IOSR Journal of Computer Engineering (IOSRJCE)
ISSN: 2278-0661 Volume 2, Issue 3 (July-Aug. 2012), PP 12-24
www.iosrjournals.org

Message Passing Algorithm: A Tutorial Review

Kavitha Sunil, Poorna Jayaraj, K.P. Soman
Centre for Excellence in Computational Engineering and Networking
Amrita Vishwa Vidyapeetham, India-641112.

Abstract: *This tutorial paper reviews the basics of error correcting codes like linear block codes and LDPC. The error correcting codes which are also known as channel codes enable to recover the original message from the message that has been corrupted by the noisy channel. These block codes can be graphically represented by factor graphs. We mention the link between factor graphs, graphical models like Bayesian networks, channel coding and compressive sensing. In this paper, we discuss an iterative decoding algorithm called Message Passing Algorithm that operates in factor graph, and compute the marginal function associated with the global function of the variables. This global function is factorized into many simple local functions which are defined by parity check matrix of the code. We also discuss the role of Message Passing Algorithm in Compressive Sensing reconstruction of sparse signal.*

Index Terms*—Linear block code, factor graph, LDPC, Bayesian networks, belief propogation, Message passing algorithm, sum product algorithm, Compressive sensing*.

I. Introduction

This paper provides tutorial introduction to linear block codes, factor graph, Bayesian network and message passing algorithm. In coding theory, to enable reliable delivery of bit stream from its source to sink over noisy communication channel error correcting codes like linear block codes and LDPC are introduced. While the message is sent from source to sink, error is introduced by the noisy channel. Error correcting techniques help us to recover the original data from the distorted one. These error correcting codes are graphically represented using factor graphs and an iterative decoding algorithm for the same is developed.

Message passing algorithm which is an iterative decoding algorithm factorizes the global function of many variables into product of simpler local functions, whose arguments are the subset of variables. In order to visualize this factorization we use factor graph. Here we discuss the message passing algorithm, called the sum product algorithm. This sum product algorithm operates in a factor graph and compute various marginal function associated with global function.

Then we link factor graphs with graphical models like Bayesian (belief) networks. Bayesian networks show the factorization of joint distribution function (JDF) of several random variables. MacKay and Neal, was the first to connect Pearl's 'Belief Propagation' algorithm with coding. In message passing algorithm the messages passed along the edges in the factor graph are probabilities or beliefs.

In this paper we tried to unify the work [9], [10],[11],[5],[7],[17].In Section II, we review the fundamentals of binary linear block codes, factor graphs and LDPC. Graphical models like Bayesian networks are developed as a key to iterative decoding algorithm in Section III. In Section IV we discuss the message passing algorithm which decodes the original data from the distorted data. Section V discuss the probability domain version of sum product algorithm. Section VI discuss the role of Message Passing Algorithm in Compressive sensing reconstruction of sparse signal. *Approximate Message Passing*(AMP)algorithm for compressive sensing was recently developed by David L. Donoho,Arian Maleki and Andrea Montanaria in [17].In section VII we conclude.

II. Binary Linear Block Codes

Linear block codes are conceptually simple codes that are basically an extension of single-bit parity check codes for error detection.

In channel encoder a block of k message bits is encoded into a block of n bits by adding $(n-k)$ number of check bits. These $(n-k)$ number of check bits are derived from k information bits. Clearly $n > k$ and such a code is called (n,k) block code. A (n,k) block code is said to be a (n,k) linear block code if it satisfies the condition that any linear combination of codeword is still a codeword. A (n,k) binary linear block code is a finite set $C \in F_2^n$ of codewords x. Each codeword is binary with length n. C contains 2^k codewords. An (n,k) block code C is a mapping between a k -bit message vector u

$$u = [u_1, u_2, ..., u_k] \tag{1}$$

and an n length codeword vector x

$$x=[x_1, x_2, ..., x_n] \tag{2}$$

The code C can also be viewed as the mapping of k -space to n -space by a $k \times n$ generator matrix G which is given by:

$$G = [I_k \mid P_{k\times(n-k)}]_{k\times n} \tag{3}$$

Where I_k is the identity matrix of order k and P is the parity matrix.

The generator matrix is a compact description of how codewords are generated from information bits in a linear block code.

$$x = u \otimes G \tag{4}$$

The row of generator matrix constitutes a basis of the code subspace.

A. Parity Check Matrix

Every linear subspace of dimension k has an orthogonal linear subspace $C^{\perp}$ of dimension $(n-k)$ such that for all $x \in C$ and $x' \in C^{\perp}$, we have $\langle x, x' \rangle = 0$. This $C^{\perp}$ is dual code to C. This dual code also has a basis, which is called the parity check matrix is given by

$$H = [P^T_{(n-k)\times k} \mid I_{n-k}]_{(n-k)\times n} \tag{5}$$

Since we have $\langle x, x' \rangle = 0$ for all $x \in C$ and $x' \in C^{\perp}$, it must be the case that for all $x \in C$, we have

$$H \otimes x = 0_{m\times 1} \forall x \in C \tag{6}$$

and so we have

$$H \otimes G^T = 0 \tag{7}$$

The matrix H is called the parity check matrix of the code C and has the property that a word x is in the code if and only if x is orthogonal to every row of H. The matrix H is called the parity check matrix because every row induces a parity check on the codeword. i.e., the elements of the row that are equal to 1 define a subset of the code bits that must have even parity.

B. Factor Graphs

Factor graph is a generalization of a "Tanner graph". In coding theory, tanner graph is a bipartite graph which means that there are two kinds of nodes and the same kind of nodes are never connected directly with an edge. In probabilistic theory, factor graph is a particular type of graphical model, with applications in Bayesian inference, that enables efficient computation of marginal distributions through the sum-product algorithm.

Factor graphs are partitioned into variable nodes and check nodes. For linear block codes, the check nodes denote rows of the parity-check matrix H . The variable nodes represent the columns of the matrix H . An edge connects a variable node to a check node if a nonzero entry exists in the intersection of the corresponding row and column. From this we can deduce that there are $m = n - k$ check nodes and n variable nodes [5].

Let $x = \{1, ..., n\}$ and $c = \{1, ..., m\}$ be indices for the columns and rows of the $m \times n$ parity check matrix of the code [5]. L is a bipartite graph with independent node sets x and c .We refer to the nodes in x as variable nodes, and the nodes in c as check nodes. All edges in L have one endpoint in x and the other in c .Generally, use i to denote variable nodes, and j to denote parity checks. V_j , is the set of variable nodes i that are incident to j in L .Similarly, C_i is the set of check nodes incident to a particular variable node i in L .

Let us take (6, 2) hamming code *example* [14]:
Consider

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

and for any valid codeword x .

$$H = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = 0$$

This expression serves as the starting point for constructing the decoder. The matrix/vector multiplication in above equation defines a set of *parity check equation:*

$$H \otimes x = \begin{Bmatrix} chk(A): x_2 \oplus x_3 = 0 \\ chk(B): x_1 \oplus x_2 \oplus x_4 = 0 \\ chk(C): x_1 \oplus x_5 = 0 \\ chk(D): x_2 \oplus x_6 = 0 \end{Bmatrix}$$

In figure. 1, the factor graph has 4 check nodes, 6 variable nodes, for example, there is an edge between each variables x_1, x_3 and check node A.

A cycle of length1 in a factor graph is a path of 1 distinct edges which closes on itself. The girth of a factor graph is the minimum cycle length of the graph.

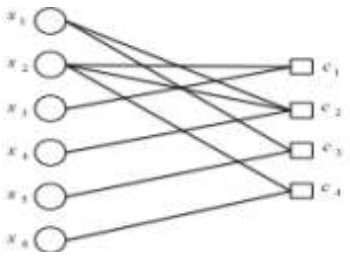

Figure 1: *Factor graph corresponding to above parity check matrix*

C. Low Density Parity Check Codes

In information theory, a low-density parity-check (LDPC) code is a linear error correcting code, a method of transmitting a message over a noisy transmission channel, and decoded using a sparse bipartite graph.

The name comes from the characteristic of their parity-check matrix which contains only a few number of $1's$ in comparison to the amount of $0's$.

LDPC code is similar to any other block code except for the requirement that H must be sparse [4]. Indeed existing block codes can be successfully used with the LDPC iterative decoding algorithms if they can be represented by a sparse parity-check matrix. Finding a sparse parity-check matrix for an existing code is impractical. So inorder to design LDPC codes a sparse parity-check matrix is constructed first and a generator matrix for the code is determined afterwards.

The biggest difference between LDPC codes and classical block codes is the way they are decoded [4]. Classical block codes are generally decoded with Maximum Likelihood like decoding. LDPC codes however are decoded iteratively using a graphical representation of their parity-check matrix.

III. Graphical Code Model As A Key To Iterative Decoding

Graphical models not only help us to describe Low Density Parity Check (LDPC) codes, but also help us to derive iterative decoding algorithms like Message Passing Algorithms [9]. Message Passing Algorithm is a kind of probability propagation algorithm that operates in the graphical model of the code. MacKay and Neal, was the first to connect Pearl's 'Belief Propagation' algorithm with coding. Here we will discuss graphical models like Bayesian Networks. In the factor graph the messages sent across the edges are probability or beliefs which can be used to compute the conditional probability of a message symbol x given the observed channel output y which is the a posteriori probability ($p(x \mid y)$).

Given a set $X = \{x_1, x_2, ..., x_n\}$ of random variables with joint probability distribution $p\{x_1, x_2, ..., x_n\}$ and graphical model attempts to express factorization of this distribution as a product of local functions(conditional probabilities) involving various subsets of random variables.

Given a directed graph $L=(V,E)$, let the parents $N(x)$ of vertex x be connected to x through directed edges, where $N(x)\in X$ .For a Bayesian Network , the joint probability distribution can be written as

$$p(x_1,x_2,....,x_n)=\prod_{i=1}^{n}p(x_i\mid N(x_i)) \tag{8}$$

If x_i has no parents i.e. $N(x_i)=\varnothing$, then $p(x_i\mid\varnothing)=p(x_i)$.

Bayesian Network can be used to describe any distribution, by the chain rule of probability we can write the joint probability distribution as

$$\begin{aligned}p(x_1,x_2,....,x_n)&=p_1(x_1)p_2(x_2\mid x_1)p_3(x_3\mid x_1,x_2)\\&\quad .....p_n(x_n\mid x_1,x_2,....,x_{n-1})\end{aligned} \tag{9}$$

Since the last factor $p_n(x_n\mid x_1,x_2,....,x_{n-1})$ contains all n variables just like the full joint distribution, which makes the computation complex.

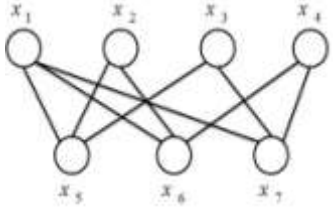

Figure 2: *Example taken from [9] Bayesian Network for the (7, 4) Hamming code*

For instance consider a Bayesian Network for the Hamming code as shown in figure 2.The joint distribution using parent-child relationship is

$$\begin{aligned}p(x_1,x_2,....,x_7)&=p_1(x_1)p_2(x_2)p_3(x_3)p_4(x_4)\\&\quad p_5(x_5\mid x_1,x_2,x_3)p_6(x_6\mid x_1,x_2,x_4)\\&\quad p_7(x_7\mid x_1,x_3,x_4)\end{aligned} \tag{10}$$

The first four factors have no parents, so it expresses the prior probabilities of x_1,x_2,x_3 and x_4 , and the last three factors express the conditional probabilities that capture the parity checks. Parity check equation is satisfied if the parent and child have even parity, i.e. $p_6(x_6\mid x_1,x_2,x_4)$ satisfies the parity check equation if x_1,x_2,x_4,x_6 have even number of ones. Factor graph for (7,4) Hamming code in Figure 3.

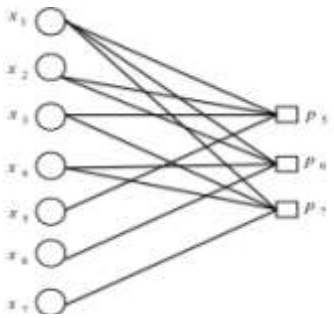

Figure 3: *Factor graph for (7,4) Hamming code*

Since the Bayesian network is a directed graph, the arrows help us to determine how each variable are influenced by other variables.

Elementary operations in a Bayesian Network involve conditioning and marginalization [12]. Conditioning involves the notion "What if y were known"

$$p(x\bigcap y)=p(x\mid y)p(y)\Rightarrow\text{conditioning} \tag{11}$$

for events x and y . Marginalization involves the notion "not wanting to know about", i.e. Marginal distribution is obtained by marginalizing over the distribution of the variables being discarded, and the discarded

variables are said to have been marginalized out. Suppose if we want to compute the Marginal distribution [10] for the joint probability distribution $p\{x_1, x_2, x_3...., x_7\}$ with respect to each variable x_i

$$p(x_i) = \sum_{\{x_j\}, j \neq i} p(x), \qquad i = 1,....n \tag{12}$$

For example to compute the marginal we have from above equation

$$p(x_1) = \sum_{x_2}\sum_{x_3}\sum_{x_4}\sum_{x_5}\sum_{x_6}\sum_{x_7} p(x_1, x_2, x_3, x_4, x_5, x_6, x_7) \tag{13}$$

For marginalization we convert the global function $p(x)$ into product of many local functions. i.e. JDF allows the factorization of the form

$$p(x) = \prod_{a=1}^{m} p_a(x_a) \tag{14}$$

Consider the example from [8] .Let $p(x_1, x_2, x_3, x_4, x_5)$ be a global function which can factorized as

$$p(x_1, x_2, x_3, x_4, x_5) = p_1(x_1)\left(\sum_{x_2} p_2(x_2)\right.$$
$$\left(\sum_{x_3} p_3(x_1, x_2, x_3).\left(\sum_{x_4} p_4(x_3, x_4)\right)\right.$$
$$\left.\left.\left(\sum_{x_5} p_5(x_3, x_5)\right)\right)\right)$$

Factor graph for above equation is shown in figure 4.Factor graph of figure 4 as tree is shown in figure 5.

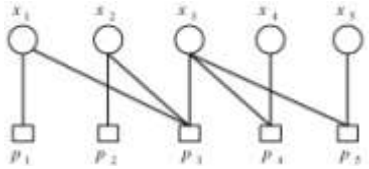

Figure 4: *Factor graph for above factorization*

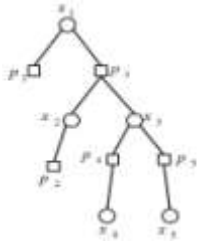

Figure 5: *Above factor graph as tree*

A. Bayesian Inference

The application of Baye's Theorem to update beliefs is known as Bayesian Inference. Bayesian Inference is used to compute the posteriori probability according to Baye's rule.

$$p(x \mid y) = p(x)p(y \mid x) \tag{15}$$

$p(x \mid y)$ is the posterior probability ,which determines the probability of the transmitted codeword given the received codeword.

$p(x)$ is the prior probability.

$p(y \mid x)$ is the likelihood, it shows how compatible is the received codeword x with the transmitted codeword y . The channel likelihood $p(y \mid x)$ represents the noise introduced by the channel.

IV. Iterative Decoding Algorithm

The class of decoding algorithm used to decode Linear Block Codes and LDPC codes are termed as Message Passing Algorithm (MPA) [4]. The reason for their name is that at each round of the algorithm messages are passed from variable nodes to check nodes, and from check nodes back to variable nodes in factor graph. The MPA is also known as iterative algorithm as message pass back and forth between the variable node and check node iteratively until result is achieved or the process is halted.

Different MPA are named for the type of message passed or the type of operation performed at the nodes [6]. Important subclass of MPA is Belief Propagation Algorithm where the messages passed along the edges are probabilities, or beliefs. More precisely, the message passed from a variable node x to a check node c is the probability that x has a certain value given the observed value of that variable node, and all the values communicated to x in the prior round from check nodes incident to x other than c. On the other hand, the message passed from c to x is the probability that x has a certain value given all the messages passed to c in the previous round from variable nodes other than x.

A. Message Passing Algorithm:

Assume a binary codeword $(x_1, x_2, ..., x_n)$ is transmitted using binary phase shift keying modulation. Then the sequence is transmitted over an additive white Gaussian noise (AWGN) channel and the received symbol is $(y_1, y_2, ..., y_n)$.Let $y = x + n$ where n is an iid vector of Gaussian random variable where each component has variance σ^2. We assume that $x_i = 0$ is transmitted as -1 and $x_i = 1$ is transmitted as $+1$ .The n code bits must satisfy all parity checks and we will use this fact to compute the posterior probability $p(x_i = b \mid S_i, y)$, $b \in \{0,1\}$and S_i is the event that all parity checks associated with x_i have been satisfied [5].

Prior to decoding, the decoder has the following: a parity check matrix H, bipartite graph, n channel outputs y .Let V_j be the set of variable nodes connected to check node c_j, $V_j \setminus i$ be the set of variable node connected to check node c_j excluding variable node x_i. C_i be the set of check nodes connected to variable node x_i, $C_i \setminus j$ be the set of check node connected to variable node x_i excluding c_j. $M_v(\sim i)$ be the message from all variable nodes except x_i . $M_c(\sim j)$ be the message from all check nodes except c_j .

The MPA for the computation of $\Pr(x_i = 1 \mid y)$ is an iterative algorithm based on factor graph. Let us imagine that variable node (v-node) represent processors of one type, check node(c-node) represent processors of another type and the edge represent message paths.

In first half of the iteration each v-node x_i processes its input message received from the channel y_i and passes its resulting output message to neighboring c-node because in first pass there is no incoming message from the c-node. Now the c-node has got new input messages and it has to check whether the check equation is satisfied. And then passes its resulting output messages to neighboring v-node using the incoming messages from all other v- nodes connected to c- node c_j excluding the information from x_i . This is shown in figure 6.The information passed concerns $\Pr(\text{check equation } c_1 \text{ is satisfied} \mid \text{input messages})$ .Note that in the figure 6 the information passed to v-node x_4 is all the information available to c-node c_1 from the neighboring v-node, excluding v-node x_4 . Such extrinsic information is computed for each connected c-node/v-node pair at each half iteration.

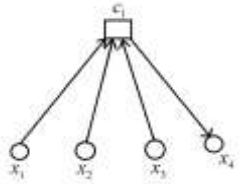

Figure 6: *Sub graph of tanner graph corresponding to row in a H matrix. The arrow indicate message passing from c-node c_1 to v-node x_4 .*

In the other half iteration, each v-node processes its input message and passes its resulting output message to neighboring c-node using channel samples and incoming messages from all other c- nodes connected to v- node x_i , excluding check node c_j . This is shown in figure 7.The information passed concerns $\Pr(x_1 = b \mid \text{input messages}),\ b \in \{0,1\}$ .Note that in the figure 7 the information passed to c-node c_3 is all the information available to v-node x_1 from the channel sample y_1 and through its neighboring c-nodes, excluding c-node c_3 .Such extrinsic information is computed for each connected v-node/c-node pair at each half iteration.

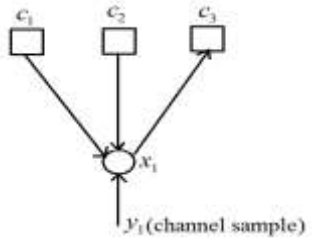

Figure 7: *Sub graph of tanner graph corresponding to column in a H matrix. The arrow indicates message passing from v-node x_1 to c-node c_3 .*

Now the c-node has got new messages from the v-node and it has to check whether the check equation is satisfied and passes all the information to v-node. And then every bit in v-node x_i is updated by using the channel information and messages from neighboring c-nodes, without excluding any c-node c_j information. After every one complete iteration it will check whether valid codeword is found or not. If the estimated codeword is valid then

$$H \otimes \hat{x} = 0$$

Then the iteration terminate otherwise continue.

V. Probability Domain Spa (Sum Product Algorithm) Decoder

Let us consider AWGN channel with x_i be the i^{th} transmitted binary value. Then the i^{th} received channel sample is $y_i = x_i + n_i$, where n_i are independent and normally distributed as $\eta(0, \sigma^2)$ .Then it is easy to show that

$$\Pr(x_i = x \mid y_i) = [1 + \exp(-2yx / \sigma^2)]^{-1} \tag{17}$$

where $x \in \{\pm 1\}$. For $i = 1, ..., n$, find probability P_i such that set $P_i = \Pr(x_i = 1 \mid y_i)$ .Initially these probabilities P_i is sent from variable node to check node as q_{ij} i.e. set $q_{ij}(0) = 1 - P_i$ and $q_{ij}(1) = P_i$ for all i, j for which $h_{ij} = 1$. Then update the check nodes and check whether the check equation is satisfied. In order to update the variable node message r_{ji} is sent from c-node to v-node.

To develop an expression for $r_{ji}(b)$, we need the following result [7]. Consider a sequence of m independent binary digits x_i , for which $\Pr(x_i = 1) = p_i$ then the probability that $\{x_i\}_{i=1}^{m}$ contains an even number of $1's$ is

$$\frac{1}{2} + \frac{1}{2}\prod_{l=i}^{m}(1 - 2p_i) \tag{18}$$

As a preliminary calculation, suppose two bits satisfy a parity check constraint $x_1 \oplus x_2 = 0$, and it is known that $p_1 = P(x_1 = 1)$ and $p_2 = P(x_2 = 1)$ .Let $q_1 = 1 - p_1$ and $q_2 = 1 - p_2$ .Then probability that the check is satisfied is

$$\left.\begin{aligned} P(x_1 \oplus x_2 = 0) &= (1-p_1)(1-p_2) + p_1 p_2 \\ &= 2p_1 p_2 - p_1 - p_2 + 1 \end{aligned}\right\} \tag{19}$$

which can be written as

$$2P(x_1 \oplus x_2 = 0) - 1 = (1-2p_1)(1-2p_2) = (q_1 - p_1)(q_2 - p_2) \tag{20}$$

Now suppose that l bits satisfy an even parity check constraint

$$x_1 \oplus x_2 \oplus ..... \oplus x_l = 0 \tag{21}$$

Then for known probabilities $\{p_1, p_2, ..., p_l\}$ corresponding to the bits $\{x_1, x_2, ..., x_l\}$. Generalizing to find the probability distribution on the binary sum $z_l = x_1 \oplus x_2 \oplus ..... \oplus x_l$

$$2P(z_l = 0) - 1 = \prod_{i=1}^{l} (1-2p_i) \tag{22}$$

Or

$$P(z_l = 0) = \frac{1}{2}(1 + \prod_{i=1}^{l} (q_i - p_i) \tag{23}$$

In view of this result, together with the correspondence $p_i \leftrightarrow q_{ij}(1)$, we have

$$r_{ji}(0) = \frac{1}{2} + \frac{1}{2} \prod_{i' \in V_j \setminus i} (1 - 2q_{i'j}(1)) \tag{24}$$

Since, when $x_i = 0$, the bits $\{x_{i'} : i' \in V_j \setminus i\}$ must contain an even number of $1's$ in order for check equation c_j to be satisfied. Clearly,

$$r_{ji}(1) = 1 - r_{ji}(0) \tag{25}$$

Half iteration of message passing for the computation of $r_{ji}(b)$ is shown in figure 8.

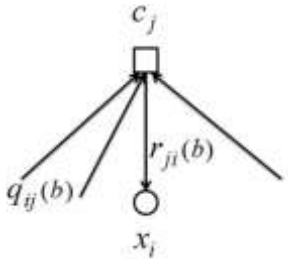

Figure 8: *Illustration of message passing half iteration for the computation of* $r_{ji}(b)$

Consider the factor graph shown in the figure 9 for the computation of $r_{21}(b)$

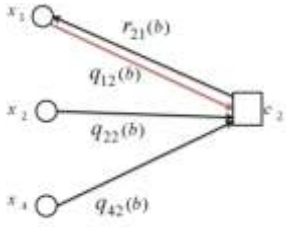

Figure 9: *Computation of* $r_{21}(b)$

In figure 9 to calculate $r_{21}(b)$ from c_2 to x_1, we consider the all v-nodes connected to c_2 excluding x_1 i.e. x_2 and x_4 .

$$r_{21}(0) = \frac{1}{2} + \frac{1}{2}(1-2q_{22}(0))\ (1-2q_{42}(0))$$

$$r_{21}(1) = 1 - r_{21}(0)$$

Then we have to update variable nodes by considering the previously calculated $r_{ji}(b)$ and channel sample and develop an expression for $q_{ij}(b)$ using the following description: APP (a-posterior probability) $p(x_i = b \mid S_i, y)$ can be rewritten using the assumption of code bit independence and Baye's rule as

$$p(x_i = b \mid S_i, y) = K\ p(y_i \mid x_i = b)\ p(S_i \mid x_i = b, y) \tag{26}$$

This equation is derived from Baye's theorem:

$$P(C \mid BA) = P(A \mid C)\ P(B \mid CA) \tag{27}$$

where K is a constant for both $b = 0,1$.The first term in equation (26) is

$$p(y_i \mid x_i = 1) = P_i = [1 + \exp(-2yx/\sigma^2)]^{-1}$$
$$\Downarrow$$
$$\text{Likelihood} \tag{28}$$

The second term in equation (26) is the probability that all parity checks connected to x_i are satisfied given y and $x_i = b$. $S_i = \{S_{0i}, S_{1i}, ...S_{mi}\}$ is a collection of events where S_{ji} is the event that j^{th} parity node connected to x_i is satisfied. Again by independence of code bits $(x_1, ..., x_n)$ this can be written as:

$$p(S_i \mid x_i = b, y) = p(S_{0i}, S_{1i}, ...S_{mi} \mid x_i = b, y)$$

$$= \prod_{j \in C_i} p(S_{ji} \mid x_i = b, y) \tag{29}$$

where $p(S_{ji} \mid x_i = b, y)$ is the probability that the j^{th} parity check connected to the bit x_i is satisfied given $x_i = b$ and y .If $b = 0$ this the probability that the code bits other than x_i connected to the j^{th} parity check have an even number of $1's$ .If $b = 1$ the other code bits must have odd parity.

$$p(S_{ji} \mid x_i = 0, y) = r_{ji}(0) = \frac{1}{2} + \frac{1}{2} \prod_{i' \in V_j \setminus i} (1 - 2q_{i'j}(1)) \tag{30}$$

From equation (26) we can write

$$q_{ij}(0) = p(x_i = 0 \mid S_i, y) \tag{31}$$

So equation (26) can be re written as

$$q_{ij}(0) = K_{ij}(1 - P_i) \prod_{j' \in C_i \setminus j} (r_{j'i}(0)) \tag{32}$$

Similarly

$$q_{ij}(1) = K_{ij} P_i \prod_{j' \in C_i \setminus j} (r_{j'i}(1)) \tag{33}$$

The constants K_{ij} are chosen to ensure that $q_{ij}(0) + q_{ij}(1) = 1$

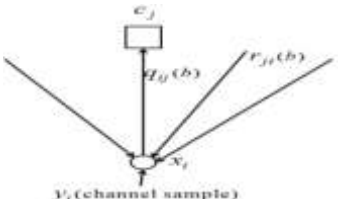

Figure 10: Illustration of message passing half iteration for the computation of $q_{ij}(b)$

Half iteration of message passing for the computation of $q_{ij}(b)$ is shown in figure 10.

Consider the factor graph shown in the figure 11 for the computation of $q_{12}(b)$

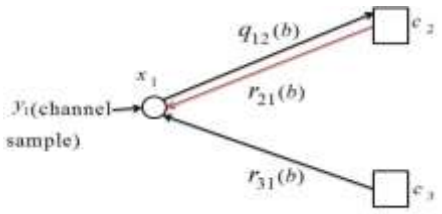

Figure 11: *Computation of* $q_{12}(b)$

In figure 11 to calculate $q_{12}(b)$ from x_1 to c_2, we consider the information from the channel and from the c-nodes connected to x_1 excluding c_2 i.e. c_3.

$$q_{12}(0) = K_{12}(1-P_1)r_{31}(0)$$

$$q_{12}(1) = K_{12}(P_1)r_{31}(1)$$

In order to fix the value for variable node we calculate $Q_i(b)$ using the information from the channel and from all the check nodes connected to x_i.

$$Q_i(0) = K_i(1-P_i)\prod_{j\in C_i} r_{ji}(0) \tag{34}$$

and

$$Q_i(1) = K_i P_i \prod_{j\in C_i} r_{ji}(1) \tag{35}$$

Where the constants K_i are chosen to ensure that $Q_i(0)+Q_i(1)=1$.

For $i = 1,2,....,n$,set

$$\hat{x}_i = \begin{cases} 1, & if\ Q_i(1) > Q_i(0) \\ 0, & else \end{cases} \tag{36}$$

If the estimated codeword is valid then,

$$\hat{x}H^T = 0 \tag{37}$$

And the iteration will stop else it will continue iterating till stopping criterion is fulfilled.

A.Probability -domain Sum Product Algorithm:

1) Compute prior-probabilities for the received vector y

$$P_i^1 = [1+\exp(-2y_i/N_0)]^{-1}$$
$$P_i^0 = 1-P_i^1$$

2) Initialization

$$q_{ij}^{\ 0} = 1-P_i$$
$$q_{ij}^{\ 1} = P_i$$

For all i, j for which $h_{ij} = 1$

3) Iteration

For $n = 1 : I_{\max}$ do

4) Horizontal Step (Check node updates)

For $j = 1 : m$ do

For $\mathrm{i} \in \mathrm{V}_j$ do

$$r_{ji}^{\ 0} = \frac{1}{2} + \frac{1}{2}\prod_{i'\in V_j\setminus i}(1-2q_{ij}^1)$$
$$r_{ji}^{\ 1} = 1-r_{ji}^{\ 0}$$

end
end

5) Vertical Step (Variable node updates)

For $i = 1:n$ do

For $j \in C_i$ do

$$q_{ij}^{\ 0} = K_{ij}(1-P_i)\prod_{j' \in C_i \setminus j}(r_{ji}^0)$$

$$q_{ij}^{\ 1} = K_{ij}P_i\prod_{j' \in C_i \setminus j}(r_{ji}^1)$$

The constants K_{ij} are chosen to ensure that $q_{ij}^{\ 0} + q_{ij}^{\ 1} = 1$

end
end

6) Marginalization

Q_i^b's are updated as

For $i = 1.....n$ do

$$Q_i^0 = K_i(1-P_i)\prod_{j \in C_i} r_{ji}^0$$

$$Q_i^1 = K_i P_i \prod_{j \in C_i} r_{ji}^1$$

Where the constants K_i are chosen to ensure that $Q_i^0 + Q_i^1 = 1.$

7) Verification

For $i = 1.....n$ do

If $(Q_i^1 > Q_i^0)$ then $\hat{x}_i = 1$

else $\hat{x}_i = 0$

end

8) If $\hat{x}H^T = 0$ or the number of iterations equals the maximum limit,

stop

else go to step 4.

end

VI. Compressive Sensing

Compress sensing concept asserts that a sparse signal can be reconstructed from far fewer samples or measurements than that is specified by the Nyquist theorem. This section discusses how Message Passing Algorithm in its simplified form is applied to compressive sensing reconstruction.

Consider a signal $f \in \mathbb{R}^N$, acquired via n-linear measurements,

$$y_j = \langle f, \varphi_j \rangle, \quad j = 1,.....n \tag{38}$$

we try to correlate the signal we wish to acquire with the measurement vector $\varphi_j \in \mathbb{R}^N$.CS accomplishes reconstruction of an $N-$dimensional signal f from $n-$measurements, where $n << N$ .Equation (38) can also be written as matrix product,

$$y = \Phi f \tag{39}$$

where $y \in \mathbb{R}^n$ is called the measurement vector and $\Phi \in \mathbb{R}^{n \times N}$ is the sensing or measurement matrix with vectors $\varphi_1^*,.....\varphi_n^*$ stacked as rows. In this undersampled situation, CS relies on the fact that most of the signals can be approximated in some convenient basis. We can approximate f with small number of non-zero coefficients by finding a suitable orthonormal basis[16].

Using linear algebraic notations, $f \in \mathbb{R}^N$ can be expressed as a linear combination of the basis vectors $\Psi = [\psi_1 \psi_2 ... \psi_N]$

$$f = \sum_{i=1}^{N} x_i \psi_i \tag{40}$$

where $x_i = \langle f, \psi_i \rangle$ are the sparse coefficients of f in the basis. Equation (40) can be rewritten as a matrix product,

$$f = \Psi x \tag{41}$$

where $x \in \mathbb{R}^N$ is the vector of coefficients, $\Psi \in \mathbb{R}^{N \times N}$ is the matrix with basis vectors $\psi_1, \psi_2, ..., \psi_N$ as columns.

The k-sparse approximation of the signal f is obtained by sorting the coefficients of x in descending order and keeping the largest k elements, while setting the rest of the elements to zero. $x_k \in \mathbb{R}^N$ denotes the vector containing only the largest k coefficients of x.The approximation $f_k \in \mathbb{R}^N$ of f is obtained as

$$f_k = \psi x_k \tag{42}$$

A signal is said to be *compressible* if the sorted magnitudes of x_i decay quickly. Compressible signals can be well approximated such that for $k << N$ the error $\|f - f_k\|_{\ell_2} = \|x - x_k\|_{\ell_2}$ is small. CS attempts to optimize the acquisition process. Compressive sensing allow us to take small amount $(n << N)$ of linear and *non-adaptive* measurements as in equation (39)

We recover the signal by determining the sparsest vector $\tilde{x} \in \mathbb{R}^N$ that is consistent with the measurements y. We get the sparsest solution by solving the *Basis Pursuit* or ℓ_1*-minimization problem*.

$$\begin{aligned} &\min \|\tilde{x}\|_{\ell_1} \\ &\text{s.t } y_k = \langle \varphi_k, \Psi \tilde{x} \rangle, \quad k = 1, ..., n \end{aligned} \tag{43}$$

where $\|x\|_{\ell_1} := \sum_i |x_i|$. One sufficient condition for the signal recovery via ℓ_1-minimization is that measurement matrix Φ should satisfy the Restricted Isometry Property (RIP)[11] i.e. it should approximately preserve the Euclidean length of k-sparse signals, i.e. $\|\Phi x_k\|_{\ell_2}^2 \approx \|x_k\|_{\ell_2}^2$.

A random measurement matrix Φ, efficiently captures the information of a sparse signal with few measurements. Sparse error correcting codes such as LDPC codes recommends the use of sparse compressed sensing matrices to encode the signals [18].

A. Message Passing for Compressive Sensing

Message Passing Algorithm can be applied for compressive sensing reconstruction of sparse signal. Consider a CS estimation problem [11] shown in figure.12.Here $X \in \mathbb{R}^N$ is the vector to be estimated. $Z \in \mathbb{R}^n$ is the measurements and $Y \in \mathbb{R}^n$ is the noisy measurements.

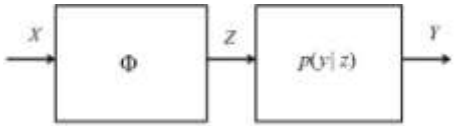

Figure.12:*Generalized CS estimation problem. Here $X \in \mathbb{R}^N$ is the vector to be estimated. $Z \in \mathbb{R}^n$ is the measurements and $Y \in \mathbb{R}^n$ is the noisy measurements.*

When a random input vector X with i.i.d. components is transformed by a matrix Φ, we get the measurements Z.These measurements when transmitted over a noisy channel gets corrupted. Here the noise is characterized as a channel, which is represented as a conditional probability $p(y \mid z)$.. Goal is to estimate X from Y given the matrix Φ, the prior $p_X(x)$, and the noise $p(y \mid z)$ [11].

To find the posterior distribution i.e. the probability of random vector X given the measurements Y is,

$$p(x \mid y) = \prod_{i=1}^{N} p(x_i) \prod_{j=1}^{n} p(y_j \mid z_j) \tag{44}$$

where $z_j = (\Phi x)_j$ .Estimation of x_i is possible by the marginalization of $p(x \mid y)$ through the following Message Passing Algorithm rules.

$$q_{ij}^{t+1}(x_i) \propto p(x_i) \prod_{j' \in C_i \setminus j} r_{j'i}^{t}(x_i)$$

$$r_{ji}^{t}(x_i) \propto \int p(y_j \mid z_j) \prod_{i' \in V_j \setminus i} q_{ij}^{t}(x_i) dx \quad (45)$$

The integration is performed over all the elements of x except x_i . The probability densities of the messages exchanged via MPA is tracked by density evolution (DE).DE allows to predict the condition for the successful decoding.

In the CS framework X takes value in $\mathbb{R}^N$,so it is difficult to keep track of probability densities across the iterations of MPA.MPA can be simplified through various Gaussian approximations[11].One such approximation to MPA is *Approximate Message Passing*(AMP) Algorithm[17]. Here we can track the evolution of mean square error(MSE) from iteration to iteration through recursive equations called State Evolution (SE)[17], which provide reliable estimates of the reconstruction error and predicts that when it has a unique fixed point the AMP algorithm will obtain minimum MSE estimates of the signal[11].

In AMP, since the messages exchanged are Gaussian we need to track only the means and variances through the factor graph[17].

Approximate Message Passing algorithm (AMP) exhibits low computational complexity of iterative thresholding algorithms and reconstruction power of the Basis Pursuit. Algorithm starts with $x^0 = 0$ and then proceeds according to the following iteration,

$$x^{t+1} = \eta_t\left(\sum_a A^* z^t + x^t\right), \quad (46)$$

$$z^t = y - Ax^t + \frac{1}{\delta} z^{t-1} \left\langle \eta'_{t-1}(A^* z^{t-1} + x^{t-1}) \right\rangle \quad (47)$$

where η_t is the soft thresholding function , $x^t \in \mathbb{R}^N$ is the current estimate. $z^t \in \mathbb{R}^n$ is the residual. A^* is the transpose of the matrix A.

It is similar to iterative thresholding algorithm which is given by

$$x^{t+1} = \eta_t(A^* z^t + x^t)$$

$$z^t = y - Ax^t \quad (48)$$

The only difference is the second term included in the right hand side of the equation (47).

VII. Conclusion

In this paper, we tried to unify various papers associated with message passing algorithm. We have reviewed the fundamentals of error correcting codes, factor graphs, Bayesian networks, sum-product algorithm and compressive sensing. Here the joint distribution function is factorized as a product of local functions and then marginalized by message passing algorithm. Bayesian Inference is used to compute the posterior probability for channel decoding. Then we have the probability domain version of sum product algorithm that computes the a-posteriori probabilities (APPs).We also tried to give a small insight to Approximate Message Passing Algorithm which was derived from MPA for compressive sensing reconstruction.

References

[1] R.W. Hamming, "Error Detecting and Error Correcting Codes", Bell system, Technical 29(1950), pp.147-160.

[2] R. Gallager, "Low Density Parity Check codes", *IEEE Trans. Information Theory* 8 (1962) no-1, pp.21-28.

[3] Yunghsiang S. Han, "Introduction to Binary Linear Block Codes", Graduate Institute of Communication Engineering, National Taipei University, pp.8-17.

[4] Sarah J. Johnson, "Introducing Low Density Parity Check Codes", University of Newcastle, Australia.

[5] William E. Ryan, " An Introduction to LDPC codes", Department of Electrical and Computer Engineering, University of Arizona, August 2003

[6] Amin Shokrollahi, "LDPC codes- An Introduction", Digital Fountain, Inc. April 2 2003.

[7] Steve,"Basic Introduction to LDPC",University of Clifornia,March 2005

[8] H.A. Loeliger, "An Introduction to Factor Graphs", *IEEE Signal Proc. Magazine* (2004), pp.28-41.

[9] F.R. Kschischang, B.J. Frey, and H.A. Loeliger, "Factor Graphs and the Sum Product Algorithm", *IEEE Trans. Information Theory* 47 (2001), pp.498-519.

[10] Frank R.Kschischang, Brenden J.Frey, "Iterative Decoding of Compound Codes by Probability Propagation in Graphical Models", *IEEE Journal on selected areas in communication,* vol.16,no.2 February 1998,pp.219-230

[11] Ulugbek Kamilov, "Optimal Quantization for Sparse Reconstruction with Relaxed Belief Propagation", Master's thesis ,MIT ,March 2011

[12] Nana Traore, Shashi Kant and Tobias Lindstrm Jensen, "Message Passing Algorithm and Linear Programming Decoding for LDPC and Linear Block Codes", Institute of Electronic Systems Signal and Information Processing in Communications, Aalborg University, January 2007.

[13] Matthias Seeger, "Bayesian Modeling in Machine Learning: A Tutorial Review", Probabilistic Machine Learning and Medical Image Processing, Saarland University, March 2009.

[14] Tinoosh Mohsenin, "Algorithm and Architecture for Efficient Low Density Parity Check Decoder Hardware", Ph.D. thesis, University of California, 2010.
[15] Alexios Balatsoukas-Stimming, "Analysis and Design of LDPC codes for the Relay Channel", Thesis, Technical University of Crete,Department of Electronic and Computer Engineering, February 2010.
[16] Emmanuel J. Cands and Michael B. Wakin,"An Introduction To Compressive Sampling",IEEE signal processing magazine,March 2008
[17] David L. Donohoa, Arian Malekib, and Andrea Montanaria,"Messagepassing algorithms for compressed sensing",Proc. Natl. Acad. Sci. 106,2009 ,18914-18919
[18] D. Baron, S. Sarvotham, and R. G. Baraniuk, Bayesian Compressive Sensing via Belief Propagation, accepted to IEEE Transactions on Signal Processing,2009