XV Reunión de Trabajo en Procesamiento de la Información y Control, 16 al 20 de septiembre de 2013

A new Algorithm to construct LDPC codes with large stopping sets

Juan Camilo Salazar Ripoll† and Néstor R. Barraza‡

†*Universidad de los Andes.*
palazar43@gmail.com
‡*Universidad Nacional de Tres de Febrero y Facultad de Ingeniería, UBA*
nbarraza@untref.edu.ar

Abstract— **A new algorithm to construct good low-density parity-check (LDPC) codes with large stopping sets is presented. Since the minimum stopping set characterizes an LDPC code, searching for stopping sets in LDPC codes is an important issue. Large minimum stopping sets avoid the LDPC code to get trapped in cycles specially on the binary erasure channel. Dealing with stopping sets is not an easy task since their discovering is a well known NP hard problem. Conversely, we propose an algorithm in order to construct an LDPC code from a stopping set which is demonstrated to be large. Results of simulations showing the performance of the LDPC code obtained this way are analyzed.**

Keywords— **LDPC, stopping set, BEC.**

1 INTRODUCTION

LDPC codes have been an important matter of study since they were rediscovered by Mackay and Neal [1] many years after they were invented by Gallager [2]. Due to rapid development in processors and memory technologies, important low complexity algorithms like belief propagation or sum-product were developed for decoding. At present, LDPC and Turbo codes are the two important branches in coding theory having performance close to the channel capacity. Many issues are important to be analyzed related to getting good performance in LDPC codes. One of them is the size of stopping sets; the larger size, the better the performance since large stopping sets avoid the LDPC code to be trapped in cycles and guarantee convergence, mainly when the binary erasure channel (BEC) is analyzed. Usually the method consists in looking for stopping sets in given LDPC codes in order to analyze performance, see [3] and [4]. The problem of finding stopping sets is well known to be NP Hard, see [5]. On the other hand, some algorithms to construct LDPC codes having large stopping sets are also proposed, see for example [6]. Other techniques are based on coset graphs, see for example [7], and in multidimensional finite lattice, see for example [8]. In this work, we propose a new algorithm to construct an LDPC code by expanding a graph which represents a minimum stopping set increasing the variable node degree for a given girth. This paper is organized as follows: some remarks on graphs and their properties are exposed in Section 2. The algorithm we propose is carefully described in Section 3. In Section 4, the performance of the code in a BEC is analyzed and results are shown. Conclusions of this work are given in Section 5.

2 GRAPHS AND PROPERTIES

A simple graph is a set of tuples $G = \{V, E\}$ where V is a given set whose elements are called nodes and E is a subset of $\begin{pmatrix} V \\ 2 \end{pmatrix} = \{\{a,b\}\,|a,b \in V, a \neq b\}$. The degree of a node $v \in V$ is the number of edges connected to it, it means, $\#\{\{v,u\} \in E | u \in V, u \neq v\}$.

A path between nodes, $a, b \in V$, is a sequence of nodes $a = v_0, v_1, \ldots, v_n = b$ such that $\{v_{n-1}, v_n\} \in E$, for $n = 1, \ldots, n$. A cycle is a path where $a = b$. The length of the smallest cycle of a graph is called girth. If a graph is cycle free its girth is defined as infinity.

A bipartite graph is a graph whose nodes can be separated into 2 sets: the right set and the left set such that the neighbors of the nodes in the right set lie in the left set. A "vertex-edge incident" graph of a graph G is a bipartite graph such that the left set represent the nodes V in G, the right set represent the edges E in G and a node u in the left set is connected to a node v in the right set if there is a node $w \in V$ such that $v = \{u, w\}$.

Every linear LDPC code can be described using a bipartite graph, where one set represent the variable nodes and the other set of nodes represent the check nodes. A stopping set S of an LDPC code is a subset of the variable nodes such that every neighbor of S is connected to S at least twice.

3 THE ALGORITHM

3.1 The method

The aim is to get a graph which determines the minimum stopping set of the obtained code. The parity check matrix of the code is obtained as the transpose of the vertex-edge incidence matrix of the graph. This method allows to construct LDPC codes up to a girth of 24, and with a slight variation the girth can be increased to 28.

The mentioned graph is obtained from the following steps:

1. Take a core C which is a simple graph, its girth determines the girth of the LDPC code.
2. Make $2|C|+1$ copies of the core obtaining $2|C|+2$ subgraphs.
3. Divide the subgraphs into two sets: a left set and a right set, each one of $|C|+1$ subgraphs. Lets name the subgraphs in the left set $0, 1, \cdots, |C|$ and the subgraphs in the right set $0', 1', \cdots, |C|'$.
4. Connecting the nodes
   (a) Take the node i from the graph j and connect it to the node j of the graph i' for $i \neq j$ with $1 \leq i, j \leq |C|$.
   (b) Connect the node i from the graph i to the node i of the graph $0'$, in a similar way connect the node i from the graph i' to the node i of the graph 0.

From the previous general procedure we get a graph with $2|C|(|C|+1)$ nodes. Notice that the degree of each node is increased by one.

As an example, we will show how to obtain a regular graph with girth 12. A regular core with girth at least 12 must be chosen since girth(graph) = $\min$(girth(core),12).

In order to show that the obtained graph has girth 12, we will show that no cycle of length less than 12 is generated using the previous construction. First, it is easy to see that the girth of each subgraph is the same as the chosen core. Then, the cycles involving several subgraphs must be analyzed. In order to do that, some graphs considering the worst cases are shown next.

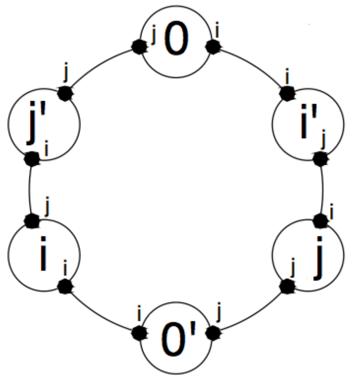

Figure 1: The smallest cycle involving 0 and $0'$

These particular graphs are those involving subgraphs 0 and $0'$. Let i and j be adjacent nodes in each subgraph, the cycle shown in Fig 1 is obtained from our method.

For cycles not involving subgraphs 0 and $0'$, the worst case is such having cycles of length 14. In order to see that, let i, j, k be a path in each subgraph, then, a cycle of length 14 is obtained. This case is shown in Fig. 2.

In order to improve the previous method to get a final graph with girth 14, we modify the connecting procedure. As it was explained, a core with girth at least 14 must be chosen. This modification consists of renaming the nodes in subgraphs 0 and $0'$ such that edges in the original core are not present in the modified subgraph. This means, let $C = (V, E)$, the original core, then the modified subgraphs $0, 0' = (V, E')$, where $E' \cap E = \emptyset$.

To show that the obtained graph has girth 14, we will show that no cycle of length less than 14 is generated. As previously shown, the smallest cycles not involving subgraphs 0 and $0'$ have length 14. The worst case involving 0 and $0'$ contains a cycle of length 14. Let i and j be adjacent nodes in the original core, which means that they are not adjacent in 0 and $0'$, then the worst case involving the subgraphs 0 and $0'$ is shown in figure 3.

The other case involving subgraphs 0 and $0'$ has a cycle of length 16. Let k and l be adjacent nodes in the modified subgraphs 0 and $0'$, which means that they are not adjacent in the original core. The worst case is shown in figure 4.

3.2 Getting the LDPC code

As it was explained before, the parity check matrix H is obtained as the transpose of the vertex-edge incidence matrix of the graph. The nodes of the graph are the check nodes of the code and the edges of the graph are the variable nodes. Cycles of length k give cycles of length $2k$ in the Tanner graph. Then, the size of the stopping set in the LDPC code will not be less than the girth of the graph. Consequently, this method guarantees that the size of the stopping set will not be small.

If a regular graph is chosen as the core, being d_v the degree of each node, then the number of edges in the generated graph is $(d_v+1)(|C|)(|C|+1)$. As a result, the rate of the code generated by this method will be $R = \frac{1}{d_v+1}$.

4 SIMULATION

For our simulation, we chose a ring of length 22 as the core. This ring is a simple regular graph of girth 22. In order to rename the nodes in the subgraphs 0 and $0'$, we use a simple formula: $i := i \cdot 5 mod(22)$, for the given example, the permutation is as follows [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22] $\rightarrow$ [9, 18, 5, 14, 1, 10, 19, 6, 15, 2, 11, 20, 7, 16, 3, 12, 21, 8, 17, 4, 13, 22].

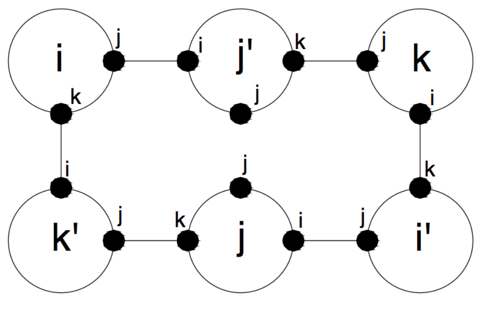

Figure 2: The smallest cycle not involving 0 and $0'$

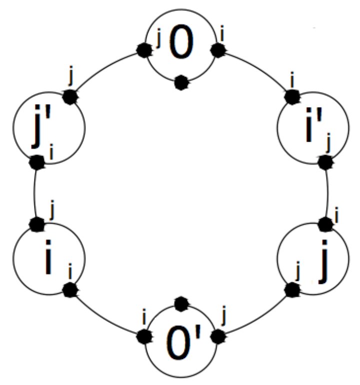

Figure 3: The smallest cycle with adjacent nodes in the original core

The generated graph has $2(22)(23) = 1012$ nodes and $3(22)(23) = 1518$ edges. The parity check matrix H has dimensions 1012×1518, where every row has exactly two 1's.

Performance of the obtained LDPC code on a binary erasure channel (BEC) using the message-passing algorithm for decoding are shown in figure 5.

5 CONCLUSIONS

A new algorithm to construct an LDPC code from a generated graph was presented. This graph is generated by making some connections between several copies of a given core. Since the stopping set of the LDPC code is related to the girth of the graph, a large stopping set size is obtained. The parity check matrix is quite sparse, then, the generated LDPC code converges in just a few iterations.

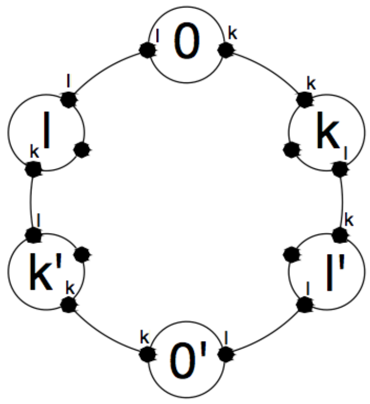

Figure 4: The smallest cycle with adjacent nodes in the modified subgraphs

It is possible to generate bigger codes using the obtained graph as the core. We are working now on this issue and it will be shown in future work.

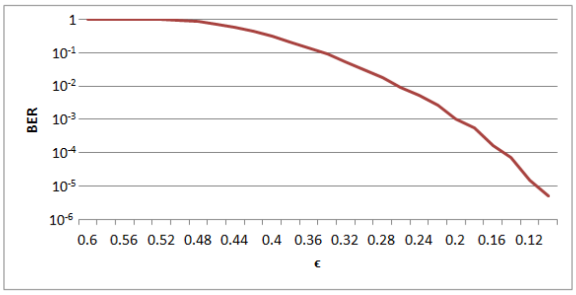

Figure 5: Performance of the LDPC code in a BEC (R = 1/3, n = 1518) with error probability ϵ.

ACKNOWLEDGMENTS

One of us (NRB) wants to thank ”Universidad Nacional de Tres de Febrero” and ”Facultad de Ingeniería. UBA” for support.

REFERENCES

[1] D. J. C. MacKay and R. M. Neal, “Near Shannon limit performance of low density parity check codes,” *Electronics Letters*, vol. 32, no. 18, pp. 1645–1646, August 1996, reprinted *Electronics Letters*, vol 33, no 6, 13th March 1997, p.457–458.

[2] R. G. Gallager, “Low-density parity-check codes,” *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.

[3] G. Richter, “Finding small stopping sets in the tanner graphs of ldpc codes,” in *In 4th International Symposium on Turbo Codes and Related Topics*, 2006.

[4] E. Rosnes and O. Ytrehus, “An efficient algorithm to find all small-size stopping sets of low-density parity-check matrices,” *IEEE Trans. Inf. Theor.*, vol. 55, no. 9, pp. 4167–4178, Sep. 2009. [Online]. Available: http://dx.doi.org/10.1109/TIT.2009.2025573

[5] K. M. Krishnan and P. Shankar, “On the complexity of finding stopping distance in tanner graphs,” *CoRR*, vol. abs/cs/0512101, 2005.

[6] G. Richter and A. Hof, “On a construction method of irregular ldpc codes without small stopping sets.” in *ICC*. IEEE, 2006, pp. 1119–1124.

[7] J. Lauri and O. Tjhai, C.J., “Coset graphs for low-density parity check codes: performance on the binary erasure channel,” *IET Comm.*, vol. 5, no. 5, pp. 719–727, Mar. 2011.

[8] J. Craddock, M. F. Flanagan, S. J. Redmond, and A. D. Fagan, “Construction of girth 8 ldpc codes based on multidimensional finite lattices.” in *ISCC*. IEEE, 2007, pp. 655–659.