Convergent tree-reweighted message passing for energy minimization

Vladimir Kolmogorov
Microsoft Research
Cambridge, UK
vnk@microsoft.com

Abstract

Tree-reweighted max-product message passing (TRW) is an algorithm for energy minimization introduced recently by Wainwright et al. [7]. It shares some similarities with Pearl's loopy belief propagation. TRW was inspired by a problem of maximizing a lower bound on the energy. However, the algorithm is not guaranteed to increase this bound - it may actually go down. In addition, TRW does not always converge. We develop a modification of this algorithm which we call *sequential tree-reweighted message passing*. Its main property is that the bound is guaranteed not to decrease. We also give a *weak tree agreement* condition which characterizes local maxima of the bound with respect to TRW algorithms. We prove that our algorithm has a limit point that achieves weak tree agreement. Experimental results demonstrate that on certain synthetic and real problems our algorithm outperforms both the ordinary belief propagation and tree-reweighted algorithm [7].

1 Introduction

Sum-product algorithms Pearl's loopy belief propagation ("sum-product BP") is a popular algorithm for inference in Bayesian networks. If the network is a tree, then it converges in a finite number of iterations, and the solution gives exact marginals. However, if the network contains loops then convergence is not guaranteed.

Fixed points of BP have been shown to correspond to extrema of the so-called Bethe free energy [12]. This motivated algorithms for direct minimization of the Bethe free energy, such as CCCP [13] and UPS [5]. They are guaranteed to converge; however, the computation cost is often higher than the cost of BP.

Several researchers proposed alternatives to the Bethe free energy [6, 9, 11, 3]. Functionals used in [6, 9] are convex upper bounds on the log partition function, and functionals in [3] are convex upper bounds on the Bethe free energy. In order to minimize these functionals, belief propagation algorithm was modified in such a way that its fixed points correspond to extrema of the functional.

The algorithm proposed in [9] is called *tree-reweighted message passing* (TRW). Interestingly, its fixed point achieves the *global* minimum of the upper bound. Unfortunately, as in the case of ordinary BP, convergence is not guaranteed.

Max-product algorithms Closely related to sum-product are "max-product" (or "min-sum") BP algorithms. Their goal is to find a configuration with the maximum a posteriori (MAP) probability, or a configuration with the smallest energy. Generally, max-product algorithms can be obtained from sum-product versions in the zero temperature limit. There are caveats, however; for example, it is not known whether fixed points of max-product BP correspond to extrema of some functional.

In this paper we focus on the max-product version of tree-reweighted algorithm [7]. In fact, two different TRW algorithms are developed in [7]. They are inspired by the problem of maximizing a concave lower bound on the energy. These algorithms have the following property: if their fixed point satisfies a certain condition ("tree agreement") then it is guaranteed to give a MAP solution (i.e. a global minimum of the energy).

However, TRW algorithms in [7] cannot be viewed as algorithms for direct maximization of the bound. Indeed, in our experiments we observed that sometimes they decrease it. Also, the algorithms do not always converge; when it happens, the value of the bound of-

ten goes into a loop.

Our main contribution is as follows: we show how to modify TRW algorithms so that the value of the bound is guaranteed not to decrease. Thus, we are guaranteed to find at least a "local" maximum of the bound. The word "local" is in quotes since for concave functions all local maxima are global, if the standard metric space topology is used. Here we use a weaker topology: our maxima are local with respect to the TRW algorithms. We formulate the *weak tree agreement* condition (WTA) which gives a precise characterization of such maxima. We prove that our algorithm has a subsequence converging to a vector satisfying WTA.

An interesting question is whether WTA always gives a global maximum of the bound. We show that this is not the case by providing a counterexample. This is a difference between sum-product and max-product cases.

TRW algorithms require some choice of trees covering the graph. If the trees have a special structure (namely, chains which are *monotonic* with respect to some ordering on the graph) then our algorithm reduces to the TRW message-passing algorithm of Wainwright et al. [7], but with a significant distinction: we update messages in a specific *sequential* order rather than in parallel. In the context of ordinary BP it is a well-known experimental fact that sequential updates are superior to parallel updates, although convergence is still not guaranteed. We give a theoretical justification of sequential updates in the case of tree-reweighted algorithms.

Our experimental results include both synthetic and real problems. In particular, we consider an energy function arising in the stereo matching problem [1]. We demonstrate that our algorithm outperforms both the ordinary BP and the TRW algorithms of Wainwright et al. [7]. Moreover, we obtain a slightly lower energy than the expansion move method [1], which is generally considered to be the most accurate minimization technique for such energy functions.

Outline The paper is organized as follows. In section 2 we introduce our notation and review some results from [7], in particular the lower bound on the energy function via convex combination of trees and duality result. Our new tree-reweighted algorithm and its analysis are given in section 3. Experimental results are described in section 4. Finally, we give conclusions in section 5.

2 Notation and background

In this paper we closely follow the notation used in [7]. However, instead of maximizing posterior probability we minimize an energy function. Therefore, we replace "min" with "max", "inf" with "sup" and vice versa.

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph with the set of vertices $\mathcal{V}$ and the set of edges $\mathcal{E}$. For each $s \in \mathcal{V}$, let x_s be a variable taking values in some discrete space $\mathcal{X}_s$. By concatenating the variables at each node, we obtain a vector $\mathbf{x}$ with $n = |\mathcal{V}|$ elements. This vector takes values in the space $\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times \ldots \times \mathcal{X}_n$. Unless noted otherwise, symbols s and t will denote nodes in $\mathcal{V}$, (s,t) - edge in $\mathcal{E}$, j and k - variables in $\mathcal{X}_s$ and $\mathcal{X}_t$, respectively.

A *potential function* is a mapping $\phi : \mathcal{X} \to \mathbb{R}$. We can also consider a family of potential functions $\{\phi_\alpha \mid \alpha \in \mathcal{I}\}$ where $\mathcal{I}$ is some index set. This family defines a vector-valued mapping $\boldsymbol{\phi} : \mathcal{X} \to \mathbb{R}^d$. Associated with $\boldsymbol{\phi}$ is a real-valued vector $\theta = \{\theta_\alpha \mid \alpha \in \mathcal{I}\}$, which we call an *energy parameter vector*[1]. This vector defines an energy function $E(\cdot \mid \theta) : \mathcal{X} \to \mathbb{R}$ as follows:

$$E(\mathbf{x} \mid \theta) = \langle \theta, \boldsymbol{\phi}(\mathbf{x}) \rangle = \sum_{\alpha \in \mathcal{I}} \theta_\alpha \phi_\alpha(\mathbf{x}) \qquad (1)$$

where $\langle \cdot, \cdot \rangle$ is the ordinary Euclidean product in $\mathbb{R}^d$.

We will use the collection of potential functions called the *canonical overcomplete representation* [8, 10]. It is defined as follows:

$$\{\delta_j(x_s)\} \cup \{\delta_j(x_s)\delta_k(x_t)\} \cup \{\phi_{const}(\mathbf{x})\}$$

where $\delta_j(x_s)$ is an *indicator function* - it is equal to one if $x_s = j$, and zero otherwise. Function $\phi_{const}(\mathbf{x})$ equals to 1 for all arguments. The index set for this representation is

$$\mathcal{I} = \{(s;j)\} \cup \{(st;jk)\} \cup \{const\}$$

Note that $(st;jk) \equiv (ts;kj)$, so $\theta_{st;jk}$ and $\theta_{ts;kj}$ are the same element.

Sometimes it will be convenient to denote elements $\theta_{s;j}$ and $\theta_{st;jk}$ as $\theta_s(j)$ and $\theta_{st}(j,k)$, respectively. We will also use notation θ_s to denote a vector of size $|\mathcal{X}_s|$ and θ_{st} to denote a vector of size $|\mathcal{X}_s \times \mathcal{X}_t|$.

Note that energy function 1 corresponding to this representation can be written as a sum of unary terms corresponding to nodes in $\mathcal{V}$, and pairwise terms corresponding to edges in $\mathcal{E}$:

$$E(\mathbf{x} \mid \theta) = \theta_{const} + \sum_{s \in \mathcal{V}} \theta_s(x_s) + \sum_{(s,t) \in \mathcal{E}} \theta_{st}(x_s, x_t)$$

Let us introduce another notation which we will use extensively throughout the paper. Functions $\Phi, \Phi_{s;j}, \Phi_{st;jk} : \mathbb{R}^d \to \mathbb{R}$ give information about the

[1] In [8, 10] parameter θ is called an *exponential parameter vector*. We use a different name to emphasize the fact that our θ is the negative of θ used in [8, 10].

minimum values of the energy under different constraints:

$$\begin{array}{lll} \Phi(\theta) & = \min_{\mathbf{x}\in\mathcal{X}} & E(\mathbf{x} \mid \theta) \\ \Phi_{s;j}(\theta) & = \min_{\mathbf{x}\in\mathcal{X}, x_s=j} & E(\mathbf{x} \mid \theta) \\ \Phi_{st;jk}(\theta) & = \min_{\mathbf{x}\in\mathcal{X}, x_s=j, x_t=k} & E(\mathbf{x} \mid \theta) \end{array}$$

Values $\Phi_{s;j}(\theta)$ and $\Phi_{st;jk}(\theta)$ are called *min-marginals* for node s and edge (s,t), respectively.

2.1 Reparameterization and max-product belief propagation

If two parameter vectors θ and θ' define the same energy function (i.e. $E(\mathbf{x} \mid \theta') = E(\mathbf{x} \mid \theta)$ for all $\mathbf{x} \in \mathcal{X}$) then θ' is called a *reparameterization* of θ [8, 10]. We will write this as $\theta' \equiv \theta$. Note that this condition does not necessarily imply that $\theta' = \theta$ since there are various linear relations among potential functions ϕ_α.

Reparameterization provides an alternative tool for the analysis of belief propagation algorithm. Recall that a basic operation of BP is *passing a message* from node s to node t. As shown in [8, 10], this operation can be implemented without any messages: it is equivalent to a certain reparameterization of vectors θ_{st} and θ_t for edge (s,t) and node t, respectively. We will say that θ is in a *normal form* if it is a fixed point of BP.

If graph $\mathcal{G}$ is a tree, then values $\theta_{s;j}$ and $\theta_{st;jk}$ for vector θ in a normal form have a particularly simple interpretation [10] - they correspond to min-marginals (up to a constant):

$$\begin{array}{lll} \Phi_{s;j}(\theta) & = & \theta_{s;j} + const_s \\ \Phi_{st;jk}(\theta) & = & \theta_{s;j} + \theta_{st;jk} + \theta_{t;k} + const_{st} \end{array} \quad (2)$$

where $const_s$ and $const_{st}$ are constants independent of j and k.

2.2 Lower bound on the energy function

In this section we review the bound proposed in [7]. In order to describe it, we need to introduce some notation.

Let $\mathcal{T}$ be a collection of trees in graph $\mathcal{G}$ and $\rho^T,\ T \in \mathcal{T}$ be some distribution on $\mathcal{T}$. Throughout the paper we assume that each tree has a non-zero probability and each edge in $\mathcal{E}$ is covered by at least one tree.

For a given tree $T = (\mathcal{V}^T, \mathcal{E}^T)$ we define a set

$$\mathcal{I}^T = \{(s;j) \mid s \in \mathcal{V}^T\} \cup \{(st;jk) \mid (s,t) \in \mathcal{E}^T\} \cup \{const\}$$

corresponding to those indexes associated with vertices and edges in the tree.

To each tree $T \in \mathcal{T}$, we associate an energy parameter θ^T that must respect the structure of T. More precisely, the parameter θ^T must belong to the following linear constraint set:

$$\mathcal{A}^T = \{\theta^T \in \mathbb{R}^d \mid \theta^T_\alpha = 0 \ \ \forall\, \alpha \in \mathcal{I} \backslash \mathcal{I}^T\}$$

By concatenating all of the tree vectors, we form a larger vector $\boldsymbol{\theta} = \{\theta^T \mid T \in \mathcal{T}\}$, which is an element of $\mathbb{R}^{d \times |\mathcal{T}|}$. Vector $\boldsymbol{\theta}$ must belong to the constraint set

$$\mathcal{A} = \{\boldsymbol{\theta} \in \mathbb{R}^{d \times |\mathcal{T}|} \mid \theta^T \in \mathcal{A}^T \text{ for all } T \in \mathcal{T}\}$$

Consider function $\Phi_\rho : \mathcal{A} \to \mathbb{R}$ defined as follows:

$$\Phi_\rho(\boldsymbol{\theta}) = \sum_T \rho^T \Phi(\theta^T) = \sum_T \rho^T \min_{\mathbf{x}\in\mathcal{X}} \langle \theta^T, \boldsymbol{\phi}(\mathbf{x}) \rangle$$

[7] shows that if $\sum_T \rho^T \theta^T = \bar{\theta}$ then $\Phi_\rho(\boldsymbol{\theta})$ is a lower bound on the optimal value of the energy for vector $\bar{\theta}$ (this follows from Jensen's inequality). To get the tightest bound we can consider the following maximization problem:

$$\max_{\boldsymbol{\theta}\in\mathcal{A}, \sum_T \rho^T \theta^T = \bar{\theta}} \Phi_\rho(\boldsymbol{\theta}) \quad (3)$$

Interestingly, the optimal value of this problem does not depend on the choice of trees [7]. A proof of this fact can be summarized as follows. Φ_ρ is a concave function of $\boldsymbol{\theta}$; moreover, the constraints of problem 3 are linear in $\boldsymbol{\theta}$. Thus, we can consider its Lagrangian dual. This dual problem turns out to be a certain linear programming relaxation of the original minimization problem, and does not involve any trees[2].

3 New tree-reweighted message passing algorithm

Maximization problem 3 inspired two algorithms in [7] - tree-reweighted message passing with edge based updates (TRW-E) and with tree based updates (TRW-T). However, neither algorithm maintains constraint $\sum_T \rho^T \theta^T = \bar{\theta}$ of problem 3. Indeed, they perform reparameterizations of the original parameter vector, so this equality may become violated[3]. Let us replace it with the constraint $\sum_T \rho^T \theta^T \equiv \bar{\theta}$. Thus, we are now interested in the following maximization problem:

$$\max_{\boldsymbol{\theta}\in\mathcal{A}, \sum_T \rho^T \theta^T \equiv \bar{\theta}} \Phi_\rho(\boldsymbol{\theta}) \quad (4)$$

The following lemma justifies this formulation.

Lemma 3.1. *The optimal value of problem 4 equals to the optimal value of problem 3.*

[2] [7] formulated the duality theorem for the case when trees in $\mathcal{T}$ are *spanning*. However, their proof never uses this assumption. In this paper we do not assume that trees are spanning.

[3] Note that lemmas 5 and 11 in [7] seem to contain a mistake. Lemma 11, for example, says that TRW-T algorithm maintains the property $\sum_T \rho^T \theta^T = \bar{\theta}$. However, the proof does not take into account reparameterization step.

Proof. See [4]. The proof involves showing that any reparameterization can be expressed via messages. It is omitted due to space limitations. □

As shown in [7], TRW-E and TRW-T algorithms maintain the constraint of problem 4. Unfortunately, they do not guarantee that the objective function Φ_ρ monotonically increases - in our experiments we have observed that sometimes it goes down. In fact, when the algorithms failed to converge the value of $\Phi_\rho(\boldsymbol{\theta})$ often had gone into a loop. Next we design a new algorithm with the property that Φ_ρ never decreases.

Our algorithm is shown in Fig. 1. Unlike TRW-E and TRW-T algorithms which use parallel updates, we update vectors $\{\theta^T\}$ sequentially. Therefore, we call our algorithm "sequential tree-reweighted message passing" (TRW-S).

Reparameterization step 1(a) can be implemented in many different ways. One possibility is to convert vectors θ^T to normal forms my running the ordinary max-product BP[4]. However, this would be very expensive if trees are large. A more efficient technique is discussed in section 3.3.

3.1 Weak tree agreement

The algorithm in Fig. 1 does not specify what the stopping criterion is. In this section we address this issue by giving *weak tree agreement* condition (WTA). Later we will show that it characterizes local maxima of the algorithm with respect to function Φ_ρ. More precisely, we will prove that the algorithm has a subsequence converging to a vector satisfying WTA condition. Moreover, if a vector satisfies these conditions, then the algorithm will not make any progress, i.e. it will not increase function Φ_ρ.

In order to define this condition, it is convenient to introduce some notation. Let $\mathrm{OPT}^T(\theta^T)$ be the set of optimal configurations for parameter θ^T. Let $\mathrm{OPT}(\boldsymbol{\theta})$ be the collection $\{\mathrm{OPT}^T(\theta^T) \mid T \in \mathcal{T}\}$ of the sets of optimal configurations for vectors θ^T. It belongs to the set $(2^{\mathcal{X}})^{|\mathcal{T}|} = 2^{\mathcal{X}} \times \ldots \times 2^{\mathcal{X}}$ ($|\mathcal{T}|$ times). For two collections $\mathbb{S}, \tilde{\mathbb{S}} \in (2^{\mathcal{X}})^{|\mathcal{T}|}$ we will write $\mathbb{S} \subset \tilde{\mathbb{S}}$ if $\mathbb{S}^T \subset \tilde{\mathbb{S}}^T$ for any tree T.

Consider some collection of sets of configurations $\mathbb{S} = \{\mathbb{S}^T\} \in (2^{\mathcal{X}})^{|\mathcal{T}|}$. We say that $\mathbb{S}$ is *consistent* if it satisfies the following three conditions:

(a) For any tree T set $\mathbb{S}^T$ is non-empty.

[4]With this scheme a connection to TRW-T algorithm is more apparent. Basically, TRW-T iterates between two phases: (a) running max-product BP for *all* trees, and (b) performing averaging operation for *all* nodes and edges.

0. Initialize $\boldsymbol{\theta}$ so that $\boldsymbol{\theta} \in \mathcal{A}$ and $\sum_T \rho^T \theta^T \equiv \bar{\theta}$.

1. Select some order for nodes and edges in $\mathcal{V} \cup \mathcal{E}$. For each element $\omega \in \mathcal{V} \cup \mathcal{E}$ find all trees $\mathcal{T}_\omega \subset \mathcal{T}$ containing ω. If there is more than one tree, then do the following:

(a) For all trees $T \in \mathcal{T}_\omega$ reparameterize θ^T such that values $\theta^T_{s;j}$ (if $\omega = s$ is a node) or $\theta^T_{s;j} + \theta^T_{st;jk} + \theta^T_{t;k}$ (if $\omega = (s,t)$ is an edge) give correct min-marginals for tree T as in formula 2.

(b) "Averaging" operation:
If $\omega = s$ is a node in $\mathcal{V}$ then
- Compute $\tilde{\theta}_s = \frac{1}{\rho_s} \sum_{T \in \mathcal{T}_s} \rho^T \theta^T_s$
- Set $\theta^T_s := \tilde{\theta}_s$ for trees $T \in \mathcal{T}_s$

If $\omega = (s,t)$ is an edge in $\mathcal{E}$ then
- Compute

$$\tilde{\nu}_{st;jk} = \frac{1}{\rho_{st}} \sum_{T \in \mathcal{T}_{st}} \rho^T (\theta^T_{s;j} + \theta^T_{st;jk} + \theta^T_{t;k})$$

- Set θ^T_s, θ^T_{st}, θ^T_t for trees $T \in \mathcal{T}_{st}$ so that

$$\theta^T_{s;j} + \theta^T_{st;jk} + \theta^T_{t;k} = \tilde{\nu}_{st;jk}$$

2. Check whether a stopping criterion is satisfied; if yes, terminate, otherwise go to step 1.

Figure 1: **Sequential tree-reweighted algorithm (TRW-S).**

(b) If node s is contained in trees T and T', then for any configuration $\mathbf{x}^T \in \mathbb{S}^T$ there exists configuration $\mathbf{x}^{T'} \in \mathbb{S}^{T'}$ which agrees with $\mathbf{x}^T$ on node s, i.e. $x^T_s = x^{T'}_s$.

(c) If edge (s,t) is contained in trees T and T', then for any configuration $\mathbf{x}^T \in \mathbb{S}^T$ there exists configuration $\mathbf{x}^{T'} \in \mathbb{S}^{T'}$ which agrees with $\mathbf{x}^T$ on nodes s and t, i.e. $x^T_s = x^{T'}_s$, $x^T_t = x^{T'}_t$.

Now we can define WTA condition.

Definition 3.2. *Vector $\boldsymbol{\theta} = \{\theta^T\} \in \mathcal{A}$ is said to satisfy the* weak tree agreement *condition if there exists collection $\mathbb{S} \subset OPT(\boldsymbol{\theta})$ which is consistent.*

Note that it can be viewed as a generalization of the *tree agreement* condition introduced in [7]: vectors satisfying tree agreement also satisfy WTA condition.

Also note that WTA condition is different from the *fixed point* condition of TRW algorithms. The latter means that any step of the algorithm does not change vector $\boldsymbol{\theta}$. This in turn implies that all vectors θ^T are in a normal form and $\theta^T_\omega = \theta^{T'}_\omega$ for every element $\omega \in \mathcal{V} \cup \mathcal{E}$ and for every pair of trees $T, T' \in \mathcal{T}_\omega$. It is easy

to see that every fixed point of TRW satisfies WTA condition, but not the other way around.

3.2 Analysis of the algorithm

First we show that similarly to TRW-T algorithm, TRW-S maintains the constraint of problem 4.

Lemma 3.3. *TRW-S algorithm performs reparameterization of the original parameter vector $\bar{\theta}$, i.e. it maintains the property $\sum_T \rho^T \theta^T \equiv \bar{\theta}$.*

This lemma follows directly from the algorithm's construction; see [4] for details.

Next we analyze the behaviour of objective function Φ_ρ during the algorithm. To be specific, we assume for the rest of this section that after reparameterization step 1(a) we have $\min_j \{\theta^T_{s;j}\} = 0$ (if $\omega = s$ is a node) or $\min_{j,k} \{\theta^T_{s;j} + \theta^T_{st;jk} + \theta^T_{t;k}\} = 0$ (if $\omega = (s,t)$ is an edge). This assumption is not essential, however; the behaviour of the algorithm will be the same for any other normalization.

Theorem 3.4. *(a) After any number of steps functions Φ and Φ_ρ do not decrease.*

(b) If vector $\boldsymbol{\theta}$ does not satisfy WTA condition then after a finite number of steps Φ_ρ will increase.

(c) If vector $\boldsymbol{\theta}$ satisfies WTA condition with collection $\mathbb{S}$ then after any number of steps it will still satisfy WTA with the same collection $\mathbb{S}$. Function Φ_ρ will not change.

Proof. Due to space limitations we prove only part (a) for the case of averaging operation for node s. The rest of the proof is in [4].

Suppose that this operation transforms vectors θ^T to vectors $\tilde{\theta}^T$. We need to show that $\Phi(\tilde{\theta}^T) \geq \Phi(\theta^T)$ for any tree $T \in \mathcal{T}_s$. This will imply $\sum_T \rho^T \Phi(\tilde{\theta}^T) \geq \sum_T \rho^T \Phi(\theta^T)$.

Because of our normalization assumption we have

$$\Phi(\theta^T) = \min_{j \in \mathcal{X}_s} \Phi_{s;j}(\theta^T) = \min_{j \in \mathcal{X}_s} \{\theta^T_{s;j}\} + const_s = const_s$$

where $const_s$ is the constant in formula 2. Plugging this into formula 2 we get $\Phi_{s;j}(\theta^T) = \Phi(\theta^T) + \theta^T_{s;j}$.

Vectors θ^T and $\tilde{\theta}^T$ agree on all nodes and edges other than node s. Thus, they have the same optimal configuration $\mathbf{x}^j$ with the constraint $x^j_s = j$. We can write

$$\Phi_{s;j}(\tilde{\theta}^T) = E(\mathbf{x}^j \mid \tilde{\theta}^T) = E(\mathbf{x}^j \mid \theta^T) - \theta^T_{s;j} + \tilde{\theta}^T_{s;j} =$$
$$= \Phi_{s;j}(\theta^T) - \theta^T_{s;j} + \tilde{\theta}^T_{s;j} = \Phi(\theta^T) + \tilde{\theta}^T_{s;j}$$
$$\Phi(\tilde{\theta}^T) = \min_{j \in \mathcal{X}_s} \Phi_{s;j}(\tilde{\theta}^T) = \Phi(\theta^T) + \min_{j \in \mathcal{X}_s} \{\tilde{\theta}^T_{s;j}\}$$

We have $\theta^T_s \geq 0$; inspecting update rule of step 1(b) we conclude that $\tilde{\theta}^T_s \geq 0$ as well. Therefore, the minimum on the RHS is non-negative, so $\Phi(\tilde{\theta}^T) \geq \Phi(\theta^T)$. □

As an immediate consequence of theorem 3.4(b) we get the following result:

Corollary 3.5. *If vector $\boldsymbol{\theta}$ maximizes problem 4 then $\boldsymbol{\theta}$ satisfies WTA condition.*

Unfortunately, the converse is not necessarily true as example in [4] demonstrates.

Finally, we give the convergence theorem. A proof is given in [4]; here we omit it due to space limitations.

Theorem 3.6. *Let $\{\boldsymbol{\theta}^{(i)}\}_i$ be an infinite sequence of vectors obtained by applying step 1 of TRW-S algorithm. Then there exists a subsequence $\{\boldsymbol{\theta}^{(i(m))}\}_m$ such that*

(a) It has reparameterization $\{\tilde{\boldsymbol{\theta}}^{i(m)}\}_m$ (i.e. $\tilde{\boldsymbol{\theta}}^{i(m)} \in \mathcal{A}$ and $(\theta^{i(m)})^T \equiv (\tilde{\theta}^{i(m)})^T$ for any m, T) that converges to some vector $\boldsymbol{\theta}^* \in \mathcal{A}$: $\{\tilde{\boldsymbol{\theta}}^{i(m)}\} \overset{m\to\infty}{\longrightarrow} \boldsymbol{\theta}^*$.

(b) Sequence $\{\Phi_\rho(\boldsymbol{\theta}^{(i)})\}_i$ converges to $\Phi_\rho(\boldsymbol{\theta}^*)$.

(c) Vector $\boldsymbol{\theta}^*$ satisfies WTA condition.

3.3 TRW-S algorithm for a graph with monotonic chains

In this section we focus on step 1(a) - reparameterizing vector θ^T. For simplicity, consider the case when $\omega = s$ is a node. In general, a complete forward pass of the ordinary max-product BP is needed for trees $T \in \mathcal{T}_s$ - sending messages from leaves to node s which we treat as a root[5]. However, this would make the algorithm very inefficient if trees are large. Fortunately, a complete forward pass is not always necessary.

The key idea is that the averaging operation does not invalidate messages in trees $T \in \mathcal{T}_s$ oriented *towards* node s[6]. Therefore, we can "reuse" messages passed in previous steps, i.e. not pass them again. This observation allows us to reduce the number of messages drastically if trees and the order of averaging operations are chosen in a particular way. Specifically, we require trees to be chains which are *monotonic* with respect to some ordering on the graph:

Definition 3.7. *Graph $\mathcal{G}$ and chains $T \in \mathcal{T}$ are said to be* monotonic *if there exists an ordering of nodes $i(u), u \in \mathcal{V}$ such that each chain T satisfies the following property: if $u^T_1, \ldots, u^T_{n(T)}$ are the consecutive nodes in the chain, then the sequence $i(u^T_1), \ldots, i(u^T_{n(T)})$ is monotonic.*

[5] Note that the backward pass (sending messages from the root to leaves) is not needed. It would convert vectors θ^T to normal forms but would not change vectors θ^T_s.

[6] Recall that our algorithm is message-free. The phrase "message is valid for directed edge $(s \to t)$ in tree T" means that sending a message from node s to node t as discussed in section 2.1 would not modify vector θ^T.

0. Initialize $\boldsymbol{\theta}$ so that $\boldsymbol{\theta} \in \mathcal{A}$ and $\sum_T \rho^T \theta^T \equiv \bar{\theta}$.

1. For nodes $t \in \mathcal{V}$ do the following operations in the order of increasing $i(t)$:

   (a) For every edge $(s,t) \in \mathcal{E}$ with $i(s) < i(t)$ do the following:

   - If $\mathcal{T}_{st}$ contains more than one chain then perform the averaging operation for edge (s,t).
   - For chains in $\mathcal{T}_{st}$ pass a message from s to t.

   (b) Perform the averaging operation for node t.

2. Reverse the ordering: set $i(u) := |\mathcal{V}| + 1 - i(u)$.

3. Check whether a stopping criterion is satisfied; if yes, terminate, otherwise go to step 1.

Figure 2: **TRW-S algorithm for a graph with monotonic chains.**

As an example, we could choose $\mathcal{T}$ to be the set of edges; it is easy to see that they are monotonic for any ordering of nodes. However, it might be advantageous to choose longer trees since the information might propagate faster through the graph.

The algorithm for a graph with monotonic chains is shown in Fig. 2. Its properties are summarized by the following lemma whose proof is given in [4].

Lemma 3.8. *Starting with the second pass, the following properties hold during step 1 for node t:*

(a) For each edge $(u,v) \in \mathcal{E}$ with $i(u) < i(v) < i(t)$ messages $(u \to v)$ in chains $T \in \mathcal{T}_{uv}$ are valid. This property also holds for node $v = t$ in the beginning and in the end of step 1(b).

(b) For each edge $(u,v) \in \mathcal{E}$ with $i(u) \geq i(t)$ messages $(v \to u)$ in chains $T \in \mathcal{T}_{uv}$ are valid.

In addition, property (a) holds during the first pass of the algorithm.

Thus, the algorithm in Fig. 2 is a special case of the algorithm in Fig. 1, if we treat the first pass of former algorithm as part of initialization of the latter one.

Efficient implementation The algorithm in Fig. 2 requires $O(|\mathcal{T}_s| \cdot |\mathcal{X}_s|)$ storage for node s and $O(|\mathcal{T}_{st}| \cdot |\mathcal{X}_s| \cdot |\mathcal{X}_t|)$ storage for edge (s,t). However, we can reduce it to $O(|\mathcal{X}_s|)$ and $O(|\mathcal{X}_s| + |\mathcal{X}_t|)$, respectively, using two ideas[7]. First, it can be seen that the algorithm maintains the following equalities: $\theta_s^T = \theta_s^{T'}$ for $T, T' \in \mathcal{T}_s$ and $\theta_{st}^T = \theta_{st}^{T'}$ for $T, T' \in \mathcal{T}_{st}$ (assuming that they hold after initialization). Second,

[7] We assume that storage required for vectors $\bar{\theta}_{st}$ is negligible. This holds for many energy functions used in practice, e.g. for functions with Potts terms.

0. Set all messages to zero.

1. For nodes $t \in \mathcal{V}$ do the following operation in the order of increasing $i(t)$:

   - For every edge $(s,t) \in \mathcal{E}$ with $i(s) < i(t)$ update message M_{st} as follows:
     - Compute $\theta_s = \frac{1}{\rho_s}\bar{\theta}_s + \sum_{(u,s)\in\mathcal{E}} \frac{\rho_{us}}{\rho_s} M_{us}$
     - Set $M_{st;k} := \min_j \{(\theta_{s;j} - M_{ts;j}) + \frac{1}{\rho_{st}}\bar{\theta}_{st;jk}\}$

2. Reverse the ordering: set $i(u) := |\mathcal{V}| + 1 - i(u)$.

3. Check whether a stopping criterion is satisfied; if yes, terminate, otherwise go to step 1.

Figure 3: **Efficient implementation of the algorithm in Fig. 2 using messages.**

reparameterizations θ^T can be stored using messages $M_{st} = \{M_{st;k} \,|\, k \in \mathcal{X}_t\}$ for directed edges $(s \to t) \in \mathcal{E}$, which correspond to vector $\boldsymbol{\theta}$ as follows:

$$\begin{aligned} \theta_t^T &= \tfrac{1}{\rho_t}\bar{\theta}_t + \textstyle\sum_{(s,t)\in\mathcal{E}} \frac{\rho_{st}}{\rho_t} M_{st} \\ \theta_{st;jk}^T &= \tfrac{1}{\rho_{st}}\bar{\theta}_{st;jk} - M_{st;k} - M_{ts;j} \end{aligned}$$

The resulting algorithm is shown in Fig. 3. Note that for many important choices of terms $\bar{\theta}_{st}$ message update in step 1 can be done very efficiently using distance transforms [2].

The message update rule of our algorithm is very similar to that of TRW-E algorithm (they would be identical if trees were spanning). However, TRW-E performs the updates in parallel, while we do it sequentially. As a result, we get convergence properties proved earlier.

4 Experimental results

We have compared four algorithms: ordinary max-product algorithm (BP) and three tree-reweighted algorithms (TRW-E, TRW-T, TRW-S). For TRW-E algorithm we also experimented with the damping parameter $\gamma \in (0,1]$; as reported in [7], TRW-E converges if sufficiently damped. We tuned this parameter for the experiments below. We used rectangular graphs with 4-neighborhood systems. The trees were horizontal and vertical chains with uniform probability. We treated them as two forests with probabilities 0.5; then we have $\rho_s = 1$ for nodes s and $\rho_{st} = 0.5$ for edges (s,t). We processed nodes in the raster order.

For BP algorithm we implemented a sequential update scheme whose order is similar to that of TRW-S: we pass messages from the top left corner to the bottom right corner and back. We experimented with the parallel update scheme and found that it was much slower.

The running time of the algorithms (number of iterations) was measured in terms of the number of passed messages divided by the number of directed edges in the graph.

For TRW algorithms we measured two quantities as functions of time: the value of $\Phi_\rho(\boldsymbol{\theta})$ and the value of the energy $E(\mathbf{x} \mid \bar{\theta})$. Note that the former is a lower bound on the optimal value of the energy, and the latter is an upper bound. Solution $\mathbf{x}$ for vector $\boldsymbol{\theta}$ was computed as follows: for each node s we determine label x_s minimizing $\sum_T \rho^T \theta_s^T(x_s)$. For BP algorithm, we can determine only one of these quantities, namely the value of the energy.

TRW algorithms were deemed to converge if we could find a consistent collection $\mathbb{S} = \{\mathbb{S}^T\}$ such that $E(\mathbf{x}^T \mid \theta^T) - \Phi(\theta^T) < 10^{-8}$ for any tree T and configuration $\mathbf{x}^T \in \mathbb{S}^T$. Note that checking this condition is expensive. In practice it may be better to use a test based on the behaviour of function Φ_ρ.

4.1 Synthetically generated problems

We have tested two types of problems: Ising model with attractive potentials and with mixed potentials. Single-node potentials were generated as independent gaussians: $\bar{\theta}_{s;0}, \bar{\theta}_{s;1} \sim \mathcal{N}(0, (0.25)^2)$. Pairwise potentials were set as follows: $\bar{\theta}_{st;00} = \bar{\theta}_{st;11} = 0$, $\bar{\theta}_{st;01} = \bar{\theta}_{st;10} = \lambda_{st}$ where λ_{st} was generated as $|\mathcal{N}(0,1)|$ for attractive potentials and as $\mathcal{N}(0,1)$ for mixed potentials. The size of the problem was 20x20. We tested the algorithms on 100 sample problems.

Attractive potentials First we tested the behaviour of TRW-E algorithm with respect to the damping parameter γ. For $\gamma = 1$, only 40% of the trials converged; however, for smaller values of γ the algorithm always converged. The smallest number of iterations was achieved at $\gamma = 0.96$. We used this value for the subsequent experiment.

Next we tested convergence rate of the four algorithms. TRW-S and BP converged in 100% of the cases, and TRW-T never converged (we observed that it went into a loop). The average number of iterations until convergence was as follows: 73.1 for TRW-E, 21.8 for TRW-S and 7.1 for BP. Note that it is not very fair to compare convergence for TRW algorithms and for BP, since convergence criteria are different. Next test provides a better comparison.

In this test we measured average values of lower bound $\Phi_\rho(\boldsymbol{\theta})$ and energy $E(\mathbf{x} \mid \bar{\theta})$ as functions of time for the first 50 iterations. Results are shown in Fig. 4(a,b). In most cases TRW-E and TRW-S found an optimal solution, and BP found an optimal solution for a smaller number of cases. TRW-S was decreasing the energy faster than the other algorithms. Surprisingly, TRW-T algorithm failed completely. Moreover, it demonstrated similar behaviour for the tests below so it is not shown.

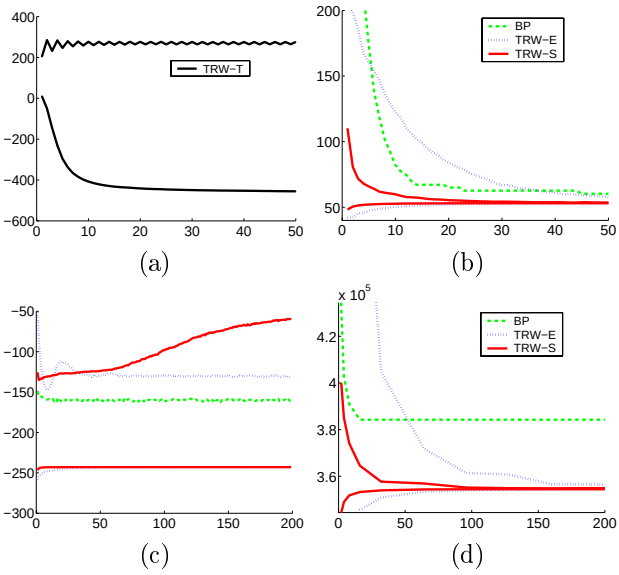

Figure 4: **Energy plots. Horizontal axis: number of iterations. Vertical axis, upper curves: average value of the energy $E(\mathbf{x} \mid \bar{\theta})$. Vertical axis, lower curves for TRW algorithms: average value of $\Phi_\rho(\boldsymbol{\theta})$. (a) Ising model, attractive potentials, TRW-T algorithm. (b) Ising model, attractive potentials, three other algorithms. (c) Ising model, mixed potentials. (d) Tsukuba dataset, Potts model.**

Mixed potentials The behaviour of TRW-E and TRW-S algorithms was substantially different from the previous case. We observed that the smallest value of the energy was achieved after a small number of iterations (5-10), and after that the energy increased. TRW-E with damping and TRW-S always converged, however the value of the energy at convergence was larger than in the middle. Moreover, the energy obtained by BP was smaller than the energy of TRW algorithms, despite the fact BP did not converge. This behaviour is shown in Fig. 4(c). For TRW-E algorithm we picked empirically $\gamma = 0.3$.

Poor performance of TRW algorithms for the problem with mixed potentials needs further investigation.

4.2 Stereo matching problem

We have tested the algorithms on the energy function arising in the stereo matching problem [1]. The input is two images taken from different viewpoints, and the goal is to find a disparity for every pixel in the left image. Similarly to [1], we used Potts interaction model.

For TRW-E algorithm we picked empirically $\gamma = 0.95$.

Fig. 4(d) shows energy plots for the Tsukuba dataset used in [1]. Precise numbers are as follows:

alg.	number of iterations				
	4	16	64	256	1024
expansion move [1]	719				
BP	48479	29815	29815	29815	29815
TRW-E	1073357	175433	17475	905	889
	-29418.0	-8966.5	-1058.7	-26.8	-0.47
TRW-S	30663	10101	2465	463	643
	-5496.0	-1189.0	-92.7	-0.057	0

Top rows show the value of the energy, bottom rows for TRW-E and TRW-S - the value of lower bound $\Phi_\rho(\boldsymbol{\theta})$. Note that we subtracted 354453 from all values - it appears to be the maximum of problem 4. Using a different initialization for TRW-S algorithm, we were also able to obtain a configuration with energy 354453+103.

TRW-S algorithm is a clear winner - it decreases the energy more rapidly than the other algorithms. The second best is TRW-E algorithm with damping, although in the beginning the energy decreases more slowly than the energy for BP. Our algorithm obtains lower energy than BP, and slightly lower energy than the expansion move algorithm [1].

5 Discussion and conclusions

We have developed a new algorithm which can be viewed as a method for direct maximization of objective function Φ_ρ subject to the constraint of problem 4. We gave a precise characterization of local maxima of this function with respect to TRW-S algorithm. We showed that the algorithm is guaranteed to have a subsequence converging to such a maximum.

As all tree-reweighted algorithms, our method is not guaranteed to find a *global* maximum. Nevertheless, experimental results suggest that this is not an issue for certain synthetic and real problems. For the stereo matching problem we were able to obtain slightly lower energy than the expansion move algorithm [1] which is considered to be the most accurate energy minimization technique for such problems. TRW-S algorithm outperforms both TRW algorithms in [7] and the ordinary max-product BP.

Our algorithm can be implemented efficiently only for a particular choice of trees. Fortunately, nothing prevents us from using this choice - the optimal value of the lower bound does not depend on it.

In our experiments we noticed that TRW-S algorithm would always converge to a fixed point of TRW, although such convergence would usually take much longer than achieving a weak tree agreement. However, we have not been able to prove this general convergence. On the other hand, in practice it does not make much sense to run the algorithm after WTA has been achieved since function Φ_ρ and, more importantly, the output configuration will not change.

In the future we plan to extend techniques proposed in this paper to sum-product TRW algorithm [9].

Acknowledgements

I would like to thank Thomas Minka for helpful discussions.

References

[1] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 23(11), November 2001.

[2] P.F. Felzenszwalb and D.P. Huttenlocher. Efficient belief propagation for early vision. *CVPR*, 2004.

[3] T. Heskes, K. Albers, and B. Kappen. Approximate inference and constrained optimization. *UAI*, 2003.

[4] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. Technical Report MSR-TR-2004-90, Misrosoft Research, September 2004.

[5] Y.W. Teh and M. Welling. The unified propagation and scaling algorithm. *NIPS*, 2002.

[6] M.J. Wainwright, T.S. Jaakkola, and A.S. Willsky. A new class of upper bounds on the log partition function. *UAI*, 2002.

[7] M.J. Wainwright, T.S. Jaakkola, and A.S. Willsky. MAP estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. Technical Report UCB/CSD-03-1269, UC Berkeley CS Division, August 2003.

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

[9] M.J. Wainwright, T.S. Jaakkola, and A.S. Willsky. Tree-reweighted belief propagation and approximate ML estimation by pseudo-moment matching. *AISTATS*, 2003.

[10] M.J. Wainwright, T.S. Jaakkola, and A.S. Willsky. Tree consistency and bounds on the performance of the max-product algorithm and its generalizations. *Statistics and Computing*, 14(2):143–166, April 2004.

[11] W. Wiegerinck and T. Heskes. Fractional belief propagation. *NIPS*, 2000.

[12] J.S. Yedidia, W.T. Freeman, and Y.Weiss. Generalized belief propagation. *NIPS*, 2000.

[13] A.L. Yuille. CCCP algorithms to minimize the Bethe and Kikuchi free energies: Convergent alternatives to belief propagation. *Neural Computation*, 14:1691–1722, 2002.