Pairwise Matching through Max-Weight Bipartite Belief Propagation

Zhen Zhang[1] Qinfeng Shi[2] Julian McAuley[3] Wei Wei[1] Yanning Zhang[1] Anton van den Hengel[2]
[1]School of Computer Science and Engineering, Northwestern Polytechnical University, Xi'an, China
[2]School of Computer Science, The University of Adelaide, Australia
[3]Computer Science and Engineering Department, University of California, San Diego, USA

Abstract

Feature matching is a key problem in computer vision and pattern recognition. One way to encode the essential interdependence between potential feature matches is to cast the problem as inference in a graphical model, though recently alternatives such as spectral methods, or approaches based on the convex-concave procedure have achieved the state-of-the-art. Here we revisit the use of graphical models for feature matching, and propose a belief propagation scheme which exhibits the following advantages: (1) we explicitly enforce one-to-one matching constraints; (2) we offer a tighter relaxation of the original cost function than previous graphical-model-based approaches; and (3) our sub-problems decompose into max-weight bipartite matching, which can be solved efficiently, leading to orders-of-magnitude reductions in execution time. Experimental results show that the proposed algorithm produces results superior to those of the current state-of-the-art.

1. Introduction

The feature matching problem aims to find consistent correspondences between two sets of features or points, and is a key step in many tasks in computer vision and pattern recognition including self-calibration, image registration, and multiple object tracking[8, 23, 24]. While finding one-to-one correspondences between local feature points can be done efficiently, the problem becomes NP-hard as soon as pairwise affinities are introduced, leading to a range of relaxation and approximation techniques.

Graphical models are often used to encode pairwise matching problems as they naturally express the interdependence among a set of putative matches where each feature can be matched only once. Bayati, Shah and Sharma[2] first formulated feature matching as a MAP inference problem in a graphical model (*without* pairwise constraints) and incorporating pairwise constraints was addressed by Duchi *et al*. [9], and others[29]. The primary disadvantage of the approach of Duchi *et al*. [9] is that convergence cannot be guaranteed; Torresani, Kolmogorov and Rother[29] addressed this limitation, devising a method which was considered the state-of-the-art at the time. Even so, the looseness of the relaxation limited the extent to which the matching quality could be guaranteed. This limitation served as a primary motivator for recent approaches, and a shift away from methods based on inference in graphical models.

A range of alternative approaches to the matching problem have been proposed (see [5] for a review). One approach has been to relax the one-to-one matching constraints, and to recover them later through post-processing; this is the approach typically adopted by Spectral Matching (SM) [7, 19], and Local Sparse Models (LSM)[15]. This produces good results despite the lack of theoretical justification for the post-processing step. The Convex-Concave Relaxation Procedure (CCRP) used in [23, 33], in contrast, maintains the constraints, but involves solving a series of non-convex optimization problems; it thus is not guaranteed to find a high-quality solution, and may converge slowly. Despite these limitations the CCRP-based method Factorised Graph Matching[33] is considered to be the current state-of-the-art as its performance is superior to that of competing methods in many scenarios.

In this paper, we revisit the problem of using graphical models for feature matching. We find that graphical model based feature matching can be both fast and accurate. We first formulate the matching problem as MAP inference in a graphical model with one-to-one matching constraints, and use a *tighter* linear programming (LP) relaxation than has previously been identified. We then propose an augmented belief propagation scheme to solve the dual problem of the LP relaxation. Unlike other dual methods such as [29], we decompose the dual problem into sub-problems which either can be efficiently solved in terms of max-weight bipartite matching (e.g. using the Hungarian algorithm), and others which have a closed-form solution. Note that the decomposition into sub-problems which can be cast as max-weight bipartite matching is possible only because of the specific form of our novel LP relaxation. The fact that such a relaxation is available in turn depends on the nature of the

one-to-one matching constraints, which is perhaps why a corresponding approach has not been discovered for general MAP inference problems. In this study, a classic Hungarian method is adopted for max-weight bipartite matching, but it should be noted that other methods (including fast approximations) instead of the Hungarian algorithm could also be used. Promising experimental results show the effectiveness and efficiency of the proposed Hungarian-BP method.

2. Notation and Problem Formulation

Assume that we have two sets of features: the model features $\mathcal{M} = \{f_{ij}^M | (i,j) \in [n] \times [n]\}$ and the data features $\mathcal{D} = \{f_{ij}^D | (i,j) \in [n] \times [n]\}$, where $[n]$ denotes the set $\{1, 2, \ldots, n\}$. Each set consists of n unary features and several pairwise features. Specifically, $f_{ii}^M, i \in [n]$ and $f_{jj}^D, j \in [n]$ are unary features, while f_{ij}^M, f_{ij}^D where $i, j \in [n], i \neq j$ are pairwise features. In a graph matching problem, for example, the unary features might represent nodes, and the pairwise features edges. We assume here that $\mathcal{M}$ and $\mathcal{D}$ have the same size, though this can easily be relaxed (e.g. to handle outliers) by generating additional 'dummy' points in either $\mathcal{M}$ or $\mathcal{D}$. Feature matching is the problem of finding the best correspondence between $\mathcal{M}$ and $\mathcal{D}$.

Given a correspondence vector $y_i = j$ ($i \in [n], j \in [n]$), meaning that feature i in $\mathcal{M}$ corresponds to feature j in $\mathcal{D}$, finding an optimal one-to-one matching can be formulated as a constrained MAP inference problem in a graphical model[1],

$$\mathbf{y}^* = \underset{\mathbf{y}}{\operatorname{argmax}} \left[\sum_{i \in \mathcal{V}} \theta_i(y_i) + \sum_{\{i,j\} \in \mathcal{E}} \theta_{ij}(y_i, y_j) \right], \quad (1)$$
$$\text{s.t. } \forall l \in [n], \sum_{i \in \mathcal{V}} \mathbb{1}(y_i = l) = 1,$$

where $y_i \in \{1, 2, \ldots, n\}$ and $\mathcal{V} = \{1, 2, \ldots, n\}$. The set $\mathcal{E} \subseteq \{\{i,j\} | i,j \in \mathcal{V}\}$ is known as the edge set. $\mathbb{1}(S)$ is the indicator function, which outputs 1 if S is true and 0 if S is false. $\theta_i(y_i)$ and $\theta_{ij}(y_i, y_j)$ are real-valued functions, known as potentials and are given by

$$\theta_i(y_i) = \phi(f_{ii}^M, f_{y_i y_i}^D),$$
$$\theta_{ij}(y_i, y_j) = \varphi(f_{ij}^M, f_{y_i y_j}^D) + \varphi(f_{ji}^M, f_{y_j y_i}^D), \quad (2)$$

where ϕ and φ are real-valued functions, which measure the similarity between the features (ϕ for unary features, and φ for pairwise features). The edge set $\mathcal{E}$ is a subset of $\mathcal{V} \times \mathcal{V}$, which is defined as follows

$$\mathcal{E} = \{\{i,j\} | \exists y_i, y_j \in [n], \text{ s.t. } \theta_{ij}(y_i, v_j) \neq 0\}. \quad (3)$$

[1]Details are provided in the supplementary file.

3. Inference by Iterative Bipartite Matching

Problem (1) is NP-hard in general, thus we search for efficient and accurate approximations. Belief propagation (BP) provides an efficient means of producing approximate solutions[26, 32] for the MAP inference problem, and exact solutions in certain specific cases. However, general BP methods cannot encode one-to-one matching constraints (as they introduce an interdependence among *all* variables), thus unfortunately they can not be used to solve (1) directly. To address this problem, we propose an augmented BP method for solving (1) using an LP relaxation. One of the reasons to adopt LP relaxations is that they allow a tighter approximation to the original cost function than many other relaxations[18].

3.1. LP Relaxation and Its Dual

Previously, an LP relaxation was proposed via composition of roof-duality-relaxation and the one-to-one matching constraints[29]. Then a dual decomposition method was proposed to solve the relaxed problem. One drawback of the relaxation in [29] is that it is relatively loose; thus it often leads to inferior solutions. To improve the solution quality, they added so called local sub-problems solved by exhaustive search, which makes the resulting algorithm computationally expensive.

To overcome the issues of looseness and speed, we propose a new LP relaxation for (1), which provides a better approximation than [29], and can be solved much more efficiently with a speed up of two orders of magnitude. First we introduce the typical LP relaxation for an unconstrained MAP inference problem,

$$\max_{\boldsymbol{\mu} \in \mathbb{L}} \Big[\sum_{i \in \mathcal{V}} \langle \mu_i, \theta_i \rangle + \sum_{\{i,j\} \in \mathcal{E}} \langle \mu_{ij}, \theta_{ij} \rangle \Big], \quad (4)$$

where $\langle \mu_i, \theta_i \rangle = \sum_{y_i} \mu_i(y_i) \theta_i(y_i)$, and $\langle \mu_{ij}, \theta_{ij} \rangle = \sum_{y_i, y_j} \mu_{ij}(y_i, y_j) \theta_{ij}(y_i, y_j)$. The set $\mathbb{L}$, known as the local marginal polytope[30], is defined as

$$\mathbb{L} = \left\{ \boldsymbol{\mu} \geqslant \mathbf{0} \,\middle|\, \begin{array}{l} \sum_{y_i} \mu_i(y_i) = 1, \\ \sum_{y_j} \mu_{ij}(y_i, y_j) = \mu_i(y_i), \\ \sum_{y_i} \mu_{ij}(y_i, y_j) = \mu_j(y_j) \end{array} \right\}. \quad (5)$$

In the LP relaxation (4), $\mu_i(y_i)$ plays a similar role to a permutation matrix by capturing a loose notion of a one-to-one correspondence. Following this we propose the following LP relaxation with one-to-one matching constraints:

$$\max_{\boldsymbol{\mu} \in \mathbb{L}} \Big[\sum_{i \in \mathcal{V}} \langle \mu_i, \theta_i \rangle + \sum_{\{i,j\} \in \mathcal{E}} \langle \mu_{ij}, \theta_{ij} \rangle \Big], \quad (6)$$
$$\text{s.t. } \forall l \in [n], \sum_{i \in \mathcal{V}} \mu_i(l) = 1.$$

Table 1. Introduced dual variables

Constraints	Dual Variables
$\forall i \in \mathcal{V}, \sum_{y_i} \mu_i(y_i) = 1$	u_i
$\forall l \in [n], \sum_{i \in \mathcal{V}} \mu_i(l) = 1$	v_l
$\forall \{i,j\}, y_i, \sum_{y_j} \mu_{ij}(y_i, y_j) = \mu_i(y_i)$	$\lambda_{j \to i}(y_i)$
$\forall \{i,j\}, y_j, \sum_{y_i} \mu_{ij}(y_i, y_j) = \mu_j(y_j)$	$\lambda_{i \to j}(y_j)$

In (6), the additional constraints $\sum_{i \in \mathcal{V}} \mu_i(l) = 1$ enforce the matrix $\mathrm{mat}(\boldsymbol{\mu})$ with $\mu_i(y_i)$ as its entry i^{th} row and y_i^{th} column to be a permutation matrix. As local marginal polytopes provide a more compact feasible set than roof-duality-relaxations, the proposed LP relaxation constitutes a closer approximation than the one in [29]. Commercial LP solvers such as CPLEX can be used to solve problems such as (6) but have been shown to be very slow in doing so[32] as they do not exploit the particular structure of the problem. We have thus tailor-made an efficient solver for this problem using dual coordinate descent and max-weight bipartite matching.

The Lagrangian dual of previous LP-relaxation-based BPs[26] only involves dual variables (*i.e.* Lagrange multipliers), which are referred to as messages (*i.e.* $\lambda_{i \to j}(y_j)$ and $\lambda_{j \to i}(y_i)$). Thus a straightforward way to derive the dual of (6) is to introduce dual variables for additional one-to-one matching constraints in (6).

However, since $\mu_i(y_i)$ serves a similar role to a permutation matrix, we find that introducing dual variables u_i corresponding to the constraints $\sum_{y_i} \mu_i(y_i) = 1$, and dual variables (*i.e.* v_l) for the additional one-to-one matching constraints in (6) leads to a more clear connection between the proposed dual and the max-weight bipartite matching problem[3]. Introducing the dual variables in Table 1 yields the dual problem below,

$$\begin{aligned}
&\min_{\mathbf{u},\mathbf{v},\boldsymbol{\lambda}} g(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{v}) = \sum_{i \in \mathcal{V}} u_i + \sum_{l \in [n]} v_l \\
&+ \sum_{i \in \mathcal{V}} \max_{y_i} \left[\theta_i(y_i) + \sum_{j \in N(i)} \lambda_{j \to i}(y_i) - u_i - v_{y_i} \right] \\
&+ \sum_{\{i,j\} \in \mathcal{E}} \max_{y_i, y_j} \left[\theta_{ij}(y_i, y_j) - \lambda_{j \to i}(y_i) - \lambda_{i \to j}(y_j) \right], \quad (7)
\end{aligned}$$

where $\mathbf{u} = [u_1, \ldots, u_n]$, $\mathbf{v} = [v_1 \ldots, v_n]$, $\boldsymbol{\lambda} = [\lambda_{i \to j}(y_j)]_{\{i,j\} \in \mathcal{E}}$ and $N(i) = \{j | \{i,j\} \in \mathcal{E}\}$ is the set of neighbours of node i. The variables $\boldsymbol{\lambda}$ are referred to as messages as in previous work[26].

A nice property of this dual problem is that one of its sub-problems can be smoothly transformed to a max-weight bipartite matching problem[3] (see Proposition 1), which can be efficiently solved by the Hungarian algorithm and a variety of other methods[11]. Without u_i, the dual form is not as convenient.

Since the variables $\mathbf{u}$ and $\mathbf{v}$ correspond to one-to-one matching constraints, we name them matching variables throughout the paper.

3.2. Sub-Problems Solved by Max-Weight Bipartite Matching

Fixing $\boldsymbol{\lambda}$, the problem of updating $\mathbf{u}$ and $\mathbf{v}$ in (7) becomes

$$\begin{aligned}
\min_{\mathbf{u},\mathbf{v}} &\sum_{i \in \mathcal{V}} \max_{y_i} \left[\theta_i(y_i) + \sum_{j \in N(i)} \lambda_{j \to i}(y_i) - u_i - v_{y_i} \right] \\
&+ \sum_{i \in \mathcal{V}} u_i + \sum_{l \in [n]} v_l. \quad (8)
\end{aligned}$$

Instead of solving sub-problem (8) directly, one can solve the following weighted bipartite matching problem

$$\begin{aligned}
\textbf{Primal: } \mathbf{X}^\star &= \operatorname*{argmax}_{\mathbf{X}} \sum_{i \in [n]} \sum_{l \in [n]} c_{il} \mathbf{X}_{il} \quad (9a) \\
\text{s.t. } &\sum_{i \in [n]} \mathbf{X}_{il} = 1; \sum_{l \in [n]} \mathbf{X}_{il} = 1; \mathbf{X}_{il} \geqslant 0, \forall i, l, \\
\textbf{Dual: } [\mathbf{u}^\star, \mathbf{v}^\star] &= \operatorname*{argmin}_{\mathbf{u},\mathbf{v}} \sum_{i \in \mathcal{V}} u_i + \sum_{l \in [n]} v_l \quad (9b) \\
\text{s.t. } &u_i + v_l \geqslant c_{il}, \forall i, l,
\end{aligned}$$

where $\mathbf{X}$ is a permutation matrix and the coefficients c_{il} are determined via

$$c_{il} = \theta_i(l) + \sum_{j \in N(i)} \lambda_{j \to i}(l), \forall i, l \in [n].$$

Proposition 1. *Sub-problem* (8) *is equivalent to the max-weight bipartite matching problem in* (9).

Proof. Associating u_i to constraints $\sum_{l \in [n]} \mathbf{X}_{il} = 1$ and v_l to constraints $\sum_{i \in [n]} \mathbf{X}_{il} = 1$ results in the dual of (9a) as follows,

$$\begin{aligned}
h(\mathbf{u}, \mathbf{v}) = &\max_{\mathbf{X} \geqslant 0} \Bigg[\sum_{i \in [n]} \sum_{l \in [n]} c_{il} \mathbf{X}_{il} - \sum_{i \in [n]} u_i \Big(\sum_{l \in [n]} \mathbf{X}_{il} - 1 \Big) \\
&- \sum_{l \in [n]} v_l \Big(\sum_{i \in [n]} \mathbf{X}_{il} - 1 \Big) \Bigg] \\
= &\max_{\mathbf{X} \geqslant 0} \sum_{i \in [n]} \sum_{l \in [n]} (c_{il} - u_i - v_l) \mathbf{X}_{il} + \sum_{i \in [n]} u_i + \sum_{l \in [n]} v_l.
\end{aligned}$$

Now we have two different approaches to further simplify the dual. One approach is adding the constraints

$\sum_{l\in[n]} \mathbf{X}_{il} = 1, \forall i \in [n]$ back, which yields

$$\begin{aligned} h(\mathbf{u},\mathbf{v}) =& \max_{\mathbf{X}\geqslant 0} \sum_{i\in[n]}\sum_{l\in[n]} (c_{il}-u_i-v_l)\,\mathbf{X}_{il} \\ &+ \sum_{i\in[n]} u_i + \sum_{l\in[n]} v_l, \text{s.t.} \sum_{l\in[n]} \mathbf{X}_{il} = 1, \forall i \in [n] \\ =& \sum_{i\in[n]} \max_{l} (c_{il}-u_i-v_l) + \sum_{i\in[n]} u_i + \sum_{l\in[n]} v_l \\ =& \sum_{i\in\mathcal{V}} \max_{y_i} \big[\theta_i(y_i) + \sum_{j\in N(i)} \lambda_{j\to i}(y_i) - u_i - v_{y_i}\big] \\ &+ \sum_{i\in\mathcal{V}} u_i + \sum_{l\in[n]} v_l. \end{aligned}$$

The other approach is as follows

$$\begin{aligned} h(\mathbf{u},\mathbf{v}) &= \sum_{i\in[n]} \max_{\mathbf{X}_{il}\geqslant 0} \sum_{l\in[n]} (c_{il}-u_i-v_l)\,\mathbf{X}_{il} + \sum_{i\in[n]} u_i + \sum_{l\in[n]} v_l \\ &= \begin{cases} \sum_{i\in[n]} u_i + \sum_{l\in[n]} v_l & \text{all } c_{il}-u_i-v_l \leqslant 0, \\ +\infty, & \text{otherwise.} \end{cases} \end{aligned}$$

Thus the two dual formulation (8) and (9b) are equivalent. □

Various methods, for example the Hungarian algorithm, can be used to determine the optimal $\mathbf{u}^\star$, $\mathbf{v}^\star$, as well as the optimal assignment $\mathbf{X}^\star$. In this study, we use the classical Hungarian algorithm implemented by Jonker and Volgenant[16]. However in practice the running time associated with the bipartite matching method did not prove to be a limiting factor.

Next, we show how to update $\boldsymbol{\lambda}$. Here, at each step, we choose some $\{i,j\} \in \mathcal{E}$ and update $\lambda_{j\to i}(y_i)$, $\lambda_{i\to j}(y_j)$ with all other dual variables fixed. Thus the sub-problem becomes

$$\begin{aligned} &\min_{\lambda_{j\to i},\lambda_{i\to j}} \max_{y_i} \big[\theta_i(y_i) + \sum_{j'\in N(i)} \lambda_{j'\to i}(y_i) - u_i - v_{y_i}\big] \\ &+ \max_{y_j} \big[\theta_j(y_j) + \sum_{j'\in N(j)} \lambda_{j'\to j}(y_j) - u_j - v_{y_j}\big] \\ &+ \max_{y_i,y_j} \big[\theta_{ij}(y_i,y_j) - \lambda_{j\to i}(y_i) - \lambda_{i\to j}(y_j)\big]. \end{aligned} \tag{10}$$

Fortunately, a closed-form solution of (10) exists. Let

$$\begin{aligned} b_i(y_i) &= \theta_i(y_i) + \sum_{j\in N(i)} \lambda_{j\to i}(y_i) - u_i - v_{y_i}, \\ b_{ij}(y_i,y_j) &= \theta_{ij}(y_i,y_j) - \lambda_{j\to i}(y_i) - \lambda_{i\to j}(y_j). \end{aligned} \tag{11}$$

By using a similar derivation as in [12], one MPLP-like

Algorithm 1: The Hungarian-BP procedure

input : Potentials $\theta_i(y_i), i\in\mathcal{V}, \theta_{ij}(y_i,y_j), \{i,j\}\in\mathcal{E}$; MaxIter; threshold ϵ_1 and ϵ_2.
output: $\mathbf{y}^\star$.

$f_{\max} = -\infty$, $\mathbf{u} = \mathbf{0}$, $\mathbf{v} = \mathbf{0}$;
for $k \in \{1,2,\ldots,\text{MaxIter}\}$ **do**
  for $\{i,j\} \in \mathcal{E}$ **do**
    Compute $\lambda^\star_{j\to i}(y_i)$ and $\lambda^\star_{i\to j}(y_j)$ as in (12);
    $[\lambda_{j\to i}(y_i), \lambda_{i\to j}(y_j)] \leftarrow [\lambda^\star_{j\to i}(y_i), \lambda^\star_{i\to j}(y_j)]$;
  Compute optimal $\mathbf{u}^\star$, $\mathbf{v}^\star$ and $\mathbf{X}^\star$ of (9) by the Hungarian algorithm;
  $[\mathbf{u},\mathbf{v}] \leftarrow [\mathbf{u}^\star,\mathbf{v}^\star]$;
  Decode $\hat{\mathbf{y}}$ as in (13),
    $f_k = \sum_{i\in\mathcal{V}} \theta_i(\hat{y}_i) + \sum_{\{i,j\}\in\mathcal{E}} \theta_{ij}(\hat{y}_i,\hat{y}_j)$;
  If $f_k > f_{\max}$ **then** $f_{\max} = f_k$, $\mathbf{y}^\star = \hat{\mathbf{y}}$;
  $g_k \leftarrow$ current dual objective of (7);
  if $|f_{\max} - g_k| < \epsilon_1$ *or* $|g_k - g_{k-1}| < \epsilon_2$ **then**
    break;

closed-form solution of (10) is

$$\begin{aligned} \lambda^\star_{j\to i}(y_i) =& \lambda_{j\to i}(y_i) - \frac{1}{2} b_i(y_i) \\ &+ \frac{1}{2} \max_{y_j} \big[b_{ij}(y_i,y_j) + b_j(y_j)\big], \\ \lambda^\star_{i\to j}(y_j) =& \lambda_{i\to j}(y_j) - \frac{1}{2} b_j(y_j) \\ &+ \frac{1}{2} \max_{y_i} \big[b_{ij}(y_i,y_j) + b_i(y_i)\big]. \end{aligned} \tag{12}$$

The overall procedure is summarized in Algorithm 1. We call this Hungarian-BP, since the Hungarian algorithm is used to update matching variables $\mathbf{u}$ and $\mathbf{v}$, and BP updates the messages $\boldsymbol{\lambda}$. Since it is a dual method, we use a decoding strategy to obtain a feasible integer (primal) solution.

Decoding In traditional belief propagation methods such as MPLP[12], the integer solution is decoded via $\bar{y}_i = \text{argmax}_{y_i}\, b_i(y_i)$. When multiple maiximizers exist, one of them ($\bar{y}_i$) can be chosen randomly. However, this scheme may give rise to an infeasible integer solution. On the contrary, in our Hungarian-BP framework, as the optimal $\mathbf{X}^\star$ is also provided, a feasible $\hat{\mathbf{y}}$ can be decoded from $\mathbf{X}^\star$ via

$$\hat{y}_i = \operatorname*{argmax}_{l} \mathbf{X}_{il}\,. \tag{13}$$

We would like to point out that $\hat{y}_i$ also maximizes $b_i(y_i)$.

Remarks In other dual decomposition frameworks such as [17, 31] for MAP inference, constraints are often considered as higher order potentials (HOPs), and they essentially

consider the following sub-problem,

$$\min_{\boldsymbol{\delta}} \max_{\mathbf{y}} \left[\theta(\mathbf{y}) - \sum_{i\in\mathcal{V}} \delta_i(y_i)\right] + \sum_{i\in\mathcal{V}} \max_{y_i} \left[\bar{\theta}_i(y_i) + \delta_i(y_i)\right],$$

where the high-order potential $\theta(\mathbf{y})$ enforces the one-to-one matching constraints,

$$\theta(\mathbf{y}) = \begin{cases} 0, & \sum_{i\in\mathcal{V}} \mathbb{1}(y_i = l) = 1, \forall l \in [n], \\ -\infty, & \text{otherwise}, \end{cases}$$

and $\bar{\theta}_i(y_i) = \theta_i(y_i) + \sum_{j\in N(i)} \lambda_{j\to i}(y_i)$. Here $\boldsymbol{\delta}$ are the messages from the high-order term to node.

In our setting, one can show that one optimal $\boldsymbol{\delta}^*$ can be obtained from the optimal $\mathbf{u}^*$ and $\mathbf{v}^*$ by $\delta_i^*(y_i) = -u_i^* - v_{y_i}^*, \forall i \in \mathcal{V}, y_i \in [n]$. Expressing $\boldsymbol{\delta}$ in terms of $\mathbf{u}$ and $\mathbf{v}$ yields our sub-problem (9), which can be efficiently solved as shown.

3.3. Fast Dual Objective Evaluation

In Hungarian-BP, the dual and primal objectives are evaluated iteratively. Dual objective evaluation can be expensive because it requires a lot of maximization operations. For example evaluating the dual via (7) has a time complexity of $\mathcal{O}(n^2|\,\mathcal{E}\,|)$. Thus a fast evaluation of the dual objective is desirable. For this purpose, we precisely arranged the order of dual variable updates. In Algorithm 1, we update messages first and then update matching variables. This is because our framework has convenient properties given by the following propositions.

Proposition 2. *The closed-form solution of* (10) *in* (12) *satisfies*[2]

$$\max_{y_i, y_j} \left[\theta_{ij}(y_i, y_j) - \lambda^\star_{j\to i}(y_i) - \lambda^\star_{i\to j}(y_j)\right] = 0.$$

Proposition 3. *The optimal* $\mathbf{u}^\star$ *and* $\mathbf{v}^\star$ *of* (8) *satisfies that* $\forall i \in \mathcal{V}$

$$\max_{y_i} \left[\theta_i(y_i) + \sum_{j\in N(i)} \lambda_{j\to i}(y_i) - u_i^* - v_{y_i}^*\right] = 0.$$

By these two properties, after message updating, at each iteration of Algorithm 1, the dual objective can be evaluated as

$$g_k = \sum_{i\in\mathcal{V}} u_i + \sum_{l\in[n]} v_l, \tag{14}$$

because all other terms in (7) are zero. Using (14), we can evaluate the dual with a time complexity of $\mathcal{O}(n)$ instead of $\mathcal{O}(n^2|\,\mathcal{E}\,|)$.

[2]Proof of all propositions excepet Proposition 1 are provided in supplementary file.

3.4. Analysis

In this section, we analyse the time complexity, convergence and exactness conditions for Hungarian-BP.

Time complexity Hungarian-BP has a time complexity of $\mathcal{O}(n^3 + |\,\mathcal{E}\,|n^2)$ per iteration, which is much better than that of [29] of $\mathcal{O}(n^6|\,\mathcal{E}\,|^2)$ per iteration. In the Hungarian-BP procedure, solving the weighted bipartite matching problem has complexity $\mathcal{O}(n^3)$. Finding closed-form solutions of each small-scale problem like (10) has complexity $\mathcal{O}(n^2)$. As there are $|\,\mathcal{E}\,|$ such problems, the time complexity of each iteration in total is $\mathcal{O}(n^3 + |\,\mathcal{E}\,|n^2)$. The work of Torresani, Kolmogorov and Rother[29] needs to solve a maxflow problem with n^2 nodes and $n^2|\,\mathcal{E}\,|$ edges at each iteration. Thus the complexity per iteration is at least $\mathcal{O}(n^6|\,\mathcal{E}\,|^2)$. Though neither Hungarian matching nor max-flow are nearly as slow as their worst-case running time might imply, in our experiments we find that Hungarian-BP can be up to two orders of magnitude faster than a max-flow-based approach.

In each iteration of our algorithm, other methods besides the Hungarian algorithm could be used to solve weighted bipartite matching problems as in (9). As far as we know, the best complexity for solving (9) is $\mathcal{O}(n^{5/2}\log(n\,\mathcal{C}))$[3][11, 14, 25], where $\mathcal{C} = \max c_{il}$. However, as in the worst case $|\,\mathcal{E}\,|$ can be n^2, solving weighted bipartite matching problems is not the bottleneck of the whole complexity. Thus switching to a faster algorithm instead of the Hungarian algorithm may only marginally improve the time complexity.

Convergence The dual objective of Hungarian-BP decreases at each iteration. By the fact that the dual objective is bounded from below by the true MAP value, Hungarian-BP provides a sequence of converged dual objectives.

Exactness Here we provide two sufficient conditions for Hungarian-BP to provide exact solutions of the constrained MAP inference problem.

Proposition 4. *In each iteration of Algorithm 1, after updating message and matching variables, if* $\sum_{\{i,j\}\in\mathcal{E}} b_{ij}(\hat{y}_i, \hat{y}_j) = 0$, *then* $\hat{\mathbf{y}}$ *is a solution of* (1).

Proposition 5. *If in Algorithm 1, for some* k *the two conditions hold: (1)* $g_k = g_{k-1}$*; (2) for each* $b_i(y_i)$*, there is a unique maximizer* $\bar{y}_i = \operatorname{argmax}_{y_i} b_i(y_i)$*; then* $\mathbf{y}^\star$ *is a solution of* (1).

When there is a gap between the dual objective and decoded integer solutions, various approaches including cluster pursuit techniques[1, 27] and branch-and-bound

[3]To simplify the derivation, we assume that all c_{il} are bounded.

techniques[28] can be used to tighten the initial relaxation. In our implementation, we use a similar branch-and-bound framework as [28], where the split strategies are most-fractional-first. In each branch step of branch-and-bound, we run the relaxation solver (with MaxIter=5 and $\epsilon_1 = \epsilon_2 = 10^{-6}$). If there is a gap between the dual objective and decoded primal, we use most-fractional-first to select a node. Then we branch the state space of $\mathbf{x}$ in two parts as follows. In one part the selected node must be assigned its current label, and in the other part it must not be assigned to its current label. The branched state space is organized as a queue and the procedure terminates once the queue is empty or maximal iterations are reached.

4. Experiments

In this section, we apply Hungarian-BP[4] to several matching tasks. Our Hungarian-BP method is compared with several existing popular and state-of-the-art matching algorithms, including:

- the graduated assignment algorithm (denoted as "GA" for short)[13];
- the spectral matching algorithm (denoted as "SM" for short)[19];
- the spectral matching algorithm with affine constraints (denoted as "SMAC" for short)[6];
- the integer projected fixed point matching algorithm with initialization $\mathbf{X}^0 = 1_{n\times n}/n$ (denoted as "IPFP-U" for short)[20];
- the integer projected fixed point matching algorithm with the result of SM as initialization (denoted as "IPFP-S" for short)[20];
- the reweighted random walks matching algorithm (denoted as "RRWM" for short)[4];
- the factorized graph matching algorithm (denoted as "FGM" for short)[33];
- the local sparse model matching algorithm (denoted as "LSM" for short)[15];
- the MAP inference based dual decomposition matching algorithm (denoted as "DD" for short)[29].

We conducted experiments on a server with two 12-core Xeon X5650 CPUs and 96 GB memory. The seven algorithms, GA, SM, SMAC, IPFP, RRWM, FGM[5] and DD[6] were based on public implementations, and the LSM algorithm was implemented by us. Our algorithm is implemented in Matlab with mex files. In the experiments we set MaxIter $= 5$, and $\epsilon_1 = \epsilon_2 = 10^{-6}$ for Algorithm 1. Then if there is a gap between the dual objective and decoded primal, we run at most 600 branch-and-bound iterations to tighten the bound. For LSM, we run at most 10000 iterations, or stop when the objective difference is less than 10^{-6}, whichever comes first. For DD, we add the linear subproblems, maxflow subproblems and local subproblems with size 2 (for details on parameter settings we refer to [29]). For all other algorithms, we use the same parameter setting as [33].

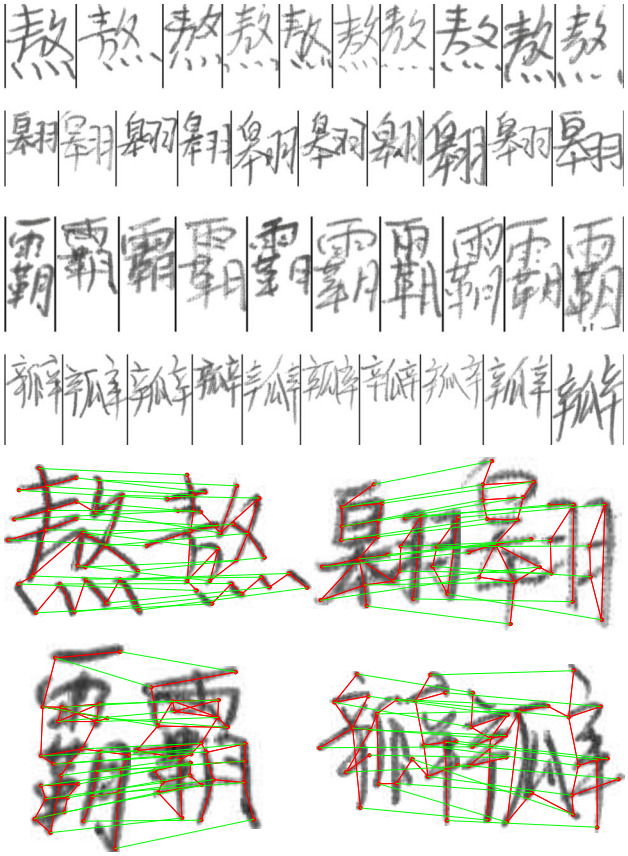

Figure 1. The four hand-written Chinese characters used in the experiment, and typical matching results. In the matching results, the red lines show the structure representations of characters given by Liu *et al*. [23].

4.1. Chinese Character Matching

As our first experiment we report feature matching results on four hand-written Chinese characters shown in Figure 1, where each character has 10 different samples[22]. In this experiment, we use the manually labelled feature points (each character has 28, 23, 28 and 23 feature points)

[4]The code is available at http://zzhang.org/software/.

[5]The code of "GA, SM, SMAC, IPFP, RRWM and FGM" is available at http://humansensing.cs.cmu.edu/down_fgm.html

[6]http://pub.ist.ac.at/~vnk/software/GraphMatching-v1.01.src.zip

Table 2. Matching results on Chinese characters. The objective is normalized to $[0, 1]$, and the best objective and accuracy are in **bold**. For each character, we report the average performance on 45 matching problems.

	GA	SM	IPFP-U	IPFP-S	SMAC	RRWM	FGM	LSM	DD	Ours
Character1(Acc)	0.7429	0.4127	0.6690	0.4587	0.5849	0.8651	0.8246	0.1389	0.9119	**0.9159**
Character1(Obj)	0.8672	0.5964	0.9113	0.8416	0.7043	0.9256	0.9787	0.2104	**1.0000**	**1.0000**
Character1(Time)	0.0443	0.0290	0.0434	0.0686	0.0291	0.3819	4.7767	0.1978	2.2333	0.0948
Character2(Acc)	0.8937	0.6860	0.8116	0.8512	0.7643	0.9014	0.8280	0.3285	0.9140	**0.9188**
Character2(Obj)	0.9720	0.7690	0.9634	0.9609	0.8486	0.9942	0.9789	0.3977	**1.0000**	**1.0000**
Character2(Time)	0.0723	0.0194	0.0288	0.0397	0.0169	0.1894	4.2974	0.0335	0.9873	0.0914
Character3(Acc)	0.5762	0.7198	0.7087	0.8484	0.5278	0.7730	0.7698	0.1444	**0.8778**	**0.8778**
Character3(Obj)	0.6090	0.7532	0.9371	0.9483	0.6176	0.8902	0.9779	0.2482	0.9990	**0.9999**
Character3(Time)	0.0299	0.0207	0.0435	0.0600	0.0285	0.3883	4.6312	0.0346	5.1402	0.4438
Character4(Acc)	0.9314	0.8609	0.8754	0.9546	0.7681	0.9411	0.9353	0.3130	**0.9961**	**0.9961**
Character4(Obj)	0.9545	0.8290	0.9525	0.9838	0.8409	0.9740	0.9872	0.3792	**1.0000**	**1.0000**
Character4(Time)	0.0470	0.0214	0.0309	0.0449	0.0165	0.1840	3.7027	0.0247	0.8655	0.0625

and structure representations provided by Liu *et al*. [23][7]. The similarity between unary features is set to zero (*i.e*. ϕ in (2)), and the similarity between pairwise features (*i.e*. φ in (2)) is computed as follows

$$\varphi_{ij}(k,l) = \exp(-\frac{1}{2}|d_{ij}^M - d_{kl}^D| - \frac{1}{2}|\theta_{ij}^M - \theta_{kl}^D|)\, \mathbf{A}_{ij}^M\, \mathbf{A}_{kl}^D,$$

where d_{ij}^M is the Euclidean distance between feature points i, j, θ_{ij}^M is the angle of edge ij in $\mathcal{M}$, and similarly for d_{kl}^D and θ_{kl}^D in $\mathcal{D}$. $\mathbf{A}^D$ and $\mathbf{A}^M$ are adjacency matrices of feature points in $\mathcal{D}$ and $\mathcal{M}$ provided by Liu *et al*. [23]. As in [23], we test the algorithms on all possible sample pairs, *i.e*. 45 pairs for each character pair. The experimental results are shown in Table 2, and typical matching results by Hungarian-BP are shown in Figure 1. The proposed Hungarian-BP uniformly obtains the best results in terms of both accuracy and normalised objective. DD can produce results competitive with ours, however its speed is at least 10 times slower than Hungarian-BP.

4.2. Wide Baseline Image Matching

In this section, we perform feature matching on the CMU *house* sequence[4, 33]. The CMU *house* sequence consists of 111 images of a toy house captured from different view points. In each image there are 30 manually marked landmark points with known correspondences. We have matched all images spaced by 10, 20, ...90 frames and compute the average performance per separation gap. In the experiments, as in [33], the similarity between unary features is set to zero (*i.e*. ϕ in (2)), and the similarity between pairwise features (*i.e*. φ in (2)) is computed as follows,

$$\varphi_{ij}(k,l) = \exp(-(d_{ij}^M - d_{kl}^D)^2/2500)\, \mathbf{A}_{ij}^M\, \mathbf{A}_{kl}^D,$$

where d_{ij}^M and d_{kl}^D are Euclidean distances between two landmarks points. $\mathbf{A}^M$ and $\mathbf{A}^D$ are adjacency matrices of landmark points created through Delaunay triangulation. Results are shown in Figure 2, and we note that as the separation between frames increases, the accuracy of several algorithms drops precipitously. The four methods: IPFP-S, RRWM, FGM, and our Hungarian-BP exactly identify the correct match in all scenarios. Furthermore, in our matching BP, an upper bound of the matching problem is also provided. Thus we can conclude that the objective provided by our Hungarian-BP is within 0.5% of the global optimum.

4.3. Real-World Image Matching

In this section, we evaluate our method on the dataset from [21], which consists of 30 pairs of images of cars and 20 pairs of images of motorbikes from the Pascal 2007 dataset[10]. Each pair contains 30-60 ground-truth correspondences and several outliers. The similarity measure function ϕ and φ are the same as that of [33]. The results of experiments without outliers are shown in Table 3. For further investigation, we randomly added 1-20 outliers from the background to the matching problems, with the result shown in Figure 3.

Without outliers, our method always achieves the highest accuracy. It also achieves the best objective in the "Motorbikes" dataset, and the second best in the "Car" dataset. The speed of the Hungarian-BP is also quite competitive. In the Motorbike dataset, DD achieves the second best accuracy, but its speed is hundreds of times slower than that of our method. In the Car dataset, FGM achieves the second best accuracy, but its speed is 10 times slower than that of Hungarian-BP. When outliers exist, the running time of our algorithm increases with the number of outliers. However it is sill faster than the FGM method. From Figure 3, we can see that Hungarian-BP achieves the best accuracy in most cases, and its normalized objectives are also quite close to being the best.

[7] http://www.escience.cn/system/file?fileId=62549

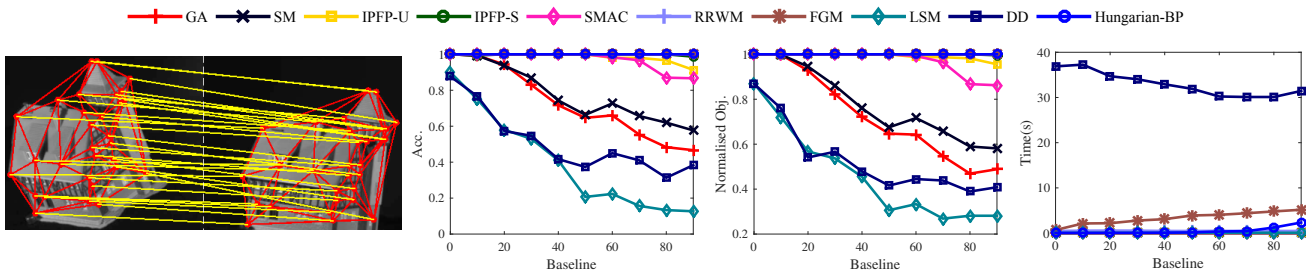

Figure 2. Matching results across image sequences with wide baseline.

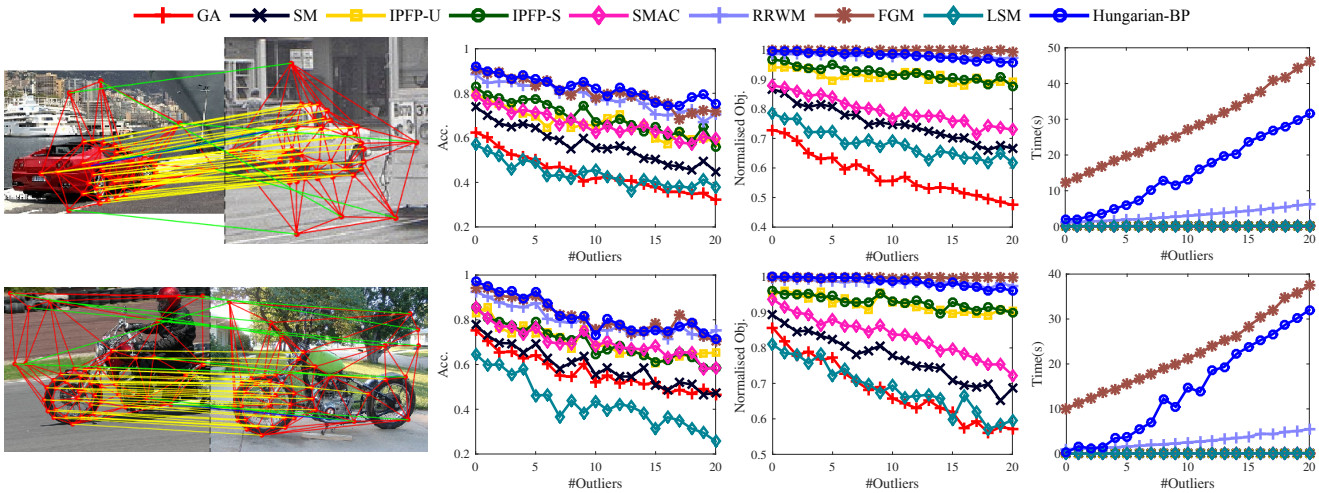

Figure 3. Matching results on real-world images. Typical matching result are shown on the left. Yellow lines indicate correct matches, blue lines indicates incorrect matches, and green lines indicate matches between outliers. The results of DD are not shown due to the prohibitive execution time.

Table 3. Matching results on real-world image matching, without outliers. The objective value is normalized to $[0, 1]$. The best accuracy and objective are in **bold**, and the second best are in *italic*.

	GA	SM	IPFP-U	IPFP-S	SMAC	RRWM	FGM	LSM	DD	Ours
Car (Acc.)	0.6246	0.7381	0.7976	0.8281	0.7913	0.8841	*0.9077*	0.5706	0.8883	**0.9218**
Car (Obj.)	0.7283	0.8666	0.9398	0.9662	0.8787	0.9865	**0.9991**	0.7854	0.9533	*0.9952*
Car (Time)	0.0523	0.0936	0.0564	0.0639	0.0692	1.1546	11.2468	0.0519	278.9577	1.1974
Motor (Acc.)	0.7531	0.7764	0.8298	0.8565	0.8565	0.9258	0.9405	0.6465	*0.9610*	**0.9713**
Motor (Obj.)	0.8573	0.8928	0.9492	0.9617	0.9377	0.9962	0.9983	0.8103	*0.9993*	**1.0000**
Motor (Time)	0.0613	0.1237	0.0442	0.0542	0.0880	0.9423	9.4517	0.0572	206.1469	0.2986

5. Conclusions

We have shown that it is possible to formulate matching problems as a constrained graphical model MAP inference problem suitable for the application of a novel LP relaxation. The advantage of this relaxation is that, as we have shown, it is possible using dual coordinate descent to devise an efficient solver, which we have named Hungarian-BP. Under certain conditions, the proposed LP relaxation is tight, in which case the Hungarian-BP method is guaranteed to achieve the global optimum. Experiments show that our algorithm often provides the best accuracy in real world matching problems at a greatly reduced computational cost.

Acknowledgments. This work was supported by NSFC Project 61301193, 61231016 and 61301192, ARC Project DP140102270 and DP160100703.

References

[1] D. Batra, S. Nowozin, and P. Kohli. Tighter relaxations for MAP-MRF inference: A local primal-dual gap based separation algorithm. In *AISTATS*, pages 146–154, 2011. 5

[2] M. Bayati, D. Shah, and M. Sharma. Maximum weight matching via max-product belief propagation. In *ISIT*, pages 1763–1767. IEEE, 2005. 1

[3] R. E. Burkard, M. Dell'Amico, and S. Martello. *Assignment Problems, Revised Reprint*. Siam, 2009. 3

[4] M. Cho, J. Lee, and K. M. Lee. Reweighted random walks

for graph matching. *Computer Vision Eccv*, pages 492–505, 2010. 6, 7

[5] D. Conte, P. Foggia, C. Sansone, and M. Vento. 30 years of graph matching in pattern recognition. *International Journal of Pattern Recognition & Artificial Intelligence*, 18(3):265–298, 2004. 1

[6] T. Cour, P. Srinivasan, and J. Shi. Balanced graph matching. *Advances in Neural Information Processing Systems*, 19:313, 2007. 6

[7] O. Duchenne, F. Bach, I.-S. Kweon, and J. Ponce. A tensor-based algorithm for high-order graph matching. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1980–1987. IEEE, 2009. 1

[8] O. Duchenne, A. Joulin, and J. Ponce. A graph-matching kernel for object categorization. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1792–1799. IEEE, 2011. 1

[9] J. Duchi, D. Tarlow, G. Elidan, and D. Koller. Using combinatorial optimization within max-product belief propagation. In *Advances in Neural Information Processing Systems 19: Proceedings of the 2006 Conference*, volume 19, page 369. MIT Press, 2007. 1

[10] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge 2009. In *2th PASCAL Challenge Workshop*, 2009. 7

[11] H. N. Gabow and R. E. Tarjan. Faster scaling algorithms for network problems. *SIAM Journal on Computing*, 18(5):1013–1036, 1989. 3, 5

[12] A. Globerson and T. Jaakkola. Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In *NIPS*, 2007. 4

[13] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 18(4):377–388, 1996. 6

[14] A. V. Goldberg and R. Kennedy. An efficient cost scaling algorithm for the assignment problem. *Mathematical Programming*, 71(2):153–177, 1995. 5

[15] B. Jiang, J. Tang, C. Ding, and B. Luo. A local sparse model for matching problem. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015. 1, 6

[16] R. Jonker and A. Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38(4):325–340, 1987. 4

[17] N. Komodakis, N. Paragios, and G. Tziritas. MRF optimization via dual decomposition: Message-passing revisited. In *ICCV*, pages 1–8. IEEE, 2007. 4

[18] M. P. Kumar, V. Kolmogorov, and P. H. Torr. An analysis of convex relaxations for MAP estimation of discrete MRFs. *The Journal of Machine Learning Research*, 10:71–106, 2009. 2

[19] M. Leordeanu and M. Hebert. A spectral technique for correspondence problems using pairwise constraints. In *Proceedings / IEEE International Conference on Computer Vision. IEEE International Conference on Computer Vision*, pages 1482–1489 Vol. 2, 2005. 1, 6

[20] M. Leordeanu, M. Hebert, and R. Sukthankar. An integer projected fixed point method for graph matching and map inference. In *Advances in neural information processing systems*, pages 1114–1122, 2009. 6

[21] M. Leordeanu, R. Sukthankar, and M. Hebert. Unsupervised learning for graph matching. *International journal of computer vision*, 96(1):28–45, 2012. 7

[22] C.-L. Liu, F. Yin, D.-H. Wang, and Q.-F. Wang. Casia online and offline chinese handwriting databases. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 37–41. IEEE, 2011. 6

[23] Z.-Y. Liu, H. Qiao, X. Yang, and S. C. Hoi. Graph matching by simplified convex-concave relaxation procedure. *International Journal of Computer Vision*, 109(3):169–186, 2014. 1, 6, 7

[24] J. Maciel and J. P. Costeira. A global solution to sparse correspondence problems. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(2):187–199, 2003. 1

[25] J. B. Orlin and R. K. Ahuja. New scaling algorithms for the assignment and minimum mean cycle problems. *Mathematical programming*, 54(1-3):41–56, 1992. 5

[26] D. Sontag, A. Globerson, and T. Jaakkola. Introduction to dual decomposition for inference. In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*. MIT Press, 2011. 2, 3

[27] D. Sontag, T. Meltzer, A. Globerson, Y. Weiss, and T. Jaakkola. Tightening LP relaxations for MAP using message-passing. In *UAI*, pages 503–510. AUAI Press, 2008. 5

[28] M. Sun, M. Telaprolu, H. Lee, and S. Savarese. Efficient and exact MAP-MRF inference using branch and bound. In *AISTATS*, pages 1134–1142, 2012. 6

[29] L. Torresani, V. Kolmogorov, and C. Rother. Feature correspondence via graph matching: Models and global optimization. In *Computer Vision–ECCV 2008*, pages 596–609. Springer, 2008. 1, 2, 3, 5, 6

[30] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305, 2008. 2

[31] T. Werner. Revisiting the linear programming relaxation approach to gibbs energy minimization and weighted constraint satisfaction. *TPAMI*, 32(8):1474–1488, 2010. 4

[32] C. Yanover, T. Meltzer, and Y. Weiss. Linear Programming Relaxations and Belief Propagation–An Empirical Study. *JMLR*, 7:1887–1907, 2006. 2, 3

[33] F. Zhou and F. De la Torre. Factorized graph matching. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 127–134. IEEE, 2012. 1, 6, 7

Supplementary file

A. Reformulation of The Quadratic Assignment Problem

The one-to-one feature matching problems are also often modelled as the following quadratic assignment problems

$$\begin{aligned} \max_{\mathbf{X}} \ & \mathrm{vec}(\mathbf{X})^\top \mathbf{K} \mathrm{vec}(\mathbf{X}), \\ \text{s.t.} \ & \forall i \in [n], \sum_{l=1}^{n} \mathbf{X}_{il} = 1, \\ & \forall l \in [n], \sum_{i=1}^{n} \mathbf{X}_{il} = 1, \end{aligned} \tag{15}$$

where $\mathbf{X}$ is a permutation matrix[8], $\mathbf{X}_{ij}$ is the entry in the i-th row and j-th column of $\mathbf{X}$, and $\mathrm{vec}(\mathbf{X}) = (\mathbf{X}_{11} \dots \mathbf{X}_{1n}, \dots, \mathbf{X}_{n1} \dots \mathbf{X}_{nn})^\top$ is the vectorised form of $\mathbf{X}$. The matrix $\mathbf{K}$ is in $\mathbb{R}^{n^2 \times n^2}$ and $\mathbf{K}_{ij,kl}$ measures similarity between the feature f_{ij}^M and f_{kl}^M.

It is obvious, any problem like (1) can be reformulated as (15) if we let

$$\mathbf{K}_{ii,ll} = \phi(f_{ii}^M, f_{ll}^D), \tag{16a}$$

$$\mathbf{K}_{ij,kl} = \varphi_(f_{ij}^M, f_{kl}^D), i \neq j, k \neq l. \tag{16b}$$

On the other hand, any problem like (15) can be reformulated as (1) if we let

$$\theta_i(y_i) = \mathbf{K}_{ii,y_i y_i}, \tag{17a}$$

$$\theta_{ij}(y_i, y_j) = \mathbf{K}_{ij,y_i y_j} + \mathbf{K}_{ji,y_j y_i}. \tag{17b}$$

B. Derivation of dual

To derive the dual, several redundant constraints will be added to the local marginal polytope $\mathbb{L}$, as these constraints does corresponding to any dual variables. For simplifying the derivation, we define

$$\mathcal{L} = \left\{ \boldsymbol{\mu} \;\middle|\; \begin{array}{l} \mu_i(y_i) \geqslant 0; \sum_{y_i} \mu_i(y_i) = 1; \forall i \in \mathcal{V}, y_i \in [n] \\ \mu_{ij}(y_i, y_j) \geqslant 0; \sum_{y_i, y_j} \mu_{ij}(y_i, y_j) = 1; \forall \{i, j\} \in \mathcal{E}, y_i, y_j \in [n] \end{array} \right\}. \tag{18}$$

Then with the dual variables in Table 1, the dual of (6) is

$$\begin{aligned} g(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{v}) = \max_{\boldsymbol{\mu} \in \mathcal{L}} \Bigg\{ & \sum_{i \in \mathcal{V}} \langle \mu_i, \theta_i \rangle + \sum_{\{i,j\} \in \mathcal{E}} \langle \mu_{ij}, \theta_{ij} \rangle - \sum_{\{i,j\} \in \mathcal{E}} \lambda_{j \to i}(y_i) \big[\sum_{y_j} \mu_{ij}(y_i, y_j) - \mu_i(y_i) \big] \\ & - \sum_{\{i,j\} \in \mathcal{E}} \lambda_{i \to j}(y_j) \big[\sum_{y_i} \mu_{ij}(y_i, y_j) - \mu_j(y_j) \big] \\ & - \sum_{i \in \mathcal{V}} u_i \big[\sum_{y_i} \mu_i(y_i) - 1 \big] - \sum_{l \in [n]} v_l \big[\sum_{i \in \mathcal{V}} \mu_i(l) - 1 \big] \Bigg\}. \end{aligned} \tag{19}$$

[8] $\mathbf{X}$ can also be a non-square partial permutation matrix. However, by adding zero-valued elements in matrix $\mathbf{K}$, such a problem can be reformulated as (15).

Obviously, the maximisation in the above equation is decomposable, which means that

$$
\begin{aligned}
g(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{v}) =& \sum_{i \in \mathcal{V}} \max_{\boldsymbol{\mu} \in \mathcal{L}} \sum_{y_i} \mu_i(y_i) \Big[\theta_i(y_i) + \sum_{j \in N(i)} \lambda_{j \to i}(y_i) - u_i - v_{y_i} \Big] \\
&+ \sum_{\{i,j\} \in \mathcal{E}} \max_{\boldsymbol{\mu} \in \mathcal{L}} \sum_{y_i, y_j} \mu_{ij}(y_i, y_j) \Big[\theta_{ij}(y_i, y_j) - \lambda_{i \to j}(y_j) - \lambda_{j \to i}(y_i) \Big] + \sum_{i \in \mathcal{V}} u_i + \sum_{l \in [n]} v_l \\
=& \sum_{i \in \mathcal{V}} \max_{\mu_i(y_i) \text{ is distribution.}} \sum_{y_i} \mu_i(y_i) \Big[\theta_i(y_i) + \sum_{j \in N(i)} \lambda_{j \to i}(y_i) - u_i - v_{y_i} \Big] \\
&+ \sum_{\{i,j\} \in \mathcal{E}} \max_{\mu_{ij}(y_i, y_j) \text{ is distribution.}} \sum_{y_i, y_j} \mu_{ij}(y_i, y_j) \Big[\theta_{ij}(y_i, y_j) - \lambda_{i \to j}(y_j) - \lambda_{j \to i}(y_i) \Big] + \sum_{i \in \mathcal{V}} u_i + \sum_{l \in [n]} v_l \\
=& \sum_{i \in \mathcal{V}} \max_{y_i} \Big[\theta_i(y_i) + \sum_{j \in N(i)} \lambda_{j \to i}(y_i) - u_i - v_{y_i} \Big] \\
&+ \sum_{\{i,j\} \in \mathcal{E}} \max_{y_i, y_j} \Big[\theta_{ij}(y_i, y_j) - \lambda_{i \to j}(y_j) - \lambda_{j \to i}(y_i) \Big] + \sum_{i \in \mathcal{V}} u_i + \sum_{l \in [n]} v_l,
\end{aligned} \tag{20}
$$

which finish the derivation of dual.

C. Proof of proposition 2

Proposition 2. *The closed-form solution of* (10) *in* (12) *satisfies*

$$
\max_{y_i, y_j} \left[\theta_{ij}(y_i, y_j) - \lambda^{\star}_{j \to i}(y_i) - \lambda^{\star}_{i \to j}(y_j) \right] = 0.
$$

Proof. Combining (12) and (11) yields

$$
\begin{aligned}
& \theta_{ij}(y_i, y_j) - \lambda^{\star}_{j \to i}(y_i) - \lambda^{\star}_{i \to j}(y_j) \\
=& \theta_{ij}(y_i, y_j) - \lambda_{j \to i}(y_i) + \frac{1}{2} b_i(y_i) - \frac{1}{2} \max_{y_j} \Big[b_{ij}(y_i, y_j) + b_j(y_j) \Big] \\
& - \lambda_{i \to j}(y_j) + \frac{1}{2} b_j(y_j) - \frac{1}{2} \max_{y_i} \Big[b_{ij}(y_i, y_j) + b_i(y_i) \Big] \\
=& \theta_{ij}(y_i, y_j) - \lambda_{j \to i}(y_i) - \lambda_{i \to j}(y_j) + b_i(y_i) + b_j(y_j) \\
& - \frac{1}{2} \max_{y_j} \Big[b_{ij}(y_i, y_j) + b_j(y_j) + b_j(y_i) \Big] - \frac{1}{2} \max_{y_i} \Big[b_{ij}(y_i, y_j) + b_i(y_i) + b_j(y_j) \Big] \\
=& b_{ij}(y_i, y_j) + b_i(y_i) + b_j(y_j) && \text{(21a)} \\
& - \frac{1}{2} \max_{y_j} \Big[b_{ij}(y_i, y_j) + b_j(y_j) + b_j(y_i) \Big] && \text{(21b)} \\
& - \frac{1}{2} \max_{y_i} \Big[b_{ij}(y_i, y_j) + b_i(y_i) + b_j(y_j) \Big]. && \text{(21c)}
\end{aligned}
$$

By (21) and the amplyfying trick, we have that

$$
\begin{aligned}
&\max_{y_i,y_j}\left[\theta_{ij}(y_i,y_j)-\lambda^\star_{j\to i}(y_i)-\lambda^\star_{i\to j}(y_j)\right]\\
=&\max_{y_i,y_j}\left\{\frac{1}{2}\left[b_{ij}(y_i,y_j)+b_i(y_i)+b_j(y_j)-\frac{1}{2}\max_{y_i}\left[b_{ij}(y_i,y_j)+b_j(y_j)+b_j(y_i)\right]\right]\right.\\
&\left.+\frac{1}{2}\left[b_{ij}(y_i,y_j)+b_i(y_i)+b_j(y_j)-\frac{1}{2}\max_{y_j}\left[b_{ij}(y_i,y_j)+b_j(y_j)+b_j(y_i)\right]\right]\right\}\\
\leqslant&\frac{1}{2}\max_{y_i,y_j}\left[b_{ij}(y_i,y_j)+b_i(y_i)+b_j(y_j)-\max_{y_i}\left[b_{ij}(y_i,y_j)+b_j(y_j)+b_j(y_i)\right]\right]\\
&+\frac{1}{2}\max_{y_i,y_j}\left[b_{ij}(y_i,y_j)+b_i(y_i)+b_j(y_j)-\max_{y_j}\left[b_{ij}(y_i,y_j)+b_j(y_j)+b_j(y_i)\right]\right].
\end{aligned}
\tag{22}
$$

Obviously $b_{ij}(y_i,y_j)+b_i(y_i)+b_j(y_j)\leqslant\max_{y_i}\left[b_{ij}(y_i,y_j)+b_j(y_j)+b_j(y_i)\right]$ holds, and the equality holds for entry $[\hat{y}_i,\hat{y}_j]=\operatorname{argmax}_{y_i,y_j}\left[b_{ij}(y_i,y_j)+b_j(y_j)+b_j(y_i)\right]$. As a result, we have that

$$
\max_{y_i,y_j}\left[b_{ij}(y_i,y_j)+b_i(y_i)+b_j(y_j)-\max_{y_i}\left[b_{ij}(y_i,y_j)+b_j(y_j)+b_j(y_i)\right]\right]=0.
$$

Similarly, we have that

$$
\max_{y_i,y_j}\left[b_{ij}(y_i,y_j)+b_i(y_i)+b_j(y_j)-\max_{y_j}\left[b_{ij}(y_i,y_j)+b_j(y_j)+b_j(y_i)\right]\right]=0.
$$

Thus we have that

$$
\theta_{ij}(y_i,y_j)-\lambda^\star_{j\to i}(y_i)-\lambda^\star_{i\to j}(y_j)\leqslant 0,
$$

On the other hand, it is obvious that

$$
\begin{aligned}
\max_{y_i}\left[b_{ij}(y_i,y_j)+b_j(y_j)+b_j(y_i)\right]&\leqslant\max_{y_i,y_j}\left[b_{ij}(y_i,y_j)+b_j(y_j)+b_j(y_i)\right],\\
\max_{y_j}\left[b_{ij}(y_i,y_j)+b_j(y_j)+b_j(y_i)\right]&\leqslant\max_{y_i,y_j}\left[b_{ij}(y_i,y_j)+b_j(y_j)+b_j(y_i)\right],
\end{aligned}
\tag{23}
$$

and thus we have

$$
\begin{aligned}
&\max_{y_i,y_j}\left[\theta_{ij}(y_i,y_j)-\lambda^\star_{j\to i}(y_i)-\lambda^\star_{i\to j}(y_j)\right]\\
\geqslant&\max_{y_i,y_j}\left\{b_{ij}(y_i,y_j)+b_i(y_i)+b_j(y_j)\right.\\
&\left.-\frac{1}{2}\max_{y_i,y_j}\left[b_{ij}(y_i,y_j)+b_j(y_j)+b_j(y_i)\right]-\frac{1}{2}\max_{y_i,y_j}\left[b_{ij}(y_i,y_j)+b_i(y_i)+b_j(y_j)\right]\right\}\\
=&\max_{y_i,y_j}\left\{b_{ij}(y_i,y_j)+b_i(y_i)+b_j(y_j)-\max_{y_i,y_j}\left[b_{ij}(y_i,y_j)+b_j(y_j)+b_j(y_i)\right]\right\}\\
=&\max_{y_i,y_j}\left[b_{ij}(y_i,y_j)+b_i(y_i)+b_j(y_j)\right]-\max_{y_i,y_j}\left[b_{ij}(y_i,y_j)+b_j(y_j)+b_j(y_i)\right]=0,
\end{aligned}
\tag{24}
$$

which completes the proof. □

D. Proof of Proposition 3

Proposition 3. *The optimal* $\mathbf{u}^\star$ *and* $\mathbf{v}^\star$ *of* (8) *satisfies that* $\forall i\in\mathcal{V}$

$$
\max_{y_i}\left[\theta_i(y_i)+\sum_{j\in N(i)}\lambda_{j\to i}(y_i)-u^*_i-v^*_{y_i}\right]=0.
$$

Proof. Considering the equivalence between (8) and (9), by complementary slackness, we have that

$$\mathbf{X}^{\star}_{ij}(c_{ij} - u^{\star}_i - v^{\star}_i) = 0,$$

then by the constraints $u_i + v_l \geqslant c_{il}$ we have that

$$\forall c_{il} - u^{\star}_i - v^{\star}_l \leqslant 0, \mathbf{X}^{\star}_{il} = 0,$$
$$\forall c_{il} - u^{\star}_i - v^{\star}_l = 0, \mathbf{X}^{\star}_{il} = 1.$$

By the fact that $\mathbf{X}$ is a permutation matrix, there must be one l' for each i, s.t. $\mathbf{X}^{\star}_{il'} = 1$, which means that

$$\begin{aligned}\max_{y_i}\left[\theta_i(y_i) + \sum_{j\in N(i)}\lambda_{j\to i}(y_i) - u^*_i - v^*_{y_i}\right] &= \max_{l}\left[c_{il} - u^*_i - v^*_l\right] \\ &= c_{il'} - u^{\star}_i - v^{\star}_{l'} = 0, \end{aligned} \tag{25}$$

which completes the proof. □

E. Proof of Proposition 4

Proposition 4. *In each iteration of Algorithm 1, after updating message and matching variables, if* $\sum_{\{i,j\}\in\mathcal{E}} b_{ij}(\hat{y}_i, \hat{y}_j) = 0$, *then* $\hat{\mathbf{y}}$ *is a solution of* (1).

Proof. The decoded primal can be computed as

$$\begin{aligned} f_k =& \sum_{i\in\mathcal{V}}\theta_i(\hat{y}_i) + \sum_{\{i,j\}\in\mathcal{E}}\theta_{ij}(\hat{y}_i,\hat{y}_j) \\ =& \sum_{i\in\mathcal{V}}\left[\theta_i(\hat{y}_i) + \sum_{j\in N(i)}\lambda_{j\to i}(\hat{y}_i) - \sum_{j\in N(i)}\lambda_{j\to i}(\hat{y}_i) + u_i + v_{\hat{y}_i} - u_i - v_{\hat{y}_i}\right] + \sum_{\{i,j\}\in\mathcal{E}}\theta_{ij}(\hat{y}_i,\hat{y}_j), \end{aligned}$$

where $\hat{\mathbf{y}}$ is the decoded integer solution in current iteration as (13). Then by the fact that there is exactly one y_i to be assigned to a particular label $l \in [n]$, f_k can be reformulated as

$$\begin{aligned} f_k =& \sum_{i\in\mathcal{V}}\left[\theta_i(\hat{y}_i) + \sum_{j\in N(i)}\lambda_{j\to i}(\hat{y}_i) - \sum_{j\in N(i)}\lambda_{j\to i}(\hat{y}_i) + u_i + v_{\hat{y}_i} - u_i - v_{\hat{y}_i}\right] + \sum_{\{i,j\}\in\mathcal{E}}\theta_{ij}(\hat{y}_i,\hat{y}_j) \\ =& \sum_{i\in\mathcal{V}}\left[\theta_i(\hat{y}_i) + \sum_{j\in N(i)}\lambda_{j\to i}(\hat{y}_i) - u_i - v_{\hat{y}_i}\right] + \sum_{\{i,j\}\in\mathcal{E}}\left[\theta_{ij}(\hat{y}_i,\hat{y}_j) - \lambda_{j\to i}(\hat{y}_i) - \lambda_{i\to j}(\hat{y}_j)\right] \\ &+ \sum_{i\in[n]}u_i + \sum_{l\in[n]}v_l. \end{aligned}$$

By the decoding scheme in (13), we have $\mathbf{X}_{i\hat{y}_i} = 1$, and then by complementary slackness we must have

$$\theta_i(\hat{y}_i) + \sum_{j\in N(i)}\lambda_{j\to i}(\hat{y}_i) - u_i - v_{\hat{y}_i} = 0.$$

Then by the fact that the dual objective can be evaluated as $g_k = \sum_{i\in[n]}u_i + \sum_{l\in[n]}v_l$, we have

$$\begin{aligned} f_k &= \sum_{\{i,j\}\in\mathcal{E}}\left[\theta_{ij}(\hat{y}_i,\hat{y}_j) - \lambda_{j\to i}(\hat{y}_i) - \lambda_{i\to j}(\hat{y}_j)\right] + \sum_{i\in[n]}u_i + \sum_{l\in[n]}v_l \\ &= \sum_{\{i,j\}\in\mathcal{E}}b_{ij}(\hat{y}_i,\hat{y}_j) + \sum_{i\in[n]}u_i + \sum_{l\in[n]}v_l \\ &= \sum_{\{i,j\}\in\mathcal{E}}b_{ij}(\hat{y}_i,\hat{y}_j) + g_k, \end{aligned}$$

which indicates the gap between f_k and g_k is $-\sum_{\{i,j\}\in\mathcal{E}}b_{ij}(\hat{y}_i,\hat{y}_j)$. Thus if $\sum_{\{i,j\}\in\mathcal{E}}b_{ij}(\hat{y}_i,\hat{y}_j) = 0$, we must have f_k attains g_k, which means that $\hat{\mathbf{y}}$ is the exact solution. □

F. Proof of Proposition 5

Proposition 5. *If in Algorithm 1, for some k the two conditions hold: (1) $g_k = g_{k-1}$; (2) for each $b_i(y_i)$, there is a unique maximizer $\bar{y}_i = \operatorname{argmax}_{y_i} b_i(y_i)$; then $\mathbf{y}^\star$ is a solution of (1).*

Proof. By Proposition 2 and 3, we have that after updating $\lambda_{i\to j}(y_j)$ and $\lambda_{j\to i}(y_i)$ to $\lambda^\star_{i\to j}(y_j)$ and $\lambda^\star_{j\to i}(y_i)$, the dual decrease is

$$\begin{aligned}
&\max_{y_i} b_i(y_i) + \max_{y_j} b_j(y_j) + \max_{y_i,y_j} b_{ij}(y_i,y_j) - \max_{y_i} b^\star_i(y_i) - \max_{y_j} b^\star_j(y_j) - \max_{y_i,y_j} b^\star_{ij}(y_i,y_j) \\
=&\max_{y_i} b_i(y_i) + \max_{y_j} b_j(y_j) + \max_{y_i,y_j} b_{ij}(y_i,y_j) - \max_{y_i,y_j} \Big[b_i(y_i) + b_j(y_j) + b_{ij}(y_i,y_j) \Big].
\end{aligned} \tag{26}$$

Thus if $g_k = g_{k-1}$, we must have

$$\begin{aligned}
&\max_{y_i} b_i(y_i) + \max_{y_j} b_j(y_j) + \max_{y_i,y_j} b_{ij}(y_i,y_j) - \max_{y_i} b^\star_i(y_i) - \max_{y_j} b^\star_j(y_j) - \max_{y_i,y_j} b^\star_{ij}(y_i,y_j) \\
=&\max_{y_i} b_i(y_i) + \max_{y_j} b_j(y_j) + \max_{y_i,y_j} b_{ij}(y_i,y_j) - \max_{y_i,y_j} \Big[b_i(y_i) + b_j(y_j) + b_{ij}(y_i,y_j) \Big] = 0,
\end{aligned} \tag{27}$$

which means that $b_{ij}(y_i,y_j)$ and $b_i(y_i)$, $b_j(y_j)$ must have common maximiser. By the fact that each $b_i(y_i)$ has a unique maximiser $\bar{y}_i$, the maximiser of $b_{ij}(y_i,y_j)$ is $\bar{y}_i, \bar{y}_j$. Then by Lemma 2 we must have

$$b_{ij}(\bar{y}_i,\bar{y}_j) = 0, \forall \{i,j\} \in \mathcal{E}\,.$$

Then by complementary slackness, we also have

$$\mathbf{X}^\star_{i\bar{y}_i} = 1,$$

and thus by Proposition 4, $\mathbf{y}^*$ must be a solution of (1). □

G. Derivation of Message Updating

In this section, we prove that (12) is a closed-form solution of (10). First we introduce the following definitions and lemma.

Firstly, we introduce the following definitions

$$\begin{aligned}
b^\star_i(y_i) &= \theta_i(y_i) + \sum_{j'\in N(i), j'\neq j} \lambda_{j'\to i}(y_i) + \lambda^\star_{j\to i}(y_i) - u_i - v_{y_i}, \\
b^\star_j(y_j) &= \theta_j(y_j) + \sum_{j'\in N(j), j'\neq i} \lambda_{j'\to j}(y_j) + \lambda^\star_{i\to j}(y_j) - u_i - v_{y_j}, \\
b^\star_{ij}(y_i,y_j) &= \theta_{ij}(y_i,y_j) - \lambda^\star_{j\to i}(y_i) - \lambda^\star_{i\to j}(y_j).
\end{aligned} \tag{28}$$

Then we introduce the following lemma.

Lemma 1. $\max_{y_i} b^\star_i(y_i) = \max_{y_j} b^\star_j(y_j) = \frac{1}{2}\max_{y_i,y_j}\big[b_{ij}(y_i,y_j) + b_i(y_i) + b_j(y_j)\big]$.

Proof.

$$
\begin{aligned}
\max_{y_i} b_i^\star(y_i) &= \max_{y_i} \left[\theta_i(y_i) + \sum_{j' \in N(i), j' \neq j} \lambda_{j' \to i}(y_i) + \lambda^\star_{j \to i}(y_i) - u_i - v_{y_i} \right] \\
&= \max_{y_i} \left[b_i(y_i) - \lambda_{j \to i}(y_i) + \lambda^\star_{j \to i}(y_i) \right] \\
&= \max_{y_i} \left[b_i(y_i) - \lambda_{j \to i}(y_i) + \lambda_{j \to i}(y_i) - \frac{1}{2} b_i(y_i) + \frac{1}{2} \max_{y_j} \left[b_{ij}(y_i, y_j) + b_j(y_j) \right] \right] \\
&= \max_{y_i} \left[\frac{1}{2} \max_{y_j} \left[b_{ij}(y_i, y_j) + b_i(y_i) + b_j(y_j) \right] \right] \\
&= \max_{y_i} \left[\frac{1}{2} \max_{y_j} \left[b_{ij}(y_i, y_j) + b_i(y_i) + b_j(y_j) \right] \right] \\
&= \frac{1}{2} \max_{y_i, y_j} \left[b_{ij}(y_i, y_j) + b_i(y_i) + b_j(y_j) \right]
\end{aligned}
$$

and with similar derivation, we also have $\max_{y_j} b_j^\star(y_j) = \frac{1}{2} \max_{y_i, y_j} \left[b_{ij}(y_i, y_j) + b_i(y_i) + b_j(y_j) \right]$. □

Then by Lemma 1 and Proposition 2, we have that

$$
\begin{aligned}
\max_{y_i} b_i^\star(y_i) + \max_{y_j} b_j^\star(y_j) + \max_{y_i, y_j} b_{ij}^\star(y_i, y_j) &= \max_{y_i, y_j} \left[b_i(y_i) + b_j(y_j) + b_{ij}(y_i, y_j) \right] \\
&\leqslant \max_{y_i} b_i(y_i) + \max_{y_j} b_j(y_j) + \max_{y_i, y_j} b_{ij}(y_i, y_j) \\
&= \max_{y_i} \left[\theta_i(y_i) + \sum_{j' \in N(i)} \lambda_{j' \to i}(y_i) - u_i - v_{y_i} \right] \\
&\quad + \max_{y_j} \left[\theta_j(y_j) + \sum_{j' \in N(j)} \lambda_{j' \to j}(y_j) - u_j - v_{y_j} \right] \\
&\quad + \max_{y_i, y_j} \left[\theta_{ij}(y_i, y_j) - \lambda_{j \to i}(y_i) - \lambda_{i \to j}(y_j) \right],
\end{aligned}
$$

which indicates that (12) is a closed-form solution of (10).