Optimal Binary Switch Codes with Small Query Size

Zhiying Wang
Center for Science of Information
Stanford University
Stanford, CA 94305, USA
zhiyingw@stanford.edu

Han Mao Kiah
School of Physical and Mathematical Sciences
Nanyang Technological University
21 Nanyang Link, 637371, Singapore
hmkiah@ntu.edu.sg

Yuval Cassuto
Department of Electrical Engineering
Technion-Israel Institute of Technology
Haifa, 3200003, Israel
ycassuto@ee.technion.ac.il

***Abstract*—In this paper, we study a construction of binary switch codes. A switch code is a code such that a multi-set request of information symbols can be simultaneously recovered from disjoint sets of codeword symbols. Our construction is optimal in the sense that it has the smallest codeword length given its average encoding degree, which is logarithmic in the code dimension. Moreover, the number of queries needed to recover any information symbol in the request is at most 2. As a result, our construction is the first family of switch codes with low encoding and decoding complexity.**

I. Introduction

We study memory sub-systems of network switches that are used to store packets between arrival from input ports and departure to output ports. In particular, we consider a coding scheme for network switches such that they scale with information exchange speed and network size. Suppose that a switch has k input ports and R output ports. In order to parallelize the process of writing and reading in the memory for different ports, multiple memory banks are used in a switch. However, there is a contention problem for reading if the requested packets from several output ports are written in the same memory bank. To solve this contention problem, Wang *et. al* proposed the use of switch codes [7].

Consider a toy example with $k = R = 2$ and two memory banks. Suppose at each time slot every port transmits one packet with fixed size, and every memory bank can support one packet write and one packet read. At time slot t, $t \in T$, denote the information of two input packets by A_t, B_t, which are written in the first and second bank, respectively. At some time slot, suppose A_t, A_s are requested from the two output ports, respectively, $t \neq s \in T$. Since every bank can only support one read, an undesired delay is incurred. An alternative would be a coding scheme that uses three memory banks. At every time $t \in T$, $A_t, B_t, A_t + B_t$ are written in the three banks, respectively, where "+" means bit-wise XOR of the two packets. Then any request of size two can be solved by reading at most one packet from every bank. Therefore any request can be solved within a single time slot. In particular, A_t can be solved by reading $\{A_t\}$, while A_s can be solved by reading $\{B_s, A_s + B_s\}$. Observe that the two sets of banks used are disjoint.

More generally, an (n, k, R) *switch code* is a code of length n and dimension k, such that for every multi-set request of size R, there is a solution such that the sets of codeword symbols for recovering the requested symbols are disjoint. The maximum number of codeword symbols used to recover an information symbol, denoted by r, is called *query size*. Typically, we require k, R to be of the same order.

A switch code is a specialization of the primitive multi-set batch codes by Ishai *et. al* [1] to the case where the number of input information symbols k is close to the request size R. This case captures well the regime in switching applications where the input and output data rates are instantaneously similar, whereas the outcome of most of the previous work on batch codes has been codes with R much smaller than k, and cannot support steady-state switching.

A related notion is locally decodable codes (e.g., [8]), which satisfy the smoothness property: for any information symbol, all the r-queries used to decode that symbol cover the n codeword symbols uniformly. This property ensures that if the same information symbol is requested multiple times, there exist disjoint solutions. However, it is not tight enough to achieve non-asymptotic optimality. Moreover, probabilistic decoding is considered for such codes, while for switch codes deterministic decoding is required.

In addition, switch codes are related to locally repairable codes with multiple availabilities (e.g. [3], [4], [5], [6]), but known codes for the latter model do not give request lengths R close to k.

Previously, only switch codes with encoding degree two (two information symbols are combined to get each codeword symbol) and codes that can solve burst requests (only one requested symbol is requested more than once) is known [7]. It is an open problem to construct switch codes solving arbitrary requests and having degree larger than two.

In this paper, we construct the first family of switch codes with optimal length given its average encoding degree, which is $O(\log k)$. Furthermore, the code is binary, and the query size is $r = 2$. As a result, it has low complexity from a practical point of view. The construction is based on simplex code and the concatenation of multiple codewords. For $R = k$, the codeword length is $n = O(k^2/\log k)$, whereas the previous constructions [7] have codeword length $n = O(k^2)$.

The paper is organized as follows. In Section II, we formally define the switch codes and introduce necessary notations. Section III gives the construction and the proof of its correctness and optimality. Comparison with previous results are shown in Section IV. Finally, we conclude in Section V.

II. Definitions and Notations

In the rest of the paper, we use $[i]$ to denote the set $\{1, 2, \ldots, i\}$ for $i \in \mathbb{N}^+$, and $[i, j]$ to denote the set $\{i, i + 1, \ldots, j\}$ for $i \leq j \in \mathbb{Z}$. We use boldface to represent a vector. For a vector $\mathbf{x}$, its length is represented by $|\mathbf{x}|$. For a set S, its

Y. Cassuto was supported by a joint ISF-UGC project and by the Israel Ministry of Science and Technology. This work was completed when H. M. Kiah was at University of Illinois, Urbana-Champagin. Z. Wang was supported by the Center for Science of Information.

cardinality is denoted by $|S|$. For a vector $\mathbf{x} = (x_0, \ldots, x_{n-1})$ and a subset $S = \{s_1, \ldots, s_{|S|}\} \subseteq [0, n-1]$, where $0 \leq s_1 < \cdots < s_{|S|} \leq n-1$, we denote by $\mathbf{x}_S = (x_{s_1}, \ldots, x_{s_{|S|}})$ the vector of elements with coordinates in S. We use $\log$ to denote logarithm of base 2.

In formally, a *switch code* over the alphabet $\mathcal{X}$ encodes an information vector $\mathbf{u} = (u_0, \ldots, u_{k-1})$ of length k into a codeword vector $\mathbf{x} = (x_0, \ldots, x_{n-1})$ of length n. For any request of information indices $(i_1, \ldots, i_R)$, $i_j \in [0, k-1]$, there exists disjoint sets $S_1, \ldots, S_R \subseteq [0, n-1]$, such that u_{i_j} can be recovered from the codeword symbols indexed by S_j, namely, $\mathbf{x}_{S_j}$, for any $j \in [R]$. More formally, a switch code can be defined as follows.

Definition 1. An (n, k, R) switch code on the alphabet $\mathcal{X}$ consists of an encoding function

$$\varphi : \mathcal{X}^k \to \mathcal{X}^n,$$

a decoding set function

$$\xi : [0, k-1]^R \to \mathcal{S},$$

where $\mathcal{S} = \{(S_1, S_2, \ldots, S_R) : S_i \subseteq [0, n-1], S_i \cap S_j = \emptyset, \text{ for all } 1 \leq i \neq j \leq R\}$ is the collection of all vectors of R disjoint sets, and decoding recovery functions

$$\psi_S^i : \mathcal{X}^{|S|} \to \mathcal{X}$$

for $S \subseteq [0, n-1]$, and $i \in [0, k-1]$. The functions satisfy the following: for all $\mathbf{u} \in \mathcal{X}^k$ and $(i_1, \ldots, i_R) \in [0, n-1]^R$, if $\varphi(\mathbf{u}) = \mathbf{x}$ and $\xi(i_1, \ldots, i_R) = (S_1, \ldots, S_R)$, then for every $j \in [R]$,

$$\psi_{S_j}^{i_j}(\mathbf{x}_{S_j}) = u_j.$$

We call k the *input size* or the *code dimension*, and R the *request length*. If a codeword symbol is systematic, namely, if it equals to an information symbol, then it is called a *singleton*. For a linear code, if a codeword symbol is a linear combination of d information symbols, then its *encoding degree* is d.

For some code and a request, if there exist disjoint sets to recover the requested symbols, then we say there is a *solution* to a request, or the request is *solvable*. If the j-th information symbol is requested l_j times, $j \in [0, k-1]$, then we write the *request (vector)* as $\mathbf{L} = (l_0, \ldots, l_{k-1})$. Notice that the total multiplicity is the request length, or $\sum_{j=0}^{k-1} l_j = R$. Moreover, to show a code solves arbitrary requests, it is sufficient to show that it solves all request vectors. The set of codeword symbols in S_j is called *helpers* or a *helper set* for the j-th requested information symbol.

If r is the smallest integer such that for any request of length R and any information index $j \in [0, k-1]$, we have

$$|S_j| \leq r,$$

then we say the *query size* is r.

III. Code Constructions

In this section, we first construct binary switch codes from simplex codes, and get codes with length $n = (2R-1)k/(1+\log R)$, dimension k, and query size 2, where $\log R$ and $k/(1+\log R)$ are integers. We then show that this construction solves arbitrary requests of length R. Lastly, we prove the optimality of our construction.

An (N, K) *simplex code* is constructed as follows. For every non-empty subset of $[0, K-1]$, form a bit in the codeword that is the XOR of the elements in the subset. Hence, a simplex code of dimension K, $K \geq 1$, has codeword length $N = 2^K - 1$.

Construction 1. Fix k and R such that $\log R$ and $\frac{k}{1+\log R}$ are integers. Partition the k information bits into $\frac{k}{1+\log R}$ groups of size $K = 1 + \log R$. Use an $(N = 2R-1, K = 1+\log R)$ simplex code on every group, and then concatenate the $\frac{k}{1+\log R}$ codewords. The constructed code therefore has a codeword length

$$n = \frac{(2R-1)k}{1+\log R}.$$

Suppose the generator matrix of the $(N = 2R-1, K = 1+\log R)$ simplex code is given by (I_K, G), where I_K is the $K \times K$ identity matrix, and G is an $K \times (N-K)$ matrix. Then the code given by Construction 1 has generator matrix

$$\begin{pmatrix} I_K & 0 & \cdots & 0 & G & 0 & \cdots & 0 \\ 0 & I_K & \cdots & 0 & 0 & G & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & I_K & 0 & 0 & \cdots & G \end{pmatrix}.$$

We next show that the above construction solves arbitrary R requests and has query size 2. To prove that, we first focus on the (N, K) simplex code, and show that it solves an arbitrary request of length $2^{K-1} = \frac{N+1}{2}$.

For example, the code $(A, B, A+B)$ described in the introduction is a simplex code with $K = 2$, and any request of length $2^{K-1} = 2$ can be solved.

In the following, we use the subset $S \subseteq [0, K-1]$ to represent the corresponding bit in the codeword of the simplex code. The information bits are the sets of size one, namely, $\{i\}$ for any $i \in [0, K-1]$. By abuse of notation, we write "+" to denote the XOR of two bits, or equivalently, the symmetric difference of two sets. It is clear that any codeword bit S can be recovered from the XOR of the two bits $(S + S')$ and S', for arbitrary S'. Therefore, the simplex code has query size of 2 for any information or parity bit. In particular, any information bit can be computed from two codeword bits.

Consider a graph where every non-empty subset S is a vertex, and every edge (S, S') corresponds to a solution to an information bit, namely, $|S + S'| = 1$. Also add to this graph K dummy vertices corresponding to the empty set, denoted by ϕ_i, $i \in [0, K-1]$, along with K edges $(\{i\}, \phi_i)$. See Figure 1 for an example. First, notice that this graph represents all possible solutions of information bits with query size no more than 2. Next, notice that this is a bipartite graph where the partition of the vertices is determined by the parity of $|S|$. The even partition is of size $2^{K-1} + K - 1$ (including K copies of the empty set), while the odd partition is of size 2^{K-1}. A disjoint solution for some request vector can be viewed as a matching in the graph, and apparently the size of the matching, or the request length, cannot exceed 2^{K-1}. We show later that this upper bound on the request length, 2^{K-1}, is also achievable.

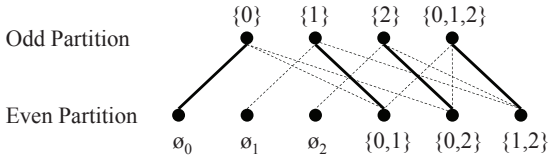

Fig. 1. A bipartite graph on $K = 3$ input bits. Every edge corresponds to a possible helper pair. The set of solid-line edges is a solution to the request $\mathbf{L} = (4, 0, 0)$, or four times the input bit $\{0\}$.

Definition 2. A request vector $\mathbf{L}$ on K input bits is said to be *short* if its length satisfies

$$|\mathbf{L}| \leq f(K) \stackrel{\triangle}{=} \frac{K}{K+1} 2^{K-1}.$$

Definition 3. A solution to a request vector is said to be *type I* if singletons are not used in the solution, and the query size is 2.

For example, let $K = 4$, and consider a short request $\mathbf{L} = (1, 1, 1, 1)$ of length no more than $f(K) = 32/5$. Namely, every information bit is requested once. It can be solved by the helper pairs $(\{0, 1, 2\}, \{1, 2\})$, $(\{1, 2, 3\}, \{2, 3\})$, $(\{2, 3, 0\}, \{3, 0\})$, $(\{3, 0, 1\}, \{0, 1\})$, which is a type I solution, since no singletons are used.

For a request vector $\mathbf{L} = (l_0, \ldots, l_{K-1})$, one potential solution is to first solve $\mathbf{L}' = (l_0 - 1, \ldots, l_{K-1} - 1)$ without singletons, and then $(1, 1, \ldots, 1)$ with singletons only. More precisely, we later show that if $\mathbf{L}'$ is a short request for a large enough K, then it has a type I solution (Lemma 7), hence $\mathbf{L}$ is solvable.

Definition 4. Let I be a set of integers of size K. Consider all subsets of I. They are also the codeword bits on K inputs labeled by I, together with a dummy empty set. For any $i \in I$, define a partition of the subsets into two parts:

$$A_i = \{S \subseteq I : i \in S\},$$
$$\overline{A_i} = \{S \subseteq I : i \notin S\}.$$

Also define a 1-1 mapping between them:

$$\delta_i : A_i \to \overline{A_i},$$

such that for any $S \in A_i$, we have

$$\delta_i(S) = S \backslash \{i\}.$$

Figure 2 shows an example of the partitions on $K = 3$ inputs. The above partition forms a recursive structure of the codeword bits. Apparently, any solution to the information bit $\{i\}$ with query size 2 must be a pair

$$(S, \delta_i(S)), \tag{1}$$

for some $S \in A_i$, where the ordering of the pair is not considered. Besides, if the information bit $\{j\}$, $j \neq i$, can be solved by a pair (R, S), with $R + S = \{j\}$ and $R, S \in \overline{A_i}$, then it can also be solved by the pair

$$(\delta_i^{-1}(R), \delta_i^{-1}(S)) = (R \cup \{i\}, S \cup \{i\}).$$

Before proving the solvability of the simplex code, we outline the proof steps with an example.

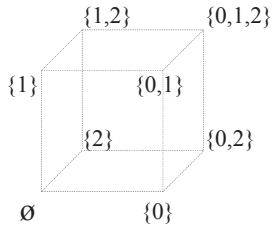

Fig. 2. Partitions on $K = 3$ input bits labeled $I = \{0, 1, 2\}$. Every two parallel faces form a partition. For example, the face on the right is $A_0 = \{\{0\}, \{0, 1\}, \{0, 2\}, \{0, 1, 2\}\}$ containing element "0", and the face on the left is $\overline{A_0}$. One can see that any edge connecting the left and the right faces corresponds to a solution to the bit $\{0\}$. Moreover, since the pair $(\emptyset, \{1\})$ solves the bit $\{1\}$, we have that the pair $(\delta_0^{-1}(\emptyset), \delta_0^{-1}(\{1\})) = (\{0\}, \{0, 1\})$ also solves the same bit.

Example 5. Consider a request

$$\mathbf{L} = (62, 59, 58, 55, 51, 50, 49, 45, 42, 41)$$

for $K = 10$. Crucial to the proof is that the entries in the request vector are in non-increasing order. We write $\mathbf{L}$ as $\mathbf{L} = \mathbf{L}_1 + 2\mathbf{L}_2 + \mathbf{L}_3$ with

$$\begin{aligned}
\mathbf{L}_1 &= (62, \quad 0, \quad 0, \quad 0, \quad 0, \quad 0, \quad 0, \quad 0, \quad 0, \quad 0),\\
\mathbf{L}_2 &= (0, \quad 29, \quad 29, \quad 27, \quad 25, \quad 25, \quad 24, \quad 22, \quad 21, \quad 20),\\
\mathbf{L}_3 &= (0, \quad 1, \quad 0, \quad 1, \quad 1, \quad 0, \quad 1, \quad 1, \quad 0, \quad 1).
\end{aligned}$$

Lemma 7 below shows that $\mathbf{L}_2$ can be solved by type I solution on $K - 1 = 9$ inputs, which we explain in the next paragraph. This solution can then be duplicated in the two partitions of A_0 and $\overline{A_0}$, respectively. Thus we can solve $2\mathbf{L}_2$, while we use the singletons to solve $\mathbf{L}_3$. Finally, we demonstrate in Theorem 8 that there are sufficiently many pairs as in (1) still available for the information bit $\{0\}$ in the request vector $\mathbf{L}_1$.

To show that $\mathbf{L}_2$ has a type I solution, we view $\mathbf{L}_2$ as a short request on $K - 1$ inputs and write $\mathbf{L}_2 = (29, 29, 27, 25, 25, 24, 22, 21, 20)$. Again this is in non-increasing order. Now consider

$$\mathbf{L}_2' = (29, 30, 28, 26, 26, 24, 22, 22, 20),$$

and write $\mathbf{L}_2' = \mathbf{L}_4 + 2\mathbf{L}_5$, where

$$\begin{aligned}
\mathbf{L}_4 &= (29, \quad 0, \quad 0, \quad 0, \quad 0, \quad 0, \quad 0 \quad 0, \quad 0),\\
\mathbf{L}_5 &= (0, \quad 15, \quad 14, \quad 13, \quad 13, \quad 12, \quad 11, \quad 11, \quad 10).
\end{aligned}$$

Notice that if $\mathbf{L}_2'$ is type-I solvable, so is $\mathbf{L}_2$. We show $\mathbf{L}_5$ is type-I solvable using the base case in Lemma 7 on $K - 2 = 8$ inputs. In general if $K - 2 > 8$, $\mathbf{L}_5$ can be shown to be a short request on $K - 2$ inputs, and thus have a type I solution using induction hypothesis in Lemma 7. Similar to the argument in the previous paragraph, consider all codeword bits on the $K - 1$ inputs labeled $I = \{1, 2, \ldots, K - 1\}$. The solution to $\mathbf{L}_5$ can be duplicated in the two partitions A_1 and $\overline{A_1}$, respectively. Finally, we solve $\mathbf{L}_4$ using the remaining pairs as in (1), which is proved in the last part of Lemma 7.

The following is a lemma on a small input size, and forms the base case of our proof.

Lemma 6. Consider a simplex code with K input bits.

(i) There is a type I solution to any request of length $2^{K-1} - K$, for all $K \leq 8$.

(ii) There is a type I solution to any short request for $K = 8$.
(iii) There is a solution of query size 2 to any request of length 2^{K-1}, for all $K \leq 8$.

Proof: (i) This is proved by computer search.

(ii) This is obvious since a short request also has length no more than $f(K) \leq 2^{K-1} - K$ when $K = 8$. We can simply use a subset of pairs of helpers from (i).

(iii) For any request vector $\mathbf{L} = (l_0, \ldots, l_{K-1})$, without loss of generality, assume $l_0 \geq l_1 \geq \cdots \geq l_{K-1}$. If $l_{K-1} \geq 1$, let $\mathbf{L}' = (l_0 - 1, \ldots, l_{K-1} - 1)$, and $\mathbf{L}'' = (1, \ldots, 1)$. Notice that $\mathbf{L}'$ can be solved by (i) and $\mathbf{L}''$ can be solved by singletons, and the two sets of helpers for $\mathbf{L}'$ and $\mathbf{L}''$ are disjoint. Also notice that $\mathbf{L} = \mathbf{L}' + \mathbf{L}''$. Therefore, $\mathbf{L}$ is solvable. When $l_{K-1} = 0$, this is proven by computer search. ∎

Lemma 7. For the (N, K) simplex code, $K \geq 8$, there is a type I solution to any short request.

Proof: We prove by induction. When $K = 8$, this is proven in Lemma 6 (ii). Let $K \geq 9$ and consider a short request $\mathbf{L} = (l_0, \ldots, l_{K-1})$. Without loss of generality, assume $l_0 \geq \cdots \geq l_{K-1}$. Partition all codeword bits on K inputs labeled $\{0, \ldots, K-1\}$ into two parts: $A_0, \overline{A_0}$.

Let $\mathbf{L}' = (0, \lceil \frac{l_1}{2} \rceil, \ldots, \lceil \frac{l_{K-1}}{2} \rceil)$, $\mathbf{L}'' = (l_0, 0, \ldots, 0)$. We next show that $2\mathbf{L}' + \mathbf{L}''$ is type-I solvable. Noticing that $\mathbf{L}$ is short and l_0 is the largest component, the length satisfies

$$\begin{aligned} |\mathbf{L}'| &\leq \frac{1}{2}\left(\sum_{i=1}^{K-1} l_i + K - 1\right) \\ &\leq \frac{1}{2}\left(\frac{K-1}{K} f(K) + K - 1\right) \leq f(K-1) \end{aligned}$$

for $K \geq 9$. So we can view $\mathbf{L}'$ as a short request on $K-1$ inputs labeled $[K-1]$. By induction hypothesis, $\mathbf{L}'$ has a type I solution. For every helper pair (S, R) in this solution on $K-1$ inputs, $|S + R| = 1$, we generate two pairs on K inputs labeled $[0, K-1]$, that solve the same information bit: the first is

$$(S \cup \{0\}, R \cup \{0\}), \tag{2}$$

and both helpers belong to A_0; the second is (S, R) and both helpers belong to $\overline{A_0}$. Since S, R are not singletons by induction hypothesis, we know the generated helpers are not singletons, either. Moreover, all the generated helpers are disjoint. So we have a type I solution to the request $2\mathbf{L}'$.

Let B, C be the set of helpers for $2\mathbf{L}'$ generated in $A_0, \overline{A_0}$, respectively, which are both of size $2|\mathbf{L}'|$. Notice that the function δ_0 defines a 1-1 mapping from B to C, and accordingly from $A_0 \backslash B$ to $\overline{A_0} \backslash C$. In other words, if we pick any unused element $S \in A_0 \backslash B$, then we have $S \backslash \{0\} \in \overline{A_0} \backslash C$, and they can be the helper pair for $\{0\}$. As a result, there are $|A_0 \backslash B| = 2^{K-1} - 2|\mathbf{L}'|$ remaining ways to solve $\{0\}$, and K of them involve singletons. Hence the number of helper sets not using singletons for $\{0\}$ satisfies for $K \geq 9$,

$$\begin{aligned} |A_0 \backslash B| - K &= 2^{K-1} - 2|\mathbf{L}'| - K \\ &\geq 2^{K-1} - (|\mathbf{L}| - l_0 + K - 1) - K \\ &\geq 2^{K-1} - (f(K) - l_0 + K - 1) - K \\ &\geq l_0. \end{aligned} \tag{3}$$

So we have a type I solution for $2\mathbf{L}' + \mathbf{L}''$, and hence for $\mathbf{L}$. ∎

Theorem 8. Let $\mathbf{L}$ be a request of length 2^{K-1} for the (N, K) simplex code, then it is solvable with query size 2.

Proof: When $K \leq 8$, this is true by Lemma 6 (iii). Assume $K \geq 9$. Assume without loss of generality $\mathbf{L} = (l_0, \ldots, l_{K-1})$ with $l_0 \geq \cdots \geq l_{K-1}$. Then rewrite $\mathbf{L}$ as $\mathbf{L} = (l_0, 0, \ldots, 0) + 2\mathbf{L}_2 + \mathbf{L}_3$, where $\mathbf{L}_2 = (0, \lfloor \frac{l_1}{2} \rfloor, \ldots, \lfloor \frac{l_{K-1}}{2} \rfloor)$, and $\mathbf{L}_3 = (0, l_1 \bmod 2, \ldots, l_{K-1} \bmod 2)$. It is easy to see that $\mathbf{L}_2$ is a short request on $K-1$ inputs, namely $|\mathbf{L}_2| \leq f(K-1)$, and has a type I solution by Lemma 7 for $K \geq 9$. Then, with singletons, we can solve $\mathbf{L}_3$. Finally, with the same argument as in (3), we have l_0 pairs to solve $\{0\}$ (using possibly some singletons):

$$|A_0 \backslash B| - |\mathbf{L}_3| = 2^{K-1} - 2|\mathbf{L}_2| - |\mathbf{L}_3| = l_0, \tag{4}$$

where A_0 is the partition of subsets of $[0, K-1]$ containing "0", and B is the set of helpers for $\mathbf{L}_2$ belonging to A_0, defined similar to (2). ∎

Remark: For any request of length no more than 2^{K-1}, the above proofs provide us with a recursive algorithm to find a solution. The recursion ends at the base case of 8 input bits, and the complexity of the algorithm is linear in K.

Corollary 9. Construction 1 can solve any request of length R.

Proof: Set $K = 1 + \log R$ in the simplex code. Consider any request of length R. If it only contains information bits of the same group of size K, then the statement holds by Theorem 8. If it contains information bits from different groups, then for every group we get a request of length less than $R = 2^{K-1}$, and can solve it by Theorem 8 considering codeword bits from that group. ∎

Next, we show the optimality of our construction. Consider a linear switch code. Suppose a codeword symbol indexed i is a linear combination of d_i information symbols, then we say the *degree* of the codeword symbol is d_i. Similar to [7], we have the following lower bound on the codeword length.

Lemma 10. An (n, k, R) switch code with average degree d for the codeword symbols satisfies

$$n \geq kR/d.$$

Proof: Notice that every information symbol has to appear in R codeword symbols, resulting in a total of kR appearances. Therefore, the sum of degrees satisfies $\sum_{i=0}^{n-1} d_i \geq kR$. And the lemma holds for the average degree. ∎

Proposition 11. Construction 1 is optimal in terms of codeword length with respect to its average degree.

Proof: For Construction 1, the average degree is also the average degree of the (N, K) simplex code:

$$d = \frac{\sum_{i=1}^{K} i\binom{K}{i}}{2^K - 1} = \frac{K2^{K-1}}{2^K - 1} = \frac{(1 + \log R)R}{2R - 1}.$$

Given this value of d, the upper bound from Lemma 10 is given by

$$n \geq \frac{kR}{(1 + \log R)R/(2R-1)} = \frac{(2R-1)k}{1 + \log R},$$

establishing the optimality of Construction 1. ∎

IV. Comparison with Batch Codes

In this section, we consider switch codes, or multi-set primitive batch codes, proposed in [1]. Specifically, we focus on the two classes of *binary* multi-set primitive batch codes: subcube codes and subset codes.

A. Subcube codes

Fix parameters l and t to be positive integers. Let G_l be the $l \times (l+1)$ matrix given by $(I_l, \mathbf{1})$, where I_l is the $l \times l$ identity matrix, and $\mathbf{1}$ is the all-ones column vector. In other words, G_l is the generator matrix of a code with a single parity bit.

A *subcube code* with parameters l and t is then the linear code generated by the matrix $G(l,t) \triangleq G_l^{\otimes t}$, where $A^{\otimes t}$ denotes the Kronecker product of t A's.

Hence, we check easily that $n = (l+1)^t$, $k = l^t$. In [6] this code is viewed as a $((l+1)^t, l^t, t+1)$ switch code with query size l. If we require a large request length, Ishai *et. al* [1] demonstrated that the subcube code can solve requests of length 2^t. Therefore, in the subsequent discussions the subcube code is viewed as a $((l+1)^t, l^t, 2^t)$ switch code.

Due to the structure of $G(l,t)$, we observe that the average encoding degree is given by $d = (2l/(l+1))^t = kR/n$. Hence, by Lemma 10, the subcube code is optimal in terms of codeword length with respect to its average degree.

However, unlike Construction 1, the *query size of the subcube code is* k. In particular, consider 2^t requests for any given information bit. Then there exists a helper set of size $l^t = n^{\log l / \log(l+1)}$. Hence, the query size of the subcube code is $l^t = k$ and approaches n as l grows.

B. Subset codes

Fix parameters w and l to be positive integers with $w < \ell$. Consider subsets of $[l]$ and let the information bit x_T correspond to a subset T of size w. Then a *subset code* with parameters l and w is a code whose codewords are indexed by subsets of $[l]$ with size at most w. Every codeword bit x_S is given by $\sum_{S \subseteq T,\ |T|=w} x_T$. Hence, we have the codeword length $n = \sum_{j=0}^{w} \binom{\ell}{j}$ and the dimension $k = \binom{\ell}{w}$.

Fix a subset T. In order to solve a request for the information bit x_T, Ishai *et. al* considered the following collection of subsets. For each $T' \subseteq T$, consider the line

$$L(T,T') = \{X : X \cap T = T', |X| \leq w\}.$$

Ishai *et. al* then demonstrated that the set of codeword bits $\{x_S : S \in L(T,T')\}$ on the line $L(T,T')$ solves the request for x_T. Furthermore, given requests $T_1, T_2, \ldots, T_R$ with $R = 2^{cw}$, where $0 < c < 1$ is some constant, they provided a randomized algorithm to choose $T_1', T_2', \ldots, T_R'$ and showed that the lines $L(T_j, T_j')$ are pairwise disjoint with positive probability. However, no deterministic solutions with guaranteed decodability is known.

We demonstrate below that when $l = w+1$, the subset code can in fact solve requests of length $R = 2^w$. Furthermore, the helper sets can be *deterministically* computed. We make use of the following observation.

Lemma 12. A subset code with parameters w and $l = w+1$ is a $(2^w - 1, w)$ simplex code.

Proof: By definition, the codeword bits of subset code are indexed by the subsets of $[l]$ with size at most w. Fix T to be a subset of size at most w and let $\overline{S} = [l] \setminus S$. We change the indices of the subset code such that the bit x_S corresponds to the bit $y_{\overline{S}}$. Since $l = w+1$, the new index set is given by the nonempty subsets of $[l]$.

Furthermore, if $|T| = w$, the information bit x_T now corresponds to $y_{\overline{T}} = y_{\{i\}}$ for some $i \in [l]$. For any subset S with $|S| \leq w$,

$$y_{\overline{S}} = x_S = \sum_{S \subseteq T,\ |T|=w} x_T = \sum_{S \subseteq ([l] \setminus \{i\})} x_{[l]\setminus\{i\}} = \sum_{i \in \overline{S}} y_{\{i\}}.$$

This then coincides with the definition of a $(2^w - 1, w)$ simplex code. ∎

Applying Theorem 8, we obtain the fact that a subset code with parameters w and $l = w+1$ is a $(2^w - 1, w, 2^{w-1})$ switch code. As pointed by a reviewer, the same fact was observed by Ishai *et al.* [1]. Specifically, Ishai *et al.* observed that a subset code is a subcode of some binary Reed Muller code. Indeed a simplex code is a shortened subcode of the first order Reed Muller code [2, Fig 1.13].

V. Concluding Remarks

In this paper, we construct a family of binary switch codes with optimal codeword length given the average degree. Compared to probabilistic decoding methods studied for locally decodable codes, such as Reed-Muller codes, our construction provides a deterministic solution to an arbitrary request of maximum length, as well as a small field size and a small query size. Moreover, our algorithm in finding a solution to any request has complexity $O(k)$, for k input bits.

Interesting open problems on this topic include tight lower bounds on the codeword length, and constructions with an arbitrary encoding degree, d.

References

[1] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai, "Batch codes and their applications," in *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*. ACM, 2004, pp. 262–271.

[2] F. J. MacWilliams and N. J. A. Sloane, *The theory of error correcting codes*. Elsevier, 1977, vol. 16.

[3] F. Oggier and A. Datta, "Self-repairing homomorphic codes for distributed storage systems," in *INFOCOM, 2011 Proceedings IEEE*. IEEE, 2011, pp. 1215–1223.

[4] L. Pamies-Juarez, H. D. L. Hollmann, and F. E. Oggier, "Locally repairable codes with multiple repair alternatives," *CoRR*, vol. abs/1302.5518, 2013.

[5] A. Rawat, D. Papailiopoulos, A. Dimakis, and S. Vishwanath, "Locality and availability in distributed storage," in *Information Theory (ISIT), 2014 IEEE International Symposium on*, June 2014, pp. 681–685.

[6] I. Tamo and A. Barg, "Bounds on locally recoverable codes with multiple recovering sets," in *Information Theory (ISIT), 2014 IEEE International Symposium on*, June 2014, pp. 691–695.

[7] Z. Wang, O. Shaked, Y. Cassuto, and J. Bruck, "Codes for network switches," in *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*. IEEE, 2013, pp. 1057–1061.

[8] S. Yekhanin, "Locally decodable codes," *Computer Science–Theory and Applications*, pp. 289–290, 2011.