An Algorithm for Finding the Most Similar Given Sized Subgraphs in Two Weighted Graphs

Xu Yang, Hong Qiao, *Senior Member, IEEE*, and Zhi-Yong Liu, *Senior Member, IEEE*

***Abstract*— We propose a weighted common subgraph (WCS) matching algorithm to find the most similar subgraphs in two labeled weighted graphs. WCS matching, as a natural generalization of equal-sized graph matching and subgraph matching, has found wide applications in many computer vision and machine learning tasks. In this brief, WCS matching is first formulated as a combinatorial optimization problem over the set of partial permutation matrices. Then, it is approximately solved by a recently proposed combinatorial optimization framework—graduated nonconvexity and concavity procedure. Experimental comparisons on both synthetic graphs and real-world images validate its robustness against noise level, problem size, outlier number, and edge density.**

***Index Terms*— Adjacency matrix, graph algorithms, graph matching, weighted common subgraph (WCS) matching.**

I. Introduction

Graph matching aims to find the optimal correspondence between vertices of two graphs. It is a fundamental problem in theoretical computer science, and also plays a key role in many computer vision and pattern recognition tasks [1], [2].

Bipartite graph matching can be effectively and efficiently solved by the Hungarian algorithm [3] or linear programming methods [4]. When further considering pairwise constraints, the matching problem becomes in general NP-hard. Approximation methods, which make certain relaxations to the original problem, are necessary for efficiency reasons [1].

In the last ten years, significant progresses have been achieved on the approximation methods. For instance, the computational complexity has been decreased to as low as $\mathcal{O}(N^3)$—the complexity of matrix multiplication. On the other hand, the accuracy, taking the benchmark data set "House sequence"[1] for example, has been increased from about 60% [5] to nearly 100% [6], [7], due to the new graph similarity measures as well as new optimization techniques. Typical algorithms in the literature include graduated assignment (GA) [8], spectral technique (SM) [9], path following [7], and probabilistic graph matching (PGM) [10].

In this brief, we consider the matching problem involving outliers. Most existing methods treat it as a part-in-whole (PIW) problem [11], commonly known as *subgraph matching*, which recognizes the smaller graph as a part of the bigger one. Moreover, some recently proposed effective graph matching algorithms [7], [12] are only applicable to equal-sized graphs. However, in real applications, such as object recognition in computer vision, outliers may exist in both images because of image background, object occlusion, or geometric transformations. Thus, it is reasonable to formulate the matching problem as finding the most similar subgraphs within two graphs extracted from the images. Furthermore, to obtain a robust similarity measure, the number of matched vertices should sometimes be specified and kept smaller than the estimated number of inliers representing the objects [13]. Then, the problem can be defined as finding the most similar subgraphs of a given size in two labeled weighted graphs. Such a problem is named weighted common subgraph (WCS) matching, a generalization of the equal-sized graph matching and subgraph matching from the perspective of outlier distribution.

Another similar term in the literature is the maximum common subgraph (MCS) problem, or known as MCS isomorphism [14]. Given two graphs, MCS aims to find the largest subgraph in one graph isomorphic to an unknown subgraph in the other one. MCS has a long tradition in structural data processing, such as cheminformatics. MCS and WCS are different mainly in the following two aspects. First, MCS requires the two common subgraphs to be strictly isomorphic to each other, even on weighted graphs, while WCS tolerates some disparities between them. The latter one is more reasonable in most real tasks when involving noises. Second, MCS searches for the largest common subgraphs while WCS for the common subgraphs of a given size.

In the literature, there also exist some algorithms [9], [10], which can be used for WCS matching, by a two-step schema, which first matches all the vertices, and then finds a specified number of best assignments by ranking techniques. Such a two-step idea is, however, not consistent with the WCS matching problem. That is, even both the two steps are optimally solved, and the obtained subgraphs may not be the optimal pair.

Different from the above-mentioned methods, in this brief, we propose a novel WCS matching algorithm, which unifies the two steps and targets directly at the specified number of best assignments. Specifically, we first formulate WCS matching as a combinatorial problem over the partial permutation matrices, and then develop a graduated nonconvexity and concavity procedure (GNCCP)-based [15], [16] optimization algorithm. To this end, we also propose two relaxations of the objective function to make the calculation tractable.

The WCS matching algorithm is proposed in Section II, and experimentally evaluated in Section III on both synthetic graphs and real-world images. Finally, concluding remarks and future extensions are discussed in Section IV.

II. Method

In this section, we first formulate WCS matching as a combinatorial optimization problem, and then approximately solve it by the GNCCP. Finally, we give two relaxations to make the algorithm implementable.

Manuscript received June 7, 2016; revised February 23, 2017 and May 29, 2017; accepted June 4, 2017. This work was supported in part by the National Natural Science Foundation of China under Grant 61503383, Grant 61633009, Grant U1613213, Grant 61375005, Grant 61303174, and Grant 61210009, in part by the National Key Research and Development Plan of China under Grant 2016YFC0300801, and in part by the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant XDB02080003. *(Corresponding author: Zhi-Yong Liu.)*

X. Yang is with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: xu.yang@ia.ac.cn).

H. Qiao and Z.-Y. Liu are with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China; with the Center for Excellence in Brain Science and Intelligence Technology, Chinese Academy of Sciences, Shanghai 200031, China; and also with the University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: hong.qiao@ia.ac.cn; zhiyong.liu@ia.ac.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2017.2712794

[1]Available at http://vasc.ri.cmu.edu//idb/html/motion/house/index.html

A. Formulation

A graph $G = (V, E)$ of size M is defined by a finite vertex set $V = \{1, 2, \cdots, M\}$ and an edge set $E \subseteq V \times V$. The labeled weighted graph is further defined by assigning a real number vector l_i^G as a label to vertex i, and assigning a nonnegative real number w_{ij}^G as a weight to edge ij in G. Taking feature correspondence for example, by treating feature points as vertices, some local descriptor, e.g., the scale invariant feature transform (SIFT) descriptor, can be used as the vertex label, and the distance between two feature points as the edge weight. The weighted adjacency matrix $\mathbf{A}_G \in \mathbb{R}^{M\times M}$ is commonly used to record adjacency and weights of edges. Hereafter, by terms *graph* and *adjacency matrix*, we mean the labeled weighted graph and weighted adjacency matrix, respectively.

Given two graphs G and H of size M and N, respectively, and an integer $L \le M \le N$, WCS matching is formulated as the following combinatorial programming problem:

$$\min_{\mathbf{X}} \; F(\mathbf{X})$$
$$\text{s.t. } \mathbf{X} \in \mathcal{P}, \mathcal{P} := \left\{ \mathbf{X} \left| \sum_{i=1}^{M} \mathbf{X}_{ij} \le 1, \sum_{j=1}^{N} \mathbf{X}_{ij} \le 1, \right. \right.$$
$$\left. \sum_{i=1}^{M}\sum_{j=1}^{N} \mathbf{X}_{ij} = L, \mathbf{X}_{ij} \in \{0,1\} \right\}, \quad L \le M \le N \tag{1}$$

where $\mathcal{P}$ is the set of partial permutation matrices shown in Fig. 1. As a generalization of the recent adjacency matrix-based objective functions [7], [15], which have the advantage of low computational complexity and storage complexity. In this brief, the objective function $F(\mathbf{X})$ is constructed by

$$F(\mathbf{X}) = \alpha \|\mathbf{U} \circ \mathbf{A}_G - \mathbf{X}\mathbf{A}_H\mathbf{X}^T\|_F^2 + (1-\alpha)\text{tr}(\mathbf{C}^T\mathbf{X}). \tag{2}$$

For further derivation convenience, the higher order term is denoted by $H_0(\mathbf{X})$, that is

$$H_0(\mathbf{X}) = \|\mathbf{U} \circ \mathbf{A}_G - \mathbf{X}\mathbf{A}_H\mathbf{X}^T\|_F^2. \tag{3}$$

The two adjacency matrices $\mathbf{A}_G$ and $\mathbf{A}_H$ are, respectively, associated with the graphs G and H. $\|\cdot\|_F$ denotes the Frobenius matrix norm defined as $\|\mathbf{A}\|_F = (\sum_i \sum_j \mathbf{A}_{ij}^2)^{1/2} = (\text{tr}(\mathbf{A}^T\mathbf{A}))^{1/2}$ with $\text{tr}(\cdot)$ denoting the matrix trace. The symbol $\circ$ denotes the Hadamard product (entrywise product) of two matrices defined as $(\mathbf{A} \circ \mathbf{B})_{ij} = \mathbf{A}_{ij}\mathbf{B}_{ij}$ assuming conformability. The matrix $\mathbf{U} = \mathbf{X}\mathbf{1}_{N\times N}\mathbf{X}^T$ is to "pick out" the vertices with corresponding relations in G, where every entry in $\mathbf{1}_{N\times N}$ is "1," as shown in Fig. 2. In the unary term $\text{tr}(\mathbf{C}^T\mathbf{X})$, $\mathbf{C}$ is a cost matrix with $\mathbf{C}_{ij}$ measuring the dissimilarity between labels l_i^G and l_j^H. The parameter α satisfying $0 \le \alpha \le 1$ is used to balance the two terms.

When $L = M \le N$, WCS matching degenerates to the PIW problem [15], and when $L = M = N$, it further degenerates to the equal-sized matching problem [7], [12].

By defining the problem over $\mathcal{P}$, we actually introduce the one-to-one constraints on the WCS matching, a commonly used assumption in graph matching [6]. Particularly, if $\mathbf{X}_{ij} = 1$, vertex i in G is assigned to j in H. If $\sum_i^M \mathbf{X}_{ij} = 0$, there are no corresponding vertices in G for vertex j in H. It is similar when $\sum_j^N \mathbf{X}_{ij} = 0$, as shown in Fig. 1.

The formulation equation (1) directly targets at the WCS matching problem, without resorting to the conventional two-step schema [9], [10]. However, it meanwhile becomes more difficult to handle, as to be discussed later in Section II-C.

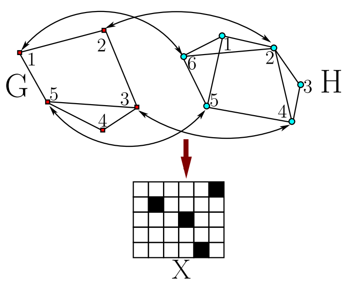

Fig. 1. Matching $L = 4$ vertices between the graphs G and H with size $M = 5$ and $N = 6$. The black box and white box in $\mathbf{X}$ mean 1 and 0, respectively.

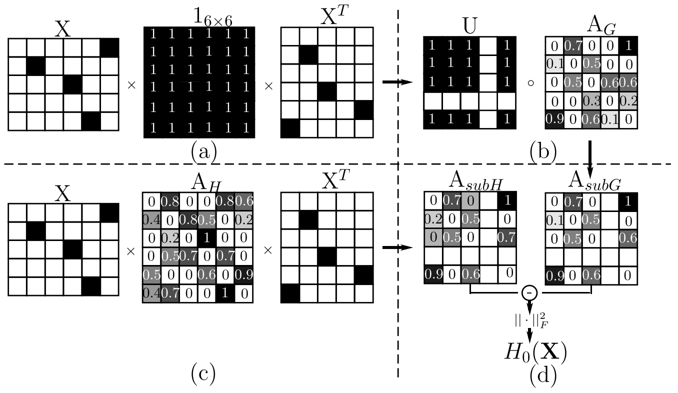

Fig. 2. Graphic description for the objective. Here, $M = 5$, $N = 6$, and $L = 4$. $\mathbf{A}_{\text{subG}}$ and $\mathbf{A}_{\text{subH}}$ denote the adjacency matrices for the subgraphs from G and H, respectively. (a) $\mathbf{U} = \mathbf{X}\mathbf{1}_{N\times N}\mathbf{X}^T$. (b) $\mathbf{A}_{\text{subG}} = \mathbf{U} \circ \mathbf{A}_G$. (c) $\mathbf{A}_{\text{subH}} = \mathbf{X}\mathbf{A}_H\mathbf{X}^T$. (d) $H_0(\mathbf{X}) = \|\mathbf{A}_{\text{subG}} - \mathbf{A}_{\text{subH}}\|_F^2$.

B. Optimization

The combinatorial optimization problem (1) is NP-hard with a factorial complexity, which calls for some approximations in realistic applications. In the following, we propose an approximation algorithm based on the GNCCP [15], [16], a relaxation technique.

The GNCCP has its root in the convex–concave relaxation procedures (CCRPs) [7], [12], [17]. Combining both convex and concave relaxations, CCRP achieved a superior performance on the equal-sized graph matching. The GNCCP realizes exactly a type of CCRP, but in a much simpler way without needing to construct the convex or concave relaxations explicitly, making it very easy to use in practice. This is particularly important for WCS matching, because both the convex and concave relaxations are difficult to construct.

To utilize the GNCCP to solve (1), first, we need to get the convex hull $\mathcal{D}$ of $\mathcal{P}$ as follows.

Theorem 1: The convex hull of the set of partial matrices $\mathcal{P}$ is $\mathcal{D}$, where

$$\mathcal{D} := \left\{ \mathbf{X} \left| \sum_{i=1}^{M} \mathbf{X}_{ij} \le 1, \sum_{j=1}^{N} \mathbf{X}_{ij} \le 1, \sum_{i=1}^{M}\sum_{j=1}^{N} \mathbf{X}_{ij} = L, \mathbf{X}_{ij} \ge 0 \right. \right\}.$$

Proof: See Appendix A in the Supplementary Material.

Note that $\mathcal{D}$ can be regarded as a generalization of the set of doubly stochastic matrices [18], the convex hull of the set of permutation matrices. Then, the GNCCP takes the following form:

$$J_\zeta(\mathbf{X}) = \begin{cases} (1-\zeta)F(\mathbf{X}) + \zeta\text{tr}(\mathbf{X}^T\mathbf{X}), & \text{if } 1 \ge \zeta \ge 0 \\ (1+\zeta)F(\mathbf{X}) + \zeta\text{tr}(\mathbf{X}^T\mathbf{X}), & \text{if } 0 > \zeta \ge -1 \end{cases}$$
$$\mathbf{X} \in \mathcal{D}. \tag{4}$$

In implementation, ζ decreases gradually from 1 to 0 (graduated nonconvexity) and finally to -1 (graduated concavity). During the process, GNCCP implicitly transfers from the convex relaxation to the concave relaxation. When reaching the concave relaxation whose minimum points locate exactly in $\mathcal{P}$, the algorithm terminates. The readers are referred to [15] for more details of GNCCP.

For a specific ζ, $J_\zeta(P)$ is optimized by the Frank–Wolfe algorithm [19], which iteratively updates $\mathbf{X}$ by $\mathbf{X} \leftarrow \mathbf{X} + \lambda d$ until convergence. The initial $\mathbf{X}$ is the solution of $J_\zeta(\mathbf{X})$ obtained at the previous ζ. And the optimal search direction $d = \mathbf{Y} - \mathbf{X}$ is given by solving the following linear programming problem:

$$\begin{aligned} \mathbf{Y} = \arg\max\ & \operatorname{tr}(-\nabla J_\zeta(\mathbf{X})^T \mathbf{Y}) \\ \text{s.t. } & \mathbf{Y} \in \mathcal{D} \end{aligned} \tag{5}$$

which can be solved by, for example, the interior point method [20]. The gradient $\nabla J_\zeta(\mathbf{X})$ in (5) takes the following form:

$$\begin{aligned} \nabla J_\zeta(\mathbf{X}) &= \begin{cases} (1-\zeta)\nabla F(\mathbf{X}) + 2\zeta\mathbf{X}, & \text{if } 1 \geq \zeta \geq 0 \\ (1+\zeta)\nabla F(\mathbf{X}) + 2\zeta\mathbf{X}, & \text{if } 0 > \zeta \geq -1 \end{cases} \\ \mathbf{X} &\in \mathcal{D} \end{aligned} \tag{6}$$

where

$$\nabla F(\mathbf{X}) = \nabla H_0(\mathbf{X}) + (1-\alpha)\mathbf{C}. \tag{7}$$

The optimal step size λ is given by

$$\begin{aligned} \lambda = \arg\min\ & J_\zeta(\mathbf{X} + \lambda(\mathbf{Y} - \mathbf{X})) \\ \text{s.t. } & 0 \leq \lambda \leq 1 \end{aligned} \tag{8}$$

which can be solved by an inexact line search, e.g., backtracking algorithm [20].

Finally, the GNCCP-based WCS matching algorithm is summarized by Algorithm 1.

Algorithm 1: Weighted Common Subgraph Matching Algorithm

```
Input: Two graphs $G$ and $H$
Initialization: $\mathbf{X} \leftarrow \mathbf{1}_{M\times N}\frac{L}{M\times N}, \zeta \leftarrow 1$
GNCCP:
    Repeat
        FW process
        $\zeta = \zeta - d\zeta$
    Until $\zeta < -1 \vee \mathbf{X} \in \mathcal{P}$
Output: The matching result $\mathbf{X}$
```

C. Implementation Details

In implementation, it is difficult to directly calculate $\nabla H_0(\mathbf{X})$, which involves a Hadamard product. Instead, in the following, we propose two types of relaxations of $H_0(\mathbf{X})$ to make it calculable. Similar relaxation techniques were widely used in graph matching [7], [12].

The first relaxation $H_1(\mathbf{X})$ is given as follows:

$$\begin{aligned} H_1(\mathbf{X}) &= \|\mathbf{U} \circ \mathbf{A}_G - \mathbf{X}\mathbf{A}_H\mathbf{X}^T\|_F^2 = \operatorname{tr}((\mathbf{A}_G \circ \mathbf{A}_G)U^T) \\ &\quad -2\operatorname{tr}(\mathbf{A}_G\mathbf{X}\mathbf{A}_H^T\mathbf{X}^T) + \operatorname{tr}(\mathbf{X}\mathbf{A}_H\mathbf{X}^T\mathbf{X}\mathbf{A}_H^T\mathbf{X}^T) \end{aligned} \tag{9}$$

where we take advantage of

$$\mathbf{U} \circ \mathbf{U} = \mathbf{U} \tag{10a}$$

$$\mathbf{U} \circ (\mathbf{X}\mathbf{A}_H\mathbf{X}^T) = \mathbf{X}\mathbf{A}_H\mathbf{X}^T. \tag{10b}$$

See Appendix B in the Supplementary Material for the derivation details of (9). Its gradient is then figured out as follows:

$$\begin{aligned} \nabla H_1(\mathbf{X}) &= (\mathbf{A}_G^T \circ \mathbf{A}_G^T + \mathbf{A}_G \circ \mathbf{A}_G)\mathbf{X}\mathbf{1}_{N\times N} - 2(\mathbf{A}_G^T\mathbf{X}\mathbf{A}_H \\ &\quad + \mathbf{A}_G\mathbf{X}\mathbf{A}_H^T) + 2(\mathbf{X}\mathbf{A}_H\mathbf{X}^T\mathbf{X}\mathbf{A}_H^T + \mathbf{X}\mathbf{A}_H^T\mathbf{X}^T\mathbf{X}\mathbf{A}_H). \end{aligned} \tag{11}$$

The second relaxation $H_2(\mathbf{X})$ is derived as follows:

$$\begin{aligned} H_2(\mathbf{X}) &= \|\mathbf{U} \circ \mathbf{A}_G - \mathbf{X}\mathbf{A}_H\mathbf{X}^T\|_F^2 = \|(\mathbf{X}\mathbf{X}^T)\mathbf{A}_G(\mathbf{X}\mathbf{X}^T) - \mathbf{X}\mathbf{A}_H\mathbf{X}^T\|_F^2 \\ &= \operatorname{tr}(\mathbf{X}\mathbf{X}^T\mathbf{A}_G^T\mathbf{X}\mathbf{X}^T\mathbf{A}_G\mathbf{X}\mathbf{X}^T) - 2\operatorname{tr}(\mathbf{X}\mathbf{X}^T\mathbf{A}_G^T\mathbf{X}\mathbf{A}_H\mathbf{X}^T) \\ &\quad + \operatorname{tr}(\mathbf{X}\mathbf{A}_H^T\mathbf{X}^T\mathbf{X}\mathbf{A}_H\mathbf{X}^T) = T_1(\mathbf{X}) - 2T_2(\mathbf{X}) + T_3(\mathbf{X}) \end{aligned} \tag{12}$$

where we take advantage of

$$U \circ A = \mathbf{X}\mathbf{X}^T A \mathbf{X}\mathbf{X}^T \tag{13a}$$

$$\mathbf{X}\mathbf{X}^T\mathbf{X}\mathbf{X}^T = \mathbf{X}\mathbf{X}^T \tag{13b}$$

$$\mathbf{X}\mathbf{X}^T\mathbf{X} = \mathbf{X} \tag{13c}$$

$$\mathbf{X}^T\mathbf{X}\mathbf{X}^T = \mathbf{X}^T. \tag{13d}$$

Then, the gradient is given as follows:

$$\nabla H_2(\mathbf{X}) = \nabla T_1(\mathbf{X}) - 2\nabla T_2(\mathbf{X}) + \nabla T_3(\mathbf{X}) \tag{14}$$

where

$$\begin{aligned} \nabla T_1(\mathbf{X}) &= 2(\mathbf{X}\mathbf{X}^T\mathbf{A}_G^T\mathbf{X}\mathbf{X}^T\mathbf{A}_G\mathbf{X} + \mathbf{A}_G^T\mathbf{X}\mathbf{X}^T\mathbf{A}_G\mathbf{X}\mathbf{X}^T\mathbf{X} \\ &\quad + \mathbf{A}_G\mathbf{X}\mathbf{X}^T\mathbf{X}\mathbf{X}^T\mathbf{A}_G^T\mathbf{X}) \\ \nabla T_2(\mathbf{X}) &= \mathbf{X}\mathbf{A}_H^T\mathbf{X}^T\mathbf{A}_G\mathbf{X} + \mathbf{A}_G^T\mathbf{X}\mathbf{A}_H\mathbf{X}^T\mathbf{X} + \mathbf{A}_G\mathbf{X}\mathbf{X}^T\mathbf{X}\mathbf{A}_H^T \\ &\quad + \mathbf{X}\mathbf{X}^T\mathbf{A}_G^T\mathbf{X}\mathbf{A}_H \\ \nabla T_3(\mathbf{X}) &= 2(\mathbf{X}\mathbf{A}_H^T\mathbf{X}^T\mathbf{X}\mathbf{A}_H + \mathbf{X}\mathbf{A}_H\mathbf{X}^T\mathbf{X}\mathbf{A}_H^T). \end{aligned}$$

Consequently, by replacing $H_0(\mathbf{X})$ with $H_1(\mathbf{X})$ or $H_2(\mathbf{X})$, the GNCCP can be implemented to solve the WCS matching problem. It is noted that both $H_1(\mathbf{X})$ and $H_2(\mathbf{X})$ are relaxations of $H_0(\mathbf{X})$, because $H_0(\mathbf{X}) = H_1(\mathbf{X}) = H_2(\mathbf{X})$, $\forall \mathbf{X} \in \mathcal{P}$. However, the equivalence becomes in general unsatisfied when $\mathbf{X} \in \mathcal{D}\backslash\mathcal{P}$. It is similar to previous works [7], [15] where the relaxation functions are not necessarily equivalent to the original objective function in the continuous domain.

It is hard to evaluate the two relaxations theoretically by error bounds, because neither of them is convex relaxation. However, as revealed by the experimental comparisons in Section III, $H_1(\mathbf{X})$ outperforms $H_2(\mathbf{X})$ in most of the experiments. The advantage of $H_1(\mathbf{X})$ may be due to the fact that it has the same fourth order as $H_0(\mathbf{X})$, which is lower than the sixth order of $H_2(\mathbf{X})$. Furthermore, $H_1(\mathbf{X})$ is computationally more efficient than $H_2(\mathbf{X})$.

Last but not least, when degenerating to the PIW problem, i.e., $L = M$, the GNCCP can be directly implemented, with $H_0(\mathbf{X})$ and its gradient $\nabla H_0(\mathbf{X})$ given as follows [15]:

$$H_0'(\mathbf{X}) = \|\mathbf{A}_G - \mathbf{X}\mathbf{A}_H\mathbf{X}^T\|_F^2 \tag{15}$$

$$\begin{aligned} \nabla H_0'(\mathbf{X}) &= 2\mathbf{X}(\mathbf{A}_H^T\mathbf{X}^T\mathbf{X}\mathbf{A}_H + \mathbf{A}_H\mathbf{X}^T\mathbf{X}\mathbf{A}_H^T) \\ &\quad - 2(\mathbf{A}_G\mathbf{X}\mathbf{A}_H^T + \mathbf{A}_G^T\mathbf{X}\mathbf{A}_H). \end{aligned} \tag{16}$$

D. Storage and Computational Complexity

The proposed method formulates the graph matching problem based on the adjacency matrix. Compared with the *affinity matrix*[2]-based algorithms [6], [8]–[10], [21], one most important advantage of the adjacency matrix-based methods is storage saving. Without

[2]The affinity matrix can be seen as the adjacency matrix for the associate graph of G and H, whose size is $MN \times MN$.

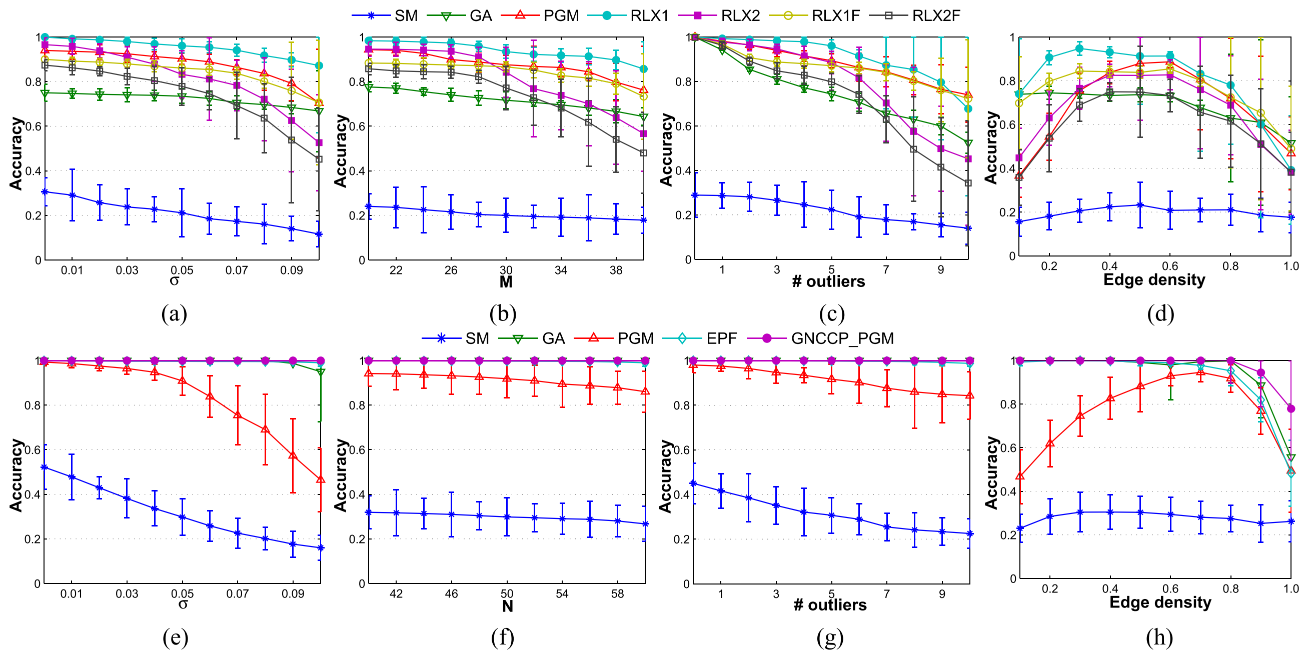

Fig. 3. Results on synthetic data with respect to noise level, problem size, outlier number, and edge density, summarized from 30 random runs for each fixed configuration. The WCS and PIW results are in the top and bottom rows, respectively. Matching accuracy comparison results with respect to (a) noise level in WCS, (b) problem size in WCS, (c) outlier number in WCS, (d) edge density in WCS, (e) noise level in PIW, (f) problem size in PIW, (g) outlier number in PIW, and (h) edge density in PIW.

considering the sparsity, the storage complexity of the affinity matrix-based methods is $\mathcal{O}(M^2N^2)$, while that of adjacency matrix-based methods, including the proposed method, is as low as $\mathcal{O}(N^2)$.

The computational complexity of the proposed method is mainly determined by the linear programming problem (5), which can be solved in polynomial time. When $L = M \leq N$, (5) can be solved by the rectangular Hungarian algorithm [22] with an $\mathcal{O}(M^2N)$ computational complexity. It is smaller than the matrix multiplication complexity $\mathcal{O}(MN^2)$, so the overall complexity is $\mathcal{O}(MN^2)$. When $L < M \leq N$, the Hungarian algorithm or other efficient linear assignment algorithms [4] are inapplicable. Linear programming algorithms involving an $\mathcal{O}(N^6)$ computational complexity, such as the interior point method, are usually used to solve (5).

To make the algorithm more efficient, a fast method based on the rectangular Hungarian algorithm [22] is presented to approximately solve (5), which still enjoys the complexity $\mathcal{O}(MN^2)$. Specifically, it first finds a solution $\mathbf{Y}_1$ for (5) with M "1"s by the rectangular Hungarian algorithm [22]. Then, it removes the $M - L$ "1"s values of $\mathbf{Y}_1$ that correspond to the $M - L$ smallest values of $-\nabla J_{\zeta}(\mathbf{X})$ to get the final approximate solution $\mathbf{Y}$. The cost for the computational efficiency is a slight loss of accuracy, as to be experimentally demonstrated in Section III.

III. Experimental Results

We apply the proposed algorithm on synthetic graphs as well as real-world images,[3] to evaluate its performance against noise level, problem size, outlier number, and edge density. The experiments are conducted on both WCS matching and PIW problems. The methods included for comparison are SM [9], GA [8], PGM [10], and *(extended) path following method* (EPF) [7], [12]. The proposed algorithm with two relaxations is denoted by RLX1 and RLX2, respectively. When the fast method described in Section II-D is used to solve the linear programming (5), the two algorithms are then, respectively, denoted by RLX1F and RLX2F. When used on the PIW problem, the proposed algorithm becomes exactly the one proposed in [15], denoted by GNCCP_PGM.

The algorithms are implemented by MATLAB R2011 on a personal computer with a 3.07-GHz CPU (two core) and 2-GB RAM, using mex (dll) files *lpsolve* toolbox[4] and the *rectangular assignment* toolbox[5] for the linear programming problem (5).

A. On Synthetic Data

1) Experimental Settings: In this experiment, the accuracies of different algorithms are first compared on randomly generated synthetic graphs. First, two spatial point sets $\mathcal{G} = \{g_i\}_{i=1}^{M}$, $\mathcal{H} = \{h_i\}_{j=1}^{N}$ are randomly generated by uniform sampling, i.e., $g_i,\ h_j \sim U(0,1)^{1\times 2}$. A partial permutation matrix $\mathbf{X}^{gt} \in \mathbb{R}^{M\times N}$ with L "1"s is randomly generated as the ground truth correspondence. Then, part of points in $\mathcal{G}$ are updated by permutating $\mathcal{H}$ with $\mathbf{X}^{gt}$ as

$$g_i = h_j + \eta, \quad \eta \sim N(0,\sigma^2), \quad \text{if } \mathbf{X}_{ij}^{gt} = 1$$

where η is the additive gaussian noise. Finally, the distances between points are utilized as the edge weights. And the adjacency, i.e., the graph structure, is built in a sparse manner by adjusting the edge density. The graph structure is disturbed by the noise η following the way in [7]. Specifically, $(1/2)\sigma\#Edge$ edges are randomly added to and removed from each sparse graph, where $\#Edge$ denotes the number of edges. The adjacency matrices $\mathbf{A}_G$, $\mathbf{A}_H$, and the affinity matrix are then generated, where the affinity matrix, required by SM, GA, and PGM, is built in the same way as in [10]. The parameter α is set as 1.

For WCS matching, the following four scenarios are implemented.

1) *Noise Level:* Set $M = 30$, $N = M + 5$, $L = M - 5$, set edge density as 0.5, and increase σ from 0 to 0.1 by a step 0.01.

[3]More experimental results are given in the supplementary materials (Sup_Mat_II_Add_Exp.pdf), including an experiment on handwritten Chinese character recognition and typical results on *Motorbike* and *Pisa* images.

[4]Available at http://sourceforge.net/projects/lpsolve/files/lpsolve/

[5]Available http://www.mathworks.com/matlabcentral/fileexchange/6543

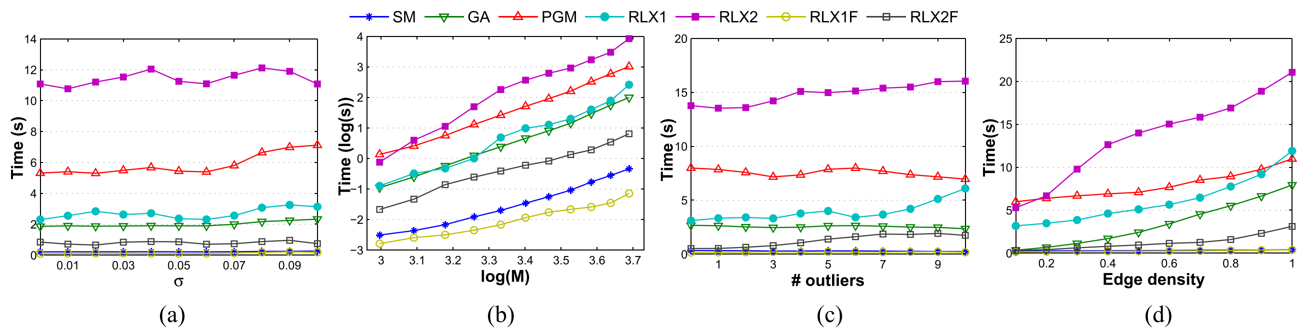

Fig. 4. Running time comparison in WCS matching with respect to noise level, problem size, outlier number, and edge density, summarized from 50 random runs for each fixed configuration. Running time comparison results with respect to (a) noise level, (b) problem size, (c) outlier number, and (d) edge density.

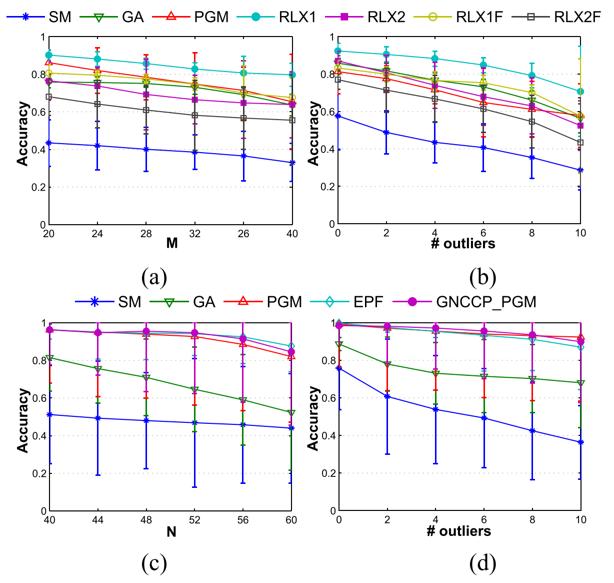

Fig. 5. Results on real-world images with respect to problem size and outlier number, summarized from 200 random runs for each fixed configuration. The WCS and PIW results are in the top and bottom rows, respectively. Matching accuracy comparison results with respect to (a) problem size in WCS, (b) outlier number in WCS, (c) problem size in PIW, and (d) outlier number in PIW.

2) *Problem Size:* Set $\sigma = 0.05$, $N = M + 5$, $L = M - 5$, set edge density as 0.5, and increase M from 20 to 40 by a step 2.
3) *Outlier Number:* Set $\sigma = 0.05$, $M = 30$, $N = M + 5$, set edge density as 0.5, and decrease L from 30 to 20 by a step 1.
4) *Edge Density:* Set $\sigma = 0.05$, $M = 30$, $N = M + 5$, $L = M - 5$, and increase the density from 0.1 to 1 by a step 0.1.

For the PIW problem, the following four similar scenarios are implemented.

1) *Noise Level:* Set $N = 50$, $L = M = N - 5$, set edge density as 0.5, and increasing σ from 0 to 0.1 by a step 0.01.
2) *Problem Size:* Set $\sigma = 0.05$, $L = M = N - 5$, set edge density as 0.5, and increase N from 40 to 60 by a step 2.
3) *Outlier Number:* Set $\sigma = 0.05$, $N = 50$, $L = M$, set edge density as 0.5, and decrease L from 50 to 40 by a step 1.
4) *Edge Density:* Set $\sigma = 0.05$, $N = 50$, $L = M = N - 5$, and increase the density from 0.1 to 1 by a step 0.1.

2) Results: The WCS matching performance is shown in Fig. 3, from which we can draw the following observations. First, generally the proposed algorithm RLX1 achieves better results than other algorithms, and the performance of RLX2 is comparable with PGM, a state-of-the-art algorithm in the literature. Second, RLX1 outperforms RLX2, probably because $H_1(\mathbf{X})$ provides a better relaxation for $H_0(\mathbf{X})$. Third, the fast algorithm for approximating (5) introduced in Section II-D shows good performance. For instance, RLX1F achieves a comparable performance to PGM. It is also reasonable to observe that RLX1 and RLX2 outperform slightly RLX1F and RLX2F, respectively, but the latter ones enjoy a much smaller computational complexity, as to be discussed in the following. Finally, it is also reasonable to witness that the accuracies of all the algorithms decrease as the noise level, problem size, or outlier number increase. Since the number of local minimum points is $C_M^L C_N^L L!$, a larger M, N, or L implies that the algorithm is easier to be trapped in a poor local minimum. Besides, the algorithms achieve their best results when the edge density is set about 0.5, at which the objective function seems to get a good balance between the appearance and structure cues. When the edge density is 0, the matching problem degenerates to a pure appearance matching without structural cues. As the edge density increases, incorporation of more structural information results in a better performance. However, high edge density may make the graphs less distinctive, and thus result in a decrease in performance.

The computation time of different algorithms in WCS matching is compared in Fig. 4. It can be observed that generally the computation time with respect to the varying noise level and outlier number is relatively stable, but they are positively correlated with problem size and edge density. The computation time with respect to problem size is plotted in log scale in Fig. 4(b), and the slopes of the lines indicate the computational complexities of different algorithms, which are, respectively, 5.4 ± 0.5 for RLX1 and RLX2, 4.3 ± 0.5 for SM, GA, and PGM, and 3.5 ± 0.5 for RLX1F and RLX2F.

The PIW results are also given in Fig. 3. All the algorithms achieve better performances on the PIW problem than on WCS matching, implying that WCS matching is a more difficult problem. EPF transforms the subgraph matching problem into the equal-sized adjacency matrix matching by adding dummy nodes, which may, however, change the original problem [15]. By contrast, GNCCP_PGM directly optimizes the subgraph matching objective, and thus achieves better results.

B. On Real-World Images

We also apply the proposed method on a data set fetched from Pascal 2007 Challenge. The data set consists of ten pairs of *Motorbike* images and ten pairs of *Pisa* images. Each image pair is manually labeled with 60 ground-truth correspondence points. The graph structure is constructed by Delaunay triangulation. SIFT descriptor is utilized as the vertex label with $\alpha = 0.5$. The comparisons with respect to different problem sizes and outlier numbers are carried out, with the same experimental settings as those on the previous synthetic graph matching. The smaller number of ground truth points are randomly selected from the original ones. For instance, when the problem size is 35 and the outlier number is 5, only 30 ground-truth correspondence points are randomly selected from the 60 ground truth ones. For each image pair, the outliers are randomly sampled for ten times. Thus for each fixed configuration, the matching are repeated randomly for 200 times.

The real-image matching results are shown in Fig. 5, which reveals that the proposed algorithms achieve better or at least comparable performances with the state-of-the-art algorithms on both WCS and PIW.

IV. Conclusion and Future Works

A new algorithm is proposed to match subgraphs with given size in two labeled weighted graphs. Different from the commonly used two-step strategy, the proposed algorithm could directly find the most similar subgraphs in one optimization framework. In the current version, the common subgraph size must be prespecified, but in some tasks, it needs automatic determination. We would try to solve the problem in our future works.

References

[1] D. Conte, P. Foggia, C. Sansone, and M. Vento, "Thirty years of graph matching in pattern recognition," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 18, no. 3, pp. 265–298, May 2004.

[2] M. Houbraken, S. Demeyer, T. Michoel, P. Audenaert, D. Colle, and M. Pickavet, "The index-based subgraph matching algorithm with general symmetries (ISMAGS): Exploiting symmetry for faster subgraph enumeration," *PLoS ONE*, vol. 9, no. 5, p. e97896, 2014.

[3] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Res. Logistics Quart.*, vol. 2, nos. 1–2, pp. 83–97, Mar. 1955.

[4] A. Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency*. Berlin, Germany: Springer, 2003.

[5] H. J. Qiu and E. R. Hancock, "Graph matching and clustering using spectral partitions," *Pattern Recognit.*, vol. 39, no. 1, pp. 22–34, Jan. 2006.

[6] F. Zhou and F. De la Torre, "Factorized graph matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2012, pp. 127–134.

[7] M. Zaslavskiy, F. Bach, and J. P. Vert, "A path following algorithm for the graph matching problem," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 12, pp. 2227–2242, Dec. 2009.

[8] S. Gold and A. Rangarajan, "A graduated assignment algorithm for graph matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 4, pp. 377–388, Apr. 1996.

[9] M. Leordeanu and M. Hebert, "A spectral technique for correspondence problems using pairwise constraints," in *Proc. Int. Conf. Comput. Vis.*, Beijing, China, Oct. 2005, pp. 1482–1489.

[10] A. Egozi, Y. Keller, and H. Guterman, "A probabilistic approach to spectral graph matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 18–27, Jan. 2013.

[11] S. Biasotti, S. Marini, M. Spagnuolo, and B. Falcidieno, "Sub-part correspondence by structural descriptors of 3D shapes," *Comput. Aided Design*, vol. 38, no. 9, pp. 1002–1019, Sep. 2006.

[12] Z.-Y. Liu, H. Qiao, and L. Xu, "An extended path following algorithm for graph-matching problem," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1451–1456, Jul. 2012.

[13] J. Maciel and J. Costeira, "Robust point correspondence by concave minimization," *Image Vis. Comput.*, vol. 20, nos. 9–10, pp. 683–690, Aug. 2002.

[14] H.-C. Ehrlich and M. Rarey, "Maximum common subgraph isomorphism algorithms and their applications in molecular science: A review," *Wiley Interdiscipl. Rev., Comput. Mol. Sci.*, vol. 1, no. 1, pp. 68–79, Jan./Feb. 2011.

[15] Z.-Y. Liu and H. Qiao, "GNCCP—Graduated NonConvexityand concavity procedure," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 6, pp. 1258–1267, Jun. 2014.

[16] X. Yang, H. Qiao, and Z.-Y. Liu, "Partial correspondence based on subgraph matching," *Neurocomputing*, vol. 122, pp. 193–197, Dec. 2013.

[17] Z.-Y. Liu, H. Qiao, X. Yang, and S. C. H. Hoi, "Graph matching by simplified convex-concave relaxation procedure," *Int. J. Comput. Vis.*, vol. 109, no. 3, pp. 169–186, Sep. 2014.

[18] J. M. Borwein and A. S. Lewis, *Convex Analysis and Nonlinear Optimization*, vol. 3. Berlin, Germany: Springer, 2006.

[19] M. Frank and P. Wolfe, "An algorithm for quadratic programming," *Nav. Res. Logistics Q.*, vol. 3, nos. 1–2, pp. 95–110, Mar./Jun. 1956.

[20] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge, U.K.: Cambridge Univ. Press, 2004.

[21] T. Cour, P. Srinivasan, and J. Shi, "Balanced graph matching," in *Proc. Adv. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, Dec. 2006, p. 6.

[22] F. Bourgeois and J.-C. Lassalle, "An extension of the munkres algorithm for the assignment problem to rectangular matrices," *Commun. ACM*, vol. 14, no. 12, pp. 802–804, Dec. 1971.