Matching Hierarchical Structures Using Association Graphs

Marcello Pelillo, *Member*, *IEEE*, Kaleem Siddiqi, *Member*, *IEEE*, and Steven W. Zucker, *Fellow*, *IEEE*

Abstract—It is well-known that the problem of matching two relational structures can be posed as an equivalent problem of finding a maximal clique in a (derived) "association graph." However, it is not clear how to apply this approach to computer vision problems where the graphs are hierarchically organized, i.e., are trees, since maximal cliques are not constrained to preserve the partial order. Here, we provide a solution to the problem of matching two trees by constructing the association graph using the graph-theoretic concept of connectivity. We prove that, in the new formulation, there is a one-to-one correspondence between maximal cliques and maximal subtree isomorphisms. This allows us to cast the tree matching problem as an indefinite quadratic program using the Motzkin-Straus theorem, and we use "replicator" dynamical systems developed in theoretical biology to solve it. Such continuous solutions to discrete problems are attractive because they can motivate analog and biological implementations. The framework is also extended to the matching of attributed trees by using weighted association graphs. We illustrate the power of the approach by matching articulated and deformed shapes described by shock trees.

Index Terms—Maximal subtree isomorphisms, association graphs, maximal cliques, replicator dynamical systems, shock trees, shape recognition.

1 Introduction

The relationships between discrete and continuous mathematics have always been a subject of intensive study since the discovery of the irrationals by the Pythagorean school. Apart from the underlying philosophical implications, the interaction between the two domains can provide new insights into old problems and often allows techniques from one side to be profitably imported into the other. Entire branches of modern mathematics have been created with the specific motivation of exploring such connections, examples of which are singularity theory, combinatorial topology, and spectral graph theory. In more recent years, with the introduction of the ellipsoid and the interior point methods for linear programming, there has also been a tremendous interest in computer science and operations research in solving combinatorial optimization problems using continuous methods [21], [44].

Following the seminal works of Hopfield and Tank [24], and Durbin and Willshaw [14], the neural network community also became interested in using continuous approaches for combinatorial optimization. The basic idea consists of deriving a continuous "energy" function whose minimizers are in correspondence with the solutions of the discrete problem, and then minimizing it using continuous- or discrete-time dynamical systems, typically embedded in a parallel network of locally interacting processing elements. Such continuous solutions to discrete problems are attractive not only because they offer the advantage of biological plausibility, but also because they can motivate parallel, analog VLSI implementations. Examples of problems attacked within this framework include the traveling salesman problem [14], [24], graph bipartitioning [16], the maximum clique problem [26], [46], the linear assignment problem [29], the knapsack problem [43], and the graph/subgraph isomorphism problems [20], [53].

Thus far, the focus has been on "flat" problems in the sense that there is no partial ordering imposed on the data. In many practical problems, however, data are organized in a hierarchical manner, i.e., are *trees*, and the problem of *matching* such representations is of interest for pattern recognition. Applications in domains like computer vision [32], [36], [55], [57], [62], [66], molecular biology [58], and natural language processing [41] abound, and many traditional, discrete algorithms have been developed [31], [34], [37], [54], [59]. On the other hand, no attempt has yet been made to approach such problems within a continuous framework, using analog continuous-time dynamics. The main difficulty is that it is not clear how to map the hierarchy embedded in the representations onto a "flat" optimization network.

The matching of relational structures is a related (but different) problem which has also received considerable attention in computer vision and pattern recognition because of its applications in such problems as object recognition, motion, and stereo analysis, etc. [2]. A classical

- *M. Pelillo is with the Dipartimento di Informatica, Università Ca' Foscari di Venezia, Via Torino 155, 30172 Venezia Mestre, Italy. E-mail: pelillo@dsi.unive.it.*
- *K. Siddiqi is with the School of Computer Science and Center for Intelligent Machines, McGill University, 3480 University Street, Montreal, PQ, H3A 2A7, Canada. E-mail: siddiqi@cim.mcgill.ca.*
- *S.W. Zucker is with the Departments of Computer Science and Electrical Engineering and the Center for Computational Vision and Control, Yale University, PO Box 208285, New Haven, CT 06520-8285. E-mail: xucker-steven@cs.yale.edu.*

Manuscript received 15 Dec. 1998; revised 20 July 1999.
Recommended for acceptance by K. Bowyer.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 108453.

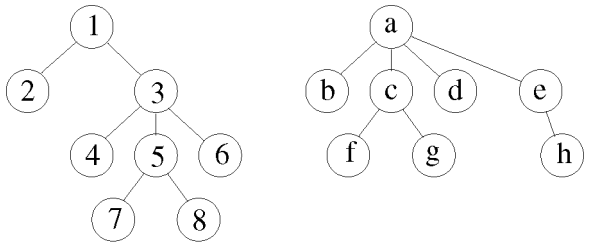


Fig. 1. An example of matching two trees. In the standard formulation of the association graph, the maximum cliques do not preserve the hierarchical structure of the two trees (see text for details).

solution to this problem consists of transforming it into the equivalent problem of finding a maximum clique in an auxiliary graph structure, known as the *association graph* [2], [3]. The idea goes back to Ambler et al. [1] and has since been successfully employed in a variety of different tasks, e.g., [6], [25], [42], [51], [52], [65]. This framework is attractive because it casts relational structure matching as a pure graph-theoretic problem for which a solid theory and powerful algorithms have been developed. Although the maximum clique problem is known to be NP-complete [17], powerful heuristics have been developed which efficiently find good approximate solutions [9] and there exist many classes of graphs for which the problem is solvable in polynomial-time [21], [9].

Since, in the standard association graph formulation, the solutions are not constrained to preserve the required partial order, it is not clear how to apply the framework for matching hierarchical structures. The extension of association graph techniques to tree matching problems is therefore of considerable interest. To illustrate the difficulties with the standard formulation, consider the problem of finding the largest subtree in the left tree of Fig. 1, which is isomorphic to a subtree in the right tree. Up to permutations, the correct solution is clearly given by 3 $\rightarrow$ a, 4 $\rightarrow$ b, 5 $\rightarrow$ c, 6 $\rightarrow$ d, 7 $\rightarrow$ f, and 8 $\rightarrow$ g. In other words, the subtree rooted at node 3 is matched against that rooted at node a in the tree on the right. However, using the standard association graph formulation (cf. [2, p. 366]), it is easily verified that the solutions induced by the maximum cliques correspond (up to permutations) to the following: 2 $\rightarrow$ h, 3 $\rightarrow$ a, 4 $\rightarrow$ b, 5 $\rightarrow$ c, 6 $\rightarrow$ d, 7 $\rightarrow$ f, and 8 $\rightarrow$ g, which, while perfectly in accordance with the usual subgraph isomorphism constraints, *do* violate the requirement that the matched subgraphs be trees (note, in fact, that nodes 2 and h are isolated from the rest of the matched subtrees).

In this paper, we introduce a solution to this problem by providing a novel way of deriving an association graph from two (rooted) trees, based on the graph-theoretic notions of connectivity and the distance matrix. We prove that, in the new formulation, there is a one-to-one correspondence between maximal (maximum) cliques in the derived association graph and maximal (maximum) subtree isomorphisms. As an obvious corollary, the computational complexity of finding a maximum clique in such graphs is therefore the same as that of the subtree isomorphism problem, which is known to be polynomial in the number of nodes [17]. This formulation allows us to map the hierarchical information contained in the trees onto a flat structure and this turns out to be the key to the proposed framework.

Following the development in [48], [49], we use the Motzkin-Straus theorem [40] to formulate the maximum clique problem on the association graph as a (continuous) quadratic program whose solutions are in one-to-one correspondence with the solutions of the original tree matching problem. To solve it, we employ *replicator equations*, a class of continuous- and discrete-time dynamical systems developed and studied in various branches of mathematical biology [23], [63], which are also closely related to parallel relaxation labeling networks [56]. In addition, we extend the framework to handle the matching of attributed trees by casting the problem as that of finding a maximum weight clique in a weighted association graph. A recent generalization of the Motzkin-Straus theorem applies [19], allowing the use of the same replicator dynamics as in the unweighted case. We illustrate the power of the proposed approach via several examples of matching articulated and deformed shapes described by *shock* trees [62].

2 Tree Isomorphism and Maximal Cliques

2.1 Notations and Definitions

Before going into the details of the proposed framework, we need to introduce some graph-theoretical notations and definitions. More details can be found in standard textbooks of graph theory, such as [22]. Let $G = (V, E)$ be a graph, where V is the set of nodes and E is the set of (undirected) edges. The *order* of G is the number of nodes in V, while its *size* is the number of edges. Two nodes $u, v \in V$ are said to be *adjacent* (denoted $u \sim v$) if they are connected by an edge. A *path* is any sequence of distinct nodes $u_0 u_1 \dots u_n$ such that, for all $i = 1 \dots n$, $u_{i-1} \sim u_i$; in this case, the *length* of the path is n. If $u_0 = u_n$, the path is called a *cycle*. A graph is said to be *connected* if any pair of nodes is joined by a path. The *distance* between two nodes u and v, denoted by $d(u, v)$, is the length of the shortest path joining them (by convention, $d(u, v) = \infty$ if there is no such path). Given a subset of nodes $C \subseteq V$, the *induced subgraph* $G[C]$ is the graph having C as its node set and two nodes are adjacent in $G[C]$ if and only if they are adjacent in G.

A connected graph with no cycles is called a *tree*. A *rooted* tree is one which has a distinguished node, called the *root*. The *level* of a node u in a rooted tree, denoted by $\mathrm{lev}(u)$, is the length of the path connecting the root to u. Note that there is an obvious equivalence between rooted trees and directed trees, where the edges are assumed to be oriented. We shall therefore use the same terminology typically used for directed trees to define the relation between two adjacent nodes. In particular, if $u \sim v$ and $\mathrm{lev}(v) - \mathrm{lev}(u) = +1$, we say that u is the *parent* of v and, conversely, v is a *child* of u. Trees have a number of interesting properties. One which turns out to be very useful for our characterization is that in a tree any two nodes are connected by a *unique* path.

2.2 Deriving the Association Graph

Let $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$ be two rooted trees. Any bijection $\phi : H_1 \rightarrow H_2$, with $H_1 \subseteq V_1$ and $H_2 \subseteq V_2$, is called a

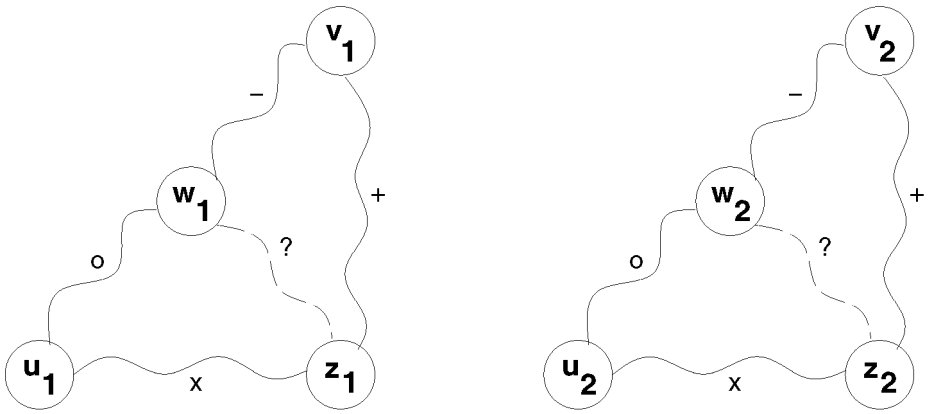


Fig. 2. An illustration of the hypotheses of Lemma 1. Each curved line represents a path between two nodes; when two paths are labeled by the same symbol, the corresponding path-strings are assumed to be the same. The lemma states that $\mathrm{str}(w_1, z_1) = \mathrm{str}(w_2, z_2)$.

subtree isomorphism if it preserves the adjacency and hierarchical relationships between the nodes and, in addition, the subgraphs obtained when we restrict ourselves to H_1 and H_2, i.e., $T_1[H_1]$ and $T_2[H_2]$, are trees. The former condition amounts to stating that, given $u, v \in H_1$, we have $u \sim v$ if and only if $\phi(u) \sim \phi(v)$, and u is the parent of v if and only if $\phi(u)$ is the parent of $\phi(v)$. A subtree isomorphism is *maximal* if there is no other subtree isomorphism $\phi' : H_1' \to H_2'$ with H_1 a strict subset of H_1', and *maximum* if H_1 has largest cardinality. The maximal (maximum) subtree isomorphism problem is to find a maximal (maximum) subtree isomorphism between two rooted trees.

We now introduce the notion of a *path-string*, which will be central to the subsequent development.

Definition 1. *Let u and v be two distinct nodes of a rooted tree T, and let $u = x_0 x_1 \ldots x_n = v$ be the (unique) path joining them. The* path-string *of u and v, denoted by $\mathrm{str}(u, v)$, is the string $s_1 s_2 \ldots s_n$ on the alphabet $\{-1, +1\}$ where, for all $i = 1 \ldots n$, $s_i = \mathrm{lev}(x_i) - \mathrm{lev}(x_{i-1})$. By convention, when $u = v$ we define $\mathrm{str}(u, v) = \varepsilon$, where ε is the null string (i.e., the string having zero length).*

The path-string concept has a very intuitive meaning. Suppose that you stand on a particular node in a rooted tree and want to move to another adjacent node. Because of the orientation induced by the root, only two types of moves can be done, i.e., going down to one of the children (if one exists) or going up to the parent (if you are not on the root). Let us assign to the first move the label $+1$ and, to the second, the label -1. Now, suppose that you want to move from node u to v, following the unique path joining them. Then, the path-string of u and v is simply the string of elementary moves required to reach v, starting from u. It may be thought of as the degree of relationship between two relatives in a "family" tree. As an illustrative example, referring to Fig. 1, we have $\mathrm{str}(2, 8) = -1 + 1 + 1 + 1$. Note that if $\mathrm{str}(u, v) = s_1 \ldots s_{n-1} s_n$, then $\mathrm{str}(v, u) = \overline{s}_n \overline{s}_{n-1} \ldots \overline{s}_1$, where $\overline{s}_i = -1$ if $s_i = +1$ and $\overline{s}_i = +1$ otherwise ($i = 1 \ldots n$).

Definition 2. *The* tree association graph *(TAG) of two rooted trees $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$ is the graph $G = (V, E)$ where*

$$V = V_1 \times V_2$$

and, for any two nodes (u, w) and (v, z) in V, we have

$$(u, w) \sim (v, z) \Leftrightarrow \mathrm{str}(u, v) = \mathrm{str}(w, z).$$

Intuitively, two nodes (u, w) and (v, z) are adjacent in the TAG if and only if the relationship between u and v in T_1 is the same as that between w and z in T_2. Note that this definition of the association graph is stronger than the standard one used for matching arbitrary relational structures [2], [3]. A subset of vertices of G is said to be a *clique* if all its nodes are mutually adjacent. A *maximal* clique is one which is not contained in any larger clique, while a *maximum* clique is a clique having largest cardinality. The maximum clique problem is to find a maximum clique of G.

Our main goal in this section is to establish a one-to-one correspondence between maximal cliques in the TAG and maximal subtree isomorphisms. To this end, we need the following result.

Lemma 1. *Let $u_1, v_1, w_1, z_1 \in V_1$ and $u_2, v_2, w_2, z_2 \in V_2$ be distinct nodes of rooted trees $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$, and suppose that the following conditions hold (see Fig. 2):*

1. *w_1 is on the $u_1 v_1$-path and w_2 is on the $u_2 v_2$-path*
2. $\mathrm{str}(u_1, w_1) = \mathrm{str}(u_2, w_2)$
3. $\mathrm{str}(w_1, v_1) = \mathrm{str}(w_2, v_2)$
4. $\mathrm{str}(u_1, z_1) = \mathrm{str}(u_2, z_2)$
5. $\mathrm{str}(v_1, z_1) = \mathrm{str}(v_2, z_2)$

Then, $\mathrm{str}(w_1, z_1) = \mathrm{str}(w_2, z_2)$.

Proof. See the Appendix. □

The following theorem, which is the basis of the work reported here, establishes a one-to-one correspondence between the maximum subtree isomorphism problem and the maximum clique problem.

Theorem 1. *Any maximal (maximum) subtree isomorphism between two rooted trees induces a maximal (maximum) clique in the corresponding TAG, and vice versa.*

Proof. Let $\phi : H_1 \to H_2$ be a maximal subtree isomorphism between rooted trees T_1 and T_2, and let $G = (V, E)$

denote the corresponding tree association graph. Let $C_\phi \subseteq V$ be defined as:

$$C_\phi = \{(u, \phi(u)) : u \in H_1\}.$$

From the definition of a subtree isomorphism, it follows that ϕ maps the path between any two nodes $u, v \in H_1$ onto the path joining $\phi(u)$ and $\phi(v)$. This clearly implies that $\mathrm{str}(u, v) = \mathrm{str}(\phi(u), \phi(v))$ for all $u \in H_1$ and, therefore, C_ϕ is a clique. Trivially, C_ϕ is a maximal clique because ϕ is maximal. This proves the first part of the theorem.

Suppose now that $C = \{(u_1, w_1), \cdots, (u_n, w_n)\}$ is a maximal clique of G, and let $H_1 = \{u_1, \cdots, u_n\} \subseteq V_1$ and $H_2 = \{w_1, \cdots, w_n\} \subseteq V_2$. Define $\phi : H_1 \to H_2$ as $\phi(u_i) = w_i$, for all $i = 1 \ldots n$. From the definition of a tree association graph and the hypothesis that C is a clique, it is simple to see that ϕ is a one-to-one and onto correspondence between H_1 and H_2, which trivially preserves both the adjacency and the hierarchical relationships between nodes. The fact that ϕ is a maximal isomorphism is a straightforward consequence of the maximality of C.

To conclude the proof we have to show that the subgraphs that we obtain when we restrict ourselves to H_1 and H_2, i.e., $T_1[H_1]$ and $T_2[H_2]$, are trees and this is equivalent to showing that they are connected. Suppose, by contradiction, that this is not the case and let $u_i, u_j \in H_1$ be two nodes which are not joined by a path in $T_1[H_1]$. Since both u_i and u_j are nodes of T_1, however, there must exist a path $u_i = x_0 x_1 \ldots x_m = u_j$ joining them in T_1. Let $x^* = x_k$, for some $k = 1 \ldots m$, be a node on this path which is not in H_1. Moreover, let $y^* = y_k$ be the kth node on the path $w_i = y_0 y_1 \ldots y_m = w_j$ which joins w_i and w_j in T_2 (remember that $\mathrm{str}(u_i, u_j) = \mathrm{str}(w_i, w_j)$ and, hence, $d(w_i, w_j) = m$). We now show that the set $\{(x^*, y^*)\} \cup C \subseteq V$ is a clique. To this end, let $(u, w) \in C$. Since (u_i, w_i) and (u_j, w_j) are also nodes in C, we have $\mathrm{str}(u_i, u) = \mathrm{str}(w_i, w)$, and $\mathrm{str}(u_j, u) = \mathrm{str}(w_j, w)$. Furthermore, we have that x^* and y^* are on the $u_i u_j$- and $w_i w_j$-paths, respectively, and, clearly, $\mathrm{str}(u_i, x^*) = \mathrm{str}(w_i, y^*)$ and $\mathrm{str}(x^*, u_j) = \mathrm{str}(y^*, w_j)$. Therefore, all the hypotheses of Lemma 1 are satisfied and this implies that $\mathrm{str}(x^*, u) = \mathrm{str}(y^*, w)$, which amounts to stating that node (x^*, y^*) is adjacent to (u, w), for all $(u, w) \in C$. This means that $\{(x^*, y^*)\} \cup C$ is a clique, thereby contradicting the hypothesis that C is a maximal clique and proving the second part of the theorem.

The "maximum" part of the statement is proven similarly. □

The next proposition provides us with a straightforward criterion to construct the TAG.

Proposition 1. *Let $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$ be two rooted trees, $u, v \in V_1$, and $w, z \in V_2$. Then, $\mathrm{str}(u, v) = \mathrm{str}(w, z)$ if and only if the following two conditions hold:*

1. $d(u, v) = d(w, z)$
2. $\mathrm{lev}(u) - \mathrm{lev}(v) = \mathrm{lev}(w) - \mathrm{lev}(z)$.

Proof. The proposition is a straightforward consequence of the observation that, given any two nodes u and v in a tree, with $\mathrm{str}(u, v) = s_1 s_2 \ldots s_n$, we have $\mathrm{lev}(u) - \mathrm{lev}(v) = \sum_i s_i$, and the fact that $s_i = +1$ implies $s_j = +1$ for all $j \geq i$. □

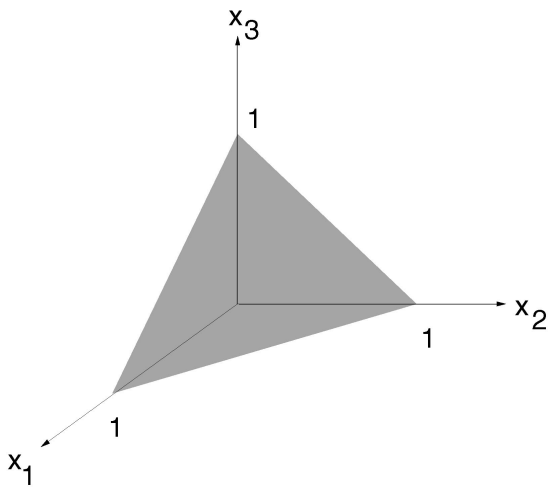


Fig. 3. The simplex S_3.

This property allows us to efficiently derive the TAG by using a classical representation for graphs, i.e., the so-called *distance matrix* (see, e.g., [22]) which, for an arbitrary graph $G = (V, E)$ of order n, is the $n \times n$ matrix $D = (d_{ij})$ where $d_{ij} = d(u_i, u_j)$, the distance between nodes u_i and u_j.

3 A Continuous Formulation of the Maximum Clique Problem

We now exploit the interplay between discrete and continuous mathematics, alluded to in Section 1. Let $G = (V, E)$ be an arbitrary graph of order n and let S_n denote the standard simplex of $\mathbb{R}^n$ (see Fig. 3):

$$S_n = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{e}'\mathbf{x} = 1 \text{ and } x_i \geq 0, i = 1 \ldots n\},$$

where $\mathbf{e}$ is the vector whose components equal 1 and a prime denotes transposition. Given a subset of vertices C of G, we will denote by $\mathbf{x}^c$ its *characteristic vector* which is the point in S_n defined as

$$x_i^c = \begin{cases} 1/|C|, & \text{if } u_i \in C \\ 0, & \text{otherwise,} \end{cases}$$

where $|C|$ denotes the cardinality of C.

Now, consider the following quadratic function

$$f(\mathbf{x}) = \mathbf{x}' A \mathbf{x}, \tag{1}$$

where $A = (a_{ij})$ is the adjacency matrix of G, i.e., the $n \times n$ symmetric matrix defined as

$$a_{ij} = \begin{cases} 1, & \text{if } u_i \sim u_j \\ 0, & \text{otherwise.} \end{cases}$$

A point $\mathbf{x}^* \in S_n$ is said to be a *global* maximizer of f in S_n if $f(\mathbf{x}^*) \geq f(\mathbf{x})$, for all $\mathbf{x} \in S_n$. It is said to be a *local* maximizer if there exists an $\epsilon > 0$ such that $f(\mathbf{x}^*) \geq f(\mathbf{x})$ for all $\mathbf{x} \in S_n$ whose distance from $\mathbf{x}^*$ is less than ϵ and if $f(\mathbf{x}^*) = f(\mathbf{x})$

implies $\mathbf{x}^* = \mathbf{x}$, then $\mathbf{x}^*$ is said to be a *strict* local maximizer. Note that $f(\mathbf{x}) \leq 1$ for all $\mathbf{x} \in S_n$.

The Motzkin-Straus theorem [40] establishes a remarkable connection between global (local) maximizers of the function f in S_n and maximum (maximal) cliques of G. Specifically, it states that a subset of vertices C of a graph G is a maximum clique if and only if its characteristic vector $\mathbf{x}^c$ is a global maximizer of f on S_n. A similar relationship holds between (strict) local maximizers and maximal cliques [19], [50]. This result has an intriguing computational significance in that it allows us to shift from the discrete to the continuous domain. Such a reformulation is attractive for several reasons: It suggests how to exploit the full arsenal of continuous optimization techniques, thereby leading to the development of new algorithms, and may also reveal unexpected theoretical properties. Additionally, continuous optimization methods are often described in terms of sets of differential equations and are, therefore, potentially implementable in analog circuitry. The Motzkin-Straus theorem has served as the basis of several clique-finding procedures [10], [18], [45], [46] and has also been used to determine theoretical bounds on the cardinality of the maximum clique [45], [64].

One drawback associated with the original Motzkin-Straus formulation relates to the existence of spurious solutions, i.e., maximizers of f which are not in the form of characteristic vectors. This was observed empirically by Pardalos and Phillips [45] and more recently formalized by Pelillo and Jagota [50]. In principle, spurious solutions represent a problem since, while providing information about the cardinality of the maximum clique, they do not allow us to easily extract its vertices. Fortunately, there is a solution to this problem which has recently been introduced and studied by Bomze [7]. Consider the following regularized version of f:

$$\hat{f}(\mathbf{x}) = \mathbf{x}'A\mathbf{x} + \frac{1}{2}\mathbf{x}'\mathbf{x}, \tag{2}$$

which is obtained from (1) by substituting the adjacency matrix A of G with

$$\hat{A} = A + \frac{1}{2}I_n,$$

where I_n is the $n \times n$ identity matrix. The following is the spurious-free counterpart of the original Motzkin-Straus theorem (see [7] for a proof).

Theorem 2. *Let C be a subset of vertices of a graph G, and let $\mathbf{x}^c$ be its characteristic vector. Then the following statements hold:*

1. *C is a maximum clique of G if and only if $\mathbf{x}^c$ is a global maximizer of the function $\hat{f}$ in S_n. In this case, $|C| = 1/2(1 - f(\mathbf{x}^c))$.*
2. *C is a maximal clique of G if and only if $\mathbf{x}^c$ is a local maximizer of $\hat{f}$ in S_n.*
3. *All local (and, hence, global) maximizers of $\hat{f}$ in S_n are strict.*

Unlike the original Motzkin-Straus formulation, the previous result guarantees that *all* maximizers of $\hat{f}$ on S_n are strict, and are characteristic vectors of maximal/maximum cliques in the graph. In a formal sense, therefore, a one-to-one correspondence exists between maximal cliques and local maximizers of $\hat{f}$ in S_n on the one hand and maximum cliques and global maximizers on the other hand.

4 Replicator Equations and Tree Matching

We now turn our attention to a class of dynamical systems that we use for solving our quadratic optimization problem. Let W be a nonnegative real-valued $n \times n$ matrix and consider the following dynamical system:

$$\dot{x}_i(t) = x_i(t)\big[(W\mathbf{x}(t))_i - \mathbf{x}(t)'W\mathbf{x}(t)\big], \quad i = 1 \ldots n, \tag{3}$$

where a dot signifies derivative w.r.t. time t, and its discrete-time counterpart

$$x_i(t+1) = x_i(t)\frac{(W\mathbf{x}(t))_i}{\mathbf{x}(t)'W\mathbf{x}(t)}, \quad i = 1 \ldots n. \tag{4}$$

It is readily seen that the simplex S_n is invariant under these dynamics, which means that every trajectory starting in S_n will remain in S_n for all future times. Moreover, it turns out that their *stationary points*, i.e., the points satisfying $\dot{x}_i(t) = 0$ for (3) or $x_i(t+1) = x_i(t)$ for (4), coincide and are the solutions of the equations:

$$x_i[(W\mathbf{x})_i - \mathbf{x}'W\mathbf{x}] = 0, \quad i = 1 \ldots n.$$

A stationary point $\mathbf{x}$ is said to be *asymptotically stable* if every solution to (3) or (4) which starts close enough to $\mathbf{x}$ converges to $\mathbf{x}$ as $t \to \infty$.

Both (3) and (4) are called *replicator equations* in theoretical biology since they are used to model evolution over time of relative frequencies of interacting, self-replicating entities [23]. The discrete-time dynamical equations turn out to be a special case of a general class of dynamical systems introduced by Baum and Eagon [5] in the context of the theory of Markov chains. They also represent an instance of the original Rosenfeld-Hummel-Zucker relaxation labeling algorithm [56], whose dynamical properties have recently been clarified [47] (specifically, it corresponds to the 1-object, n-label case).

We are now interested in the dynamical properties of replicator equations; it is these properties that will allow us to solve our original tree matching problem.

Theorem 3. *If $W = W'$, then the function $\mathbf{x}(t)'W\mathbf{x}(t)$ is strictly increasing with increasing t along any nonstationary trajectory $\mathbf{x}(t)$ under both continuous-time (3) and discrete-time (4) replicator dynamics. Furthermore, any such trajectory converges to a stationary point. Finally, a vector $\mathbf{x} \in S_n$ is asymptotically stable under (3) and (4) if and only if $\mathbf{x}$ is a strict local maximizer of $\mathbf{x}'W\mathbf{x}$ on S_n.*

The previous result is known in mathematical biology as the fundamental theorem of natural selection [13], [23], [63] and, in its original form, traces back to Fisher [15]. As far as the discrete-time model is concerned, it can be regarded as a straightforward implication of the more general Baum-Eagon theorem [5]. The fact that all trajectories of the replicator dynamics converge to a stationary point has been proven more recently [33], [35].

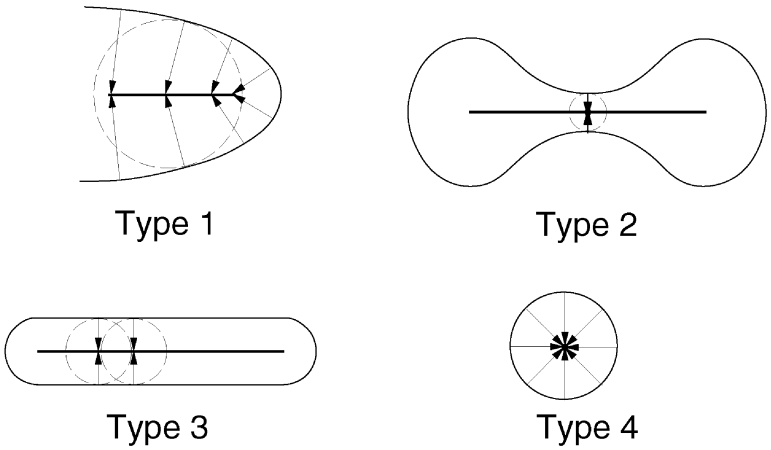


Fig. 4. A coloring of shocks into four types [28]. A 1-shock derives from a *protrusion* and traces out a curve segment of adjacent 1-shocks, along which the radius function varies monotonically. A 2-shock arises at a *neck*, where the radius function attains a strict local minimum, and is immediately followed by two 1-shocks flowing away from it in opposite directions. 3-shocks correspond to an annihilation into a curve segment due to a *bend*, along which the radius function is constant, and a 4-shock is an annihilation into a point or a *seed*, where the radius function attains a strict local maximum. The loci of these shocks gives Blum's medial axis.

In light of their dynamical properties, replicator equations naturally suggest themselves as a simple heuristic for solving the maximal subtree isomorphism problem. Let $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$ be two rooted trees and let A denote the N-node adjacency matrix of the corresponding TAG. By letting

$$W = A + \frac{1}{2} I_N,$$

where I_N is the $N \times N$ identity matrix, we know that the replicator dynamical systems (3) and (4), starting from an arbitrary initial state, will iteratively maximize the function $\hat{f}$ defined in (2) over S_N and will eventually converge with probability 1 to a strict local maximizer which, by virtue of Theorem 2, will then correspond to the characteristic vector of a maximal clique in the association graph. As stated in Theorem 1, this will in turn induce a maximal subtree isomorphism between T_1 and T_2.

Clearly, in theory, there is no guarantee that the converged solution will be a *global* maximizer of $\hat{f}$ and, therefore, that it will induce a *maximum* isomorphism between the two original trees. Previous experimental work done on the maximum clique problem [10], [46], and also the results presented in this paper, however, suggest that the basins of attraction of optimal or near-optimal solutions are quite large and, very frequently, the algorithm converges to one of them, despite its inherent inability to escape from local optima.

Since the process cannot leave the boundary of S_N, it is customary to start out the relaxation process from some interior point, a common choice being the barycenter of S_N, i.e., the vector $(\frac{1}{N}, \cdots, \frac{1}{N})'$. This prevents the search from being initially biased in favor of any particular solution.

5 An Example: Matching Shock Trees

We now illustrate our framework for matching hierarchical structures with numerical examples of shape matching. Because of the subtleties associated with generating random trees of relevance to applications in computer vision and pattern recognition, we use a class of example trees derived from a real system. Our representation for shape is based on an abstraction of the *shocks* (or singularities) of a curve evolution process, acting on a simple closed curve in the plane, into a shock tree. We begin by providing some background on the representation (for details see [28], [62]) and then present experimental results on matching shock trees. In Section 6, we extend the framework to incorporate attributes associated with shock tree nodes.

5.1 The Shock Tree

In [27], [28], the following evolution equation was proposed for visual shape analysis:

$$\begin{aligned} \mathcal{C}_t &= (1 + \alpha\kappa)\mathcal{N} \\ \mathcal{C}(p, 0) &= \mathcal{C}_0(p). \end{aligned} \tag{5}$$

Here, $\mathcal{C}(p,t)$ is the vector of curve coordinates, $\mathcal{N}(p,t)$ is the inward normal, p is the curve parameter, and t is the evolutionary time of the deformation. The constant $\alpha \geq 0$ controls the regularizing effects of curvature κ. When α is large, the equation becomes a geometric heat equation; When $\alpha = 0$, the equation is hyperbolic and *shocks* [30], or entropy-satisfying singularities, can form. In the latter case the locus of points through which the shocks migrate is related to Blum's grassfire transformation [12], [28], although significantly more information is available via a "coloring" of these positions. Four types can arise, according to the local variation of the radius function along the medial axis (Fig. 4). Intuitively, the radius function varies monotonically at a type 1, reaches a strict local minimum at a type 2, is constant at a type 3, and reaches a strict local maximum at a type 4. The classification of shock positions according to their colors and an enumeration of the possible local neighborhoods around each shock type is at the heart of the representation.

Shocks of the same type that form a connected component are grouped together to comprise the nodes of a *shock graph*, with the 1-shock groups separated at branch-points of the skeleton. Directed edges in the graph are

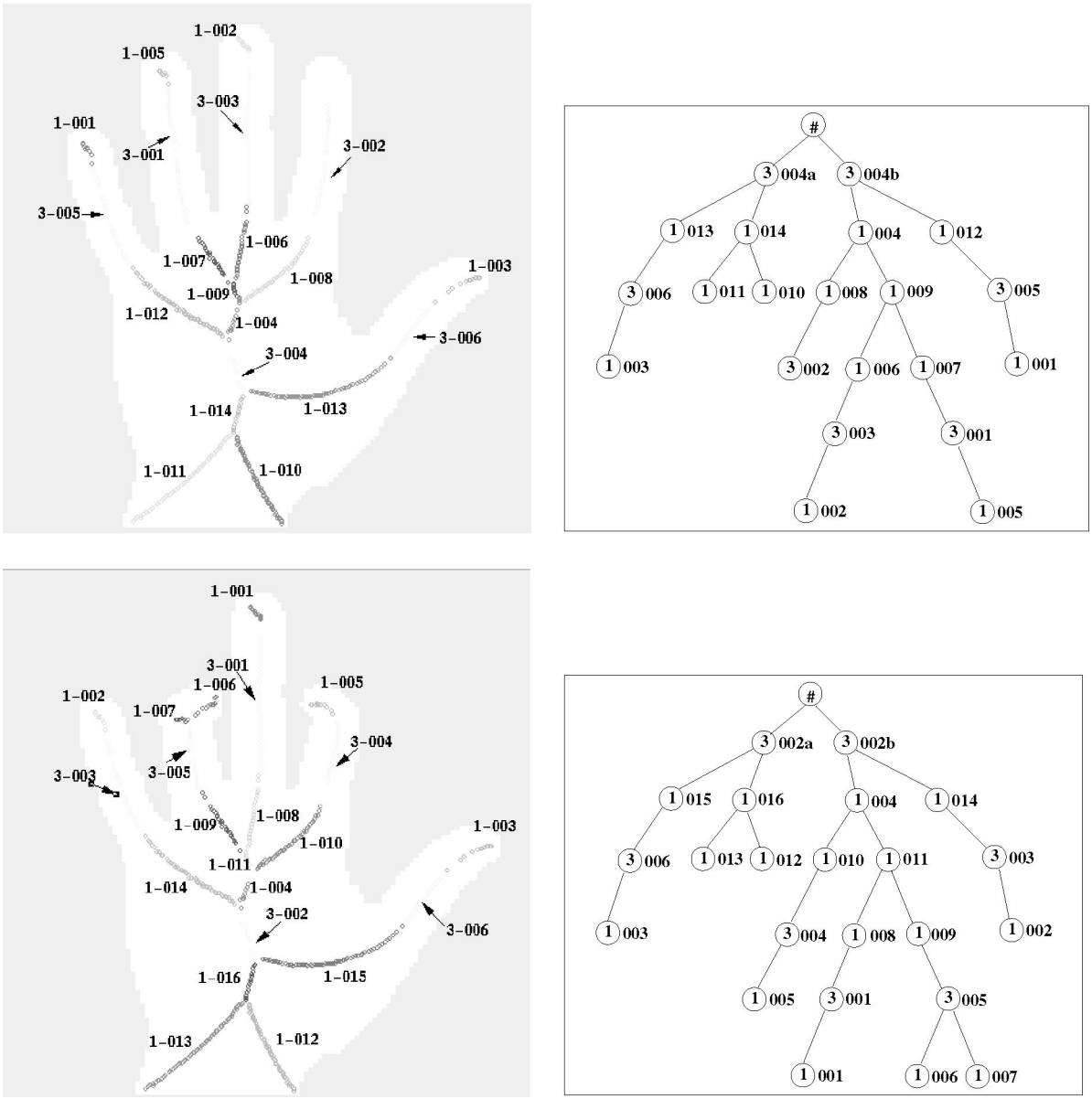


Fig. 5. Two illustrative examples of the shocks obtained from curve evolution (from [62]). Left: The notation associated with the locus of shock points is of the form shock_type-identifier. Right: The corresponding trees have the shock_type on each node, with the identifier adjacent. The last shocks to form during the curve evolution process appear under a root node labeled #.

placed between shock groups that touch one another such that each parent node contains no shocks that formed prior to any shocks in a child node. This corresponds to a "reversal" in time of the curve evolution process to obtain a hierarchy of connected components. The graph is rooted at a unique vertex #, the children of which are the last shock groups to form, e.g., the palms of the hand silhouettes in Fig. 5. (The letters "a" and "b" denote different sides of the same shock group). A key property of the shock graph is that its topological structure is highly constrained because the events that govern the birth, combination, and death of shock groups are completely characterized by a shock grammar, with a small number of rewrite rules [62]. In particular, each shock graph can be reduced to a unique rooted shock tree.

5.2 Experimental Results

We now illustrate the power of the hierarchical structure matching algorithm on shock trees. Whereas each node has geometric attributes, related to properties of the shocks in each group, for now we shall consider only the shock tree topologies. (We shall consider the geometry shortly.) We selected 25 silhouettes representing eight different object classes (Table 1, first column); the tool shapes were taken from the Rutgers Tools database. Each shape was then matched against *all* entries in the database. Fig. 6 shows the maximal subtree isomorphisms found by the algorithm for three examples. The top eight matches for each query shape, along with the associated scores, are shown in Table 1. The scores indicate the average of n/n_1 and n/n_2, where n is the size of the maximal clique found, and n_1 and n_2 are the number of nodes in each tree. We observed that, in all our

TABLE 1
A Tabulation of the Top Eight Topological Matches for Each Query

Query Shape	Top 8 Topological Matches							
	1	2	3	4	5	6	7	8
	 1.00	 1.00	 .916	 .900	 .825	 .771	 .750	 .750
	 1.00	 1.00	 .916	 .900	 .825	 .771	 .750	 .750
	 1.00	 .900	 .900	 .833	 .833	 .833	 .833	 .807
	 1.00	 .958	 .875	 .825	 .729	 .666	 .641	 .641
	 1.00	 .909	 .875	.826	.738	.738	.738	.711
	1.00	.958	.909	.859	.755	.668	.668	.609
	1.00	1.00	.966	.966	.900	.800	.784	.771
	1.00	.937	.937	.904	.875	.750	.731	.731
	1.00	1.00	.966	.966	.937	.928	.773	.771
	1.00	.928	.928	.900	.900	.875	.833	.825
	1.00	1.00	.966	.966	.937	.928	.773	.771
	1.00	1.00	.966	.966	.904	.900	.800	.784
	1.00	1.00	.801	.750	.733	.733	.720	.708
	1.00	1.00	.801	.750	.733	.733	.720	.708
	1.00	.807	.801	.801	.801	.801	.722	.721
	1.00	.900	.826	.755	.750	.729	.675	.675
	1.00	.900	.859	.825	.738	.675	.600	.600
	1.00	.977	.956	.800	.800	.785	.785	.773
	1.00	.977	.933	.784	.784	.772	.772	.759
	1.00	.956	.933	.771	.771	.760	.760	.746
	1.00	.916	.916	.833	.833	.833	.785	.772
	1.00	1.00	.833	.833	.801	.733	.733	.720
	1.00	1.00	.833	.833	.801	.733	.733	.720
	1.00	.687	.675	.675	.656	.612	.600	.600
	1.00	.693	.692	.692	.687	.673	.673	.661

The scores indicate the average of the fraction of nodes matched in each of the two trees (see text). Note that only the topology of the shock trees was used; the addition of geometric information permits finer discrimination (compare with Table 2).

experiments, the maximal cliques found were also maximum cliques. This is due to the property that global maximizers of the objective function typically have large basins of attraction, as also shown experimentally in [46], [49], [10]. The matching algorithm generally takes only two to three seconds to converge on a Sparc 10.

Note that, despite the fact that metric/label information associated with nodes in the shock trees was discounted altogether, all exemplars in the same class as the query shape are typically within the top five matches, illustrating the potential of a topological matching process for indexing into a database of shapes. Nevertheless, there exist a few matches which appear to be counterintuitive, e.g., matches [(Row 1, Column 3); (Row 2, Column 3); (Row 13, Column 4); (Row 14, Column 4); (Row 21, Column 2) and (Row 21, Column 3)]. These correspond to shapes with similar shock-tree topologies (hierarchies of parts), but drastically different shock geometries (part shapes). In the following section, we extend our framework to incorporate the latter geometric information contained in each shock sequence (the location, time of formation, speed, and direction of each shock) as attributes on the nodes. We show that this leads to better discrimination between shapes than that provided by shock tree topologies alone.

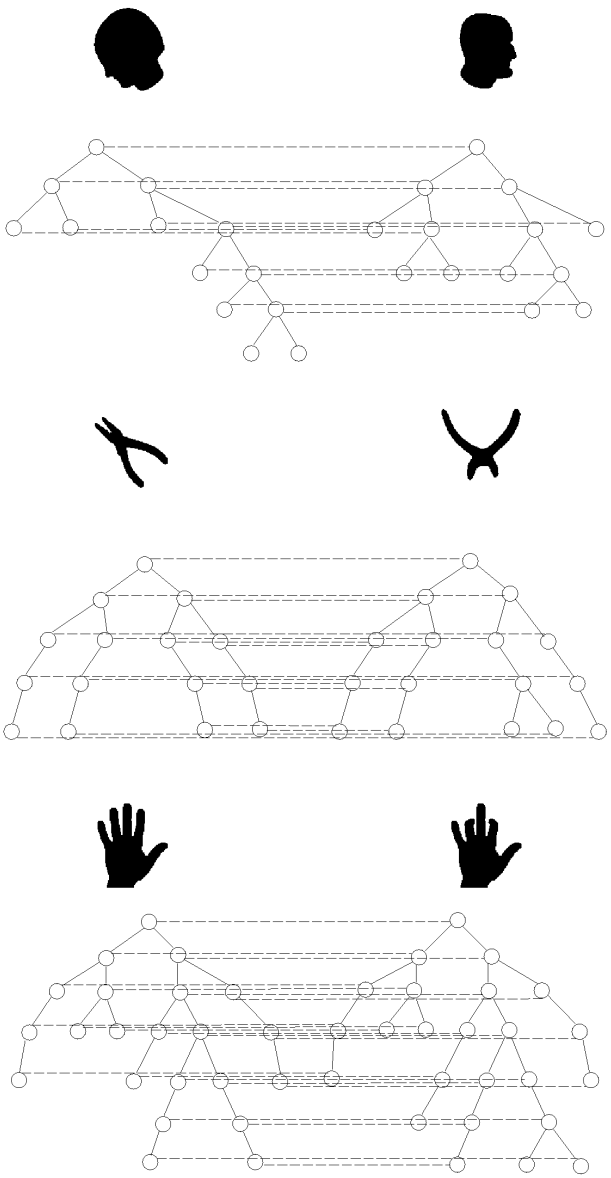


Fig. 6. Maximal subtree isomorphisms found for three illustrative examples. The shock-based descriptions of the hand silhouettes are shown in Fig. 5; the shock-trees for the other silhouettes were computed in a similar fashion.

6 Attributed Tree Matching and Weighted TAGs

In many computer vision and pattern recognition applications, the trees being matched have nodes with an associated vector of symbolic and/or numeric attributes. In this section, we show how the proposed framework can naturally be extended for solving attributed tree matching problems.

6.1 Attributed Tree Matching as Weighted Clique Search

Formally, an *attributed tree* is a triple $T = (V, E, \alpha)$, where (V, E) is the "underlying" rooted tree and α is a function which assigns an attribute vector $\alpha(u)$ to each node $u \in V$. It is clear that, in matching two attributed trees, our objective is to find an isomorphism which pairs nodes having "similar" attributes. To this end, let σ be any similarity measure on the attribute space, i.e., any (symmetric) function which assigns a positive number to any pair of attribute vectors. If $\phi : H_1 \rightarrow H_2$ is a subtree isomorphism between two attributed trees $T_1 = (V_1, E_1, \alpha_1)$ and $T_2 = (V_2, E_2, \alpha_2)$, the overall similarity between the induced subtrees $T_1[H_1]$ and $T_2[H_2]$ can be defined as follows:

$$S(\phi) = \sum_{u \in H_1} \sigma(\alpha_1(u), \alpha_2(\phi(u))).$$

The isomorphism ϕ is called a *maximal similarity subtree isomorphism* if there is no other subtree isomorphism $\phi' : H_1' \rightarrow H_2'$ such that H_1 is a strict subset of H_1' and $S(\phi) < S(\phi')$. It is called a *maximum similarity subtree isomorphism* if $S(\phi)$ is largest among all subtree isomorphisms between T_1 and T_2.

The *weighted TAG* of two attributed trees T_1 and T_2 is the weighted graph $G = (V, E, \omega)$, where V and E are defined as in Definition 2 and ω is a function which assigns a positive weight to each node $(u, v) \in V = V_1 \times V_2$ as follows:

$$\omega(u, v) = \sigma(\alpha_1(u), \alpha_2(v)).$$

Given a subset of nodes C of V, the total weight assigned to C is simply the sum of all the weights associated with its nodes. A *maximal weight* clique in G is one which is not contained in any other clique having larger total weight, while a *maximum weight* clique is a clique having largest total weight. The maximum weight clique problem is to find a maximum weight clique of G [9]. Note that the unweighted version of the maximum clique problem arises as a special case when all the nodes are assigned a constant weight.

The following result, which is the weighted counterpart of Theorem 1, establishes a one-to-one correspondence between the attributed tree matching problem and the maximum weight clique problem (the proof is essentially identical to that of Theorem 1).

Theorem 4. *Any maximal (maximum) similarity subtree isomorphism between two attributed trees induces a maximal (maximum) weight clique in the corresponding weighted TAG, and vice versa.*

6.2 Matching Attributed Trees

Recently, the Motzkin-Straus formulation of the maximum clique problem has been extended to the weighted case [19]. Let $G = (V, E, \omega)$ be an arbitrary weighted graph of order n. The (weighted) *characteristic vector* of any subset of nodes $C \subseteq V$, denoted $\mathbf{x}^c$, is defined as follows:

$$x_i^c = \begin{cases} \omega(u_i)/\Omega(C), & \text{if } u_i \in C \\ 0, & \text{otherwise,} \end{cases}$$

where $\Omega(C) = \sum_{u_j \in C} \omega(u_j)$ is the total weight on C.

Now, consider the following class of $n \times n$ symmetric matrices:

$$\mathcal{M}(G) = \Big\{ B = (b_{ij}) : b_{ii} = \frac{1}{2\omega(u_i)},\ b_{ij} = 0 \text{ if } u_i \sim u_j,$$
$$b_{ij} = b_{ji} \geq b_{ii} + b_{jj} \text{ otherwise} \Big\}$$

and the quadratic function

$$g(\mathbf{x}) = \mathbf{x}' B \mathbf{x}, \tag{6}$$

where $B \in \mathcal{M}(G)$. The following theorem, which is the weighted counterpart of Theorem 2, expands on a recent result by Gibbons et al. [19], which in turn generalizes the Motzkin-Straus theorem to the weighted case (see [8], [11] for a proof).

Theorem 5. *Let $G = (V, E, \omega)$ be an arbitrary weighted graph and let $B \in \mathcal{M}(G)$. Then, the following hold:*

1. *A vector $\mathbf{x} \in S_n$ is a local minimizer of g on S_n if and only if $\mathbf{x} = \mathbf{x}^c$, where C is a maximal weight clique of G.*
2. *A vector $\mathbf{x} \in S_n$ is a global minimizer of g on S_n if and only if $\mathbf{x} = \mathbf{x}^c$, where C is a maximum weight clique of G.*
3. *All local (and hence global) minimizers of g on S_n are strict.*

In contrast to the Gibbons et al. formulation [19], which is plagued by the presence of spurious solutions, as is the original Motzkin-Straus problem [11], the previous result guarantees that all minimizers of g on S_n are strict and are characteristic vectors of maximal/maximum weight cliques in the graph. Note that the class $\mathcal{M}(G)$ is isomorphic to the positive orthant in $\binom{n}{2} - |E|$ dimensions, where $|E|$ is the number of edges in G. This class is a polyhedral pointed cone with its apex given by the following matrix, which is the one used in the experiments described below:

$$b_{ij} = \begin{cases} \frac{1}{2\omega(u_i)} & \text{if } i = j, \\ 0 & \text{if } i \neq j \text{ and } u_i \sim u_j, \\ \frac{1}{2\omega(u_i)} + \frac{1}{2\omega(u_j)} & \text{otherwise.} \end{cases} \tag{7}$$

Having formulated the maximum weight clique problem as a quadratic program over the standard simplex, we can use the replicator equations to approximately solve it. However, note that replicator equations are maximization procedures, while ours is a minimization problem. It is straightforward to see that the problem of minimizing the quadratic form $\mathbf{x}' B \mathbf{x}$ on the simplex is equivalent to that of maximizing $\mathbf{x}'(\gamma \mathbf{e}\mathbf{e}' - B)\mathbf{x}$, where γ is an arbitrary constant.[1]

Now, let T_1 and T_2 be two attributed trees, $G = (V, E, \omega)$ be the corresponding weighted TAG, and define

$$W = \gamma \mathbf{e}\mathbf{e}' - B, \tag{8}$$

where $B = (b_{ij})$ is any matrix in the class $\mathcal{M}(G)$ and $\gamma = \max b_{ij}$. From the fundamental theorem of natural selection (Theorem 3), we know that the replicator dynamical systems (3) and (4) will find local maximizers of the function $\mathbf{x}' W \mathbf{x}$ (and, hence, minimizers of $\mathbf{x}' B \mathbf{x}$) over the standard simplex and, by virtue of Theorem 5, these will correspond to the characteristic vectors of maximal weight cliques in the weighted TAG. As stated in Theorem 4, these will in turn induce maximal similarity subtree isomorphisms between T_1 and T_2. As in the unweighted case, there is no theoretical guarantee that the solutions found will be the globally optimal ones, but the experiments reported in [11] on the maximum weight clique problem suggest that, here too, the attraction basins of global maximizers are quite large. This observation is also confirmed by the experiments reported below, each of which typically took 5 to 10 seconds to run on a Sparc 10.

6.3 Experimental Results

We now provide examples of weighted shock tree matching, using the geometric attributes associated with shock tree nodes. The vector of attributes assigned to each node $u \in V$ of the attributed shock tree $T = (V, E, \alpha)$ is given by $\alpha(u) = (x_1, y_1, r_1, v_1, \theta_1; ...; x_m, y_m, r_m, v_m, \theta_m)$. Here, m is the number of shocks in the group, and $x_i, y_i, r_i, v_i, \theta_i$ are, respectively, the x coordinate, the y coordinate, the radius (or time of formation), the speed, and the direction of each shock i in the sequence, obtained as outputs of the shock detection process [60], [61]. In order to apply our framework, we must define a similarity measure between the attributes of two nodes u and v.

The similarity measure we use is a linear combination of four terms, incorporating the differences in lengths, radii, velocities, and curvature of the two shock sequences, respectively. Each term is normalized to provide a unitless quantity so that these different geometric properties can be combined. Let u contain m shocks and v contain n shocks and, without loss of generality, assume that $m \geq n$. The Euclidean length of each sequence of shocks is given by:

$$L(u) = \sum_{i=1}^{m-1} \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2},$$
$$L(v) = \sum_{j=1}^{n-1} \sqrt{(x_j - x_{j+1})^2 + (y_j - y_{j+1})^2}.$$

Let $\xi(i) : i \to \lceil \frac{i.n}{m} \rceil$ be the many-to-one mapping of each index $i \in \{1, ..., m\}$ to an index $j \in \{1, ..., n\}$. The similarity measure between the two attribute vectors used in our experiments is defined as:

1. Note that the conversion of the minimization problem to a maximization problem is driven by a purely algorithmic (as opposed to formulation) issue. One could minimize $\mathbf{x}' B \mathbf{x}$ on the simplex using alternative optimization techniques [29], [53]. However, since we are matching trees and not graphs, we expect the difference in performance to be marginal.

$$\sigma(\alpha(u),\alpha(v)) = 1 - \beta_l \frac{|L(u)-L(v)|}{max(L(u),L(v))} - \beta_r \left\{ \frac{1}{m}\sum_{i=1}^{m}\left(\frac{r(i)-r(\xi(i))}{max(r(i),r(\xi(i)))}\right)^2 \right\}^{\frac{1}{2}} - \beta_v \left\{ \frac{1}{m}\sum_{i=1}^{m}\left(\frac{v(i)-v(\xi(i))}{max(v(i),v(\xi(i)))}\right)^2 \right\}^{\frac{1}{2}} - \beta_{\Delta\theta} \left\{ \frac{1}{m}\sum_{i=1}^{m}\left(\frac{\Delta\theta(i)-\Delta\theta(\xi(i))}{max(\Delta\theta(i),\Delta\theta(\xi(i)))}\right)^2 \right\}^{\frac{1}{2}}, \quad (9)$$

where $\beta_l, \beta_r, \beta_v, \beta_{\Delta\theta}$ are nonnegative constants summing to 1 and $\Delta\theta$ is the change in orientation at each shock. The measure provides a number between 0 and 1, which represents the overall similarity between the geometric attributes of the two nodes being compared. The measure is designed to be invariant under rotations and translations of two shapes and to satisfy the requirements of the weight function discussed in Section 6.1.

We repeated the earlier experiments, but now with weights $w(u,v) = \sigma(\alpha(u),\alpha(v))$ placed on each node (u,v) of a weighted TAG. The weights were defined by (9), with $\beta_l, \beta_r, \beta_v, \beta_{\Delta\theta}$ set to 0.25, 0.4, 0.2, and 0.15, respectively. We verified that the overall results were not sensitive to the precise choice of these parameters or to slight variations in the similarity measure such as the use of a different norm. We ranked the results using a score given by the quantity $W \times \frac{1}{2}\{\sum_{i=1}^{n}(m(u_i)/M_1 + m(v_i)/M_2)\}$, where n is the size of the maximal weight clique found, W its weight, M_1 and M_2 the total mass associated with the nodes of each tree, and $m(u), m(v)$ the masses of nodes u and v, respectively. The score represents the weight of the maximal clique scaled by the average of the total (relative) mass of nodes in each tree that participates in the match. As before, the top eight matches are shown for each query shape, in Table 2. It is evident that, for almost all queries, performance improves with the incorporation of geometric attributes, with better overall discrimination between the shapes (compare with Table 1). Nonetheless, there is an inherent trade-off between geometry and topology, e.g., observe that the short fat screwdriver in row 15 scores better with the fatter hand shapes than with the thin, elongated screwdrivers.

We note that the qualitative results, specifically, the partial ordering of matches in each row of Table 2, compare favorably with those obtained using an alternate approach described in [62]. The latter approach has been applied successfully to the same database of shapes used here and is a sequential (level by level) approach with backtracking. At each step, an eigenvalue labeling of the adjacency matrix of the subtrees rooted at the two nodes being matched is used. In other words, the cost of matching two nodes incorporates not only their geometries, but a measure of the similarity between their subtrees, which is *global*. Furthermore, the algorithm tolerates noise by allowing for jumps between levels. Hence, strictly speaking, it solves an *approximation* to the subtree isomorphism problem. For these reasons, a one-to-one comparison between the two algorithms is not possible; they solve different problems and the maximal clique formulation invokes much weaker constraints. In [4], we present an extension of the maximal clique formulation to the case of many-to-one correspondences, which is of particular interest in computer vision and pattern recognition applications where the trees being matched are noisy, and vertices are deleted or added.

7 Conclusions

We have developed a formal approach for matching hierarchical structures by constructing an association graph whose maximal cliques are in one-to-one correspondence with maximal subtree isomorphisms. The framework is general and can be applied in a variety of computer vision and pattern recognition domains: We have demonstrated its potential for shape matching. The formulation allows us to cast the tree-matching problem as an indefinite quadratic program owing to the Motzkin-Straus theorem. The solution is found by using a dynamical system, which makes it amenable to hardware implementation and offers the advantage of biological plausibility. In particular, these relaxation equations are related to putative neuronal implementations [38], [39]. We have also extended the framework to the problem of matching hierarchical structures with attributes. The attributes result in weights being placed on the nodes of the association graph and a conversion of the maximum clique problem to a maximum weight clique problem. An extension of the proposed framework to problems involving many-to-one correspondences is presented in [4].

Characterizing the complexity of our approach appears to be difficult since it involves the simulation of a dynamical system. However, we have observed experimentally that the basins of attraction of the global maximizers are large, both in the unweighted and weighted cases, and that the system converges quickly when applied to shock-tree matching. Conversely, whereas polynomial time algorithms exist for the maximum common subtree problem [37], [54], [17], to our knowledge no such algorithm exists for the case of weighted tree matching. This provides further justification for our framework.

Appendix

Proof of Lemma 1

Before presenting the proof of Lemma 1, we need some preliminary remarks and definitions. First, note that path-strings cannot be arbitrary strings of -1s and $+1$s. Since trees do not have cycles, in fact, once we go down one level along a path (i.e., we make a "+1" move), we cannot return to the parent. This is formally stated by saying that if $\mathrm{str}(u,v) = s_1 s_2 \ldots s_n$ is the path-string between any two nodes u and v, then $s_i = +1$ implies $s_j = +1$ for all $j \geq i$.

Now, we define the *path-pair* of any two nodes u and v in a tree as $\mathrm{pair}(u,v) = (n,p)$, where n is the number of negative components in $\mathrm{str}(u,v)$ and p is the number of positive components in $\mathrm{str}(u,v)$. It is clear from the previous observation that path-pairs and path-strings are equivalent concepts. In fact, we have: $\mathrm{str}(u,v) = \mathrm{str}(w,z)$ if and only if

TABLE 2
A Tabulation of the Top Eight Attributed Toplogical Matches for Each Query

Query Shape	Top 8 Attributed Topological Matches							
	1	2	3	4	5	6	7	8
	 10.00	 6.97	 5.89	 4.94	 4.85	 4.83	 4.74	 4.54
	 10.00	 6.97	 5.33	 5.22	 5.10	 5.04	 4.98	 4.87
	 8.00	 5.89	 5.33	 4.73	 4.73	 4.67	 4.66	 4.61
	 12.00	 8.91	 6.82	 5.73	 4.73	 4.26	 3.49	 3.38
	 9.00	 6.82	 6.78	5.00	4.93	4.68	3.90	3.76
	11.00	8.91	6.78	6.06	4.88	4.34	3.60	3.58
	15.00	10.79	10.26	10.13	8.90	7.83	6.65	6.29
	16.00	9.22	9.21	9.05	8.90	6.68	5.53	5.32
	14.00	10.13	9.61	9.48	9.21	8.20	6.10	6.10
	12.00	8.28	8.20	7.83	7.68	6.68	5.64	5.53
	14.00	10.79	9.94	9.61	9.05	8.28	5.77	5.70
	15.00	10.26	9.94	9.48	9.22	7.68	6.20	6.16
	12.00	8.96	5.15	4.61	4.53	4.52	4.49	4.28
	12.00	8.96	4.99	4.47	4.43	4.36	4.33	3.87
	13.00	6.67	6.20	6.10	5.15	4.99	4.70	4.65
	8.00	5.58	4.68	4.34	4.26	3.16	3.01	2.93
	10.00	6.06	5.73	5.58	5.00	3.90	2.94	2.73
	21.00	19.23	18.49	7.37	6.45	6.22	6.20	6.20
	22.00	18.49	17.83	7.35	6.29	6.21	6.16	6.10
	23.00	19.23	17.83	7.44	6.92	6.67	6.16	6.10
	12.00	7.44	7.37	7.35	5.93	5.63	5.22	4.64
	12.00	9.67	5.93	5.35	5.28	5.28	4.70	3.34
	12.00	9.67	5.63	5.21	5.15	5.11	4.57	3.23
	40.00	12.44	6.92	6.65	6.45	6.21	5.52	5.32
	31.00	12.44	6.10	6.04	6.02	5.38	5.87	5.28

The scores indicate the weight of the maximal clique multiplied by the average of the total relative mass of nodes in each tree matched (see text). The addition of geometric information permits finer discrimination between the shapes (compare with Table 1).

$\mathrm{pair}(u,v) = \mathrm{pair}(w,z)$. Moreover, note that if a node w is on the path between any two nodes u and v in a rooted tree, then $\mathrm{str}(u,v)$ can be obtained by concatenating $\mathrm{str}(u,w)$ and $\mathrm{str}(w,v)$. This implies that

$$\mathrm{pair}(u,v) = \mathrm{pair}(u,w) + \mathrm{pair}(w,v),$$

where "+" denotes the usual sum between vectors. In a sense, then, path-pairs allow us to do "arithmetic" on path-strings, a fact which will be technically useful in the sequel. (The full algebraic structure will not be needed here.)

We are now in a position to prove Lemma 1. For convenience, we repeat its statement below.

Lemma 1. *Let* $u_1, v_1, w_1, z_1 \in V_1$ *and* $u_2, v_2, w_2, z_2 \in V_2$ *be distinct nodes of rooted trees* $T_1 = (V_1, E_1)$ *and* $T_2 = (V_2, E_2)$*, and suppose that the following conditions hold:*

1. w_1 *is on the* u_1v_1*-path, and* w_2 *is on the* u_2v_2*-path*
2. $\mathrm{str}(u_1, w_1) = \mathrm{str}(u_2, w_2)$
3. $\mathrm{str}(w_1, v_1) = \mathrm{str}(w_2, v_2)$
4. $\mathrm{str}(u_1, z_1) = \mathrm{str}(u_2, z_2)$
5. $\mathrm{str}(v_1, z_1) = \mathrm{str}(v_2, z_2)$

Then, $\mathrm{str}(w_1, z_1) = \mathrm{str}(w_2, z_2)$.

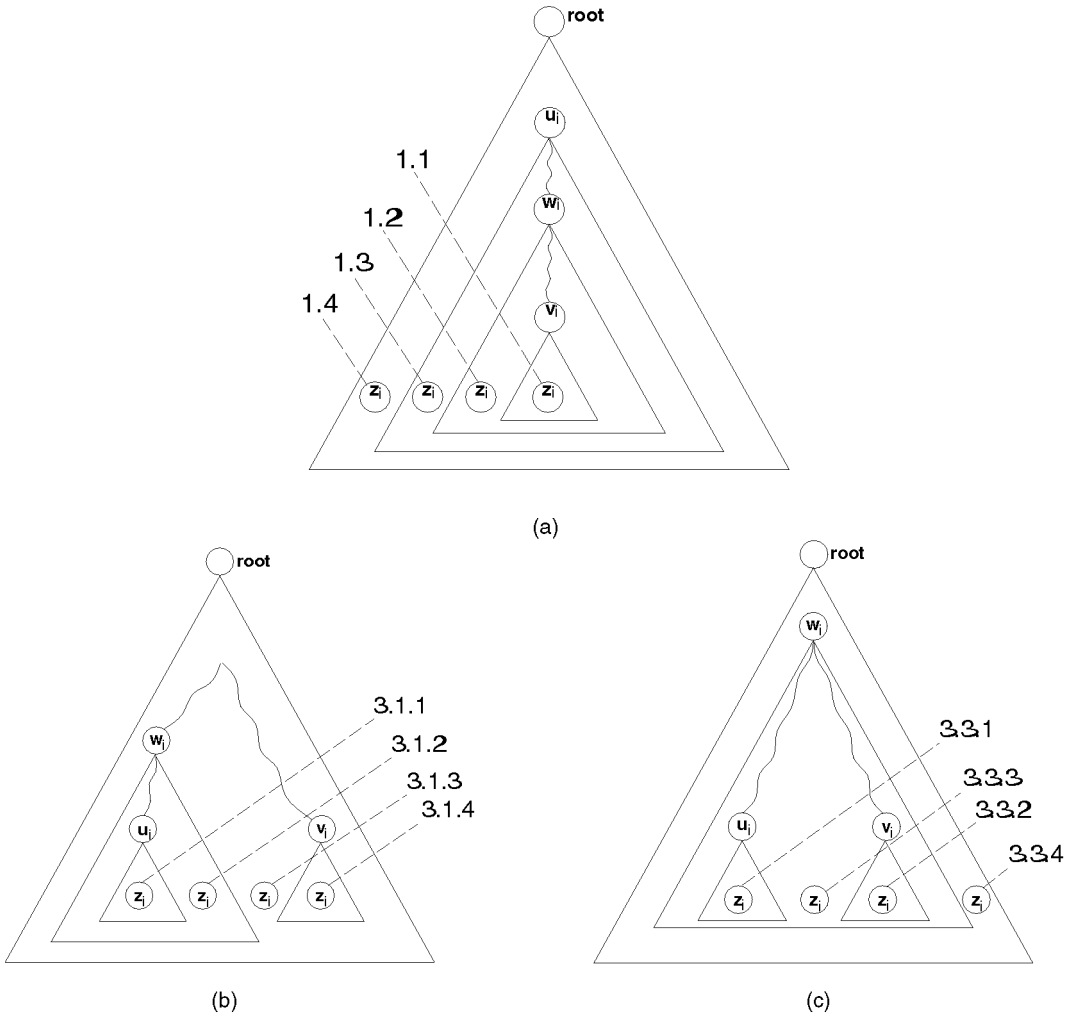


Fig. 7. An illustration of the cases arising in the proof of Lemma 1.

Proof. First, note that, from conditions 1-3, we also have

6. $\mathrm{str}(u_1, v_1) = \mathrm{str}(u_2, v_2)$.

We shall prove that $\mathrm{pair}(w_1, z_1) = \mathrm{pair}(w_2, z_2)$, which is, of course, equivalent to the thesis of the lemma. The proof consists of enumerating all possible cases and exploiting the previous observation that, when w is on a uv-path, then $\mathrm{pair}(u, v) = \mathrm{pair}(u, w) + \mathrm{pair}(w, v)$. Before doing so, however, we need an auxiliary notation. Let u be a node of a rooted tree $T = (V, E)$. The set of nodes belonging to the subtree rooted at u will be denoted by $V(u)$. This can be formally defined as follows:

$$V(u) = \left\{ v \in V : \mathrm{str}(u, v) = \underbrace{+1 + 1 \ldots + 1}_{n}, \text{ for some } n \geq 0 \right\}.$$

Note that $u \in V(u)$.

We shall enumerate all the possible ways in which u_1, v_1, w_1, and z_1 can relate to one another in T_1, but it is clear from conditions 1-6 that each such configuration induces a perfectly symmetric situation in T_2 and vice versa. Therefore, from now on, we shall use the index $i = 1, 2$ to simplify the discussion. Technically, this means that we are assuming something about one tree and, because of our hypotheses, the same situation arises in the other.

The enumeration of all possible cases starts from the observation that, given two different subtrees of a tree, either one is a strict subset of the other or they are disjoint (otherwise, in fact, there will be cycles in the graph). Therefore, considering the subtrees rooted at nodes u_i and v_i ($i = 1, 2$), only the following cases can arise:

1. $V(v_i) \subset V(u_i)$
2. $V(u_i) \subset V(v_i)$
3. $V(u_i) \cap V(v_i) = \emptyset$

The first two are symmetric and, therefore, we shall only consider Case 1. In this case, since w_i is on the $u_i v_i$-path, we have $V(v_i) \subset V(w_i)$, $i = 1, 2$. Clearly, only four subcases are possible (cf. Fig. 7a), that is:

1.1 $z_i \in V(v_i)$
1.2 $z_i \in V(w_i) \setminus V(v_i)$
1.3 $z_i \in V(u_i) \setminus V(w_i)$
1.4 $z_i \notin V(u_i)$

where $i = 1, 2$.

Let us consider Case 1.1. In this case, the v_is are on the w_iz_i-paths and, therefore:

$$\begin{aligned}\text{pair}(w_1, z_1) &= \text{pair}(w_1, v_1) + \text{pair}(v_1, z_1)\\ &= \text{pair}(w_2, v_2) + \text{pair}(v_2, z_2)\\ &= \text{pair}(w_2, z_2).\end{aligned}$$

In Case 1.2, we have that the w_is are on the u_iz_i-paths and, hence, $\text{pair}(u_i, z_i) = \text{pair}(u_i, w_i) + \text{pair}(w_i, z_i)$, $i = 1, 2$. Therefore, we have:

$$\begin{aligned}\text{pair}(w_1, z_1) &= \text{pair}(u_1, z_1) - \text{pair}(u_1, w_1)\\ &= \text{pair}(u_2, z_2) - \text{pair}(u_2, w_2)\\ &= \text{pair}(u_2, w_2) + \text{pair}(w_2, z_2) - \text{pair}(u_2, w_2)\\ &= \text{pair}(w_2, z_2).\end{aligned}$$

Case 1.3 is similar to Case 1.2. In this case, we have that w_i is on the z_iv_i-path and, therefore

$$\begin{aligned}\text{pair}(w_1, z_1) &= \text{pair}(v_1, z_1) - \text{pair}(v_1, w_1)\\ &= \text{pair}(v_2, z_2) - \text{pair}(v_2, w_2)\\ &= \text{pair}(w_2, z_2).\end{aligned}$$

Finally, Case 1.4 is similar to Case 1.1, since u_i is on the path joining w_i and z_i.

Now, let us consider Case 3, i.e., $V(u_i) \cap V(v_i) = \emptyset$ ($i = 1, 2$). Here, the situation is slightly more complicated. Since w_i is on the u_iv_i-path and this has the form $-1 - 1 \ldots - 1 + 1 + 1 \ldots + 1$, we have the following three subcases:

3.1 $V(u_i) \subset V(w_i)$ and $V(w_i) \cap V(v_i) = \emptyset$
3.2 $V(v_i) \subset V(w_i)$ and $V(w_i) \cap V(u_i) = \emptyset$
3.3 $V(v_i) \subset V(w_i)$ and $V(u_i) \subset V(w_i)$

Cases 3.1 and 3.2 are symmetric, so we shall only consider Cases 3.1 and 3.3.

Let us start with Case 3.1. Four possible subcases arise (Fig. 7b):

3.1.1
$z_i \in V(u_i)$
3.1.2
$z_i \in V(w_i) \setminus V(u_i)$
3.1.3
$z_i \notin V(w_i)$ and $z_i \notin V(v_i)$
3.1.4
$z_i \in V(v_i)$

All these cases are similar to the previous ones and the corresponding proofs are therefore analogous.

Finally, let us consider Case 3.3. Again, four possible subcases arise (Fig. 7c):

3.3.1.
$z_i \in V(u_i)$
3.3.2.
$z_i \in V(v_i)$
3.3.3.
$z_i \notin V(u_i)$ and $z_i \notin V(v_i)$ and $z_i \in V(w_i)$
3.3.4.
$z_i \notin V(w_i)$

Here, Cases 3.3.1, 3.3.2, and 3.3.4 are similar to those seen before and, hence, we omit the corresponding proofs. As to Case 3.3.3, we note that w_i must necessarily be either on the path joining u_i and z_i or on that joining v_i and z_i (or on both), otherwise the graph would have a cycle. So, the proof in this case is analogous to the previous ones, and this concludes the proof of the lemma. □

ACKNOWLEDGMENTS

This work was done while Marcello Pelillo was visiting the Center for Computational Vision and Control at Yale University. It was supported by Consiglio Nazionale delle Ricerche (Italy), NSERC, FCAR, NSF, and AFOSR. We thank the reviewers for their helpful comments and Sven Dickinson for the use of shapes from the Rutgers Tools database.

REFERENCES

[1] A.P. Ambler, H.G. Barrow, C.M. Brown, R.M. Burstall, and R.J. Popplestone, "A Versatile Computer-Controlled Assembly System," *Proc. Int'l Joint Conf. Artificial Intelligence,* pp. 298-307, Stanford, Calif., 1973.

[2] D.H. Ballard and C.M. Brown, *Computer Vision.* Englewood Cliffs, N.J.: Prentice Hall, 1982.

[3] H.G. Barrow and R.M. Burstall, "Subgraph Isomorphism, Matching Relational Structures, and Maximal Cliques," *Information Processing Letters,* vol. 4, no. 4, pp. 83-84, 1976.

[4] M. Bartoli, M. Pelillo, K. Siddiqi, and S.W. Zucker, "Attributed Tree Homomorphism Using Association Graphs," Technical Report CS-99-12, Dipartimento di Informatica, Università Ca' Foscari di Venezia, 1999.

[5] L.E. Baum and J.A. Eagon, "An Inequality with Applications to Statistical Estimation for Probabilistic Functions of Markov Processes and to a Model for Ecology," *Bulletin Am. Math. Soc.,* vol. 73, pp. 360-363, 1967.

[6] R.C. Bolles and R.A. Cain, "Recognizing and Locating Partially Visible Objects: The Locus-Feature-Focus Method," *Int'l J. Robotics Research,* vol. 1, no. 3, pp. 57-82, 1982.

[7] I.M. Bomze, "Evolution Towards the Maximum Clique," *J. Global Optimization,* vol. 10, pp. 143-164, 1997.

[8] I.M. Bomze, "On Standard Quadratic Optimization Problems," *J. Global Optimization,* vol. 13, pp. 369-387, 1998.

[9] I.M. Bomze, M. Budinich, P.M. Pardalos, and M. Pelillo, "The Maximum Clique Problem," *Handbook of Combinatorial Optimization,* D.-Z. Du and P.M. Pardalos, eds., vol. 4. Boston, Mass.: Kluwer Academic, 1999.

[10] I.M. Bomze, M. Pelillo, and R. Giacomini, "Evolutionary Approach to the Maximum Clique Problem: Empirical Evidence on a Larger Scale," *Developments in Global Optimization,* I.M. Bomze, T. Csendes, R. Horst, and P.M. Pardalos, eds., pp. 95-108, Dordrecht, The Netherlands: Kluwer, 1997.

[11] I.M. Bomze, M. Pelillo, and V. Stix, "Approximating the Maximum Weight Clique Using Replicator Dynamics," Technical Report CS-99-13, Dipartimento di Informatica, Università Ca' Foscari di Venezia, 1999.

[12] R. Brockett and P. Maragos, "Evolution Equations for Continuous-Scale Morphology," *Proc. IEEE Conf. Acoustics, Speech, and Signal Processing,* San Francisco, Mar. 1992.

[13] J.F. Crow and M. Kimura, *An Introduction to Population Genetics Theory.* New York: Harper & Row, 1970.

[14] R. Durbin and D. Willshaw, "An Analog Approach to the Travelling Salesman Problem Using an Elastic Net Method," *Nature,* vol. 326, pp. 689-691, 1987.

[15] R.A. Fisher, *The Genetical Theory of Natural Selection.* London: Oxford Univ. Press, 1930.

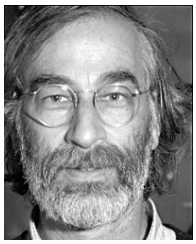
[16] Y. Fu and P.W. Anderson, "Application of Statistical Mechanics to NP-Complete Problems in Combinatorial Optimization," *J. Physics A,* vol. 19, pp. 1,605-1,620, 1986.

[17] M.R. Garey and D. Johnson, *Computer and Intractability: A Guide to the Theory of NP-Completeness.* San Francisco: Freeman, 1979.
[18] L.E. Gibbons, D.W. Hearn, and P.M. Pardalos, "A Continuous Based Heuristic for the Maximum Clique Problem," *Cliques, Coloring, and Satisfiability—Second DIMACS Implementation Challenge,* D.S. Johnson and M. Trick, eds., pp. 103-124, 1996.
[19] L.E. Gibbons, D.W. Hearn, P.M. Pardalos, and M.V. Ramana, "Continuous Characterizations of the Maximum Clique Problem," *Math. Operations Research,* vol. 22, pp. 754-768, 1997.
[20] S. Gold and A. Rangarajan, "A Graduated Assignment Algorithm for Graph Matching," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 18, no. 4, pp. 377-388, Apr. 1996
[21] M. Grötschel, L. Lovász, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization.* Berlin: Springer-Verlag, 1988.
[22] F. Harary, *Graph Theory.* Reading, Mass.: Addison-Wesley, 1969.
[23] J. Hofbauer and K. Sigmund, *The Theory of Evolution and Dynamical Systems.* Cambridge, U.K.: Cambridge Univ. Press, 1988/
[24] J.J. Hopfield and D.W. Tank, "Neural Computation of Decisions in Optimization Problems," *Biological Cybernetics,* vol. 52, pp. 141-152, 1985.
[25] R. Horaud and T. Skordas, "Stereo Correspondence through Feature Grouping and Maximal Cliques," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 11, no. 11, pp. 1,168-1,180, Nov. 1989.
[26] A. Jagota, "Approximating the Maximum Clique with a Hopfield Network," *IEEE Trans. Neural Networks,* vol. 6, pp. 724-735, 1995.
[27] B.B. Kimia, A. Tannenbaum, and S.W. Zucker, "Toward a Computational Theory of Shape: An Overview," *Lecture Notes in Computer Science,* vol. 427, pp. 402-407, 1990.
[28] B.B. Kimia, A. Tannenbaum, and S.W. Zucker, "Shape, Shocks, and Deformations I: The Components of Two-Dimensional Shape and the Reaction-Diffusion Space," *Int' J. Computer Vision,* vol. 15, pp. 189-224, 1995.
[29] 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.
[30] P.D. Lax, "Shock Waves and Entropy," *Contributions to Nonlinear Functional Analysis,* E.H. Zarantonello, ed., pp. 603-634, New York: Academic Press, 1971.
[31] J.T. Li, K. Zhang, K. Jeong, and D. Shasha, "A System for Approximate Tree Matching," *IEEE Trans. Knowledge and Data Eng.,* vol. 6, pp. 559-571, 1994.
[32] T.L. Liu, D. Geiger, and R.V. Kohn, "Representation and Self-Similarity of Shapes," *Proc. Int'l Conf. Computer Vision,* pp. 1,129-1,135, Bombay, 1998.
[33] V. Losert and E. Akin, "Dynamics of Games and Genes: Discrete versus Continuous Time," *J. Math. Biology,* vol. 17, pp. 241-251, 1983.
[34] S.Y. Lu, "A Tree-Matching Algorithm Based on Node Splitting and Merging," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 6, pp. 249-256, 1984.
[35] Y. Lyubich, G.D. Maistrowskii, and Y.G. Ol'khovskii, "Selection-Induced Convergence to Equilibrium in a Single-Locus Autosomal Population," *Problems of Information Transmission,* vol. 16, pp. 66-75, 1980.
[36] D. Marr and K.H. Nishihara, "Representation and Recognition of the Spatial Organization of Three-Dimensional Shapes," *Proc. Royal Soc. London B,* vol. 200, pp. 269-294, 1978.
[37] D.W. Matula, "An Algorithm for Subtree Identification," *SIAM Review,* vol. 10, pp. 273-274, 1968.
[38] D. Miller and S.W. Zucker, "Efficient Simplex-Like Methods for Equilibria of Nonsymmetric Analog Networks," *Neural Computation,* vol. 4, no. 2, pp. 167-190, 1992.
[39] D. Miller and S.W. Zucker, "Computing with Self-Excitatory Cliques: A Model and an Application to Hyperacuity-Scale Computation in Visual Cortex," *Neural Computation,* vol. 11, no. 1, pp. 21-66, 1999.
[40] T.S. Motzkin and E.G. Straus, "Maxima for Graphs and a New Proof of a Theorem of Turán," *Canadian J. Math.,* vol. 17, pp. 533-540, 1965.
[41] M. Neff, R. Byrd, and O. Rizk, "Creating and Querying Hierarchical Lexical Databases," *Proc. Conf. Applied Natural Language Processes,* pp. 84-93, 1988.
[42] H. Ogawa, "Labeled Point Pattern Matching by Delaunay Triangulation and Maximal Cliques," *Pattern Recognition,* vol. 19, pp. 35-40, 1986.
[43] M. Ohlsson, C. Peterson, and B. Söderberg, "Neural Networks for Optimization Problems with Inequality Constraints: The Knapsack Problem," *Neural Computation,* vol. 5, pp. 331-339, 1993.
[44] P.M. Pardalos, "Continuous Approaches to Discrete Optimization Problems," *Nonlinear Optimization and Applications,* G.D. Pillo and F. Giannessi, eds., pp. 313-328. Plenum Press, 1996.
[45] P.M. Pardalos and A.T. Phillips, "A Global Optimization Approach for Solving the Maximum Clique Problem," *Int'l J. Computer Math.,* vol. 33, pp. 209-216, 1990.
[46] M. Pelillo, "Relaxation Labeling Networks for the Maximum Clique Problem," *J. Artificial Neural Networks,* vol. 2, pp. 313-328, 1995.
[47] M. Pelillo, "The Dynamics of Nonlinear Relaxation Labeling Processes," *J. Math. Imaging and Vision,* vol. 7, pp. 309-323, 1997.
[48] M. Pelillo, "A Unifying Framework for Relational Structure Matching," *Proc. Int'l Conf. Pattern Recognition,* pp. 1,316-1,319, Brisbane, Australia, 1998.
[49] M. Pelillo, "Replicator Equations, Maximal Cliques, and Graph Isomorphism," *Neural Computation,* vol. 11, no. 8, 1999.
[50] M. Pelillo and A. Jagota, "Feasible and Infeasible Maxima in a Quadratic Program for Maximum Clique," *J. Artificial Neural Networks,* vol. 2, pp. 411-420, 1995.
[51] F. Pla and J.A. Marchant, "Matching Feature Points in Image Sequences through a Region-Based Method," *Computer Vision and Image Understanding,* vol. 66, no. 3, pp. 271-285, 1997.
[52] B. Radig, "Image Sequence Analysis Using Relational Structures," *Pattern Recognition,* vol. 17, pp. 161-167, 1984.
[53] A. Rangarajan and E. Mjolsness, "A Lagrangian Relaxation Network for Graph Matching," *IEEE Trans. Neural Networks,* vol. 7, no. 6, pp. 1,365-1,381, 1996.
[54] S.W. Reyner, "An Analysis of a Good Algorithm for the Subtree Problem," *SIAM J. Computing,* vol. 6, pp. 730-732, 1977.
[55] H. Rom and G. Medioni, "Hierarchical Decomposition and Axial Shape Description," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 15, no. 10, pp. 973-981, Oct. 1993.
[56] A. Rosenfeld, R.A. Hummel, and S.W. Zucker, "Scene Labeling by Relaxation Operations," *IEEE Trans. Systems, Man, and Cybernetics,* vol. 6, pp. 420-433, 1976.
[57] H. Samet, *Design and Analysis of Spatial Data Structures.* Reading, Mass.: Addison-Wesley, 1990.
[58] B.A. Shapiro and K. Zhang, "Comparing Multiple RNA Secondary Structures Using Tree Comparisons," *Computer Applications Bioscience,* vol. 6, pp. 309-318, 1990.
[59] D. Shasha, J.T.L. Wang, K. Zhang, and F.Y. Shih, "Exact and Approximate Algorithms for Unordered Tree Matching," *IEEE Trans. Systems, Man, and Cybernetics,* vol. 24, pp. 668-678, 1994.
[60] K. Siddiqi and B.B. Kimia, "A Shock Grammar for Recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 507-513, 1996.
[61] K. Siddiqi, B.B. Kimia, A. Tannenbaum, and S.W. Zucker, "Shapes, Shocks and Wiggles," *Image and Vision Computing,* vol. 17, nos. 5-6, pp. 365-373, 1999.
[62] K. Siddiqi, A. Shokoufandeh, S.J. Dickinson, and S.W. Zucker, "Shock Graphs and Shape Matching," *Int'l J. Computer Vision,* to appear 1999.
[63] J.W. Weibull, *Evolutionary Game Theory.* Cambridge, Mass.: MIT Press, 1995.
[64] H.S. Wilf, "Spectral Bounds for the Clique and Independence Numbers of Graphs," *J. Combinatorial Theory, Series B,* vol. 40, pp. 113-117, 1986.
[65] B. Yang, W.E. Snyder, and G.L. Bilbro, "Matching Oversegmented 3D Images Using Association Graphs," *Image and Vision Computing,* vol. 7, pp. 135-143, 1989.
[66] S. Zhu and A.L. Yuille, "FORMS: A Flexible Object Recognition and Modeling System," *Int'l J. Computer Vision,* vol. 20, no. 3, pp. 187-212, 1996.

Marcello Pelillo received the "Laurea" degree with honors in computer science from the University of Bari, Italy, in 1989. From 1988 to 1989, he was with the IBM Scientific Center in Rome, where he was involved in studies on natural language and speech processing. In 1991, he joined the Department of Computer Science at the University of Bari, Italy, as an assistant professor. Since 1995, he has been with the Department of Computer Science at the University of Venice, Italy. He has held visiting research positions at Yale University, McGill University, Canada, the University of Vienna, Austria, and the University of York.

His research interests are in the areas of pattern recognition, computer vision, and neural networks, where he has published more than 60 papers in refereed journals, handbooks, and conference proceedings. He was the program co-chair of the First and Second International Workshops on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR) held in Venice in May 1997, and in York, U.K., in July 1999, respectively. He has been a guest co-editor of a special issue of the journal *Pattern Recognition* devoted to this theme and is guest co-editor of a special issue of the *IEEE Transactions on Pattern Analysis and Machine Intelligence* devoted to "graph algorithms in computer vision." He is on the editorial board of *Pattern Recognition* and is a member of the IEEE, the IEEE Computer Society and the Pattern Recognition Society.

Kaleem Siddiqi received his BS degree from Lafayette College in 1988 and his MS and PhD degrees from Brown University in 1990 and 1995, all in the field of electrical engineering. He is an assistant professor in the School of Computer Science at McGill University. Before moving to McGill in 1998, he was a postdoctoral associate in the Department of Computer Science at Yale University (1996-1998) and held a visiting position in the Department of Electrical Engineering at McGill University (1995-1996). His research interests are in the areas of computer vision, image processing and human psychophysics. He is a member of the IEEE, the IEEE Computer Society, Phi Beta Kappa, Tau Beta Pi, and Eta Kappa Nu.



Steven W. Zucker obtained his education at Carnegie-Mellon University in Pittsburgh and at Drexel University in Philadelphia, and was a postdoctoral research fellow in computer science at the University of Maryland, College Park. He is the David and Lucille Packard Professor of Computer Science and Electrical Engineering at Yale University. Before moving to Yale in 1996, he was a professor of electrical engineering at McGill University, director of the Program in Artificial Intelligence and Robotics of the Canadian Institute for Advanced Research, and the co-director of the Computer Vision and Robotics Laboratory in the McGill Research Center for Intelligent Machines. He was elected a fellow of the Canadian Institute for Advanced Research, a fellow of the IEEE, and (by) a fellow of Churchill College, Cambridge.

He was Professor Inviteé at the Institut National de Recherche en Informatique et en Automatique, Sophia-Antipolis, France, in 1989, a visiting professor of computer science at Tel Aviv University in January 1993, and an SERC fellow of the Isaac Newton Institute for Mathematical Sciences, University of Cambridge. He has authored or coauthored more than 140 papers on computational vision, biological perception, artificial intelligence, and robotics, and serves on the editorial boards of eight journals.