Provably Good Codes for Hash Function Design

Charanjit S. Jutla[1] and Anindya C. Patthak[2,⋆]

[1] IBM Thomas J. Watson Research Center
csjutla@watson.ibm.com
[2] University of Texas at Austin
anindya@cs.utexas.edu

Abstract. We develop a new technique to lower bound the minimum distance of quasi-cyclic codes with large dimension by reducing the problem to lower bounding the minimum distance of a few significantly smaller dimensional codes. Using this technique, we prove that a code which is similar to the SHA-1 message expansion code has minimum distance at least 82, and that too in just the last 64 of the 80 expanded words. Further the minimum weight in the last 60 words (last 48 words) is at least 75 (52 respectively). We expect our technique to be helpful in designing future practical collision-resistant hash functions. We also use the technique to find the minimum weight of the SHA-1 code (25 in the last 60 words), which was an open problem.

Keywords: linear codes, minimum distance, collision-resistant hash functions, SHA-1.

1 Introduction

Recall the SHA-1 message expansion code which is a binary linear code of dimension 512: the 512 information bits are packed into 16 32-bit words $\langle W_0, \cdots, W_{15}\rangle$, and 64 additional words are generated by the recurrence:

$$W_i = (W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16}) \lll 1 \quad \text{for } i = 16, \cdots, 79 \qquad (1)$$

The 80 words $\langle W_0, \cdots, W_{79}\rangle$ can be seen as constituting a code-word in a linear code over $\mathbb{F}_2$ with the above parity check equations. Unfortunately, this code has a minimum distance or weight of no more than 44. Further, the weight restricted to the last 60 words is only 25. This has been exploited in [21] to give a differential attack on SHA-1 with complexity 2^{69} hash operations. Recently, the complexity has further been improved to 2^{63} hash operations [19].

The code for SHA-0, which is same as (1) but without the rotation (see [14], has an even worse minimum weight. The small minimum weight of these codes is an integral part of the attack strategies on these hash functions (see [22,23,3,2,1,20,21]). The question naturally arises as to why codes with better

⋆ This work was done while the author was visiting IBM T.J. Watson Research Center, N.Y.

E. Biham and A.M. Youssef (Eds.): SAC 2006, LNCS 4356, pp. 376–393, 2007.

minimum weight were not employed, even though the coding theory literature [17] is rife with codes with proven good minimum weight. However, as we point out later in section 2, none of them comes close to being as efficient to implement in software (i.e., do not have an efficient encoder) as the code (1) above. One is then led to ask if codes more complex than (1), but still easy to implement, could be shown to have a better minimum distance. Surprisingly, it was not even known how to lower bound the minimum weight of the above SHA-1 code, even though it is related to codes such as Hadamard code [17] (we address this relationship in section 2).

The purpose of this paper is three-fold. First, to introduce a novel technique for lower bounding efficient-to-implement codes such as given by (1). Second, to use this technique to lower bound this particular code (which was an open problem). Third, to show how one can design efficient-to-implement codes with a much better minimum distance, and to actually give such a code. We expect our technique to be helpful in designing future practical collision-resistant hash functions.

Before we describe our technique, we mention the specific code we analyze, as this specific example will help in understanding the complexity of the problem and the intricacy of the technique. The code, $\mathcal{C}$, we consider is a 80×32 length binary code of dimension 16×32, given by the following recurrence relation (or parity check equations): Let $V_i \stackrel{def}{=} W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16}$.

$$W_i \stackrel{def}{=} \begin{cases} V_i \oplus ((W_{i-1} \oplus W_{i-2} \oplus W_{i-15}) \ <<< \ 1) & \text{if } 16 \leq i < 36 \\ V_i \oplus ((W_{i-1} \oplus W_{i-2} \oplus W_{i-15} \oplus W_{i-20}) \ <<< \ 1) & \text{if } 36 \leq i \leq 79 \end{cases} \tag{2}$$

We will show that this code has minimum distance 82, and that too in just the last 64 words (contrast this with SHA-1 which has minimum weight at most 30 [21], and 192 for the highly inefficient Reed Solomon code described in Section 2). Of course, since the dimension of this code is 16×32, a brute force search of $2^{16 \times 32}$ is infeasible. Further, it is known that computing minimum weight of an arbitrary linear code is NP-hard (see [18]), and that approximating within a constant factor is NP-hard under randomized reduction (see [5]).

Still, there is additional structure in the above codes (i.e. (1) and (2)), and we intend to exploit that. Note that a codeword of the above codes can be seen as an 80×32 matrix, with each column representing the codeword projected on a particular bit position. Further, the above codes have the property that they are closed under column rotations. Such codes are called *quasi-cyclic codes* in the literature, and have been studied extensively (see [13,4,8,9]). As for estimating the minimum weight of such codes by algorithmic means, the presently known techniques are computationally infeasible [4,8].

Our novel technique reduces the problem of lower bounding the minimum weight of a $k \times n$ dimensional quasi-cyclic code to a function of the minimum weight of a few $k \times n'$ dimensional codes, where n' is much smaller than n. We now briefly explain the main idea of our technique, using the above example code given by (2). For any codeword represented as a 80×32 matrix, note that either (a) there are

no all-zero columns in the codeword, in which case we would like to show that on average there are a few (three, e.g.) non-zero bits in each column, or (b) there is a zero column in the codeword, in which case we would like to show that the code projected on a few columns (say, $m << n$) has a large minimum distance.

Unfortunately, there are two major hurdles in this plan (related to case (b)). Consider the first non-zero column next to a zero column (either to the left or the right). It turns out that the code projected on that column is not expected to be any better than the code for SHA-0, and hence we do not expect a minimum weight of more than 15-20 for that column. Thus, we would need m to be about five to get a minimum weight of 75, in which case the dimension of the projected code is still too large, i.e. 16×5. Further, there are *pathological cases* (and which cannot be avoided) where the code projected on a column yields a minimum weight as low as 1. Thus, we may be forced to consider m to be much larger than five. The novelty of our approach lies in tackling these two major hurdles. We show that the minimum weight of the sub-code in case (b) can be lower bounded by a function of the minimum weight of a few codes (some of which are subspaces), each of dimension at most 16×3. A "lazy" brute force search with early-stopping then yields a lower bound of 82.

Other Contributions: We also use the techniques developed to give a lower bound of 25 (in the last 60 words) on the minimum weight of the codewords of SHA-1 (this was an open problem). A codeword of weight smaller than 25, could potentially lead to an even more drastic attack on SHA-1. As for further advancement of our techniques, we also prove that the minimum weight of our example code (2) is at least 75 (52) when restricted to the last 60 (48 resp.) words. We will give detailed proof of this in the full version of the paper. Note that front truncations are not equivalent to back truncations for this code.

Organization: The rest of the paper is organized as follows: In section 2 we review limitations of known algebraic techniques. In section 3 we give an informal description of the proof technique and the intuition behind why certain codes are easier to analyze. In section 4 we give a detailed proof of a lower bound on the code given by (2). In the Conclusion section, we describe applications of our methods to designing hash functions. In Appendix A, we give detailed algebraic proofs of some lemmas in section 4.

2 Limitations of Purely Algebraic Techniques

We first investigate the SHA-0 code restricted to a single column, which is a length 80 binary code of dimension 16, given by the binary parity check equations:

$$a_i = a_{i-3} \oplus a_{i-8} \oplus a_{i-14} \oplus a_{i-16} \quad \text{for } i = 16, \cdots, 79 \tag{3}$$

Consider the polynomial $h(X) = X^{16} + X^{13} + X^8 + X^2 + 1$, which is known to be a primitive polynomial, as the smallest n such that $h(X)$ divides $X^n - 1$ is $2^{16}-1$. Hence, if the above code was extended up to length $2^{16}-1$, it would be the code generated by the LFSR (Linear Feedback Shift Register) given by primitive

polynomial $h(X)$. However, such codes are well known [24,7] to be a subcode of first-order Reed Muller codes (also known as Hadamard codes) with one digit dropped. Such codes have an extremely good minimum distance of $2^{15} - 1$, or fractional distance $1/2$. Unfortunately, nothing useful can be said about this code truncated to just the first 80 bits, based purely on known algebraic methods. In fact, any such code (i.e. using any degree 16 primitive polynomial) has a minimum weight of at most 26, i.e. a fractional distance of less than $1/3$ (as can be checked by a computer).

The lack of purely algebraic techniques to lower bound even this single column code emphasizes the difficulty of analyzing the more complex codes such as SHA-1 and that given by Equation (2). Of course, if $h(X)$ above was not primitive, and divided $X^{80} - 1$, then we would get a cyclic code of length 80. Such codes can be analyzed much more easily, and it is not too difficult to see that the best cyclic code gives a minimum distance of only 8. However, there are non-cyclic linear codes known of minimum distance 31, though they are really difficult to encode. One could also consider cyclic codes of length 85, which have a much better minimum distance and then truncate them. However, the analysis does not extend to codes which do column mixing like SHA-1.

Instead of quasi-cyclic codes as SHA-1 or Equation (2), one could consider cyclic codes of length 80×32, or of an appropriate length. First note that a random code will give minimum distance roughly 475 for a code with rate $1/4$ and length 64×32 (follows from the Gilbert-Varshamov bound). Of course, finding such a code is infeasible. Alternatively, one can try a Reed Solomon code over $\mathbb{F}_{2^8}$ of length $2^8 - 1$ (bytes), and dimension 64 (bytes). Such a code has distance $256 - 64 = 192$ (over bytes). However, the encoder for this code requires multiplication by various elements in $\mathbb{F}_{2^8}$, and is not at all suitable for software implementations. A binary cyclic code of dimension 16×32 would also be extremely cumbersome to implement. Similar considerations rule out known good quasi-cyclic codes.

3 Intuition Behind the Code

Let us start by examining why the message expansion code in SHA-1 given by Equation (1) is not satisfactory (observed independently in [11] and [10]). We can rewrite Equation (1) as follows:

$$\forall i, 0 \leq i \leq 63, \quad W_i = W_{i+2} \oplus W_{i+8} \oplus W_{i+13} \oplus (W_{i+16} >>> 1), \tag{4}$$

where ">>> 1" denotes a one bit rotation to the right. The above clearly shows that a difference created in the last 16 words propagates to only up to 4 different bit positions.

One way to remedy this situation is to let $W_i = (W_{i+2} >>> 1) \oplus W_{i+8} \oplus W_{i+13} \oplus (W_{i+16} >>> 1)$. Now Equation (1) becomes $W_i = (W_{i-3} \oplus W_{i-8} \oplus W_{i-16}) <<< 1 \oplus W_{i-14}$. Thus, whether you consider the evaluation in the *forward direction* or in the *reverse direction*, the spread of differences to the neighboring columns (i.e. neighboring bits) is more frequent. However, it is not

enough to just have a good intuition about the code, but one also needs to prove a good lower bound on the minimum weight of such codes.

The strategy we use to prove lower bounds on such codes is to divide the proof into two main cases. We argue that either there are no zero columns in a codeword (a column in the codeword is the codeword projected on a particular bit position) or starting from an all zero column, the first neighboring non-zero column is actually a codeword in a good code, and so on.

Elaborating on the first case, i.e., when there are no zero columns, if every column has at least 3 bits ON, we are done. So, assume that there is some column which has 1 or 2 bits ON. Thus, there are $(64 \times 63)/2 + 64$ choices for picking these bits in the column. Having picked these bits, the neighboring column is completely specified by at most 16 bits in that column. Now the two columns together either have weight 6, in which case we are maintaining an average of 3 per column, or the weight of these two columns is at most 5. Thus, our search is quite restricted. We continue in this fashion, noting that the code has to be designed carefully so as to satisfy a property as in Claim 3.

As for the second case, we consider a contiguous band of zero columns, bordered on both sides with non-zero columns (we prove that they cannot be same; in fact we prove by a rank argument that there must be at least four consecutive non-zero columns). We have to assure that when a column is zero, and the neighboring column is non-zero (whether to the right or left), the resulting code for the neighboring column is a good code, i.e., with a good minimum weight. Note that this is important since we may possibly have at most 5-6 non-zero columns. Therefore it is desired that the disturbance propagates fast across columns. Unfortunately, this is impossible for the codes we are considering so far.

Consider a SHA-1 like code, with dimension 16×32, and which is invariant under column rotations. Moreover, suppose that the code is of the form

$$W_i = \sum_{t=1}^{16} a_t W_{i-t} + \left(\left(\sum_{t=1}^{16} b_t W_{i-t} \right) <<< 1 \right), \tag{5}$$

where $a_1, \cdots, a_{16}, b_1, \cdots, b_{16}$ are boolean. If a_{16} and b_{16} are equal, then there is a codeword which is zero everywhere, except for W_0 which is the all one 32-bit word. Thus for the sake of the argument, assume that $b_{16} = 0$ and $a_{16} = 1$. However in this case, suppose $t' < 16$ is the largest t such that $b_{t'}$ is non-zero. First note that if a column, say C^j, is zero, then in the column to its right, say C^{j-1}, C^{j-1}_k (for $k = 0$ to $15 - t'$) can take any value (i.e., are free variables), and the rest of the column C^{j-1} can be all zero. Further, the propagation to columns C^{j-2}, C^{j-3} etc. can be rather weak.

A similar situation arises when the code is evaluated in the backward direction. The trick is to keep the above free variables few in number, so that the subspace of such *pathological cases* is of a relatively small dimension. This small dimension is absolutely necessary to keep the exhaustive search over this space tractable. One way to get rid of these pathological free variables is to include a term like W_{i-20}, as we do in our code. This in fact gets rid of all the pathological variables

in the forward direction and thereby yields a fast expansion. In the backward direction at least one pathological free variable per column remains, and we must search over such subspaces.

4 A Lower Bound on the Minimum Distance

In this section we will prove a lower bound on the code described in the introduction. As mentioned earlier, this is a general technique for reducing the problem to smaller dimensional codes. However, if the reduction is to codes with dimensions too large, then a brute force search may not be feasible. On the other hand, if the reduction is to codes which have really low minimum weight, then we will not obtain a good bound.

We will see in Claim 7 and Claim 8 (in Appendix A) that if the polynomials describing the parity check equations (5) have a certain algebraic property, namely that the polynomial corresponding to coefficients a_t is irreducible, and does not divide the polynomial corresponding to coefficients b_t, then some key reduced codes have low dimensions. Although, these are not necessary conditions, they make a good choice. Similarly, if the coefficients b_1 and b_{15} are both one, then the number of pathological variables per column is small.

We will prove a lower bound on the minimum weight of the code given by Equation (2), but projected on the last 64 words. Clearly, the same bound holds for the full 80 words. The reason we focus on the last 64 words is because the recent attacks on hash functions have shown that the weight of the code in early words (the information words, and a few following words) is mostly immaterial (see "message modification technique" in [21]), and hence the weight in the latter words decides the complexity of the attack.

Since we will be arguing about the weight of this code in the last 64 words, we instead consider the following code $\mathcal{C}64$: Let $W_0, \cdots, W_{15}$ be the message blocks. Let $V_i \stackrel{def}{=} W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16}$.

$$W_i \stackrel{def}{=} \begin{cases} V_i \oplus ((W_{i-1} \oplus W_{i-2} \oplus W_{i-15}) <<< 1) & \text{if } 16 \leq i < 20 \\ V_i \oplus ((W_{i-1} \oplus W_{i-2} \oplus W_{i-15} \oplus W_{i-20}) <<< 1) & \text{if } 20 \leq i \leq 63 \end{cases} \quad (6)$$

We first prove that this is indeed sufficient.

Lemma 1. *If the code $\mathcal{C}64$ described above has minimum weight at least w, then $\mathcal{C}$ has minimum weight at least w in its last 64 words.*

Proof. Consider any nonzero codeword in $\mathcal{C}$, say $U = \langle U_0, \cdots, U_{79} \rangle$. Denote $X = \langle U_0, \cdots, U_{15} \rangle$ and $Y = \langle U_{16} \cdots, U_{31} \rangle$ and $Z = \langle U_{32} \cdots, U_{79} \rangle$. Therefore $U = \langle X, Y, Z \rangle$. From Equation 2 observe that the code $\mathcal{C}$ is completely determined by specifying any consecutive 16 word block provided the block starts anywhere in 0 to 20, since the rest can then be obtained by solving the recurrence relation. We therefore choose to specify $Y = \langle U_{16}, \cdots, U_{31} \rangle$, that is we treat Y as the message symbols. Note that a fixed choice of Y also fixes X and Z. Following this observation it is now clear that $\langle Y, Z \rangle$ is a codeword in $\mathcal{C}64$.

Assume that the minimum weight of $\mathcal{C}64$ is d. Then we need to show that any non-zero codeword in $\mathcal{C}$ has weight at least d in its last 64 words. This follows provided a non-zero X implies a non-zero Y. However, if Y is zero then X is zero, as X is a linear function of Y.

Therefore the minimum weight of $\mathcal{C}64$ is exactly the minimum weight of code $\mathcal{C}$ in its last 64 words. ■

Next we prove a lower bound on the minimum distance of $\mathcal{C}64$. We break down the proof into several sub-cases. In each sub-case, we argue often following an exhaustive search over a small space that the minimum weight of the code is at least 82. We mention that a naive algorithm may require to search a space as large as $2^{32\times 16}$ which is clearly not feasible. Therefore the novelty in our approach lies in a careful sub-division of the problem into a small number of tractable cases.

Theorem 2. *The code $\mathcal{C}64$ as defined by Equation 6 has minimum distance at least 82.*

Proof. It is easy to see that the code $\mathcal{C}64$ is a quasi-cyclic code by noting that it is invariant under a 64 bit cyclic shift. From now onwards, we view the codewords of $\mathcal{C}64$ as a matrix that has 32 columns where each column is 64-bit long. The quasi-cyclic property then just mean that the code is invariant under column rotations. Unless otherwise specified, *the arithmetic in the superscript will be modulo 32*.

Now consider any non-zero codeword. Since the code is linear, it suffices to prove that it has weight at least 82. We break down the proof into two main cases depending upon whether or not a codeword has zero columns.

1. (**All Columns Non-Zero Case:**) Consider any such codeword. Also, consider any non-zero column, w.l.o.g., let it be C^0. Denote the columns, to the left of it by $C^1, C^2, \cdots, C^{31}$. Note that all C^i's are non-zero. In this case the following claim holds.

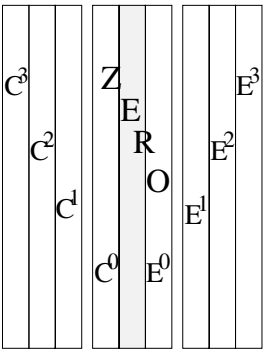
   Claim 3. *For any non-zero column C^j, there exists $k, 0 \leq k \leq 7$ such that the combined weight of columns $C^j, C^{j+1}, \cdots, C^{j+k}$ is at least $3 \cdot (k+1)$.*

   Proof. This is easily verified by a computer program. We mention that for $k \leq 6$, an average of 3 cannot be assured (see Appendix B for an example). ■

   Next we create a partition of the 32 columns into several groups. We pick a non-zero column C^j. Now following Claim 3, there exists $(k+1)$-columns $(0 \leq k \leq 7)$ such that the average weight of each column is at least 3. Consider the smallest k that achieves this. Then put these $(k+1)$ columns $C^j, C^{j+1}, \cdots, C^{j+k}$ into a group. Call these columns good columns and the group a good group. We then choose C^{k+j+1} and form another group. We continue like this till no more good groups can be created. The remaining columns are then grouped together. Call this group a bad group. Note that the bad group has average weight at least 1. Now let e be the size of this bad group. Then we have $(32-e)$ good columns. Also following Claim 3, e

could be at most 7. Therefore the total weight of the codeword is at least $3 \cdot (32 - e) + e = 96 - 2 \cdot e \geq 82$.

2. (**At least One column Zero Case:**) Assume that there is at least one zero column. W.l.o.g. let C^0 be a zero column such that the column to the left of it is non-zero (note that such a column always exists since we are considering a non-zero codeword). Denote the columns to the left of C^0 by $C^1, C^2, \cdots$ (see figure).



Also, going towards the right of C^0, denote the first non-zero column by E^1 and thereafter $E^2, E^3, \cdots$. Denote the column to the left of E^1 by E^0. (Note that it may be possible that C^0 and E^0 are the same column.) We argue that a few columns to the left and right of a band of zero columns must contribute a total weight of at least 82.

It will be immaterial in our analysis below if there are some non-zero columns between C^0 and E^0. All we require in our analysis is that C^0 and E^0 are zero.

Next consider $C^1, C^2, \cdots$. How soon can the sequence yield a zero column, i.e., what is the smallest value of j such that $C^j = E^0$? In order to answer this question, first note that since C^0 is everywhere zero, C^1 is essentially generated by the code whose parity check equations over $\mathbb{F}_2$ are given as follows: Denote $C^1 = \langle y_0, \cdots, y_{63} \rangle$. Then

$$\forall i, 16 \leq i \leq 63, \quad 0 = y_i + y_{i-3} + y_{i-8} + y_{i-14} + y_{i-16}. \tag{7}$$

Similarly for a fixed C^1, the column C^2 is generated by the code whose parity check equations over $\mathbb{F}_2$ are given as follows: Denote $C^2 = \langle x_0, \cdots, x_{63} \rangle$. Let $u_i \stackrel{def}{=} x_i + x_{i-3} + x_{i-8} + x_{i-14} + x_{i-16}$.

$$0 = \begin{cases} u_i + y_{i-1} + y_{i-2} + y_{i-15} & \text{for } 16 \leq i \leq 19 \\ u_i + y_{i-1} + y_{i-2} + y_{i-15} + y_{i-20} & \text{for } 20 \leq i \leq 63 \end{cases} \tag{8}$$

On the other hand E^1 is generated by the code whose parity check equations over $\mathbb{F}_2$ are given as follows: Denote $E^1 = \langle w_0, \cdots, w_{63} \rangle$. Then

$$0 = \begin{cases} w_{i-1} + w_{i-2} + w_{i-15} & \text{for } 16 \leq i \leq 19 \\ w_{i-1} + w_{i-2} + w_{i-15} + w_{i-20} & \text{for } 20 \leq i \leq 63 \end{cases} \tag{9}$$

Similarly for a fixed E^1, the column E^2 is generated by the code whose parity check equations over $\mathbb{F}_2$ are given as follows: Denote $E^2 = \langle z_0, \cdots, z_{63} \rangle$. Let $v_i \stackrel{def}{=} w_i + w_{i-3} + w_{i-8} + w_{i-14} + w_{i-16}$. Then

$$0 = \begin{cases} v_i + z_{i-1} + z_{i-2} + z_{i-15} & \text{for } 16 \leq i \leq 19 \\ v_i + z_{i-1} + z_{i-2} + z_{i-15} + z_{i-20} & \text{for } 20 \leq i \leq 63 \end{cases} \tag{10}$$

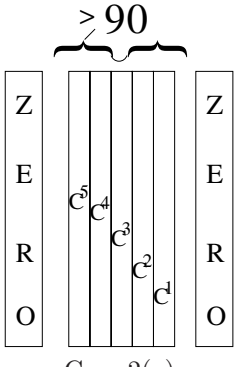
The following claim shows that at least four consecutive columns have to be non-zero.

Claim 4. *If C^0 is everywhere zero, and C^1 is non-zero, then C^2, C^3 and C^4 are all non-zero.*

Proof. Suppose for a j it is the case that $C^j = E^1$, i.e., C^{j+1} is all zero. Then a homogeneous system of linear equations over $\mathbb{F}_2$ can be set up. Consider the $64 \times j$ variables in column C^1 through C^j. There are 48 equations for each of the columns C^1 through C^j. Also, there are 48 more equations for C^{j+1}. It is well known that such a system can have a non-trivial solution if and only if the rank of the co-efficient matrix is strictly smaller than the number of variables. It can easily be verified by a computer program that for $j = 1, 2, 3$, the system has full rank, that is exactly $64 \times j$. This can also be proved algebraically for $j = 1, 2$. We give a simple algebraic proof in the appendix (see Appendix A). ■

This proof also highlights that for the rank to be full the recurrence relation must satisfy nice properties. Ranks of all linear systems considered in this paper have been computed using Gaussian elimination. We now divide the proof into two cases.

(a) (**Number Of Consecutive Non-Zero Columns is at most Five):** By the claim above, we can safely assume that we have at least four consecutive non-zero columns. Also, if we assume $C^4 = E^1$, then the number of nontrivial solutions can be at most $2^{16} - 1$ (since the co-rank or nullity of the matrix is 16, as verified by implementing a Gaussian elimination program). Similarly, assuming $C^5 = E^1$, the number of nontrivial solutions can be at most $2^{32} - 1$. We do an exhaustive search to conclude that the minimum weight in the latter case is at least 90. (Note that this latter case alone is sufficient.)



Case 2(a)

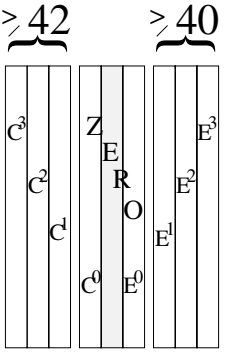
(b) (**Number Of Consecutive Non-Zero Columns is at least Six**): If case 1 and case 2(a) do not hold then, the only case that remains to be considered is the one where at least six consecutive columns are non-zero. Note that C^1, C^2, C^3 are then distinct from E^1, E^2, E^3. We use a computer program to verify that in this case the combined weight of C^1, C^2 and C^3 is at least 42. However the same cannot be said of E^1, E^2, E^3, and we have to do a more detailed analysis.

Now recall Equation 9, the constraints induced on E^1. A quick observation reveals that its free variables are the first 15 bits and the very

last bit. Depending on the values taken by E^1s first 15 bits we sub-divide our proof into two cases:

i. (**Non-Pathological Case:**) Not all of the first 15 bits of E^1 are zero.

This is the simpler case. In this case, the recurrence induces a good expansion. By an exhaustive search we obtain that in this case the combined weight of E^1, E^2 and E^3 is at least 40. Since the combined weight of C^1, C^2 and C^3 is at least 42, and that C^i, E^i are all distinct, together they establish this case.



Case 2(b)i

ii. (**Pathological Case:**) Here we assume that the first 15 variables of E^1 are all zero. This is the most subtle and difficult case. Going back to Equation 9, we note that in this case it must hold that $w_{63} = 1$ and for all $0 \leq i \leq 62, w_i = 0$. We call such w *pathological*.

Now consider Equation 10. We can have two cases here.

In the first case, assume that the first 15 variables of z are zero. In that case, it must hold that $z_{62} = 1$. (Plugging in $i = 16$ to 62 in Equation 10 will yield $z_j = 0$ for all $15 \leq j \leq 61$ since $w_i = 0$ for these values.) Also note that z_{63} is free. In this case, we also call z pathological. In fact this may continue along the diagonal i.e., $E^3, E^4, \cdots$ may be pathological. If that happens then it is easy to show that the first non-zero bits of E^3 will be its 61^{st} bit, that of E^4 will be 60^{th} bit and so on. Also each column will have a free variable in its 63^{rd} bit.

In the second case, we assume that not all of its first 15 variables are zero. We call such z's to be non-pathological.

We now sub-divide into many small cases depending primarily on the number of pathological columns (and thus on the number of free variables).

A. (**# Pathological Columns** ≤ 8) We break this case into two sub-cases. That each of these sub-cases holds has been verified using a computer program.

(I). 6^{th} **and earlier non-pathological columns are non-zero:** In this case, we verify that the combined weight of the pathological columns and the first three non-pathological columns to the right of the pathological columns is at least 40. This ensures that in this case the minimum weight is at least 82.

We mention that the search space has dimension

$$\#\text{ of Pathological vars} + \#\text{ of Non-Pathological Cols.} \times 16,$$

which is at most 40 in this case.

We next consider the case where the non-pathological columns are same as one of C^1, C^2 or C^3.

(II). **6^{th} or earlier non-pathological column is identically zero:** Firstly note that it suffices to check the case where the 6^{th} non-pathological column is identically zero (that is $E^3 = C^3$), since other cases do fall in this case. Now we consider the parity check equations induced on the pathological columns and the six non-pathological columns. Note that C^1 satisfies Equation 7 and that E^1 satisfies Equation 9. Also note that in between columns satisfy equations similar to Equations 8 and 10. These equations then set up a homogeneous system of linear equations whose nullity can be verified (by a computer program) to be at most 40.

Let the number of pathological columns be p and the number of non-pathological columns be n. Specifically then the nullity of the system can then be shown to be exactly (see Appendix A Claim 9) : $p + 64 \times n - 48 \times (n+1) = p + 16 \cdot n - 48$, which is at most 40 in this case. We do an exhaustive search over the null space to establish that the min-weight is at least 82.

B. ($8 <$ **# Pathological Columns** ≤ 16): We also break this case into two sub-cases. That each of these sub-cases holds has been verified using a computer program.

(I). **5^{th} and earlier non-pathological columns are non-zero:** In this case, we verify that the combined weight of the pathological columns and the first two non-pathological columns to the right of the pathological columns is at least 40. This ensures that in this case the minimum weight is at least 82.

Therefore the case that remains to be considered is the one where the non-pathological columns are same as one of C^2 or C^3 which leads us to the next case.

(II). **5^{th} or earlier non-pathological column is identically zero:** Firstly, note that it suffices to check the case when the 5^{th} non-pathological column is identically zero (that is $E^2 = C^3$), since other cases do fall in this case. As in the 2^{nd} subcase of the previous case (i.e., Case 2(b)(ii)(A)(II)), we verify that the min-weight is at least 82.

C. ($16 <$ **Pathological Columns** ≤ 28**):** First of all, notice that 28 columns is enough, since by our assumption there is at least one zero column and three non-pathological column (i.e., C^1, C^2, C^3). Now, we also break this case into two sub-cases. That each of these sub-cases holds has been verified using a computer program.

(I). **4^{th} and earlier non-pathological columns are non-zero:** In this case, we verify that the combined weight of the pathological columns and the first non-pathological column to the right of the pathological columns is at least 40. This ensures that in this case the minimum weight is at least 82.

Therefore the case that remains to be considered is the one where the 1^{st} non-pathological column is the same as C^3.

(II). **4^{th} non-pathological column is identically zero:** As in the 2^{nd} sub-case of the previous case (or Case 2(b)(ii)(A)(II)), we verify that the min-weight is at least 82.

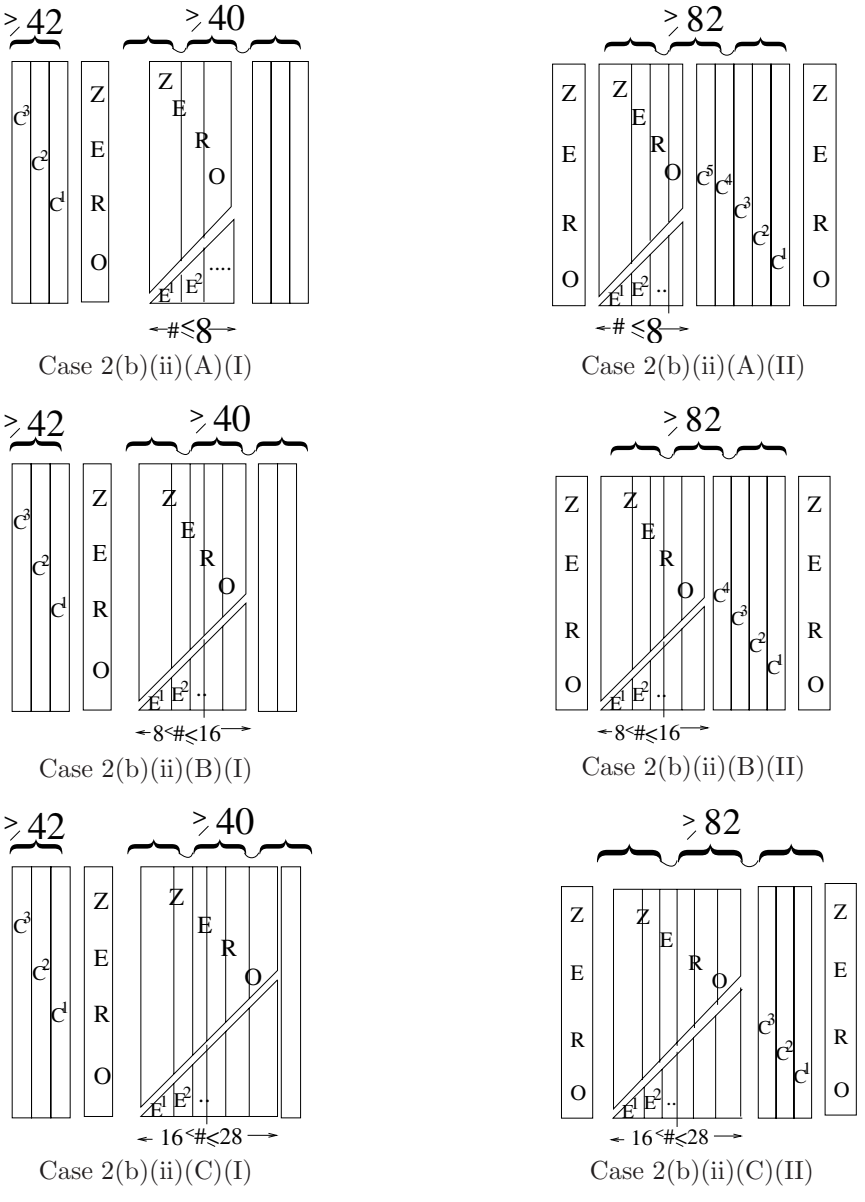


Fig. 1. Illustrations of various cases in the proof of Theorem 2

We remark that the minimum weight of this code can at most be 82 and therefore our result is tight (see Appendix B). Extending our approach, we can further prove the following theorem whose proof has been deferred to the full version.

Theorem 5. *The code $\mathcal{C}64$, as defined by Equation (6), has minimum weight at least 75 (and at least 52) in its last 60 words (and in its last 48 words, respectively).*

We remark that a simple variants of the above technique can be used to give a lower bound on the minimum weight of SHA-1 (of course, there are much fewer cases to consider here). Specifically we have the following theorem whose proof has been deferred to the full version of this paper (also see [6]).

Theorem 6. *SHA-1 message expansion code has minimum weight 25 in the last 60 words.*

5 Conclusion

In this paper we have shown how lower bounds on minimum weight of quasi-cyclic linear codes of dimension $m \times n$ given by parity equations of the form

$$W_i = \sum_{t=1}^{i} a_{it} W_{i-t} + \left(\left(\sum_{t=1}^{i} b_{it} W_{i-t} \right) <<< 1 \right) \qquad \text{for } i \geq n,$$

can be obtained by reducing the problem to the minimum weight of significantly smaller dimensional codes. Note that this equation is more general than Equation (5), and Equation (2) is of this form rather than the simpler Equation (5). In some cases, we obtain the exact minimum weight, including the example codes we considered. An obvious generalization is to consider three or more column mixing (the equation above has only two column mixing), which could lead to codes with even better minimum distance.

A common paradigm for designing hash functions, including MD5[12], SHA-0, SHA-1 and SHA-2[16] is the following: the 512-bit message is first expanded into N words, and then the N words are used as step keys (sometimes known as round keys) in N steps of a (non-linear) block cipher invoked on an initial vector. The output of the block cipher is the output of the compression function. As pointed out in the Introduction, one of the key ingredients of the recent differential attacks on MD5, SHA-0, and SHA-1 has been their poor message expansion (in terms of minimum weight) into the N words. We propose SHA1-IME which is SHA-1 with the original message expansion (see [15]) substituted by our improved message expansion as given in Equation 2. A preliminary evaluation has shown that this proposed compression function has at most a 5% overhead in speed over SHA-1 in a software implementation, and at most a 10% overhead in gate count in a high performance hardware implementation. However, on the positive side this proposed compression function resists all presently known attacks against SHA-1. Thus, we consider our novel technique to be an important advance in the design of collision-resistant hash functions.

References

1. Biham, E., Chen, R.: Near collisions of SHA-0. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, Springer, Heidelberg (2004)
2. Biham, E., Chen, R.: New results on SHA-0 and SHA-1. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, Springer, Heidelberg (2004)
3. Chabaud, F., Joux, A.: Differential collisions in SHA-0. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, Springer, Heidelberg (1998)
4. Chepyzhov, V.V.: New lower bounds for minimum distance of linear quasi-cyclic and almost linear cyclic codes. Problems of information Transmission 28(1) (1992)
5. Dumer, I., Micciancio, D., Sudan, M.: Hardness of approximating the minimum distance of a linear code. IEEE Transaction on Information Theory 49(1) (2003)
6. Jutla, C.S., Patthak, A.C.: A Matching Lower Bound on the Minimum Weight of SHA-1 Expansion Code. Cryptology ePrint Archive, Report 2005/266 (2005), `http://eprint.iacr.org/`
7. Kasami, T., Lin, S., Peterson, W.W.: New Generalization of the Reed-Muller Codes Part I: Primitive Codes. IEEE Transactions on Information Theory IT-14(2), 189–199 (1968)
8. Lally, K.: Quasicyclic codes of index ℓ over $\mathbb{F}_q$ Viewed as $\mathbb{F}_q[x]$-submodules of $\mathbb{F}_{q^l}[x]/\langle x^m - 1\rangle$. In: Fossorier, M.P.C., Høholdt, T., Poli, A. (eds.) Applied Algebra, Algebraic Algorithms and Error-Correcting Codes. LNCS, vol. 2643, Springer, Heidelberg (2003)
9. Ling, S., Solé, P.: Structure of quasi-clcyic codes III: Generator theory. In: IEEE Transaction on Information Theory (2005)
10. Matusiewicz, K., Pieprzyk, J.: Finding good differential patterns for attacks on SHA-1. In: International Workshop on Coding and Cryptography (2005)
11. Rijmen, V., Oswald, E.: Update on SHA-1. In: Menezes, A.J. (ed.) CT-RSA 2005. LNCS, vol. 3376, Springer, Heidelberg (2005)
12. Rivest, R.: RFC1321: The MD5 message-digest algorithm. In: Internet Activities Board (1992)
13. Townsend, R.L., Weldon, E.J.: Self-orthogonal quasi-cyclic codes. IEEE Transaction on Information Theory (1967)
14. United States Department of Commerce, National Institute of Standards and Technology, Federal Information Processing Standard Publication #180. Secure Hash Standard (1993)
15. United States Department of Commerce, National Institute of Standards and Technology, Federal Information Processing Standard Publication #180-1 (addendum to [14]). Secure Hash Standard (1995)
16. United States Department of Commerce, National Institute of Standards and Technology, Federal Information Processing Standard Publication #180-2. Secure Hash Standard (August 2002)
17. van Lint, J.H.: Introduction to Coding Theory. Springer, Heidelberg (1998)
18. Vardy, A.: The intractability of computing the minimum distance of a code. IEEE Transaction on Information Theory 43(6) (1997)
19. Wang, X., Yao, A., Yao, F.: New collision search for SHA-1. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, Springer, Heidelberg (2005)
20. Wang, X., Yu, H., Yin, Y.L.: Efficient collision search attacks in SHA-0. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, Springer, Heidelberg (2005)
21. Wang, X., Yu, H., Yin, Y.L.: Finding collisions in the full SHA-1. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, Springer, Heidelberg (2005)

22. Wang, X.Y.: The collision attack on SHA-0. In Chinese (1997)
23. Wang, X.Y.: The Improved collision attack on SHA-0. In Chinese (1997), `http://www.infosec.edu.cn/`
24. Zierler, N.: On a variation of the first-order reed-muller codes. In: M.I.T. Lincoln Lab., Group Report, 34-80, Lexington, Mass (October 1958)

A Rank Proofs

Claim 7. *If C^0 is zero, and C^1 is non-zero, then C^2 is non-zero.*

Proof. Assume otherwise i.e., that C^2 is zero. Consider the 48×64 dimensional parity check matrices (essentially Equations (7) and (9)) over $\mathbb{F}_2$.

$$\begin{pmatrix} 1010000010000100100000 \cdots & 0000000000000000 \\ 0101000001000010010000 \cdots & 0000000000000000 \\ \ddots & \cdots & \ddots \\ 00000000000000000000 \cdots & 0101000001000010 01 \end{pmatrix}$$

H_1

$$\begin{pmatrix} 01000000000000011000000 \cdots & 00000000000000000000 \\ 00100000000000001100000 \cdots & 00000000000000000000 \\ 00010000000000000110000 \cdots & 00000000000000000000 \\ 00001000000000000011000 \cdots & 00000000000000000000 \\ 10000100000000000001100 \cdots & 00000000000000000000 \\ 01000010000000000000110 \cdots & 00000000000000000000 \\ \ddots & \cdots & \ddots \\ 00000000000000000000 \cdots & 10000100000000000110 \end{pmatrix}$$

H_2

Then we need to show that $H = \begin{pmatrix} H_1 \\ H_2 \end{pmatrix}$ has full rank. For that it is enough to show that there are 64 linearly independent rows. We consider the 48 rows of H_1 and 16 additional rows, namely 5^{th} through 20^{th} rows of H_2. We reduce the problem to showing that a certain equation over polynomial ring $\mathbb{F}_2[x]$ does not have solutions in a restricted set of polynomials. We associate with the vector $c = \langle c_0, \cdots, c_{63} \rangle$ in $\mathbb{F}_2^{64}$ the polynomial $c(x) = \sum_{i=0}^{63} c_i x^i$ in $\mathbb{F}_2[x]$. Then the following polynomials can be associated with the 1^{st} and 5^{th} rows of matrix H_1 and H_2, respectively:

$$p(x) \stackrel{def}{=} x^{16} + x^{13} + x^8 + x^2 + 1, \text{ and } r(x) \stackrel{def}{=} x^{19} + x^{18} + x^5 + 1.$$

Further note that the i^{th} (note $1 \leq i \leq 48$) row of H_1 then gets associated with $x^{i-1}p(s)$. Similarly the j^{th} (note we restrict ourselves to $5 \leq j \leq 20$) row of H_2 then gets associated with $x^{j-5}r(s)$. Therefore, observe that if the 80 rows that

we are considering were dependent then we would have a non-zero solution of the following polynomial equation : $p(x)\alpha(x) + \beta(x)r(x) = 0$, with additional constraints that degree(α) ≤ 47 and degree(β) ≤ 15. However, it is easy to check that $p(x)$ is irreducible, therefore if such a equation holds then it must be the case that $p(x)$ divides $r(x)$. However, it is easy to check that $p(x)$ does not divide $r(x)$, thus leading to a contradiction.
Therefore H has full rank. ∎

We now strengthen the claim slightly.

Claim 8. *If C^0 is zero, and C^1 is non-zero, then both C^2, C^3 are non-zero.*

Proof. Consider the following polynomials :

$$p(x) \stackrel{def}{=} x^{16} + x^{13} + x^8 + x^2 + 1,$$

$$q(x) \stackrel{def}{=} x^{15} + x^{14} + x, \text{ and } r(x) \stackrel{def}{=} x^{19} + x^{18} + x^5 + 1 = x^4 \cdot q(x) + 1.$$

Let H_1 and H_2 be as above. First of all note that H_2 has full rank. (This is clear from the matrix. Otherwise, note that we would have an identity: $q(x) \cdot a(x) + r(x) \cdot b(x) = 0$, with degree($a$) ≤ 3 and degree(b) ≤ 43. Since degree($q \cdot a$) $<$ degree(r), this cannot happen.) Now we will show that the rank of the matrix

$$\begin{pmatrix} H_2 & 0 \\ H_1 & H_2 \\ 0 & H_1 \end{pmatrix}$$

is at least 128. Since H_1 has full rank, observe that $\begin{pmatrix} H_1 & H_2 \\ 0 & H_1 \end{pmatrix}$ has rank at least 96. So consider the following 92 independent rows from the above matrix, namely 5^{th} row onwards. We also argue that another additional 5^{th} through 40^{th} rows of the top H_2 are also independent. If not, then they would satisfy the following polynomial equations

$$\left.\begin{aligned} \alpha(x)p(x) + \beta(x)r(x) = 0 \quad (11) \\ x^4\beta(x)p(x) + \gamma(x)r(x) = 0 \quad (12) \end{aligned}\right| \quad \begin{aligned} &\text{with restrictions} \\ &\text{degree}(\alpha) \leq 47, \\ &\text{degree}(\beta) \leq 43, \text{ and} \\ &\text{degree}(\gamma) \leq 35. \end{aligned}$$

Since $p(x)$ is an irreducible polynomial, and $p(x) \nmid r(x)$, observe from Equation (11) that $p(x)|\beta(x)$. Hence, set $\beta(x) = \mu(x)p(x)$. Substituting in Equation (12) we get

$$x^4 p(x)^2 \mu(x) + \gamma(x)r(x) = 0.$$

Since $p(x)$ is irreducible, and $p(x) \nmid r(x)$, and $x \nmid r(x)$, it must hold that $x^4 p(x)^2 | \gamma(x)$. But that is impossible, since degree(γ) $\leq 35 < 36 =$degree($x^4 p(x)^2$). ∎

Recall that we used E^0 to denote a column that is zero everywhere. Also, recall that the columns left to E^0 are denoted E^1, E^2 and so on. In the following claim, we will assume $3 \leq n$.

Claim 9. *Let $E^1, E^2, \cdots, E^p$ be p pathological columns. Also, let $E^{p+1}, E^{p+2}, \cdots, E^{p+n}$ be n non-pathological columns. Further assume that $E^{p+n+1} = C^0$ is everywhere zero. If the nullity of the parity check equations resulting from these columns for $p = 0$ is $16 \cdot n - 48$, then the nullity of the parity check equations resulting from these columns for any $p \leq 28$ is $p + 16 \cdot n - 48$.*

Proof. Let $N_{i,j}, (1 \leq i \leq n, 0 \leq j \leq 63)$ denote the entries in the non-pathological columns. Also let $P_{i,j}, (1 \leq i \leq p,$ for each $i,\ 64 - i \geq j \leq 63)$ be the pathological variables. We will denote $N_i = \langle N_{i,0}, \cdots, N_{i,63} \rangle$ and $P_i = \langle P_{i,64-i}, \cdots, P_{i,63} \rangle$. Let $H_{1|i}$ denote the matrix H_1 restricted to the last i columns. (Note that only the last i rows will be non-zero.) Also let $H_{2|i}$ denote the matrix H_2 restricted to the last i columns. (Note that only the last $i-1$ rows will be non-zero.) Note that $\langle P_1, \cdots, P_p, N_1, \cdots, N_n \rangle$ must belong to the null space of the following matrix:

$$\mathcal{H} = \begin{pmatrix} H_{1|1} & H_{2|2} & & & & & & & & \\ & H_{1|2} & H_{2|3} & & & & & & & \\ & & \ddots & \ddots & & & & & & \\ & & & & H_{1|p-1} & H_{2|p} & & & & \\ & & & & & H_{1|p} & H_2 & & & \\ & & & & & & H_1 & H_2 & & \\ & & & & & & \ddots & \ddots & & \\ & & & & & & & & H_1 & H_2 \\ & & & & & & & & & H_1 \end{pmatrix}$$

Note that when we restrict H_1 or H_2 to the last few columns, the top rows in that restricted entries may become zero. We remove such rows if the entire row in the above matrix $\mathcal{H}$ becomes everywhere zero. Note that with this modification, the following sub-matrix is already in the echelon form:

$$\mathcal{H}_1 = \left.\begin{pmatrix} H_{1|1} & H_{2|2} & & & \\ & H_{1|2} & H_{2|3} & & \\ & & \ddots & \ddots & \\ & & & H_{1|p-1} & H_{2|p} \end{pmatrix}\right\} (p-1) \text{ blocks}$$

(Observe that first block corresponding to $(H_{1|1}\ H_{2|2})$ reduces to (1 10), and that corresponding to $(H_{1|2}\ H_{2|3})$ reduces to $\begin{pmatrix} 10 & 100 \\ 01 & 110 \end{pmatrix}$.)

Furthermore, since by assumption the following sub-matrix has full rank:

$$\mathcal{H}_2 = \left.\begin{pmatrix} H_2 & & & \\ H_1 & H_2 & & \\ \ddots & \ddots & & \\ & & H_1 & H_2 \\ & & & H_1 \end{pmatrix}\right\} (n+1) \text{ blocks}$$

the matrix $\mathcal{H}$ has full rank. Note here that in the top $48-p$ rows, $H_{1|p}$ is entirely zero. However these rows in $\mathcal{H}$ are independent since $\mathcal{H}_2$ has full rank. In the

remaining rows $H_{1|p}$ is in echelon form and hence independent. Note that the number of rows i.e., number of constraints is:

$$48 \times (n+1) + \sum_{i=1}^{p-1} i = 48(n+1) + \frac{p(p-1)}{2}.$$

Also, note the number of variables i.e., columns is

$$64 \times n + \sum_{i=1}^{p} i = 64 \cdot n + \frac{p(p+1)}{2}.$$

Thus the nullity of the system is

$$64 \cdot n + \frac{p(p+1)}{2} - \left(48(n+1) + \frac{p(p-1)}{2}\right) = p + 16 \cdot n - 48.$$

This completes the proof. ■

B Examples

We cite below an example where over 7 columns an average of 3 does not hold. Below we only give 8 columns and the columns are placed horizontally. Note that the 8 columns yield 29, whereas the first 7 columns yield only 14.

```
00000000000000000000000000000000000000000000000000000000000001000000
00000000000000000000000000000000000000000000000000000000000000110110
00000000000000000000000000000000000000000000000000000000000000010100
00000000000000000000000000000000000000000000000000000000000000001110
00000000000000000000000000000000000000000000000000000000000000000100
00000000000000000000000000000000000000000000000000000000000000000011
00000000000000000000000000000000000000000000000000000000000000000001
10001010100000000010010000100000100001001011000000100010000010000
```

Below is a codeword in the code defined by Equation (6) with optimal minimum weight. We found the following codeword while searching for Case 2(b)(ii)(A)(II). Below we only give eight columns that includes six non-zero and two zero columns. The rests are all zero columns. Below the columns are placed horizontally.

```
0000000000000000 0000000000000000 0000000000000000 0000000000000000
0011110010011110 1000000001101001 1101001001010110 0000110010010000
1011000101000100 0010111101001000 1011100010101100 1101000000101111
1010101000111011 0010100100110010 1000000101001000 0110011000000000
0000000000000000 0000000000000000 0000000000000000 0000000000000100
0000000000000000 0000000000000000 0000000000000000 0000000000000011
0000000000000000 0000000000000000 0000000000000000 0000000000000001
0000000000000000 0000000000000000 0000000000000000 0000000000000000
```