Covering Trees and Lower-bounds on Quadratic Assignment

Julian Yarkony, Charless Fowlkes, Alexander Ihler
Dept. of Computer Science, University of California, Irvine, CA 92697
{jyarkony,fowlkes,ihler}@ics.uci.edu

Abstract

Many computer vision problems involving feature correspondence among images can be formulated as an assignment problem with a quadratic cost function. Such problems are computationally infeasible in general but recent advances in discrete optimization such as tree-reweighted belief propagation (TRW) often provide high-quality solutions. In this paper, we improve upon these algorithms in two ways. First, we introduce covering trees, a variant of TRW which provide the same bounds on the MAP energy as TRW with far fewer variational parameters. Optimization of these parameters can be carried out efficiently using either fixed–point iterations (as in TRW) or sub-gradient based techniques. Second, we introduce a new technique that utilizes bipartite matching applied to the min-marginals produced with covering trees in order to compute a tighter lower-bound for the quadratic assignment problem. We apply this machinery to the problem of finding correspondences with pairwise energy functions, and demonstrate the resulting hybrid method outperforms TRW alone and a recent related subproblem decomposition algorithm on benchmark image correspondence problems.

1. Introduction

Image correspondence is a classical problem in computer vision. Identifying corresponding points or parts across a pair of images is the basis for a wide range of successful techniques in 3D reconstruction, camera self-calibration, image registration and motion estimation. Correspondence also arises in object recognition with pictorial structures or deformable templates where parts in a model are matched to locations in a target image.

It is common to formulate correspondence as a discrete optimization problem in which each point in a source image or model may be assigned to some set of discrete locations in a target image. A good objective function for such an optimization should assure that local appearance of matched points are similar and that the spatial layout of points in the target image is consistent with the source image or model. It may also be desirable to enforce other constraints, for example that the matching be one-to-one or that the matching satisfies the epipolar constraint. The choice of objective function is also greatly influenced by which objectives can be efficiently optimized.

From the perspective of tractability, a natural choice is the linear assignment problem in which the matching cost depends only on similarity of matched points and one-to-one constraints can be easily enforced. While such problems can be efficiently optimized (using for example the Hungarian algorithm) they provide no direct control over whether the resulting matching is spatially coherent. There have been various attempts to remedy this by using richer unary scores which take into account some spatial context along with multiple iterations of matching [1, 4].

Including spatial coherence in the objective directly requires the introduction of quadratic or higher-order terms which assign a cost to the joint assignment of two or more points. One classic formulation is that of graph matching, in which the layout of features are described by attributed graphs and the goal is to find a permutation of the vertices of one graph which aligns it with the other (see e.g., [11, 13]). Depending on the exact measure of alignment, this is formally equivalent to the Quadratic Assignment Problem studied in the operations research literature. We consider a more somewhat more general formulation in which the cost is an arbitrary quadratic function of the assignment variables.

While such cost functions are often sparse (e.g., only defined between neighboring points in the source image) such discrete quadratic optimization problems are generally intractable and one must resort to some approximation scheme such as spectral rounding [7], loopy belief propagation, dual decomposition [10] or local search [8, 2]. Interestingly, these approximation techniques often require that we drop the one-to-one matching constraint which is so easily incorporated in the linear assignment problem.

In this paper, we tackle the problem of finding solutions which approximately minimize the quadratic objective and also satisfy the one-to-one constraint. Our approach consists of two components. The first is a variant on

tree-reweighted belief propagation (TRW) which efficiently computes lower-bounds on the minimum achievable costs of assigning a point to a given location using a *covering tree* of the graph describing the original quadratic objective. We then solve a variant on the linear assignment problem we term *bottleneck assignment rounding* which uses edge costs derived from the lower-bounds on the min-marginal costs computed with the covering tree. This can be thought of as a procedure for "rounding" solutions to the nearest one-to-one assignment while simultaneously computing a tighter lower-bound on the true constrained objective minimum. The result is both a matching and a lower-bound on the minimum of the true objective providing a measure of the approximation quality.

In the first half of the paper we discuss the covering tree algorithm (CT) which can be applied to optimize general pairwise energy functions found in a variety of vision problems. In section 2 we give a background on the problem decomposition technique on which TRW, CT and other variants are based. In section 3 we prove the equivalence of the bounds given by the CT and TRW decompositions. Section 4 gives a simple algorithm for optimizing the CT bound which guarantees monotonic convergence. We also present experiments on randomly generated problems comparing the convergence rate to other proposed techniques in the literature. In the later half of the paper we describe bottleneck assignment rounding and show theoretically that it gives tighter bounds than CT alone (section 5). We demonstrate that the rounding procedure yields improvement in matching accuracy on a simple image matching problem and compare to other recently proposed techniques in section 6.

2. Energy Minimization Bounds

Consider the problem of minimizing an energy function of the form:

$$E(X,\theta) = \sum_i \theta_i(X_i) + \sum_{i,j} \theta_{ij}(X_i, X_j) \qquad (1)$$

where each X_i takes on values in some finite set of states. The pairwise and singleton functions are described by the collection of parameters $\theta = \{\theta_{i;k}, \theta_{ij;kl}\}$ with

$$\theta_i(X_i) = \sum_k \theta_{i;k} X_{i;k}$$

and

$$\theta_{ij}(X_i, X_j) = \sum_{k,l} \theta_{ij;kl} X_{i;k} X_{j;l}$$

respectively, where $X_{i;k} = \mathbf{1}_{[X_i=k]}$ denotes a binary indicator that variable X_i takes on state k. Typically, the parameters $\theta_{ij;kl}$ are sparse. For example if the X_i correspond to part locations in a model, $\theta_{ij;kl}$ may be non-zero only for nearby parts i, j. We can define a graph $G(\theta)$ whose vertices correspond to variables $\{X_i\}$ in the energy function and with an edge connecting X_i and X_j if and only if $\theta_{ij;kl} \neq 0$ for some pair of states k, l.

In the matching problems we consider, $\theta_{ij;kl}$ encodes the cost of matching some part i to location k while simultaneously matching part j to location l. In the classical quadratic assignment problem, $\theta_{ij;kl}$ factors into the the product of two terms $\theta_{ij;kl} = A_{ij}B_{kl}$ where, e.g., A_{ij} represents the amount of material that needs to be transported from facility i to j and B_{kl} is the distance between possible facility locations k and l. We are interested in the more general case in which θ_{ij} does not factor. These more general problems commonly arise in pictorial structures or deformable template matching where the entries of θ_{ij} may be some arbitrary function of image features and geometry, typically learned from training data.

Minimizing the energy function $E(X, \theta)$ is hard in general but efficient solutions can be found for certain classes of problems. For example, if the graph $G(\theta)$ forms a tree then an optimal solution can be found using dynamic programming. Dynamic programming passes messages in the tree to compute the min-marginal or cost-to-go functions

$$\mu_{ij;kl} = \min_{\substack{X: X_i=k, \\ X_j=l}} E(X;\theta) \qquad \mu_{i;k} = \min_l \mu_{ij;kl}$$

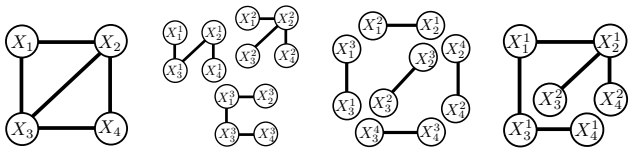
Wainwright et al. [12] suggest an approach to lower bounding the energy of the minimum energy configuration by decomposing the problem θ into a set of tractable subproblems (i.e., tree structured subgraphs of G) and solving each subproblem independently. This technique is generally known as dual-decomposition. Let t index a collection of subproblems defined over the same set of variables X and whose parameters sum up to the original parameter values, so that $\theta = \sum_t \theta^t$. Since the energy function is linear in θ we have

$$\begin{aligned} E_{MAP} &= \min_X E(X, \Theta) = \min_X \sum_t E(X, \Theta^t) \\ &\geq \sum_t \min_{X^t} E(X^t, \Theta^t) \end{aligned}$$

where the inequality arises because minimizing X independently in each subproblem may yield a set of solutions $\{X^t\}$ which are not consistent, so that $X_i^t \neq X_i^s$ for some s, t, i. However, if the solutions of all the subproblems do agree, then the inequality is tight and we have found a minimizing solution to the original problem.

Since there is some freedom in how we decompose the original problem, we can try to find an optimal decomposition which maximizes the lower bound:

$$E_{MAP} \geq \max_{\substack{\{\theta^t\} \\ \sum_t \theta^t = \theta}} \sum_t \min_{X^t} E(X^t, \theta^t) = E_{TRW}$$



(a) (b) (c) (d)

Figure 1. A graphical depiction of bound equivalence. We show that the original collection of spanning trees used in the TRW bound (b) can be split into a collection of single edges (c) and then joined into a covering tree (d). Each of these steps preserves the same lower-bound on the minimal energy configuration of (a). Here superscripts index copies of nodes in the original problem which are free to take on different states (e.g., X_4^1, X_4^2 correspond to X_4)

Computing the optimal lower bound, E_{TRW}, amounts to finding a maximum of the lower envelope of a set of linear functions, one for each possible setting of the X^t; this forms a convex optimization problem. Imposing the constraint that each θ^t has a particular tree structure, the inner minimization for each X^t can be solved efficiently using dynamic programming, allowing one to use an iterative optimization procedures such as projected subgradient or a cutting plane techniques. Wainwright et al. [12] show that this bound can also be optimized using a message passing scheme called tree reweighted belief propagation (TRW).

3. Covering Tree Bounds

The work of [12] and [5] suggest bounding the minimum energy of the original problem by a convex combination of spanning trees. We now show that it is sufficient to consider a single *covering tree* which includes each edge in the original graph exactly once, but includes duplicate copies of some nodes as necessary. Such a covering tree can be constructed, for example, by performing a breadth first search of the original graph without revisiting any edge.

Figure 1 shows an example graph (a) and one possible covering tree (d). In the covering tree, copies of a variable from the original problem are now free to take on different states and the original singleton term in the energy function is split among the copies. As with the TRW decomposition into a collection of spanning trees, we have the possibility that different copies of a node X_i in the covering tree take on different minimizing states. For any given allocation of the the singleton parameters to copies of their respective variables, a minimizing configuration of the variables in the covering tree provides a lower bound on E_{MAP}, and again this lower bound is tight if and only if there is a minimizing configuration in which all corresponding variable copies agree.

With a slight abuse of notation, we will use X_i^t to refer to the copies of variable X_i, where the range of the index t depends on which variable X_i we are considering. Let

$$E_{CT} = \max_{\substack{\{\theta^t\} \\ \sum_t \theta^t_{i;k} = \theta_{i;k}}} \min_X \sum_{i,t} \sum_k \theta^t_{i;k} X^t_{i;k} + \sum_{(i,j)} \sum_{k,l} \theta_{ij;kl} X^{t_{ij}}_{i;k} X^{t_{ji}}_{j;l}$$

where (i, j) are the edges in the original graph, and t_{ij} is the copy of node i which is adjacent to a copy of node j in the covering tree.

Intuitively, optimizing the covering tree lower bound appears quite attractive since there is only a single minimization over X, computable by a dynamic program on the covering tree. Furthermore, there are fewer variational parameters than TRW, since there is no decomposition of the edge parameters into multiple subproblems. In the following section we show that despite possessing fewer variational parameters, the covering tree bound achieves the same optimum as TRW, so that $E_{MAP} \geq E_{CT} = E_{TRW}$.

3.1. Equivalence to TRW bounds

Figure 1 provides a graphical overview of the proof. The basic idea is that it is possible to perform various sorts of surgery on the tree subproblems to produce a new set of subproblems which give the exact same variational bound. This includes splitting a tree at a node and joining two trees at a duplicate node. In splitting a node, we introduce new variational parameters, while joining removes parameters from the bound. We show that at optimal parameter settings, the bound before and after these operations is the same.

We introduce some notation to characterize the optimal parameters and configurations. Let $\Theta = \{\theta^1, \theta^2, \ldots\}$ denote the set of parameters for the whole collection of spanning trees. We say $\Theta \in \mathcal{T}$ if Θ is a valid decomposition of the original problem, i.e., $\sum_t \theta^t = \theta$. For a given setting of parameters Θ there may be more than one minimizing configuration for some variable X_i^t. Let $M(X_i^t, \Theta)$ denote the set of optimizing configurations for node X_i^t given the parameters Θ. If $A = \{X_i^1, X_i^2, \ldots\}$ is a set of copies of X_i then let $M(A, \Theta) = \cap_{X_i^t \in A} M(X_i^t, \Theta)$.

We first state a lemma which describes a necessary condition for optimality of the bound, namely that at optimal settings of Θ all copies of a given node should share a common set of optimizing configurations. This condition is equivalent to "weak tree agreement" derived by Kolmogorov [5].

Lemma 3.1 *If for some $\Theta \in \mathcal{T}$ there is a set of nodes $A = \{X_i^1, X_i^2, \ldots\}$ corresponding to X_i with $M(A, \Theta) = \emptyset$ then there exists $\hat{\Theta} \in \mathcal{T}$ with a strictly greater minimum energy $\sum_t \min_{X^t} E(X^t, \hat{\theta}^t) > \sum_t \min_{X^t} E(X^t, \theta^t)$*

Proof See the supplementary material. Intuitively, if none of the optimal values of a copy X_i^t are optimal for some

other copy $X_i^{t'}$, we can transfer some of $\theta_i^{t'}$ to increase the energy contribution of copy X_i^t, without changing any of the variables' optimal configurations.

The next two lemmas establish that for optimal settings of the variational parameters, we can split or join trees that make up the bound without changing the minimal energy.

Lemma 3.2 Joining *copies of a node from two trees which share an optimizing configuration does not change the minimum energy.*

Proof Let $k \in M(\{X_i^u, X_i^v\}, \Theta)$ for optimal Θ. Then by the definition of an optimizing configuration, we have

$$\begin{aligned}\min_{X^t} \sum_t E(X^t, \theta^t) &= \min_{X^t : X_i^u = X_i^v = k} \sum_t E(X^t, \theta^t) \\ &= \min_{X^t : X_i^u = X_i^v} \sum_t E(X^t, \theta^t)\end{aligned}$$

so a reduced problem in which nodes X_i^u and X_i^v are identified will yield the same bound.

Lemma 3.3 Splitting *a tree by duplicating a node and optimizing the allocation of singleton parameters between the duplicates does not change the minimum energy.*

Proof Let $E(X, \theta)$ be the energy of some tree T. If we split the tree at node X_i let $E(X^1, \theta^1), E(X^2, \theta^2)$ denote the energy of the resulting pieces with X_i^1 and X_i^2 denoting the duplicates of X_i and $\Theta = \{\theta^1, \theta^2\}$ the set of parameters. Let

$$\Theta^* = \arg\max_{\Theta} \min_{X^1, X^2} E(X^1, \theta^1) + E(X^2, \theta^2)$$

By Lemma 3.1 we know there exists $k \in M(\{X_i^1, X_i^2\}, \Theta^*)$ (otherwise Θ^* would not be optimal). By Lemma 3.2 we know that joining the two trees back into a single tree will not increase the energy, so we have the equalities

$$\begin{aligned}&\max_{\Theta} \min_{X^1, X^2} E(X^1, \theta^1) + E(X^2, \theta^2) \\ &\quad = \max_{\Theta} \min_{X_i^1 = X_i^2 = k} E(X^1, \theta^1) + E(X^2, \theta^2) \\ &\quad = \max_{\Theta} \min_{X : X_i = k} E(X, \theta) \\ &\quad = \max_{\Theta} \min_{X} E(X, \theta)\end{aligned}$$

Note that splitting only applies to trees. In general we cannot, for example, split a cycle into a chain and expect to find the minimum energy configuration simply by optimizing the singleton parameters of the end points.

Theorem 3.4 $E_{TRW} = E_{CT}$

Proof We proceed by tree surgery. Start with any collection of edge spanning trees which cover every edge of $G(\theta)$ at least once. We repeatedly *split* each of these trees until we are left with a collection of doublets, or trees consisting of only a single edge and two nodes. By Lemma 3.3, if we optimize the variational (singleton) parameters across these doublets, we will achieve the same lower-bound as with the set of spanning trees. At this point, we can *merge* any doublets which are duplicates by simply adding together their parameters. This yields a minimal number of doublets, one for each edge in the original graph. Lastly, we *join* these doublets into a single covering tree. As stated in Lemma 3.2, for optimal settings of the parameters, all copies of a node have at least one optimizing configuration in common so constraining them to take on the same state will not change the bound.

3.2. Discussion

There are a number of potentially useful insights to observe from this proof. First, the proof provides a slightly different outlook which hopefully sheds some light on the work of Wainwright and Kolmogorov. For example, it is easy to see why the exact choice of spanning trees in TRW is inconsequential as long as every edge is covered at least once, since we can always split and merge any such choice of trees down to the same set of doublets. In terms of the achievable bound, there is no advantage to working with (say) the entire set of spanning trees. In fact, the covering tree provides the smallest representation (in terms of the fewest number of variational parameters) which yields the same bound. Finally, Lemma 3.1 points out a fundamental feature of this approach: at the optimal bounds, unless there is tree agreement there will always be some degeneracy in the minimum energy configuration of the covering tree.

4. Algorithms for Bound Optimization

Now that we have seen that the covering tree representation provides the same bound as TRW at optimal settings of the parameters θ_i^t, we turn to optimizing the bound over these parameters. We show both a fixed point update which monotonically increases the bound, generalizing TRW-S [5] and tree-block coordinate ascent [9], as well as giving a sub-gradient update for the parameters similar to [6].

To initialize, we choose a covering tree of the original graph and initial setting of parameters. Each iteration then consists of a pass of dynamic programming in the covering tree (computing cost-to-go functions μ and an optimizing configuration $\hat{X}$) and a parameter update step to any subset S of the parameters of copied nodes. The algorithm itself is given in Figure 2.

The sub-gradient rule takes a gradient step to increase the bound given the configuration $\hat{X}$. If this update does not

Until convergence:

1. Choose any subset S of the copied nodes (potentially all nodes), and define $S_i = \{t : X_i^t \in S\}$, the copies of node i in S
2. Run dynamic programming to compute the cost-to-go functions μ_i^t and an optimizing configuration $\hat{X}$ at each node.
3. Do either of the following updates for each θ_i^t in S, with step-size γ:

   (a) Fixed point update:

   $$\bar{\theta}_{i;k}^t = \theta_{i;k}^t - \gamma(\mu_{i;k}^t - \frac{1}{|S_i|}\sum_{t \in S_i} \mu_{i;k}^t)$$

   (b) Subgradient update:

   $$\bar{\theta}_{i;k}^t = \theta_{i;k}^t + \gamma(\hat{X}_{i;k}^t - \frac{1}{|S_i|}\sum_{t \in S_i} \hat{X}_{i;k}^t)$$

   where $\hat{X}_{i;k}^t$ is an indicator function that X_i^t takes on value k in $\hat{X}$.

Figure 2. Covering Tree Algorithm.

change the optimal configuration it corresponds to gradient ascent, but in general can decrease the bound if the configuration changes. For an appropriate schedule of step-sizes, subgradient ascent is guaranteed to find a global optimum. This update method corresponds to the algorithm proposed in [6].

The fixed point update has a similar form, and attempts to match the μ_i^t at different copies t of node i. This update can be made to monotonically increase the bound for any choice of S (including all nodes); however, like other fixed-point parameter updates for TRW it can become stuck at suboptimal settings which satisfy weak tree agreement [5].

The covering tree algorithm is exceptionally simple, and does not require building or maintaining forests of spanning trees. It contains a strictly smaller number of parameters than previous methods, and never modifies the edge parameters θ_{ij}. This last point may be useful if the pairwise parameters are themselves sparse (for example, a few preferred or undesirable configurations). Finally, covering trees generalize TRW-S and tree-block updates in the sense that these algorithms consist of a subset of updates on the covering tree, and the covering tree can be used to update a strictly greater set of parameters if desired, up to and including all variational parameters in one step.

4.1. Monotone Updates

We would like to determine a "step size" γ which guarantees that the fixed point update always increases our lower-bound on the objective. We show that for any selection of nodes in step 1 of the covering tree algorithm, there is a step size which guarantees monotonic increase in the lower-bound.

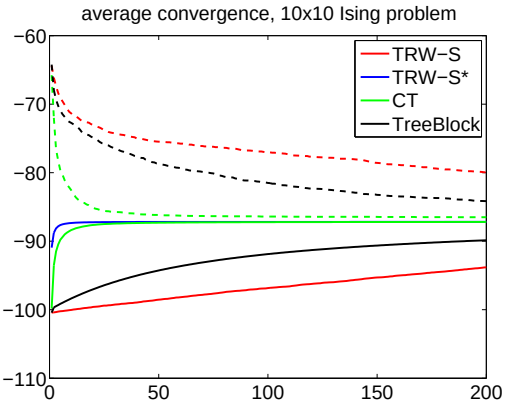


Figure 3. Comparison of bound optimization algorithms on random 10x10 Ising grid problems with repulsive potentials. Curves show energies averaged across 100 problem instances. The x-axis indicates the number of rounds of dynamic programming over the graph. Dashed lines indicate an upper-bound given by the energy of the lowest energy state found among the subproblems. TRW-S and TreeBlock updates update a single node or spanning tree on each pass while the CT updates every node in the covering tree. TRW-S* is the efficient version of TRW-S for ordered chains.

Theorem 4.1 *The fixed point parameter update given in Figure 2 with step size $\gamma = 1/|S|$ strictly increases the bound: $E_{CT}(\bar{\Theta}) \geq E_{CT}(\Theta)$.*

Proof See supplementary material. We modify the reparameterization construction of Sontag and Jaakkola [9] to apply to the covering tree data structure (in which the individual copies of each variable are connected by some path), and to update an arbitrary subset S of the variables.

4.2. Convergence

Figure 3 shows an experimental comparison of the the convergence rates for different bound optimization approaches. Here we compare the covering tree update in which all parameters are updated simultaneously. In this setting, each step incorporates longer-range information than tree-block coordinate descent, providing a similar improvement to tree-block updates over naïve TRW-S and generalizing both, in the sense that these algorithms can be thought of as acting on subsets of the covering tree. In practice, we also find that the step size $\frac{1}{|S|}$ may be overly conservative; an alternative strategy is to begin with a relatively large step size (we use $\gamma = \frac{1}{2}$), compute the bound after each step, and decrease the stepsize anytime non-monotonic behavior is observed.

Kolmogorov [5] also provides a highly efficient update strategy for TRW-S which makes use of ordered chains

along which parameters are updated as the messages are computed (TRW-S* in Figure 3). This essentially interleaves the parameter updates with the dynamic programming and gives much larger gains in the bound per message passed than the naïve implementation. We note in passing that, given an ordering, this same strategy can also be implemented on a covering tree to provide the same benefits.

5. Bottleneck Assignment Rounding

We now turn to the quadratic assignment problem we use for finding correspondences. We can formulate the quadratic assignment problem in the same discrete minimization framework:

$$\begin{aligned} E_{AMAP} &= \min_{X \in \mathcal{A}} E(X, \theta) \\ &= \min_{X \in \mathcal{A}} \sum_i \theta_i(X_i) + \sum_{i,j} \theta_{i,j}(X_i, X_j) \end{aligned}$$

where we have now added a constraint; $\mathcal{A}$ is the space of 1-to-1 assignments, so that for solutions X, if $X_i = k$ then $X_j \neq k \; \forall j \neq i$.

The addition of the assignment constraint often makes this problem considerably more difficult to solve. The assignment constraints can be encoded by adding hard pairwise potentials between each pair of nodes in the model, giving energy ∞ to configurations which violate the constraint. Although this yields an unconstrained QP, its corresponding graph structure is dense (fully connected), so that TRW approximations will be forced to break many of these dependencies and cannot guarantee a feasible solution. We suggest an alternative scheme which finds a low-cost 1-to-1 assignment guided by the covering tree solution for the unconstrained problem.

Suppose for each variable X_i we have computed a lower bound $H(i,k)$ on the energy of a solution in which X_i takes on state k. We can compute such a lower bound via dynamic programming on the covering tree to compute the cost-to-go functions for each node and set $H(i,k) = \max_t \mu^t_{i;k}$[1]. We then can compute a lower-bound on the minimum energy assignment as

$$E_{BAR} = \min_{X \in \mathcal{A}} \max_i H(i, X_i)$$

This is a variant of the linear assignment problem known as linear bottleneck assignment (see [3]). We refer to this procedure as *bottleneck assignment rounding* since it finds an assignment which is, in some sense, nearest to the unconstrained solution given by the covering tree.

It is easy to show that E_{BAR} can be computed efficiently. A simple algorithm to compute E_{BAR} is to construct a sequence of bipartite graphs G_t for different thresholds t which contain an edge (i,k) iff $H(i,k) > t$. For a threshold t, $E_{BAR} \leq t$ only if there exists a matching in G. The minimal value of t for which a matching still exists specifies the minima of Equation 5 and can be found by performing a binary search over the possible values of t.

We now establish that E_{BAR} is a lower-bound on the minimum energy assignment and is tighter than the covering tree bound E_{CT} alone.

Theorem 5.1 $E_{CT} \leq E_{BAR} \leq E_{AMAP}$

Proof Let $\hat{X} = \arg\min_{X \in \mathcal{A}} E(X, \theta)$ be the minimizing assignment. Let $Q(i,k) = \min_{X; X_i = k, X \in \mathcal{A}} E(X, \theta)$ be the true costs-to-go in the original graph (with the assignment constraint) and $H(i,k) = \max_t \min_{X: X^t_i = k} E(X, \Theta)$ the lower bound on the costs-to-go given by the covering tree so that $H(i,k) \leq Q(i,k)$. We then have:

$$\begin{aligned} E_{CT} = \min_X E(X, \Theta) = \min_X \max_i H_{i,X_i} \\ \leq \min_{X \in \mathcal{A}} \max_i H_{i,X_i} \quad = E_{BAR} \end{aligned}$$

where the first line is by definition of the cost-to-go H and the inequality arises because we are adding the additional constraint that X is an assignment. In turn,

$$E_{BAR} \leq \max_i H_{i,\hat{X}_i} \leq \max_i Q_{i,\hat{X}_i} = E_{AMAP}$$

The first inequality arises since any valid assignment (including the optimal one $\hat{X}$) must cost at least as much as the minimal bottleneck assignment and the second since H is a lower bound on Q. The final equality is by definition of the true cost-to-go energy Q. This establishes that E_{BAR} is an upper bound on E_{CT} and is a lower bound on E_{AMAP}

6. Experimental Comparisons

We tested the efficacy of bottleneck assignment rounding for finding matched features in wide-baseline matching tasks on the CMU "house" image sequence. Image feature detection positions, along with ground truth matchings for comparison, were obtained from Caetano et al. [4]. A pair of images from the sequence and its 30 features are shown in Figure 4, with true correspondences indicated by matching colors.

Following Caetano et al. [4] and Torresani et al. [10], we use Shape Context features to define the singleton (node) potentials:

$$\theta_{i;k} = -\|F^1_i - F^2_k\|^2$$

where F^a_b is the shape context vector for point b in image a. For pairwise similarity cost parameters, we use a weighted

[1] the max is only necessary if we have stopped the CT algorithm prior to convergence

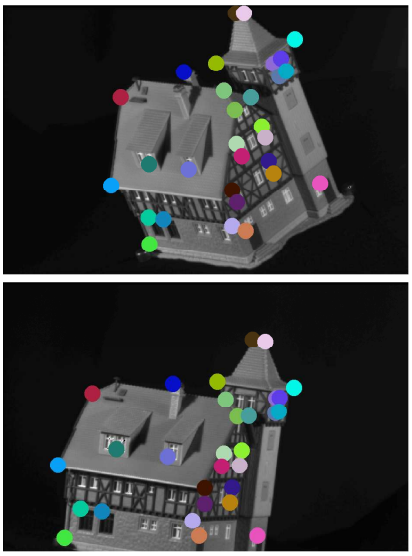


Figure 4. CMU "house" data set, showing corresponding feature points by matching color. Singleton energy functions capture similarity of appearance, and pairwise energies measure similarity in the points' relative geometry.

combination of angle and distance similarity between the feature positions in the first and second images:

$$\theta_{ij;kl} = -w_a \|\angle_{ij} - \angle_{kl}\|^2 - w_d \|d_{ij} - d_{kl}\|^2$$

This is similar to the geometric energy function given in [10]. As in Torresani et al. [10], we restrict our set of pairwise interaction terms to a small neighborhood of features, so that the edges in G correspond to the k nearest neighbors of each point in the reference image.

We compare three algorithms for correspondence: our bottleneck assignment rounding applied to the covering tree marginals (BAR), a naïve covering tree which does not take into account any additional assignment constraints (beyond those included in the nearest neighbors' pairwise functions), and the graph matching algorithm of Torresani et al. [10]. The latter approach also utilizes a decomposition into simpler subproblems, but with a more sophisticated collection of subproblems than trees. The graph matching code is written in Visual C++, while the covering tree and bottleneck assignment rounding algorithms are implemented in Matlab. We also considered a fourth baseline algorithm based on the classical Gilmore-Lawler bound which has been previous used, e.g. in [8, 2], as an initialization for local search. However, we found that while efficient, it was generally not competitive in terms of bounds or accuracy.

Figure 5 shows the performance of the algorithms as a function of the image baseline. For nearby images, the correspondence problem is not difficult and all methods do well. However, as the problem becomes more difficult, we begin to see the differences among the algorithms. To better illustrate these differences, in our comparisons we focus

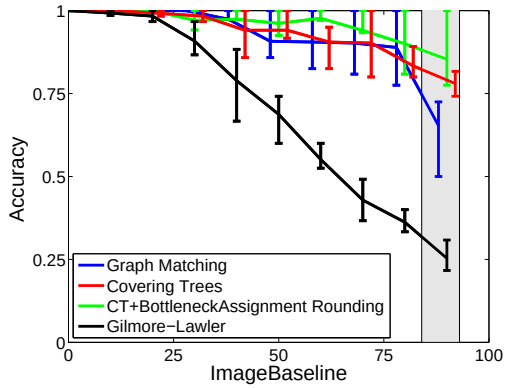
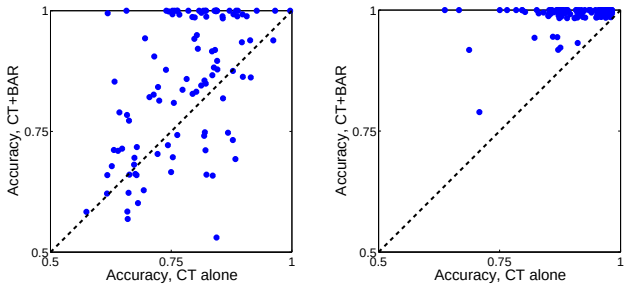


Figure 5. Accuracy of the algorithms as a function of image baseline separation. Curves show the mean accuracy with error bars indicating quartiles computed over N=15 image pairs at each baseline separation. For frames which are more widely separated in the image sequence (larger baseline), the change in perspective increases, making the task of finding a good set of correspondences more difficult. For nearby image pairs from the sequence, all three methods perform well. However, by the indicated region (85–90 frames) there are more measurable differences between the methods.



(a) 3-NN (b) 4-NN

Figure 6. Comparing bottleneck assignment to covering trees without enforcing additional assignment constraints, on graphs with 3- or 4-nearest-neighbor (3-NN, 4-NN) interactions. In both cases, the inclusion of assignment constraints improves the accuracy of the solutions.

specifically on the indicated region of large baseline, between 85 and 90 frames of separation between the images.

We first compare the performance of covering trees (CT) with and without the bottleneck assignment rounding (CT+BAR). There are 105 pairs of images in total with baseline change in the selected range, and Figure 6 shows a scatterplot of the accuracy of CT+BAR (y-axis) compared with CT (x-axis) on models created using 3- and 4-nearest-neighbor interactions. For 3-NN graphs, the accuracy increases slightly with inclusion of assignment constraints (85.6% compared to 78.2% on average); however, with the inclusion of 4-NN interactions, the assignment constraints lead to considerably better solutions (average accuracy 99.4% compared to 89.9% for covering trees).

Next, we compare the CT+BAR solution to the graph

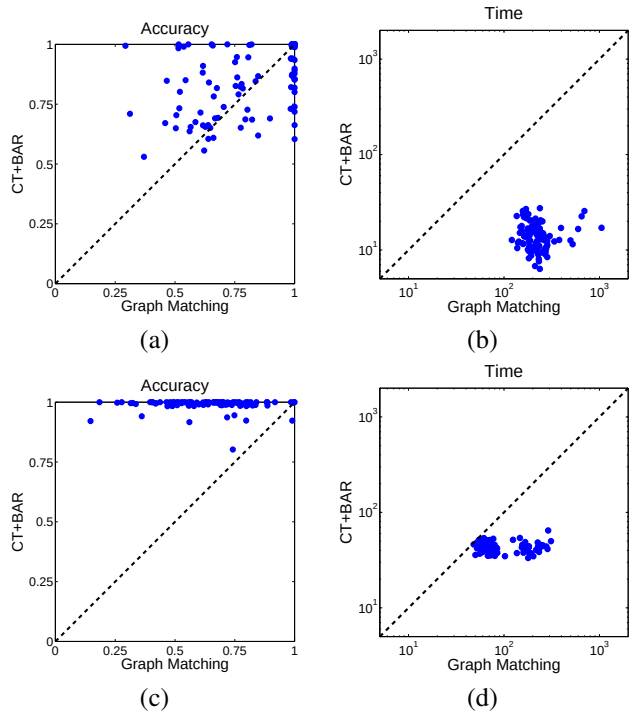


Figure 7. Comparing covering trees with bottleneck assignment rounding (CT+BAR) to Graph Matching [10]. (a)-(b) For problems of equal complexity (3-nearest-neighbor interactions), CT+BAR gives similar accuracy but is significantly faster. (c)-(d) Using 4-NN interactions in CT+BAR and only 2-NN in graph matching, CT+BAR provides significantly more accuracy in a similar amount of time.

matching methods of Torresani et al. [10]. Timing and accuracy values are plotted in Figure 7. The top row shows the accuracy and timing comparison using 3-NN interactions in both algorithms. The accuracies of the two methods are reasonably similar (85.6% for BAR compared to 81.8% for graph matching overall); however, the CT+BAR algorithm is considerably faster in all cases (by factors ranging between 6 and 60 times).

To compare the algorithms on a more equal timing footing, we simplify the model used by graph matching to include only 2-NN interactions, while increasing the interactions used by CT+BAR to 4-NN. Under these conditions, the algorithms are similar in runtime (CT+BAR remains a factor of 1–7$\times$ faster), but now CT+BAR significantly outperforms graph matching in terms of accuracy, achieving an average of 99.4% compared to 64.7%. Within our experiments, using the covering tree with bottleneck assignment rounding compared to graph matching was on average always faster, more accurate, or both.

7. Conclusion

In this paper we have introduced two new techniques which are applicable to combinatorial optimization problems of interest to vision researchers. It is worth noting that these two techniques are largely independent. Covering Trees can be applied to general energy minimizations problems which abound in computer vision (such as inference in Markov random fields). Similarly, the bottleneck assignment rounding can be applied to strengthen the output of any algorithm which gives lower-bounds on the cost-to-go for partial matches. Taken together, these algorithms improve upon the state-of-the art speed and accuracy for correspondence as seen in our preliminary image matching experiments.

References

[1] S. Belongie, J. Malik, and J. Puzicha. Shape context: A new descriptor for shape matching and object recognition. In *NIPS*, pages 831–837, 2001.

[2] A. C. Berg, T. L. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondence. In *CVPR*, 2005.

[3] R. Burkard, M. Dell'Amico, and S. Martello. *Assignment problems*. Society for Industrial Mathematics, 2009.

[4] T. S. Caetano, J. J. McAuley, L. Cheng, Q. V. Le, and A. J. Smola. Learning graph matching. *IEEE Trans. Pattern Anal. Machine Intell.*, 31(6):1048–1058, 2009.

[5] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Trans. Pattern Anal. Machine Intell.*, 28(10):1568–1583, 2006.

[6] N. Komodakis, N. Paragios, and G. Tziritas. MRF optimization via dual decomposition: Message-passing revisited. In *ICCV*, Rio de Janeiro, Brazil, Oct. 2007.

[7] M. Leordeanu and M. Hebert. A spectral technique for correspondence problems using pairwise constraints. In *ICCV*, 2005.

[8] J. Maciel and J. Costeira. A global solution to sparse correspondence problems. *IEEE Trans. Pattern Anal. Machine Intell.*, 25(2):187–199, 2003.

[9] D. Sontag and T. Jaakkola. Tree block coordinate descent for MAP in graphical models. In *AI & Statistics*, pages 544–551, Clearwater Beach, Florida, 2009. JMLR: W&CP 5.

[10] L. Torresani, V. Kolmogorov, and C. Rother. Feature correspondence via graph matching: Models and global optimization. In *ECCV*, pages 596–609, 2008.

[11] S. Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE Trans. Pattern Anal. Machine Intell.*, pages 695–703, 1988.

[12] M. J. Wainwright, T. Jaakkola, and A. S. Willsky. MAP estimation via agreement on (hyper)trees: message-passing and linear programming approaches. *IEEE Trans. Inform. Theory*, 51(11):3697–3717, 2005.

[13] M. Zaslavskiy, F. Bach, and J. Vert. A path following algorithm for the graph matching problem. *IEEE Trans. Pattern Anal. Machine Intell.*, pages 2227–2242, 2009.

SUPPLEMENT: Covering Trees and Lower-bounds on Quadratic Assignment

Julian Yarkony, Charless Fowlkes, Alexander Ihler
Dept. of Computer Science, University of California, Irvine, CA 92697
{jyarkony,fowlkes,ihler}@ics.uci.edu

1. Proof of Lemma 3.1

We first give two lemmas which show that since there is an energy gap between optimizing and non-optimizing configurations, there must always exist some "slack" in the singleton potential parameters corresponding to those states.

Lemma A *If* $k \notin M(X_i^t, \Theta)$ *then there exists* $\hat{\Theta}$ *with* $\hat{\theta}_{i;k}^t < \theta_{i;k}^t$ *and all other entries the same, so that* $M(X_i^t, \Theta) = M(X_i^t, \hat{\Theta})$.

Proof If $k \notin M(X_i^t, \Theta)$ and $l \in M(X_i^t, \Theta)$ then there exists some gap $\epsilon > 0$ with $\min_{X^t: X_i^t = l} E(X^t, \theta^t) > \min_{X^t: X_i^t = l} E(X^t, \theta^t) + \epsilon$. Choose $\hat{\Theta} = \Theta$ except for $\hat{\theta}_{i;k}^t = \theta_{i;k}^t - \epsilon$. Then, even though we have decreased the energy of X_i^t taking on state k, it is still the case that $k \notin M(X_i^t, \hat{\Theta})$ since

$$\begin{aligned} \min_{X^t: X_i^t = l} E(X^t, \hat{\theta}^t) &= \min_{X^t: X_i^t = l} E(X^t, \theta^t) \\ &< \min_{X^t: X_i^t = k} E(X^t, \theta^t) - \epsilon \\ &= \min_{X^t: X_i^t = k} E(X^t, \hat{\theta}^t) \end{aligned}$$

Lemma B *There exists* $\hat{\Theta}$ *where* $\forall l \in M(X_i^t, \Theta), \hat{\theta}_{i;l}^t > \theta_{i;l}^t$ *and all other entries the same so that* $M(X_i^t, \Theta) = M(X_i^t, \hat{\Theta})$.

Proof For each $k \notin M(X_i^t, \Theta)$ we can choose $\epsilon_k > 0$ (as in the previous lemma) so that $\forall l \in M(X_i^t, \Theta)$

$$\min_{X^t: X_i^t = k} E(X^t, \theta^t) > \min_{X^t: X_i^t = l} E(X^t, \theta^t) + \epsilon_k$$

Then set $\hat{\theta}_{i;l}^t = \theta_{i;l}^t + \min_k \epsilon_k$ for all $l \in M(X_i^t, \Theta)$. The optimizing configurations are unaltered by this perturbation since $\forall k \notin M(X_i^t, \Theta), l \in M(X_i^t, \Theta)$ we have

$$\min_{X^t: X_i^t = l} E(X^t, \hat{\theta}^t) < \min_{X^t: X_i^t = k} E(X^t, \hat{\theta}^t)$$

and $\forall k, l \in M(X_i^t, \Theta)$ we have

$$\min_{X^t: X_i^t = l} E(X^t, \hat{\theta}^t) = \min_{X^t: X_i^t = k} E(X^t, \hat{\theta}^t)$$

Now we are ready to prove Lemma 3.1 which establishes a necessary condition for optimality, namely that at optimal settings of Θ all copies of a given node should share a common set of optimizing configurations.

Lemma 3.1 If for some $\Theta \in \mathcal{T}$ there is a set of nodes $A = \{X_i^1, X_i^2, \ldots\}$ corresponding to X_i with $M(A, \Theta) = \emptyset$ then there exists $\hat{\Theta} \in \mathcal{T}$ with a greater minimum energy $\sum_t \min_{X^t} E(X^t, \hat{\theta}^t) > \sum_t \min_{X^t} E(X^t, \theta^t)$

Proof Let G and H be non-empty subsets of A where H contains a single node X_i^h, $M(G, \Theta) \neq \emptyset$, $M(H, \Theta) \neq \emptyset$ and G and H don't share any optimizing configurations, $M(G, \Theta) \cap M(H, \Theta) = \emptyset$ (see Lemma C below for proof that G and H always exist).

We first modify Θ to assure that each node in G has the same set of optimizing configurations. Let S be the largest subset of G with some state $k \in M(S, \Theta)$ but for which $k \notin M(G, \Theta)$. There must exist some $X_i^t \in G$ for which $k \notin M(X_i^t, \Theta)$. Let $\hat{\theta}_{i;k}^t = \theta_{i;k}^t - \epsilon$ and $\forall s \in S$ set $\hat{\theta}_{i;k}^s = \theta_{i;k}^s + \frac{\epsilon}{|S|}$. Clearly $\hat{\theta} \in \mathcal{T}$ and by Lemma A we can choose $\epsilon > 0$ small enough that k doesn't join the optimizing configurations of X_i^t but the positive increment will drive it out of the optimizing configurations of the nodes in S. This cleanup can be repeated without changing the energy until $M(X_i^t, \hat{\Theta}) = M(G, \hat{\Theta})$ for all $X_i^t \in G$.

We now make a final perturbation to $\hat{\Theta}$ which increases the minimal energy without changing the optimizing configurations. Let $\epsilon_G > 0$ be smaller than the energy gap between the optimizing and non-optimal configurations of G (as in Lemma B). Let $\epsilon_H > 0$ be smaller than the gap for $H = \{X_i^h\}$. Let $\epsilon = \min(\epsilon_G, \epsilon_H)$. For all $l \in M(G, \hat{\Theta})$ set $\hat{\theta}_{i;l}^h \leftarrow \hat{\theta}_{i;l}^h - \epsilon$ and for all t with $X_i^t \in G$ set $\hat{\theta}_{i;l}^t \leftarrow \hat{\theta}_{i;l}^t + \frac{\epsilon}{|G|}$. The value of ϵ is small enough so as to not change the optimizing configuration, however, we have increased the parameters associated with those configurations so the minimizing energy increases

$$\begin{aligned} \sum_t \min_{X^t} E(X^t, \hat{\theta}^t) &= \sum_t \min_{X^t} E(X^t, \theta^t) + \frac{\epsilon}{|G|} \\ &> \sum_t \min_{X^t} E(X^t, \theta^t) \end{aligned}$$

while $\hat{\Theta} \in \mathcal{T}$.

Lemma C *Sets G and H as prescribed in the proof of Lemma 3.1 always exist.*

Proof The following procedure constructs subsets G and H satisfying the properties prescribed in the proof of Lemma 3.1. Let A be a set of nodes $X_i^1, X_i^2, X_i^3 ... X_i^T$. Let A have at least one element and let $M(A, \Theta) \neq \emptyset$ given potentials Θ. We will show by construction $G \subset A$, $H \subset A$, $G \cap H = \emptyset$,$M(G, \Theta) \neq \emptyset$,$M(H, \Theta) \neq \emptyset$, $M(G, \Theta) \neq \emptyset$, $M(H, \Theta) \neq \emptyset$, $M(G, \Theta) \cap M(H, \Theta) = \emptyset$, $|H| = 1$

Let $G \leftarrow \{X_a^1\}$
let $i \leftarrow 2$
while $M(G \cup \{X_a^i\}) \neq \emptyset$
. $\quad G \leftarrow \{G \cup \{X_a^i\}\}$
. $\quad i \leftarrow i + 1$
$H \leftarrow \{X_a^i\}$

Note this procedure must terminate for some $i \leq T$ since $M(A, \Theta) \neq \emptyset$. Since an element will never be added to G which makes $M(G, \Theta)$ empty and H only has one element so both $M(G, \Theta)$ and $M(H, \Theta)$ can not be empty. Since each element in A can be added to G,H or neither but not both $G \cap H = \emptyset$. Since all elements in G are member of A and one element in A is in H and $G \cap H = \emptyset$ then $G \subset A$ and $H \subset A$. Lastly, the termination condition assure that $M(G \cup \{X_a^i\}, \Theta) \neq \emptyset$ and sets $H \leftarrow \{X_a^i\}$ so $M(G \cup H, \Theta) = \emptyset$.

2. Proof of Theorem 4.1

We now show that for any selection of nodes in step 1 of the covering tree algorithm, there is a step size which guarantees monotonic increase in the lower-bound.

Theorem 4.1 The fixed point parameter update given in Figure 2 with step size $\gamma = 1/|S|$ strictly increases the bound: $E_{CT}(\bar{\Theta}) \geq E_{CT}(\Theta)$.

Proof Suppose our covering tree is specified by parameters $\Theta = \{\theta_i^t\} \cup \{\theta_{ij}\}$ and we have computed the cost-to-go functions $\mu_{ij;kl}$, $\mu_{i;k}^t$. We can reparameterize the factors into a form similar to that proposed in Sontag and Jaakkola [9],

$$\lambda_{i;k}^t = \frac{1}{|S|}(\mu_{i;k}^t - \alpha_i^t)$$

$$\lambda_{ij;kl} = \mu_{ij;kl} - \alpha_{ij} - \frac{S_{ij}}{|S|}(\mu_{i;k}^{t_{ij}} - \alpha_i^{t_{ij}}) - \frac{S_{ji}}{|S|}(\mu_{j;l}^{t_{ji}} - \alpha_j^{t_{ji}})$$

where S_{ij} is the number of nodes in S nearer to $X_i^{t_{ij}}$ than to $X_j^{t_{ji}}$ in the covering tree (including $X_i^{t_{ij}}$ itself, if in S), and we choose $\alpha_i^t = \min_k \mu_{i;k}^t$ and $\alpha_{ij}^t = \min_{kl} \mu_{ij;kl}$. A straightforward counting argument modified from Sontag and Jaakkola [9] demonstrates that the λ is a valid reparameterization plus a scalar, i.e., $E(X; \Theta) = E(X; \Lambda) + \sum \alpha$. Furthermore, all parameters λ are nonnegative. Since by definition $\mu_{i;k}, \mu_{j;l} \leq \mu_{ij;kl}$ it must be that $\mu_{ij;kl} - w\mu_{i;k} - (1-w)\mu_{j;l} \geq 0$ for $0 \leq w \leq 1$ so there exists a configuration $\hat{X}$ of the covering tree nodes $\{X_i^t\}$ which achieves $E(\hat{X}; \Lambda) = 0$.

Now, consider the averaging update,

$$\bar{\lambda}_i^t = \frac{1}{|S_i|} \sum_t \lambda_i^t \qquad \text{and} \qquad \bar{\lambda}_{ij} = \lambda_{ij}.$$

Again, all entries in $\bar{\lambda}_i^t$ and $\bar{\lambda}_{ij}$ are nonnegative. Moreover, if $M(S_i, \Lambda) = \emptyset$ (no optimizing configuration is shared by all copies of X_i), then $\bar{\lambda}_i$ will be strictly positive. Thus, $E(\bar{\Lambda}) \geq E(\Lambda)$, with equality if and only if weak tree agreement holds at all nodes in S.

Finally, to obtain our update rule, we simply consider reversing the reparameterization $\Theta \leftrightarrow \Lambda$. Reparameterization simply shifts values between the singleton and pairwise parameters, preserving the energy function itself. Defining the net reparameterization amounts by β, so that

$$\theta_{ij;kl} = \lambda_{ij;kl} + \beta_{i;k}^{t_{ij}} + \beta_{j;l}^{t_{ji}} + \alpha_{ij}$$
$$\theta_{i;k}^t = \lambda_{i;k}^t - \sum_{j \in \Gamma_i^t} \beta_{i;k}^t + \alpha_i^t$$

we can immediately see that

$$\bar{\theta}_{ij;kl} = \bar{\lambda}_{ij;kl} - \beta_{i;k}^{t_{ij}} - \beta_{j;l}^{t_{ji}} - \alpha_{ij} \qquad = \theta_{ij;kl}$$
$$\bar{\theta}_{i;k}^t = \bar{\lambda}_{i;k}^t + \sum_{j \in \Gamma_i^t} \beta_{i;k}^t - \alpha_i^t \qquad = \theta_{i;k}^t + (\bar{\lambda}_{i;k}^t - \lambda_{i;k}^t)$$

which, applying the definition of $\lambda_{i;k}^t$ gives the update in Figure 2. Thus we have that $E(\bar{\Theta}) = E(\bar{\Lambda}) + \sum \alpha \geq E(\Lambda) + \sum \alpha = E(\Theta)$, and our fixed-point update provides a monotonic increase in the lower bound, with equality at weak tree agreement.