

LETTER

Fast Decoding Algorithm for Low-Density Parity-Check Codes

DAN WANG[†], LI PING^{††}, Xiao YU HU^{†††}, and Xin MEI WANG[†], *Nonmembers*

SUMMARY A fast decoding algorithm for low-density parity-check codes is presented based on graph decomposition and two-way message passing schedule. Simulations show that the convergence speed of the proposed algorithm is about twice that of the conventional belief propagation algorithm.

key words: LDPC code turbo-type two-way algorithm

1. Introduction

The belief propagation algorithm (BPA) is considered to be a standard decoding technique for low-density parity-check (LDPC) codes[1-3]. The conventional BPA is based on the flooding schedule[2] (we will refer to this type of BPA as the flood algorithm (FA) for convenience in this paper), in which all the variable nodes and, successively, all the check nodes pass new messages to their neighbors in a decoding iteration. In this algorithm, the messages from each node only affect the state of its immediate neighbors, which can lead to a slow convergence speed when serial processors are involved. In order to improve the convergence speed, a turbo-type two-way algorithm (TTWA) is presented here for decoding a general LDPC code. This new algorithm is developed from a two-way schedule technique for local decoding combined with a turbo mechanism for global decoding. A pre-processing stage is involved to decompose the general LDPC code into several tree-structured component codes so that the LDPC code can then be viewed as a concatenated tree (CT) code [4], TTWA can then be used to accomplish the decoding task. The proposed method is particularly suited to the numerical evaluation of LDPC codes since modern computers are still mainly based on serial processors.

2. Two-Way Schedule

This technique is suitable for decoding tree code, that is, a code whose Tanner graph contains no loops. In this schedule, messages are only passed once in each direction along each edge in the tree during the whole decoding process to achieve optimal performance[2]. The

[†]The author is with the State Key Lab of ISN, Xidian University, Xi'an, Shaanxi, China

^{††}The author is with Department of Electronic Engineering, City University of Hong Kong, TatChee Avenue, Kowloon, Hong Kong

^{†††}The author is with IBM Zurich Research Laboratory, Saumerstrasse 4, CH-8803, Rueschlikon, Switzerland

two-way schedule is more efficient than the flooding schedule when decoding a tree code using a serial processor (detailed discussions can be found in[2]). However it cannot be directly applied to a code containing loops.

3. CT codes and Turbo-Type decoder

CT codes are formed from the parallel concatenation of several tree codes[4], and can be decoded by a serial Turbo-type algorithm. Suppose that there are M local decoders and each is responsible for one component tree code. The main decoding process is:

1. At the first iteration, an initial *a priori* LLR vector for code bits is fed into the first local decoder.
2. The local decoders run successively on the turbo principle [5]. Let $L^{(m)}$ denote the *a posteriori* LLR vector of the m th component code and $E_{last}^{(m)}$ the extrinsic LLR vector generated in the last iteration. Firstly, $E_{last}^{(m)}$ is initialized to 0 for all m . With $\tilde{L}^{(m)} = L^{(m)} - E_{last}^{(m)}$ as its input, the m th local decoder performs *a posteriori* probability decoding and then outputs $L^{(m)}$ for the next decoder. Finally, the M th (last) local decoder $L^{(M)}$ feeds into the 1st (first) local decoder for the next iteration.
3. The process iterates until some termination condition is satisfied.

With this algorithm, CT codes combine the advantages of LDPC codes and Turbo codes and can achieve good performance with low decoding cost. In the following, we will show that this mechanism can be applied to general LDPC codes to improve convergence speed.

4. TTWA and Graph Decomposition

TTWA, as the name indicates, uses the turbo mechanism in the global decoder and the two-way schedule in the local decoders. If properly used, this algorithm can provide fast convergence when a serial processor is involved. Before it can be applied to a general LDPC code, we need to decompose the code into several trees by graph decomposition. Many methods can be employed for this purpose. One approach is as follows. Given a Tanner graph T with variable nodes $\nu_1, \nu_2, \dots, \nu_N$ and check nodes c_1, c_2, \dots, c_M (N and M are the number of variable nodes and check nodes respectively), we initially construct the first subtree T_1 by taking all of the variable nodes (and without

any check nodes). Next, we test each check node c_k one by one. If adding c_k into T_1 results in loops, it will not be acceptable; otherwise, we can add c_k into T_1 . After testing all the check nodes, the process of constructing T_1 is completed. In the same way, we construct T_2 from the check nodes that were not added to T_1 . The process is repeated to build sub trees T_3, T_4, \dots until no check nodes are left. The above decomposing process is illustrated by Fig.1, where Fig.1.a is a Tanner graph with loops while Fig.1.b gives its two sub trees. After the decomposition, the LDPC code can be viewed as a CT code, and so the TTWA can be used for decoding.

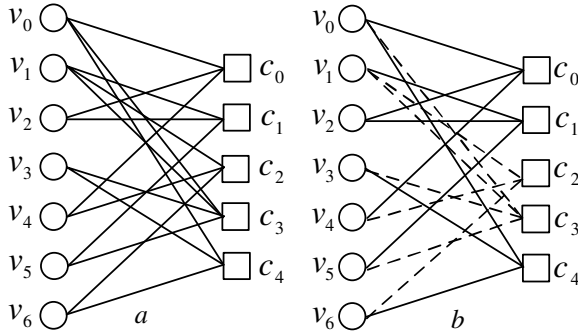


Fig.1 a.Tanner graph of LDPC code b.Tree graphs of subcode1 (solid line) and subcode2 (dot line)

5. Complexity

The proposed TTWA and the conventional FA involve almost the same number of operations at each iteration, i.e. about $6N\lambda$ multiplications and $5N\lambda$ additions each iteration, where λ is the average variable node degree.

6. Numerical Results

Fig.2 shows the performance of two irregular LDPC codes (code length=1008 and 5000 respectively) decoded using the TTWA and FA methods. In the graphs, solid lines mark the TTWA results and dashed lines mark the FA performance; the iteration number can also be read from the figures. The LDPC codes used in the simulation are constructed by the PEG algorithm [5] with the following degree distributions:

$$\lambda(x) = 0.238687x^1 + 0.209597x^2 + 0.0035803x^3 + 0.135505x^4 + 0.380408x^{14} \tag{1}$$

$$\rho(x) = 0.017401x^6 + 0.982599x^7 \tag{2}$$

Evidently, the TTWA converges faster than the FA. This is due to the fact that the TTWA can achieve locally optimal solution for a component tree code within one iteration (this property does not hold for the FA). Experimentally, we observed that different decompositions of an LDPC code make little difference to the convergence speed and performance. We also observed that FA requires about twice the number of

iterations to achieve the same BER as TTWA.

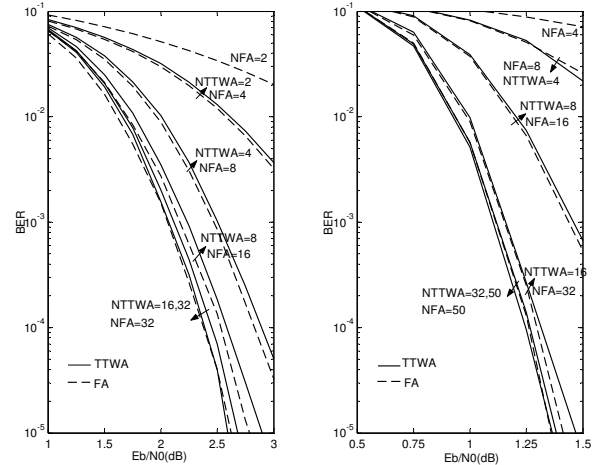


Fig.2 Comparison of convergence speed of TTWA and FA Rate = 1/2 length = 1008(left)length = 5000 (right)

7. Conclusion

We have presented a TTWA algorithm for decoding general LDPC codes using a serial processor. Simulation results show that the new algorithm converges twice as fast as the conventional FA algorithm with almost the same computational complexity and can achieve the nearly same decoding performance. Thus it can explore full advantage of the Turbo mechanism and two-way schedule.

Acknowledgement

This work was supported by CERG under grand N-CityU N-101/01 and NSFC under Grant 60131160742

8. References

1. R.G.Gallager, "Low-density parity-check codes", IRE Trans. Inf. Theory, Vol. IT-8, pp. 21-28, Jan. 1962
2. F.R.Kschischang, and B.J.Frey, "Iterative decoding of compound codes by probability propagation in graphical models", IEEE J. Selected Areas Commun., Vol.16, pp. 219-230, Feb.1998
3. F.R.Kschischang, B.J.Frey, and H.A.Loeliger, "Factor graph and the sum-product algorithm", IEEE Trans. Inf. Theory, Vol. 47, pp.498-518, Feb.2001,
4. LI Ping, and K.Y.Wu, "Concatenated tree codes: a low complexity, high performance approach", IEEE Trans. Inf. Theory, Vol.47, pp.791-799, Feb. 2001
5. X.Y.Hu, E.Elefteriou, and D.M.Arnold, "Regular and irregular progressive edge-growth Tanner graphs", IEEE Trans. Inform. Theory, vol. 51, pp. 386 C 398, Jan. 2005