Constraint Relaxation and Annealed Belief Propagation for Binary Networks

Yen-Chih Chen and Yu T. Su
Department of Communications Engineering
National Chiao Tung University
1001 Dar Hsueh Road, Hsinchu 30056, TAIWAN
joechen.cm88g@nctu.edu.tw, ytsu@mail.nctu.edu.tw

***Abstract*— In this paper, we propose two novel generalized belief propagation (BP) algorithms to improve the convergence behavior of the conventional BP algorithm. By incorporating a dynamic temperature into the free energy formulation, message passing is performed on a dynamic surface of energy cost. The proposed cooling process helps BP converge to a stable fixed point with a lower energy value that leads to better estimations. For decoding turbo-like error correcting codes, we adopt a parametric Gaussian approximation to relax the binary parity check constraints and generalize the conventional binary networks as well. Both the computational complexity and the convergence rate of the proposed annealed BP algorithms are almost the same as those of the conventional BP algorithm. Simulated performance of the proposed algorithms when they are used to decode a low density parity check (LDPC) code and the (23,12) Golay code is presented to validate our proposals.**

I. Introduction

Belief propagation (BP) algorithms have been recognized as efficient and powerful inference tools for applications in computer vision, artificial intelligence, error-correcting codes, and digital communications. The BP algorithms, e.g. BCJR for turbo decoding and sum-product algorithm (SPA) for LDPC codes, have received much attention for their near Shannon-limit performance. Recent studies show that fixed points of BP algorithms correspond to the stationary points of the Bethe approximation of the free energy for a factor graph [1]. However, BP algorithms do not promise convergence when the associated graph contains cycles. Even if they converge, they are only guaranteed to converge to a local minimum of the Bethe free energy [1]. Some new algorithms attempt to avoid local minimums by directly minimizing the free energy [2], [3] through using the conventional optimization schemes. These algorithms, however, are often much more complex and yield much slower convergence rate.

Inspired by the deterministic annealing methods used in optimization problems [4], we propose two annealed BP (ABP) algorithms that alleviate ill-convergence. By incorporating a temperature parameter into the formulation of the free energy, one can start an inference task at a higher temperature to have a smoother energy cost surface. According to the result from the statistical physics, there is only one local minimum of the free energy function when the temperature is above a critical value [1]. Thus, a BP algorithm can always converge to the unique minimum at that temperature. However, inference results at high temperatures may be poor due to the unfaithful modelling of the real free energy. On the other hand, there may be more than one local minimum at low temperatures. When searching on the energy cost surface at low temperatures, one can easily get trapped in one of these local extreme points. But if the global minimal free energy can be located, the corresponding inference result is much more accurate as a result of more accurate modelling of the free energy function. Hence, the idea is to start an inference at a high temperature and smoothly decrease the temperature using some cooling strategies to track the minimum point. Hopefully, the algorithm will stop at the global minimum and gives accurate estimations when the temperature approaches to zero. Such a cooling strategy prevents a BP algorithm from sticking to a local minimum, and helps the algorithms to converge to the global minimum with a larger probability.

The paper is organized as follows. In section II, we formulate the free energy of a belief network by taking into consideration the temperature effect and give the corresponding BP equations. Section III develops a Gaussian approximation scheme to relax the parity check constraints and generalizes the conventional binary networks that are commonly used when decoding error correcting codes. Section IV discusses cooling schedules of the proposed ABP algorithms. In section V, we provide simulation results for ABP-based Golay and LDPC decoders and give some concluding remarks.

II. Minimum Free Energy under Bethe Approximation

It is now well known that the conventional SPA can be derived from the minimization of Bethe free energy when the temperature equals to one [1]. We begin with the Bethe free energy formulation that retains the temperature parameter. Consider a factor graph whose Bethe free energy is given by

$$F_{\text{Bethe}} = U - TH. \tag{1}$$

U in (1) is the variational average energy

$$U = \langle E_\mu \rangle_{\mathbf{x}_\mu} + \langle E_k \rangle_{x_k}, \tag{2}$$

where

$$\langle E_\mu \rangle_{\mathbf{x}_\mu} \stackrel{def}{=} -\sum_{\mu}\sum_{\mathbf{x}_\mu} b_\mu(\mathbf{x}_\mu)\ln f_\mu(\mathbf{x}_\mu) \quad (3)$$

$$\langle E_k \rangle_{x_k} \stackrel{def}{=} -\sum_{k}\sum_{x_k} b_k(x_k)\ln g_k(x_k). \quad (4)$$

are the average energy functions associated with factor node μ and variable node k respectively. $f_\mu(\mathbf{x}_\mu)$ denotes the factor function with $\mathbf{x}_\mu \stackrel{def}{=} \{x_k \mid x_k \in \text{neighbors of } \mu\}$. $g_k(x_k)$ is the local function associated with variable node x_k.

The second term H in (1) denotes the variational entropy under the Bethe approximation,

$$H = -\sum_{\mu}\sum_{\mathbf{x}_\mu} b_\mu(\mathbf{x}_\mu)\ln b_\mu(\mathbf{x}_\mu) - \sum_{k} c_k \sum_{\mathbf{x}_k} b_k(x_k)\ln b_k(x_k). \quad (5)$$

In (5), c_k is defined as $1-$(numbers of neighbors of x_k).

Parameter T in (1) mimics the temperature effect in the field of statistic physics. Under the marginalization and normalization constraints

$$\sum_{\mathbf{x}_\mu \setminus x_k} b_\mu(\mathbf{x}_\mu) = b_k(x_k) \quad (6)$$

$$\sum_{\mathbf{x}_\mu} b_\mu(\mathbf{x}_\mu) = 1 \quad (7)$$

$$\sum_{x_k} b_k(x_k) = 1, \quad (8)$$

one can derive a BP algorithm by minimizing the the parametric Bethe free energy given in (1). Define $m_{\mu k}$ as the message passed from factor node μ to variable node k, and $n_{k\mu}$ as that passed from variable node k to factor node μ, one can prove

$$m_{\mu k}(x_k) \propto \sum_{\mathbf{x}_\mu \setminus x_k} \left\{ f_\mu^\beta(\mathbf{x}_\mu) \prod_{j \in \mathcal{N}(\mu)\setminus k} n_{\mu j}(x_j) \right\} \quad (9)$$

and message $n_{k\mu}$ is updated by

$$n_{k\mu}(x_k) = g_k^\beta(x_k) \cdot \prod_{\xi \neq \mu, \xi \in \mathcal{N}(k)} m_{\xi k}(x_k), \quad (10)$$

where $\beta \stackrel{def}{=} 1/T$. Equations (9) and (10) degenerate to those of the conventional BP algorithm when $\beta = 1$.

On the other hand, the belief of variable node x_k can be estimated by

$$b_k^{(*)}(x_k) \propto g_k^\beta(x_k) \prod_{\mu \in \mathcal{N}(k)} m_{\mu k}. \quad (11)$$

(9) and (10) indicate that the BP algorithm can be generalized by incorporating a temperature parameter T, and replacing $f_\mu(\mathbf{x}_\mu)$ and $g_k(x_k)$ by their βth powers, $f_\mu^\beta(\mathbf{x}_\mu)$ and $g_k^\beta(x_k)$.

III. Annealed Belief Propagation in Binary Networks

As the first step to implement an ABP algorithm is to parameterize the original local and factor functions with an auxiliary parameter $\beta = 1/T$. In the following, a parametric approach for binary networks is developed. Such a parametric relaxation scheme not only works for the proposed annealed BP algorithms but also provides a new degree of freedom for network and code constructions. It can also be used for applications in signal processing, speech processing and pattern recognition, etc.

A. Local Function Modelling

For an LDPC coded BPSK waveform in a white Gaussian noise channel, the local likelihood function $g_k(x_k)$ associated with each code bit node x_k is modelled as

$$g_k(x_k) = p(y_k|x_k) = \sqrt{\frac{1}{2\pi}} \exp\left[\frac{-(y_k - x_k)^2}{2\sigma^2}\right]. \quad (12)$$

where x_k represents a transmitted bit and y_k is the corresponding receiver matched filter output sample. The likelihood ratio (LR) of x_k is

$$l_k = \frac{g_k(x_k = -1)}{g_k(x_k = 1)} = \exp\left(-\frac{2y_k}{\sigma^2}\right). \quad (13)$$

and the generalized likelihood ratio (GLR) becomes

$$l_k^\beta = \frac{g_k^\beta(x_k = -1)}{g_k^\beta(x_k = 1)} = \exp\left(-\beta\frac{2y_k}{\sigma^2}\right). \quad (14)$$

which converges to three limiting points

$$l_k^\beta = \begin{cases} 1, & \text{when } \beta \to 0 \\ l_k, & \text{when } \beta \to 1 \\ \phi(y_k < 0)\cdot\infty, & \text{when } \beta \to +\infty,\ y_k \neq 0, \end{cases} \quad (15)$$

where $\phi(\cdot)$ denotes the indicator function,

$$\phi(A) = \begin{cases} 1, & \text{if event } A \text{ is true,} \\ 0, & \text{if event } A \text{ is false.} \end{cases} \quad (16)$$

(14) generalizes the statistical modelling of a variable function so that the conventional LR l_k becomes the special case $T \to 1$. On the other hand, as $T \to 0$, i.e., when the code bit node is in a very low temperature state, l_k^β approaches an impulse function and renders a deterministic decision of y_k. Otherwise, when $T \to +\infty$, l_k becomes a constant whence every code bit node is equally probable. Thus, by changing the temperature parameter T, one actually models the local LR l_k^β in different quantization precisions.

B. Factor Function Modelling

When decoding an error correcting code, a factor function $f_\mu(\mathbf{x}_\mu)$ can be described by an indicator function of parity check equation associated with vector $\mathbf{x}_\mu$, i.e.,

$$f_\mu(\mathbf{x}_\mu) = \delta\left(\{\oplus_{j \in \mathcal{N}(\mu)} c_j\}\right), \quad (17)$$

where $\oplus$ denotes exclusive-or, $c_j = \begin{cases} 0, & \text{if } x_j = 1 \\ 1, & \text{if } x_j = -1 \end{cases}$ and

$$\delta(y) = \begin{cases} 1, & \text{if } y = 0 \\ 0, & \text{otherwise} \end{cases}$$

Since $\delta(\cdot) = \delta^{\beta}(\cdot)$ for $\beta > 0$, it is of no use to replace the conventional factor function $f_\mu(\mathbf{x}_\mu)$ by $f_\mu^\beta(\mathbf{x}_\mu)$.

An alternative way to parameterize the factor function is to approximate the delta function by the limit of a Gaussian density of a real random variable, i.e.,

$$\delta(y) = \lim_{\beta\to\infty} \sqrt{\frac{\beta}{2\pi}} \exp\left(-\frac{\beta}{2}y^2\right) \stackrel{def}{=} \lim_{\beta\to\infty} \delta_\beta(y) \qquad (18)$$

With the approximation $\delta(y) \approx \delta_\beta(y)$, one naturally incorporates the temperature effect β into the modelling of the factor function $f_\mu^\beta(\mathbf{x}_\mu)$. When $0 < \beta < +\infty$, $\delta_\beta(y)$ has non zero value for any finite y. One thus relaxes a "deterministic" parity check constraint $f_\mu(\mathbf{x}_\mu) = \delta(y)$ to a "stochastic" parity check constraint $f_\mu^\beta(\mathbf{x}_\mu) = \delta_\beta(y)$.

From (9) and (18), one obtains (20) and (21) in the next page, representing the message passed from $f_\mu^\beta(\mathbf{x}_\mu)$ to x_k, and the LR of $m_{\mu k}(x_k)$. There are also a number of limit points when β changes from 0^+ to $+\infty$:

$$l_{m_{\mu k,\beta}} \to \begin{cases} 1, & \text{when } \beta \to 0^+ \\ l_{m_{\mu k}}. & \text{when } \beta \to +\infty, \end{cases} \qquad (19)$$

where $l_{m_{\mu k}}$ given in (22) is exactly the LR when the factor node is a delta function, i.e., $f_\mu(\mathbf{x}_\mu) = \delta(\mathbf{x}_\mu)$.

Similar to the case of local function modelling, the probabilities of $x_k = 1$ and $x_k = -1$ are equally likely when the temperature is high. In the following, we give an example to elaborate the proposed idea of "stochastic" parity check node.

1) Example: Consider a factor node $f_\mu^\beta(\mathbf{x}_\mu)$ associated with the parity check equation

$$c_1 \oplus c_2 \oplus c_3 = 0. \qquad (23)$$

Following (20), one has

$$\begin{aligned} m_{\mu1,\beta}(x_1 = 1) \;\propto\; & n_{2\mu}(x_2 = -1)n_{3\mu}(x_3 = -1)e^{-\frac{\beta}{2}\cdot 0} \\ +\; & n_{2\mu}(x_2 = -1)n_{3\mu}(x_3 = 1)e^{-\frac{\beta}{2}\cdot 1} \\ +\; & n_{2\mu}(x_2 = 1)n_{3\mu}(x_3 = -1)e^{-\frac{\beta}{2}\cdot 1} \\ +\; & n_{2\mu}(x_2 = 1)n_{3\mu}(x_3 = 1)e^{-\frac{\beta}{2}\cdot 0} \qquad (24) \\ m_{\mu1,\beta}(x_1 = -1) \;\propto\; & n_{2\mu}(x_2 = -1)n_{3\mu}(x_3 = -1)e^{-\frac{\beta}{2}\cdot 1} \\ +\; & n_{2\mu}(x_2 = -1)n_{3\mu}(x_3 = 1)e^{-\frac{\beta}{2}\cdot 0} \\ +\; & n_{2\mu}(x_2 = 1)n_{3\mu}(x_3 = -1)e^{-\frac{\beta}{2}\cdot 0} \\ +\; & n_{2\mu}(x_2 = 1)n_{3\mu}(x_3 = 1)e^{-\frac{\beta}{2}\cdot 1}. \qquad (25) \end{aligned}$$

(24) and (25) show that, when $\beta \neq +\infty$, there is a non-zero probability $e^{-\frac{\beta}{2}}$ associated with the terms that disappear in the calculation of $m_{\mu1}$ in the deterministic case. This nonzero probability defines "stochastic" parity checks, i.e., one allows the existence of non-zero parity check value with a specified probability.

It is interesting to note that as one decreases the temperature T from $+\infty$ to 0^+, a stochastic parity check will reach two limit states. First, when $T = +\infty$ (i.e., $\beta = 0^+$), both $e^{-\frac{\beta}{2}\cdot 0}$ and $e^{-\frac{\beta}{2}\cdot 1}$ equal to 1 such that $m_{\mu1}(x_k = 1) = m_{\mu1}(x_k = -1)$. Hence, the messages provided by check node μ at $T = +\infty$ are equally probable and one just cannot obtain any information about bit node x_1 from f_μ^β.

On the other hand, as $T = 0^+$ (i.e., $\beta = +\infty$), $e^{-\frac{\beta}{2}\cdot 0} = 1$ and $e^{-\frac{\beta}{2}\cdot 1} = 0$. Message $m_{\mu1,\beta}(x_k)$ will become

$$\begin{aligned} m_{\mu1,\beta=0}(x_1 = 1) \;\propto\; & n_{2\mu}(x_2 = 1)n_{3\mu}(x_3 = 1) \\ +\; & n_{2\mu}(x_2 = -1)n_{3\mu}(x_3 = -1) \\ m_{\mu1,\beta=0}(x_1 = -1) \;\propto\; & n_{2\mu}(x_2 = -1)n_{3\mu}(x_3 = 1) \\ +\; & n_{2\mu}(x_2 = 1)n_{3\mu}(x_3 = -1), \end{aligned}$$

which gives exactly the deterministic parity check function. Hence, as $T \to 0$, the stochastic parity check will approach the deterministic case, which has been expected by (18).

IV. Annealed Belief Propagation Algorithm

The basic principle of annealing type algorithms is to start an inference task at a high temperature and iteratively decrease the temperature according to some chosen annealing schedules. In the literatures, there are two main categories when applying the annealing type algorithms. The first category, simulated annealing (SA), based on the Markov Chain Monte Carlo (MCMC) method, is quite useful in non-convex optimization problems. It generates a sequence of random walks to reach a new state with an acceptance probability that depends on the current state. SA directly simulates the system dynamics and can theoretically converge to the global optimum value. But convergence is only guaranteed under a proper annealing schedule that starts at a high enough initial temperature. Exponential execution time is needed for SA to converge to the global optimum, making it an impractical option for many real-time applications.

The second category, deterministic annealing (DA), has a deterministic processing time. Instead of simulating the stochastic dynamics of the system, DA directly minimizes the expected free energy while avoiding many poor local minima of the cost function. According to [4], the DA method performs annealing as it maintains the free energy at the thermal equilibrium while gradually lowering the temperature. Since the processing time of DA is deterministic and controllable, DA is a candidate solution if there is a fixed processing delay constraint. In fact it has already been successfully employed to solve optimization problems in source coding, pattern recognition, pattern classification, and many other fields.

We use DA to anneal the belief network when decoding an error correcting code for a number of reasons. First, a communication system usually has constraints on the decoding delay in order to satisfy the total system latency requirement. Second, DA combined with the proposed ABP algorithms needs only minor modifications of the conventional BP. In fact, ABP generalizes BP while keeping most of the implementation details intact, including the message passing schedule, the way

$$m_{\mu k,\beta}(x_k) \propto \sum_{\substack{\{o_j\} \\ j\in\mathcal{N}(\mu),j\neq k}} \left\{ \exp\left[-\frac{\beta}{2} \left(\left(\oplus_{j\in\mathcal{N}(\mu),j\neq k} c_j \right) \oplus c_k \right)^2 \right] \prod_{j\in\mathcal{N}(\mu),j\neq k} n_{j\mu}(x_k) \right\}, \tag{20}$$

$$l_{m_{\mu k,\beta}} \stackrel{def}{=} \frac{m_{\mu k,\beta}(x_k=-1)}{m_{\mu k,\beta}(x_k=1)} = \frac{\sum_{\substack{\{o_j\} \\ j\in\mathcal{N}(\mu),j\neq k}} \left\{ \exp\left[-\frac{\beta}{2} \left(\left(\oplus_{j\in\mathcal{N}(\mu),j\neq k} c_j \right) \oplus 1 \right)^2 \right] \prod_{j\in\mathcal{N}(\mu),j\neq k} n_{j\mu}(x_k) \right\}}{\sum_{\substack{\{o_j\} \\ j\in\mathcal{N}(\mu),j\neq k}} \left\{ \exp\left[-\frac{\beta}{2} \left(\left(\oplus_{j\in\mathcal{N}(\mu),j\neq k} c_j \right) \oplus 0 \right)^2 \right] \prod_{j\in\mathcal{N}(\mu),j\neq k} n_{j\mu}(x_k) \right\}} \tag{21}$$

$$l_{m_{\mu k}} \stackrel{def}{=} \frac{\sum_{\substack{\{o_j\} \\ j\in\mathcal{N}(\mu),j\neq k}} \left\{ \delta\left[\left(\left(\oplus_{j\in\mathcal{N}(\mu),j\neq k} c_j \right) \oplus 1 \right) \right] \prod_{j\in\mathcal{N}(\mu),j\neq k} n_{j\mu}(x_k) \right\}}{\sum_{\substack{\{o_j\} \\ j\in\mathcal{N}(\mu),j\neq k}} \left\{ \delta\left[\left(\left(\oplus_{j\in\mathcal{N}(\mu),j\neq k} c_j \right) \oplus 0 \right) \right] \prod_{j\in\mathcal{N}(\mu),j\neq k} n_{j\mu}(x_k) \right\}}. \tag{22}$$

that message exchanges, or even the total iteration numbers. We summarize below a generic ABP algorithm.

1) Initialize :
   Set the initial temperature T_{init} and the minimum temperature T_{min}.
   Set the maximum iteration number i_{max} and let the iteration index $i = 1$ with temperature index $T_i = T_{init}$ and $\beta_i = \frac{1}{T_i}$.
2) Update :
   Calculate the local function $g_k^{\beta_i}(x_k)$ of variable node x_k for all k.
   Calculate the factor function $f_\mu^{\beta_i}(\mathbf{x}_\mu)$ of the μ-th factor node for all $\mathbf{x}_\mu$.
   Calculate messages $m_{\mu k,\beta_i}(x_k)$ and $n_{k\mu,\beta_i}(x_k)$ for all $\mu,\ k$.
3) Check Temperature : If $T_i \leq T_{\min}$, set $T_i = 0$.
4) Cooling Step : Calculate T_{i+1} and β_{i+1} by decreasing T_i according to the specific annealing schedule.
5) Check Status : Set $i = i + 1$. Stop the algorithm when $i > i_{\max}$ or some convergent check is passed.
6) Go to 2).

Following the basic framework of ABP, we propose two types of ABP algorithms: local function annealing (LFA) and factor function annealing (FFA).

A. Local Function Annealing

Local function annealing is also referred to as code node annealing when applying ABP to decode an error-correcting code. For LDPC codes, local functions of code nodes are set to the generalized Gaussian model described in (14) Section III-A. The factor nodes, however, keep the deterministic definition, which means that we still use deterministic parity checks for LDPC codes when applying local function annealing. In this case, the exchanged message $m_{\mu k,\beta_i}$ is identical to $m_{\mu k,0}$ and $n_{k\mu,\beta_i} = n_{k\mu,0}$. Local function annealing follows the generic framework of ABP which begins with a high temperature, performs message exchange, checks the convergence and then decreases the temperature. Since the way message exchanges at factor nodes remains the same for LFA and BP, the only difference between them is that the former changes the local functions during iterations, while the latter keeps them unchanged. Because LFA only modifies the exponent of local function, only one additional multiplication per code bit is needed according to (14) at every update period. Thus, the computational complexity of LFA is almost the same as conventional BP.

Based on (15), we propose two rules for LFA scheduling. The first rule requires that the initial temperature be set high enough to avoid being trapped in a local minimum at the beginning. The second rule necessitates the lower limit of temperature be bounded below by one to make the final state equal to the observed likelihood. An empirical annealing schedule which satisfies both rules is

$$T_{local,(j+1)\eta} = T_{local,init}^{\gamma^{j+1}} = T_{local,j\eta}^{\gamma}, \tag{26}$$

$$T_{local,i} = T_{local,j\eta},\ j\eta \leq i < (j+1)\eta. \tag{27}$$

where j is the update index and $0 < \gamma \leq 1$ is a constant used to control the annealing rate. The initial temperature is defined as $T_0 \stackrel{def}{=} T_{local,init}$, η denotes the update period of the annealing process, and i is the iteration index of the ABP algorithm. The parameter η fixes the temperature $T_i = T_{local,j\eta}$ for $j\eta \leq i < (j+1)\eta$ to ensure that the ABP algorithm converges to a lower energy state at that temperature. When γ equals 1, $T_{local,i} = T_{local,init}$ for any i. For small γ, $T_{local,i}$ converges to 1, the low temperature limit, in just a few iterations.

B. Factor Function Annealing

Factor function annealing is also called check node annealing when applying an ABP algorithm to decode an error correcting code. The reason for the name "factor function annealing" is that we only perform annealing on the factor nodes. For decoding an LDPC code, we replace the deterministic parity check function by the stochastic one described in Section III-B while the local function $g_k(x_k)$ is kept constant,

$$g_k(x_k) = p(y_k|x_k) = \sqrt{\frac{1}{2\pi}} \exp\left(\frac{-(y_k - x_k)^2}{2\sigma^2} \right). \tag{28}$$

According to (19), the generalized parity check function reaches a limit as $T \to +\infty$ or $T \to 0$. Hence, one suggests the following geometric schedule for FFA

$$T_{factor,(j+1)\eta} = \alpha^{j+1} T_{factor,init} = \alpha T_{factor,j\eta} \quad (29)$$

$$T_{factor,i} = T_{factor,j\eta}, \; j\eta \le i < (j+1)\eta \quad (30)$$

The definitions of j, η, i are the same as the LFA case. The initial temperature is $T_0 \stackrel{def}{=} T_{factor,init}$ and $0 < \alpha \le 1$ is a scaling constant that controls the annealing rate. When $\alpha = 1$, temperature $T_{factor,i}$ equals $T_{factor,init}$ for all iteration. On the other hand, $T_{factor,i}$ approaches 0 in few iterations when α is small. Empirically, the geometric scaling factor α is set between 0.8 and 0.99 [5]. Both LFA and FFA are stopped when the maximum iteration limit is reached or all the deterministic parity check functions are satisfied.

Since the combinations of incoming information at every check node are the same for both FFA and the conventional BP, only two extra additions and four additional multiplications are needed for each check node compared with the conventional BP. Hence, the complexity of FFA is only slightly higher than the original BP, and can be reduced by limiting the lowest temperature of the annealing process.

V. Numerical Results and Conclusion

Simulated behaviors of the proposed algorithms when decoding an LDPC code and the (23,12) Golay code is reported in this section. The latter code has cycles of length 4 that prevent the BP algorithm from converging to the maximum likelihood decoding (MLD) performance [6]. Fig. 1 shows that, for the Golay code, the simplest LFA offers the best frame error rate (FER) performance although the performance of FFA is only slightly worse. Also, we find that for some codes, FFA performs better than LFA.

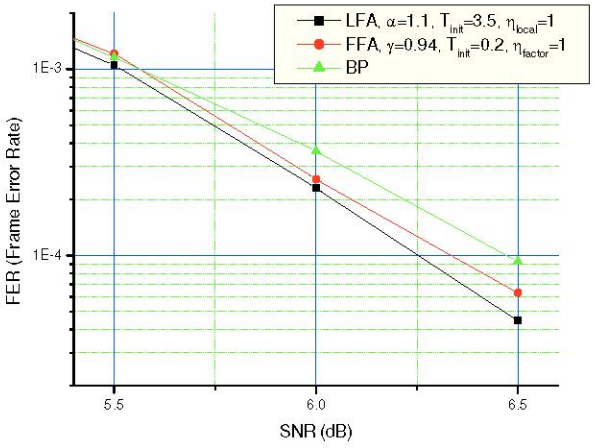

Fig. 1. Golay Code $N = 23, K = 12$, Code Rate: 0.522, Max Iteration Number: 50, Frame Error Count: 50.

Fig. 2 show the BER performance of an LDPC code [7] when the generalized BP algorithm of (9) and (10) is used without annealing (i.e., at a fixed temperature). The performance curve in Fig. 2 indicates that the best BER performance is achieved at a non-zero temperature. It is likely that, at some specific temperatures, the proposed free energy function retains the global minimum of the conventional free energy while "smoothes out" many local minimums. Nevertheless, locating the optimum temperature analytically remains an open issue.

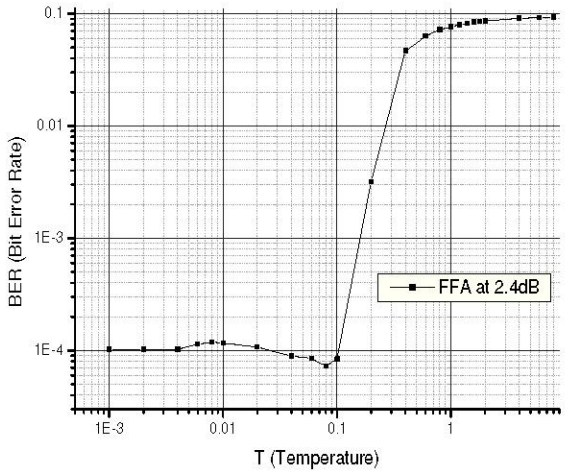

Fig. 2. Mackay 816.33.164, $N = 816$, $K = 408$, Code Rate: 0.5, Max Iteration Number: 200, Frame Error Count: 200.

We have derived generalized message passing rules and developed appropriate schemes to relax the deterministic constraints. Higher order energy approximations and the method proposed in [1] can be extended by the proposed ABP scheme as well. Furthermore, the same relaxation idea can also be used to generalize the trellis structures for turbo decoding algorithms. Alternative scheduling rules, theoretical analysis concerning the temperature effect as well as the application for new code design are to be reported in another article.

References

[1] J. S. Yedidia, "Constructing free-energy approximations and generalized belief propagation algorithms," *IEEE Trans. Inf. Theory*, vol. 51, no. 7, pp.2282-2312, July 2005.

[2] M. Welling and Y. W. Teh, "Belief optimization for binary networks: A stable alternative to belief propagation," in *Proc. Conf. Uncertainty in Artificial Intelligence*, pp. 554-561, Seattle, WA, Aug. 2001.

[3] A. Yuille, "CCCP algorithms to minimize the Bethe and Kikuchi free energies: Convergent alternatives to belief propagtion," *Neural Computation*, vol. 14, no.7, pp. 1691-1722, 2002.

[4] K. Rose, "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems," *IEEE Proc.*, vol. 86, no. 11, pp. 2210-2239, Nov. 1998.

[5] E. H. L. Aarts and J. Korst, *Simulated annealing and Boltzmann machines,* New York, USA: Wiley, 1989.

[6] S. Lin and D. J. Costello, "Error control coding : Fundamentals and applicatioins," *Englewood Cliffs, NJ: Prentice-Hall,* , 2004.

[7] *Mackay's encyclopedia of sparse graph codes*, http://www.inference.phy.cam.ac.uk/mackay/codes/data.html.