IEEE Transactions on Neural Networks (in press)

A Lagrangian Relaxation Network for Graph Matching

Anand Rangarajan
Departments of Diagnostic Radiology
and Computer Science
Yale University
New Haven, CT 06520-8042
e-mail: anand@noodle.med.yale.edu

Eric Mjolsness
Department of Computer Science
and Engineering
University of California San Diego (UCSD)
La Jolla, CA 92093-0114
e-mail: emj@cs.ucsd.edu

Abstract

A Lagrangian relaxation network for graph matching is presented. The problem is formulated as follows: given graphs G and g, find a permutation matrix M that brings the two sets of vertices into correspondence. Permutation matrix constraints are formulated in the framework of deterministic annealing. Our approach is in the same spirit as a Lagrangian decomposition approach in that the row and column constraints are satisfied separately with a Lagrange multiplier used to equate the two "solutions." Due to the unavoidable symmetries in graph isomorphism (resulting in multiple global minima), we add a symmetry-breaking self-amplification term in order to obtain a permutation matrix. With the application of a fixpoint preserving algebraic transformation to both the distance measure and self-amplification terms, we obtain a Lagrangian relaxation network. The network performs minimization with respect to the Lagrange parameters and maximization with respect to the permutation matrix variables. Simulation results are shown on 100 node random graphs and for a wide range of connectivities.

Index Terms: Deterministic annealing, free energy, permutation matrix, Lagrangian decomposition, algebraic transformation, Lagrangian relaxation, graph matching, self-amplification, symmetry-breaking, merit function.

1 Introduction

Graph matching is an important part of object recognition systems in computer vision. The problem of matching structural descriptions of an object to those of a model is posed as weighted graph matching [1]-[26]. Good, approximate solutions to inexact, weighted graph matching are usually required. Neural network approaches to graph matching [6, 7, 8, 9, 11, 12, 28, 15, 23, 21, 25, 27] share the feature of other recent approaches to graph matching [29, 30, 14, 31, 20, 24] in that they are not restricted to looking for the right isomorphism. Instead, a measure of distance between the two graphs [32] is minimized. The focus shifts from brittle, symbolic, subgraph isomorphism distance metrics [13, 33] to an "energy" or

distance measure between graphs which is minimized by the correct correspondence of the vertex labels of one graph with respect to the other. Subgraph isomorphism is a special case and is achieved when the distance is zero.

Exact graph isomorphism is an open problem. It is not known to be either NP-hard or have a polynomial time solution[1] [37, 38]. In contrast, subgraph isomorphism is NP-complete [37, 39]. In this paper, we are interested in exact graph isomorphism and matching as they seem to be problems intermediate in difficulty. Our approach handles isomorphism and matching in the same manner. We follow the *deterministic annealing* framework formulated in [40, 41]. This particular approach is useful since it readily generalizes to inexact, weighted graph matching.

We set up our problem as follows: given graphs G and g, find a *permutation matrix* M that minimizes the distance between the two graphs. A permutation matrix is a zero-one matrix whose rows and columns sum to one. When the problem is made more difficult—subgraph isomorphism or inexact graph matching—the permutation matrix constraints get modified. The rows and columns can add up to one or zero. Permutation matrix constraints are formulated quite naturally in the framework of deterministic annealing. The row or column constraints are *winner-take-alls* (WTAs). Our approach is in the same spirit as a *Lagrangian decomposition* approach [42, 43] in that the row and column constraints are satisfied separately with Lagrange multipliers used to equate the two "solutions." Application of a fixpoint preserving transformation [44, 45] to the graph matching distance allows us to express the combination of the distance measure and the permutation matrix constraint using Lagrange parameters.

Due to the unavoidable symmetries involved in graph isomorphism (resulting in multiple global minima), we add a symmetry-breaking *self-amplification* term [46] in order to obtain a permutation matrix. The self-amplification term is similar to the hysteretic annealing performed in [47, 48] and is suited to the deterministic annealing framework employed here. Unlike hysteretic annealing, the self-amplification parameter is held fixed. A fixpoint preserving transformation is also applied to the self-amplification term. With the application of the fixpoint preserving transformations to both the distance measure and self-amplification terms, the energy function becomes linear in the two "solutions". The two permutation matrix variables can now be eliminated, considerably simplifying the energy function. Since the fixpoint preserving transformations reverse the dynamics, a saddle point of the energy function is sought. The resulting Lagrangian relaxation network is tested on the graph isomorphism problem with 100 node random, undirected graphs and on the weighted graph matching problem with 100 node weighted, random graphs and uniform noise added to the links.

In Section 2, we describe the deterministic annealing, Lagrangian decomposition approach to graph matching. Using this approach, we derive the Lagrangian relaxation algorithm in Section 3. Experimental results for graph isomorphism and matching are presented in Section 4. We relate our approach to Yuille and Kosowsky's barrier function approach [49] in in Section 5.

[1] There is significant recent evidence indicating that graph isomorphism may not be NP-hard [34, 35, 36].

2 Graph matching via deterministic annealing

2.1 Problem Formulation

The graph matching problem is formulated in terms of the adjacency matrices of the two graphs. Given the adjacency matrices G_{ab} and g_{ij} of two graphs $G(V, E)$ and $g(v, e)$ respectively, the problem is to find the permutation match matrix M such that the distance between the two graphs is minimized. The adjacency matrices G_{ab} and g_{ij} of the two undirected graphs are represented as symmetric, sparse matrices with zero diagonal entries. The problem is stated as follows:

$$\min_{M} \sum_{ai} \left(\sum_{b} G_{ab} M_{bi} - \sum_{j} M_{aj} g_{ji} \right)^2 \tag{1}$$

$$\text{subject to } \sum_{a} M_{ai} = 1, \sum_{i} M_{ai} = 1$$

with $M_{ai} \in \{0, 1\}$. Equation (1) describes a distance measure between the two graphs G and g. While the distance measure handles isomorphism and matching, these two cases should be differentiated.

For isomorphism, the adjacency matrices of the two graphs (as seen from the simple example below) contain only zero-one entries. When a link exists between nodes "a" and "b", the corresponding entry G_{ab} is one, otherwise zero. When a correct isomorphism is found, $GM = Mg$ or $g = M^T G M$ and the distance measure in (1) is zero.

For matching, the adjacency matrices of the two graphs can be real-valued. Typically, the graphs are derived from underlying patterns [9, 14, 24] using, for example, the Euclidean distance between feature points. The distance measure in (1) may be non-zero expressing the fact that g is a distorted version of G. (An example of weighted graphs arising from handwritten numerals is shown in Section 4.3.) The distance measure in (1) corresponds to the following statistical generative model: randomly permute the nodes of G and then add additive, white, Gaussian noise (AWGN) to the real-valued links. The resulting expression $\|MgM^T - G\|^2$ can be reduced to $\|GM - Mg\|^2$ using the fact that $M^T M = I$ for a permutation matrix.

The distance measure in (1) contains the rectangle rule of graph matching. To see this, examine the middle term of the distance measure $(-G_{ab} M_{bi} g_{ji} M_{aj})$. Restricting the edges to be either zero or one, we see that all four elements in this chain have to be "on" in order to secure a match. When this happens, vertices "a" and "b" in G are matched to vertices "j" and "i" in g respectively and edges exist between "a" and "b" and "j" and "i" in G and g respectively. This is depicted in Figure 1. The rectangle rule plays an important role in subgraph isomorphism; here, one looks for the match matrix yielding the largest number of rectangles [9].

Figure 1 about here

2.2 A simple example

An example of isomorphic graphs is shown in Figure 2. For this example, we can generate the adjacency matrices of the two graphs G (person) and g (face) as follows:

Figure 2 about here.

$$G = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \text{ and } g = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The two graphs are also displayed in Figure 3. Note the similarities between Figure 2 and Figure 3.

Figure 3 about here.

The graph are planar enabling easy visualization. We ran the Lagrangian relaxation algorithm on the two graphs displayed in Figure 3. Since there is a vertical axis of symmetry for both graphs, a number of isomorphisms (multiple global minima) exist. Two isomorphisms obtained by our algorithm are shown in Figure 4. The dotted lines in Figure 4 depict the correspondences between nodes in G and g. It is easy to check that the correspondences represent true isomorphisms.

Permutations are natural representations of isomorphisms and can be equally well expressed as lists or matrices. The example on the left in Figure 4 shows the list permutation $[0, 9, 1, 8, 7, 6, 5, 4, 3, 2]$ indicating that node 0 in G matches to node 0 in g, $1 \rightarrow 9$, $2 \rightarrow 1$, $3 \rightarrow 8$ etc. Similarly, the example on the right in Figure 4 shows the list permutation $[1, 9, 0, 8, 7, 5, 6, 4, 3, 2]$ indicating that node 0 in G matches to node 1 in g and $1 \rightarrow 9$, $2 \rightarrow 0$, $3 \rightarrow 8$ etc. Both solutions are correct. A permutation matrix as the name suggests is a matrix representation of permutation. It is a square zero-one matrix with rows and columns summing to one. Below, we show both correct isomorphisms as lists (P) and matrices (M). The relationship between the permutation list and the permutation matrix is formally stated as follows: If $P(a) = i,\ a, i, \in \{0, \ldots, N-1\}$, then $M_{ai} = 1$, with all other elements of M_{ai} being zero. From this it follows that in graph isomorphism, node a of G matches to node i in g when $M_{ai} = 1$.

Figure 4 about here.

$$
\begin{array}{c|c}
P = [0,9,1,8,7,6,5,4,3,2] & P = [1,9,0,8,7,5,6,4,3,2] \\
M = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} &
M = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\end{array}
\tag{2}
$$

2.3 Lagrangian Decomposition

We now turn our attention to the constraints. The distance measure contains the match matrix M which has to satisfy permutation matrix constraints. Besides the usual integrality constraint on the match variables ($M_{ai} \in \{0,1\}$), these constraints include row and column WTAs ($\sum_a M_{ai} = 1$ and $\sum_i M_{ai} = 1$). The constraints on M for graph matching are the same as the constraints for the traveling salesman problem (TSP).

Neural net approaches (beginnning with the celebrated Hopfield-Tank network [50]) have typically expressed these constraints via penalty functions. However, contrary to the standard penalty function approach [51], the penalty function parameters are seldom varied. Regardless of the manner of constraint satisfaction, the original discrete problem is converted into a nonlinear optimization problem, with the match matrix assuming values inside the unit hypercube. The permutation matrix constraints get transformed into *doubly stochastic matrix* constraints ($\sum_a M_{ai} = 1$, $\sum_i M_{ai} = 1$ and $M_{ai} \in [0,1]$). An $x \log(x) + (1-x)\log(1-x)$ barrier function constrains the match matrix to be inside the unit hypercube. Gradually changing the barrier function parameter is identical to *mean-field annealing* (MFA) [52, 53].

Beginning with Kanter and Sompolinsky [54], several authors [55, 40, 56, 57, 58] have noted that a more compact representation of the WTA constraint is available via what is known as a Potts glass. The Potts glass approach is also a mean field method. This method utilizes the fact that there are only N configurations (and not 2^N) satisfying the WTA constraint ($\sum_i s_i = 1,\ s_i \in \{0,1\}$). The WTA constraint is kept satisfied in a soft manner by having the mean field variables sum to one ($\sum_i s_i = 1$) and is termed *softmax* [59]. However, the two WTA constraints required by the doubly stochastic matrix cannot be kept simultaneously satisfied. (Despite having relaxed the integrality constraint, we shall continue to refer to the row and column constraints as WTAs.)

The basic idea in this paper is to start with two match matrices $M_{ai}^{(1)}$ and $M_{ai}^{(2)}$ which have the

respective properties

$$\sum_a M_{ai}^{(1)} = 1, \text{ and } \sum_i M_{ai}^{(2)} = 1 \tag{3}$$

always satisfied. Two softmax Potts glasses are used. Assuming that the claim in (3) can be established, the new objective is

$$\min_{M^{(1)}, M^{(2)}} D(M^{(1)}, M^{(2)}) = \sum_{ai} \left(\sum_b G_{ab} M_{bi}^{(1)} - \sum_j M_{aj}^{(2)} g_{ji} \right)^2 \tag{4}$$

$$\text{subject to } M_{ai}^{(1)} = M_{ai}^{(2)}. \tag{5}$$

The constraint in (5) is set up using a Lagrange parameter λ. This technique of having two copies of the same variable satisfy different constraints and subsequently equating them is referred to as Lagrangian decomposition in the operations research literature [42, 43].

We now return to our simple person-face example of the previous section. One possible $M^{(1)}$ and $M^{(2)}$ generated from our simple graph isomorphism example is shown below.

$$M^{(1)} = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \text{ and } M^{(2)} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6}$$

While the two match matrices are somewhat similar, they violate the row and column constraints respectively. The columns of $M^{(1)}$ sum to one, whereas the rows of $M^{(2)}$ sum to one. Clearly, if our Lagrangian relaxation network converges to this particular $M^{(1)}$ and $M^{(2)}$, it has not produced a permutation matrix. This can happen if and only if the Lagrange parameter λ does not converge to its correct value.

Our energy function is constructed with four ingredients: (i) the graph matching distance measure, (ii) Lagrangian decomposition with the twin softmaxes $M^{(1)}$ and $M^{(2)}$, (iii) equating the two "solutions" via the Lagrange parameter λ and (iv) a self-amplification term $(\frac{\gamma}{2} \sum_{ai} M_{ai} (1 - M_{ai}))$ with a parameter γ. In the next section, we present the free energy with the above four ingredients.

2.4 A Free Energy from Statistical Physics

First, we introduce the machinery of Gibbs distributions [60]. A Gibbs probability distribution is specified by an energy function $E(S)$:

$$\Pr(\mathbf{S} = S) = \frac{1}{Z(\beta)} \exp\left(-\beta E(S)\right) \tag{7}$$

where β is the inverse temperature to be used subsequently in the deterministic annealing procedure. In (7), $Z(\beta)$ is the *partition function* which is a sum over all configurations of S ($Z(\beta) = \sum_S \exp\left[-\beta E(S)\right]$). Rewriting (7), we get

$$-\frac{1}{\beta}\log\left(Z(\beta)\right) = E(S) - \frac{1}{\beta}\log\left(\Pr(\mathbf{S} = S)\right). \tag{8}$$

Taking expectations (8) and noting that the left side remains the same, we get

$$-\frac{1}{\beta}\log\left(Z(\beta)\right) = \sum_S \Pr(\mathbf{S} = S)E(S) - \frac{1}{\beta}\sum_S \Pr(\mathbf{S} = S)\log\left(\Pr(\mathbf{S} = S)\right).$$

This is a well known statistical physics identity [61] and immediately leads to the definition of the *free energy*:

$$F(\beta) \stackrel{\text{def}}{=} -\frac{1}{\beta}\log\left(Z(\beta)\right) = \mathcal{E}_\beta\left(E(S)\right) - \frac{1}{\beta}W(\beta) \tag{9}$$

where $\mathcal{E}$ is the expectation operator and $W(\beta)$ is the entropy. As the temperature is reduced (and fewer configurations become likely) the entropy decreases to zero. Also, the expected value of the energy approaches the minimum value of the energy function. Deterministic annealing minimizes the free energy F instead of $E(S)$. The free energy (at low β) is a smooth approximation to the original, non-convex energy [62, 63, 53] and approaches the minimum of $E(S)$ as β tends to infinity.

Unfortunately, the free energy involves the logarithm of the partition function which is intractable. For a more tractable computation, the partition function (at each temperature) is approximated via the saddle-point approximation [58, 40, 49]. The partition function for graph matching is written as follows:

$$Z(\beta) = \sum_{\left\{\begin{array}{c} S^{(1)}, S^{(2)} \\ \sum_a S_{ai}^{(1)} = 1 \\ \sum_i S_{ai}^{(2)} = 1 \\ S_{ai}^{(1)} = S_{ai}^{(2)} \end{array}\right\}} \exp\Big(-\beta\Big[\frac{1}{2}\sum_{ai}\Big(\sum_b G_{ab}S_{bi}^{(1)} - \sum_j S_{aj}^{(2)}g_{ji}\Big)^2 + \frac{\gamma}{2}\sum_{ai}\frac{S_{ai}^{(1)} + S_{ai}^{(2)}}{2}\Big(1 - \frac{S_{ai}^{(1)} + S_{ai}^{(2)}}{2}\Big)\Big]\Big). \tag{10}$$

In Appendix A, we present the derivation of the free energy arising from an application of the saddle-

point approximation to the above partition function. The free energy obtained is

$$F(U^{(1)}, M^{(1)}, U^{(2)}, M^{(2)}, \lambda, \beta) = \frac{1}{2}\sum_{ai}\left(\sum_b G_{ab}M_{bi}^{(1)} - \sum_j M_{aj}^{(2)}g_{ji}\right)^2 \tag{11}$$

$$+\sum_{ai}\left[\lambda_{ai}\left(M_{ai}^{(1)} - M_{ai}^{(2)}\right) + \frac{\gamma}{2}\left(\frac{M_{ai}^{(1)} + M_{ai}^{(2)}}{2}\left[1 - \frac{M_{ai}^{(1)} + M_{ai}^{2}}{2}\right]\right)\right]$$

$$+\frac{1}{\beta}\left(\sum_{ai} U_{ai}^{(1)}M_{ai}^{(1)} - \sum_i \log\sum_a \exp\left\{U_{ai}^{(1)}\right\}\right) + \frac{1}{\beta}\left(\sum_{ai} U_{ai}^{(2)}M_{ai}^{(2)} - \sum_a \log\sum_i \exp\left\{U_{ai}^{(2)}\right\}\right).$$

In (11), $U^{(1)}$ and $U^{(2)}$ are dummy variables involved in the implementation of the two WTA constraints. $M^{(1)}$ and $M^{(2)}$ are continuous-valued versions of the discrete match matrices $S^{(1)}$ and $S^{(2)}$. As the temperature is reduced to zero, $M^{(1)}$ and $M^{(2)}$ converge to the corners of the unit hypercube while respecting the softmax constraints.

2.5 Algebraic Transformations

The free energy in (11) can be considerably simplified. We apply a fixed–point preserving algebraic transformation, essentially a Legendre transformation [44, 45, 27] which transforms

$$X^2 \;\rightarrow\; 2X\tau - \tau^2$$

where it is seen that

$$\max_\tau 2X\tau - \tau^2 = X^2.$$

Both the distance and the self-amplification terms are modified using this algebraic transformation. The distance becomes

$$D(M, \mu) = \sum_{ai}\mu_{ai}\left[\sum_b G_{ab}M_{bi} - \sum_j M_{aj}g_{ji}\right] - \frac{1}{2}\sum_{ai}\mu_{ai}^2. \tag{12}$$

With the application of the above algebraic transformation, the distance measure becomes linear in the match matrix and the dynamics are reversed. We ascend on μ as well as the Lagrange parameters λ. Introducing a new parameter μ does not significantly change the number of variables $(2N^2)$ on which ascent is performed.

The self-amplification term becomes

$$-\frac{\gamma}{2}\sum_{ai}\left(\frac{M_{ai}^{(1)} + M_{ai}^{2}}{2}\right)^2 \rightarrow -\gamma\sum_{ai}\frac{M_{ai}^{(1)} + M_{ai}^{(2)}}{2}\sigma_{ai} + \frac{\gamma}{2}\sum_{ai}\sigma_{ai}^2. \tag{13}$$

The distance measure and the self-amplification term are transformed so that the energy function becomes linear in $M^{(1)}$ and $M^{(2)}$. Also, we have separate control via μ and σ over the amount of ascent/descent on the distance measure and self-amplification terms respectively.

The fixed points of the free energy w.r.t. $U^{(1)}$, $U^{(2)}$, $M^{(1)}$ and $M^{(2)}$ are evaluated by direct differentiation. These variables are eliminated from F and after dropping the terms $\sum_{ai} M_{ai}^{(1)}$ and $\sum_{ai} M_{ai}^{(2)}$ (which are constrained to be a constant) and reversing the sign of F, we get

$$\begin{aligned} F(\mu,\lambda,\sigma) &= \frac{1}{2}\sum_{ai}\mu_{ai}^2 - \frac{\gamma}{2}\sum_{ai}\sigma_{ai}^2 \\ &+ \frac{1}{\beta}\sum_i \log\sum_a \exp\left(-\beta\left[\sum_b G_{ab}\mu_{bi} + \lambda_{ai} - \frac{\gamma}{2}\sigma_{ai}\right]\right) \\ &+ \frac{1}{\beta}\sum_a \log\sum_i \exp\left(+\beta\left[\sum_j g_{ij}\mu_{aj} + \lambda_{ai} + \frac{\gamma}{2}\sigma_{ai}\right]\right). \end{aligned} \tag{14}$$

This is the final form of the free energy. Note that the match variables $M^{(1)}$ and $M^{(2)}$ have been eliminated. The energy function in (14) is to be minimized w.r.t. μ and λ and maximized w.r.t. σ. The dynamics on (μ,λ) and σ can be separated.

When the fixed points are evaluated (partial derivatives with respect to μ, λ and σ are set to zero), we get

$$\begin{aligned} \tfrac{\partial F}{\partial \mu_{ai}} &= 0 \;\Rightarrow\; \mu_{ai} = \textstyle\sum_b G_{ab} M_{bi}^{(1)} - \sum_j M_{aj}^{(2)} g_{ji}, \\ \tfrac{\partial F}{\partial \lambda_{ai}} &= 0 \;\Rightarrow\; M_{ai}^{(1)} = M_{ai}^{(2)}, \text{ and} \\ \tfrac{\partial F}{\partial \sigma_{ai}} &= 0 \;\Rightarrow\; \sigma_{ai} = \tfrac{M_{ai}^{(1)} + M_{ai}^{(2)}}{2}, \end{aligned}$$

where

$$M_{ai}^{(1)} = \frac{\exp\left(-\beta\left[\sum_c G_{ac}\mu_{ci} + \lambda_{ai} - \frac{\gamma}{2}\sigma_{ai}\right]\right)}{\sum_b \exp\left(-\beta\left[\sum_c G_{bc}\mu_{ci} + \lambda_{bi} - \frac{\gamma}{2}\sigma_{bi}\right]\right)}, \text{ and} \tag{15}$$

$$M_{ai}^{(2)} = \frac{\exp\left(\beta\left[\sum_k g_{ik}\mu_{ak} + \lambda_{ai} + \frac{\gamma}{2}\sigma_{ai}\right]\right)}{\sum_j \exp\left(\beta\left[\sum_k g_{jk}\mu_{ak} + \lambda_{aj} + \frac{\gamma}{2}\sigma_{aj}\right]\right)} \tag{16}$$

are the two softmaxes corresponding to $\sum_a M_{ai}^{(1)} = 1$ and $\sum_i M_{ai}^{(2)} = 1$ respectively. At its fixed point, the reversed parameter μ is equal to the distance measure. For graph isomorphism, μ converges to zero, and hence, the μ^2 term can be dropped from the free energy. The parameter μ then becomes a Lagrange parameter satisfying the graph isomorphism constraint—graph matching distance equals zero.

We have obtained a free energy which expresses the graph matching and permutation matrix constraints. In the next section, we describe the algorithm that extremizes the above energy function.

3 The Lagrangian relaxation algorithm

A saddle point of the energy function in (14) is sought. The energy function is minimized w.r.t. μ and λ and maximized w.r.t. σ. In Appendix B, we show that the energy function in (14) is weakly convex w.r.t. the reversed parameter μ and the Lagrange parameters λ. Hence, local minima do not occur during

minimization. Relaxation of the constraints via μ and λ and subsequent optimization w.r.t. them is referred to as *Lagrangian relaxation* [64, 42]. While minimizing the energy function, the main objective is not to find the Lagrange parameters but to satisfy the constraints in (4) and (5). To this end, we use the *merit function* [51] rather than the energy function to determine the line search parameter during each iteration of a conjugate-gradient (CG) update. The merit function for our energy function is

$$\begin{aligned} E_{\text{merit}}(\mu, \lambda) &= \frac{1}{2} \sum_{ai} \left[\left(\frac{\partial F}{\partial \mu_{ai}} \right)^2 + \left(\frac{\partial F}{\partial \lambda_{ai}} \right)^2 \right] \\ &= \frac{1}{2} \sum_{ai} \left[\left(\mu_{ai} - \sum_b G_{ab} M_{bi}^{(1)} + \sum_j M_{aj}^{(2)} g_{ji} \right)^2 + \left(M_{ai}^{(1)} - M_{ai}^{(2)} \right)^2 \right]. \end{aligned} \tag{17}$$

Using the merit function to obtain the line search parameter allows us to escape the typically slow, oscillatory convergence of the Lagrange parameters [65]. The merit function in (17) is a combination of the graph matching distance measure and doubly stochastic matrix constraint deviation. Reducing the merit function satisfies the twin requirements of distance minimization and constraint satisfaction. The $l^{(2)}$ norm of the constraint violation is obtained from the merit function ($l^{(2)}$ norm $= \frac{\sqrt{2E_{\text{merit}}}}{N}$) and used to set convergence thresholds. Since the energy function is weakly convex, the merit function (and therefore the $l^{(2)}$ norm as well) are guaranteed not to increase for gradient descent dynamics [51].

While maximizing the energy function w.r.t. σ, the objective function is very slowly and gradually increased (not maximized) at each temperature. But it is always minimized w.r.t. μ and λ. This approach is used since graph matching distance minimization and constraint satisfaction are more crucial than self-amplification.

THE LAGRANGIAN RELAXATION ALGORITHM

I. INITIALIZE: Set $\beta = \beta^{(0)}$ and λ, μ, σ to zero.

II. DESCEND: Run a CG algorithm w.r.t. (μ, λ) in the free energy (14) until $l^{(2)}$ norm $< l^{(2)}_{\text{thr}}$. Use the merit function in (17) to determine the line search parameter.

III. ASCEND: Set $\sigma_{ai} = \frac{M_{ai}^{(1)} + M_{ai}^{(2)}}{2} + \xi$ (ξ is a uniform random number in $[-\varepsilon, \varepsilon]$). This performs self-amplification.

IV. ANNEAL: Increase β. $\beta^{(k+1)} = (1-\delta)\beta^{(k)} + \delta\beta_{\text{max}}$. Return to step II unless all M_{ai} have converged ($\forall ai,\ M_{ai}(1 - M_{ai}) < \kappa$).

Note from the description of the algorithm that while the energy function is minimized w.r.t. the Lagrange parameters, it is merely increased (one iteration) w.r.t. σ at each temperature.

4 Simulation results

All reported simulations were executed on Silicon Graphics Indigo workstations with R4000 and R4400 processors.

4.1 Graph Isomorphism

We are first interested in testing our method on the isomorphism problem which requires that the distance between the two graphs be equal to zero. This is achieved by dropping the second term of (12) and treating μ as a Lagrange parameter instead of as a reversed "neuron" [44] arising from the fixpoint preserving transformation.

We ran the Lagrangian relaxation algorithm on 100 node undirected, random graphs $(N = 100)$ with weights in $\{0, 1\}$ and for a wide range of connectivities. It is difficult to generate random graphs with the exact number of edges as specified by the desired connectivity [66]. Instead, we generated random graphs with the number of edges approximating the desired connectivity. The chosen connectivities are $\{0.001, 0.01, 0.02, 0.03, 0.04\}$ and from 0.05 to 0.5 in steps of 0.025. Sixty examples were generated at each connectivity giving a total of 1440 examples. The results are presented in Figures 5 and 6. In all experiments, the following parameters were used: $\delta = 0.01$, $\beta^{(0)} = 1.0$ and $\beta_{\max} = 100$. The self-amplification parameter γ was set to $\frac{N}{20} = 5$. For most connectivities, a number of permutation matrices map G onto g. These different solutions correspond to multiple global minima. The self-amplification term performs the useful function of *symmetry-breaking* [67, 68]. The match matrix traverses a series of bifurcations in starting from an almost uniform doubly stochastic matrix $(M_{ai} = \frac{1}{N} +$ random noise$)$ and ending with the final permutation matrix. The bifurcations due to symmetry-breaking start occurring at $\beta = \frac{2N}{\gamma} = 40$ for fully connected 100 node graphs but much before that for sparsely connected graphs. The $l^{(2)}$ norm threshold ($l^{(2)}_{\text{thr}}$), the M_{ai} convergence threshold κ and the symmetry-breaking parameter ε were set to 0.02, 0.001 and 0.0001 respectively. Our method always gave a correct permutation matrix for all the examples except for 3 failures out of 60 runs taken at an average connectivity percentage of 1%. From Figures 5 and 6, it is seen that the number of iterations and the relaxation time are large for both extremely small connectivities and for moderate to large connectivities. The method can also be tried for connectivities larger than 0.5 but for this problem it is easier to reverse the weights on all the edges. These results are clearly superior to the results obtained by Simić [28]. Simić's deterministic annealing network [28] could not reliably find isomorphisms for all connectivities less than 30% in 75 node random graphs.

Figure 5 about here.

Figure 6 about here.

Figure 7 about here.

Figure 7 shows the change in energy as the iterations proceed for the sixty examples of graph isomorphism at a connectivity of 0.01. With the exception of the three failures (indicated by the flatlines), the pattern is the same. An isomorphism is obtained when the energy goes to zero and when all the match matrix variables have converged. This corresponds to a saddle point. The energy first decreases rapidly, passes through zero (with the match matrix very far from convergence) and then increases gradually approaching zero. The gradual progress towards integral solutions is seen in Figure 8 where the changes in the match matrix are depicted through 36 temperatures. The black dots appearing for the first time in Figure 8(c) indicate convergence to integral solutions of the respective rows *and* columns. These stay in place and the remaining rows and columns gradually converge. In Figure 8(f), the permutation matrix obtained is shown. The black dots are ones with the white spaces being zeros—similar to the permutation matrices in (2).

Figure 8 about here.

4.2 Weighted Graph Matching

Next, we ran the Lagrangian relaxation algorithm on 100 node weighted graphs. The weights are chosen randomly in the interval $[0, 1]$. Uniform noise in the interval $[-\epsilon, \epsilon]$ is then added to all the edges. While the distance measure in (1) calls for additive, white, Gaussian noise (AWGN), we added independent uniform noise because it facilitates comparison with three other non-neural methods of graph matching, namely, eigendecomposition [29], symmetric polynomial transform [30] and linear programming [31]. All these methods have been tested with uniform noise, allowing for a straightforward comparison. The noise variance corresponding to uniform noise is $\epsilon^2/3$. We chose nine noise variances roughly ranging from 0.0025 to 0.02 with sixty examples at each value giving us a total of 540 examples. The results are presented in Figure 9. [In the figure, the number of iterations is plotted against $\ln(\text{variance}/0.0025)/2$.] In all experiments, we used the following parameters: $\delta = 0.01$, $\beta^{(0)} = 1.0$ and $\beta_{\max} = 100$. The self-amplification parameter γ was set to $\frac{N}{10} = 10$. The $l^{(2)}$ norm threshold ($l^{(2)}_{\text{thr}}$), the M_{ai} convergence threshold κ and the symmetry-breaking parameter ε were set to 0.02, 0.001 and 0.0001 respectively. Our method always gave a correct permutation matrix for all the examples indicating that we have not "broken" our algorithm. These results are superior to the results obtained by the eigendecomposition approach [29], the symmetric polynomial transform approach [30] and the linear programming approach [31]. In the other approaches, 10 node fully connected graphs were matched with uniform noise added to the links. The linear programming approach was the most accurate giving exact results on 50 trials only when the noise variance was less than 0.0035. In contrast, we get exact results in our 100 node fully

connected graph matching experiments on 60 trials when the noise variance is less than 0.02—a factor of 10 better both in terms of the noise variance and size of graphs handled. We are extending this approach to inexact, weighted graph matching where the limitations of this approach can be better studied.

Figure 9 about here.

4.3 Matching handwritten numerals

In order to test the algorithm in a more realistic setting, we examined the problem of finding correspondences between a model and data. This problem frequently arises in object recognition (computer vision) [69].

To this end, we generated a handwritten numeral "model" (along with rotated and scaled versions) using an X windows interface. The interface also provided us with "ground-truth" coordinate point locations for the models. Let $X_a,\ a = 1, \ldots, N$ denote the N 2-D points describing the model numeral. (Each X_a is a 2-D vector.) A weighted graph G—invariant to rotation, translation and scaling of the 2-D point set X—was obtained in the following manner. Let the distance between all pairs of points in X

$$d_X(a,b) = \|X_a - X_b\|^2,\ \forall\ a, b\ \in \{1, \ldots, N\}.$$

The distance d_X is invariant to rotation and translation of X. In order to get scale invariance, we first sorted the distances $d_X(a,b)$. Then we assigned the *normalized rank order* $k/\frac{N(N-1)}{2}$, where $k \in \{1, \ldots, \frac{N(N-1)}{2}\}$ is the rank of a distance element $d_X(a,b)$, to the corresponding element (a,b) of the adjacency matrix of graph G. The diagonal elements of G_{ab} were set to zero. The matrix is symmetric since $d_X(a,b) = d_X(b,a)$. Using the rank order rather than the distance makes graph G scale invariant. We have described the process by which a weighted graph G is obtained from point set X.

Figure 10 about here.

The same procedure is repeated for all the "data" point sets obtained by rotating, scaling and adding noise to the model X. The correspondences between model and data numerals is depicted in Figure 10. Figures 10(a) and (b) illustrate the case for pure rotation and scale. The model "9" is rotated by 180° in Figure 10(a) and scaled down by a factor of four—we have increased the scale somewhat for better visualization—in Figure 10(b). Since the graphs are rotation and scale invariant, the correspondences are near perfect as expected. In Figures 10(c) and (d), we present an anecdotal study of the noise performance. The effect of noise is simulated by adding independent additive Gaussian (AWGN) random numbers to the *point set X and not the weights of graph G*. The same procedure for obtaining a graph from the point set was carried out. The correspondences shown in Figures 10(c) and (d) are degraded somewhat. The graphs G and g resulting in the procedure described above have 91 nodes. We ran the Lagrangian

relaxation algorithm with the following parameters: $\delta = 0.01$, $\beta^{(0)} = 1.0$ and $\beta_{\max} = 100$. The self-amplification parameter γ was set to 5. The $l^{(2)}$ norm threshold ($l^{(2)}_{\text{thr}}$), the M_{ai} convergence threshold κ and the symmetry-breaking parameter ε were set to 0.02, 0.01 and 0.0001 respectively. This anecdotal study demonstrates an application of the graph matching technique to the problem of model matching in computer vision.

We have recently developed a related graph matching algorithm based on an objective function which is equivalent up to fixpoints to the free energy in (11) [27, 26]. While the objective function is similar, the optimization dynamics are very different from those of the Lagrangian relaxation algorithm. Line searches and Lagrangian relaxation are eschewed in favor of a discrete algorithm which satisfies the doubly stochastic matrix constraints via iterated row and column normalization. The method is much faster (by a factor of 10 or more) but less accurate. The Lagrangian relaxation algorithm is amenable to analog VLSI implementation and its utility may lie there. The discrete algorithm may be competitive (in terms of speed and accuracy) with other algorithms implemented on digital computers.

5 Relationship to Yuille and Kosowsky

The free energy in (11) is closely related to the approach of Yuille and Kosowsky [49][2]. Their approach, applicable to quadratic assignment problems in general, expresses the doubly stochastic matrix constraint using an $x \log(x)$ barrier function and two Lagrange parameters (enforcing WTA constraints). The barrier function parameter plays the role of deterministic annealing.

The free energy before elimination of the match matrix variables is

$$\begin{aligned} F(U^{(1)}, M^{(1)}, U^{(2)}, M^{(2)}, \lambda, \beta) = \frac{1}{2} \sum_{ai} \left(\sum_b G_{ab} M^{(1)}_{bi} - \sum_j M^{(2)}_{aj} g_{ji} \right)^2 \\ + \sum_{ai} \left[\lambda_{ai} \left(M^{(1)}_{ai} - M^{(2)}_{ai} \right) + \frac{\gamma}{2} \left(\frac{M^{(1)}_{ai} + M^{(2)}_{ai}}{2} \left[1 - \frac{M^{(1)}_{ai} + M^{2}_{ai}}{2} \right] \right) \right] \\ + \frac{1}{\beta} \left(\sum_{ai} U^{(1)}_{ai} M^{(1)}_{ai} - \sum_i \log \sum_a \exp \left\{ U^{(1)}_{ai} \right\} \right) + \frac{1}{\beta} \left(\sum_{ai} U^{(2)}_{ai} M^{(2)}_{ai} - \sum_a \log \sum_i \exp \left\{ U^{(2)}_{ai} \right\} \right). \end{aligned}$$

Differentiating the above free energy w.r.t. $U^{(1)}$ and equating the result to zero, we get

$$M^{(1)}_{ai} = \frac{\exp\left(U^{(1)}_{ai}\right)}{\sum_b \exp\left(U^{(1)}_{bi}\right)}. \tag{18}$$

This is rewritten to reflect the dependence of $U^{(1)}$ on $M^{(1)}$:

$$M^{(1)}_{ai} \sum_b \exp\left(U^{(1)}_{bi}\right) = \exp\left(U^{(1)}_{ai}\right)$$

[2]We acknowledge Alan Yuille for pointing out the close relationship between our Lagrangian decomposition approach and the barrier function approach developed in [49].

$$\Rightarrow U_{ai}^{(1)} = \log\left(M_{ai}^{(1)}\right) + \log\sum_b \exp\left(U_{bi}^{(1)}\right). \tag{19}$$

Equation (19) consists of a set of self-consistent equations from which $U^{(1)}$ cannot be eliminated. Substituting (19) in the entropy term

$$S = \sum_{ai} U_{ai}^{(1)} M_{ai}^{(1)} - \sum_i \log\sum_b \exp\left(U_{bi}^{(1)}\right),$$

we get

$$S = \sum_{ai} M_{ai}^{(1)} \log\left(M_{ai}^{(1)}\right) + \sum_i \left[\log\sum_b \exp\left(U_{bi}^{(1)}\right)\right]\left(\sum_a M_{ai}^{(1)} - 1\right).$$

The $\log\sum_a \exp\left(U_{ai}^{(1)}\right)$ term is replaced by a new variable ν_i which is a Lagrange parameter enforcing the WTA constraint $\sum_a M_{ai}^{(1)} = 1$. A new $x\log(x)$ barrier function is obtained. This was first shown by Yuille and Kosowsky [49]. Unlike traditional barrier functions [51], this barrier function (shown in Figure 11) does not go to infinity as $x \to 0$. Nevertheless, it achieves the objective of keeping the match matrix positive.

Figure 11 about here.

An $x\log(x)$ barrier function and a Lagrange parameter μ_a are similarly associated with the WTA constraint $\sum_i M_{ai}^{(2)} = 1$. When this is done, the new free energy takes the form

$$F(M^{(1)}, M^{(2)}, \lambda, \mu, \nu, \beta) = \frac{1}{2}\sum_{ai}\left(\sum_b G_{ab} M_{bi}^{(1)} - \sum_j M_{aj}^{(2)} g_{ji}\right)^2 \tag{20}$$

$$+\sum_{ai}\left[\lambda_{ai}\left(M_{ai}^{(1)} - M_{ai}^{(2)}\right) + \frac{\gamma}{2}\left(\frac{M_{ai}^{(1)} + M_{ai}^{(2)}}{2}\left[1 - \frac{M_{ai}^{(1)} + M_{ai}^{2}}{2}\right]\right)\right]$$

$$+\frac{1}{\beta}\left\{\sum_{ai} M_{ai}^{(1)}\log(M_{ai}^{(1)}) + \sum_i \nu_i\left(\sum_a M_{ai}^{(1)} - 1\right) + \sum_{ai} M_{ai}^{(2)}\log(M_{ai}^{(2)}) + \sum_a \mu_a\left(\sum_i M_{ai}^{(2)} - 1\right)\right\}.$$

Differentiating the above free energy w.r.t. λ and setting the result to zero, we get $M_{ai}^{(1)} = M_{ai}^{(2)}$. Substituting this in (20), we get

$$F(M, \mu, \nu, \beta) = \frac{1}{2}\sum_{ai}\left(\sum_b G_{ab} M_{bi} - \sum_j M_{aj} g_{ji}\right)^2 + \frac{\gamma}{2}\sum_{ai} M_{ai}(1 - M_{ai}) \tag{21}$$

$$+\frac{1}{\beta}\left\{\sum_{ai} 2M_{ai}\log(M_{ai}) + \sum_i \nu_i\left(\sum_a M_{ai} - 1\right) + \sum_a \mu_a\left(\sum_i M_{ai} - 1\right)\right\}.$$

A Lagrangian relaxation network similar to the one reported in this paper can be obtained from the above free energy. At a theoretical level, the two formulations are equivalent. No penalty functions have been used to express the WTA constraints and doubly stochastic match matrices are obtained at each temperature setting when the Lagrange parameters converge.

Other optimizing network approaches using penalty functions [50, 28] can be related to ours by replacing Lagrange parameters with penalty functions. For example, Simić [28] uses a Potts glass softmax and a penalty function for the two WTA constraints. The penalty functions satisfy the doubly stochastic matrix constraint asymptotically as opposed to our approach where this constraint is satisfied at each temperature.

6 Conclusions

We have formulated and tested a Lagrangian relaxation network for graph matching. A distinguishing feature of our approach is exact constraint satisfaction at each temperature within deterministic annealing. In addition, the reversal of the dynamics on the graph matching term allows us to simultaneously ascend on the matching and constraint satisfaction terms. The results for both matching and isomorphism are impressive. Perfect results are obtained for almost all instances. We are currently extending our Lagrangian relaxation approach to inexact, weighted graph matching with subsequent application to object recognition. Finally, Lagrangian relaxation networks similar to the ones presented here can be designed for other quadratic assignment problems like the traveling salesman problem.

Acknowledgements

This work is supported by AFOSR grant F49620-92-J-0465, ONR/ARPA grant N00014-92-J-4048 and the Yale Neuroengineering and Neuroscience Center (NNC). A brief summary of this paper was presented at ICNN '94 [70]. We thank Charles Garrett, Gene Gindi, Steven Gold, Chien-Ping Lu, Suguna Pappu and Alan Yuille for stimulating discussions.

A Derivation of the free energy

In this section, we begin with the partition function in (10) and systematically demonstrate the change of variables and other calculations leading to the final free energy in (14).

$$Z(\beta) = \sum_{\left\{ \begin{array}{c} S^{(1)}, S^{(2)} \\ \sum_a S_{ai}^{(1)} = 1 \\ \sum_i S_{ai}^{(2)} = 1 \\ S_{ai}^{(1)} = S_{ai}^{(2)} \end{array} \right\}} \exp\Big(-\beta \Big[\frac{1}{2} \sum_{ai} \Big(\sum_b G_{ab} S_{bi}^{(1)} - \sum_j S_{aj}^{(2)} g_{ji} \Big)^2$$

$$+\frac{\gamma}{2}\sum_{ai}\frac{S_{ai}^{(1)}+S_{ai}^{(2)}}{2}\left(1-\frac{S_{ai}^{(1)}+S_{ai}^{(2)}}{2}\right)\Big]\Big). \tag{22}$$

We now illustrate the Potts glass encoding of the WTA constraint. The treatment follows [40]. Consider the following partition function arising from an energy function $E(S)$, $\sum_i S_i = 1$, $S_i \in \{0,1\}$. Following [40], we get

$$\begin{aligned}
Z(\beta) &= \sum_{\left\{\begin{array}{c} S| \\ \sum_i S_i = 1 \end{array}\right\}} \exp\left(-\beta E(S)\right) \\
&= \sum_{\left\{\begin{array}{c} S| \\ \sum_i S_i = 1 \end{array}\right\}} \int_R dM\, \delta(S-M) \exp\left(-\beta E(M)\right) \\
&= \int_R dM \exp(-\beta E(M)) \int_I dU \sum_{\left\{\begin{array}{c} S| \\ \sum_i S_i = 1 \end{array}\right\}} \exp\left[\sum_i U_i (S_i - M_i)\right] \\
&= \int_R dM \int_I dU \exp\left[-\beta E(M) - \sum_i U_i M_i + \log\left(\sum_i \exp(U_i)\right)\right].
\end{aligned}$$

The deterministic annealing trick evaluates the above integrand at its saddle points and uses the result as an approximation to the partition function integral.

$$Z(\beta) \approx Z(\hat{M},\hat{U},\beta) = \exp\left[-\beta\left(E(\hat{M}) + \frac{1}{\beta}\left[\sum_i \hat{U}_i\hat{M}_i - \log\sum_i \exp(\hat{U}_i)\right]\right)\right] \stackrel{\text{def}}{=} \exp\left(-\beta F\right) \tag{23}$$

where $\hat{M}$ and $\hat{U}$ are the solutions to

$$\frac{\partial F}{\partial M_i} = 0 \Rightarrow \hat{U}_i = -\beta\frac{\partial E}{\partial M_i}, \text{ and } \frac{\partial F}{\partial U_i} = 0 \Rightarrow \hat{M}_i = \frac{\exp(U_i)}{\sum_i \exp(U_i)}. \tag{24}$$

Here, $\sum_i \hat{M}_i = 1$ and a single winner is obtained as β is increased. A continuation method is obtained from the saddle-point approximation. At each temperature setting, a relaxation network is used to obtain the pair $\hat{M}$ and $\hat{U}$. The inverse temperature β is increased and the earlier solution is used as an initial condition for the next stage.

Writing the details of the saddle-point approximation for the graph matching objective and its two WTAs is awkward. Below, we present a "bare bones" version of the derivation.

$$Z(\beta) = \int_R dM^{(1)} \int_I dU^{(1)} \int_R dM^{(2)} \int_I dU^{(2)} \int_I d\lambda \sum_{\left\{\begin{array}{c} S^{(1)}| \\ \sum_a S_{ai}^{(1)} = 1 \end{array}\right\}} \exp\left[\sum_{ai} U_{ai}^{(1)}(M_{ai}^{(1)} - S_{ai}^{(1)})\right]$$

$$\sum_{\left\{ \begin{array}{c} S^{(2)} | \\ \sum_i S_{ai}^{(2)} = 1 \end{array} \right\}} \exp\left[\sum_{ai} U_{ai}^{(2)}(M_{ai}^{(2)} - S_{ai}^{(2)})\right] \exp\Big(-\beta\Big[\frac{1}{2}\sum_{ai}\Big(\sum_b G_{ab}M_{bi}^{(1)} - \sum_j M_{aj}^{(2)}g_{ji}\Big)^2$$

$$+\frac{\gamma}{2}\sum_{ai}\frac{M_{ai}^{(1)} + M_{ai}^{(2)}}{2}\Big(1 - \frac{M_{ai}^{(1)} + M_{ai}^{(2)}}{2}\Big) + \sum_{ai}\lambda_{ai}(M_{ai}^{(1)} - M_{ai}^{(2)})\Big]\Big)$$

$$\approx \exp\left(-\beta F(\hat{U}^{(1)}, \hat{M}^{(1)}, \hat{U}^{(2)}, \hat{M}^{(2)}, \hat{\lambda}, \beta)\right)$$

where

$$F(U^{(1)}, M^{(1)}, U^{(2)}, M^{(2)}, \lambda, \beta) = \frac{1}{2}\sum_{ai}\left(\sum_b G_{ab}M_{bi}^{(1)} - \sum_j M_{aj}^{(2)}g_{ji}\right)^2$$

$$+\sum_{ai}\left[\lambda_{ai}\left(M_{ai}^{(1)} - M_{ai}^{(2)}\right) + \frac{\gamma}{2}\sum_{ai}\frac{M_{ai}^{(1)} + M_{ai}^{(2)}}{2}\left(1 - \frac{M_{ai}^{(1)} + M_{ai}^{(2)}}{2}\right)\right]$$

$$+\frac{1}{\beta}\left(\sum_{ai} U_{ai}^{(1)}M_{ai}^{(1)} - \sum_i \log\sum_a \exp\left\{U_{ai}^{(1)}\right\}\right) + \frac{1}{\beta}\left(\sum_{ai} U_{ai}^{(2)}M_{ai}^{(2)} - \sum_a \log\sum_i \exp\left\{U_{ai}^{(2)}\right\}\right).$$

B Non-negative definiteness of the Hessian

In this section, we show that the Hessian of the objective function in (14) with respect to the reversed "neuron" μ and the Lagrange parameter λ is non-negative definite. The proof is carried out for the isomorphism case alone. Since the only difference between the energy function for graph isomorphism and graph matching is a convex $\sum_{ai}\mu_{ai}^2$ term, the proof also carries over to the graph matching energy function. For the Hessian to be non-negative definite, we require for any x and y,

$$D = \sum_{aibj}\left[x_{ai}\left(\frac{1}{\beta}\frac{\partial^2 F}{\partial\mu_{ai}\partial\mu_{bj}}\right)x_{bj} + 2y_{ai}\left(\frac{1}{\beta}\frac{\partial^2 F}{\partial\lambda_{ai}\partial\mu_{bj}}\right)x_{bj} + y_{ai}\left(\frac{1}{\beta}\frac{\partial^2 F}{\partial\lambda_{ai}\partial\lambda_{bj}}\right)y_{bj}\right] \geq 0. \tag{25}$$

Now

$$\frac{1}{\beta}\frac{\partial^2 F}{\partial\mu_{ai}\partial\mu_{bj}} = -\frac{1}{\beta}\left(\sum_c G_{ac}\frac{\partial M_{ci}^{(1)}}{\partial\mu_{bj}} - \sum_k \frac{\partial M_{ak}^{(2)}}{\partial\mu_{bj}}g_{ki}\right), \tag{26}$$

$$\frac{1}{\beta}\frac{\partial^2 F}{\partial\lambda_{ai}\partial\lambda_{bj}} = -\frac{1}{\beta}\left(\frac{\partial M_{ai}^{(1)}}{\partial\lambda_{bj}} - \frac{\partial M_{ai}^{(2)}}{\partial\lambda_{bj}}\right), \text{ and} \tag{27}$$

$$\frac{1}{\beta}\frac{\partial^2 F}{\partial\lambda_{ai}\partial\mu_{bj}} = -\frac{1}{\beta}\left(\frac{\partial M_{ai}^{(1)}}{\partial\mu_{bj}} - \frac{\partial M_{ai}^{(2)}}{\partial\mu_{bj}}\right). \tag{28}$$

Rewriting $M_{ci}^{(1)}$ and $M_{ak}^{(2)}$, for convenience:

$$M_{ci}^{(1)} = \frac{\exp\left(-\beta\left[\sum_b G_{bc}\mu_{bi} + \lambda_{ci}\right]\right)}{\sum_d \exp\left(-\beta\left[\sum_b G_{bd}\mu_{bi} + \lambda_{di}\right]\right)}, \quad M_{ak}^{(2)} = \frac{\exp\left(\beta\left[\sum_j g_{kj}\mu_{aj} + \lambda_{ak}\right]\right)}{\sum_l \exp\left(\beta\left[\sum_j g_{lj}\mu_{aj} + \lambda_{al}\right]\right)}. \tag{29}$$

From (29), the relevant partial derivatives required in (26), (27) and (28) are evaluated.

$$\frac{\partial M_{ci}^{(1)}}{\partial \mu_{bj}} = -\beta\delta_{ij} M_{ci}^{(1)} \left(G_{bc} - \sum_d G_{bd} M_{di}^{(1)}\right), \tag{30}$$

$$\frac{\partial M_{ak}^{(2)}}{\partial \mu_{bj}} = +\beta\delta_{ab} M_{ak}^{(2)} \left(g_{kj} - \sum_l g_{lj} M_{al}^{(2)}\right), \tag{31}$$

and

$$\frac{\partial M_{ai}^{(1)}}{\partial \lambda_{bj}} = -\beta\delta_{ij} M_{ai}^{(1)} \left(\delta_{ab} - M_{bi}^{(1)}\right), \tag{32}$$

$$\frac{\partial M_{ai}^{(2)}}{\partial \lambda_{bj}} = \beta\delta_{ab} M_{ai}^{(2)} \left(\delta_{ij} - M_{aj}^{(2)}\right), \tag{33}$$

where δ_{ij} and δ_{ab} are Kronecker delta functions.
From (26), (30) and (31), we get

$$\begin{aligned}\frac{1}{\beta}\frac{\partial^2 F}{\partial \mu_{ai}\partial \mu_{bj}} &= \delta_{ij}\left[\sum_c G_{ac}G_{bc}M_{ci}^{(1)} - \left(\sum_c G_{ac}M_{ci}^{(1)}\right)\left(\sum_d G_{bd}M_{di}^{(1)}\right)\right] \\ &+ \delta_{ab}\left[\sum_k g_{ki}g_{kj}M_{ak}^{(2)} - \left(\sum_k g_{ki}M_{ak}^{(2)}\right)\left(\sum_l g_{lj}M_{al}^{(2)}\right)\right].\end{aligned} \tag{34}$$

From (27), (32) and (33), we get

$$\frac{1}{\beta}\frac{\partial^2 F}{\partial \lambda_{ai}\partial \lambda_{bj}} = \delta_{ij} M_{ai}^{(1)}\left(\delta_{ab} - M_{bi}^{(1)}\right) + \delta_{ab} M_{ai}^{(2)}\left(\delta_{ij} - M_{aj}^{(2)}\right). \tag{35}$$

And finally, from (28), (30) and (31), we get

$$\frac{1}{\beta}\frac{\partial^2 F}{\partial \lambda_{ai}\partial \mu_{bj}} = \delta_{ij} M_{ai}^{(1)}\left(G_{ba} - \sum_c G_{bc}M_{ci}^{(1)}\right) + \delta_{ab} M_{ai}^{(2)}\left(g_{ij} - \sum_k g_{kj}M_{ak}^{(2)}\right). \tag{36}$$

Using (34), (35) and (36), we evaluate D in (25):

$$\begin{aligned}D &= \sum_{aibj} x_{ai}\delta_{ij}\left[\sum_c G_{ac}G_{bc}M_{ci}^{(1)} - \left(\sum_c G_{ac}M_{ci}^{(1)}\right)\left(\sum_d G_{bd}M_{di}^{(1)}\right)\right]x_{bj} \\ &+ \sum_{aibj} x_{ai}\delta_{ab}\left[\sum_k g_{ki}g_{kj}M_{ak}^{(2)} - \left(\sum_k g_{ki}M_{ak}^{(2)}\right)\left(\sum_l g_{lj}M_{al}^{(2)}\right)\right]x_{bj}\end{aligned}$$

$$
\begin{aligned}
&+ \quad 2\sum_{aibj} y_{ai}\left[\delta_{ij}M_{ai}^{(1)}\left(G_{ba}-\sum_c G_{bc}M_{ci}^{(1)}\right)x_{bj}+\delta_{ab}M_{ai}^{(2)}\left(g_{ij}-\sum_k g_{kj}M_{ak}^{(2)}\right)x_{bj}\right] \\
&+ \quad \sum_{aibj} y_{ai}\left[\delta_{ij}M_{ai}^{(1)}\left(\delta_{ab}-M_{bi}^{(1)}\right)+\delta_{ab}M_{ai}^{(2)}\left(\delta_{ij}-M_{aj}^{(2)}\right)\right]y_{bj}. \qquad (37)
\end{aligned}
$$

This is rearranged to give

$$
\begin{aligned}
D \quad = \quad & \sum_{i,c}\left(\sum_a G_{ac}x_{ai}\right)\left(\sum_b G_{bc}x_{bi}\right)M_{ci}^{(1)}-\sum_i\left[\sum_c\left(\sum_a G_{ac}x_{ai}\right)M_{ci}^{(1)}\right]\left[\sum_d\left(\sum_b G_{bd}x_{bi}\right)M_{di}^{(1)}\right] \\
+ \quad & \sum_{a,k}\left(\sum_i g_{ki}x_{ai}\right)\left(\sum_j g_{kj}x_{aj}\right)M_{ak}^{(2)}-\sum_a\left[\sum_k\left(\sum_i g_{ki}x_{ai}\right)M_{ak}^{(2)}\right]\left[\sum_l\left(\sum_j g_{lj}x_{aj}\right)M_{al}^{(2)}\right] \\
+ \quad & 2\sum_{i,a} y_{ai}\left(\sum_b G_{ba}x_{bi}\right)M_{ai}^{(1)}-2\sum_{i,a} y_{ai}\left[\sum_c\left(\sum_b G_{bc}x_{bi}\right)M_{ci}^{(1)}\right] \\
+ \quad & 2\sum_{a,i} y_{ai}\left(\sum_j g_{ij}x_{aj}\right)M_{ai}^{(2)}-2\sum_{a,i} y_{ai}\left[\sum_k\left(\sum_j g_{kj}x_{aj}\right)M_{ak}^{(2)}\right] \\
+ \quad & \sum_{i,a} y_{ai}^2 M_{ai}^{(1)}-\sum_i\left(\sum_a y_{ai}M_{ai}^{(1)}\right)\left(\sum_b y_{bi}M_{bi}^{(1)}\right) \\
+ \quad & \sum_{a,i} y_{ai}^2 M_{ai}^{(2)}-\sum_a\left(\sum_i y_{ai}M_{ai}^{(2)}\right)\left(\sum_j y_{aj}M_{aj}^{(2)}\right).
\end{aligned}
$$

The above expression is simplified as follows. Define

$$
u_{ci} \stackrel{\text{def}}{=} \sum_a G_{ac}x_{ai},\ v_{ak} \stackrel{\text{def}}{=} \sum_i g_{ki}x_{ai}. \qquad (38)
$$

D is rewritten in a more compact form:

$$
\begin{aligned}
D \quad = \quad & \sum_i\left[\sum_a u_{ai}^2 M_{ai}^{(1)}-\left(\sum_a u_{ai}M_{ai}^{(1)}\right)^2\right]+\sum_a\left[\sum_i v_{ai}^2 M_{ai}^{(2)}-\left(\sum_i v_{ai}M_{ai}^{(2)}\right)^2\right] \\
+ \quad & 2\sum_i\left[\sum_a y_{ai}u_{ai}M_{ai}^{(1)}-\left(\sum_a y_{ai}M_{ai}^{(1)}\right)\left(\sum_a u_{ai}M_{ai}^{(1)}\right)\right] \\
+ \quad & 2\sum_a\left[\sum_i y_{ai}v_{ai}M_{ai}^{(2)}-\left(\sum_i y_{ai}M_{ai}^{(2)}\right)\left(\sum_i v_{ai}M_{ai}^{(2)}\right)\right] \\
+ \quad & \sum_i\left[\sum_a y_{ai}^2 M_{ai}^{(1)}-\left(\sum_a y_{ai}M_{ai}^{(1)}\right)^2\right]+\sum_a\left[\sum_i y_{ai}^2 M_{ai}^{(2)}-\left(\sum_i y_{ai}M_{ai}^{(2)}\right)^2\right]. \qquad (39)
\end{aligned}
$$

And after further simplification,

$$
D \quad = \quad \sum_{i,a}\left[u_{ai}^2 M_{ai}^{(1)}+2y_{ai}u_{ai}M_{ai}^{(1)}+y_{ai}^2 M_{ai}^{(1)}\right]+\sum_{a,i}\left[v_{ai}^2 M_{ai}^{(2)}+2y_{ai}v_{ai}M_{ai}^{(2)}+y_{ai}^2 M_{ai}^{(2)}\right]
$$

$$- \sum_i \left[\left(\sum_a u_{ai} M_{ai}^{(1)} \right)^2 + 2 \left(\sum_a y_{ai} M_{ai}^{(1)} \right) \left(\sum_a u_{ai} M_{ai}^{(1)} \right) + \left(\sum_a y_{ai} M_{ai}^{(1)} \right)^2 \right]$$

$$- \sum_a \left[\left(\sum_i v_{ai} M_{ai}^{(2)} \right)^2 + 2 \left(\sum_i y_{ai} M_{ai}^{(2)} \right) \left(\sum_i v_{ai} M_{ai}^{(2)} \right) + \left(\sum_i y_{ai} M_{ai}^{(2)} \right)^2 \right].$$

A second round of definitions completes the "compactification" of D.

$$w_{ai} \stackrel{\text{def}}{=} u_{ai} + y_{ai}, \;\; z_{ai} \stackrel{\text{def}}{=} v_{ai} + y_{ai}. \tag{40}$$

And finally, D is shown to be

$$D = \sum_i \left[\sum_a w_{ai}^2 M_{ai}^{(1)} - \left(\sum_a w_{ai} M_{ai}^{(1)} \right)^2 \right] + \sum_a \left[\sum_i z_{ai}^2 M_{ai}^{(2)} - \left(\sum_i z_{ai} M_{ai}^{(2)} \right)^2 \right]. \tag{41}$$

For any i, setting $f_a = w_{ai} \sqrt{M_{ai}^{(1)}}$ and $g_a = \sqrt{M_{ai}^{(1)}}$ and using the Cauchy-Schwartz inequality

$$\sum_a f_a^2 \sum_a g_a^2 \;\geq\; \left(\sum_a f_a g_a \right)^2$$

with equality occurring when $f_a = h g_a$ and h is an arbitrary non-zero constant, we get [41]

$$\begin{aligned} \sum_a w_{ai}^2 M_{ai}^{(1)} \sum_a M_{ai}^{(1)} &\geq \left(\sum_a w_{ai} M_{ai}^{(1)} \right)^2 \\ \Rightarrow \sum_a w_{ai}^2 M_{ai}^{(1)} &\geq \left(\sum_a w_{ai} M_{ai}^{(1)} \right)^2 \end{aligned} \tag{42}$$

using the fact that $\sum_a M_{ai}^{(1)} = 1$ is always satisfied.

Similarly, for any a, setting $f_i = z_{ai} \sqrt{M_{ai}^{(2)}}$ and $g_i = \sqrt{M_{ai}^{(2)}}$ and using the Cauchy-Schwartz inequality, we get

$$\begin{aligned} \sum_i z_{ai}^2 M_{ai}^{(2)} \sum_i M_{ai}^{(2)} &\geq \left(\sum_i z_{ai} M_{ai}^{(2)} \right)^2 \\ \Rightarrow \sum_i z_{ai}^2 M_{ai}^{(2)} &\geq \left(\sum_i z_{ai} M_{ai}^{(2)} \right)^2 \end{aligned} \tag{43}$$

using the fact that $\sum_i M_{ai}^{(2)} = 1$ is always satisfied.

Therefore $D \;\geq\; 0$ with equality occurring for those vectors $w_{ai} = s_i, \; \forall \; a$ and $z_{ai} = t_a, \; \forall \; i$, where s and t are arbitrary vectors.

References

[1] L. S. Davis, "Shape matching using relaxation techniques", *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 1, pp. 60–72, Jan. 1979.

[2] L. G. Shapiro and R. M. Haralick, "Structural descriptions and inexact matching", *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 3, pp. 504–519, Sept. 1981.

[3] M. A. Eshera and K. S. Fu, "An image understanding system using attributed symbolic representation and inexact graph-matching", *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 8, pp. 604–619, Sept. 1986.

[4] C. von der Malsburg and E. Bienenstock, "Statistical coding and short-term synaptic plasticity: A scheme for knowledge representation in the brain", in E. Bienenstock, F. Fogelman-Soulie, and G. Weisbuch, editors, *Disordered Systems and Biological Organization*, pp. 247–252. Springer–Verlag, Berlin, 1986.

[5] J. A. Feldman, M. A. Fanty, and N. H. Goddard, "Computing with structured neural networks", *IEEE Computer*, vol. 21, pp. 91–103, March 1988.

[6] C. von der Malsburg, "Pattern recognition by labeled graph matching", *Neural Networks*, vol. 1, pp. 141–148, 1988.

[7] R. Kree and A. Zippelius, "Recognition of topological features of graphs and images in neural networks", *J. Phys. A*, vol. 21, pp. L813–L818, 1988.

[8] P. Kuner and B. Ueberreiter, "Pattern recognition by graph matching–combinatorial versus continuous optimization", *Intl. J. Pattern Recognition and Artificial Intelligence*, vol. 2, pp. 527–542, 1988.

[9] E. Mjolsness, G. Gindi, and P. Anandan, "Optimization in model matching and perceptual organization", *Neural Computation*, vol. 1, pp. 218–229, 1989.

[10] J. Segen, "Model learning and recognition of nonrigid objects", *in IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 597–602. IEEE Press, 1989.

[11] W. Li and N. M. Nasrabadi, "Object recognition based on graph matching implemented by a Hopfield–style neural network", *in Intl. Joint Conf. on Neural Networks (IJCNN)*, vol. 2, pp. 287–290. IEEE Press, 1989.

[12] V. Tresp, "A neural network approach for three–dimensional object recognition", in R. P. Lippmann, J. E. Moody, and D. S. Touretzky, editors, *Advances in Neural Information Processing Systems 3*, pp. 306–312, San Mateo, CA, 1991. Morgan Kaufmann.

[13] E. Wong, "Model matching in robot vision by subgraph isomorphism", *Pattern Recognition*, vol. 25, pp. 287–303, 1992.

[14] S. Z. Li, "Matching: invariant to translations, rotations and scale changes", *Pattern Recognition*, vol. 25, pp. 583–594, 1992.

[15] S.-S. Yu and W.-H. Tsai, "Relaxation by the Hopfield neural network", *Pattern Recognition*, vol. 25, pp. 197–208, Feb. 1992.

[16] R. Bergevin and M. D. Levine, "Generic object recognition: Building and matching coarse descriptions from line drawings", *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 15, pp. 19–36, Jan. 1993.

[17] M. Lades, J. C. Vorbruggen, J. Buhmann, J. Lange, C. von der Malsburg, R. P. Wurtz, and W. Konen, "Distortion invariant object recognition in the dynamic–link architecture", *IEEE Trans. Computers*, vol. 42, pp. 300–311, Mar. 1993.

[18] A. H. Gee, S. V. B. Aiyer, and R. W. Prager, "An analytical framework for optimizing neural networks", *Neural Networks*, vol. 6, pp. 79–97, 1993.

[19] E. Mjolsness, "Connectionist grammars for high-level vision", in V. Honavar and L. Uhr, editors, *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration*, pp. 423–451. Academic Press, San Diego, CA, 1994.

[20] M. Krcmar and A. P. Dhawan, "Application of genetic algorithms in graph matching", *in IEEE Intl. Conf. on Neural Networks (ICNN)*, vol. 6, pp. 3872–3876. IEEE Press, 1994.

[21] T.-W. Chen and W.-C. Lin, "A neural network approach to CSG-based 3-D object recognition", *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 16, pp. 719–725, July 1994.

[22] A. Pearce, T. Caelli, and W. F. Bischof, "Rulegraphs for graph matching in pattern recognition", *Pattern Recognition*, vol. 27, pp. 1231–1247, 1994.

[23] S. S. Young, P. D. Scott, and N. M. Nasrabadi, "Object recognition using multi-layer Hopfield neural network", *in IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 417–422. IEEE Press, 1994.

[24] W. J. Christmas, J. Kittler, and M. Petrou, "Structural matching in computer vision using probabilistic relaxation", *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 17, pp. 749–764, Aug. 1995.

[25] P. N. Suganthan, E. K. Teoh, and D. P. Mital, "Pattern recognition by graph matching using the Potts MFT neural networks", *Pattern Recognition*, vol. 28, pp. 997–1009, 1995.

[26] S. Gold and A. Rangarajan, "A graduated assignment algorithm for graph matching", 1996, *IEEE Trans. Patt. Anal. Mach. Intell.*, (in press).

[27] A. Rangarajan, S. Gold, and E. Mjolsness, "A novel optimizing network architecture with applications", 1996, *Neural Computation*, (in press).

[28] P. D. Simic, "Constrained nets for graph matching and other quadratic assignment problems", *Neural Computation*, vol. 3, pp. 268–281, 1991.

[29] S. Umeyama, "An eigendecomposition approach to weighted graph matching problems", *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 10, pp. 695–703, Sept. 1988.

[30] H. A. Almohamad, "A polynomial transform for matching pairs of weighted graphs", *J. Applied Math. Modeling*, vol. 15, April 1991.

[31] H. A. Almohamad and S. O. Duffuaa, "A linear programming approach for the weighted graph matching problem", *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 15, pp. 522–525, May 1993.

[32] A. K. C. Wong and M. You, "Entropy and distance of random graphs with applications to structural pattern recognition", *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 7, pp. 599–609, Sept. 1985.

[33] W.-H. Tsai and K.-S. Fu, "Error–correcting isomorphisms of attributed relational graphs for pattern analysis", *IEEE Trans. Syst. Man, Cyber.*, vol. 9, pp. 757–768, Dec. 1979.

[34] R. Motwani, "personal communication", Sept. 1994.

[35] D. S. Johnson, "The NP–Completeness column: An ongoing guide", *Journal of Algorithms*, vol. 13, pp. 502–524, 1992.

[36] D. S. Johnson, "The NP–Completeness column: An ongoing guide", *Journal of Algorithms*, vol. 9, pp. 426–444, 1988.

[37] M. R. Garey and D. S. Johnson, *Computers and intractability: a guide to the theory of NP–completeness*, W. H. Freeman, San Francisco, CA, 1979.

[38] D. G. Corneil and C. C. Gotlieb, "An efficient algorithm for graph isomorphism", *J. Ass. Comput. Mach.*, vol. 17, pp. 51–64, Jan. 1970.

[39] J. R. Ullman, "An algorithm for subgraph isomorphism", *J. Ass. Comput. Mach.*, vol. 23, pp. 31–42, Jan. 1976.

[40] C. Peterson and B. Soderberg, "A new method for mapping optimization problems onto neural networks", *Intl. Journal of Neural Systems*, vol. 1, pp. 3–22, 1989.

[41] J. J. Kosowsky and A. L. Yuille, "The invisible hand algorithm: Solving the assignment problem with statistical physics", *Neural Networks*, vol. 7, pp. 477–490, 1994.

[42] K. Jornsten and M. Nasberg, "A new Lagrangian relaxation approach to the generalized assignment problem", *European Journal of Operational Research*, vol. 27, pp. 313–323, 1986.

[43] M. Guignard and S. Kim, "Lagrangean decomposition: A model yielding stronger Lagrangean bounds", *Mathematical Programming*, vol. 39, pp. 215–228, 1987.

[44] E. Mjolsness and C. Garrett, "Algebraic transformations of objective functions", *Neural Networks*, vol. 3, pp. 651–669, 1990.

[45] I. Elfadel, "Convex potentials and their conjugates in analog mean-field optimization", *Neural Computation*, vol. 7, pp. 1079–1104, Sept. 1995.

[46] C. von der Malsburg, "Network self-organization", in S. F. Zornetzer, J. L. Davis, and C. Lau, editors, *An Introduction to Neural and Electronic Networks*, pp. 421–432. Academic Press, San Diego, CA, 1990.

[47] S. V. B. Aiyer, M. Niranjan, and F. Fallside, "A theoretical investigation into the performance of the Hopfield model", *IEEE Trans. Neural Networks*, vol. 1, pp. 204–215, 1990.

[48] A. H. Gee and R. W. Prager, "Polyhedral combinatorics and neural networks", *Neural Computation*, vol. 6, pp. 161–180, Jan. 1994.

[49] A. L. Yuille and J. J. Kosowsky, "Statistical physics algorithms that converge", *Neural Computation*, vol. 6, pp. 341–356, May 1994.

[50] J. J. Hopfield and D. Tank, "'Neural' computation of decisions in optimization problems", *Biol. Cyber.*, vol. 52, pp. 141–152, 1985.

[51] D. Luenberger, *Linear and Nonlinear Programming*, Addison–Wesley, Reading, MA, 1984.

[52] G. L. Bilbro, W. E. Snyder, R. Mann, D. Van den Bout, T. Miller, and M. White, "Optimization by mean field annealing", in D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 1*, pp. 91–98. Morgan Kaufmann, San Mateo, CA, 1988.

[53] G. L. Bilbro, W. E. Snyder, S. J. Garnier, and J. W. Gault, "Mean field annealing: A formalism for constructing GNC-like algorithms", *IEEE Trans. Neural Networks*, vol. 3, pp. 131–138, Jan. 1992.

[54] I. Kanter and H. Sompolinsky, "Graph optimisation problems and the Potts glass", *J. Phys. A*, vol. 20, pp. L673–L677, 1987.

[55] D. E. Van den Bout and T. K. Miller III, "Improving the performance of the Hopfield–Tank neural network through normalization and annealing", *Biol. Cyber.*, vol. 62, pp. 129–139, 1989.

[56] D. Geiger and A. L. Yuille, "A common framework for image segmentation", *Intl. Journal of Computer Vision*, vol. 6, pp. 227–243, Aug. 1991.

[57] D. E. Van den Bout and T. K. Miller III, "Graph partitioning using annealed networks", *IEEE Trans. Neural Networks*, vol. 1, pp. 192–203, June 1990.

[58] P. D. Simic, "Statistical mechanics as the underlying theory of 'elastic' and 'neural' optimisations", *Network*, vol. 1, pp. 89–103, 1990.

[59] J. S. Bridle, "Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters", in D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pp. 211–217, San Mateo, CA, 1990. Morgan Kaufmann.

[60] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images", *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 6, pp. 721–741, Nov. 1984.

[61] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation*, vol. 1 of *A Lecture Notes volume in the Santa Fe Institute Studies in the Sciences of Complexity*, Addison–Wesley, New York, 1991.

[62] D. Geiger and F. Girosi, "Parallel and deterministic algorithms from MRFs: Surface reconstruction", *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 13, pp. 401–412, May 1991.

[63] A. Rangarajan and R. Chellappa, "A continuation method for image estimation and segmentation", Technical Report CAR-TR-586, Center for Automation Research, University of Maryland, Oct. 1991.

[64] M. L. Fisher, "The Lagrangian relaxation method for solving integer programming problems", *Management Science*, vol. 27, pp. 1–18, 1981.

[65] J. C. Platt and A. H. Barr, "Constraint methods for flexible models", *Computer Graphics*, vol. 22, pp. 279–288, Aug. 1988, SIGGRAPH '88.

[66] S. Skiena, *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*, Addison–Wesley, Reading, Mass., 1990.

[67] H. Haken, *Synergetics–An Introduction*, vol. 1 of *Springer Series in Synergetics*, Springer–Verlag, New York, 3rd edition, 1983.

[68] H. Haken, *Advanced Synergetics*, vol. 20 of *Springer Series in Synergetics*, Springer–Verlag, New York, 2nd edition, 1987.

[69] W. E. L. Grimson, *Object Recognition by Computer: The Role of Geometric Constraints*, Artificial Intelligence. M.I.T. Press, 1990.

[70] A. Rangarajan and E. Mjolsness, "A Lagrangian relaxation network for graph matching", *in IEEE Intl. Conf. on Neural Networks (ICNN)*, vol. 7, pp. 4629–4634. IEEE Press, 1994.

Figure Captions

Figure 1: The rectangle rule for graph isomorphism: vertices "a" and "b" in G are matched to vertices "j" and "i" in g respectively and edges exists between "a" and "b" and "j" and "i" in G and g respectively. A rectangle is constructed from the cycle of vertices $a \stackrel{G_{ab}}{\rightarrow} b \stackrel{M_{bi}}{\rightarrow} i \stackrel{g_{ji}}{\rightarrow} j \stackrel{M_{aj}}{\rightarrow} a$. As a match to "i", "b" will be preferred to "c" or "d" because it has the same degree (3) and can therefore participate in more rectangles. The same holds for a match between "a" and "j".

Figure 2: An example of isomorphic graphs: The person and the face are seen to be isomorphic. A link between any two nodes in one graph has a counterpart in the other graph. There's a redundancy in that the mirror image of the face (person) is also isomorphic to the person (face). Redrawn with permission from *"Compared to what? an introduction to the analysis of algorithms."*, Gregory J. E. Rawlins, page 306, chapter 5, Computer Science Press, W. H. Freeman and Co., New York, 1992.

Figure 3: Graphs of the person (on the left) and the face (on the right) corresponding to the drawings in Figure 2. Both graphs have ten vertices and ten edges.

Figure 4: Illustration of multiple solutions in graph isomorphism. Left: One correct labelling. Right: Another correct labelling.

Figure 5: Number of iterations vs. connectivity: Each "+" in the figure denotes the number of iterations taken for a 100 node graph isomorphism run. A total of 1440 graph isomorphism examples are presented. The interval $[0, 5]\%$ was tested more heavily to obtain a better sampling of the decrease and then increase in the number of iterations. The apparent quantization of the iterations is not understood. The figure on the right is an errorbar plot of the data presented in the figure on the left. Note the decrease in the variance of the number of iterations in the interval $[0, 5]\%$.

Figure 6: Relaxation time vs. connectivity: Each "x" in the figure denotes the wall clock time. The code was not optimized. The figure on the right is an errorbar plot of the data presented in the figure on the left.

Figure 7: Energy vs. number of iterations: In the figure on the left, the approach to a saddle point is depicted for a connectivity of 1%. In all 60 cases, the energy decreases rapidly, goes through zero with the match matrix far from convergence and then slowly ascends. Three failures are indicated by flat lines. The energy does not make the ascent to zero. In the right figure, the same approach to a saddle point of the energy is depicted for one particular example.

Figure 8: Match matrix at the (a) first, (b) ninth, (c) sixteenth, (d) nineteenth, (e) thirty-third, and (f) thirty-sixth temperatures: Gradual evolution from a nearly uniform doubly stochastic matrix to a permutation matrix (with black dots representing ones and white spaces representing zeros) is shown.

Figure 9: Number of iterations vs. noise variance: Each "+" in the figure denotes the number of iterations taken for a 100 node graph matching run. A total of 540 graph matching examples are presented.

Independent, uniform noise is added to each edge. The number of iterations is plotted against $\ln(\text{variance}/0.0025)/2$. The figure on the right is an errorbar plot of the data presented in the figure on the left. Note the gradual increase in the variance of the number of iterations.

Figure 10: Correspondences in matching handwritten numerals: (a) Rotation (b) Scale (c) Additive Gaussian Noise and (d) All of the above.

Figure 11: The $x \log(x)$ barrier function. The free energy resulting from the Lagrangian decomposition approach is related via a change of variables to the approach of Yuille and Kosowsky. The latter approach uses an $x \log(x)$ barrier function and two Lagrange parameters for the two WTA constraints.

Figures

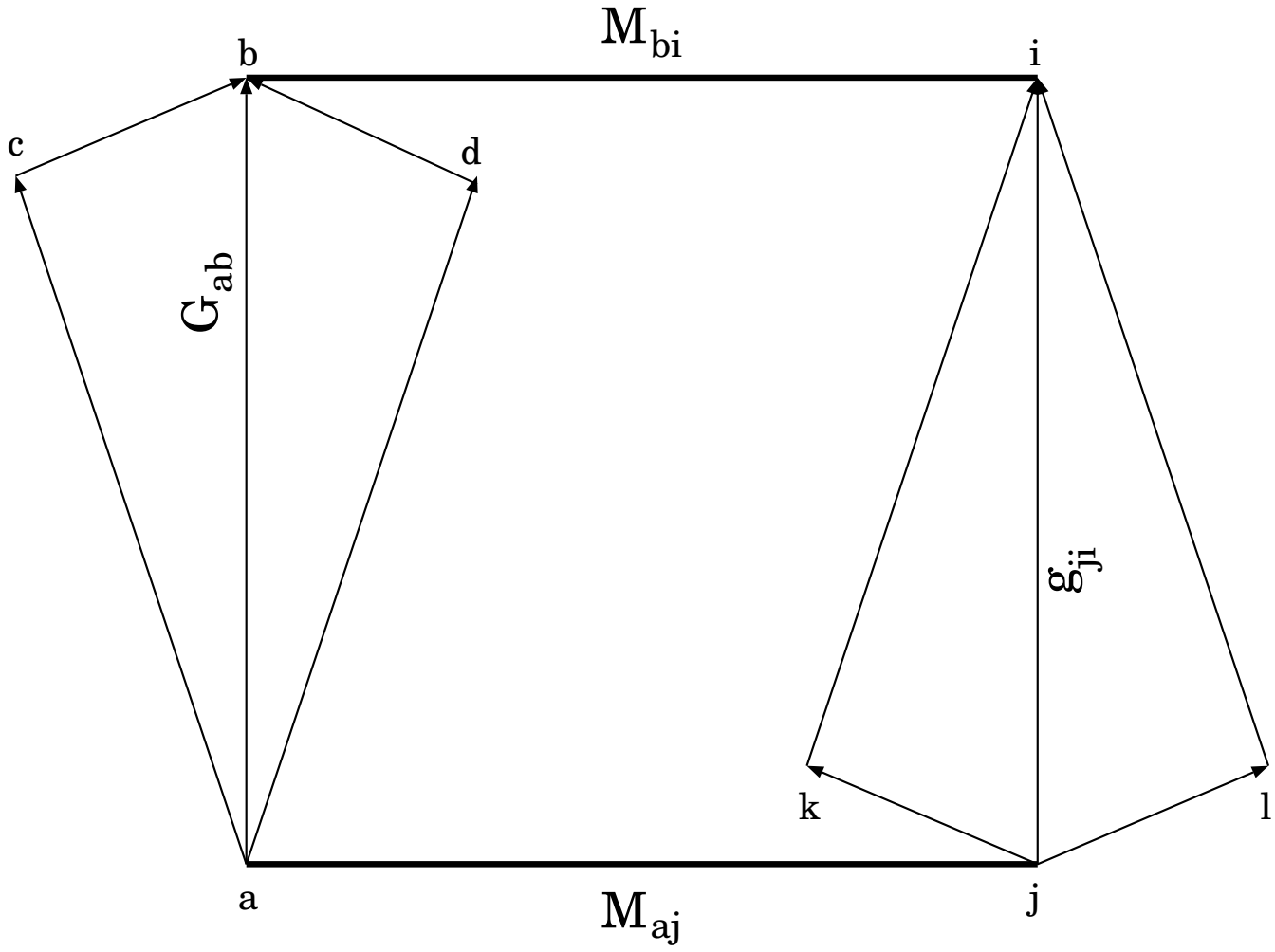


Figure 1:

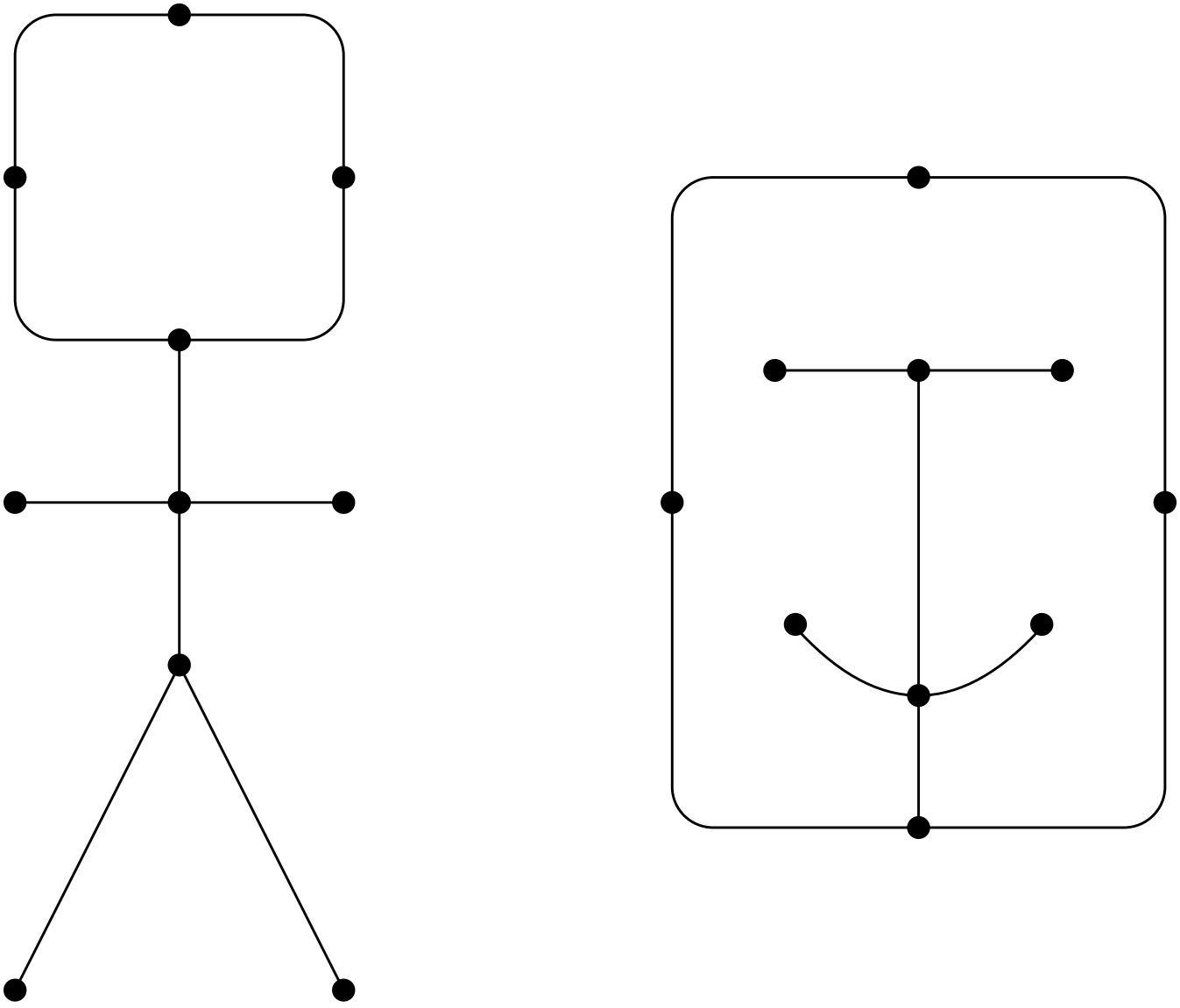


Figure 2:

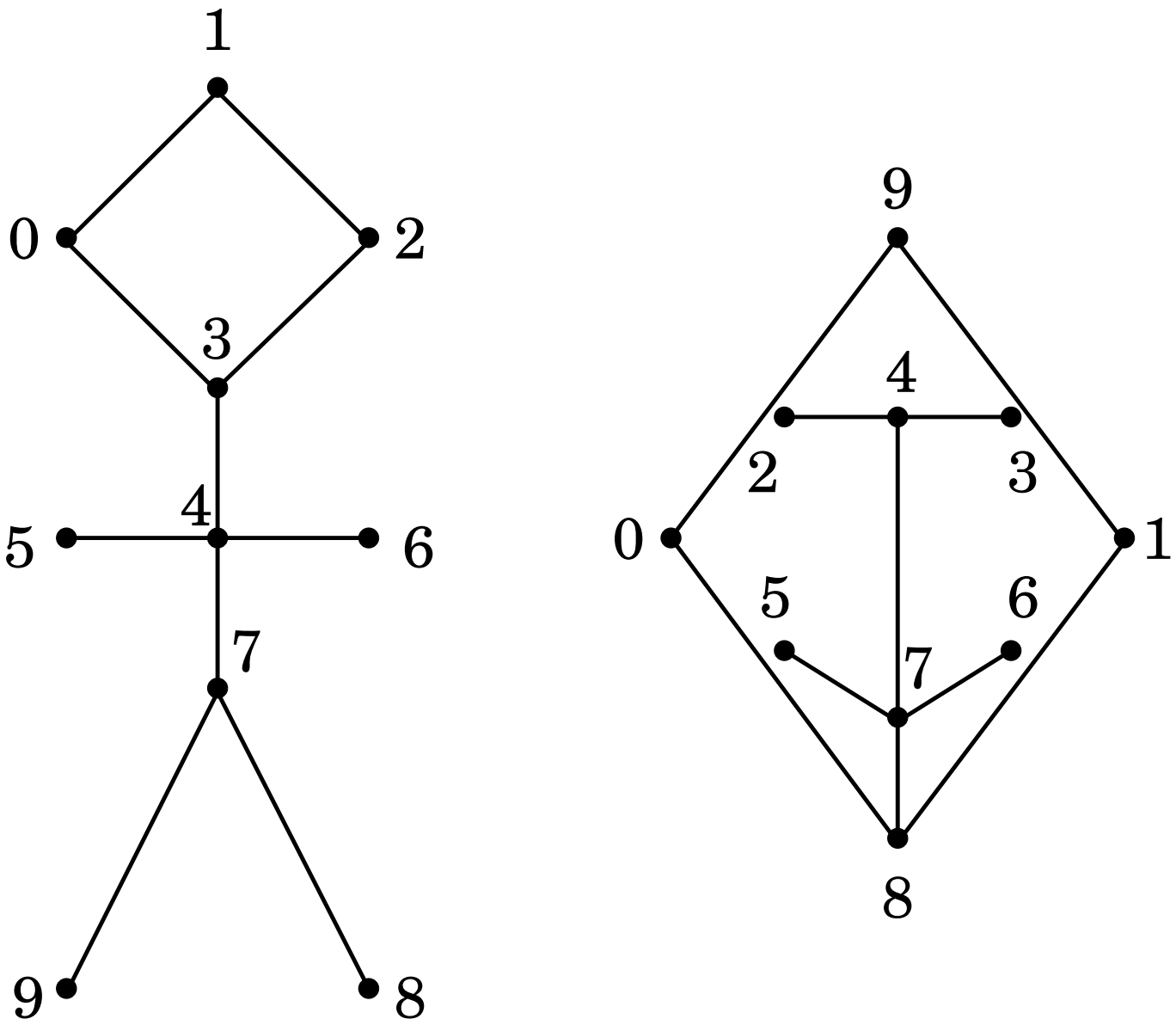


Figure 3:

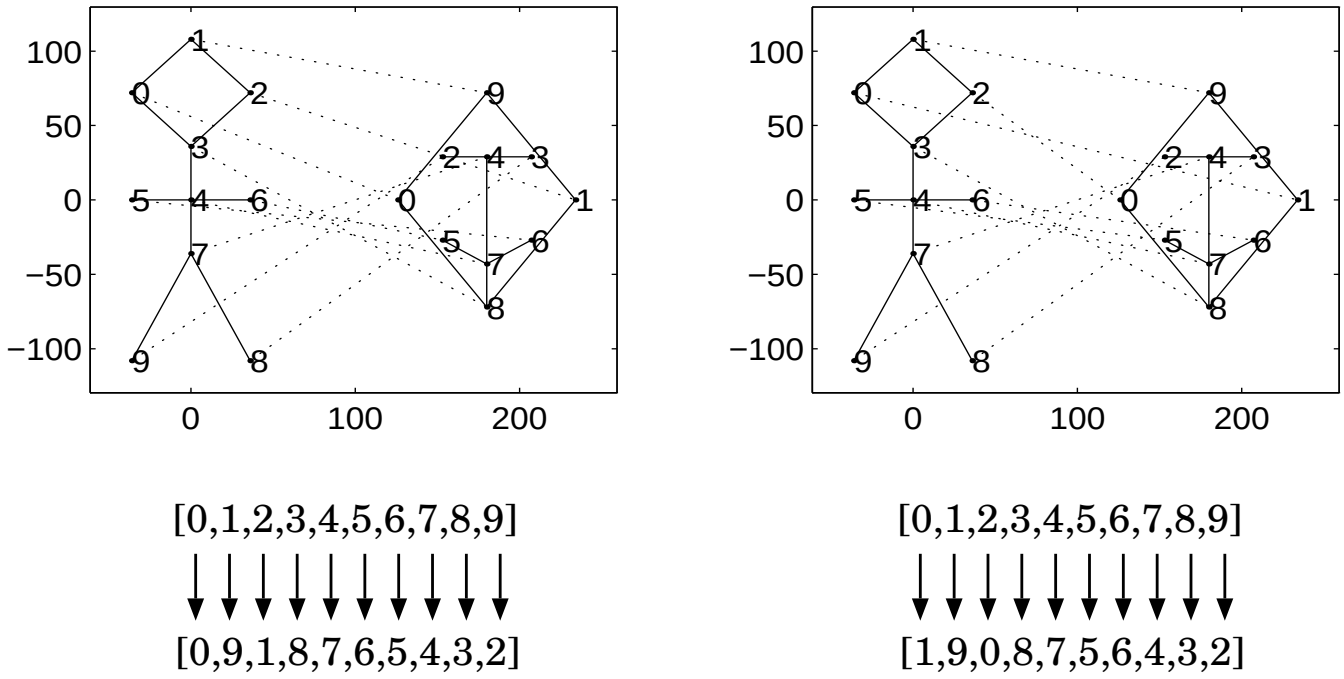


Figure 4:

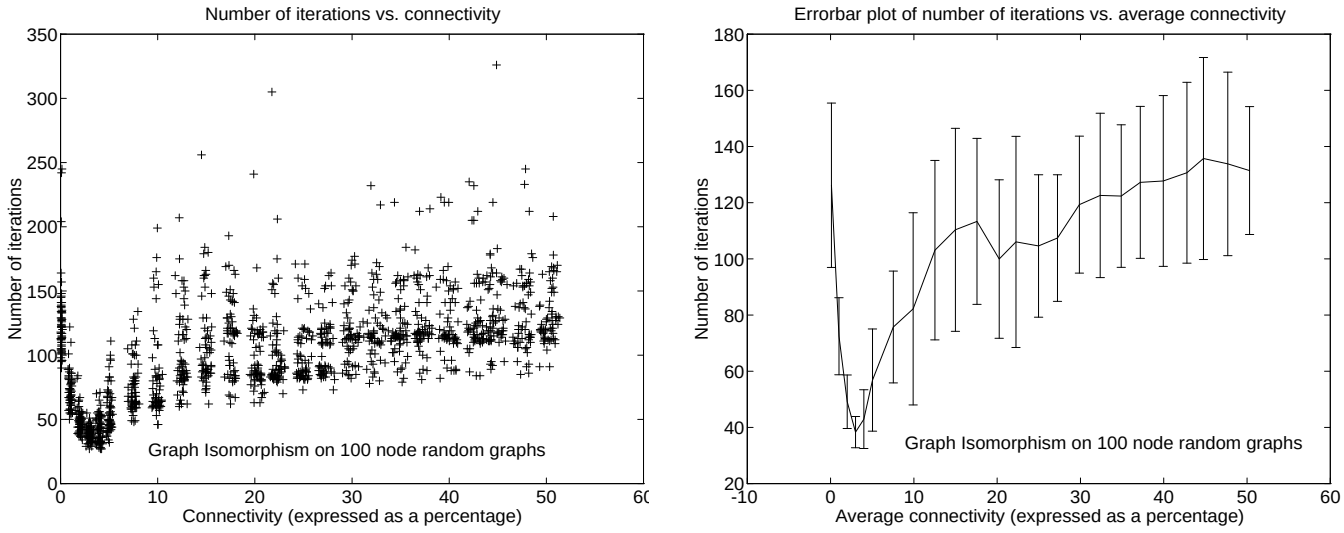


Figure 5:

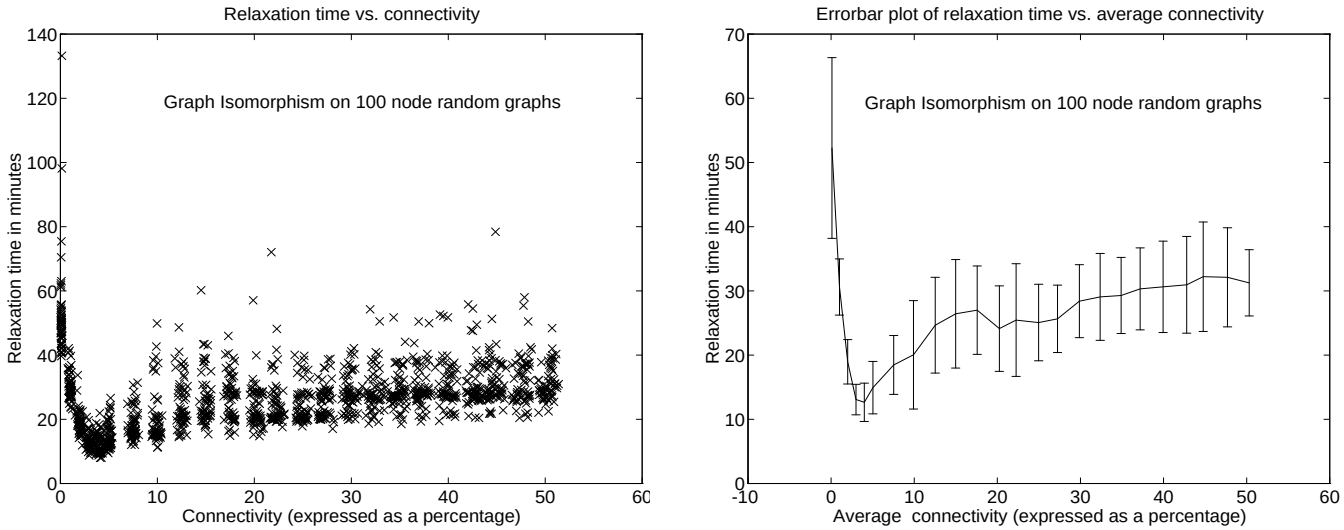


Figure 6:

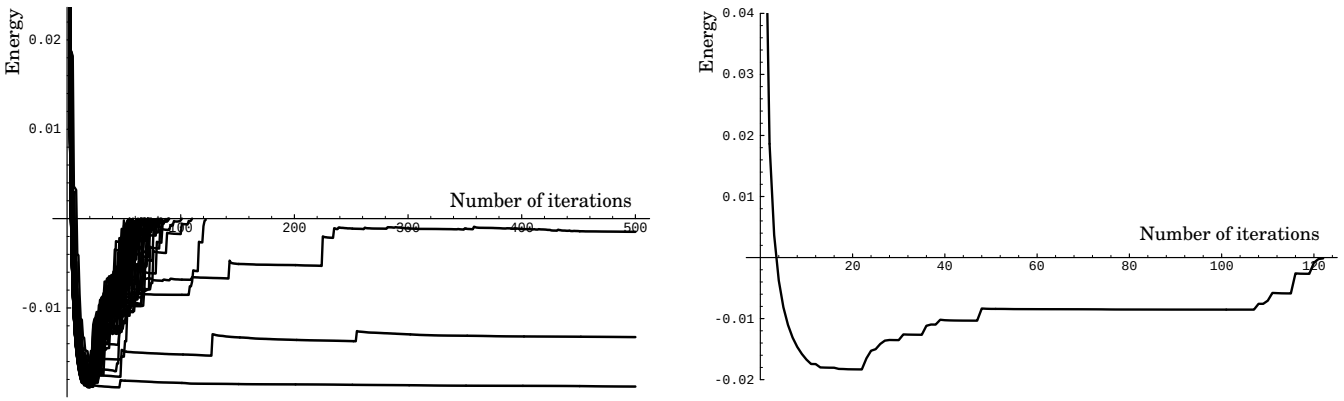


Figure 7:

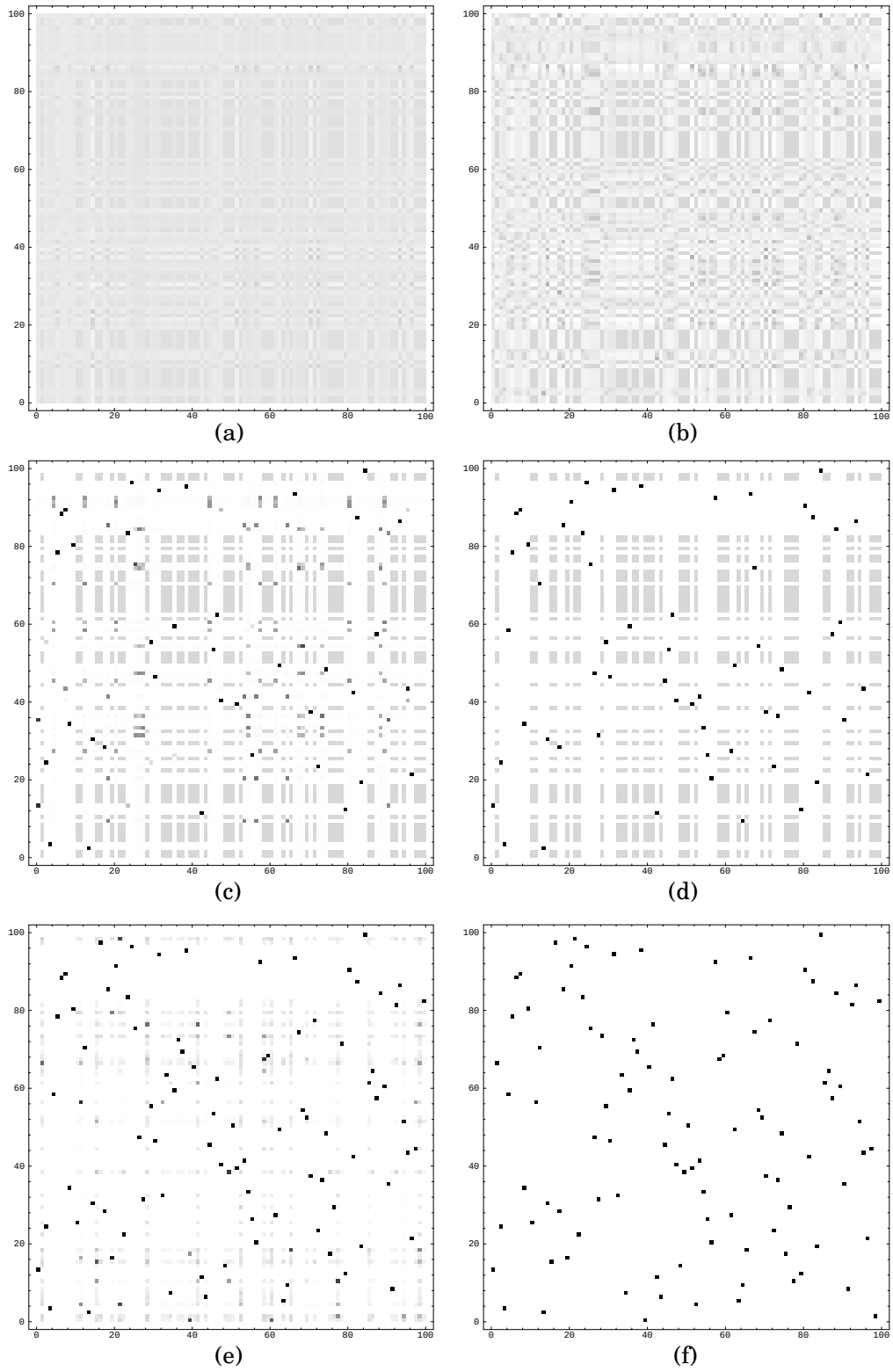
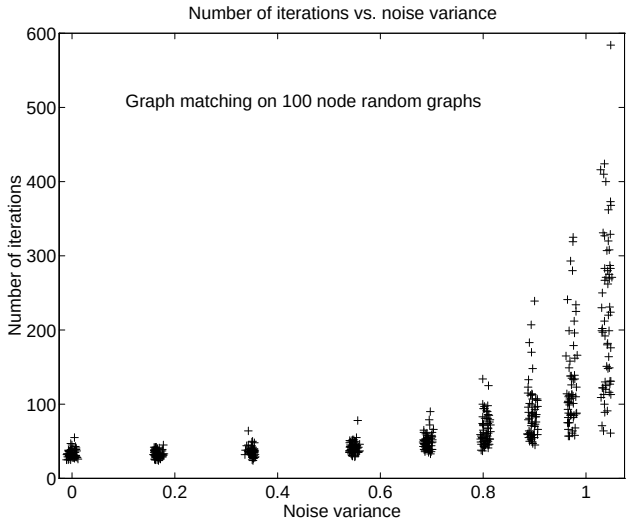


Figure 8:


Number of iterations vs. noise variance
Graph matching on 100 node random graphs
Number of iterations
Noise variance
0
100
200
300
400
500
600
0.2
0.4
0.6
0.8
1

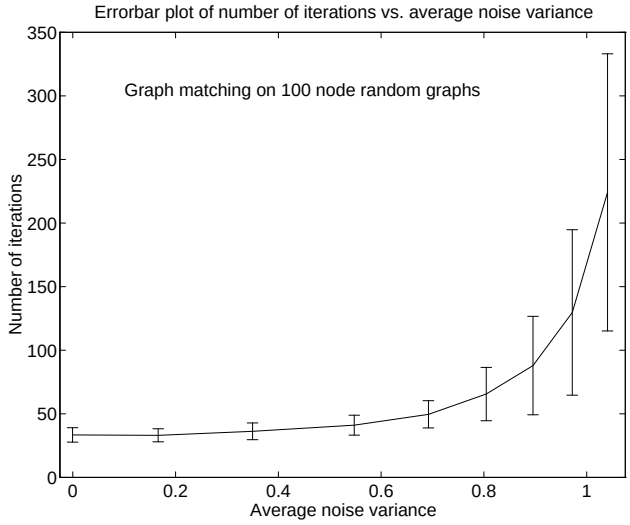

Errorbar plot of number of iterations vs. average noise variance
Graph matching on 100 node random graphs
Number of iterations
Average noise variance
0
50
100
150
200
250
300
350
0.2
0.4
0.6
0.8
1

Figure 9:

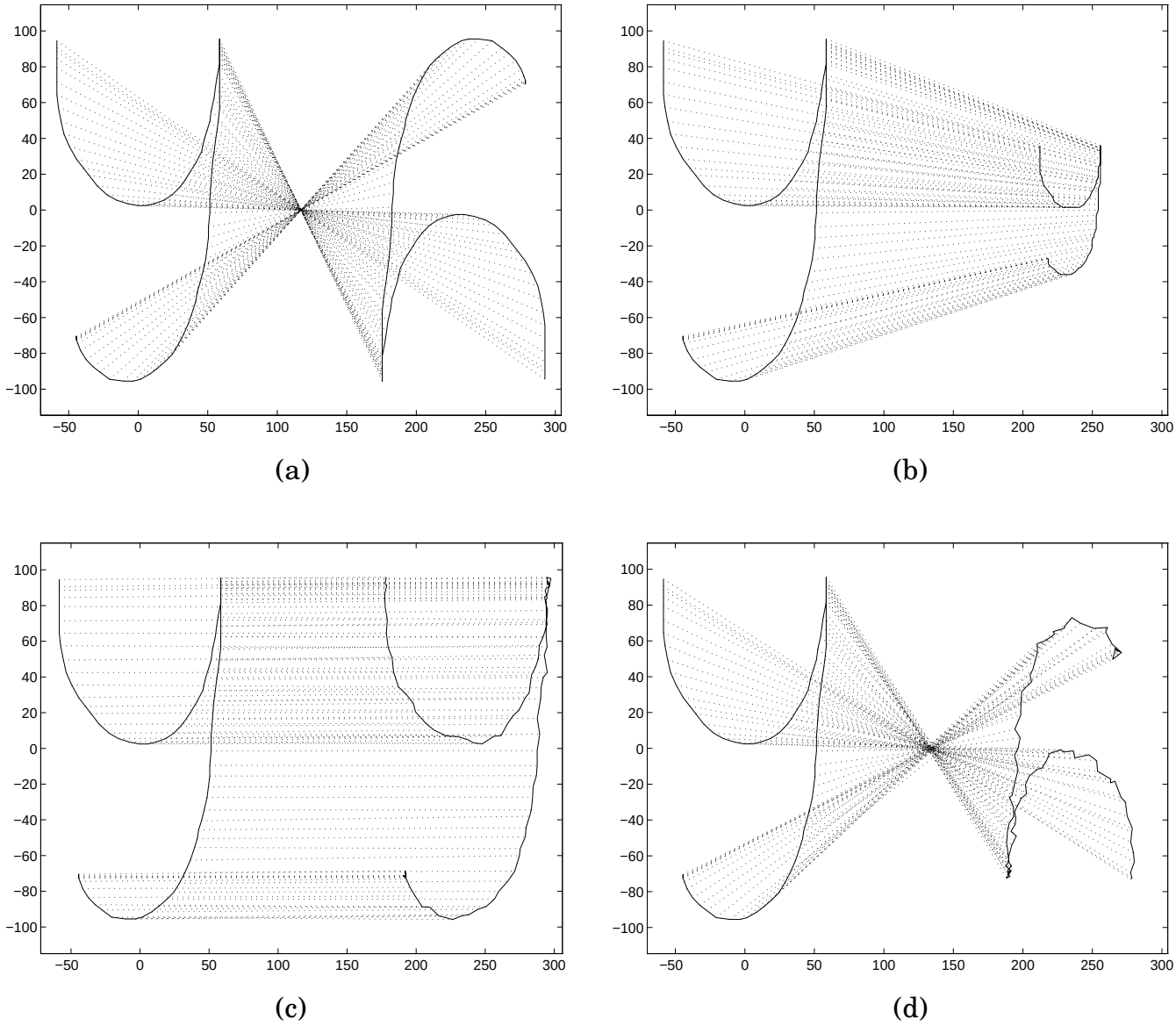


Figure 10:

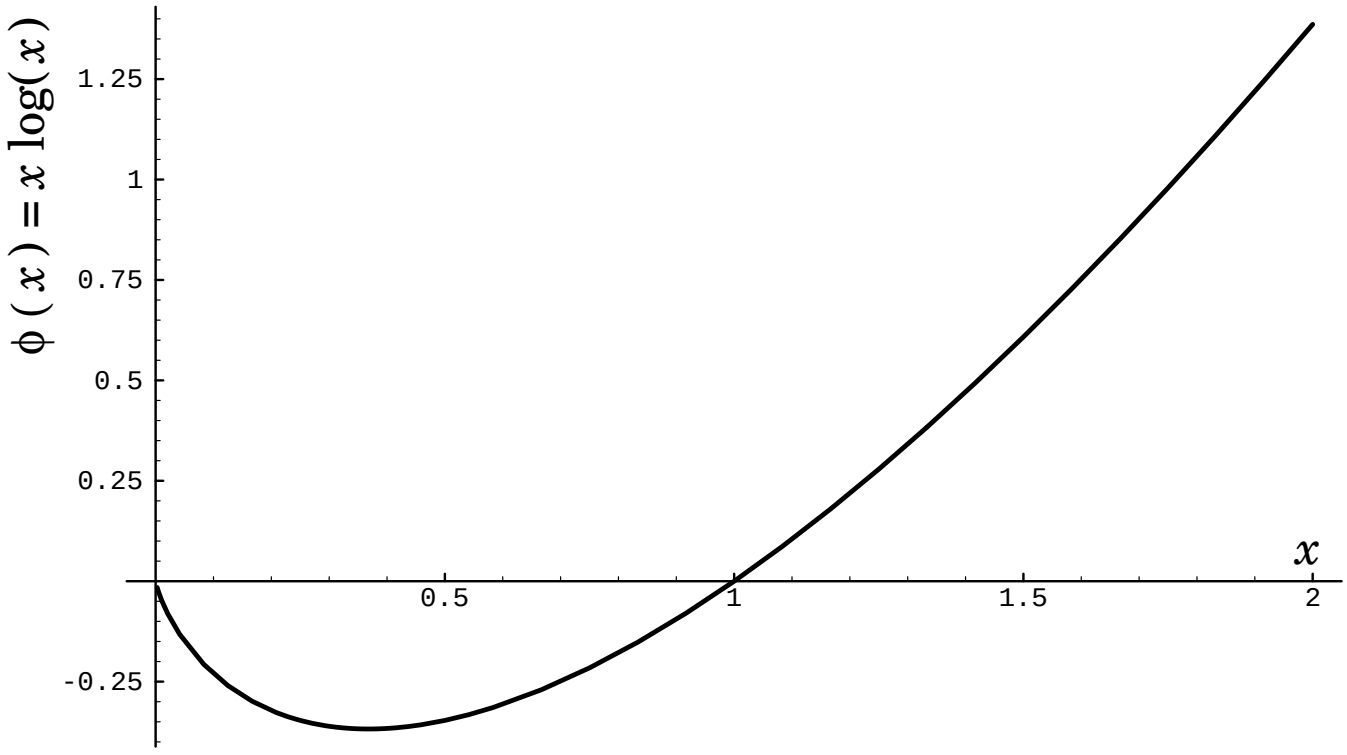


Figure 11: