Cut-Node Tree and Loops of LDPCs

Huanming Zhang

Institute of Electronics and Information Engineering
Foshan University, Guangdong, China, 528000

Abstract. Bipartite graph is used to illustrate Low-density parity-check (LDPC) code by Tanner [1]. In this thesis, we introduce a new algorithm to depict LDPC codes. Cut-node tree(CNT), although appearing to be a tree defined in graph theory, can figure out all cycles of an LDPC. And this method can apply to decoding, help to prevent superfluous iteration in belief propagation (BP).

1. Introduction

In coding and decoding domain, Tanner graph(TG) is suitable for indicating constraints or equations in LDPC, and iterative decoding of forward error correcting codes [2]. See Fig.1. It can commendably compose longer codes, [3][4].

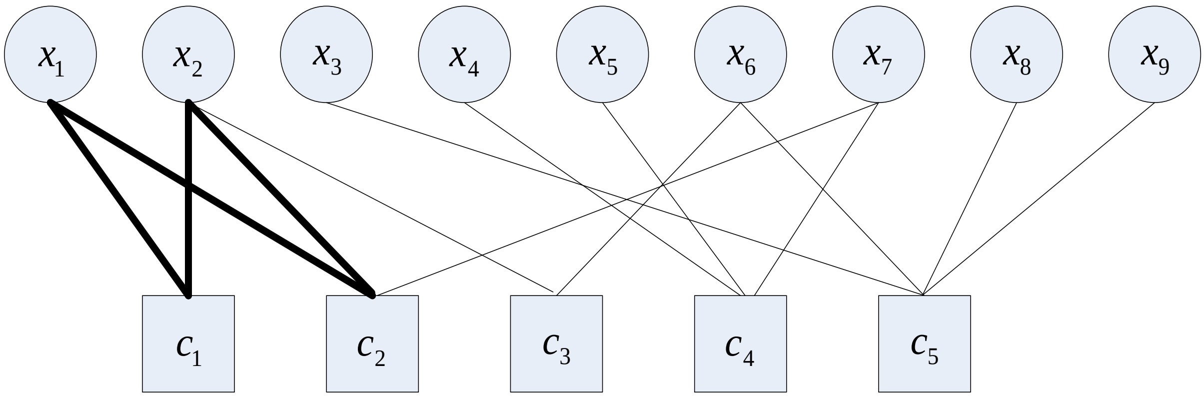

Figure 1. TG of an LDPC code

For a codeword of FEC, N is the length of codeword. the transmitted variable, supposing, is $x=\{x_1,\cdots,x_N\}$ and unknown; instead of received value $y=\{y_1,\cdots,y_N\}$, M Parity-check equations can be got, $c=\{c_1,c_2,\cdots,c_M\}$. $f=\{f_1^a,\cdots,f_N^a\}$ is initial value of transmitted codeword(variable). Where x_i is ith bit for a binary system of variable.

Belief propagation (BP) algorithm of FEC or particularly LDPC code can be elaborated by TG [5]-[7], See Fig.2.

R_{ji}^a represents check value from check c_j to variable v_i, and Q_{ij}^a variable value from v_i to check c_j.

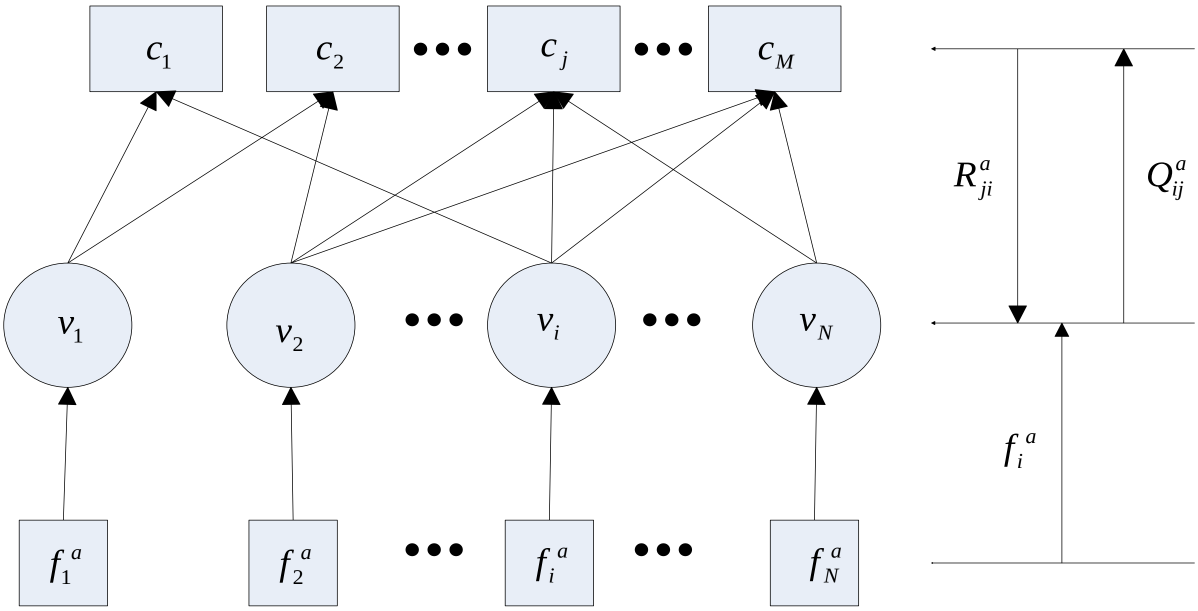

Figure 2. BP decoding Algorithm of LDPC

$$R_{ji}^0(t+1)=\frac{1}{2}\left(1+\prod_{j'\in N(i)\setminus j}\left(1-2Q_{ij'}^1(t)\right)\right) \tag{1}$$

$$R_{ji}^1(t+1)=1-R_{ji}^0(t+1) \tag{2}$$

$$Q_{ij}^0(t+1)=\alpha_{ij}(1-P_i)\prod_{i'\in M(j)\setminus i}R_{ji}^0(t+1) \tag{3}$$

$$Q_{ij}^1(t+1)=\alpha_{ij}P_i\prod_{i'\in M(j)\setminus i}R_{ji}^1(t+1) \tag{4}$$

Where α_{ij} is normalization value, satisfy $Q_{ij}^0(t+1)+Q_{ij}^1(t+1)=1$.

2. Theory Of CNT

In order to figure out all cycles of code. TG is re-drew to a tree, cut-node tree:

(1)selecting a value ‘1’ in H, its variable node considered as root-node;

(2)correspondingly, all check nodes connected to the root-node as first-order child nodes;

(3)when growing, a node, it is a leaf originally or it appeared in previous node, will be forbid to grow and become an end-node;

(4)repeating(2) (3), at the end, a CNT can be got. If all nodes are connected, a single CNT can be got. Repeating this process until all end nodes are either cut-nodes or leaf nodes. See Fig.3:

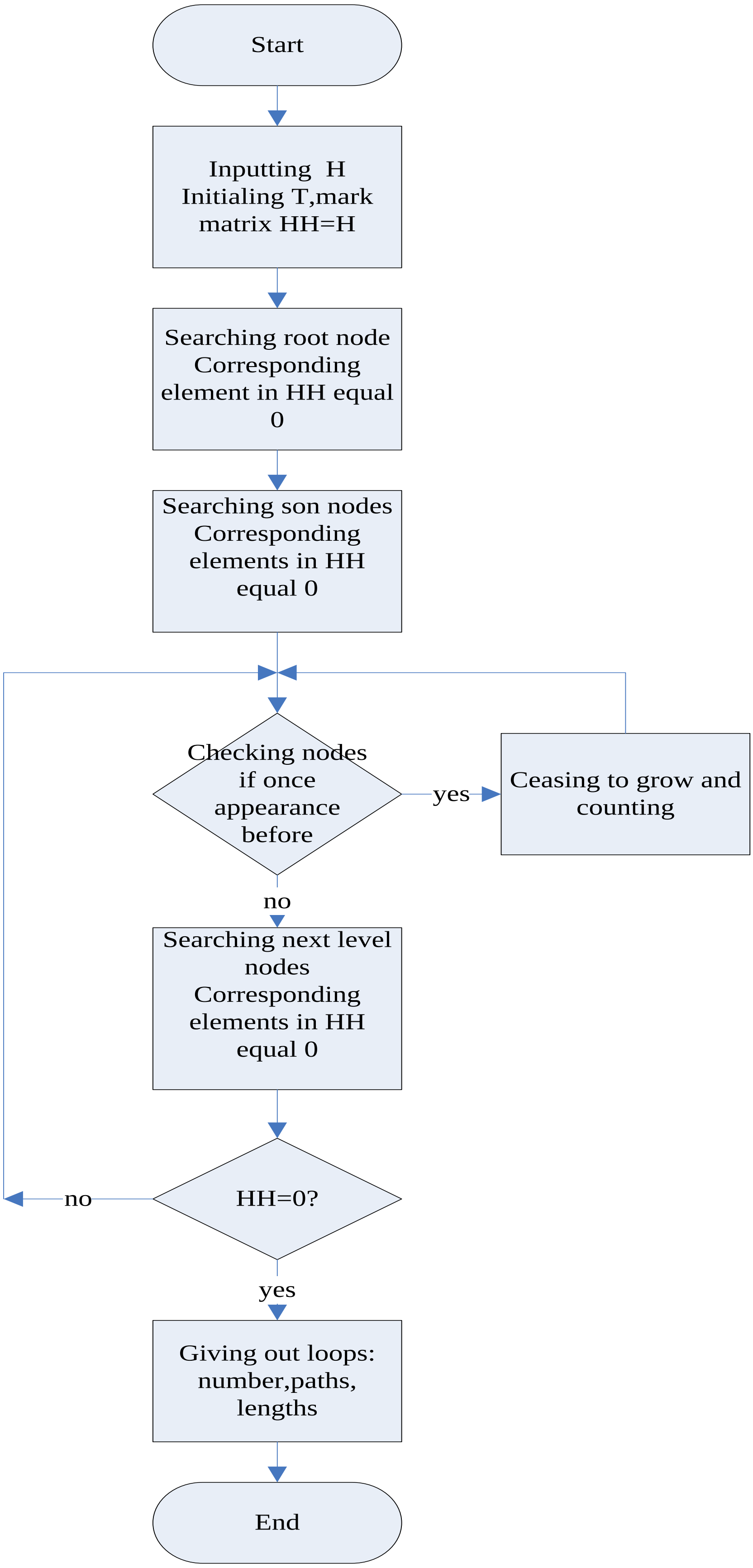

Figure 3. Diagram of CNT algorithm

The initial value of matrix HH is H, when a node appeared in CNT , the corresponding element in HH replaced by zero; all elements of HH change to zero at the end of algorithm.

3. Implement of Principle and Results

Here for one example:

(1)first, choosing a variable node $v_i, h_{ij} = 1$, as root-node, the son nodes of this root-node are those check nodes which connect to it, $v_i, h_{ij} = 1$;

(2)according to this principle, all child-nodes can be obtained. [6].

Such as an LDPC code with check *H*:

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

CNT matrix used to express *H* by matlab. In fact, here tree matrix matches TG by element [i j k l m], [i j] states current node, [k l] states father node, m states times being cut, m=-1inherited, m=0 leaf.

Root-node: [1 1 0 0 -1]

Son-node Level1: [2 1 1 1 -1] [3 1 1 1 -1]

Son-node Level2: [2 2 2 1 -1] [2 4 2 1 -1] [2 6 2 1 0] [3 3 3 1 -1] [3 4 3 1 1] [3 7 3 1 0]

Son-node Level3: [1 2 2 2 -1] [3 4 2 4 2] [1 3 3 3 -1]

Son-node Level4: [1 1 1 2 1] [1 3 1 2 1] [1 5 1 2 0]

So, searched by a computer, all cycles of Eq.5 can be get by this algorithm. See Table 1.

Cycle 1: [1 1]-[1 2]-[2 2]-[2 1]-[1 1]

Cycle 2: [1 3]-[1 2]-[2 2]-[2 1]-[1 1]-[3 1]-[3 3]-[1 3]

Cycle 3: [3 4]-[2 4]-[2 1]-[1 1]-[3 1]-[3 4]

Table 1. Features of cycles about above example

Parameters	Values
Sparsity	0.43
Cycles	3
Total length of cycles	14
Average length	4.67
Girth	4
Maximum length	6
Relativity	1.5

Here, *H* is a single CNT, with 5 cut-nodes, 3 leaf node and 3 cycles.

In a graph without cycle, CNT just a tree defined by graph theory, but with cycles CNT can be got by cutting all cycles.

CNT has all characters which TG contains.

4. Conclusion

In this paper, we give out CNT algorithm. If matrix *H* is large, such as *H* in DVB-S2, cycles entangle with each other, it is hard to solve all cycles. LDPC's performances have certain relationship with characters of cycles. By CNT and matlab's cell array, cycles can be figure out relatively easily.

On the other hand, CNT can be applied in decoding algorithm to prevent redundant iteration in belief propagation algorithm.

5. Reference

[1] R. G. Gallager, Low-Density Parity Check Codes, MIT Press, Cambridge, MA, 1963.

[2] D. J. C. Mackay, "Good Error-Correcting Codes Based on Very Sparse Matrices", IEEE Trans.

Inform. Theory, vol. 45, 399-431, Mar. 1999.

[3] R. M. Tanner, "A recursive approach to low complexity codes," IEEE Trans. Inf Theory, pp. 533-547, Sept. 1981

[4] R.Kschischang, B.J.Frey, and H.A.Loeliger, "Factor graphs and the sum-product algorithm" IEEE Trans. Inform. Theory, vol.47, no.2, pp.498–519, Feb 2001.

[5] N.Wiberg, Codes and decoding on general graphs, Linkoping Studies in Science and Technology, Ph.D. dissertation No. 440, Univ. Link oping, Sweden, 1996.

[6] Reinhard D, Graph Theory[M]. New York: Springer-Verlag, 1997

[7] Martin J. Wainwright, Sparse Graph Codes for Side Information and Binning, IEEE SIGNAL PROCESSING MAGAZINE [47],47-57, SEPTEMBER 2007

[8] Martin Wainwright, Tommi Jaakkola, and Alan Willsky, Tree-based reparameterization analysis of belief propagation and related algorithms for approximate inference on graphs with cycles, ISIT2002, Lausanne, Switzerland, June 30-July 5, 2002

[9] Gholami Mohammad, Nassaj Akram, LDPC codes based on Mobius transformations, IET Communications,v13,n11,p1615-1624;ISSN:17518628,DOL:10.1049/iet-com.2018.6192

[10] Xu, Hengzhou, Zhu, Hai,Xu, Mengmeng, Zhang, Bo, Zhu, Sifeng,Girth analysis of tanner (5,11) quasi-cyclic LDPC codes, Proceedings - 14th International Conference on Computational Intelligence and Security, CIS 2018, p 210-214, December 5, 2018, Proceedings - 14th International Conference on Computational Intelligence and Security, CIS2018; ISBN-13: 9781728101699; DOI: 10.1109/CIS2018.2018.00053; Article number:8564291; Conference: 14th International Conference on Computational Intelligence and Security, CIS 2018, November 16, 2018 - November 19, 2018;