Analysis of One Step Majority Logic Decoders Constructed From Faulty Gates

Shashi Kiran Chilappagari
Dept. of Electrical and Computer Eng.
University of Arizona
Tucson, AZ 85719, USA
Email: shashic@ece.arizona.edu

Miloš Ivković
Department of Mathematics
University of Arizona
Tucson, AZ 85719, USA
Email: milos@math.arizona.edu

Bane Vasić
Dept. of ECE and Dept. of Mathematics
University of Arizona
Tucson, AZ 85719, USA
Email: vasic@ece.arizona.edu

***Abstract*— In this paper we propose an analytical method to evaluate the performance of one step majority logic decoders constructed from faulty gates. We analyze the decoder under the assumption that the gates fail independently. We calculate the average bit error probability of such a decoder and apply the method to the special case of projective geometry codes. The method, however, applies to any regular low-density parity-check code of girth at least six but the calculations are much simpler for the projective geometry codes. We present results for the bit error rate performance of four codes from projective planes.**

I. Introduction

During the past four decades reducing transistor size has been a driving force for microchip technology and has lead to very fast, power efficient, and highly integrated silicon chips. However, decreases in voltage and transistor capacitance have already started affecting transistor reliability and it is widely believed that transistor failures (temporal and permanent) will become one of the main technological obstacles as this trend continues.

In the traditional memory and communication models it is assumed that the designer of a system has full control over error correction encoders and decoders, and that errors/noise only affect the memory or a communication link (channel). However, digital logic in the encoder and decoder is constructed using non-ideal components as well, and errors can affect the computation performed in these two devices.

Research in the area of systems constructed from faulty gates has progressed along two directions: computation and storage. Study of computation by circuits with faulty gates was started by von Neumann [1]. He showed that, under certain conditions, increased gate redundancy can lead to increased reliability of a circuit. He introduced a technique called *multiplexing* which consists of two phases. In the first phase, the basic function of the circuit is performed in several copies of the circuit. In the second phase, referred to as restoration phase, errors introduced in the first phase are corrected by random coupling of the copies, and the final outputs are considered to be "'1" if majority of the considered copies are "1" and "0" otherwise.

Note that multiplexing can be seen as error control coding. For example, Triple Modular Redundancy is nothing but a repetition code with code rate 1/3. Restoration organ of the multiplexing technique is in fact an error correcting decoder. Similar observation was made by Spielman [2]; for a tutorial on the subject see [3].

All the current fault tolerant constructions are based on redundancy of the gates, and it was shown that, in general, computation by faulty gates with non-zero computational capacity is not possible (see [4], [5], [6]). *Computational capacity* is the reciprocal of the minimum redundancy for which the probability of error can be made arbitrarily small [7]. The best known result for a general model of computation is due to Speilman [8] who coupled ideas of von Neumann with Reed-Solomon codes.

On the other hand, the study of storage circuits made of unreliable components lead to much more optimistic results. Taylor in [7] proposed construction of a storage circuit with non-zero computation capacity (or rather storage capacity) based on Low-Density Parity-Check (LDPC) codes (this construction was further studied by Kuznetsov [9]).

It is important to stress that non-zero capacity is possible only until the moment a codeword is transformed by some unencodable action, such as extracting message bits. For example, even if a systematic code is used in the construction of Taylor and Kuznetsov, at the end of the correcting procedure information will still be confined within a multiple of codeword copies with possibly some errors, i.e. discrepancy in content. So, the final step, extracting user information, must be done by a majority logic gate (or some equivalent circuit), which is by assumption faulty, resulting in probability of error grater than or equal to the probability of failure of the used majority logic gate. Therefore, it is reasonable to use reliable gates for the final step of decoding (see also [10] and [8]). Such gates can be realized, for example, by using larger transistor sizes. We employ such "perfect" or "golden" majority logic gates in our decoder architecture.

In this paper we present an analytical method to evaluate the performance of decoders constructed from faulty gates. We consider one step majority logic decoders which are amenable for easy analysis and moreover require very little circuitry. These decoders do not suffer from the problem of dependence

of the messages passed, which in general is a major obstacle for analyzing iterative decoders. The method we propose can be applied to any LDPC code with girth at least six. However, we will restrict our attention to the codes constructed from projective planes which form a large class of majority logic decodable codes and whose combinatorial properties are well understood.

The rest of the paper is organized as follows: In Section II we discuss some preliminaries about majority logic decoding and projective geometry codes. In Section III we describe our decoder architecture and present a method to calculate the average bit error probability of a single step majority decoder. In Section IV we provide a combinatorial interpretation of the method. In Section V we present numerical results and we conclude in Section VI.

II. Preliminaries

Let $\mathcal{C}_s$ represent the LDPC code constructed based on the family of the two-dimensional projective geometries PG(2,2^s). This code is characterized by $n_v = 2^{2s} + 2^s + 1$ variable nodes, $n_c = 2^{2s} + 2^s + 1$ check nodes, minimum distance $d_{min} = 2^s + 2$ and column weight $\gamma = 2^s + 1$. It is well known that such a code can correct up to $\lfloor \gamma/2 \rfloor$ errors when decoded using a one step majority logic decoder. Note that for all the codes considered in this paper, γ is odd.

Consider transmission over a binary symmetric channel (BSC) characterized by transition probability α. For a variable node v (check node c), let $E(v)$ ($E(c)$) denote the edges incident on v (c). Also, let $r(v)$ denote the received value of node v.

A one step majority logic decoder can be implemented in one of the following two ways:

A. Syndrome Based Decoder

The syndrome based decoder performs the following steps to decode a received vector.

- Each variable node v sends $r(v)$ along each edge in $E(v)$.
- Each check node c sends $m(e)$ along each edge in $E(c)$ where

$$m(e) = (\sum_{e' \in E(c)} m(e'))\ mod\ 2$$

- At each v an estimate of the error $e(v)$ is made

$$e(v) \quad = \quad \begin{cases} 1, & |\{e' \in E(v) : m(e') = 1\}| \geq \lceil \gamma/2 \rceil \\ 0, & \text{otherwise} \end{cases}$$

- Each node is decoded as $(r(v) + e(v)) mod 2$

Such a decoder needs a γ-input XOR gate at each of the check nodes and a γ-input majority logic gate and a two-input XOR gate at each variable node. As is clear from the decoder implementation, a check node tells its neighboring variable node if the check is satisfied or not. If majority of checks connected to a variable node are unsatisfied then the variable node is flipped. Also, a check node sends the same message to all its neighboring variable nodes. This implementation is attractive from a practical point of view.

B. Message Passing Decoder

A message passing decoder differs slightly from the syndrome based decoder. The following steps are involved in this decoder.

- Each variable node v sends $r(v)$ along each edge in $E(v)$.
- Each check node c sends $m(e)$ along each edge in $E(c)$ where

$$m(e) = (\sum_{e' \in E(c) \setminus \{e\}} m(e'))\ mod\ 2$$

- At each v an estimate of the value of v is made

$$\hat{v} \quad = \quad \begin{cases} 1, & |\{e' \in E(v) : m(e') = 1\}| \geq \lceil \gamma/2 \rceil \\ 0, & |\{e' \in E(v) : m(e') = 0\}| \geq \lceil \gamma/2 \rceil \\ r(v), & \text{otherwise} \end{cases}$$

This decoder needs γ of $(\gamma - 1)$-input XOR gates at each check node and a γ-input majority logic gate at each variable node. In this implementation the check node makes an estimate of the value of a variable node based on the values of other variable nodes. The value of the variable node is itself not used in its estimation. The final decision is made on the basis of majority of the estimates.

It should be noted that the both the decoders are equivalent for one step majority logic decoding when constructed from perfect gates. However, when built from faulty gates their performance can be quite different. In this paper we consider the message passing decoder. This is mainly because of the fact that the decoder architecture in this case agrees well with our assumption of "golden" majority logic decoders. The syndrome based decoder needs "golden" XOR gates also, as the final step in the decoding is that of bit flipping. Otherwise, the bit error probability will be bounded below by the failure probability of that 2-input XOR gate. Moreover, a gate failure in the message passing like decoder affects only one variable node whereas the effect is more drastic in a syndrome based decoder.

III. Analysis of the Decoder

The decoder is assumed to be built of faulty XOR gates which are used on the check node end and "golden" majority logic gates which are used at the variable node end. It is assumed that the each of the $(\gamma - 1)$-input XOR gate fails independently with a probability ϵ. By gate failure we mean that the correct output of the gate is flipped with a fixed probability ϵ. We use the term perfect check to denote the checks in a perfect decoder and faulty check to denote the checks in a faulty decoder. Typically ϵ is of order $10^{-6} - 10^{-}3$.

Let α denote the transition probability of the BSC. Let the received bits be $(r(v_1), r(v_2), \ldots, r(v_n))$ where n is length of the code.

In the analysis which follows we consider a particular variable node and calculate the probability that it is miscorrected. Since a check node sends an estimate of the variable node, we say that a message sent by check node is incorrect if the estimate made by the check node is wrong. In a perfect decoder the check node sends an incorrect message when odd number of errors are present in the estimation. In a faulty decoder the check may send a wrong message either due to wrong estimate or due to a correct estimate followed by gate failure. We will make use of these facts in the analysis. The theorem below is useful in finding error configurations of received bits and checks that are most likely to lead to a miscorrection of the considered bit. Corollaries that follow lead to the formula for the probability that a bit is miscorrected during decoding.

Theorem 1: Let the variable node v be connected to checks $c_1, c_2, \ldots, c_\gamma$. Let f_i be number of variable nodes in error (excluding v) connected to check c_i and let

$$\begin{aligned} G(e_1,\ldots,e_\gamma,x_1,\ldots,x_\gamma) &= \prod_{i=1}^{\gamma}\Big[(1-\epsilon)M_c^{(i)}+\epsilon M_w^{(i)} \\ &+ x_i \quad \Big((1-\epsilon)M_w^{(i)}+\epsilon M_c^{(i)}\Big)\Big] \end{aligned}$$

where

$$\begin{aligned} M_w^{(i)} &= \sum_{k=1}^{\frac{\gamma-1}{2}} \binom{\gamma-1}{2k-1}\alpha^{2k-1}(1-\alpha)^{\gamma-2k}e_i^{2k-1} \\ M_c^{(i)} &= \sum_{k=0}^{\frac{\gamma-1}{2}} \binom{\gamma-1}{2k}\alpha^{2k}(1-\alpha)^{\gamma-1-2k}e_i^{2k} \end{aligned}$$

The coefficient of $e_1^{f_1}e_2^{f_2}\ldots e_\gamma^{f_\gamma}x_{j_1}x_{j_2}\ldots x_{j_k}$, $k\le\gamma$ is the probability that the node v will receive incorrect messages from the checks $c_{j_1}, c_{j_2}\ldots c_{j_k}$ if f_i nodes (excluding v) connected to the check c_i are in error ($i=1,\ldots,\gamma$).

To prove this theorem the following lemma is used.

Lemma 1: Probability that a faulty check will send a correct message is

$$\begin{aligned} (1-\epsilon) &\quad \sum_{k=0}^{\frac{\gamma-1}{2}} \binom{\gamma-1}{2k}\alpha^{2k}(1-\alpha)^{\gamma-1-2k} \\ +\ \epsilon &\quad \sum_{k=1}^{\frac{\gamma-1}{2}} \binom{\gamma-1}{2k-1}\alpha^{2k-1}(1-\alpha)^{\gamma-2k} \end{aligned}$$

and probability that this check will send an incorrect message is

$$\begin{aligned} \epsilon &\quad \sum_{k=0}^{\frac{\gamma-1}{2}} \binom{\gamma-1}{2k}\alpha^{2k}(1-\alpha)^{\gamma-1-2k} \\ +\ (1-\epsilon) &\quad \sum_{k=1}^{\frac{\gamma-1}{2}} \binom{\gamma-1}{2k-1}\alpha^{2k-1}(1-\alpha)^{\gamma-2k} \end{aligned}$$

Proof of Lemma 1: Notice that a *perfect* check will send a correct message if number of variable nodes connected to the check (excluding v) that are in error is even. Considering that the check itself can produce an error with probability ϵ statement of the lemma follows.

Proof of Theorem 1: The degree of the variables e_i ($i=1,\ldots,\gamma$) in G is the number of erroneous received bits that are used in calculation of the check c_i, that is, this degree is f_i. In G, the expression for the probability that check c_i will send an incorrect message (from Lemma 1) is multiplied with x_i, completing the theorem.

Theorem 1 gives the probabilities of specific error configurations of received bits and checks. Assuming BSC, all bit configurations with i errors (excluding v) are equally likely ($\alpha^i(1-\alpha)^{(n-1)-i}$). Also, within our model, any configuration of k check failures is equally likely ($\epsilon^k(1-\epsilon)^{\gamma-k}$). Therefore, we can talk about *average probability* that i channel and errors and k check failures will lead to a miscorrection. For this purpose we define:

$$\overline{G}(e,x)=\sum_{i=0}^{n-1}\sum_{k=0}^{\gamma}b_{i,k}e^ix^k \stackrel{\text{def}}{=} G(\underbrace{e,e,\ldots,e}_{\gamma},\underbrace{x,x,\ldots,x}_{\gamma}) \tag{1}$$

It should be noted that for codes with girth at least six, all the variable nodes connected to the check node neighbors of a variable node v are distinct. The PG codes have the extra property that the variable node neighbors of all the check node neighbors of v contain all the other variable nodes exactly once. Hence we can say that the parameter i is in fact the total number of errors in locations other than v. For all other codes a similar analysis can be carried out with slight modifications for the parameter i.

Corollary 1: The average probability that the node v receives k incorrect messages if i errors have occurred in nodes other than v is equal to $b_{i,k}$ (as given in Equation 1).

Proof of Corollary 1: Note that the coefficient of e^ix^k in $\overline{G}(e,x)$ is the sum of probabilities for all the received bit configurations with i erroneous bits and all the possible ways k out of γ checks can fail.

Corollary 2: The average probability that node v receives k incorrect messages, given the channel introduced i errors is

$$P_v^{i,k}(\alpha,\epsilon) = (1-\alpha)b_{i,k}+(\alpha)b_{i-1,k}$$

Proof of Corollary 2: $(1-\alpha)$ is the probability that the node v is not in error initially and $b_{i,k}$ is the probability that v receives k incorrect messages when there are i errors are outside v. Similarly α is the probability that v is in error initially and $b_{i-1,k}$ is the probability that v receives k incorrect messages when there are $i-1$ errors are outside v. The result now follows from total probability theorem. Note that $P_v^{0,k}=(1-\alpha)b_{0,k}$ and $P_v^{n,k}=\alpha b_{n-1,k}$.

Corollary 3: The Probability that node v is miscorrected is

$$\begin{aligned} P_v(\alpha,\epsilon) &= \sum_{i=0}^{n}\sum_{k=\lfloor\frac{\gamma}{2}\rfloor+1}^{\gamma}P_v^{i,k} \\ &= \sum_{i=0}^{n-1}\sum_{k=\lfloor\frac{\gamma}{2}\rfloor+1}^{\gamma}b_{i,k} \end{aligned}$$

Proof of Corollary 3: A node will be miscorrected if it receives incorrect messages from more than half of the checks. Hence the second summation starts at $\lfloor \frac{\gamma}{2} \rfloor + 1$.

IV. Combinatorial Interpretation

The focus of this section is to provide a combinatorial interpretation of the values of $b_{i,k}$ for $\mathcal{C}_s$ for different values of s.

Let $p_\gamma(i)$ denote the number of partitions of i with at most γ parts and no part exceeding $\gamma - 1$. Let $\lambda_i^j = (\lambda_1, \lambda_2, \ldots, \lambda_l), l \leq \gamma, j \in \{1, \ldots, p_\gamma(i)\}$ denote a specific partition of i. Let $(1^{f_1^{i,j}} 2^{f_2^{i,j}} 3^{f_3^{i,j}})$ denote the frequency representation of a partition $\lambda_i^j = (\lambda_1, \lambda_2, \ldots, \lambda_l)$ where exactly $f_t^{i,j}$ of $\lambda_m, m \in 1, \ldots, l$ are equal to t. Any configuration of i errors in nodes other than a given node v can be associated with a λ_i^j for some j. The partition λ_i^j completely determines the number of incorrect messages sent to node v by a perfect decoder.

Lemma 2: The number of i error configurations corresponding to $\lambda_i^j = (\lambda_1, \lambda_2, \ldots, \lambda_l)$ is

$$N(\lambda_i^j) = \binom{\gamma}{l}\binom{\gamma-1}{\lambda_1}\binom{\gamma-1}{\lambda_2}\cdots\binom{\gamma-1}{\lambda_l}\text{Perms}(\lambda_i^j)$$

where Perms(λ_i^j) is the number of permutations on entries of λ_i^j

Note that $\binom{\gamma-1}{\lambda_1}\binom{\gamma-1}{\lambda_2}\cdots\binom{\gamma-1}{\lambda_l}$Perms($\lambda_i^j$) is the number of compositions (ordered partitions) corresponding to λ_i^j.

Lemma 3: The number of incorrect messages $I_p^{i,j}$ sent to a node v by a perfect decoder in a configuration corresponding to λ_i^j is

$$I_p^{i,j} = \sum_{t=1}^{2t-1 \leq \gamma} f_{2t-1}^{i,j}$$

Proof of Lemma 3: Note that in a perfect decoder, a check c sends an incorrect message to a neighboring variable v if odd number of errors have occurred in the variable nodes (excluding v) connected to c.

So corresponding to a partition λ_i^j, a node v receives $I_p^{i,j}$ incorrect messages and $C_p^{i,j} = \gamma - I_p^{i,j}$ correct messages.

Lemma 4: The probability that a node v receives k incorrect messages in a configuration corresponding to λ_i^j in a faulty decoder is

$$P(k, \lambda_i^j) = \sum_{u=0}^{k} P_w\{u, I_p^{i,j}\} P_c\{k-u, C_p^{i,j}\}$$

where

$$\begin{aligned} P_w(u, I_p^{i,j}) &= \binom{I_p^{i,j}}{u}(\epsilon)^{I_p^{i,j}-u}(1-\epsilon)^u \\ P_c(k-u, C_p^{i,j}) &= \binom{C_p^{i,j}}{k-u}(\epsilon)^{k-u}(1-\epsilon)^{C_p^{i,j}-(k-u)} \end{aligned}$$

Proof of Lemma 4: $P_w(u, I_p^{i,j})$ is the probability that u of the $I_p^{i,j}$ incorrect messages remain incorrect and $P_c(k-u, C_p^{i,j})$ is the probability that $k-u$ of the $C_p^{i,j}$ correct messages become incorrect. The product hence accounts for a total of k incorrect messages.

Theorem 2:

$$b_{i,k} = \alpha^i (1-\alpha)^{(n-1-i)} \sum_{j=1}^{p_\gamma(i)} N(\lambda_i^j) P(k, \lambda_i^j)$$

Proof of Theorem 2: Follows from Lemma 2, Lemma 3 and Lemma 4 and the fact that each i error configuration occurs with probability $\alpha^i(1-\alpha)^{(n-1-i)}$.

V. Numerical Results

The average bit error probabilities for different values of ϵ can be calculated and compared to the case of $\epsilon = 0$, i.e., a perfect decoder. However even for the PG code of length 73, the number of values of $b_{i,k}$ to be calculated is very large. Since we are generally interested in the performance of the decoder for low values of α we can consider only the first few values of i and neglect the higher values. For example, for the PG code of length 73 at values of $\alpha < 10^{-2}$ the probability of more than 9 errors is of the order 10^{-9}. So the bit error probability for this code can be estimated as

$$BER_f(\alpha, \epsilon) \approx \sum_{i=0}^{9}\sum_{k=5}^{9} P_v^{i,k}(\alpha, \epsilon)$$

A perfect decoder for the above code can correct up to 4 errors. So, the bit error for this decoder is given by

$$BER_p(\alpha) \approx \sum_{i=5}^{9}\sum_{k=5}^{9} P_v^{i,k}(\alpha, 0)$$

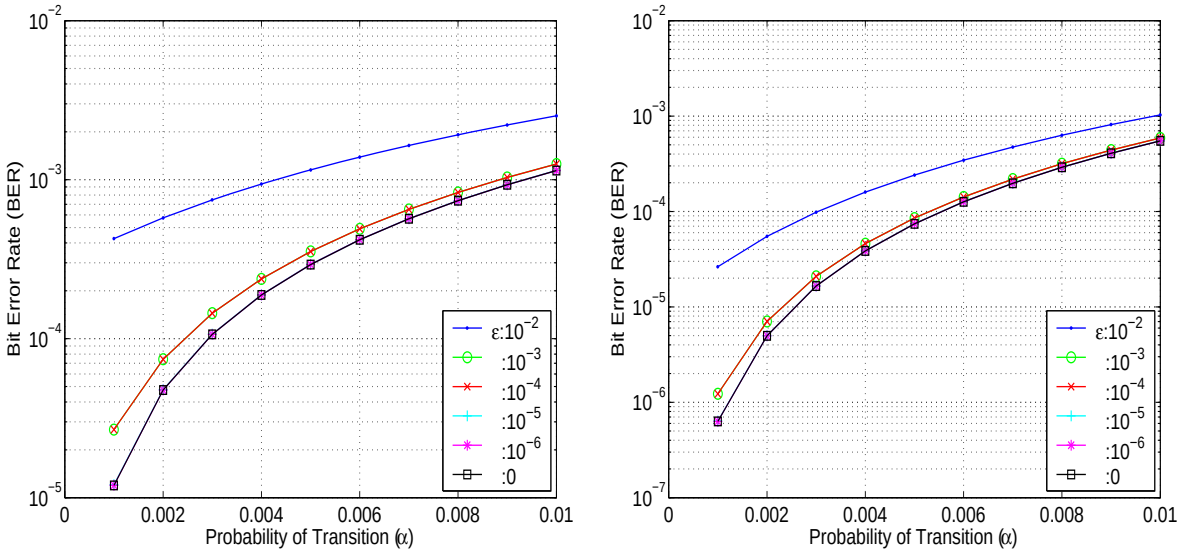
In general we calculate the BER using the formula

$$BER_f(\alpha, \epsilon) \approx \sum_{i=0}^{\gamma}\sum_{k=\lfloor \frac{\gamma}{2} \rfloor + 1}^{\gamma} P_v^{i,k}(\alpha, \epsilon)$$

The BERs for different PG codes are plotted for different values of ϵ in Figs. 1(a)-1(d). Notice that even for large probability of gate failures (10^{-3}), the performance loss is small.

VI. Conclusion

In this paper we have proposed an analytical method to compute the bit error probability of a faulty one step majority logic decoder. The method was applied to a class of PG codes and approximate BER performance was calculated. This method is a first step towards the analysis of iterative decoders constructed from faulty gates. The analysis presented in Section III can be applied to any regular LDPC code of girth at least six, and in principle it is possible to calculate the bit error probability after the first iteration. Future work involves extending the method to multi-step majority logic decoders, deriving tight bounds on frame error rate (FER) performance by analyzing dependence of node failures and investigating schemes to achieve non-zero computational capacity.



(a) PG(2,2) Code

(b) PG(2,4) Code

(c) PG(2,8) Code

(d) PG(2,16) Code

Fig. 1. Performance of Faulty Decoders for Different PG Codes

ACKNOWLEDGMENT

This work was supported by the NSF under Grant CCR-0208597.

REFERENCES

[1] J. Von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," in *Automata Studies*, C.E. Shannon and J. McCarty, eds., Princeton University Press, pp. 43-98, 1956.

[2] M. Sipser and D. Spielman, "Expander Codes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 1710-1723, Nov. 1996.

[3] P.Gács, "Lectures on Reliable Cellular Automata," http://citeseer.ist.psu.edu/20653.html

[4] R. L. Dobrushin and S. I. Ortyukov, "Lower bound for the redundancy of self-correcting arrangements of unreliable functional elements," *Prob. Inf. Trans.*, 13:59–65, Jul. 1977.

[5] P. Gács and A. Gál, "Lower bounds for the complexity of reliable Boolean circuits with noisy gates," *IEEE Trans. Information Theory*, vol. 40, pp. 579-583, Mar. 1994.

[6] N. Pippenger, G. D. Stamoulis and J. N. Tsitsiklis, "On a lower bound for the redundancy of reliable networks with noisy gates," *IEEE Trans. Inf. Theory* vol. 37, pp. 639-643, May 1991.

[7] M. Taylor, "Reliable Information Storage in Memories Designed from Unreliable Components," *Bell System Technical Journal* vol. 47, pp 2299-2337, Dec. 1968.

[8] Daniel Spielman, "Highly Fault-Tolerant Parallel Computation," *IEEE Conference on Foundations of Computer Science*, pp 154-163, 1996.

[9] A. Kuznetsov, "Information Storage in a Memory Assembled from Unreliable Components," *Problems of Information Transmission* vol. 9, pp. 254-264, Jul. 1973.

[10] C. N. Hadjicostis and G. C. Verghese, "Coding Approaches to Fault Tolerance in Linear Dynamic Systems," *IEEE Transactions on Information Theory*, vol. 51, no. 1, pp. 210-228, Jan. 2005.