MATHEMATICS OF COMPUTATION
VOLUME 57, NUMBER 195
JULY 1991, PAGES 415–434

COVERING RADIUS COMPUTATIONS FOR BINARY CYCLIC CODES

RANDALL DOUGHERTY AND HEERALAL JANWA

ABSTRACT. We compute the covering radius of each binary cyclic code of length ≤ 64 (for both even and odd lengths) and redundancy ≤ 28. We also compute the covering radii of their punctured codes and shortened codes. Thus we give exact covering radii of over six thousand codes. For each of these codes (except for certain composite codes), we also determine the number of cosets of each weight less than or equal to the covering radius. These results are used to compute the minimum distances of the above cyclic codes. We use the covering radii of shortened codes and other criteria for normality to show that all but eight of the cyclic codes for which we determine the covering radius are normal. For all but seven of these normal codes, we determine the norm using some old results and some new results proved here. We observe that many cyclic codes are among the best covering codes discovered so far, and some of them lead to improvements on the previously published bounds on $t[n, k]$, the smallest covering radius of any binary linear $[n, k]$ code.

Among some other applications of our results, we use our table of covering radii and a code augmentation argument to give four improvements on the values of $d_{\max}(n, k)$, where $d_{\max}(n, k)$ is the largest minimum distance of any binary $[n, k]$ code. These results show that the covering radius is intimately connected with the other three parameters of a linear code, n, k, and d. We also give a complete classification (up to isomorphism) of cyclic self-dual codes of lengths 42, 56, and 60.

The computations were carried out mainly on concurrent machines (hypercubes and Connection Machines); we give a description of our algorithm.

1. INTRODUCTION

Coding theory consists of the study of transmission of information reliably over a noisy channel (deep space, telephone, telegraph, magnetic tape, disc drive, etc.). Algebra, combinatorics, graph theory, finite geometries, combinatorial geometries, number theory, representation theory, and algebraic geometry have contributed to this discipline and in turn have profited from it (see, for example, MacWilliams and Sloane [20], Berlekamp [1], Goppa [13], or Hirschfeld [15]).

Let F_2 denote the finite field of two elements. An $[n, k]$ binary linear block code C is a k-dimensional subspace of F_2^n. For $u, v \in F_2^n$, let $d(u, v)$ denote the number of coordinate places where u and v differ; $d(u, v)$ is called the Hamming distance between u and v. The minimum distance of C

Received December 7, 1989.
1980 *Mathematics Subject Classification* (1985 *Revision*). Primary 94B15, 94B05.

is defined as $d = \min\{d(u, v) \mid u, v \in C \text{ and } u \neq v\}$. The *Hamming weight* $\mathrm{wt}(u)$ of a vector u is defined to be $d(u, 0)$. Since C is a linear subspace of F_2^n, the minimum distance of C is the minimum of the set of weights of nonzero vectors in C. The *covering radius* of a block code C of length n is the smallest integer $R = R(C)$ such that all vectors in the containing space are within Hamming distance R of some codeword of C. Cohen et al. [8] gives a survey of present knowledge about R. For basic results from coding theory, see Berlekamp [1], van Lint [28], MacWilliams and Sloane [20], or Pless [25]. From now on, an $[n, k, d]R$ code is a linear code of length n, dimension k, minimum distance d, and a covering radius R.

For the last forty years, the main problems of coding theory have been optimization problems involving only the three parameters n, k, and d. Some particular problems have been to maximize k and d (for fixed n) and to find linear $[n, k, d]$ codes (i.e., $[n, k]$ codes with minimum distance d) with simple decoding algorithms. However, recently there has been an intense interest in codes with optimum covering radius [8]. It has been established that the covering radius is a basic geometric parameter of a code and has many important applications (see Cohen et al. [8] for further references). This paper concerns Open Problem 6 of [8], which asks for the determination of the covering radii of some classes of codes (e.g., Goppa codes, Justesen codes, cyclic codes, quadratic residue codes, or Reed-Muller codes). We note that quadratic residue codes are cyclic codes and Reed-Muller codes are extended cyclic codes. Also, some extended Goppa codes are cyclic.

We have computed the covering radius of binary cyclic codes of length ≤ 64 (for both even and odd lengths) and redundancy ≤ 28. Downey and Sloane [12] give the covering radii of some cyclic codes of odd length ≤ 31 (see also Mattson [21] and Cohen et al. [8]), and we verify these results. The covering radii of a few other codes, such as the [47, 24, 11] quadratic residue code and the 1-, 2-, and 3-error-correcting BCH (Bose-Chaudhuri-Hocquenghem) codes of lengths $2^m - 1$, $m \geq 2$, are also known [8]. The problem of computing covering radii is known to be both NP-hard and co-NP-hard (in fact, Π_2^p-complete [23]), and hence strictly harder than any NP-complete problem unless NP = co-NP.

We show that all but eight of the cyclic codes for which we compute the covering radius are normal, and we determine the norm for all but seven of the normal ones. Normal codes are important because they can be efficiently combined to give good codes by using the "amalgamated direct sum" construction given in Graham and Sloane [14]. We prove normality and determine the norms of most codes by determining the covering radii of shortened codes and punctured codes and then applying several results, some of which are new. Computing the norm seems to be at least as hard as computing the covering radius (see §3).

We also determine the number of cosets of each weight less than or equal to R; this is extremely difficult to calculate for a general code and is known only for very few codes [20, pp. 18–19]. The coset weights give the exact value for

the decoding error probability if C is used for error correction and maximum likelihood decoding is used [20, pp. 15–19]. As an application of these numbers, we determine the minimum distance for all cyclic codes for which we compute the covering radius.

One goal of this paper is to give tighter bounds on the values of the following two functions connected with the covering radius:

(1) The smallest covering radius of any $[n, k]$ binary linear code, denoted by $t[n, k]$. Graham and Sloane [14] give a table of bounds on $t[n, k]$ for $n \leq 64$. We give twelve improvements of upper bounds in this table. Some of our codes achieve the exact value of $t[n, k]$.

(2) The least length of a code of codimension m and covering radius R, denoted $l(m, R)$ [4] (the codimension of an $[n, k]$ code is $n - k$). Brualdi and Pless [3] include a table of bounds on $l(m, R)$ for $m \leq 24$; we give improvements of three upper bounds in this table.

For all of the cyclic codes in our table, we also determine the covering radius of the corresponding punctured and shortened codes. In all, we determine the covering radius for over six thousand codes.

As an application of all the parameters we compute, we give a complete classification (up to isomorphism) of cyclic self-dual codes of lengths 42, 56, and 60.

Among other applications of our results, we use our table of covering radii and a code augmentation argument to give four improvements on the known bounds of $d_{\max}(n, k)$, where $d_{\max}(n, k)$ is the largest minimum distance of any binary $[n, k]$ code. These results show that the covering radius is closely connected with the other three parameters of a linear code, n, k, and d.

The organization of the paper is as follows. In §2 we collect the necessary background from coding theory. Section 3 lists the required results to determine the normality and the norm for the cyclic codes mentioned above. Section 4 contains the details of the algorithm used. Section 5 consists of a description of how we generate our list of cyclic codes, including a discussion of trivial and composite codes. In §6, we discuss the results we obtain and their applications. The Appendix, on microfiche, contains two long tables containing the following data: Table 1 lists the cyclic codes of length at most 64 and codimension at most 28 (omitting some trivial and composite codes), with the roots and computed parameters for each code; Table 2 gives the coset weight distribution of each listed cyclic, punctured cyclic, or shortened cyclic code.

2. Background

We use the notation of Cohen et al. [8].

Let G be a $k \times n$ binary matrix whose rows form a basis for an $[n, k]$ binary linear code C. Then G is called a *generator matrix* for C. If we define the inner product of two vectors in F_2^n in the usual way, then the orthogonal complement of C is called its *dual code* and denoted $C^{\perp}$. The dimension

of $C^{\perp}$ equals $n-k$, and a generator matrix for $C^{\perp}$ is called a *check matrix* for C.

We will need the following well-known facts about R, for which references are given in Cohen et al. [8, pp. 328–331].

Fact 1. *If H is any check matrix of C, then $R(C)$ is the least integer such that every syndrome is a sum of $R(C)$ or fewer columns of H, where a syndrome is any column of length $n-k$ over $GF(2)$.*

Fact 2 (Redundancy bound). $R(C) \le n-k$.

Fact 3 (Sphere-covering-bound). $2^{n-k} \le \sum_{i=0}^{R(C)} \binom{n}{i}$.

A code C is called *even* if all vectors in C have even weight.

Fact 4. *If C is even, then*

$$\sum_{2i \le R(C)} \binom{n}{2i} \ge 2^{n-k-1} \quad \text{and} \quad \sum_{2i+1 \le R(C)} \binom{n}{2i+1} \ge 2^{n-k-1}.$$

Fact 5 (Delsarte bound). *If s' is the total number of nonzero weights in $C^{\perp}$, then $R(C) \le s'$.*

Fact 6 (Supercode lemma). *If C_1 is a proper subcode of C_2, then*

$$R(C_1) \ge \max\{\min\{\mathrm{wt}(x) \mid x \in C_2 \setminus C_1\}, R(C_2)\}.$$

Fact 7. *Appending an overall parity check or an extra zero bit increases the covering radius by one. Puncturing a code on p coordinates (i.e., removing these coordinates) reduces the covering radius by at most p.*

Fact 8. *The $[n, 1, n]$ code has covering radius $\lfloor n/2 \rfloor$.*

Let $d(C)$ denote the minimum distance of C.

Fact 9 (Sphere-packing bound). *If $e(C) = \lfloor (d(C)-1)/2 \rfloor$, then*

$$\sum_{i=0}^{e(C)} \binom{n}{i} \le 2^{n-k}.$$

If we take all code vectors in C which have a zero in the first coordinate (assuming that this does not include all of C) and drop this coordinate from these codewords, then we get the $[n-1, k-1, d_s]$ *shortened code* C_s (where $d_s \ge d$). A check matrix for C_s can be obtained by dropping the first column from a check matrix for C. In Table 1, $R_s = R(C_s)$. From Fact 1, it is clear that $R \le R_s$. For an important application of R_s, see Proposition 1 in §3.

Let C_p denote the code obtained from C by dropping the first coordinate. (C_p is called a *punctured code.*) Then Fact 7 implies that $R_p := R(C_p) \le R(C)$. For some applications of R_p, see §6.1 below.

The weight $\mathrm{wt}(D)$ of a coset D of C is defined to be the least weight of a vector in D. A vector of weight $\mathrm{wt}(D)$ in D is called a *coset leader.* Let α_i denote the number of cosets of C of weight i.

Let $C_{\#}^{n}$, $C_{\#p}^{n}$, and $C_{\#s}^{n}$ denote, respectively, cyclic code number # of length n in Table 1, its punctured code, and its shortened code.

An $[n, k, d]R$ code is a linear code of length n, dimension k, minimum distance d, and covering radius R.

3. Norms and normal codes

Fix $i \in \{0, 1\}$, and let $C_0^{(i)}$ (respectively $C_1^{(i)}$) denote the subcode of C consisting of codewords with ith coordinate equal to 0 (respectively 1). We assume that C is not identically zero at i. Let

$$N^{(i)} := \max\{d(x, C_0^{(i)}) + d(x, C_1^{(i)}) \mid x \in F_2^n\}. \tag{1}$$

Then $N^{(i)}$ is called the *norm of C with respect to the coordinate position i*, and $N(C) := N := \min_{1 \le i \le n} N^{(i)}$ is called the *norm* of C; coordinate positions i for which $N = N^{(i)}$ are called *acceptable.* (Note that all coordinate positions of a cyclic code are acceptable.) The code is called *normal* if $N \le 2R + 1$.

Cohen, Lobstein, and Sloane [9] give the following sufficient condition for normality.

Proposition 1. *Let R_s denote the covering radius of a shortened subcode of a binary linear code C. If $R_s \le R + 1$, then C is normal.*

Other conditions ensuring normality have been established in a series of papers, starting with Graham and Sloane [14], where all $[n, k]$ codes with $n \le 8$ or $k \le 2$ were proved normal. After extensions in Cohen, Lobstein, and Sloane [9] and Sloane [26], the latest improvements appeared in Kilby and Sloane [19] and Janwa and Mattson [18].

Proposition 2. *An $[n, k, d]$ binary linear code C is normal provided that $n \le 14$, or $k \le 5$, or $d \le 5$, or $R \le 2$ [19], or $n = 15$, or $n - k \le 8$ [18].*

We use Propositions 1 and 2 to prove normality for all but the following codes in Table 1: C_{21}^{21}, C_{86}^{30}, C_{17}^{31}, C_{18}^{31}, C_{23}^{31}, C_{1}^{41}, C_{2}^{46}, C_{1}^{47}, and C_{554}^{63}.

The code C_{21}^{21} has been shown to be normal [12]. We thus have eight cyclic codes of lengths ≤ 64 and redundancy ≤ 28 for which we have not shown normality. Downey and Sloane [12] show that all cyclic codes of odd lengths ≤ 25 and some codes of lengths 27 and 31 are normal; we verify their results for these codes.

3.1. **Determination of norms.** In this section we prove that the parameters R, R_p, and R_s and the evenness of cyclic codes are enough to determine the norm for all of the cyclic codes of length ≤ 64 and redundancy ≤ 28 except for the eight codes listed in the preceding section and seven additional codes. The norm for cyclic codes of odd lengths ≤ 25 and some codes of lengths 27 and 31 has already been computed by Downey and Sloane [12], and we verify their results for these codes.

Determining the norm of a binary linear code (theoretically or computationally) seems to be at least as difficult as determining the covering radius. This is clearly true for normal codes, for which we have $R(C) = \lfloor N(C)/2 \rfloor$. One can in fact show that the problem of computing covering radii is polynomial-time reducible to the problem of computing norms: if C is a code with check matrix H, and C' is the code whose check matrix consists of two copies of H side by side, then Theorem 2(a) in §5 implies that $R(C) = \lfloor N(C')/2 \rfloor$.

Proposition 3 (Graham and Sloane [14]). *If C is an even code, then the norm of C is even. Furthermore, if C is normal, then $N = 2R(C)$.*

To apply this result to cyclic codes, we note that any cyclic code with root 0 among its roots is an even code.

Theorem 2.3 from Janwa and Mattson [18] immediately yields the following tool for the determination of norms.

Proposition 4. *Let C be a normal cyclic code, and let C^* denote its punctured code at the coordinate $*$. If $R(C^*) = R(C)$, then $N(C) = 2R(C) + 1$.*

We now prove a partial converse to Proposition 4.

Theorem 1. *If $R(C^*) + 1 = R(C) = R(C_s)$, then $N(C) = 2R(C)$ (where C_s is the shortened subcode of C at $*$).*

Proof. First note that since $R(C) = R(C_s)$, C is normal at $*$. We may assume that $*$ is the first coordinate. Clearly, $C^* = (C_0)^* \cup (C_1)^*$. Let $x = (\varepsilon, x^*)$ be any vector. Then

$$d(x^*, C^*) = \min\{d(x^*, (C_0)^*), d(x^*, (C_1)^*)\} \le R(C^*) = R(C) - 1.$$

On the other hand,

$$\max\{d(x^*, (C_0)^*), d(x^*, (C_1)^*)\} \le R(C_s) = R(C).$$

Now,

$$\begin{aligned} d(x, C_0) + d(x, C_1) &= 1 + d(x^*, (C_0)^*) + d(x^*, (C_1)^*) \\ &= 1 + \min\{d(x^*, (C_0)^*), d(x^*, (C_1)^*)\} \\ &\quad + \max\{d(x^*, (C_0)^*), d(x^*, (C_1)^*)\} \\ &\le 1 + (R(C) - 1) + R(C). \end{aligned}$$

Therefore, $N(C) \le 2R(C)$. On the other hand, by (1), $N(C) \ge 2R(C)$. □

Theorem 1 implies that the norm of C_3^{15} (which is not an even code) is 6, as is known from Downey and Sloane [12].

Propositions 3 and 4, Theorem 1, and the results in §5 suffice to determine the norm for all codes of lengths ≤ 64 and redundancy ≤ 28 except for the eight listed in the preceding section and the following seven: C_{66}^{30}, C_7^{33}, C_{79}^{45}, C_{82}^{45}, C_{81}^{63}, C_{477}^{63}, and C_{480}^{63}.

4. Details of the algorithm

The covering radius computations are done using a very efficient algorithm which is, so far as we can determine, different from previously used algorithms.

The starting point is Fact 1, which tells us that, in order to determine whether a code has covering radius at most j, we must find out whether every syndrome is a sum of j or fewer columns of the check matrix H. Therefore, we compute checklists L_0, L_1, etc., where L_j is an array of length 2^m $(m = n - k)$ in which entry number i is 1 if syndrome number i is a sum of j or fewer columns of H, and 0 otherwise; when we reach a j for which L_j is all 1's, we are done. At each step j we compute the number N_j of 1's in L_j; then the numbers $N_j - N_{j-1}$ give the coset weight distribution of the code.

Computing L_j directly would involve looking at all $\sum_{l=0}^{j} \binom{n}{l}$ sets of j or fewer columns of H and checking off their sums on the checklist; of course, if we already have L_{j-1}, we can just look at the $\binom{n}{j}$ new subsets. This becomes infeasible quite rapidly as j grows. However, there is a much better way to get L_j from L_{j-1}. This is to note that a sum of j or fewer columns of H is either a sum of $j-1$ or fewer columns of H or the sum of one column of H and a syndrome which is a sum of $j-1$ or fewer columns of H. Therefore, if we assume that the lth column of H is syndrome number h_l, we can write the algorithm for computing L_j from L_{j-1} as follows:

$L_j \leftarrow L_{j-1}$
for l from 1 to n do
  for i from 0 to $2^m - 1$ do
    $L_j(i \oplus h_l) \leftarrow L_j(i \oplus h_l) \vee L_{j-1}(i)$

Here $\vee$ is just the "or" operation, and $\oplus$ is the operation on syndrome numbers corresponding to addition of syndromes. If we use the obvious representation in which syndrome number i is the m-bit binary expansion of i, then $\oplus$ is just an exclusive-or operation on binary representations, which is a built-in operation on most computers.

Note that, as i runs through all numbers from 0 to $2^m - 1$, so does $i \oplus h_l$; hence, we can interchange i and $i \oplus h_l$ in the last line above. Also, instead of getting $L_{j-1}(i \oplus h_l)$ directly from L_{j-1}, we can get it from $L_{j-1}(i \oplus h_{l-1})$: if $d_l = h_l \oplus h_{l-1}$, then $L_{j-1}(i \oplus h_l) = L_{j-1}((i \oplus h_{l-1}) \oplus d_l)$. (The reason for doing this will be explained later.) Therefore, we can write the algorithm as

$L_j \leftarrow L_{j-1}$
$L \leftarrow L_{j-1}$
for l from 1 to n do begin
  $L' \leftarrow L$ translated by d_l
  $L \leftarrow L'$
  $L_j \leftarrow L_j \vee L$
end

where $d_1 = h_1$ and "$L' \leftarrow L$ translated by d_l" is:

$$\text{for } i \text{ from } 0 \text{ to } 2^m - 1 \text{ do}$$
$$L'(i) \leftarrow L(i \oplus d_l)$$

Since each entry of the checklist is a single bit, we can store a number of such entries in a single computer word; it will be convenient to restrict this number to be a power of 2, say 2^b. Then we can consider an m-bit syndrome number i to consist of two parts; the first $m - b$ bits of i identify a word within the checklist, and the remaining b bits choose a particular bit within this word. Then operations such as "$L_j \leftarrow L_j \vee L$" can easily be done a word at a time. Somewhat surprisingly, so can the operation "$L' \leftarrow L$ translated by d_l," because the operation $\oplus$ acts on the bits of a syndrome number separately. If we now index checklists by the top $m - b$ bits of a syndrome number, and if we write d_l as $2^b d_l' + d_l^{(b)}$, then "$L' \leftarrow L$ translated by d_l" becomes

$$\text{for } i \text{ from } 0 \text{ to } 2^{m-b} - 1 \text{ do}$$
$$L'(i) \leftarrow S(L(i \oplus d_l'), d_l^{(b)})$$

and the rest of the algorithm in the preceding paragraph is unchanged. (Here, $S(K, x)$ does bit-swapping in K as specified by x; bit number i of $S(K, x)$ is equal to bit number $i \oplus x$ of K.)

This completes the outline of the sequential algorithm. (Of course, there are a number of tricks one can use to speed up the bit-swapping, the counting of 1's in L_j, and so on.) It is not difficult to convert this to a concurrent algorithm; simply split the checklists up among the processors. Again, it will be convenient to assume that the number of processors is a power of 2, say 2^t. We now split each m-bit syndrome number into three parts: the top t bits give a processor number, the next $m - t - b$ bits give a word number for the part of the checklist within this processor, and the bottom b bits give a bit number within the word. The algorithm can be written out exactly as above; the only communications needed to get L_j from L_{j-1} are in the translation step when the top bits of d_l are not all 0. The fact that $\oplus$ acts on top bits, middle bits, and bottom bits separately means that each processor needs to communicate with only one other processor during this translation. On a hypercube architecture, each of the t top bits refers to communication along one of the t dimensions of the cube; if t' of the top t bits of d_l are 1, then the translation by d_l requires swaps along t' dimensions.

Certain manipulations of the matrix H do not change the covering radius of a code; these include row interchanges, column interchanges, and elementary row operations (but not elementary column operations). Therefore, one can preprocess H in order to reduce the number of nonzero top bits or bottom bits in the syndromes d_l. For example, one can arrange that the first m columns of H form an upper triangular matrix with all 1's above the diagonal (so d_l

only has one nonzero bit for $l \leq m$) and then sort the remaining columns by their top and/or bottom bits; the best thing to do depends on the machine one is using. Note that column interchanges would not be helpful if we were using the numbers h_l directly; this is why we use the differences d_l instead.

We have implemented this algorithm on several machines; in each case it was efficient enough that the main constraint on our computations was not the available CPU time but the size of main memory. The algorithm requires enough space for three checklists of 2^m bits apiece, plus a small amount of additional memory. For each machine, there were maximum numbers T, M, and B of top bits, middle bits, and bottom bits, respectively; this allowed covering radius computations for codes of codimension up to $T+M+B$. We have the program running on an IBM PC-compatible computer ($T=2$, $M=14$, $B=4$; T is nonzero because the segmented memory prevents us from using $M=16$), various Caltech hypercubes (a 32-node Mark III has $T=5$, $M=18$, and $B=5$), and a 16K-node Connection Machine 2 ($T=M=14$, $B=0$). The PC-compatible took about 12 minutes to handle a [41, 21]6 code; an average [63, 35]9 code took about 80 minutes on the Mark III and 7 minutes on the CM-2. Note that the overall running time for the algorithm on a particular machine is basically $O(Rn2^m)$ (so the [63, 35]9 code requires about 600 times as much work as the [41, 21]6 code); differences based on the particular bits in the check matrix, rather than on its size, were a relatively minor factor in the computation times.

We performed a number of consistency checks to make sure that the programs were giving correct results. Of course, we ran some smaller codes for which we already knew the coset weight distributions, and we compared our results with those in previously published tables (Peterson and Weldon [24], Downey and Sloane [12]). We computed the parameters $R(C)$ and $e(C)$ (see Fact 9) and verified the following: $e(C_p) \leq e(C) \leq e(C_p)+1$, $e(C)=e(C_s)$, $R(C_p) \leq R(C) \leq R(C_p)+1$, $R(C) \leq R(C_s) < n$, and $R(C) \leq n/2$. Finally, we verified the properties listed in the next section for the relevant codes in the list.

We also implemented a backtrack search algorithm using the original definition of covering radius, in the hope that it would give useful results for codes of small dimension; it has given improved lower bounds for the covering radius of some codes, but it takes too long to run to completion (giving an exact value for R) in almost all cases.

5. Code list and composite codes

Because of the difficulties we have had in comparing our data to that in previously published tables (e.g., the table in Peterson and Weldon [24]), we have concluded that it will be useful to other researchers for us to include an explicit description of how we generated our list of cyclic codes.

A binary cyclic code of length n and dimension k is determined by a generating polynomial g, a polynomial over $GF(2)$ of degree $n-k$ which divides

$x^n - 1$. (If s is the sequence of $n-k+1$ 0's and 1's giving the coefficients of g, then the code is generated by the vectors $0^i s 0^{k-1-i}$ for $0 \le i < k$.) If we write $n = n_o n_e$, where n_o is odd and n_e is a power of 2, then $x^n - 1 = (x^{n_o} - 1)^{n_e}$, where $x^{n_o} - 1$ has no repeated roots; in fact, if ζ is a primitive n_oth root of unity over $GF(2)$, then the roots of $x^{n_o} - 1$ are ζ^i for $0 \le i < n_o$. Two roots ζ^i and ζ^j are roots of the same irreducible factor of $x^{n_o} - 1$ if and only if $j \equiv 2^m i \pmod{n_o}$ for some m (and vice versa). This defines an equivalence relation on the set $\{0, 1, \ldots, n_o - 1\}$; the equivalence classes for this relation are known as *cyclotomic cosets*. In order to specify the polynomial g, we must give a multiplicity (from 0 to n_e) for the roots corresponding to each cyclotomic coset. We can refer to a cyclotomic coset by giving its least member; hence, g can be represented by a list (with multiplicities and, for convenience, sorted in increasing order) of cyclotomic coset numbers.

The generating polynomial h for the dual code (also known as the *check polynomial* of the original code) has degree k and is related to g by the formula $x^k h(x^{-1}) g(x) = x^n - 1$.

Suppose g is given by a list l of cyclotomic coset numbers, and we obtain a new list l' as follows: fix some odd number m relatively prime to n_o, and get l' from l by replacing each number i with the least number equivalent to mi (and then sorting). If g' is the corresponding polynomial, then $g'(x) \equiv g(x^m) \pmod{x^n - 1}$, so the code for g' can be obtained from the code for g by just permuting the coordinates (moving coordinate number i to the position of coordinate number $mi \bmod n$). Such codes are called *equivalent*; equivalent codes will have the same parameters, so we only need to work with one of them (say, the one whose root list comes first in lexicographic order). This also means that it did not matter which primitive n_oth root of unity we chose above.

To generate our list of codes, we started by generating all possible root lists, keeping only the first one in lexicographic order from each class of equivalent codes. For a given n, these were sorted primarily in decreasing order of k and secondarily in lexicographic order. (We did not sort on d because d could not be computed in advance.) We then eliminated certain trivial and composite codes. Consider the following list of possible properties of a code C given as above:

(1) $k \le 1$.
(2o) n and the listed root numbers have a common odd divisor $m > 1$.
(2e) No root has multiplicity greater than $n_e/2$.
(3o) n and the cyclotomic coset numbers which do not occur in the list with multiplicity n_e have a common odd divisor $m > 1$.
(3e) All cyclotomic coset numbers occur in the list with multiplicity at least $n_e/2$.
(4o) There is an odd number $m > 1$ dividing n_o such that $g(x)$ is of the form $g'(x^m)$ for some polynomial g'.
(4e) All roots have even multiplicity.

In case (1), the code is trivial (either the all-0 vector and the all-1 vector, or just the all-0 vector). In each of the other six cases, the code is composite, obtained from a code of length $n' = n/m$, where we define m to be 2 in the "e" cases.

In cases (2o) and (2e), $g(x)$ divides $x^{n'} - 1$, so we can consider the cyclic code C' of length n' with generator polynomial g. Then a member of $GF(2)^n$ is in C if and only if, when it is broken up into m blocks of length n', the sum of these blocks is in C'. In other words, C is the inverse image of C' under the linear map from $GF(2)^n$ to $GF(2)^{n'}$ which divides a vector into m parts and adds them.

In cases (3o) and (3e), the check polynomial $h(x)$ divides $x^{n'} - 1$, so we can define C' to be the cyclic code of length n' with check polynomial $h(x)$. Then C is a repetition code based on C'; a member of $GF(2)^n$ is in C if and only if its m blocks of length n' are identical and in C'. Theorem 2 shows that the covering radius of a repetition code is close to half its length, so such codes are not of interest for covering radius studies. (Codes of type (3o) were left in our code list anyway, for reasons explained below.)

In cases (4o) and (4e), we have $g(x) = g'(x^m)$, and we can define C' to be the cyclic code of length n' with generating polynomial g'. Here we break a vector of length n into m pieces each of length n' by putting coordinates $0, m, 2m, \ldots$ in one piece, coordinates $1, m+1, 2m+1, \ldots$ in another piece, and so on; the vector is in C if and only if all these pieces are in C'. Hence, except for a rearrangement of coordinates, C is just the direct sum of m copies of C'.

Most of the parameters of composite codes can be obtained from those of the corresponding smaller codes as follows:

Theorem 2. *Suppose C is a binary linear code of length n, C' is a binary linear code of length n', and $n = mn'$, where $m > 1$.*

(a) *If C is an m-ary inverse image of C' (i.e., a vector is in C if and only if, when it is broken into m parts of length n', the sum of those parts is in C'), then the covering radius and coset weight distribution of C are the same as those of C'; furthermore, $d(C) = \min(d(C'), 2)$, $R(C_p) = R(C'_p)$, $R(C_s) = R(C')$, and $N(C)$ is $2R(C)+1$ if $R(C') = R(C'_p)$, $2R(C)$ otherwise (so C is always normal).*

(b) *If C is an m-fold repetition code based on C', then $d(C) = md(C')$. If C' has no identically-0 coordinates, then $\lfloor m/2 \rfloor n' + R(C') \le R(C) \le \lfloor n/2 \rfloor$ if m is odd and $R(C) = n/2$ if m is even; furthermore, when m is even, we have $N(C) = 2R(C) = n$ (so C is normal), $R(C_p) = n/2 - 1$, and $R(C_s) = (n' + z + 1)m/2 - 1$, where z is the number of coordinates at which C'_s is identically 0. (If C' is cyclic, then $z = 0$ unless C' is itself a repetition code, in which case $z = m' - 1$, where m' is the maximal repetition multiplicity of C'.)*

(c) *If* C *is the direct sum of* m *copies of* C', *then* $d(C) = d(C')$, $R(C) = mR(C')$, *and if* w *and* w' *are the polynomials with coefficients given by the coset weight distributions of* C *and* C', *then* $w = w'^m$. *Also,* $R(C_p) = R(C'_p) + (m-1)R(C')$, $R(C_s) = R(C'_s) + (m-1)R(C')$, *and* $N(C) = N(C') + 2(m-1)R(C')$ (*so* C *is normal if and only if* C' *is normal*).

Note. In each case above, any coordinate of C corresponds to a particular coordinate of C'; when we make a statement such as $R(C_p) = R(C'_p)$, we mean that C' is to be punctured in the coordinate corresponding to the coordinate in which C is punctured. Shortened codes and norms are treated similarly.

Proof of Theorem 2. (a) Let $\psi: GF(2)^n \to GF(2)^{n'}$ be the linear map defined by $\psi(u)_j = \sum_{i \equiv j \,(\mathrm{mod}\, n')} u_i$. Then C is the inverse image of C' under ψ, and the cosets of C are just the inverse images of the cosets of C'. Clearly, $\mathrm{wt}(\psi(u)) \leq \mathrm{wt}(u)$ (every nonzero $\psi(u)_j$ requires at least one corresponding nonzero u_i), while, for v of length n', $\mathrm{wt}(v) = \mathrm{wt}(v')$, where v' is v followed by $n - n'$ 0's (so $\psi(v') = v$). This shows that each coset of C has the same weight as the corresponding coset of C', so C and C' have the same coset weight distribution and hence the same covering radius. It is easy to verify that $d(C) = 1$ if and only if $d(C') = 1$, and that $d(C) \leq 2$ in any case (the vector with 1's at coordinates 1 and $n'+1$ and 0's elsewhere is in C), so $d(C) = \min(d(C'), 2)$.

If we puncture C and C' at corresponding coordinates i and j, then we can define a linear map $\psi': GF(2)^n \to GF(2)^{n'-1}$ by letting $\psi'(u)$ be $\psi(u)$ without coordinate j. Then $\psi'(u)$ will not depend on u_i, so we get an induced map $\psi'': GF(2)^{n-1} \to GF(2)^{n'-1}$. It is easy to verify that C_p is the inverse image of C'_p under ψ'', so $R(C_p) = R(C'_p)$ as before.

If u has length n and $u_i = 0$, then we can convert $\psi(u)$ into a member of C' by modifying at most $R(C')$ coordinates. By doing corresponding modifications at coordinates of u other than u_i (which is possible because each coordinate of C' corresponds to more than one coordinate of C), we can convert u into a member of C which also is 0 at coordinate i. This shows that $R(C_s) \leq R(C')$; since $R(C_s) \geq R(C)$, we have $R(C_s) = R(C')$. A similar argument shows that any u with $u_i = 1$ is within distance $R(C')$ of a member of C which is also 1 at coordinate i. It follows that $N(C) \leq 2R(C) + 1$, so C is normal. If $R(C') = R(C'_p)$, we can choose u such that $d(\psi'(u), C'_p) = R(C)$; this implies $d(u, C_\varepsilon^{(i)}) \geq R(C)$ and $d(u, C_{1-\varepsilon}^{(i)}) \geq R(C) + 1$, where $\varepsilon = u_i$, so $N^{(i)}(C) = 2R(C) + 1$. On the other hand, if $R(C') > R(C'_p)$, then for any u we have $d(u, C_\varepsilon^{(i)}) \leq R(C)$ and $d(u, C_{1-\varepsilon}^{(i)}) \leq R(C) + 1$, and equality can never hold in both places because $d(\psi'(u), C'_p) < R(C)$, so $N^{(i)}(C) = 2R(C)$.

(b) If a word in C' has weight w, then the corresponding repeated word of C has weight mw, so $d(C) = md(C')$. If m is even and u consists of $n/2$ 1's followed by $n/2$ 0's, then u is at distance $n/2$ from any repeated word

at all, so $R(C) \geq n/2$. If m is odd and u' is at distance $R(C')$ from C', then the vector u consisting of u' followed by $n'(m-1)/2$ 1's followed by $n'(m-1)/2$ 0's must be at distance at least $R(C') + n'(m-1)/2$ from any repeated member of C' (one must change $R(C)$ of the first n' coordinates and $n'(m-1)/2$ of the rest), so $R(C) \geq R(C') + n'(m-1)/2$. For $N(C) \leq n$ and hence $R(C) \leq n/2$ when C has no identically-0 coordinates, see Graham and Sloane [14].

(Similar arguments appear in Graham and Sloane [14], Mattson [22], and Kilby and Sloane [19].)

Now assume m is even; then $N(C) \geq 2R(C) = n$, so $N(C) = n$. If C has no identically-0 coordinates, then neither does C_p, so $R(C) - 1 \leq R(C_p) \leq \lfloor (n-1)/2 \rfloor$, which gives $R(C_p) = n/2 - 1$. But C_s has exactly $(z+1)m - 1$ identically-0 coordinates, and what remains when these are deleted is another repetition code of multiplicity m, so $R(C_s) = (z+1)m - 1 + m(n' - z - 1)/2 = m(n' + z + 1)/2 - 1$.

(c) For $d(C) = d(C')$, note that any nonzero word in C must be nonzero in at least one of the m blocks of length n' and hence must have weight at least $d(C')$ in that block; on the other hand, one nonzero block will suffice. Each coset of C is a product of a sequence of m cosets of C', and the weight of the C-coset is clearly the sum of the weights of the C'-cosets; it is now easy to derive the relation between the coset weight polynomials of C and C', and the equation $R(C) = mR(C')$ follows immediately. The last three equations are easy to prove, using the fact that, when working with a particular coordinate of C, the block containing that coordinate can be manipulated independently of the remaining blocks. □

In our list of cyclic codes, we omitted those codes of types (1), (2o), (2e), (3e), and (4e); these can all easily be identified just by looking at n and the list of roots (but we did not do it manually, of course). Codes of types (3o) and (4o) are not so easily identified by inspection; one needs a list of cyclotomic coset numbers as well for (3o), and (4o) requires knowing which cyclotomic coset numbers have to have the same multiplicity in order for $g(x)$ to be of the form $g'(x^m)$. Furthermore, such codes were included in the published code lists we looked at. (We found no published lists of cyclic codes of even length, so we had no examples to go by there.) Therefore, we included these codes in our code list, and then verified the properties listed above to check the correctness of our computations; the property $w = w'^m$ for codes of type (4o) was an especially convincing test. (Table 1 indicates which codes are of types (3o) and (4o).)

6. Results and applications

The Appendix on microfiche consists of two long tables containing the results of our covering radius computations. Table 1 lists the cyclic codes of length at most 64 and codimension at most 28 (omitting the trivial or composite codes of

types (1), (2e), (2o), (3e), or (4e)), with the roots and the following parameters for each code: the minimum distance, the sphere-covering lower bound, the covering radii of the code and its punctured and shortened codes, and the norm (if known).

From the data, we conclude that all but eight of the cyclic codes for which we compute the covering radius are normal, and we determine the norm for all but seven of the normal ones (see §3 for details).

To compute the sphere-covering lower bound on the covering radius, we have applied Fact 4 to cyclic codes having 0 among their roots because such codes are even-weight codes. Note that Fact 4 gives a better lower bound (on the R of an even code) than Fact 3 for some codes, such as C_2^{12} and C_2^{15}.

Table 2 gives the coset weight distributions of the codes in the list and the corresponding punctured and shortened codes. As an application of these numbers, we determine the minimum distance d of a cyclic code C as follows: Let the distribution of coset weights of C be $\{\alpha_0, \alpha_1, \ldots, \alpha_n\}$. Let e be the largest integer such that $\alpha_i = \binom{n}{i}$ for $0 \le i \le e$. Then it is not difficult to see that d is either $2e+1$ or $2e+2$. Therefore, since e can be determined from Table 2, d can be determined to within 1. Since C is a cyclic code, C_p has minimum distance $d-1$. Therefore, if $e_p = e$ from Table 2, then $d = 2e+2$, and if $e_p = e-1$, then $d = 2e+1$. Once we determine d, we know that, since C is cyclic, the minimum distance of its shortened code C_s is also d.

We give the following applications of our results.

6.1. **The amalgamated direct sum construction.** Suppose A is an $[n_1, k_1]R_1$ code and B is an $[n_2, k_2]R_2$ code. The *amalgamated direct sum* (ADS) of A and B, denoted $A \dot{\oplus} B$ [14], is an $[n_1 + n_2 - 1, k_1 + k_2 - 1]$ code. As an application of normality, if A and B are normal, then $R(A \dot{\oplus} B) \le R(A) + R(B)$ [14]. As an application of norms, if we know that the normal codes A and B have even norms, then the upper bound on $R(A \dot{\oplus} B)$ can be improved to $R(A \dot{\oplus} B) = R(A) + R(B) - 1$, and the resulting code is normal [18].

Janwa and Mattson [18] give several applications of the covering radii of punctured and shortened codes in the ADS construction.

6.2. **Improvements to the table of $t[n, k]$.** We show for twelve new values of n and k that the function $t[n, k]$ attains the sphere-covering lower bound. Thus, we give a dozen improvements in Table I of $t[n, k]$ in Graham and Sloane [14]. We also show that each of these covering radii is attained by a normal code; this lets us use the ADS construction to get good covering codes of larger lengths.

1. $t[49, 33] = t[50, 34] = t[51, 35] = 4$: Graham and Sloane [14] have $4 \le t[n, k] \le 5$ for each of these values of n and k. The $[51, 35]$ cyclic code $C = C_8^{51}$ with roots $1, 9$ has $R = 4$. By computation, $R(C_s) = R(C_{ss}) = 4$. Therefore, $t[49, 33] = t[50, 34] = t[51, 35] = 4$. Since $R(C_{sss}) = 5$, again by computation, the normality of the three codes follows from Proposition 1.

Note that the code C_{12p}^{51} also attains the bound $t[50, 34] = 4$. It is normal, since another computation shows that the covering radius of its shortened code is 4.

2. $t[52, 36] = 4$: If we add any column to a check matrix for C_8^{51}, then we get a $[52, 36]$ code C^+ whose covering radius is at most 4. Since $4 \leq t[52, 36] \leq 5$ [14], we conclude that $t[52, 36] = 4$. For the normality of C^+, observe that C_8^{51} is a shortened code of C^+.

3. $t[59 + i, 39 + i] = 5$ for $0 \leq i \leq 5$: Graham and Sloane [14] have $5 \leq t[n, k] \leq 6$ for each of these values of n and k. The $[59, 39]$ code C_{206p}^{60} has $R = 5$. Therefore, $t[59, 39] = 5$. If we take any check matrix for $D = C_{206p}^{60}$ and augment it by adjoining any i columns $(i \geq 0)$, we get a $[59 + i, 39 + i]$ code D_i with covering radius at most 5. Therefore, $t[59 + i, 39 + i] = 5$ for $1 \leq i \leq 5$. For normality of D_i $(1 \leq i \leq 5)$, note that D_{i-1} is its shortened code and apply Proposition 1.

The $[63, 43]$ cyclic code C_{162}^{63} with roots $1, 5, 21, 31$ also has $R = 5$. It is normal, because the covering radius of its shortened code is 6.

4. $t[63, 49] = t[64, 50] = 3$: Graham and Sloane [14] have $3 \leq t[n, k] \leq 4$ for each of these values of n and k. The $[63, 49]$ cyclic code C_{36}^{63} with roots $1, 5, 21$ has $R = 3$. Therefore, $t[63, 49] = 3$. It is normal, because its shortened code has $R = 4$. By adding a column to a check matrix for C_{36}^{63}, we get a $[64, 50]$ code with R at most 3. Thus, $t[64, 50] = 3$. It is normal because C_{36}^{63} is its shortened code.

We also have:

$$[15, 11]1 \dot{\oplus} [49, 33]4 = [63, 43]5,$$
$$[16, 12]1 \dot{\oplus} [49, 33]4 = [64, 44]5,$$
$$[15, 11]1 \dot{\oplus} [50, 34]4 = [64, 44]5.$$

Theorem 19 from Graham and Sloane [14] implies that the resulting codes are normal.

Calderbank and Sloane [7] have also shown that $t[64, 44] = 5$, since $[42, 33]2 \dot{\oplus} [23, 12]3 = [64, 44]5$. (For $t[42, 33] = 2$, see Brualdi, Pless, and Wilson [4].)

(*Note.* The $[42, 33]2$ code constructed by Brualdi, Pless, and Wilson [4] is normal by Proposition 2 (see also Calderbank and Sloane [7, p. 1280]). Therefore, by Theorem 20 of Graham and Sloane [14], there exists a $[43, 34]2$ normal code; since Graham and Sloane [14] give $2 \leq t[43, 34] \leq 3$, we must have $t[43, 34] = 2$.)

Our computations also show that many of the bounds given in [14], which were achieved by codes coming from an amalgamated direct sum construction, are in fact attained by cyclic codes, whose symmetry makes them easier to work with.

6.3. **Improved bounds on** $l(m, R)$. The results of §6.2 yield improvements in the known values of $l(m, R)$ [3] for the following three values of (m, R):

1. $47 \leq l(14, 3) \leq 63$ (from $47 \leq l(14, 3) \leq 71$);
2. $36 \leq l(16, 4) \leq 49$ (from $36 \leq l(16, 4) \leq 51$); and
3. $43 \leq l(20, 5) \leq 59$ (from $43 \leq l(20, 5) \leq 63$).

6.4. **Improved bounds on** $d_{\max}(n, k)$.

Definition 1. For given n and k, let

$$d_{\max}(n, k) = \max\{d | \text{ there exists an } [n, k, d] \text{ code}\}.$$

The latest table of upper and lower bounds on $d_{\max}(n, k)$ appears in Verhoeff [29].

Theorem 3. *We have the following four improvements of the lower bounds on* $d_{\max}(n, k)$:

(i) $d_{\max}(45, 16) \geq 13$ (*from* 12);
(ii) $d_{\max}(46, 16) \geq 14$ (*from* 13);
(iii) $d_{\max}(51, 25) \geq 11$ (*from* 10); *and*
(iv) $d_{\max}(52, 25) \geq 12$ (*from* 11).

For the proof we need the following lemma.

Lemma 1. *Let* C *be an* $[n, k, d]$ *code with* $R(C) \leq d$. *Then there exists an* $[n + d - R, k + 1, d]$ *code.*

Proof. Let x be a coset leader of C of weight R. We adjoin $d - R$ 1's to x to get the vector $x' = (x, 1, \ldots, 1)$. Let C' denote the code C extended by adjoining $d - R$ columns of zeros to C. Then $D = \langle C', x' \rangle$ is an $[n + d - R, k + 1, d]$ code. □

Proof of Theorem 3. (1) The code C_3^{43} is a $[43, 15, 13]11$ code. Therefore, by Lemma 1, there exists a $[45, 16, 13]$ code.

(2) If we append an overall parity check to the $[45, 16, 13]$ code in (1), we get a $[46, 16, 14]$ code.

(3) The quadratic residue code C_1^{47} is a $[47, 24, 11]7$ code ($R = 7$ was first proved by Delsarte [10]). Therefore, by Lemma 1, there exists a $[51, 25, 11]$ code.

(4) Append an overall parity check to the $[52, 25, 11]$ code in (3). □

Codes which improve known bounds for $d_{\max}(n, k)$ often also improve known bounds for $r(n, d)$, which is defined to be the minimal redundancy of a binary code (linear or nonlinear) of length n and minimum distance d; the redundancy of a code of length n containing M code words is defined to be $n - \log_2 M$. The best known redundancies are tabulated in Zinov'ev and Litsyn [30], which gives the bounds $r(45, 13) \leq 29$, $r(46, 14) \leq 30$, $r(51, 11) \leq 27$, and $r(52, 12) \leq 28$. Theorem 3 improves the latter two of these to $r(51, 11) \leq 26$ and $r(52, 12) \leq 27$. It also gives linear codes meeting

the best known bounds for $r(45, 13)$ and $r(46, 14)$, whereas the only previously known codes meeting these bounds were nonlinear [30].

Remark 1. The vector $1^5 0^2 1^3 0^6 10^3 10^3 10^{18}$ is a coset leader of C_3^{43} of weight 11. A check matrix for this code was generated by cyclic shifts of the vector $10^{27} 101^3 0^2 1^2 0^2 1^3 0$.

From C_3^{43} we generated a [45, 16, 13] code using the method outlined in the proof of Lemma 1. The weight distribution of the resulting [45, 16, 13] code is: $A_0 = 1$, $A_{13} = 318$, $A_{14} = 415$, $A_{15} = 453$, $A_{16} = 1515$, $A_{17} = 2105$, $A_{18} = 3067$, $A_{19} = 4800$, $A_{20} = 5554$, $A_{21} = 6938$, $A_{22} = 7540$, $A_{23} = 7372$, $A_{24} = 7372$, $A_{25} = 5648$, $A_{26} = 4444$, $A_{27} = 3447$, $A_{28} = 1790$, $A_{29} = 1304$, $A_{30} = 901$, $A_{31} = 311$, $A_{32} = 152$, $A_{33} = 71$, $A_{34} = 17$, $A_{43} = 1$. Here, A_i denotes the number of codewords of weight i; only nonzero A_i are listed.

The vector $1^6 0^3 10^{37}$ is a coset leader of C_1^{47} of weight 7. A check matrix for C_1^{47} was generated by cyclic shifts of the vector $10^{22} 10^3 1^2 0^2 1^2 01^2 0^2 10^2 1010^2 1$.

The weight distribution of the resulting [51, 25, 11] code is: $A_0 = 1$, $A_{11} = 4333$, $A_{12} = 13017$, $A_{13} = 160$, $A_{14} = 608$, $A_{15} = 180433$, $A_{16} = 362934$, $A_{17} = 16800$, $A_{18} = 40800$, $A_{19} = 1754332$, $A_{20} = 2509820$, $A_{21} = 326688$, $A_{22} = 544480$, $A_{23} = 4672220$, $A_{24} = 5004772$, $A_{25} = 1496352$, $A_{26} = 1768416$, $A_{27} = 4252526$, $A_{28} = 3586630$, $A_{29} = 1768416$, $A_{30} = 1496352$, $A_{31} = 1520662$, $A_{32} = 1009745$, $A_{33} = 544480$, $A_{34} = 326688$, $A_{35} = 192156$, $A_{36} = 93916$, $A_{37} = 40800$, $A_{38} = 16800$, $A_{39} = 6204$, $A_{40} = 2068$, $A_{41} = 608$, $A_{42} = 160$, $A_{43} = 45$, $A_{44} = 9$, $A_{47} = 1$.

6.5. **New quasi-perfect codes.** The following codes are quasi-perfect (i.e., $R = e + 1$) and could be added to the list of quasi-perfect codes in Peterson and Weldon [24, p. 122]: C_{1p}^7, C_{2p}^7, C_{2p}^9, C_{1p}^{12}, C_{1p}^{14}, C_{2p}^{14}, C_{1p}^{15}, C_{2p}^{15}, C_{4p}^{15}, C_{10p}^{15}, C_1^{21}, C_{1s}^{21}, C_{2p}^{21}, C_3^{21}, C_{3s}^{21}, C_{4p}^{21}, C_9^{21}, C_{1p}^{28}, C_{1p}^{30}, C_{5p}^{30}, C_{6p}^{30}, C_{1p}^{31}, C_{2p}^{31}, C_{2p}^{33}, C_{1p}^{35}, C_{1p}^{42}, C_2^{42}, C_{2s}^{42}, C_{3p}^{42}, C_{4p}^{42}, C_5^{42}, C_{5s}^{42}, C_{7p}^{42}, C_{1s}^{51}, C_{2p}^{51}, C_{1p}^{60}, C_{1p}^{62}, C_{2p}^{62}, C_{1p}^{63}, C_{2p}^{63}, C_{6p}^{63}, C_{32p}^{63}, and C_{36}^{63}.

The following codes are known to be quasi-perfect, and we verify this: C_4^{31} and C_5^{31} [11]; C_1^{51} [16]; C_7^{15}, C_1^{17}, and C_1^{23} [24, pp. 122–123]; C_1^7, C_1^{15}, C_1^{31}, C_3^{31}, C_1^{63}, and C_{17}^{63} [8]; C_{1p}^{23} [5]; and C_{1s}^7, C_{1s}^{15}, C_{1s}^{31}, and C_{1s}^{63} [17].

Of these quasi-perfect codes, the following are uniformly packed (i.e., for these codes equality holds in Fact 5) and could be added to the lists of uniformly packed codes in Calderbank and Kantor [6, pp. 117–118] and Brouwer, Cohen, and Neumaier [2, pp. 355–357]; C_{2p}^7, C_{1p}^{12}, C_{1p}^{15}, C_{2p}^{15}, C_3^{21}, C_9^{21}, C_{1p}^{23}, C_{1p}^{31}, C_{2p}^{31}, C_4^{31}, C_{1s}^{51}, C_{1p}^{60}, C_{1p}^{63}, and C_{2p}^{63}.

The cyclic code C_1^{17} [24, p. 122], the punctured Golay code C_{1p}^{23} [5], the code C_1^{51} [16], 2-error-correcting BCH codes of lengths $2^{2m+1} - 1$ (including C_3^{31}) [20], and the shortened subcodes of the Hamming perfect codes (including C_{1s}^7, C_{1s}^{15}, C_{1s}^{31}, and C_{1s}^{63}) [18] are known to be uniformly packed.

Calderbank and Goethals [5] prove that any uniformly packed code of length 21 has the same parameters as the code obtained by puncturing the Golay code C_1^{23} twice. One such code is C_9^{21} in our list. It is natural to conjecture that there is only one uniformly packed code of length 21; J. I. Hall informs us that this is indeed the case.

For connection between uniformly packed codes, strongly regular graphs, and other combinatorial structures, see Chapter 11 of Brouwer, Cohen, and Neumaier [2].

6.6. **Equivalence of codes.** Since equivalent linear codes have the same parameters given in Tables 1 and 2, our computations can be used to show nonequivalence of cyclic codes or their punctured or shortened codes. We illustrate this by completing the classification of all *cyclic self-dual* codes of length n up to 60 begun in Sloane and Thompson [27]. In that paper, an upper bound on the number $c(n)$ of nontrivial, nonequivalent cyclic self-dual codes of length n is given. It is shown there that for $n \leq 54$, only $c(42)$ is not precisely known and that $1 \leq c(42) \leq 4$. The upper bound is given by the four codes C_{133}^{42}, C_{134}^{42}, C_{136}^{42}, and C_{137}^{42}. Since their respective parameters (d, R) are different, we conclude that $c(42) = 4$.

From the upper bound on $c(n)$ mentioned above, we also conclude that $1 \leq c(56) \leq 4$ and $1 \leq c(60) \leq 2$. For $n = 56$ the upper bound is given by the four cyclic codes C_{106}^{56}, C_{107}^{56}, A, and B, where A is a direct sum code with roots $0^4 1^8$ and B is a direct sum code with roots $0^4 1^6 3^2$. From Table 1, we have $d(C_{106}^{56}) = 6$ and $d(C_{107}^{56}) = 4$, and from §5 we have $d(A) = d(C_2^{14}) = 4$ and $d(B) = d(C_{16}^{28}) = 4$. Also, $R(A) = 4R(C_2^{14}) = 12$, $R(B) = 2R(C_{16}^{28}) = 14$, and $R(C_{107}^{56}) = 14$. To distinguish B from C_{107}^{56}, note that the number of cosets of weight 2 is 1498 for B and 1435 for C_{107}^{56} (see Table 2 and §5). Therefore, the four codes are nonequivalent, so $c(56) = 4$. For $n = 60$ the two distinct nontrivial cyclic self-dual codes are a code C with roots $0^2 1^3 3^2 5^2 7$ and a direct sum code D with roots $0^2 1^4 3^2 5^2$. By computation, $R(C) = 15$, while from §5, $R(D) = 2R(C_{42}^{30}) = 14$. Therefore, $c(60) = 2$.

Acknowledgments

We would like to thank Eric Van de Velde for useful discussions of the algorithm and for his help in getting the program to run on hypercube systems. We would also like to thank the Caltech Concurrent Computation Project, the UCLA Cognitive Science group, and the Advanced Computing Laboratory at Los Alamos National Laboratory for providing computing resources. We are thankful to Henk van Tilborg for helpful discussions regarding the code extension techniques used in §6.4.

IBM PC is a trademark of IBM Corporation. Connection Machine is a trademark of Thinking Machines Corporation.

BIBLIOGRAPHY

1. E. R. Berlekamp, *Algebraic coding theory*, McGraw-Hill, New York, 1968.
2. A. E. Brouwer, A. M. Cohen, and A. Neumaier, *Distance-regular graphs*, Springer-Verlag, New York, 1989.
3. R. A. Brualdi and V. S. Pless, *On the length of codes with a given covering radius*, preprint.
4. R. A. Brualdi, V. S. Pless, and R. M. Wilson, *Short codes with a given covering radius*, IEEE Trans. Inform. Theory **IT-35** (1989), 99–109.
5. A. E. Calderbank and J.-M. Goethals, *On a pair of dual subschemes of the Hamming scheme* $H_n(q)$, European J. Combin. **6** (1985), 133–147.
6. A. E. Calderbank and W. M. Kantor, *The geometry of two-weight codes*, Bull. London Math. Soc. **18** (1986), 97–122.
7. A. R. Calderbank and N. J. A. Sloane, *Inequalities for covering codes*, IEEE Trans. Inform. Theory **IT-34** (1988), 1276–1280.
8. G. D. Cohen, M. G. Karpovsky, H. F. Mattson, Jr., and J. R. Schatz, *Covering radius—survey and recent results*, IEEE Trans. Inform. Theory **IT-31** (1985), 328–343.
9. G. D. Cohen, A. C. Lobstein, and N. J. A. Sloane, *Further results on the covering radius of codes*, IEEE Trans. Inform. Theory **IT-32** (1986), 680–694.
10. Ph. Delsarte, *Four fundamental parameters of a code and their combinatorial significance*, Inform. and Control **23** (1973), 407–438.
11. S. M. Dodunekov, *Some quasi-perfect double error correcting codes*, Problemy Peredachi Informatsii **3** (1984), 17–23.
12. D. E. Downey and N. J. A. Sloane, *The covering radius of cyclic codes of length up to* 31, IEEE Trans. Inform. Theory **IT-31** (1985), 446–447.
13. V. D. Goppa, *Algebraico-geometry codes*, Math. USSR-Izv. **21** (1983), 75–91.
14. R. L. Graham and N. J. A. Sloane, *On the covering radius of codes*, IEEE Trans. Inform. Theory **IT-31** (1985), 385–401.
15. J. W. P. Hirschfeld, *Linear codes and algebraic curves*, Geometrical Combinatorics (F. C. Holroyd and R. J. Wilson, eds.), Pitman, 1984, pp. 35–53.
16. H. Janwa, *Some new upper bounds on the covering radius of binary linear codes*, IEEE Trans. Inform. Theory **IT-35** (1989), 110–122.
17. H. Janwa and H. F. Mattson, Jr., *The covering radius and normality of t-dense codes*, presented in part at the IEEE Internat. Sympos. on Inform. Theory, Ann Arbor, Michigan, October 1986 (to appear).
18. ——, *On the normality of binary linear codes*, IEEE Trans. Inform. Theory (to appear).
19. 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 Discrete Methods **8** (1987), 604–627.
20. F. J. MacWilliams and N. J. A. Sloane, *The theory of error-correcting codes*, North-Holland, Amsterdam, 1977.
21. H. F. Mattson, Jr., *Another upper bound on covering radius*, IEEE Trans. Inform. Theory **IT-29** (1983), 356–359.
22. ——, *An improved upper bound on covering radius*, Applied Algebra, Algorithmics, and Error-correcting Codes (Toulouse, 1984), Lecture Notes in Comput. Sci., vol. 288, Springer-Verlag, New York, 1986, pp. 90–106.
23. A. McLoughlin, *The complexity of computing the covering radius of a code*, IEEE Trans. Inform. Theory **IT-30** (1984), 800–804.
24. W. W. Peterson and E. J. Weldon, Jr., *Error-correcting codes*, 2nd ed., MIT Press, Cambridge, MA, 1972.
25. V. Pless, *Introduction to the theory of error-correcting codes*, Wiley, New York, 1981.

26. N. J. A. Sloane, *A new approach to the covering radius of codes*, J. Combin. Theory Ser. A **42** (1986), 61–86.
27. N. J. A. Sloane and J. G. Thompson, *Cyclic self-dual codes*, IEEE Trans. Inform. Theory **IT-29** (1983), 364–366.
28. J. H. van Lint, *Introduction to coding theory*, Springer-Verlag, New York, 1982.
29. T. Verhoeff, *An updated table of minimum-distance bounds for binary linear codes*, IEEE Trans. Inform. Theory **IT-33** (1987), 665–680.
30. V. A. Zinov′ev and S. N. Litsyn, *Table of best known binary codes*, preprint, 1984.

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF CALIFORNIA, LOS ANGELES, CALIFORNIA 90024
Current address: Department of Mathematics, Ohio State University, Columbus, Ohio 43210
E-mail address: rld@function.mps.ohio-state.edu

DEPARTMENT OF MATHEMATICS, CALIFORNIA INSTITUTE OF TECHNOLOGY, PASADENA, CALIFORNIA 91125
Current address: School of Mathematics, Tata Institute of Fundamental Research, Colaba, Bombay 400 005, India
E-mail address: janwa@tifrvax.bitnet