Linköping Electronic Articles in
Computer and Information Science
Vol. n(2002): nr nn

Computing Kikuchi approximations by Cluster BP

Taisuke Sato

Department of Computer and Information Science
Tokyo Institute of Technology / CREST
Tokyo, Japan

Linköping University Electronic Press
Linköping, Sweden

http://www.ep.liu.se/ea/cis/2002/nnn/

Published on February nn,, 2002 by
Linköping University Electronic Press
581 83 Linköping, Sweden

Linköping Electronic Articles in
Computer and Information Science
ISSN 1401-9841
Series editor: Erik Sandewall

Typeset by the author using LaTeX
Formatted using étendu *style*

Recommended citation:
<Author>. <Title>. Linköping Electronic Articles in Computer and Information Science, Vol. n(2002): nr nn.
http://www.ep.liu.se/ea/cis/2002/nnn/. *February nn,, 2002.*

This URL will also contain a link to the author's home page.

Abstract

In this paper, we investigate *loopy BP* (*belief propagation*) which is at the crossroads of Bayesian networks, statistical physics and error correction decoding. It is a method of computing exact and approximate marginals based on a minimization of variational free energy, and offers a quite powerful means for otherwise computationally intractable problems. We re-examine its theoretical background, the Kikuchi approximation, and propose *Cluster BP*, a mild generalization of loopy BP that can compute some class of Kikuchi approximations. We also propose *Cluster CCCP* as a convergent version of Cluster BP to compute local minima of Kikuchi approximations.

1 Introduction

In this paper, we investigate[1] *loopy BP* (*belief propagation*) which is at the crossroads of Bayesian networks, statistical physics and error correction decoding. It is a method of computing exact and approximate marginals based on the minimization of variational free energy, and offers a quite powerful means for otherwise computationally intractable problems.

Loopy BP came out of the study of Bayesian networks[2] [13, 3, 5]. Pearl proposed the BP (belief propagation) algorithm for computing marginal distributions [12]. It is a message passing algorithm that propagates messages (probability distributions) over a Bayesian network. When the network is singly connected, i.e. no loop appears when viewed as an undirected graph, it computes exact marginals in time linear in the size of the network [12].[3] As BP is just a local message passing algorithm, it is applicable even when the network contains loops (hence the name "loopy BP"), though at the risk of non-convergence. Historically while what loopy BP does was obscure in the case of loopy networks, experiments revealed that it is very versatile and effective, and sometimes can give remarkably good approximate marginals especially when used for error correction decoding [8, 10, 18]. It was not until recently that Yedidia et al. discovered that what loopy BP actually does is computing a stationary point of a certain quantity known as the Bethe approximation to the free energy in statistical physics [19]. The Bethe approximation is just one of the possible approximations to the free energy. Yedidia et al. proposed *GBP* (generalized BP), a new message passing algorithm that yields, when it converges, a stationary point of the Kikuchi approximation which is more complicated but more accurate than the Bethe approximation as a generalization of the Bethe approximation [9, 19].[4] Yuille proposed an alternative to GBP, *CCCP double-loop algorithm*[5], that is guaranteed to converge to local minima of the Bethe and Kikuchi approximations. This convergence property is achieved by his new optimizing technique for the class of functions decomposable as a sum of convex and concave functions [21].

While loopy BP is only able to compute stationary points of Bethe approximations, it is simple and efficient. Contrastingly GBP and CCCP for Kikuchi approximations are quite general but rather complex. In this

[1] Our investigation is motivated by PRISM, a general symbolic-statistical modeling language which we have been developing for years [14, 16, 17]. It is based on a rigorous mathematical semantics and enables one to build a complex statistical model as a PRISM program and efficiently learn parameters embedded in the program through the graphical EM algorithm [15, 17]. The computational efficiency is achieved however by restricting programming style so that it complies with assumptions of probability computation employed by PRISM. We are seeking for a general yet robust method of probability computation to realize more freedom of PRISM programming.

[2] A Bayesian network is a finite directed acyclic graph representing probabilistic causal relationships between random variables. Vertices are random variables which are connected by directed edges from parent vertices to child vertices. A conditional distribution $p(X = x \mid Y_1 = y_1, \ldots, Y_m = y_m)$ $(m \geq 0)$ is associated with each vertex X and its parents $Y_1, \ldots, Y_m$. The graph defines a joint distribution as a product of these conditional distributions.

[3] For Bayesian networks which are not singly connected, the junction tree algorithm [7] is available to efficiently compute marginal distributions.

[4] Descriptions of the Bethe and Kikuchi approximations in this paper are largely due to [19].

[5] "CCCP" stands for the Concave-Convex Procedure.

paper, we take an intermediate approach. Keeping the simplicity of loopy BP, we extend it to the *Cluster BP* algorithm allowing messages to contain more than one variable so that it can compute Kikuchi approximations when a certain condition is satisfied. Furthermore, to remove the possibility of non-convergence, we derive the *Cluster CCCP* algorithm following Yuille's convex-concave decomposition. It always converges and gives the same result as the Cluster BP algorithm.

In the sequel, we first quickly review the problem to be solved. We then introduce the Kikuchi approximation in a slightly generalized form by dropping the requirement that clusters used in the approximation must be closed under intersection. We also introduce a class of undirected labeled graphs called cluster graphs. After these preparations, two new algorithms, *Cluster BP* and *Cluster CCCP*, are proposed which run on cluster graphs. Both compute stationary points of (generalized) Kikuchi approximations but the latter is convergent and guaranteed to compute a local minimum. The reader is assumed to be familiar with Bayesian networks and the junction tree algorithm [4, 3].

2 Preliminaries

2.1 Variable free energy

We assume a discrete random vector $X = (X_1, \ldots, X_n)$ of n dimension is given and has a joint distribution $p(x) = p(x_1, \ldots, x_n)$ such that

- $p(x)$ is always positive, $p(x) > 0$ for any x.
- $p(x)$ is presented as a product of positive functions $\psi_\alpha(x_\alpha)$ called *potentials*.

$$
\begin{aligned}
p(x) &= \kappa \prod_{\alpha \in \mathbf{P}} \psi_\alpha(x_\alpha) && (1)\\
&= \frac{1}{Z} e^{-E(x)} \\
\text{where} & \\
E(x) &\stackrel{\text{def}}{=} \sum_{\alpha \in \mathbf{P}} -\ln \psi_\alpha(x_\alpha) && (2)\\
Z &\stackrel{\text{def}}{=} \kappa^{-1} && (3)\\
&= \sum_x \prod_{\alpha \in \mathbf{P}} \psi_\alpha(x_\alpha)
\end{aligned}
$$

Here κ is a normalizing constant and Z is a partition function. $E(x)$ is considered to be the "energy" of a state x. We call each i $(1 \leq i \leq n)$ a *variable index*. Let α denote a sub-vector of $(1, \ldots, n)$, the vector of all variable indecies, and $\mathbf{P}$ be a collection of such sub-vectors. x_α then stands for a sub-vector of $(x_1, \ldots, x_n)$ corresponding to α. For instance, if $x = (x_1, x_2, x_3, x_4)$ and $\alpha = (2, 4)$, we have $x_\alpha = (x_2, x_4)$. For convenience, we identify a sub-vector $\alpha = (i_1, \ldots, i_k)$ $(1 \leq k \leq n)$ with the set $\{i_1, \ldots, i_k\}$ like $\alpha = (2, 4) = \{2, 4\}$. Because both notations are mutually convertible, such treatment does not cause confusion.

A non-empty subset α of $\{1, \ldots, n\}$ is called a *cluster* and if α is in $\mathbf{P}$, it is referred to as a *potential cluster*. When α and β are clusters, so are $\alpha \cap \beta$,

$\alpha \cup \beta$ and $\alpha \setminus \beta$. Operations on clusters reflect on vectors. For example, put $\alpha = \{1,2,3\}$ and $\beta = \{2,3,4\}$. Then $x_{\alpha\cap\beta} = x_{\{2,3\}} = (x_2, x_3)$ and $x_{\alpha\setminus\beta} = x_{\{1\}} = x_1$. $A \subset B$ means A is a proper subset of B whereas $A \subseteq B$ means $A \subset B$ or $A = B$.

We define the *variational free energy* $F(b)$ w.r.t. the energy $E(x)$ by

$$\begin{aligned} F(b) &\stackrel{\text{def}}{=} \text{(average energy)} - \text{(entropy)} \\ &= \sum_x b(x)E(x) + \sum_x b(x)\ln b(x) \end{aligned} \tag{4}$$

where $b(\cdot)$ is a test distribution. Because $F(b) = \sum_x b(x)\ln\left(\frac{b(x)}{p(x)}\right) - \ln Z$ and the Kullback-Leibler divergence $\sum_x b(x)\ln\left(\frac{b(x)}{p(x)}\right)$ is non-negative and zero only when $b(\cdot) = p(\cdot)$, $F(b)$ takes a minimum value if and only if $b(\cdot) = p(\cdot)$. In other words, we have an equation $p = \text{argmin}_b F(b)$.

Suppose a distribution $p(x)$ is given by (1). For the variational free energy $F(b)$, we can define its approximation, the *Kikuchi approximation* $F_K(\{b_\alpha\})$ such that $F(b) \approx F_K(\{b_\alpha\})$.[6] Combining the Kikuchi approximation with $p = \text{argmin}_b F(b)$, we obtain an approximation scheme for marginal distributions $\{p_\alpha(x_\alpha)\}$ of $p(x)$:

$$\{p_\alpha\} \approx \text{argmin}_{\{b_\alpha\}} F_K(\{b_\alpha\}).$$

We read this equation from right to left and calculate approximate marginal distributions $\{p_\alpha\}$ by minimizing the functional $F_K(\{b_\alpha\})$ with respect to $\{b_\alpha\}$ as variational variables. In view of the significance of computing marginals in many fields including pattern recognition, natural language processing, robotics, bioinformatics, coding theory etc on one hand and the difficulty of their exact computation on the other hand, it is quite important to develop efficient algorithms for computing $\text{argmin}_{\{b_\alpha\}} F_K(\{b_\alpha\})$. The objective of this paper is to propose such algorithms while preserving the simplicity of BP.

2.2 The Kikuchi approximation

In this subsection, we introduce the Kikuchi approximation [9, 19] in a slightly generalized form. Let $\mathbf{P}$ be a set of potential clusters appearing in (1). For $\mathbf{P}$ we introduce another set $\mathbf{U}$ of clusters such that

- for any α in $\mathbf{P}$, there exists the smallest $\beta \in \mathbf{U}$ that includes α, i.e. $\beta \supseteq \alpha$ and if $\beta' \supseteq \alpha$ then $\beta' \supseteq \beta$ for any $\beta' \in \mathbf{U}$.[7]

$\mathbf{U}$ is called *a set of clusters for* $\mathbf{P}$. We consider $\mathbf{U}$ as a *partially ordered set* ordered by set inclusion ordering. Let $\mathbf{B}$ be the set of maximal clusters in $\mathbf{U}$. If $\mathbf{U}$ is generated from $\mathbf{B}$ by taking all possible non-empty intersections of elements in $\mathbf{B}$, i.e. $\mathbf{U} = \{\alpha_1 \cap \ldots \cap \alpha_h \mid \alpha_i \in \mathbf{B}, 0 < h, 1 \le i \le h\}$, a cluster in $\mathbf{U}$ is called a *Kikuchi cluster*, and $\mathbf{U}$ *a set of Kikuchi clusters for* $\mathbf{P}$.

We associate an *overcounting number* a_α with a cluster α in $\mathbf{U}$ which is inductively defined from maximal clusters as follows[9, 19].

---

[6] $b_\alpha(x_\alpha)$ (resp. $p_\alpha(x_\alpha)$) denotes a marginal distribution of $b(x)$ (resp. $p(x)$) marginalized to x_α.

[7] This condition is trivially satisfied by putting $\mathbf{U} = \mathbf{P}$, but for the sake of freedom of approximations, we prefer to introduce $\mathbf{U}$ independently of $\mathbf{P}$.

$$\begin{cases} a_\alpha \stackrel{\text{def}}{=} 1 \text{ if } \alpha \text{ is maximal in } \mathbf{U} \\ a_\alpha \stackrel{\text{def}}{=} 1 - \sum_{\beta:\beta\supset\alpha,\beta\in\mathbf{U}} a_\beta \text{ o.w.} \end{cases}$$

By definition

$$\sum_{\alpha:\alpha\in\mathbf{U},\alpha\supseteq\beta} a_\alpha = 1 \text{ for } \forall\beta\in\mathbf{U}.$$

Introduce $E^+_\alpha(x_\alpha) \stackrel{\text{def}}{=} -\ln\psi_\alpha(x_\alpha)$ and transform the average energy $\sum_x b(x)E(x)$ using the above property of overcounting numbers as follows.

$$\begin{aligned} \sum_x b(x)E(x) &= \sum_x b(x) \sum_{\beta:\beta\in\mathbf{P}} E^+_\beta(x_\beta) \\ &= \sum_{\beta:\beta\in\mathbf{P}} \sum_{x_\beta} b_\beta(x_\beta)E^+_\beta(x_\beta) \\ &= \sum_{\beta:\beta\in\mathbf{P}} \left(\sum_{\alpha\in\mathbf{U},\alpha\supseteq\beta} a_\alpha \right) \sum_{x_\beta} b_\beta(x_\beta)E^+_\beta(x_\beta) \\ &= \sum_{\alpha\in\mathbf{U}} a_\alpha \sum_{x_\alpha} b_\alpha(x_\alpha)E_\alpha(x_\alpha) \end{aligned} \tag{5}$$

Here $b_\beta(x_\beta) \stackrel{\text{def}}{=} \sum_{x_{[n]\setminus\beta}} b(x)$ ($[n] = \{1,\ldots,n\}$) is a marginal distribution and $E_\alpha(x_\alpha) \stackrel{\text{def}}{=} \sum_{\beta:\beta\in\mathbf{P},\beta\subseteq\alpha} E^+_\beta(x_\beta)$ is the energy of cluster α. Now the average energy is expressed as a linear combination of average cluster energies. It is unfortunate that a similar transformation of the entropy $S = -\sum_x b(x)\ln b(x)$ is impossible because of non-linearity of the entropy function. Yet we can derive its approximation as a linear combination of cluster entropies. First introduce S_α, the entropy of a cluster α by

$$S_\alpha \stackrel{\text{def}}{=} -\sum_{x_\alpha} b_\alpha(x_\alpha)\ln b_\alpha(x_\alpha).$$

Inductively define S^+_α from minimal elements of $\mathbf{U}$ by

$$\begin{cases} S^+_\alpha \stackrel{\text{def}}{=} S_\alpha \text{ if } \alpha \text{ is minimal in } \mathbf{U} \\ S^+_\alpha \stackrel{\text{def}}{=} S_\alpha - \sum_{\beta:\beta\subset\alpha,\beta\in\mathbf{U}} S^+_\beta \text{ o.w.} \end{cases}$$

For every $\alpha\in\mathbf{U}$, we have

$$S_\alpha = \sum_{\beta:\beta\subseteq\alpha,\beta\in\mathbf{U}} S^+_\beta.$$

Accordingly if we *assume* $S \approx \sum_{\beta\in\mathbf{U}} S^+_\beta$,[8] we see

$$\begin{aligned} S &\approx \sum_{\beta\in\mathbf{U}} S^+_\beta \\ &= \sum_{\beta\in\mathbf{U}} \left(\sum_{\alpha\in\mathbf{U},\alpha\supseteq\beta} a_\alpha \right) S^+_\beta \\ &= \sum_{\alpha\in\mathbf{U}} a_\alpha S_\alpha. \end{aligned} \tag{6}$$

[8] This assumption is justified empirically in statistical physics. The reader is advised to see [9] for the development of the Kikuchi approximation.

Putting (5) and (6) together, the variational free energy $F(b)$ is approximated as follows.

$$\begin{aligned} F(b) &= \sum_x b(x)E(x) + \sum_x b(x)\ln b(x) \\ &\approx \sum_{\alpha\in\mathbf{U}} a_\alpha \left(\sum_{x_\alpha} b_\alpha(x_\alpha)E_\alpha(x_\alpha) + \sum_{x_\alpha} b_\alpha(x_\alpha)\ln b_\alpha(x_\alpha) \right) \end{aligned}$$

We define the *generalized* [9] *Kikuchi approximation* $\tilde{F}_K(\{b_\alpha\})$ for $F(b)$ by

$$\begin{aligned} \tilde{F}_K(\{b_\alpha\}) &\stackrel{\text{def}}{=} \sum_{\alpha\in\mathbf{U}} a_\alpha \left(\sum_{x_\alpha} b_\alpha(x_\alpha)E_\alpha(x_\alpha) + \sum_{x_\alpha} b_\alpha(x_\alpha)\ln b_\alpha(x_\alpha) \right) \\ &= \sum_{\alpha\in\mathbf{U}} a_\alpha \sum_{x_\alpha} b_\alpha(x_\alpha)\ln\left(\frac{b_\alpha(x_\alpha)}{\phi_\alpha(x_\alpha)}\right) \qquad (7) \end{aligned}$$

where

$$\begin{aligned} \phi_\alpha(x_\alpha) &\stackrel{\text{def}}{=} e^{-E_\alpha(x_\alpha)} \\ &= \exp\left(\sum_{\beta:\beta\in\mathbf{P},\beta\subseteq\alpha} -E^+_\beta(x_\beta) \right) \\ &= \prod_{\beta:\beta\in\mathbf{P},\beta\subseteq\alpha} \psi_\beta(x_\beta). \qquad (8) \end{aligned}$$

When $\mathbf{U}$ is a set of Kikuchi clusters for $\mathbf{P}$, $\tilde{F}_K(\{b_\alpha\})$ is identical to the Kikuchi approximation defined in [19]. The Bethe approximation is a special case of the Kikuchi approximation where $\mathbf{U}$ is a union of clusters $\mathbf{B}$ of the form $\{i,j\}$ $(i \neq j)$ and a set of intersections $\{\alpha_1 \cap \alpha_2 \mid \alpha_1,\alpha_2 \in \mathbf{B}\}$.

3 Cluster graphs

Just like Pearl's BP runs on Bayesian networks and the junction tree algorithm runs on junction trees, our new algorithms, *Cluster BP* and *Cluster CCCP*, run on *cluster graph*s which can be seen as a generalization of junction trees to graphs and also can be seen as a refinement of "junction graphs" [2] with a partial ordering over clusters and a suitable condition for computing the Kikuchi approximation. From here on for brevity, we interchangeably use $n_1 \cdots n_h$ for cluster $\{n_1,\ldots,n_h\}$ like 123 for $\{1,2,3\}$.

Let $\mathbf{U}$ be a set of clusters for $\mathbf{P}$. A labeled undirected graph $\mathcal{G}_U$ is said to be a *cluster graph for* $\mathbf{U}$ if vertices and edges are labeled by clusters in $\mathbf{U}$ as follows and if there is such a graph, we say $\mathbf{U}$ has a cluster graph.[10]

- Every maximal element in $\mathbf{U}$ appears exactly once in $\mathcal{G}_U$ as a label of a vertex.

[9] The term "generalized" is added to make clear the distinction between the Kikuchi approximation defined in [19] and the one defined here which does not require $\mathbf{U}$ to be closed under intersection.

[10] $\mathbf{U}$ may not have a cluster graph. For instance $\mathbf{U}_1 = \{1467, 2457, 3567, 47, 57, 67\}$ has a cluster graph but $\mathbf{U}_2 = \mathbf{U}_1 \cup \{7\}$ has no cluster graph. If the Hasse diagram $H_\mathbf{U}$ of $\mathbf{U}$ is a tree, then $H_\mathbf{U}$ is a cluster graph for $\mathbf{U}$. In general however, finding an efficient algorithm for the construction of cluster graphs is a future research topic.

- Suppose vertices v_1 and v_2 are respectively labeled α_1 and α_2. The edge connecting v_1 and v_2, if any, is labeled $\beta \subseteq \alpha_1 \cap \alpha_2$ ($\beta \neq \emptyset$). When the equality $\beta = \alpha_1 \cap \alpha_2$ holds for every label on an edge, we say $\mathcal{G}_U$ is *regular*.
- The *tree condition* is satisfied: for every $\alpha \in \mathbf{U}$, the subgraph of $\mathcal{G}_U$ consisting only of the vertices and edges whose label include α is a tree.

Please note that we allow the same cluster to occur (as a label) more than once in a cluster graph. So vertices and edges can be labeled by a common cluster. By the way it is obvious that junction trees are cluster graphs such that nodes are cliques and the tree condition is satisfied.

We have a look at examples of cluster graphs.

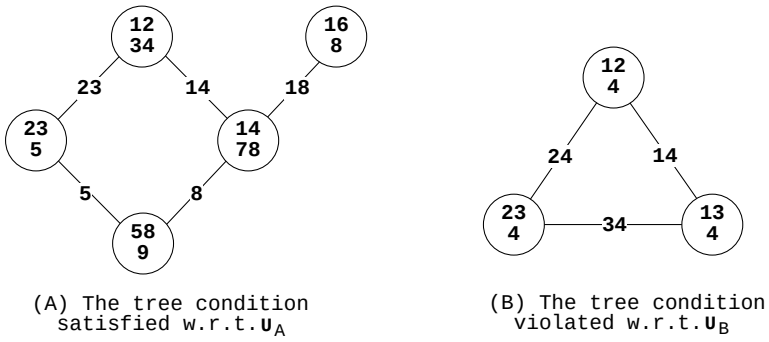

Figure 1: The left graph (A) satisfies the tree condition w.r.t. $\mathbf{U}_{\mathrm{A}}$. The right graph (B) violates the tree condition w.r.t. $\mathbf{U}_{\mathrm{B}}$.

The left graph (A) in Figure 1 is a cluster graph for $\mathbf{U}_{\mathrm{A}} = \{1234, 2356, 589, 1478, 168, 14, 18, 23, 1, 5, 8\}$. It is regular. The maximal clusters are $\mathbf{B}_{\mathrm{A}} = \{1234, 2356, 589, 1478, 168\}$. Since their intersections yield $\mathbf{U}_{\mathrm{A}}$, $\mathbf{U}_{\mathrm{A}}$ is a set of Kikuchi clusters for $\mathbf{P} = \mathbf{B}_{\mathrm{A}}$. Associated overcounting numbers are $a_{1234} = \cdots = a_{168} = 1$ for the maximal clusters, $a_{23} = a_5 = a_8 = a_{14} = a_{18} = -1$ and $a_1 = 0$ for the rest. We check the tree condition. For example the subgraph comprised of vertices and edges that include cluster 23 is a tree. Similarly, the one containing 1, though it does not appear as a label, is also a tree and so on. So (A) satisfies the tree condition. The right graph (B) however does not satisfies the tree condition w.r.t. $\mathbf{U}_{\mathrm{B}} = \{124, 234, 134, 14, 24, 34, 4\}$ because all vertices and edges include 4 and they form a loop. (B) is not a cluster graph for $\mathbf{U}_{\mathrm{B}}$ (instead it is a cluster graph for $\mathbf{U}'_{\mathrm{B}} = \mathbf{U}_{\mathrm{B}} \setminus \{4\}$).

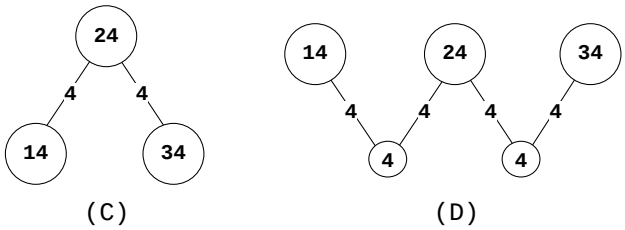

Figure 2: Cluster graphs with a multiple occurrence of the same label

In Figure 2 both (C) and (D) are cluster graphs for the same cluster set $\mathbf{U}_{\mathrm{C}} = \{14, 24, 34, 4\}$.

We next prove a proposition that states the relationship between the tree condition and overcounting numbers.

Proposition 3.1
Let $\mathcal{G}_U$ be a cluster graph for **U** *and a_α an overcounting number of $\alpha \in$* **U**. *Also let $V(\alpha)$ and $E(\alpha)$ respectively be*

$$V(\alpha) \stackrel{\text{def}}{=} \text{the number of vertices whose label is } \alpha$$
$$E(\alpha) \stackrel{\text{def}}{=} \text{the number of edges whose label is } \alpha.$$

We have $a_\alpha = V(\alpha) - E(\alpha)$.

(Proof) By induction the size of clusters. Suppose α is a maximal cluster. Then there is exactly one vertex labeled α whereas there is no edge labeled α by the tree condition. So $V(\alpha) - E(\alpha) = 1 = a_\alpha$. Now suppose the proposition holds for all $\beta \supset \alpha$.

$$\begin{aligned} a_\alpha &= 1 - \sum_{\beta:\alpha\subset\beta\in\mathbf{U}} a_\beta \\ &= 1 - \sum_{\beta:\alpha\subset\beta\in\mathbf{U}} (V(\beta) - E(\beta)) \\ &= \left\{\begin{array}{l}\text{the number of edges}\\ \text{whose label properly}\\ \text{include } \alpha\end{array}\right\} - \left\{\begin{array}{l}\text{the number of vertices}\\ \text{whose label properly}\\ \text{include } \alpha\end{array}\right\} + 1 \end{aligned}$$

Recalling that the following holds by the tree condition

$$V(\alpha) + \left\{\begin{array}{l}\text{the number of vertices}\\ \text{whose label properly}\\ \text{include } \alpha\end{array}\right\} = E(\alpha) + \left\{\begin{array}{l}\text{the number of edges}\\ \text{whose label properly}\\ \text{include } \alpha\end{array}\right\} + 1$$

$$= V(\alpha) - E(\alpha) \qquad \text{Q.E.D.}$$

Proposition 3.2
Suppose **U** *has a cluster graph.*
Then for every $\alpha \in$ **U**, *$a_\alpha \leq 0$ if α is not maximal in* **U**.

(Proof) Let α be a non-maximal element in **U**. There is a maximal $\beta \in$ **U** such that $\alpha \subset \beta$. If no vertex bears α as a label, then $a_\alpha \leq 0$ by Proposition 3.1. Suppose otherwise and a vertex v is labeled by α. Due to the tree condition, there is a path connecting v and w where w is the vertex labeled by β. Let u, β' and γ respectively be a vertex on the path adjacent to v, its label and the label of the edge connecting u and v. We have $\alpha \subseteq \beta'$ and $\alpha \subseteq \gamma$ because all vertices and edges on the path have a label containing α. On the other hand, due to the labeling condition, $\gamma \subseteq \alpha \cap \beta' = \alpha$. Hence $\gamma = \alpha$. We may assume $\beta' \neq \alpha$ (o.w. we merge u and v and their labels β'). So there is a map from the vertex v labeled α and its edge labeled α connecting to an adjacent node whose label is not α. Consequently, from Proposition 3.1, we conclude $a_\alpha \leq 0$. Q.E.D.

We remark that "overcounting numbers of non-maximal clusters are not positive" is a necessary condition for the tree condition but not a sufficient condition.

4 Cluster BP

In this section, we derive an iterative algorithm, *Cluster BP*, running on cluster graphs which computes a stationary point of the generalized Kikuchi approximation by exchanging messages between vertices in a cluster graph.

4.1 Deriving Cluster BP

We restate our assumptions and notational conventions.

- A joint distribution $p(x) = \kappa \prod_{\alpha \in \mathbf{P}} \psi_\alpha(x_\alpha)$ is specified by using potential clusters $\mathbf{P}$ and potentials $\psi_\alpha(x_\alpha)$. Potentials are always positive. We aim is to efficiently compute approximate marginals $\{p_\alpha(x_\alpha)\}$ by minimizing the generalized Kikuchi approximation $\tilde{F}_K(\{b_\alpha\})$ defined by (7).
- $\mathbf{U}$ is a set of clusters for $\mathbf{P}$. $\phi_\alpha(x_\alpha)$ is the α's potential defined by $\phi_\alpha(x_\alpha) \stackrel{\text{def}}{=} \prod_{\beta:\beta\in\mathbf{P},\beta\subseteq\alpha} \psi_\beta(x_\beta)$.
- $\mathcal{G}_U$ is a cluster graph for $\mathbf{U}$. We denote by V (resp. E) the set of vertices (resp. edges) in $\mathcal{G}_U$, and by $\underline{v}$ (resp. by $\underline{e}$) the cluster labeling a vertex $v \in V$ (resp. the cluster labeling an edge $e \in E$).

Now back to the generalized Kikuchi approximation $\tilde{F}_K(\{b_\alpha\})$. We rewrite it using Proposition 3.1 as follows.

$$
\begin{aligned}
\tilde{F}_K(\{b_\alpha\}) &= \sum_{\alpha\in\mathbf{U}} a_\alpha \sum_{x_\alpha} b_\alpha(x_\alpha) \ln\left(\frac{b_\alpha(x_\alpha)}{\phi_\alpha(x_\alpha)}\right) \\
&= \sum_{\alpha\in\mathbf{U}} (V(\alpha) - E(\alpha)) \sum_{x_\alpha} b_\alpha(x_\alpha) \ln\left(\frac{b_\alpha(x_\alpha)}{\phi_\alpha(x_\alpha)}\right) \\
&= \sum_{\alpha\in\mathbf{U}} V(\alpha) \sum_{x_\alpha} b_\alpha(x_\alpha) \ln\left(\frac{b_\alpha(x_\alpha)}{\phi_\alpha(x_\alpha)}\right) \\
&\qquad - \sum_{\alpha\in\mathbf{U}} E(\alpha) \sum_{x_\alpha} b_\alpha(x_\alpha) \ln\left(\frac{b_\alpha(x_\alpha)}{\phi_\alpha(x_\alpha)}\right) \\
&= \sum_{v\in V} \sum_{x_{\underline{v}}} b_{\underline{v}}(x_{\underline{v}}) \ln\left(\frac{b_{\underline{v}}(x_{\underline{v}})}{\phi_{\underline{v}}(x_{\underline{v}})}\right) - \sum_{e\in E} \sum_{x_{\underline{e}}} b_{\underline{e}}(x_{\underline{e}}) \ln\left(\frac{b_{\underline{e}}(x_{\underline{e}})}{\phi_{\underline{e}}(x_{\underline{e}})}\right)
\end{aligned}
$$

Note the last equation is expressed solely in terms of marginal distributions $\{b_{\underline{v}}(x_{\underline{v}}), b_{\underline{e}}(x_{\underline{e}})\}$ that correspond to labels of vertices and edges in $\mathcal{G}_U$.

We minimize $\tilde{F}_K(\{b_\alpha\})$ as a functional over $\{b_{\underline{v}}(x_{\underline{v}}), b_{\underline{e}}(x_{\underline{e}})\}$ to obtain approximate marginals but the minimization must be carried out under two constraints.

Distribution: $\sum_{x_{\underline{v}}} b_{\underline{v}}(x_{\underline{v}}) = 1$ for every vertex v.

Consistency: $\sum_{x_{\underline{v}\setminus\underline{e}}} b_{\underline{v}}(x_{\underline{v}}) = b_{\underline{e}}(x_{\underline{e}})$ for every vertex v and its edge e.[11]

[11] For any clusters β, α ($\beta \subset \alpha$), the tree condition guarantees the existence of a path connecting β and α regardless of whether they label a vertex or not. Hence, if the consistency holds along this path, $\sum_{x_{\alpha\setminus\beta}} b_\alpha(x_\alpha) = b_\beta(x_\beta)$ must hold transitively.

So we introduce a Lagrangian $\mathcal{L} = F_K(\{b_{\underline{v}}(x_{\underline{v}}), b_{\underline{e}}(x_{\underline{e}})\}) + \mathcal{C}$ with a constraint term $\mathcal{C}$.

$$\begin{aligned}\mathcal{C} \stackrel{\text{def}}{=} \sum_{e\in E}\sum_{x_{\underline{e}}} \Bigg\{ & \lambda^{e}_{\underline{v_1}}(x_{\underline{e}}) \left(\sum_{x_{\underline{v_1}\setminus\underline{e}}} b_{\underline{v_1}}(x_{\underline{v_1}}) - b_{\underline{e}}(x_{\underline{e}}) \right) \\ & + \lambda^{e}_{\underline{v_2}}(x_{\underline{e}}) \left(\sum_{x_{\underline{v_2}\setminus\underline{e}}} b_{\underline{v_2}}(x_{\underline{v_2}}) - b_{\underline{e}}(x_{\underline{e}}) \right) \Bigg\} + \sum_{v\in V} \lambda_{\underline{v}} \left(\sum_{x_{\underline{v}}} b_{\underline{v}}(x_{\underline{v}}) - 1 \right)\end{aligned}$$

One thing must be noted here. Setting the derivatives of $\mathcal{L}$ with respect to $\{b_{\underline{v}}(x_{\underline{v}}), b_{\underline{e}}(x_{\underline{e}})\}$ equal to zero does not give an algorithm which exchanges messages among vertices because $\mathcal{L}$'s variational variables correspond to labels of vertices, not to vertices themselves (some vertices may have a common label). We therefore inflate variational variables by introducing new variables $\{b_v(x_{\underline{v}}), b_e(x_{\underline{e}})\}$ which have one-to-one correspondence to vertices v and edges e.[12] However this change does not affect the solution because the tree condition ensures that those vertices or edges with a common label will have the same distribution. So our final Lagrangian becomes

$$\begin{aligned}\mathcal{L}' &= \tilde{F}'_K + \mathcal{C}' \\ \tilde{F}'_K &= \sum_{v\in V}\sum_{x_{\underline{v}}} b_v(x_{\underline{v}}) \ln \left(\frac{b_v(x_{\underline{v}})}{\phi_{\underline{v}}(x_{\underline{v}})} \right) - \sum_{e\in E}\sum_{x_{\underline{e}}} b_e(x_{\underline{e}}) \ln \left(\frac{b_e(x_{\underline{e}})}{\phi_{\underline{e}}(x_{\underline{e}})} \right) \qquad (9) \\ &= \sum_{v\in V}\sum_{x_{\underline{v}}} b_v(x_{\underline{v}}) \ln \left(\frac{b_v(x_{\underline{v}})}{\phi'_{\underline{v}}(x_{\underline{v}})} \right) - \sum_{e\in E}\sum_{x_{\underline{e}}} b_e(x_{\underline{e}}) \ln b_e(x_{\underline{e}}) \qquad (10) \\ \mathcal{C}' &= \sum_{e\in E}\sum_{x_{\underline{e}}} \Bigg\{ \lambda^{e}_{v_1}(x_{\underline{e}}) \left(\sum_{x_{\underline{v}1\setminus\underline{e}}} b_{v_1}(x_{\underline{v_1}}) - b_e(x_{\underline{e}}) \right) \\ & \qquad + \lambda^{e}_{v_2}(x_{\underline{e}}) \left(\sum_{x_{\underline{v}2\setminus\underline{e}}} b_{v_2}(x_{\underline{v_2}}) - b_e(x_{\underline{e}}) \right) \Bigg\} \\ & \qquad + \sum_{v\in V} \lambda_v \left(\sum_{x_{\underline{v}}} b_v(x_{\underline{v}}) - 1 \right) \qquad (11)\end{aligned}$$

Here $\lambda^{e}_{v_i}(x_{\underline{e}})$ $(i = 1, 2)$ is a Lagrange multiplier for the vertex v_i connected by an edge e. It is introduced to ensure the consistency of marginal distributions associated with v_i and e. In the transition from (9) to (10), we performed *an adjustment of potentials* to displace potentials from edges, which is performed nondeterministically as follows (there is a lot of freedom).

1. For each edge e, choose a vertex v from those connected by e. We write $v = \theta(e)$.
2. Let $\phi_{uv}(x_{\underline{v}})$ be the potential of v and $\phi_{\underline{e}}(x_{\underline{e}})$ the potential of e. Define the *adjusted potential for* v by

[12] If $v \neq v'$, $b_v(x_{\underline{v}})$ and $b_{v'}(x_{\underline{v}'})$ are independent variables. If they have a common label, i.e. $\underline{v} = \underline{v}'$ however, $b_v(x_{\underline{v}}) = b_{v'}(x_{\underline{v}'})$ must hold. The same applies to edges.

$$\phi'_v(x_{\underline{v}}) = \frac{\phi_{\underline{v}}(x_{\underline{v}})}{\prod_{e:\theta(e)=v} \phi_{\underline{e}}(x_{\underline{e}})}$$

We say $\phi_{\underline{e}}(x_{\underline{e}})$ *is moved to* v.

We show examples. In Figure 3, suppose the potential of edge 23 is moved to vertex 234, that of edge 4 to vertex 145, that of edge 1 to vertex 123 and that of edge 14 to vertex 145. The adjusted potentials are respectively $\phi'_{123}(x_{123}) = \frac{\phi_{123}(x_{123})}{\phi_1(x_1)}$, $\phi'_{234}(x_{234}) = \frac{\phi_{234}(x_{234})}{\phi_{23}(x_{23})}$, $\phi'_{145}(x_{145}) = \frac{\phi_{145}(x_{145})}{\phi_4(x_4)\phi_{14}(x_{14})}$ and $\phi'_{14}(x_{14}) = \phi_{14}(x_{14})$.

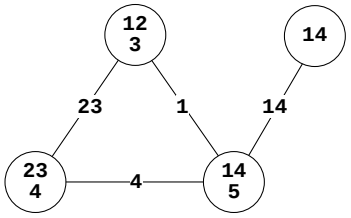

Figure 3: An adjustment of potentials

Now we calculate the derivatives of $\mathcal{L}'$ with respect to $\{b_v(x_{\underline{v}}), b_e(x_{\underline{e}})\}$ and put them equal to 0. We here change notations. We use i, j, k for vertices and denote by (ij) the edge connecting i and j and by $\underline{ij}$ the cluster labeling (ij). The results are

$$\begin{aligned} b_i(x_{\underline{i}}) &= \kappa\, \phi'_i(x_{\underline{i}}) \prod_{k \in N(i)} e^{-\lambda_k^{(ki)}(x_{\underline{ki}})} \\ b_{(ij)}(x_{\underline{ij}}) &= \kappa'\, e^{-\lambda_i^{(ij)}(x_{\underline{ij}})} e^{-\lambda_j^{(ij)}(x_{\underline{ij}})} \end{aligned}$$

where $N(i)$ is a set of the vertices adjacent to i and κ, κ' are normalizing constants. Applying the distribution and consistency conditions to a pair of $b_i(x_{\underline{i}})$ and $b_{(ij)}(x_{\underline{ij}})$ and introducing

$$m_{i \triangleright j}(x_{\underline{ij}}) \stackrel{\text{def}}{=} \kappa\, e^{-\lambda_i^{(ij)}(x_{\underline{ij}})}$$

we reach the *Cluster BP algorithm* in Figure 4 that computes stationary points of generalized Kikuchi approximations. A multi-variate function $m_{i \triangleright j}(x_{\underline{ij}})$ is considered as a message from vertex i to vertex j. $\phi'_i(x_{\underline{i}})$ is an adjusted potential associated with i. $N(i) \setminus j$ denotes the set $N(i)$ with the vertex j deleted.

$$m_{i \triangleright j}(x_{\underline{ij}}) = \kappa \sum_{x_{\underline{i} \setminus \underline{ij}}} \phi'_i(x_{\underline{i}}) \prod_{k \in N(i) \setminus j} m_{k \triangleright i}(x_{\underline{ki}}) \tag{12}$$

$$b_i(x_{\underline{i}}) = \kappa'\, \phi'_i(x_{\underline{i}}) \prod_{k \in N(i)} m_{k \triangleright i}(x_{\underline{ki}}) \tag{13}$$

Figure 4: Cluster BP

This algorithm starts with initial messages $\{m_{i \triangleright j}(x_{\underline{ij}}) = 1\}$ and iteratively updates $\{m_{i \triangleright j}(x_{\underline{ij}})\}$ by substituting the right hand side of (12) for the left hand side infinitely (sufficiently) many times. After all $\{m_{i \triangleright j}(x_{\underline{ij}})\}$ converge, i.e. they reach a fixed point satisfying (12), we substitute them for $\{m_{k \triangleright i}(x_{\underline{ki}})\}$ in (13) and obtain approximate marginals $\{b_i(x_{\underline{i}})\}$ of $p(x)$.

4.2 Special cases

As cluster graphs are rather arbitrarily defined, we should not expect accurate approximations without appropriate restriction on $\mathbf{U}$. Proposition 4.1 is obvious from the derivation of Cluster BP.

Proposition 4.1
Suppose $\mathbf{U}$ *is a set of Kikuchi clusters and there is a cluster graph* $\mathcal{G}_U$ *for* $\mathbf{U}$. *Cluster BP running on* $\mathcal{G}_U$ *computes a stationary point of the Kikuchi approximation.*

We now look into cases where $\mathcal{G}_U$ is a tree. In this case observe that $b_i(x_{\underline{i}})$ is uniquely determined by (12) and (13) because starting from (13) and replacing the l.h.s. $m_{i \triangleright j}(x_{\underline{ij}})$ of (12) with the r.h.s., we obtain, in time linear in the size of the tree, $b_i(x_{\underline{i}})$ expressed in terms of the sum and product of adjusted potentials.

Proposition 4.2
Suppose $\mathbf{U}$ *is closed under intersection*[13] *and has a cluster graph* $\mathcal{G}_U$ *which is a tree. Then marginal distributions computed by Cluster BP using* $\mathcal{G}_U$ *are exact.*

(Proof) First of all $\mathcal{G}_U$ satisfies the so-called junction tree property.[14] This is because if z is in some cluster $\alpha \in \mathbf{U}$, there is the smallest cluster $\beta \in \mathbf{U}$ containing z as $\mathbf{U}$ is closed under intersection. The tree condition ensures that vertices and edges whose label include β form a tree. As a result the junction tree property is satisfied.

Second, $\mathcal{G}_U$ must be regular because otherwise there are vertices labeled α and β respectively, and connected by an edge e whose label is γ such that $\gamma \subset \alpha \cap \beta$. As $\alpha \cap \beta \in \mathbf{U}$, there is a path comprised of vertices and edges whose label include $\alpha \cap \beta$. The path does not contain the edge e however, which means that there is a loop containing γ, contradicting the tree condition.

Thirdly, we can deduce using $\sum_{\alpha : \alpha \in \mathbf{U}, \alpha \supseteq \beta} a_\alpha = 1$ that

$$\begin{aligned}
\prod_{\alpha \in \mathbf{P}} \psi_\alpha(x_\alpha) &= \prod_{\alpha \in \mathbf{U}} (\phi_\alpha(x_\alpha))^{a_\alpha} \\
&= \prod_{\alpha \in \mathbf{U}} \frac{\phi_\alpha(x_\alpha)^{V(\alpha)}}{\phi_\alpha(x_\alpha)^{E(\alpha)}} \\
&= \frac{\prod_{v \in V} \phi_{\underline{v}}(x_{\underline{v}})}{\prod_{e \in E} \phi_{\underline{e}}(x_{\underline{e}})} \\
&= \prod_{v \in V} \phi'_{\underline{v}}(x_{\underline{v}})
\end{aligned}$$

[13] If $\alpha, \beta \in \mathbf{U}$, then $\alpha \cap \beta \in \mathbf{U}$ if $\alpha \cap \beta \neq \emptyset$. Note that this condition is less restrictive than the condition that $\mathbf{U}$ is a set of Kikuchi clusters. For example $\mathbf{U} = \{12, 1, 2\}$ is not a set of Kikuchi clusters but closed under intersection.

[14] If a variable z occurs in the labels of vertices v and v', there is a path connecting v and v' such that every vertex and edge on the path contains z.

where $\phi_\alpha(x_\alpha)$ is a potential for the cluster α, $\phi'_v(x_{\underline{v}})$ an adjusted potential for v, and V is the set of vertices in $\mathcal{G}_U$.

From $\prod_{\alpha\in\mathbf{P}}\psi_\alpha(x_\alpha) = \prod_{v\in V}\phi'_{\underline{v}}(x_{\underline{v}})$ and the fact that $\mathcal{G}_U$ is a regular cluster tree, it is straightforward to prove that marginal distributions $\{b_i(x_{\underline{i}})\}$ defined by (12) and (13) are exact (details omitted). Q.E.D.

We state some remarks below. When $\mathcal{G}_U$ is a junction tree (considering separators as clusters labeling edges), Cluster BP is equivalent to the celebrated junction tree algorithm [7]. Proposition 4.2 covers however also cases where some clusters are subsets of others thereby $\mathcal{G}_U$ being not a junction tree.

Aji and McEliece introduced junction graphs and the Bethe-Kikuchi approximation to the variational free energy [2]. They showed how an iterative message passing algorithm running on junction graphs called the GDL (generalized distribution law) [1] which is identical to Cluster BP, is derived from the Bethe-Kikuchi approximation. Their introduction of the Bethe-Kikuchi approximation however was done intuitively and it remained unclear under what condition the GDL algorithm computes the Kikuchi approximation. Let $\mathcal{G}_J$ be a junction graph for $\mathbf{U}$. In $\mathcal{G}_J$, edges are labeled by clusters not necessarily from $\mathbf{U}$ and the tree condition is imposed on all but only singleton clusters regardless of whether $\mathbf{U}$ includes singleton clusters or not. As a result there are cases where the Bethe-Kikuchi approximation defined by $\mathcal{G}_J$ does not coincide with the Kikuchi approximation. By imposing the tree condition on every cluster in $\mathbf{U}$, we assure that when $\mathbf{U}$ is a set of Kikuchi clusters, any cluster graph for $\mathbf{U}$ defines the (same) Kikuchi approximation thereby Cluster BP computing it.

Lastly, suppose $\mathbf{U}$ consists of potential clusters and singletons, i.e. $\mathbf{U} = \mathbf{P} \cup \{\{1\},\ldots,\{n\}\}$. If $\mathcal{G}_U$ is a bipartite graph with $\mathbf{P}$ on one side and $\{1\},\ldots,\{n\}$ on the other side, connecting $\{k\}$ with $\alpha \in \mathbf{P}$ containing k, Cluster BP is reduced to the sum-product algorithm [6].

5 Cluster CCCP

Loopy BP (resp. GBP) can compute stationary points of Bethe approximations (resp. Kikuchi approximations) but has the risk of non-convergence. Yuille recently proposed a new approach to the minimization of Bethe and Kikuchi approximations based on the convex-concave decomposition of a target function [21]. His algorithm is called the *CCCP double-loop algorithm* and computes local minima of Bethe and Kikuchi approximations.

The basic idea is as follows. Suppose a multi-variate function $f(x)$ is represented as a sum of a convex function and a concave function. For convenience we regard this sum, *convex-concave decomposition*, as a difference of convex functions $g(x)$ and $h(x)$ like $f(x) = g(x) - h(x)$ (note that $-h(x)$ is concave). By convexity, we have $(1-r)g(x) + rg(x+d) \geq g(x+rd)$ for any real vector d and any real number r $(0 < r < 1)$. By taking the limit $r \to 0$, we know $g(x+d) - g(x) \geq (d \cdot \bigtriangledown g(x))$ where $\cdot$ denotes an inner product. Hence we have

$$\begin{aligned} g(x_n) &\geq g(x_{n+1}) + ((x_n - x_{n+1}) \cdot \bigtriangledown g(x_{n+1})) \\ h(x_{n+1}) &\geq h(x_n) + ((x_{n+1} - x_n) \cdot \bigtriangledown h(x_n)) \end{aligned}$$

from which we obtain a recurrence formula generating a series $\{x_n\}$ that locally minimizes $f(x)$.

$$f(x_n) = g(x_n) - h(x_n) \;\geq\; f(x_{n+1}) = g(x_{n+1}) - h(x_{n+1})$$
$$\text{if } \bigtriangledown\, g(x_{n+1}) = \bigtriangledown h(x_n)$$

When the problem is a minimization of $f(x)$ under linear constrains $(a_i \cdot x) = b_i$ $(1 \leq i \leq h)$, a similar treatment is possible and we can derive a series that leads to a solution.

$$f(x_n) \geq f(x_{n+1}) \text{ if}$$
$$\bigtriangledown g(x_{n+1}) = \bigtriangledown h(x_n) + \textstyle\sum_{i=1}^{h} c_i a_i \text{ for some } c_i \; (1 \leq i \leq h)$$

Figure 5: Minimizing $f(x)$ under linear constraints

Since Bethe approximations (resp. Kikuchi approximations) are representable as a difference of two convex functions (of the form $x \ln \left(\frac{x}{a}\right)$ and $x \ln x$), Yuille applied the above method and derived an iterative algorithm that is guaranteed to converge and guaranteed to compute local minima of Bethe approximations (resp. Kikuchi approximations) [21].

We here follow [21] and apply his decomposition method to minimize the generalized Kikuchi approximation $\tilde{F}_K(\{b_\alpha\})$ in (7) or equivalently $\tilde{F}'_K$ in (10) and derive a new algorithm called *Cluster CCCP*. However, we carefully choose a decomposition of $\tilde{F}_K(\{b_\alpha\})$ so that the resulting Cluster CCCP becomes a convergent generalization of Cluster BP.

We assume as before a distribution $p(x) = \kappa \prod_{\alpha \in \mathbf{P}} \psi_\alpha(x_\alpha)$ and a cluster graph $\mathcal{G}_U$ for a set of clusters $\mathbf{U}$ for $\mathbf{P}$ and use V for the set of vertices and E for the set of edges in $\mathcal{G}_U$. When $i \in V$ is a vertex, $\underline{i}$ is a cluster labeling i. Also when $(ij) \in E$ is an edge connecting i and j, $\underline{ij}$ is a cluster labeling the edge (ij). $\phi'_i(x_{\underline{i}})$ is an adjusted potential of i. We rewrite $\tilde{F}'_K$ in (10) as follows.

$$\tilde{F}'_K = \sum_{i \in V} \sum_{x_{\underline{i}}} b_i(x_{\underline{i}}) \ln \left(\frac{b_i(x_{\underline{i}})}{\phi'_i(x_{\underline{i}})} \right) - \sum_{(ij) \in E} \sum_{x_{\underline{ij}}} b_{(ij)}(x_{\underline{ij}}) \ln b_{(ij)}(x_{\underline{ij}}) \quad (14)$$

$$= \left\{ \sum_{i \in V} \sum_{x_{\underline{i}}} b_i(x_{\underline{i}}) \ln \left(\frac{b_i(x_{\underline{i}})}{\phi'_i(x_{\underline{i}})} \right) + \sum_{(ij) \in E} \sum_{x_{\underline{ij}}} b_{(ij)}(x_{\underline{ij}}) \ln b_{(ij)}(x_{\underline{ij}}) \right\}$$
$$-2 \sum_{(ij) \in E} \sum_{x_{\underline{ij}}} b_{(ij)}(x_{\underline{ij}}) \ln b_{(ij)}(x_{\underline{ij}}) \quad (15)$$

We apply the convex-concave decomposition to (15) and derive the Cluster CCCP algorithm that locally minimizes the generalized Kikuchi approximation. The result is displayed in Figure 6.[15]

The Cluster CCCP algorithm is executed as follows. First we initialize $\{b_i(x_{\underline{i}}),\, \lambda_i,\, \lambda^j_{(ij)}(x_{\underline{ij}})\}$ to 1 and sort edges $\{(ij) \mid (ij) \in E\}$ in some order. We then enter the inner loop (to satisfy linear constraints). In the loop, we take an edge (ij) serially and update Lagrange multipliers $\lambda^j_{(ij)}(x_{\underline{ij}})$, $\lambda^i_{(ij)}(x_{\underline{ij}})$ and λ_i respectively using (16), (17) and (18) *in this order*. After updates for all $\{(ij) \mid (ij) \in E\}$ are finished, we enter the next cycle of

[15] It is possible to apply the convex-concave decomposition to (14) but the resulting algorithm is messier than Cluster CCCP.

$$e^{2\lambda^{j}_{(ij)}(x_{\underline{ij}})} = \frac{\sum_{x_{\underline{i}\setminus\underline{ij}}} \phi'_i(x_{\underline{i}})\, e^{-\sum_{h\in N(i)\setminus j} \lambda^{h}_{(ih)}(x_{\underline{ih}})}}{\{b_{(ij)}(x_{\underline{ij}})\}^2\, e^{\lambda^{i}_{(ij)}(x_{\underline{ij}})}} \left(\frac{e^{-(1+\lambda_i)}}{e}\right) \quad (16)$$

$$e^{2\lambda^{i}_{(ij)}(x_{\underline{ij}})} = \frac{\sum_{x_{\underline{j}\setminus\underline{ij}}} \phi'_j(x_{\underline{j}})\, e^{-\sum_{h\in N(j)\setminus i} \lambda^{h}_{(jh)}(x_{\underline{jh}})}}{\{b_{(ij)}(x_{\underline{ij}})\}^2\, e^{\lambda^{j}_{(ij)}(x_{\underline{ij}})}} \left(\frac{e^{-(1+\lambda_j)}}{e}\right) \quad (17)$$

$$e^{(1+\lambda_i)} = \sum_{x_{\underline{i}}} \phi'_i(x_{\underline{i}}) e^{-\sum_{h\in N(i)} \lambda^{h}_{(ih)}(x_{\underline{ih}})} \quad (18)$$

(Inner loop)

$$b'_{(ij)}(x_{\underline{ij}}) = \{b_{(ij)}(x_{\underline{ij}})\}^2\, e^{\lambda^{i}_{(ij)}(x_{\underline{ij}})+\lambda^{j}_{(ij)}(x_{\underline{ij}})+1} \quad (19)$$

(Outer loop)

Figure 6: Cluster CCCP

the inner loop. We repeat the inner loop until all values of $\{\lambda_i, \lambda^{j}_{(ij)}(x_{\underline{ij}})\}$ converge (the convergence is assured). After the convergence of the inner loop, we execute the outer loop (19) once to update $b_{(ij)}(x_{\underline{ij}})$ to $b'_{(ij)}(x_{\underline{ij}})$. At each update of the outer loop, the variational free energy decreases.

We iterate this inner loop–outer loop process enough times until the outer loop converges. After the convergence of the outer loop (again the convergence is assured), we calculate

$$b'_i(x_{\underline{i}}) = e^{-(1+\lambda_i)} \phi'_i(x_{\underline{i}})\, e^{-\sum_{h\in N(i)} \lambda^{h}_{(ih)}(x_{\underline{ih}})} \quad (20)$$

to obtain approximate marginals $b'_i(x_{\underline{i}})$ for vertex i.

We check the relationship between Cluster CCCP and Cluster BP. When the outer loop converges, we have $b'_{(ij)}(x_{\underline{ij}}) = b_{(ij)}(x_{\underline{ij}})$ from which it follows that

$$b_{(ij)}(x_{\underline{ij}}) = e^{-\lambda^{i}_{(ij)}(x_{\underline{ij}})-\lambda^{j}_{(ij)}(x_{\underline{ij}})-1}.$$

By substituting this into (16) and (17), and replacing $e^{-\lambda^{i}_{(ij)}(x_{\underline{ij}})}$ with $m_{i\triangleright j}(x_{\underline{ij}})$, we reproduce Cluster BP in Figure 4. One may say therefore that Cluster CCCP is a convergent version of Cluster BP which takes advantage of information about neighbor points of a stationary point.

6 Related work and discussion

As we pointed out before, the GDL algorithm for Bethe-Kikuchi approximations proposed by Aji and McEliece [2] does not necessarily compute Kikuchi approximations. To ensure the computation of the latter, we introduced cluster graphs that are similar to junction graphs but more restrictive in that all clusters must satisfy the tree condition.[16] We derived the Cluster

[16] Note that this does not mean cluster graphs are a subclass of junction graphs. This is because a cluster set $\mathbf{U}$ may lack some or all singleton clusters. For

BP algorithm from the Kikuchi approximation with the help of the tree condition and proved that when run on a cluster graph (for the set of Kikuchi clusters), it computes a stationary point of the Kikuchi approximation. We further proposed Cluster CCCP as a convergent generalization of Cluster BP though it is not in the style of message passing.

(Generalized) Kikuchi approximations which Cluster BP and Cluster CCCP can compute are however a proper subset of the ones computed by a more general method such as GBP [19, 20]. By Proposition 3.2 we know that Cluster BP does not work for Kikuchi approximations with positive overcounting numbers for some nonmaximal clusters. For instance look at again the left graph (B) in Figure 7 (= the right graph (B) in Figure 1) and $\mathbf{U}_{\mathrm{B}} = \{124, 234, 134, 14, 24, 34, 4\}$. $\mathbf{U}_{\mathrm{B}}$ is the set of Kikuchi clusters for potential clusters $\{124, 234, 134\}$ corresponding to the distribution $p(x_{1234}) = \kappa\, \psi_1(x_{124})\psi_2(x_{234})\psi_3(x_{134})$. Since the overcounting number of $\{4\}$, the minimum cluster in $\mathbf{U}_{\mathrm{B}}$, is 1, there is no cluster graph for $\mathbf{U}_{\mathrm{B}}$. So Cluster BP is not applicable to compute the Kikuchi approximation for $p(x_{1234})$.

Nevertheless it is still possible to compute generalized Kikuchi approximations by Cluster BP. One way is just deleting from the cluster set $\mathbf{U}$ the culprits violating the tree condition. We for example delete $\{4\}$ from $\mathbf{U}_{\mathrm{B}}$. Then the same graph, (B), is qualified as a cluster graph for the deleted set $\mathbf{U}'_{\mathrm{B}} = \{124, 234, 134, 14, 24, 34\}$ and hence Cluster BP would compute a generalized Kikuchi approximation defined by $\mathbf{U}'_{\mathrm{B}}$. Another one, more reasonable one, is to modify $\mathbf{U}$ a little and at the same time modify the original graph accordingly so that the modified $\mathbf{U}$ has a cluster graph. In our case we choose one of the clusters, say 34 from $\mathbf{U}_{\mathrm{B}}$, and replace it with 3. We also transform the original graph (B) into another graph (B') so that (B') is a cluster graph for the modified set $\mathbf{U}_{\mathrm{B}'} = \{124, 234, 134, 14, 24, 3, 4\}$. See the right graph (B') in Figure 7.[17]

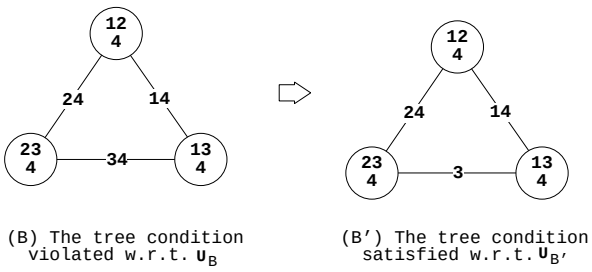

Figure 7: Turning into a cluster graph for $\mathbf{U}_{\mathrm{B}'}$

In this case Cluster BP computes a generalized Kikuchi approximation defined by $\mathbf{U}_{\mathrm{B}'}$ using addition and multiplication over six messages. GBP on the other hand can compute a more accurate approximation, the Kikuchi approximation defined by $\mathbf{U}_{\mathrm{B}}$, but requires addition, multiplication and division over nine messages some of which are listed below.

---

example, although the right graph (B) in Figure 1 is a cluster graph for $\mathbf{U}'_{\mathrm{B}} = \{124, 234, 134, 14, 24, 34\}$, it is not a junction graph for $\mathbf{U}'_{\mathrm{B}}$.

[17] By this transformation we lose the assurance of the consistency of marginals between b_{234} and b_{134}. We cannot expect $\sum_{x_2} b_{234}(x_{234}) = \sum_{x_1} b_{134}(x_{134})$, because clusters 234 and 134 have no path conveying information about the correlation of x_3 and x_4.

$$
\begin{aligned}
m_{124\to 14}(x_{14}) &= \kappa_1 \frac{\sum_{x_2} \psi(x_{124}) m_{234\to 24}(x_{24})}{m_{24\to 4}(x_4)} \\
m_{234\to 24}(x_{24}) &= \kappa_2 \frac{\sum_{x_3} \psi(x_{234}) m_{134\to 34}(x_{34})}{m_{34\to 4}(x_4)} \\
m_{134\to 34}(x_{34}) &= \kappa_3 \frac{\sum_{x_1} \psi(x_{134}) m_{124\to 14}(x_{14})}{m_{14\to 4}(x_4)} \\
&\dots
\end{aligned}
$$

As indicated by the above example, it seems that there is a trade-off between the complexity of the algorithm and the accuracy of the approximation. We leave it as a future work to investigate how to balance computational burden against approximation accuracy.[18]

Pakzad and Anantharam generalized the Kikuchi approximation by allowing Kikuchi clusters to be just a partially ordered set represented by a Hasse diagram [11]. They proposed to vary the degree of approximation by a number k in such a way that for every cluster $|\alpha| \leq k$, $\sum_{\beta:\beta\subseteq\alpha} a_\beta = 1$.

7 Conclusion

We have proposed cluster graphs to compute generalized Kikuchi approximations by Cluster BP using multi-variate messages. It has been shown that Cluster BP computes when converged the subclass of Kikuchi approximations for which cluster graphs exist. We also have proposed Cluster CCCP which is more complicated than Cluster BP but is assured to compute local minima of generalized Kikuchi approximations.

References

[1] A.M. Aji and R.J. McEliece. The generalized distributive law. *IEEE Trans. Inform. Theory*, 46(2):325–343, 2000.

[2] A.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*, 2001.

[3] E. Castillo, J. M. Gutierrez, and A. S. Hadi. *Expert Systems and Probabilistic Network Models*. Springer-Verlag, 1997.

[4] F. V. Jensen. *An Introduction to Bayesian Networks*. UCL Press, 1996.

[5] F. V. Jensen. *Bayesian Networks and Decision Graphs*. Springer, 2001.

[6] F.R. Kschischang, B.J. Frey, and H.A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transaction on Information Theory*, 47(2):498–519, 2001.

[18] In a more recent publication Yedidia et al. proposed the region graph method [20]. Region graphs are a generalization of junction graphs with additional structures, a partial ordering over regions and a distinction between variable nodes and factor nodes like factor graphs for the sum-product algorithm [6]. In the Appendix E, they derive a two-way message passing algorithm for computing a stationary point of the region free energy, and suggest a sufficient condition for the two-way algorithm to be reduced to the standard BP, which seems equivalent to the tree condition though there is a subtle difference due to the fact that cluster graphs admit of a multiple occurrence of the same label.

[7] S.L. Lauritzen and D.J. Spiegelhalter. Local computations with probabilities on graphical structures and their applications to expert systems. *Journal of the Royal Statistical Society, B*, 50:157–224, 1988.

[8] D.J.C. MacKay, R.J. McEliece, and J.-F. Cheng. Turbo decoding as an instance of pearl's "belief propagation" algorithm. *IEEE Journal of Selected Areas of Communication*, pages 140–152, 1998.

[9] T. Morita. Formal structure of the cluster variation method. *Progress of Theoretical Physics Supplement*, 115:27–39, 1994.

[10] Kevin Murphy, Yair Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Uncertainty in Artificial Intelligence: Proceedings of the Fifteenth Conference (UAI-1999)*, pages 467–475, San Francisco, CA, 1999. Morgan Kaufmann Publishers.

[11] P. Pakzad and V. Anantharam. Belief Propagation and Statistical Physics. In *Proceedings of 2002 Conference on Information Sciences and Systems*, 2002.

[12] J. Pearl. Fusion, propagation and structuring in belief networks. *Artificial Intelligence*, 29:357–369, 1986.

[13] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.

[14] T. Sato. A statistical learning method for logic programs with distribution semantics. In *Proceedings of the 12th International Conference on Logic Programming (ICLP'95)*, pages 715–729, 1995.

[15] T. Sato, S. Abe, Y. Kameya, and K. Shirai. A separate-and-learn approach to EM learning of PCFGs. In *Proceedings of the 6th Natural Language Processing Pacific Rim Symposium (NLRPS2001)*, pages 255–262, 2001.

[16] T. Sato and Y. Kameya. PRISM: a language for symbolic-statistical modeling. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI'97)*, pages 1330–1335, 1997.

[17] T. Sato and Y. Kameya. Parameter learning of logic programs for symbolic-statistical modeling. *Journal of Artificial Intelligence Research*, 15:391–454, 2001.

[18] Y. Weiss and W.T. Freeman. Correctness of belief propagation in gaussian graphical models of arbitrary topology. *Neural Computation*, 13(10):2173–2200, 2001.

[19] J.S. Yedidia, W.T. Freeman, and Y. Weiss. Bethe free energy, kikuchi approximations and belief propagation algorithms. Technical Report MERL TR2001-16, Mitsubishi Electric Research Laboratories, 2001.

[20] J.S. Yedidia, W.T. Freeman, and Y. Weiss. Constructing free energy approximations and generalized belief propagation algorithms. Technical Report MERL TR2002-35, Mitsubishi Electric Research Laboratories, 2002.

[21] A.L. Yuille. CCCP algorithms to minimize the bethe and kikuchi free energies: Convergent alternatives to belief propagation. In *Neural Computation 14 (7)*, pages 1691–1722. MIT Press, 2002.