Dynamic Tree Block Coordinate Ascent

Daniel Tarlow DTARLOW@CS.TORONTO.EDU
University of Toronto

Dhruv Batra DBATRA@TTIC.EDU
TTI Chicago

Pushmeet Kohli PKOHLI@MICROSOFT.COM
Microsoft Research Cambridge

Vladimir Kolmogorov V.KOLMOGOROV@CS.UCL.AC.UK
University College London

Abstract

This paper proposes a novel Linear Programming (LP) based algorithm, called Dynamic Tree-Block Coordinate Ascent (DT-BCA), for performing maximum *a posteriori (MAP)* inference in probabilistic graphical models. Unlike traditional message passing algorithms, which operate uniformly on the whole factor graph, our method dynamically chooses regions of the factor graph on which to focus message-passing efforts. We propose two criteria for selecting regions, including an efficiently computable upper-bound on the increase in the objective possible by passing messages in any particular region. This bound is derived from the theory of primal-dual methods from combinatorial optimization, and the forest that maximizes the bounds can be chosen efficiently using a maximum-spanning-tree-like algorithm. Experimental results show that our dynamic schedules significantly speed up state-of-the-art LP-based message-passing algorithms on a wide variety of real-world problems.

1. Introduction

Performing maximum *a posteriori* (MAP) inference in probabilistic models such as Markov and Conditional Random Fields (MRFs and CRFs) is a fundamental problem in Machine Learning, and plays an important role in the solution of many labeling problems in Computer Vision such as image segmentation, stereo and optical flow. MAP inference is often formulated as the problem of minimizing the *energy* of the probabilistic model, which is the negative logarithm of the posterior distribution. Although computing the MAP solution in general models is known to be an NP-hard problem, significant advances have been made in obtaining good approximate solutions. In particular, formulations based on Linear Programming (LP) relaxations have inspired fast convergent message-passing algorithms for MAP inference. Despite recent advances, models using internet-scale datasets, high-resolution images and video, and large-scale learning remain computationally demanding beyond practical limits (without significant investment in hardware). These models have motivated research on faster inference algorithms.

Energy-Oblivious Methods. Many recently proposed LP-based algorithms such as Tree-reweighted message passing (Wainwright et al., 2005; Kolmogorov, 2006), Max-sum diffusion (Werner, 2007), MPLP (Globerson & Jaakkola, 2007), and Tree-based coordinate descent (Sontag & Jaakkola, 2009) operate by passing messages uniformly on the whole factor graph using a particular schedule. These methods have schedules that depend only on the structure of the probabilistic model and not on its posterior distribution (energy). In other words, they are *energy-oblivious*. Furthermore, these schedules are static; *i.e.*, they remain fixed during the operation of the algorithm. This is a suboptimal strategy since it can—and often does—lead to wasted computation: effort is expended on regions of the graph where additional (messages) computation cannot lead to significant improvement in solution quality.

Energy-Aware Methods. The last decade has seen the increasing use of efficient max-flow based algorithms for minimizing certain classes of energy functions (Boykov et al., 2001; Kolmogorov & Zabih, 2004). This has primarily been due to their low running time compared to message-passing algorithms (Szeliski et al., 2006). It has also been shown that flow-based

Appearing in *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, WA, USA, 2011.

algorithms, like message-passing algorithms, work by reparameterizing the energy functions (Kohli & Torr, 2007). In this interpretation, flow can be seen as a encoding of the messages, and the sequence of paths used for passing flow can be seen as particular schedules for message passing (Tarlow et al., 2011b).

We believe that the reason for better performance of flow-based algorithms (compared to conventional message-passing algorithms like BP and TRW-S) lies in their use of sophisticated energy-aware dynamic schedules. These schedules are *energy-aware* in the sense that they depend not only on the structure of the factor graph but also on the exact form and values of the energy function; and they are dynamic in the sense that the schedule is not fixed beforehand, but is computed at run-time and changes in each iteration.

Contributions. We believe *dynamic*, *energy-aware* schedules are crucial to the development of the next generation of computationally efficient algorithms. In this work, we develop a novel method for choosing regions of the problem graph on which to focus inference efforts. We propose two region-scoring functions – Local Primal-Dual Gap (LPDG) and Weak Tree Agreement score (WTA) – that are based on the theory of Primal-Dual methods from combinatorial optimization. These two scoring functions are natural generalizations of complementary slackness conditions and virtual arc consistency in the primal-dual LP formulation of MAP. LPDG is an upper-bound on the increase in the objective possible by passing messages in any particular region. Most importantly, both score can be computed with low overhead, and the forest that maximizes the score can be chosen efficiently using a maximum-spanning-tree-like algorithm. Experimental results show that our dynamic schedules significantly speed up state-of-the-art LP-based message-passing algorithms on a wide variety of real-world problems.

2. Related Work

Our work is perhaps most directly related to message scheduling approaches that have been applied to max-product belief propagation, such as Residual Belief Propagation (RBP) (Elidan et al., 2006) and its followups (Sutton & McCallum, 2007). Also related is Chandrasekaran et al. (2007), which considers dynamic tree-based schedules for continuous-valued graphical models. Though Tree-Block Coordinate Ascent (TBCA) algorithms have their roots in max-product, the algorithms and scheduling considerations are actually quite different. Perhaps most importantly, unlike in ordinary max-product, the update for TBCA depends on the entire block that is chosen. Thus, while the motivations of our work are similar to those of RBP, we focus on methods that are applicable to the LP-based message passing algorithms.

Another related algorithm is the Augmenting DAG algorithm (Werner, 2007). This algorithm uses a a measure similar to ours (WTA) in order to choose parameters to update. There are two important differences: first, Augmenting DAG requires a search at each iteration to choose an update; and second, the update is not a full block-coordinate ascent step over the subset of variables involved in the update. In our work, we avoid running a search at each iteration (which reduces overhead), and we make a full block-coordinate ascent update. Finally, there are other algorithms such as TRW-S (Kolmogorov, 2006) that operate using a static update schedule. Implementations of these algorithms have been optimized to run very quickly, but they make no attempt to identify regions of the graph to update. As we show later, in certain cases this leads to an order of magnitude more updates than necessary.

3. Preliminaries

Notation. For any positive integer n, let $[n]$ be shorthand for the set $\{1, 2, \dots, n\}$. We consider a set of discrete random variables $\mathcal{X} = \{x_1, x_2, \dots, x_n\}$, each taking value in a finite label set, $x_i \in X_i$. For a set $A \subseteq [n]$, we use x_A to denote the tuple $\{x_i \mid i \in A\}$, and X_A to be the joint label space $\times_{i \in A} X_i$. For ease of notation, we use x_{ij} as a shorthand for $x_{\{i,j\}}$.

MAP. Let $G = (\mathcal{V}, \mathcal{E})$ be a graph defined over these variables, *i.e.* $\mathcal{V} = [n]$, $\mathcal{E} \subseteq \binom{\mathcal{V}}{2}$, and let $\theta_A : X_A \to \mathbb{R}$, $\forall A \in \mathcal{V} \cup \mathcal{E}$ be functions defining the cost/energy at each node and edge for the labeling of variables in scope. Let $\vec{\mathcal{E}} = \{(i \to j), (j \to i) \mid \{i, j\} \in \mathcal{E}\}$ be a set holding directed edges for each undirected edge. The goal of MAP inference can then be succinctly written as: $\min_{\mathcal{X} \in X_{\mathcal{V}}} \sum_{i \in \mathcal{V}} \theta_i(x_i) + \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(x_i, x_j)$.

Schlesinger's LP. It is well-known (Wainwright & Jordan, 2008) that the above discrete optimization problem can be written as a linear programming problem over the so-called marginal polytope $\mathcal{M}(G)$:

$$\min_{\boldsymbol{\mu} \in \mathcal{M}(G)} \boldsymbol{\theta} \cdot \boldsymbol{\mu} \tag{1}$$

$$\mathcal{M}(G) = \left\{ \boldsymbol{\mu} \;\middle|\; \exists p(\mathcal{X}) \; s.t. \; \begin{array}{c} \mu_i(x_i) = \sum_{X_{\mathcal{V} \setminus i}} p(\mathcal{X}) \\ \mu(x_i, x_j) = \sum_{X_{\mathcal{V} \setminus i,j}} p(\mathcal{X}) \end{array} \right\},$$

where $\boldsymbol{\theta} \cdot \boldsymbol{\mu} \doteq \sum_{i \in \mathcal{V}} \theta_i(x_i)\mu_i(x_i) + \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(x_i, x_j)\mu_{ij}(x_i, x_j)$, $p(\mathcal{X})$ is a Gibbs distribution that factorizes over the graph G, and $\boldsymbol{\mu}$ is a vector holding node and edge marginal distributions, *i.e.*, $\boldsymbol{\mu} = \{\mu_i(\cdot), \mu_e(\cdot) \mid i \in \mathcal{V}, e \in \mathcal{E}\}$.

Problem (1) is NP-hard to solve in general and the standard LP relaxation of the MAP problem (1),

$$
\begin{array}{llll}
\min\limits_{\boldsymbol{\mu}} & \boldsymbol{\theta}\cdot\boldsymbol{\mu} & \max\limits_{\boldsymbol{h},\boldsymbol{y}} & \mathbf{1}\cdot\boldsymbol{h} \\
s.t. & \sum\limits_{x_A}\mu_A(x_A)=1, & s.t. & h_A\in\mathbb{R}, \quad \forall A\in\mathcal{V}\cup\mathcal{E} \\
 & \sum\limits_{x_i}\mu_{ij}(x_i,x_j)=\mu_j(x_j), & & y_{ij\to j}(x_j)\in\mathbb{R}, \quad \forall(i\to j)\in\vec{\mathcal{E}} \\
 & \mu_i(x_i)\ge 0, & & h_i\le\tilde{\theta}_i(x_i)\doteq\theta_i(x_i)+\sum\limits_{e|e\in\mathcal{E},i\in e}y_{ji\to i}(x_i) \quad \forall i\in\mathcal{V},x_i\in X_i \\
 & \mu_{ij}(x_i,x_j)\ge 0. & & h_{ij}\le\tilde{\theta}_{ij}(x_i,x_j)\doteq\theta_{ij}(x_i,x_j)-y_{ij\to j}(x_j)-y_{ji\to i}(x_i), \quad \forall\{i,j\}\in\mathcal{E},x_i,x_j.
\end{array}
$$

Figure 1. Linear programming relaxation. Both the primal and dual LPs are shown.

also known as Schlesinger's bound (Schlesinger, 1976; Werner, 2007), is given by $\min_{\boldsymbol{\mu}\in\mathcal{L}(G)}\boldsymbol{\theta}\cdot\boldsymbol{\mu}$, with

$$
\mathcal{L}(G)=\left\{\boldsymbol{\mu}\ge 0\;\middle|\;\begin{array}{c}\sum_{x_i}\ \mu_i(x_i)=1,\\ \sum_{x_i,x_j}\ \mu_{ij}(x_i,x_j)=1,\\ \sum_{x_i}\ \mu_{ij}(x_i,x_j)=\mu_j(x_j)\end{array}\right\}, \quad (2)
$$

where $\boldsymbol{\mu}$ is now a vector of local *pseudomarginals*, which are forced to be "edge-consistent", meaning that the marginalized edge beliefs agree with the node beliefs (marginalization constraints). Optimizing over (2) is possible in polytime, and by $\mathcal{L}(G)\supseteq\mathcal{M}(G)$ this provides a lower bound on the optimal objective.

3.1. Dual Formulation and Reparameterization

Fig. 3 shows both the primal and dual programs corresponding to the LP-relaxation over $\mathcal{L}(G)$. Vector $\tilde{\theta}$ used in Fig. 3 is called a *reparameterization* of θ; its components are defined as

$$
\tilde{\theta}_i(x_i)\doteq\theta_i(x_i)+\sum_{e|e\in\mathcal{E},i\in e}y_{ji\to i}(x_i), \quad (3)
$$

$$
\tilde{\theta}_{ij}(x_i,x_j)\doteq\theta_{ij}(x_i,x_j)-y_{ij\to j}(x_j)-y_{ji\to i}(x_i). \quad (4)
$$

Intuitively, $\tilde{\theta}_A(x_A)$ can be thought of as the new cost for state x_A at node/edge A after incorporating the dual variables $\{y_{ji\to i}(x_i)\}$. Energy is preserved under reparameterization, *i.e.*, for any labeling $\mathcal{X}$, we have $\sum_{A\in\mathcal{V}\cup\mathcal{E}}\tilde{\theta}_A(x_A)=\sum_{A\in\mathcal{V}\cup\mathcal{E}}\theta_A(x_A)$. Note, however, that this does not imply that $\tilde{\theta}_A(x_A)=\theta_A(x_A)$.

3.2. Weak Tree Agreement

The stopping criterion for message passing-based LP algorithms is given by the *weak tree agreement* (WTA) condition (Kolmogorov, 2006). Its equivalent for the dual problem in Fig. 3 is known as *Virtual Arc Consistency* (VAC) (Cooper et al., 2008). A reparameterized vector $\tilde{\theta}$ is said to satisfy WTA if there exist non-empty subsets $D_i\subseteq X_i$ for each node $i\in\mathcal{V}$ such that

$$
\tilde{\theta}_i(x_i)=h_i^* \qquad \forall i\in\mathcal{V},x_i\in D_i \quad (5a)
$$

$$
\min_{x_j\in D_j}\theta_{ij}(x_i,x_j)=h_{ij}^* \qquad \forall(i\to j)\in\vec{\mathcal{E}},x_i\in D_i \quad (5b)
$$

where we denoted $h_A^*=\min_{x_A}\tilde{\theta}_A(x_A)$. If $|D_i|=1$ for all $i\in\mathcal{V}$ then the WTA condition is equivalent to the *strong tree agreement* (Wainwright & Jordan, 2008); and the labeling encoded by sets D_i is then the global minimum of the energy.

3.3. Tree Block Coordinate Ascent

In this paper we follow the *tree block coordinate ascent (TBCA)* scheme of Sontag & Jaakkola (2009) for maximizing the dual objective. In each iteration of this scheme we first select a subset of edges $T\subseteq\mathcal{E}$ that form a forest. We then we fix all dual variables $y_{ji\to i}(x_i)$ outside T and update remaining dual variables $y_{ji\to i}(x_i)$, $\{i,j\}\in T$ using the sequential tree-block update algorithm of Sontag & Jaakkola (2009). It requires $2|T|$[1] edge-reparameterization operations[2] and produces an *optimal* reparameterization for forest T, i.e. a reparamaterization that maximizes the dual objective function under the restriction above. Clearly, in this approach the dual objective $\sum_{i\in\mathcal{V}}h_i+\sum_{(i,j)\in\mathcal{E}}h_{ij}$ (which is a lower bound on the energy) never decreases. This scheme has the same convergence guarantee as the TRW-S algorithm (Kolmogorov, 2006): the sequence of vectors $\tilde{\theta}^t$ has a limit point $\tilde{\theta}^*$ that satisfies the WTA condition (assuming that each edge is covered by T every constant number of iterations and the sequence of $\tilde{\theta}^t$ is bounded).

Remark 1. The scheme above is actually slightly different from the overall scheme described by Sontag & Jaakkola (2009) in section 4.1. Namely, at each iteration Sontag *et al.* propose to "split" $\tilde{\theta}$ between overlapping trees, perform the block update for each tree in parallel, and then combine the reparameterized vectors. In contrast, we select a single forest T at each step and fully assign all covered unary and pairwise potentials to T. Note, however, that in the end of section 4.1 Sontag *et al.* mention that "sequential" tree block updates (which we believe to coincide with the scheme that we use) empirically converge more quickly than the parallel updates they used.

[1] Fig. 2 in (Sontag & Jaakkola, 2009) may appear to use $3|T|$ distance transform operations, but the max operation used for computing $\delta_{ji}(x_i)$ in Fig. 2 is the same as the max operation in step 2, so we do not need to compute it again.

[2] Following the notation of Felzenszwalb & Huttenlocher (2004), we refer to these as Distance Transforms.

4. Measuring Local Agreement

We now present two methods for identifying regions to pass messages on. At a high level, our goal is to come up with measures that identify how locally consistent a region of the graph is. We will then pass messages on regions that are locally inconsistent. Our two measures are: a) Local Primal Dual Gap (LPDG), which was first proposed by Batra et al. (2011) and is derived from the complementary slackness conditions in the primal-dual LPs; and b) a score based on the Weak Tree Agreement (WTA) conditions of Kolmogorov (2006).

4.1. Local Primal Dual Gap

Complementary Slackness. Let $\{\mu_i^*(\cdot), \mu_{ij}^*(\cdot,\cdot)\}$ and $\{h_i^*, h_{ij}^*, y_{ij\to j}^*(\cdot), y_{ji\to i}^*(\cdot)\}$ be a pair of optimal primal and dual solutions of the LPs in Fig. 3. Let $\tilde\theta_i(\cdot), \tilde\theta_{ij}(\cdot,\cdot)$ be the reparameterized node/edge vectors corresponding to these dual variables (as given by (3),(4)). The complementary slackness condition (Bertismas & Tsitsiklis, 1997) for this pair of LPs is:

$$\mu_A(x_A)\cdot\left(\tilde\theta_A(x_A) - \min_{\hat x_A}\tilde\theta_A(\hat x_A)\right) = 0, \qquad (6)$$

where $A \in \mathcal{V}\cup\mathcal{E}$. Batra et al. (2011) defined LPDG for a node/edge as a generalization of this complementary slackness condition:

Definition (Local Primal-Dual Gap). *Given a reparameterized cost vector $\tilde\theta$ and a (possibly non-optimal) integral primal labeling $\mathcal{X}^p = \{x_1^p, x_2^p, \ldots, x_n^p\}$, the* local primal-dual gap *$l(A)$ for each node/edge $A \in \mathcal{V}\cup\mathcal{E}$ is:*

$$l(A) = \underbrace{\tilde\theta_A(x_A^p)}_{\text{primal}} - \underbrace{\min_{x_A}\ \tilde\theta_A(x_A)}_{\text{dual}}. \qquad (7)$$

Comparing (6) and (7), we can see that intuitively LPDG quantifies by how much the complementary slackness condition is violated, and attributes violations to individual potentials. More precisely, we state the following properties of LPDG:

Proposition 1 *Distributed Primal-Dual Gap: If $P = \sum_{A\in\mathcal{V}\cup\mathcal{E}}\theta_A(x_A^p)$ is the energy of the primal labeling, and $D = \mathbf{1}\cdot\boldsymbol{h}$ the dual value, then the sum of LPDG for all cliques is the primal-dual gap,* i.e., *$\sum_{A\in\mathcal{V}\cup\mathcal{E}} l(A) = P - D$.*

Proposition 2 *Slackness: If LPDG for no node/edge is strictly positive,* i.e. *$\max_{A\in\mathcal{V}\cup\mathcal{E}} l(A) = 0$, then complementary slackness conditions hold and the LP is tight,* i.e. *we have found the MAP solution.*

Proof Sketches. Prop. 1 just utilizes that $h_A = \min_{x_A}\tilde\theta(x_A)$ (for the dual) and conservation of total energy under reparameterization (for the primal). Prop. 2 follows from complementary slackness.

4.2. WTA edge measure

While the LPDG measure is quite simple and easy to compute, it can easily overestimate the "usefulness" of an edge. If, for example, the LP relaxation of the energy is not tight then some edges will always have nonzero LPDG value, so the algorithm will try to perform reparameterizations even in regions where the WTA condition has been reached. We now define an alternative measure which is inspired by the WTA condition (5) and can potentially overcome this problem.

This alternative measure is based on a subset of labels $D_i \subseteq X_i$ for each node i. One way to compute these subsets would be to choose, for each hyperedge A, the labelings x_A such that $\tilde\theta_A(x_A)$ is sufficiently close to h_A (say, within some $\epsilon \geq 0$). However, this requires a significant computation effort. To avoid such computations, we choose D_i for node i locally by including labels that were assigned to this node during the last R iterations. Note, we only count iterations in which node i was actually involved in reparamterizations, *i.e.* covered by the forest T in which the messages were passed. We then set

$$wta(i,j) = e_i + e_j + \max(e_{ij}, e_{ji}), \qquad (8)$$

e_i and e_{ij} are *temporal* local agreement measures:

$$e_i = \max_{x_i\in D_i}\tilde\theta_i(x_i) - \min_{x_i}\tilde\theta_i(x_i), \quad \text{and} \qquad (9)$$

$$e_{ij} = \max_{x_i\in D_i}\min_{x_j\in D_j}\tilde\theta_{ij}(x_i,x_j) - \min_{x_i,x_j}\tilde\theta_{ij}(x_i,x_j). \qquad (10)$$

5. Dynamic TBCA

Our proposed algorithm Dynamic Tree-Block Coordinate Ascent (DTBCA) uses the sequential TBCA formulation described in Section 3. The pseudo-code for the method is listed as Algorithm 1. At each iteration, we first isolate the set of nodes and edges in the graph whose scores have been invalidated by the updates in the previous iteration. The local consistency measures (presented in Section 4) for these 'dirty' nodes and edges are then recomputed. The exact procedure for determining the set of dirty nodes and edges is discussed section in Section 5.1. ADD-TO-HISTORY stores in D_i the most recent R settings of x_i^p. The setting $R = 1$ gives us the LPDG measure, while larger values of R imply the WTA measure. After combining node and edge scores into a single undirected score per edge, we choose a forest by running a modified version of Kruskal's maximum weight spanning tree algorithm. Kruskal's algorithm sorts edges by descending weight then greedily adds the largest weight edge that does not create a cycle. Typically the algorithm proceeds until $n-1$ edges have been used, but we instead terminate early if we reach an edge score of less than ϵ

Algorithm 1 Dynamic Tree-Block Coordinate Ascent

```
$\hat{\mathcal{V}} \leftarrow \mathcal{V}$   {Dirty nodes}
$\hat{\mathcal{E}} \leftarrow \mathcal{E}$   {Dirty edges (see Sec. 5.1 for details)}
$R \leftarrow$ RUN-LPDG ? $1 : R_{WTA}$   {History size}
for $t = 1 : t_{\max}$ do
  for $i \in \hat{\mathcal{V}}$ do {Node scores}
    $x_i^p \leftarrow \arg\min_{x_i} \tilde{\theta}_i(x_i)$
    Add-To-History($x_i^p, D_i, R$)
    $e_i \leftarrow \max_{x_i \in D_i} \tilde{\theta}_i(x_i) - \min_{x_i} \tilde{\theta}_i(x_i)$
  end for
  for $(i,j) \in \hat{\mathcal{E}}$ do {Directed edge scores}
    $h_{ij} \leftarrow \min_{x_i, x_j} \tilde{\theta}_{ij}(x_i, x_j)$
    $e_{ij} \leftarrow \max_{x_i \in D_i} \min_{x_j \in D_j} \tilde{\theta}_{ij}(x_i, x_j) - h_{ij}$
    $e_{ji} \leftarrow \max_{x_j \in D_j} \min_{x_i \in D_i} \tilde{\theta}_{ij}(x_i, x_j) - h_{ij}$
  end for
  for $(i,j) \in \mathcal{E}$ do {Undirected edge scores}
    $w_{ij} \leftarrow \max(e_{ij}, e_{ji}) + e_i + e_j$
  end for
  $T \leftarrow$ Kruskal-Forest($\boldsymbol{w}$)
  $\tilde{\theta} \leftarrow$ Reparameterize-Forest($T, \tilde{\theta}$)
end for
```

($= 10^{-10}$). This modified Kruskal's algorithm is referred to as Kruskal-Forest in Algorithm 1.

Finally, given the forest T, we apply TBCA updates on maximal trees in T. Updates on distinct trees do not interact and thus can be performed in parallel without affecting the resulting reparameterization.

5.1. Caching

An important property of DTBCA is that it may operate on only a small fraction of the edges in the graph at each iteration. Our formulation is amenable to caching many computations. In particular, we only need to update e_i if $\tilde{\theta}_i$ changed in the previous iteration, and we only need to update e_{ij} and e_{ji} if either edge (i,j) was used in T in the previous iteration or if one of the sets D_i or D_j changed in the previous iteration. Further, we only consider edges with nonzero w_{ij} in Kruskal-Forest, so the sort operation typically only needs to be performed on a small subset of edges.

Label history sets D are typically small, so unless pairwise potentials admit fast distance transforms (e.g., Potts models), the dominant overhead is the minimizations in computing e_{ij} that require considering $|X_i| \cdot |X_j|$ entries in $\tilde{\theta}_{ij}$. One of these is required per iteration per dirty edge; the minimization can be shared between the e_{ij} and e_{ji} computation.

6. Experiments

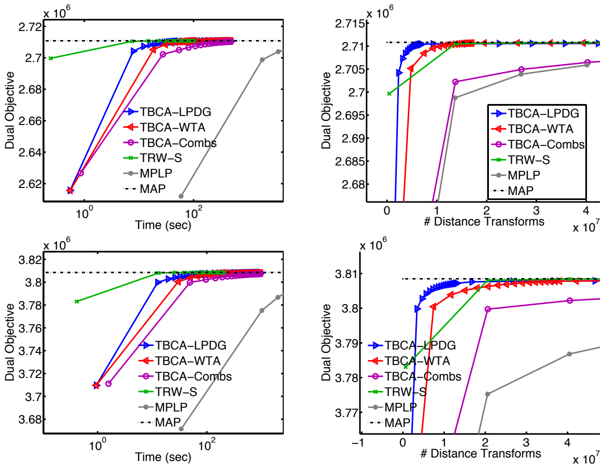
Problem Instances. We tested our algorithm on the problems of stereo vision, object-category segmentation and protein design. These problems have been extensively studied and are known to yield difficult discrete optimization instances of a large scale (hundreds of thousands of nodes and almost a million edges). Our experiments show that our proposed TBCA methods solve the LP relaxation faster than baseline schemes for all the above problems.

We also conducted a "Dynamic Segmentation" experiment that simulates a branch-and-bound or A^* approach for tightening loose relaxations. Specifically, we first solve a particular relaxation to convergence, then branch on certain variables taking certain states. This requires solving a slightly-updated problem by warm-starting from the converged messages. Our proposed TBCA methods solve the updated problem faster than baselines because they are able to identify the regions in the graphs that are affected by the conditioning.

Baseline Methods. We compare the performance of our methods to standard-TBCA, which uses exactly the same message passing implementation as our methods (TBCA-LPDG, TBCA-WTA) and differs only in the schedule of the messages. Thus the relative performance difference can be directly attributed to the different scheduling choices.

To demonstrate the practical impact or our method, we compare the performance of our method with the well known and widely used TRW-S (Kolmogorov, 2006) and MPLP (Globerson & Jaakkola, 2007) algorithms, both of which are fixed-schedule block coordinate ascent methods with the block being dual variables corresponding to individual nodes and edges. For both these methods, we used the implementations made publicly available by the respective authors. From an implementation perspective, these two baseline are fundamentally different. The TRW-S implementation is orders of magnitude faster than MPLP, which was prohibitively slow on larger problems. However, it is important to tease out the difference between implementation optimizations and "inherent efficiency" of various algorithms. One way to do this is to count the number of Distance Transforms and we measure the number of calls to a native `distance_transform()` function for each baseline. Note that this does include the dominant operations needed to compute the region-scoring measures.

Evaluation Metrics. We evaluate the performance of different methods in two ways: 1) dual-objective vs time and 2) dual-objective vs # distance transforms. Dual objective at each iteration is a lower-bound on the MAP value, and a larger value indicates a tighter relaxation. At each iteration, integral primal labeling are generated by *node-decoding*, *i.e.* independently minimizing the reparameterized energy vectors. The energy of the primal is an upper bound on the MAP and when the primal-dual gap is smaller than a threshold (10^{-4}), we report that the MAP solution has been found. All experiments were performed on a 64-bit 8-Core Macbook Pro with 8GB RAM.



(a) Dual vs Time (b) Dual vs Dist. Trans.

Figure 2. **Stereo experiments.** (a) Dual objective versus time for Tsukuba problem, which is a 288x384 pixel, 16 label Potts model with pairwise strength $\lambda = 20$. (b) Dual objective versus number of distance transforms for same problem. (c-d) Same comparisons, but on Venus problem.

Convergence Heuristics. For the image labeling problems on grid graphs, we found that two heuristic strategies led to faster convergence. First, starting inference with a small number of iterations using a static combs schedule (we use 4 iterations) improved performance. Second, a strategy of "dilating" the edge scores led to faster convergence. The dilation procedure is defined as follows. For each edge $e = \{i, j\}$, a set of "neighboring" edges $\mathcal{N}_e$ $(=\emptyset)$ is created. For each node n adjacent to an edge with non-zero w_{ij}, with small probability (1%) we add to $\mathcal{N}_e$ the horizontal or vertical chain of length $2d$ centered at n (we use d=10). We then add a small score $\epsilon = 10^{-5}$ to all edges in $\mathcal{N}_e$. Intuitively, there are often strong correlations between pixels in a local neighborhood that extends beyond just the immediate neighbors. With the dilation strategy, relevant information can preemptively be brought from neighbors several pixels away.

Edge Improvement Measure. We also experimented with an edge measure inspired by the strategy that Sontag et al. (2008) used to choose regions of the graph to add constraints for tightening the LP relaxation. In this measure, the undirected edge score for $\{i, j\}$ is the increase in the dual objective achieved by optimally reparameterizing the subgraph containing $\tilde{\theta}_{ij}$, $\tilde{\theta}_i$, and $\tilde{\theta}_j$: $\Delta_{ij}^{Dual} = \min_{x_i,x_j} \left[\tilde{\theta}_i(x_i) + \tilde{\theta}_{ij}(x_i, x_j) + \tilde{\theta}_j(x_j)\right] - \left[\min_{x_i} \tilde{\theta}_i(x_i) + \min_{x_i,x_j} \tilde{\theta}_{ij}(x_i, x_j) + \min_{x_j} \tilde{\theta}_j(x_j)\right]$.

This measure has an undesirable theoretical property: it does not capture whether WTA has been achieved. Consider *e.g.*, reparameterizations with zero node potentials: this measure will be 0 for all edges, but WTA may not hold. Empirically, we found it to be inferior to our measures. See Tarlow et al. (2011a) for details.

6.1. Stereo Experiments

Stereo vision involves predicting disparity labels for each pixel given a pair of stereo images. Graphical models for this problem have been proposed by Tappen & Freeman (2003) and extensively studied by Meltzer et al. (2005). We tested on the Tsukuba (288x383, 16 labels) and Venus (383x434, 20 labels) image pairs from the Middlebury Stereo Dataset[3], and used the publicly available energies [4] from Alahari et al. (2010). The graph was a 4-connected grid for which the natural fixed scheduling scheme is to alternate on horizontal and vertical combs, *i.e.* spanning trees containing all horizontal edges connected by a single column of vertical edges (and vice versa).

Fig. 2 shows the results of our experiments. As we can see, both our methods TBCA-LPDG and TBCA-WTA significantly outperform TBCA-Combs, which demonstrates the efficacy of our proposed scoring functions. Interestingly, we note that while TRW-S is the fastest implementation and performs the best in terms of runtime, it performs worse than our methods in terms of number of distance transforms. MPLP performs worst in terms of both metrics.

6.2. Segmentation Experiments

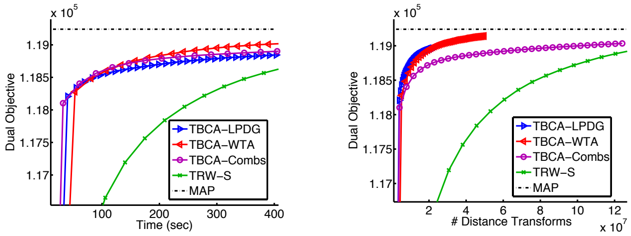
The goal in this application is dense object-category labeling, *i.e.*, to label every image pixel with a particular category label. We worked with some images from the PASCAL Visual Object Category dataset (Everingham et al.), which is considered to be one of the most challenging datasets for this problem. Typical images are 375x500 pixels and the dataset contains 20 foreground and 1 background category. We use the energies from Ladickỳ et al. (2009), who use pairwise energy functions based on Shotton et al. (2009).

We do not show plots for MPLP because it was significantly worse than other methods. For this application, our methods TBCA-LPDG, TBCA-WTA outperform TBCA-static and TRW-S, both in terms of number of distance transforms *and* runtime, in some instances cutting runtime by more than 80%. See Fig. 3.

6.3. Dynamic Segmentation Experiments

In this experiment, we simulate a problem where dynamic inference is used within learning to reuse inference computation between iterations. First, we

[3] http://vision.middlebury.edu/stereo/data/
[4] http://cms.brookes.ac.uk/staff/Karteek/data.tgz



(a) Dual vs Time. (b) Dual vs Dist. Trans.

Figure 3. **Object class labeling experiments.** (a) Dual objective versus time for the segmentation problem shown in the second row of Fig. 4. 375x500 pixel, 21 label 4-connected grid model with pairwise potentials based on object co-occurrence counts. (b) Dual objective versus number of distance transforms for same problem.

run inference on the original problem to convergence (Fig. 4(b)). We then update a subset of unary potentials—which may correspond to updating a weight on a sparsely activated feature—and re-solve the problem. Here, all white pixels in Fig. 4(c) have unary potentials incremented to more strongly prefer being labeled background. We now solve this slightly-updated problem by warm-starting from the converged messages. Fig. 4(f) shows the dual objective (and energy of primal decoding) as a function of number of distance transform calls for this updated problem. Both TRW-S and TBCA-WTA are able to find the MAP (duality gap $< 10^{-4}$), however TBCA-WTA passes orders of magnitude fewer messages than TRW-S (x-axis is in log-scale). The reason for this wide gap is that TBCA-WTA is able to accurately localize the regions where messages need to be passed for convergence, while TRW-S indiscriminately passes messages everywhere in the graph. Fig. 4(e) shows where the messages were passed in a heat-map visualization.

6.4. Protein Design

The goal of this problem to find a sequence of amino-acids that are most stable in a given 3D configuration. We use the energy functions studied by Yanover et al. (2006). This is also a difficult dataset because the LP relaxation is tight only on 2 of the 97 protein instances.

Fig. 5 shows scatter plots for one of the smaller proteins '1bx7'. Each point is a sub-forest of the MRF. X-axis shows the sum of LPDG/WTA measures over it's nodes and edges, and Y-axis shows the improvement in dual-objective by performing a block-coordinate ascent step over this forest block. We can see that our proposed measure are highly correlated with improvements in the dual objective.

In the protein problems, there is not a natural grid structure, so we compare our measures to a static sequential star-based schedule, iterating over nodes and reparameterizing subgraphs containing all edges adjacent to the current node. Our dynamic schedules are faster and use fewer distance transforms. On protein '1bx7', where the relaxation is tight, the WTA schedule converges to the optimum in .39 seconds using 24400 distance transforms. The stars baseline converges in .86 seconds using 53700 distance transforms. On protein '1kth', where the relaxation is not tight, after 500 seconds and roughly 9M distance transforms (time and distance transforms correspond closely between the two in this case), the WTA schedule achieves a dual objective of -115.94, while the stars baseline achieves a worse dual objective of -116.03. We also tried a dynamic experiment similar to Sec. 6.3. Specifically, we considered a variable that was alternating between two labels during message passing (*i.e.*, its label history D_i contained 2 states). We forced it to take one of these labels, and warm-started message-passing with the converged messages. Similar to Sec. 6.3, we found that TBCA-WTA is significantly more efficient than TRW-S, *e.g.* achieves a similar improvement in dual as TRW-S in 80% fewer distance transforms (~200K vs. 1.7M) and 90% less time (5 vs 50 sec).

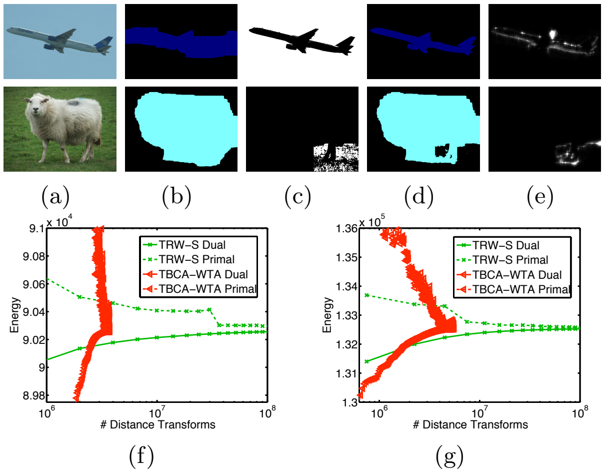
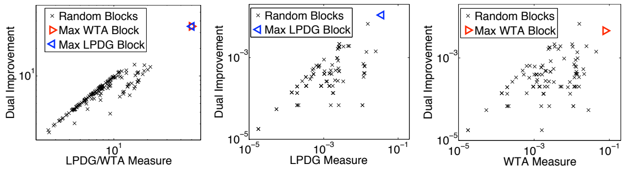


Figure 4. **Dynamic Segmentation Experiment.** For each image(a), global optimum is shown in (b). We then update white pixels in (c) to prefer background. The "warm started" inference then runs until convergence (d) of the modified problem. (e) shows a heat map visualization of where our algorithms passes message updates. Note the localized updates. (f) and (g) compare our method to TRW-S run dynamically with the same setup.

7. Conclusions

In summary, we presented a novel dynamic block selection algorithm for tree-block coordinate ascent in LP-relaxations of MAP inference. Our proposed measures are natural generalizations of complementary slackness conditions and virtual arc consistency criteria. Since our measures incorporate both primal and dual infor-



(a) Iteration 1 (b) Iteration 100 (c) Iteration 100

Figure 5. **Scatter Plots.** (a) Dual improvement vs LPDG/WTA scores at the start of the algorithm. Each point represents a forest, and the total LPDG/WTA scores of all nodes/edges covered are shown. Dual improvement for each point is computed by performing a block-coordinate ascent step corresponding to that forest. At the first iteration, LPDG and WTA scores are precisely the same, hence shown on a single plot. (b,c) Dual improvement versus LPDG and WTA respectively at iteration 100. Correlation coefficients for the three plots are 0.9051, 0.7009 and 0.4955. We can see that our scores are well correlated with the increase in the dual objective.

mation, they alert us to regions where messages need to be passed and where decoding needs to be improved.

Just as synchronous message schedules are understood to be wasteful on tree-like and grid-like graphs, we hope our work provides further evidence that energy-oblivious message schedules are analogously wasteful and inefficient. Our implementation is orders of magnitude faster than the MPLP implementation made available by its authors, but we have not optimized our code to the degree that has been done for TRW-S. We believe there is still substantial opportunity for us to improve the run-time of our method, especially considering the performance in terms of number of distance transforms performed.

An important consideration with dynamic scheduling is whether the benefit gained from more intelligent scheduling outweighs the cost of computing schedules. This tradeoff changes as inference progresses: static schedules can be reasonable in early iterations; our method is most useful in later iterations, where there are fewer dirty edges (leading to a low overhead) and few regions where additional inference is beneficial.

We have also found LPDG to be useful for cluster pursuit in tightening the standard LP-relaxation (Batra et al., 2011), and for speeding-up α-expansion by label reordering (Batra & Kohli, 2011). More generally, since LPDG is a natural generalization of complementary slackness conditions for *any* LP relaxation of an integer program, we expect it to be useful in settings even beyond message-passing algorithms.

References

Alahari, K., Kohli, P., and Torr, P. H. S. Dynamic hybrid algorithms for MAP inference in discrete mrfs. *PAMI*, pp. 1846–1857, 2010.

Batra, D. and Kohli, P. Making the Right Moves: Guiding Alpha-Expansion using Local Primal-Dual Gaps. In *CVPR*, 2011.

Batra, D., Nowozin, S., and Kohli, P. Tighter Relaxations for MAP-MRF Inference: A Local Primal-Dual Gap based Separation Algorithm. In *AISTATS*, 2011.

Bertismas, D. and Tsitsiklis, J. *Introduction to linear optimization*. Athena Scientific, 1997.

Boykov, Y., Veksler, O., and Zabih, R. Efficient approximate energy minimization via graph cuts. *PAMI*, 20(12):1222–1239, 2001.

Chandrasekaran, V., Johnson, J. K., and Willsky, A. S. Adaptive embedded subgraph algorithms using walk-sum analysis. In *NIPS*, 2007.

Cooper, M., De Givry, S., Sanchez, M., Schiex, T., and Zytnicki, M. Virtual arc consistency for weighted csp. In *National conference on Artificial intelligence*, 2008.

Elidan, G., McGraw, I., and Koller, D.Residual Belief Propagation: Informed scheduling for asynchronous message passingIn *UAI*, 2006.

Everingham, M., Gool, L. Van, Williams, C. K. I., Winn, J., and Zisserman, A. The PASCAL Visual Object Classes Challenge 2009.

Felzenszwalb, Pedro F. and Huttenlocher, Daniel P. Distance transforms of sampled functions. Technical report, Cornell Computing and Information Science, 2004.

Globerson, A. and Jaakkola, T. Fixing Max-Product: Convergent Message Passing Algorithms for MAP LP-RelaxationsIn*NIPS*, 2007.

Kohli, P. and Torr, P. H. S. Dynamic graph cuts for efficient inference in Markov Random Fields. *PAMI*, 29(12):2079–2088, 2007.

Kolmogorov, V. Convergent Tree-Reweighted Message Passing for Energy Minimization. *PAMI*, 28(10):1568–1583, 2006.

Kolmogorov, V. and Zabih, R. What energy functions can be minimized via graph cuts? *PAMI*, 26(2):147–159, 2004.

Ladickỳ, L., Russell, C., Kohli, P., and Torr, P. H. S. Associative hierarchical CRFs for object class image segmentation. *ICCV*, 2009.

Meltzer, Talya, Yanover, Chen, and Weiss, Yair. Globally optimal solutions for energy minimization in stereo vision using reweighted belief propagation. In *ICCV*, pp. 428–435, 2005.

Schlesinger, M. Syntactic analysis of two-dimensional visual signals in noisy conditions (in Russian). *kibernetika*, 4:113–130, 1976.

Shotton, J.D.J., Winn, J., Rother, C., and Criminisi, A. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *IJCV*, 2009.

Sontag, D. and Jaakkola, T. Tree Block Coordinate Descent for MAP in Graphical Models. In *AISTATS*, 2009.

Sontag, D., Meltzer, T., Globerson, A., Jaakkola, T., and Weiss, Y. Tightening LP Relaxations for MAP using Message Passing. In *UAI*, 2008.

Sutton, C. and McCallum, A. Improved Dynamic Schedules for Belief Propagation. In *UAI*, 2007.

Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M. F., and Rother, C. A comparative study of energy minimization methods for Markov Random Fields. In *ECCV*, pp. 16–29, 2006.

Tappen, M.F. and Freeman, W.T. Comparison of graph cuts with belief propagation for stereo, using identical mrf parameters. In *ICCV*, pp. 900 –906 vol.2, oct. 2003.

Tarlow, D., Batra, D., Kohli, P., and Kolmogorov, V. Dynamic tree block coordinate ascent. Technical report, 2011a. Available at http://www.cs.toronto.edu/∼dtarlow/papers/tbkk2011.html.

Tarlow, D., Givoni, I. E., Zemel, R. S., and Frey, B. J. Interpreting Graph Cuts as a Max-Product Algorithm. Technical report, 2011b. Available at http://www.cs.toronto.edu/∼dtarlow/papers/tgzf2011.html.

Wainwright, M. J. and Jordan, M. I. Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2008.

Wainwright, M.J., Jaakkola, T.S., and Willsky, A.S. Map estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. *Trans. Inf. Th.*, 51(11):3697–3717, 2005.

Werner, Tomáš. A linear programming approach to max-sum problem: A review. *PAMI*, 29(7):1165–1179, 2007.

Yanover, Chen, Meltzer, Talya, and Weiss, Yair. Linear programming relaxations and belief propagation – an empirical study. *J. Mach. Learn. Res.*, 7:1887–1907, 2006. ISSN 1532-4435.