Complexity of Bethe Approximation

Jinwoo Shin
Georgia Institute of Technology

Abstract

This paper resolves a common complexity issue in the Bethe approximation of statistical physics and the sum-product Belief Propagation (BP) algorithm of artificial intelligence. The Bethe approximation reduces the problem of computing the partition function in a graphical model to that of solving a set of non-linear equations, so-called the Bethe equation. On the other hand, the BP algorithm is a popular heuristic method for estimating marginal distribution in a graphical model. Although they are inspired and developed from different directions, Yedidia, Freeman and Weiss (2004) established a somewhat surprising connection: the BP algorithm solves the Bethe equation if it converges (however, it often does not). This naturally motivates the following important question to understand their limitations and empirical successes: the Bethe equation is computationally easy to solve?

We present a message passing algorithm solving the Bethe equation in polynomial number of bitwise operations for arbitrary binary graphical models of n nodes where the maximum degree in the underlying graph is $O(\log n)$. Our algorithm, an alternative to BP fixing its convergence issue, is the first fully polynomial-time approximation scheme for the BP fixed point computation in such a large class of graphical models. Moreover, we believe that our technique is of broader interest to understand the computational complexity of the cavity method in statistical physics.

Appearing in Proceedings of the 15^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2012, La Palma, Canary Islands. Volume XX of JMLR: W&CP XX. Copyright 2012 by the authors.

1 Introduction

In the recent years, graphical models (also known as Markov random fields) defined on graphs have been studied as powerful formalisms modeling inference problems in numerous areas including computer vision, speech recognition, error-correcting codes, protein structure, networking, statistical physics, game theory and combinatorial optimization. Two central problems, commonly addressed in these applications involving graphical models, are computing the marginal distribution and the so-called partition function. It is well-known that the inference problems are computationally hard in general [Chandrasekaran et al. 2008]. Due to such a theoretical barrier, efforts have been made to develop heuristic methods.

The sum-product Belief Propagation (BP) algorithm, first proposed by Pearl (1988), and its variants (e.g. Survey Propagation) are such heuristics, driven by certain experimental thoughts, for computing the marginal distribution. Their appeal lie in the ease of implementation as well as optimality in tree-structured graphical models (models which contain no cycles). BP (and message-passing algorithms in general) can be thought as an updating rule on a set of messages:

$$m^{t+1} \;=\; f\left(m^t\right),$$

where m^t is the multi-dimensional vector of messages at the t-th iteration, and f describes the updating rule (or BP operator).[1] Two major hurdles to understand such a message-passing algorithm are about its convergence (i.e. m^t converges to m^*?) and correctness (i.e. m^* is good enough?). It is known that the BP iterative procedure always has a fixed point m^* due to the Brouwer fixed point theorem. However, BP can oscillate far from a fixed point in models with cycles, and only several sufficient convergence conditions [Weiss 2000, Tatikonda and Jordan 2002, Heskes 2004, Ihler et al. 2006] have been established in the last decade. More importantly, BP can have multiple fixed points, and even when it is unique, it may not be the correct answer. Significant efforts [Wainwright

[1] See Section 2.1 for the precise definitions of m^t and f.

et al. 2003, Heskes 2004, Yedidia 2004] were made to understand BP fixed points, while the precise approximation qualities and the rigorous understandings on their limitations still remain mystery. Regardless of those theoretical understandings, the BP algorithm performs empirically well in many applications [Freeman and Pasztor 1999, Murphy et al. 1999, Forney 2001]. For example, the highly successful turbo codes [Berrou et al. 1993] in practice can be interpreted as BP [McEliece et al. 1998] and decisions guided by BP is also known to work well to solve satisfiability problems [Ricci-Tersenghi and Semerjian 2009].

The Bethe approximation [Bethe 1935] and its variants (e.g. Kikuchi approximation [Domb and Green 1972]), originally developed in statistical physics of lattice models, are currently used as powerful approximation schemes for computing the (logarithm of) partition function in many applications. The Bethe approximation suggests to use the following quantity as an approximation for the logarithm of the partition function:

$$F(\mathbf{y}^*) \qquad \text{where} \quad \nabla F(\mathbf{y}^*) = 0.$$

F, $\nabla F(\mathbf{y}^*) = 0$ and $\mathbf{y}^*$ are called the (minus) Bethe free energy function, Bethe equation and Bethe equilibrium, respectively.[2] The statistical physics prediction suggests its asymptotic correctness in random sparse graphical models, and several rigorous evidences in particular models are known [Bandyopadhyay and Gamarnik 2006, Dembo and Montanari 2010, Chandrasekaran et al. 2011]. Efforts have also been made to estimate and characterize its error [Chertkov and Chernyak 2006, Sudderth et al. 2008]. However, the error still remains uncontrollable for models with many cycles.

Yedidia, Freeman and Weiss (2004) established a somewhat surprising connection between the BP algorithm and the Bethe approximation: if BP converges, it solves the Bethe equation. Equivalently, the BP fixed point equation $f(m^*) = m^*$ is in essence equivalent to the Bethe equation $\nabla F(\mathbf{y}^*) = 0$. This naturally leads to the following common computational question for both: the BP fixed point computation is computationally easy? Formally speaking,

$\mathcal{Q}$. Given $\varepsilon > 0$, is it possible to design a deterministic iterative algorithm finding m^* satisfying

$$(1-\varepsilon)\, f(m^*) \;\le\; m^* \;\le\; (1+\varepsilon)\, f(m^*),$$

in polynomial number of bitwise operations with respect to $1/\varepsilon$ and the dimension of vector m^*?

Such $\varepsilon > 0$ is necessary since the exact computation (i.e. $\varepsilon = 0$) is impossible since BP fixed points are irrational in general. An algorithm in $\mathcal{Q}$ can be used an alternative to BP with provably fast convergence rate (i.e. fixing the convergence issue of BP) and eliminates a need for the convergence analysis of BP. Even though it may not converge to the correct answer, it can, at least, provide a guidance toward it [Ricci-Tersenghi and Semerjian 2009]. Further, it confirms the efficiency of the Bethe approximation scheme as well since m^* satisfying the above inequality provides $\mathbf{y}^*$ with $\|\nabla F(\mathbf{y}^*)\| \le \varepsilon$. Efforts to design such algorithms were made [Teh and Welling 2001, Yuille 2002], but no rigorous analysis on their convergence rates is known. Chandrasekaran et al. (2011) recently proposed an algorithm with provable polynomial convergence rate, but the work is for a specific graphical model, i.e., the uniform distributions on independent sets of sparse graphs. An ideal algorithm should work for a large class of graphical models.

1.1 Our Contribution

The main result of this paper is the following answer $\mathcal{A}$ for the question $\mathcal{Q}$ for the BP operator f and arbitrary sparse binary graphical models. To state it formally, we let n be the number of nodes and Δ be the maximum degree in the underlying graph, respectively.

$\mathcal{A}$. Given $\varepsilon > 0$, there exists a deterministic iterative algorithm finding m^* satisfying

$$(1-\varepsilon)\, f(m^*) \;\le\; m^* \;\le\; (1+\varepsilon)\, f(m^*)$$

in $2^{O(\Delta)} n^2 \varepsilon^{-4} \log^3(n\varepsilon^{-1})$ iterations.

In this paper, we call the message m^* satisfying the above inequality as an ε-approximate BP fixed point. In what follows, we explain the algorithm in details.

The known equivalence [Yedidia et al. 2004] between the BP fixed point equation and the Bethe equation implies that the question $\mathcal{Q}$ is equivalent to the following.

$\mathcal{Q}'$. Given $\varepsilon > 0$, is it possible to design a deterministic iterative algorithm finding $\mathbf{y}^*$ satisfying

$$\|\nabla F(\mathbf{y}^*)\| \;\le\; \varepsilon,$$

in polynomial number of bitwise operations with respect to $1/\varepsilon$ and the dimension of the domain D of the Bethe free energy function F?

However, we note that it is still far from being obvious whether it is computationally ‘easy’ to find such a near

[2] See Section 2.2 for the precise definition of F.

stationary point.[3] Natural attempts are gradient descent algorithms to find a local minimum or maximum of F: iteratively update $\mathbf{y}(t)$ as

$$\mathbf{y}(t+1) \;=\; \mathbf{y}(t) + \alpha \nabla F(\mathbf{y}(t)),$$

where $\alpha \in \mathbb{R}$ is the (appropriately chosen) step-size. The main issue here is that the gradient algorithm may not find a near stationary point if $\mathbf{y}(t)$ hits (or moves across) the boundary of D in one of its iterations (and some projection may be required). Hence, the main strategy in the work by Chandrasekaran et al. (2011) to avoid the hitting issue lies in (a) understanding the behavior of gradient ∇F close to the boundary of D and (b) designing an appropriate small step-size in the gradient algorithm based on the understanding (a).

The main technical challenge to apply the strategy to general binary graphical model, beyond the specific uniform independent-set model studied by Chandrasekaran et al. (2011), is on (a). The domain D is simply $\left[0, \frac{1}{2}\right]^n$ in the uniform independent-set model since the Bethe free energy function F is determined by node marginal probabilities in the model. One can observe that the proof strategy by Chandrasekaran et al. (2011) immediately fails even for the non-uniform independent-set model, which has the domain $D = [0,1]^n$. Furthermore, the more significant issue is that in general graphical model the domain becomes more complex, i.e. $D = [0,1]^{n+m}$ where m is the number of edges in the underlying graph. This is because the Bethe free energy should consider pairwise (or edge) marginal probabilities as well. One can check that any similar approaches to that of Chandrasekaran et al. (2011) fail in the larger domain. To overcome such a technical issue, we first observe that at stationary points of F, pairwise marginal probabilities should satisfy certain quadratic equations in terms of node marginal probabilities. This allows to express the Bethe free energy again in terms of node marginal probabilities i.e. $D = [0,1]^n$. Now we study this 'modified' Bethe expression to avoid the hitting issue, which we end up with an appropriate small step-size in the gradient algorithm. Moreover, we eliminate a need to decide such a small step-size explicitly in the algorithm, by designing an elegant time-varying projection scheme. The algorithm is presented in Section 3.

We later realized that the 'modified' Bethe expression was already proposed by Teh and Welling (2001), where they suggested gradient algorithms to minimize it using sigmoid functions. The main difference in our work is that our gradient algorithm does the minimization task with a projection scheme, instead of using sigmoid functions. This is possible since our algorithm design was motivated from avoiding the hitting issue. The success of our rigorous convergence rate analysis, which was missing in the work of Teh and Welling (2001), crucially relies on this difference.

One can observe that our gradient algorithm is implementable as a 'BP-like' iterative, message passing algorithm: each node maintains a message at each iteration and passes it to its neighbors. We prove it terminates in $2^{O(\Delta)} n^2 \varepsilon^{-4} \log^3(n\varepsilon^{-1})$ iterations until it finds an ε-approximate BP fixed point. In a complexity point of view, the only remaining issue is that each node may require to maintain irrational messages (of infinitely long bits). We further show in Section 4 that a polynomial number (with respect to $1/\varepsilon$, n and 2^{Δ}) of bits to approximate each message suffices, and hence the algorithm consists of only a polynomial number of bitwise operations in total. Namely, it is a fully polynomial-time approximation scheme (FPTAS) to compute an approximate BP fixed point for sparse binary graphical models where $\Delta = O(\log n)$.

1.2 Organization

In Section 2, we provide backgrounds for graphical models, Belief Propagation and Bethe approximation. In Section 3 and 4, we describe our algorithms and their running times.

2 Graphical Models

We first introduce a class of joint distributions defined with respect to (undirected) graphs, which are called (pairwise) *Markov random fields* (MRFs) [Lauritzen 1996]. Specifically, let $G = (V,E)$ be a undirected graph with the vertices being denoted by V with $|V| = n$, and the edges $E \subseteq \binom{V}{2}$ denoting a set of unordered pairs of vertices. The vertices of G label a collection of random variables $\mathbf{x} = \{x_v \mid v \in V\}$. Our focus in this paper is on binary random variables, i.e., $x_v \in \{0,1\}$ for all $v \in V$.

Now consider the following joint distribution on $\{0,1\}^n$ that factors according to G: for $\mathbf{x} \in \{0,1\}^n$,

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{v \in V} \psi_v(x_v) \prod_{(u,v) \in E} \psi_{u,v}(x_u, x_v).$$

Here, each $\psi_{u,v}$ and ψ_v are non-negative functions on $\{0,1\}^2$ and $\{0,1\}$, respectively. These local functions are called *potential* functions or *compatibility* functions. The normalizing factor Z is called the *partition* function:

$$Z = \sum_{\mathbf{x} \in \{0,1\}^n} \prod_{v \in V} \psi_v(x_v) \prod_{(u,v) \in E} \psi_{u,v}(x_u, x_v). \qquad (1)$$

[3] PPAD and PLS are complexity classes to capture the hardness of computations for fixed points and local optima, respectively.

Finally, some notations. Let $\mathcal{N}(v)$ be the set of neighbors of a vertex $v \in V$, $d_v := |\mathcal{N}(v)|$ be the degree of $v \in V$, and $\Delta := \max_v d_v$ be the maximum degree in the graph G. Further, we define

$$\psi_* := \max_{(u,v)\in E, x_u, x_v \in \{0,1\}} \left\{ e^{|\ln \psi_v(x_v)|}, e^{|\ln \psi_{u,v}(x_u,x_v)|} \right\}.$$

In this paper, we primarily focus on the case $\psi_* = O(1)$.[4]

2.1 Belief Propagation

The sum-product Belief Propagation (BP) algorithm has messages

$$\{m^t_{u\to v}(x_v), m^t_{v\to u}(x_u) : (u,v) \in E, x_v, x_u \in \{0,1\}\}$$

at the t-th iteration are on the both sides of edges and it updates them as

$$m^{t+1}_{u\to v}(x_v) \propto \sum_{x_u} \psi_{u,v}(x_u,x_v)\psi_u(x_u) \prod_{w\in\mathcal{N}(u)\setminus v} m^t_{w\to u}(x_u),$$

where $\sum_{x_v\in\{0,1\}} m^{t+1}_{u\to v}(x_v) = 1$. This is equivalent to the following updating rule on (reduced) messages $\{m^t_{u\to v}, m^t_{v\to u}\}$.

$$m^{t+1}_{u\to v} \;=\; f_{u\to v}\left(\prod_{w\in\mathcal{N}(u)\setminus v} m^t_{w\to u} \right),$$

where $m^t_{u\to v} := m^t_{u\to v}(1)/m^t_{u\to v}(0)$ and the function $f_{u\to v} : \mathbb{R}_+ \to \mathbb{R}_+$ is defined as

$$f_{u\to v}(x) \;:=\; \frac{\psi_{u,v}(0,1)\psi_u(0) + \psi_{u,v}(1,1)\psi_u(1)\cdot x}{\psi_{u,v}(0,0)\psi_u(0) + \psi_{u,v}(1,0)\psi_u(1)\cdot x}.$$

Now the *BP fixed point* of messages $\{m_{u\to v}, m_{v\to u}\}$ can be naturally defined as

$$m_{u\to v} \;=\; f_{u\to v}\left(\prod_{w\in\mathcal{N}(u)\setminus v} m_{w\to u} \right),$$

where one can easily argue the existence of such a fixed point using the Brouwer fixed point theorem. This motivates the following notion of ε-approximate BP fixed point.

Definition 1 *The set of messages $\{m_{u\to v}, m_{v\to u} : (u,v) \in E\}$ is called an ε-approximate BP fixed point if*

$$\left| \frac{m_{u\to v}}{f_{u\to v}\left(\prod_{w\in\mathcal{N}(u)\setminus v} m_{w\to u}\right)} - 1 \right| \le \varepsilon, \quad \forall\ (u,v) \in E.$$

The BP estimates for node and edge marginal probabilities based on messages, denoted by $\tau_v(\cdot), \tau_{u,v}(\cdot)$ for $v \in V, (u,v) \in E$, are defined as

$$\frac{\tau_v(x_v)}{\psi_v(x_v)} \;\propto\; \prod_{u\in\mathcal{N}(v)} m_{u\to v}(x_v) \tag{2}$$

$$\frac{\tau_{u,v}(x_u,x_v)}{\psi_u(x_u)\psi_v(x_v)\psi_{u,v}(x_u,x_v)} \propto \prod_{w\in\mathcal{N}(u)\setminus v} m_{w\to u}(x_u) \prod_{w\in\mathcal{N}(v)\setminus u} m_{w\to v}(x_v), \tag{3}$$

where $\sum_{x_v} \tau_v(x_v) = 1$ and $\tau_v(x_v) = \sum_{x_u} \tau_{u,v}(x_u,x_v)$.

2.2 Bethe Approximation

The Bethe approximation is an approximation to the logarithm of the partition function (i.e. $\ln Z$), given by

$$\sum_{v\in V} \sum_{x_v} \tau_v(x_v)\left[\ln \psi_v(x_v) - \ln \tau_v(x_v)\right] + \sum_{\{u,v\}\in E} \sum_{x_u,x_v} \tau_{u,v}(x_u,x_v) \left[\ln \psi_{u,v}(x_u,x_v) - \ln \frac{\tau_{u,v}(x_u,x_v)}{\tau_u(x_u)\tau_v(x_v)} \right].$$

Under the constraints $\sum_{x_v} \tau_v(x_v) = 1$ and $\tau_v(x_v) = \sum_{x_u} \tau_{u,v}(x_u,x_v)$, it can be reduced as a function $F(\mathbf{y})$[5] of $\mathbf{y} = [y_v, y_{u,v}]$ where $y_v = \tau_v(1)$, $y_{u,v} = \tau_{u,v}(1,1)$ and the following substitutions:

$$\begin{aligned} \tau_v(0) &= 1 - y_v \\ \tau_{u,v}(0,0) &= 1 - y_u - y_v + y_{u,v} \\ \tau_{u,v}(0,1) &= y_v - y_{u,v} \\ \tau_{u,v}(1,0) &= y_u - y_{u,v}. \end{aligned}$$

One can obtain the gradient $\nabla F(\mathbf{y}) = \left[\frac{\partial F}{\partial y_v}, \frac{\partial F}{\partial y_{u,v}}\right]$ as

$$\frac{\partial F}{\partial y_v} = \psi^{(v)} + \ln \frac{1-y_v}{y_v} + \sum_{u\in\mathcal{N}(v)} \ln\left(\frac{1-y_v-y_u+y_{u,v}}{1-y_v} \cdot \frac{y_v}{y_v-y_{u,v}} \right) \tag{4}$$

$$\frac{\partial F}{\partial y_{u,v}} = \psi^{(u,v)} + \ln\left(\frac{y_u - y_{u,v}}{1-y_u-y_v+y_{u,v}} \cdot \frac{y_v - y_{u,v}}{y_{u,v}} \right), \tag{5}$$

where

$$\psi^{(v)} := \ln \frac{\psi_v(1)}{\psi_v(0)} + \sum_{u\in\mathcal{N}(v)} \ln \frac{\psi_{u,v}(0,1)}{\psi_{u,v}(0,0)}$$

$$\psi^{(u,v)} := \ln \frac{\psi_{u,v}(0,0)\,\psi_{u,v}(1,1)}{\psi_{u,v}(1,0)\,\psi_{u,v}(0,1)}.$$

[4] This excludes the case $\psi_{u,v}(\cdot,\cdot) = 0$. However, we note that our algorithm and its analysis still work even for the case $\psi_{u,v}(\cdot,\cdot) = 0$ such as the independent set model in the work [Chandrasekaran et al. 2011].

[5] $-F$ is called the Bethe free energy function.

It is known that there is one-to-one correspondence between BP fixed points and zero gradient points of F. In particular, one can obtain the following lemma, where its proof can be done easily using the algebraic expressions (4) and (5) of gradients.

Lemma 1 *Given $\varepsilon \in [0,1)$, suppose $\mathbf{y} = [y_v, y_{u,v}]$ satisfies $\|\nabla F(\mathbf{y})\|_\infty \leq \varepsilon$. Then, the set of messages $\{m_{u\to v}, m_{v\to u} : (u,v) \in E\}$ is a 6ε-approximate BP fixed point if it is given as*

$$m_{u\to v} = \frac{\psi_{u,v}(0,1)}{\psi_{u,v}(0,0)} \cdot \frac{1 - y_v - y_u + y_{u,v}}{1 - y_v} \cdot \frac{y_v}{y_v - y_{u,v}}. \tag{6}$$

The proof of Lemma 1 is omitted due to space constraints.

3 Message Passing Algorithm for BP Fixed Point Computation

In this section, we present the main result of this paper, a new message passing algorithm for approximating a BP fixed point. From the (algebraic) relationship between approximate BP fixed points and near gradient points of the Bethe free energy function F in Lemma 1, it is equivalent to compute a near gradient point $\mathbf{y}$ i.e. $\|\nabla F(\mathbf{y})\|_2 \leq \varepsilon$.

3.1 Algorithm Description

Our algorithm, described next, for finding such a point is essentially motivated by the standard (projected) gradient algorithm. The non-triviality (and novelty) lies in our choice of appropriate (time-varying) 'projection $[\cdot]_*$' with respect to the (time-varying) 'step-size $\frac{1}{\sqrt{t}}$' at each iteration and subsequent analysis of rate of convergence.

Algorithm A1

1. Algorithm parameters:

   $$\varepsilon \in (0,1) \quad \text{and} \quad \mathbf{y}(t) = [\, y_v(t) \in (0,1) : v \in V\,]$$

   at the t-th iteration.

2. $\mathbf{y}(t)$ is updated as:

   $$\mathbf{y}(t+1) \quad = \quad \left[\mathbf{y}(t) + \frac{1}{\sqrt{t}} \nabla F(\mathbf{y}(t), \mathbf{y}_E(t))\right]_*,$$

   where the projection $[\cdot]_*$ at the t-th iteration is defined as

   $$[x]_* = \begin{cases} x & \text{if } \frac{1}{t^{1/4}} \leq x \leq 1 - \frac{1}{t^{1/4}} \\ \frac{1}{t^{1/4}} & \text{if } x < \frac{1}{t^{1/4}} \\ 1 - \frac{1}{t^{1/4}} & \text{if } x > 1 - \frac{1}{t^{1/4}} \end{cases},$$

   and $\mathbf{y}_E(t) = [y_{u,v}(t)] \in (0,1)^{|E|}$ is computed as the unique solution satisfying

   $$\frac{y_u(t) - y_{u,v}(t)}{1 - y_u(t) - y_v(t) + y_{u,v}(t)} \cdot \frac{y_v(t) - y_{u,v}(t)}{e^{-\psi^{(u,v)}} y_{u,v}(t)} = 1$$

   $$0 < y_{u,v}(t) < \min\{y_v(t), y_u(t)\}.$$

3. Compute messages $\{m_{u\to v}, m_{v\to u}\}$ from $[y_v(t), y_{u,v}(t)]$ using the formula (6).

4. Terminate if $\{m_{u\to v}, m_{v\to u}\}$ is an ε-approximate BP fixed point.

The algorithm is clearly implementable through message-passing where each node u sends $y_u(t)$ to all of its neighbors $v \in \mathcal{N}(u)$ at each iteration. We also note that solving the second step for computing $y_{u,v}(t)$ can be done efficiently since it is solving a quadratic equation whose coefficients are decided by $y_v(t)$ and $y_u(t)$. We establish the following running time of the algorithm.

Theorem 2 **A1** *terminates in $2^{O(\Delta)} n^2 \varepsilon^{-4} \log^3 \frac{n}{\varepsilon}$ iterations as long as $\psi_* = O(1)$.*

The proof of Theorem 2 is presented in the following section. Note that the algorithm may require to maintain irrational messages or rational messages of long bits. In Section 4, we present a minor modification of the algorithm to fix the issue, which leads to a fully poly-time approximation algorithm (FPTAS) to compute an approximate BP fixed point.

3.2 Proof of Theorem 2

We first define F^* on $(0,1)^n$: for $\mathbf{y} = [y_v] \in (0,1)^n$

$$F^*(\mathbf{y}) \ = F(\mathbf{y}, \mathbf{y}_E),$$

where F is the (original) Bethe free energy function defined in Section 2.2 and the additional vector $\mathbf{y}_E = [y_{u,v}] \in (0,1)^{|E|}$ is defined as the solution satisfying that $y_{u,v} < \min\{y_v, y_v\}$ and

$$\psi^{(u,v)} + \ln\left(\frac{y_u - y_{u,v}}{1 - y_u - y_v + y_{u,v}} \cdot \frac{y_v - y_{u,v}}{y_{u,v}}\right) = 0. \tag{7}$$

Observe that each $y_{u,v}$ is a function of y_u, y_v, i.e. $y_{u,v} = y_{u,v}(y_u, y_v)$. One can check that the gradient of F^* has the same form with that of F as follows.

$$\begin{aligned} \frac{\partial F^*}{\partial y_v} \ &= \ \psi^{(v)} + \ln \frac{1 - y_v}{y_v} \\ &\quad + \sum_{u \in \mathcal{N}(v)} \ln\left(\frac{1 - y_v - y_u + y_{u,v}}{1 - y_v} \cdot \frac{y_v}{y_v - y_{u,v}}\right), \end{aligned}$$

where we recall that $y_{u,v}$ is decided in terms of y_u, y_v from (7). This implies that the updating procedure of $\mathbf{y}(t)$ in the algorithm is simply as

$$\mathbf{y}(t+1) \;=\; \left[\mathbf{y}(t) + \frac{1}{\sqrt{t}}\,\nabla F^*(\mathbf{y}(t))\right]_* . \tag{8}$$

Based on this interpretation, we start to prove the running time of the algorithm by stating the following key lemma.

Lemma 3 *Define $\delta > 0$ as the largest real number that satisfies the followings.*

$$\delta \;\le\; \frac{1/2}{2(\Delta+1)\psi_*^{4\Delta+1}+1} \qquad \textit{and} \qquad 4\delta \ln\frac{1}{2\delta} \;\le\; 1.$$

Then,

$$\mathbf{y}(t) \in D := [\delta, 1-\delta]^n, \qquad \forall\; t \ge t_* := \delta^{-4}.$$

Proof. First observe that $\mathbf{y}(t_*) \in D$ due to our choice of projection $[\cdot]_*$ and $\frac{1}{t_*^{1/4}} = \delta$. Hence, it suffices to establish the following three steps: for all $v \in V$ and $t > t_*$,

$$\frac{\partial F^*}{\partial y_v} \;\le\; 0 \qquad \text{if } \; y_v \ge 1-2\delta \;\text{ and }\; \mathbf{y}\in D, \tag{9}$$

$$\frac{\partial F^*}{\partial y_v} \;\ge\; 0 \qquad \text{if } \; y_v \le 2\delta \;\text{ and }\; \mathbf{y}\in D, \tag{10}$$

$$\frac{1}{\sqrt{t}}\left|\frac{\partial F^*}{\partial y_v}\right| \;\le\; \frac{\delta}{2} \qquad \text{if } \; \mathbf{y}\in D. \tag{11}$$

From (8), (9), (10) and (11), it clearly follows that $\mathbf{y}(t) \in D$ for all $t \ge t_*$.

Proof of (9)**.** We first provide a proof of (9). To this end, if $\mathbf{y} \in (0,1)^n$, we have

$$\begin{aligned}
&\frac{1-y_v-y_u+y_{u,v}}{1-y_v}\cdot\frac{y_v}{y_v-y_{u,v}}\\
&\quad= \frac{1}{1+\frac{y_u-y_{u,v}}{1-y_v-y_u+y_{u,v}}}\cdot\frac{y_v}{y_v-y_{u,v}}\\
&\quad= \frac{1}{1+e^{-\psi^{(u,v)}}\cdot\frac{y_{u,v}}{y_v-y_{u,v}}}\cdot\left(1+\frac{y_{u,v}}{y_v-y_{u,v}}\right)\\
&\quad\le \max\left\{1, e^{\psi^{(u,v)}}\right\}\\
&\quad\le \psi_*^4,
\end{aligned}\tag{12}$$

where we use the definition (7) of $y_{u,v}$. Using this, (9) follows as

$$\begin{aligned}
\frac{\partial F^*}{\partial y_v} &= \psi^{(v)} + \ln\frac{1-y_v}{y_v}\\
&\qquad + \sum_{u\in\mathcal{N}(v)} \ln\left(\frac{1-y_v-y_u+y_{u,v}}{1-y_v}\cdot\frac{y_v}{y_v-y_{u,v}}\right)\\
&\le \ln\left(2(\Delta+1)\psi_*\right) + \ln\frac{2\delta}{1-2\delta} + \Delta\ln\psi_*^4\\
&= \ln\frac{2(\Delta+1)\psi_*^{4\Delta+1}}{\frac{1}{2\delta}-1}\\
&\le 0,
\end{aligned}$$

where the last inequality is from our choice of $\delta \le \frac{1/2}{2(\Delta+1)\psi_*^{4\Delta+1}+1}$.

Proof of (10)**.** Second, we provide a proof of (10). Similarly as we did in (12), we have

$$\begin{aligned}
\frac{1-y_v-y_u+y_{u,v}}{1-y_v}\cdot\frac{y_v}{y_v-y_{u,v}} &\ge \min\left\{1, e^{\psi^{(u,v)}}\right\}\\
&\ge \frac{1}{\psi_*^4}.
\end{aligned}\tag{13}$$

Hence, (10) follows as

$$\begin{aligned}
\frac{\partial F^*}{\partial y_v} &= \psi^{(v)} + \ln\frac{1-y_v}{y_v}\\
&\qquad + \sum_{u\in\mathcal{N}(v)} \ln\left(\frac{1-y_v-y_u+y_{u,v}}{1-y_v}\cdot\frac{y_v}{y_v-y_{u,v}}\right)\\
&\ge -\ln\left(2(\Delta+1)\psi_*\right) + \ln\frac{1-2\delta}{2\delta} + \Delta\ln\frac{1}{\psi_*^4}\\
&= \ln\frac{\frac{1}{2\delta}-1}{2(\Delta+1)\psi_*^{4\Delta+1}}\\
&\ge 0,
\end{aligned}$$

where the last inequality is again from our choice of $\delta \le \frac{1/2}{2(\Delta+1)\psi_*^{4\Delta+1}+1}$.

Proof of (11)**.** Finally, we provide a proof of (11). Using (12) and (13), it follows as

$$\begin{aligned}
\left|\frac{\partial F^*}{\partial y_v}\right| &= \left|\psi^{(v)}\right| + \left|\ln\frac{1-y_v}{y_v}\right|\\
&\qquad + \sum_{u\in\mathcal{N}(v)} \left|\ln\left(\frac{1-y_v-y_u+y_{u,v}}{1-y_v}\cdot\frac{y_v}{y_v-y_{u,v}}\right)\right|\\
&\le \ln\left(2(\Delta+1)\psi_*\right) + \ln\frac{1-2\delta}{2\delta} + \Delta\ln\psi_*^4\\
&= 2\ln\frac{1-2\delta}{2\delta}\\
&\le 2\ln\frac{1}{2\delta}\\
&\le \frac{\delta}{2}\cdot\sqrt{t_*},
\end{aligned}$$

where the last equality is from our choices of δ, t_* which imply

$$\sqrt{t_*} \;=\; \delta^{-2} \;\geq\; 4\frac{1}{\delta}\ln\frac{1}{2\delta}.$$

This completes the proof of Lemma 3. □

Using the above lemma, we will show that the algorithm terminates in $2^{O(\Delta)}n^2\varepsilon^{-4}\log^3(n\varepsilon^{-1})$ iterations until it outputs an ε-approximate BP fixed point. We first explain why it suffices to show the following:

$$\sum_{t=t_*}^{T} c_t \cdot \|\nabla F^*(\mathbf{y}(t))\|_2^2 = \frac{2^{O(\Delta)}\, n\, \log T}{\sqrt{T}}, \tag{14}$$

where $c_t = \frac{t^{-1/2}}{\sum_{t=t^*}^{T} t^{-1/2}}$. The above equality suggests that we can choose $T = 2^{O(\Delta)}n^2\varepsilon^{-4}\log^3(n\varepsilon^{-1})$ such that

$$\sum_{t=t_*}^{T} c_t \cdot \|\nabla F^*(\mathbf{y}(t))\|_2^2 \leq \left(\frac{\varepsilon}{6}\right)^2.$$

From $\sum_{t=t_*}^{T} c_t = 1$, there exists $t \in [t_*, T]$ such that $\|\nabla F^*(\mathbf{y}(t))\|_2 \leq \varepsilon/6$. Further, observe that if $\mathbf{y}_E(t)$ is defined from (7),

$$\|\nabla F(\mathbf{y}(t), \mathbf{y}_E(t))\|_2 \;\leq\; \|\nabla F^*(\mathbf{y}(t))\|_2 \;\leq\; \varepsilon/6.$$

Then, Lemma 1 implies that the computed messages at the t-th iteration is an ε-approximate BP fixed point.

Now we proceed toward establishing the desired inequality (14). The important implication of Lemma 3 is that the algorithm does not need the projection $[\cdot]_*$ after the t_*-th iteration. In other words, from Lemma 3 and (8), we have that

$$\mathbf{y}(t+1) \;=\; \mathbf{y}(t) + \frac{1}{\sqrt{t}}\nabla F^*(\mathbf{y}(t)), \qquad \forall\; t \geq t_*.$$

In what follows, we will assume $t \geq t_*$ and $\mathbf{y}(t) \in D$ from Lemma 3.

Using the Taylor's expansion, we have

$$\begin{aligned} F^*(\mathbf{y}(t+1)) & \\ &= F^*\left(\mathbf{y}(t) + \frac{1}{\sqrt{t}}\nabla F^*(\mathbf{y}(t))\right) \\ &= F^*(\mathbf{y}(t)) + \nabla F^*(\mathbf{y}(t))' \cdot \frac{1}{\sqrt{t}}\nabla F^*(\mathbf{y}(t)) \\ &\quad + \frac{1}{2}\frac{1}{\sqrt{t}}\nabla F^*(\mathbf{y}(t))' \cdot R \cdot \frac{1}{\sqrt{t}}\nabla F^*(\mathbf{y}(t)), \end{aligned} \tag{15}$$

where R is a $n \times n$ matrix such that

$$|R_{vw}| \leq \sup_{\mathbf{y}\in B}\left|\frac{\partial^2 F^*}{\partial y_v \partial y_w}\right|,$$

and B is a L_∞-ball in $\mathbb{R}^n$ centered at $\mathbf{y}(t) \in D$ with its radius

$$r = \max_{v\in V}\left|\frac{1}{\sqrt{t}}\frac{\partial F^*}{\partial y_v}(\mathbf{y}(t))\right|.$$

From (11), we know $r \leq \frac{\delta}{2}$. Hence, $y_v \in [\delta/2, 1-\delta/2]$ for $\mathbf{y} = [y_v] \in B$. Using this with $\delta = 1/2^{O(\Delta)}$, one can check that

$$\sup_{\mathbf{y}\in B}\left|\frac{\partial^2 F^*}{\partial y_v \partial y_w}\right| \;=\; \begin{cases} 2^{O(\Delta)} & \text{if } v = w, (v,w) \in E \\ 0 & \text{otherwise} \end{cases}.$$

Therefore, using these bounds the equality (15) reduces to

$$\begin{aligned} F^*(\mathbf{y}(t+1)) & \\ &\geq F^*(\mathbf{y}(t)) + \frac{1}{\sqrt{t}}\|\nabla F^*(\mathbf{y}(t))\|_2^2 - \frac{O\left(2^{O(\Delta)}|E|\right)}{t} \\ &\geq F^*(\mathbf{y}(t)) + \frac{1}{\sqrt{t}}\|\nabla F^*(\mathbf{y}(t))\|_2^2 - \frac{O\left(2^{O(\Delta)}n\right)}{t}, \end{aligned}$$

since $|E| \leq \Delta \cdot n$. If we sum the above inequality over t from t_* to $T-1$, we have

$$F^*(\mathbf{y}(T)) \;\geq\; F^*(\mathbf{y}(t_*)) + \sum_{t=t_*}^{T-1}\frac{1}{\sqrt{t}}\|\nabla F^*(\mathbf{y}(t))\|_2^2 - O\left(2^{O(\Delta)}n\right)\sum_{t=t_*}^{T-1}\frac{1}{t}.$$

Since $|F^*(\mathbf{y})| = O(\Delta n)$ for $\mathbf{y} \in D$, we obtain

$$\sum_{t=t_*}^{T-1}\frac{1}{\sqrt{t}}\|\nabla F^*(\mathbf{y}(t))\|_2^2 \;\leq\; O(\Delta n) + O\left(2^{O(\Delta)}n\right)\sum_{t=t_*}^{T-1}\frac{1}{t}.$$

Thus, we finally obtain the desired conclusion (14) as follows.

$$\begin{aligned} \sum_{t=t_*}^{T} c_t \cdot \|\nabla F^*(\mathbf{y}(t))\|_2^2 & \\ &= \frac{1}{\sum_{t=t_*}^{T}\frac{1}{\sqrt{t}}}\sum_{t=t_*}^{T}\frac{1}{\sqrt{t}}\|\nabla F^*(\mathbf{y}(t))\|_2^2 \\ &\leq \frac{1}{\sum_{t=t_*}^{T}\frac{1}{\sqrt{t}}}\left(O(\Delta n) + O\left(2^{O(\Delta)}n\right)\sum_{t=t_*}^{T}\frac{1}{t}\right) \\ &= \frac{2^{O(\Delta)}n\log T}{\sqrt{T}}, \end{aligned}$$

where we recall that $t_* = \delta^{-4} = 2^{O(\Delta)}$. This completes the proof of Theorem 2.

4 Modification to FPTAS

In this section, we provide a minor modification of Algorithm **A1** in Section 3, to establish a fully polynomial-time approximation scheme (FPTAS) for the BP fixed point computation. We will show only a polynomial number of bits are enough to maintain for each message $y_v(t)$. To this end, we define the following function $g^t = [g_v^t]$ which describe the updating rule (at the t-th iteration) of Algorithm **A1**, i.e.

$$\mathbf{y}(t+1) \;=\; g^t(\mathbf{y}(t)) \qquad \text{under Algorithm } \mathbf{A1}.$$

Now we formally propose the following algorithm of minor modification.

Algorithm A2

1. Algorithm parameters:

   $$\varepsilon \in (0,1) \qquad \text{and} \qquad \mathbf{z}(t) = [\, z_v(t) \in (0,1) : v \in V\,]$$

   at the t-th iteration, where $z_v(t)$ has k-bits (i.e. $2^k z_v(t) \in \mathbb{Z}$) for all $t \geq 0$, $v \in V$.

2. $\mathbf{z}(t)$ is updated as:

   $$\left| z_v(t+1) - g_v^t(\mathbf{z}(t)) \right| \;\leq\; \frac{1}{2^k}.$$

3. Compute a set of messages $\{m_{u\to v}, m_{v\to u}\}$ satisfying

   $$m_{u\to v} = \frac{\psi_{u,v}(0,1)}{\psi_{u,v}(0,0)} \cdot \frac{1 - z_v(t) - z_u(t) + z_{u,v}(t)}{1 - z_v(t)} \cdot \frac{z_v(t)}{z_v(t) - z_{u,v}(t)},$$

   where $z_{u,v}(t) > 0$ is computed to satisfy

   $$\left| \psi^{(u,v)} + \ln\Big(\frac{z_u(t) - z_{u,v}(t)}{1 - z_u(t) - z_v(t) + z_{u,v}(t)} \cdot \frac{z_v(t) - z_{u,v}(t)}{z_{u,v}(t)} \Big) \right| \;\leq\; \frac{\varepsilon}{6}.$$

4. Terminate if $\{m_{u\to v}, m_{v\to u}\}$ is an ε-approximate BP fixed point.

We note that each step in the above algorithm is executable in a polynomial number of bitwise operations with respect to Δ, $1/\varepsilon$, k and n. The second step to compute g_v^t consists of $O(\Delta)$ arithmetic operations, logarithm, division, addition, square root and multiplication. Furthermore, the equations in the third step to compute $\{m_{u\to v}, m_{v\to u}\}$ can be solvable in a polynomial number of bitwise operations with respect to $1/\varepsilon$ and k.

Now we state the following theorem, which shows that one can choose k as a polynomial in terms of n, $1/\varepsilon$ and 2^{Δ}. This implies that Algorithm **A2** is a FPTAS for such a choice of k as long as $\Delta = O(\log n)$. We note that one can obtain the explicit bound of k in terms of Δ, ψ_*, n and ε via explicitly calculating each step in our proof.[6]

Theorem 4 **A2** *terminates in* $2^{O(\Delta)} n^2 \varepsilon^{-4} \log^3 \frac{n}{\varepsilon}$ *iterations for some* $k = 2^{O(\Delta)} n^2 \varepsilon^{-4} \log^4 \frac{n}{\varepsilon}$ *as long as* $\psi_* = O(1)$.

The proof of Theorem 4 is omitted due to space constraints.

5 Conclusion

In the last decade, exciting progresses have been made on understanding computationally hard problems in computer science using a variety of methods in statistical physics. The belief propagation (BP) algorithm or its variants are on this line and suggest to solve certain relaxations of hard problems. In this paper, we address the question whether the relaxation is indeed computationally easy to solve in a strong sense. We believe that our rigorous complexity analysis of the BP-relaxation is the important step to guarantee the computational complexity of BP-based algorithms.

Acknowledgements

We are grateful to Devavrat Shah for fruitful comments on this paper. We are also grateful to Max Welling for pointing out the similarity between our algorithm and that by Teh and Welling (2001).

References

[1] A. Bandyopadhyay and D. Gamarnik. Counting without sampling: new algorithms for enumeration problems using statistical physics. *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, 890-899, 2006.

[2] H. A. Bethe. Statistical theory of superlattices. *Proc. Roy. Soc. London A*, 150:552-558, 1935.

[6] Another naive way to avoid such an explicit choice of k is to run Algorithm **A2** 'polynomially' many times by increasing k (as well as the number of iterations) until it succeeds.

[3] C. Berrou, A. Glavieux and P. Thitimajshima. Near Shannon Limit Error-correcting Coding and Decoding: Turbo codes (I). *Proceeding of ICC (Geneva)*, 1993.

[4] V. Chandrasekaran, M. Chertkov, D. Gamarnik, D. Shah and J. Shin. Counting independent sets using the Bethe approximation. *SIAM Jouurnal on Discrete Mathematics*, 25(2):1012-1034, 2011.

[5] V. Chandrasekaran, N. Srebro and P. Harsha. Complexity of inference in graphical models, *Uncertainty in Artificial Intelligence*, 2008.

[6] M. Chertkov and V. Y. Chernyak. Loop series for discrete statistical models on graphs. *Journal of Statistical Mechanics: Theory and Experiment*, 2006.

[7] A. Dembo and A. Montanari. Ising models on locally tree-like graphs. *The Annals of Applied Probability*, 20(2): 565-592, 2010.

[8] C. Domb and M. S. Green. Phase Transitions and Critical Phenomena. *Vol. 2. Academic Press. London*, 1972.

[9] C. D. Forney, Jr. Codes on Graphs: News and Views. *Conference on Information Sciences and Systems.* The John Hopkins University, 2001.

[10] D. A. Forsyth, J. Haddon and S. Ioffe. The Joy of Sampling. *International Journal of Computer Vision*, 41(1):109-134, 2001.

[11] W. T. Freeman and E. C. Pasztor. Learning Low Level Vision. *In Proceeding of International Conference of Computer Vision*, 1999.

[12] T. Heskes. On the uniqueness of loopy belief propagation fixed points. *Neural Computation*, 16(11):2379. 2413, 2004.

[13] A. T. Ihler, J. W. Fischer III and A. S. Willsky. Loopy Belief Propagation: Convergence and Effects of Message Errors. *The Journal of Machine Learning Research*, 6:905-936, 2006

[14] S. L. Lauritzen. *Graphical models.* Oxford University Press, USA, 1996.

[15] R. J. McEliece, D. J. C. Mackay and J. F. Cheng. Turbo decoding as an instance of Pearl's belief propagation algorithm. *IEEE Journal on Selected Areas in Communication*, 16(2):140-152, 1998.

[16] K. P. Murphy, Y. Weiss and M. Jordan. Loopy belief propagation for approximate inference: an empirical study. *In Proceedings of Uncertainty in Artificial Intelligence*, 1999.

[17] J. Pearl. Reverend Bayes on inference engines: A distributed hierarchical approach. *Proceedings of the Second National Conference on Artificial Intelligence*, AAAI Press, USA, 1982.

[18] F. Ricci-Tersenghi and G. Semerjian. On the cavity method for decimated random constraint satisfaction problems and the analysis of belief propagation guided decimation algorithms. *J. Stat. Mech.*, 2009

[19] E. B. Sudderth, M. J. Wainwright, and A. S. Willsky. Loop series and Bethe variational bounds in attractive graphical models. *Advances in neural information processing systems*, 20:1425–1432, 2008.

[20] S. Tatikonda and M. Jordan. Loopy belief propagation and gibbs measures. *Uncertainty in Artificial Intelligence*, 2002.

[21] Y. W. Teh and M. Welling. Belief Optimization for Binary Networks: A stable Alternative to Loopy Belief Propagation. *Uncertainty in Artificial Intelligence*, 2001.

[22] M. J. Wainwright, T. Jaakkola and A. S. Willsky. Tree-based reparameterization framework for analysis of sum-product and related algorithms. *IEEE Transactions on Information Theory*, 45(9):1120-1146, 2003.

[23] Y. Weiss. Correctness of local probability propagation in graphical models with loops. *Neural Computation*, 12(1), 2000.

[24] J. Yedidia, W. Freeman, and Y. Weiss. Constructing free energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, 51:2282-2312, 2004.

[25] A. L. Yuille. CCCP algorithms to minimize the Bethe and Kikuchi free energies: Convergent alternatives to belief propagation. *Neural Computation*, 14, 2002.