MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

# Constructing Free Energy Approximations and Generalized Belief Propagation Algorithms

Jonathan S. Yedidia, William T. Freeman, and Yair Weiss

TR2002-35 August 2002

**Abstract**

Note: This technical report is superseded by MERL TR2004-040, available at http://www.merl.com/papers/TR2004-040/.The region graph method is the most general of these methods, and it subsumes all the other methods. Region graphs also provide the natural graphical setting for GBP algorithms. We explain how to obtain three different versions of GBP algorithms and show that their fixed points will always correspond to stationary points of the region graph approximation to the free energy. We also show that the region graph approximation is exact when the region graph has no cycles.

MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

# Constructing Free Energy Approximations and Generalized Belief Propagation Algorithms

Jonathan S. Yedidia, William T. Freeman, and Yair Weiss

TR-2002-35 August 2002

**Abstract**

Important inference problems in statistical physics, computer vision, error-correcting coding theory, and artificial intelligence can all be reformulated as the computation of marginal probabilities on factor graphs. The belief propagation (BP) algorithm is an efficient way to solve these problems that is exact when the factor graph is a tree, but only approximate when the factor graph has cycles.

We show that BP fixed points correspond to the stationary points of the Bethe approximation to the free energy for a factor graph. We explain how to obtain region-based free energy approximations that improve the Bethe approximation, and corresponding generalized belief propagation (GBP) algorithms.

We emphasize the conditions a free energy approximation must satisfy in order to be a "valid" approximation. We describe the relationship between four different methods that can be used to generate valid approximations: the "Bethe method," the "junction graph method," the "cluster variation method," and the "region graph method."

The region graph method is the most general of these methods, and it subsumes all the other methods. Region graphs also provide the natural graphical setting for GBP algorithms. We explain how to obtain three different versions of GBP algorithms and show that their fixed points will always correspond to stationary points of the region graph approximation to the free energy. We also show that the region graph approximation is exact when the region graph has no cycles.

This technical report has been superseded by MERL TR2004-040, which was posted in May, 2004.

# Constructing Free Energy Approximations and Generalized Belief Propagation Algorithms

Jonathan S. Yedidia †, William T. Freeman ‡, and Yair Weiss §

*Abstract*— **Important inference problems in statistical physics, computer vision, error-correcting coding theory, and artificial intelligence can all be reformulated as the computation of marginal probabilities on factor graphs. The belief propagation (BP) algorithm is an efficient way to solve these problems that is exact when the factor graph is a tree, but only approximate when the factor graph has cycles.**

**We show that BP fixed points correspond to the stationary points of the Bethe approximation to the free energy for a factor graph. We explain how to obtain region-based free energy approximations that improve the Bethe approximation, and corresponding generalized belief propagation (GBP) algorithms.**

**We emphasize the conditions a free energy approximation must satisfy in order to be a "valid" approximation. We describe the relationship between four different methods that can be used to generate valid approximations: the "Bethe method," the "junction graph method," the "cluster variation method," and the "region graph method."**

**The region graph method is the most general of these methods, and it subsumes all the other methods. Region graphs also provide the natural graphical setting for GBP algorithms. We explain how to obtain three different versions of GBP algorithms and show that their fixed points will always correspond to stationary points of the region graph approximation to the free energy. We also show that the region graph approximation is exact when the region graph has no cycles.**

## I. INTRODUCTION

Problems involving probabilistic inference using graphical models are important in a wide variety of disciplines, including statistical physics, signal processing, artificial intelligence, and digital communications [1], [2]. Message-passing algorithms are a practical and powerful way to solve such problems. The centrality of such problems and the utility of message-passing algorithms for solving them is an explanation for the fact that equivalent or very closely-related message-passing algorithms have now been independently invented many times. They are well-known by names like the forward-backward algorithm for Hidden Markov Models [3], the Viterbi algorithm [4], [5], Gallager's sum-product algorithm for decoding low-density parity check codes [6], the "turbo-decoding" algorithm [7], [8], Pearl's "belief propagation" algorithm for inference on Bayesian networks [9], the "Kalman filter" for signal processing [10], [11], and the "transfer matrix" approach in statistical mechanics [12].

† MERL Cambridge Research Lab, 201 Broadway, 8th Floor, Cambridge MA 02139. yedidia@merl.com

‡ Electrical Engineering and Computer Science, MIT Artificial Intelligence Laboratory, NE43a, Cambridge MA 02139. wtf@ai.mit.edu

§ School of Computer Science and Engineering, The Hebrew University of Jerusalem, 91904 Jerusalem, Israel. yweiss@cs.huji.ac.il

In this list of "standard" belief propagation (BP) algorithms, we have blurred a distinction between two different objectives that one might have, and the slightly different algorithms that result. Sometimes, one might be interested in obtaining the one global state of a system that is most probable or otherwise optimal. In other cases, one is interested in obtaining marginal probabilities for some subset of the nodes of the system, given evidence about other nodes in the system. In this paper, we will focus exclusively on this latter problem.

In all standard BP algorithms, messages are sent from one node in a graphical model to a neighboring node. The algorithms are exact when the graphical model is free of cycles. Thus, a common approach for dealing with graphical models that do have cycles is to try to convert them to equivalent cycle-free graphical models, and then to use the standard BP algorithm [13]. In some cases, this is possible, but for many other cases of practical interest, such an approach is intractable, and one must settle for approximate methods.

Fortunately, the standard BP algorithms are well-defined, and often give surprisingly good approximate results, for graphical models with cycles. Nevertheless, in such cases there are no guarantees, and sometimes the results are quite poor, or the algorithm fails to give any result at all because it does not converge [14]. Two major goals of this paper are to explain why the standard BP algorithm often works so well even for graphical models with cycles, and to use that understanding to develop improved algorithms for cases when it does not work well.

The class of algorithms that we will describe, which we call *generalized belief propagation* (GBP) algorithms, all have the characteristic that sets or *regions* of nodes will send messages to other regions of nodes. The regions of nodes that communicate with each other can be easily visualized in terms of a *region graph*. The standard BP algorithm is a special case of a GBP algorithm, with a particular choice of regions. Different choices of region graphs will give different GBP algorithms, and the user can choose to trade off complexity for accuracy.

In practice, GBP algorithms can often dramatically outperform BP algorithms in terms of either their accuracy or their convergence properties, for minimal computational cost, if one makes an intelligent choice of regions. However, how to optimally choose regions for a GBP algorithm remains at this point more an art than a science. We hope that this paper contributes to this problem by delineating what classes of constructions are likely to give good results.

We shall give a theoretical justification of GBP algorithms by showing that their fixed points are identical to the stationary points of a *region-based free energy*, which is an approximation to another free energy that can be justified by a rigorous variational principle. The first specialized examples of such free energies were introduced long ago in the physics literature by by Bethe [15] and Kikuchi [16]. For the important special case of the standard BP algorithm, we show that its fixed points are the same as the stationary points of the *Bethe free energy*, thus establishing an important basic link between a classical algorithm and a classical approximation from physics.

One must be careful in constructing a region graph in order to ensure that the resulting approximations are accurate. In our original work introducing GBP algorithms [17], we focused on a sub-class of GBP algorithms that were equivalent to free energy approximations based on Kikuchi's *cluster variation method* [16], [18], [19]. We shall show that this method is only one of a variety of methods to generate region graphs and their corresponding free energies and message-passing algorithms.

In our original work, we also focused on graphical models defined in terms of pair-wise or higher-order Markov random fields (MRFs). In this paper, we shall instead focus on graphical models defined in terms of *factor graphs.* All our results can be re-expressed for other graphical models without difficulty. Using factor graphs has certain practical advantages–in particular we can refer the neophyte reader to the excellent review by Kschischang et.al. [20]. That review explains the equivalence to factor graphs of other graphical models such as Bayesian networks, Tanner graphs for error-correcting codes, or pair-wise MRFs, and explains the standard BP algorithm in its various guises as an algorithm that operates on factor graphs.

Other formulations of the standard BP algorithm provide different insights, and we refer the interested reader to a number of important recent papers that exploit alternative views of the BP algorithm [21], [22], [23], [24], [25], [26].

After our original work which introduced region-based free energies and GBP algorithms based on the cluster variation method, Aji and McEliece introduced a class of free energy approximations and GBP algorithms based on *junction graphs* [27]. One of the goals of this paper is to unify our previous approach with the one that Aji and McEliece presented. McEliece and Yildirim have independently developed a unified approach to belief propagation which is largely equivalent to our region graph approach, and we recommend their elegant exposition [28]. Pakzad and Anantharam have also recently presented parallel ideas in a brief paper [29].

The outline for the rest of the paper is as follows. In section II, we review and introduce our notation for factor graphs and the standard BP algorithm. In sections III and IV, we introduce and explain the physical intuition behind variational free energies and region-based approximations to them. In section V, we consider the *Bethe Method* which can be used to obtain particularly simple region-based free energy approximations. We also show in that section that the standard BP algorithm has fixed points equivalent to the stationary points of the Bethe approximation to the free energy. In section VI, we develop a theory that can be used to determine which region-based free energy approximations will be likely to give accurate results. In particular, we describe the *Region Graph Method*, a very general method for generating valid region graphs and their associated free energies. In section VII, we introduce GBP algorithms, and show that there are actually a variety of ways to define GBP algorithms for any given region graph, all of which have identical fixed points. We focus on one particular type of GBP algorithm, which we call the *parent-to-child* algorithm. Finally, in section VIII, we give a detailed example of the implementation of the parent-to-child GBP algorithm.

We have chosen to put an unusually large amount of material in the appendices of this paper. We did this in an attempt to help the reader grasp the fundamental concepts behind our work and not lose sight of the forest because of all the trees. The appendices describe a variety of other methods to generate region graphs and GBP algorithms which could easily prove to be as important in practice as the methods described in the main text.

## II. Factor Graphs and Belief Propagation

Let $\{X_1, X_2, ..., X_N\}$ be a set of $N$ discrete-valued random variables and let $x_i$ represent the possible realizations of random variable $X_i$. We consider the joint probability mass function $p(X_1 = x_1, X_2 = x_2, ..., X_N = x_N)$, which we shall write more succintly as $p(\mathbf{x})$, where $\mathbf{x}$ stands for $\{x_1, x_2, ..., x_N\}$. We suppose that $p(\mathbf{x})$ factors into a product of functions. That is, we suppose that $p(\mathbf{x})$ has the very general form

$$p(\mathbf{x}) = \frac{1}{Z} \prod_a f_a(\mathbf{x}_a). \tag{1}$$

Here $a$ is an index labeling $M$ functions $f_A, f_B, f_C, ..., f_M$, where the function $f_a(\mathbf{x}_a)$ has arguments $\mathbf{x}_a$ that are some subset of $\{x_1, x_2, ..., x_N\}$. $Z$ is a normalization constant.

A *factor graph* [20] is a bipartite graph that expresses the factorization structure in equation (1). A factor graph has a *variable node* (which we draw as a circle) for each variable $x_i$, a *factor node* (which we draw as a square) for each function $f_a$, with an edge connecting variable node $i$ to factor node $a$ if and only if $x_i$ is an argument of $f_a$. (We shall always index variable nodes with letters starting with $i$, and factor nodes with letters starting with $a$.) As an example, the factor graph corresponding to

$$p(x_1, x_2, x_3, x_4) = \frac{1}{Z} f_A(x_1, x_2) f_B(x_2, x_3, x_4) f_C(x_4) \tag{2}$$

in shown in figure 1.

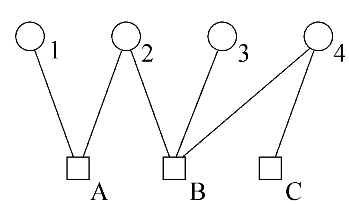


Fig. 1. A small factor graph representing the joint probability distribution $p(x_1, x_2, x_3, x_4) = \frac{1}{Z} f_A(x_1, x_2) f_B(x_2, x_3, x_4) f_C(x_4)$.

We shall focus on the problem of computing marginal probability distributions. We call the possible values of $X_i$ the *states* of variable node $i$. If $S$ is a set of variable nodes, we use $\mathbf{x}_S$ to denote the states of the corresponding variable nodes. $p_S(\mathbf{x}_S)$ will denote the marginal probability function obtained by marginalizing $p(\mathbf{x})$ onto the set of variable nodes $S$, i.e.,

$$p_S(\mathbf{x}_S) = \sum_{\mathbf{x} \backslash \mathbf{x}_S} p(\mathbf{x}). \tag{3}$$

Here the sum over $\mathbf{x} \backslash \mathbf{x}_S$ indicates that we sum over the states of all the variable nodes *not* in the set $S$. We shall write $p_i(x_i)$ for the marginal probability function when the set $S$ consists of the single node $i$. One should note that the problem of computing marginal probability functions is in general hard because it can require summing an exponentially large number of terms.

The *belief propagation* (BP) algorithm is a method for computing marginal probability functions. We describe the algorithm in terms of operations on a factor graph. As we already mentioned in the introduction, the computed marginal probability functions will be exact if the factor graph has no cycles, but the BP algorithm is still well-defined and empirically often gives good approximate answers even when the factor graph does have cycles.

To define the BP algorithm, we first introduce *messages* between variable nodes and their neighboring factor nodes and vice versa. The message $m_{a \to i}(x_i)$ from the factor node $a$ to the variable node $i$ is a vector over the possible states of $x_i$. This message can be interpreted as a statement from factor node $a$ to variable node $i$ about the relative probabilities that $i$ is in its different states, based on the function $f_a$. The message $n_{i \to a}(x_i)$ from the variable node $i$ to the factor node $a$ may in turn be interpreted as a statement about the relative probabilities that node $i$ is in its different states, based on all the information $i$ has *except* for that based on the function $f_a$.

The messages are initialized to $m_{a \to i}(x_i) = n_{i \to a}(x_i) = 1$ for all factor nodes $a$, variable nodes $i$, and states $x_i$. In fact, other initializations are also possible, and the overall normalization of the messages can also be chosen arbitrarily. The only important normalization condition is on the beliefs, introduced below, which must sum to one in order to properly represent probabilities. The messages are updated according to the following rules:

$$n_{i \to a}(x_i) := \prod_{b \in N(i) \backslash a} m_{b \to i}(x_i). \tag{4}$$

and

$$m_{a \to i}(x_i) := \sum_{\mathbf{x}_a \backslash x_i} f_a(\mathbf{x}_a) \prod_{j \in N(a) \backslash i} n_{j \to a}(x_j) \tag{5}$$

Here, $N(i) \backslash a$ denotes all the nodes that that are neighbors of node $i$ except for node $a$, and $\sum_{\mathbf{x}_a \backslash x_i}$ denotes a sum over all the variables $\mathbf{x}_a$ that are arguments of $f_a$ except $x_i$. The messages may be normalized in any way that is convenient, as only the ratios of the terms in a message are relevant. This standard BP algorithm is sometimes called the "sum-product" algorithm because of the sum and product that occurs on the right-hand-side of equation (5).

In some cases, it is convenient to eliminate the $n_{i \to a}(x_i)$ messages and write the message-update equations entirely in terms of the $m_{a \to i}(x_i)$ messages. Alternatively, of course, one could choose to eliminate the $m_{a \to i}(x_i)$ messages in favor of the $n_{i \to a}(x_i)$ messages.

These message-update rules may initially appear quite mysterious–a major goal of this paper will be to explain, justify, and ultimately improve upon them. First though, to complete our preliminary description of the standard BP algorithm, we introduce the *belief* $b_i(x_i)$ at a variable node $i$, which is the BP approximation to the exact marginal probability function $p_i(x_i)$. The belief $b_i(x_i)$ can be computed from the equation

$$b_i(x_i) \propto \prod_{a \in N(i)} m_{a \to i}(x_i), \tag{6}$$

where we have used the proportionality symbol $\propto$ to indicate that one must normalize the beliefs so that they sum to one. The BP message-update equations are iterated until they (hopefully) converge, at which point the beliefs can be read off from equation (6).

We can also use the BP algorithm to compute joint beliefs $b_S(\mathbf{x}_S)$ over sets of variable nodes $S$ that may contain more than one node. Consider the important case when the set $S$ consists of all the variable nodes attached to the $a$th function $f_a(\mathbf{x}_a)$. We will denote the corresponding belief by $b_a(\mathbf{x}_a)$, which will be given within the BP approximation by

$$\begin{aligned} b_a(\mathbf{x}_a) &\propto f_a(\mathbf{x}_a) \prod_{i \in N(a)} n_{i \to a}(x_i) \\ &\propto f_a(\mathbf{x}_a) \prod_{i \in N(a)} \prod_{b \in N(i) \backslash a} m_{b \to i}(x_i). \end{aligned} \tag{7}$$

We can directly *derive* the message update rules (4) and (5) from the belief equations (6) and (7), along with the marginalization condition

$$b_i(x_i) = \sum_{\mathbf{x}_a \backslash x_i} b_a(\mathbf{x}_a) \tag{8}$$

which holds when $x_i$ is one of the arguments in the set $\mathbf{x}_a$. Thus, the belief equations (6) and (7) can be considered to define the BP algorithm, a point of view that will prove useful later.

The BP algorithm is normally justified as being an exact algorithm when the factor graph has no cycles (i.e., it has the topology of a tree.) We shall not prove that property here, but will simply give a small example: consider the joint probability distribution given by equation (2) as illustrated in figure 1. Suppose that we would like to compute $p_1(x_1)$, the marginal probability distribution at variable node 1. Repeatedly using the BP equations, we find

$$\begin{aligned} b_1(x_1) &\propto m_{A \to 1}(x_1) \\ &\propto \sum_{x_2} f_A(x_1, x_2) n_{2 \to A}(x_2) \\ &\propto \sum_{x_2} f_A(x_1, x_2) m_{B \to 2}(x_2) \\ &\propto \sum_{x_2} \sum_{x_3} \sum_{x_4} f_A(x_1, x_2) f_B(x_2, x_3, x_4) n_{3 \to B}(x_3) n_{4 \to B}(x_4) \end{aligned}$$

$$\propto \sum_{x_2}\sum_{x_3}\sum_{x_4} f_A(x_1,x_2) f_B(x_2,x_3,x_4) m_{C\to 4}(x_4)$$
$$\propto \sum_{x_2}\sum_{x_3}\sum_{x_4} f_A(x_1,x_2) f_B(x_2,x_3,x_4) f_C(x_4) \tag{9}$$

which is exactly the desired marginal probability function. We could similarly demonstrate that equation (7) would give exact multi-node marginal probabilities for graphs with no cycles. We can already see from this example that for graphs with no cycles, the BP algorithm is essentially a dynamic programming algorithm that organizes the computations necessary to compute marginal probability distributions in such a way that they become tractable.

The BP algorithm was introduced into the coding literature by Gallager as a sub-optimal probabilistic decoding algorithm for linear block error-correcting codes, and some readers may be most familiar with the BP algorithm in that context [6]. Other readers may be most familiar with the form of the BP algorithm introduced and popularized by Pearl [9] for probabilistic inference with Bayesian networks. Readers who are more familiar with the BP algorithm written on one of these forms may want to consult the review by Kschischang et.al. [20], which explains the equivalence between these forms of the BP algorithm and the one we have chosen to use here.

## III. Free Energies

In this section, we turn from simply describing the BP algorithm to explaining its success. In section II, we saw that the BP algorithm can be defined in terms of the belief equations (6) and (7). We shall eventually show that these belief equations correspond to the stationarity conditions for a functional of the beliefs called the *Bethe free energy*, $F_{\mathrm{Bethe}}(b_i, b_a)$. This fact serves in some sense to justify the BP algorithm even when the factor graph it operates on has cycles, because minimizing the Bethe free energy is a sensible approximation procedure that has a long and successful history in physics. It also points to a variety of ways to improve upon or generalize BP, especially by improving upon the approximations used in the Bethe free energy. In the rest of the paper, we will discuss all of these issues, but we first turn to an explanation of the notion of a *free energy*.

Suppose that one has a system of $N$ particles, each of which can be in one of a discrete number of states, where the states of the $i$th particle are labeled by $x_i$. (As an example, one might make a variety of simplifications and characterize the states of the atoms in a magnetic crystal by whether a given electron in each atom has an "up" spin or a "down" spin.) The overall state of the system will be denoted by the vector $\mathbf{x} = \{x_1, x_2, ..., x_N\}$. Each state of the system has a corresponding *energy* $E(\mathbf{x})$. A fundamental result of statistical mechanics is that, in thermal equilibrium, the probability of a state will be given by *Boltzmann's Law*

$$p(\mathbf{x}) = \frac{1}{Z(T)} e^{-E(\mathbf{x})/T}. \tag{10}$$

Here, $T$ is the temperature, and $Z(T)$ is simply a normalization constant, known as the *partition function*:

$$Z(T) = \sum_{\mathbf{x}\in S} e^{-E(\mathbf{x})/T} \tag{11}$$

where $S$ is the space of all possible states $\mathbf{x}$ of the system.

A substantial part of statistical mechanics theory is devoted to the justification of Boltzmann's Law. On the other hand, if one begins with a joint probability distribution $p(\mathbf{x})$ for some non-physical system, one can view Boltzmann's law as a postulate that serves to define an energy for the system, where the temperature can be set arbitrarily, as it simply sets a scale for the units in which one measures energy. We shall take this point of view and set $T = 1$ throughout the rest of this paper. For the case of a factor graph probability distribution function $p(\mathbf{x}) = (1/Z)\prod_{a=1}^{M} f_a(\mathbf{x}_a)$, we define the *energy* $E(\mathbf{x})$ of a state $\mathbf{x}$ to be

$$E(\mathbf{x}) = -\sum_{a=1}^{M} \ln f_a(\mathbf{x}_a) \tag{12}$$

in order to be consistent with Boltzmann's Law.

The *Helmholtz free energy* $F_{\mathrm{Helmholtz}}$ of a system is

$$F_{\mathrm{Helmholtz}} = -\ln Z. \tag{13}$$

This free energy is a fundamentally important quantity in statistical mechanics, because if one can calculate the functional dependence of $F_{\mathrm{Helmholtz}}$ on quantities like a macroscopic magnetic field $H$ or temperature $T$, then it is easy to compute experimentally measurable quantities like the response of the system to a change in $H$ or $T$. Physicists have therefore devoted considerable energy to developing techniques which give good approximations to $F_{\mathrm{Helmholtz}}$.

One important technique is based on a variational approach. Suppose again that $p(\mathbf{x})$ is the true probability distribution of the system, which obeys Boltzmann's Law $p(\mathbf{x}) = e^{-E(\mathbf{x})}/Z$. It may be that even if we know $p(\mathbf{x})$ exactly, it is of a form that makes the computation of $F_{\mathrm{Helmholtz}}$ difficult. We therefore introduce a "trial" probability distribution $b(\mathbf{x})$, and a corresponding *variational free energy* (often called the *Gibbs free energy*) defined by

$$F(b) = U(b) - H(b). \tag{14}$$

where $U(b)$ is the *variational average energy*:

$$U(b) = \sum_{\mathbf{x}\in S} b(\mathbf{x}) E(\mathbf{x}) \tag{15}$$

and $H(b)$ is the *variational entropy*:

$$H(b) = -\sum_{\mathbf{x}\in S} b(\mathbf{x}) \ln b(\mathbf{x}). \tag{16}$$

It follows directly from our definitions that

$$F(b) = F_{\mathrm{Helmholtz}} + D(b||p) \tag{17}$$

where

$$D(b||p) \equiv \sum_{\mathbf{x}\in S} b(\mathbf{x}) \ln \frac{b(\mathbf{x})}{p(\mathbf{x})} \tag{18}$$

is the Kullback-Leibler divergence between $b(\mathbf{x})$ and $p(\mathbf{x})$. Since there exists a theorem [30] that $D(b||p)$ is always non-negative and is zero if and only if $b(\mathbf{x}) = p(\mathbf{x})$, we see that $F(b) \geq F_{\mathrm{Helmholtz}}$, with equality precisely when $b(\mathbf{x}) = p(\mathbf{x})$.

Minimizing the Gibbs free energy $F(b)$ is therefore an exact procedure for computing $F_{\text{Helmholtz}}$ and recovering $p(\mathbf{x})$. Of course, as $N$ becomes large, this procedure is also totally intractable, as $b(\mathbf{x})$ will take exponentially large memory just to store. A more practical possibility is to upper-bound $F_{\text{Helmholtz}}$ by minimizing $F(b)$ over a restricted class of probability distributions. This is the basic idea underlying the *mean field* approach. One very popular mean-field form for $b(\mathbf{x})$ is the factorized form:

$$b_{MF}(\mathbf{x}) = \prod_{i=1}^{N} b_i(x_i). \tag{19}$$

Using this $b_{MF}(\mathbf{x})$, and an energy function $E(\mathbf{x})$ of the factor graph form given in equation (12), we can easily compute the mean field free energy $F_{MF} = U_{MF} - H_{MF}$ for an arbitrary factor graph:

$$U_{MF}(\{b_1, ..., b_N\}) = -\sum_{a=1}^{M} \sum_{\mathbf{x}_a} \ln f_a(\mathbf{x_a}) \prod_{i \in N(a)} b_i(x_i), \tag{20}$$

$$H_{MF}(\{b_1, ..., b_N\}) = -\sum_{i=1}^{N} \sum_{x_i} b_i(x_i) \ln b_i(x_i). \tag{21}$$

Minimizing $F_{MF}(b_1, ..., b_N)$ over the $b_i$ will give us self-consistent equations for the $b_i$, which can be solved numerically to obtain a mean-field approximation for the beliefs $b_i(x_i)$.

Instead of a factorized form, one might consider other more complicated forms for $b(\mathbf{x})$ which still lead to tractable approximations. This is the idea behind the "structured mean-field" approach [31]. We will not follow that path, and will instead describe a quite different approach to approximating $F(b)$ in the next section; one which underlies the BP algorithm.

## IV. Region-based Free Energy Approximations

Kikuchi and the other physicists who further developed the so-called *cluster variation method* [16], [18], [19] introduced a class of approximations to the Gibbs free energy $F(b)$. The idea behind these approximations is similar, but slightly different from the mean field approximation. Whereas the factorized mean-field free energy $F_{MF}$ is a function of single-node beliefs $b_i(x_i)$, in a Kikuchi approximation, the approximate free energy $F_{Kikuchi}$ will be a function of beliefs $b_S(\mathbf{x}_S)$ over larger sets $S$ of variable nodes. One drawback of the cluster variation method is that in contrast with the mean-field approach, one cannot normally explicitly construct an overall "trial" belief vector $b(\mathbf{x})$ that is consistent with the multi-node beliefs $b_S(\mathbf{x}_S)$, and therefore one does not normally obtain any upper bound on $F$ [32]. On the other hand, one can make approximations that are much more accurate than the factorized mean-field approximation, and there is a great deal of flexibility in the exact choice of approximation. As we shall also see in further detail, these approximations can be exploited to yield message-passing algorithms, and a particularly simple version–the Bethe approximation–will give results that are equivalent to the standard BP algorithm.

We shall actually describe here a class of approximations that generalize those generated by the cluster variation method as it has been described in the physics literature, and will therefore refer to such approximations as *region-based approximations*. We refer to the sub-class of approximations specifically generated using the cluster variation method as *Kikuchi approximations*.

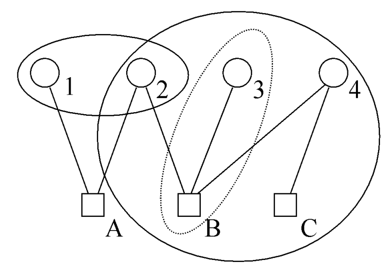


Fig. 2. An illustration of the definition of a *region*. Regions are sets of variable and factor nodes in a factor graph such that all variable nodes connected to any included factor nodes are included. Thus, the sets of nodes $\{1, 2\}$ and $\{B, C, 2, 3, 4\}$ could be regions, but $\{B, 3\}$ could *not* be a region (since factor node $B$ was included, variable nodes 2 and 4 must also be included.)

We begin by assuming that $p(\mathbf{x})$ has the factor graph form of equation (1). We define a *region* $R$ of a factor graph to be a set $V_R$ of variable nodes and a set $F_R$ of factor nodes, such that if a factor node $a$ belongs to $F_R$, all the variable nodes neighboring $a$ are in $V_R$. We give examples of sets of nodes that could or could not be considered regions in figure 2. Note that the set $F_R$ may be empty, and that a factor $a$ need not be included in $F_R$ even if all its neighboring variable nodes are in $V_R$.

We define the state $\mathbf{x}_R$ of a region $R$ to be the collective set of variable node states $\{x_i | i \in V_R\}$. The marginal probability function over a region $R$ will be denoted by $p_R(\mathbf{x}_R)$, by which we mean a marginalization of $p(\mathbf{x})$ onto the variable nodes in $V_R$. The corresponding belief $b_R(\mathbf{x}_R)$ will be an approximation to the true $p_R(\mathbf{x}_R)$.

We define the *region energy* $E_R(\mathbf{x}_R)$ to be

$$E_R(\mathbf{x}_R) = -\sum_{a \in F_R} \ln f_a(\mathbf{x}_a). \tag{22}$$

where, since all the variable nodes neighboring a factor node $a \in F_R$ are guaranteed to be in the region $R$, we can always determine any needed state $\mathbf{x}_a$ from the state $\mathbf{x}_R$. We further define the *region average energy* $U_R(b_R)$, the *region entropy* $H_R(b_R)$, and the *region free energy* $F_R(b_R)$, by

$$U_R(b_R) = \sum_{\mathbf{x}_R} b_R(\mathbf{x}_R) E_R(\mathbf{x}_R) \tag{23}$$

$$H_R(b_R) = -\sum_{\mathbf{x}_R} b_R(\mathbf{x}_R) \ln b_R(\mathbf{x}_R) \tag{24}$$

and

$$F_R(b_R) = U_R(b_R) - H_R(b_R). \tag{25}$$

The intuitive idea behind a region-based free energy approximation is that we will try to break up the factor graph into a set of large regions that include every factor and variable node, and say that the overall free energy is the sum of the free energies of all the regions. Of course, if some of the large regions overlap, then we will have erred by counting the free energy contributed by some nodes two or more times, so we then need to subtract out the free energies of these overlap regions in such a way that each factor and variable node is counted exactly once.

To make these notions precise, we say that a region-based approximation $F_{\mathcal{R}}$ for the Gibbs free energy will be defined in terms of a set of regions $\mathcal{R}$, and an associated set of *counting numbers* $c_R$. $c_R$ will always be an integer, but might be zero or negative for some $R$.

We say that a set of regions $\mathcal{R}$ and counting numbers $c_R$ give a *valid* region-based approximation when, for every factor node $a$ and every variable node $i$ in the factor graph,

$$\sum_{R \in \mathcal{R}} [a \in F_R]\, c_R = \sum_{R \in \mathcal{R}} [i \in V_R]\, c_R = 1, \qquad (26)$$

where $[z \in Z]$ is the set-membership indicator function equal to 1 if $z \in Z$ and equal to 0 otherwise.

These conditions ensure that every factor and variable node will be counted exactly one time in the approximation to the free energy. If a given factor or variable node is added into the free energy in two different regions, then there must be another region where it is subtracted back out.

Given a valid set of regions $\mathcal{R}$ and counting numbers $c_R$, the region-based approximation to the Gibbs free energy is simply

$$F_{\mathcal{R}}(\{b_R\}) = \sum_{R \in \mathcal{R}} c_R F_R(b_R). \qquad (27)$$

Note that the region-based average energy

$$\begin{aligned} U_{\mathcal{R}}(\{b_R\}) &= \sum_{R \in \mathcal{R}} c_R U_R(b_R) \\ &= -\sum_{R \in \mathcal{R}} c_R \sum_{\mathbf{x}_R} b_R(\mathbf{x}_R) \sum_{a \in F_R} \ln f_a(\mathbf{x}_a) \qquad (28) \end{aligned}$$

will always be *exact*, provided that the beliefs $\{b_R(\mathbf{x}_R)\}$ are equal to the corresponding exact marginal probabilities $\{p_R(\mathbf{x}_R)\}$. We can see this by comparing with the exact average energy

$$U = \sum_{\mathbf{x} \in S} p(\mathbf{x}) E(\mathbf{x}) = -\sum_{a=1}^{M} \sum_{\mathbf{x}_a} p_a(\mathbf{x}_a) \ln f_a(\mathbf{x}_a) \qquad (29)$$

and noting that in the overcounting numbers $c_R$ guarantee that each factor node is counted exactly once in equation (28), and that if all the $\{b_R\}$ are exact in equation (28), they will properly marginalize to give the necessary factors of $p_a(\mathbf{x}_a)$ in equation (29).

On the other hand, the region-based entropy

$$\begin{aligned} H_{\mathcal{R}}(\{b_R\}) &= \sum_{R \in \mathcal{R}} c_R H_R(b_R) \\ &= -\sum_{R \in \mathcal{R}} c_R \sum_{\mathbf{x}_R} b_R(\mathbf{x}_R) \ln b_R(\mathbf{x}_R) \qquad (30) \end{aligned}$$

will normally only be an approximation even if the beliefs $b_R(\mathbf{x}_R)$ were exactly equal to the true marginal probabilities, although the condition that each variable node is counted once makes it a quite "reasonable" approximation, in the sense that if the probability distribution $p(\mathbf{x})$ was flat, this entropy would at least count the number of degrees of freedom correctly. The region-based entropy will also be exact in certain cases that we describe later.

How does one select a valid set of regions $\mathcal{R}$ and counting numbers $c_R$ for a given factor graph? There are in fact an infinite number of ways to do that. In the next section we will describe a very straightforward approach which we call the *Bethe method*, which is guaranteed to return valid sets of regions and counting numbers. We then prove that the fixed points of the standard BP algorithm correspond to stationary points of the Bethe approximation to the free energy.

In the following section, we will introduce the *region graph method*, which is a very general approach to finding valid approximations, based on constructing a *region graph*. Region graphs play a central role in the description both of the region graph free energy, and in the construction of corresponding GBP algorithms, and provide the clear way of visualizing and understanding a region-based approximation.

The Bethe method is a special case of the much more general region graph method. In appendices A and B, we discuss two other important methods that are also special cases of the region graph method: the *junction graph method* and the *cluster variation method*. In appendix C, we discuss in detail the relationship between the different methods.

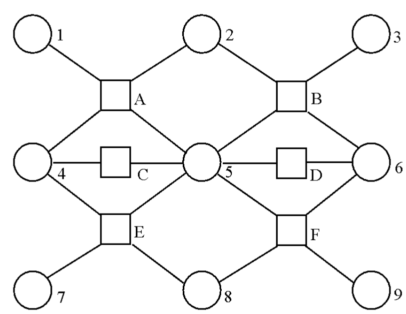


Fig. 3. A factor graph which we use to illustrate a variety of region-based free energy approximations.

## V. The Bethe Method

The origins of the Bethe method date back to 1935, and Bethe's famous approximation method for magnets [15]. Kikuchi, in his 1951 paper that pioneered the cluster variation method [16], recognized that Bethe's approximation was the simplest example of an approximation that could be generated using that method. Of course, from the modern point of view, these early papers focused on very special graphical models, and we warn the reader who wants to read the original papers that our description of Bethe's and Kikuchi's methods will bear little resemblance to their expositions.

In the *Bethe method*, we take the set of regions included in $\mathcal{R}$ to be of two types. First, we have a set of *large* regions $\mathcal{R}_L$ such that the $M$ regions in $\mathcal{R}_L$ each contain exactly one factor node and all the variable nodes neighboring that factor node. Second, we have a set of *small* regions $\mathcal{R}_S$, such that the $N$ regions in $\mathcal{R}_S$ each contain a single variable node.

We take as an example the factor graph shown in figure 3, which has six factor nodes which we label $A, B, C, D, E, F$ and nine variable nodes which we label $1, 2, ..., 9$. For this example, we would have the following large regions in $\mathcal{R}_L$: $\{A, 1, 2, 4, 5\}$, $\{B, 2, 3, 5, 6\}$, $\{C, 4, 5\}$, $\{D, 5, 6\}$, $\{E, 4, 5, 7, 8\}$, and $\{F, 5, 6, 8, 9\}$, and the following small regions in $\mathcal{R}_S$: $\{1\}$, $\{2\}$, $\{3\}$, $\{4\}$, $\{5\}$, $\{6\}$, $\{7\}$, $\{8\}$, and $\{9\}$. The complete set of regions $\mathcal{R}_{\text{Bethe}}$ included in the Bethe approximation is $\mathcal{R}_{\text{Bethe}} = \mathcal{R}_L \cup \mathcal{R}_S$.

If $R_1$ and $R_2$ are two regions, we say that $R_1$ is a *sub-region* or $R_2$ and $R_2$ is a *super-region* of $R_1$ if the set of variable and factor nodes in $R_1$ are a subset of those in $R_2$.

In the Bethe method, the counting numbers $c_R$ for each region $R \in \mathcal{R}$ are given by

$$c_R = 1 - \sum_{S \in \mathcal{S}(R)} c_S \tag{31}$$

where $\mathcal{S}(R)$ is the set of regions that are super-regions of $R$.

Using this definition we see that for every region $R \in \mathcal{R}_L$, $c_R = 1$, while for every region $R \in \mathcal{R}_S$, $c_R = 1 - d_i$, where $d_i$ is the degree (number of neighboring factor nodes) of the variable node $i$. It is easy to confirm that the Bethe approximation will always be a *valid* approximation, as each factor and variable node will clearly be counted once as required in equation (26).

We can use our expressions for $c_R$ in equation (28) to obtain the *Bethe approximation* to the Gibbs free energy $F_{\text{Bethe}} = U_{\text{Bethe}} - H_{\text{Bethe}}$, where

$$U_{\text{Bethe}} = -\sum_{a=1}^{M} \sum_{\mathbf{x}_a} b_a(\mathbf{x}_a) \ln f_a(\mathbf{x}_a) \tag{32}$$

and

$$\begin{aligned} H_{\text{Bethe}} = & - \sum_{a=1}^{M} \sum_{\mathbf{x}_a} b_a(\mathbf{x}_a) \ln b_a(\mathbf{x}_a) \\ & + \sum_{i=1}^{N} (d_i - 1) \sum_{x_i} b_i(x_i) \ln b_i(x_i). \end{aligned} \tag{33}$$

Note that the Bethe entropy will be exact if the factor graph has no cycles, because in that case we have the exact formula [13]

$$p(\mathbf{x}) = \frac{\prod_{a=1}^{M} p_a(\mathbf{x}_a)}{\prod_{i=1}^{N} (p_i(x_i))^{d_i - 1}}, \tag{34}$$

which we can substitute into the formula for the variational entropy to recover $H_{\text{Bethe}}$.

We shall now show that minimizing the Bethe approximation to the free energy will always give results that are equivalent to the standard BP algorithm, so the exactness of the Bethe approximation for factor graphs with no cycles is no surprise.

### A. Equivalence of the Bethe Approximation and Standard BP

We now show that the standard BP algorithm is equivalent to the Bethe approximation, and explore some of the implications of that equivalence. In particular, we show that the "messages" sent in BP are exponentiated combinations of Lagrange multipliers.

*Theorem:* Let $\{m_{a \to i}(x_i), n_{i \to a}(x_i)\}$ be a set of BP messages and let $\{b_a(\mathbf{x}_a), b_i(x_i)\}$ be the beliefs calculated from those messages. Then the beliefs are fixed points of the BP algorithm if and only if they are zero gradient points of the Bethe free energy $F_{\text{Bethe}}$, subject to the constraint that all the beliefs are normalized and consistent.

*Proof:* We want to minimize the Bethe free energy, while insisting that all the beliefs $b_i(x_i)$ and $b_a(\mathbf{x}_a)$ are consistent. To this end, we add Lagrange multipliers $\lambda_a(x_i)$ which enforce the constraint that $\sum_{\mathbf{x}_a \setminus x_i} b_a(\mathbf{x}_a) = b_i(x_i)$ for every factor node $a$ and all its neighboring variable nodes $i$. We also need to add Lagrange multipliers to normalize the beliefs, but we do not clutter our equations with them, as their effects are automatically taken into account if we simply normalize our beliefs.

Setting the derivative of the resulting Lagrangian $L_{\text{Bethe}}$ with respect to the beliefs $b_i(x_i)$ and $b_a(\mathbf{x}_a)$ equal to zero gives:

$$b_a(\mathbf{x}_a) \propto f_a(\mathbf{x}_a) \prod_{i \in N(a)} e^{\lambda_{ai}(x_i)} \tag{35}$$

and

$$b_i(x_i) \propto \left( \prod_{a \in N(i)} e^{\lambda_{ai}(x_i)} \right)^{\frac{1}{d_i - 1}}. \tag{36}$$

If we make the identification

$$\lambda_{ai}(x_i) = \ln n_{ia}(x_i) = \ln \prod_{b \in N(i) \neq a} m_{a \to i}(x_i) \tag{37}$$

then we find that we recover the standard BP belief equations (6) and (7), which means that the standard BP fixed points correspond to stationary points of the constrained Bethe free energy.•

The fact that $L_{\text{Bethe}}$ is bounded below implies that the BP equations always possess a fixed point (obtained at the global minimum of $L_{\text{Bethe}}$). To our knowledge, this is the first proof of the existence of BP fixed points for a general graph with arbitrary potentials. Of course, the existence of a fixed point does not imply that the BP algorithm will converge starting from arbitrary initial conditions.

The conditions for the *uniqueness* of BP fixed points are also clarified by the equivalence with the Bethe approximation. In graphs with no more than a single cycle, it was known that if all factors are strictly positive ($f_a(\mathbf{x}_a) > 0$ for all $a$ and $\mathbf{x}_a$), then there was a unique BP fixed point.[33] For general graphs, we can use the equivalence established above to answer a question about the uniqueness of stationary points for the Bethe free energy. The issue of the number of stationary points of approximate free energies is well studied in physics. To be more precise, we can imagine defining a sequence of probability distributions where some or all of our original functions are all raised by a power: $f_a(\mathbf{x}_a; T) = f_a(\mathbf{x})^{1/T}$. This is equivalent to changing the temperature in a physical system, where

$T$ is the temperature. Many systems, for example Ising ferromagnets, will have different numbers of solutions above or below a *critical temperature* $T_c$ within the Bethe approximation [34]. Above $T_c$, the constrained free energy is convex and has a unique stationary point, while below $T_c$, there are multiple stationary points. Using this equivalence it is easy to define small factor graphs that show a similar behavior. Although the topology does not change and the factors are always positive, as we smoothly change the factors we go from a regime with a unique fixed point to one with multiple fixed points.

While we have shown that standard BP can only converge to stationary points of the constrained Bethe free energy, it is important to realize that BP does not perform constrained minimization of the Bethe free energy; i.e. it does not decrease $F_{\text{Bethe}}$ at every iteration. Indeed, the marginalization constraints are typically not satisfied at intermediate iterations of BP: it is only at a fixed point that the beliefs are in a feasible set. Based on the equivalence, first noted in our earlier work [17], others have recently devised algorithms that directly minimize the free energy on the feasible set [35], [36], [37]. Such free energy minimizations are somewhat slower than the BP algorithm, but they are guaranteed to converge.

## VI. The Region Graph Method

We now introduce *region graphs*, which are central to the region graph method for generating valid free energy approximations, and also will provide a graphical framework for GBP algorithms.

Let $I$ be the set of indices for the factor and variable nodes in a factor graph. A *region graph* is a labeled, directed graph $\mathcal{G} = (V, E, L)$ in which each vertex $v \in V$ (corresponding to a region) is labeled with a subset of $I$. We denote the label of vertex $v$ by $L(v)$. A directed edge (or *arc*) may exist pointing from vertex $v_p$ to vertex $v_c$ if $L(v_c)$ is a subset of $L(v_p)$. If such an arc exists, we say that $v_c$ is a *child* of $v_p$, that $v_p$ is a *parent* of $v_c$, and that they belong to different *generations*.. If there exists a directed path from vertex $v_a$ to vertex $v_d$, we say that $v_a$ is an *ancestor* of $v_d$, and $v_d$ is a *descendant* of $v_a$. Note that because of the transitivity of the subset relationship, a region graph must be a directed acyclic graph, in the sense that the arrows cannot loop around.

A region graph is closely related to the *Hasse diagram* for a *partially ordered set*, or *poset* [38], if we consider our regions to be organized into a poset, with the ordering relationship between the regions to be given by the ancestor-descendant relationship [28], [29]. There are, however, some differences between region graphs and Hasse diagrams. First, region graphs are labeled graphs, and we will insist on some "region graph conditions," described below, that the labels must satisfy. Second, region graphs can include an arc between two regions that are also connected by a path of length two or greater, which is forbidden for Hasse diagrams.

We define a counting number $c_v$ for every vertex in the region graph, by

$$c_v = 1 - \sum_{u \in A(u)} c_u, \tag{38}$$

where $A(u)$ is the set of vertices that are ancestors of $u$. Thus, the counting numbers for the regions of a region graph correspond to the *Möbius function* of the corresponding partially ordered set [38].

For a graph $\mathcal{G}$ to qualify as a region graph, we further insist on the *region graph condition*, which requires that for every $i \in I$, the subgraph $\mathcal{G}(i) = (V(i), E(i), L(i))$ formed by just those vertices whose labels include $i$ is a connected graph that satisfies the condition

$$\sum_{v \in V(i)} c_v = 1. \tag{39}$$

Having defined region graphs, it is almost trivial to define a corresponding method for generating valid region-based free energy approximations. We simply create a region graph such that the vertices correspond to regions, with labels corresponding to the factor and variable nodes in a region, and we require that every factor and variable node be contained in at least one region. We associate the counting numbers $c_R$ for regions directly with the counting numbers $c_v$ for the region graph, and the region graph free energy $F_{RG}$ will be given by $F_{RG} = \sum_R c_R F_R$, where $F_R$ is the free energy of the region $R$.

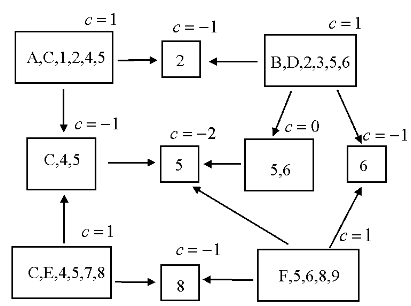


Fig. 4. An example of a region graph. We have listed the counting number $c_R$ next to each region.

In figure 4, we give an example of a region graph for the factor graph that we already introduced in figure 3. This region graph was constructed to demonstrate what is and is not permitted in a legal region graph, rather than what would likely give good results. Note that a region graph need not obey some properties that one might consider important (including some which are enforced in the junction graph method described in appendix A and the cluster variation method described in appendix B). For example there need not be any clear delineation of "generations" (region $\{8\}$ is a child of both regions $\{C, E, 4, 5, 7, 8\}$ and regions $\{F, 5, 6, 8, 9\}$, while region $\{5\}$ is a grand-child of region $\{C, E, 4, 5, 7, 8\}$ and a child of region $\{F, 5, 6, 8, 9\}$.) Note also that regions may have counting number equal to zero (e.g. region $\{5, 6\}$), and that the fact that a region is a sub-set of another region need not imply that it is also a descendant of that region (e.g. regions $\{F, 5, 6, 8, 9\}$ and $\{5, 6\}$).

What is essential is that the *region graph conditions* that we described above are obeyed. We insist on these conditions for the following reasons. First, to reiterate the comments we made about valid region-based free energy approximations, the condition that every factor node in the factor graph is counted once when we do the weighted sum over all regions ensures that the region graph average energy is exact if the region beliefs are exact; and the condition that every variable node is counted once ensures that the region graph entropy is a reasonable approximation. The condition that the regions containing a particular variable node form a connected sub-graph will ensure that the marginal probability at any node is *consistent* irrespective of which region's beliefs one uses to compute it. Empirically, we have found in limited experiments that if one attempts to run a GBP algorithm (as described below) on graphs that do not satisfy all the region graph conditions, the results will be poor.

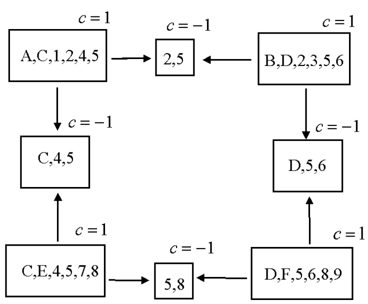


Fig. 5. An example of a graph of regions that is *not* a region graph because the sum of the counting numbers of regions containing variable node 5 is not one.

An example of a "false region graph" or graph of regions that does *not* satisfy the region graph conditions is shown in figure 5. The problem with this plausible-looking construction is that the sum of the counting numbers of the regions containing variable node 5 is zero, rather than one. We could modify this false region graph in a variety of ways to obtain a real region graph. For example, we could simply remove node 5 from the region $\{2, 5\}$. The resulting region graph would be an example of a *junction graph*; see appendix A. Alternatively, we could add a region $\{5\}$ which just contained variable node 5, and connect the regions $\{2, 5\}$, $\{C, 4, 5\}$, $\{D, 5, 6\}$, and $\{5, 8\}$ to it (the result of using the cluster variation method; see appendix B).

Just as the Bethe approximation will be exact when the factor graph is a tree, a region graph approximation will always be exact when the corresponding region graph is a tree. This can be demonstrated by recursively applying the following junction graph formula for the probability distribution of a factor graph divided into large regions $\mathcal{R}_L$, and small regions $\mathcal{R}_S$ which separate the large regions (see Appendix A for more details):

$$p(\mathbf{x}) = \frac{\prod_{R \in \mathcal{R}_L} p_R(\mathbf{x}_R)}{\prod_{R \in \mathcal{R}_S} p_R(\mathbf{x}_R)^{d_R - 1}}. \tag{40}$$

We illustrate the idea with an example, that has the factor graph given in figure 6, and the region graph given in figure 7. We will recursively break down the full joint probability distribution and show that it is equal to a product of marginal probability distributions over regions that has precisely the form necessary so that the region graph free energy is exact.

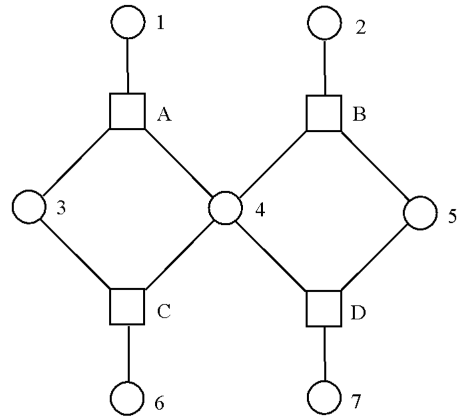


Fig. 6. A factor graph that has a tree region graph shown in figure 7.

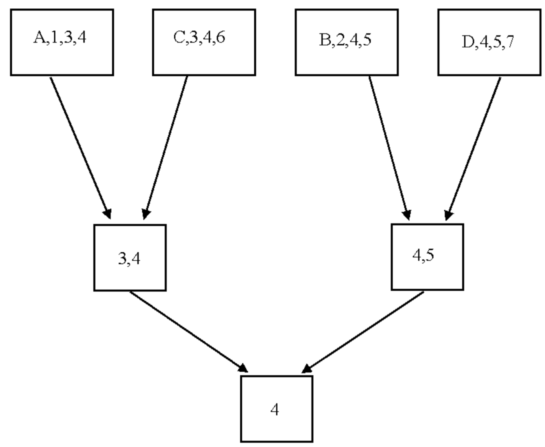


Fig. 7. A region graph with no cycles that has a corresponding region graph free energy approximation which is exact.

Note that for this region graph, the region $\{4\}$ separates the left part of the tree and the right part of the tree. That means that we have

$$p(x_1, ..., x_7) = \frac{p(x_1, x_3, x_4, x_6) p(x_2, x_4, x_5, x_7)}{p(x_4)}. \tag{41}$$

The marginal probability distributions $p(x_1, x_3, x_4, x_6)$ and $p(x_2, x_4, x_5, x_7)$ can in turn be written in terms of marginal probabilities of smaller regions. For example, we see that the region $\{3, 4\}$ separates the regions $\{A, 1, 3, 4\}$ and $\{C, 3, 4, 6\}$, so that

$$p(x_1, x_3, x_4, x_6) = \frac{p(x_1, x_3, x_4) p(x_3, x_4, x_6)}{p(x_3, x_4)}. \tag{42}$$

Expanding everything out, we obtain that the joint probability distribution $p(x_1, ..., x_7)$ equals

$$\frac{p(x_1, x_3, x_4) p(x_3, x_4, x_6) p(x_2, x_4, x_5) p(x_4, x_5, x_7)}{p(x_3, x_4) p(x_4, x_5) p(x_4)}. \tag{43}$$

Substituting this result into the formula for the exact entropy, we recover the region graph entropy. Since the region graph average energy is always exact when the region beliefs are, this demonstrates that the approximation is exact in this case.

We note that each term in the numerator of the expression (43) has a power of 1, and each term in the denominator has a power of $-1$, corresponding exactly to the counting number of the corresponding region. In the general case of a region graph with no cycles, the recursive application of the junction tree separator formula (40) will always similarly reproduce the counting numbers given by the region graph prescription equation (38).

We have already seen that the stationary points of the Bethe approximation to the free energy are equivalent to the fixed points of the standard BP algorithm, which operates on a factor graph. In the following sections, we shall introduce *generalized belief propagation* algorithms which operate on region graphs, and demonstrate that their fixed points correspond to the stationary points of the region graph free energy.

In appendices A, B, we discuss two other methods (the *junction graph method* and the *cluster variation method*) that generate valid free energy approximations. Both of these methods can be considered special cases of the region graph method. In appendix C, we describe the relationship between all the different methods described in this paper in more detail.

## VII. Generalized Belief Propagation Algorithms

Just as the standard BP algorithm corresponds to the Bethe approximation, one can construct generalized belief propagation (GBP) algorithms corresponding to any region graph free energy approximation. In fact, there are many ways to construct message-passing algorithms whose fixed points are equivalent to the stationary points of a region graph free energy. In all these algorithms, messages of some sort are sent between regions on a region graph.

One should first note that one can obtain different GBP algorithms corresponding to the same free energy by using different region graphs that have the same free energy. For example, one could modify a region graph by connecting a grandparent region directly to a grandchild region. The GBP algorithms that we describe below would be modified, but the approximate free energy would not be changed. Making such a modification will alter the dynamics of a GBP algorithm, but not its fixed points.

Even if one fixes one's attention on a particular region graph, there are still a variety of different GBP algorithms that one can create. In the main text of this paper, we will describe one possible approach, which we call the *parent-to-child algorithm*. In appendices D and E, we describe two other approaches (the *child-to-parent algorithm* and the *two-way algorithm*) which give algorithms with equivalent fixed points, and which have their own advantages. An important advantage of the parent-to-child algorithm is that the message-passing algorithm makes no reference to region counting numbers, just as in the standard BP algorithm.

The standard BP algorithm is a special case of all three algorithms when the region graph is obtained using the Bethe method.

### A. The Parent-to-Child Algorithm

As we saw, the standard BP message-passing equations can be derived using the fact that the belief at a single variable node is just the product of all the messages bearing information from neighboring factor nodes, while the belief at the region of variable nodes adjoining a single factor node is the product of their internal factors, multiplied by all the messages coming into the group of nodes from factor nodes outside the region.

The parent-to-child algorithm generalizes this idea. In this algorithm (which in previous expositions we called the "canonical" GBP algorithm [17]) the belief at any region $R$ will be the product of the local factors in that region, multiplied by all the messages coming into region $R$ from outside regions. There is one complication, however: to make the algorithm equivalent to minimizing the region graph free energy, we need to include additional messages into regions which are descendants of $R$ from other parent regions that are not themselves descendants of region $R$.

To be more specific, in the parent-to-child algorithm, we only have one kind of message $m_{P\to R}(\mathbf{x}_R)$ from a parent region to a child region. Each region $R$ has a belief $b_R(\mathbf{x}_R)$ given by

$$\begin{aligned} b_R(\mathbf{x}_R) \quad &\propto \quad \prod_{a\in A_R} f_a(\mathbf{x}_a) \left( \prod_{P\in\mathcal{P}(R)} m_{P\to R}(\mathbf{x}_R) \right) \ldots \\ &\ldots \quad \left( \prod_{D\in\mathcal{D}(R)} \prod_{P'\in\mathcal{P}(D)\backslash\mathcal{E}(R)} m_{P'\to D}(\mathbf{x}_D) \right) \end{aligned} \tag{44}$$

Here $\mathcal{P}(R)$ is the set of regions that are parents to region $R$, $\mathcal{D}(R)$ is the set of all regions that are descendants of region $R$, $\mathcal{E}(R) \equiv R \cup \mathcal{D}(R)$ is the set of all regions that are descendants of $R$ and also region $R$ itself, and $\mathcal{P}(D)\backslash\mathcal{E}(R)$ is the set of all regions that are parents of region $D$ except those that are also descendants of region $R$ or region $R$.

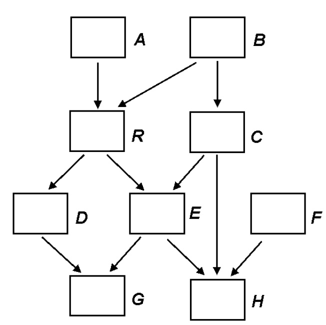


Fig. 8. A region graph used to illustrate the parent-to-child GBP algorithm. Note that we do not explicitly give the variable and factor node labels for each region, as for our purposes, we are only interested in the topology of the region graph.

An example may help make the belief equation clearer. Consider the example shown in figure 8. The belief $b_R(\mathbf{x}_R)$ at region $R$ is the product of its local factors $\prod_{a\in A_R} f_a(\mathbf{x}_a)$, the messages from its parents $m_{A\to R}(\mathbf{x}_R)$ and $m_{B\to R}(\mathbf{x}_R)$, and the messages into descendants from other parents who are not descendants: $m_{C\to E}(\mathbf{x}_E)$, $m_{C\to H}(\mathbf{x}_H)$, and $m_{F\to H}(\mathbf{x}_H)$.

One obtains self-consistent equations for the messages by requiring consistency between the beliefs between every pair of parent and child regions. Thus in figure 8, we might focus on the region $R$ and its child $E$. The belief at region $R$ is given by

$$b_R \propto m_{A\to R}\, m_{B\to R}\, m_{C\to E}\, m_{C\to H}\, m_{F\to H} \prod_{a\in A_R} f_a(\mathbf{x}_a) \tag{45}$$

(where we have lightened the notation by removing the obvious functional dependencies of the messages) and the belief at region $E$ is given by

$$b_E \propto m_{R\to E}\, m_{C\to E}\, m_{D\to G}\, m_{C\to H}\, m_{F\to H} \prod_{a\in A_E} f_a(\mathbf{x}_a) \tag{46}$$

Using the marginalization constraint, $b_R(\mathbf{x}_R) = \sum_{\mathbf{x}_A\setminus\mathbf{x}_R} b_A(\mathbf{x}_A)$ we obtain a relation between messages that we can interpret as the message update rule

$$m_{R\to E}(\mathbf{x}_E) m_{D\to G}(\mathbf{x}_G) := \sum_{\mathbf{x}_R\setminus\mathbf{x}_E} m_{A\to R}(\mathbf{x}_R) m_{B\to R}(\mathbf{x}_R) \prod_{a\in A_R\setminus A_E} f_a(\mathbf{x}_a). \tag{47}$$

Of course, similar message update rules would be obtained for all the pairs of parent and children regions. There will be enough conditions to determine every message.

In general, the parent-to-child message-update rules will be

$$m_{P\to R}(x_R) := \frac{\sum_{x_{P\setminus R}} \prod_{a\in F_{P\setminus R}} f_a(x_a) \prod_{(I,J)\in N(P,R)} m_{I\to J}(x_J)}{\prod_{(I,J)\in D(P,R)} m_{I\to J}(x_J)} \tag{48}$$

where the sets $N(P,R)$ and $D(P,R)$ can be calculated in advance. Recall that $\mathcal{E}(R) = R \cup \mathcal{D}(R)$. Then $N(P,R)$ is the set of all connected pairs of regions $(I,J)$ such that $J$ is in $\mathcal{E}(P)$ but not $\mathcal{E}(R)$ while $I$ is *not* in $E(P)$. $D(P,R)$ is the set of all connected pairs of regions $(I,J)$ such that $J$ is in $\mathcal{E}(R)$, while $I$ is in $\mathcal{E}(P)$, but not $\mathcal{E}(R)$.

We now prove a central theorem about the parent-to-child GBP algorithm, which is defined by the message update rules (48) combined with the belief equations (44).

*Theorem:* A set of messages and beliefs define a fixed point of the parent-to-child GBP algorithm if and only if the beliefs are a stationary point of the region graph free energy, where the region graph free energy is constrained to have beliefs that are consistent and normalized.

*Proof:* To simplify the proof, we will assume that no region $R$ in the region graph has counting number $c_R = 0$. In appendix F, we discuss this technically useful assumption in detail. In particular, we show that it is easy to remove any $c_R = 0$ regions to get an equivalent region graph; and also that even if we do permit them, the parent-to-child GBP algorithm will still work properly, although the following proof no longer holds.

Recall that the region graph free energy is simply

$$F_{\mathcal{R}}(\{b_R\}) = \sum_{R\in\mathcal{R}} c_R F_R(b_R). \tag{49}$$

To derive the stationarity conditions, we need to create a Lagrangian $L$ for the free energy which enforces consistency between the beliefs in every pair of connected regions. To that end, we add Lagrange multipliers $\lambda_{PC}(\mathbf{x}_C)$ which enforce that

$$b_C(\mathbf{x}_C) = \sum_{\mathbf{x}_P\setminus\mathbf{x}_C} b_P(\mathbf{x}_P) \tag{50}$$

for every pair of parent and child regions $P$ and $C$.

Of course, we also need to include Lagrange multipliers $\gamma_R$ which enforce the normalization of the beliefs: $\sum_{\mathbf{x}_R} b_R(\mathbf{x}_R) = 1$. Setting the derivatives of $L$ with respect to the beliefs $b_R(\mathbf{x}_R)$ equal to zero gives us the following stationarity conditions:

$$c_R \ln b_R(\mathbf{x}_R) = \gamma_R + c_R \sum_{a\in A_R} \ln f_a(\mathbf{x}_a)... \\ - \sum_{P\in\mathcal{P}(R)} \lambda_{PR}(\mathbf{x}_R) + \sum_{C\in\mathcal{C}(R)} \lambda_{RC}(\mathbf{x}_C), \tag{51}$$

where $\mathcal{P}(R)$ is the set of regions that are parents of region $R$, and $\mathcal{C}(R)$ is the set of regions that are children of region $R$. In this expression, $\mathbf{x}_a$ and $\mathbf{x}_C$ are entirely determined by the value of $\mathbf{x}_R$.

Our proof will now work backwards from the belief equations that we want to derive. We want to show that there exists a "rotation" from our Lagrange multipliers $\lambda$ to another set of Lagrange multipliers $\mu$ such that the stationary point conditions can be re-written as

$$c_R \ln b_R(\mathbf{x}_R) = \gamma_R + c_R \sum_{a\in A_R} \ln f_a(\mathbf{x}_a)... \tag{52}$$

$$+ \sum_{P\in\mathcal{P}(R)} \mu_{PR}(\mathbf{x}_R) + \sum_{D\in\mathcal{D}(R)} \sum_{P'\in\mathcal{P}(D)\setminus\mathcal{E}(R)} \mu_{P'D}(\mathbf{x}_D).$$

Clearly, if we can show this, then by identifying the message $m_{P\to R}(\mathbf{x}_R) = \exp(\mu_{PR}(\mathbf{x}_R))$, we will recover our desired belief equations.

So what do the Lagrange multipliers $\mu_{PR}(\mathbf{x}_R)$ constrain? The answer is that they impose the constraint

$$c_R b_R(\mathbf{x}_R) + \sum_{A\in\mathcal{A}(R)\setminus(P\cup\mathcal{A}(P))} c_A \sum_{\mathbf{x}_A\setminus\mathbf{x}_R} b_A(\mathbf{x}_A) = 0. \tag{53}$$

In words, the Lagrange multiplier $\mu_{PR}$ constrains the weighted belief in region $R$ plus the sum of the weighted beliefs in all the ancestor regions of region $R$, *except* for regions $P$ and all its ancestors, to be equal to zero. If we make a Lagrangian using these Lagrange multipliers, it is straightforward to work out that its stationary points are given by equation (52).

Now we need to show that the new set of $\mu$ Lagrange multipliers and their associated constraints are equivalent to the old set of $\lambda$ Lagrange multipliers and their constraints. We first note that because $c_R + \sum_{A\in\mathcal{A}(R)} c_A = 1$, and $c_P + \sum_{A\in\mathcal{A}(P)} C_A = 1$, we can subtract these two equations and obtain

$$c_R + \sum_{A\in\mathcal{A}(R)\setminus(P\cup\mathcal{A}(P))} c_A = 0 \tag{54}$$

If we start with the $\lambda_{PC}(\mathbf{x}_C)$ constraints that $b_C(\mathbf{x}_C) = \sum_{\mathbf{x}_P\setminus\mathbf{x}_C} b_P(\mathbf{x}_P)$ for every pair of parent and child regions, we can use equation (54) as a basis for deriving the constraints associated with the $\mu$ Lagrange multipliers. It is also always possible to go in the other direction: The $\mu$ constraints will be linearly independent, so that if we begin with them, we can derive the $\lambda$ constraints [28]. (This is where the proof breaks down if there are regions with counting number $c_R = 0$; the $\mu$ constraints may be linearly dependent in that case.) •

Note that we have not given a general formula relating the new $\mu$ Lagrange multipliers to the $\lambda$ Lagrange multipliers, as we only need to show the existence of a rotation to a new set of Lagrange multipliers, without constructing it explicitly. It is difficult to derive a general formula relating the two sets of Lagrange multipliers, but for region graphs with only two "generations" of regions like those constructed using the junction graph method (see appendix A), we can in fact give the relationship explicitly:

$$\lambda_{PR}(\mathbf{x}_R) = \sum_{P' \in \mathcal{P}(R) \setminus P} \mu_{P'R}(\mathbf{x}_R). \tag{55}$$

## VIII. Detailed Example of a GBP Algorithm

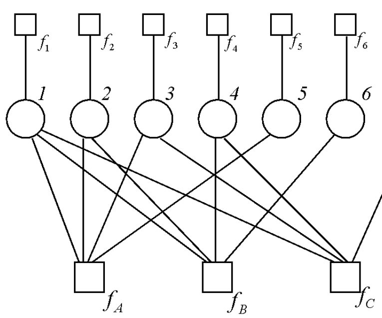


Fig. 9. A factor graph that we will use for our detailed example of how to construct a GBP algorithm.

We will now give a detailed example of how to construct a GBP algorithm. Consider the factor graph drawn in figure 9, which has seven variable nodes and ten factor nodes. For this factor graph, it is convenient to slightly alter our labeling conventions so that some of the factor nodes (the ones attached to a single variable node) are labeled with a number rather than a letter. This factor graph corresponds to the joint probability distribution

$$p(x_1, x_2, ..., x_7) = \frac{1}{Z} \left( \prod_{i=1}^{7} f_i(x_i) \right) ... \tag{56}$$
$$f_A(x_1, x_2, x_3, x_5) f_B(x_1, x_2, x_4, x_6) f_C(x_1, x_3, x_4, x_7)$$

We will work out a GBP algorithm making no assumptions about the actual forms of the functions, but we note that this particular factor graph can be used to represent the probability distribution that occurs when decoding a block error-correcting code [20]. In particular, if each of the variable nodes is binary, with possible states `0` or `1`, and the functions $f_A$, $f_B$, and $f_C$ are parity-check functions (equal to 1 if the sum of their arguments are even, and 0 otherwise), then this factor graph corresponds to the linear block $(7, 4, 3)$ Hamming code with parity check matrix

$$H = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}. \tag{57}$$

For the decoding problem, the functions $f_i(x_i)$ represent the likelihoods of the possible states of the bits, in light of the received block from the channel and the assumed channel model.

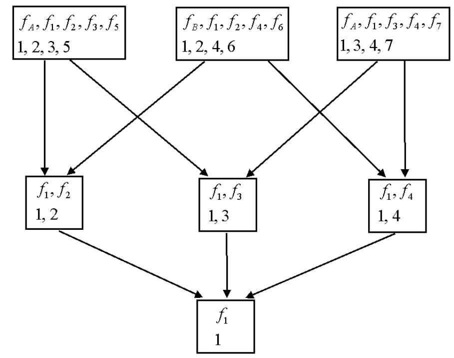


Fig. 10. A region graph obtained for the factor graph of figure 9 using the cluster variation method.

To obtain a GBP algorithm, we first need to create a region graph. We use the cluster variation method, with largest regions $\{f_A, f_1, f_2, f_3, f_5, 1, 2, 3, 5\}$, $\{f_B, f_1, f_2, f_4, f_6, 1, 2, 4, 6\}$ and $\{f_C, f_1, f_3, f_4, f_7, 1, 3, 4, 7\}$. Following the cluster variation method prescription for finding intersection regions detailed in appendix B, we obtain the region graph shown in figure 10.

Now that we have a region graph, we need to choose what kind of GBP algorithm we want to use and then write down the belief and message equations for the GBP algorithm. We choose to use the parent-to-child algorithm.

Note that although the region graph free energy is useful for theoretically *justifying* a GBP algorithm, it will not be necessary for *constructing* the algorithm. Instead, we can work directly with the belief equations.

Recall that in the parent-to-child algorithm, we only have one kind of message $m_{P \to R}(\mathbf{x}_R)$ from a parent region to a child region. Each region $R$ has a belief $b_R(\mathbf{x}_R)$ given by equation (44) which we re-write here:

$$\begin{aligned} b_R(\mathbf{x}_R) \quad &\propto \quad \prod_{a \in A_R} f_a(\mathbf{x}_a) \left( \prod_{P \in \mathcal{P}(R)} m_{P \to R}(\mathbf{x}_R) \right) ... \\ &... \quad \left( \prod_{D \in \mathcal{D}(R)} \prod_{P' \in \mathcal{P}(D) \setminus \mathcal{E}(R)} m_{P' \to D}(\mathbf{x}_D) \right) \end{aligned} \tag{58}$$

In words, this equation says that the belief at each region is a product of the local factors in that region, the messages from parents, and the messages into descendant regions from other parents who are not also descendants.

In our region graph, we have seven regions that can be grouped into three types of regions: the three regions exemplified by $\{f_A, f_1, f_2, f_3, f_5, 1, 2, 3, 5\}$ that contain five factor nodes and four variable nodes; the three regions exemplified by $\{f_1, f_2, 1, 2\}$ that contain two factor nodes and two variable nodes; and the single region $\{f_1, 1\}$ that contains one factor node and one variable node.

We will use an abbreviated notation, dropping explicit $\mathbf{x}_R$ dependence, for beliefs and messages and factor functions. The notation is best explained with some examples: we write $b_{1235}$, $b_{12}$ and $b_1$ for the beliefs at the regions listed in the previous paragraph; we write $m_{35\to 12}$ for the message from region $\{f_A, f_1, f_2, f_3, f_5, 1, 2, 3, 5\}$ to region $\{f_1, f_2, 1, 2\}$, $m_{2\to 1}$ for the message from region $\{f_1, f_2, 1, 2\}$ to region $\{f_1, 1\}$, and we abbreviate $f_A(x_1, x_2, x_3, x_5)$ as $f_A$.

In this abbreviated notation, the belief equations for the largest regions will be

$$b_{1235} \propto f_A f_1 f_2 f_3 f_5 m_{46\to 12} m_{47\to 13} m_{4\to 1}, \tag{59}$$

$$b_{1246} \propto f_B f_1 f_2 f_4 f_6 m_{35\to 12} m_{47\to 14} m_{3\to 1}, \tag{60}$$

and

$$b_{1347} \propto f_C f_1 f_3 f_4 f_7 m_{25\to 13} m_{26\to 14} m_{2\to 1}. \tag{61}$$

Note that since these regions do not have parents, all the relevant messages are into descendant regions from other parents who are not descendants.

The belief equations for the intermediate-sized regions will be

$$b_{12} \propto f_1 f_2 m_{35\to 12} m_{46\to 12} m_{3\to 1} m_{4\to 1}, \tag{62}$$

$$b_{13} \propto f_1 f_3 m_{25\to 13} m_{47\to 13} m_{2\to 1} m_{4\to 1} \tag{63}$$

and

$$b_{14} \propto f_1 f_4 m_{26\to 14} m_{37\to 14} m_{2\to 1} m_{3\to 1}. \tag{64}$$

Finally, the belief equation for the region $\{f_1, 1\}$ will be

$$b_1 \propto f_1 m_{2\to 1} m_{3\to 1} m_{4\to 1}. \tag{65}$$

The message-update rules are obtained by combining these belief equations with the marginalization conditions between parent and child regions:

$$b_C(\mathbf{x}_C) = \sum_{\mathbf{x}_P \setminus \mathbf{x}_C} b_P(\mathbf{x}_P). \tag{66}$$

For example, requiring consistency between the beliefs at the region $\{f_1, 1\}$ and the region $\{f_1, f_2, 1, 2\}$ tells us that

$$b_1(x_1) = \sum_{x_2} b_{12}(x_1, x_2) \tag{67}$$

from which we obtain

$$m_{2\to 1} := \sum_{x_2} f_2 m_{35\to 12} m_{46\to 12}. \tag{68}$$

The other message-update rules, obtained in the same way (or equivalently by using equation (48), will be

$$m_{3\to 1} := \sum_{x_3} f_3 m_{25\to 13} m_{47\to 13}, \tag{69}$$

$$m_{4\to 1} := \sum_{x_4} f_4 m_{26\to 14} m_{37\to 14}, \tag{70}$$

$$m_{3\to 1} m_{35\to 12} := \sum_{x_3, x_5} f_A f_3 f_5 m_{47\to 13}, \tag{71}$$

$$m_{2\to 1} m_{25\to 13} := \sum_{x_2, x_5} f_A f_2 f_5 m_{46\to 12}, \tag{72}$$

$$m_{4\to 1} m_{46\to 12} := \sum_{x_4, x_6} f_B f_4 f_6 m_{37\to 14}, \tag{73}$$

$$m_{2\to 1} m_{26\to 14} := \sum_{x_2, x_6} f_B f_2 f_6 m_{35\to 12}, \tag{74}$$

$$m_{4\to 1} m_{47\to 13} := \sum_{x_4, x_7} f_C f_4 f_7 m_{26\to 14}, \tag{75}$$

and

$$m_{3\to 1} m_{37\to 14} := \sum_{x_3, x_7} f_C f_3 f_7 m_{25\to 13}. \tag{76}$$

In practice, it often helps convergence to only step the messages part-way to their newly computed values. This simple heuristic can eliminate "over-shooting" problems.

We note here one potential practical pitfall to avoid when using inertia. Let us suppose that we have a set of old messages $\{m^{\text{old}}\}$, which we use in the update equations to calculate a set of messages $\{m^{\text{update}}\}$, and that we want to set our new messages to be half-way between the old messages and the updated messages: $\{m^{\text{new}}\} = \frac{1}{2}\{m^{\text{old}}\} + \frac{1}{2}\{m^{\text{update}}\}$. We recommend when using an update equation with more than one message on the left hand side, that all those messages are $m^{\text{update}}$ equations. Mixing in $m^{\text{new}}$ or $m^{\text{old}}$ messages on the left hand side empirically often results in poor convergence properties. For example, the update equation (71) given above should explicitly be

$$m^{\text{update}}_{3\to 1} m^{\text{update}}_{35\to 12} := \sum_{x_3, x_5} f_A f_3 f_5 m^{\text{old}}_{47\to 13}. \tag{77}$$

Fortunately, it is always possible to schedule the message updates so that one computes the updated messages into the smallest regions first (e.g. messages like $m^{\text{update}}_{3\to 1}$), so that they are available when needed to compute the updated messages into larger regions.

There are many other details that can be handled in different ways in iterating the message update equations. For example, the messages can be initialized in any way one likes; two popular choices are random or uniform messages. The algorithm typically terminates after a fixed number of iterations, or after some convergence criterion is satisfied, but other termination conditions are possible. In a decoding application, one typically checks at each iteration whether the thresholded beliefs correspond to a code-word, and terminates the decoding algorithm if they do, stopping otherwise when some fixed number of iterations has passed.

## IX. Discussion

In this paper, we have presented a general theory, based on region graphs, for constructing generalized belief propagation (GBP) algorithms. Region graphs permit easy visualization of the structure of GBP algorithms–messages are always sent between the neighboring regions on the graph. For region graphs that have no cycles, the GBP algorithms are exact. We have also seen that the fixed points of the GBP algorithm always correspond to the stationary points of an approximate region-based free energy, so that even when the region graph has cycles, GBP algorithms seem to be do something reasonable. The standard BP algorithm turned out to be a special case of a GBP algorithm obtained when the region graph is constructed using the Bethe method.

Given a factor graph and limited computational resources, a key remaining problem is how to choose an "optimal" region graph–i.e. one that gives the most accurate results with the least computational effort. We limit ourselves here to suggesting two sensible heuristics. First, it is wise to try to collect the shortest cycles in a factor graph into regions, so that they are handled as accurately as possible. Second, in order that the region graph free energy be as accurate as possible, one should try to make the region graph resemble a tree–that is, one should avoid short cycles in the region graph.

We have previously reported promising numerical results obtained using GBP algorithms for inference on random Markov Random Fields [17] and for decoding error-correcting codes [39].


## ACKNOWLEDGEMENTS

We thank Dave Forney and Robert McEliece for helpful and encouraging discussions, and David MacKay for his comments on a draft of this paper.


## APPENDIX A: THE JUNCTION GRAPH METHOD

A natural idea to generalize the Bethe Method is to keep the notion that $\mathcal{R}$ should be the union of a set of large regions $\mathcal{R}_L$ and a set of small regions $\mathcal{R}_S$, but to let the regions in $\mathcal{R}_L$ or $\mathcal{R}_S$ contain more nodes. The *junction graph method*, that we describe here, exploits this idea, and is based on a generalization of the "junction graphs" that were introduced by Aji and McEliece [27].

We define a *junction graph* to be a labeled bipartite graph $\mathcal{G} = (V_L, V_S, E, L)$ in which there are *large vertices* (corresponding to large regions) $v_l \in V_L$, *small vertices* (corresponding to small regions) $v_s \in V_S$, and directed edges (or *arcs*) $e \in E$ connecting large vertices to small vertices. The vertices in the junction graph are labeled, and the label of vertex $v_i$ is denoted $L(v_i)$. The labels will be subsets of a set of indices $I$ representing factor or variable nodes of a factor graph.

For the graph $\mathcal{G}$ to be considered a junction graph, we insist upon two conditions. First, if $v_s$ is a small vertex neighboring the $k$ large vertices $v_{l_1}, v_{l_2}, ..., v_{l_k}$, then we require that $L(v_s)$ is a subset of each of $L(v_{l_1}), L(v_{l_2}), ...., L(v_{l_k})$, or equivalently, that

$$L(v_s) \subseteq L(v_{l_1}) \cap L(v_{l_2}) \cap ... \cap L(v_{l_k}). \tag{A-1}$$

Secondly, we require that for any index $i \in I$, the subgraph of $\mathcal{G}$ consisting only of the vertices which contain $i$ in their labels, is a connected tree.

The "junction graphs" introduced by Aji and McEliece [27] are a special case of those described here. In their junction graphs, small vertices were restricted to have precisely two neighboring large vertices, so that the small vertices can be interpreted as labeled "edges" between the large vertices. They further required that small region labels not include any indices representing factor nodes.

Given a set of regions $\mathcal{R}_{JG} = \mathcal{R}_L \cup \mathcal{R}_S$ that are organized into a junction graph, we can always obtain a valid region-based approximation by defining a set of counting numbers $c_R$ as follows. For all regions $R \in \mathcal{R}_L$, we let $c_R = 1$, while for all region $R \in \mathcal{R}_S$, we let $c_R = 1 - d_R$ where $d_R$ is the degree (numbering of neighboring large regions) of region $R$. It is through this prescription that the arcs the junction graph become relevant–a small region's contribution to the free energy is subtracted out from that of a large region only if the two regions are connected by an arc. It is straightforward to confirm that this prescription for the counting numbers gives us a valid region-based free energy approximation, as the junction graph condition that the sub-graph for each variable or factor node is a tree guarantees that each variable and factor node will be counted once as required in equation (26).

Aji and McEliece proved a theorem that tells us that given *any* set of large regions $\mathcal{R}_L$ that contain all the factor and variable nodes in a factor graph, we can find a corresponding set of small regions $\mathcal{R}_S$ and organize the regions in $\mathcal{R}_{JG} = \mathcal{R}_L \cup \mathcal{R}_S$ into a junction graph. Their theorem generalizes without difficulty to our version of junction graphs.

As an example, consider the factor graph which we introduced in the main text and re-draw in figure 11. We could take as our set of large regions $\mathcal{R}_L$ the four regions $\{A, C, 1, 2, 4, 5\}$, $\{B, D, 2, 3, 5, 6\}$, $\{C, E, 4, 5, 7, 8\}$, and $\{F, 5, 6, 8, 9\}$. An acceptable set of corresponding small regions $\mathcal{R}_S$ would be $\{2, 5\}$, $\{C, 4, 5\}$, $\{5, 6\}$, and $\{8\}$, with a junction graph as shown in figure 11. Because in this case each of the small regions is connected to two large regions, they would each have an counting number of $-1$.

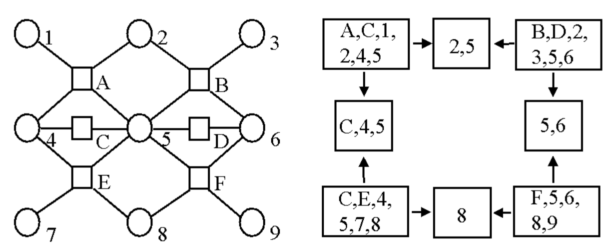


Fig. 11. A junction graph (on the right) for the factor graph on the left.

The set of regions given by the Bethe method can also always be organized into a junction graph (though not necessarily the restricted Aji-McEliece version of a junction graph); using as an example the same factor graph, the resulting junction graph is shown in figure 12. It is obvious from this example that there

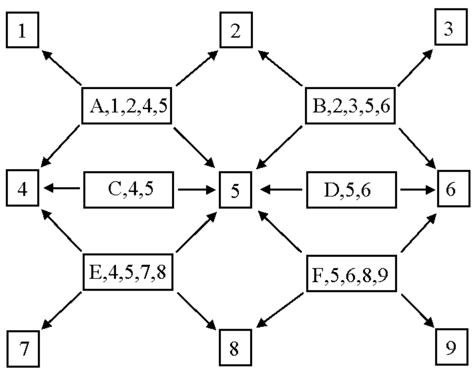


Fig. 12. A junction graph for the factor graph shown in figure 3 generated using the Bethe method. Note the isomorphism between this junction graph and the original factor graph.

will always be a one-to-one isomorphism between the original factor graph and the corresponding junction graph obtained from the Bethe method.

The junction graph approximation for the Gibbs free energy is

$$F_{JG}(\{b_R\}) = U_{JG}(\{b_R\}) - H_{JG}(\{b_R\}), \qquad \text{(A-2)}$$

where

$$U_{JG}(\{b_R\}) = \sum_{R\in\mathcal{R}_L} U_R(b_R) + \sum_{R\in\mathcal{R}_S} (1-d_R)U_R(b_R), \quad \text{(A-3)}$$

and

$$H_{JG}(\{b_R\}) = \sum_{R\in\mathcal{R}_L} H_R(b_R) + \sum_{R\in\mathcal{R}_S} (1-d_R)H_R(b_R). \qquad \text{(A-4)}$$

Junction graphs are a special case of region graphs, where there are only two “generations” of regions. It follows that minimizing the junction graph free energy $F_{JG}$ will once again give beliefs $\{b_R\}$ that are equivalent to those obtained from a message-passing BP algorithm. That algorithm is sometimes known as the *generalized distributive law* [24]. Again it follows as a corrollary of our more general results for region graphs that the junction graph approximation to the Gibbs free energy will be exact, and the generalized distributive law will give exact results, when the junction graph is a tree. In that case, we can call the junction graph a *junction tree*, and the generalized distributive law reduces to the famous *junction tree algorithm*.

Our junction trees are actually a slight generalization of what is normally called a “junction tree,” in that we allow *separators* (i.e., the small regions) to neighbor more than just two large regions. We can generalize the well-known result [13] for the joint probability function in junction trees to our case and obtain

$$p(\mathbf{x}) = \frac{\prod_{R\in\mathcal{R}_L} p_R(\mathbf{x}_R)}{\prod_{R\in\mathcal{R}_S} p_R(\mathbf{x}_R)^{d_R-1}}. \qquad \text{(A-5)}$$

To obtain this result, we note that while we have described region graphs and junction graphs as directed graphs, from the point of view of statistical grphical models, they are equivalent to undirected graphs. In particular, one can re-write the full joint probability distribution $p(\mathbf{x})$ for a factor graph in the form

$$p(\mathbf{x}) = \frac{1}{Z}\prod_{(RS)} \Psi_{RS}(\mathbf{x}_R, \mathbf{x}_S) \prod_R \Phi_R(\mathbf{x}_R) \qquad \text{(A-6)}$$

where $(RS)$ denotes pairs of connected regions in a given region graph for that factor graph. Specifically, when we set $\Phi_R(\mathbf{x}_R) = \left(\prod_{a\in A_R} f_a(\mathbf{x}_a)\right)^{c_R}$ and $\Psi_{RS}(\mathbf{x}_R, \mathbf{x}_S)$ equal to 1 if $\mathbf{x}_R$ is consistent with $\mathbf{x}_S$ and equal to 0 otherwise, this form of the joint probability distribution will be equivalent to the one in the original factor graph form. Since the formula (A-5) is true for pairwise Markov Random Fields when the set of nodes in $\mathcal{R}_L$ are separated by the set of nodes in $\mathcal{R}_S$, and we have shown how to convert a region graph into an equivalent pairwise Markov Random Field, we have justified using formula (A-5) for region graphs as well.

## Appendix B: The Cluster Variation Method

Another method for selecting a valid set of regions $\mathcal{R}$ and counting numbers $c_R$ is the *cluster variation method* introduced by Kikuchi in 1951 and further developed in the physics literature since then [19]. The main feature distinguishing this method from the junction graph method is that $\mathcal{R}$ may be the union of more than just two generations of regions.

In the cluster variation method, we begin with a set of distinct large regions $\mathcal{R}_0$ such that every factor node $a$ and every variable node $i$ in our factor graph is included in at least one region $R \in \mathcal{R}_0$. We also require that no region $R \in \mathcal{R}_0$ be a subregion of any other region in $\mathcal{R}_0$. We then construct the set of regions $\mathcal{R}_1$ by forming all possible intersections between regions in $\mathcal{R}_0$, but discarding from $\mathcal{R}_1$ any intersection regions that are sub-regions of other intersection regions. If possible, we then construct in the same way the set of regions $\mathcal{R}_2$ from the intersections between regions in $\mathcal{R}_1$. As long as there continue to be intersection regions, we construct sets of regions $\mathcal{R}_3, \mathcal{R}_4, ...\mathcal{R}_K$ in the same way. Finally, the set of regions used in the cluster variation method will be $\mathcal{R} = \mathcal{R}_0 \cup \mathcal{R}_1 \cup ... \cup \mathcal{R}_K$.

We define the counting numbers in the cluster variation method to be

$$c_R = 1 - \sum_{S\in\mathcal{S}(R)} c_S \qquad \text{(B-1)}$$

where $\mathcal{S}(R)$ is the set of all regions which are super-regions of region $R$.

Returning to our example factor graph drawn in figure 3, we can choose the base set of regions $\mathcal{R}_0$ to consist of the four regions $\{A, C, 1, 2, 4, 5\}$, $\{B, D, 2, 3, 5, 6\}$, $\{C, E, 4, 5, 7, 8\}$, and $\{D, F, 5, 6, 8, 9\}$. Once the set of base regions $\mathcal{R}_0$ is chosen, there is no further choice in the cluster variation method. In our case, the set of intersection regions $\mathcal{R}_1$ would be the regions $\{2, 5\}$ $\{C, 4, 5\}$, $\{D, 5, 6\}$, $\{5, 8\}$, and the set of intersection regions $\mathcal{R}_2$ would be $\{5\}$.

Each of the regions $R \in \mathcal{R}_0$ would have an counting number $c_R = 1$. Because each of the regions $R \in \mathcal{R}_1$ is the subregion of two regions in $\mathcal{R}_0$, they each have an counting number of $c_R = 1 - 2 = -1$. Finally since every region in $\mathcal{R}_0$ and $\mathcal{R}_1$ is a super-region of $\{5\}$, its counting number is $1 - 4 + 4 = 1$.

We can represent this set of regions and counting numbers with the region graph shown in figure 13.

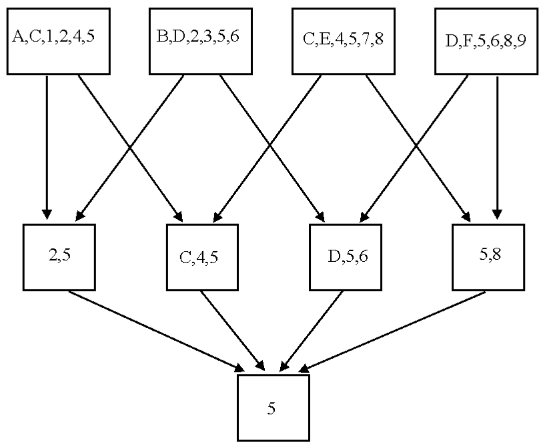


Fig. 13. A region graph generated using the cluster variation method.

Note that the Bethe approximation will be a special case of the cluster variation method if and only if no factor node shares more than one variable node with another factor node (or equivalently, there are no cycles of length four in the factor graph.) The factor graph shown in figure 13 is therefore one example of a factor graph for which the Bethe approximation can *not* be generated by the cluster variation method.

We remark that in the physics literature, the cluster variation method has normally been applied to a restricted class of factor graphs that are particularly relevant as models of magnetic materials. In particular, the factor graph normally represents a translationally invariant crystal lattice, and the factor nodes normally have degree two, corresponding to two-body interactions. Translational symmetry in the factor graph often dramatically simplifies the problem of minimizing the Kikuchi free energy, and when the factor nodes have degree two, the Bethe method *will* always be a special case of the cluster variation method.

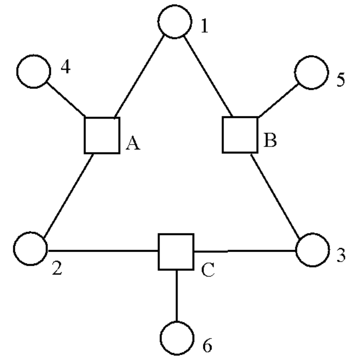


Fig. 14. For this factor graph, the choice of regions $\{A,1,2,4\}$, $\{B,1,3,5\}$, $\{C,2,3,6\}$, and $\{1,2,3\}$, with corresponding counting numbers of 1,1,1,and $-1$, will give a valid region-based approximation that cannot be represented by a region graph.

Note, however, that although the region graph method is the most general method we have introduced, there do exist valid region-based free energy approximations that do not have a region graph representation. We demonstrate an example in figure 14.

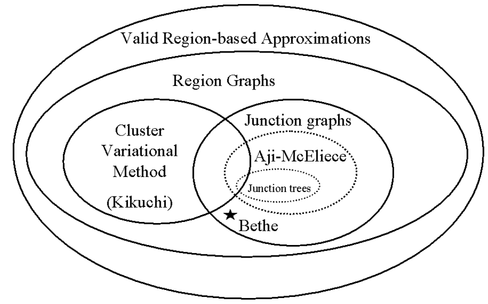


Fig. 15. A Venn diagram illustrating the relationships between different methods of generating valid region-based free energy approximations. The Bethe method is always an exemplar of the junction graph method, but is only a special case of the cluster variation method if the factor graph has no pair of factor nodes that share more than one variable node, and is only a special case of Aji and McEliece's junction graph method if the relevant factor graph is a Forney "normal" graph (no variable node is connected to more than two factor nodes).

## Appendix C: Relationships Between Different Methods

In this appendix, we summarize the relationships between the different methods for generating valid sets of regions for a region-based free energy approximation. First of all, as is clear from its definition, a junction graph will always be a region graph (though the converse is not true). The sets of regions and counting numbers generated by the cluster variation method can also always be represented by a region graph. We already saw one example in figure 13.

We emphasize that one can construct region graph approximations that cannot be generated with either the junction graph or cluster variation methods. We already saw such an example when we introduced region graphs in the main text in section VI. Constructions that are more general than those constructed using the cluster variation method or the junction graph method may be useful for a variety of reasons, including reducing the computational complexity of a GBP algorithm.

In summary, we have the following relationships, as illustrated in the Venn diagram of figure 15. For a given factor graph, the cluster variation method and the generalized junction graph method each generate valid region-based free energy approximations that are subclasses of all the possible valid approximations. Neither the cluster variation method nor the generalized junction graph method is more general than the other, and both are subsumed by the more general region graph method. The set of regions generated by the Bethe method is always an examplar of those generated by the junction graph method, and will be an examplar of those generated by the cluster variation method if and only if the factor graph has no cycles of length four. In general, the Bethe method will not be a special case of the Aji-McEliece junction graph method, though it will be for factor graphs such that each variable node is adjacent to no more than two factor nodes (Forney's so-called "normal" factor graphs [21]).

In addition to being a more general method than the cluster variation method or the junction graph method, we feel that the region graph method is easier to understand on an intuitive level. We simply select a set of regions and counting numbers such that every variable and factor node gets counted once, and such that we can enforce consistency for the belief over any variable node, no matter which region we choose. Region graphs also have the important advantage of being a natural graphical structure for describing generalized belief propagation algorithms.

Pakzad and Anantharam have suggested strengthening the region graph requirements described in section VI so that for every *sub-set* of variable nodes in the factor graph, the sub-graph of regions containing that sub-set must be connected and must have a sum of counting numbers equal to one [29]. Such a strengthening would ensure that the beliefs computed for any sub-set of nodes would be consistent, no matter which regions were used to compute it. The cluster variation method produces region graphs that satisfy these stronger requirements, but we chose not to insist on these stronger requirements in general, because region graphs created using the Bethe Method will not necessarily satisfy them.

## Appendix D: The Child-to-Parent Algorithm

The observation underlying the "child-to-parent algorithm" is that when we minimize the Bethe free energy, the Lagrange multipliers enforcing the marginalization constraints correspond exactly (after exponentiation) to the $n_{i\to a}(x_i)$ messages from variable nodes to factor nodes in the BP algorithm. Considering these messages as messages from child regions to parent regions in a region graph, we can try to generalize the approach to arbitrary region graphs. Thus, we construct a GBP algorithm by simply minimizing a region graph free energy and identifying Lagrange multipliers that enforce consistency between beliefs with messages from child regions to parent regions. Such an approach was considered in detail by Kappen and Wiegerinck for region graphs constructed using the cluster variation method [37].

We begin with the stationary point equations obtained from differentiating a Lagrangian $L$ that represents a region graph free energy $F_{\mathcal{R}}(\{b_R\})$ with beliefs $\{b_R\}$ that are constrained to be consistent with their neighbors on the region graph. We obtained this equation previously (see equation (51)), and re-write it here:

$$c_R \ln b_R(\mathbf{x}_R) = \gamma_R + c_R \sum_{a\in A_R} \ln f_a(\mathbf{x}_a)... \\ - \sum_{P\in\mathcal{P}(R)} \lambda_{PR}(\mathbf{x}_R) + \sum_{C\in\mathcal{C}(R)} \lambda_{RC}(\mathbf{x}_C), \quad \text{(D-1)}$$

where $\mathcal{P}(R)$ is the set of regions that are parents of region $R$, and $\mathcal{C}(R)$ is the set of regions that are children of region $R$, and $\lambda_{PR}(\mathbf{x}_R)$ are the Lagrange multipliers that enforce consistency between the beliefs in region $P$ and those in region $R$.

For $c_R \neq 0$, we can re-write this equation as

$$b_R(\mathbf{x}_R) \propto \prod_{a\in A_R} f_a(\mathbf{x}_a) \left( \frac{\prod_{C\in\mathcal{C}(R)} n_{C\to R}(\mathbf{x}_C)}{\prod_{P\in\mathcal{P}(R)} n_{R\to P}(\mathbf{x}_R)} \right)^{1/c_R}, \quad \text{(D-2)}$$

where $n_{C\to P}(\mathbf{x}_C) = \exp(\lambda_{PC}(\mathbf{x}_C))$ is a "message" from a child region $C$ to a parent region $P$, in analogy with the messages $n_{i\to a}(x_i)$ in standard BP. If $c_R = 0$, we do not get a condition on $b_R(\mathbf{x}_R)$ ($b_R(\mathbf{x}_R)$ can still be determined from beliefs in super-regions via the marginalization conditions); instead we obtain the following condition on the messages into and out of region $R$:

$$\left( \frac{\prod_{C\in\mathcal{C}(R)} n_{C\to R}(\mathbf{x}_C)}{\prod_{P\in\mathcal{P}(R)} n_{R\to P}(\mathbf{x}_R)} \right) = 1. \quad \text{(D-3)}$$

The message update rules are then obtained by applying the marginalization conditions $b_C(\mathbf{x}_C) = \sum_{\mathbf{x}_P\setminus\mathbf{x}_C} b_P(\mathbf{x}_P)$.

A small example might help clarify the meaning of these equations for the reader. Consider the probability distribution

$$p(x_1, x_2, x_3) = \frac{1}{Z} f_A(x_1, x_2) f_B(x_2, x_3). \quad \text{(D-4)}$$

We use the Bethe approximation, which should be exact in this case because the factor graph is a tree. Thus, we obtain large regions $\{A, 1, 2\}$ and $\{B, 2, 3\}$, with counting numbers 1, and small regions $\{1\}$, $\{2\}$, and $\{3\}$, with counting numbers 0, 1, and 0 respectively. We obtain the following belief equations for the regions with $c_R \neq 0$:

$$b_A(x_1, x_2) \propto f_A(x_1, x_2) n_{1\to A}(x_1) n_{2\to A}(x_2), \quad \text{(D-5)}$$

$$b_B(x_2, x_3) \propto f_B(x_2, x_3) n_{2\to B}(x_2) n_{3\to B}(x_3), \quad \text{(D-6)}$$

$$b_2(x_2) \propto n_{2\to A}(x_2) n_{2\to B}(x_2), \quad \text{(D-7)}$$

and the following conditions on messages for the regions with $c_R = 0$:

$$n_{1\to A}(x_1) = 1, \quad \text{(D-8)}$$

and

$$n_{3\to B}(x_3) = 1. \quad \text{(D-9)}$$

Using these conditions and the marginalization conditions, we find that

$$n_{2\to A}(x_2) = \sum_{x_3} f_B(x_2, x_3), \quad \text{(D-10)}$$

and

$$n_{2\to B}(x_2) = \sum_{x_1} f_A(x_1, x_2). \quad \text{(D-11)}$$

We can now easily check that in this example, the computed beliefs give back the desired marginal probabilities exactly.

The child-to-parent algorithm, by its construction, clearly gives a generalized BP algorithm whose fixed points correspond to the stationary points of the region graph free energy. On the other hand, it might be considered inelegant both because it focuses only on the messages from child regions to parent regions and because the message update equations will inevitably be complicated and involve the counting numbers $c_R$. The *two-way* algorithm described in Appendix E and the *parent-to-child* described in the main text in section VII-A are different GBP algorithms that attempt to ameliorate these flaws.

## APPENDIX E: THE TWO-WAY ALGORITHM

To motivate the two-way algorithm, we return to the standard BP algorithm, where we recall that the belief equations can be written in the form

$$b_i(x_i) = \prod_{a \in N(i)} m_{a\to i}(x_i) \tag{E-1}$$

and

$$b_a(\mathbf{x}_a) = f_a(\mathbf{x}_a) \prod_{i \in N(a)} n_{i\to a}(x_i) \tag{E-2}$$

where

$$n_{i\to a}(x_i) = \prod_{b \in N(i)\backslash a} m_{b\to i}(x_i). \tag{E-3}$$

Given these equations, it is natural to aim for a generalization where the belief equations will have the form

$$b_R(\mathbf{x}_R) = \tilde{f}_R(\mathbf{x}_R) \prod_{C \in \mathcal{C}(R)} n_{C\to R}(\mathbf{x}_C) \prod_{P \in \mathcal{P}(R)} m_{P\to R}(\mathbf{x}_P). \tag{E-4}$$

In other words, we aim to write the belief equations so that the belief in a region is a product of local factors and messages arriving from all the connected regions, whether they are parents or children. It will turn out that we can do this, but in order that the GBP algorithm be correspond to the region graph free energy, we will need to use modified factors and a rather complicated relation between the $n_{C\to P}(\mathbf{x}_C)$ messages and $m_{P\to C}(\mathbf{x}_P)$ messages generalizing the relation for standard BP given in equation (E-3).

It will be convenient to denote the number of parents of region $R$ by $p_R$, and define the numbers $q_R \equiv (1-c_r)/p_r$ and $\beta_R \equiv 1/(2-q_r)$. When a region has no parent so that $p_R = 0$ and $c_R = 1$, we take $q_R = \beta_R = 1$. Note that within the Bethe approximation, $q_R = \beta_R = 1$ for all regions. We will assume that $q_R \neq 2$ so that $\beta_R$ is well-defined (normally, if one has a region graph with a region such that $q_R = 2$, one should be able to change the connectivity of $R$ to avoid this problem).

We first define the set of pseudo-messages for all regions $R$ and their parents $P$ and children $C$:

$$n^0_{R\to P}(\mathbf{x}_R) = \tilde{f}_R(\mathbf{x}_R) \prod_{P' \in \mathcal{P}(R)\backslash P} m_{P'\to R}(\mathbf{x}_R) \prod_{C \in \mathcal{C}(R)} n_{C\to R}(\mathbf{x}_C) \tag{E-5}$$

and

$$m^0_{R\to C}(\mathbf{x}_C) = \sum_{\mathbf{x}_R\backslash \mathbf{x}_C} \tilde{f}_R(\mathbf{x}_R) \prod_{P \in \mathcal{P}(R)} m_{P\to R}(\mathbf{x}_R) \prod_{C' \in \mathcal{C}(R)\backslash C} n_{C'\to R}(\mathbf{x}_{C'}), \tag{E-6}$$

where $\tilde{f}_R(\mathbf{x}_R) \equiv \left(\prod_{a\in A_r} f_a(\mathbf{x}_a)\right)^{c_R}$.

Aside from the fact that we raised the product of the local factors to a power of $c_R$, these pseudo-messages are what one would naively expect the message updates to look like. To obtain the true message updates, however, one needs to combine the pseudo-messages going in the two directions of a link as follows:

$$n_{R\to P}(\mathbf{x}_R) = \left(n^0_{R\to P}(\mathbf{x}_R)\right)^{\beta_R} \left(m^0_{P\to R}(\mathbf{x}_R)\right)^{\beta_R - 1} \tag{E-7}$$

and

$$m_{P\to R}(\mathbf{x}_R) = \left(n^0_{R\to P}(\mathbf{x}_R)\right)^{\beta_R - 1} \left(m^0_{P\to R}(\mathbf{x}_R)\right)^{\beta_R} \tag{E-8}$$

Note that when $\beta_R = 1$, the messages are precisely the same as the pseudo-messages.

The two-way algorithm is completed by the belief equations, which have the form already given in equation (E-4). We now claim that the above sets of messages and beliefs are fixed points of two-way GBP if and only if they are stationary points of the region graph free energy.

*Proof:* We form a Lagrangian from the region graph energy as already indicated in the previous section on the child-to-parent algorithm. If we exponentiate equation (51) derived there, we obtain the equation

$$b_R(\mathbf{x}_R)^{c_R} \propto \tilde{f}_R(\mathbf{x}_R) \prod_{C \in \mathcal{C}(R)} e^{\lambda_{RC}(\mathbf{x}_C)} \left(\prod_{P \in \mathcal{P}(R)} e^{\lambda_{PR}(\mathbf{x}_R)}\right)^{-1}. \tag{E-9}$$

Suppose that we are given a set of $\lambda$ and $b_R$ that satisfy these stationary conditions of the Lagrangian. Now we define

$$n_{R\to P}(\mathbf{x}_R) = e^{\lambda_{PR}(\mathbf{x}_R)} \tag{E-10}$$

and

$$m_{P\to R}(\mathbf{x}_R) = b_R(\mathbf{x}_R)^{q_R} e^{-\lambda_{PR}(\mathbf{x}_R)} \tag{E-11}$$

Of course, we have one $m$ message and one $n$ message for every Lagrange multiplier $\lambda$, so for these definitions to hold, we also need to have constraints relating the $m$'s and $n$'s. The constraints will be given by the definitions of the pseudo-messages and the relations between the messages and the pseudo-messages that we defined above. We want to show that these relations, as well as the two-way GBP belief equations previously defined, must hold.

First, we show that the belief equations (E-4) hold. We have

$$\begin{aligned}
b_R(\mathbf{x}_R)^{c_R} &\propto \tilde{f}_R(\mathbf{x}_R) \prod_{C \in \mathcal{C}(R)} e^{\lambda_{RC}(\mathbf{x}_C)} \prod_{P \in \mathcal{P}(R)} e^{-\lambda_{PR}(\mathbf{x}_R)} \\
&\propto \tilde{f}_R(\mathbf{x}_R) \prod_{C \in \mathcal{C}(R)} n_{C\to R}(\mathbf{x}_C) \prod_{P \in \mathcal{P}(R)} \left(\frac{f_R(\mathbf{x}_R)}{b_R(\mathbf{x}_R)}\right)^{q_R} m_{P\to R}(\mathbf{x}_R) \\
&\propto (b_R(\mathbf{x}_R))^{-q_R p_R} \tilde{f}_R(\mathbf{x}_R) \prod_{C \in \mathcal{C}(R)} n_{C\to R}(\mathbf{x}_C) \prod_{P \in \mathcal{P}(R)} m_{P\to R}(\mathbf{x}_R) \\
&\propto (b_R(\mathbf{x}_R))^{c_R - 1} \tilde{f}_R(\mathbf{x}_R) \prod_{C \in \mathcal{C}(R)} n_{C\to R}(\mathbf{x}_C) \prod_{P \in \mathcal{P}(R)} m_{P\to R}(\mathbf{x}_R)
\end{aligned}$$

so that indeed $b_R(\mathbf{x}_R)$ is product of local potentials and incoming messages.

Turning to the constraints, we have from the definition of $n^0_{R\to P}(\mathbf{x}_R)$, that

$$\begin{aligned}
n^0_{R\to P}(\mathbf{x}_R)\, m_{P\to R}(\mathbf{x}_R) &= b_R(\mathbf{x}_R) && \text{(E-12)}\\
&= \sum_{\mathbf{x}_P\backslash \mathbf{x}_R} b_P(\mathbf{x}_P) && \text{(E-13)}\\
&= n_{R\to P}(\mathbf{x}_R) m^0_{P\to R}(\mathbf{x}_R). && \text{(E-14)}
\end{aligned}$$

Equations (E-10) and (E-11) imply that

$$n_{R\to P}(\mathbf{x}_R)m_{P\to R}(\mathbf{x}_R) = b_R(\mathbf{x}_R)^{q_R} \tag{E-15}$$

$$= \left(n^0_{R\to P}(\mathbf{x}_R)m_{P\to R}(\mathbf{x}_R)\right)^{q_R}. \tag{E-16}$$

Together these equations give us two equations for the two unknowns $m_{P\to R}(\mathbf{x}_R)$ and $n_{R\to P}(\mathbf{x}_R)$:

$$\frac{m_{P\to R}(\mathbf{x}_R)}{n_{R\to P}(\mathbf{x}_R)} = \frac{m^0_{P\to R}(\mathbf{x}_R)}{n^0_{R\to P}(\mathbf{x}_R)} f_R(\mathbf{x}_R)^{-q_R} \tag{E-17}$$

and

$$n_{R\to P}(\mathbf{x}_R)m_{P\to R}(\mathbf{x}_R)^{1-q_R} = \left(n^0_{R\to P}(\mathbf{x}_R)\right)^{q_R} \tag{E-18}$$

The unique solution of these equations is given by equations (E-7) and (E-8). Thus, we have shown that the message passing algorithm previously defined has fixed points that are equivalent to the stationary points of the region graph free energy.

The two-way algorithm will be particularly elegant when $\tilde{f}_R(\mathbf{x}_R) = f_R(\mathbf{x}_R)$ and when $\beta_R = 1$ for all regions. In that case, each region will send messages to all adjacent regions, and the message update rules will be the natural generalization of the ordinary BP rules written with two kinds of messages. It is interesting to note that the condition that $\tilde{f}_R(\mathbf{x}_R) = f_R(\mathbf{x}_R)$ can be ensured by requiring that only regions with no parents contain factor nodes, while the condition that $\beta_R = 1$ for all regions can be ensured by requiring that the sub-graph obtained by taking any region and all of its ancestor regions must always form a tree.

When $\beta_R = 1$ for all regions, the two-way GBP algorithm is equivalent to Pearl's method of clustering [9]: we form new nodes from clusters of variables in the original graph (these are the regions) and run an ordinary BP algorithm on the resulting graph. It is important to bear in mind that this equivalence only holds for a subset of possible region graphs: if one uses this method on a set of regions that does not satisfy the region graph conditions, or on a region graph for which $\beta_r \neq 1$ for some regions, the resulting beliefs will generally be a poor approximation.

## Appendix F: Region Graphs with $c_R = 0$ Regions

In our proof that the fixed points of the parent-to-child GBP algorithm are equivalent to the stationary points of the region graph free energy (given in section VII-A), we assumed that no region has counting number $c_R = 0$. That is never difficult to arrange: if one has a region graph with regions whose counting number equals zero, one can remove them, and then connect directly any regions that were previously ancestors or descendants of each other, but are no longer after the removal of the $c_R = 0$ regions. The remaining regions will have identical counting numbers by construction, and since the regions with $c_R = 0$ did not contribute to the region graph free energy in any case, it will be unchanged. In figure 16, we illustrate the "surgery" that needs to be performed on a region graph to remove regions with counting number zero.

In fact, however, the parent-to-child algorithm is well-defined even when some of the regions have counting numbers equal to zero, and when one implements it, one finds that the results at its fixed points are identical to those obtained when one surgically removes the $c_R = 0$ regions. The reason that the algorithm still gives proper results, even though the above proof breaks down, is that the $\lambda$ constraints that cannot be derived from the $\mu$ constraints are actually not necessary–they all involve $c_R = 0$ regions that do not contribute to the free energy in any case.

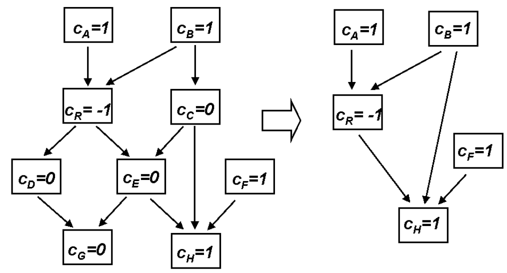


Fig. 16. An illustration of how one can take a region graph with some regions that have counting number zero, and obtain another region graph with no such regions but with an identical free energy. One first removes regions with a counting number of zero, and then directly connects any ancestor-descendant pairs that have become disconnected. In this example, we form new direct connections between regions $R$ and $H$ and between regions $B$ and $H$.

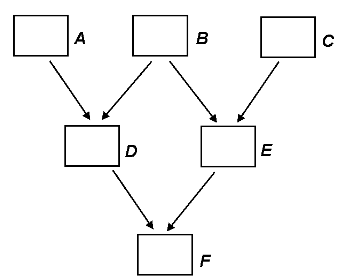


Fig. 17. A small illustrative region graph (see text). Note that region $F$ has counting number $c_F = 0$.

A small example may make this point more comprehensible. Consider the small region graph shown in figure (17). The counting numbers of the regions are $c_A = c_B = c_C = 1$, $c_D = c_E = -1$, and $c_F = 0$, so that region $F$ could clearly be removed to obtain an equivalent region graph. For the purpose of illustration, we leave it in. We have six $\lambda$ constraints, each of which is very straightforward. For example, the constraint associated with $\lambda_{AD}(\mathbf{x}_D)$ is $b_D(\mathbf{x}_D) = \sum_{\mathbf{x}_A \setminus \mathbf{x_D}} b_A(\mathbf{x}_A)$, while the constraint associated with $\lambda_{DF}(\mathbf{x}_F)$ is $b_F(\mathbf{x}_F) = \sum_{\mathbf{x}_D \setminus \mathbf{x_F}} b_D(\mathbf{x}_D$.

The six $\mu$ constraints are somewhat less straightforward. Going back to the prescription given in equation (53), we see for example that the constraint associated with $\mu_{AD}(\mathbf{x}_D)$ is

$$c_D b_D(\mathbf{x}_D) + c_B \sum_{\mathbf{x}_B \setminus \mathbf{x}_D} b_B(\mathbf{x}_B) = 0 \tag{F-1}$$

or equivalently,

$$b_D(\mathbf{x}_D) = \sum_{\mathbf{x}_B \setminus \mathbf{x}_D} b_B(\mathbf{x}_B) \tag{F-2}$$

while the constraint associated with $\mu_{DF}(\mathbf{x}_F)$ is

$$c_F b_F(\mathbf{x}_F) + c_E \sum_{\mathbf{x}_E \setminus \mathbf{x}_F} b_E(\mathbf{x}_E) + c_C \sum_{\mathbf{x}_C \setminus \mathbf{x}_F} b_C(\mathbf{x}_C) = 0 \tag{F-3}$$

or equivalently

$$\sum_{\mathbf{x}_C \setminus \mathbf{x}_F} b_C(\mathbf{x}_C) = \sum_{\mathbf{x}_E \setminus \mathbf{x}_F} b_E(\mathbf{x}_E). \tag{F-4}$$

Because $c_F = 0$, there will not be any $\mu$ constraint directly involving $b_F(\mathbf{x}_F)$, so we cannot derive some of the $\lambda$ constraints. On the other hand, these constraints are not necessary, because the region graph free energy itself also does not depend directly on $b_F(\mathbf{x}_F)$. We also see that the $\mu$ constraints are still sufficient to ensure that all the beliefs are consistent when they are marginalized down to region $F$. Finally, if we do surgery on this region graph and remove region $F$, we can then easily verify that the $\lambda$ constraints are then entirely equivalent to the $\mu$ constraints.

## REFERENCES


[1] B. J. Frey. *Graphical Models for Machine Learning and Digital Communication.* MIT Press, 1998.

[2] M. I. Jordan, editor. *Learning in graphical models.* MIT Press, 1998.

[3] L. R. Rabiner. A tutorial on hidden markov models and selected applications. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[4] A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. Info. Theory*, 13(2):260–269, 1967.

[5] G. D. Forney. The Viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

[6] R. G. Gallager. *Low-density parity check codes.* MIT Press, 1963.

[7] C. Berrou, A. Glavieux, and P. Thitimajshima. Near Shannon limit error-correcting coding and decoding: Turbo-codes. In *Proceedings 1993 IEEE International Conference on Communications*, pages 1064–1070, Geneva, Switzerland, 1993.

[8] R. J. McEliece, D. J. C. MacKay, and J. F. Cheng. Turbo decoding as an instance of Pearl's 'belief propagation' algorithm. *IEEE J. on Sel. Areas in Comm.*, 16(2):140–152, 1998.

[9] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference.* Morgan Kaufmann, 1988.

[10] R. E. Kalman. A new approach to linear filtering and prediction problems. *J. Basic Eng.*, 82:34–45, 1960.

[11] A. Gelb, editor. *Applied optimal estimation.* MIT Press, 1974.

[12] R. J. Baxter. *Exactly Solved Models in Statistical Mechanics.* Academic Press, 1982.

[13] R. Cowell. Advanced inference in Bayesian networks. In M.I. Jordan, editor, *Learning in Graphical Models.* MIT Press, 1998.

[14] K.P. Murphy, Y. Weiss, and M.I. Jordan. Loopy belief propagation for approximate inference: an empirical study. In *Proc. Uncertainty in AI*, 1999.

[15] H. A. Bethe. *Proc. Roy. Soc. London A*, 150:552, 1935.

[16] R. Kikuchi. A theory of cooperative phenomena. *Phys. Rev.*, 81(6):988–1003, 1951.

[17] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Generalized belief propagation. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13, Cambridge, MA, 2001. MIT Press.

[18] T. Morita. Cluster variation method for non-uniform Ising and Heisenberg models and spin-pair correlation function. *Prog. Theor. Phys.*, 85(2):243–255, 1991.

[19] Eds. T. Morita, M. Suzuki, K. Wada, and M. Kaburagi. Foundations and applications of cluster variation method and path probability method. In *Prog. Theor. Phys. Supplement*, volume 115, 1994.

[20] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. Info. Theory*, 47:498–519, 2001.

[21] G. D. Forney. Codes on graphs: normal realizations. *IEEE Trans. Info. Theory*, 47(2):520–548, 2001.

[22] T. P. Minka. Expectation propagation for approximate Bayesian inference. UAI-01, 2001.

[23] M. J. Wainwright, T. Jaakkola, and A. S. Willsky. Tree-based reparameterization for approximate estimation on loopy graphs. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2002.

[24] S. M. Aji and R. J. McEliece. The generalized distributive law. *IEEE Trans. Info. Theory*, 46:325–343, 2000.

[25] E. Offer and E. Soljanin. An algebraic description of iterative decoding schemes. In B. Marcus and J. Rosenthal, editors, *Codes, Systems and Graphical Models, Volumes in Mathematics and its Applications*. Springer Verlag, 2000.

[26] M. Mezard and R. Zecchina. The random k-satisfiability problem: from an analytic solution to an efficient algorithm. Available online at http://xxx.lanl.gov/pdf/cond-mat/0207194, 2002.

[27] S. M. Aji and R. J. McEliece. The generalized distributive law and free energy minimization. In *Proceedings of the 39th Allerton Conference on Communication, Control and Computing*, Allerton, Illinois, 2001.

[28] R. J. McEliece and M. Yildirim. Belief propagation on partially ordered sets. Fifteenth International Symposium on Mathematical Theory of Networks and Systems, 2002.

[29] P. Pakzad and V. Anantharam. Belief propagation and statistical physics. In *Proceedings of the Conference on Information Sciences and Systems*, 2002.

[30] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* John Wiley and Sons, 1991.

[31] M. I. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. An introduction to variational methods for graphical models. In M.I. Jordan, editor, *Learning in Graphical Models.* MIT Press, 1998.

[32] D. J. C. Mackay, J. S. Yedidia, W. T. Freeman, and Y. Weiss. A conversation about the Bethe free energy and sum-product (discussion document). Available online at http://www.merl.com/reports/TR2001-18/index.html, 2001.

[33] Y. Weiss. Correctness of local probability propagation in graphical models with loops. *Neural Computation*, 12:1–41, 2000.

[34] K. Huang. *Statistical Mechanics.* John Wiley and Sons, 1963.

[35] A. L. Yuille. A double-loop algorithm to minimize the Bethe and Kikuchi free energies. Unpublished, 2001.

[36] M. Welling and Y. W. Teh. Belief optimization: A stable alternative to belief propagation. UAI-01, 2001.

[37] H. J. Kappen and W. Wiegerinck. Novel iteration schemes for the cluster variation method. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2002.

[38] R. P. Stanley. *Enumerative Combinatorics*, volume 1. Cambridge University Press, 1997.

[39] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Characterizing belief propagation and its generalizations. Available online at http://www.merl.com/reports/TR2001-15/index.html, 2001.