Generalizing graph matching beyond quadratic assignment model

Tianshu Yu  
Arizona State University  
tianshuy@asu.edu

Junchi Yan  
Shanghai Jiao Tong University  
yanjunchi@sjtu.edu.cn

Yilin Wang  
Arizona State University  
yilwang@adobe.com

Wei Liu  
Tencent AI Lab  
wl2223@columbia.edu

Baoxin Li  
Arizona State University  
baoxin.li@asu.edu

Abstract

Graph matching has received persistent attention over several decades, which can be formulated as a quadratic assignment problem (QAP). We show that a large family of functions, which we define as Separable Functions, can approximate discrete graph matching in the continuous domain asymptotically by varying the approximation controlling parameters. We also study the properties of global optimality and devise convex/concave-preserving extensions to the widely used Lawler’s QAP form. Our theoretical findings show the potential for deriving new algorithms and techniques for graph matching. We deliver solvers based on two specific instances of Separable Functions, and the state-of-the-art performance of our method is verified on popular benchmarks.

1 Introduction

Given two graphs, graph matching algorithms (GM) seek to find node-to-node correspondences by optimizing a pre-defined affinity score function. This problem falls into the category of quadratic assignment problem (QAP) \([1]\), and has wide applications from object categorization \([2]\) to protein alignment \([3]\). While a line of works using combinatorial heuristics \([4, 5]\) attempt to solve graph matching, relaxation of original problem into the continuous domain is mostly employed and solved with different optimization techniques e.g. gradient \([6]\) or multiplication \([7, 8]\) based methods. The dominance of continuous relaxation may be partly because it is easier to analyze the local behavior of continuous functions, and one can often find a local optimum. In this paper, we focus on continuous relaxation of graph matching.

Graph matching seeks the solution to the quadratic assignment problem \(\max_X \text{vec}(X)^\top A \text{vec}(X)\), where \(\text{vec}(X) \in \{0, 1\}^{n^2}\) is the column-wise vectorized version of the binary (partial) assignment matrix \(X \in \{0, 1\}^{n \times n}\) and the so-called affinity matrix \(A \in \mathbb{R}^{n^2 \times n^2}\) in the real domain consists of the affinity score measuring how one edge in one graph is similar to another from the other graph. Traditionally, the common practice is relaxing \(\text{vec}(X)\) into the continuous real domain \(\text{vec}(X) \in \mathbb{R}^{n^2}\) \([9, 7, 10]\).

In this paper, we show that a large family of functions, defined as Separable Functions, can asymptotically approximate the discrete matching problem by varying the approximation controlling parameters. With this function family, there exist infinite modelings of graph matching problem, thereby providing the feasibility of adapting different practical problems with different models. This provides a new perspective of considering graph matching. We also give analysis on the conditions based on which
these approximations have good properties. Novel solvers on instances of Separable Functions are proposed based on the path-following and multiplicative techniques respectively.

Notations We use bold lower-case $x$ and upper-case $A$ to represent vector and matrix, respectively. Function $\text{vec}(\cdot)$ transforms a matrix to its column-wise vectorized replica. Conversely, function $\text{mat}(\cdot)$ transfers a vector back to its matrix form. Denote $\mathcal{R}_+, S$ as non-negative real numbers and symmetric matrices respectively. Function $K = \text{diag}(k)$ transforms a vector $k$ into a diagonal matrix $K$ such that $K_{ij} = k_i$ if $i = j$, and $K_{ij} = 0$ otherwise.

2 Related Work

Different from linear assignment [11], the quadratic assignment problem (QAP) in terms of graph matching in the literature is often formulated in two forms: i) the Koopmans-Beckmann’s QAP [12]: $\text{tr}(X^\top A_i X A_j) + \text{tr}(A_p^\top X)$ where $X$ is the assignment matrix, $A_i$ and $A_j$ are the weighted adjacency matrices, and $A_p$ is the node-to-node similarity matrix. Methods based on this formula include [13] [14] [15], to name a few; ii) the more general Lawler’s QAP [16] by $\text{vec}(X)^\top A \text{vec}(X)$. Note that the Koopmans-Beckmann’s QAP can always be represented as a special case of the Lawler’s by setting $A = A_j \otimes A_i$, and many previous works [9] [10] [6] [17] adopt the Lawler’s form, which is also the main focus of this paper for its generality. A recent survey [18] provides a more comprehensive literature review.

Though there are a few (quasi-)discrete methods [5] [19] [20] that directly work in the binary domain, the major line of research falls into the following tracks in the continuous domain. Our relaxation techniques do not fall into any of these categories and opens up new possibility for new algorithms.

Spectral relaxation: The authors of the seminal work in [7] proposed to relax $X$ to be of unit length $\|\text{vec}(X)\|^2 = 1$, and the resulting optimization problem can be efficiently solved by computing the leading eigen-vector of the affinity matrix $A$. Better approximation has been made in [21] by adding an affine constraint. In contrast to the above Lawler’s QAP based models, there are a few earlier methods [13] [22] based on the Koopmans-Beckmann’s QAP, and the relaxation is often fulfilled by setting $X^\top X = I$, where $I$ is the identify matrix. In general, spectral relaxation is efficient while not tight, which hinders the matching accuracy.

Semi-definite programming relaxation: SDP has been a standard tool for combinatorial problem, and it has been adopted to tackle the graph matching problem. In existing work, a variable $Y$ subject to $Y = \text{vec}(X)\text{vec}(X)^\top$ is introduced. As a result, the raw problem is approximated by SDP which relaxes the non-convex constraint on $Y$ into a semi-definite one: $Y \succeq \text{vec}(X)\text{vec}(X)^\top$. The final matching $X$ is then recovered by different heuristics such as winner-take-all [23] or randomization [24]. However, the SDP solver is not very popular in graph matching, as the variable $Y$ squares the raw variable size, resulting in high complexity.

Doubly-stochastic relaxation: Based on the fact that doubly-stochastic matrix is the convex hull of the permutation matrix, various methods have been proposed in this line to formulate the relaxed problem into a non-convex quadratic programming problem for both Koopmans-Beckmann’s and Lawler’s QAP. Linear programming is adopted in [25] to approximate the quadratic problem, followed by more complex path following methods [14] [15] to approximate the relaxed quadratic problem – all are based on the Koopmans-Beckmann’s QAP. For the more general Lawler’s QAP, the seminal work termed graduated assignment [9] approximates the relaxed QAP via solving a series of linear approximations via iterative Taylor expansions. A random walk perspective to the graph matching problem is adopted in [10], whereby the method can also be seen as a weighting solution by [9] and the multiplication method [7]. More recently factorized graph matching is devised in [17], which also follows the doubly-stochastic relaxation on top of other relaxations on the objective function.

Finally we also briefly review recent advances on hyper and multiple graph matching. There are studies addressing the more general hypergraph matching problem, whereby the third-order or higher is considered in the objective and usually an affinity tensor is adopted. Many current hypergraph matching methods approximate the third-order objective via iterative approximations. In each iteration, often a Lawler’s QAP is involved [26] [27]. Moreover, the Lawler’s QAP model solvers can also be used in matching a batch of graphs beyond two graphs. In [28], an alternating optimization method was proposed, whereby in each iteration a Lawler’s QAP problem is derived and solved. [29] further extends multi-graph matching problem to an online version. These connections further
highlight the importance of the Lawler’s QAP for not only traditional graph matching, but also
hypergraph matching and multiple graph matching.

3 Generalizing the QAP for Graph Matching

We re-visit the graph matching problem in this section. We propose an equivalent model to the
discrete one over continuous domain \([0, 1]\), provided the relaxation gap is 0. This gives rise to the
possibility to relax graph matching with much tighter ways. Mathematically, graph matching can be
formulated as the following quadratic assignment problem which is also called Lawler’s QAP:

\[
\max_{\mathbf{X}} \mathbf{vec}(\mathbf{X})^T \mathbf{A} \mathbf{vec}(\mathbf{X}) \\
\text{s.t. } \mathbf{X} \mathbf{1} = \mathbf{1}, \mathbf{X}^T \mathbf{1} = \mathbf{1}, x_{ia} \in \{0, 1\}
\]

where \(\mathbf{A} \in \mathbb{R}^{n^2 \times n^2}\) is a non-negative affinity matrix, which encodes node similarities on
diagonal elements and edge similarities on the rest. Note \(x_{ia}\) denotes the element of \(\mathbf{X}\) indexed by row \(i\) and
column \(a\) indicating the matching status of node \(i\) to node \(a\) from the other graph. If we break down
problem (1) into element-wise product, it becomes:

\[
\max_{\mathbf{x}} \sum_{i,j,a,b} \mathbf{A}_{ij,ab} x_{ia} x_{jb} \\
\text{s.t. } \mathbf{H} \mathbf{x} = \mathbf{1}, \mathbf{x} \in \{0, 1\}^{n^2}
\]

where \(\mathbf{A}_{ij,ab}\) corresponds to the edge similarity between edge \((i, j) \in \mathcal{G}_1\) and \((a, b) \in \mathcal{G}_2\). Here
\(\mathbf{H} \in \{0, 1\}^{2n \times n^2}\) is a selection matrix over the elements of \(\mathbf{x}\) sufficing assignment constraints
according to (1).

In particular, we relax \(\mathbf{x}\) into continuous domain and let \(f_{\text{prod}}(x_{ia}, x_{jb}) = x_{ia} x_{jb}\):

\[
\max_{\mathbf{x}} \sum_{i,j,a,b} \mathbf{A}_{ij,ab} f_{\text{prod}}(x_{ia}, x_{jb}) \\
\text{s.t. } \mathbf{H} \mathbf{x} = \mathbf{1}, \mathbf{x} \in [0, 1]^{n^2}
\]

We generalize problem (3) by replacing \(f_{\text{prod}}\) with \(f_{\delta}\):

\[
\max_{\mathbf{x}} \sum_{i,j,a,b} \mathbf{A}_{ij,ab} f_{\delta}(x_{ia}, x_{jb}) \\
\text{s.t. } \mathbf{H} \mathbf{x} = \mathbf{1}, \mathbf{x} \in [0, 1]^{n^2}
\]

where \(f_{\delta}\) is a 2D quasi-delta function in the continuous domain \((f_{\delta}(x, y) = 1\ if\ x = 1\ and\ y = 1,\ and\ f_{\delta} = 0\ otherwise)\). We have the following theorem that establishes the connection between (2)
and (4):

**Theorem 1** The optimal objective \(p^\ast\) to problem (2) is equal to the optimal objective \(p^\ast_{\delta}\) to problem (4).

**Remark** Based on Theorem 1 one can devise a sampling procedure to find the optimal solution to
problem (2) from the solution to problem (4): Given optimal \(x^\ast_{\delta}\) to problem (4), if all the elements
are in the set \{0, 1\}, then \(x^\ast_{\delta}\) is automatically optimal to problem (2). If not, we first remove all 1
elements and corresponding columns and rows, yielding a subvector (submatrix) \(\mathbf{x}^1\) with all elements
in range \([0, 1)\). Then any sampling subject to one-to-one constraint on \(\mathbf{x}^1\), together with the removed
discrete values, forms the optimal solution to problem (2).

For the time being, a discrete assignment problem (2) is relaxed into (4) with continuous feasible
domain. However, function \(f_{\delta}\) is not continuous as there is a jump at value \((1, 1)\), ending up with
much difficulty to solve (recall (4) is equivalent to (2)). In the next section, we will show some
approximate techniques to tackle problem (4).

---

1Here the number of nodes in two graphs are assumed the same. In case \(m \neq n\) one can add dummy nodes
as a standard technique as in literature [10][17].
Figure 1: Three examples of approximations (Laplacian, Gaussian, Polynomial) to function $f_\delta$ with varying $\theta$. The closer for $\theta \to 0$ (from red to green), the better approximation to $f_\delta$.

4 Separable Functions

4.1 Separable Approximation Function Family

It is important to find an appropriate approximate function for $f_\delta$, otherwise it may lead to intractable models to solve. To avoid high computational cost, we narrow our focus on a specific family of functions, called Separable Functions.

Definition 1 A function $f_\theta(x, y)$ is called Separable Function (SF)\(^2\) if it satisfies the following properties:

1. $f_\theta(x, y) = h_\theta(x) \times h_\theta(y)$ where $h_\theta$ is defined on $[0, 1]$.
2. $h_\theta(0) = 0$ and $h_\theta(1) = 1$. $h_\theta \in C^1$.
3. $h_\theta$ is non-decreasing and $\lim_{\theta \to 0} h_\theta(x) - h_\delta(x) = 0$ for any $x \in [0, 1]$, where $h_\delta$ is defined on $[0, 1]$, $h_\delta(x) = 1$ if $x = 1$ and $h_\delta(x) = 0$ otherwise.

We also call such a function $h_\theta$ univariate SF, where $\theta$ is a controlling parameter. Being seemingly a simple formulation, SF has three fine properties for computation.

Firstly, SF shows similar behavior as a probabilistic distribution on two independent variables. That is, if two nodes are impossible to match, then any pair of edges containing the two nodes will never match neither. Mathematically, assuming the matching score of node pair $\langle i, a \rangle$ is $h_\theta(x_{ia})$, we have $f_\theta(x_{ia}, x_{jb}) = 0$ for any $\langle j, b \rangle$ if $h_\theta(x_{ia}) = 0$.

Secondly, the definition of SF eases gradient computing. For a given $f_\theta(x, y) = h_\theta(x) h_\theta(y)$, the approximate version of problem (4) can be expressed in matrix form as:

$$\max_{x} h_\theta^\top A h_\theta$$

s.t. $Hx = 1, x \in [0, 1]^n$ \(^5\)

where $h_\theta = [h_\theta(x_1), ..., h_\theta(x_n)]^\top$. The gradient of objective \(^5\) with respect to $x$ is $\nabla x = 2G A h$, where $G$ is a diagonal matrix with the $i$th element $\partial h_\theta(x_i) / \partial x_i$.

The third advantage of SF is that we can construct new approximation functions via reweighted summation and multiplication of existing ones, e.g. if $h_1$ and $h_2$ are two univariate SFs, it can be trivially verified that $\alpha h_1 + (1 - \alpha) h_2$ for $0 \leq \alpha \leq 1$ and $h_1 \times h_2$ are also univariate SFs.

If we keep the constraints on $x$ intact as in problem \(^5\), and let $p_\theta^* = \max_{x} h_\theta(x)^\top A h_\theta(x)$, where $h_\theta(x) = [h_\theta(x_1), ..., h_\theta(x_n)]^\top$, we have the following theorem:

Theorem 2 $\lim_{\theta \to 0} p_\theta^* = p_\delta$

See supplementary material for proof details. The above theorem guarantees that, if we approximate the quasi-delta function by letting $\theta \to 0$, problem \(^4\) can also be approximated asymptotically. As $h_\theta \in C^1$, gradient-base algorithms can be applied to such approximations.

\(^2\)In fact separable function has its traditional meanings in mathematics, we re-define it in the graph matching context.
4.2 Approximations to Function $f_\delta$

Though we have proved that using $f_\delta$ can derive an equivalent problem i.e. (4), finding its optimal solution is still notoriously difficult. Instead of solving (4) directly, based on the analysis in Sec. 4.1, we introduce approximation functions to $f_\delta$. To simplify the expression, we only present the univariate SF $h$, and the SF $f$ can be obtained using definition (1). It is trivial to show that the SFs derived from the following functions approximate $f_\delta$ when $\theta \rightarrow 0^+$ under the properties in definition (1):

$$h_{\text{Lap}}(x) = \frac{1}{m} \left\{ \exp \left( \frac{x - 1}{\theta} \right) - d \right\}$$  \hspace{1cm} (6a)

$$h_{\text{Gauss}}(x) = \frac{1}{m} \left\{ \exp \left( -\frac{(x - 1)^2}{\theta} \right) - d \right\}$$  \hspace{1cm} (6b)

$$h_{\text{Poly}}(x) = x^\frac{1}{\theta}$$  \hspace{1cm} (6c)

where $d = \exp(-\frac{1}{\theta})$ and $m = 1 - d$. The usage of $m$ and $d$ is to normalize the SFs to satisfy the second property. Figure 1 shows some examples of such functions with varying $\theta$ values. Note that traditional quadratic graph matching model in fact is a special case of our model, which seeks to optimize a model where the SF is derived from $h_{\text{Poly}}$ and $\theta = 1$. Specifically, for the univariate SFs (6a) and (6c), we also have the following proposition.

Proposition 1 For univariate SF $h_{\text{Lap}}$, $h_{\text{Poly}}$, suppose $p_1^*$ and $p_2^*$ are the optimal objectives for (5) with $\theta_1$ and $\theta_2$, respectively. Then we have $p_1^* \geq p_2^*$ if $0 < \theta_2 < \theta_1$.

Together with Theorem 2, this claim means that, given univariate SF $h_{\text{Lap}}$ or $h_{\text{Poly}}$, the optimal objective of (5) will converge as $\theta \rightarrow 0^+$ monotonically.

4.3 Convexity/Concavity Analysis

Sections 4.1 and 4.2 show that original problem (4) can be asymptotically approximated using SFs as $\theta \rightarrow 0$. In this section, we analyze the properties of convexity/concavity under such approximations. We believe this effort is worthwhile as one can employ techniques e.g. self-amplification (10), to convert non-convex/non-concave problems into convex/concave ones with the beneficial properties of convexity/concavity. We first show the equivalence of problem (3) and (5) under global convexity.

Theorem 3 Assume that affinity $A$ is positive definite. If the univariate SF $h_\theta(x) \leq x$ on $[0, 1]$, then the global maxima of problem (2), which is discrete, must also be the global maxima of problem (5).

The above theorem builds up a link from problem (2) to problem (5) when $A$ is positive definite. In this case, we first conclude that the optimum to problem (3) is discrete, hence also optimal to (2). Then as long as $h_\theta(x) < x$ on $[0, 1]$ and $h_\theta$ satisfies the second property in Definition 1, this solution is also optimal to problem (5). In this case the optimal objective gap of these three problems becomes 0. We give the following proposition showing under mild conditions, the generalized problem (5) is convexity/concavity-preserving.

Proposition 2 Assume affinity matrix $A$ is positive/negative semi-definite, then as long as the univariate SF $h_\theta$ is convex, the objective of (5) is convex/concave.

Any matrix $A$ can be transformed to positive definite by adding up a diagonal matrix $\lambda I$. The lower bound of $\lambda$ is $\lambda \geq |A^\dagger|$, where $A^\dagger$ is the smallest eigenvalue of $A$ below 0. We define $A^\dagger = A + \lambda I$.

Proposition 3 Assume affinity matrix $A$ is positive definite and univariate SF $h_\theta$ is convex. The optimal value to the following problem is:

$$E_{\text{conv}} = \max_{x} h_\theta^T A^\dagger h_\theta$$ \hspace{1cm} (7)

Then there exists a permutation $x^*$, s.t. $E_{\text{conv}} - E(x^*) \leq n \lambda$ where $E(x^*)$ is the objective value w.r.t. problem (5).
5 Two Optimization Strategies for Generalized GM

Algorithm 1 Path following for GGM

**Input:** $A$, $h_\theta$, $\theta_0$, $0 < \alpha < 1$, initial $x_0$, $k$;
**Output:** $x$

$x \leftarrow x_0$, $\theta \leftarrow \theta_0$

repeat

- make problem according to (5) with $\theta$
- repeat

- compute $V$ using formula (8)
  $x = x + \epsilon \text{vec}(V)$

until Converge

$\theta \leftarrow \alpha \theta$

until $\theta < k$

Algorithm 2 Multiplicative strategy for GGM

**Input:** $A$, $h_\theta$, initial $x_0$; **Output:** $x$

$x \leftarrow x_0$

repeat

- $h \leftarrow h_\theta(x)$
- $h \leftarrow Ah$
- $x \leftarrow h_\theta^{-1}(h)$

until Converge

5.1 Path Following Strategy

It is observed that solving the problem when $\theta$ is too close to 0 is highly non-convex, suggesting the existence of many local optima. Instead, moderate smoothness is desired when we initiate the optimization. This naturally leads to the path following strategy. Such optimization is involved in [9,17,31]. In our implementation, we start by obtaining a local optimum $x_1^*$ from a relatively tractable problem $P_{h_0}$, then we shrink the value of $\theta_1$ by letting $\theta_2 = \alpha \theta_1$ where $0 < \alpha < 1$. Let the starting point for next iteration be $x_1^*$, we solve the updated problem $P_{h_2}$. The iteration continues until convergence condition is satisfied. To verify the convergence, we calculate the energy gap between two consecutive iterations. Formally, for current $x^{(t)}$ at iteration $t$, we calculate the corresponding energy $E^{(t)} = x^{(t)\top} Ax^{(t)}$. The energy at previous iteration $t - 1$ is analogously calculated as $E^{(t-1)} = x^{(t-1)\top} Ax^{(t-1)}$. Then if $|E^{(t)} - E^{(t-1)}| < \eta$, where $\eta$ is a small positive value, we identify the convergence of the iteration. If there is no such $t$, the algorithm stops when reaching the pre-defined maximal iteration number. In all the following experiments, we let $\eta = 10^{-8}$.

Note the problem $P_{h}$ is a general objective with affine constraints. For any gradient-based strategy, projection is necessary to mapping the current solution back to the feasible set. As discussed in [6], projection in variable domain may lead to weak optima. Instead, we use Iterative Bregmann Gradient Projection (IBGP), which is performed in the gradient domain and the convergence is guaranteed [32]. Given current gradient $U = \text{mat}(\nabla x)$, previous matching $X$ and step length $\epsilon$, IBGP performs the following calculations iteratively to obtain $V$ until convergence:

$$V = U - \frac{1}{n} U11^T - \frac{1}{n} 11^TU + \frac{2}{n^2} 11^TU11^T$$  \hspace{1cm} (8a)$$

$$V_{ij} = -X_{ij}/\epsilon \text{ if } V_{ij} < -X_{ij}/\epsilon$$  \hspace{1cm} (8b)$$

$$V_{ij} = (1 - X_{ij})/\epsilon \text{ if } V_{ij} > (1 - X_{ij})/\epsilon$$  \hspace{1cm} (8c)$$

where $V$ is the update direction within the feasible set. Note the iterative procedure in the above equation is a projection. As the constraint set is convex (affinity set), the projection convergence is ensured. Thus in each iteration of update, the algorithm seeks a direction $V$ with ascending guarantee and proceeds a fixed length $\epsilon$. This procedure iterates until convergence or maximal step number. The path following method is summarized in Algorithm 1

5.2 Multiplicative Strategy

Multiplicative strategy on optimizing quadratic objective proved to be convergent under the assumption that $A$ is positive semi-definite [33]. In this strategy, each step amounts to a multiplication $x^{(t+1)} = Ax^{(t)}$ and the objective score over the solution path is non-decreasing. There are works [10,9,6] falling into this category. However, in general affinity $A$ is barely positive semi-definite. While some methods handle this circumstance by adding reweighted identity matrix to $A$ [34], others simply neglect the non-decreasing constraint including some popular algorithms [10,9]. The empirical success of such methods suggests pursuing objective ascending and enhancing matching
accuracy sometimes are paradox. Moreover, the recent study further shows due to noise and the parametric modeling limitation of the affinity function, high accuracy may even corresponds to lower affinity score. Inspired by these observations, we devise a simple yet effective multiplicative strategy by omitting the non-decreasing check. The procedure is shown in Algorithm. In this strategy, the update rule involves calculating inverse function of $h_\theta$. While it is found the multiplicative method converges much faster and hence the overall run time is less compared with the path following method.

6 Experiments

Three popular benchmarks are used including Random Graph Matching, CMU house sequence and Caltech-101/MSRC object matching. accuracy, score and ratio are evaluated, where accuracy measures the portion of correctly matched nodes with respect to all nodes, score represents the value of the objective function and ratio emphasizes the ratio between current objective value and the maximal one. The algorithms for comparison include Spectral Matching (SM), Integer Projected Fixed Point (IPFP), Graduated Assignment (GAGM), Reweighted Random Walk (RRWM), Soft-restricted Graduated Assignment (SRGA) and Branching Path Following Matching (BPM). We term our algorithm Generalized Graph Matching (GGM) with a subscript indicating the corresponding Separable Function and optimization strategy. Namely, GGM$_{xy}$ represents the method with Separable Function $x \in \{1: h_{Lap}; p: h_{path}\}$ and optimizing strategy $y \in \{p: Path\; following; m: Multiplication\}$.

In all the experiments, the algorithms with any updating rules are initialized with a uniform matching. For path following strategy of GGM, we set $\theta_0 = 2, \alpha = 0.5, k = 0.2$.

Figure 2: Performance on random graphs. Note BPM’s runtime is significantly more expensive than other methods (empirically an order higher than ours using the public source code) as it simultaneously seeks multiple paths for the best score (though accuracy is similar to ours). In contrast, our method focus on one path no matter the path following or multiplicative strategy is used.

Random Graph Matching This test is performed following the protocol in [10]. For each trial, source graph $G_S$ and destination graph $G_D$ with $n_{in}$ inlier nodes are generated, consisting of vector attributes $a_i^S$ and $a_i^D$ for both nodes and edges (note $a_i$ is a node attribute and $a_{ij}$ is an edge attribute when $i \neq j$). In the initial setting, $G_D$ is the replica of $G_S$. Three types of sub-experiments are conducted with varying graph deformation $\sigma$, number of outliers $n_{out}$ and edge density $\rho$. To deform a graph, we add an independent Gaussian noise $\varepsilon \sim \mathcal{N}(0, \sigma)$ to attribute $a_i^D$ such that $a_i^{D} = a_i^{S} + \varepsilon$.

Thus the resulting affinity is calculated by $A_{ij,ab} = \exp(-|a_{ij}^{S} - a_{ij}^{D}|^2 / \sigma_x^2)$. The parameter $\sigma_x$ is empirically set to be 0.15. In outlier test, we generate the same number of outlier nodes for both graph. In edge density test, we randomly sample $\rho$ portion of corresponding edges from two fully connected graphs. Each type of sub-experiment is independently carried out 500 times, while average accuracy and score are calculated.

Results are shown in Fig[1]: In the deformation and the edge density tests, GGM$_{lp}$ and GGM$_{ip}$ achieve competitive performance compared to state-of-the-art algorithms. Especially when there
Figure 4: Top and bottom row shows examples on CMU house sequence with gap 20 and 80 respectively, by setting \((n^S = 30, n^D = 20)\).

is combination of severe deformation and edge density is low, GGM\(_{pp}\) and GGM\(_{lp}\) outperform the selected counterparts. On the other hand, GGM\(_{pm}\) and GGM\(_{lm}\) reach significant performance close to state-of-the-art e.g. BPM \([31]\). Though multiplicative strategies cannot guarantee ascending objective in each iteration, GGM\(_{pm}\) and GGM\(_{lm}\) are still effective. This supports the discussion of the paradox between matching accuracy and objective score in Section 5.2. We only show results of GGM\(_{lp}\) in the following experiments, as we see no notable performance gap compared to the other settings.

To examine the algorithm sensitivity to the parameters, we also conduct an extra Random Graph Matching experiment with SFs SFs \(h\)\(_{\text{Poly}}\) and \(h\)\(_{\text{Lap}}\) on Algorithm 1. In this test, we let deformation noise 0.15 and edge density 0.8, 20 inliers and 5 outliers. Test is carried out independently for 20 times and the average accuracy is reported. For both the SFs, we observe that \(k = 0.2\) is sufficient to produce satisfying matching accuracy. Thus we conduct the test by varying the values of \(\theta_0\) and \(\alpha\). The results are demonstrated in Table 1. As larger \(\theta_0\) and \(\alpha\) indicate more iterations, and \(\theta_0 < 2\) and \(\alpha < 0.5\) result in decreasing behavior, we employ the setting \(\theta_0 = 2\) and \(\alpha = 0.5\) throughout all experiments.

Table 1: Sensitivity test on \(h\)\(_{\text{Poly}}\) and \(h\)\(_{\text{Lap}}\)

<table>
<thead>
<tr>
<th>(h)(_{\text{Poly}})</th>
<th>(\alpha)</th>
<th>0.7</th>
<th>0.6</th>
<th>0.5</th>
<th>0.4</th>
<th>0.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\theta_0)</td>
<td>3</td>
<td>0.842</td>
<td>0.839</td>
<td>0.841</td>
<td>0.721</td>
<td>0.610</td>
</tr>
<tr>
<td>2</td>
<td>0.905</td>
<td>0.905</td>
<td>0.904</td>
<td>0.848</td>
<td>0.725</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.910</td>
<td>0.905</td>
<td>0.908</td>
<td>0.851</td>
<td>0.717</td>
<td></td>
</tr>
<tr>
<td>0.5</td>
<td>0.823</td>
<td>0.814</td>
<td>0.770</td>
<td>0.652</td>
<td>0.422</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(h)(_{\text{Lap}})</th>
<th>(\alpha)</th>
<th>0.7</th>
<th>0.6</th>
<th>0.5</th>
<th>0.4</th>
<th>0.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\theta_0)</td>
<td>4</td>
<td>0.912</td>
<td>0.909</td>
<td>0.910</td>
<td>0.872</td>
<td>0.685</td>
</tr>
<tr>
<td>3</td>
<td>0.911</td>
<td>0.907</td>
<td>0.903</td>
<td>0.836</td>
<td>0.672</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.904</td>
<td>0.904</td>
<td>0.906</td>
<td>0.811</td>
<td>0.567</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.853</td>
<td>0.844</td>
<td>0.810</td>
<td>0.728</td>
<td>0.472</td>
<td></td>
</tr>
</tbody>
</table>

CMU House Sequence We perform feature point matching on widely used CMU house sequence dataset following the settings in \([36, 10]\). The dataset consists of 111 house images with gradually changing view points. There are 30 landmark points in each frame. Following the protocol in \([10, 31]\), matching test is conducted on totally 560 pairs of images, spaced by varying frame gaps \((10, 20, ..., 100)\). We use 2 settings of nodes \((n^S, n^D) = (30, 30)\) and \((20, 30)\). In case \(n^S < 30\),

Table 2: Performance on natural images from Caltech-101 and MSRC dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>GAGM</th>
<th>IPFP</th>
<th>SRGA</th>
<th>RRWM</th>
<th>SM</th>
<th>FGM</th>
<th>BPM</th>
<th>GGM(_{lp})</th>
</tr>
</thead>
<tbody>
<tr>
<td>accuracy (%)</td>
<td>73.66</td>
<td>75.77</td>
<td>72.86</td>
<td>72.95</td>
<td>65.78</td>
<td>76.35</td>
<td>75.14</td>
<td><strong>76.69</strong></td>
</tr>
<tr>
<td>score ratio</td>
<td>0.933</td>
<td>0.942</td>
<td>0.940</td>
<td>0.946</td>
<td>0.735</td>
<td>0.969</td>
<td>1</td>
<td>0.972</td>
</tr>
</tbody>
</table>
\(n^S\) nodes are randomly sampled from the source graph. The affinity is conducted by 

\[
A_{ij}^{ab} = \exp(-|a_i^S - a_j^D|^2 / \sigma_s^2),
\]

where \(a_i^S\) measures the Euclidean distance between point \(i\) and \(j\), and \(\sigma_s^2 = 2500\). The edge density is set by \(\rho = 1\). One can see when there is no outlier, all methods except for IPFP and SM achieve perfect matching on any gap setting, and we only show the results with outliers. Figure 4 and Figure 3 depict the matching samples and performance curve, respectively.

We also show typical converging behavior of GGMlp and GGMlm on the upper of Figure 3. We note our path following strategy (Alg. 1) converges slower than multiplicative one (Alg. 2) and they obtain similar final accuracy. One can see when there exist outlier points, GAGM and RRWM suffer notable degraded performance. Our algorithm, on the other hand, achieves competitive performance to state-of-the-arts and behaves stably even under severe degradations.

Natural Image Matching

This is a challenging dataset as it includes natural images in arbitrary backgrounds. In line with the protocol in 10, 30 pairs of images are included in this test collected from Caltech-101 37 and MSRC3. In each pair of images, MSER detector 38 and SIFT descriptor 39 are used to obtain the key points and the corresponding node feature. Mutual projection error function 40 is further adopted to calculate the edge affinity. The ground-truth are manually labeled. The results are shown in Table 2 and matching examples are shown in Fig. 5. Our method outperforms selected algorithms w.r.t. accuracy regardless of objective score. This also suggests the paradox between accuracy and score under complex affinity modeling as discussed in 35.

7 Conclusion

By using Separable Functions, we present a family of continuous approximations to the vanilla QAP formulation widely used in graph matching. We explore the relation of such approximations to the original discrete matching problem, and show convergence properties under mild conditions. Based on the theoretical analysis, we propose a novel solver GGM, which achieves remarkable performance in both synthetic and real-world image tests. This gives rise to the possibility of solving graph matching with many alternative approximations with different solution paths.

Acknowledgement

This work was supported in part by a grant from ONR. Junchi Yan is supported in part by NSFC 61602176 and Tencent AI Lab Rhino-Bird Joint Research Program (No. JR201804). Any opinions expressed in this material are those of the authors and do not necessarily reflect the views of ONR.

References


