои от номерата означава, че множителят е реципрочен на полинома под този номер в списъка.

Таблица 5.4.1. CRC полиноми от степен до 5-та.

N	Име	Полином	Множители	Ред
1	CRC-1	$x+1$		1
2		x^2+x+1		3
3		x^3+x^2+1		7
4		x^3+x^2+x+1	1^3	4
5	CCITT-4	x^4+x+1		15
6		x^4+x^2+x+1	1.3	7
7		x^4+x^3+x+1	$1^2.2$	6
8		x^4+x^2+1	2^2	6
9		$x^4+x^3+x^2+x+1$		5
10		$x^5+x^4+x^3+x^2+1$		31
11		$x^5+x^4+x^2+x+1$		31
12	USB-5	x^5+x^2+1		31
13		x^5+x+1	2.3	21
14	CCITT-5	x^5+x^3+x+1	$1.5'$	15
15		$x^5+x^4+x^3+1$	$1^2.3$	14
16		$x^5+x^3+x^2+1$	$1^3.2$	12
17		x^5+x^4+x+1	1^5	8
18		$x^5+x^4+x^3+x^2+x+1$	1.2^2	6

За кодовете от списъка и за всевъзможните им скъсявания бяха пресметнати с програмата `QWD` тегловните спектри на дуалните им кодове. След това с програмата `MacWillTransform` са определени тегловните спектри за всеки от разглежданите кодове, а с програмата `CovRadM1` са пресметнати тегловните разпределения на лидерите на съседните им класове.

Резултатите за CRC кодовете от таблица 5.4.1 са представени в таблица 5.4.2 в приложението. В първата колона е посочен поредният номер според таблица 5.4.1. Тъй като са дадени резултатите за кодовете на пълната им дължина (редът на пораждащия им полином) и всички техни скъсявания, във втората колона са посочени съответните дължини на кодовете. В останалите колони са дадени минималното разстояние, броя на кодовите думи с минимално тегло, тегловното разпределение на дуалните им кодове $\bar{B}$ и на лидерите на съседните им класове $\bar{\alpha}$. $\bar{B}$ и $\bar{\alpha}$ са представени като двойка от числа, в която първото е теглото на кодовата дума или на лидера на съседен клас, а второто - броят им.

За разлика от предишните изследвания върху поведението на скъсените циклични кодове, тук не предлагаме най-добри полиноми, а процедура за избор на най-подходящия за конкретното приложение код в зависимост от дължината на която той ще работи и вероятността за грешка на комуникационния канал. Про-

цедурата се състои от четири стъпки. Тя е проста и лесна за изпълнение, тъй като всички необходими пресмятания могат да се направят за линейно време благодарение на пресметнатите по-горе данни. Ще отбележим, че тази процедура ще избере същите най-добри полиноми както в [39, 40, 92, 93, 128, 136, 155, 172, 202, 208, 210], ако бъде приложена за съответните кодови дължини и вероятности за грешка на канала.

Съществена роля в процедурата за избор играе редът на полинома. Тази характеристика на пораждащия полином не винаги се взема под внимание при изследванията, но дава важна информация за минималното разстояние на CRC кода.

Твърдение 5.4.3. *Всеки CRC код с пораждащ полином, който не се дели на x, има минимално разстояние поне 3.*

Доказателство. Нека да приемем, че $c(x) = q(x)g(x)$ е кодова дума на CRC код с пораждащ полином $g(x)$ от ред n_c. Тъй като $g(x)$ има поне два не нулеви коефициента, $c(x)$ също има поне два не нулеви коефициента. Нека сега да приемем, че $c(x) = x^i + x^j$, $i > j$. Тогава $c(x) = q(x)g(x) = x^j(x^{i-j} + 1)$ и $q'(x)g(x) = (x^{i-j} + 1)$. Това е невъзможно, тъй като $i - j < n_c$ и най-малкото число m, за което $x^m + 1 \equiv 0 \bmod g(x)$ е $ord\ g(x) = n_c$. ◇

Стъпките на процедурата са следните.

1. Първото ограничение, което всяко приложение налага, е броят на проверочните символи, които трябва да бъдат добавени към данните, за да се осъществи контрола на грешките. Този брой определя степента p на полиномите, които трябва да бъдат разглеждани.

2. Измежду всички полиноми с фиксирана степен p избираме само тези от ред по-голям или равен на максималната дължина n, на която кодът ще бъде използван. Ако редовете на всички полиноми са по-малки от n, избираме тези с максимален ред.

3. От множеството полиноми, формирано на предишната стъпка, разглеждаме само тези чието минимално разстояние е максимално. Ако има твърде много такива полиноми, избираме само тези, които имат най-малък брой кодови думи с минимално тегло.

4. Пресмятаме и сравняваме стойностите на $P_{ue}(C, \varepsilon)$ за избраните кодове за конкретната вероятност за грешка на канала ε, при която ще работи кода, и избираме този с най-малка стойност на $P_{ue}(C, \varepsilon)$. В допълнение, ако кодът ще бъде използван и за коригиране на грешки, сравняваме и вероятностите им за коректно предаване P_{corr} и избираме този с най-голяма стойност на тази вероятност.

Много от комуникационните протоколи не налагат ограничения за дължината на съобщенията, които се защитават, т.е. кодирането със CRC кода се прилага към съобщения с дължина многократно по-голяма от $n_c - p$, където n_c е редът на CRC полинома. Такъв пример е CCITT-5 (полиномът с номер 14 в таблица 5.4.1), където съобщение с дължина 3151 бита се защитава от 5 проверочни бита (ITU-T Recommendation G.704). Ясно е, че няма полином от 5-та степен и от ред 3156, каквато е необходимата дължина на кода в този случай. Решението е да се използва повторение на оригиналния код и ние ще наричаме този код *повторен* CRC код. Кодиращите и декодиращите процедури за тези кодове са същите както и при CRC кодовете, но тяхното минимално разстояние е 2.

Нека съобщение $i(x)$ с дължина $m > n_c - p$ да се защитава от p бита със CRC код. Кодирането на това съобщение води до получаването на кодова дума $c(x) = x^p i(x) + r(x)$ с дължина $n = m + p$, където $r(x)$ е остатъка по модул $g(x)$. Тъй като $x^{n_c} + 1 \equiv 0 \bmod g(x)$, то

$$x^a \equiv x^b \bmod g(x),$$

където x^b е остатъкът на a по модул n_c и

$$x^c + x^d \equiv 0 \bmod g(x),$$

където $c - d \equiv 0 \bmod n_c$. Следователно някои съобщения с тегло едно или две водят до получаването на кодови думи с тегло две, което значително влошава поведението при контрол на грешки на кода. Броят на кодовите думи с тегло две може да се пресметне лесно.

Твърдение 5.4.4. *Нека $n = qn_c + r$, $0 \leq r \leq n_c$. Броят на кодовите думи с тегло две се определя от равенството*

$$A_2 = \frac{q(n - n_c + r)}{2}.$$

Доказателство. Кодовата дума може да бъде разделена на q блока от по n_c бита и един блок от r бита, т.е.

$$[n_c][n_c] \ldots [n_c][r].$$

Не нулевите битове на всяка кодова дума с тегло две са разположени по следния начин:

а) Един в последния блок от r бита и един в позиция със същия номер в един от първите q блока. Има точно qr такива кодови думи.

b) В позиции с един и същи номер в два от първите q блока. В този случай, има точно $\binom{q}{2} n_c$ такива кодови думи.

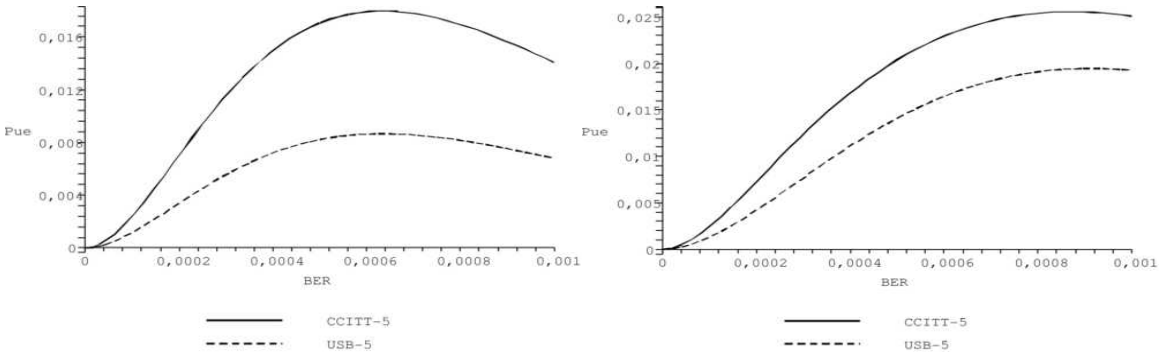
Следователно

$$A_2 = qr + \frac{q(q-1)}{2} n_c = \frac{q(n - n_c + r)}{2}.$$

◇

Забележка. Броят на кодовите думи с тегло две в повторен CRC код зависи от дължината на кода и от реда на пораждащия полином. Колкото е по-голям редът на пораждащия полином, толкова е по-малък броя на кодовите думи с тегло две. Следователно може да се очаква, че по-добро поведение като повторени CRC кодове ще имат CRC кодовете с по-висок ред.

Ще отбележим, че когато искаме да сравним $P_{ue}(C, \varepsilon)$ на повторени CRC кодове можем да получим достатъчно точна информация, ако пресметнем само първите няколко събираеми в израза за P_{ue}. Двете графики на фигура 5.7 са потвърждение на нашето твърдение. На фигурата са показани графиките на вероятността за неоткрита грешка на два стандартизирани 5-битови CRC кода - CCITT-5 и USB-5 на дължина 3156. На лявата графика е използвано само първото събираемо от израза за $P_{ue}(C, \varepsilon)$, за да бъде оценена вероятността за неоткрита грешка, т.е. $A_2\varepsilon^2(1-\varepsilon)^{n-2}$. На дясната графика $P_{ue}(C, \varepsilon)$ е пресметнато като са използвани първите няколко не нулеви елемента от тегловния спектър на кодовете. По-точно, A_2, A_4 за CCITT-5 и A_2, A_3, A_4 за USB-5. Тези бройки са вземати от [136]. Ясно е, че макар че сравнението на дясната графика е по-прецизно, тенденцията за безспорно по-доброто поведение на USB-5 в сравнение с CCITT-5 се вижда достатъчно добре и на лявата.



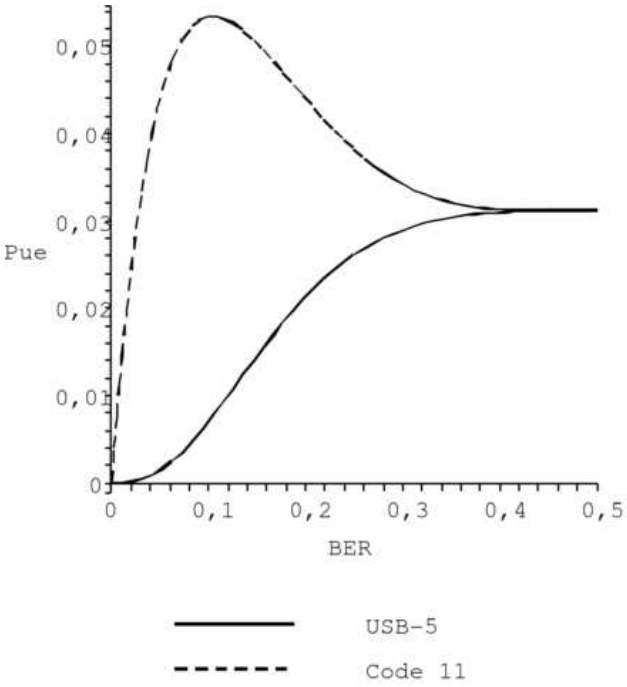
Фигура 5.7: P_{ue} на CCITT-5 и USB-5 пресметната чрез A_2 и $\{A_2, A_3, A_4\}$.

За да оценим по-точно вероятността за неоткрита грешка на дълги повторени кодове, за които пресмятането на пълния тегловен спектър е изчислително невъзможно дори и за дуалните им кодове, бихме могли да използваме вместо целия спектър само първите няколко елемента от него. Така ще получим приемлива оценка за вероятността за неоткрита грешка, тъй като първите няколко

събираеми в израза за $P_{ue}(C, \varepsilon)$ в голяма степен определят неговата стойност. Следователно ние трябва да пресметнем броя на кодовите думи с тегла 3, 4, 5, ... За да направим това, трябва да генерираме всички $\binom{n}{3}$, $\binom{n}{4}$, $\binom{n}{5}$, ... двоични n-мерни вектори (n е дължината на кода) с тегла 3, 4, 5, ... съответно. Като делим всеки от тези вектори на пораждащия полином (или еквивалентно, ако умножим вектора по проверочната матрица), проверяваме кои от тях са кодови думи. Това е сравнително бавна процедура за кодове с големи дължини, но е възможно да бъде изпълнена за реално време, ако я приложим само за няколко от първите тегла. Можем да ускорим тези пресмятания, ако вземем под внимание, че ако цикличния код на пълната си дължина n_c има само кодови думи с четно тегло, то и всички негови повторени кодове ще имат само кодови думи с четно тегло. Следователно, за такива кодове, трябва да се тестват само векторите с четно тегло.

Пример. Като илюстрация на това как можем да използваме резултатите получени в тази работа, ще представим сравнение (виж фиг. 5.8) между стандартизиран код и не стандартизирани такива. $x^5 + x^2 + 1$ (ред 12 от таблица 5.4.1) е стандартизираният пораждащ полином USB-5 и кода, породен от него, се използва за откриване на грешки в Universal Serial Bus (USB), за да защитава данни с дължина 11 бита. Следователно дължината на кода трябва да е $n = 16$. От таблица 5.4.1 се вижда, че първите четири полинома от 5-та степен са подходящи (техните редове са по-големи от 16), за да бъдат използвани като пораждащи полиноми на такъв код. Всички те имат минимално разстояние 3 (виж таблица 5.4.2) на тази дължина и затова сравняваме броя на кодовите им думи с минимално тегло. Кодовете с номера 10 и 12 имат $A_3 = 19$, а тези с номера 11 и 13 - $A_3 = 20$. В допълнение, на дължина 16, кодовете 10,12 и 11,13 имат еднакви тегловни спектри на техните дуални кодове. Тогава можем да сравняваме само кодовете с номера 11 и 12 (USB-5). Резултатът от сравнението е показан на фигура 5.8 и от нея се вижда, че поведението на USB-5 е значително по-добро за голяма част от стойностите на вероятността за грешка на канала. Следователно USB-5 е оптимален когато се използва да защитава 11 бита с данни, т.е. дължината на кода е $n = 16$. Кодът с номер 10 има същия тегловен спектър на дуалния код и спектър на лидерите на съседни класове за $n = 16$ като USB-5. Следователно тези два кода имат напълно идентично поведение при контрол на грешки за дължина 16.

Изготвеният списък с полиноми до 10-та степен, които могат да са пораждащи полиноми на CRC кодове, показва значимостта на пълното изследване на всички характеристики на CRC кодовете и важността на реда на CRC полинома за поведението на породения от него CRC код. От друга страна определянето на всички необходими характеристики на CRC кодовете е изключително трудоемко.



Фигура 5.8: Сравнение на P_{ue} на USB-5 и на код номер 11 от таблицата.

Дори само класификацията на полиномите и определянето на редовете им става все по-сложно с увеличаване на степента им. Броят на полиномите, които трябва да се изследват за всяка следваща степен също се увеличава, както и необходимото време за определяне на минималното разстояние и тегловните спектри на кодовете. Следователно разширяването на представения тук списък с полиноми от по-висока степен определено представлява изследователски интерес.

Резултатите в този раздел са самостоятелни и са публикувани в [224].

Глава 6

Някои характеристики на шумозащитни кодове свързани с техните възможности за контрол на грешки

6.1 Нютонов радиус на покритие

Известно е, че за един линеен $[n, k, d]$ код C всички грешки с тегло $t \leq (d-1)/2$ са коригируеми по единствен начин. Съществуват обаче грешки с тегло по-голяво от t, които също са коригируеми по единствен начин. Това са случите когато съседните класове с тегла по-големи от t имат единствен лидер. Така възникват няколко важни въпроса. Кои са грешките, които са коригируеми по единствен начин? Колко са те за предварително зададено тегло? Какво е най-голямото тегло на грешка, която може да се коригира по единствен начин? В настоящия раздел отговаряме на тези въпроси за двоичните циклични кодове с дължина до 31, двоичните кодове с максимално минимално разстояние с дължини до 33 и троичните циклични и негациклични кодове с дължини до 22.

В [106] е въведен параметърът *Нютонов радиус на покритие* за двоичен линеен код като най-голямото тегло на коригируем по единствен начин вектор-грешка. Граници и точни стойности за Нютоновия радиус на някои класове линейни кодове са дадени в [95], [107] и [133]. По-късно тези граници са обобщени за линейни кодове над произволна азбука в [96].

Нека C да е двоичен линеен $[n,k]$ код. За вектора $x \in F_2^n$, *фалшив съсед* е не нулева кодова дума, която се намира на същото разстояние от x както и нулевата кодова дума.

Векторът-грешка e е коригируем по единствен начин, тогава и само тогава когато няма лъжливи съседи. Грешка, която не може да бъде коригирана по единствен начин, ще наричаме *некоригируема грешка*. Ще отбележим, че e е коригируема по единствен начин (или за краткост ще я наричаме коригируема), тогава и само тогава когато е единствен лидер на съседен клас.

Нека да означим с $E(C)$ множеството от всички некоригируеми грешки и с

$$E_t(C) = \{x \in E(C) | wt(x) = t\}$$

множеството от некоригируеми грешки с тегло t. Нека още $\bar{E}(C)$ да е множеството от всички коригируеми грешки и $\bar{E}_t(C)$ да е множеството от всички коригируеми грешки с тегло t. Да означим също

$$N_t = N_t(C) = |E_t(C)|,$$

$$\bar{N}_t = \bar{N}_t(C) = |\bar{E}_t(C)| = \binom{n}{t} - N_t(C).$$

Тогава дефинираме Нютоновия радиус $\nu(C)$ по следния начин. Ако

$$\bar{E}(C) = \{x | d(x,c) > wt(x),\ \forall c \in C \backslash \{0\}\},$$

то

$$\nu(C) = \max\{wt(x) | x \in \bar{E}(C)\}.$$

От дефиницията непосредствено получаваме следната връзка между Нютоновия радиус, радиуса на сферичната опаковка и радиуса на покритие на кода.

Лема 6.1.1. *[107] Нека C да е $[n,k,d]$ код. Тогава*

$$\left\lfloor \frac{d-1}{2} \right\rfloor \leq \nu(C) \leq r(C).$$

В частност, ако кодът е съвършен, то

$$\left\lfloor \frac{d-1}{2} \right\rfloor = \nu(C) = r(C).$$

Може да се очаква, че разликата между Нютоновия радиус и радиуса на покритие няма да е твърде голяма. Оказва се обаче, че това не винаги е така. В [107] е показано, че за всяко k съществуват еквидистантни $[n,k]$ кодове, такива че $R(C) - \nu(C) = k$. И това е най-лошия случай, както се вижда от следващата теорема.

Теорема 6.1.2. *[107] За всеки $[n,k]$ код C имаме*

$$0 \leq R(C) - \nu(C) \leq k.$$

Проблемът за Нютоновия радиус на кодове над произволни азбуки без нулеви координати (т.е. за всяка позиция на кода съществува поне една кодова дума, която има не нулева стойност в тази позиция) е разгледан в [96]. Там са изведени следните горни граници за радиуса на покритие и за Нютоновия радиус на такива кодове.

Лема 6.1.3. *Нека C да е $[n,k]_q$ код без нулеви координати. Тогава*

$$R(C) \leq n - \frac{n}{q}$$

и

$$\nu(C) < n - \frac{n}{q}.$$

В [95] е показано, че $R(C) + \nu(C) \leq n - k$. По-късно в [96] Gabidulin и Klöve обобщават тази граница за не двоични кодове.

Теорема 6.1.4. *Нека C да е $[n,k]_q$ код без нулеви координати. Тогава*

$$\nu(C) \leq \frac{(q-1)n - k - (q-2) - R(C)}{q-1}.$$

Равенство в тази граница се достига от двоични кодове с $k \leq 3$ и от два пъти повторени двоични кодове.

Известни са и точните стойности на Нютоновия радиус на покритие за някои класове кодове. В [107] е определен Нютоновият радиус на покритие на двоичните еквидистантни кодове и на кодовете на Reed-Muller от първи ред, а в [96] на кодовете с размерност 1 над произволна азбука.

В този раздел ние определяме точните стойности на Нютоновия радиус на покритие на някои двоични и троични кодове. Разглеждаме точно тези кодове, а именно двоични циклични кодове, двоични кодове с максимално минимално разстояние, както и троични циклични и негациклични кодове, защото те са интересни и от гледна точка на приложенията. Цикличните кодове са измежду най-често използваните в практиката заради ефективните алгоритми за кодиране и декодиране, които имат. Кодовете с максимално минимално разстояние могат да откриват и коригират еднозначно най-голям брой грешки измежду всички кодове със същите дължини и размерности.

Класификации на разглежданите от нас кодове както и пресмятане на точните стойности на радиусите им на покритие са направени в [10], [75], [117] и [211]. Тези класификации бяха използвани като основа за изследването. За двоичните кодове с размерности 2 и 3 имаме равенство в границата от теорема 6.1.4 и тези

стойности се определят веднага, тъй като са ни известни радиусите им на покритие. За останалите кодове първо бяха пресметнати представените по-горе горни и долни граници за Нютоновия им радиус. В случаите, когато някои от тези граници съвпадат, се определя еднозначно и Нютоновия им радиус на покритие. За останалите кодове беше използван компютър, за да се пресметне точната му стойност. За целта беше използвана модификация на Метод 2 за пресмятане на радиуса на покритие на код. Тъй като всички съседни класове с тегло до t имат единствени лидери на съседни класове, бяха разглеждани само векторите с тегла от $t+1$ до горната граница за Нютоновия радиус. Така определяме броя на всички единствени лидери на съседни класове с тегла по-големи от t. Програмната реализация е направена на C++ с програмата `NewtonRad`.

Предимство на това изследване е, че не само определяме точните стойности на Нютоновия радиус на покритие на разглежданите кодове, но посочваме и броя на единствените лидери за всички тегла на съседни класове между $t+1$ и $\nu(C)$. Пълните резултати от изследването могат да се намерят на
http://www.math.bas.bg/∼tsonka/nutable.pdf.

За илюстрация на резултатите, в таблица 6.1.1 сме представили само двоичните циклични кодове, които имат над 90% единствени лидери на съседни класове. В таблицата са посочени стойността на Нютоновия радиус на покритие на съответния код, както и броя и теглото на единствените лидери. В скоби е даден процентът на тези единствени лидери по отношение на общия брой съседни класове с такова тегло.

Таблица 6.1.1. Двоични циклични кодове.

No	[n,k,d]	ν	Единствени лидери
1.	[15,3,5]	6	425, w=3 (93%); 1050, w=4 (77%); 1500, w=5 (50%), 1000, w=6 (20%)
2.	[21,9,3]	4	189, w=2 (90%); 945, w=3 (71%); 1890, w=4 (32%)
3.	[21,7,3]	7	189, w=2 (90%); 945, w=3 (71%); 2835, w=4 (47%); 5103, w =5 (25%); 5103, w=6 (9%); 2187, w=7 (2%)
4.	[21,6,7]	6	5880, w=4 (98%); 13020, w=5 (64%); 1680, w=6 (3%)
5.	[21,3,7]	9	5880, w=4 (98%); 18816, w=5 (92%); 43806, w=6 (81%); 72030, wt=7 (62%); 77175, w=8 (38%); 42875, w=9 (15%)
6.	[21,2,14]	9	105987, w=7 (91%); 123480, w=8 (61%); 42875, wt=9 (15%)
7.	[25,5,5]	10	2250, w=3 (98%); 11625, w=4 (92%); 43125, w=5 (81%); 116250, w=6 (66%); 225000, w=7 (47%); 300000, w=8 (28%) ; 250000, w=9 (12%); 100000, w=10 (3%)
8.	[25,4,10]	10	50625, w=5 (95%); 141250, w=6 (80%); 256250, w=7 (53%); 300000, w=8 (28%); 250000 w=9 (12%); 100000, w=10 (3%)
9.	[27,9,3]	9	324, w=2 (92%); 2268, w=3 (78%); 10206, w=4 (58%); 30618 , wt=5 (38%); 61236, w=6 (21%); 59049, w=8 (3%); 19683, w=9 (0,42%)

10.	[27,7,6]	9	2754, w=3 (94%); 14094, w=4 (80%); 45198, w=5 (56%); 94041, w=6 (32%); 118098, w=7 (13%); 78732, w=8 (4%); 19683, w=9 (0,4%)
11.	[27,6,6]	9	2754, w=3 (94%); 14094, w=4 (80%); 47385, w=5 (59%); 101331, w=6 (34%); 124659, w=7 (14%); 78732, w=8 (4%); 19683, wt=9 (0,4%)
12.	[27,3,9]	12	80352, w=5 (99,5%); 288954, w=6 (98%); 825552, w=7 (93%) ; 1871100, w=8 (84%); 3307500, w=9 (71%); 4381776, w=10 (52 %); 4000752, w=11 (31%); 2000376, w=12 (12%)
13.	[27,2,18]	12	4540968, w=9 (97%); 6994512, w=10 (83%); 6667920, w=11 (51%); 2000376, w=12 (12%)
14.	[31,16,5]	5	4185, w=3 (93%); 13485, w=4 (43%); 186, w=5 (0,1%)
15.	[31,11,10]	7	162099, w=5 (95%); 394599, w=6 (54%); 22475, w=7 (1%)
16.	[31,10,10]	7	162099, w=5 (95%); 451019, w=6 (61%); 103695, w=7 (4%)
17.	[31,6,15]	10	7291200, w=8 (92%); 11179840, w=9 (55%); 1833216, w=10 (4 %)
18.	[31,5,16]	11	7490220, w=8 (95%); 14069660, w=9 (70%); 9634304, w=10 (22%); 317688, w=11 (0,4%)

Резултатите в този раздел са получени самостоятелно, представени са на International Workshop on Algebraic and Combinatorial Coding Theory в Царское село, Русия и са публикувани в сборника с доклади на конференцията [213].

6.2 LUEP кодове

Голямата част от блоковите кодове разглеждани в литературата имат свойството, че техните възможности за коригиране на грешки се описват в контекста на коректното получаване на цялото съобщение. Тези кодове могат да се използват успешно в случаите когато всички позиции на съобщението се нуждаят от еднакво ниво на защита от грешки. Съществуват обаче много приложения при които някои от позициите на съобщението са по-важни от други. Например комуникационна система при която информационно съобщение се състои от няколко части и всяка част има различна степен на значимост. Така по-значимите части изискват по-сигурна защита от грешки при комуникацията. В този случай се нуждаем от код, който предлага различна степен на защита на различните части на предаваното съобщение. Друга ситуация, където това се налага, е едновременното предаване на съобщения от различни източници обединени в една кодова дума, като различните източници имат различни изисквания за нивото на защита на данните. Например наблюдател, който изпраща по сателитен канал резултатите от различни експерименти. Някои от тях могат да са по-важни от други и следователно да се нуждаят от по-високо ниво на защита от грешки. Вместо да се

използват различни кодове за различните източници би било по-ефективно да се използва един и така да имаме един кодер и един декодер.

Линейни кодове, които защитават някои от позициите на съобщението срещу по-голям брой грешки отколкото други негови позиции, се наричат линейни кодове с неравномерна защита от грешки (linear unequal error protection (LUEP)). Концепцията за неравномерна защита на информационните символи в кодовата дума е въведена от Masnick и Wolf [149]. Те разглеждат неравномерна защита на един бит от кодовата дума и описват някои свойства, намират граници и дават примери на систематични кодове с неравномерна защита на символите. В работите [26, 27, 28, 77, 78, 101, 131, 142, 146, 149] са представени подходи за конструиране на пораждащите или проверочните матрици на кодове с неравномерна защита на символите.

В този раздел ще разглеждаме кодове с неравномерна защита на един от информационните символи като следваме дефиницията на Dunning и Robbins [77]. Те въвеждат така наречения отделящ вектор, за да оценят възможностите за коригиране на грешки на линеен код с неравномерна защита от грешки (LUEP код). Основният проблем е да се намери такъв код с неравномерна защита от грешки с фиксирана размерност и отделящ вектор, че дължината му да е минимална, а следователно скоростта му да е максимална. Ние използваме компютърно търсене и определяме възможностите за неравномерна защита на информационен символ на троичните циклични и негациклични кодове с дължини до 26, имащи минимално разстояние поне 3.

Нека C да е линеен $[n, k]$ код над $GF(q)$ и G да е пораждащата му матрица. Тогава, ако $x = (x_1, \ldots, x_k)$ е информационната последователност, която ще се кодира, то $v(x) = xG = (v_1, \ldots, v_n)$ е кодова дума от кода C.

Dunning и Robbins [77] дават следната дефиниция за отделящ вектор.

Дефиниция 6.2.1. *За линеен $[n, k]$ код C над $GF(q)$, отделящ вектор $s(G) = (s(G)_1, \ldots, s(G)_k)$ с дължина k по отношение на пораждащата матрица G на C се дефинира като*

$$s(G)_i = min\{wt(mG)|m \in GF(q)^k, m_i \neq 0\}, i = 1, \ldots, k.$$

Това означава, че за всяко $\alpha, \beta \in GF(q), \alpha \neq \beta$, множествата $\{mG|m \in GF(q)^k$, $m_i = \alpha\}$ и $\{mG|m \in GF(q)^k, m_i = \beta\}$ са на разстояние $s(G)_i$ от $(i = 1, \ldots, k)$.

Ако линейният код C има пораждаща матрица G, такава че компонентите на отделящият вектор $\mathbf{s}(G)$ не са еднакви, то тогава кодът C се нарича *линеен код с неравномерна защита от грешки* (LUEP код).

Ние използваме модификация на тази дефиниция за отделящия вектор както

това е направено в [142]. Нека C да е $[n, k_1 + k_2 + \ldots + k_m]$ код за пространството на съобщенията $M_1 \times M_2 \times \ldots \times M_m$, където $M_i = \{0, 1, \ldots q-1\}^{k_i}$, за $i = 1, 2, \ldots m$. Всяко съобщение x спрямо кода C може да бъде записано като $x = (x_1, x_2, \ldots, x_m)$, където x_i е k_i-орка и е компонент на съобщението x спрямо пространството на съобщенията $M_i, i = 1, 2, \ldots, m$. Тогава отделящият вектор $s = (s_1, s_2, \ldots, s_m)$ се дефинира от

$$s_i = min\{d[v(x), v(x')] : x = (x_1, \ldots, x_i, \ldots, x_m), x' = (x'_1, \ldots, x'_i, \ldots, x'_m),$$
$$v(x), v(x') \in C, x_l, x'_l \in M_l, l = 1, 2, \ldots, m, x_i \neq x'_i\}.$$

Ясно е, че минималното разстояние на C е

$$d_{min} = min\{s_1, s_2, \ldots, s_m\}.$$

Нека $v(x)$ да е изпратената кодова дума, а r да е получения вектор. Може да се покаже, че x_i може да бъде коректно декодиран от r, ако $d[v(x), r] \leq \lfloor (s_i - 1)/2 \rfloor$ за $i = 1, 2, \ldots, m$. Следователно отделящият вектор s определя възможностите за неравномерна защита на символите на кода C.

Линейният код C може да бъде представен като директна сума на линейните кодове $C_1, C_2, \ldots, C_m$, където C_i е $[n, k_i]$ код за пространството на съобщенията $M_i, i = 1, 2, \ldots m$. Следващата теорема показва един начин за изследване на възможностите за неравномерна защита на символите на кода C.

Теорема 6.2.2. *[142] Нека минималното разстояние на кода C_1 е d_1 и $d_1 < d_2 < \ldots < d_m$. Ако минималното тегло на кодова дума от $C \backslash C_1 \oplus C_2 \oplus \ldots \oplus C_{i-1}$ е $d_i, i = 2, 3, \ldots, m$, то отделящия вектор s на C за пространството на съобщенията $M_1 \times M_2 \times \ldots \times M_m$ е $s = (d_1, d_2, \ldots, d_m)$.*

Трябва да се отбележи, че отделящият вектор или, еквивалентно, възможността за неравномерна защита на кода зависи от използвания метод за кодиране. Следователно отделящият вектор за линеен код зависи от избора на пораждащата му матрица.

Да приемем, че C е q-ичен цикличен код. В [149] е показано, че всички циклични кодове в систематичен вид нямат възможности за неравномерна защита на информационните символи. Следователно всички циклични кодове с неравномерна защита на символите не са в систематичен вид. Нека $h(X) = q_1(X)q_2(X) \ldots q_l(X)$ да е проверочен полином на C, където $q_1(X), q_2(X), \ldots, q_l(X)$ са q-ични неразложими полиноми. Цикличният код V_i с проверочен полином $q_i(X)$ е подкод на C. Кодът C е директна сума на $V_1, V_2, \ldots, V_l$. Следващата теорема представя метод за кодиране, при който се постига максималната възможна степен на неравномерна защита на символите на цикличен код.

Теорема 6.2.3. *[142] Матрицата* $G = [G_1^T G_2^T \dots G_l^T]^T$ *е пораждаща матрица за* C*, която осигурява максимална неравномерна защита на символите на* C*, като* G_i *е пораждаща матрица на минималния идеал* $V_i, i = 1, 2, \dots, l$*.*

Директната сума на $V_{i_1}, V_{i_2}, \dots, V_{i_r}$ е цикличен подкод на C с проверочен полином $q_{i_1}(X)q_{i_2}(X)\dots q_{i_r}(X)$ и пораждаща матрица $G = [G_{i_1}^T G_{i_2}^T \dots G_{i_r}^T]^T$. От теорема 6.2.3 се получава следния резултат.

Следствие 6.2.4. *[142] Да допуснем, че* C *има максимална неравномерна защита на символите представена от отделящия вектор* $s = (d_1, d_2, \dots, d_m)$ *за пространството на съобщенията* $M_1 \times M_2 \times \dots \times M_m$*, където* $d_1 < d_2 < \dots < d_m$*. Тогава пораждащата матрица на* C *е от вида* $G = [G_1^T G_2^T \dots G_l^T]^T$*, където* G_j *е пораждаща матрица на цикличния подкод* C_j *на* C *с проверочен полином* $h_j(X)$ *като* $deg(h_j(X)) = k_j, j = 1, 2, \dots m$*. В допълнение пораждащият полином на* C *е* $h(X) = h_1(X)h_2(X)\dots h_m(X)$ *и* $C = C_1 \oplus C_2 \oplus \dots \oplus C_m$*.*

На базата на тези твърдения предлагаме следния алгоритъм за определяне на възможностите за неравномерна защита на символите на линейния код C.

Първа стъпка. Намираме всички кодови думи с минимално тегло d_1 в C и подкода C_1 породен от тези кодови думи. Ако $C = C_1$, то C не е код с неравномерна защита на символите. В противен случай преминаваме към стъпка 2.

Втора стъпка. Намираме всички кодови думи с минимално тегло d_2 в $C \backslash C_1$ и подкода C_2', породен от тези кодови думи. $C_1 + C_2'$ е еквивалентен на $C_1 \oplus C_2$, където C_2 е подкод на C_2'. Ако $C = C_1 \oplus C_2$, то C е код с неравномерна защита на символите с две нива на защита и отделящ вектор $s = (d_1, d_2)$, където $d_1 < d_2$. Иначе преминаваме към следващата стъпка.

i-та стъпка. Намираме всички кодови думи с минимално тегло d_i в $C \backslash C_1 \oplus C_2 \oplus \dots \oplus C_{i-1}$ и подкода C_i', породен от тези кодови думи. Тогава $C_1 \oplus C_2 \oplus \dots \oplus C_{i-1} + C_i' = C_1 \oplus C_2 \oplus \dots \oplus C_i$ и C е код с неравномерна защита на символите с i нива на защита и отделящ вектор $s = (d_1, d_2, \dots, d_i)$, където $d_1 < d_2 < \dots < d_i$. В противен случай C е код с неравномерна защита на символите с ниво на защита поне $(i + 1)$ и трябва да повторим тази процедура за $i + 1$.

С помощта на програма, която реализира този алгоритъм бяха, пресметнати отделящите вектори на троичните циклични и нега циклични кодове с дължини до 26 и минимално разстояние поне 3. При пресмятането на възможностите на тези кодове за неравномерна защита на символите съществено използвахме факта, че те са циклични и според теорема 6.2.3 и следствие 6.2.4 трябваше да тестваме само подкодовете, които са директна сума на минимални идеали. Резултатите са представени в таблиците по-долу като са включени само кодовете имащи неравномерна

защита на символите. Всеки код е описан с експонентите на не нулите си, които са също и корени на проверочния му полином. За краткост на записа е даден само по един представител от всеки циклотомичен клас. Например всички не нули на първия код от първата таблица са 7; 2,4,6,8,10,12. Оказа се, че всички изследвани кодове, които имат неравномерна защита на информационните символи, са с ниво на защита 2. Затова сме отделили с ';' не нулите от двете множества. В първото подмножество са не нулите на подкода C_2, а във второто - тези на подкода C_1.

Таблица 6.2.1. Троични циклични UEP кодове

No	$[n, k, d]$	k_2	k_1	s_2	s_1	ненули
1.	$[14, 7, 4]$	1	6	7	4	7;2
2.	$[16, 10, 4]$	2	8	5	4	10;0,4,5,8
3.	$[16, 9, 4]$	1	8	6	4	8;2,5,10
4.	$[16, 8, 5]$	2	6	7	5	10;0,5,8
5.	$[16, 7, 6]$	3	4	7	6	8,10;5
6.	$[16, 7, 5]$	1	6	8	5	8;4,5
7.	$[16, 6, 6]$	2	4	9	6	10;5
8.	$[16, 6, 4]$	2	4	5	4	10;0,4,8
9.	$[16, 5, 6]$	1	4	10	6	8;5
10.	$[16, 5, 4]$	1	4	8	4	8;2,10
11.	$[20, 14, 4]$	1	13	5	4	10;0,1,4,11
12.	$[20, 13, 4]$	4	9	6	4	11;2,4,10
13.	$[20, 13, 4]$	1	12	6	4	10;1,4,11
14.	$[20, 12, 4]$	7	5	5	4	5,10,11;0,4
15.	$[20, 11, 4]$	2	9	5	4	10;2,4,5
16.	$[20, 11, 4]$	6	5	6	4	5,11;2,10
17.	$[20, 10, 4]$	5	5	6	4	10,11;0,4
18.	$[20, 10, 4]$	2	8	5	4	0,10;1,11
19.	$[20, 9, 6]$	1	8	10	4	10;1,11
20.	$[20, 9, 6]$	4	5	8	4	11;2,10
21.	$[20, 8, 5]$	4	4	8	5	11;0,5,10
22.	$[20, 7, 8]$	1	6	10	8	10;5,11
23.	$[20, 7, 4]$	2	5	10	4	5;2,10
24.	$[20, 6, 8]$	2	4	10	8	5;4
25.	$[20, 6, 4]$	1	5	10	4	10;0,4
26.	$[20, 5, 8]$	1	4	10	8	10;4
27.	$[22, 15, 4]$	5	10	6	4	7;2,4
28.	$[22, 11, 6]$	1	10	10	6	11;2,7
29.	$[22, 11, 4]$	1	10	11	4	11;2,4
30.	$[22, 7, 10]$	1	6	11	10	0;7,11
31.	$[22, 6, 12]$	1	5	13	12	11;4

Таблица 6.2.2. Троични негациклични UEP кодове

No.	$[n,k,d]$	k_2	k_1	s_2	s_1	ненули
1.	$[20,14,3]$	4	10	4	3	6;2,4,5
2.	$[20,12,3]$	2	10	5	3	1;2,4,5
3.	$[20,8,5]$	4	4	8	5	6;1,2

Резултатите в този раздел са съвместни с Ирина Ганчева и са публикувани в [216]. Представени са на International congress MASSEE в Боровец, България през 2003 г.

6.3 Нормализирани двоични линейни кодове

Свойството на един код да е нормализиран е въведено през 1985 година от Graham и Sloane в [103]. Това свойство трябва да притежават кодовете, за да могат да участват в конструкцията смесена директна сума на кодове (amalgamated direct sum (ADS)) представена в същата работа. Целта на тази конструкция е да се получат кодове с колкото е възможно по-малък радиус на покритие в сравнение с други кодове със същата дължина и размерност.

Свойството нормализираност и директната сума на кодове вече бяха използвани в раздел 3.4, за да бъдат конструирани двоични кодове с минимален радиус на покритие и размерности до 6 и тези две понятия бяха въведени там.

Резултата за смесената директна сума на Graham и Sloane е разширен в [119] така.

Теорема 6.3.1. *Ако A и B са нормализирани с четни норми, то $R(A\dot{\oplus}B) = R(A) + R(B) - 1$ и $A\dot{\oplus}B$ е нормализиран код.*

Дефиниция 6.3.2. *Нека C да е $[n,k]$ код и нека $c \in C$ да е с тегло w. Нека I да е множеството от не нулеви координати на c. Тогава остатъчен код на кода C по отношение на вектора x, означен с $R(C;x)$, е код с дължина $n-w$ свързан по всички координати от I.*

В [49] е доказан следният резултат като разширение на идеята за остатъчен код върху дума, която не е кодова дума.

Твърдение 6.3.3. *Нека C да е двоичен линеен код. Ако x е лидер на съсед клас на C с тегло $R = R(C)$, то $R(C;x)$ е $[n-R,k,d' \geq \lceil d/2 \rceil]$ код.*

Обобщение на това твърдение е направено в [119]. Нека да означим с $|x|$ теглото по Хеминг на вектора x.

Твърдение 6.3.4. *Ако x е лидер на съседен клас на C, то $R(C;x)$ е $[n-|x|, k, d' \geq \lceil d/2 \rceil]$ код.*

Да означим с C_σ кодът получен от C чрез премахване на i-тата му координата. Cohen, Lobstein и Sloane [50] дават следното достатъчно условие за един код да е нормализиран.

Твърдение 6.3.5. *Ако за някоя координата i, $R(C_\sigma) \leq R(C) + 1$, то C е нормализиран.*

Интересен изследователски въпрос е да се определи кои кодове са нормализирани. След работата на Graham и Sloane са направени още изследвания ([47], [50], [80], [109]-[113], [115], [119], [130]) и в следващото твърдение са обобщени параметрите, за които е известно, че двоичните кодове са нормализирани.

Твърдение 6.3.6. *(i) [130] Ако C е $[n, k, d]$ код с $n \leq 14$ или $k \leq 5$, или $d \leq 4$, или $R \leq 2$, то C е нормализиран.*
(ii) [115] Всички $[n, k, d]R$ кодове с $d \geq 2R - 1$ или $R = 3$ са нормализирани.
(iii) [119] Всички двоични $[n, k, d]$ кодове с $n = 15$ и $n - k \leq 9$ са нормализирани.

В този раздел ще покажем, че всички двоични кодове с дължини 16, 17 и 18 и ко-размерност 10 са нормализирани.

В [119] е получен резултат за това кога двоичните линейни кодове с дължина 16 и ко-размерност 10 са нормализирани.

Теорема 6.3.7. *Всички двоични линейни кодове с дължина 16 са нормализирани евентуално с изключение на $[16, 6, 5$ или $6]4$ кодовете имащи като подкодове всички кодове от типа $[15, 5, 5$ или $6]6$.*

Теорема 6.3.8. *Всички кодове с ко-размерност 10 са нормализирани с евентуално изключение на кодовете*
(i) $[16, 6, 5$ или $6]4$,
(ii) $[17 + j, 7 + j, 6]4$, $j = 0, 1, 2$,
(iii) $[17 + j, 7 + j, 5]4$, $0 \leq j \leq 5$,
в които за всяка координата скъсения код има радиус на покритие 6.

Ние разширяваме този резултат за кодове с дължини 17 и 18.

Теорема 6.3.9. *(i) Всички двоични линейни кодове с дължина 17 са нормализирани евентуално с изключение на $[17, 6, 5$ или $6]5$ кодовете имащи като подкодове всички кодове от типа $[16, 5, 5$ или $6]7$.*

(ii) Всички двоични линейни кодове с дължина 18 са нормализирани евентуално с изключение на $[18, 6, d \geq 5]5$ *кодовете имащи като подкодове всички кодове от типа* $[17, 5, d \geq 5]7$ *и* $[18, 7, 5$ *или* $6]5$ *кодовете имащи като подкодове всички кодове от типа* $[17, 6, 5$ *или* $6]7$.

Доказателство (i) В съответствие с твърдение 6.3.6 можем да приемем, че $[17, 6, d \geq 5]$ са единствените кодове, за които не е известно дали са нормализирани. Максималното минимално разстояние на двоичния $[17, 6]$ код е 7 (виж [29]). Нека $d = 7$. Тогава C_σ е $[16, 5, d' \geq 7]$ код. От [47, Таблица 7.1] се вижда, че минималния радиус на покритие на $[17, 6]$ кодовете е 5. От друга страна според твърдение 6.3.4, ако C е с радиус на покритие 7, то неговия остатъчен код $R(C; x)$ трябва да е $[10, 6, d' \geq 4]$ код. От [29] е известно, че максималното минимално разстояние на $[10, 6]$ кодовете е 3 и следователно радиуса на покритие на кода C е най-много 6. Следователно C е $[17, 6, 7]5$ или 6 код с подкод C_σ от типа $[16, 5, d' \geq 7]5$ или 6. Накрая базирайки се на твърдение 6.3.5 можем да заключим, че всички $[17, 6, 7]$ кодове са нормализирани.

Нека сега $d = 5$ или $d = 6$. C_σ е $[16, 5, d' \geq 5]$ код. Според [47, Таблица 7.1] и твърдение 6.3.4 $5 \leq R(C) \leq 7$ и $5 \leq R(C_\sigma) \leq 7$. От твърдение 6.3.5 получаваме, че единствения нерешен случай са $[17, 6, 5$ или $6]5$ кодовете с подкодове от типа $[16, 5, 5$ или $6]7$.

(ii) Нерешените случаи за дължина 18 са кодовете с размерности 6 и 7 (виж твърдение 6.3.6). Нека първо C да е $[18, 6, d \geq 5]$ код. За $d = 7$ или $d = 8$, $5 \leq R(C) \leq 7$ според [47, Таблица 7.1] и твърдение 6.3.4, а C_σ е $[17, 5, d' \geq 7]$ подкод с $R(C_\sigma) = 6$ или 7 ([47, Таблица 7.1] и твърдение 6.3.4). Тези възможности се редуцират до $[18, 6, 7$ или $8]5$ кодове с $[17, 5, 7$ или $8]7$ подкодове, след прилагането на твърдение 6.3.5. По аналогичен начин получаваме, че за $d = 5$ или $d = 6$, C би могъл да е само $[18, 6, 5$ или $6]5$ код с $[17, 5, d' \geq 5]7$ подкод. Като комбинираме резултатите за $d = 5$ или $d = 6$ и $d = 7$ или $d = 8$ получаваме, че C би могъл да е само $[18, 6, d \geq 5]5$ код с подкод от типа $[17, 5, d \geq 5]7$.

Нека сега C да е $[18, 7, d \geq 5]$ код. Неговото максимално минимално разстояние е 7 ([29]). За $[18, 7, 7]$ кодовете $R(C) = 5$ или $R(C) = 6$ и C_σ е от типа $[17, 6, d' \geq 7]5$ или 6 ([47, Таблица 7.1] и твърдение 6.3.4). Според твърдение 6.3.5 всички такива кодове са нормализирани. Ако $d = 5$ или $d = 6$ от [47, Таблица 7.1] и твърдение 6.3.4 получаваме $5 \leq R(C) \leq 7$ и $5 \leq R(C_\sigma) \leq 7$. Според твърдение 6.3.5, ако код с дължина $n = 18$ и размерност $k = 7$ не е нормализиран, то той трябва да е $[18, 7, 5$ или $6]5$ код с подкод от типа $[17, 6, 5$ или $6]7$. $\diamond$

За да завършим доказателството, че кодовете от теореми 6.3.7, 6.3.8 и 6.3.9 са нормализирани, ние конструираме всички такива кодове и проверяваме дали са

нормализирани. Класификацията е направена с разработения от Илия Буюклиев програмен пакет Q-EXTENSION [22], а определянето на радиусите на покритие на класифицираните кодове с програмата `CovRadM2`. Нормата на кода по отношение на всяка една от координатите му е определена с програма написана на C, като е използван следния резултат от [103].

Теорема 6.3.10. *Нека H да е проверочна матрица на кода C с параметри $[n,k]R$ и нека i да е не нулева негова координата. За всяка двоична $(n-k)$-орка s, нека $h_0(s)$ да е минималният брой стълбове на H, чиято сума дава s, без да се използва i-тия стълб и нека $h_1(s)$ да е минималният брой стълбове, чиято сума дава s, като i-тата координата задължително участва в тази сума. Тогава*

$$N^{(i)} = \max_s\{h_0(s) + h_1(s)\}$$

и C има норма N, тогава и само тогава когато за някоя координата i,

$$\max_s\{h_0(s) + h_1(s)\} = N.$$

Например, за да приключим случая $n = 16$, трябва да класифицираме всички $[15, 5, 5$ или $6]6$ кодове. Оказа се, че има 2638 такива кода и 114 от тях са с радиус на покритие $R = 6$. Тези 114 кода се разширяват с Q-EXTENTION до 5828 $[16, 6, 5$ или $6]$ кодове. Сред тях 4 са с радиус на покритие 4 и всички преминават теста за нормализираност.

Този подход е неприложим за останалите кодове, тъй като броя на кодовете, които трябва да се тестват нараства много бързо. За да направим възможна класификацията, доказваме следния резултат.

Твърдение 6.3.11. *Нека B да е $[n,k,d]R$ код и нека да го разширим до $[n+1, k+1, d' \leq d]$ код B_1. Тогава $R(B_1) \leq R(B)$.*

Доказателство. Без загуба на общност ще считаме, че пораждащата матрица на B_1 може да бъде представена по следния начин:

$$G(B_1) = \left[\begin{array}{c|c} A & C \\ \hline 0 & G_B \end{array}\right],$$

където A е $[1,1]0$ код и C е $[n,1]$ код. Тогава според [150]

$$R(B_1) \leq R(A) + R(B) = 0 + R(B) = R(B).$$

◇

Следователно, за да проверим дали кодовете от теорема 6.3.9 са нормализарани може да започнем от вече класифицираните $[15, 5, 5$ или $6]$ с радиус на покритие $R = 6$, да ги разширим до $[16, 6, 5$ или $6]R \leq 6$ кодове и след това да определим кои имат радиус на покритие $R = 6$, т.е. да намерим $[16, 6, 5$ или $6]6$ кодовете. На следващата стъпка разширяваме тези $[16, 6, 5$ или $6]6$ кодове до $[17, 7, 5$

или 6]$R \leq 6$ кодове. От получените кодове отделяме тези с радиус на покритие $R = 4$ и $R = 6$. Проверяваме дали $[17, 7, 5$ или $6]4$ кодовете са нормализирани и разширяваме $[17, 7, 5$ или $6]6$ кодовете, за да получим $[18, 8, 5$ или $6]R \leq 6$ кодове. По същия начин проверяваме дали $[18, 8, 5$ или $6]4$ кодовете са нормализирани и разширяваме $[18, 8, 5$ или $6]6$ кодовете до $[19, 9, 5$ или $6]R \leq 6$ кодове. Продължаваме тази йерархия на разширявания докато получим всички $[22, 12, 5]4$ кодове и проверим дали са нормализирани. По аналогичен начин построяваме кодовете от теорема 6.3.10 и проверяваме дали са нормализирани. Оказа се, че всички кодове са нормализирани и можем да заключим, че всички двоични линейни кодове с дължини 16, 17 и 18 или ко-размерност 10 са нормализирани.

Важен резултат от това изследване е и получената класификация на всички кодове с параметри от теореми 6.3.7, 6.3.8 и 6.3.9. Тогава като приложим смесената директна сума към тях, можем да получим нови кодове с малък радиус на покритие. Някои от кодовете имат четна норма и можем да приложим теорема 6.3.3, за да получим и по-добър резултат. В много случаи можем да използваме тази конструкция, за да получим кодове, които имат минималния възможен радиус на покритие $t_2[n, k]$. Например известно е от [47, Таблица 7.1], че $t_2[19, 7] = t_2[20, 7] = 5$ и $t_2[22, 11] = t_2[23, 11] = 4$. Смесената директна сума на класифицираните в тази работа нормализирани $[17, 6]5$ и $[20, 10]4$ кодове с четни норми (10 и 8 съответно) и нормализирания $[3, 2]1$ код с норма 2 дава $[19, 7]5$ и $[22, 11]4$ кодове. Или смесената директна сума на същите $[17, 6]5$ и $[20, 10]4$ кодове с четна норма и нормализирания $[4, 2]1$ код с норма 2 дава $[20, 7]5$ и $[23, 11]4$ кодове.

Резултатите в този раздел са получени самостоятелно, публикувани са в [227], представени са на International Workshop Optimal Codes and Related Topics във Варна, България и са публикувани в сборника с доклади на конференцията.

Глава 7

Оптични ортогонални кодове и свързани с тях комбинаторни структури

В тази глава са представени класификационни резултати за оптични ортогонални кодове. Това са първите класификационни резултати за такива кодове, които представляват интерес както от изследователска гледна точка, така и заради практическото им приложение. Наличието на класификационните резултати позволява конструираните кодове да се използват директно за различни практически приложения, като основа на нови безкрайни фамилии от кодове, както и да се докаже несъществуването на кодове с определени параметри. Заради връзката им с други комбинаторни структури, получените резултати за OOC водят до получаване на нови резултати и за тези структури.

Първите класификационни резултати за OOC с тегла 3 и 4, авто-корелация 2, крос-корелация 1 и малки дължини са получени в [228, 229, 232] и са представени в раздел 7.1. Също така, с компютърно търсене, са получени нови кодове с по-големи дължини, за които не бяха известни конструкции.

Един от интересните класове константно-тегловни кодове, намиращи разнообразни практически приложения, са циклично-пермутационните константно-тегловни (cyclically permutable constant weight, CPCW) кодове. В [230] са класифицирани с точност до мултипликативна еквивалентност оптималните циклично-пермутационни константно-тегловни кодове с тегло $k = 3$ и дължини $v \leq 61$. Заради еднозначното съответствие между тези кодове и оптималните двоични константно-тегловни кодове с тегло $k = 3$ и минимално разстояние 4, $(v, 3, \lfloor (v-1)/6 \rfloor)$ разностните пакетирания (difference packings) и оптималните $(v, 3, 1)$ оптич-

ни ортогонални кодове са получени класификации и за тези комбинаторни структури. Част от класифицираните циклично-пермутационни константно-тегловни кодове са съвършени и са еквивалентни на Щайнерови системи от тройки от ред v ($STS(v)$) и на $(v, 3, 1)$ циклични разностни множества (cyclic difference families, CDF). По този начин са получени нови класификационни резултати за $STS(61)$ и $(61, 3, 1)$ циклични разностни множества.

Като е използвана модификация на алгоритъма за класификация на оптични ортогонални кодове, в [231] са класифицирани циклични разностни множества с $k \leq 11$ и малки дължини. По този начин са повторени известни вече резултати, получени са нови класификации и е доказано несъществуване на някои циклични разностни множества.

7.1 Класификация на оптимални $(v, 4, 2, 1)$ и $(v, 5, 2, 1)$ оптични ортогонални кодове с малки дължини

Да означим с Z_v пръстена от целите числа по модул v.

Дефиниция 7.1.1. $(v, k, \lambda_a, \lambda_c)$ *оптичен ортогонален код (OOC) е колекция* $\mathcal{C} = \{C_1, \ldots, C_s\}$ *от* k*-елементни подмножества (кодови думи) на* Z_v*, такива че всеки две различни транслации на кодова дума имат най-много* λ_a *общи елементи, а всеки две транслации на две различни кодови думи имат най-много* λ_c *общи елемента:*

$$|C_i \cap (C_i + t)| \leq \lambda_a, \quad 1 \leq i \leq s, \quad 1 \leq t \leq v - 1 \tag{7.1}$$

$$|C_i \cap (C_j + t)| \leq \lambda_c, \quad 1 \leq i < j \leq s, \quad 0 \leq t \leq v - 1. \tag{7.2}$$

Условието (7.1) се нарича *автокорелационно свойство*, а условието (7.2) - *кроскорелационно свойство. Размер* на кодът $\mathcal{C}$ наричаме броя s на кодовите му думи. Когато $\lambda_a = \lambda_c$, кодът се означава с (v, k, λ).

Да разгледаме кодовата дума $C = \{c_1, c_2, \ldots, c_k\}$. Означаваме с $\triangle' C$ мултимножеството от стойности на разликите $c_i - c_j$, $i \neq j$, $i, j = 1, 2, \ldots, k$. Автокорелационното свойство означава, че най-много λ_a разлики са еднакви. Нека да означим с $\triangle C$ съответното на мултимножеството $\triangle' C$ множество. *Тип* на C е броят на елементите в $\triangle C$, т.е. броят различни стойности на разликите. Ако $\lambda_c = 1$, кроскорелационното свойство означава, че $\Delta C_1 \bigcap \Delta C_2 = \emptyset$ за две кодови думи C_1 и C_2 от $(v, k, \lambda_a, 1)$ оптичен ортогонален код.

Пример за $(v, 5, 2, 1)$ оптичен ортогонален код е представен на фигура 7.1.

Фигура 7.1: $(31, 5, 2, 1)$ **оптичен ортогонален код** $\mathcal{C}$

$\mathcal{C}$	$\triangle C$	Тип
{0,1,2,8,25}	1 2 6 7 8 14 17 23 24 25 29 30	12
{0,3,13,18,22}	3 4 5 9 10 12 13 15 16 18 19 21 22 26 28	15

Дефиниция 7.1.2. $(v, k, \lambda_a, 1)$ *оптичен ортогонален код е съвършен ако* $|\bigcup_{i=1}^{s} \Delta C_i| = v - 1$, *което означава, че всички възможни не нулеви разлики са покрити.*

Дефиниция 7.1.3. *Един оптичен ортогонален код е оптимален ако има максималния възможен брой кодови думи.*

Дефиниция 7.1.4. *Циркулантна матрица от ред* v *е (0,1) квадратна матрица* $M = (m_{i,j})_{v \times v}$ *с* v *реда и стълба, такива че* $m_{i+1,j+1} = m_{i,j}$, *където* $i, j = 0, 1, \ldots, v-1$ *и събирането на индексите е по модул* v.

Дефиниция 7.1.5. *Насочен граф с* v *върха, които съответстват на елементите на* Z_v, *се нарича циркулантен граф или циркуланта, ако има циклична симетрия, т.е. ако пермутацията* $(0, 1, 2, \ldots, n-1)$ *е автоморфизъм на графа.*

Матрицата на съседство на циркуланта е циркуланта. Ето защо има еднозначно съответствие между циркулантни графи и циркулантни матрици.

Дефиниция 7.1.6. *Let* $V = \{P_i\}_{i=1}^{v}$ *да е крайно множество от точки и* $\mathcal{B} = \{B_j\}_{j=1}^{b}$ *да е крайна колекция от* k*-елементни подмножества на* V, *наречени блокове.* $D = (V, \mathcal{B})$ *е дизайн (частичен дизайн) с параметри* t*-(*v*,*k*,*λ*), ако всяко* t*-елементно подмножество на* V *се съдържа в точно (най-много)* λ *блока на* $\mathcal{B}$.

Частичните дизайни се наричат също *опаковки* [184] или *пакетиращи дизайни* [89]. Ние ще ги наричаме частични дизайни следвайки [45].

Дефиниция 7.1.7. t*-(*v*,*k*,*λ*) дизайн (частичен дизайн) е цикличен, ако съществува автоморфизъм* α, *който пермутира точките му в един цикъл.*

От тук нататък, когато говорим за блокови орбити, ще разбираме блокови орбити под действието на автоморфизъм α, който пермутира точките в един цикъл.

От $(v, k, \lambda_a, \lambda_c)$ оптичен ортогонален код $\mathcal{C}$ може да бъде получен цикличен t-(v,k,λ) частичен дизайн D, чиито блокове са кодовите думи на $\mathcal{C}$ заедно с техните транслации. Всяка кодова дума заедно с транслациите си формира една блокова орбита. Матрицата на инцидентност на такъв цикличен дизайн се състои от циркулантни матрици от ред v, които съответстват на блоковите орбити. И обратно.

Един оптичен ортогонален код може да бъде получен от съответния му частичен дизайн като за кодови думи се изберат по точно един блок от всяка блокова орбита. В частност частичният дизайн, съответстващ на $(v, 4, 2, 1)$ оптичен ортогонален код, е частичен 2-(v,4,2) дизайн и в същото време частичен 3-(v,4,1) дизайн. Да разгледаме например оптимален $(20, 4, 2, 1)$ оптичен ортогонален код:

кодови думи	разлики	различни разлики	тип
{0,1,5,6}	1 19 5 15 4 16 6 14 5 15 1 19	1 4 5 6 14 15 16 19	8
{0,2,9,12}	2 18 9 11 7 13 12 8 10 10 3 17	2 3 7 8 9 10 11 12 13 17 18	11

Съответният му цикличен частичен $2-(20, 4, 2)$ дизайн се състои от орбитите на блоковете $\{0, 1, 5, 6\}$ и $\{0, 2, 9, 12\}$.

Дефиниция 7.1.8. *(n, w, d) двоичен константно-тегловен код с дължина n, тегло w и минимално разстояние d е колекция от двоични вектори с дължина n (кодови думи), които имат точно w не нулеви позиции и разстоянието по Хеминг между кои да е две кодови думи е поне d. Константно-тегловният код е цикличен, ако цикличното завъртане на всяка кодова дума е също кодова дума.*

$(v, k, \lambda_a, \lambda_c)$ оптичен ортогонален код съответства на цикличен $(v, k, 2(k-\lambda))$ константно-тегловен код за $\lambda = max(\lambda_a, \lambda_c)$. На фигура 7.2 на следващата страница е представен такъв пример.

Връзката с частичните дизайни показва, че оптичните ортогонални кодове могат да се разглеждат също и като циклични комбинаторни структури. Автоморфизмите на цикличната група от ред v съпоставят на всяка циркулантна матрица от ред v циркулантна матрица от ред v. Ще означаваме с $\varphi_0, \varphi_1, \ldots, \varphi_{m-1}$ автоморфизмите на елементите на Z_v. Пермутацията φ_i съпоставя на всеки елемент $z \in Z_v$ елемента $a_i z$, където a_i е примитивен корен по модул v, а φ_0 е идентитета. Ето защо за циклични комбинаторни структури се дефинира *мултипликативна еквивалентност* [44], [88]. Следователно тя може да се дефинира и за оптичните ортогонални кодове.

Дефиниция 7.1.9. *Два циркулантни графа G и G' с v върха са изоморфни, ако съществува пермутация φ на Z_v, такава че, всеки два върха u и v от G са съседни в G, тогава и само тогава когато $\varphi(u)$ и $\varphi(v)$ са съседни в G'.*

Дефиниция 7.1.10. *Два циркулантни графа G и G' с v върха са мултипликативно еквивалентни, ако съществува $m \in Z_v$, такова че m е примитивен корен по модул v и всеки два върха u и v от G са съседни в G, тогава и само тогава когато mu и mv са съседни в G'.*

Фигура 7.2: Връзки между оптичен ортогонален код и други комбинаторни структури

a) - Оптимален (31,5,2,1) оптичен ортогонален код - {0,1,2,8,25}, {0,3,13,18,22}.

b) - Свързаните с него

- цикличен $(31, 5, 6)$ константно-тегловен код (кодовите му думи са стълбовете на матрицата)

- цикличен $2 - (31, 5, 2)$ частичен дизайн (матрицата от фигурата е матрицата му на инцидентност).

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 1	1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0
1	1 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 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0
2	1 1 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 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
3	0 1 1 1 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	1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0
4	0 0 1 1 1 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 1 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0
5	0 0 0 1 1 1 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 1 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0
6	0 0 0 0 1 1 1 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 1 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0
7	0 0 0 0 0 1 1 1 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 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0
8	1 0 0 0 0 0 1 1 1 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 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0
9	0 1 0 0 0 0 0 1 1 1 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 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0
10	0 0 1 0 0 0 0 0 1 1 1 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 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0
11	0 0 0 1 0 0 0 0 0 1 1 1 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 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0
12	0 0 0 0 1 0 0 0 0 0 1 1 1 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 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1
13	0 0 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0	1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0
14	0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0	0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0
15	0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 0 0 0 0 0 0	0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0
16	0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 0 0 0 0 0	0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0
17	0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 0 0 0 0	0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1
18	0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 0 0 0	1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0
19	0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 0 0 0	0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 0
20	0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 0 0	0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0
21	0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 0	0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1
22	0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0 0	1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0
23	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0 0 1 0	0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0
24	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0 0 1	0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0
25	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0 0	0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0
26	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 1 1 0 0 0 0	0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0
27	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 1 1 0 0 0	0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0
28	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 1 1 0 0	0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0
29	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 1 1 0	0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 1 0 0
30	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 1 1	0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1

Дефиниция 7.1.11. *Два $(v, k, \lambda_a, \lambda_c)$ оптични ортогонални кода C и C' са изоморфни, ако съществува пермутация на Z_v, която изобразява колекцията от транслации на всяка кодова дума от C в колекция от транслации на кодова дума от C'.*

Дефиниция 7.1.12. *Два $(v, k, \lambda_a, \lambda_c)$ оптични ортогонални кода C и C' са мултипликативно еквивалентни, ако могат да бъдат получени един от друг чрез прилагане на автоморфизъм на Z_v и замяна на кодова дума с някоя от нейните транслации.*

Ще отбележим, че два оптични ортогонални кода (две циркуланти) могат да са изоморфни, но да са мултипликативно не еквивалентни.

Известни са много конструкции на оптични ортогонални кодове (виж например [34], [35], [45], [46], [89], [157], [206]). (v, k, λ) оптичните ортогонални кодове са широко изследвани, като особено сериозно внимание е отделено на $(v, k, 1)$ и $(v, k, 2)$. Дълъг списък с публикации върху тези изследвания е представен в [35], а някои последни резултати са описани в [156] и [199].

7.1.1 Класификация на оптимални $(v, 4, 2, 1)$ оптични ортогонални кодове

Оптични ортогонални кодове с параметри $(v, 4, 2, 1)$ са разгледани за първи път в [206]. По-късно в [157] е доказано, че ако s е размера на $(v, 4, 2, 1)$ оптичен ортогонален код, то

$$s \leq \lceil v/8 \rceil \text{ ако } v \equiv 7, 14 \ (mod\ 56) \tag{7.3}$$

$$s \leq \lfloor v/8 \rfloor \text{ в останалите случаи.} \tag{7.4}$$

Разнообразни конструкции на безкрайни фамилии от оптимални $(v, 4, 2, 1)$ оптични ортогонални кодове и някои резултати за несъществуване са представени в [35] и [157]. Все още обаче има много стойности на v, за които не е известно дали съществуват оптични ортогонални кодове. В нашата работа ние отговаряме на този въпрос за всички нерешени случаи за $v < 182$ и класифицираме с точност до изоморфизъм оптималните $(v, 4, 2, 1)$ оптични ортогонални кодове с $v < 76$, $v \neq 71$. Доказателствата в [35] показват, че за някои безкрайни фамилии е трудно теоретично да се покаже съществуването на оптични ортогонални кодове с малки параметри и в този случай компютърното търсене е по-ефективно (виж например [35], Теорема 4.6, Теорема 6.3). За конструирането на други безкрайни фамилии пък са нужни оптични ортогонални кодове с малки параметри и с допълнителни

свойства и в този случай класификационните резултати могат да са полезни. Например в забележката след Теорема 4.7 в [35] е показано, че полученият в теоремата резултат би могъл да бъде значително разширен, ако съществува $(88, 4, 2, 1)$ ООС, който има една кодова дума, чиито разлики са точно не нулевите елементи на подгрупата от ред 8 на Z_{88}. В нашата работа ние намираме такъв код (виж първата кодова дума на ООС с $v = 88$ в Таблица 7.1.1). В този смисъл, получените от нас резултати за съществуване и направените класификации на оптични ортогонални кодове с малки дължини могат да допринесат за бъдещите изследвания на кодове с по-големи дължини.

Класификация на оптималните $(v, 4, 2)$ оптични ортогонални кодове с $v \leq 44$ (с 3 изключения) е направена в [45]. В нея авторите конструират кодовете с помощта на компютър и алгоритъм базиран на задачата за максимална клика в граф. Нашият подход е съвършено различен, тъй като нашата цел е не само да конструираме по един оптимален код за всяко v, но и да направим класификация на тези кодове. В [44] и [88] е направена класификация с точност до изоморфизъм на циклични дизайни с определени параметри като първо е направена класификация с точност до мултипликативна еквивалентност. Ние също първо правим класификация до мултипликативна еквивалентност при класификацията с точност до изоморфизъм на $(v, 4, 2, 1)$ оптичните ортогонални кодове с $v < 76$, $v \neq 71$. Настоящата класификация обаче е по-сложна понеже някои от съвкупностите от транслации на кодовите думи са мултипликативно не еквивалентни, но изоморфни. По-сложна е и класификацията с точност до мултипликативна еквивалентност, защото трябва да се вземе предвид и типа на кодовите думи, който при кодовете с $\lambda_a = \lambda_c$ е един и същ, а когато $\lambda_a \neq \lambda_c$ е различен.

Ние класифицираме $(v, 4, 2, 1)$ оптичните ортогонални кодове с точност до мултипликативна еквивалентност като прилагаме добре известната техника на търсене с връщане с тест за минималност на частичните решения [126, раздел 7.1.2]. За целта първо подреждаме всички възможни кодови думи в съответствие с дефинирана върху тях лексикографска наредба.

Приемаме, че $c_1 < c_2 < c_3 < c_4$ за всяка кодова дума $C = \{c_1, c_2, c_3, c_4\}$. Дефинираме лексикографска наредба на кодовите думи, при която $C' = \{c'_1, c'_2, c'_3, c'_4\}$ е лексикографски по-малка от $C'' = \{c''_1, c''_2, c''_3, c''_4\}$, ако типа на C' е по-малък от типа на C'', или ако типовете на двете кодови думи са равни, но $c'_i = c''_i$ за $i < a$, а $c'_a < c''_a$. Ако заменим кодова дума от $C \in \mathcal{C}$ с нейна транслация $C + t \in \mathcal{C}$, то получаваме еквивалентен оптичен ортогонален код. Затова без загуба на общност, ние приемаме, че всяка кодова дума на оптимален $(v, 4, 2, 1)$ оптичен ортогонален код е лексикографски по-малка от всички свои транслации. Това означава, че $c_1 = 0$

и когато казваме, че пермутацията φ изобразява C_1 в C_2, ние имаме предвид, че C_2 е най-малката транслация на $\varphi(C_1)$.

Нека $\varphi_0, \varphi_1, \ldots, \varphi_{m-1}$ да са автоморфизмите на Z_v, като φ_0 е идентитета. Конструираме масив от всички множества от 4 елемента на Z_v, които биха могли да са кодови думи, т.е. които удовлетворяват автокорелационното свойство и са по-малки от всички свои транслации. Подреждаме ги според лексикографската наредба. На всяко възможно множество прилагаме пермутациите $\varphi_i, i = 1, 2, \ldots, m-1$. Ако някои от тях съответстват на по-малко множество, не добавяме текущото, защото то вече е в масива. С добавянето на текущото множество към масива, добавяме след него и $m-1$ множества, в които то се изобразява при $\varphi_1, \varphi_2, \ldots, \varphi_{m-1}$. Така получаваме следния масив L:

$\mathbf{L_0}$

$L_1 = \varphi_1 L_0$

$L_2 = \varphi_2 L_0$

$\vdots$

$L_{m-1} = \varphi_{m-1} L_0$

$\mathbf{L_m}$

$L_{m+1} = \varphi_1 L_m$

$L_{m+2} = \varphi_2 L_m$

$\vdots$

$L_{2m-1} = \varphi_{m-1} L_m$

$\vdots$

$\mathbf{L_{im}}$

$L_{im+1} = \varphi_1 L_m$

$L_{im+2} = \varphi_2 L_m$

$\vdots$

$L_{(i+1)m-1} = \varphi_{m-1} L_m$

След това прилагаме търсене с връщане, за да изберем кодовите думи на оптичния ортогонален код измежду всички възможности записани в масива L. При търсенето, първа кодова дума избираме само измежду множествата $\mathbf{L_i}$.

Тази наредба е удобна за ефективното изпълнение на теста за минималност на частичните решения. С теста за минималност ние проверяваме дали текущото решение може да бъде съпоставено на лексикографски по-малко от автоморфизмите на Z_v.

Също така прилагаме и тест за типа на частичните решения. Да приемем, че вече сме избрали r кодови думи от кода. Нека T да е типа на r-тата кодова дума

и нека d да е броя различни разлики, покрити от избраните r кодови думи. Тъй като търсим единствено оптимални кодове, т.е. кодове с s кодови думи, то, заради наредбата в масива L, типа на оставащите да бъдат избрани кодови думи трябва да е не по-малък от този на избраната r-та кодова дума. Ето защо $d + (s-r)T \leq v-1$. Ако това условие не се изпълнява, се връщаме една стъпка назад и търсим следваща възможност за $r-1$-ва кодова дума.

Така класифицираме оптичните ортогонални кодове с точност до мултипликативна еквивалентност.

След това прилагаме тест за изоморфизъм като съпоставяме на всяка кодова дума циркулатна матрица, получена от всички нейни транслации. Чрез проверка с компютър доказваме верността на следващата теорема.

Теорема 7.1.13. *За всички $(v, 4, 2, 1)$ оптични ортогонални кодове с $16 \leq v \leq 75$ и $v \not\equiv 0 \pmod 8$ всеки две циркуланти съответни на две възможни кодови думи са изоморфни, ако са мултипликативно еквивалентни.*

За някои от стойностите на v обаче тестът за изоморфизъм на циркулантите се оказа бавен за разработения от нас софтуер. Ето защо, за някои стойности на v, ние конструирахме множество от *обобщени множители*, дефинирани от Muzychuk [159]. Те формират множество от пермутации на Z_v наречени *множество на решенията*. Обобщените множители и съответното им множество на решенията се дефинират за всяка стойност на v. Muzychuk доказва, че две циркуланти от ред v са изоморфни, тогава и само тогава когато се изобразяват една в друга от някоя пермутация от множеството на решенията. За да получим множеството на решенията, ние първо намираме делителите на Z_v, такива че $Z_v = a_1 a_2 \dots a_r$, където $a_i = p_i^{\alpha_i}$ и $p_i \neq p_j$ за $i, j = 1, 2, \dots, r, i \neq j$. Всяко $x \in Z_v$ може да бъде представено от r-орката $(x_1, x_2, \dots, x_r)$, където $x_i \in Z_{a_i}$ и $x = \sum_{i=1}^{r} x_i \prod_{l=1}^{i-1} a_l$. От своя страна, цялото число x_i може да бъде представено като $(x_{i_0}, x_{i_1}, \dots, x_{i_{\alpha_i - 1}})$, където $x_{i_j} \in Z_{p_i}$ и $x_i = \sum_{j=0}^{\alpha_i - 1} x_{i_j} p_i^j$. Обобщеният множител е r-орката $(\vec{m}_1, \vec{m}_2, \dots, \vec{m}_r)$, където $\vec{m}_i = (m_{i_1}, m_{i_2}, \dots, m_{i_{\alpha_i}})$, m_{i_j} е цяло число, което е взаимно просто с p_i и $m_{i_j} < p_i^j \prod_{l=1}^{i-1} a_l$. Съответната на обобщения множител пермутация φ от множеството на решенията се дефинира като $\varphi x_i = \sum_{j=0}^{\alpha_i - 1} m_{i_{\alpha_i - j}} x_{i_j} p_i^j$.

За всички $v \equiv 0 \pmod 8$ с $16 \leq v \leq 75$ обаче съществуват кодови думи, които са мултипликативно не еквивалентни, но изоморфни.

Теорема 7.1.14. *Всеки два мултипликативно еквивалентни $(v, 4, 2, 1)$ оптични ортогонални кода с $16 \leq v \leq 75$ са не изоморфни.*

Доказателство. За $(v, 4, 2, 1)$ оптичните ортогонални кодове с $16 \leq v \leq 75$ и

$v \not\equiv 0 \pmod 8$ твърдението следва от теорема 7.1.13 и дефиницията на изоморфни оптични ортогонални кодове.

За $v = 16, 24, 32, 40, 48, 56, 64$ и 72 направихме проверката за еквивалентност с компютър. Тествахме за изоморфизъм всички оптични ортогонални кодове, които съдържат поне една кодова дума, която е мултипликативно не еквивалентна, но изоморфна на някоя друга кодова дума от същия или друг оптичен ортогонален код. За всеки оптичен ортогонален код проверихме дали има пермутация, която го преобразува в друг мултипликативно не еквивалентен код. За целта, тествахме всички пермутации от множеството на решенията, които съответстват на обобщените множители, дефинирани от Muzychuk [159]. Не бяха намерени изоморфни, но мултипликативно не еквивалентни кодове. ◇

Получените от нас класификационни резултати за $(v, 4, 2, 1)$ оптичните ортогонални кодове с $v \leq 75$ и $v \neq 71$ са достъпни на http://www.moi.math.bas.bg/∼tsonka и са обобщени в Таблица 7.1.1 от приложението. В някои от случаите не намираме кодове, които достигат границите (7.3) или (7.4) и класифицираме кодове с една кодова дума по-малко от границите. Затова в колоната s_b сме поставили стойността на границата, а в следващата най-големия намерен брой кодови думи s. Даваме също така общия брой на кодовете, броя на съвършените и по един пример за код със съответните параметри. В случаите, когато е имало съвършени кодове, примерът е на съвършен код. Тъй като всички кодови думи съдържат 0 в първа позиция сме я пропуснали, за да спестим място. В Таблица 7.1.2 са представени конструираните по един оптичен ортогонален код за $v \leq 181$. Знакът $\surd$ в колоната p означава, че кодът е съвършен.

7.1.2 Класификация на оптимални $(v, 5, 2, 1)$ оптични ортогонални кодове

Оптимални $(v, 5, 2, 1)$ оптични ортогонални кодове са разгледани в работата на Buratti, Pasotti и Wu [36]. В нея те показват, че за $(v, 5, 2, 1)$ оптични ортогонални кодове

$$s \leq \lceil v/12 \rceil \text{ ако } v \equiv 11 \ (mod\ 132) \text{ или } v \equiv 154 \ (mod\ 924) \tag{7.5}$$

$$s \leq \lfloor v/12 \rfloor \text{ в останалите случаи.} \tag{7.6}$$

Освен това те предлагат конструкции на $(v, 5, 2, 1)$ оптични ортогонални кодове достигащи тази граница с кодови думи от типа $\{0, x, -x, y, -y\}$, няколко рекурсивни конструкции на такива кодове, както и отговарят на въпроса за съществуването им за $v \leq 62$. В нашето изследване решаваме задачата за съществуването

на $(v, 5, 2, 1)$ оптични ортогонални кодове достигащи границата от [36] за всички нерешени случаи за $v \leq 155$ и класифицираме тези с $v \leq 114$. За целта използваме алгоритъма за класификация на оптималните $(v, 4, 2, 1)$ оптични ортогонални кодове, като разработваме негова версия за пресмятане на паралелен компютър. Така част от резултатите ни са получени като е използван българския суперкомпютър BlueGene.

Тъй като алгоритъмът е подобен на този за класификацията на оптималните $(v, 4, 2, 1)$ оптични ортогонални кодове, тук ще представим само особеностите на паралелната му реализация. Алгоритъмът е от типа търсене с връщане и може успешно да се имплементира на паралелен компютър с ускоряване близко до оптималното [122].

Да разгледаме претърсване в дълбочина имплементирано чрез стек. При последователния алгоритъм винаги работим с най-горния елемент от стека, т.е. използваме го в текущото решение, премахваме го от стека и поставяме на негово място наследниците му. За да направим търсенето паралелно, заменяме стека с *deque* (double ended queue) и реализираме търсенето по същия начин. Винаги даваме на друг процесор последния елемент от deque и той започва да го заменя с неговите наследници.

В [122] са разгледани два типа паралелни алгоритми за търсене с връщане. Рандомизирано търсене и глобален контрол. При търсенето с връщане с глобален контрол един или няколко процесора се използват единствено за управление на комуникацията, докато при рандомизираното търсене с връщане свободния процесор се обръща за работа към произволно избран друг процесор. В нашата реализация използваме глобален контрол като избираме за *мениджър* процесора с номер 0. Този процесор изпълнява само функции свързани с комуникацията с останалите.

В началото на работата на алгоритъма, всички върхове на дървото на решенията от първо ниво се конструират от всички процесори. Нека техният брой да е F. Така всеки процесор с номер $0 < p \leq F$ започва да разширява p-тия връх от първо ниво.

Мениджърът:

- Знае кой процесор работи с връх от най-ниско ниво.
- Когато процесорът завърши работата си, му казва
  - от кой процесор да получи работа,
  - да приключи работа.

Всеки от останалите процесори:

- Изпраща съобщение на мениджъра когато остане без работа.
- Изпраща съобщение на мениджъра когато сменя нивото на първия си връх.
- Поделя работата си с друг процесор, ако мениджърът му каже.

Тъй като получаваме решенията в лексикографски ред, при замяната на връх с неговите наследници, прилагаме теста за минималност, за да проверим наличието на еквивалентни лексикографски по-малки решения. Така добавяме в deque само наследници на решения, за които няма вече конструирани еквивалентни на текущото. Съобщението, което мениджъра изпраща към съответния процесор съдържа върхът, който трябва да се замени с неговите наследници и текущото частично решение. Това е достатъчно за процесора, за да започне работата си отново. Предоставяме му връх от най-ниското налично ниво, който е вероятно да се нуждае от най-много време, за да бъде обработен напълно и така ще заеме ресурса на процесора за дълго. Така минимизираме комуникацията с мениджъра, която отнема много време.

Конструираните от нас $(v, 5, 2, 1)$ оптични ортогонални кодове са достъпни на http://www.moi.math.bas.bg/∼tsonka. В таблица 7.1.3 от приложението са представени резултатите от класификацията до мултипликативна еквивалентност на достигащите границата от [36] $(v, 5, 2, 1)$ оптични ортогонални кодове с $v \leq 114$. В колоната s е посочен броят на кодовите думи на кода. След това е посочен броят на всички кодове и този на съвършените измежду тях. Както и в таблицата за $(v, 4, 2, 1)$, представяме и един от конструираните кодове. Когато измежду конструираните кодове има съвършени, е даден такъв код.

В таблица 7.1.4 от приложението сме обобщили нашите резултати за съществуване на достигащите границата от [36] $(v, 5, 2, 1)$ оптични ортогонални кодове с $115 \leq v \leq 155$. В колоната *съвършен* има отметка $\surd$, когато кодът е съвършен.

Примери на оптични ортогонални кодове с $29 \leq v \leq 62$ са представени в [36], но те се състоят само от кодови думи от вида $\{0, x, -x, y, -y\}$ и затова са съвършени само за $v \equiv 1 \ (mod\ 12)$. В таблица 7.1.4 са дадени съвършени кодове чиито кодови думи са без ограничение за вида им.

Преди настоящата работа не бяха известни примери на достигащи границата от [36] $(v, 5, 2, 1)$ оптични ортогонални кодове за

- v се дели на 12 и $s > 1$
- $v \equiv 154 \ (mod\ 924)$

- 76 стойности на v за $63 \leq v \leq 155$, за които не е установено несъществуване.

В нашата работа

- Намерени са примери на достигащи границата от [36] оптични ортогонални кодове с v, което се дели на 12 за дължини 108, 120, 132 и 144.
- Установено е, че не съществува достигащ границата от [36] $(154, 5, 2, 1)$ оптичен ортогонален код. Проблемът за съществуването на достигащи границата от [36] оптични ортогонални кодове с $v \equiv 154 \ (mod\ 924)$ остава отворен.
- Установено е несъществуването на достигащи границата от [36] оптични ортогонални кодове с $63 \leq v \leq 155$ за $v = 63$, 72, 84, 86, 96, 122 и 154. В останалите случаи са представени примери на достигащи границата от [36] оптични ортогонални кодове. Направена е и класификация за $v \leq 114$.

Получените в тази работа резултати за малки стойности на v могат да бъдат използвани и в рекурсивни конструкции. Ако рекурсивните конструкции от теореми 9.5, 9.6, 9.7, 9.8 и 9.9 от [36] се приложат към резултатите от директните конструкции от [36], следва че съществуват достигащи границата от [36] $(v, 5, 2, 1)$ оптични ортогонални кодове за 222 стойности на $v \leq 1000$, за 883 стойности на $v \leq 5000$ и за 1641 стойности на $v \leq 10000$. Ако се приложат същите рекурсивни конструкции към резултатите от [36] и към получените от нас резултати, следва че съществуват достигащи границата от [36] $(v, 5, 2, 1)$ оптични ортогонални кодове за 305 стойности на $v \leq 1000$, за 1067 стойности на $v \leq 5000$ и за 1964 стойности на $v \leq 10000$.

За да проверим коректността на получените компютърни резултати:

1. сравнихме нашите резултати с тези, получени в [36] и не намерихме различия;
2. за $v \leq 92$ получихме еднакви резултати с последователна и паралелна версия на разработените от нас компютърни програми.

7.2 Оптимални $(v, 3, 1)$ двоични циклично - пермутационни константно - тегловни кодове с дължини до 61

Един от най-интересните класове на константно-тегловните кодове е този на циклично-пермутационните константно-тегловни кодове. *Циклично - пермутационните кодове* (cyclically permutable codes, CPC) са въведени от Gilbert [100] през

1963 година за използване при CDMA комуникациите. Това са двоични блокови кодове с дължина n, такива че всяка кодова дума има n различни циклични завъртания и нито една кодова дума не може да бъде получена като циклично завъртане на някоя друга кодова дума. *Циклично-пермутационните константно-тегловни* кодове са едновременно константно-тегловни и циклично-пермутационни. Тези кодове са изследвани в [18, 158, 161] и са приложени при конструирането на последователности за комуникационни канали без обратна връзка използвани едновременно от много потребители.

Да означим с Z_v пръстена на целите числа по модул v, а с $\oplus$ и $\odot$ събирането и умножението в него.

Двоичният (v, k, λ) циклично-пермутационен константно-тегловен (CPCW) код $\mathcal{C}$ е колекция от $\{0, 1\}$ последователности с дължина v и тегло по Хеминг k такъв, че:

$$\sum_{i=0}^{v-1} x(i)x(i \oplus j) \leq \lambda, \ 1 \leq j \leq v-1 \tag{7.7}$$

$$\sum_{i=0}^{v-1} x(i)y(i \oplus j) \leq \lambda, \ 0 \leq j \leq v-1 \tag{7.8}$$

за всички двойки различни последователности $x, y \in \mathcal{C}$. Същата дефиниция е валидна и за (v, k, λ) оптичен ортогонален код.

Два (v, k, λ) циклично-пермутационни константно-тегловни кода C и C' са *изоморфни*, ако съществува пермутация на Z_v, която съпоставя на съвкупност от транслации на всяка кодова дума от C съвкупност от транслации на кодова дума от C'.

Два (v, k, λ) циклично-пермутационни константно-тегловни кода са *мултипликативно еквивалентни*, ако се получават един от друг чрез прилагане на автоморфизъм на Z_v и замяна на кодови думи с някои техни транслации.

Нека $\Phi(v, k, 1)$ да е най-големият възможен размер на $(v, k, 1)$ циклично-пермутационен константно-тегловен код. За кодове с $\lambda = 1$ имаме следната горна граница [46]

$$\Phi(v, k, \lambda_a, \lambda_c) \leq \left\lfloor \frac{(v-1)}{k(k-1)} \right\rfloor .$$

Ако $\Phi(v, k, 1)$ е точно $(v-1)/k(k-1)$, то $(v, k, 1)$ циклично-пермутационния константно-тегловен код се нарича *съвършен* и съответства на цикличен 2-$(v, k, 1)$ дизайн и на циклично $(v, k, 1)$ разностно множество. 2-$(v, 3, 1)$ дизайнът се нарича също Щайнерова система от тройки и се означава с $STS(v)$.

В [6] и [46] е показано, че циклично-пермутационни константно-тегловни кодове съществуват, тогава и само тогава когато $v \not\equiv 14$ или 20 $(mod\ 24)$. До нашата

работа не бяха известни класификационни резултати за циклично-пермутационни константно-тегловни кодове, но имаше такива за Щайнерови системи от тройки от ред v с $v \leq 57$ [52]. По-точно за $v = 19, 21, 25, 27, 31, 33, 37, 39, 43, 45, 49, 51, 55,$ и 57. Измежду тях, дизайните с $v = 19, 25, 31, 37, 43, 49$ и 55 са еквивалентни на $(v, 3, 1)$ циклично-пермутационни константно-тегловни кодове, докато дизайните с $v = 121, 27, 33, 39, 45, 51$ и 57 имат по една къса орбита. Щайнеровите системи от тройки са интересен клас дизайни с много различни приложения в теорията на кодирането (виж например [183] за връзката им със съвършени кодове).

Ние класифицираме с точност до мултипликативна еквивалентност оптималните $(v, 3, 1)$ циклично-пермутационни константно-тегловни кодове с $v \leq 61$. По този начин към известните вече класификации на циклични $STS(v)$ добавяме и такава за $v = 61$ като същевременно повтаряме и резултатите за циклични $STS(v)$ за $v \leq 57$.

За целта на класификацията, съпоставяме на всяка кодова дума $C = \{c_1, c_2, c_3\}$ вектор $\vec{C} = (c_1, c_2, c_3)$, такъв че $c_1 < c_2 < c_3$. Без загуба на общност приемаме, че $c_1 = 0$.

Нека $\vec{C_1}$ и $\vec{C_2}$ са два вектора съответстващи на кодовите думи C_1 и C_2. Когато казваме, че пермутацията φ съпоставя на $\vec{C_1}$ вектора $\vec{C_2}$, $\varphi(\vec{C_1}) = \vec{C_2}$, имаме предвид, че $\varphi(C_1) = C_2'$, C_2 е най-малката транслация на C_2', а $\vec{C_2}$ е векторът съпоставен на кодовата дума C_2. По тази причина е възможно две различни пермутации да съпоставят на даден вектор един и същ вектор.

Както при класификацията на $(v, 4, 2, 1)$ и $(v, 5, 2, 1)$ оптичните ортогонални кодове, създаваме масив L от всички 3-мерни вектори над Z_v, които могат да са кодови думи, т.е. по-малки са от своите транслации. Подреждаме ги в лексикографски ред. Към всеки вектор прилагаме пермутациите $\varphi_i, i = 1, 2, ...m - 1$ и добавяме само тези, които не са вече в масива. В масива съхраняваме и още $m - 1$ вектора, които съответстват на $\vec{C}$ след като се приложат пермутациите $\varphi_i, i = 1, 2, ...m - 1$. По този начин елементите на масива L, $L_x, x = 0, 1, ..., f$ са всички възможни вектори на кодови думи и са свързани по следния начин:

- ако $x \equiv 0 \ (mod\ m)$, то L_x не може да бъде съпоставен на някой вектор преди него в масива,
- ако $x \equiv i \ (mod\ m)$, то L_x се получава от L_{x-i} чрез пермутацията φ_i.

При тази организация на масива L е възможно някои кодови думи да се появят по-вече от веднъж в него. Затова за всяко L_x пазим също и най-малкото a, такова че $L_x = L_y$ и $y \equiv a \ (mod\ m)$. Записваме стойността на a на мястото на първата координата на кодовата дума c_1, тъй като тя е винаги 0. Така за

всяко x можем директно да определим най-малкото y, такова че L_y е получен чрез прилагане върху L_x на съответна пермутация от групата от автоморфизми на Z_v.

Конструираме кодовете като прилагаме търсене с връщане в елементите на масива L докато намерим s кодови думи $L_{x_1}, L_{x_2}, ..., L_{x_s}$. Без загуба на общност приемаме, че $x_1 \equiv 0 \ (mod\ m)$. На практика ние работим с номерата x_i на кодовите думи от L, които съхраняваме в масив, така че всеки елемент е по-голям от следващия, т.е. $x_1 \leq x_2 \leq ... \leq x_s$.

Избираме $r+1$-та кодова дума $L_{x_{r+1}}$ $(x_{r+1} > x_r)$, така че да няма общи разлики с предишните, вече избрани, r. Прилагаме също $\varphi_i, i = 1, 2, ..., m$ към текущото частично решение. Ако получения масив е лексикографски по-малък от текущия, това означава, че вече е бил разглеждан еквивалентен подкод с $r+1$ кодови думи и ние търсим следваща възможност за $r+1$-ва кодова дума.

В таблица 7.2.1 по-долу са представени класификационните резултати за оптималните $(v, 3, 1)$ циклично-пермутационни константно-тегловни кодове с $13 \leq v \leq 61$. След стойността на v е поставено p, ако кодът е съвършен. Както и в другите таблици, броят на кодовите думи е означен с s.

Таблица 7.2.1 Мултипликативно не еквивалентни оптимални (v,3,1) циклично-пермутационни константно-тегловни кодове

v	s	# (v,3,1) CPCW кодове
13p	2	1
14	1	3
15	2	5
16	2	3
17	2	5
18	2	12
19p	3	4
20	2	23
21	3	25
22	3	20
23	3	40
24	3	107
25p	4	12
26	4	36
27	4	128
28	4	164
29	4	400
30	4	1376

v	s	# (v,3,1) CPCW кодове
31p	5	80
32	5	242
33	5	1212
34	5	1360
35	5	6762
36	5	12784
37p	6	820
38	5	35120
39	6	15678
40	6	19794
41	6	68784
42	6	185376
43p	7	9508
44	6	621888
45	7	257886
46	7	231616
47	7	1137664
48	7	2712394

v	s	# (v,3,1) CPCW кодове
49p	8	157340
50	8	550528
51	8	3642484
52	8	4204688

v	s	# (v,3,1) CPCW кодове
53	8	21282112
54	8	54243072
55p	9	3027456
56	9	8660480

v	s	# (v,3,1) CPCW кодове
57	9	68638238
58	9	74974976
59	9	446472448
60	9	$\geq$ 455000000
61p	10	42373196

В случаите, когато кодовете са съвършени и отговарят на циклични Щайнерови системи от тройки, получаваме същите резултати както и в [52]. От класификацията на съвършените $(61, 3, 1)$ циклично-пермутационни константно-тегловни кодове, получаваме че съществуват 42373196 не еквивалентни циклични $STS(61)$, което е нов резултат.

За да проверим коректността на получените от нас резултати, конструирахме също и цикличните $STS(v)$, които имат една къса орбита и не са циклично-пермутационни константно-тегловни кодове. Орбитата е с дължина $v/3$ и е от вида $\{0, v/3, 2v/3\}$. Повторихме класификацията на цикличните $STS(v)$ с една къса орбита за $v = 15, 21, 27, 33, 39, 45, 51$ и 57 и получихме същите резултати както и в [52].

7.3 Класификация на $(v, k, 1)$ циклични разностни множества с малки параметри

Дефиниция 7.3.1. *Нека* $\mathbf{B}$ *да е подмножество на адитивната група* G*. Означаваме с* $\Delta\mathbf{B}$ *всички възможни разлики* $b - b'$*, където* (b, b') *е наредена двойка от различни елементи на* $\mathbf{B}$*. По-общо, ако* $\mathbf{F} = \{B_1, B_2, \ldots, B_n\}$ *е колекция от подмножества на* G*, то разликите от* $\mathbf{F}$*, които ще означаваме с* $\Delta\mathbf{F}$*, формират мултимножество получено чрез обединяване на* $\Delta B_1, \ldots, \Delta B_n$*.* $\mathbf{F}$ *е* $(v, k, 1)$ *разностно множество (DF), ако* G *е от ред* v*, всяко* B_i *е с размер* $k \geq 3$ *и* $\Delta\mathbf{F}$ *покрива всеки не нулев елемент на* G *точно веднъж. Когато* $G = Z_v$*, разностното множество се нарича циклично разностно множество (CDF).*

Класификацията на цикличните разностни множества представлява интерес

както от изследователска гледна точка, така и заради приложението им при конструирането на други видове комбинаторни структури. Известни са техни приложения при едно-факторизация на пълни графи и при циклично разрешими циклични Щайнерови системи от тройки [90], както и за конструирането на регулярни LDPC кодове [91]. В [114] е представена ефективна конструкция на нови оптимални системи със съвършена сигурност на базата на разностни множества. Оптимални последователности за прескачане на честоти могат също да бъдат конструирани от $(v, k, 1)$ циклични разностни множества.

Цикличните разностни множества са еквивалентни на няколко други комбинаторни структури. Една от дефинициите на $(v, k, 1)$ оптичен ортогонален код е следната

Дефиниция 7.3.2. $(v, k, 1)$ *оптичен ортогонален код е множество от* k*-елементни подмножества на* Z_v*, чиито разлики нямат повтарящи се елементи.*

Списъкът от разлики на съвършен $(v, k, 1)$ оптичен ортогонален код покрива всички не нулеви елементи на Z_v и следователно формира $(v, k, 1)$ циклично разностно множество.

$(v, k, 1)$ циклично разностно множество може да бъде получено от всеки оптимален съвършен $(v, k, 1)$ циклично-пермутационен константно-тегловен код (оптимален съвършен $(v, k, 1)$ оптичен ортогонален код) и от всеки оптимален двоичен цикличен константно-тегловен код с тегло k и минимално разстояние $2(k-1)$. Цикличните разностни множества с $v \equiv k \ (mod\ k(k-1))$ не съответстват на циклично-пермутационни константно-тегловни кодове (оптични ортогонални кодове) и на двоични циклични константно-тегловни кодове.

Има еднозначно съответствие между $(v, k, 1)$ цикличните раностни множества и цикличните 2-$(v, k, 1)$ дизайни. На фиг. 7.3, на следващата страница, е представена връзката между цикличните разностни множества и изброените комбинаторни структури.

Тъй като целта на нашето изследване е класификация на циклични разностни множества, е необходимо да знаем кога две такива комбинаторни структури са еквивалентни.

Дефиниция 7.3.3. *Две разностни множества* $\mathbf{F} = \{B_1, B_2, \ldots, B_n\}$ *и* $\mathbf{F}' = \{B'_1, B'_2, \ldots, B'_n\}$ *над групата* G *са еквивалентни, ако съществува автоморфизъм* α *на* G*, такъв че за всяко* $i = 1, 2, \ldots, n$ *съществува* B'_j*, което е транслация на* $\alpha(B_i)$*.*

Ще отбележим също, че броят на мултипликативно не еквивалентните оптимални съвършени 2-$(v, k, 1)$ циклично-пермутационни константно-тегловни кодове (оптични ортогонални кодове) е същия както и броя на не еквивалентните

Фигура 7.3: Връзки на циклични разностни множества с други комбинаторни структури

a) - Съвършен оптичен ортогонален код и циклично разностно множество

кодови думи	разлики
{0,1,4}	1 3 4 9 10 12
{0,2,8}	2 5 6 7 8 11

b) - Свързаните с тях цикличен 2-(13,3,1) дизайн и двоичен цикличен (13,3,4) константно-тегловен код

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

2-$(v,k,1)$ циклични разностни множества. Броят на не еквивалентните $(v,k,1)$ циклични разностни множества е същия както и броя на мултипликативно не еквивалентните циклични 2-$(v,k,1)$ дизайни и обратно.

В следващите теореми са обобщени известните резултати за съществуване на циклични разностни множества с $k = 3,4,5,6,7$.

Теорема 7.3.4. $(v,k,1)$ *циклично разностно множество съществува за* $v \equiv 1,k \ (mod\ k(k-1))$.

Теорема 7.3.5. *([63])* $(v,3,1)$ *разностно множество съществува за всяко* $v \equiv 1,3 \ (mod\ 6)$ *с изключение на* $v = 9$.

Теорема 7.3.6. *([21],[32],[33],[43],[139]) 1. За всяка степен на просто* $q \equiv 1 \ (mod\ 12)$ *съществува* $(q,4,1)$ *разностно множество в* $GF(q)$.

2. За всяка степен на просто $q \equiv 1 \ (mod\ 20)$ *съществува* $(q,5,1)$ *разностно множество в* $GF(q)$.

Теорема 7.3.7. *([6],[7]) 1.* $(12t+1,4,1)$ *разностно множество съществува за* $1 \leq t \leq 50$ *с изключение на* $t = 2$.

2. $(20t+1,5,1)$ *разностно множество съществува за* $1 \leq t \leq 50$ *с изключение на* $t = 16,25,31,34,40,45$.

Теорема 7.3.8. *([41])* $(q,6,1)$ *разностно множество съществува за всяка степен на просто* $q \equiv 1 \ (mod\ 30)$ *с изключение на* $q = 61$.

Теорема 7.3.9. *([42])* $(q,7,1)$ *разностно множество съществува за всяка степен на просто* $q \equiv 1 \ (mod\ 42)$ *с изключение на* $q = 43$ *и е възможно да съществува за* $q = 127,211,31^6$ *и простите числа в интервала* $q \in [261239791, 1.236597.10^{13}]$, *такива че* $(-3)^{\frac{(q-1)}{14}} = 1$ *в* $GF(q)$.

До нашата работа бяха известни класификации за $k = 3$ и $v \leq 57$ [52], $k = 4$ и $v \leq 64$ [44], $k = 5$ и $v \leq 65$ [44], $k = 6$ и $v = 91$ [51], [118] и $k = 7$ и $v = 91$ [16]. Ние разширяваме тези резултати като класифицираме циклични разностни множества с $k \leq 11$ и малки стойности на v.

Заради връзката на цикличните разностни множества с оптичните ортогонални кодове и циклично-пермутационните константно-тегловни кодове, можем да използваме подобен на алгоритъма описан в предишния раздел. Отново формираме масив с всички множества $\mathbf{B}$ от k елемента на Z_v, които могат да са от $\mathbf{F}$, т.е. те са по-малки от всички техни транслации и $\Delta\mathbf{B}$ не съдържа повтарящи се разлики. Голяма част от множествата на едно циклично разностно множество

имат $v-1$ транслации. Ако $v \equiv k \ (mod\ k(k-1))$, едно от множествата има v/k транслации и е еквивалентно на $\{0, v/k, 2v/k, \ldots, (k-1)v/k\}$. Първо добавяме това множество. След това прилагаме търсене с връщане, за да изберем множествата с $v-1$ транслации от масива.

Така повтаряме всички известни предишни класификации за циклични разностни множества като във всички случаи получаваме същите резултати. Тези резултати са получени от много автори в различни работи и смятаме, че събирането им на едно място е много полезно както за различни практически приложения, така и за бъдещи научни изследвания. Към тях добавяме и класификациите за $k \leq 7$ на $(61, 3, 1)$, $(73, 4, 1)$, $(76, 4, 1)$, $(81, 5, 1)$ и $(85, 5, 1)$ цикличните разностни множества, както и правим първите класификации за $k \geq 7$. Резултатите са обобщени в следващата таблица.

Таблица 7.3.1 Не еквивалентни циклични (v,k,1) разностни множества (мултипликативно не еквивалентни циклични 2-(v,k,1) дизайни)

v	k	брой
7	3	1
9	3	0
13	3	1
15	3	2
19	3	4
21	3	7
25	3	12
27	3	8
31	3	80
33	3	84
37	3	820
39	3	798
43	3	9508
45	3	11616
49	3	157340
51	3	139828
55	3	3027456
57	3	2353310
61	3	42373196

v	k	брой
13	4	1
16	4	0
25	4	0
28	4	0
37	4	2
40	4	10
49	4	224
52	4	206
61	4	18132
64	4	12048
73	4	1426986
76	4	1113024
21	5	1
25	5	0
41	5	1
45	5	0
61	5	10
65	5	2
81	5	528

v	k	брой
85	5	170
31	6	1
36	6	0
61	6	0
66	6	0
91	6	4
96	6	0
43	7	0
49	7	0
85	7	0
91	7	2
57	8	1
64	8	0
73	9	1
81	9	0
91	10	1
100	10	0
111	11	0
121	11	0

Резултатите в тази глава са съвместни със Светлана Топалова и са публикувани в [228, 229, 230, 231, 232].

Приложение А

Радиуси на покритие на троичните негациклични кодове с дължини до 26

Таблица 3.1.1

№	n	k	d	$R(C)$	Пораждащ полином
1	4	2	3	1	112
2	6	4	2	2	101
3	6	2	3	4	10201
4	8	4	3	2	10102
5	10	8	2	2	101
6	10	6	4	2	11021
7	10	4	6	5	1110121
8	10	2	5	6	102010201
9	12	10	2	2	112
10	12	8	2	4	10001
11	12	8	3	2	12211
12	12	6	3	4	1220122
13	12	6	3	3	1002002
14	12	4	3	8	100020001
15	12	4	6	6	122122211
16	12	2	9	8	12202110122
17	14	12	2	2	101
18	14	8	5	3	1101011
19	14	6	6	5	111202111
20	14	2	7	8	1020102010201
21	16	8	3	4	100010002
22	18	16	2	2	101
23	18	14	2	4	10201
24	18	12	2	6	1000001
25	18	10	3	6	101000101
26	18	8	3	8	10201010201
27	18	6	3	12	1000002000001

28	18	4	6	12	101000202000101
29	18	2	9	12	10201020102010201
30	20	18	2	1	112
31	20	16	2	4	10001
32	20	16	3	2	10111
33	20	14	3	3	1102102
34	20	14	4	3	1100222
35	20	12	5	4	112212211
36	20	12	4	4	102000101
37	20	12	4	4	101120111
38	20	12	3	4	122020211
39	20	10	7	5	12201101002
40	20	10	3	5	10000200002
41	20	10	6	4	11121011012
42	20	8	6	10	1020100010101
43	20	8	6	8	1220021100211
44	20	8	8	7	1122201022211
45	20	8	5	7	1102000121201
46	20	6	9	9	101212021220212
47	20	6	9	8	120121010211212
48	20	4	5	12	10002000100020001
49	20	4	12	11	11021100212101201
50	20	2	15	13	1120221011202210112
51	22	20	2	2	101
52	22	12	5	4	10002020201
53	22	10	6	10	1010201010001
54	22	2	11	14	10201020102010201
55	24	20	2	2	10102
56	24	16	2	8	100000001
57	24	16	3	4	102020101
58	24	12	3	8	1020200010202
59	24	12	3	6	1000001000002
60	24	8	3	16	10000000200000001
61	24	8	6	12	10102020201020201
62	24	4	9	16	1020200020101000102020
63	26	24	2	2	101
64	26	20	3	3	1000201
65	26	20	3	2	1010201
66	26	18	3	6	102000001
67	26	18	3	5	101020001
68	26	14	5	6	1020200020201
69	26	14	4	6	1010101020101

70	26	12	6	9	100010202010001
71	26	12	6	8	102020200000201
72	26	8	7	11	10000010200010101 01
73	26	6	9	14	10100010002010202020 1
74	26	2	13	16	10201020102010201020102 01

Приложение Б

Минимален радиус на покритие на троичните циклични кодове с дължини до 40

Таблица 3.5.1 $t_3[n,k]$

$k\backslash n$	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	2	2	3	4	4	5	6	6	7	8	8	9	10	10	11	12	12
2	1	1	2	3	4	4	5	6	6	7	8	8	9	10	10	11	12
3		1	1	2	3	3	4	5	5	6	7	7	8	9	9	10	11
4			1	1	2	2	3	4^{O_2}	4	5^{O_2}	5-6	5-6	6-7	6-8	7-9	7-9	8-10
5				1	1	2	2	3	3^{D}	4	4-5	5^{O_1}	5-6	6-7	6-8	7-8	7-9
6					1	1	2	2	2	3	3-4	4-5	4-5	5-6	5-7	6-7	6-8
7						1	1	2	2	2	3	3	4	4-5	5-6	5-6	6-7
8							1	1	2	2	2	3	3	4^{1}	4-5	5^{2}	5-6
9								1	1	2	2	2	3	3	4	4-5	4-5
10									1	1	1	2	2	3	3	3-4	4-5
11										1	1	1	2	2	3	3	3-4
12											1	1	1	2	2	3	3
13												1	1	1	2	2	3
14													1	1	1	2	2
15														1	1	1	2
16															1	1	1
17																1	1
18																	1

$t_3[n,k]$

$k\backslash n$	20	21	22	23	24	25	26	27	28	29	30
1	13	14	14	15	16	16	17	18	18	19	20
2	12	13	14	14	15	16	16	17	18	18	19
3	11	11	12	13	14	15	15	16	17	17	18
4	8-11	9-11	10-12	10-13	11-13	11-14	12-15	12-15	13-16	13-17	14-17
5	8-9	8-10	9-11	9-11	10-12	10-13	11-13	11-14	12-15	12-15	13-16
6	7-8	7-9	8-9	8-10	9-11	9-12	10-12	10-13	11-14	11-14	12-15
7	6-8	7-8	7-8	8-9	8-10	9-11	9-12	10-12	10-13	11-14	11-14
8	5-7	6-7	6-8	7-8	7-9	8-10	8-11	9-11	9-12	10-13	10-13
9	5-6	5-7	6-7	6-8	7-8	7-9	8-10	8-10	9-11	9-12	10-12
10	4^n	5^n	5-6	6-7	6-8	7-8	7-9	7-10	8-10	8-11	9-11
11	4^n	4	5	5-6	6^3	6-7	6-8	7-8	7-9	8-9	8-10
12	3-4	4	4	5^c	5^n	5-6	6-7	6-8	7-8	7-9	8-9
13	3	3-4	4	4	4-5	5	5-6	6-7	6-7	7-8	7 -9
14	2-3	3	3-4	4	4	4-5	5	5-6	6^4	6-7	7 -8
15	2	2-3	3	3-4	4	4	4-5	5	5-6	6	6- 7
16	2	2	2	3	3^D	4	4	4-5	5	5-6	5-6
17	1	2	2	2	3	3^D	3-4	4	4-5	5	5-6
18	1	1	2	2	2	3	3^D	3-4	4	4-5	5
19	1	1	1	2	2	2	3	3	3-4	4	4-5
20		1	1	1	2	2	2	3	3	3-4	4
21			1	1	1	2	2	2	3	3	3- 4
22				1	1	1	2	2	2	3	3
23					1	1	1	2	2	2	3
24						1	1	1	2	2	2
25							1	1	1	2	2
26								1	1	1	2
27									1	1	1
28										1	1
29											1

$t_3[n,k]$

$k\backslash n$	31	32	33	34	35	36	37	38	39	40
1	20	21	22	22	23	24	24	25	26	26
2	20	20	21	22	22	23	24	24	25	26
3	19	19	20	21	21	22	23	23	24	25
4	15-18	15-19	16-19	16-20	17-21	17-21	18-22	19-23	19-23	20- 24
5	14-17	14-17	15-18	15-19	16-19	16-20	17-21	17-21	18-22	19- 23
6	13-16	13-16	14-17	14-18	15-18	15-19	16-20	16-20	17-21	17 -22
7	12-15	12-16	13-16	13-17	14-18	14-18	15-19	15-20	16-20	16- 21
8	11-14	11-15	12-15	12-16	13-17	13-17	14-18	14-18	15-20	16- 20
9	10-13	11-14	11-15	12-15	12-16	13-17	13-17	14-18	14-18	15- 19
10	9-12	10-13	10-14	11-15	11-15	12-16	12-17	13-17	13-18	14- 18
11	9-11	9-12	10-12	10-13	11-14	11-14	12-15	12-16	13-16	13- 17
12	8-10	9-11	9-11	9-12	10-13	10-14	11-14	11-15	12-16	12- 16
13	7-9	8-10	8-10	9-11	9-12	10-13	10-13	11-14	11-15	12- 15
14	7-9	7-9	8-10	8-10	9-11	9-12	10-12	10-13	11-13	11- 14
15	6-7	7-8	7-9	8-10	8-10	9-11	9-11	9-12	10-13	10- 13
16	6^5	6-7	7-8	7-9	8-9	8-10	8-11	9-11	9-12	10- 13
17	5-6	6	6-7	7^7	7-8	7-9	8-10	8-10	9-11	9- 11
18	5-6	5-6	6	6-7	7	7^8	7-9	8-9	8-10	9- 11
19	4-5	5-6	5-6	6	6-7	6-7	7	7-8	8-9	8 -9
20	4-5	4-5	5^6	5-6	6	6-7	6-7	7	7-8	8^9
21	4	4-5	4-5	5	5-6	6	6-7	6-7	7	7-8
22	3-4	4	4-5	4-5	5	5-6	5-6	6-7	6-7	7
23	3	3-4	4	4-5	4-5	5	5-6	5-6	6-7	6-7
24	3	3	3-4	4	4-5	4-5	5	5-6	5-6	6- 7
25	2	3	3	3-4	4	4-5	4-5	5	5-6	5-6
26	2	2	3	3	3-4	3-4	4-5	4-5	5	5-6
27	2	2	2	2-3	3	3-4	3-5	4-5	4-5	5
28	1	2	2	2	2-3	3	3-4	3-4	4-5	4-5
29	1	1	2	2	2	2-3	3	3-4	3-4	4-5
30	1	1	1	2	2	2	2-3	3	3-4	3-4
31		1	1	1	2	2	2	2-3	3	3 -4
32			1	1	1	2	2	2	2-3	3
33				1	1	1	2	2	2	2-3
34					1	1	1	2	2	2
35						1	1	1	2	2
36							1	1	1	2
37								1	1	1
38									1	1
39										1

Приложение В

Подходящи двоични и троични циклични, троични негациклични и двоични кодове с максимално минимално разстояние

Таблица 4.1.1. Двоични циклични кодове.

№	[n,k,d]	пораждащ полином	подходящ
1.	$[7,4,3]^*$	1101	t=0,1
2.	[7,3,4]	10111	t=0,1
3.	$[9,3,3]^*$	1001001	t=0,1
4.	$[9,2,6]^{o*}$	11011011	t=0,1,2
5.	$[15,11,3]^{o*}$	11001	t=0,1
6.	$[15,10,4]^o$	101011	t=0,1
7.	[15,9,3]	1001111	t=0,1
8.	$[15,9,4]^o$	1011101	t=0,1
9.	$[15,8,4]^o$	11010001	t=0,1
10.	$[15,8,4]^o$	11100111	t=0,1
11.	$[15,7,3]^*$	110111011	t=0
12.	$[15,7,5]^{o*}$	100010111	t=0,1,2
13.	$[15,6,6]^o$	1011001101	t=0,1,2
14.	$[15,6,6]^{o*}$	1100111001	t=0,1,2
15.	$[15,5,7]^{o*}$	10000100001	t=0,1,2,3
16.	[15,4,6]	110001100011	t=0,1
17.	$[15,4,8]^o$	100110101111	t=0,1,2,3
18.	$[15,2,10]^{o*}$	11011011011011	t=0,1,2,3,4
19.	$[17,9,5]^{o*}$	100111001	t=0,1,2
20.	$[17,8,6]^o$	1101001011	t=0,1,2
21.	$[21,16,3]^{o*}$	100011	t=0,1
22.	$[21,15,3]^*$	1110101	t=0,1
23.	$[21,15,4]^o$	1100101	t=0,1
24.	$[21,14,4]^o$	0011111	t=0,1
25.	$[21,13,3]^*$	01001011	t=0,1
26.	$[21,13,4]^{o*}$	101111101	t=0,1
27.	[21,12,4]	1111011101	t=0,1

28.	$[21,12,5]^*$	1100110111	t=0,1,2
29.	$[21,11,6]^o$	10101011001	t=0,1,2
30.	[21,10,5]	100110000101	t=0,1,2
31.	[21,9,6]	1011001010011	t=0,1,2
32.	$[21,9,8]^o$	1001001000001	t=0,1,2,3
33.	[21,8,6]	11101011110101	t=0,1,2
34.	[21,8,6]	10110111101101	t=0,1,2
35.	$[21,7,8]^o$	110001110111001	t=0,1,2,3
36.	$[21,6,7]^*$	10101100111011111	t=0,1,2,3
37.	$[21,6,8]^o$	1010010011001011	t=0,1,2,3
37.	$[21,5,10]^{o*}$	11111010100110001	t=0,1,2,3,4
39.	$[21,4,9]^*$	110100011010001101	t=0,1,2,3,4
40.	$[21,3,12]^o$	10111001011100101111	t=0,1,2,3,4,5
41.	$[21,2,14]^{o*}$	110110111011011011011	t=0,1,2,3,4,5,6
42.	$[23,12,7]^{o*}$	110001110101	t=0,1,2,3
43.	$[23,11,8]^o$	1010010011111	t=0,1,2,3
44.	$[27,2,18]^{o*}$	11011011011011011011011011	t=0,1,2,3,4,5,6,7,8
45.	$[31,26,3]^{o*}$	101001	t=0,1
46.	$[31,25,4]^o$	1111011	t=0,1
47.	$[31,21,5]^{o*}$	10110101101	t=0,1,2
48.	$[31,21,5]^{o*}$	11001110101	t=0,1,2
49.	$[31,21,5]^{o*}$	10010110111	t=0,1,2
50.	$[31,20,6]^o$	111011110111	t=0,1,2
51.	$[31,20,6]^o$	101010011111	t=0,1,2
52.	$[31,20,6]^o$	110111011001	t=0,1,2
53.	[31,16,5]	1001000011000111	t=0,1,2
54.	[31,16,6]	1100011110110101	t=0,1,2
55.	[31,16,7]	1101000100000001	t=0,1,2,3
56.	[31,16,7]	1001110000101101	t=0,1,2,3
57.	[31,15,6]	10100100011011111	t=0,1,2
58.	$[31,15,8]^o$	10111001100000011	t=0,1,2,3
59.	$[31,15,8]^o$	11011000101001001	t=0,1,2,3
60.	$[31,15,8]^o$	11100111111001101	t=0,1,2,3
61.	$[31,11,11]^{o*}$	1000011001011001110 11	t=0,1,2,3,4,5
62.	$[31,11,11]^{o*}$	1010100000111001101 11	t=0,1,2,3,4,5
63.	[31,11,10]	1110110011100000101 01	t=0,1,2,3,4
64.	$[31,10,12]^o$	1100010101110101001101	t=0,1,2,3,4,5
65.	$[31,10,12]^o$	1111110000100101011001	t=0,1,2,3,4,5
66.	[31,10,10]	10001110100001011100 01	t=0,1,2,3,4
67.	$[31,6,15]^{o*}$	11011001111010010101110001	t=0,1,2,3,4,5,6,7
68.	$[31,5,16]^{o*}$	10110101000111011111001001 11	t=0,1,2,3,4,5,6,7

Таблица 4.1.2. Троични циклични кодове.

№	[n,k,d]	пораждащ полином	подходящ
1.	$[4,1,4]^{o*}$	1211	t=0,1
2.	$[8,5,3]^{o*}$	1011	t=0,1
3.	$[8,4,4]^{o}$	11012	t=0,1
4.	$[8,3,5]^{o}$	102111	t=0,1,2
5.	[8,3,4]	120012	t=0,1
6.	[8,2,6]	1120221	t=0,1,2
7.	[8,2,4]	1020102	t=0,1
8.	$[8,1,8]^{o*}$	12121212	t=0,1,2,3
9.	$[10,5,4]^{*}$	112122	t=0,1
10.	$[10,1,10]^{o*}$	1212121212	t=0,1,2,3,4
11.	$[11,6,5]^{o*}$	102122	t=0,1,2
12.	$[11,5,6]^{o}$	1222101	t=0,1,2
13.	$[11,1,11]^{o*}$	11111111111	t=0,1,2,3,4,5
14.	$[13,10,3]^{o*}$	1112	t=0
15.	$[13,9,3]^{o}$	10011	t=0
16.	$[13,7,5]^{o*}$	1022201	t=0,1,2
17.	$[13,7,4]^{*}$	1222121	t=0,1
18.	$[13,6,6]^{o}$	12200112	t=0,1,2
19.	[13,6,6]	11002122	t=0,1,2
20.	$[13,4,7]^{o}$	1120102201	t=0,1,2,3
21.	$[13,3,9]^{o*}$	10111220121	t=0,1,2,3,4
22.	$[13,1,13]^{o*}$	1111111111111	t=0,1,2,3,4,5,6
23.	$[14,1,14]^{o*}$	12121212121212	t=0,1,2,3,4,5,6 g&p
24.	$[16,10,4]^{o*}$	1101121	t=0
25.	$[16,9,5]^{o}$	10210122	t=0,1
26.	[16,8,5]	111210221	t=0,1
27.	$[16,7,6]^{o}$	1001222022	t=0,1
28.	[16,6,6]	11010112212	t=0,1
29.	$[16,3,10]^{o}$	10211100102111	t=0,1,2,3,4
30.	$[16,2,12]^{o}$	112022101120221	t=0,1,2,3,4,5
31.	$[16,1,16]^{o*}$	12121212121212	t=0,1,2,3,4,5,6,7 g&p
32.	[20,11,5]	1202000202	t=0
33.	[20,9,6]	121102020102	t=0
34.	[20,8,8]	1002122221122	t=0,1
35.	[20,7,8]	10022121211122	t=0,1
36.	$[20,6,10]^{o}$	1121001010002221	t=0,1,3
37.	[20,6,8]	122110102022112	t=0,1,3
38.	[20,5,11]	1012201212020022	t=0,1,2

39.	$[20,4,12]^o$	11101210002220212	t=0,1,2,3
40.	$[20,1,20]^{o*}$	121212121212121212	t=0,1,2,3,4,5,6,7,8,9

Таблица 4.1.3. Троични негациклични кодове.

№	[n,k,d]	пораждащ полином	подходящ
1.	[6,2,3]	10201	t=0,1
2.	$[10,6,4]^{o*}$	11021	t=0,1
3.	$[10,4,6]^o$	1110121	t=0,1,2
4.	$[12,8,3]^{o*}$	12211	t=0,1
5.	[12,4,6]	122122211	t=0,1,2
6.	$[12,2,9]^o$	12202110122	t=0,1,2,3,4
7.	$[14,8,5]^{o*}$	1101011	t=0,1,2
8.	$[14,6,6]^o$	111202111	t=0,1,2
9.	[20,12,5]	112212211	t=1
10.	[20,10,7]	12201101002	t=0
11.	$[20,10,6]^*$	11121011012	t=0,1,2
12.	[20,8,8]	1122201022211	t=0,1,2
13.	[20,6,9]	120121010211212	t=0,1
14.	[20,4,12]	11021100212101201	t=0,1,2,3
15.	[20,2,15]	1120221011202210112	t=0,1,2,3,4,5,6

Таблица 4.1.4. Двоични кодове с максимално минимално разстояние.

№	[n,k,d]	R	№	[n,k,d]	R	№	[n,k,d]	R
1.	[12,4,6]*	4	31.	[24,7,10]	8	61.	[27,6,12]	10
2.	[16,11,4]*	2	32.	[24,5,12]	10	62.	[27,6,12]	11
3.	[16,7,6]*	4	33.	[25,5,12]	11	63.	[27,6,12]	11
4.	[16,7,6]*	4	34.	[25,5,12]	11	64.	[27,6,12]	11
5.	[16,7,6]	5	35.	[25,5,12]*	10	65.	[28,10,10]	8
6.	[17,5,8]*	6	36.	[25,5,12]*	10	66.	[28,10,10]	8
7.	[17,5,8]	7	37.	[25,5,12]*	10	67.	[28,10,10]	8
8.	[18,9,6]	4	38.	[25,5,12]*	10	68.	[28,10,10]	8
9.	[18,6,8]	7	39.	[25,5,12]*	10	69.	[28,10,10]	8
10.	[18,6,8]	7	40.	[26,6,12]	11	70.	[28,10,10]	8
11.	[19,7,8]	7	41.	[26,6,12]	11	71.	[28,10,10]	8
12.	[20,8,8]	7	42.	[27,7,12]	10	72.	[28,10,10]	8
13.	[22,10,8]	7	43.	[27,6,12]	11	73.	[28,10,10]	8
14.	[23,12,7]*	3	44.	[27,6,12]	11	74.	[28,10,10]	8
15.	[24,12,8]	7	45.	[27,6,12]	11	75.	[28,10,10]	8
16.	[21,8,8]	6	46.	[27,6,12]	11	76.	[28,5,14]	12
17.	[21,8,8]	7	47.	[27,6,12]	10	77.	[29,5,14]*	12
18.	[21,8,8]	7	48.	[27,6,12]	10	78.	[29,5,14]*	12
19.	[21,8,8]	7	49.	[27,6,12]	11	79.	[29,5,14]*	12
20.	[21,8,8]	7	50.	[27,6,12]	10	80.	[29,5,14]*	12
21.	[21,8,8]	7	51.	[27,6,12]	10	81.	[29,5,14]*	12
22.	[21,8,8]	8	52.	[27,6,12]	11	82.	[29,5,14]*	12
23.	[21,8,8]	8	53.	[27,6,12]	11	83.	[29,5,14]*	12
24.	[21,5,10]*	8	54.	[27,6,12]	10	84.	[30,6,14]	11
25.	[24,14,6]	4	55.	[27,6,12]	11	85.	[30,6,14]	11
26.	[24,7,10]	8	56.	[27,6,12]	10	86.	[31,13,9]	7
27.	[24,7,10]	8	57.	[27,6,12]	11	87.	[32,17,8]	6
28.	[24,7,10]	8	58.	[27,6,12]	10	88.	[32,6,16]	12
29.	[24,7,10]	8	59.	[27,6,12]	11	89.	[33,8,14]	11
30.	[24,7,10]	8	60.	[27,6,12]	11	90.	[33,12,11]	9

Приложение Г

Тегловни разпределения на лидерите на съседни класове на троичните циклични кодове с дължини до 14

Таблица 4.2.1

№	n	k	d	Корени	$\bar{\alpha}$
1.	4	3	2	2	$\alpha_1 = 2$
2.	4	2	2	1	$\alpha_1 = 4, \alpha_2 = 4$
3.	4	1	4	0,1	$\alpha_1 = 8, \alpha_2 = 18$
4.	8	7	2	4	$\alpha_1 = 2$
5.	8	6	2	1	$\alpha_1 = 8$
6.	8	6	2	2	$\alpha_1 = 4, \alpha_2 = 4$
7.	8	5	3	0,1	$\alpha_1 = 16, \alpha_2 = 10$
8.	8	5	2	0,2	$\alpha_1 = 8, \alpha_2 = 18$
9.	8	4	4	1,2	$\alpha_1 = 16, \alpha_2 = 60, \alpha_3 = 4$
10.	8	4	2	1,5	$\alpha_1 = 8, \alpha_2 = 24, \alpha_3 = 32, \alpha_4 = 16$
11.	8	3	5	0,1,2	$\alpha_1 = 16, \alpha_2 = 112, \alpha_3 = 108, \alpha_4 = 6$
12.	8	3	4	0,1,5	$\alpha_1 = 16, \alpha_2 = 82, \alpha_3 = 96, \alpha_4 = 48$
13.	8	2	6	0,1,2,4	$\alpha_1 = 16, \alpha_2 = 112, \alpha_3 = 368, \alpha_4 = 216, \alpha_5 = 16$
14.	8	2	4	0,1,4,5	$\alpha_1 = 16, \alpha_2 = 100, \alpha_3 = 288, \alpha_4 = 324$
15.	8	1	8	0,1,2,5	$\alpha_1 = 16, \alpha_2 = 112, \alpha_3 = 448, \alpha_4 = 1050, \alpha_5 = 560$
16.	10	9	2	5	$\alpha_1 = 2$
17.	10	8	2	0,5	$\alpha_1 = 4, \alpha_2 = 4$
18.	10	6	2	1	$\alpha_1 = 10, \alpha_2 = 40, \alpha_3 = 30$
19.	10	5	4	0,1	$\alpha_1 = 20, \alpha_2 = 132, \alpha_3 = 90$
20.	10	5	2	0,2	$\alpha_1 = 10, \alpha_2 = 40, \alpha_3 = 80, \alpha_4 = 80, \alpha_5 = 32$
21.	10	4	4	0,1,5	$\alpha_1 = 20, \alpha_2 = 132, \alpha_3 = 240, \alpha_4 = 240,$ $\alpha_5 = 96$
22.	10	2	5	1,2	$\alpha_1 = 20, \alpha_2 = 180, \alpha_3 = 860, \alpha_4 = 2200, \alpha_5 = 2400,$ $\alpha_6 = 900$
23.	10	1	10	0,1,2	$\alpha_1 = 20, \alpha_2 = 180, \alpha_3 = 960, \alpha_4 = 3360, \alpha_5 = 7812,$ $\alpha_6 = 7350$
24.	11	6	5	1	$\alpha_1 = 22, \alpha_2 = 220$

25.	11	5	6	0,1	$\alpha_1 = 22, \alpha_2 = 220, \alpha_3 = 440, \alpha_4 = 44, \alpha_5 = 2$
26.	11	1	11	1,2	$\alpha_1 = 22, \alpha_2 = 220, \alpha_3 = 1320, \alpha_4 = 5280, \alpha_5 = 14784,$ $\alpha_6 = 25872, \alpha_7 = 11550$
27.	13	10	3	1	$\alpha_1 = 26$
28.	13	9	3	0,1	$\alpha_1 = 26, \alpha_2 = 52, \alpha_3 = 2$
29.	13	7	5	1,4	$\alpha_1 = 41, \alpha_2 = 362, \alpha_3 = 324$
30.	13	7	4	1,2	$\alpha_1 = 41, \alpha_2 = 302, \alpha_3 = 384$
31.	13	6	6	0,1,4	$\alpha_1 = 29, \alpha_2 = 348, \alpha_3 = 1274, \alpha_4 = 32, \alpha_5 = 3$
32.	13	6	6	0,1,2	$\alpha_1 = 29, \alpha_2 = 352, \alpha_3 = 1432, \alpha_4 = 373$
33.	13	4	7	1,2,4	$\alpha_1 = 26, \alpha_2 = 312, \alpha_3 = 2288, \alpha_4 = 8788, \alpha_5 = 8060,$ $\alpha_6 = 208$
34.	13	3	9	0,1,2,4	$\alpha_1 = 26, \alpha_2 = 312, \alpha_3 = 2288, \alpha_4 = 11440, \alpha_5 = 30342,$ $\alpha_6 = 14352, \alpha_7 = 288$
35.	13	1	13	1,2,4,7	$\alpha_1 = 26, \alpha_2 = 312, \alpha_3 = 2288, \alpha_4 = 11440, \alpha_5 = 41184,$ $\alpha_6 = 109824, \alpha_7 = 204204, \alpha_8 = 162162$
36.	14	13	2	7	$\alpha_1 = 2$
37.	14	12	2	0,7	$\alpha_1 = 4, \alpha_2 = 4$
38.	14	8	2	1	$\alpha_1 = 14, \alpha_2 = 84, \alpha_3 = 280, \alpha_4 = 350$
39.	14	7	4	0,1	$\alpha_1 = 30, \alpha_2 = 300, \alpha_3 = 1015, \alpha_4 = 841$
40.	14	7	2	0,2	$\alpha_1 = 14, \alpha_2 = 84, \alpha_3 = 280, \alpha_4 = 560, \alpha_5 = 672$ $\alpha_6 = 448, \alpha_7 = 128$
41.	14	6	4	0,1,7	$\alpha_1 = 44, \alpha_2 = 343, \alpha_3 = 1102, \alpha_4 = 1930, \alpha_5 = 1935,$ $\alpha_6 = 1003, \alpha_7 = 202$
42.	14	2	7	1,2	$\alpha_1 = 28, \alpha_2 = 364, \alpha_3 = 2912, \alpha_4 = 15596, \alpha_5 = 56840,$ $\alpha_6 = 137200, \alpha_7 = 196000, \alpha_8 = 122500$
43.	14	1	14	0,1,2	$\alpha_1 = 28, \alpha_2 = 364, \alpha_3 = 2912, \alpha_4 = 16016, \alpha_5 = 64064,$ $\alpha_6 = 192192, \alpha_7 = 435864, \alpha_8 = 630630, \alpha_9 = 252252$

Приложение Д

CRC полиноми от 8-ма степен.

Таблица 5.2.1. Неразложими полиноми от 8-ма степен.

Коеф. на $g(x)$	Ред	Монот.	Добър	Подх.	Минимално разстояние	Радиус на покритие
111111001	85	10..85	10..85	10..69	4,10..28;3,29..85	5,10;4,11..17; 3,18..37;2,38..85
101110111	85	10..85	10..85	10..69	6,10;5,11;3,12..85	4,10..19;3,20 ..37; 2,38..88
110011111	51	10..51	10..51	10..51	4,10..34;3,34..51	5,10..14;4,15 ..20; 3,21..34;2,35..51
111001111	255	10..127	10..127	10..69	4,10..28;3,29..127	6,10..12;4,13 ..17; 3,18..45;2,46..127
100101011	255	10..89	10..80	10..69	5,10..14;4,14..34; 3,35..127	5,10;4,11..17 ; 3,18..42;2,43..127
110110001	51	10..51	10..51	10..51	5,10..14;4,14..34; 3,35..51	5,10..1;4,12 ..16; 3,17..36;2,37..51
101100101	255	10..127	10..127	10..69	5,10..14;4,15..23; 3,24..127	5,10;4,11..16 ; 3,17..44;2,45..127
100101101	85	10..85	10..85	10..69	5,10..12;4,12..16; 3,17..85	5,10;4,11..15 ; 3,16..37;2,38..85
110111101	85	10..85	10..85	10..69	6,10;5,11..13; 4,14..16;3,17..85	4,10..17;3,18..33 ; 2,34..85
110101001	255	10..127	10..127	10..69	5,10..13;3,14..127	5,10;4,11..17 ; 3,18..42;2,43..127
110000111	255	13..127	13..127	13..69	4,10..27;3,28..127	6,10..11;5,12 ;4,13..16; 3,17..35;2,36..127
100011101	255	10..127	10..127	10..69	5,10..13;4,14..21; 3,22..127	5,10..11;4,12 ..15; 3,16..45;2,46..127
111110101	255	10..127	10..127	10..69	5,10..13;4,14..21; 3,22..127	4,10..21; 3,22..45;2,46..127
101101001	255	10..127	10..127	10..69	5,10..11;4,12..16; 3,17..127	5,10;4,11..15 ; 3,16..37;2,38..127
110001101	255	10..127	10..127	10..69	5,10..12;4,13..20; 3,21..127	5,10..11;4,12 ..16; 3,17..34;2,35..127
110100011	85	10..85	10..85	10..69	5,10..12;4,13..33; 3,34..85	5,10..11;4,12 ..15; 3,16..39;2,40..85

Таблица 5.2.2. Неразложими полиноми от 7-ма степен умножени по $x+1$.
Редът на всички полиноми е 127.

Коеф. на $g(x)$	Монот.	Добър	Подх.	Минимално разстояние	Радиус на покритие
100000111	14..127	14..127	15..69	4,10..127	7,10;6,11..12;5,13..15; 4,16..32;3,33..127
101110101	10..127	10..127	10..69	6,10;4,11..127	5,10..21;4,22..33;3,34 ..127
111111101	10..127	10..127	10..69	4,10..127	5,10..19;4,20..30;3,31..127
100101111	10..127	10..127	10..69	6,10..11;4,12..127	5,10..14;4,15..24;2 ,25..127
100110001	11..127	11..127	11..69	4,10..127	6,10;5,11..19;4,20..29; 3,30..127
110011011	10..127	10..127	10..69	6,10;4,10..127	5,10..14;4,15..27;3,28 ..127
101000011	13..127	13..127	13..69	4,10..127	6,10..11;5,12..14;4,15..26; 3,27..127
111001011	10..127	10..127	10..69	6,10..11;4,10..127	5,10..12;4,13..22;3 ,23..127
100010011	11..17, 20..127	11..17, 20..127	11..16, 20..69	4,10..127	6,10;5,11..19; 4,20..28;3,29..127

Таблица 5.2.3. Неразложими полиноми от 6-та степен умножени по $(x+1)^2$.
Редът на всички полиноми е 63.

Коеф. на $g(x)$	Монот.	Добър	Подх.	Минимално разстояние	Радиус на покритие
110111111	10..63	10..63	10..63	6,10..12;4,13..63	5,10..12;4,13..63
111011001	10..63	10..63	10..63	6,10..12;4,13..63	5,10..11;4,12..63
101001111	10..63	10..63	10..63	4,10..63	5,10..11;4,12..63

Таблица 5.2.4. Неразложими полиноми от 6-та степен умножени по x^2+x+1.
Редът на всички полиноми е 63.

Коеф. на $g(x)$	Монот.	Добър	Подх.	Минимално разстояние	Радиус на покритие
101011001	10..63	10..63	10..63	5,10..13;4,14..17;3,18..63	5,10;4,11..17 ;3,18..63
100000011	51..63	51..63	51..63	3,10..63	7,10;6,11..12;5,13..14; 4,15..26;3,27..63
111001001	10..63	10..63	10..63	5,10..12;4,13..25;3,26..63	5,10;4,11..16 ;3,17..63

Приложение E

Стандартизирани CRC кодове и кодове с най-малки стойности на функцията P_{ue} с 16 проверочни бита.

Таблица 5.3.1.

Коефициенти на пол.	Най-добър за	Ред	Монот.	Добър	Подх.	Минимално разстояние
11021^{CCITT}	-	32767	252..1024	-	-	4,18..1024
14003^{ANSI}	-	32767	-	-	-	4,18..1024
$1A097^{IBM}$	-	32766	22..352	22..65	22..65	4,84..1024; 6,25..83;8,18..24
$16F63^{IEEE}$	254,255	255	18..255	18..65	18..25 28..65	5,31..255;6,30; 7,19..29,10,18
$15B93^{IEC}$	-	254	18..128	18..65	18..65	6,26..128; 8,20..25;10,18.19
$13D65^{CGG,C_1}$	19, 148..151	32767	18..151	18..65	18..65	2,152..1024;6,23..151; 8,21..22;10,18..20
$1F29F^{CGG,C_2}$	-	32766	18..42 49..189	18..42 49..65	18..42 49..65	2,259..1024;4,131..258; 6,23..130;8,18..22
$15935^{CGG,C_3}$	256,257	65535	18..258	18..65	18..65	2,258..1024;5,52..257 ; 6,27..51;≥ 7, $N \leq 26$
$1011B^{CGG,C_4}$	-	28658	29..30 59..1024	29..30 59..65	29..30 59..65	4,116..1024; 6,18..115
$1A2EB^{CGG,C_5}$	-	32767	18..1024	18..65	18..65	4,110..1024;6,28..109; 8,19..27;10,18
1EDF3	935..1024	65535	18..35, 176..1024	18..35	18..35	4,34..1024;6, 28..33 7,23..27;8,18..22
10595	884..934	57337	22..60,64,	22..59	22.59 67..1024	4,109..1024;5, 39..108; 6,26..38;7,18..25
1FAC3	818..883	57337	18..1024	18..65	18..65	3,879..1024;4, 102..878; 5,32..101;$\geq 6, N \leq 31$
1FC9F	728..817 707..720	65535	24..1024	24..65	27..65	3,818..1024;4 ,78..817; 5,38..77;6,18..37
1E667	721..727	57337	18..21, 30..1024	18..21, 30..65	18..21, 30..65	3,728.. 1024;4,90..727 5,46..89;6 ,20..45;8,18..19
1BE8D	694..706	63457	18..1024	18..65	18..65	3,705.. 1024;4,72..704 4,44..71;$\geq 6, N \leq 43$

19E91	682..693	65535	18..1024	18..65	18..65	3,700..1024;4, 124..699; 5,50..123;$\geq 6, N \leq 49$
16087	595..681	16087	22..46, 55..1024	22..46 58..65	22..46 59..65	3,680..1024;4,64..679; 5,35..63;6,31..34;7,18..30
1941F	529..594 362..503	57337	22..1024	22..65	22..27, 30..65	3,756..1024 ;4,89..755; 5,35..88;6, 23..34;8,18..22
131A1	504..528	57337	20..1024	20..65	20..65	3,529..1024 ;4,106..528; 5,40..105;$\geq 6, N \leq 39$
1F737	327..361	64897	18..512	18..65	18..65	3,364..512;4, 103..363; 5,39..102;$\geq 6, N \leq 38$
19E91	326	65535	18..1024	18..65	18..65	3,700..1024;4,124 ..699; 5,50..123;$\geq 6, N \leq 49$
1A801	310..325	65535	69..1024	-	-	3,684..1024;4,105..683; 5,18..104;
1B6CB	258..309	4095	18, 47..1024	18, 48..65	18, 48..65	3,310..1024;4,101..309 5,19..100;11,18
1F6ED	256..257	64897	18..27, 77..258	18..27	18..27	2,258..1024;5,28.. 257; 6,20..27;8,19;10,18
1333F	236..251 152..165	255	18..22, 26..255	18..22, 26..65	18..22, 26..65	5,34..255; 6,20..33;8,18..19
1989D	211..235	255	18..42, 58..83, 99..255	18..41, 60..65	18..41, 60..65	5,28..255;6,26..27 ; 7,25;8,21..24; 9,18..20;
19ED5	176..210	255	18..255	18..65	18..65	5,46..255;6,25..45; 7,23..24; $\geq 8, N \leq 22$
12109	166..175	65535	66..258	-	-	2,258..1024; 5,18..257
190D9	131..151	32767	19..22, 29..151	19..22, 30..65	19..21, 30..65	6,22..151;8,18..21
11c71	129..130	32766	20..21, 31..188	20..21, 31..65	20, 32..65	2,259..1024;4,131..258 ; 6,21..130;8,18..20
184D5	117...128	254	18..128	18..65	18..65	6,25..128;8,18..24
1C447	101..116	254	20..21, 28..128	20..21, 29..65	20, 30..65	6,21..128; 8,18..20
14825	19..100	32766	26..189	26..65	2,259..1024;	4,131..258;6,18..130

Приложение Ж

CRC полиноми с до 5 проверочни бита.

Таблица 5.4.2.

№	n	d	A_d	$\{B_i\}$	$\{\alpha_i\}$
3	7	3	7	0,1 3,7	0,1 1,7
	6	3	4	0,1 3,4 4,3	0,1 1,6 2,1
	5	3	2	0,1 2,2 3,4 4,1	0,1 1,5 2,2
	4	3	1	0,1 1,1 2,3 3,3	0,1 1,4 2,3
4	4	1	1	0,1 2,6 4,1	0,1 1,4 2,3
5	15	3	35	0,1 8,15	0,1 1,15
	14	3	28	0,1 7,8 8,7	0,1 1,14 2,1
	13	3	22	0,1 6,4 7,8 8,3	0,1 1,13 2,2
	12	3	17	0,1 5,2 6,6 7,6 8,1	0,1 1,12 2,3
	11	3	13	0,1 4,1 5,4 6,6 7,4	0,1 1,11 2,4
	10	3	9	0,1 4,3 5,6 6,4 7,2	0,1 1,10 2,5
	9	3	6	0,1 3,1 4,5 5,6 6,2 7,1	0,1 1,9 2,6
	8	3	4	0,1 3,4 4,5 5,4 6,2	0,1 1,8 2,7
	7	3	3	0,1 2,2 3,4 4,5 5,4	0,1 1,7 2,8
	6	3	2	0,1 1,1 2,2 3,6 4,5 5,1	0,1 1,6 2,7 3,2
	5	3	1	0,1 1,2 2,4 3,6 4,3	0,1 1,5 2,7 3,3
6	7	4	7	0,1 3,7 4,7 7,1	0,1 1,7 2,7 3,1
	6	4	3	0,1 2,3 3,8 4,3 6,1	0,1 1,6 2,7 3,2
	5	4	1	0,1 1,1 2,6 3,6 4,1 5,1	0,1 1,5 2,7 3,3
7	6	4	3	0,1 2,3 3,8 4,3 6,1	0,1 1,6 2,7 3,2
	5	4	1	0,1 1,1 2,6 3,6 4,1 5,1	0,1 1,5 2,7 3,3
8	6	3	2	0,1 2,6 4,9	0,1 1,6 2,9
	5	3	1	0,1 1,2 2,4 3,6 4,3	0,1 1,5 2,7 3,3
9	5	5	1	0,1 2,10 4,5	0,1 1,5 2,10
10	31	3	155	0,1 16,31	0,1 1,31
	30	3	140	0,1 15,16 16,15	0,1 1,30 2,1
	29	3	126	0,1 14,8 15,16 16,7	0,1 1,29 2,2

	28	3	113	0,1 13,4 14,12 15,12 16,3	0,1 1,28 2,3
	27	3	101	0,1 12,2 13,8 14,12 15,8 16,1	0,1 1,27 2,4
	26	3	90	0,1 11,1 12,5 13,10 14,10 15,5	0,1 1,26 2,5
	25	3	80	0,1 11,5 12,5 13,10 14,10 15,1	0,1 1,25 2,6
	24	3	71	0,1 10,3 11,4 12,7 13,12 14,5	0,1 1,24 2,7
	23	3	63	0,1 9,2 10,3 11,4 12,11 13,10 14,1	0,1 1,23 2,8
	22	3	55	0,1 8,1 9,2 10,4 11,8 12,10 13,6	0,1 1,22 2,9
	21	3	47	0,1 8,2 9,3 10,6 11,10 12,7 13,3	0,1 1,21 2,10
	20	3	40	0,1 8,4 9,5 10,6 11,10 12,5 13,1	0,1 1,20 2,11
	19	3	34	0,1 7,1 8,6 9,6 10,6 11,9 12,3	0,1 1,19 2,12
	18	3	28	0,1 7,3 8,8 9,6 10,6 11,7 12,1	0,1 1,18 2,13
	17	3	23	0,1 7,8 8,7 9,4 10,8 11,4	0,1 1,17 2,14
	16	3	19	0,1 6,3 7,9 8,5 9,6 10,7 11,1	0,1 1,16 2,15
	15	3	16	0,1 5,1 6,7 7,6 8,5 9,9 10,3	0,1 1,15 2,16
	14	3	13	0,1 5,4 6,8 7,4 8,7 9,8	0,1 1,14 2,17
	13	3	10	0,1 4,1 5,7 6,6 7,6 8,8 9,3	0,1 1,13 2,18
	12	3	7	0,1 4,3 5,9 6,6 7,6 8,6 9,1	0,1 1,12 2,19
	11	3	5	0,1 3,1 4,5 5,10 6,6 7,5 8,4	0,1 1,11 2,20
	10	3	3	0,1 3,3 4,7 5,10 6,6 7,3 8,2	0,1 1,10 2,20 3,1
	9	3	1	0,1 3,6 4,10 5,8 6,4 7,2 8,1	0,1 1,9 2,20 3,2
	8	4	3	0,1 2,1 3,10 4,11 5,4 6,3 7,2	0,1 1,8 2,20 3,3
	7	4	1	0,1 2,5 3,12 4,7 5,4 6,3	0,1 1,7 2,18 3,6
	6	5	1	0,1 1,1 2,10 3,10 4,5 5,5	0,1 1,6 2,15 3,10
11	31	3	155	0,1 16,31	0,1 1,31
	30	3	140	0,1 15,16 16,15	0,1 1,30 2,1
	29	3	126	0,1 14,8 15,16 16,7	0,1 1,29 2,2
	28	3	113	0,1 13,4 14,12 15,12 16,3	0,1 1,28 2,3
	27	3	101	0,1 12,2 13,8 14,12 15,8 16,1	0,1 1,27 2,4
	26	3	90	0,1 11,1 12,5 13,10 14,10 15,5	0,1 1,26 2,5
	25	3	80	0,1 11,5 12,5 13,10 14,10 15,1	0,1 1,25 2,6
	24	3	71	0,1 10,3 11,4 12,7 13,12 14,5	0,1 1,24 2,7
	23	3	62	0,1 9,1 10,4 11,6 12,9 13,9 14,2	0,1 1,23 2,8
	22	3	54	0,1 9,3 10,6 11,6 12,9 13,7	0,1 1,22 2,9
	21	3	47	0,1 8,1 9,6 10,4 11,8 12,10 13,2	0,1 1,21 2,10
	20	3	41	0,1 8,5 9,4 10,4 11,12 12,6	0,1 1,20 2,11
	19	3	35	0,1 7,2 8,5 9,4 10,8 11,10 12,2	0,1 1,19 2,12
	18	3	30	0,1 6,1 7,3 8,5 9,6 10,9 11,7	0,1 1,18 2,13
	17	3	25	0,1 6,3 7,3 8,5 9,10 10,7 11,3	0,1 1,17 2,14
	16	3	20	0,1 5,1 6,3 7,4 8,9 9,9 10,3 11,2	0,1 1,16 2,15
	15	3	15	0,1 5,2 6,3 7,9 8,9 9,4 10,3 11,1	0,1 1,15 2,16
	14	3	11	0,1 5,3 6,7 7,10 8,5 9,3 10,3	0,1 1,14 2,17
	13	3	8	0,1 5,7 6,9 7,6 8,5 9,3 10,1	0,1 1,13 2,18

	12	3	6	0,1 4,2 5,10 6,8 7,4 8,5 9,2	0,1 1,12 2,19
	11	3	4	0,1 4,6 5,12 6,4 7,4 8,5	0,1 1,11 2,20
	10	3	3	0,1 3,2 4,10 5,8 6,4 7,6 8,1	0,1 1,10 2,21
	9	3	2	0,1 3,7 4,9 5,6 6,6 7,3	0,1 1,9 2,21 3,1
	8	3	1	0,1 2,2 3,9 4,9 5,6 6,4 7,1	0,1 1,8 2,19 3,4
	7	4	1	0,1 2,5 3,12 4,7 5,4 6,3	0,1 1,7 2,18 3,6
	6	5	1	0,1 1,1 2,10 3,10 4,5 5,5	0,1 1,6 2,15 3,10
12	31	3	155	0,1 16,31	0,1 1,31
	30	3	140	0,1 15,16 16,15	0,1 1,30 2,1
	29	3	126	0,1 14,8 15,16 16,7	0,1 1,29 2,2
	28	3	113	0,1 13,4 14,12 15,12 16,3	0,1 1,28 2,3
	27	3	101	0,1 12,2 13,8 14,12 15,8 16,1	0,1 1,27 2,4
	26	3	90	0,1 11,1 12,5 13,10 14,10 15,5	0,1 1,26 2,5
	25	3	79	0,1 11,3 12,9 13,10 14,6 15,3	0,1 1,25 2,6
	24	3	69	0,1 11,9 12,9 13,6 14,6 15,1	0,1 1,24 2,7
	23	3	60	0,1 10,3 11,12 12,7 13,4 14,5	0,1 1,23 2,8
	22	3	52	0,1 9,1 10,8 11,10 12,5 13,5 14,2	0,1 1,22 2,9
	21	3	45	0,1 9,5 10,10 11,6 12,5 13,5	0,1 1,21 2,10
	20	3	38	0,1 8,1 9,10 10,8 11,4 12,6 13,2	0,1 1,20 2,11
	19	3	32	0,1 8,5 9,12 10,4 11,4 12,6	0,1 1,19 2,12
	18	3	27	0,1 7,2 8,9 9,8 10,4 11,6 12,2	0,1 1,18 2,13
	17	3	23	0,1 6,1 7,5 8,9 9,6 10,5 11,5	0,1 1,17 2,14
	16	3	19	0,1 6,3 7,9 8,5 9,6 10,7 11,1	0,1 1,16 2,15
	15	3	16	0,1 5,1 6,7 7,6 8,5 9,9 10,3	0,1 1,15 2,16
	14	3	13	0,1 5,5 6,5 7,6 8,9 9,5 10,1	0,1 1,14 2,17
	13	3	11	0,1 4,2 5,6 6,4 7,8 8,9 9,2	0,1 1,13 2,18
	12	3	9	0,1 4,6 5,4 6,4 7,12 8,5	0,1 1,12 2,19
	11	3	7	0,1 3,2 4,6 5,4 6,8 7,10 8,1	0,1 1,11 2,20
	10	3	5	0,1 3,5 4,5 5,6 6,10 7,5	0,1 1,10 2,20 3,1
	9	3	4	0,1 2,2 3,5 4,5 5,10 6,8 7,1	0,1 1,9 2,18 3,4
	8	3	3	0,1 2,5 3,4 4,7 5,12 6,3	0,1 1,8 2,17 3,6
	7	3	2	0,1 1,1 2,6 3,6 4,9 5,9	0,1 1,7 2,15 3,9
	6	3	1	0,1 1,3 2,6 3,10 4,9 5,3	0,1 1,6 2,12 3,10 4,3
13	21	3	42	0,1 10,21 12,7 14,3	0,1 1,21 2,10
	20	3	36	0,1 9,10 10,11 11,4 12,3 13,2 14,1	0,1 1,20 2,11
	19	3	31	0,1 8,5 9,10 10,8 11,4 12,2 13,2	0,1 1,19 2,12
	18	3	27	0,1 7,3 8,6 9,10 10,6 11,3 12,3	0,1 1,18 2,13
	17	3	23	0,1 6,2 7,3 8,8 9,10 10,4 11,3 12,1	0,1 1,17 2,14
	16	3	20	0,1 5,1 6,3 7,4 8,9 9,9 10,3 11,2	0,1 1,16 2,15
	15	3	17	0,1 5,3 6,3 7,6 8,9 9,7 10,3	0,1 1,15 2,16
	14	3	14	0,1 4,1 5,4 6,3 7,8 8,10 9,4 10,1	0,1 1,14 2,17
	13	3	11	0,1 4,3 5,3 6,6 7,10 8,6 9,3	0,1 1,13 2,18

	12	3	9	0,1 3,1 4,3 5,6 6,6 7,9 8,6	0,1 1,12 2,19
	11	3	7	0,1 3,2 4,6 5,4 6,8 7,10 8,1	0,1 1,11 2,20
	10	3	5	0,1 3,5 4,5 5,6 6,10 7,5	0,1 1,10 2,20 3,1
	9	3	4	0,1 2,2 3,5 4,5 5,10 6,8 7,1	0,1 1,9 2,18 3,4
	8	3	3	0,1 1,1 2,2 3,6 4,9 5,9 6,4	0,1 1,8 2,15 3,8
	7	3	2	0,1 1,2 2,3 3,8 4,11 5,6 6,1	0,1 1,7 2,13 3,9 4,2
	6	3	1	0,1 1,3 2,6 3,10 4,9 5,3	0,1 1,6 2,12 3,10 4,3
14	15	4	105	0,1 7,15 8,15 15,1	0,1 1,15 2,15 3,1
	14	4	77	0,1 6,7 7,16 8,7 14,1	0,1 1,14 2,15 3,2
	13	4	55	0,1 5,3 6,12 7,12 8,3 13,1	0,1 1,13 2,15 3,3
	12	4	38	0,1 4,1 5,8 6,12 7,8 8,1 12,1	0,1 1,12 2,15 3,4
	11	4	25	0,1 4,5 5,10 6,10 7,5 11,1	0,1 1,11 2,15 3,5
	10	4	16	0,1 3,2 4,7 5,12 6,7 7,2 10,1	0,1 1,10 2,15 3,6
	9	4	10	0,1 2,1 3,3 4,11 5,11 6,3 7,1 9,1	0,1 1,9 2,15 3,7
	8	4	5	0,1 2,2 3,8 4,10 5,8 6,2 8,1	0,1 1,8 2,15 3,8
	7	4	2	0,1 2,6 3,9 4,9 5,6 7,1	0,1 1,7 2,15 3,9
	6	4	1	0,1 1,2 2,7 3,12 4,7 5,2 6,1	0,1 1,6 2,12 3,10 4,3
15	14	4	77	0,1 6,7 7,16 8,7 14,1	0,1 1,14 2,15 3,2
	13	4	55	0,1 5,3 6,12 7,12 8,3 13,1	0,1 1,13 2,15 3,3
	12	4	38	0,1 4,1 5,8 6,12 7,8 8,1 12,1	0,1 1,12 2,15 3,4
	11	4	25	0,1 4,5 5,10 6,10 7,5 11,1	0,1 1,11 2,15 3,5
	10	4	16	0,1 3,2 4,7 5,12 6,7 7,2 10,1	0,1 1,10 2,15 3,6
	9	4	9	0,1 3,6 4,9 5,9 6,6 9,1	0,1 1,9 2,15 3,7
	8	4	5	0,1 2,2 3,8 4,10 5,8 6,2 8,1	0,1 1,8 2,15 3,8
	7	4	3	0,1 1,1 2,3 3,11 4,11 5,3 6,1 7,1	0,1 1,7 2,13 3,9 4,2
	6	4	1	0,1 1,2 2,7 3,12 4,7 5,2 6,1	0,1 1,6 2,12 3,10 4,3
16	12	4	39	0,1 4,3 6,24 8,3 12,1	0,1 1,12 2,15 3,4
	11	4	26	0,1 3,1 4,2 5,12 6,12 7,2 8,1 11,1	0,1 1,11 2,15 3,5
	10	4	16	0,1 3,2 4,7 5,12 6,7 7,2 10,1	0,1 1,10 2,15 3,6
	9	4	9	0,1 3,6 4,9 5,9 6,6 9,1	0,1 1,9 2,15 3,7
	8	4	5	0,1 2,2 3,8 4,10 5,8 6,2 8,1	0,1 1,8 2,15 3,8
	7	4	2	0,1 2,6 3,9 4,9 5,6 7,1	0,1 1,7 2,15 3,9
	6	4	1	0,1 1,2 2,7 3,12 4,7 5,2 6,1	0,1 1,6 2,12 3,10 4,3
17	8	4	6	0,1 2,4 4,22 6,4 8,1	0,1 1,8 2,13 3,8 4,2
	7	4	3	0,1 1,1 2,3 3,11 4,11 5,3 6,1 7,1	0,1 1,7 2,13 3,9 4,2
	6	4	3	0,1 1,2 2,7 3,12 4,7 5,2 6,1	0,1 1,6 2,12 3,10 4,3
18	6	6	1	0,1 2,15 4,15 6,1	0,1 1,6 2,15 3,10

Приложение 3

Оптимални $(v, 4, 2, 1)$ и $(v, 5, 2, 1)$ оптични ортогонални кодове

Таблица 7.1.1 Класификация на оптимални $(v, 4, 2, 1)$ оптични ортогонални кодове с $v \leq 75$

v	s_O	s	всички	съвършени	кодови думи на един ООС							
16	2	1	20	0	1 2 9							
17	2	2	1	1	1 4 5	2 8 10						
18	2	1	30	0	1 2 10							
19	2	2	1	0	1 4 5	2 8 10						
20	2	2	10	1	1 5 6	2 10 13						
21	2	2	7	0	1 3 19	4 10 14						
22	2	2	23	0	1 2 12	3 8 17						
23	2	2	19	0	1 2 4	5 11 16						
24	3	2	113	0	1 2 13	4 9 14						
25	3	3	1	1	1 4 22	2 10 12	5 11 16					
26	3	3	5	0	1 2 14	3 7 10	5 11 20					
27	3	2	192	0	1 2 4	5 10 20						
28	3	3	44	5	2 8 22	3 12 19	1 11 24					
29	3	3	21	0	1 2 4	5 10 17	6 14 20					
30	3	3	156	0	1 2 16	3 7 10	5 11 24					
31	3	3	119	0	1 2 4	5 10 18	6 15 22					
32	4	3	642	2	2 8 10	1 4 17	5 14 25					
33	4	3	585	4	1 4 30	5 13 24	2 12 18					
34	4	4	21	4	1 9 26	3 15 22	4 14 28	2 13 18				
35	4	4	28	7	1 2 4	6 12 24	7 15 22	5 10 26				
36	4	4	72	18	1 17 18	7 15 28	4 14 26	2 27 33				
37	4	4	155	13	1 3 35	7 17 24	6 15 21	4 12 23				
38	4	4	1467	68	1 2 20	4 13 17	3 6 30	5 15 31				
39	4	4	797	86	1 2 4	5 14 19	6 12 22	7 15 28				
40	5	5	11	11	1 2 21	3 6 12	4 14 18	5 16 29	7 15 32			
41	5	5	3	3	1 3 39	4 14 31	6 19 25	7 15 33	9 20 29			
42	5	5	139	26	6 12 24	1 2 22	3 7 38	8 16 25	5 15 28			
43	5	5	107	30	1 2 4	7 19 26	6 21 27	9 20 29	5 10 35			
44	5	5	1938	377	1 2 23	5 10 20	8 19 27	6 18 32	3 31 40			
45	5	5	624	158	1 2 4	8 16 32	6 20 31	9 19 28	5 12 27			
46	5	5	16962	1550	1 2 24	3 6 12	8 16 35	5 20 33	4 29 36			
47	5	5	12214	1277	1 2 4	5 10 20	9 21 30	6 25 39	7 18 31			
48	6	5	113629	4674	1 2 25	7 14 28	6 19 38	4 22 37	3 8 39			
49	6	6	96	82	7 14 28	1 2 4	5 18 31	6 22 33	8 20 37	9 24 39		
50	6	6	2447	1130	1 2 26	3 6 12	4 19 23	5 16 21	7 20 37	8 18 36		

51	6	5	514731	10504	1 2 4	5 10 20	8 29 38	6 18 32	7 24 35					
52	6	6	18853	4748	1 2 27	3 6 12	5 19 33	10 20 41	4 22 39	7 36 44				
53	6	6	15030	4020	1 2 4	5 10 20	6 12 24	7 16 23	11 22 39	8 27 40				
54	6	6	214583	25000	1 2 28	5 10 20	3 19 35	9 23 40	4 12 33	6 13 24				
55	6	6	169834	23774	1 2 4	6 12 24	13 27 40	10 21 44	5 30 38	7 26 46				
56	7	7	811	663	8 16 32	1 2 29	3 12 15	7 18 25	5 19 42	6 26 36	4 17 39			
57	7	6	1279381	79334	1 2 4	5 10 20	8 16 32	7 18 46	9 23 35	6 19 36				
58	7	7	10271	3903	1 2 30	3 6 12	11 22 44	7 26 39	5 20 43	8 24 42	4 31 41			
59	7	7	7932	3291	1 2 4	5 10 20	7 14 28	6 24 30	9 26 42	8 27 40	11 22 47			
60	7	7	289139	60646	1 2 31	7 14 28	11 22 44	8 18 26	3 6 12	5 20 25	4 23 47			
61	7	7	122586	29927	1 2 4	5 10 20	7 14 28	12 24 48	8 19 50	6 29 45	9 26 43			
62	7	7	1672477	193886	1 2 32	3 6 12	7 14 28	10 23 36	5 25 42	8 24 43	4 22 51			
63	8	8	2823	2823	9 18 36	1 2 4	5 10 20	6 23 29	7 14 28	8 16 32	11 22 41	12 25 37		
64	8	8	2354	2354	1 2 33	3 6 12	4 14 18	5 24 45	7 27 34	8 23 49	11 28 39	13 29 42		
65	8	8	2610	2610	1 2 4	5 10 20	6 19 25	7 28 35	8 26 34	9 23 32	11 22 38	12 24 41		
66	8	8	238215	89236	1 2 34	5 10 20	7 14 28	4 8 39	9 26 49	3 16 53	6 12 24	11 22 47		
67	8	8	59871	24721	1 2 4	5 10 20	7 14 28	11 22 44	8 27 35	13 29 42	9 26 50	6 37 55		
68	8	8	1364771	304507	1 2 35	3 6 12	5 10 20	7 14 28	11 27 52	8 31 39	13 30 49	4 26 50		
69	8	8	2365589	540319	1 2 4	5 10 20	7 14 28	11 22 44	9 26 43	6 18 57	8 27 46	13 37 53		
70	9	9	1417	1417	10 20 40	1 2 36	3 6 12	4 18 56	5 26 49	7 22 55	8 27 51	11 28 39	13 29 54	
71	8	8	$\geq$6116889	$\geq$744092	1 8 9	2 16 18	3 24 27	6 17 60	12 32 51	4 33 37	13 26 48	5 15 46		
72	9	9	86028	86028	1 2 37	3 6 12	4 8 38	5 28 49	7 20 27	10 29 53	11 33 50	14 32 46	15 31 56	
73	9	9	17021	17021	1 2 4	5 43 48	12 27 39	23 40 56	24 42 55	26 45 54	37 44 66	41 52 62	53 59 67	
74	9	9	740033	287113	1 12 13	9 34 43	8 27 55	3 20 23	7 22 29	2 35 41	4 30 48	10 24 60	5 37 58	
75	9	9	1944427	774968	1 11 65	2 22 55	5 13 18	4 27 52	7 31 38	15 34 49	12 28 40	6 36 45	3 32 61	

Таблица 7.1.2 Примери на оптимални $(v, 4, 2, 1)$ оптични ортогонални кодове

v	s_b	s	p	кодови думи							
76	9	9		1 2 39	3 24 27	4 22 58	9 32 41	10 15 20	11 19 30	13 29 60	14 42 48
				17 43 50							
77	9	9		11 22 44	1 52 53	2 15 64	3 9 74	4 14 18	5 32 37	7 36 43	8 38 46
				16 35 58							
78	9	9		1 2 40	3 20 23	4 47 51	6 24 30	7 12 19	8 21 29	9 25 34	10 32 42
				11 26 37							
79	9	9		1 2 4	5 24 29	6 21 64	7 20 27	8 17 25	10 28 38	11 37 48	12 34 46
				14 30 44							
80	10	10	$\surd$	1 2 41	3 6 74	4 26 58	5 23 28	13 34 59	14 24 38	15 30 50	16 33 49
				29 36 73	53 61 72						
81	10	10	$\surd$	1 2 4	5 19 24	6 39 45	7 58 65	17 46 63	21 32 70	25 40 66	27 37 71
				50 59 72	53 61 73						
82	10	10		1 2 42	3 72 75	4 25 29	5 38 43	6 22 28	8 26 34	9 36 55	11 35 58
				12 32 62	14 31 45						
83	10	10		1 2 4	5 13 18	6 31 58	7 26 45	10 43 53	11 39 55	12 24 48	14 23 37
				15 32 49	20 41 61						
84	10	10		12 24 48	25 42 59	1 22 63	2 29 57	3 10 13	4 9 79	6 38 44	8 31 39
				11 26 37	16 35 51						
85	10	10		1 2 4	5 65 70	6 24 30	7 33 40	8 29 37	9 28 47	10 23 72	11 42 53
				12 34 46	14 41 58						
86	10	10		1 2 44	3 50 53	4 12 78	5 21 70	7 22 29	9 19 28	11 24 35	14 40 54
				18 41 59	25 31 56						
87	10	10		1 2 4	5 54 59	6 18 24	7 41 48	8 35 60	9 19 77	11 26 37	13 30 70
				16 36 67	21 43 64						
88	11	11	$\surd$	11 22 44	1 2 4	5 10 20	7 14 28	9 18 36	13 38 51	6 35 41	17 43 62
				8 31 39	12 42 54	16 32 56					
89	11	11	$\surd$	1 64 65	9 76 85	14 46 60	18 23 84	31 50 81	33 40 73	34 44 78	35 47 77
				36 38 74	41 62 68	52 69 72					
90	11	11		1 2 46	7 54 61	12 28 40	15 48 63	18 38 70	21 34 55	23 26 49	25 31 84
				51 68 73	60 71 79	66 76 80					
91	11	11		13 39 78	18 27 82	23 61 84	28 48 76	32 51 72	33 34 67	35 41 85	37 62 66
				46 60 77	69 79 81	80 83 88					
92	11	11		1 45 46	7 38 61	16 59 75	20 42 70	30 56 86	32 35 67	37 51 78	39 58 73
				52 63 81	79 83 88	80 82 90					
93	11	11		1 2 4	8 45 53	12 70 82	19 36 55	28 44 72	29 34 63	31 41 83	33 46 79
				43 50 68	54 69 78	61 67 87					
94	11	11		1 46 47	7 24 31	11 43 54	18 20 38	34 57 91	35 44 79	36 52 78	55 65 84
				61 67 88	64 72 86	12 25 53					
95	11	11		1 2 48	8 34 42	12 45 62	18 57 75	21 43 64	23 28 51	24 30 54	29 32 92
				55 68 82	70 79 86	76 80 91					
96	12	11		1 2 49	5 10 20	7 14 28	11 22 44	4 27 50	13 39 70	17 42 71	18 37 55
				8 43 51	3 6 12	16 32 72					
97	12	12	$\surd$	1 2 95	8 25 33	12 30 79	21 50 71	27 42 82	32 75 86	34 44 78	35 58 74
				45 59 83	51 60 88	56 61 92	77 84 90				
98	12	12		14 56 84	6 75 81	15 37 76	18 48 68	21 31 88	25 60 63	32 34 96	33 44 87
				58 74 82	72 79 91	85 89 94					
99	12	12		1 2 97	8 15 92	16 33 49	19 39 59	21 65 86	26 37 63	27 41 68	28 52 75
				29 38 67	43 48 94	54 64 89	57 69 87				
100	12	12		1 49 50	6 69 75	14 84 98	21 43 78	24 33 91	34 61 73	36 40 96	38 53 85
				41 46 87	55 74 81	72 82 90	77 80 97				
101	12	12		1 2 4	7 82 89	11 57 68	17 35 83	24 52 76	41 54 88	43 58 72	51 61 91

				56 65 92	59 64 96	62 70 93	63 79 85				
102	12	12		30 60 81	15 52 65	25 49 78	34 41 75	43 44 101	56 76 82	62 74 90	63 80 85
				67 69 100	83 91 94	84 88 98	9 32 64				
103	12	12		1 2 4	7 64 71	11 42 72	19 69 88	27 47 74	51 63 91	54 60 97	55 68 90
				62 80 85	67 77 93	70 78 95	14 38 59				
104	13	13	$\surd$	1 2 53	6 31 79	14 21 97	23 38 61	32 36 100	34 67 101	39 47 96	41 60 82
				46 62 88	54 74 84	59 77 86	69 80 93	75 87 92			
105	13	12		15 30 60	1 2 4	8 16 32	11 22 44	13 26 52	17 34 68	19 38 76	18 41 64
				12 43 55	9 49 58	6 42 69	7 14 28				
106	13	13		1 2 54	6 28 34	9 90 99	27 69 96	30 66 70	33 50 83	35 43 98	38 49 95
				41 67 80	47 61 92	62 74 94	77 82 101	85 88 103			
107	13	13		1 2 4	7 42 72	14 69 83	17 46 78	21 68 89	23 59 71	25 33 58	26 41 92
				28 34 62	37 50 87	43 53 97	67 76 98	80 91 96			
108	13	13		1 2 55	6 25 31	9 72 81	12 32 44	18 41 59	35 50 93	37 40 74	46 51 97
				47 61 75	48 70 86	52 56 82	79 87 100	84 91 101			
109	13	13		1 2 4	7 33 83	10 75 85	13 30 43	23 51 74	25 46 88	27 32 104	29 41 97
				31 47 62	36 54 90	38 49 60	39 48 100	89 95 103			
110	13	13		4 8 59	13 30 93	21 50 81	28 34 62	33 72 105	46 56 100	49 67 92	65 84 91
				73 74 109	78 87 101	83 95 98	86 88 108	11 52 68			
111	13	13		1 17 95	14 44 58	22 49 84	28 66 73	41 46 106	48 50 98	52 71 92	54 64 101
				55 80 86	69 77 103	82 91 102	93 96 108	4 36 72			
112	14	14		48 64 80	5 81 86	12 66 78	15 37 52	17 42 87	24 43 93	27 57 84	33 53 92
				38 51 89	41 44 109	63 77 98	72 83 101	102 106 108	103 104 111		
113	14	14	$\surd$	1 2 58	7 17 103	21 60 74	24 54 83	26 72 98	33 36 69	34 79 102	35 63 85
				37 42 108	38 51 100	43 52 61	65 73 105	82 88 107	93 97 109		
114	14	13		1 2 58	5 10 20	7 14 28	11 22 44	13 26 52	17 34 68	9 41 73	8 16 69
				3 40 43	25 54 85	12 47 59	19 42 91	18 36 66			
115	14	14		1 2 4	10 27 98	11 89 100	13 18 110	14 77 91	16 35 96	22 68 90	28 34 109
				29 62 82	30 51 94	31 39 70	32 41 106	36 48 103	42 49 108		
116	14	14		1 2 59	5 65 70	7 31 92	8 37 45	9 25 100	10 52 74	15 34 97	17 20 113
				18 48 66	21 33 54	26 69 73	28 39 67	32 38 110	76 89 103		
117	14	14	$\surd$	1 2 4	6 95 101	8 44 81	9 49 58	10 35 92	15 71 86	20 27 110	21 32 53
				23 52 75	24 50 91	38 55 100	39 51 105	70 84 103	5 18 48		
118	14	14		1 2 60	5 51 72	12 16 28	14 40 54	24 30 112	27 65 80	32 63 95	33 41 74
				34 37 71	35 42 111	45 56 101	48 57 105	50 68 93	79 89 108		
119	15	15	$\surd$	17 34 85	9 83 92	14 73 87	22 35 57	25 40 104	30 61 91	33 38 71	37 56 100
				41 52 93	42 54 107	43 47 90	49 50 99	80 96 103	95 98 116	109 111 117	
120	15	14		1 2 61	7 14 28	11 22 44	13 26 52	4 31 62	17 51 86	19 57 82	9 46 55
				6 47 79	20 43 97	5 29 96	3 15 18	8 16 72	10 40 50		
121	15	15	$\surd$	1 2 62	9 67 76	14 55 80	21 65 77	31 37 115	34 38 72	36 69 105	40 51 91
				46 74 93	50 82 89	64 79 106	68 92 97	73 86 108	95 103 113	98 101 118	
122	15	15		1 60 61	18 49 67	30 76 99	35 83 118	40 52 110	44 50 116	57 71 108	68 77 113
				75 95 102	79 90 111	80 96 106	81 100 103	88 93 117	89 97 114	94 107 109	
123	15	14		1 2 4	5 10 20	7 14 28	8 16 32	11 22 44	13 26 52	17 34 68	19 38 76
				23 46 92	12 49 61	9 50 59	27 56 83	6 36 42	18 53 98		
124	15	15		1 2 63	8 29 37	18 24 118	25 70 79	40 56 96	53 64 113	58 85 97	59 74 109
				67 80 111	69 73 120	78 83 119	81 101 104	82 92 114	91 98 117	14 48 86	
125	15	15		1 2 63	9 42 51	30 68 98	35 52 108	48 70 118	53 67 111	65 78 112	66 76 115
				71 82 114	79 99 105	80 88 117	84 89 120	85 97 113	101 104 122	102 106 121	
126	16	16	$\surd$	18 36 72	1 2 64	5 10 20	11 22 44	13 26 52	17 34 68	19 38 76	3 43 83
				8 61 73	29 60 95	7 37 96	23 47 102	9 25 110	4 49 81	6 27 105	14 28 56
127	15	15		1 2 65	11 15 26	16 66 82	20 71 76	31 39 70	37 84 121	58 79 106	59 68 93
				67 81 113	74 92 109	75 87 115	83 86 124	91 98 120	94 104 117	19 49 73	

128	16	16	✓	10 74 84	9 101 110	15 56 71	19 66 85	30 35 65	33 50 111	39 55 94	45 58 103
				48 86 90	51 52 127	59 88 99	60 91 97	79 102 105	82 104 106	96 108 116	100 107 114
129	16	15		1 2 4	5 10 20	7 14 28	8 16 32	11 22 44	13 26 52	17 34 68	19 38 76
				23 46 92	25 50 100	12 47 82	27 58 98	9 64 73	18 63 81	6 36 42	
130	16	16		7 72 79	9 71 80	21 38 113	25 70 85	29 49 78	32 54 86	35 41 124	46 56 120
				48 88 90	61 77 114	63 91 102	87 106 111	94 97 127	96 104 122	99 103 126	116 117 129
131	16	16		1 2 4	10 68 78	16 22 38	28 59 100	32 37 126	35 49 84	44 61 105	50 79 102
				51 62 113	54 88 97	57 76 112	64 85 110	65 92 104	71 86 101	75 83 123	98 111 118
132	16	16		1 67 68	10 61 122	16 73 89	21 33 54	28 63 91	31 84 115	45 49 94	52 74 110
				55 70 117	72 90 114	76 82 126	86 93 125	92 95 129	102 113 121	103 105 130	109 118 123
133	16	16		19 57 114	11 63 81	17 64 86	26 72 87	30 59 89	33 75 91	36 73 109	48 79 127
				49 50 99	51 53 131	56 88 101	65 93 105	71 92 112	90 98 125	110 119 124	123 126 130
134	16	16		1 2 68	4 95 99	7 54 61	11 34 111	12 81 93	13 116 129	22 60 96	28 79 107
				37 52 89	46 72 118	50 64 114	71 101 104	75 92 117	76 85 125	78 102 110	86 105 115
135	16	16		1 2 68	6 13 19	11 53 93	23 79 112	25 60 110	28 58 105	39 65 109	45 59 121
				52 86 101	57 94 98	63 81 99	64 74 125	80 92 123	97 106 126	103 111 119	108 113 130
136	17	17	✓	1 67 68	2 63 65	4 20 120	9 56 89	11 32 43	14 78 92	23 31 54	25 59 84
				26 29 55	35 62 97	38 48 86	45 85 130	46 53 99	57 87 106	76 94 118	95 108 123
				114 119 131							
137	17	17	✓	2 43 96	5 44 98	13 55 68	18 34 52	21 110 131	24 49 112	35 64 99	46 47 92
				51 59 129	57 77 97	58 75 133	61 71 127	65 87 115	100 111 126	101 104 134	105 114 128
				106 118 125							
138	17	17		1 35 104	2 70 136	10 61 87	11 74 85	12 60 90	13 118 131	17 62 79	23 65 96
				28 95 123	29 56 111	38 84 122	41 91 132	44 80 102	49 52 101	81 99 120	98 106 130
				119 124 133							
139	17	17		1 2 4	10 15 134	11 72 78	17 46 110	21 109 130	22 77 84	24 74 98	28 40 68
				35 66 108	38 57 76	39 92 131	43 69 113	48 80 107	54 90 103	56 81 114	79 102 116
				87 105 121							
140	17	17		20 40 80	2 59 83	4 11 133	9 32 41	12 61 73	17 52 69	19 106 125	22 76 86
				42 85 127	46 90 96	47 74 113	48 84 132	62 65 137	63 89 114	82 103 119	102 107 135
				109 110 139							
141	17	17		1 2 4	6 103 109	8 43 106	10 52 99	11 34 118	16 65 81	17 31 48	18 72 87
				21 91 112	24 51 75	25 104 129	41 80 102	46 59 105	53 83 111	55 64 132	56 96 101
				108 115 134							
142	17	17		1 2 72	10 76 109	13 82 95	24 98 122	26 79 105	31 61 92	38 86 124	49 78 113
				54 77 131	57 96 103	59 80 121	75 100 117	84 87 139	97 106 133	102 114 130	108 123 127
				120 128 134							
143	17	17		1 3 141	6 33 116	9 13 139	14 103 117	16 51 67	18 113 131	24 63 87	28 74 97
				34 68 75	42 78 107	49 57 106	50 61 132	55 77 99	62 81 100	64 79 111	73 98 118
				91 112 122							
144	18	18	✓	1 2 73	5 46 51	6 121 127	7 125 132	11 111 122	13 28 129	16 63 79	30 38 68
				31 83 92	34 48 130	35 45 134	36 39 75	49 74 123	50 54 140	53 85 112	56 80 120
				66 84 126	67 87 124						
145	18	18	✓	1 3 4	9 73 82	17 75 87	22 49 118	30 36 139	35 50 130	43 71 114	46 83 108
				48 59 107	53 94 104	56 61 140	67 90 122	77 101 121	85 111 119	88 106 127	91 98 138
				93 112 126	100 116 129						
146	18	18		1 2 74	8 38 46	9 110 119	10 113 123	11 76 87	15 68 83	20 37 57	22 44 84
				24 71 99	25 105 130	34 52 86	48 67 115	56 95 107	64 69 133	85 91 140	88 92 142
				111 125 132	114 117 143						
147	18	18		21 42 84	6 13 140	8 88 96	10 66 76	11 131 142	17 102 119	18 107 125	29 64 93
				36 50 86	37 46 92	43 74 117	44 69 113	48 87 135	49 82 114	52 90 109	53 68 121
				77 100 124	143 144 145						
148	18	18		1 74 147	4 103 107	10 32 126	14 21 35	15 67 96	16 92 108	17 97 131	23 53 118

				36 64 100	39 58 129	43 46 89	47 85 110	60 69 139	61 98 111	66 86 128	91 117 122
				93 104 137	124 130 136						
149	18	18		1 13 137	11 59 101	14 44 119	18 40 58	21 50 71	23 110 133	32 35 146	34 49 134
				41 72 113	46 93 102	51 57 143	52 60 141	69 76 142	70 96 123	82 106 125	84 94 139
				88 116 121	95 112 132						
150	18	18		1 74 75	4 94 98	7 46 53	9 28 37	16 45 61	17 66 101	21 83 88	24 68 92
				32 80 102	38 110 148	50 93 107	55 63 142	59 85 144	69 103 116	86 109 127	99 119 130
				108 123 135	114 117 147						
151	18	18		1 8 144	6 9 148	11 41 52	16 122 138	19 115 134	22 56 78	25 75 101	31 70 112
				32 65 118	35 92 127	37 64 124	38 48 141	63 67 130	69 74 146	71 89 133	83 97 137
				85 108 128	100 102 149						
152	19	19	✓	2 76 150	5 13 144	6 57 101	11 47 116	17 75 94	18 45 63	20 70 102	22 68 106
				28 31 149	30 42 140	33 93 126	49 86 115	56 80 136	65 88 129	71 85 138	73 112 113
				90 97 145	91 100 143	117 127 142					
153	19	19	✓	1 2 77	9 26 35	15 22 146	21 61 82	23 36 140	24 69 108	31 50 81	41 68 126
				43 94 137	46 83 116	47 57 143	53 97 109	54 74 128	55 73 135	63 105 111	65 93 125
				87 101 139	89 119 123	142 145 150					
154	19	19		22 44 88	2 33 123	3 72 75	5 55 60	11 28 137	15 56 113	19 86 87	20 81 93
				43 51 146	46 80 120	47 84 131	53 71 136	58 96 106	62 76 138	91 118 127	100 124 130
				102 115 128	105 112 147	125 129 150					
155	19	19		1 2 4	6 36 42	7 76 86	11 52 114	12 26 141	13 122 135	17 60 77	22 124 146
				24 80 99	25 92 117	28 51 132	34 100 134	37 110 147	47 65 137	49 54 150	57 84 128
				58 87 116	70 85 120	91 107 123					
156	19	19		1 2 79	7 102 109	10 74 92	11 73 94	13 110 123	15 81 96	17 39 56	19 132 151
				25 28 53	31 40 71	32 55 87	35 76 111	48 99 105	49 65 114	50 88 118	52 86 122
				67 93 130	84 98 142	136 144 148					
157	19	19		1 2 4	6 28 135	7 67 97	8 59 106	9 27 36	11 54 65	12 116 128	14 24 38
				31 57 131	34 86 120	42 91 108	45 64 109	46 78 124	55 70 142	56 96 117	62 75 137
				63 88 132	68 73 141	76 99 134					
158	19	19		1 2 80	4 127 131	7 71 94	8 63 103	9 33 134	10 48 58	12 41 53	13 26 132
				14 75 89	16 76 98	20 88 108	32 43 147	37 73 122	56 102 130	62 107 113	65 99 124
				67 72 139	81 111 128	137 140 155					
159	19	19		1 2 157	5 89 94	8 25 142	9 111 120	11 22 96	15 46 61	20 115 135	28 86 114
				35 69 125	36 54 141	41 81 119	43 49 92	50 77 132	68 97 130	76 88 147	79 93 145
				106 129 136	108 127 140	10 26 47					
160	20	20	✓	1 2 81	4 101 105	6 16 22	9 95 104	11 121 132	14 32 46	17 34 126	25 62 123
				29 60 129	30 77 113	38 43 155	40 94 134	41 76 125	57 102 115	64 72 136	70 89 141
				75 78 153	87 107 140	93 116 137	118 133 145				
161	20	20		23 46 92	3 57 107	4 99 103	8 34 135	10 39 49	13 76 89	19 140 159	24 31 154
				25 53 78	30 45 75	36 71 126	37 51 88	41 59 143	65 87 139	67 94 114	81 93 149
				82 91 152	97 113 129	117 118 160	144 150 155				
162	20	20		1 2 82	4 108 112	6 18 24	7 128 135	10 76 96	14 75 101	15 74 89	16 114 146
				19 68 113	23 56 79	29 40 151	30 92 100	31 126 157	38 55 93	43 85 120	44 65 109
				57 60 159	71 110 123	90 115 137	116 125 153				
163	20	20		1 2 4	8 76 95	11 43 54	13 27 149	16 86 102	22 31 53	25 49 74	28 34 157
				29 58 125	33 83 113	39 65 104	47 84 126	48 71 140	63 107 119	64 82 146	66 112 117
				73 108 128	85 106 142	88 103 148	91 101 153				
164	20	20		1 2 83	4 89 93	7 42 129	8 30 38	9 68 105	11 32 43	12 24 140	16 33 49
				19 34 53	26 70 120	27 73 118	28 67 95	29 80 109	52 58 110	57 104 161	62 103 123
				64 78 150	74 99 139	88 98 154	133 146 151				
165	20	20		1 3 164	5 69 101	10 43 53	18 107 125	20 36 149	24 87 102	28 84 109	30 65 95
				31 48 79	34 108 142	38 60 143	45 106 151	49 75 124	50 97 118	67 94 138	72 80 157
				92 103 154	99 111 153	110 119 156	146 152 159				

166	20	20		1 2 84	4 26 144	6 67 105	9 103 112	16 28 44	17 35 148	20 109 129	23 47 70
				27 91 102	31 121 152	33 120 153	36 68 104	41 56 151	50 93 123	52 92 126	55 60 115
				58 66 124	59 80 145	78 81 159	117 127 156				
167	20	20		1 2 4	5 59 118	7 107 114	9 48 128	12 155 161	19 36 55	22 101 145	25 45 147
				27 62 97	28 92 120	29 124 153	33 63 137	40 90 130	41 56 152	51 83 135	57 91 133
				68 106 129	69 82 151	73 81 159	136 146 157				
168	21	20		24 48 96	1 2 85	5 10 20	11 22 44	13 26 52	17 34 68	19 38 76	4 45 86
				23 69 122	25 75 118	27 74 101	3 58 61	7 73 102	37 77 114	16 65 81	9 80 89
				28 59 137	8 70 106	6 18 156	14 35 147				
169	21	21	√	3 60 112	7 74 102	9 89 160	11 27 38	14 65 118	19 31 157	22 56 78	30 75 105
				35 76 111	37 77 114	39 68 107	46 90 136	47 106 110	48 121 145	49 50 99	54 69 154
				61 82 148	81 83 164	127 144 152	133 146 156	137 143 163			
170	21	21		4 81 85	7 140 147	10 18 162	11 69 112	12 83 99	15 55 130	19 51 70	20 22 42
				21 124 145	27 86 113	31 76 125	33 39 72	34 82 116	35 38 167	60 74 134	68 97 165
				90 118 142	91 104 157	92 109 153	105 114 161	106 107 169			
171	21	21		2 21 23	6 24 153	8 33 146	9 80 100	10 89 99	11 73 109	14 65 120	15 27 159
				28 69 130	29 86 115	34 87 121	37 59 96	38 101 139	43 88 131	60 107 124	67 93 119
				68 81 158	74 105 140	95 125 141	117 122 166	164 167 168			
172	21	21		2 88 90	3 92 95	5 98 103	7 20 27	9 33 42	10 147 157	12 56 68	21 87 106
				22 48 70	36 99 135	39 57 96	41 81 122	45 75 142	49 107 156	53 64 117	54 71 125
				60 61 171	94 126 140	100 134 138	113 121 164	120 143 149			
173	21	21		1 7 8	9 12 21	13 43 56	14 97 111	15 121 136	18 78 113	20 31 162	23 73 146
				41 65 149	46 75 144	47 105 115	48 80 128	49 54 103	51 110 114	53 87 140	66 91 157
				71 109 135	77 79 171	99 118 154	112 134 151	28 72 116			
174	21	21		1 2 88	3 79 98	5 83 96	8 51 131	9 59 124	12 130 142	15 119 134	18 63 129
				20 113 133	24 31 167	26 36 164	27 74 101	28 62 90	33 102 135	42 106 110	49 97 126
				53 67 120	89 114 149	92 122 144	116 137 153	151 157 168			
175	22	22	√	25 50 100	3 37 40	5 82 87	6 58 64	7 83 99	8 81 102	14 71 85	17 130 147
				18 137 155	23 47 70	30 79 109	31 134 165	32 78 110	36 91 120	42 122 164	44 59 160
				60 114 174	63 106 132	89 108 156	107 140 142	124 136 163	153 162 166		
176	22	22	√	1 2 89	4 39 141	5 63 118	6 40 46	10 83 103	11 86 101	12 21 167	16 77 115
				17 42 151	18 65 129	24 43 157	27 98 105	31 45 76	32 91 123	44 80 140	48 51 173
				56 82 150	66 79 163	69 106 139	95 124 147	104 126 154	119 127 168		
177	22	21		1 2 4	5 10 20	7 14 28	8 16 32	11 22 44	13 26 52	17 34 68	19 38 76
				23 46 92	25 50 100	29 58 116	31 62 124	35 70 140	18 65 112	12 67 79	9 80 89
				30 73 103	40 81 121	27 86 113	36 78 114	6 60 66			
178	22	22		1 2 90	4 37 145	5 62 121	7 135 142	8 64 122	9 158 167	12 18 24	17 84 111
				23 78 101	25 73 98	32 127 159	34 69 103	53 93 138	54 96 150	63 110 131	65 106 137
				71 97 168	86 99 165	87 102 163	104 134 148	118 140 156	126 129 175		
179	22	22		1 2 4	5 73 111	7 143 150	8 93 101	9 44 144	10 42 52	11 142 153	12 84 107
				13 132 145	15 77 117	18 64 133	27 98 125	28 49 158	33 122 155	53 112 165	55 80 154
				58 75 162	83 103 159	88 118 149	97 113 163	116 138 157	128 134 173		
180	22	22		1 2 91	4 38 146	5 60 125	7 128 135	9 12 21	13 71 84	14 117 131	19 94 105
				23 127 150	24 74 130	25 61 144	32 78 110	40 73 113	43 80 123	51 69 162	62 101 163
				66 88 154	81 116 145	82 126 136	85 112 153	132 152 160	133 149 164		
181	22	22		1 2 4	5 67 124	6 89 95	8 33 156	10 82 109	15 76 120	18 29 170	20 59 79
				22 63 85	32 56 88	34 46 80	37 91 127	42 55 168	43 64 107	45 110 155	49 84 133
				77 94 164	78 106 153	81 100 131	112 121 172	115 129 167	128 151 158		

Таблица 7.1.3 Класификация на (v,5,2,1) оптични ортогонални кодове с $v \leq 114$

v	s	всички	съвършени	кодови думи на един оптичен ортогонален код					
29	2	1	0	1 2 13 18	3 6 10 25				
30	2	1	0	1 4 7 8	2 11 16 21				
31	2	2	0	1 2 6 27	3 11 19 22				
32	2	5	0	1 2 9 25	3 6 17 21				
33	2	9	0	1 2 6 29	3 11 18 25				
34	2	23	0	1 2 6 30	3 11 19 22				
35	2	48	0	1 2 5 32	6 12 19 28				
36	3	0	0						
37	3	1	1	1 2 7 32	3 11 19 22	4 13 17 27			
38	3	0	0						
39	3	1	0	1 2 10 31	4 11 15 27	3 6 20 25			
40	3	0	0						
41	3	2	0	1 2 6 37	3 14 25 28	7 15 24 33			
42	3	1	0	1 10 19 20	3 6 11 37	2 4 17 29			
43	3	5	1	1 2 7 38	3 14 25 28	4 8 24 34			
44	3	14	0	1 2 12 34	3 6 21 29	4 17 24 31			
45	3	27	0	1 2 4 42	6 12 22 35	8 17 26 34			
46	3	51	0	1 2 5 43	7 17 27 34	6 15 24 30			
47	3	93	0	1 2 4 44	8 17 26 34	6 12 22 37			
48-51	4	0	0						
52	4	1	0	1 2 14 40	3 6 25 33	4 11 28 45	5 15 20 36		
53	4	2	0	1 2 15 40	5 10 16 47	3 12 21 24	4 8 27 34		
54	4	3	0	1 2 7 49	4 19 34 38	9 18 31 41	3 11 14 40		
55	4	34	0	1 2 4 52	8 16 30 41	9 19 29 38	6 12 27 40		
56	4	27	0	1 2 15 43	3 7 10 33	6 22 31 40	5 24 32 44		
57	4	90	3	1 2 12 47	7 14 27 44	5 23 31 39	3 32 36 51		
58	4	193	0	1 2 4 55	8 19 30 38	9 18 32 44	6 21 31 37		
59	4	576	0	1 2 4 56	6 12 24 41	8 16 27 48	9 22 31 45		
60	5	0	0						
61	5	1	1	1 2 12 51	3 6 31 36	7 26 34 42	9 18 32 47	4 17 21 41	
62-63	5	0	0						
64	5	2	0	1 2 17 49	5 13 18 41	3 7 10 37	6 20 26 45	9 21 33 42	
65	5	46	2	1 2 9 58	4 19 34 38	10 20 32 53	3 17 28 54	5 23 29 52	
66	5	12	0	1 2 4 63	8 19 30 38	9 23 32 49	10 20 35 51	6 18 24 45	
67	5	101	10	1 2 7 62	3 21 39 42	11 22 38 51	8 17 43 58	4 14 34 48	
68	5	469	28	1 2 18 52	6 12 27 53	5 24 43 48	9 23 45 55	3 7 11 40	
69	5	863	18	1 2 6 65	3 19 36 53	7 20 27 48	11 25 37 51	8 23 38 47	
70	5	2743	158	5 10 20 40	1 2 14 58	3 6 45 51	4 11 27 63	8 17 41 49	
71	5	6715	114	1 2 5 68	6 12 30 53	9 26 43 52	10 21 35 60	7 22 38 51	
72	6	0	0						
73	6	7	7	1 2 7 68	3 13 23 26	8 22 36 44	11 27 43 54	9 18 33 58	4 21 38 42
74	6	5	3	1 2 7 69	11 26 41 52	9 27 36 55	8 16 29 61	4 14 39 64	3 23 40 43
75	6	54	48	1 2 4 72	6 23 40 46	10 21 31 53	8 16 36 55	9 27 42 57	12 24 37 61
76	6	44	6	1 19 37 38	6 34 41 48	5 10 21 65	4 8 17 67	3 15 27 30	2 25 45 47
77	6	145	22	7 14 28 49	1 2 5 74	9 31 43 55	11 26 44 62	10 27 37 57	6 25 38 54
78	6	219	36	1 2 5 75	12 28 40 59	6 29 42 55	9 18 33 48	10 35 56 67	7 14 34 51
79	6	1115	160	1 2 4 76	10 20 39 60	9 27 44 61	13 26 41 64	6 12 37 48	8 16 32 65
80	6	6385	988	1 2 21 61	3 36 47 58	13 28 43 56	4 10 42 74	5 34 46 73	8 17 31 62
81	6	13006	686	1 2 4 78	6 25 44 50	10 20 36 65	9 18 42 57	11 23 34 64	8 22 54 68
82	6	18816	1480	1 2 4 79	6 19 32 38	10 20 37 65	9 24 33 66	11 23 41 59	8 22 43 51
83	6	72866	3519	1 2 4 80	6 39 50 61	15 31 47 62	8 18 45 64	9 23 43 57	12 25 42 66

84	7	0	0						
85	7	7	7	1 2 7 80 10 24 38 48	3 21 44 67	8 16 43 58	13 32 49 66	9 20 31 40	4 29 33 59
86	7	0	0						
87	7	63	33	1 2 4 84 9 30 63 75	8 26 44 52	6 17 28 34	10 20 39 58	15 31 47 62	13 27 50 73
88	7	34	5	1 2 23 67 11 28 39 75	5 10 25 73	3 6 32 62	8 16 35 69	7 14 45 57	4 37 46 55
89	7	503	141	1 2 4 86 10 28 51 61	8 21 55 68	6 12 32 69	14 31 45 67	9 24 49 74	11 30 46 73
90	7	2939	960	1 2 4 87 10 37 55 72	8 33 49 65	14 29 52 75	13 26 47 69	6 12 54 66	9 40 59 79
91	7	9118	1375	1 2 4 88 9 44 54 65	6 12 30 73	15 32 49 64	13 33 53 66	8 16 39 68	14 28 50 69
92	7	21655	2477	1 2 24 70 5 18 49 62	3 6 12 83	7 14 34 72	10 26 42 52	11 39 56 75	4 8 33 41
93	7	104401	7906	1 2 4 90 8 25 62 70	11 22 44 60	14 28 43 78	9 41 51 61	6 18 30 36	13 26 47 66
94	7	118707	11706	1 2 4 91 12 37 51 65	9 18 36 67	16 35 54 70	10 20 33 81	8 30 52 60	6 17 32 79
95	7	889778	50016	1 2 4 92 11 32 48 63	6 12 24 77	14 34 54 68	13 35 57 70	10 33 56 66	8 17 36 86
96	8	0	0						
97	8	8	8	1 2 4 94 14 28 46 79	8 16 39 74 10 30 50 60	6 19 25 61	9 26 43 52	12 24 53 68	11 22 49 70
98	8	548	384	7 14 28 56 4 43 51 59	5 10 20 83 6 12 29 52	9 18 45 71	3 22 41 44	1 31 32 65	2 13 50 87
99	8	663	180	9 18 36 63 12 24 50 73	1 2 4 96 6 52 66 91	16 32 51 80	13 30 43 71	15 37 57 77	10 21 31 65
100	8	511	168	1 2 26 76 11 33 44 72	3 6 12 91 10 40 57 80	8 16 35 81	13 31 49 62	4 45 52 59	5 34 63 68
101	8	2453	620	1 2 4 98 9 18 36 64	10 20 40 71 11 22 44 67	6 12 47 66	8 16 29 88	14 38 62 76	15 32 58 84
102	8	4454	1349	1 2 4 99 9 29 58 67	11 50 63 76 10 31 41 87	14 28 47 83	17 34 57 79	6 18 48 90	8 16 59 86
103	8	32438	5679	1 2 4 100 11 46 68 91	8 16 32 79 13 28 61 89	10 36 62 72	6 25 44 50	9 18 39 82	17 34 54 83
104	8	57203	9633	1 2 27 79 13 31 64 82	3 6 12 95 8 36 47 75	4 49 54 59	7 24 41 48	19 38 61 81	14 30 46 60
105	8	1452405	158564	1 2 12 95 6 33 42 78	4 8 48 65 5 25 46 71	7 30 56 82	14 28 43 90	3 38 54 70	13 31 50 81
106	8	599984	72186	1 2 4 103 8 30 38 87	6 18 24 65 10 33 54 64	13 29 45 58	15 35 55 70	11 39 67 92	9 46 72 89
107	8	4612857	320346	1 2 4 104 12 32 61 73	9 18 45 80 6 37 53 76	11 22 55 74	10 24 38 48	13 26 56 77	8 25 50 65
108	9	1	1	1 28 55 56 6 23 57 91	5 10 20 93 7 37 44 76	2 4 13 99 3 24 45 48	12 31 50 62	8 16 49 75	14 43 61 79
109	9	79	79	1 2 4 106 13 29 61 93	10 51 68 85 9 39 59 79	15 43 62 81 8 26 44 52	11 22 53 78	14 37 60 74	6 12 33 88
110	9	1	0	1 2 4 107 10 39 60 81	8 16 31 95 11 22 59 73	6 41 58 75 12 32 44 77	13 26 53 83	18 36 61 85	9 28 47 56
111	9	785	344	1 2 4 108 10 20 34 97	16 38 73 92 15 30 51 90	11 28 61 94 8 49 67 93	9 32 55 64	13 40 53 82	6 12 37 80

112	9	1003	435	1 2 29 85 7 30 53 60	3 6 12 103 13 35 57 70	19 38 58 92 8 25 41 49	5 36 67 72	4 14 18 65	11 37 48 80
113	9	9548	2479	1 2 4 110 9 34 59 68	6 12 24 95 14 28 60 81	11 40 62 84 16 36 55 93	13 26 61 78	8 23 31 72	10 37 47 80
114	9	15322	4502	1 2 4 111 6 20 60 100	11 22 44 81 16 32 51 83	10 49 62 75 9 36 57 78	12 24 41 97	8 61 84 99	18 43 68 86

Таблица 7.1.4 Примери на оптимални (v,5,2,1) оптични ортогонални кодове с $115 \leq v \leq 155$

v	s	съвършени	кодови думи на един оптичен ортогонален код					
115	9		1 2 5 112	6 20 34 40	9 21 62 103	10 25 35 70	11 47 63 79	13 26 56 85
			17 39 61 78	18 42 66 84	19 38 65 88			
116	9		1 29 57 58	2 37 72 74	3 6 19 103	5 10 25 101	7 31 55 62	9 26 43 52
			14 47 65 83	4 8 49 71	11 23 50 89			
117	9		1 2 4 114	6 12 36 93	8 16 55 86	9 18 53 82	11 22 59 80	13 27 41 54
			15 32 66 100	19 38 61 94	20 45 65 91			
118	9		1 2 5 115	6 12 53 77	7 57 75 100	11 31 51 62	13 26 49 95	14 28 44 102
			15 37 52 85	17 34 55 97	9 19 54 108			
119	9		1 2 4 116	6 12 57 74	8 24 40 48	9 18 38 99	10 33 66 76	11 30 65 100
			13 26 60 85	14 28 55 92	15 46 67 88			
120	10	√	5 10 35 95	7 19 31 38	11 22 63 79	8 47 64 81	13 26 53 80	1 2 45 77
			9 23 37 46	3 6 61 65	20 49 70 91	15 33 48 84		
121	10		11 22 44 77	1 2 4 118	8 16 32 97	6 12 42 91	9 35 61 70	13 34 47 84
			19 38 67 92	18 41 64 82	10 20 63 78	14 28 45 104		
122	10							
123	10	√	1 2 4 120	8 16 32 99	9 26 35 79	14 28 66 85	13 43 73 86	22 45 68 90
			20 41 61 92	11 29 47 58	6 12 54 81	10 25 59 84		
124	10		1 2 32 94	3 6 12 115	11 22 44 91	5 56 73 90	4 41 64 87	18 42 71 100
			7 14 50 88	13 26 61 89	20 45 72 99	8 16 65 75		
125	10	√	1 2 4 122	6 12 24 107	15 44 81 103	16 32 67 90	10 20 48 97	11 36 61 72
			17 43 69 86	9 40 71 80	14 33 79 98	8 21 55 112		
126	10	√	9 18 36 72	1 2 4 123	11 22 44 93	19 39 58 92	6 43 66 89	16 47 78 94
			8 67 84 109	12 24 38 112	13 28 69 98	10 45 75 96		
127	10	√	1 2 4 124	6 12 24 109	11 22 44 94	13 41 69 82	9 26 43 52	14 37 60 74
			10 20 59 88	16 47 63 95	15 36 51 102	8 27 65 100		
128	10	√	1 2 33 97	3 6 12 119	13 26 52 89	7 27 47 54	4 8 57 79	10 35 60 70
			18 36 59 105	14 30 44 86	5 29 90 109	11 28 62 73		
129	10	√	1 2 4 126	8 16 32 105	11 22 44 96	13 26 62 93	19 42 61 95	14 39 64 78
			6 15 21 75	10 37 47 94	12 29 70 100	18 46 66 109		
130	10		1 2 4 127	9 18 36 103	11 22 44 97	15 46 84 107	16 41 73 105	12 47 71 95
			14 51 72 93	13 26 43 113	10 20 49 101	6 40 74 80		
131	10		1 2 4 128	6 12 24 113	11 22 44 98	13 26 52 92	9 28 47 56	23 46 71 106
			14 45 59 95	8 16 37 110	15 32 49 64	10 20 63 90		
132	11	√	11 22 55 88	1 2 4 129	13 26 52 93	14 28 87 115	23 46 71 107	8 43 70 97
			16 53 74 95	9 18 47 103	15 30 49 113	10 20 60 92	6 57 69 81	
133	11	√	1 2 5 130	6 36 66 72	7 56 70 84	9 18 35 116	10 29 48 58	11 22 65 90
			12 39 51 92	13 46 59 96	15 31 47 62	20 40 64 109	21 42 76 99	
134	11		1 2 13 123	15 30 57 107	9 45 54 94	3 6 35 105	8 61 71 81	7 14 33 115
			5 28 51 56	17 48 79 96	16 41 75 109	4 47 69 91	18 39 76 113	
135	11		1 2 4 132	6 12 48 99	8 16 62 89	9 43 72 101	10 25 35 85	11 33 55 66
			13 39 65 78	14 28 45 104	19 38 68 105	20 40 61 114	23 47 79 111	
136	11		1 2 35 103	3 6 19 123	4 8 59 85	5 10 47 99	9 18 40 114	11 39 67 78
			12 24 73 87	15 30 53 113	17 44 71 88	20 45 70 90	7 36 79 100	
137	11		1 2 4 134	6 12 21 128	8 16 47 106	10 38 66 76	11 44 55 96	13 26 45 118
			14 37 60 74	17 42 67 84	18 36 58 115	20 49 69 103	24 48 75 110	
138	11	√	1 2 4 135	6 12 21 129	10 20 49 109	11 48 85 96	13 44 75 88	17 34 52 120
			19 38 65 92	22 47 80 113	23 51 79 102	8 16 32 77	14 40 81 95	
139	11		1 2 4 136	6 12 28 123	8 21 29 84	9 18 60 97	10 36 62 72	11 35 59 70
			14 39 53 96	15 32 47 93	19 38 68 109	20 40 74 105	23 50 73 106	
140	11		1 2 36 106	3 6 12 131	4 8 45 103	5 10 66 84	7 32 57 64	14 31 77 123

			19 38 62 116	22 51 81 111	23 49 75 98	13 33 53 100	11 27 96 112	
141	11		1 2 4 138	6 12 21 132	8 16 33 124	10 36 62 72	13 48 83 96	14 51 88 102
			18 47 65 103	19 60 80 100	22 44 68 117	27 54 82 113	11 34 77 109	
142	11		1 2 4 139	6 12 20 134	9 18 61 99	10 29 76 123	11 22 44 109	16 40 79 118
			17 51 68 105	21 42 67 117	26 53 84 115	15 30 60 101	13 36 72 85	
143	12	√	13 26 52 91	1 2 4 140	6 12 24 125	19 38 60 121	9 62 76 90	11 40 51 97
			8 44 80 88	16 50 84 100	23 54 77 110	15 32 79 126	10 20 45 118	21 48 69 106
144	12	√	1 2 37 109	14 28 67 105	7 31 55 62	19 41 60 102	3 13 23 26	5 49 93 98
			9 34 59 68	16 43 70 86	4 8 73 79	15 30 47 127	11 29 40 92	12 33 99 111
145	12	√	1 2 4 142	11 43 78 113	8 16 34 127	10 47 84 94	19 49 82 115	15 53 80 107
			14 45 76 90	6 28 50 56	13 36 59 72	12 24 41 128	9 57 77 97	21 42 81 106
146	12	√	1 2 4 143	16 32 49 81	9 18 45 128	8 47 86 94	11 22 64 104	15 30 74 102
			23 46 71 121	19 54 89 108	13 26 69 103	14 51 80 109	10 20 41 125	6 61 73 85
147	12		1 2 4 144	6 12 22 137	8 25 33 90	9 43 78 113	11 52 63 105	13 62 80 98
			14 28 72 103	15 38 61 76	19 56 83 110	20 40 79 108	21 51 81 102	24 48 74 121
148	12		1 2 38 112	3 6 12 139	4 47 76 105	5 13 18 83	7 17 24 86	11 34 91 114
			14 49 81 113	16 55 82 109	19 41 60 104	20 40 73 115	25 51 77 102	28 56 87 117
149	12		1 2 4 146	6 12 23 138	8 16 30 135	9 18 55 112	10 43 76 86	13 52 91 104
			15 51 87 102	19 54 89 108	20 40 68 121	21 42 80 111	24 49 74 123	27 56 88 120
150	12		1 2 4 147	6 12 29 133	8 21 79 137	9 18 49 119	11 22 72 111	14 51 88 102
			15 41 67 82	16 32 59 123	19 38 73 115	24 57 90 114	25 53 81 106	10 20 65 95
151	12		1 2 4 148	6 12 20 143	9 18 35 134	11 50 89 100	13 53 93 106	15 30 79 102
			16 47 63 107	19 38 74 115	21 42 67 126	24 48 81 118	27 56 83 117	10 32 75 129
152	12	√	1 2 39 115	3 6 12 143	4 8 28 132	10 45 80 90	16 32 63 121	17 34 59 127
			18 36 79 109	23 49 75 98	5 46 65 106	7 14 64 78	11 22 55 108	13 40 69 125
153	12		1 2 4 150	6 12 23 142	8 16 30 139	9 18 51 111	10 45 80 90	13 49 85 98
			15 46 84 122	19 48 67 110	20 57 94 114	21 61 87 113	24 56 88 112	25 50 78 125
154	13							
155	12		1 2 4 152	6 12 53 114	8 16 35 136	9 18 51 122	10 20 49 126	13 56 86 99
			15 40 65 80	17 55 93 110	21 52 88 124	22 44 68 131	23 57 89 121	11 37 85 96

Л И Т Е Р А Т У Р А

[1] Ц. Байчева и Кр. Манев, Намиране на линейната обвивка на множество вектори над крайно поле с характеристика различна от 2, *Доклади на XXIII Пролетна конференция на СБМ*, Ст. Загора, (1994) 313–318.

[2] Е. Великова, *Върху радиуса на покритие на класове линейни кодове*, Дисертация за присъждане на научната степен „Доктор“ (1991).

[3] Е. Николова, *Подходящи кодове за откриване на грешки*, Дисертация за присъждане на научната степен „Доктор“ (2005).

[4] И. Буюклиев, *Алгоритмични подходи за изследване на линейни кодове*, Дисертация за присъждане на научната степен „Доктор на математическите науки“, София (2007).

[5] Ст. Капралов, *Граници, конструкции и класификации на оптимални кодове*, Дисертация за присъждане на научната степен „Доктор на математическите науки“, Габрово (2004).

[6] R. J. R. Abel and M. Buratti, Some progress on (v, 4, 1) difference families and optical orthogonal codes, *J. Combin. Theory*, Ser. A, vol. 106 (2004) 59–75.

[7] R. J. R. Abel, S. Costa, N. J. Finizio, Directed-ordered whist tournaments and $(v, 5, 1)$ difference families: Existence results and some new classes of Z-cyclic solutions, *Discrete Appl. Math.*, 143 (2004) 43–53.

[8] Z. Ahang and C. Lo, Lower bounds on $t[n, k]$ from linear inequalities, *IEEE Trans. Infom. Theory*, vol. 38 (1992) 194–197.

[9] A. V. Aho, J. E. Hopcroft and J. D. Ullman, *The Analysis and Design of Computer Algorithms*, Addison-Wesley (1974).

[10] T. Baicheva, The Covering Radius of Ternary Cyclic Codes with Length up to 25, *Designs, Codes and Cryptography*, vol. 13 (1998), 223–227.

[11] T. Baicheva and E. Velikova, Covering radii of ternary linear codes of small dimensions and codimensions, *IEEE Trans. Inform. Theory*, vol. 26 (1997) 738–742.

[12] A. M. Barg and I. I. Dummer, On computing the weight spectrum of cyclic codes, *IEEE Trans. Inform. Theory*, vol. 38 (1992) 1382–1386.

[13] E. R. Berlecamp, *Algebraic Coding Theory*, New York: McGraw-Hill (1968).

[14] E. R. Berlekamp, R. J. McEliece and H. C. A. van Tilborg. On the inherent intractability of certain coding problems, *IEEE Trans. Inform. Theory*, vol. 24 (1978) 384–386.

[15] E. R. Berlecamp and L. R. Welch, Weight distributions of the cosets of the (32,6) Reed-Muller code, *IEEE Trans. Inform. Theory*, vol. 18, No. 1, Jan. (1972) 203–207.

[16] V. N. Bhat-Nayak, V. D. Kane, W. L. Kocay, R. G. Stanton, Settling some BIBD conjectures, *Ars Combin.*, vol. 16 (1983) 229–234.

[17] I. C. M. Bird, A. D. Keedwell, Design and applications of optical orthogonal codes - a survey, *Bull. Inst. Combin. Appl.*, vol. 11 (1994), 21–44.

[18] S. Bitan and T. Etzion, Constructions for optimal constant weight cyclically permutable codes and difference families, *IEEE Trans. Inform. Theory*, vol. 41 (1995) 77–87.

[19] R. E. Blahut, *The theory and practice of error control codes*, Addison-Wesley, Massachusetts (1993).

[20] R. Bosch GmbH, *CAN Specification*, Version 2.0, Sept. (1991).

[21] R. C. Bose, On the construction of balanced incomplete block designs, *Ann. Eugenics*, vol. 9 (1939) 353–399.

[22] I. Bouyukliev, On the binary projective codes with dimension 6, *Discrete Applied Mathematics*, vol. 154 (2006) 1693–1708.

[23] I. Bouyukliev, What is Q-extension? *Serdica Journal of Computing*, vol. 1 (2007) 115–130.

[24] I. Bouyukliev, About the code equivalence, in Advances in Coding Theory and Cryptography, T. Shaska, W. C. Huffman, D. Joyner and V. Ustimenko, *Series*

on Coding Theory and Cryptography, World Scientific Publishing Co. Pte. Ltd. (2007) 126–151.

[25] I. Bouyukliev and V. Bakoev, A method for efficient computing the number of codewords of fixed weights in linear codes, *Discrete Applied Mathematics*, vol. 156 (2008) 2986–3004.

[26] I. M. Boyarinov, On unequal error protection codes, *Proc. Fifth Conf. on Theory of Transmission and Coding of Inform.*, Moskow-Gorki, U.S.S.R., pt. II (1972) 22–24.

[27] I. M. Boyarinov and G. L. Katsman, On linear unequal error protection codes, *Proc. Seventh Nat. Symp. on Problems of Redundacy in Information System*, Leningrad, U.S.S.R., pt. I (1977) 66–70.

[28] I. M. Boyarinov and G. L. Katsman, Linear unequal error protection codes, *Voprosi Kibernetiki*, no. 34 (1977) 60–91.

[29] A. E. Brouwer, Bounds on the size of linear codes, in *Handbook of Coding Theory*, V. S. Pless and W. C. Huffman, eds., Elsevier, Amsterdam (1998) 295– 461.

[30] R. A. Brualdi and V. S. Pless, On the length of codes with a given covering radius, in *Coding Theory and Design Theory. Part I: Coding Theory*, Ray- Chaudhuri, ed., New York: Springer-Verlag (1990) 9–15.

[31] R. A. Brualdi, V. S. Pless and R. M. Wilson, Short codes with a given covering radius, *IEEE Trans. Inf. Theory*, vol. 36 (1990) 381–385.

[32] M. Buratti, Constructions for $(q, k, 1)$ difference families with q a prime power and $k = 4, 5$, *Discrete Mathematics*, vol. 138 (1995) 169–175.

[33] M. Buratti, From a $(G, k, 1)$ difference family to a $(C_k \oplus G, k, 1)$ difference family, *Designs, Codes and Cryptography*, vol. 11 (1997) 5–9.

[34] M. Buratti and A. Pasotti, Further progress on difference families with block size 4 or 5, *Designs, Codes and Cryptography*, vol. 56 (2010) 1–20.

[35] M. Buratti, K. Momihara, A. Pasotti, New results on optimal $(v, 4, 2, 1)$ optical orthogonal codes, *Designs, Codes and Cryptography*, vol. 58 (2011) 89—109.

[36] M. Buratti, A. Pasotti, D. Wu, On optimal $(v, 5, 2, 1)$ optical orthogonal codes, *Designs, Codes and Cryptography*, vol. 68, issue 1-3 (2013) 349–371.

[37] P. B. Busschbach, M. G. L. Gerretzen and H. C. A. van Tilborg, On the covering radius of binary linear codes meeting the Griesmer bound, *IEEE Trans. Inf. Theory*, vol. 31 (1985) 465–468.

[38] A. A. Calderbank and N. J. A. Sloane, Inequalities for covering codes, *IEEE Trans. Inf. Theory*, vol. 34 (1988) 1276–1280.

[39] G. Castagnoly, S. Bräuer and M. Herrmann, Optimization of cyclic redundancy-check codes with 24 and 32 parity bits, *IEEE Trans. Commun.*, vol. 41, June (1993) 883–892.

[40] G. Castagnoly, J. Ganz and P. Graber, Optimum cyclic redundancy-check codes with 16-bit redundancy, *IEEE Trans. Commun.*, vol. 38, January (1990) 111 –114.

[41] K. Chen, L. Zhu, Existence of $(q, 6, 1)$ difference families with q a prime power, *Designs, Codes and Cryptography*, vol. 15 (1998) 167–173.

[42] K. Chen, R. Wei, L. Zhu, Existence of $(q, 7, 1)$ difference families with q a prime power, *Journal of Combinatorial Designs*, vol. 10, Issue 2 (2002) 126–138.

[43] C. J. Colbourn and J. H. Dinitz (eds.), *The CRC Handbook of Combinatorial Designs*, CRC Press, New York (1996).

[44] M. J. Colbourn, R. A. Mathon, On cyclic Steiner 2-designs, *Ann. Discrete Math.*, vol. 7 (1980) 215–253.

[45] W. Chu, C. J. Colbourn, Optimal (n, 4, 2)- OOC of small orders, *Discrete Math.*, vol. 279 (2004) 163-–172.

[46] F. R. K. Chung, J. A. Salehi, V. K. Wei, Optical orthogonal codes: design, analysis and applications, *IEEE Trans. Inform. Theory*, vol. 35 (1989) 595–604.

[47] G. Cohen, I. Honkala, S. Litsyn and A. Lobstein, *Covering Codes*, North- Holland, Elsevier Science B.V. (1997).

[48] G. Cohen, M. Karpovsky, H. Mattson Jr., and J. Schatz, Covering radius - survey and recent results, *IEEE Trans. Inf. Theory*, vol. 31 (1985) 328– 343.

[49] G. D. Cohen, S. N. Litsyn, A. C. Lobstein and H. F. Mattson Jr., Covering radius 1985-1994, *AAECC* (Springer), vol. 8 (1997) 173–239.

[50] G. D. Cohen, A. C. Lobstein, and N. J. A. Sloane, Further results on the covering radius of codes, *IEEE Trans. on Inform. Theory*, vol. 32 (1986) 680–694.

[51] C. J. Colbourn, On cyclic Steiner systems S(2,6,91), *Abstracts Amer. Math. Soc.*, vol. 2 (1981).

[52] C. J. Colbourn and A. Rosa, *Triple systems*, Oxford University Press, Oxford (1999).

[53] C. J. Colbourn, J. H. Dinitz and D. R. Stinson, Applications of combinatorial designs to communications, cryptography, and networking, *Surveys in combinatorics*, J. D. Lamb, D. A. Preece (eds.), Cambridge University Press, London (1999) 37–100.

[54] D. Danev and S. Dodunekov, A family of ternary quasi-perfect codes, *Proc. International workshop on coding and cryptography*, Versailles, France, April 16-20 (2007) 109–115.

[55] A. A. Davydov, Constructions and families of covering codes and saturated sets of points in projective geometry, *IEEE Trans. Inform. Theory*, vol. 41 (1995) 2071–2080.

[56] A. A. Davydov, Constructions and families of nonbinary linear codes with covering radius 2, *IEEE Trans. Inf. Theory*, vol. 45, No. 5 (1999) 1679 –1686.

[57] A. A. Davydov, G. Faina, S. Marcugini and F. Pambianco, Locally optimal (non-shortening) linear covering codes and minimal saturating sets in projective spaces, *IEEE Trans. Inf. Theory*, vol. 51, No. 12 (2005) 4378–4387.

[58] A. A. Davydov, S. Marcugini and F. Pambianco, On saturating sets in projective spaces, *J. Combin. Theory, Ser. A*, vol. 103 (2003) 1–15.

[59] A. A. Davydov, S. Marcugini and F. Pambianco, Linear codes with covering radius 2,3 and saturating sets in projective geometry, *IEEE Trans. Inf. Theory*, vol. 50 (2004) 537–541.

[60] A. A. Daviydov, S. Marcugini and F. Pambianco, Minimal 1-saturating sets and complete caps in binary projective spaces, *J. Combinatorial Theory*, Ser. A, vol. 113 (2006) 647–663.

[61] M. A. de Boer, Almost MDS codes, *Designs, Codes and Cryptography*, vol. 9 (1996) 143–155.

[62] P. Delsarte, Four fundamental parametrs of a code and their combinatorial significance, *Information and Control*, vol. 23 (1973) 407–438.

[63] H. J. Dinitz, N. Shalaby, Block disjoint difference families for Steiner tripple systems: $v \equiv 3$ (*mod* 6), *J. Stat. Plann. Infer.*, vol. 106 (2002) 77–86.

[64] S. M. Dodunekov, Minimal block length of a q-ary code with prescribed dimension and code distance, *Probl. Inform. Transm.*, vol. 20, No 4 (1964) 239–249.

[65] S. M. Dodunekov, The optimal double-error correcting codes of Zetterberg and Dumer-Zinov'ev are quasiperfect, *C. R. Acad. Bulgare Sci*, vol. 38, No 9 (1985) 1121–1123.

[66] S. M. Dodunekov, Some quasiperfect double error correcting codes, *Problems Control Inform. Theory/Problemy Upravlen. Teor. Inform.*, vol. 15, No 5 (1986) 367–375.

[67] S. M. Dodunekov, Griesmer codes with maximum covering radius, *Problemy Peredachi Informatsii*, vol. 23, No. 4 (1987) 110–113. Translated in: *Problems of Inform. Transm.*, vol. 23, No. 4 (1987) 344–346.

[68] S. M. Dodunekov and I. N. Landgev, On near-MDS codes, *Report LiTH-ISY-R-1563, Dept. of Elec. Eng., Linköping Univ.* (1994).

[69] S. M. Dodunekov and N. L. Manev, Covering radius of optimal binary [15,6,6] codes (in Russian), *Proc. of the Soviet-Swedish International Workshop on Information Theory*, Sochi (1987) 64–66.

[70] R. Dodunekova and S. Dodunekov, Sufficient conditions for good and proper linear error correcting codes, *Proc. Second International Workshop on Optimal Codes and Related Topics*, Sozopol, Bulgaria (1998) 62–67.

[71] R. Dodunekova and E. Nikolova, Sufficient condotions for the monotonicity of the undetected error probability for large channel error probabilities, *Problemy Peredachi Informatsii*, vol. 41 (2005) 3-16 (in Russian); English translation: *Problems of Information Transsmision*, vol. 41 (2005) 187– 198.

[72] R. Dodunekova and E. Nikolova, Intervals for properness of binary linear error-detecting codes, *Preprint N 2004-42, Chalmers University of Thechnology and Göteberog University* (2004).

[73] R. Dodunekova and E. Nikolova, Properness of binary inear error-detecting codes in terms of basic parameters, *Proc. of the Forth Intern. Workshop on Optimal Codes and Related Topics*, Pamporovo, Bulgaria, (2005) 9–15.

[74] R. Dougherty and H. Janwa, Covering radius computations for binary cyclic codes, *Mathematics of Computation*, vol. 37, No. 195 (1991) 415–434.

[75] D. Downie and N. J. A. Sloane, The covering radius of cyclic codes of length up to 31, *IEEE Trans. Inf. Theory*, vol. 31 (1985) 446–447.

[76] I. I. Dumer and V. A. Zinov'ev, Some new maximal codes over $GF(4)$, *Problems of Information Transmission*, vol. 14, No. 3 (1978) 174–181.

[77] L. A. Dunning and W. E. Robbins, Optimal encodings of linear block codes for unequal error protection, *Inform. Contr.*, vol. 37 (1978) 150–177.

[78] V. N. Dynkin and V. A. Togonidze, Cyclic codes with unequal protection of symbols, *Probl. Peredach. Inform.*, vol. 12, no. 1 (1976) 24–28.

[79] M. C. Er, On generating the N-ary reflected Gray codes, *IEEE Trans. Comuput.*, vol. 33 (1984) 739–741.

[80] T. Etzion, G. Greenberg and I. Honkala, Normal and abnormal codes, *IEEE Trans. Inf. Theory*, vol. 39 (1993) 1453–1456.

[81] T. Etzion and B. Mounits, Quasi-perfect codes with small distance, *IEEE Trans. Inf. Theory*, vol. 51, No 11 (2005) 3938–3946.

[82] M. van Eupen and J. H. van Lint, On the minimum distance of ternary cyclic codes, *IEEE Trans. Inf. Theory*, vol. 39 (1993) 409–422.

[83] G. Faina, S. Marcugini, A. Milani and F. Pambianco, The sizes k of complete k-caps in $PG(n, q)$ for small q and $3 \leq n \leq 5$, *Ars Combinatoria*, vol. 50 (1998) 235–243.

[84] A. Faldum and W. Willems, Codes of small defect, *Designs, Codes and Cryptography*, vol. 10 (1997) 341–350.

[85] A. Faldum and W. Willems, A characterization of MMD codes, *IEEE Inform. Theory*, vol. 44 (1998) 1555–1558.

[86] FlexRay Consortioum, *FlexRay communication system protocol specification*, Version 2.1, May (2005).

[87] A. B. Fontaine and W. W. Peterson, Group code equivalence and optimum codes, *IRE Trans.*, vol. 5, Special Supplement (1959) 60–70.

[88] T. C. Frenz, *Computing Techniques for the Enumeration of Cyclic Steiner Systems*, Ph.D. Thesis, Rochester Institute of Technology School of Computer Science (1989).

[89] R. Fuji-hara, Y. Miao, Optical orthogonal codes: Their bounds and new optimal constructions, *IEEE Trans. on Inform. Theory*, vol. 46 (2000) 2396-–2406.

[90] R. Fuji-Hara, Y. Miao and S. Shinohara, Complete Sets of Disjoint Difference Families and their Applications, *Journal of Statistical Planning and Inference*, vol. 106, 1 August (2002) 87–103.

[91] M. Fujisawa, S. Sakata, A class of quasi-cyclic regular LDPC codes from cyclic difference families with girth 8, *Proceedings International Symposium on Information Theory*, 4-9 Sept. (2005) 2290–2294.

[92] T. Fujiwara, T. Kasami, A. Kitai and S. Lin, On the undetected error probability for shorthened Hamming codes, *IEEE Trans. Commun.*, vol. 33, June (1985) 570–574.

[93] G. Funk, Determination of best shortened linear codes, *IEEE Trans. Commun.*, vol. 44, January (1996) 1–6.

[94] E. M. Gabidulin, A. A. Davydov and L. M. Tombak, Linear codes with covering radius 2 and other new covering codes, *IEEE Trans. Inf. Theory*, vol. 37, No. 1 (1991) 219–224.

[95] E. Gabidulin and T. Kløve, On the Newton radius, *Reports in Informatics* (Dept. Informatics, Univ. Bergen, Bergen, Norway), no. 130, Feb. (1997).

[96] E. Gabidulin and T. Kløve, On the Newton and covering radii of linear codes, *IEEE Trans. on Inform. Theory*, vol. 45, No. 7 (1999) 2534–2536.

[97] M. R. Garey and D. S. Johanson, *Computers and Intactability: A Guide to the Theory of NP-completeness*, San Francisco: Freeman (1978).

[98] I. B. Gashkov and V. M. Sidel'nikov, Linear ternary quasiperfect codes correcting double errors, *Problems of Information Transmission*, vol. 22 , No. 4 (1986) 284–288.

[99] D. N. Gevorkijan, A. M. Avetisjan and G. A. Tigranjan, On the construction of codes correcting two errors in Hamming's metrix over Galois field, *Vichislitel'naja tehnika*, vol. 3 (1975) 19–21 (in Russian).

[100] E. N. Gilbert, Cyclically permutable error-correcting codes, *IEEE Trans. Inform. Theory*, vol. 9 (1963) 175–180.

[101] W. J. van Gils, Two topics on linear unequal error protection codes: Bounds on their length and cyclic code classes, *IEEE Trans. Inform. Theory*, vol. 29, no. 6, Nov. (1983) 866–876.

[102] M. Giulietti and F. Pasticci, Quasi-perefect linear codes with minimum distance 4, *IEEE Trans. Inf. Theory*, vol. 53, No. 5 (2007) 1928–1935.

[103] R. L. Graham and N. J. A. Sloane, On the covering radius of codes, *IEEE Trans. Inf. Theory*, vol. 31 (1985) 385–401.

[104] T. Helleseth, On the covering radius of cyclic linear codes and arithmetic codes, *Discrete Applied Mathematics*, vol. 11 (1985) 157–173.

[105] T. Helleseth, T. Kløve and J. Mykkeltveit, The weight distribution of irreducible cyclic codes, *Discrete Mathematics*, vol. 18 (1977) 179–211.

[106] T. Helleseth, T. Kløve and V. Levenshtein, The Newton radius of equidistant codes, *Proc. IEEE Intern. Symp. on Inform. Theory and its Applications*, Victoria, B.C., Canada, Sept. 17-30 (1996) 721–722.

[107] T. Helleseth and T. Kløve, The Newton radius of codes, *IEEE Trans. on Inform. Theory*, vol. 43, No. 6 (1997) 1820–1831.

[108] J. W. P. Hirschfeld and L. Storme, The packing problem in statistics, coding theory and finite projective spaces: Update 2001, in *Developments in Mathematics, vol. 3, Finite geometries*, A. Blokhuis, J. W. P. Hirshfeld, D. Jungnickel, and J. A. Thas, eds. Dordrecht, The Netherlands: Kluwer (2000) 201 –246.

[109] Xiang-Dong Hou, Some results on the norm of codes, *IEEE Trans. on Inform. Theory*, vol. 36 (1990) 683–685.

[110] Xiang-Dong Hou, *Covering radius and error-correcting codes*, PhD Thesis, University of Illinois, Chicago, US (1990).

[111] Xiang-dong Hou, Binary linear quasi-perfect codes codes are normal, *IEEE Trans. on Inform. Theory*, vol. 37, No. 2 (1991) 378–379.

[112] Xiang-dong Hou, The Reed-Muller code R(1,7) is normal, *Designs, Codes and Cryptography*, vol. 12 (1997), 75-92.

[113] Xiang-dong Hou, On the norm and covering radius of the first-order Reed- Muller codes, *IEEE Trans. on Inform. Theory*, vol. 43 (1997) 1025–1027.

[114] M. Huber, Perfect Secrecy Systems Immune to Spoofing Attacks, *International Journal of Information Security*, vol. 11, Issue 4 (2012) 281–289.

[115] J. F. Humphreys, Algebraic decoding of the ternary $[13, 7, 5]$ quadratic- residue code, *IEEE Trans. Inform. Theory*, vol. 38, No. 3 (1992) 1122– 1125.

[116] K. Imamura, K. Tokiwa and M. Kasahara, On computation of the binary weight distribution of some Reed-Solomon codes and their extended codes, *Proc. Int. Colloc. Coding Theory*, Osaka, Japan (1988) 195–204.

[117] D. Jaffe, *Binary Linear Codes: New Results on Nonexistence*, Draft (Version 0.4), Department of Mathematics and Statistics, University of Nebraska, April 14 (1997).

[118] Z. Janko and V. D. Tonchev, Cyclic 2-(91, 6, 1) designs with multiplier automorphisms, *Discrete Mathematics*, vol. 97 (1991) 265–268.

[119] H. Janwa and H. F. Mattson Jr., Some upper bounds on the covering radii of linear codes over F_q and their applications, *Designs, Codes and Cryptography*, vol. 18 (1999) 163–181.

[120] D. Julin, Two improved block codes, *IEEE Trans. Inf. Theory*, vol. 1 (1965) p. 450.

[121] R. M. Karp, Reducibility among combinatorial problems, in *Complexity of computer computations*, R. Miller and J. Thatcher, eds. New York: Plenum (1972) 85–103.

[122] R. M. Karp and Y. Zhang, Randomized parallel algorithms for backtrack search and branch-and-bound computation, *Journal Assoc. Comput. Mach. (USA)*, vol. 40, (3) (1993) 765-789.

[123] T. Kasami, The weight enumerators for several classes of subcodes of the 2nd order binary Reed-Muller codes, *Information and Control*, vol. 18 (1971) 369–394.

[124] T. Kasami and S. Lin, On the probability of undetected error for the Maximum-Distance Separable codes, *IEEE Trans. Communic.*, vol. 32 (1984) 998–1006.

[125] T. Kasami and S. Lin, The binary weight distribution of the extended $(2^m, 2^m-4)$ code of Reed-Solomon codes over $GF(2^m)$ with generator polynomial $(x-\alpha)(x-\alpha^2)(x-\alpha^3)$, *Linear Algebra an Its Applications*, vol. 98 (1984) 291–307.

[126] P. Kaski and P. R. J. Östergård, *Classification algorithms for codes and designs*, Springer, Berlin (2006).

[127] W. H. Kautz, A readily implemented single-error correcting unit-distance counting code, *IEEE Trans. Comput.*, vol. 19 (1970) 972–975.

[128] P. Kazakov, Fast Calculation on the Number of Minimum Weight Words of CRC Codes, *IEEE Trans. Inf. Theory*, vol. 49, March (2001) 1190–1195.

[129] M. Khatirinejad and P. Lisoněk, Classification and constructions of complete caps in binary spaces, *Designs, Codes and Cryptography*, vol. 39 (2006) 17–31.

[130] K. E. Kilby and N. J. A. Sloane, On the covering radius problem for codes: I Bounds on normalized covering radius, II Codes of low dimension; normal and abnormal codes, *SIAM J. Algebraic and Discrete Methods*, vol. 8 (1987) 604–627.

[131] C. C. Kilgus and W. C. Gore, A class of cyclic unequal-error-protection codes, *IEEE Trans. Inform. Theory*, vol. 18, Sept. (1972) 687–690.

[132] T.Kløve and V. Korzhik. *Error Detecting Codes*, Kluwer Academic Publishers, Boston (1995).

[133] T. Kløve, Relations between the covering and Newton radii of binary codes, *Discrete Mathematics*, vol. 238 (2001) 81–88.

[134] T.Kløve, *Codes for error detection*, Series on Coding Theory and Cryptology, vol. 2, World Scientific (2007).

[135] E. Kolev and N. Manev, The binary weight distribution of an extended $(2M, 5)$ Reed-Solomon code and its dual, *Doklady Bolgarskoi Akademii Nauk*, vol. 43, Iss. 9 (1990) 9–12.

[136] P. Koopman and T. Chakravarty, Cyclic Redundancy Code (CRC) polynomial selection for embedded networks, *Proc. of Internat. Conf. on Dependable Systems and Networks, DSN04* (2004) 145–154.

[137] P. Koopman, 32-bit cyclic redundacy codes for internet applications, *Proc. Int. Conf. Dependable Systems and Networks (DSN'02)*, June (2002) 459– 468.

[138] F. R. Kschischang and S. Pasupathy, Some ternary and quaternary codes and associated sphere packings, *IEEE Trans. Inform. Theory*, vol. 38, No 2 (1992) 227–246.

[139] R. Lidl and H. Niederreiter, *Finite fields, Encyclopedia of Mathematics and Its Applications*, Vol. 20, Cambridge University Press, Cambridge (1983).

[140] R. Lidl and H. Niederreiter, *Introduction to finite fields and their application,* Rev. Edition, Cambridge University Press (1994).

[141] S. Lin and D. J. Costello, *Error Control Coding, Fundamentals and Applications*, Prentice-Hall Publ., Englewood Cliffs, New Jersey (1983).

[142] M. C. Lin and S. Lin. Cyclic unequal error protection codes constructed from cyclic codes of composite length, *IEEE Trans. Inform. Theory*, vol. 34, no. 4, July (1988) 867–871.

[143] J. H. van Lint, *Introduction to Coding Theory*, Springer-Verlag, New York (1982).

[144] F. J. MacWilliams, A theorem on the distribution of weights in a systematic code, *The Bell System Tech. J.*, vol. 42 (1963) 79–94.

[145] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting codes*, North-Holland Publishing Company, Amsterdam, London, New York, Tokyo, Ninth impression (1996).

[146] D. Mandelbaum, Unequal-error-protection codes derived from difference sets, *IEEE Trans. Inform. Theory*, vol. 18, Sept. (1972) 686–687.

[147] K. Manev, LINCOR - a system for linear codes researches, *Proc. of the XVI Spring Conference of the UBM* (1987) 500–503.

[148] K. Manev and R. Stefanov, Yet another algorithm for addition of vectors in non binary finite field, *Proc. of International Workshop on Algebraic and Combinatorial Coding Theory*, Sozopol, Bulgaria, (1996) 190–193.

[149] B. Masnik and J. Wolf, On linear unequal error protection codes, *IEEE Trans. Inform. Theory*, vol. 13, Oct. (1967) 600–607.

[150] H. Mattson Jr., An improved upper bound on covering radius, *Lecture notes in Computer Science*, No. 228, Springer-Verlag (1986) 90–106.

[151] R. J. McEliece and H. Rumsey, Euler products, cyclotomy, and coding, *J. Number Theory*, vol. 4 (1972) 302–311.

[152] B. D. McKay, Nauty users's guide (version 1.5), Comp. Sci. Dept. Australian Nat. Univ., Tech. Rep. TR-CS-90-02 (1990).

[153] B. D. McKay, Isomorph-free exaustive generation, *J. Algorithms*, vol. 26 (1998) 306–324.

[154] A. McLoughlin, The complexity of computing the covering radius of a code, *IEEE Trans. on Inform. Theory*, vol. 30 (1984) 800–804.

[155] P. Merkey and E. C. Posner, Optimum cyclic redundacy codes for noisy channels, *IEEE Trans. Inf. Theory*, vol. 30 (1984) 865–867.

[156] K. Momihara, New optimal optical orthogonal codes by restrictions to subgroups, *Finite Fields Appl.*, vol. 17 (2011) 166–182.

[157] K. Momihara and M. Buratti, Bounds and Constructions of Optimal (n, 4, 2, 1) Optical Orthogonal Codes, *IEEE Trans. on Inform. Theory*, vol. 55 (2009) 514–523.

[158] O. Moreno, Z. Zhang, P. V. Kumar and V. A. Zinoviev, New constructions of optimal cyclically permutable constant weight codes, *IEEE Trans. on Inform. Theory*, vol. 41 (1995) 448–455.

[159] M. Muzychuk, A solution of the isomorphism problem for circulant graphs, *Proc. London Math. Soc.*, (3) 88 (2004) 1–41.

[160] S. L. Ng and B. Dewar, Parallel realization of the ATM cell header CRC, *Computer Communications*, vol. 19 (1996) 257–263.

[161] Q. A. Nguyen, L. Györfy and J. L. Massey, Construction of binary constant-weight cyclic codes and cyclically permutable codes, *IEEE Trans. on Inform. Theory*, vol. 38 (1992) 940–949.

[162] S. C. Ntafos and S. L. Hakimi, On the complexity of some coding problems,*IEEE Trans. on Inform. Theory*, vol. 27 (1981) 794–796.

[163] J. Olsson and W. Williems, A characterization of sertain Griesmer codes: MMD codes in a more general sense, *IEEE Trans. on Inform. Theory*, vol. 45 (1999) 2138–2142.

[164] P. R. J. Östergård, New constructions for q-ary covering codes, *Ars Combinatoria*, vol. 52 (1999) 51-63.

[165] P. R. J.Östergård, Classifying subspaces of Hamming spaces, *Designs, Codes and Cryptography*, vol. 27 (2002) 297–305.

[166] P. R. J. Östergård and M. K. Kaikkonen, New upper bounds for binary covering codes, *Discrete Mathematics*, vol. 178 (1998) 165–179.

[167] W. W. Peterson, *Error-Correcting Codes*, MIT Press, Cambridge, Mass. (1961).

[168] W. W. Peterson and E. J. Weldon, *Error-Correcting Codes*, MIT Press, Cambridge, Mass. (1972).

[169] V. S. Pless and W. C. Huffman, *Handbook on Coding Theory*, Elsevier Science B.V. (1998).

[170] D. K. Pradhan and S. K. Gupta, A new framework for designing and analyzing BIST techniques and zero aliasing compression, *IEEE Trans. Comput.*, vol. 40 (1991) 743–763.

[171] E. M. Reingold, J. Nievergelt and N. Deo, *Combinatorial Algorithms: Theory and Practice*, Prentice-Hall, Englewood Cliffs, NJ (1976).

[172] J. Ray and P. Koopman, Efficient high hamming distance CRCs for embedded networks, *Proc. of Dependable Systems and Networks*, Philadelphia PA, June 25-28 (2006).

[173] C. Retter, The average binary weight-enumerator for a class of generalized Reed-Solomon codes, *IEEE Trans. Infrom. Theory*, vol. 37 (1991) 346–349.

[174] C. Retter, Gaps in the binary weight distribution of Reed-Solomon codes, *IEEE Trans. Infrom. Theory*, vol. 38 (1992) 1688–1697.

[175] J. A. Salehi, Code Division Multiple-Access Techniques in Optical Fiber Networks - Part I: Fundamental Principles, *IEEE Trans. Commun.*, vol. 37, no. 8 (1989) 824–33.

[176] N. R. Saxena and E. J. McCluskey, Analysis of checksums, extended-precision checksums, and cyclic redundancy checks, *IEEE Trans. Computers*, vol. 39, July (1990) 969-975.

[177] R. Segal and R. Ward, Weight distributiona of some irreducible cyclic codes, *Mathematics of Computation*, vol. 46 (1986) 341–354.

[178] C. E. Shannon, A mathematical theory of communication, *Bell Syst. Tech. J.*, vol. 27 (1948) 379–423 and 623–656.

[179] J. Simonis, The $[23, 14, 5]$ Wagner code is unique, *Discrete Mathematics*, vol. 213 (2000) 269–282.

[180] J. Simonis, The minimal covering radius $t[15, 6]$ of a 6-dimensional binary linear code of length 15 is equal to 4, *IEEE Trans. Infrom. Theory*, vol. 34 (1998) 1344–1345.

[181] D. Slepian, A class of binary signaling alphabets, *BSTJ* vol. 35, Jan. (1956) 203–234.

[182] N. J. A. Sloane, A new approach to the covering radius of codes, *J. of Combinatorial Theory*, ser. A, vol. 42 (1986) 61–86.

[183] F. I. Solov'eva, Designs and Perfect Codes, General Theory of Information Transfer and Combinatorics, *Lecture Notes in Computer Science*, vol. 4123 (2006) 1104–1105.

[184] D. R. Stinson, R. Wei, J. Yin, Packings, *The CRC handbook of combinatorial designs*, 2nd edn., C. J. Colbourn, J. H. Dinitz (eds.), Chapman and Hall/CRC Press, Boca Raton, FL (2006) 550-–556.

[185] L. J. Stockmeyer, The polynomial-time hierarchy, *Theor. Comp. Sci.*, vol. 3 (1976) 1–22.

[186] R. Struik, On the structure of linear codes with covering radius two and three, *IEEE Trans. Infrom. Theory*, vol. 40 (1994) 1406–1416.

[187] R. Struik, *Covering codes*, PhD Thesis, Eindhoven University of Technology, the Netherlands (1994).

[188] A. Tietäväinen, On the nonexistence of perfect codes over finite fields, *SIAM J. Appl. Math.*, vol. 24 (1973) 88–96.

[189] H. C. A. van Tilborg, *Uniformly Packed Codes*, Eindhoven, The Netherlands: Eindhoven Tech. University (1976).

[190] TTTech Computertechnik AG, *Time triggered protocol TTP/C high-level specification document*, Pritocol version 1.1, specification ed. 1.4.3, Nov. (2003).

[191] A. Vardy, The intractability of computing minimum distance of a code, *IEEE Trans. on Inform. Theory*, vol. 43, No. 6 (1997) 1757–1766.

[192] E. Velikova and K. Manev, The covering radius of cyclic codes of lengths 33, 35 and 39, *Annuaire de L'Université de Sofia*, vol. 81 (1987) 215–223.

[193] E. Velikova, Bounds on the covering radius of linear codes, *Comtes-Rendus de l'Académie Bulgre des Sciences*, vol. 41 (1988) 13–16.

[194] E. Velikova, Covering radius of some cyclic codes, *Proc. of International Workshop on Algebraic and Combinatorial Coding Theory*, Varna (1988) 165–169.

[195] E. Velikova, The covering radius of two-dimensional codes over $GF(4)$, *Proc. of Workshop on Algebraic and Combinatorial Coding Theory*, Novgorod, Russia (1994) 190–193.

[196] E. Velikova, A generalization of some upper bounds on covering radius under an arbitrary additive metric, *Problems of Control and Information Theory*, vol. 19 (5-6) (1990) 445–450.

[197] T. Wagner, A search technique for quasi-perfect codes, *Information and Control*, vol. 9 (1966) 94–99.

[198] T. Wagner, Some additional quasi-perfect codes, *Information and Control*, vol. 10 (1967) p. 334.

[199] X. Wang, Y. Chang, Further results on $(v, 4, 1)$-perfect difference families, *Discrete Math.*, vol. 310 (2010) 1995–2006.

[200] R. Ward, Weight enumerators of more irreducible cyclic binary codes, *IEEE Trans. Infom. Theory*, vol. 39 (1993) 1701–1709.

[201] S. Wicker, *Error control for digital communication and storage*, Englewood Cliffs: Prentice Hall (1995).

[202] K. A. Witzke and C. Leung, A comparison of some error detecting CRC code standarads, *IEEE Trans. Commun.*, vol. 33 (1985) 996–998.

[203] J. Wolf and R. Blakeney, An exact evaluation of the probability of undetected error for certain shortened binary CRC codes, *IEEE Militariy Communications Conference*, New York, USA (1988) 287–292.

[204] J. Wolf, A. Michelson and A. Levesque, On the Probability of Undetected Error for Linear Block Codes, *IEEE Trans. Commun.*, vol. 30 (1982) 317– 325.

[205] S. K. Zaremba, Covering problems concerning abelian groups, *J. London Math. Soc.*, vol. 27 (1952) 242–246.

[206] G. C. Yang and T. E. Fuja, Optical orthogonal codes with unequal auto- and cross-correlation constraints, *IEEE Trans. on Inform. Theory*, vol. 41 (1995) 96–106.

[207] V. A. Zinov'ev and V. K. Leont'ev, On non-existence of perfect codes over Galois fields, *Problems of Control and Information Theory/Problemy Upravlenija i Teorii Informazii*, vol. 2 (1973) 123–132.

Публикации по дисертацията

[208] T. Baicheva, S. Dodunekov and P. Kazakov, On the cyclic redundancy-check codes with 8-bit redundancy, *Computer Communications*, vol. 21 (1998) 1030–1033.

[209] T. Baicheva, Binary and ternary linear codes which are good and proper for error correction, *Proc of the International Workshop on Algebraic and Combinatorial Coding Theory*, Bansko, Bulgaria (2000) 55–60.

[210] T. Baicheva, S. Dodunekov and P. Kazakov, On the Undetected Error Probability Performance of Cyclic Redundancy-Check Codes of 16-bit Redundancy, *IEE Proc. Communications*, vol. 147, No. 5 (2000) 253 –256.
T. Baicheva, S. Dodunekov and P. Kazakov, On the cyclic redundancy-check codes with 16-bit redundancy, *Proc. of the International Workshop on Algebraic and Combinatorial Coding Theory*, Pskov, Russia (1998) 17–21.

[211] T. Baicheva, On the covering radius of ternary negacyclic codes with length up to 26, *IEEE Trans. on Inform. Theory*, vol. 47, No. 1 (2001) 413–416.
T. Baicheva, On the covering radius of ternary negacyclic codes with length up to 26, *Proc of IEEE Intern. Symposium of Inform. Theory*, Sorento, Italy (2000) p. 392.

[212] T. Baicheva, S. Dodunekov and R. Kötter, On the Performance of the Ternary [13,7,5] Quadratic-Residue Codes, *IEEE Trans. Inform. Theory*, vol. 48, No. 2 (2002) 562–564.
T. Baicheva, S. Dodunekov and R. Kötter, On the Performance of the Ternary [13,7,5] Quadratic-Residue Codes, *Proc of the International Workshop on Algebraic and Combinatorial Coding Theory*, Pskov, Russia (1998) 93–97.

[213] T. Baicheva, The Newton radius of some binary and ternary cyclic codes, *Proc of the International Workshop on Algebraic and Combinatorial Coding Theory*, Tsarskoe Selo, Russia (2002) 18–21.

[214] T. Baicheva and V. Varek, On the least covering radius of binary linear codes with small lengths, *IEEE Trans. on Inform. Theory*, vol. 49, No. 3 (2003) 738–740.
T. Baicheva and V. Varek, On the least covering radius of binary linear codes with small lengths, *Proc. of the EuroWorkshop on Optimal Codes and related topics*, Sunny Beach, Bulgaria (2001) 13–18.

[215] T. Baicheva and I. Boyukliev, On the ternary projective codes with dimensions 4 and 5, *Proc of the International Workshop on Algebraic and Combinatorial Coding Theory*, Kranevo, Bulgaria (2004) 34–39.

[216] T. Baicheva and I. Gancheva, Computer search for ternary cyclic and negacyclic LUEP codes of lengths up to 26, *Mathematica Balkanica*, New Series vol. 18 (2004) 79–80.

[217] T. Baicheva, I. Boyukliev, S. Dodunekov and W. Willems, Teaching linear codes, *Mathematica Balkanica*, New Series vol. 19 (2005) 3–16.

[218] E. Velikova and T. Baicheva, On the computation of the weight distribution of the cosets of cyclic codes, *Annuarie de L'Université de Sofia 'St. Kl. Ohridski'*, vol. 97 (2005) 109–114.

[219] T. Baicheva and F. Sallam, On the error detection performance of some CRC codes, *Proc. of the workshop 'Mathematics, Informatics and Computer Sciences'*, Veliko Tarnovo, Bulgaria (2006) 107–110.

[220] T. Baicheva and F. Sallam, Error control performance of CRC codes with up to 8 bit redundancy, *Proc. of the International Workshop on Algebraic and Combinatorial Coding Theory*, Zvenigorod, Russia (2006) 11–14.

[221] T. Baicheva, On the error correcting performance of some binary and ternary linear codes, *Serdica, Journal of Computing*, vol. 1 (2007) 157–170.

[222] T. Baicheva and F. Salam, CRC codes for error control, *Mathematica Balkanica*, New series vol. 21, Fasc. 3-4 (2007) 377–388.

[223] T. Baicheva, I. Bouyukliev, S. Dodunekov, and V. Fack, Binary and Ternary Quasi-perfect Codes with Small Dimensions, *IEEE Trans. on Inform. Theory*, vol. 54, issue 9 (2008) 4335–4339.
T. Baicheva, I. Bouyukliev, S. Dodunekov and V. Fack, Binary and ternary quasi-perfect codes with small dimensions, *Proc. of the International Workshop Optimal Codes and Related Topics*, White Lagoon, Bulgaria (2007) 13–18.

[224] T. Baicheva, Determination of the best CRC codes with up to 10-bit redundancy, *IEEE Trans on Communic.*, vol. 56, issue 8 (2008) 1214 –1220.

[225] T. Baicheva, Linear codes of good error control performance, *Enchancing cryptographic primitives with techniques from error correcting codes*, IOS Press (2009) 250–259.

[226] T. Baicheva and I. Bouyukliev, On the least covering radius of the binary linear codes of dimension 6, *Advances in Mathematics of Communications*, vol. 4, No 3 (2010) 399–403.

T. Baicheva and I. Bouyukliev, On the least covering radius of the binary linear codes of dimension 6, *Proc. of the International Workshop on Algebraic and Combinatorial Coding Theory*, Pamporovo, Bulgaria (2008) 7–12.

[227] T. Baicheva, All binary linear codes of lengths up to 18 or redundancy up to 10 are normal, *Advances in Mathematics of Communications*, vol. 5, No 4 (2011) 681–686.

T. Baicheva, Normality of some binary linear codes, *Proc. of the International Workshop Optimal Codes and Related Topics*, Varna, Bulgaria (2009) 5–10.

[228] T. Baicheva and S. Topalova, Optimal (v,4,2,1) optical orthogonal codes with small parameters, *Journal of Combinatorial Designs*, vol. 20 (2) (2012) 142–160.

[229] T. Baicheva and S. Topalova, Optimal optical orthogonal codes of weight 5 and small lengths, *International Conference on Applications of Computer Algebra*, Sofia, Bulgaria (2012).

[230] T. Baicheva and S. Topalova, Optimal $(v, 3, 1)$ binary cyclically permutable constant weight codes with small v, *Proc. of the International Workshop on Algebraic and Combinatorial Coding Theory*, Pomorie, Bulgaria (2012) 33–38.

[231] T. Baicheva and S. Topalova, Classification results for (v, k, 1) cyclic difference families with small parameters, *Mathematics of Distances and Applications*, M. Deza, M. Petitjean, K. Markov (eds.), ITHEA, Sofia (2012) 24–30.

[232] T. Baicheva and S. Topalova, Optimal (v, 5, 2, 1) optical orthogonal codes of small v, *Applical Algebra in Engeneering Communication and Computing*, vol. 24, numbers 3-4 (2013) 165–177.

Списък на цитирания

[211] T. Baicheva, S. Dodunekov and P. Kazakov, On the cyclic redundancy-check codes with 8-bit redundancy, *Computer Communications*, vol. 21 (1998) 1030–1033.

1. R. Dodunekova, *On the binomial moments of linear codes and undetected error probability*, Preprint 2002:49, Chalmers University of Technology, 14 p.

2. R. Dodunekova, The duals of MMD codes are proper for error detection. *IEEE Trans. Inform. Theory*, vol.49, No.8, 2003, pp. 2034-2038.

3. R. Dodunekova, The extended binomial moments of a linear code and the undetected error probability, *Problems of Information Transmission*, v.39, No.3, 2003, pp.255-265.

4. Q. Zhou and X. Wang, Performance analysis on extended-shortened codes. *J. of Chongqing university of posts and telecommunications (natural sciences)*, V.16, No.3, 2004, pp.115-117.

5. Q. Zhou and X. Wang, Performance analysis and study on linear extended shortened codes, *Management&control technology*, V.23, No.9, 2004, pp.64-66.

6. P. Koopman and T. Chakravarty, Cyclic Redundancy Check (CRC) Polynomial Selection for Embedded Networks, *Proc. of the International Conference on Dependable Systems and Networks*, DSN-2004, pp.145-154.

7. F. Worm, P. Ienne and P. Thiran, Soft self-synchronizing codes for self-calibrating communication, *IEEE/ACM International Conference on Computer-Aided Design, Digest of Technical Papers, ICCAD*, 2004, pp.440-447.

8. Q. Zhou, X. Wang and H. Ge, Performance analysis and study on linear extended shortened codes, *Xibei Gongye Daxue Xuebao/Journal of Northwestern Polytechnical University*, 22(5), 2004, pp.640-643.

9. E. Nikolova, *Подходящи кодове за откриване на грешки*, Дисертация за присъждане на научната степен 'Доктор', 2005.

10. R. Dodunekova, O. Rabaste and J.L.V. Páez, Error detection with a class of irreducible binary cyclic codes and their dual codes, *IEEE Trans. on Inform. Theory*, vol. 51, No. 4, pp. 1206-1209, 2005.

11. A. Youssef, *Reseau de communication a haut niveau d'integrite*, PhD Thesis No. 2292, 2005, INPT, France.

12. F. Worm, *Robust checkers for self-calibrating designs*, PhD Thesis No. 3647, 2006, Lausanne, EPFL.

13. T. C. Maxino, *The effectiveness of checksums for embedded networks*, MS Thesis, Carnegie Melon University, Pttsburg, USA, May, 2006.

14. Z. Zhi, J. B. Tan, H. D. Huang and F. F. Chen, Algorithms for high-speed generating CRC error detection coding in separated ultra-precision measurement, *Journal of Physics: Conference Series*, 48, pp. 228-232, 2006.

15. Файза А. Р. Салам Муджахед, *Предизвикателства към сигурността в информационна система, базирана на УЕВ технологиите*, Дисертация за присъждане на научната степен 'Доктор', ИМИ-БАН, 2007.

16. T. Klove, *Codes for error detection*, Series on Coding Theory and Cryptology, vol 2., World Scientific, 2007.

17. T. Mattes, F. Schille1, A. Mörwald, J. Pfahler and T. Honold, Safety proof of Combinations of CRC for Industrial Communication, *Journal of Applied Computer Science*, Vol. 16. No 1, 2008, pp. 15-32.

18. F. Schiller, T. Mattes, Analysis of nested CRC with additional net data by means of stochastic automata for safety-critical communication, *Proc. of IEEE International Workshop on Factory Communication Systems*, Dresden, 21-23 May 2008, Article number 4638714, Pages 295-304.

19. H. D. Wacker and J. Börcsök, Binomial and monotonic behavior of the probability of undetected error and the 2-r-bound, *Journal WSEAS TRANSACTIONS on COMMUNICATIONS*, Vol. 7 Issue 3, March 2008, pp. 188-197.

20. T. Maxino and P. Koopman, The effectiveness of checksums for embedded control networks, *IEEE Trans. on Dependable and Secure computing*, vol. 6, No. 1, pp. 59-72, 2009.

21. O. Egwali Annie and V. V. N. Akwukwuma, Performance Evaluation of AN-VE: An Error Detection and Correction Code, *African Journal of Computing & ICT*, vol. 6, No 1, pp. 117-126, 2013.

22. X. Ji, G. Wang, F. Liu, RS-485 Bus Design of a Missile Simulation Training System, *Telkomnika*, Vol. 11, No. 2, 2013, pp. 291-296.

[212] T. Baicheva, Binary and ternary linear codes which are good and proper for error correction, *Proc of the International Workshop on Algebraic and Combinatorial Coding Theory*, Bansko, Bulgaria (2000) 55–60.

1. E. Nikolova, *Подходящи кодове за откриване на грешки*, Дисертация за присъждане на научната степен 'Доктор', 2005.

2. R. Dodunekova, S. Dodunekov and E. Nikolova, A survey on proper codes, *Discrete Applied Mathematics*, Volume 156, Issue 9, 1 May 2008, pp. 1499-1509.

[213] T. Baicheva, S. Dodunekov and P. Kazakov, On the Undetected Error Probability Performance of Cyclic Redundancy-Check Codes of 16-bit Redundancy, *IEE Proc. Communications*, vol. 147, No. 5 (2000) 253 –256.

1. D. Sheinwall, J. Satran, P. Thaler, V. Cavanna, Internet Protocol Small Computer System Interface (iSCSI) Cyclic Redundancy Check (CRC) Checksum Considerations. RFC-3385, Sept. 2002, *The Internet Society*.

2. F. Zhai, I.J. Fair, Efficient cyclic redundancy checks for turbo-coding, *Proc. Wireless and optical communications*, WOC 2002, Banff, Canada, pp. 356-196.

3. R. Dodunekova, On the binomial moments of linear codes and undetected error probability, Preprint 2002:49, Chalmers University of Technology, 14 p.

4. R. Dodunekova, The duals of MMD codes are proper for error detection, *IEEE Trans. Inform. Theory*, vol.49, No.8, 2003, pp. 2034-2038.

5. R. Dodunekova, The extended binomial moments of a linear code and the undetected error probability, *Problems of Information Transmission*, v.39, No.3, 2003, pp.255-265.

6. A. Youssef, A. de Bonneval, Y. Couzet, Dependability of Communications in Critical Real-Time Control Systems, *8-th CaberNet Radicals Workshop*, Ajaccio, Corsica, 5-8 October, 2003.

7. A. Youssef, *Systeme de commande de voldufutur: nouvelle architecture de communication*, Thesis, LAAS-CNRS, Groupe TSF 7, Toulouse, France,2003.

8. F. Zhai and I.J. Fair, Techniques for early stopping and error detection in turbo decoding, *IEEE Transactions on Communications*, 51 (10), 2003, pp.2034-2038.

9. V. A. Khitrovskyy, Schematic and technological aspects of frequency synthesizer design for advanced radars, *Proc of Int. Crimean Conference Microwave & Telecommunication Technology*, Sevastopol, Ukraine, 2003, pp. 11-14.

10. P. Koopman, T. Chakravarty, Cyclic Redundancy Check (CRC) Popynomial Selection for Embedded Networks, *Proc. The International Conference on Dependable Systems and Networks*, DSN-2004.

11. M. M. Carvalho and J. J. Garcia-Luna-Aceves, Modeling single-hop wireless networks under Rician fading channels, *2004 IEEE Wireless Communications and Networking Conference*, WCNC 2004, 1, pp.219-224.

12. E. Nikolova, *Подходящи кодове за откриване на грешки*, Дисертация за присъждане на научната степен 'Доктор', 2005.

13. A. Youssef, *Reseau de communication a haut niveau d'integrite*, PhD Thesis No. 2292, 2005, INPT, France.

14. R. Dodunekova, O. Rabaste and J.L.V. Páez, Error detection with a class of irreducible binary cyclic codes and their dual codes, *IEEE Trans. on Inform. Theory*, vol. 51, No. 4, pp. 1206-1209, 2005.

15. J. Zhao, F. Zarkeshvari and A.H. Banihashemi, On implementation of min-sum algorithm and its modifications for decoding low-density Parity-check (LDPC) codes, *IEEE Transactions on Communications*, vol. 53, Issue 4, April 2005, pp. 549 - 554. ISSN: 0090-6778

16. T.C. Maxino, *The effectiveness of checksums for embedded networks*, MS Thesis, Carnegie Melon University, Pttsburg, USA, May, 2006.

17. S. Huettinger, Low-complexity short-length error detecting codes, *Proc. of the International Workshop on Algebraic and Combinatorial Coding Theory*, Zvenigorod, Russia, pp. 118-121, 2006.

18. M.M. de Carvahlo, *Analytical modeling of medium access control protocols in wireless networks*, PhD Thesis, Univ.of California Santa Cruz, March 2006.

19. C. Nguyen and G. R. Redinbo, Detecting computer-induced errors in remote-sensing JPEG compression algorithms, *IEEE Transactions on Image Processing*, Vol. 15, Issue 7, July 2006, pp. 1728-1739. ISSN: 1057-7149

20. J. Börcsök, J. Hölzel and H. D. Wacker, Probability of Undetected Error with Redundant Data Transmission on a Binary Symmetric Channel without Memory, *Proceedings of the 6th WSEAS International Conference on Applied Computer Science*, Tenerife, Canary Islands, Spain, December 16-18, 2006, pp. 103-106.

21. Файза А. Р. Салам Муджахед, *Предизвикателства към сигурността в информационна система, базирана на УЕВ технологиите*, Дисертация за присъждане на научната степен 'Доктор', ИМИ-БАН, 2007.

22. J. Börcsök, Some Inequalities Concerning Binomial Coefficients and the Weight Distribution of Proper Linear Codes, *7th WSEAS International Conference on APPLIED COMPUTER SCIENCE*, Venice, Italy, November 21-23, 2007, pp. 43-48.

23. T. Klove, *Codes for error detection*, Series on Coding Theory and Cryptology, vol 2., World Scientific, 2007.

24. Ph. Golden, H. Dedieu, K. Jacobsen, *Implementation and Applications of DSL Technology*, CRC Press, 2007.

25. H. D. Wacker and J. Börcsök, Some inequalities concerning binomial coefficients and the weight distribution of proper linear codes, *Proceedings of the 7th WSEAS International Conference on Applied Computer Science - Volume 7*, Venice, Italy pp. 43-48, 2007.

26. H. D. Wacker and J. Börcsök, The probability of undetected error of some communication channels, *Risk, Reliability and Societal Safety* Aven & Vinnem (eds.), Taylor & Francis Group, London, 2007.

27. N. Kaabouch, A. Dhirde, S. Faruque, Improvement of the orthogonal code convolution capabilities using FPGA implementation, *IEEE International Conference on*
Electro/Information Technology, Chicago, IL, 17-20 May 2007, pp. 337 - 341.

28. A. Wang and N. Kaabouch, FPGA based design of a novel enhanced error detection and correction technique, *IEEE International Conference on Electro/Information Technology*, Ames, IA, 18-20 May 2008, pp. 25 - 29.

29. H. D. Wacker and J. Börcsök, Binomial and monotonic behaviour of the probability of undetected eror and the 2^{-r}-bound, *WSEAS Transactions on Communications*, vol. 7, March 2008, pp. 188-197.

30. H. D. Wacker and J. Börcsök, The Dual Distance of a CRC and Bounds on the Probability of Undetected Error, the Weight Distribution, and the Covering Radius, *WSEAS Transactions on Communications*, vol. 7, April 2008, pp. 188-197.

31. F. Schiller and T. Mattes, Analysis of nested CRC with additional net data by means of stochastic automata for safety-critical communication, *IEEE International Workshop on Factory Communication Systems*, 21-23 May, 2008, pp. 295 - 304

32. T. Maxino and P. Koopman, The effectiveness of checksums for embedded control networks, *IEEE Trans. on Dependable and Secure computing*, vol. 6, No. 1, pp. 59-72, 2009.

33. Damien O'Rourke, *Practical packet combining for use with cooperative and non-cooperative ARQ schemes in wireless sensor networks*, PhD thesis, Dublin City University, 2009.

34. Pavan Kumar Pendli, Michael Schwarz, Hans Dieter Wacker and Josef Boercsoek, Bluetooth for Safety Systems, *ISSC 2011*, Trinity College Dublin, June 23-24, 2011.

35. T. Zheng, W. Shaoping and A. El Kamel, Bluetooth communication reliability of mobile vehicles, *Proc. of International Conference on Fluid Power and Mechatronics*, Beijing, China, pp. 873-877, 2011.

36. P. K. Pendli, M. Schwarz, H. D. Wacker and J. Boercsoek, Bluetooth for Safety Systems, *22nd IET Irish Signals and Systems Conference*, Dublin, Ireland, 23-24 June2011.

37. Damien O'Rourke and Conor Brennanb, Practical packet combining for use with cooperative and non-cooperative ARQ schemes in resource-constrained wireless sensor networks, *Ad Hoc Networks*, Volume 10, Issue 3, pp. 339–355, 2012.

38. H. D. Wacker, P. Pendli and J. Börcsök, Data transmission via erasure type channels protected by linear codes, *J. Phys.: Conf. Ser.* **364 012058**, 2012.

39. M. Shinagawa, K. Kondou, M. Noda, *CRC generator polynomial select method, CRC coding method and CRC coding circuit*, US Patent 8341510 B2, December 25, 2012.

40. M. Shinagawa, M. Noda, H. Yamagishi, K. Kondou, *Transmission apparatus and method, reception apparatus and method, and program*, US Patent 8327251 B2, December 4, 2012.

41. You-Gang Cha, Cha-Keon Cheong, Analysis of CRC-p code performance and determination of optimal CRC code for VHF band maritime ad-hock wireless communication, *The Journal of Korea Information and Communications Society*, Vol. 37A, No. 6, pp. 438-449, 2012.

42. O. Egwali Annie and V. V. N. Akwukwuma, Performance Evaluation of AN-VE: An Error Detection and Correction Code, *African Journal of Computing & ICT*, vol. 6, No 1, pp. 117-126, 2013.

43. M. Gholase, L.P.Thakare and A.Y. Deshmuk, Enhancement of Error Detection and Correction Capability Using Orthogonal Code Convolution, *International Journal of Computational Engineering Research*, Vol, 03, Issue, 4, pp. 66-71, 2013.

44. H. D. Wacker, P. Pendli and J. Boercsoek, Data transmission via erasure type channels protected by linear codes, *Journal of Physics: Conference Series*, vol. 364, Issue 1, 2013.

45. H. D. Wacker, P. Pendli and J. Boercsoek, Error control capability of orthogonal code convolution by means of FPGA application, *International Journal of Engineering Sciences Paradigms and Researches*, Vol. 05, Issue 01, pp. 27-30, 2013.

46. Y. Zhang, X. Li, Apparatus for appending cyclic redundancy check in communication system, US Patent US 8464140 B2, June 11, 2013.

47. J. H. Collet, A brief overview of the challenges of the multicore roadmap, *Proc. of International Conference MIXDES*, Lublin, Poland, 2014.

48. J. Noh, H. Song and C. Lee, An Error Pattern Estimation Scheme for Iterative Receiver in MIMO Systems, *IEEE Communications Letters* Vol. 18, Issue 4, pp. 552-555, 2014.

49. V. K. Shendre and R. Nawkhare, Enhancement of Error Control Capability of Orthogonal Code Convolution for Digital Communication, *IJCAT International Journal of Computing and Technology*, Volume 1, Issue 3, April 2014, pp. 5-9.

50. Umberto Mattei, *Extended physical layer modeling for smart metering utility network simulators*, Master Thesis, KTH, Communication Theory, Sweden, 2014.

51. S. Jirapure, Implementation of 16 bit orthogonal code convolution with enhanced error control technique using VHDL, *Int. Journal of Pure and Applied Research in Eng. and Techn.*, vol. 2 (9), pp. 78-85, 2014.

52. A. Kant, Enchance orthogonal code convolution capabilities for efficient digital communication, *Int. Journal For Technological Research In Engineering*, vol. 2, issue 4, pp. 2347-4718, 2014.

[214] T. Baicheva, On the covering radius of ternary negacyclic codes with length up to 26, *IEEE Trans. on Inform. Theory*, vol. 47, No. 1 (2001) 413–416.

1. G. Cohen, I. Honkala, S. Litsyn and A. Lobstein, *Covering Codes*, North-Holland, Elsevier Science B.V., 1997, Updated July 12, 2010 list with bibliography at http://www.infres.enst.fr/ lobstein/bib-a-jour.pdf.

2. E. Velikova, The weight distribution of the cosets leaders of ternary cyclic codes with generator polynomial of small degree, *Annuarie de L'Université de Sofia 'St. Kl. Ohridski'*, vol. 97, pp. 109-114, 2005.

3. E. Velikova and A. Bojilov, On the Weight Distribution of the Coset Leaders of Constacyclic Codes, *Serdica J. Computing*, vol. 2, No. 2, pp. 105-110, 2008.

4. A. A. Davydov, M. Giulietti, S. Marcugini and F. Pambianco, Linear nonbinary covering codes and saturating sets in projective spaces, *Advances in Mathematics of Communications*, vol. 5, issue 2, 2011, pp. 119-147.

[215] T. Baicheva, S. Dodunekov and R. Kötter, On the Performance of the Ternary [13,7,5] Quadratic-Residue Codes, *IEEE Trans. Inform. Theory*, vol. 48, No. 2 (2002) 562–564.

1. E. Nikolova, *Подходящи кодове за откриване на грешки*, Дисертация за присъждане на научната степен 'Доктор', 2005.

2. R. Dodunekova, O. Rabaste and J.L.V. Páez, Error detection with a class of irreducible binary cyclic codes and their dual codes, *IEEE Trans. on Inform. Theory*, vol. 51, No. 4, pp. 1206-1209, 2005.

3. T. Klove, *Codes for error detection*, Series on Coding Theory and Cryptology, vol 2., World Scientific, 2007.

4. H. P. Lee, H. Y. Chen and H.C. Chang, A Method for Decoding the (24, 15, 5) Cyclic Code, *Third International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, Kaohsiung, 26-28 Nov. 2007, pp. 391-394. ISBN: 978-0-7695-2994-1

5. D. Kisku, P. Gupta and J. Sing, Feature Level Fusion of Face and Palmprint Biometrics, *Proc of Joint IAPR International Workshop SSPR&SPR*, Cesme, Izmir, Turkey, August 18-20, 2010.

6. M. Effros, G. D. Forney Jr., F. R. Kschischang, M. Médard, A. Singer, A. Vardy, The Scientific Legacy of Ralf Koetter, *IEEE Trans. Inform. Theory*, Vol. 57, N0. 2, FEBRUARY 2011, pp. 589-592.

7. K Xenoulis, List Permutation Invariant Linear Codes: Theory and Applications, *IEEE Trans. Inform. Theory*, Vol. 60, Issue 9, Sept. 2014, pp. 5263-5282.

[217] T. Baicheva and V. Varek, On the least covering radius of binary linear codes with small lengths, *IEEE Trans. On Inform. Theory*, vol. 49, No. 3 (2003) 738–740.

1. G. Cohen, I. Honkala, S. Litsyn and A. Lobstein, *Covering Codes*, North-Holland, Elsevier Science B.V., 1997., Updated July 12, 2010 list with bibliography at http://www.infres.enst.fr/ lobstein/bib-a-jour.pdf.

2. И. Буюклиев, *Алгоритмични подходи за изследване на линейни кодове*, Дисертация за присъждане на научната степен 'Доктор на математическите науки', 2007.

3. Li Ping, Zhu Shi-xin, Yu Hai-feng, Covering radius of codes over ring $F_2 + uF_2$, *Journal of University of Science and Techology of China*, 38(2), 2008.

[220] T. Baicheva, I. Boyukliev, S. Dodunekov and W. Willems, Teaching linear codes, *Mathematica Balkanica*, New Series vol. 19 (2005) 3–16.

1. A. Faldum, On the trustworthiness of error-correcting codes, *IEEE Trans. Inform. Theory*, vol. 53, No. 12, 2007, pp. 4777 - 4784.

2. R. Jurrius and R Pellikaan, Extended and Generalized Weight Enumerators, *Proceedings of the International Workshop on Coding and Cryptography*, WCC 2009, Ullensvang, May 10-15, Selmer Center, Bergen, pp. 76-91, 2009.

3. R. Jurrius and R Pellikaan, The extended coset leader weight enumerator, *Proc. 30th Symposium on Information Theory on the Benelux*, May 28-29, pp. 217-224, 2009.

4. R. Jurrius and R Pellikaan, The coset leader and list weight enumerator, *Contemporary Mathematics*, vol. 632, pp. 229-250, 2015

[226] T. Baicheva, I. Bouyukliev, S. Dodunekov, and V. Fack, Binary and Ternary Quasi-perfect Codes with Small Dimensions, *IEEE Trans. on Inform. Theory*, vol. 54, issue 9 (2008) 4335–4339.

1. G. Cohen, I. Honkala, S. Litsyn and A. Lobstein, Covering Codes, North-Holland, *Elsevier Science B.V.*, 1997., Updated July 12, 2010 list with bibliography at http://www.infres.enst.fr/ lobstein/bib-a-jour.pdf.

2. Foto N. Afrati, Anish Das Sarma, David Menestrina, Aditya Parameswaran, Jeffrey D. Ullman, Fuzzy Joins Using MapReduce, *Technical Report. Stanford InfoLab*, July 2011.

3. F. N. Afrati, A. D. Sarma, D. Menestrina, A. Parameswaran, J. D. Ullman, Fuzzy Joins Using MapReduce, *IEEE 28th International Conference on Data Engineering*, Washington, DC, 1-5 April 2012, pp. 498-509.

4. L. H. Aslanyan and H. E. Danoyan, Complexity of Elias algorithm based on codes with covering radius 3, *Proc. of the Yerevan State University*, No. 1, pp. 44-50, 2013.

5. L. H. Aslanyan and H. E. Danoyan, Complexity of Elias algorithm for hash functions based on Hamming and extended Hamming codes, *Reports of National Academy of Sciences of Armenia*, vol. 113, No. 2, 2013.

[227] T. Baicheva, Determination of the best CRC codes with up to 10-bit redundancy, *IEEE Trans on Communic.*, vol. 56, issue 8 (2008) 1214–1220.

1. T. Zhu, Z. Zhong, J. Zhang, A quick coding method based on dynamic table look-up for arbitrary bit length polynomial division in embedded system, *7th International Conference on Networked Computing and Advanced Information Management*, Gyeongju, 21-23 June, 2011, pp. 15-19.

2. E. Üstünel, I. Hökelek, O. Ileri, H. Arslan, Joint optimum message length and generator polynomial selection in Cyclic Redundancy Check (CRC) coding, *2011 IEEE 19th Signal Processing and Communications Applications Conference*, Antalya, 20-22 April 2011, pp. 222-225.

3. E. Üstünel, I. Hökelek and O. Ileri, A cross-layer goodput enhancement considering CRC coding and ARQ dinamics, *IEEE Symposium on Computer Communications*, Cappadocia, Turkey, pp. 23-28, 2012.

4. H. Patel and D. Jain, Design & check cyclic redundancy code using VERILOG HDL, *Int. Journal for Scientific Research & Development*, vol.1, issue 5, pp. 1096-1098, 2013.

5. H. Patel, D. Patel, M. Chaudhary and M. Zala, An Automated CRC engine, *International Journal for Innovative Research in Science & Technology*, Vol. 1, Issue 1, pp. 73-77, June 2014.

6. Li Chia Choo, Zander Lei, CRC codes for short control frames in IEEE 802.11ah, *The 80-th IEEE Vehicular Technology Conference*, Vancouver, Canada, 14-17 September, 2014, pp. 1-5.

7. C. Bai, M. S. Leeson, M. D. Higgins, Performance of SW-ARQ in bacterial quorum communications, *Nano Communication Networks*, vol. 6, Issue 1, pp. 3–14, March 2015.

8. M. El-Khamy, J. Lee, I. Kang, Detection Analysis of CRC-Assisted Decoding, *IEEE Commun. Letters*, vol. 19, issue 3, pp. 483-486, 2015.

9. D. A. Nugroho, S. Rizal, Dong-Seong Kim, Reconstruct unrecoverable data in real-time networks using Bézier curve, *IET Communications*, Available online: 05 January 2015, pp. 596-602

[229] T. Baicheva and I. Bouyukliev, On the least covering radius of the binary linear codes of dimension 6, *Advances in Mathematics of Communications*, vol. 4, No 3 (2010) 399–403.

1. G. Cohen, I. Honkala, S. Litsyn and A. Lobstein, *Covering Codes*, North-Holland, Elsevier Science B.V., 1997, Updated November 29, 2011 list with bibliography at http://www.infres.enst.fr/ lobstein/bib-a-jour.pdf.

[230] T. Baicheva, All binary linear codes of lengths up to 18 or redundancy up to 10 are normal, *Advances in Mathematics of Communications*, vol. 5, No 4 (2011) 681–686.

1. G. Cohen, I. Honkala, S. Litsyn and A. Lobstein, *Covering Codes*, North-Holland, Elsevier Science B.V., 1997, Updated November 29, 2011 list with bibliography at http://www.infres.enst.fr/ lobstein/bib-a-jour.pdf.

[231] T. Baicheva and S. Topalova, Optimal (v,4,2,1) optical orthogonal codes with small parameters, *Journal of Combinatorial Designs*, vol. 20 (2) (2012) 142–160.

1. X. Wang and Y. Chang, Further results on optimal (v, 4, 2, 1)-OOCs, *Discrete Mathematics*, volume 312, issue 2, pp. 331 - 340, 2012.
2. M. Buratti, A. Pasotti and D. Wu, On optimal (v, 5, 2, 1) optical orthogonal codes, *Design Codes and Cryptography*, vol. 68, Issue 1-3, pp. 349-371, 2013.
3. H. Zhao, D. Wu and R. Qin, Further results on balanced $(n, 3, 4, \lambda_a, 1)$-OOCs, *Discrete Mathematics*, Vol. 337, 28 December 2014, pp. 87–96.

[232] T. Baicheva and S. Topalova, Optimal optical orthogonal codes of weight 5 and small lengths, *International Conference on Applications of Computer Algebra*, Sofia, Bulgaria (2012).

1. S. M. Ibraheem, M. M. A. Elrazzak, S. M. S. Eldin, W. Saad and A.E. Aboelazm, A class of structured quasi-cyclic LDPC codes based on planar difference families, *International Conference on Advanced Technologies for Communications*, Ho Chi Minh City, 16-18 Oct. 2013, pp. 614-619.

[235] T. Baicheva and S. Topalova, Optimal (v, 5, 2, 1) optical orthogonal codes of small v, *Applical Algebra in Engeneering Communication and Computing*, vol. 24, numbers 3-4 (2013) 165–177.

1. H. Zhao, D. Wu and R. Qin, Further results on balanced $(n, 3, 4, \lambda_a, 1)$-OOCs, *Discrete Mathematics*, Vol. 337, 28 December 2014, pp. 87–96.