Optimizing Burst Erasure Correction of LDPC Codes by Interleaving

Gokul Sridharan, Abishek Kumarasubramanian, Andrew Thangaraj and Srikrishna Bhashyam
Electrical Engineering Department
Indian Institute of Technology Madras, Chennai, India
Email:andrew@iitm.ac.in,skrishna@ee.iitm.ac.in

***Abstract*—The performance of iterative decoding of Low Density Parity Check (LDPC) codes over Binary Erasure Channels can be completely characterized by the study of stopping sets. Therefore, the burst erasure correction capability of a given LDPC code can be readily quantified by searching for stopping sets within consecutive bit nodes. In this work we study the optimal permutation of the bit nodes that will result in the maximum possible burst erasure correction capability for a given LDPC code. Noting that this is essentially a combinatorial optimization problem that is highly likely to be NP-hard, we adopt a simulated annealing based approach for finding the optimal permutation. We present bounds based on stopping sets that limit the burst erasure correction capability. As part of our results, we provide interleavers that greatly improve the burst erasure correction capability of protograph quasi-cyclic LDPC codes used in the WiMax standard.**

I. Introduction

Low Density Parity Check(LDPC) codes have been established through their success in various different scenarios as one of the most powerful and versatile codes available today. Different standards such as Digital Video Broadcasting and the IEEE 802.16e (WiMax) standards have already adopted LDPC codes. In applications such as magnetic recording, an important requirement is that the code used should be capable of correcting bursts of erasures (in addition to random errors) caused by media defects such as scratches. Correction of burst erasures has application in wireless communication systems limited by interference. In some OFDM-based cellular systems, a contiguous band of subcarriers might be adffected by interference from neighboring cells. One way of handling such interference is to declare the affected symbols as erasures. In such settings, a code with better burst erasure correction performance will perform better.

While the threshold phenomenon of LDPC codes has been exploited to design good degree distributions to protect against random errors, burst errors and erasures need special attention. In the case of finite length LDPC codes over the Binary Erasure Channel (BEC), by searching for stopping sets that occur within a sequence of consecutive bit nodes, one can determine the maximum length burst erasure that can be corrected by a given LDPC code. A simple method to improve the burst erasure correction capability of a given LDPC code is interleaving or a permutation of bit nodes. In this paper, we study the problem of finding the optimal permutation of bit nodes that will result in the maximum possible burst erasure correction capability for a given LDPC code. Realizing that this is essentially a combinatorial optimization problem similar to the traveling salesman problem, we provide a simulated annealing based approach in trying to maximize the *span* of the given LDPC code. The simulated annealing approach greatly reduces the computational requirements of the problem and provides optimal permutations in many cases.

Note that interleaving to improve burst erasure correction results in a permutation-equivalent code that retains all the random error-correcting properties of the original code. Therefore, the design process for LDPC codes does not need any alteration in order to imrove the burst erasure correction capability.

LDPC codes have been previously designed to correct erasure bursts. Concepts such as maximum resolvable burst length (similar to span) have been introduced, and deterministic and random approaches for design of LDPC codes for correcting bursts of erasures have been studied in [1], [2], [5], [3], [6], [12]. In [8], an algorithm to improve the span of an LDPC code has been presented.

Our novel approach to the problem is essentially combinatorial. This enables a direct application to the case of protograph quasi-cyclic LDPC codes constructed using a base matrix expanded by a circulant permutation matrix [11], [13]. The stopping sets in the expanded matrix correspond to those in the base matrix as shown in [7], [9], and maximizing the *span* of the base matrix ensures that the *span* of the expanded matrix is maximized too. From a hardware perspective, it is easier to implement an interleaver for the base matrix. Thus, we greatly reduce the complexity of the problem by running the algorithm on the base matrix. We observe that our method works better than the algorithm in [8] on dense base matrices of protograph codes in the IEEE 802.16e (WiMax) standard. Based on an enumeration of the stopping sets, we provide bounds for the maximum burst erasure correction capability under iterative decoding. This is useful for moderate length base matrices such as those in the WiMax standard, where enumeration of stopping sets is feasible.

The rest of the paper is organized as follows. In Section II, we introduce the notion of span of stopping sets and relate span to the burst erasure correction capability. Section III briefly talks about the protograph quasi-cyclic LDPC codes and their stopping sets. In Section IV we provide some bounds on the maximum burst erasure correction capability by exhaustive enumeration of the stopping sets of the base matrix. Section V sets up the optimization problem and discusses the simulated annealing algorithm. In Section VI, we present the results of the algorithm run on the codes in IEEE 802.16e (WIMax) standard and compare these results with the results obtained using the algorithm discussed in [8].

II. Burst Erasure Correction Using LDPC Codes

A. Stopping sets, Span

Erasure correction capability of LDPC codes under iterative message-passing over a Tanner graph of a parity-check matrix is governed by certain subsets of the bit nodes of the Tanner graph called stopping sets [4]. A set of bit nodes is said to be a stopping set if every check node connected to the set is connected at least twice. To quantify the burst erasure correction capability of LDPC codes under iterative decoding, we define the *span* of the stopping sets of a given Tanner graph of an LDPC code.

Definition 2.1 (Span): Let S be any non empty subset of the set $\{1, 2, \ldots, n\}$. Then the **span** of S, denoted by $Span(S)$, is equal to : $\max_{x,y \in S}(|x - y| + 1)$

Definition 2.2: Let Z be some non-empty collection of non-empty subsets of the set $\{1, 2, \ldots, n\}$. Span of Z, denoted by $Span(Z)$, is defined as : $\min_{S \in Z}(Span(S))$

Let us consider a length-n LDPC code C defined by a parity-check matrix H that determines the Tanner graph used for iterative decoding over a Binary Erasure Channel (BEC). Let $Span(H)$ denote the minimum length of a burst erasure that cannot be corrected by the iterative algorithm. Let $\mathcal{X}$ denote the collection of all non-empty stopping sets of H. Then, as a consequence of the relationship between stopping sets and burst erasure correction,

$$Span(H) = Span(\mathcal{X}).$$

III. Protograph Quasi-Cyclic LDPC Codes

Protograph quasi-cyclic LDPC codes are described using a base matrix H_b of dimensions $r \times s$ and an expansion factor m [14]. Let R be the $m \times m$ permutation matrix representing a circular right shift of a length-m vector. The entries of the base matrix H_b are chosen from the set $-1, 0, 1, 2, \ldots, m-1$. To obtain the $rm \times sm$ expaned binary parity-check matrix replace each -1 in the base matrix by the $m \times m$ all-zero matrix and replace each $j \in 0, 1, \ldots, m-1$ in the base matrix with R^j. These codes are referred to as cyclic lifts by some authors [7].

A. Stopping sets

Consider the binary base matrix H' obtained from H_b by replacing the negative entries in H_b by 0 and the non-negative entriesby 1. Let H denote the $rm \times sm$ expanded matrix. Every column of the expanded matrix can be indexed by an ordered pair (i, j), where $i \in \{1, 2, \ldots, s-1\}$ denotes a column of H_b and $j \in 0, 1, \ldots, m-1$ denotes a column in a shifted version of R. Therefore (i, j) refers to the column $(si + j)$ of H.

Lemma 3.1 (from [7]): Let S be a stopping set in the expanded matrix H. Define the set $S' = \{i, ((j+1) \mod m) : (i, j) \in S\}$. Then the set S' is also a stopping set of the expanded matrix H [7].

Theorem 3.2 (from [9]): Let S be a stopping set in the expanded matrix H. Let $T = \{i : (i, j) \in S\}$ denote a set of columns in the matrix H'. The set T is a stopping set of the linear binary code defined by H' [9].

B. Burst Erasure Correction

Using Theorem 3.2, every stopping set of the expanded matrix H can be identified with a stopping set in the binary base matrix H'. Notice that this is a many-to-one map. We use this relationship to establish the following result.

Theorem 3.3: Let $b = Span(H')$ be the span of the binary base matrix. For an expansion factor m, let $t = Span(H)$ be the span of the expanded matrix. then

$$(b-2)m + 2 \leq t \leq bm$$

Proof: Suppose S is a stopping set of the expanded matrix H with minimal span t, from Theorem 3.2, we see that S can be related to a stopping set of the binary base matrix H' which by hypothesis has a span of at least b. Thus the stopping set S must span over at least b consecutive $m \times m$ expansion matrices. The minimal and maximal possible span for such a stopping set are $(b-2)m + 2$ and bm, respectively. ■

From Theorem 3.3, we see that maximization of the burst erasure correction capability of the quasi-cyclic LDPC code defined by H can be approached by finding an interleaver to maximize the span of the smaller binary base matrix H'. Since H' typically has a small number of columns (24 in the case of WiMax LDPC codes) the computational requirements of the problem are reduced.

IV. Bounds on Burst Erasure Correction Capability

In this section we present some simple bounds on the burst erasure correction capability by using an enumeration of stopping sets. These bounds are applicable especially for quasi-cyclic LDPC codes with moderate-length base matrices.

To begin with, note that the burst erasure correction capability is bounded by the number of check nodes of the $r \times s$ binary base matrix H'. i.e. $Span(H') \leq r$. This is a handy bound that is independent of the base matrix structure and is quite tight for high-rate codes.

Let $\mathcal{X}$ represent the collection of all stopping sets of the matrix H' and let $\mathcal{X}_i$ represent the collection of all stopping sets of size i. Let $N(\mathcal{X}_i)$ represent the number of stopping sets of size i. Let I_χ represent the $N(\mathcal{X}) \times s$ incidence matrix of the set $\mathcal{X}$, where every row of I_χ corresponds to the incidence vector V_{S_k} of a stopping set S_k. By the incidence vector V_{S_k} we mean the binary vector $[b_1, b_2, \ldots, bs]$ where b_i is 1 if $i \in S_k$ and 0 otherwise.

The $2(2n+1)$ cycle bound

If the incidence matrix $I_{\mathcal{X}_2}$ contains a cycle of size $2(2n+1)$ for $1 \leq n \leq \lfloor \frac{s-1}{2} \rfloor$, $span(H') \leq (\lfloor \frac{s-1}{2} \rfloor + 1)$.

Proof: Without loss of generality let us assume that s is even. Suppose p is the permutation that maximizes the span, split p into 2 halves each forming a set of size $s/2$. Denote these sets as P_1 and P_2. Let $\{k_1, k_2, \ldots, k_i\}$ be the columns of H' involved in the cycle and contained in the set P_1; let $\{l_1, l_2, \ldots, l_j\}$ be the corresponding columns of the set P_2 ($i + j$=$2n + 1$). By virtue of an odd number of columns being involved in the cycle, it is seen that one of the sets $\{k_1, k_2, \ldots, k_i\}$ or $\{l_1, l_2, \ldots, l_j\}$ is certain to completely contain one of the stopping sets of size 2 that is part of the cycle. This limits the $span(H')$ to $(\lfloor \frac{s-1}{2} \rfloor + 1)$. ■

The $\binom{n}{k}$ subset bound

If P is a subset of size n of the bit nodes of the Tanner graph of H' such that any set of k bit nodes from P *contains* a stopping set, $span(H') \leq \left\lfloor \frac{s-n}{\lfloor n/(k-1) \rfloor} \right\rfloor + k$.

Proof: Let P^c contain the bit nodes not contained in P. Let H'_P be the Tanner graph formed by only the bit nodes in P. Let p be a permutation of the n bit columns of H'_P. Since every set of k bits from P contains a stopping set, $span(H'_P) \leq k$. If suppose one bit were to be added to every *length-(k-1)* disjoint segment in p then the span can be at most increased by one. Repeating this step until all the bit nodes in P_c are exhausted gives the bound. ■

The minimum weight bound

If P is a set of n columns in H' such that the minimum weight of columns in P is k, $Span(H') \leq \left\lfloor \frac{s-n}{\lfloor n/(r-k+1) \rfloor} \right\rfloor + r - k + 2$.

Proof: For the iterative decoding to be successful on a set of erasures, the sub-matrix formed by erased bit nodes in H' must be reducible to a triangular matrix form. Irreducibility to a triangular matrix implies the presence of a stopping set. Given that the minimum weight of the columns is k, the largest set of columns that can be be reduced to a triangular matrix is r-k+1. Hence, any set of r-k+2 columns from P is certain contain a stopping set. Invoking the $\binom{n}{r-k+2}$ subset bound gives the required bound. ■

V. The Optimization Problem

The objective of this paper can now be given a precise formulation as an optmization problem. Let us consider a (n, k) LDPC code C with a Tanner graph defined by a parity check matrix H. Let $\mathcal{X} = \{S_1, S_2, \ldots, S_t\}$ be the collection of minimal stopping sets in the Tanner graph. A minimal stopping set is one that does not contain any stopping set within it. Let $p : \{1, 2, \ldots, n\} \rightarrow \{1, 2, \ldots, n\}$ denote an arbitrary permutation of the columns of H. If S is a subset of $\{1, 2, \ldots, n\}$ let $p(S) = \{p\{s\} : s \in S\}$. Let $p(C)$ denote a code equivalent to C after the bits have been interleaved according to p. It is readily seen that the stopping sets of $P(C)$ are given by $p(\mathcal{X}) = \{p(S_1), p(S_2), \ldots, p(S_t)\}$. The optimal interleaver that maximizes burst erasure correction capability is given by

$$\arg\max_p span(p(C)) = \arg\max_p span(p(\mathcal{X}))$$

A direct approach to solving the above optimization problem will involve (1) enumerating the minimal stopping sets of a tanner graph, and (2) searching through all the $n!$ permutations of the bit nodes to maximize span.

Clearly, brute force approach here is computationally prohibitive. Also, the above is an instance of combinatorial optimization searching over permutations, and problems of this kind are usually NP-hard, expect in a few instances. The combinatorial nature of the problem and the large search space, along with the unpredictable nature of the objective function, motivates us to adopt a heuristic search method to solve the problem. In the following section we provide an algorithm based on simulated annealing. Simulated annealing is a heuristic search algorithm that has worked very well for a range of optimization problems and scales pretty well as the problem size increases. Simply put, it hops from one permutation to the other, with a slight bias towards permutations that yield greater span, to find an interleaver that improves the span.

A. Simulated Annealing

Simulated annealing is a technique that has been used successfully to solve optimization problems of large scale, especially where a global extremum is hidden amongst many poorer local extrema [18]. Assume s_0 to be the initial state of the system with a finite, discrete, state space. Let t denote the temperate of the system. The initial temperature is set to t_0 (temperature is a parameter that tries to emulate the cooling process in annealing). Let s denote the current state of the system. The function $NextState(s)$ generates a candidate new state that the system can choose to step into. This function is probabilistic in nature, inducing some randomness into the algorithm. The function $E(s)$ denotes the energy of the system when in state s. In optimization, it is the value of the objective function at that state. The function $Oracle(s, NextState(s), t)$ decides whether the system must move to the candidate new state or stay in the current state. For a particular temperature t, the state generation and transition steps are repeated k_{max} times. Then, the system is cooled *i.e* $t \leftarrow \alpha t$ (typically $\alpha = 0.9$), and the state transition procedure is repeated. The algorithm runs until the system has been sufficiently cooled, or the objective function has been satisfactorily decreased/increased. The algorithm returns the best state encountered and its corresponding energy.

Of the many combinatorial problems that have been solved using simulated annealing, the traveling salesman problem(TSP) is relevant to the current context. TSP requires one to compute the least distance round-trip through a given set of cities such that each city is visited exactly once and the trip ends at the starting city. There have been instances where simulated annealing has been proven to be quite effective in solving the TSP involving more than 1000 cities [15]. Just as in the current scenario, the search space in TSP is the set of $n!$

permutations. Span is certainly a more complicated function than the relatively simple distance minimization function of the TSP, thus further strenthening the need to adopt a meta-heuristic algorithm. Having drawn an analogy between the two problems, in the next section, we make use of a simulated annealing routine designed in [17] for the TSP and adapt it to solve the current problem.

B. Simulated annealing for maximizing span

At the heart of any simulated annealing routine lie the two key functions- $Oracle()$ and $NextState()$. To a large extent the success of the routine is dependent on these two functions. To begin with, the state of the system (LDPC code with parity check matrix H) is likened to a permutation p of the columns of the matrix $\mathbf{H}$ and the span of $p(C)$ is taken as the equivalent energy attribute of the system.

Algorithm 1 GenPerm()

```
1: Input- current permutation $p$;
2: At random, select $b_1, b_2$ s.t $b_1, b_2 \in 1, 2, \ldots, n$ & $b_1 < b_2$
3: $p_n : \{1, 2, \ldots, n\} \rightarrow \{p(1), p(2), \ldots, p(b_1 - 1), p(b_2),$
   $p(b_2 - 1), \ldots, p(b_1), p(b_2 + 1), p(b_2 + 2), \ldots p(n)\}$
4: return $p_n$;
```

The $NextState()$ function that generates the candidate new state, or in this case, a new permutation of bits can be chosen in many ways. The function should be such that the energies of the new states generated should be comparable to the energy of the current state. In the current context it would mean that the new permutation p_n generated should be such that $Span(p_n(C))$ is comparable to $Span(p(C))$, where p is the current permutation of the columns of H. One such function that has been widely used while solving the TSP is the GenPerm() function as described in Algorithm 1.

The GenPerm() function picks two bit positions at random and generates a new permutation from the current permutation by flipping all the bits in between the two selected bits while retaining the locations of the remaining bits.

The $Oracle()$ used to make decisions on moving to the next state p_n for the span optimization problem, uses a very standard formula from the Metropolis-Hastings Algorithm [16] and is given below. $Oracle(s, , s_n, t) = Metrop(Span(p_n(C)), Span(p(C)), t) =$

$$\begin{cases} 1 \quad \text{if } Span(p_n(C)) > Span(p(C)). \\ \\ 1 \quad \text{with Prob. } exp(\frac{Span(p_n(C)) - Span(p(C))}{t}) \\ \quad \text{if } Span(p_n(C)) < Span(p(C)). \\ \\ 0 \quad \text{with Prob. } 1 - exp(\frac{Span(p_n(C)) - Span(p(C))}{t}) \\ \quad \text{if } Span(p_n(C)) < Span(p(C)). \end{cases} \tag{1}$$

The mathematical basis for using this formula comes from the Boltzmann distribution of the energy states [17] and has been used in many instances of simulated annealing. This formula always accepts a new permutation p_n that results in a greater span while the decision is probabilistic if the new span $Span(p_n(C))$ is lesser than the current span $Span(p(C))$. Note that the probability of moving to a permutation that gives a lesser span is greater at higher temperatures and gradually drops as the temperature is reduced.

Other parameters such as the initial temperature t_0 and k_{max} are chosen based on the size of the LDPC matrix being handled. The initial temperature is chosen such that it is considerably greater than the largest possible difference in span and the cooling schedule determined by k_{max}, is empirically adjusted for different codes. Algorithm 2 gives the simulated annealing routine designed for maximizing span. The variable $nmoves$ keeps track of the successful state transitions made by the system. The routine is terminated when further efforts to improve the span become discouraging. These are all features that have been incorporated based on the simulated annealing routine for the TSP given in [17].

Algorithm 2 Simulated annealing for maximizing burst erasure correction capability

```
1: Initialize. $p \leftarrow p_0, sp \leftarrow Span(p_0(C))t \leftarrow t_0, k = 0$
2: while ($t > t_f$) do
3:   while ($k < k_{max}$ & $nmoves < 0.2 * k_{max}$) do
4:     $p_n \leftarrow GenPerm(p)$; $sp_n \leftarrow Span(ps_n(C))$;
5:     if $Span(p_n(C)) > Span(p(C))$ then
6:       $sp_h \leftarrow sp_n; , p_h \leftarrow p_n$;
7:     end if
8:     if ($Metrop(Span(p_n(C)), Span(p(C)), t)$) then
9:       $p \leftarrow p_n; sp \leftarrow Span(p_n(C))$;
10:      nmoves++;
11:    end if
12:    k=k+1;
13:  end while
14:  if $nmoves = 0$ then
15:    break;
16:  end if
17:  $t = 0.9t$;
18: end while
19: Return $sp_h, p_h$;
```

VI. Results

Algorithm 2 was used to find interleavers that improved the span of several LDPC codes. The results obtained were compared with the bounds given in Section III and also against the results obtained using the algorithm proposed in [8].

A. WiMax Codes

The algorithm was run on the base matrices of the protograph quasi-cyclic LDPC codes in the WiMax standard. The base matrix has 24 columns. The base matrix H' can be written as a combination of two matrices- $[H_r \ H_{bd}]$ where H_{bd} has a bi-diagonal structure while H_r is a dense random matrix.

The algorithm was run on all the codes with rates varying from 1/2 to 5/6. The algorithm performs exceptionally well

$$\begin{aligned} Rate\ 1/2\ p_{opt} = \quad & [5, 14, 12, 15, 9, 4, 8, 1, 18, 6, 16, 7, 13, \\ & 21, 10, 19, 23, 22, 20, 3, 17, 2, 11, 0] \\ Rate\ 2/3A\ p_{opt} = \quad & [14, 12, 19, 8, 13, 15, 2, 18, 20, 17, 1, \\ & 0, 10, 3, 6, 9, 4, 7, 22, 23, 21, 16, 11, 5] \end{aligned}$$

TABLE I
RESULTS FOR WIMAX CODES

Code	Initial span	Improved span	Improved Span using Algo. in [8]	Bound
Rate 1/2	3	12	6	12
Rate 2/3A	4	6	5	9
Rate 3/4A	2	4	3	5
Rate 3/4B	2	4	2	7

while trying to maximize the span of the rate 1/2 code, returning an interleaver with a span of 12. As the bit pairs (5,7), (7,11) and (5,11) in H' for the rate-1/2 code form stopping sets, the cycle-6 (from Section IV) bound limits the span to 12. Thus it is seen that Algorithm 2 maximizes the span in this case. In comparison, the algorithm described in [8] returns a span of 7. The optimal permutations are given below:

The span of the rate 2/3A code was improved to 6 compared to the obvious bound of 8. It was also noted that running the algorithm on the rate 2/3B code would be a futile exercise because every pair from the set of bits $\{0, 2, 4, 6, 8, 10, 12, 14\}$ forms a stopping set thus invoking the $\binom{8}{2}$ bound and limiting the span to 4. The large number of stopping sets is due to the fact that the rate 2/3B code was designed for layered decoding. An attempt to optimize the span for the rate 2/3B code resulted in a span of just 2.

The spans of rate 3/4A code and the rate 3/4B code were both improved to 4. The minimum weight bound limits the span of the rate 3/4A code to 5. Table I summarizes the results and comparisons with [8].

B. Regular/Irregular LDPC codes

Algorithm 2 was also tried on random LDPC codes. The span optimization on a regular-(3,6), rate-1/2 code with $n = 500$ resulted in an increase of span from 193 to 206 while the optimization on a (600, 200) code of the same degree distribution saw an increase in span from to 149 to 157. The channel threshold based bound, as discussed in [8], limits the maximum burst erasure correction capability of the above codes to 215. Good results were also seen when a rate-1/3 (1920,640) was optimized; this resulted in the span being increased from 1066 to 1146. These results indicate that the simulated annealing approach scales well as n increases. The results are summarized in Table II. Span of a random LDPC code was computed using the technique used in [8].

VII. CONCLUSION

We have studied the burst erasure correction capability of LDPC codes, such as those in the WiMax standard, constructed using shifted identity matrices. We have shown that interleaving in the form of permutations of the columns of the base matrix results in a significant improvement in burst erasure correction capability. We have developed bounds for the maximum burst erasure correction capability and provided a simulated annealing based algorithm for finding good interleavers and compared the results with the bounds. Such interleaving appears to be ideal for applications in magnetic recording and interference-limited wireless channels. Incorporating burst erasure conditions in the construction of the base matrix and exploring the problem landscape for burst erasure correction maximization are possible avenues for future work.

TABLE II
RESULTS FOR RANDOM LDPC CODES

Code rate	Block size	Degree Dist .	Initial span	Improved span	Bound
1/2	500	(3,6)	193	207	215
2/3	600	(3,6)	149	157	215
1/3	1920	irregular	1066	1146	1280

REFERENCES

[1] M.Yang, W.E. Ryan, ”Performance of efficiently encodable low-density parity-check codes in noise bursts on the EPR4 channel“, *IEEE Transactions on Magnetics*, vol. 40, no. 2, pp. 507–512, March 2004

[2] Hongxin Song and J.R. Cruz ”Reduced-complexity decoding of Q-ary LDPC codes for magnetic recording”, *IEEE Transactions on Magnetics*, vol. 39, no. 2, pp. 1081–1087, March 2003

[3] F.Peng, M.Yang, W.E. Ryan, “Simplified eIRA code design and performance analysis for correlated Rayleigh fading channels”, *IEEE Transactions on Wireless Communication*, April 2006.

[4] Changyan Di, D.Proietti, I.E. Telatar, T.J Richardson, R.L Urbanke “Finite-length analysis of low-density parity-check codes on the binary erasure channel,” *IEEE Trans. on Info. Th.*, June 2002.

[5] S.J. Johnson, T. Pollock, ”LDPC codes for the classic bursty channel“ *International Symposium of Information Theory and its Applications* (ISITA’04), pp. 184–189, October, 2004

[6] Gou Hosoya, Hideki Yagi, Toshiyasu Matsushima, Shigeichi Hirasawa, ”Performance of Low-Density Parity-Check Codes for Burst Erasure Channels“Int. Symp. on Info. Th. and its App., ISITA2006, Nov-2006.

[7] Xudong Ma and En-hui Yang, “Constructing LDPC codes by 2-lifts.”,(To be published.)

[8] Paolini E. and Chiani M., “ Improves low-density parity-check codes for burst erasure channels.” *IEEE International Conference on Communications ’06*, June 2006.

[9] Chih-Chun Wang, “Code Annealing and the Suppressing Effect of the Cyclically Lifted LDPC Code Ensemble”, *IEEE Information Theory Workshop*, 2006. ITW ’06.

[10] M.Schwartz, A. Vardy, “On the stopping distance and the stopping redundancy of codes”, *IEEE Transactions on Information Theory*, Vol 52, No 3, pp:922-932, March 2006.

[11] Seho Myung, Kyeongcheol Yang, Jaeyoel Kim, “Quasi-cyclic LDPC codes for fast encoding”, *IEEE Transactions on Information Theory*, Vol 51, Issue 8, pp:2894-2901, Aug 2005.

[12] Dariush Divsalar, Sam Dolinar, Christopher Jones, “Protograph LDPC codes over burst erasure channels”,*IEEE Military Communications Conference*, MILCOM 2006, Oct 2006.

[13] R.M.Tanner, D.Sridhara, A.Sridharan, T.E.Fuja, D.J.Costello Jr, “LDPC block and convolutional codes based on circulant matrices“ *IEEE Trans. on Information Theory*, Volume 50, Issue 12, pp:2966-2984, Dec 2004.

[14] The IEEE 802.16e^{TM}-2005 (WiMax) Standard for LAN and MAN.

[15] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, “Optimization by Simulated Annealing”, Science, Vol 220, Number 4598, pages 671-680, 1983.

[16] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, E. Teller.“Equations of State Calculations by Fast Computing Machines”. Journal of Chemical Physics, 21(6):1087-1092, 1953.

[17] William H. Press, Saul A.Teukolsky, William T. Vetterling, Brian P. Flannery, “Numerical methods in Computing“, 2nd Edition, Cambridge University Press,1992.

[18] http://en.wikipedia.org/wiki/Simulated_annealing