Fixing Max-Product: Convergent Message Passing Algorithms for MAP LP-Relaxations

Amir Globerson **Tommi Jaakkola**
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139
gamir,tommi@csail.mit.edu

Abstract

We present a novel message passing algorithm for approximating the MAP problem in graphical models. The algorithm is similar in structure to max-product but unlike max-product it always converges, and can be proven to find the exact MAP solution in various settings. The algorithm is derived via block coordinate descent in a dual of the LP relaxation of MAP, but does not require any tunable parameters such as step size or tree weights. We also describe a generalization of the method to cluster based potentials. The new method is tested on synthetic and real-world problems, and compares favorably with previous approaches.

Graphical models are an effective approach for modeling complex objects via local interactions. In such models, a distribution over a set of variables is assumed to factor according to cliques of a graph with potentials assigned to each clique. Finding the assignment with highest probability in these models is key to using them in practice, and is often referred to as the MAP (maximum aposteriori) assignment problem. In the general case the problem is NP hard, with complexity exponential in the tree-width of the underlying graph.

Linear programming (LP) relaxations have proven very useful in approximating the MAP problem, and often yield satisfactory empirical results. These approaches relax the constraint that the solution is integral, and generally yield non-integral solutions. However, when the LP solution is integral, it is guaranteed to be the exact MAP. For some classes of problems the LP relaxation is provably correct. These include the minimum cut problem and maximum weight matching in bi-partite graphs [7]. Although LP relaxations can be solved using standard LP solvers, this may be computationally intensive for large problems [11]. The key problem with generic LP solvers is that they do not use the graph structure explicitly and thus may be sub-optimal in terms of computational efficiency.

The max-product method [6] is a message passing algorithm that is often used to approximate the MAP problem. In contrast to generic LP solvers, it makes direct use of the graph structure in constructing and passing messages, and is also very simple to implement. The relation between max-product and the LP relaxation has remained largely elusive, although there are some notable exceptions: For tree-structured graphs, max-product and LP both yield the exact MAP. A recent result [1] showed that for maximum weight matching on bi-partite graphs max-product and LP also yield the exact MAP [1]. Finally, Wainwright et al. [9] proposed the Tree-Reweighted max-product (TRMP) algorithm - a variation on max-product that is guaranteed to converge to the LP solution for binary x_i variables, as shown in [5].

In this work, we propose the Max Product Linear Programming algorithm (MPLP) - a very simple variation on max-product that is always guaranteed to converge, and has several advantageous properties. The algorithm is derived from the convex-dual of the LP relaxation [2], and is equivalent to block coordinate descent in this dual. Although this results in monotone improvement of the dual objective, global convergence is not always guaranteed since coordinate descent may get

stuck in suboptimal points. However, this can be remedied using various approaches (e.g., auction algorithms as in [2]), and in practice we have found MPLP to converge to the LP solution in a majority of the cases we studied. To derive MPLP we use a special form of the dual LP, which involves the introduction of redundant primal variables and constraints. We show how the dual variables corresponding to these constraints turn out to be the *messages* passed in the algorithm.

We evaluate the method on Potts models and protein design problems, and show that it compares favorably with max-product (which often does not converge for these problems) and TRMP.

1 The Max-Product and MPLP Algorithms

The max-product algorithm [6] is one of the most often used methods for solving MAP problems. Although it is neither guaranteed to converge to the correct solution, or in fact converge at all, it provides satisfactory results in some cases. Here we present two algorithms: EMPLP (edge based MPLP) and NMPLP (node based MPLP), which are structurally very similar to max-product, but have several key advantages:

- They always converge.
- No additional parameters (e.g., tree weights as in [5]) are required.
- If the beliefs $b_i(x_i)$ are not tied (i.e., $b_i(x_i)$ has a unique maximizer) then the output of MPLP is the exact MAP assignment.
- For binary variables, MPLP can be used to obtain the solution to the LP relaxation MAPLPR. This implies that when the LP relaxation is exact and variables are binary, MPLP will find the MAP solution. Moreover, for any variable whose beliefs are not tied, the MAP assignment can be found (i.e., the solution is partially decodable).

Pseudo code for the algorithms (and for max-product) is given in Fig. 1. As we show in the next sections, MPLP is essentially a coordinate descent algorithm in the dual of MAPLPR. Every update of the MPLP messages corresponds to exact minimization of a set of dual variables. For EMPLP minimization is over the set of variables corresponding to an edge, and for NMPLP it is over the set of variables corresponding to all the edges a given node appears in (i.e., a star). The coordinate descent property immediately implies both monotone improvement of the dual objective and convergence. The other properties of MPLP also result from its relation to the LP dual. In what follows we describe the derivation of the MPLP algorithms and prove their properties.

2 The MAP Problem and its LP Relaxation

We consider functions over n variables $\boldsymbol{x} = \{x_1, \ldots, x_n\}$ defined as follows. Given a graph $G = (V, E)$ with n vertices, and potentials $\theta_{ij}(x_i, x_j)$ for all edges $ij \in E$, define the function[1]

$$f(\boldsymbol{x}; \boldsymbol{\theta}) = \sum_{ij \in E} \theta_{ij}(x_i, x_j) \; . \tag{1}$$

The MAP problem is defined as finding an assignment $\boldsymbol{x}_M$ that maximizes the function $f(\boldsymbol{x}; \boldsymbol{\theta})$. Below we describe the standard LP relaxation for this problem. Denote by $\{\mu_{ij}(x_i, x_j)\}_{ij \in E}$ distributions over variables corresponding to edges $ij \in E$ and $\{\mu_i(x_i)\}_{i \in V}$ distributions corresponding to nodes $i \in V$. We will use $\boldsymbol{\mu}$ to denote a given set of distributions over all edges and nodes. The set $\mathcal{M}_L(G)$ is defined as the set of $\boldsymbol{\mu}$ where pairwise and singleton distributions are consistent

$$\mathcal{M}_L(G) = \left\{ \boldsymbol{\mu} \geq 0 \;\middle|\; \begin{array}{ll} \sum_{\hat{x}_i} \mu_{ij}(\hat{x}_i, x_j) = \mu_j(x_j) \;\; , \;\; \sum_{\hat{x}_j} \mu_{ij}(x_i, \hat{x}_j) = \mu_i(x_i) & \forall ij \in E, x_i, x_j \\ \sum_{x_i} \mu_i(x_i) = 1 & \forall i \in V \end{array} \right\}$$

Now consider the following linear program:

$$\underline{\text{MAPLPR}}: \qquad \boldsymbol{\mu}^{L*} = \arg \max_{\boldsymbol{\mu} \in \mathcal{M}_L(G)} \boldsymbol{\mu} \cdot \boldsymbol{\theta} \; . \tag{2}$$

[1] We note that some authors also add a term $\sum_{i \in V} \theta_i(x_i)$ to $f(\boldsymbol{x}; \boldsymbol{\theta})$. However, these terms can be included in the pairwise functions $\theta_{ij}(x_i, x_j)$, so we ignore them for simplicity.

where $\boldsymbol{\mu}\cdot\boldsymbol{\theta}$ is shorthand for $\boldsymbol{\mu}\cdot\boldsymbol{\theta} = \sum_{ij\in E}\sum_{x_i,x_j}\theta_{ij}(x_i,x_j)\mu_{ij}(x_i,x_j)$. It is easy to show (see e.g., [9]) that the optimum of MAPLPR yields an upper bound on the MAP value, i.e. $\boldsymbol{\mu}^{L*}\cdot\boldsymbol{\theta} \geq f(\boldsymbol{x}_M)$. Furthermore, when the optimal $\mu_i(x_i)$ have only integral values, the assignment that maximizes $\mu_i(x_i)$ yields the correct MAP assignment. In what follows we show how the MPLP algorithms can be derived from the dual of MAPLPR.

3 The LP Relaxation Dual

Since MAPLPR is an LP, it has an equivalent convex dual. In App. A we derive a special dual of MAPLPR using a different representation of $\mathcal{M}_L(G)$ with redundant variables. The advantage of this dual is that it allows the derivation of simple message passing algorithms. This does not seem to be possible in the standard MAPLPR dual . The dual is described in the following proposition.

Proposition 1 *The following optimization problem is a convex dual of MAPLPR*

$$\begin{array}{ll} \underline{\textit{DMAPLPR}}: & \\ \min & \sum_i \max_{x_i} \sum_{k\in N(i)} \max_{x_k} \beta_{ki}(x_k,x_i) \\ s.t. & \beta_{ji}(x_j,x_i) + \beta_{ij}(x_i,x_j) = \theta_{ij}(x_i,x_j)\ , \end{array} \tag{3}$$

where the dual variables are $\beta_{ij}(x_i,x_j)$ for all $ij, ji \in E$ and values of x_i and x_j.

The dual has an intuitive interpretation in terms of re-parameterizations. Consider the *star* shaped graph G_i consisting of node i and all its neighbors $N(i)$. Assume the potential on edge ki (for $k \in N(i)$) is $\beta_{ki}(x_k,x_i)$. The value of the MAP assignment for this model is $\max_{x_i}\sum_{k\in N(i)}\max_{x_k}\beta_{ki}(x_k,x_i)$. This is exactly the term in the objective of DMAPLPR. Thus the dual corresponds to individually decoding star graphs around all nodes $i \in V$ where the potentials on the graph edges should sum to the original potential. It is easy to see that this will always result in an upper bound on the MAP value. The somewhat surprising result of the duality is that there exists a $\boldsymbol{\beta}$ assignment such that *star decoding* yields the optimal value of MAPLPR.

4 Block Coordinate Descent in the Dual

To obtain a convergent algorithm for DMAPLPR we use a simple block coordinate descent strategy. At every iteration, fix all variables except a subset, and optimize over this subset. It turns out that this can be done in closed form for the cases we consider.

We begin by deriving the EMPLP algorithm. Consider fixing all the $\boldsymbol{\beta}$ variables except those corresponding to some edge $ij \in E$ (i.e., β_{ij} and β_{ji}), and minimizing DMAPLPR over the non-fixed variables. Only two terms in the DMAPLPR objective depend on β_{ij} and β_{ji}. We can write those as

$$f(\beta_{ij},\beta_{ji}) = \max_{x_i}\left[\lambda_i^{-j}(x_i) + \max_{x_j}\beta_{ji}(x_j,x_i)\right] + \max_{x_i}\left[\lambda_j^{-i}(x_j) + \max_{x_i}\beta_{ij}(x_i,x_j)\right] \tag{4}$$

where we defined $\lambda_i^{-j}(x_i) = \sum_{k\in N(i)\setminus j}\lambda_{ki}(x_i)$ and $\lambda_{ki}(x_i) = \max_{x_k}\beta_{ki}(x_k,x_i)$ as in App. A. Note that the function $f(\beta_{ij},\beta_{ji})$ depends on the other $\boldsymbol{\beta}$ values only through $\lambda_j^{-i}(x_j)$ and $\lambda_i^{-j}(x_i)$. This implies that the optimization can be done solely in terms of $\lambda_{ij}(x_j)$ and there is no need to store the β values explicitly. The optimal β_{ij},β_{ji} are obtained by minimizing $f(\beta_{ij},\beta_{ji})$ subject to the *re-parameterization* constraint $\beta_{ji}(x_j,x_i) + \beta_{ij}(x_i,x_j) = \theta_{ij}(x_i,x_j)$. The following proposition characterizes the minimum of $f(\beta_{ij}(x_i,x_j))$. In fact, as mentioned above, we do not need to characterize the optimal $\beta_{ij}(x_i,x_j)$ itself, but only the new λ values.

Proposition 1 *Maximizing the function $f(\beta_{ij},\beta_{ji})$ yields the following $\lambda_{ij}(x_j)$ and $\lambda_{ji}(x_i)$*

$$\lambda_{ji}(x_i) = -\frac{1}{2}\lambda_i^{-j}(x_i) + \frac{1}{2}\max_{x_j}\left[\lambda_j^{-i}(x_j) + \theta_{ij}(x_i,x_j)\right]$$

The proposition is proved in App. B. The λ updates above result in the EMPLP algorithm, described in Fig. 1. Note that since the $\boldsymbol{\beta}$ optimization affects both $\lambda_{ji}(x_i)$ and $\lambda_{ij}(x_j)$, both these *messages* need to be updated simultaneously.

Inputs: A graph $G = (V, E)$, potential functions $\theta_{ij}(x_i, x_j)$ for each edge $ij \in E$.

Initialization: $\forall ij, ji \in E$ set:
$$\begin{cases} m_{ij}(x_j) = 0 \\ \lambda_{ij}(x_j) = \frac{1}{2}\max_{x_i} \theta_{ij}(x_i, x_j) \\ \gamma_{ij}(x_j) = \max_{x_i}\left[\theta_{ij}(x_i, x_j) + \frac{1}{2}\sum_{k\in N(i)\setminus j} \max_{x_k} \theta_{ki}(x_k, x_i)\right] \end{cases}$$

Algorithm:

- Iterate until convergence
  - MAXPROD: Update all messages (c_{ji} shifts $\max_{x_i} m_{ji}(x_i)$ to zero)
$$m_{ji}(x_i) \leftarrow \max_{x_j}\left[m_j^{-i}(x_j) + \theta_{ij}(x_i, x_j)\right] - c_{ji}$$
  - EMPLP: Iterate over edges, and for $ij, ji \in E$ update
$$\lambda_{ji}(x_i) \leftarrow -\frac{1}{2}\lambda_i^{-j}(x_i) + \frac{1}{2}\max_{x_j}\left[\lambda_j^{-i}(x_j) + \theta_{ij}(x_i, x_j)\right]$$
  - NMPLP: Iterate over nodes $i \in V$ and update all $\gamma_{ij}(x_j)$ where $j \in N(i)$
$$\gamma_{ij}(x_j) = \max_{x_i}\left[\theta_{ij}(x_i, x_j) - \gamma_{ji}(x_i) + \frac{2}{|N(i)|+1}\sum_{k\in N(i)} \gamma_{ki}(x_i)\right]$$
- Calculate node "beliefs": Set $b_i(x_i)$ to be the sum of incoming messages into node $i \in V$ (e.g., for NMPLP set $\sum_{k\in N(i)} \gamma_{ki}(x_i)$).

Output: Return assignment $\boldsymbol{x}$ defined as $x_i = \arg\max_{\hat{x}_i} b(\hat{x}_i)$.

Figure 1: The max-product, EMPLP and NMPLP algorithms. Max-product, EMPLP and NMPLP use messages m_{ij}, λ_{ij} and γ_{ij} respectively. We use the notation $m_j^{-i}(x_j) = \sum_{k\in N(j)\setminus i} m_{kj}(x_j)$.

We proceed to derive the NMPLP algorithm. For a given node $i \in V$, we consider all its neighbors $j \in N(i)$, and wish optimize over the variables $\beta_{ji}(x_j, x_i)$ for $ji, ij \in E$ (i.e., all the edges in a *star* centered on i), while the other variables are fixed. One way of doing so is to use the EMPLP algorithm for the edges in the star, and iterate it until convergence. We now show that the result of this optimization can be found in closed form. The assumption about $\boldsymbol{\beta}$ being fixed outside the star implies that $\lambda_j^{-i}(x_j)$ is fixed. Define: $\gamma_{ji}(x_i) = \max_{x_j}\left[\theta_{ij}(x_i, x_j) + \lambda_j^{-i}(x_j)\right]$. Simple algebra yields the following relation between $\lambda_i^{-j}(x_i)$ and $\gamma_{ki}(x_i)$ for $k \in N(i)$

$$\lambda_i^{-j}(x_i) = -\gamma_{ji}(x_i) + \frac{2}{|N(i)|+1}\sum_{k\in N(i)} \gamma_{ki}(x_i) \tag{5}$$

Plugging this into the definition of $\gamma_{ji}(x_i)$ we obtain the NMPLP update in Fig. 1. Initialization for both algorithms follows from setting $\boldsymbol{\beta} = 0$.

5 Convergence Properties

The MPLP algorithm improves the dual objective at every iteration. This in itself does not guarantee convergence to the optimum of the dual, since coordinate descent algorithms may get stuck at a point where no improvement is possible over single coordinates (or subsets of coordinates as used here), but the point is not a global optimum. There are ways of overcoming this difficulty, for example by not moving ϵ away from the optimum for a given coordinate (see [2], p. 636). We leave such extensions for further work. In this section we provide several results about the properties of the MPLP fixed points and their relation to the corresponding LP. First, we claim that if all beliefs have unique maxima then the *exact* MAP assignment is obtained.

Proposition 2 *Assume MPLP converges to beliefs $b_i(x_i)$ such that for all i the function $b_i(x_i)$ has a unique maximizer x_i^*. Then $\boldsymbol{x}^*$ is the solution to the MAP problem and the LP relaxation is exact.*

Since the dual objective is always greater than or equal to the MAP value, it suffices to show that there exists a dual feasible point whose objective value is $f(x^*)$. Denote by β^*, λ^* the value of the

corresponding dual parameters at the fixed point of MPLP. Then the dual objective satisfies

$$\sum_i \max_{x_i} \sum_{k \in N(i)} \lambda^*_{ki}(x_i) = \sum_i \sum_{k \in N(i)} \max_{x_k} \beta^*_{ki}(x_k, x^*_i) = \sum_i \sum_{k \in N(i)} \beta^*_{ki}(x^*_k, x^*_i) = f(\boldsymbol{x}^*)$$

The to see why the second equality holds, note that

$$b_i(x^*_i) = \max_{x_i, x_j} \lambda_i^{-j}(x_i) + \beta_{ji}(x_j, x_i) \ , \ b_j(x^*_j) = \max_{x_i, x_j} \lambda_j^{-i}(x_j) + \beta_{ij}(x_i, x_j)$$

By the equalization property in Eq. 9 the arguments of the two max operations are equal. From the unique maximum assumption it follows that x^*_i, x^*_j are the unique maximizers of the above. It follows that β_{ji}, β_{ij} are also maximized by x^*_i, x^*_j.

In the general case, the MPLP fixed point may not correspond to a primal optimum because of the local optima problem with coordinate descent. However, when the variables are binary, fixed points do correspond to primal solutions, as the following proposition states.

Proposition 3 *When all x_i are binary variables, MPLP will converge to the dual optimum.*

The claim can be shown by constructing a primal optimal solution $\boldsymbol{\mu}^*$. For tied b_i, set $\mu^*_i(x_i)$ to 0.5 and for untied b_i, set $\mu^*_i(x^*_i)$ to 1. If b_i, b_j are not tied we set $\mu^*_{ij}(x^*_i, x^*_j) = 1$. If b_i is not tied but b_j is, we set $\mu^*_{ij}(x^*_i, x_j) = 0.5$. If b_i, b_j are tied then β_{ji}, β_{ij} can be shown to be maximized at either $x^*_i, x^*_j = (0,0), (1,1)$ or $x^*_i, x^*_j = (0,1), (1,0)$. We then set μ^*_{ij} to be 0.5 at one of these assignment pairs. The resulting $\boldsymbol{\mu}^*$ is clearly primal feasible. Setting $\delta^*_i = b^*_i$ we obtain that the dual variables $(\delta^*, \lambda^*, \beta^*)$ and primal $\boldsymbol{\mu}^*$ satisfy complementary slackness for the LP in Eq. 7 and therefore $\boldsymbol{\mu}^*$ is primal optimal. The binary optimality result implies partial decodability, since [5] shows that the LP is partially decodable for binary variables.

6 Beyond pairwise potentials: Generalized MPLP

In the previous sections we considered maximizing functions which factor according to the edges of the graph. A more general setting considers clusters $c_1, \ldots, c_k \subset \{1, \ldots, n\}$ (the set of clusters is denoted by $\mathcal{C}$), and a function[2] $f(\boldsymbol{x}; \boldsymbol{\theta}) = \sum_c \theta_c(x_c)$ defined via potentials over clusters $\theta_c(x_c)$. The MAP problem in this case also has an LP relaxation (see e.g. [10]). To define the LP we introduce the following definitions: $\mathcal{S} = \{c \cap \hat{c} : c, \hat{c} \in \mathcal{C}, c \cap \hat{c} \neq \emptyset\}$ is the set of intersection between clusters and $\mathcal{S}(c) = \{s \in \mathcal{S} : s \subseteq c\}$ is the set of overlap sets for cluster c.We now consider marginals over the variables in $c \in \mathcal{C}$ and their intersection and require that cluster marginals *agree* on their overlap. Denote this set by $\mathcal{M}_L(\mathcal{C})$. The LP relaxation is then to maximize $\boldsymbol{\mu} \cdot \boldsymbol{\theta}$ subject to $\boldsymbol{\mu} \in \mathcal{M}_L(\mathcal{C})$.

As in the Sec. 4, we can derive message passing updates that result in monotone decrease of the dual LP of the above relaxation. The derivation is similar and we omit the details. The key observation is that one needs to introduce $|\mathcal{S}(c)|$ copies of each marginal $\mu_c(x_c)$ (instead of the two copies in the pairwise case). Next, as in the EMPLP derivation we assume all $\boldsymbol{\beta}$ are fixed except those corresponding to some cluster c. The resulting messages are $\lambda_{c \to s}(x_s)$ from a cluster c to all of its intersection sets $s \in \mathcal{S}(c)$. The update on these messages turns out to be:

$$\lambda_{c \to s}(x_s) = -\left(1 - \frac{1}{|\mathcal{S}(c)|}\right) \lambda_s^{-c}(x_s) + \frac{1}{|\mathcal{S}(c)|} \max_{x_{c \setminus s}} \left[\sum_{\hat{s} \in \mathcal{S}(c) \setminus s} \lambda_{\hat{s}}^{-c}(x_{\hat{s}}) + \theta_c(x_c) \right]$$

where for a given $c \in \mathcal{C}$ all $\lambda_{c \to s}$ messages should be updated simultaneously for $s \in \mathcal{S}(c)$. We refer to this algorithm as Generalized EMPLP (GEMPLP). It is also possible to derive an algorithm that like NMPLP updates several clusters simultaneously but its structure is more involved and we do not address it here.

7 Related Work

Weiss et al. [10] recently studied the fixed points of a class of *max-product like* algorithms. Their analysis focused on properties of fixed points rather than convergence guarantees. Specifically, they

[2]The clusters are sometimes required to correspond to cliques in a graph. We do not pose that requirement here, and the analysis does not require an underlying graph.

showed that if the counting numbers used in a generalized max-product algorithm satisfy certain properties, then its fixed points will be the exact MAP if the beliefs have unique maxima, and for binary variables the solution can be partially decodable. Both these properties are obtained for the MPLP fixed points, and in fact we can show that MPLP satisfies the conditions in [10], so that we obtain these properties as corollaries of [10]. We stress however, that [10] does not address convergence of algorithms, but rather properties of their fixed points, if they converge.

MPLP is similar in some aspects to Kolmogorov's TRW-S algorithm [4]. TRW-S is also monotone coordinate descent method in a certain dual of the LP relaxation and its fixed points also have similar guarantees to those of MPLP [5]. Furthermore, convergence to a local optimum may occur, as it does for MPLP. One advantage of MPLP lies in the simplicity of its updates and the fact that it is parameter free . The other is its simple generalization to potentials over clusters of nodes (Sec. 6). Although TRW-S may be extended to such a setting (e.g., by considering pairwise interaction between nodes that correspond to cluster, and using spanning trees on this graph) this seems less straightforward than the MPLP extension.

Vontobel and Koetter [8] recently introduced a coordinate descent algorithm for decoding LDPC codes. There are several key differences between MPLP and their method. First, we consider the general MAP problem and not only the LDPC model. Second, MPLP generalizes to non pairwise interactions. Third, NMPLP optimizes a set of edges simultaneously, whereas in [8] one edge is optimized at a time (as in the EMPLP updates), resulting in slower convergence. Finally, coordinate descent was also recently considered in the context of marginal estimation. In [3] the authors present a coordinate descent algorithm for a variational bound on the partition function. Their approach uses similar ideas to the MPLP dual , but importantly does not obtain a closed for solution for the coordinates. Instead, a *gradient like* step is taken at every iteration to decrease the dual objective.

8 Experiments

We compared NMPLP to three other message passing algorithms:[3] Tree-Reweighted max-product (TRMP), standard max-product (MP), and GEMPLP. For MP and TRMP we used the standard approach of damping messages using a factor of $\alpha = 0.5$. We ran all algorithms for a maximum of 2000 iterations, and used two measures to compare their convergence times: 1) *hit-time* - At every iteration the beliefs $b_i(x_i)$ can be used to obtain an assignment $\hat{\boldsymbol{x}}$ with value $f(\boldsymbol{x})$. We consider the first iteration at which the maximum value of $f(\hat{\boldsymbol{x}})$ is achieved.[4] 2) *belief-change* - we calculated $e^{b_i(x_i)}$ at every iteration, and normalize it to one. We then find the first iteration for which the maximum change in these normalized beliefs (in $L1$ distance) is smaller than 10^{-4}.[5]

We first experimented with a 10×10 grid graph, with 5 values per state. The function $f(\boldsymbol{x})$ was a Potts model: $f(\boldsymbol{x}) = \sum_{ij \in E} \theta_{ij} \mathcal{I}(x_i = x_j) + \sum_{i \in V} \theta_i(x_i)$. The values for θ_{ij} and $\theta_i(x_i)$ were randomly drawn from $[-c_I, c_I]$ and $[-c_F, c_F]$ respectively, and we used values of c_I and c_F in the range range $[0.1, 2.1]$ (with intervals of 0.25). The clusters for GEMPLP were the faces of the graph [12]. To see if NMPLP converges to the LP solution we also used an LP solver to solve the LP relaxation. We found that the the normalized difference between NMPLP and LP objective was at most 10^{-3} (median 10^{-7}), suggesting that NMPLP typically converged to the LP solution. Fig. 2 (top row) shows the results for the three algorithms. It can be seen that while all algorithms obtain similar $f(\boldsymbol{x})$ values, NMPLP has better *hit-time* than TRMP and comparable *belief-change* (see median), and MP does not converge in many cases (see caption). GEMPLP is comparable to MPLP in *hit-time*, although each iteration is more costly. The values for GEMPLP are considerably better than NMPLP. In fact, in 95% of the cases the normalized difference between the GEMPLP objective and the $f(\boldsymbol{x})$ value was less than 10^{-5}, suggesting that the exact MAP solution was found.

We next applied the algorithms to the real world problems of protein design. In [11], Yanover et al. show how these problems can be formalized in terms of finding a MAP in an appropriately constructed graphical model.[6] We used all algorithms except GNMPLP (since there is no natural

[3] As expected, NMPLP was faster than EMPLP so only NMPLP results are given.

[4] This is clearly a post-hoc measure since it can only be obtained after the algorithm has exceeded its maximum number of iterations. However, it is a reasonable algorithm-independent measure of convergence.

[5] Beliefs for all algorithms were scaled such that they agree on a graph with one edge.

[6] Data available from http://jmlr.csail.mit.edu/papers/volume7/yanover06a/Rosetta_Design_Dataset.tgz

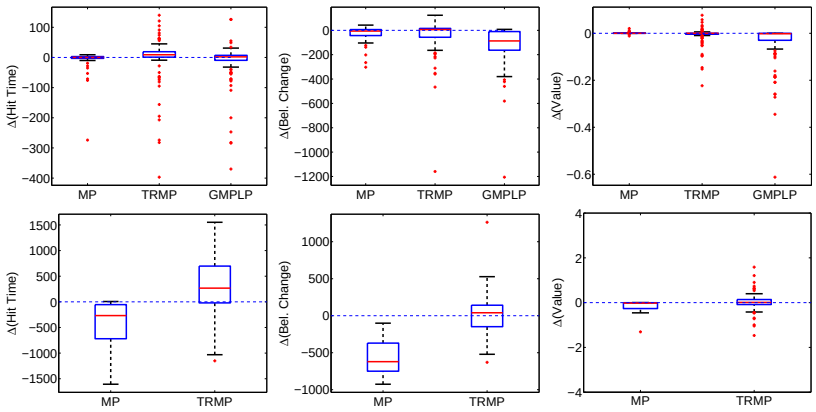

Figure 2: Evaluation of message passing algorithms on a Potts model (top row) and protein design problems (bottom row). Left column: box-plot (horiz. red line indicates median) of the difference between the *hit-time* for the other algorithms and NMPLP. Middle: box-plot of the difference between the *belief-change* time for the other algorithms and NMPLP . Right: box-plot of normalized difference between the value of $f(\boldsymbol{x})$ for NMPLP and the other algorithms. Thus, figures are such that better MPLP performance yields positive values on the Y axis. Max-product converged on 57% of the cases for the Potts model, and on 10% for the protein problems. Only convergent max-product runs are shown in the box-plots.

choice for clusters in this case) to approximate the MAP solution on the 97 models used in [11]. The graph structure in these models is irregular with number of states per variable $2-81$. Fig. 2 (bottom) shows results for all the design problems. In this case the majority of MP runs did not converge, and NMPLP was better than TRMP in terms of *hit-time* and comparable in *belief change* and value.

9 Conclusion

We presented a convergent algorithm for MAP approximation that is based on block coordinate descent of the MAP-LP relaxation dual. The algorithm can also be extended to cluster based functions, which result empirically in improved MAP estimates. This is in line with the observations in [12] that generalized belief propagation algorithms can result in significant performance improvements. However generalized max-product algorithms [12] are not guaranteed to converge whereas GMPLP is. Furthermore, the GMPLP algorithm does not require a region graph and only involves intersection between pairs of clusters. In conclusion, MPLP has the advantage of resolving the convergence problems of max-product while retaining its simplicity, and offering the theoretical guarantees of LP relaxations. We thus believe it should be useful in a wide array of applications.

A Derivation of the dual

Before deriving the dual, we first express the constraint set $\mathcal{M}_L(G)$ is a slightly different way. The definition of $\mathcal{M}_L(G)$ in Sec. 2 uses a single distribution $\mu_{ij}(x_i, x_j)$ for every $ij \in E$. In what follows, we use *two* copies of this pairwise distribution for every edge, which we denote $\bar{\mu}_{ij}(x_i, x_j)$ and $\bar{\mu}_{ji}(x_j, x_i)$, and we add the constraint that these two copies both equal the *original* $\mu_{ij}(x_i, x_j)$. For this extended set of pairwise marginals, we consider the following set of constraints which is clearly equivalent to $\mathcal{M}_L(G)$. On the rightmost column we give the dual variables that will correspond to each constraint.

$$\begin{array}{lll|l}
\bar{\mu}_{ij}(x_i, x_j) = \mu_{ij}(x_i, x_j) & \forall ij \in E, x_i, x_j & \beta_{ij}(x_i, x_j) \\
\bar{\mu}_{ji}(x_j, x_i) = \mu_{ij}(x_i, x_j) & \forall ij \in E, x_i, x_j & \beta_{ji}(x_j, x_i) \\
\sum_{\hat{x}_i} \bar{\mu}_{ij}(\hat{x}_i, x_j) = \mu_i(x_j) & \forall ij \in E, ji \in E, x_j & \lambda_{ij}(x_j) \\
\sum_{x_i} \mu_i(x_i) = 1 & \forall i \in V & \delta_i \\
\bar{\mu}_{ij}(x_i, x_j) \geq 0 \ , \ \mu_i(x_i) \geq 0 & \forall i \in V, \forall ij \in E, ji \in E, x_i, x_j &
\end{array} \tag{6}$$

We denote the set of $(\boldsymbol{\mu}, \bar{\boldsymbol{\mu}})$ satisfying these constraints by $\bar{\mathcal{M}}_L(G)$. We can now state an LP that is equivalent to MAPLPR, only with an extended set of variables and constraints. The equivalent

problem is to maximize $\boldsymbol{\mu} \cdot \boldsymbol{\theta}$ subject to $(\boldsymbol{\mu}, \bar{\boldsymbol{\mu}}) \in \bar{\mathcal{M}}_L(G)$ (note that the objective uses the *original* $\boldsymbol{\mu}$ copy). LP duality transformation of the extended problem yields the following LP

$$\begin{array}{lll} \min & \sum_i \delta_i & \\ s.t. & \lambda_{ij}(x_j) - \beta_{ij}(x_i, x_j) \geq 0 & \forall ij, ij \in E, x_i, x_j \\ & \beta_{ij}(x_i, x_j) + \beta_{ji}(x_j, x_i) = \theta_{ij}(x_i, x_j) & \forall ij \in E, x_i, x_j \\ & -\sum_{k \in N(i)} \lambda_{ki}(x_i) + \delta_i \geq 0 & \forall i \in V, x_i \end{array} \tag{7}$$

We next simplify the above LP by eliminating some of its constraints and variables. Since each variable δ_i appears in only one constraint, and the objective minimizes δ_i it follows that $\delta_i = \max_{x_i} \sum_{k \in N(i)} \lambda_{ki}(x_i)$ and the constraints with δ_i can be discarded. Similarly, since $\lambda_{ij}(x_j)$ appears in a single constraint, we have that for all $ij \in E, ji \in E, x_i, x_j$ $\lambda_{ij}(x_j) = \max_{x_i} \beta_{ij}(x_i, x_j)$ and the constraints with $\lambda_{ij}(x_j), \lambda_{ji}(x_i)$ can also be discarded. Using the eliminated δ_i and $\lambda_{ji}(x_i)$ variables, we obtain that the LP in Eq. 7 is equivalent to that in Eq. 3. Note that the objective in Eq. 3 is convex since it consists of point-wise maxima of convex functions.

B Proof of Proposition 1

We wish to minimize f in Eq. 4 subject to the constraint that $\beta_{ij} + \beta_{ji} = \theta_{ij}$. Rewrite f as

$$f(\beta_{ij}, \beta_{ji}) = \max_{x_i, x_j} \left[\lambda_i^{-j}(x_i) + \beta_{ji}(x_j, x_i)\right] + \max_{x_i, x_j} \left[\lambda_j^{-i}(x_j) + \beta_{ij}(x_i, x_j)\right] \tag{8}$$

The sum of the two arguments in the max is $\lambda_i^{-j}(x_i) + \lambda_j^{-i}(x_j) + \theta_{ij}(x_i, x_j)$ (because of the constraints on $\boldsymbol{\beta}$). Thus the minimum must be greater than $\frac{1}{2} \max_{x_i, x_j} \left[\lambda_i^{-j}(x_i) + \lambda_j^{-i}(x_j) + \theta_{ij}(x_i, x_j)\right]$. One assignment to $\boldsymbol{\beta}$ that achieves this minimum is obtained by requiring an equalization condition:[7]

$$\lambda_j^{-i}(x_j) + \beta_{ij}(x_i, x_j) = \lambda_i^{-j}(x_i) + \beta_{ji}(x_j, x_i) = \frac{1}{2}\left(\theta_{ij}(x_i, x_j) + \lambda_i^{-j}(x_i) + \lambda_j^{-i}(x_j)\right) \tag{9}$$

which implies $\beta_{ij}(x_i, x_j) = \frac{1}{2}\left(\theta_{ij}(x_i, x_j) + \lambda_i^{-j}(x_i) - \lambda_j^{-i}(x_j)\right)$ and a similar expression for β_{ji}. The resulting $\lambda_{ij}(x_j) = \max_{x_i} \beta_{ij}(x_i, x_j)$ are then the ones in Prop. 1.

References

[1] M. Bayati, D. Shah, and M. Sharma. Maximum weight matching via max-product belief propagation. *IEEE Trans. on Information Theory (to appear)*, 2007.

[2] D. P. Bertsekas, editor. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 1995.

[3] A. Globerson and T. Jaakkola. Convergent propagation algorithms via oriented trees. In *UAI*. 2007.

[4] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1568–1583, 2006.

[5] V. Kolmogorov and M. Wainwright. On the optimality of tree-reweighted max-product message passing. In *21st Conference on Uncertainty in Artificial Intelligence (UAI)*. 2005.

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

[7] B. Taskar, S. Lacoste-Julien, and M. Jordan. Structured prediction, dual extragradient and bregman projections. *Journal of Machine Learning Research*, pages 1627–1653, 2006.

[8] P.O. Vontobel and R. Koetter. Towards low-complexity linear-programming decoding. In *Proc. 4th Int. Symposium on Turbo Codes and Related Topics*, 2006.

[9] M. J. Wainwright, T. Jaakkola, and A. S. Willsky. Map estimation via agreement on trees: message-passing and linear programming. *IEEE Trans. on Information Theory*, 51(11):1120–1146, 2005.

[10] Y. Weiss, C. Yanover, and T. Meltzer. Map estimation, linear programming and belief propagation with convex free energies. In *UAI*. 2007.

[11] C. Yanover, T. Meltzer, and Y. Weiss. Linear programming relaxations and belief propagation – an empirical study. *Journal of Machine Learning Research*, 7:1887–1907, 2006.

[12] J.S. Yedidia, W.T. W.T. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Trans. on Information Theory*, 51(7):2282–2312, 2005.

[7] Other solutions are possible but may not yield some of the properties of MPLP.