Available at www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®

Pattern Recognition Letters 24 (2003) 1089–1097

Pattern Recognition Letters

www.elsevier.com/locate/patrec

Computing approximate tree edit distance using relaxation labeling

Andrea Torsello *, Edwin R. Hancock

Department of Computer Science, University of York, York, YO10 5DD, UK

Abstract

This paper presents a new method for computing the tree edit distance problem with uniform edit cost. We commence by showing that any tree obtained with a sequence of cut operations is a subtree of the transitive closure of the original tree, we show that the necessary condition for any subtree to be a solution can be reduced to a clique problem in a derived structure. Using this idea we transform the problem of computing tree edit distance into a series of maximum weight clique problems. We, then use relaxation labeling to find an approximation to the tree edit distance.

Keywords: Edit distance; Tree matching; Shock trees; Shape recognition

1. Introduction

Inexact or error-tolerant graph-matching is a problem of pivotal importance in the manipulation of relational structures that arises in many areas of machine intelligence (Eshera and Fu, 1986). One of the key issues which underlies the problem is how to measure the similarity of different graph structures. This task is frequently posed as that of computing the graph edit distance. Unfortunately, the reliable computation of edit distance has proved to be an elusive task (Wang et al., 1994). However, in a recent series of papers, Bunke has shown the intimate relationship between the size of the maximum common subgraph and edit distance (Bunke and Kandel, 2000). In particular, he demonstrated that MCS and graph edit distance computation are equivalent. This is an important observation since it has been established by Barrow and Burstall (1976) that the maximum common subgraph problem (MCSP) may be transformed into a maximum clique problem using a derived structure referred to as the association graph.

Transforming a graph matching problem into a max clique problem opens up to a wide spectrum of new possibilities. A diverse array of very powerful heuristics and theoretical results are available for solving the max clique problem. To this effect, a particularly important result is the Motzkin–Straus theorem (Motzkin and Straus, 1965) which allows the symbolic graph-matching problem to be embedded in a continuous space and to be solved via quadratic programming. Moreover, the method returns the size of the maximum common subgraph.

* Corresponding author.
E-mail address: atorsell@cs.york.ac.uk (A. Torsello).

0167-8655/03/$ - see front matter
PII: S0167-8655(02)00255-6

The observation underpinning this paper is that the Motzkin–Straus theorem provides a route to the computation of edit distance via a continuous means. Specifically, we are interested in how the observation may be applied to tree matching. While trees are a special case of graphs, the connectivity and partial order constraints that they represent require adaptation to be made to generic graph matching techniques so that they may be applied to trees. Furthermore, specific characteristics of trees suggest that posing the tree-matching problem as a variant on graph matching is not the best approach. In particular, both tree isomorphism and subtree isomorphism problems have efficient polynomial time solutions. Moreover, Tai (1979) has proposed a generalization of the string edit distance problem from the linear structure of a string to the non-linear structure of a tree. The resulting tree edit distance differs from the general graph edit distance in that edit operations are carried out only on nodes and never directly on edges. Zhang and Shasha (1989) have investigated a special case which involves adding the constraint that the solution must maintain the order of the children of a node. With this order among siblings, they showed that the tree-matching problem is still in P and gave an algorithm to solve it. In subsequent work they showed that the unordered case was indeed an NP hard problem (Zhang et al., 1992). The problem, though, returns to P when we add the constraint of strict hierarchy, that is when separate subtrees are constrained to be mapped to separate subtrees (Zhang, 1996).

For the general case we have to resort to non-linear search algorithms. For instance, Pelillo et al. (1999) transform the tree isomorphism problem into a single max clique problem, a technique already used for the generic graph isomorphism problem. They use relaxation labeling to obtain a maximal solution to the max clique problem, and, with it, a maximal tree match.

We draw a number of observations from this review of the relevant literature. First, we see that the computation of the unordered tree edit distance still presents a computational bottleneck. Most of the work reported in the literature investigates the simpler problems of subtree isomorphism or ordered tree edit distance. These problems are addressed in both exact and approximate settings. The goal of our work is therefore to introduce a framework in which we can efficiently approximate the computation of unordered tree edit distance. The approach is as follows. We commence by providing a divide and conquer method for the maximum common subtree by searching for maximal cliques of the directed association graph. With this representation to hand, we follow Bomze et al. (2000) and use a variant of the Motzkin Straus theorem to convert the maximum weighted clique problem into a quadratic programming problem which can be solved by relaxation labeling. The new tree-matching method is evaluated on the problem of shock-tree matching.

2. Inexact tree matching as a common substructure problem

The idea behind edit distance (Tsai and Fu, 1979) is that it is possible to identify a set of basic edit operations on nodes and edges of a structure, and to associate with these operations a cost. The edit distance is found by searching for the sequence of edit operations that will make the two graphs isomorphic with one another and which has minimum cost. The set of edit operations can be problem specific, but a common choice is:

- *node removal:* remove a node and link the children to its parent,
- *node insertion:* the dual of node removal,
- *node relabel:* change the label associated with a node.

There is a strong connection between the computation of maximum common subtree and the tree edit distance. In (Bunke and Kandel, 2000) Bunke showed that, under certain constraints applied to the edit-cost function, the MCSP and the graph edit distance problem are computationally equivalent. This is not directly true for trees, because of the added constraint that a tree must be connected. But, extending the concept to the common edited subtree, we can use common

substructures to find the minimum cost edited tree isomorphism.

The constraint on the edit-cost function proposed by Bunke in (Bunke and Kandel, 2000) is that the cost of deleting and reinserting the same element with a different label is not greater than the cost of relabeling it. In this way we can find an optimal edit sequence without the need for a relabel operation. We will assume a similar constraint applies to tree edit cost.

Hierarchical graphs have an order relation induced by paths: given two nodes a and b, (a,b) belongs to this relation if and only if there is a path from a to b. When the directed graph is acyclical, this relation can be shown to be an (irreflexive) order relation. The requirement that matches respect this relation and that the edited trees be connected, prevents us from applying Bunke's result directly to tree matching and the search for a common subgraph.

With the constraint described above on relabel cost, we are left with only node removal and node insertion operations to be performed on the data tree. Since a node insertion on the data tree is dual to a node removal on the model tree, we can further reduce the number of operations to be performed to only node removal, as long as we perform the operations on both trees. This allows us to use only structure reducing operations. This, in turn, means that the optimal matching is completely determined by the subset of nodes left after the minimum edit sequence. Hence, we can pose the edit distance problem as a particular substructure isomorphism problem. Since node removal operations respect the order relation implicit in the hierarchy, we can reduce the substructure isomorphism problem into subproblems in a divide and conquer approach. This approach derives from a number of observations. First, given two trees T_1 and T_2, there are two subtrees T_1' and T_2' rooted at nodes v and v' such that the matching that minimizes edit distance between those two nodes is equivalent to that obtained with the original trees. Hence the best match between T_1 and T_2 is equivalent to the best match given the association of root nodes (v,v'). Furthermore, this match can be found by examining only descendents of v and v'.

If we can express the best match given the association of root nodes (v,v') as a function of lower level matches of the descendants of v and v', we can build the solution to our matching problem bottom up. This approach immediately gives us a divide and conquer solution to the maximum common subtree problem. We can solve the matching problem rooted at v and v' given the solutions of rooted at the children $w_1,\ldots,w_n$ of v and $w_1',\ldots,w_n'$ of v'. This solution is obtained by solving a bipartite match problem: The nodes of the bipartite matching problem are children $w_1,\ldots,w_n$ of v and children $w_1',\ldots,w_n'$ of v'. The weight of the edge connecting w_i with w_j' is the cardinality of the match of between the subtrees rooted at w_i and w_j' respectively. It is clear that the edges in the maximum bipartite match represent the optimal correspondences between the children of v and the children of v' and that the cardinality of the match rooted at v, v' is equal to the value of the maximum bipartite match plus one (the sum of the cardinality of the matches rooted at the corresponding children plus one since we are adding v and v'). In the following sections we will introduce a similar divide and conquer approach to the tree edit distance problem.

2.1. Editing the transitive closure of a tree

In this section we show the relations between the graph theoretic concept of transitive closure of a DAG and the edit distance. An immediate remark is that for each node removal operation E_v by removing node v from the tree t, we can define the corresponding edit operation $\mathscr{E}_v$ on the closure $\mathscr{C}(t)$ of the tree t. In both cases the edit operation removes the node v, all the incoming edges, and all the outgoing edges. It is important to note that the transitive closure operation and the node removal operation commute, that is we have:

Lemma 1. $\mathscr{E}_v(\mathscr{C}(t)) = \mathscr{C}(E_v(t))$

Proof. If a node is in $\mathscr{E}_v(\mathscr{C}(t))$ it is clearly also in $\mathscr{C}(E_v(t))$. What is left is to show that an edge (a,b) is in $\mathscr{E}_v(\mathscr{C}(t))$ if and only if it is in $\mathscr{C}(E_v(t))$.

If (a,b) is in $\mathscr{C}(E_v(t))$ then neither a nor b is v and there is a path from a to b in $E_v(t)$. Since the

edit operation E_v preserves connectedness and the hierarchy, there must be a path from a to b in t as well. This implies that (a,b) is in $\mathscr{C}(t)$. Since neither a nor b is v, the operation $\mathscr{E}_v$ will not delete (a,b). Thus (a,b) is in $\mathscr{E}_v(\mathscr{C}(t))$.

If (a,b) is in $\mathscr{E}_v(\mathscr{C}(t))$, then it is also in $\mathscr{C}(t)$, because $\mathscr{E}_v(\mathscr{C}(t))$ is obtained from $\mathscr{C}(t)$ by simply removing a node and some edges. This implies that there is a path from a to b in t. This, in turn, implies that there is a path from a to b in $E_v(t)$ as well. Thus (a,b) is in $\mathscr{C}(E_v(t))$. Since (a,b) is in $\mathscr{E}_v(\mathscr{C}(t))$, both a and b must be nodes in $\mathscr{E}_v(\mathscr{C}(t))$ and, thus, neither can be v.

We call a subtree s of $\mathscr{C}(t)$ *obtainable* if for each node v of s if there cannot be two children a and b so that (a,b) is in $\mathscr{C}(t)$. In other words, s is obtainable if and only if there is no path from a to b in t for every two siblings nodes a and b. We can, now, prove the following: □

Theorem 1. *A tree $\hat{t}$ can be generated from a tree t with a sequence of node removal operations if and only if $\hat{t}$ is an obtainable subtree of the DAG $\mathscr{C}(t)$.*

Proof. Let us assume that there is an edit sequence $\{E_{v_i}\}$ that transforms t into $\hat{t}$, then, by virtue of the above lemma, the dual edit sequence $\{\mathscr{E}_{v_i}\}$ transforms $\mathscr{C}(t)$ into $\mathscr{C}(\hat{t})$. By construction we have that $\hat{t}$ is a subtree of $\mathscr{C}(\hat{t})$ and $\mathscr{C}(\hat{t})$ is a subgraph of $\mathscr{C}(t)$. Thus $\hat{t}$ is a subtree of $\mathscr{C}(t)$. Furthermore, since the node removal operations respect the hierarchy, $\hat{t}$ is a consistent subtree of $\mathscr{C}(t)$.

To prove the converse, assume that $\hat{t}$ is a consistent subtree of $\mathscr{C}(t)$. If (a,b) is an edge of $\hat{t}$, then it is also an edge on $\mathscr{C}(t)$, i.e., there is a path from a to b in t and we can define a sequence of edit operations $\{E_{v_i}\}$ that removes any node between a and b in such a path. Showing that the nodes $\{v_i\}$ deleted by the edit sequence cannot be in $\hat{t}$ we show that all the edit operations defined this way are orthogonal. As a result they can be combined to form a single edit sequence that solves the problem.

Let v in $\hat{t}$ be a node in the edited path and let p be the minimum common ancestor of v and a in $\hat{t}$. Furthermore, let w be the only child of p in $\hat{t}$ that is an ancestor of v in $\hat{t}$ and let q be the only child of p in $\hat{t}$ that is an ancestor of a in $\hat{t}$. Since a is an ancestor of v in t, an ancestor of v can be either a descendant of a, an ancestor of a, or a itself. This means that w has to be in the edited path. Were this not the case, then w had to be a or an ancestor of a against the hypothesis that p is the minimum common ancestor of v and a. Since q is an ancestor of a in t and a is an ancestor of w in t, it follows that q is an ancestor of w in t. On the other hand, q and w are siblings in $\hat{t}$, against the hypothesis that $\hat{t}$ is consistent.

Using this result, we can show that the minimum cost edited tree isomorphism between two trees t and t' is a maximum common consistent subtree of the two DAGs $\mathscr{C}(t)$ and $\mathscr{C}(t')$, provided that the node removal cost is uniform. The result can be extended to non-uniform cost, but in this paper we will restrict our scope to the uniform cost case.

The minimum cost edited tree isomorphism is a tree that can be obtained from both the model tree t and the data tree t' with node removal operations. By virtue of the theorem above, this tree is an obtainable subtree of both $\mathscr{C}(t)$ and $\mathscr{C}(t')$. The tree must be generated with minimum combined edit cost, and, since the node removal cost is uniform, this implies the minimum number of nodes removed. Hence the common consistent subtree must retain most nodes, i.e., it must be the maximum common consistent subtree of the two DAGs. □

2.2. Cliques and common obtainable subtrees

In this section we show how to use these results to induce a divide and conquer approach to edited tree matching. Given two trees t and t' to be matched, we calculate the transitive closures $\mathscr{C}(t)$ and $\mathscr{C}(t')$ and look for a common obtainable tree that induces the optimal matches. The maximum such tree corresponds to the maximum common obtainable subtree of $\mathscr{C}(t)$ and $\mathscr{C}(t')$.

For each pair of nodes v and w of the two trees to be matched, we divide the problem into a maximum common obtainable subtree rooted at v and w. That is, we fix the matching of v to w and we look for the maximum edited subtree common

to the two subtrees rooted at v and w. We show that, given the cardinality of the subtree rooted at each child of v and w, we can transform the search for the maximum common substructure into the search for a max weighted clique. Solving this problem for each pair of nodes, and looking for the maximum among each node pair, we can find the isomorphism linked to the minimum edit distance.

Let us assume that we know the cardinality of the isomorphism for every descendent of v and w. We want to find the set of siblings with greatest total cardinality. To do this we make use of a derived structure similar to the association graph introduced by Barrow and Burstall (1976). The nodes of this structure are pairs drawn from the Cartesian product of the descendents of v and w. Each pair corresponds to a particular association between a node in one tree to a node in the other. We connect two such associations if and only if there is no inconsistency between the two association and the corresponding subtree is obtainable. That is, we connect nodes (p,q) and (r,s) if and only if there is no path connecting p and r in t and there is no path connecting q and s in t'. Furthermore, we assign to each association node (a,b) a weight equal to the cardinality of the maximum common obtainable subtree rooted at a and b. The maximum weight clique of this graph is the set of consistent siblings root of the common substructures with maximum total cardinality. That is, the clique identifies set of children of nodes v and w that guarantee the maximum isomorphism. The cardinality of the maximum common consistent subtree rooted at v and w will be the weight of the isomorphisms rooted at the children of v and w plus the contribution of v and w, that is the weight of the clique plus one. Given a solution for the maximum weight clique problem, we can build the solution to our isomorphism problem bottom up.

2.3. Heuristics for the maximum weighted clique

In 1965, Motzkin and Straus (1965) showed that the (unweighted) maximum clique problem can be reduced to a quadratic programming problem on the n-dimensional simplex $\Delta = \{\boldsymbol{x} \in \mathbb{R}^n | x_i \geqslant 0 \text{ for all } i = 1 \ldots n,\ \sum_i x_i = 1\}$, where x_i are the components of the vector x. With this reduction, maximal cliques could be put in correspondence with local maxima of a quadratic function. The result has since been generalized to the weighted case (Bomze et al., 2000; Gibbons et al., 1997). Under such generalization, the quadratic problem is:

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{x}) = \boldsymbol{x}^{\mathrm{T}} C \boldsymbol{x} \\ &\text{subject to} && \boldsymbol{x} \in \Delta \end{aligned} \tag{1}$$

where the elements of the matrix matrix $C = (c_{ij})_{i,j \in V}$ are defined as

$$c_{ij} = \begin{cases} \dfrac{1}{2w_i} & \text{if } i = j, \\ k_{ij} \geqslant c_{ii} + c_{jj} & \text{if } (i,j) \notin E, i \neq j, \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

where w_i is the weight associated with node i.

Given a set of nodes S and its *characteristic* vector $\boldsymbol{x}^S$ defined as

$$x_i^S = \begin{cases} \dfrac{w(i)}{\sum_{j \in S} w(j)} & \text{if } i \in S, \\ 0 & \text{otherwise,} \end{cases}$$

then S is a maximum (maximal) weight clique if and only if $\boldsymbol{x}^S$ is a global (local) minimizer for the quadratic problem. Furthermore, if x is a minimum then it is the characteristic vector for a set of nodes.

To solve the quadratic problem we transform it into the equivalent maximization problem:

$$\begin{aligned} &\text{maximize} && \boldsymbol{x}^{\mathrm{T}} (\gamma \boldsymbol{e}\boldsymbol{e}^{\mathrm{T}} - C) \boldsymbol{x} \\ &\text{subject to} && \boldsymbol{x} \in \Delta, \end{aligned} \tag{3}$$

where $\boldsymbol{e} = (1, \ldots, 1)^{\mathrm{T}}$ is the vector with every component equal to 1 and γ is a positive scaling constant. Given the equivalent maximization formulation, we follow Bomze et al. (2000) and use relaxation labeling as a local maximizer for the problem.

Relaxation labeling is a evidence combining process developed in the framework of constraint satisfaction problems. Its goal is to find a classification p that satisfies pairwise constraints and interactions between its elements. The process is governed by the update rule

$$p_i^{t+1}(\lambda) = \frac{p_i^t(\lambda) q_i^t(\lambda)}{\sum_\mu p_i^t(\mu) q_i^t(\mu)},$$

where the compatibility component is

$$q_i(\lambda) = \sum_{j=1}^{n} \sum_{\mu=1}^{m} r_{ij}(\lambda, \mu) p_j(\mu),$$

where the $r_{i,j}(\lambda, \mu)$ are coefficients that represent mutual consistency between hypothesis $p_i(\lambda)$ and $p_j(\mu)$.

In (Pelillo, 1997), Pelillo showed that the function

$$A(\boldsymbol{p}) = \sum_{i\lambda} p_i(\lambda) q_i(\lambda),$$

is a Lyapunov function for the process, i.e., $A(\boldsymbol{p}^{t+1}) \geqslant A(\boldsymbol{p}^t)$, with equality if and only if $\boldsymbol{p}^t$ is stationary.

We use relaxation labeling dynamics to solve each instance of clique subproblem. For each subproblem the compatibility coefficients are initialized as $R_{(v,w)} = \gamma \boldsymbol{e}\boldsymbol{e}^{\mathrm{T}} - C$. Starting from the leaves we propagate the solutions upwards in the trees using the weight of the extracted clique to initialize the compatibility matrix of every higher level subproblem.

A common approach with relaxation labeling is to initialize the assignment vector with a uniform distribution so that there is an initial assignment close to the baricenter of the simplex. A problem with this approach is that the dimension of the basin of attraction of one maximal clique grows with the number of nodes in the clique, while with our problem decomposition the wider cliques are the ones that map nodes at lower levels. As a result the solution will be biased towards matches that are very low on the graph, even if these matches require cutting a lot of nodes and are, thus, less likely to give an optimum solution.

A way around this problem is to choose an initialization that assigns a higher initial likelihood to matches that are higher up in the subtree. In our experiments we decided to initialize the probability of the association (a, b) for the subproblem rooted at (u, v) as

$$p_{(u,v)}(a, b) = \mathrm{e}^{-(d_a + d_b + \epsilon)},$$

where d_a is the depth of a with respect to u, d_b is the depth of b with respect to v, and ϵ is a small perturbation. Of course, we then renormalize $\boldsymbol{p}_{(u,v)}$ to ensure that it is still in the simplex.

3. Experimental results

We evaluate the new tree-matching method on the problem of shock-tree matching.

Here we follow Zucker, Siddiqi, and others, by labeling points on the skeleton using so-called shock-labels (Siddiqi et al., 1999). According to this taxonomy of local differential structure, there are different classes associated with the behavior of the radius of the maximal circle inscribed in the shape. The so-called shocks distinguish between the cases where the local maximal circle has maximum radius, minimum radius, constant radius or a radius which is strictly increasing or decreasing. We abstract the skeletons as trees in which the level in the tree is determined by the time of shock formation (Siddiqi et al., 1999). The later the time of formation, and hence their proximity to the center of the shape, the higher the shock in the hierarchy. While this temporal notion of relevance can work well with isolated shocks (maxima and minima of the radius function), it fails on monotonically increasing or decreasing shock groups. To give an example, a protrusion that ends on a vertex will always have the earliest time of creation, regardless of its relative relevance to the shape.

Fig. 1 provides an illustration of the extraction of shock graphs from binary images. In the left

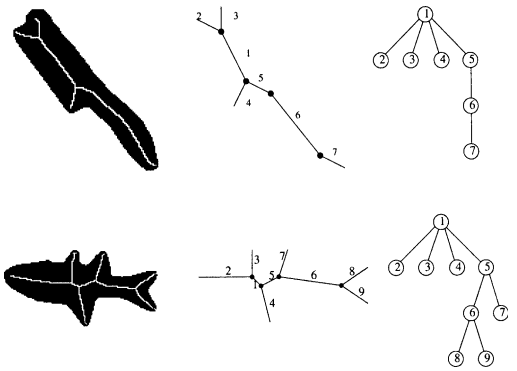

Fig. 1. Two shapes with their skeletons and shock trees.

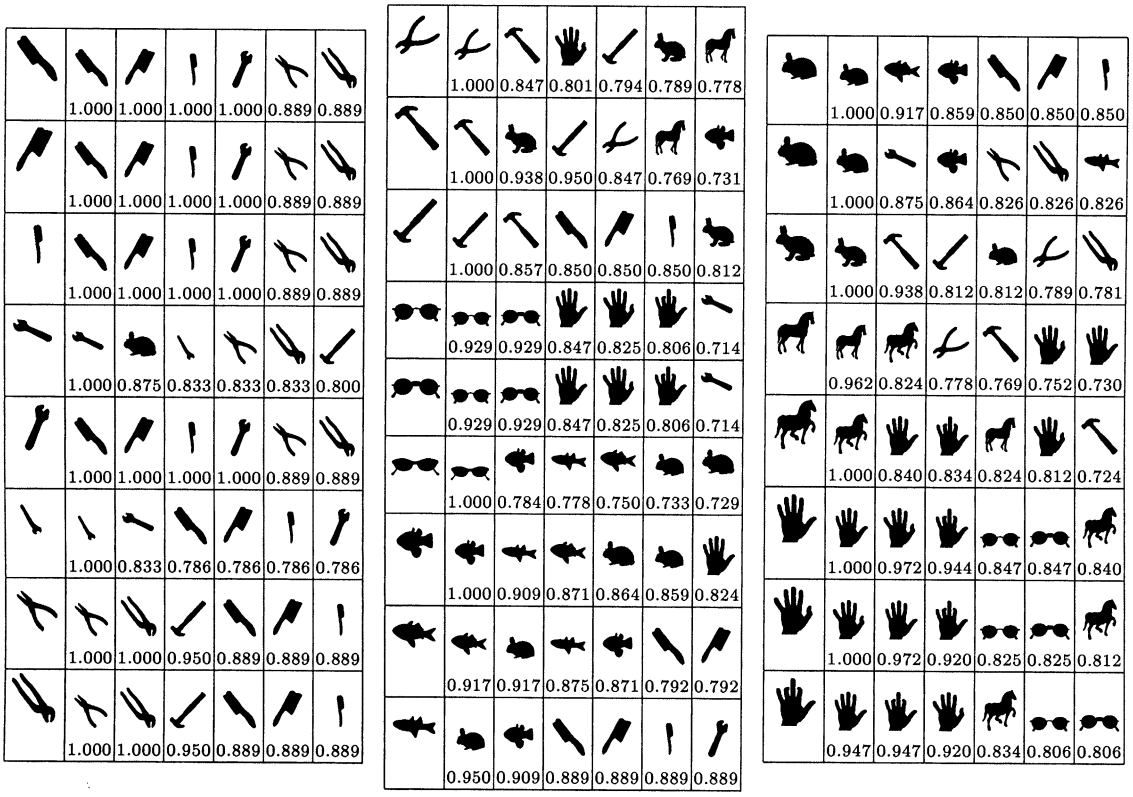

Fig. 2. Shapes and their top six matches in the shape database.

hand column, we show the binary images with the skeleton superimposed. The center column shows the branches of the skeleton with the segment labels appended. Finally, the third column shows the resulting shock-trees with the nodes labeled with the segment number.

In Fig. 2 we show 25 shapes from our shape database and their top six matches. The similarity index we show is the average ratio of matched nodes over total nodes. Our edit distance approach compares favorably against the similar shock graph experiments in (Pelillo et al., 1999). In particular our approach seems to capture better the perceptual similarity between the shapes of horses and hands. In fact, initial results with multidimensional scaling show that the different shape classes form distinct pairwise clusters, and similar shape classes are in close proximity to one-another (Luo et al., 2001).

4. Sensitivity study

To augment these real world experiments, we have performed a sensitivity analysis. The aim is to characterize the effects of structural errors.

Our analysis tests the capability of the method to cope with structural modification caused by node removal. To do this we remove an increasing fraction of nodes from a randomly generated tree. We then match the modified tree against its unedited version. Since we remove nodes only from one tree, every node of the edited tree will match against a node of the unedited version. Hence, the optimum number of nodes matched is equal to the cardinality of the edited tree.

We performed the experiments on trees with 10, 15, 20, 25, and 30 nodes. For each experimental run we used 10 randomly generated trees. The procedure for generating the random trees was as

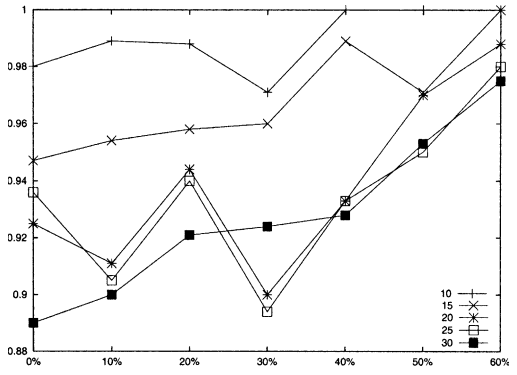

Fig. 3. Sensitivity analysis. The plot shows the ratio of computed over optimum cardinality of the maximum edited subtree under increasing structural deformation.

follows: we commence with an empty tree (i.e., one with no nodes) and we iteratively add the required number of nodes. At each iteration nodes are added as children of one of the existing nodes. The parents are randomly selected with uniform probability from among the existing nodes. This procedure will tend to generate trees in which the branch ratio is highest closest to the root. This is quite typical of real-world situations, since shock trees tend to have the same property. The fraction of nodes removed was varied from 0% to 60%. Fig. 3 plots the average ratio of the computed cardinality of the maximum common edited subtree to the optimal value as a function of fraction of nodes removed and the size of the original tree.

The main conclusion that can be drawn from these two plots is as follows. First, the effect of increasing structural error is to cause a systematic underestimation of the weighted edit distance. The different curves all exhibit a minimum value of the ratio. The reason for this is that the matching problem becomes trivial as the trees are decimated to extinction.

5. Conclusions

In this paper we have investigated a purely structural approach to tree matching. We based the work on the tree edit distance framework constraining it to uniform edit cost. We show that any tree obtained with a sequence of cut operation is a subtree of the transitive closure of the original tree. Furthermore, we show that the necessary condition for any subtree to be a solution can be reduced a clique problem in a derived structure. Using this idea we transform the tree edit distance problem into a series of maximum weight clique problems and then we use relaxation labeling to find an approximate solution.

In a set of experiments we apply this algorithm to match shock graphs, a graph representation of the morphological skeleton. The results of these experiments are very encouraging, showing that the algorithm is able to match similar shapes together. Moreover, we provide some sensitivity analysis of the method.

References

Barrow, H.G., Burstall, R.M., 1976. Subgraph isomorphism, matching relational structures and maximal cliques. Informat. Process. Lett. 4, 83–84.

Bomze, I.M., Pelillo, M., Stix, V., 2000. Approximating the maximum weight clique using replicator dynamics. IEEE Trans. Neural Networks 11.

Bunke, H., Kandel, A., 2000. Mean and maximum common subgraph of two graphs. Pattern Recognition Lett. 21, 163–168.

Eshera, M.A., Fu, K.-S., 1986. An image understanding system using attributed symbolic representation and inexact graph-matching. IEEE PAMI 8, 604–618.

Gibbons, L.E. et al., 1997. Continuous characterizations of the maximum clique problem. Math. Oper. Res. 22, 754–768.

Luo, B. et al., 2001. A probabilistic framework for graph clustering. CVPR, vol. 1. pp. 912–919.

Motzkin, T.S., Straus, E.G., 1965. Maxima for graphs and a new proof of a theorem of Turán. Can. J. Math. 17, 533–540.

Pelillo, M., 1997. The dynamics of relaxation labeling process. J. Math. Imaging Vis. 7, 309–323.

Pelillo, M., Siddiqi, K., Zucker, S.W., 1999. Matching hierarchical structures using association graphs. IEEE PAMI 21, 1105–1120.

Siddiqi, K., Shokoufandeh, A., Dickinson, S.J., Zucker, S.W., 1999. Shock graphs and shape matching. Internat. J. Comput. Vision 35, 13–32.

Tai, K.-C., 1979. The tree-to-tree correction problem. J. Assoc. Comput. Machinery 26, 422–433.

Tsai, W.H., Fu, K.S., 1979. Error-correcting isomorphism of attributed relational graphs for pattern analysis. Trans. System Man Cybernet. 9.

Wang, J.T.L., Zhang, K., Chirn, G., 1994. The approximate graph matching problem. ICPR. pp. 284–288.

Zhang, K., 1996. A constrained edit distance between unordered labeled trees. Algorithmica 15, 205–222.

Zhang, K., Shasha, D., 1989. Simple fast algorithms for the editing distance between trees and related problems. SIAM J. Comput. 18, 1245–1262.

Zhang, K., Statman, R., Shasha, D., 1992. On the editing distance between unordered labeled trees. Informat. Process. Lett. 42, 133–139.