Matching Hierarchical Structures for Shape Recognition

Andrea Torsello

Submitted for the degree of Doctor of Philosophy

Department of Computer Science
THE UNIVERSITY of York

May 2004

Abstract

In this thesis we aim to develop a framework for clustering trees and representing and learning a generative model of graph structures from a set of training samples. The approach is applied to the problem of the recognition and classification of shape abstracted in terms of its morphological skeleton. We make five contributions. The first is an algorithm to approximate tree edit-distance using relaxation labeling. The second is the introduction of the tree union, a representation capable of representing the modes of structural variation present in a set of trees. The third is an information theoretic approach to learning a generative model of tree structures from a training set.

While the skeletal abstraction of shape was chosen mainly as a experimental vehicle, we, nonetheless, make some contributions to the fields of skeleton extraction and its graph representation. In particular, our fourth contribution is the development of a skeletonization method that corrects curvature effects in the Hamilton-Jacobi framework, improving its localization and noise sensitivity. Finally, we propose a shape-measure capable of characterizing shapes abstracted in terms of their skeleton. This measure has a number of interesting properties. In particular, it varies smoothly as the shape is deformed and can be easily computed using the presented skeleton extraction algorithm.

Each chapter presents an experimental analysis of the proposed approaches applied to shape recognition problems.

Contents

List of Figures

Glossary of Symbols

$\vec{C}(t)$	Time evolution of the boundary of a shape.
∇f	Gradient of the function f.
$\nabla \cdot \vec{F}$	Divergence of the field $\vec{F}$.
$D(p)$	Distance of a point p to the boundary of a shape.
$\|A\|$	Area of region A.
$\|\|l\|\|$	Length of segment l.
$\Phi_A(\vec{F})$	Flux of the field $\vec{F}$ through region A.
$\kappa(p)$	Curvature at point p of the inward evolving boundary curve $\vec{C}$.
$N\Phi_A(\vec{F})$	Normalized flux of the field $\vec{F}$ through region A.
$\Omega(t)$	Transitive closure of the tree t.
ρ	Density field.
$v \rightsquigarrow u$	Node v is a parent of node u.
$v \dashrightarrow u$	Node v is an ancestor of node u.
w_v	Weight associated with node v.
$d(t_1, t_2)$	Edit-distance between tree t_1 and tree t_2.
r_v	Cost of removing node v.
m_{uv}	Cost of matching node u to node v.
$\mathcal{M}$	Set of matches between two trees.
$\mathcal{C}$	Set of correspondences from the training set to the tree union models.
$\mathcal{U}(\mathcal{M})$	Utility of a set of matches $\mathcal{M}$.

T	A set of trees.
$\mathcal{D}$	The complete training set of trees.
$\mathcal{T}$	Tree union.
$\mathcal{H}$	Generative tree model.
θ_i	Sampling probability for node i.
$\mathcal{L}$	Log-likelihood function.
LL	Description length.
KL	Kullback-Leiber divergence.
I	Entropy.

Acknowledgments

I wish to thank Dr. Richard Wilson and Prof. Horst Bunke for their assessment of this work and their valuable feedback and discussion. My sincere thanks also go to the good friends I made in York, with whom I shared good times and long working nights, and particularly to Fabio, to whom I owe a lot. I wish there was a way to express my gratitude to my parents, who have always encouraged me and supported me in all my achievements. A very special and loving thank you goes to Vera, who has put up with my vagrant life for so long. Finally, last but certainly not least, I want to express my deepest gratitude to Edwin Hancock. He has not only been an attentive supervisor and an insightful mentor, but also a person I feel extremely lucky to consider my friend.

Declaration

I declare that all the work in this thesis is solely my own except where attributed and cited to another author. Some of the material in the following chapters has been previously published, a full list of the publications is listed in the next page.

List of Publications

2003

- A. Torsello and E. R. Hancock, Computing approximate tree edit distance using relaxation labeling. *Pattern Recognition Letters*, 24:1089–1097, 2003.

- A. Torsello and E. R. Hancock, Curvature Correction of the Hamilton-Jacobi Skeleton. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. I, pp. 828–834, 2003.

- A. Torsello and E. R. Hancock, Curvature Dependent Skeletonization. In *IEEE International Conference on Image Processing*, Vol. I, pp. 337–340, 2003.

- A. Torsello and E. R. Hancock, Curvature Dependent Skeletonization. In *13th Scandinavian Conference on Image Analysis*, Springer-Verlag Berlin, LNCS 2749, pp. 200–207, 2003.

- A. Torsello and E. R. Hancock, Learning mixture of Tree-Unions by Minimizing Description Length. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, Springer-Verlag Berlin, LNCS 2683, pp. 130–146, 2003.

- A. Torsello and E. R. Hancock, Tree Edit Distance from Information Theory. In *4th IAPR-TC15 Workshop on Graph-based Representations in Pattern Recognition*, Springer-Verlag Berlin, LNCS 2727, pp. 71–82, 2003.

2002

- A. Torsello and E. R. Hancock, Learning Structural Variations in Shock Trees. In *Joint International Workshops on Structural, Syntactic, and Statistical Pattern Recognition*, Lecture Notes in Computer Science, Springer, LNCS2396, pp. 113–122, 2002.

- A. Torsello and E. R. Hancock, Matching and Embedding through Edit-Union of Trees. In *7th European Conference on Computer Vision*, Vol. III, Lecture Notes in Computer Science, Springer, Copenhagen, Denmark, LNCS2352, pp.822-836, 2002.

- A. Torsello and E. R. Hancock, Shape-Space from Tree-Union. In *16th International Conference on Pattern Recognition, IEEE Computer Society Press*, Vol.I, pp.188–191, 2002.

2001

- A. Torsello, B. Luo, A. Robles-Kelly, R. C. Wilson, and E. R. Hancock, A Probabilistic Framework for Graph Clustering. In *IEEE conference on Computer Vision and Pattern Recognition*, Vol. I, pp. 912–919, 2001.

- A. Torsello and E. R. Hancock, A Skeletal Measure of 2D Shape Similarity. In *Visual Form*, Lecture Notes in Computer Science, Springer, LNCS2059, pp.260–271, 2001.

- B. Luo, A. Robles-Kelly, A. Torsello, R. C. Wilson, and E. R. Hancock, Clustering Shock Trees. In *3rd IAPR-TC15 Workshop on Graph-based Representations in Pattern Recognition*, pp.217–226, 2001.

- A. Torsello and E. R. Hancock, Computing approximate tree edit distance using relaxation labeling. In *3rd IAPR-TC15 Workshop on Graph-based Representations in Pattern Recognition*, pp.125–136, 2001.

- B. Luo, A. Robles-Kelly, A. Torsello, R. C. Wilson, E. R. Hancock, Discovering Shape Categories by Clustering Shock Trees. In *Computer Analysis of Images and Patterns*, Lecture notes in Computer Science, Springer, LNCS2124, pp.151-160, 2001.

- A. Torsello and E. R. Hancock, Efficiently computing weighted tree edit distance using relaxation labeling. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pp. 438–453, 2001.

- B. Luo, A. Robles-Kelly, A. Torsello, R. Wilson and E. R. Hancock, Learning Shape Categories by Clustering Shock Trees. In *IEEE Signal Processing Society International Conference on Image Processing*, Vol. III, pp. 672–675, 2001.

Submitted Papers

- A. Torsello and E. R. Hancock, A skeletal measure of 2D shape similarity. Submitted to *Computer Vision and Image Understanding.*

- A. Torsello and E. R. Hancock, Curvature Correction of the Hamilton-Jacobi Skeleton. Submitted to *IEEE Trans. on Pattern Analysis and Machine Intelligence.*

- A. Torsello and E. R. Hancock, Learning Mixtures of Weighted Tree-Unions by Minimising Description Length. Submitted to *IEEE Trans. on Pattern Analysis and Machine Intelligence*.

Chapter 1

Introduction

1.1 The Problem

Graph-based representations have been used with considerable success in computer vision in the abstraction and recognition of object shape and scene structure. Concrete examples include the use of shock graphs to represent shape-skeletons [42, 81], the use of trees to represent articulated objects [37, 101] and the use of aspect graphs for 3D object representation [22]. The attractive feature of structural representations is that they concisely capture the relational arrangement of object primitives, in a manner which can be invariant to changes in object viewpoint. Using this framework we can transform a recognition problem into a relational matching problem. Recognition tasks are typically performed by comparing a structure extracted from an image to a structural prototype. The problem of how to measure the similarity or distance of pictorial information using graph abstractions has been a widely researched topic for over twenty years, yet the topic of how to use them to learn shape-classes has not received significant attention. The importance of the topic stems from the fact that it is a fundamental tool for learning the class structure of data abstracted in terms of relational graphs.

One of the reasons for limited progress in this area has been the lack of algorithms suitable for clustering relational structures. In particular, the problem has proved elusive to conventional central clustering techniques. This is due to the fact central clustering algorithms are designed to work on vectorial data, and graph representations cannot be easily embedded in a vector space. There are two reasons for this. First, there is no canonical ordering for the nodes or edges of a graph. Hence, before a vector-space can be constructed, the correspondences between nodes must be established. Second, structural variations in graphs manifest themselves as differences in the numbers of nodes and edges. As a result, even if a vector mapping can be established, the vectors will be of variable length.

A possible approach is to estimate a representation of the structural variation present in a set. However, despite the many advantages and attractive features of graph representations, the methodology available for representing structural variation is relatively limited. As a result, the process of constructing embedding spaces which capture the modes of structural variation for sets of graphs has proved to be elusive. The difficulty derives from the fact that in order to construct a representation for a distribution of graphs, the correspondences must be at hand. Hence, there is a chicken and egg problem in structural learning. Before the structural model can be learned, the correspondences with it must be available, and yet the model itself must be to hand to locate correspondences.

1.2 Our Goal

The principal aim of the research reported in this thesis is to develop a framework for clustering trees and learning the modes of structural variation present in a set of training samples. We propose and analyze two approaches to the problem. The first of these is to use the distance measures obtained applying standard graph-matching, and then apply pairwise clustering to the resulting set of graph distances. This approach, however, does not result in an ordering of the graphs that has metrical significance under structural variations due to graded shape-changes. The second approach is to learn structural representations from sets of training examples. To this end, we introduce the tree union, a representation which is capable of representing the modes of structural variation present in a set of trees. We adopt an information theoretic approach to learn a generative tree-union model from a training set.

We apply the proposed clustering approaches to the problem of recognizing shape abstracted in terms of shock graphs. Broadly speaking the representation and recognition of 2D shapes based on the shock representation is a three stage process. Firstly, the skeleton must be computed from the available shape boundary information. Secondly, the skeleton must be reduced to a concise representation that captures the differential structure of the original boundary. The final step is the matching and clustering of the resulting shape representation.

While the skeletal abstraction of shape was chosen mainly as a experimental vehicle, we, nonetheless, do make a number of contributions to the problem of extracting and representing the morphological skeleton. In particular, we show how to eliminate curvature-dependent error in the Hamilton-Jacobi skeleton. We also devise a shape measure capable of characterizing shapes abstracted in terms of their skeleton.

1.3 Thesis Overview

After defining the problem domain and the goals of the thesis in Chapter 1, in Chapter 2 we give a brief review of the relevant literature. The review covers the literature on skeleton extraction and representation, graph matching, and, finally, graph clustering and embedding.

The research work presented in this thesis is roughly divided into three sections. The first section, which contains Chapters 3 and 4, deals with the extraction of the skeleton and its abstraction in terms of graphs.

Chapter 3 presents an algorithm for extracting the skeleton, while Chapter 4 presents a shape-measure that can be used to compare shapes abstracted in terms of their skeleton. This measure has a number of interesting properties. In particular, it varies smoothly as the shape is deformed and can be easily computed using the skeleton extraction algorithm presented in Chapter 3.

The second section contains only Chapter 5 and deals with the tree matching problem. This chapter presents an algorithm for computing approximate tree edit-distance. It shows that the tree edit-distance problem can be reduced to a series of maximum weighted clique problems and adopts a continuous optimization approach to approximate them. In this chapter we also show how the computed edit-distance, together with a pairwise clustering algorithm, can be used to perform unsupervised classification of a set of trees.

The third section deals with the problem of defining and leaning a representation of a class of tree structures from a set of training samples. Chapter 6 presents the tree union, a structural archetype capable of describing the modes of variation present in a class of trees. Chapter 7 deals with the problem of learning these class representations from a set of trees. Here we adopt an information theoretic approach to the structural learning problem and construct the archetypes so as to minimize a description length criterion.

Finally, Chapter 8 draws some conclusions and identifies some directions for further work.

Chapter 2

Literature Review

In this chapter we will present a review of the literature relevant to the work presented in the thesis. The review covers a) the extraction of the morphological skeleton of 2D shape, b) its abstraction in terms of graphs, c) graph matching, and d) clustering and embedding of graph representations.

2.1 Skeleton Extraction

The skeletal abstraction of 2D and 3D objects has proved to be an alluring yet highly elusive goal for over 30 years in shape analysis. The topic is not only important in image analysis, where it has stimulated a number of important developments including the medial axis transform and iterative morphological thinning operators, but is also an important field of investigation in differential geometry and biometrics, where it has led to the study of the so-called morphological skeleton [7].

The morphological skeleton of a shape is defined as the set of singularities in the inward evolution of the boundary with constant velocity. The dynamics of the boundary motion is described by the eikonal equation. This is a partial differential equation that governs the motion of a wave-front through a medium. In the case of a uniform medium the equation is

$$\frac{\partial}{\partial t}\vec{C}(t) = \alpha\vec{N}(t), \tag{2.1}$$

where $\vec{C}(t) : [0, s] \rightarrow \mathbb{R}^2$ is the equation of the front at time t, $\vec{N}(t) : [0, s] \rightarrow \mathbb{R}^2$ is the equation of the normal to the wave front in the direction of motion, and α is the propagation speed. As the wave front evolves, opposing segments collide, generating a singularity.

Given the importance of skeletal representations, the quest for reliable and efficient ways of computing skeletal shape descriptors has been the subject of sustained activity [1, 45, 56, 87, 83]. The problem is a complex and elusive one because it is based on the detection of singularities in the inward evolution of the shape boundary [7]. The available methods for extracting the skeleton can be divided into three broad categories. The first class of methods are those that involve the use of *marching front techniques* which simulate the grassfire transform. These methods are concerned with iteratively propagating the boundary front over time. Singularities in the simulated evolution of the front indicate the locations of the skeleton. This class of algorithms can be further divided into a) thinning methods [1, 2], methods where layers of pixels are sequentially pealed from the shape like the skin of an onion, and b) curve evolution methods [45, 87], where curve descriptors such as splines or snakes are transformed according to the eikonal equation. Thinning algorithms have a clear advantage in terms of simplicity. However, their performance is not invariant under Euclidean transformation. Curve evolution methods, on the other hand, are invariant under Euclidean transformation, but require a functional description of the boundary curve. Concrete examples include the use of second order or higher order curves and splines [45], or local descriptors such as line segments or circular arc segments

[87]. If the shape is a binary silhouette on the image lattice, then curve evolution requires that a fit be performed to the shape-boundary, and this process not only adds to the complexity of the method, but can also be adversely affected by noise. Furthermore, the quality of the extracted boundary curve depends strongly on the reliability of the fitted curve descriptors.

A second class of skeleton extraction algorithms consists of those that rely on the relationship between the Voronoi triangulation and the skeleton [70, 56, 57]. This class is based on the property that, as the number of control points on the object boundary increases, the locus of the centers of the triangles of the corresponding Voronoi triangulation of the shape converges to the skeleton. The consequence is that as the triangulation increasingly approximates the shape boundary, then, correspondingly, the centers of the triangles increasingly approximate the skeleton. The important advantages of this approach are that it offers invariance under Euclidean transformation, that it is numerically stable, that it is fast, and that it is simple to implement. However, its major drawback is the relatively slow convergence speed of the skeleton approximation with respect to the number of control points on the boundary. Hence, this class of algorithm is the natural choice either when the shape is already triangulated (as is often the case with 3D models) or when it presents a natural triangulation, as does a polygonal object. Otherwise, the need to tessellate the shape with a large number of triangles negates the advantage of speed.

The third, and final, class of algorithms rely on the analysis of the differential structure of the boundary. An important method that falls into this class is the one resulting from an analysis of the boundary evolution dynamics using the Hamilton-Jacobi equations from classical mechanics [58]. This analysis leads to an eikonal equation which governs the boundary flow.

Whenever this flow is non-singular, the system is Hamiltonian, and, thus, conservative. However, when the system ceases to be conservative there are singularities in the flow of boundary evolution. In the Hamilton-Jacobi setting [11, 82, 83], skeletal points are detected by searching for locations where the system ceases to be Hamiltonian. The resulting skeleton search method is algorithmically simple, numerically stable, and fast. Furthermore, it works directly on the image lattice without the need to extract an intermediate curve description of the boundary.

In the first reported account of the Hamilton-Jacobi method, the analysis assumed that boundary evolution ceased to be Hamiltonian at locations where the divergence of the flow was non-zero [11, 82]. Unfortunately, this is not the case. Hence, this initial work appears to overlook the fact that the linear density of the evolving boundary front is not constant where the front is curved. The result of changes in density is that the flux is not conservative and hence the premise underpinning the skeletonization method does not hold. In a subsequent paper [83] the authors correct this oversight in the analysis by normalizing the flux by the perimeter of the integration area. The resulting normalized flux is still non-zero at non-skeletal locations. However, in the limit as the integration area shrinks to zero, the normalized flux does tend to zero at non-skeletal locations, and is negative on the skeleton itself. Unfortunately, when the integration is performed on the image lattice, the integration area is bounded from below by the pixel size and this introduces an error into the calculation of the normalized flux. Furthermore, there are locations where this error is unbounded. One way to reduce the effect of this error is to use interpolation techniques to compute the flux with sub-pixel precision [23].

2.2 Graph Representations

Once the skeleton is to hand, the next step is to devise ways of using it to characterize the original boundary shape. Most of the approaches reported in the literature opt to use a structural characterization. For instance, Zucker, Siddiqi and others have labeled points on the skeleton using so-called shock labels [85]. According to this taxonomy of local differential structure, there are different classes associated with the behavior of the radius of the bitangent circle inscribed in the shape. The so-called shocks distinguish between the cases where the local bitangent circle has maximum radius, minimum radius, constant radius or a radius which is strictly increasing or decreasing. Kimia and Giblin opt for a simpler representation which is based just on the junctions and terminations of the skeleton [88]. There are also examples in the literature of skeletal representations that are not based on the morphological skeleton. Among these are the shape axis representation of Liu and Geiger [47]. Here the skeleton is not defined using the symmetry axis, but as the mid point between two corresponding boundary points on opposite sides of the shape. Another important skeleton based representation is that used in the FORMS system [101]. In this work the medial axis is matched to a model skeleton using a branch and bound strategy.

Graph-based representations have been used with considerable success in computer vision in the abstraction and recognition of object shape and of scene structure. Concrete examples of the use of structural representations in computer vision include the use of trees to represent articulated objects [47, 37, 101] and 2D shape [42, 84, 85], and the use of aspect graphs for 3D object representation [22, 50, 21]. The attractive feature of structural representations is that they concisely capture the relational arrangement of object primitives, in a manner which can be invariant to changes in object viewpoint. The reduced memory requirement of structural descriptors makes them particularly interesting for indexing into large databases [80, 88, 73].

One of the criticisms that can be leveled at existing skeleton representations is their sensitivity to small boundary deformations and the consequent formation of so-called ligatures: segments of the skeleton that are not in correspondence with any boundary feature [4]. Although this sensitivity can be reduced via curvature dependent smoothing, it may have a significant effect on the topology of the extracted skeleton.

In order to reduce the number of spurious branches generated by local shape deformations on the border, many authors have applied some form of post-processing to the skeleton extraction, eliminating any branch that has a low relevance to the shape, according to some notion of relevance. Many features of the skeleton have been used to measure relevance [78]. Examples include the speed at which the shock propagates [7], the time of creation of the branch [85], the amount of smoothing required to eliminate the branch [53, 10], and the length of the border that generates the branch [56, 57]. Most methods are applied in a hierarchical way and can be used to determine the relevance of terminal branches only, i.e. branches that terminate on an endpoint of the skeleton. They are not applicable to internal measurements. The ratio of border length to shock length, on the other hand, maintains this informational content in the hierarchy, and does not exhibit the hierarchical bias seen on other measures.

A different approach to overcome the susceptibility of skeletal topology to noise and small deformation is to attribute the skeletal representation with some measure based on the local geometry of the shape, leaving the pruning of spurious branches to a later matching stage. Siddiqi and Zucker label the shocks generated by the eikonal equation with their time of formation

[81, 84, 85], or, which is equivalent, the distance to border. The later the time of formation, and hence the closer their proximity to the center of the shape, the higher the shock in the hierarchy. This temporal notion of relevance is based on the observation that the skeletal branches generated by noise and high frequency features are always close to the border. Unfortunately, the converse does not hold. 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. For this reason the time of formation is not an effective measure of branch relevance in the presence of sharp boundary structure or high curvature features. In order to augment the structural information, Pelillo, Siddiqi and Zucker [62] associate each node in the shock tree with the length, radii, velocities and curvatures of the corresponding shocks. Unfortunately, these attributes do not, in general, vary uniformly with continuous geometric deformation of the shape.

Blum and Nagel [8] suggested that the ratio of border length to shock length could be used, together with other measures, to characterize the shape, but the proposal was based solely on the ability of this measure to reveal whether a skeletal section is a ligature. In practice they used the measure only as a purely static measure of relevance, ignoring its properties when the shape undergoes deformation.

2.3 Graph Matching

Once the skeletal representation is to hand, then shapes may be matched by comparing their skeletons. The problem of how to measure the similarity of pictorial information which has been abstracted using graph-structures has been the focus of sustained research activity for over twenty years in the computer vision literature. Early work on the topic included Barrow and Burstall's idea [5] of locating matches by searching for maximum common subgraphs using the association graph and Shapiro and Haralick's idea [79] of locating the isomorphism that minimizes the weight of unmapped nodes. Most early approaches proposed to solve subgraph isomorphism problems adopt a tree-search algorithm [92, 79, 90, 25, 94]. Other search approaches found in the literature include genetic search [20] and Tabu search [95]. Exact search methods produce the optimum subgraph isomorphism, but are too slow to be used in practice.

A different approach to graph matching is to use permutationally invariant entities that can be easily calculated on the graph. Umeyama [93] uses eigenvalues of the association graph to solve the *weighted graph matching* problem (WGMP). The problem is formulated in terms of the isomorphism between weighted graphs that minimizes the difference between weights.

Two criticisms can be leveled at eigendecomosition methods. The first is that the approach will produce false positives. The reason for this is that there are co-spectral graphs that are structurally different but that have the same spectrum [91]. However, the speed and ease of the method combined to the relatively low probability of a false positive can compensate for the problem. The second, and more important, criticism is that the methods cannot easily cope with structural error or subgraph isomorphism problems. The reason for this is that the noise added by spurious nodes completely disrupts the spectral structure of the graph.

A third approach transforms the combinatorial problem into a continuous optimization problem and then uses the wide range of available optimization algorithms available in to find an approximate solution. In [16] Christmas and Kittler use relaxation labeling to label nodes in the data graph with the corresponding node in the model graph and use graph connectivity to combine evidence. Relaxation labeling, however, does not guarantee a one-to-one correspondence between nodes. In order to guarantee a one-to-one assignment, Gold and Ragaranjan [30] introduced the "graduated assignment" method. This is an evidence combining model that guarantees two way constraints.

Evidence combining methods like relaxation labeling and graduated assignment give a very interesting framework to iteratively improve on our initial estimate, but they are critically dependent on a good consistency model and a reliable initialization.

A more generic approach to error-tolerant graph matching derives from the extension of the concept of string edit-distance to graph-matching. This idea was independently introduced by Bunke and Allermann [12], and by Fu and his co-workers [25]. The idea behind edit-distance [90] is that it is possible to identify a set of basic edit operations on nodes and edges of a structure, and to associate a cost with these operations. 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 total cost. By making the evaluation of structural modification explicit, edit-distance provides a very effective way of measuring the similarity of relational structures. Moreover, the method has considerable potential for error tolerant object recognition and for indexing into large structural databases.

Unfortunately, the task of calculating edit-distance is a computationally hard one and most early efforts can be regarded as being goal-directed. However, in an important series of recent papers, Bunke has demonstrated the intimate relationship between the size of the maximum common subgraph and the edit distance [14]. In particular, he showed that, under certain assumptions concerning the edit-costs, computing the maximum common subgraph (MCS) and the graph edit-distance are computationally equivalent. The restriction imposed on the edit-costs is that the deletions and re-insertions of nodes and edges are not more expensive than the corresponding node or edge relabeling operations. In other words, there is no incentive to use relabeling operations and, as a result, the edit operations can be reduced to those of insertion and deletion.

The problem of assigning costs to edit operations remains an open one. Most approaches adopted in the literature are problem specific or heuristic in nature. Recently, there has been work aimed at providing a more principled approach to the estimation of the costs incurred in structural editing or deformation. For instance, adopting an information theoretic approach, Wong and You [97] have shown how the cost of edit operations can be measured using changes in entropy. Wilson and Hancock [96], on the other hand, opt for an approach in which the distribution of correspondence errors and structural errors are modeled probabilistically.

By re-casting the search for the maximum common subgraph as a max clique problem [5], we can tap into a diverse array of powerful heuristics and theoretical results available for solving the max clique problem. In particular the Motzkin-Straus theorem [54] allows us to transform the max clique problem into a continuous quadratic programming problem. An important recent development is reported by Pelillo [60] who shows how relaxation labeling can be used to find a (local) optimum of this quadratic programming problem.

In many computer vision and pattern recognition applications, such as shape recognition [74, 101, 85], pattern recognition [52], and image processing [67], the graphs at hand are trees, i.e. they are connected and acyclic. In particular, the shock graph representation of any 2D shape with no holes is a tree. Therefore, the special case of matching and comparing tree structures is an important topic by itself.

Most research in the literature adopts a structural approach to the shock-tree matching problem. For instance, Pelillo, Siddiqi and Zucker use a sub-tree matching method [62].

Kimia, Klein, and their co-workers have a potentially more robust method which matches by minimizing graph edit-distance [43, 88, 72, 74]. In particular, Sebastian, Klein and Kimia [72, 73, 42, 74] have developed a variational method which can be used to measure the cost of boundary deformation, which they refer to as "edit-distance" [72, 74]. The cost of removing a branch of the skeleton is related to the associated boundary deformation. The distance measure based on this skeleton editing procedure has been successfully used to index and retrieve shapes from a large data-base [73]. However, the method is cumbersome since it requires alignment and explicit comparison of the boundary and hence can not be encoded on the skeleton alone.

One of the criticisms of these structural matching methods is that small boundary deformations may significantly distort the topology of the skeleton. Hence, they are potentially vulnerable to structural variations or errors due to local deformations, ligature instabilities or other boundary noise.

Golland and Grimson [31] provide an interesting alternative: they minimize a boundary functional to find the optimal fit to a fixed model skeleton. This approach is very robust to boundary deformations, but is computationally very expensive. Therefore, it is not well suited to indexing large databases of shapes.

While trees are, indeed, a special case of graphs, because of the connectivity and partial order constraints which apply to them, the methods used to compare and match them require significant specific adaptation. For instance, Bartoli et al. [6], use the graph theoretic notion of a path string to transform the tree isomorphism problem into a single max weighted clique problem. This work uses a refinement of the Motzkin Strauss theorem to transform the max weighted clique problem into a quadratic programming problem on the simplex [9]. The quadratic problem is then solved using relaxation labeling.

Because of the added connectivity and partial order constraints mentioned above, Bunke's result [14] linking the computation of edit-distance to the size of the maximum common subgraph does not translate in a simple way 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 the tree isomorphism problem and the subtree isomorphism problem have efficient polynomial time solutions. Tai [86] 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. The edit operations thus defined are node deletion, node insertion, and node relabeling. This simplified set of edit operations is guaranteed to preserve the connectivity of the tree structure. Zhang and Shasha [99] 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 [100]. The NP-completeness, however, can be eliminated again by adding particular constraints to the edit operations. In particular, it can be shown that the problem returns to P when we add the constraint of strict hierarchy [98]. This is the case when separate subtrees are constrained to be mapped to separate subtrees.

2.4 Graph Clustering and Embedding

Although considerable effort has gone into the extraction and matching of shock trees, the topic of how to use shock trees to learn shape classes has not received significant attention.

Graph clustering is an important, yet relatively under-researched topic in machine learning [65, 75, 24]. The importance of the topic stems from the fact that it is a fundamental tool for learning the class structure of data abstracted in terms of relational graphs. Problems of this sort are posed by a multitude of unsupervised learning tasks in knowledge engineering, pattern recognition and computer vision. The process can be used to structure large data-bases of relational models [76] or to learn equivalence classes. One of the reasons for limited progress in this area has been the lack of algorithms suitable for clustering relational structures. In particular, the problem has proved elusive to conventional central clustering techniques. The reason for this is that it has proved difficult to define what is meant by the mean or representative graph for each cluster. However, Munger, Bunke and Jiang [38, 55] have recently taken some important steps in this direction by developing a genetic algorithm for searching for median graphs. A more fruitful avenue of investigation may be to pose the problem as pairwise clustering. This requires only that a set of pairwise distances between graphs be supplied. The clusters are located by identifying sets of graphs that have strong mutual pairwise affinities. There is therefore no need to explicitly identify a representative (mean, mode or median) graph for each cluster. Unfortunately, the literature on pairwise clustering is much less developed than that on central clustering.

When posed in a pairwise setting, the graph-clustering problem requires two computational ingredients. The first of these is a distance measure between relational structures. As illustrated before, this requirement has been addressed successfully using graph-matching methods, and, in particular, using an edit-distance approach. The second is a means of performing pairwise clustering on the distance measures. There are several possible routes available. The simplest is to transform the problem into one of central clustering. For instance, it is possible to embed the set of pairwise distances in a Euclidean space using a technique such as multi-dimensional scaling and to apply central clustering to the resulting embedding. The second approach is to use a graph-based method [77] to induce a classification tree on the data. Finally, there are mean-field methods which can be used to iteratively compute cluster-membership weights [36]. These methods require that the number of pairwise clusters be known *a priori*.

Graph matching may provide a fine measure of distance between structures, and this in turn may be used to cluster similar graphs by applying pairwise clustering to the resulting set of graph distances. However, this approach does not result in an ordering of the graphs that has metrical significance under structural variations due to graded shape changes. An alternative approach is to construct a graph-space that captures the modes of structural variation present in the data. This approach has been at the heart of recent developments in the construction of deformable models and continuous shape-spaces [33]. Here shape variations have been successfully captured by embedding them in a vector-space. The dimensions of this space span the modes of shape-variation. For instance, Cootes and Taylor [17] have shown how such shape-spaces can be constructed using the eigenvectors of a landmark covariance matrix. Sclaroff and Pentland [71], on the other hand, use the elastic modes of boundaries to define the shape-space. Luo, Wilson, and Hancock [49] use spectral features of the Laplacian matrix to embed the graphs in a low dimensional space.

Demerici and Dickinson [40] have shown how the minimum distortion embedding procedure of Linial, London, and Rabinovich [46] can be used for the purposes of correspondence matching. A recent review of methods that could be used to perform the embedding process is provided in the paper of Hjaltason and Samet [35].

Despite the many advantages and attractive features of graph representations, the methodologies available for learning structural representations from sets of training examples is relatively limited. As a result, the process of constructing shape-spaces which capture the modes of structural variation for sets of graphs has proved to be elusive. Hence, geometric representations of shape such as point distribution models [71, 33], have proved to be more amenable when variable sets of shapes must be analyzed. There are two reasons why pattern spaces are more easily constructed for curves and surfaces than for graphs. First, there is no canonical ordering for the nodes or edges of a graph. Hence, before a vector-space can be constructed, the correspondences between nodes must be established. Second, structural variations in graphs manifest themselves as differences in the numbers of nodes and edges. As a result, even if a vector mapping can be established, the vectors will be of variable length. However, there has been some progress in the area. For instance Luo, Wilson and Hancock [49] have shown how simple spectral features derived from the eigenvalues and eigenvectors of the adjacency matrix can be used both to construct pattern spaces and to locate clusters for sets of graphs.

While these spectral method can provide compact vectorial representations for the graphs, they do not provide a representation for the modes of structural variations present in the set of graphs. Recently there has been considerable interest in learning structural representations from samples of training data, in particular in the context of Bayesian networks [34, 26], mixtures of tree-classifiers [51], or general relational models [27]. The idea is to associate random variables with the nodes of the structure and to use a structural learning process to infer the stochastic dependency between these variables. Although these approaches provide a powerful way of inferring the relations between the observable quantities of the model under examination, they rely on the availability of correspondence information for the nodes of the different structures used in learning. However, in many cases the identity of the nodes and their correspondences across samples of training data are not to hand. Therefore, the correspondences must be recovered using a graph matching technique during the learning process. Hence, there is a chicken-and-egg problem in structural learning. Before the structural model can be learned, the correspondences with it must be available, and yet the model itself must be to hand to locate correspondences. Lozano and Escolano [48], and Bunke et al. [13] resolve this problem by constructing a supergraph representation from available samples. While these approaches provide a structural model of the samples, they way in which the supergraph is learned is heuristic in nature.

2.5 Conclusions

Based on this review of the relevant literature, we can draw a number of conclusions. First, the development of an efficient and reliable method for extracting the morphological skeleton is still and open issue. The Hamilton-Jacobi method provides a promising approach, but the presence of a curvature-dependent error term in the differential analysis limits its applicability. In this thesis we present a refinement on the Hamilton-Jacobi method that eliminates the curvature-dependent error term.

Second, the task of devising a stable representation of shape using the skeleton is still elusive. This is mainly due to the fact that small boundary deformations may significantly distort the topology of the skeleton. Hence, structural representations derived from the skeleton are potentially vulnerable to structural variations or errors due to local deformations, ligature instabilities or other boundary noise. In order to make the representation more stable, many measure of branch relevance or similarity have been proposed. However, most are not continuous over local regions in shape-space in which there are no topological transitions. We propose to label the skeletal representation with the ratio of border length to shock length in order to assess the similarity between the shapes.

Third, with the structural representations to hand, the correspondences between nodes must be estimated. The edit-distance framework has proven to be a powerful approach to graph matching that can be tailored to many different matching problems. Despite this, there is very little work aimed at solving or approximating the edit-distance for unordered trees. In this thesis we will show how the tree edit-distance problem can be transformed into a series of quadratic programming problems and how relaxation labeling can be used to find an approximate solution.

Finally, the topic of how to use shock trees to learn shape classes has not received significant attention. In particular, there are very few methods to learn structural archetypes capable of describing the distribution of a class of graphs. One method is to use a superstructure as a class archetype, but the existing methods to estimate these structures are heuristic in nature and more principled approaches to learn graph archetypes are needed. Central to this research is the development of a superstructure-based representation of tree classes, the tree union, and an information theoretic approach to learn different sets of union archetypes from training data.

Section I

Skeleton Extraction and Representation

Chapter 3

Curvature Correction of the Hamilton-Jacobi Skeleton

In this Chapter we present an improvement on the Hamilton-Jacobi skeleton extraction algorithm. We perform a Hamilton-Jacobi analysis of the boundary evolution under conditions where the flux-density varies due to curvature. Instead of using the gradient of the distance map, i.e. the velocity field of the eikonal equation, we use the momentum field. In other words, we multiply the velocity by the linear density of the boundary front. The resulting field is conservative, and hence zero at non-skeletal locations. Moreover, our analysis leads to a new skeleton extraction method. We compare the resulting curvature corrected skeletonization method with the Hamilton-Jacobi method. The advantages of the new method are improved localization and stability, and a reduced sensitivity to the model parameters.

3.1 Hamilton-Jacobi Skeleton

We commence by defining a distance-map that assigns to each point on the interior of an object the closest distance D from the point to the boundary (i.e. the distance to the closest point on the object boundary). The gradient of this distance-map is a field $\vec{F}$ whose domain is the interior of the shape. The field is defined to be

$$\vec{F} = \nabla D, \tag{3.1}$$

where $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})^T$ is the gradient operator. The trajectory followed by each boundary point under the eikonal equation is governed by the ordinary differential equation $\dot{\vec{x}} = \vec{F}(\vec{x})$, where $\vec{x}$ is the coordinate vector of the point. Siddiqi, Bouix, Tannenbaum, and Zucker assume that this dynamic system is Hamiltonian everywhere except on the skeleton [82, 11]. The original interpretation of this property was that at non-skeletal points the normalized flux field $\vec{F}$ is conservative, i.e. $\nabla \cdot \vec{F} = 0$. However, the total inward flux through the boundary of the shape is non zero. In fact, the flux is proportional to the length of the boundary.

The divergence theorem states that the integral of the divergence of a vector-field over an area is equal to the flux of the field over the enclosing boundary of that area. In our case, this implies that

$$\int_A \nabla \cdot \vec{F}(x)\, dx = \int_{\partial A} \vec{F} \cdot \vec{n}\, dl = \Phi_A(\vec{F}), \tag{3.2}$$

where A is an arbitrary region, $\vec{F}$ is a vector field defined in A, $d\sigma$ is the area differential in A, dl is the length differential on the boundary ∂A of A, $\vec{n}$ is a vector normal to the boundary, and $\Phi_A(\vec{F})$ is the outward flux of F through the boundary ∂A of the region A. This implies that, where the divergence is well defined, we have

$$\nabla \cdot \vec{F} = \lim_{|A| \to 0} \frac{\Phi_A(\vec{F})}{|A|}, \tag{3.3}$$

where $|A|$ is the area of the region A. With a slight abuse of notation, we extend the definition to points where the divergence is not well defined. This is done by redefining the divergence on skeletal points using equation 3.3. Since the flux through the initial boundary is non-zero, by virtue of the divergence theorem, within the interior of the shape there are points where the system is not conservative. The non-conservative points are those where the boundary trajectory is not well defined, i.e. where there are singularities in the evolution of the boundary. These points are the so-called shocks or skeleton of the shape-boundary. Shocks are thus characterized by locations where

$$\lim_{|A|\to 0} \frac{\Phi_A(\vec{F})}{|A|} < 0, \tag{3.4}$$

or, using the extended definition of the divergence,

$$\nabla \cdot \vec{F} < 0. \tag{3.5}$$

3.1.1 Curvature in the Boundary Front

Unfortunately, in general the flux of $\vec{F}$ is not conservative. To illustrate this point, let us consider an instant in time t during the inward boundary evolution. The initial shape boundary has evolved under the eikonal equation to the front $\vec{C}(t)$ which is at every location orthogonal to $\vec{F}$. We would like to select a point $p \in \vec{C}(t)$ and compute the value of $\nabla \cdot \vec{F}(p)$ at this point. The value of this divergence is more easily computed in the Frenet frame of the front $\vec{C}(t)$ passing over point p. The Frenet frame of a plane curve $\gamma : [a,b] \leftarrow \mathbb{R}^2$ is the frame provided by the orthogonal basis $\{v_\parallel, v_\perp\}$, where

$$\begin{aligned} v_\parallel(\gamma(s)) &= \tfrac{\gamma'(s)}{||\gamma'(s)||} \\ v_\perp(\gamma(s)) &= \pm \tfrac{\frac{\partial}{\partial s} v_\parallel(\gamma(s))}{||\frac{\partial}{\partial s} v_\parallel(\gamma(s))||}. \end{aligned} \tag{3.6}$$

Here the sign of $v_\perp$ depends on the chosen orientation of the curvature. We chose to orient the curvature so that $\frac{\partial}{\partial s} v_\| (\vec{C}(t,s)) = \kappa(p)\vec{F}(p)$. That is, so that the curvature $\kappa(p)$ of $\vec{C}(t)$ is positive when the curve bends towards the interior of the shape. Calculating the Frenet frame for the front $\vec{C}(t)$ at the point p, and selecting the inward orientation of the boundary curvature, we have $v_\perp = \vec{F}$. Furthermore, we have $v_\| = dl$, where dl is the arc length differential of $\vec{C}(t)$ at point p. Since the divergence operator is invariant under rotations, the divergence of the field calculated in the Frenet frame is

$$\nabla \cdot \vec{F} = \frac{\partial}{\partial v_\|}\vec{F} + \frac{\partial}{\partial v_\perp}\vec{F}. \tag{3.7}$$

Since $||\vec{F}|| = 1$ everywhere, we have $\frac{\partial}{\partial v_\perp}\vec{F} = 0$. Moreover, since $v_\|$ is an an arc-length differential for the boundary front $C(t)$ at point p, we have $\frac{\partial}{\partial v_\|}\vec{F}(p) = -\kappa(p)$, where $\kappa(p)$ is the curvature at p of $\vec{C}(t)$.

Hence, we have

$$\nabla \cdot \vec{F}(p) = -\kappa(p). \tag{3.8}$$

In other words, the divergence $\nabla \cdot \vec{F}$ is not always zero as predicted by the original Hamilton-Jacobi approach [82]. Rather, it is equal to the curvature of the front of the inward evolving boundary.

As a concrete example, consider a circle of unit radius centered in $(0,0)^T$. The gradient of the distance map at point $(x,y)^T$ is $\nabla D = -\frac{1}{\sqrt{x^2+y^2}}(x,y)^T$, and the divergence is $\nabla^2 D = -\frac{1}{\sqrt{x^2+y^2}} \neq 0$.

3.1.2 Normalized Flux

This problem was recognized by Siddiqi, Bouix, Tannenbaum, and Zucker who corrected the analysis in a subsequent publication [83] by introducing the concept of *normalized flux*. With this modification to the analysis, the non Hamiltonian points are detected by considering the flux through a circular region A of radius r normalized by the perimeter length $2\pi r$. According to the modification, non skeletal points satisfy the condition

$$\lim_{r\to 0} \frac{\Phi_A(F)}{2\pi r} = 0. \tag{3.9}$$

This condition results from the fact that $\Phi_A(F) = \nabla \cdot \vec{F}(\xi)|A|$, where $\xi \in A$ and $|A| = \pi r^2$ is the area of the circle A. Hence, the limit of the normalized flux becomes

$$\lim_{r\to 0} \frac{\Phi_A(F)}{2\pi r} = \lim_{r\to 0} \frac{\nabla \cdot \vec{F}(\xi)}{2} r = 0. \tag{3.10}$$

Furthermore, in [83] the authors proved that the limit of the normalized flux at skeletal locations is less than a negative constant c, and that this constant depends only on the characteristics of the boundary of the original shape.

While this analysis is correct, it relies on the ability to calculate the limit of the normalized flux through a region of vanishingly small area. Unfortunately, on the image lattice there is an obvious lower bound on the size of the integration area due to the pixel resolution. Hence, assuming a minimum integration radius of one pixel, the calculated normalized flux is

$$N\Phi_A(\vec{F})(p) = -\frac{\kappa(p)}{2}. \tag{3.11}$$

At most locations the absolute value of the calculated normalized flux is much smaller than the constant c. However, near the endpoints of the skeleton the curvature of the boundary front tends to infinity. Hence, at these locations the exact location of the skeletal points becomes somewhat elusive.

3.2 Momentum Field

The fact that the divergence of the field $\vec{F}$ is non-zero can be easily understood by appealing to an analogy from physics. Let us assume that a fluid

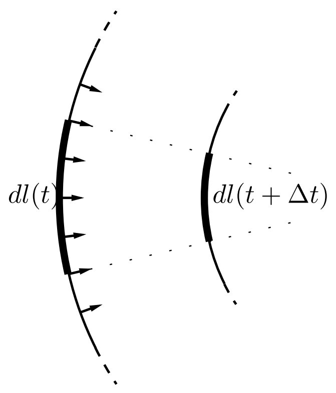


Figure 3.1: Evolution of a boundary segment.

of uniform density flows from the boundary of the shape, which acts as a source, to the skeleton, which acts as a sink. If the fluid is incompressible, then the fluid density never changes and the flux of the velocity field $\vec{F}$ is conservative everywhere except at points on the skeleton. If, on the other hand, the fluid is compressible, then as soon as a curved front compresses the fluid, the density changes and the velocity field is no longer conservative.

To develop this idea one step further, consider a segment $dl(t)$ of the boundary front $\vec{C}(t)$ at time t. We assume that this segment has average linear density $\hat{\rho}(t)$ (see Figure 3.1). Under the eikonal equation, at time $t+\Delta t$ the boundary front segment $dl(t)$ has evolved to $dl(t+\Delta t)$. Since each of the points in $dl(t)$ are now contained in $dl(t+\Delta t)$, the total mass of the two segments is the same. However, if $dl(t)$ is curved then the lengths of the segments are different, i.e. $||l(t+\Delta t)|| \neq ||l(t)||$. Thus the average density of $l(t+\Delta t)$ is $\hat{\rho}(t+\Delta t) \neq \hat{\rho}(t)$. As a result, when the front is curved, the density is not constant and we have to take into account mass effects. That is, we have to resort to the more general principle of conservation of mass.

Based on this physical intuition, we state that there is indeed a conservative field associated with the dynamics of the boundary, namely the momentum $\vec{M} = \rho\vec{F}$, where ρ is the scalar field that assigns to each point the linear density of the boundary front. As a result we have that

$$\nabla \cdot (\rho\vec{F}) = 0, \tag{3.12}$$

Applying the rules of product differentiation, we obtain the partial differential equation (PDE)

$$\nabla\rho \cdot \vec{F} = -\rho\nabla \cdot \vec{F}. \tag{3.13}$$

By setting $\sigma = \log(\rho)$, we can write the above PDE as a function of the log-density σ

$$\rho\nabla\sigma \cdot \vec{F} = -\rho\nabla \cdot \vec{F}. \tag{3.14}$$

Eliminating ρ from both sides, we obtain

$$\nabla\sigma \cdot \vec{F} = -\nabla \cdot \vec{F}. \tag{3.15}$$

This is a transport equation that can be reduced to the following set of ordinary differential equations (ODE) along the paths of the boundary points

$$\begin{cases} \frac{d}{dt}\sigma(s(t)) = -\nabla \cdot \vec{F}(s(t)) \\ \frac{d}{dt}s(t) = \vec{F}(s(t)), \end{cases} \tag{3.16}$$

where $s(t)$ is the trajectory of a boundary point.

These equations can be derived by analyzing the change in density of the segment dl in Figure 3.1. To commence, we note that $\hat{\rho}(t)||dl(t)|| = m$, where $dl(t)$ is the length of the boundary segment at time t, m is its mass, $\hat{\rho}(t)$ is its average linear density and $\kappa(t)$ is the curvature at time t. After a small interval of time Δt, the segment length will be

$$||dl(t + \Delta t)|| = ||dl(t)||\frac{\kappa(t)}{\kappa(t + \Delta t)} + O(\Delta t^2), \tag{3.17}$$

and the curvature

$$\kappa(t + \Delta t) = \frac{\kappa(t)}{1 - \kappa(t)\Delta t} + O(\Delta t^2). \tag{3.18}$$

From these equations and the conservation of mass, we have that

$$\hat{\rho}(t+\Delta t) = \frac{m}{||dl(t+\Delta t)||} = \hat{\rho}(t)\frac{1}{1-\kappa(t)\Delta t} + O(\Delta t^2). \tag{3.19}$$

Hence

$$\hat{\rho}(t+\Delta t) - \hat{\rho}(t) = \hat{\rho}(t)\frac{\kappa(t)\Delta t}{1-\kappa(t)\Delta t} + O(\Delta t^2). \tag{3.20}$$

Taking the limit for $\Delta t \to 0$ and $||dl|| \to 0$, we have

$$\frac{\frac{d}{dt}\rho(s(t))}{\rho(s(t))} = \kappa(s(t)), \tag{3.21}$$

where $s(t)$ is the trajectory of the limit point of the segment dl as $||dl|| \to 0$. Integrating (3.21) we obtain

$$\log(\rho(s(t))) = \int_0^t \kappa(s(\tau))\, d\tau. \tag{3.22}$$

From Equation 3.8, we have that $\kappa(p) = -\nabla \cdot \vec{F}$, yielding

$$\log(\rho(s(t))) = -\int_0^t \nabla \cdot \vec{F}(s(\tau))\, d\tau. \tag{3.23}$$

Hence, integrating ρ, we obtain the vector field ρF which satisfies the condition $\nabla \cdot (\rho\vec{F}) = 0$ at non-skeletal points. The analysis of skeletal points is more complex. The problem we face is that both F and ρ are multi-valued on the skeleton and hence $\nabla \cdot (\rho\vec{F})$ is not defined. Although the divergence is not well defined, we can calculate the flux through an area containing a skeletal point.

To pursue this analysis we turn our attention to Figure 3.2. Consider the skeleton segment s in the figure that is surrounded by a ribbon ϵ of half-width radius r. The skeleton segment s originates from the inward evolution of the boundary segments l_1 and l_2. The interior of the shape between the ribbon surrounding the skeleton and the two object boundary segments can be divided into two areas A_1^ϵ and A_2^ϵ. The areas are enclosed by the outside boundary of the ribbon ϵ, the boundary segments l_1 and l_2, and the

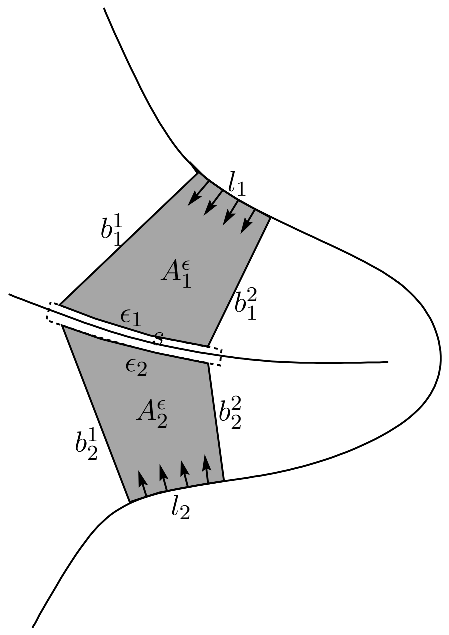


Figure 3.2: The flux through the boundary is equal to the flux through ϵ.

trajectories b_1^1, b_1^2, b_2^1 and b_2^2 of the endpoints of l_1 and l_2. Since $\nabla \cdot \rho\vec{F} = 0$ everywhere in A_1^ϵ and A_2^ϵ, by virtue of the divergence theorem the flux from the two areas are both zero, i.e. $\Phi_{A_1^\epsilon}(\rho\vec{F}) = 0$ and $\Phi_{A_2^\epsilon}(\rho\vec{F}) = 0$. The trajectories of the endpoints of the boundary are, by construction, parallel to the field, so the associated boundary normals are everywhere perpendicular to the field. Thus there is no flux through the segments b_1^1, b_1^2, b_2^1 and b_2^2. On the other hand the field on the shape boundary is always perpendicular to the boundary. Hence, the flux through the boundary segments l_1 and l_2 is equal to their respective lengths $||l_1||$ and $||l_2||$.

Since $\Phi_{A_1^\epsilon}(\rho\vec{F}) = 0$ and $\Phi_{A_2^\epsilon}(\rho\vec{F}) = 0$ the flux that enters through the boundary segments l_1 and l_2 has to exit through ϵ. That is, if ϵ_1 and ϵ_2 are respectively the sides of the areas A_1^ϵ and A_2^ϵ close to the skeleton, we have that $\Phi_{\epsilon_1}(\rho\vec{F}) = \Phi_{l_1}(\rho\vec{F})$ and $\Phi_{\epsilon_2}(\rho\vec{F}) = \Phi_{l_2}(\rho\vec{F})$. The results in turn imply that the flux through the ribbon ϵ is $\Phi_\epsilon(\rho\vec{F}) = -||l_1|| - ||l_2||$.

Since any area containing a skeleton segment s can be approximated with arbitrary precision with a series of ribbons of the sort described above, the flux through any region A containing a skeletal segment s is

$$\Phi_A(\rho\vec{F}) = -||l_1|| - ||l_2||. \tag{3.24}$$

The limit is not well-defined for an arbitrary sequence of regions of vanishingly small area. However, assuming that we can construct a sequence of circular regions A_r of radius r, containing a skeletal segment l_s, we have that

$$\lim_{r\to 0}\frac{\Phi_{A_r}(\rho\vec{F})}{||A_r||} = \lim_{r\to 0}\frac{-4r\frac{dl}{dl_s}(\xi) + O(r^2)}{\pi r^2} = -\infty \quad \text{if } \frac{dl}{dl_s} > 0. \tag{3.25}$$

Here $\frac{dl}{dl_s}$ is the ratio between boundary length and segment length and ξ is a point in the skeletal segment l_s.

When integrating the flux numerically on the pixel lattice, there will be a lower bound on the radius, hence the corresponding limiting value of the normalized flux will not be $-\infty$, but it will be negative since $\frac{dl}{dl_s} \geq 0$. In particular, the flux will be zero if and only if $\frac{dl}{dl_s} = 0$, that is, on pure ligatures. Ligatures are skeletal branches linked with high negative curvature on the boundary. They are not linked with any feature on the boundary, but serve the purpose of linking skeletal features keeping the skeleton connected.

It is worth noting that if we follow Siddiqi et al. [83] and normalize by the perimeter $||\partial A_r||$ of the region A_r, we obtain

$$\lim_{r\to 0}\frac{\Phi_{A_r}(\rho\vec{F})}{||\partial A_r||} = \lim_{r\to 0}\frac{-4r\frac{dl}{dl_s}(\xi) + O(r^2)}{2\pi r} = -\frac{2}{\pi}\frac{dl}{dl_s}. \tag{3.26}$$

Hence, factoring out the curvature effects, we obtain a clear geometrical interpretation of the value of the normalized flux at a skeletal point: It is proportional to the ratio between the boundary length and the skeletal length.

3.3 Boundary Curve Parameterization

There is another interpretation for the scalar field ρ derived from the analysis of the evolution of an arc-length parameterization of the boundary curve of the shape [41]. Let $\vec{C}(t)$ be a solution to the eikonal equation (2.1), where $\vec{C}(t) : [0, s] \to \mathbb{R}^2$ is the equation of the front at time t. Furthermore, let the differential ds of the parameterization s of the curve be, at time $t = 0$, an arc length differential. Clearly, ds will not remain a differential of arc length throughout the evolution of the curve. However, we can define a metric $g(t, s) = \frac{ds}{dl}$ that links the length at time t of the differential ds of the parameterization s, to the arc length differential dl. With this notation, we have that

$$\rho(\vec{C}(t, s)) = \frac{1}{g(t, s)}. \tag{3.27}$$

Further suppose that l is a segment on the initial boundary, and let l' be the corresponding segment on the front at time t. Since we assume unit density at time $t = 0$, the total mass of the segment l is equal to its length $|l|$. Since mass is conserved, the total mass of the segment l' will remain $|l|$. Hence, the average density is $\rho = \frac{|l|}{|l'|}$. Taking the limit as $|l'| \to 0$, we have $\rho = \frac{dl}{ds}$. With this definition of the front density, we can rewrite the momentum field as $\vec{M} = \frac{\vec{F}}{g}$. Following [41], the divergence of the momentum and velocity fields are given by

$$\nabla \cdot \left(\frac{\vec{F}}{g}\right) = \frac{\partial}{\partial l}\frac{\vec{F}}{g} + \frac{\partial}{\partial t}\frac{\vec{F}}{g} = \frac{\partial}{g\partial s}\frac{\vec{F}}{g} + \frac{\partial}{\partial t}\frac{\vec{F}}{g} = \frac{1}{g^2}(-kg) + \frac{-\frac{\partial g}{\partial t}}{g^2} - \frac{k}{g} + \frac{kg}{g^2} = 0 \tag{3.28}$$

and

$$\nabla \cdot \vec{F} = \frac{\partial}{\partial l}\vec{F} + \frac{\partial}{\partial t}\vec{F} = \frac{\partial}{g\partial s}\vec{F} + \frac{\partial \vec{F}}{\partial t} = \frac{1}{g}(-kg) + 0 = -k. \tag{3.29}$$

These are, in fact, the results obtained via our analysis of the momentum field.

3.4 Computing the Density

To obtain the momentum field we need to integrate the density field over the interior of the shape. Since images have a finite resolution, we need to discretize the solution onto the image lattice.

One approach is to express the PDE (3.15) as a system of difference equations. The difference equations form a linear system that can then be solved to obtain the log-density $\sigma = \log(\rho)$. The problem with this approach is that the skeleton is a set of singularities of the momentum field. Hence, the density can have very different values at opposite sides of a skeletal branch. The net effect is that the linear system will have no solution. In fact, even seeking an approximate solution using a residual descent method would result in oscillations near the skeleton.

3.4.1 Integration in Time

In order to overcome this problem we need to ensure that the difference operators used in the equations never cross a skeletal branch. One way to guarantee this is to integrate the equation in the time domain. This must be done so that the formulae giving the value of ρ at points on the boundary front at time t reference values of ρ only at points in the fronts at previous times. We can realize this by integrating the ODE (3.16) along the paths of the boundary points.

To do this we opt to use the second order Cranck-Nicolson method [19]. For each point $(x, y) = \vec{C}(t, s)$ in the interior of the shape, we have the equation

$$\sigma(\vec{C}(t,s)) - \sigma(\vec{C}(t-1,s)) = -\frac{1}{2}[\nabla \cdot \vec{F}(\vec{C}(t,s)) + \nabla \cdot \vec{F}(\vec{C}(t-1,s))]. \quad (3.30)$$

Solving for the log-density at time t, we obtain

$$\sigma(\vec{C}(t,s)) = \sigma(\vec{C}(t-1,s)) - \frac{1}{2}[\nabla \cdot \vec{F}(\vec{C}(t,s)) + \nabla \cdot \vec{F}(\vec{C}(t-1,s))]. \quad (3.31)$$

Using this equation we can calculate the log-density at a point on the evolving boundary at time t, referencing only values of the log-density at points that belong to the front at previous times. Since the evolution never crosses the skeleton, we are guaranteed not to cross skeletal branches during our calculations.

3.4.2 Integration in Space

Equation (3.31) allows us to integrate the log-density σ in the time domain along the evolution path followed by a boundary point. However, we have not shown how to calculate the integration path. Fortunately, we do not need to calculate every possible path. Let us assume that at time t the boundary front passes through the point $\vec{C}(t,s) = (x,y)^T$. The first order approximation of the position of this point at time $t-1$ is

$$\vec{C}(t-1,s) = (x,y)^T - \vec{F}(x,y) = (x - F_x, y - F_y)^T. \quad (3.32)$$

Using this approximation, we can write Equation (3.31) in the spatial domain instead of the time domain. As a result the density is given by

$$\sigma(x,y) = \\ \sigma(x - F_x(x,y), y - F_y(x,y)) - \frac{1}{2}[\nabla \cdot \vec{F}(x,y)) + \nabla \cdot \vec{F}(x - F_x(x,y), y - F_y(x,y))]. \quad (3.33)$$

As shown in Figure 3.3, the point $(x,y)^T - \vec{F}(x,y)$ does not belong to the image lattice. Hence, we need to interpolate it using the values at the four corners of the square containing the point. Note that the point $(x,y)^T$ is the

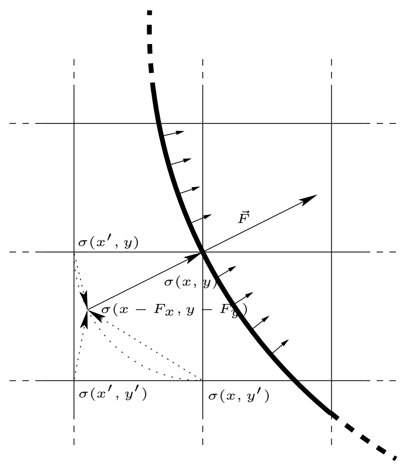


Figure 3.3: Integration along the boundary path.

last of the four points in the lattice that is visited by the evolving boundary. Hence, the interpolation is guaranteed to use points on the same side of a skeleton. We opt to compute the quantity $f(x + a, y + b)$ with $a, b \in [0, 1)$ using the bilinear interpolation

$$(a-1)(b-1)f(x,y)+a(b-1)f(x+1,y)+(a-1)bf(x,y+1)+abf(x+1,y+1). \tag{3.34}$$

With this interpolation, Equation (3.33) becomes

$$\begin{gathered}[1 - (1 - |F_x|)(1 - |F_y|)]\sigma(x, y) = |F_x|(1 - |F_y|)\sigma(x', y) \\ +(1-|F_x|)|F_y|\sigma(x,y')+|F_x||F_x|\sigma(x',y')-\frac{1}{2}[\nabla\cdot\vec{F}(x,y))+\nabla\cdot\vec{F}(x-F_x,y-F_y)],\end{gathered} \tag{3.35}$$

where $x' = x + \operatorname{sgn}(F_x(x, y))$ and $y' = y + \operatorname{sgn}(F_y(x, y))$.

Using Equation (3.35) we can compute the value of the log-density $\sigma(x, y)$ using values of σ at the points spanned by the evolving boundary front before the point $(x, y)^T$. Hence, to calculate σ all we need to do is to iterate Equation (3.35) through the interior points according to front arrival time.

We commence from the points reached first by the boundary front and proceed to those reached last. Since the evolving boundary front is moving with constant unit velocity, the time taken by the front to reach the point with position $(x, y)^T$ is equal to its distance from the initial shape boundary.

Once we have the density to hand, we need to calculate the divergence of the momentum in every point on the image lattice. We opt to discretize Equation (3.15) using the second order approximation

$$\nabla \cdot (\rho \vec{F})(x, y) = [\sigma(x, y) - \sigma(x - F_x, y - F_y)] \exp(\sigma(x, y) - \frac{1}{2}\Delta\sigma)$$
$$+\frac{1}{2}\left[\nabla \cdot \vec{F}(x - F_x, y - F_y) \exp(\sigma(x - F_x, y - F_y)) + \nabla \cdot \vec{F}(x, y) \exp(\sigma(x, y))\right]. \tag{3.36}$$

This corresponds to the second-order Cranck-Nicolson method applied to the integration of the log-density σ.

3.5 Skeletonization

Once the divergence of the momentum field is to hand, we can extract the skeleton. The extraction process we adopt is similar to the one adopted by Siddiqi et al. [83]. To perform the extraction we thin the shape by removing boundary points that have energy absorption below a certain threshold, and whose removal does not cause the shape to be split into two disjoint parts. We then further thin the remaining shape to a 1-pixel wide skeleton, being careful to maintain the connectivity of the shape and to avoid shortening of the skeleton by eliminating endpoints. Figure 3.4 provides pseudo-code for the thinning process of the shape S.

The predicate `is_simple` determines whether the shape is still connected after the removal of the point p. It does so by checking only the points

```
For each point $p$ in distance order
  if is_simple($S \setminus p$) and $-\nabla \cdot \rho\vec{F}(p) < \epsilon$
    then $S = S \setminus p$
For each remaining point $p$ in distance order
  if is_simple($S \setminus p$) and not is_endpoint($S$,$p$)
    then $S = S \setminus p$
```

Figure 3.4: Pseudo-code for the thinning process.

in the neighborhood of p. The shape $S \setminus p$ is connected if the points in the neighborhood of p, excluding p, are connected. Similarly, `is_endpoint` determines whether p is an endpoint. It does so only by inspecting the neighborhood of p. The point is an endpoint if it has at most two neighboring points and those points are horizontally or vertically adjacent.

It is worth noting that the only external help this thinning algorithm requires is the detection of endpoints. In fact, if an algorithm were to return only the set of endpoints, the thinning process would reconstruct the same skeleton. On the other hand, the algorithm is highly dependent on the quality of the detection of the endpoints. Hence, an improvement in the detection and localization of the endpoints would result in an improvement in the extraction algorithm.

3.6 Experimental Comparison

In this section we attempt to characterize the differences between the Hamilton-Jacobi skeletonization method and our density-corrected approach. We commence by providing a qualitative analysis of the difference in the divergence of the velocity and momentum fields. Then, we provide an analysis of the noise and thresholding sensitivity of the two methods. Finally, we provide a more quantitative analysis of the localization properties of the two skeletonization methods.

Figures 3.5 and 3.6 show, for a few selected shapes from our database, the values of the flux through a unit circle of the velocity field $\Phi_1(\vec{F})$, the computed log-density $\log(\rho)$, and the flux through a unit circle of $\Phi_1(\rho F)$. Note that, since we fixed the radius for the calculation of the flux at 1 pixel, the flux $\Phi_1(\vec{F})$ and the normalized flux $N\Phi_1(\vec{F})$ differ only by a multiplicative constant. In these pictures white (grey-scale 255) corresponds to a large positive value, black (grey scale value 0) to a large negative value and zero is represented by the grey scale value 128. To better show the differences, the contrast of the images is strongly enhanced. This is done by applying to the intensity of each point a sigmoidal function with slope on 0 equal to 10.

It is clear from the pictures that the divergence of the velocity field on non-skeletal points is not zero where the boundary evolution front is curved – that is, in correspondence with a curved boundary. The value of the flux through an area that does not contain any section of the skeleton is, in general, an order of magnitude smaller than the value calculated over an area that contains a skeletal branch. However, near the endpoints of the skeletal branches the values become comparable. This can be observed as a blurred dark region around the endpoints. Furthermore, quantization in the localization of the shape causes the initial boundary to be very jagged. This high-frequency, low-amplitude noise is transported and amplified throughout the velocity field, creating stripes with high local curvature in the evolving front. This in turn yields a noisy and poorly localized skeleton. By contrast, the density correction in the momentum field dampens this noise.

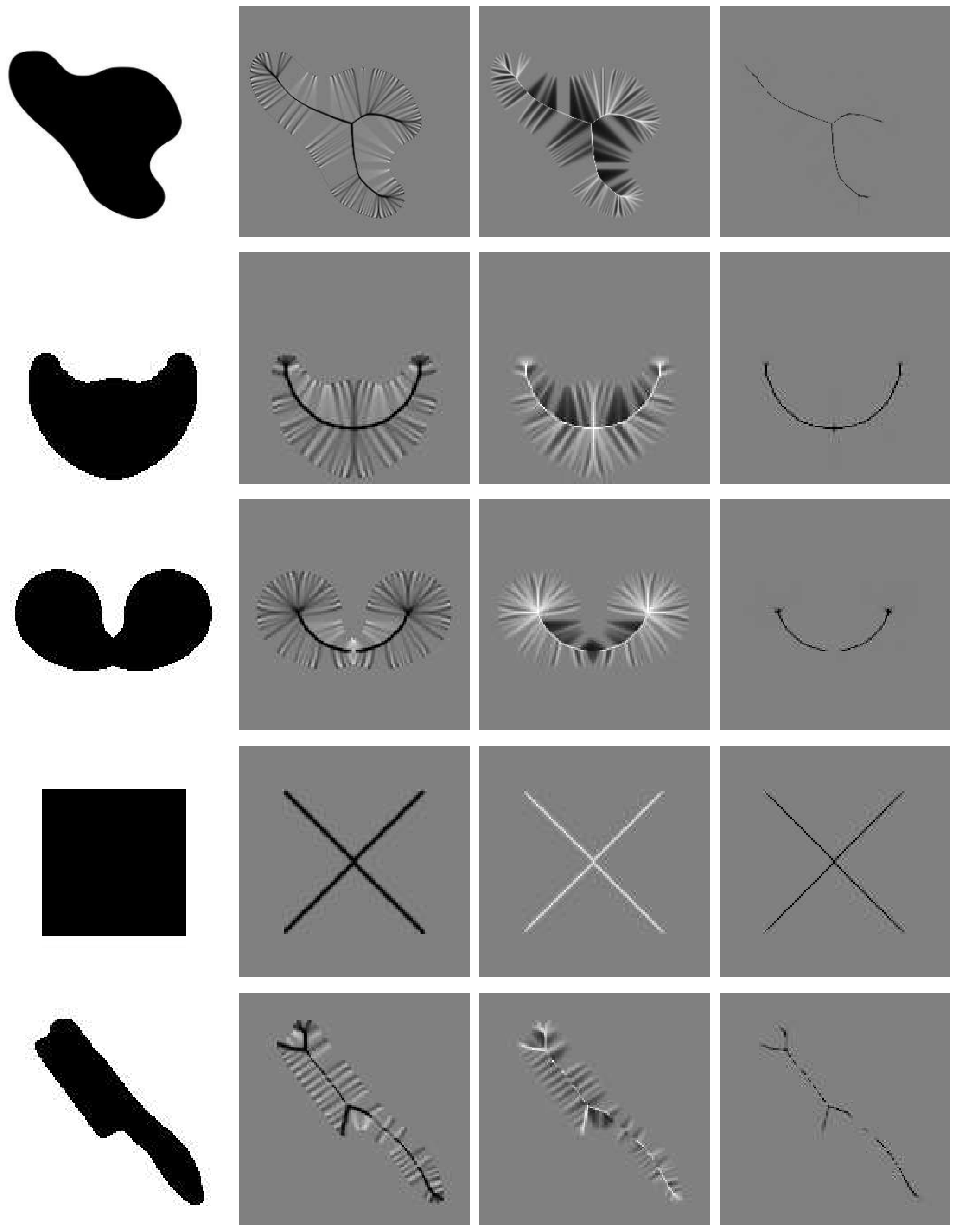


Figure 3.5: Differences in the velocity and momentum fields. Left to right: shape, (normalized) flux of $\vec{F}$, $\log(\rho)$, and flux of $\rho\vec{F}$.

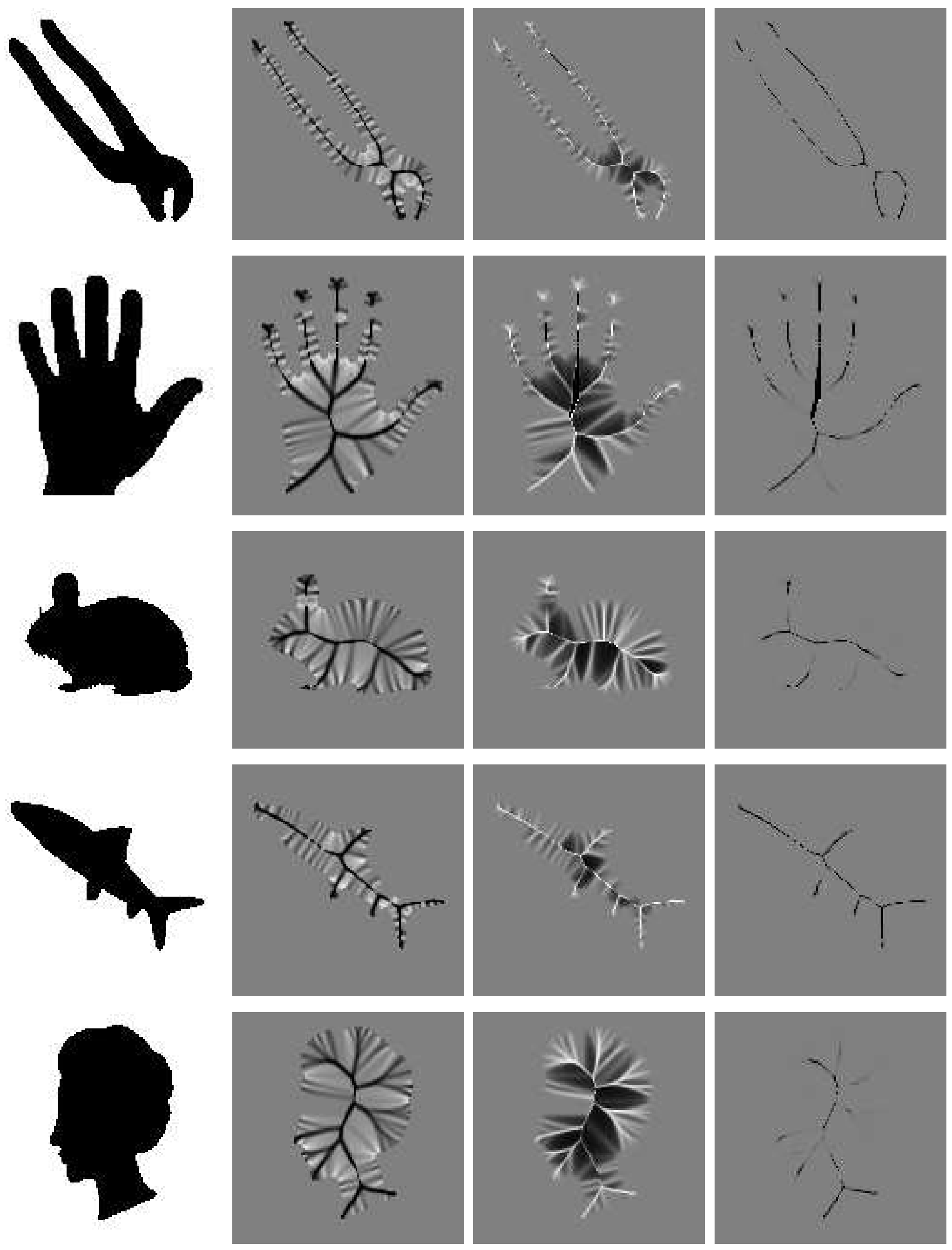


Figure 3.6: Differences in the velocity and momentum fields. Left to right: shape, (normalized) flux of $\vec{F}$, $\log(\rho)$, and flux of $\rho\vec{F}$.

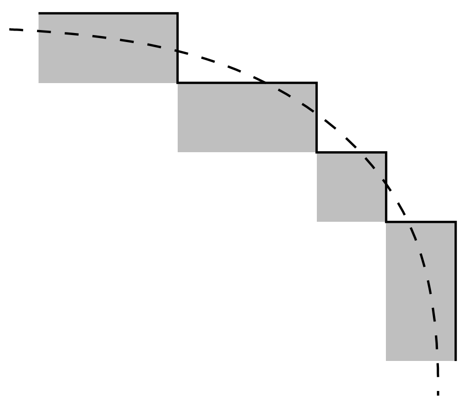


Figure 3.7: Discretization error on boundary localization.

3.6.1 Noise Sensitivity

Our skeletonization method depends on our ability to calculate the distance map D and its gradient. This, in turn, depends on the correct localization of boundary points. Unfortunately, due to the truncation effects caused by the finite precision of the image lattice, the extracted boundary presents discretization errors – most notably in the form of jagged edges. Figure 3.7 illustrates the problem. The dashed line represents the original boundary of the shape and the gray squares represent the boundary pixels in the image lattice. Due to this discretization, the observed boundary is equal to the solid line. Clearly, commencing from this observed boundary, the distance map will diverge considerably from its correct value. The effects of this discretization error will be even more dramatic on the gradient ∇D. To overcome quantization noise from the object boundary, we need to smooth the observed shape boundary and select an appropriate skeletonization threshold. To smooth the shape boundary we approximate shape diffusion [41]. In order to approximate the skeletonization of the diffused shape, it is not necessary to explicitly calculate the diffused shape and distance-map. Instead, it is sufficient to approximate the gradient ∇D of the diffused distance map.

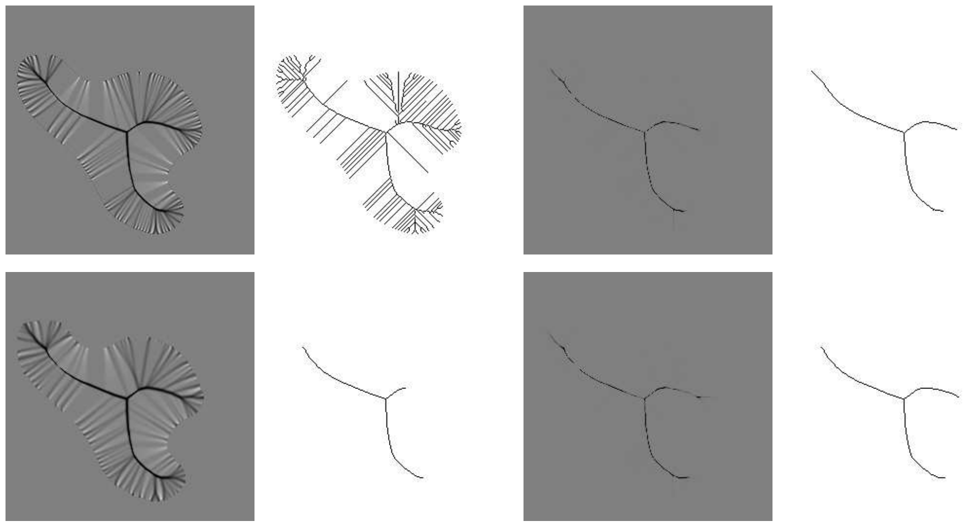


Figure 3.8: The effect of smoothing on skeleton extraction.

Let D be the distance map of the original shape and D_τ be the distance map of the shape after a diffusion with parameter τ. Furthermore, let $G_\tau * f$ be a Gaussian smoothing of a function f with standard deviation τ. For small τ we have

$$\nabla D_\tau \approx \frac{\nabla G_\tau * D}{||\nabla G_\tau * D||}. \tag{3.37}$$

Hence, we can approximate the diffusion of the image by performing Gaussian smoothing to the distance map and normalizing the resulting gradient. This approach allows us to approximate the gradient of the distance map of an ideal diffused boundary calculated with subpixel precision, without actually calculating the diffused boundary with subpixel precision.

If either the smoothing radius or the threshold is too large, then some of the branches of the skeleton will be thinned away. If, on the other hand, the selected values are too small, then the detected skeleton will have a large number of spurious branches (See Figure 3.8). In this section we characterize the effects of the smoothing radius and the skeletonization threshold on the quality of the detected skeleton.

Figure 3.8 displays the effects of very low (top) and very high (bottom) values of the smoothing radius and the skeletonization threshold on a test shape. The picture shows, left to right, the divergence of the velocity field, the uncorrected Hamilton-Jacobi skeleton, the divergence of the momentum field, and the skeleton extracted using the density-corrected method. These pictures demonstrate that the density corrected method is much less sensitive to the amount of smoothing and to the value of the threshold.

The improved extraction and the extended range of the extraction parameters have a direct effect on the usability of the method for any shape recognition process based on a skeletal representation. The extended parameter-range reduces the amount of tuning required on the extraction part of the recognition method, allowing the method to be fully automated, while improved correctness of the extraction has obvious implications for the recognition process.

Sensitivity to boundary noise is a problem with all skeleton extraction methods, and is rooted in the high sensitivity of the skeletal representation to boundary deformation. In an actual skeleton extraction algorithm this sensitivity is compounded with the accumulation of discretization errors in correspondence with high frequency boundary noise. This means that an algorithm that is correct in the continuous domain can be affected by errors in the discrete domain. Our solution is to reduce the high frequencies in the shape boundary in order to reduce the discretization error.

A complementary approach present in the literature is to extract a noisy skeleton and then prune the extraneous branches [56, 57, 10, 87]. A problem with this approach is that the calculation of the endpoints of the skeletal branches that are not pruned away is still affected by the high-frequency boundary noise. Hence a branch that should stop within the interior of the

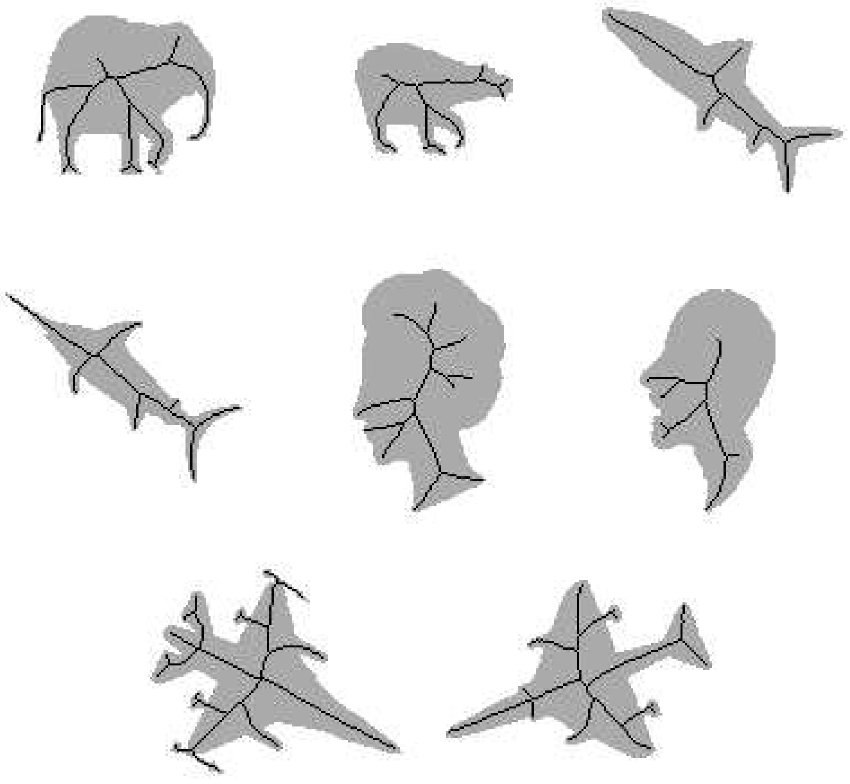
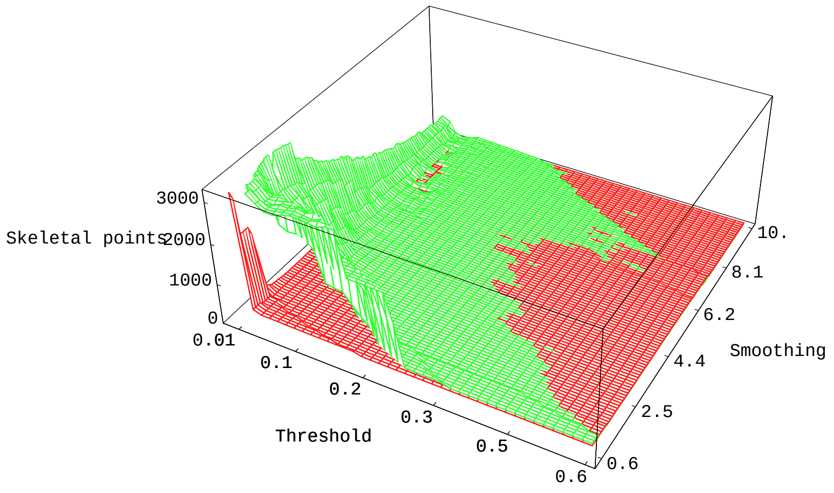
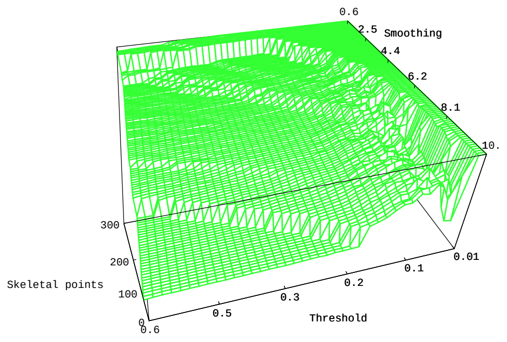


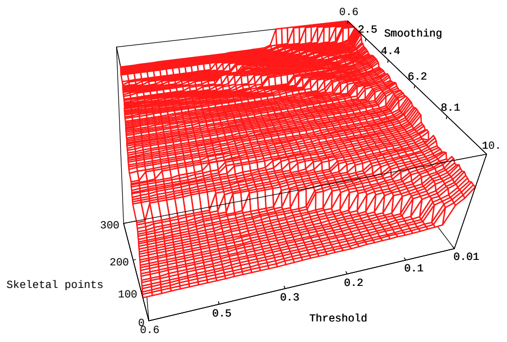
Figure 3.9: Smoothing improves localization of endpoints.

shape, on the center of a low curvature bitangent circle, will be prolonged almost up to the boundary, to the center of a high curvature bitangent circle created by a high frequency boundary feature. This effect can be seen confronting the results in Figure 3.9 with the results published in [87]. Just like our method, the algorithm described by Tek and Kimia in [87] is correct in the continuous domain. Yet they chose to adopt the pruning approach. The resulting skeletal branches are, for the most part, the same, except that their approach extends the branches inward to the center of circles that are bitangent to boundary features that have similar magnitude as the discretization noise.

To be fair, the smoothing approach has a drawback as well: in certain degenerate cases boundary diffusion could generate new skeletal features [3].







(b) Hamilton-Jacobi (c) Density corrected

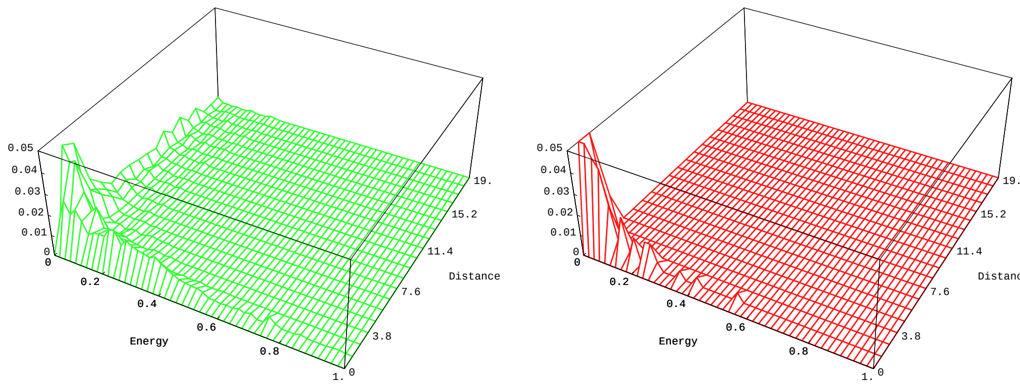
Figure 3.10: Effect of smoothing and threshold on skeleton extraction.

Figure 3.10 plots the number of detected points on the skeleton of the test shape as a joint function of the smoothing radius and the skeletonization threshold. Ideally, as the smoothing radius increases, the number of detected points should reach a plateau very quickly, and then abruptly drop to a lower plateau as a feature of the shape is smoothed away. The amount of smoothing required to reach a new plateau should be independent of the value of the threshold. Figure 3.10a shows the number of points extracted by the two methods. The results are superimposed as separate surfaces. It is clear from the plot that, of the two methods, the density-corrected method reaches the plateau faster as we increase the threshold or the smoothing radius. Moreover, it maintains the plateau for longer. Figures 3.10b and 3.10c show the results separately for the Hamilton-Jacobi approach and the density-corrected method. Here the location closest to the viewer is the plateau side of the plots. The ridges in the forefront show the drop in the number of skeletal points due to the smoothing away of an image feature. In both cases, the drop is sudden, but the ridge in the Hamilton-Jacobi plot shows a higher dependence on the threshold.

3.6.2 Skeleton Localization

In this section we characterize the localization properties of the skeleton extracted using the Hamilton-Jacobi method and the new density-corrected method on a wide variety of shapes. To this end, we investigate how the values of the divergence of the velocity and of the momentum field are distributed over the distance to the extracted skeleton. Figure 3.11 plots a histogram of the distribution of non-skeletal points as a function of distance and divergence value for the test shapes. The figure shows that the Hamilton-Jacobi skeleton has a non-negligible tail for high divergence values, even at large distance from the extracted skeleton.

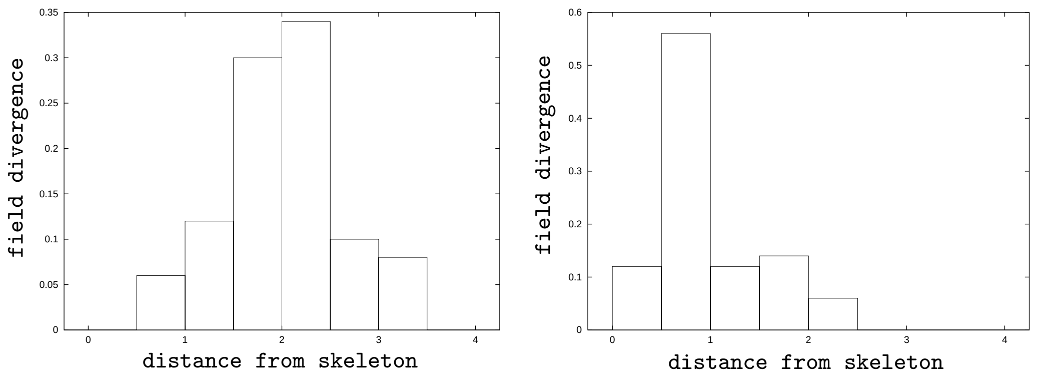
We have also performed an experiment aimed at quantifying the localization of the skeleton on a database of shapes. We have used a database of 50 shapes and have histogrammed the distribution of field divergence as



(a) Hamilton-Jacobi (b) Density corrected

Figure 3.11: Histogram over value of (negative) divergence of the field and distance to skeleton.

a function of the distance to the skeleton. We have repeated this procedure for both the velocity field and the momentum field. For each shape, we take the mean of the relevant divergence-distribution as a measure of divergence-localization.



(a) Hamilton-Jacobi (b) Density corrected

Figure 3.12: Histogram of divergence-localization on a database of 50 shapes.

Figure 3.12 shows histograms of this divergence-localization measure accumulated over all the shapes in our database. We have divided the histogram contents into 8 bins of average divergence-distance. In Figure 3.12a we show the localization histogram for the velocity field. The mean of this distribution is 2.52, while the variance is 0.34. Figure 3.12b is the corresponding histogram for the momentum field. The mean of this distribution is 1.46, while the variance is 0.28. The density correction clearly leads to a better localization of the skeleton.

3.7 Conclusions

This chapter presents a skeletonization method that corrects curvature effects in the Hamilton-Jacobi framework. Our approach addresses a shortcoming of the Hamilton-Jacobi method for skeleton extraction, namely its sensitivity to high curvature. This is due to the fact that the normalized flux, the key component of the Hamilton-Jacobi algorithm, has an error term proportional to the curvature of the inward evolving boundary front. To overcome this problem, we have presented an analysis which takes into account variations of density due to boundary curvature. This yields a skeletonization algorithm that is both better localized and less susceptible to boundary noise than the Hamilton-Jacobi method. Yet our analysis of the effects of boundary noise show that high noise still affects the extraction algorithm. This drawback is due to the intrinsic sensitivity of the skeletal representation, and hence present in every extraction algorithm. This intrinsic sensitivity is compounded with a higher incidence of discretization error in correspondence with high frequencies in the boundary features. To counter this we smooth the boundary by approximating a diffusion operator.

In the next chapter we take this work one step further by investigating how to attribute the skeleton with information concerning the way in which the skeleton varies as the boundary is deformed.

Chapter 4

Skeletal Measure

As noted in the review of the relevant literature, one of the criticisms that can be leveled at existing skeletonization methods is their sensitivity to small boundary deformations or ligatures. Although these can be reduced via curvature dependent smoothing, they may have a significant effect on the topology of the extracted skeleton. Conversely, the structural representations of shape based on the morphological skeleton suffer from the problem that perceptually distinct shapes may have topologically similar, if not identical, skeletons which can not be distinguished from one another.

From the literature review we can hence draw two observations. The first is that if a largely structural representation of the skeleton is used, then shapes which are perceptually different but which give rise to the same skeleton topology can be ambiguous with one another. For this reason in this chapter we would like to develop a representation which can be used to assess the differences in shape for objects which have topologically identical skeletons. Secondly, we would also like to be able to make comparisons between shapes that are perceptually close, but whose skeletons exhibit topological differences due to small but critical local shape deformations. Thirdly, we aim to do this without making detailed boundary comparisons. In particular we wish to construct a representation which dispenses with the boundary, but encodes information concerning its shape on the skeleton.

To meet these goals, our shape-measure must have three properties. First, it must be continuous over local regions in shape-space in which there are no topological transitions. If this is the case then it can be used to differentiate shapes with topologically identical skeletons. Secondly, it must vary smoothly across topological transitions. This is perhaps the most important property, since it allows us to define distances across transitions in skeleton topology. In other words, we can traverse the skeleton without encountering singularities. Thirdly, it must distinguish between the principal components of the skeleton and its ligatures [4]. This will allow us to suppress instabilities due to local shape deformations.

4.1 Contribution

We opt to use a shape-measure based on the rate of change of boundary length with distance along the skeleton. To compute the measure we construct at each location on the skeleton the bitangent circle inscribed in the shape. This circle is centered on a skeletal point and is bitangent to the boundary at the two boundary points. Hence, each skeletal point is in correspondence with (at least) two points on the border. The rate of change of boundary length with distance along the skeleton is computed by taking neighboring points on the skeleton. The corresponding change in boundary length is computed by determining distance along the boundary between the corresponding points of contact for the two bitangent circles. The boundary distances are averaged for the boundary segments at either side of the skeleton.

This measurement has previously been used in the literature to express *relevance* of a branch when extracting or pruning the skeleton [56, 57]. Blum and Nagel [8] suggested that the border length to shock length ratio could be used, together with other measures, to characterize the shape, but the reasons for this proposal were attributed to the measure's ability to reveal whether a skeletal section is a ligature. In practice they used the measure only as a purely static measure of relevance, ignoring its properties when the shape undergoes deformation.

We show that the rate of change of boundary length with distance along the skeleton has a number of interesting properties. The consequence of these properties is that the descriptive content of the measure extends beyond simple feature saliency, and can be used to attribute the relational structure extracted from the skeleton in order to achieve a richer description of shape. Furthermore, we demonstrate that there is an intimate relationship between the shape measure and the divergence of the Momentum field defined in the previous chapter. This is an important observation, since the divergence plays a central role when the skeleton is computed using the curvature-corrected Hamilton-Jacobi formalism to solve the eikonal equation.

Among the properties exhibited by this measure, we have that topological changes on the skeleton correspond to zero crossings. This means that ligatures are associated with a value of the measurement which is zero, and hence have neutral weight. Secondly, the measure does not change when the shape undergoes "bending".

4.2 The Shape-Measure and its Properties

The eikonal equation induces a map from a point in the skeleton to a set of points on the boundary of the shape. That is, there is a correspondence between a point on the skeleton and the set of points on the boundary whose trajectories intercept it under the motion induced by the eikonal equation. The cardinality of this set of corresponding points on the boundary can be used to classify the local topology of the skeleton in the following manner

- the cardinality is greater than or equal to 3 for junctions.
- for endpoints the cardinality is a number from 1 to a continuum.
- for the general case of points on branches of the skeleton, the cardinality is exactly 2.

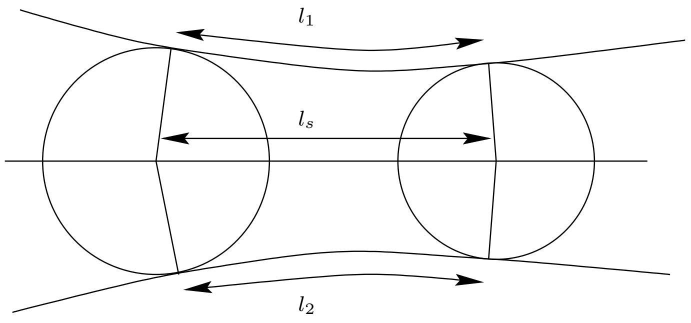


Figure 4.1: Geometric quantities used in our analysis.

As a result of this final property, any segment of a skeleton branch l_s is in correspondence with two boundary segments l_1 and l_2. This allows us to assign to a portion of the skeleton the portion of the boundary from which it arose. For each internal point in a skeleton branch, we can thus define the local ratio between the length of the generating boundary segment and the length of the generated skeleton segment. The rate of change of boundary length with skeleton length is defined as

$$\frac{dl}{dl_s} = \frac{dl_1}{dl_s} + \frac{dl_2}{dl_s}. \tag{4.1}$$

This ratio is our measure of the relevance of a skeleton segment in the representation of the 2D shape-boundary.

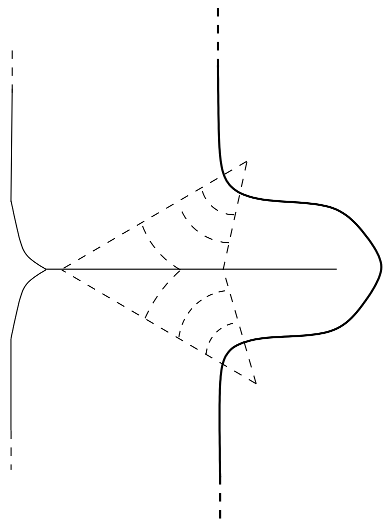


Figure 4.2: Ligature points are generated by short boundary segments.

Our proposal is to use this ratio as a measure of the local relevance of the skeleton to the boundary-shape description. In particular, we are interested in using the measure to identify ligatures [4]. Ligatures are skeleton segments that link the logically separate components of a shape. They are characterized by a high negative curvature on the generating boundary segment. The observation which motivates this proposal is that we can identify ligature by attaching to each infinitesimal segment of skeleton the length of the boundary that generated it. Under the eikonal equation, a boundary segment with high negative curvature produces a rarefaction front. This front will cause small segments to grow in length throughout their evolution, until they collide with another front and give rise to a so-called shock. This means that very short boundary segments generate very long skeleton branches. Consequently, when a skeleton branch is a ligature, then there is an associated decrease in the boundary-length to shock-length ratio. As a result our proposed skeletal shape measure “weights” ligature less than other points in the same skeleton branch.

To better understand the rate of decrease of the boundary length with skeletal length, we investigate its relationship to the local geometry of the bitangent circle inscribed within the object boundary. We have

$$\frac{dl_1}{dl_s} = \frac{\cos\theta}{1 - rk_1} \tag{4.2}$$

and, similarly,

$$\frac{dl_2}{dl_s} = \frac{\cos\theta}{1 - rk_2}, \tag{4.3}$$

where r is the radius of the bitangent circle and k_i is the curvature of the mapped segment on the boundary. The curvature is oriented inwards, that is, when the boundary bends towards the skeleton we have a positive curvature, while when the boundary bends away from the skeleton the curvature is negative. Finally, θ is the angle between the tangent to the skeleton and the tangent to the corresponding point on the boundary. These formulae show that the measure is inversely proportional to negative curvature and radius. That is, if we fix a negative curvature k_1, the measure decreases as the skeleton gets further away from the border. Furthermore, the measure decreases faster when the curvature becomes more negative.

A second important property of the shape-measure is that its value varies smoothly across shape deformations, even when these deformations impose topological transitions to the skeleton. To demonstrate this property we make use of the taxonomy of topological transition of the skeleton compiled by Giblin and Kimia [29]. According to this taxonomy, a smooth deformation of the shape induces only two types of transition on the skeleton (plus their time reversals). The transitions are *branch contraction* and *branch splicing.* A deformation *contracts* a branch joining two junctions when it moves the junctions together. Conversely, it *splices* a branch when it reduces in size, smoothes out, or otherwise eliminates the protrusion or sub-part of a shape that generates the branch.

A deformation that contracts or splices a skeleton branch causes the global value of the shape-measure along the branch to go to zero as the deformation approaches the topological transition. This means that a decreasing length of boundary generates the branch, until the branch disappears altogether.

When a deformation causes a contraction transition, both the length of the skeleton branch and the length of the boundary segments that generate the branch go to zero. A more elusive case is that of splicing. Through a splicing deformation, a decreasing length of boundary maps to the skeleton branch. This is either because the skeleton length and its associated boundary length are simultaneously reduced, or because the deformation allows boundary points to be mapped to adjacent skeleton branches. For this reduction in the length of the generating boundary, we do not have a corresponding reduction of the length of the skeleton branch. In fact, in a splice operation the length of the skeleton branch is a lower bound imposed by the presence of the ligature. This is the major cause of the perceived instability of the skeletal representation. Weighting each skeletal branch with the length of the corresponding boundary segments allows us to eliminate the contributions from ligatures, thus smoothing the instability. Since a smooth shape deformation induces a smooth change in the boundary, the total shape-measure along the branch has to vary smoothly through any deformation.

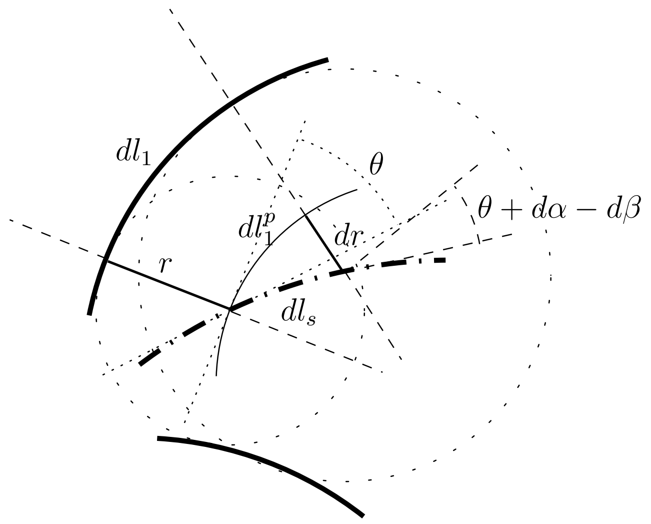


Figure 4.3: Differential geometry of a skeletal branch.

Moreover, just like the radius of the bitangent circle, key shape elements such as necks and seeds are associated with local variations of the length ratio. For instance, a neck is a point of high rarefaction and, thus, a minimum of the shape-measure along the branch. A seed is a point where the front of the evolution of the eikonal equation concentrates, and so is characterized by a maximum of the ratio.

A third important property of the shape-measure is its invariance to bending of the shape. This invariance derives from the fact that, if we bend the shape, we lose from one side the same amount of boundary-length that we gain on the opposite side. This property was already identified in [7].

To prove the bending invariance, let k_s be the curvature on the skeleton, at point p. We can assume, without loss of generality that at this point the skeleton is directed towards the border-segment dl_2. Furthermore, let k_1 and k_2 be the inward curvature on the corresponding boundary points, and let θ be the angle between the border tangents and the skeleton tangent. At the point p the tangent angle and the radius are linked by the relation

$dr/||dl_s|| = \sin(\theta)$. We define the *radius curvature* k_r as

$$k_r = \frac{d\theta}{||dls||} = \frac{d^2r/||d^2l_s||}{\sqrt{1-(dr/||dl_s||)^2}}. \tag{4.4}$$

This quantity represents the degree to which the boundary bends towards the skeleton. Positive values indicate that the boundary is convex with respect to the skeleton (i.e. bends towards the skeleton), negative values that the boundary is concave with respect to the skeleton (i.e. bends away from the skeleton). Let us now consider the segments dl_1^p and dl_2^p which are parallel to the border-segments dl_1 and dl_2, and which cross the skeleton at point p. The length of these segments is $||dl_1^p|| = ||dl_2^p|| = \cos(\theta)||dl_s||$, and their curvatures are k_1^p and k_2^p respectively. Moving along the skeleton by a distance $||dl_s||$, the tangent to the skeleton rotates by an angle $d\alpha = k_s||dl_s||$, while the tangent at the corresponding border segment dl_1^p rotates by an angle $d\beta = k_1^p||dl_p|| + O(||dl_s||^2)$. These angles are linked by the relationship

$$\begin{gathered}\frac{d\theta}{||dl_s||} = \frac{d\beta}{||dl_s||} + \frac{d\alpha}{||dl_s||}\\ k_r = k_1^p\cos(\theta) + k_s\\ k_1^p = \frac{k_r-k_s}{\cos(\theta)}.\end{gathered} \tag{4.5}$$

Similarly, remembering that on the opposite side of the skeleton, since k_s points towards dl_2, we have that

$$\begin{gathered}\frac{d\theta}{||dl_s||} = \frac{d\beta}{||dl_s||} - \frac{d\alpha}{||dl_s||}\\ k_r = k_2^p\cos(\theta) - k_s\\ k_2^p = \frac{k_r+k_s}{\cos(\theta)}.\end{gathered} \tag{4.6}$$

Recalling that $\frac{1}{k_1} = \frac{1}{k_1^p}+r$ and $\frac{1}{k_2} = \frac{1}{k_2^p}+r$, we have $||dl_1|| = [\cos(\theta)+r(k_r+k_s)]$ and $||dl_2|| = [\cos(\theta) + r(k_r - k_s)]$. Hence $||dl_1|| + ||dl_2||$ is independent of k_s, since if we bend the object sufficiently to cause a curvature k_s in the skeleton, the increase in boundary length on the one side is compensated by the decrease in boundary length on the opposite side.

4.3 Measure Extraction

The extraction of the skeletal shape-measure is a natural by-product which comes for free when we use the curvature-corrected Hamilton-Jacobi approach for skeleton extraction. This is a very important property of this shape-measure. By means of the divergence theorem, we can transport a quantity linked to a potentially distant border to a quantity local to the skeleton. In the previous chapter we proved that the border length to shock length ratio is proportional to the normalized divergence of the gradient of the momentum field. Furthermore, the flux though a circular area A containing a skeletal segment dl_s is

$$\Phi_A(\rho\vec{F}) = ||dl_1|| + ||dl_2||. \tag{4.7}$$

Given that in order to extract the skeleton we approximate the divergence using a unit-radius circle

$$\nabla \cdot (\rho\vec{F}) \approx \frac{\Phi_A(\rho\vec{F})}{\pi}, \tag{4.8}$$

we have

$$\frac{||dl_1||}{||dl_s||} + \frac{||dl_2||}{||dl_s||} = \frac{\Phi_A(\rho\vec{F})}{2\pi} \approx \frac{1}{2}\nabla \cdot (\rho\vec{F}). \tag{4.9}$$

Hence, the calculation of the skeletal measure comes for free when we extract the skeleton using the curvature-corrected Hamiton-Jacobi approach.

Figure 4.4 plots at each skeletal point the extracted value of the shape measure of each of two sample shapes.

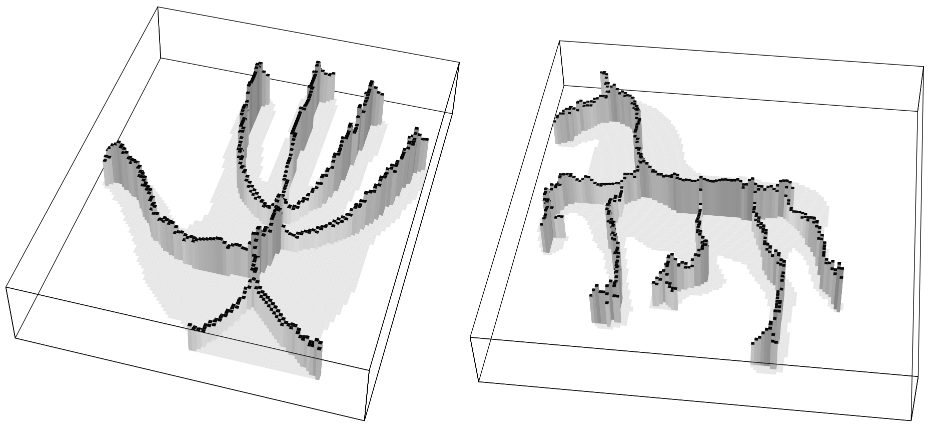


Figure 4.4: Two sample shapes. The height and intensity of the skeleton at each point is proportional to the shape measure.

4.4 Computing the Distance Between Skeletons

This result allows us to calculate a global shape-measure for each skeleton branch during the branch extraction process. For our matching experiments we have used a simple graph representation where the nodes are branches of the skeleton. When we have completed the thinning of the shape boundary and we are left only with the skeleton, we select an endpoint and start summing the values of the length ratio for each skeleton point until we reach either a junction or an extremal point. This sum $\sum_{i \in l_s} \nabla \cdot \vec{F}(\vec{x}_i)$ over every pixel x_i of our extracted skeleton branch is an approximation of

$$\int_{l_s} \nabla \cdot \vec{F}\, dl_s = \int_{l_s} (\frac{||dl_1||}{||dl_s||} + \frac{||dl_2||}{||dl_s||})\, dl_s = ||l_1|| + ||l_2||, \tag{4.10}$$

which is the length of the border that generates the skeleton branch.

At this point we have identified a branch and we have calculated the total value of the length-ratio along that branch, or, in other words, we have computed the total length of the border that generated the branch.

We continue this process until we have spanned each branch in the entire skeleton. Thus we obtain a weighted graph representation of the skeleton. In the case of a simple shape, i.e. a shape with no holes, the graph has no cycles and thus is an (unrooted) tree.

Given this representation, we can cast the problem of computing distances between different shapes as one of calculating the total difference in shape measure between corresponding branches.

4.5 Experimental Results

In this section we experiment with the new skeletal similarity measure. The experimentation is divided into two parts. First, we asses the ability of the proposed measure to discriminate between deformed shapes that give rise to skeletons with the same topology. Second, we asses how smoothly the overall similarity measure varies through transitions in skeletal topology.

There is clearly an underlying correspondence problem that must be solved before the similarity between two skeletons can be computed. This arises because we need to know how to associate branches in the two skeletons being compared. To fully perform a shape recognition task, we should recover these correspondences automatically. However, the aim of this chapter is to analyze the properties of our length ratio measure independently of the correspondence process. Therefore, for the set of experiments reported on here, we have located the branch correspondences by hand. The subsequent chapters of the thesis will present our approach to the remaining steps for the shape recognition task. In particular, the automatic recovery of the correspondences will be dealt with in the next chapter.

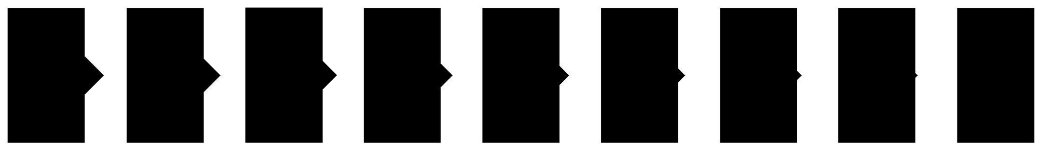


Figure 4.5: A "disappearing" protrusion which causes instability in shock-length, but not in our measure.

4.5.1 Stability under Deformation

As demonstrated earlier, we know that the length ratio measure should be stable with respect to any local shape deformation, including those that exhibit an instability in the length of a skeletal branch. This kind of behavior at local deformations is what has led to the idea that the skeleton is an unstable representation of shape.

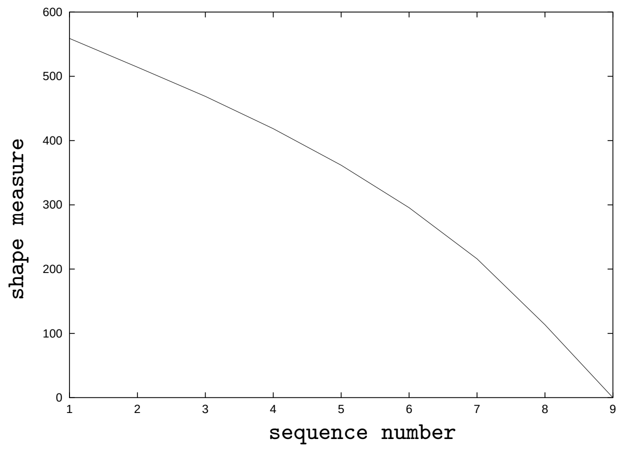


Figure 4.6: The measure of the skeleton segment generated by a protrusion.

To demonstrate the stability of the skeletal representation when augmented with the length ratio measurement, we have generated a sequence of images of a rectangle with a protrusion on one side (Figure 4.5). The size of the protrusion is gradually reduced throughout the sequence, until it is completely eliminated in the final image. In Figure 4.6 we plot the global value of the length-ratio measure for the shock branch generated by the protrusion. It is clear that the value of the length-ratio measure decreases monotonically and quite smoothly until it becomes zero when the protrusion disappears.

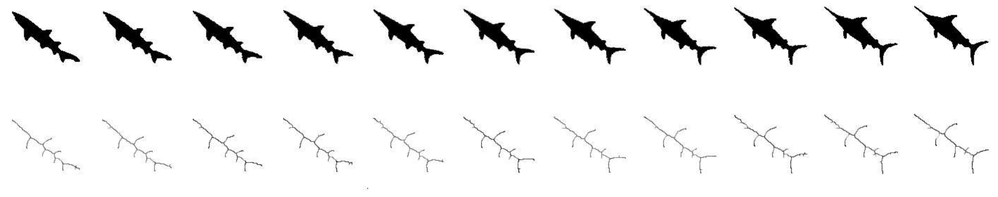
4.5.2 Changes in Skeleton Topology

In a second set of experiments we have aimed to assess the ability of the length-ratio measure to distinguish between structurally similar shapes. To do this we selected two shapes that were perceptually different, but which had skeletons with a very similar topology. We, then, generated an image sequence in which the two shapes were morphed into one another. Here the original shapes are the start and end frames of the sequence. At each frame in the sequence we calculated the distance between the start and end shapes.

We have repeated this experiment with two morphing sequences. The first sequence involved morphing a sand shark into a swordfish, while the second morphed a donkey into a hare.

To determine the difference between two shapes we made use of hand-picked correspondences between skeletal branches. The distance between the complete skeletons was defined as the Euclidean distance between the normalized weights of matched edges (skeletal branches). In other words, the distance is $D(A,B) = \sqrt{\sum_i (e_i^A - e_i^B)^2}$ where e_i^A and e_i^B are the normalized weights on the corresponding edges indexed by i on the shapes denoted by A and B. The normalized weights are computed by dividing the raw weights by the sum of the weights of each tree.

We apply this normalized length-ratio measure to ensure scale invariance. We note that two identical shapes scaled to different proportions would have different ratios due to the scale difference. However, the measure along equivalent branches of the two shapes would vary by a constant scale factor, namely the ratio of the lengths of the borders. Since the sum of the weights of the edges of a tree is equal to the total length of the border, by dividing the weights in each branch by this quantity we have reduced the two measurements to the same scale.


(a) sand shark to swordfish sequence

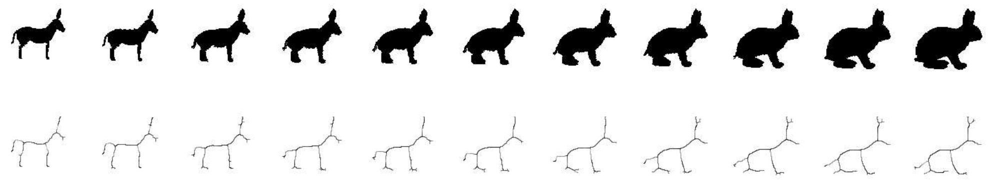
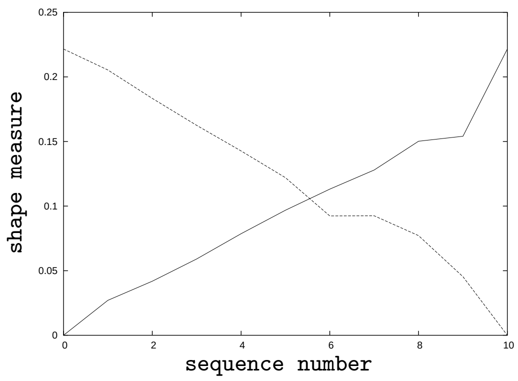

(b) donkey to hare sequence

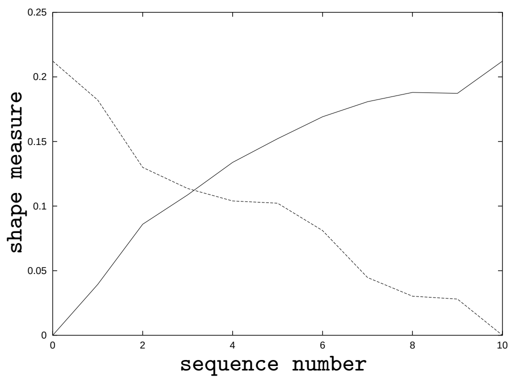
Figure 4.7: Morphing sequences and their corresponding skeletons.

For each morphing sequence, in Figure 4.8 we plot the distance between each frame in the sequence and the start and end frames. The monotonicity of the distance is evident throughout the sequences. This is proof of the ability of our length ratio measure to disambiguate between shapes with topologically similar skeletons.

To further asses the ability to discriminate between similar shapes, we selected a set of topologically similar shapes from a database of images of tools. As in the case of the previous experiments, the correspondences are hand-picked and the normalized Euclidean distance of the corresponding branch weights is used to measure the similarity of the skeletons. In the first column



(a) Distances in fish morphing sequence



(b) Distances in donkey to hare morphing sequence

Figure 4.8: Distances from first and last frame of the morphing sequences.

of Figure 4.9 we show the selected shapes. To their right are the remaining shapes sorted by increasing normalized distance. Each shape is annotated by the value of the normalized distance.

It is clear that similar shapes are usually closest to one another. However, there are problems due to a high sensitivity to occlusion, made evident by the high relative importance given to the articulation angle. This is due to the fact that, in the pliers images, articulation occludes part of the prongs of the pliers. While sensitivity to occlusion is, without a doubt, a drawback of the measure, we have to take into account that skeletal representation is, in general, highly sensitive to occlusion.

The reason that the first monkey wrench is recorded as being more similar to the pliers than the second monkey wrench is due to sensitivity to articulation and to the "closeness" of the head to the handles. Since the second monkey wrench is almost closed, the skeleton branches of the handles have a reduced overall weight. Thus the process of normalizing the edge weights reduces the significance of the small yet salient head when it is compared to

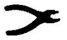
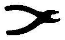
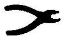
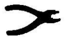
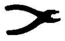
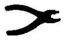
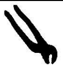
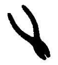
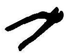
	0.068	 0.081	0.108	0.112	0.132	0.187
	 0.053	0.054	0.069	0.080	0.087	0.112
	 0.030	0.048	0.054	0.084	0.092	0.108
	0.021	0.030	0.054	0.081	0.086	0.110
	0.080	0.084	 0.086	0.095	0.102	0.132
	0.088	0.092	0.095	 0.110	0.128	0.186
	 0.021	0.048	0.068	 0.069	 0.102	 0.128

Figure 4.9: Some tools and the normalized distance between them.

the remainder of the shape.

4.6 Conclusions

This chapter introduces a shape-measure defined on the skeleton. This measure has been used in the literature as a branch-relevance measure during skeleton extraction and pruning. We show, however, that this measure has even greater informational utility, and can be used to augment the purely structural information residing in a skeleton in order to perform shape indexation and matching tasks. The shape-measure has a number of interesting properties that allow it to distinguish between structurally similar shapes. In particular, the measure a) changes smoothly through topological transitions of the skeleton, b) is able to distinguish between ligature and non-ligature points and to weight them accordingly, and c) exhibits invariance under "bending". What makes the use of this measure particularly appealing is the fact that it can be calculated with no added effort when the skeleton is computed using the skeletonization method presented in the previous chapter.

This chapter concludes our analysis and development of the skeletal representation. In the next chapter we begin a discussion of the matching of trees by introducing a method for computing approximate tree edit-distance.

Section II

Tree Matching

Chapter 5

Tree Edit-Distance

This chapter presents an energy minimization method for efficiently computing weighted tree edit-distance. Following the approach pioneered by Pelillo [60], we cast the problem in terms of the Motzkin-Straus framework. Using the graph-theoretic notion of tree closure, we show that, given a tree t, any tree obtained by cutting nodes from t is a subtree of the closure of t. Furthermore, subtrees of the closure of t that can not be obtained from t can be eliminated by solving a series of max-clique problems. This provides a divide and conquer method for finding the edit-distance by searching for maximal cliques of an auxiliary structure similar to Barrow and Burstall's association graph [5]. A variant of the Motzkin-Straus theorem introduced by Bomze et al. [9] is used to convert the maximum weighted clique problem into a quadratic programming problem which can be solved by relaxation labeling.

5.1 Problem Statement

The problem we aim to solve is the automatic extraction of correspondences between tree representations. Formally, given two trees $t_1 = (V_1, E_1)$ and

$t_2 = (V_2, E_2)$, where V_1 and V_2 are set of nodes and E_1 and E_2 set of edges, we wish to find the edit-distance between the two trees. That is, the sequence of basic edit operations that make t_1 and t_2 isomorphic with one another. Following common use, we consider three fundamental operations:

- *node removal:* this operation removes a node and links the children to the parent of said node.
- *node insertion:* the dual of node removal.
- *node relabel:* this operation changes the weight of a node.

Since a node insertion on the data tree is dual to a node removal on the model tree, we can reduce the number of operations to be performed to only node removal and node relabeling, as long as we perform the operations on both trees. Clearly, the cost of removing a node in the model tree must be equal to the cost of inserting it in the data tree. We assign a cost r_v to the operation of removing node v and a cost m_{vu} to matching node v to node u (that is, the minimum cost of relabeling nodes v and u to a common label).

With these definitions, the edit-distance between trees t_1 and t_2 is:

$$d(t_1, t_2) = \min_S \Big[\sum_{v \in R_1} r_v + \sum_{u \in R_2} r_u + \sum_{(v,u) \in M} m_{vu} \Big], \qquad (5.1)$$

where S is sequence of edit operations, R_1 and R_2 are the sets of nodes of t_1 and t_2, respectively, that are removed by S, and $M \in V_1 \times V_2$ is the set of matches between nodes of t_1 and t_2 induced by S.

The edit-distance approach is general in the sense that, by applying different costs to the edit operations, it can be equally applied to unattributed trees and to trees with either symbolic or continuous-valued attributes. In particular, to solve the correspondence problem for unattributed trees, we set $r_v = 1$ and $m_{vu} = 0$ for each node v and u of the two trees. On the other hand, to solve the correspondence problem for shock trees attributed with the weight described in the previous chapter, we set $r_v = w_v$ and $m_{vu} = |w_v - w_u|$, where w_v is the weight assigned to node v. Obviously, other cost assignments are possible.

It is easy to see that the cost of the edit-sequence is completely determined by the nodes in the two trees that are matched to one-another. In fact, given the optimal edit-sequence S, we have:

$$\begin{aligned} d(t_1, t_2) &= \textstyle\sum_{v \in R_1} r_v + \sum_{u \in R_2} r_u + \sum_{(v,u) \in M} m_{vu} \\ &= \textstyle\sum_{v \in V_1} r_v + \sum_{u \in V_2} r_u - \sum_{(v,u) \in M} \big(r_v + r_u - m_{vu} \big). \end{aligned} \tag{5.2}$$

Since $\sum_{v \in V_1} r_v$ and $\sum_{u \in V_2} r_u$ are constant and independent from S, the edit-distance is completely determined by the set of matches that maximize the utility

$$\mathcal{U}(M) = \sum_{(v,u) \in M} \big(r_v + r_u - m_{vu} \big). \tag{5.3}$$

At this point we introduce the concept of an *edited isomorphism*. Let us assume that we have two trees $t_1 = (V_1, E_1)$ and $t_2 = (V_2, E_2)$. Furthermore, let $t' = (V', E')$ be a tree that can be obtained from both t_1 and t_2 with node removal and relabel operations. The correspondences M_1 and M_2 between the nodes of t' and the nodes of t_1 and t_2, respectively, will induce an isomorphism $M' = M_1^{-1} \circ M_2$ between nodes in t_1 and t_2. This isomorphism places two nodes in correspondence with each other if and only if they are mapped to the same node in t'. We call this isomorphism an edited isomorphism induced by t'. We say that the isomorphism induced by this tree is a *maximum edited isomorphism* if it maximizes the total utility $\mathcal{U}(M')$. Clearly, finding the maximum edited isomorphism is equivalent to solving the tree edit-distance problem.

5.2 Association Graph and the Maximum Common Subtree Problem

First we describe a polynomial-time algorithm for the subtree isomorphism problem. This allows us to formalize some concepts and provide a starting point to extend the approach to the minimum tree edit-distance problem.

Let $G = (V, E)$ be a graph, where V is the set of nodes (or vertices) and $E \subseteq V \times V$ is the set of directed edges. With the notation $v \rightsquigarrow u$ we shall mean that there is a directed edge going from node v to node u. If there is a directed path from v to u, we shall write $v \dashrightarrow u$. Hierarchical trees have a canonical order relation $\mathcal{O}$ induced by paths: given two nodes v and u, we have $(v, u) \in \mathcal{O} \Leftrightarrow v \dashrightarrow u$. That is, two nodes are in the canonical relation if and only if there is a path connecting them. This relation can be shown to be an (irreflexive) order relation.

The phase-space we use to represent the matching of nodes is the directed association graph. This is a variant of the association graph, a structure that is frequently used in graph matching problems [5, 62]. The association graph $G_A = (V_A, E_A)$ of two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ has node set $V_A = V_1 \times V_2$ equal to the Cartesian products of nodes of the graphs to be matched. Hence, each node represents a possible association, or match, of a node in one graph to a node in the other. The edges represent the pairwise constraints of the problem. In particular, they represent both connectivity on the original graphs and the feasibility of a solution having both associations linked by an edge. The use of directed arcs in the association graph allows us to make use of the order provided by the tree hierarchies. For the exact isomorphism problem (maximum common subgraph) the edges of the association graph $G_A = (V_1 \times V_2, E_A)$ of two graphs $G_1 = (V_1, E_1)$ and

$G_2 = (V_2, E_2)$ are:

$$(v, v') \rightsquigarrow (u, u') \text{ iff } v \rightsquigarrow u \text{ and } v' \rightsquigarrow u', \tag{5.4}$$

where $v, u \in V_1$ are nodes of graph G_1 and $v', u' \in V_2$ are nodes of graph G_2. The graph obtained can be shown to be ordered still. Specifically, an association graph for the tree isomorphism problem can be shown to be a forest.

Proposition 1 *The directed association graph of two directed acyclic graphs (DAGs) G and G' is acyclic.*

Proof. Let us assume that $(u_1, v_1) \rightsquigarrow \ldots \rightsquigarrow (u_n, v_n)$ is a cycle. Then, since an arc $(v, v') \rightsquigarrow (u, u')$ in the association graph exists only if the arcs $v \rightsquigarrow u$ and $v' \rightsquigarrow u'$ exist in G and G' respectively, we have that $u_1 \rightsquigarrow \ldots \rightsquigarrow u_n$ is a cycle in G and $v_1 \rightsquigarrow \ldots \rightsquigarrow v_n$ is a cycle in G' against the hypothesis that they are DAGs. $\square$

Proposition 2 *The directed association graph of two trees t and t' is a forest.*

Proof. We already know that the association graph is a DAG, we have to show that for each node (u, u') there is at most one node (v, v') such that $(v, v') \rightsquigarrow (u, u')$. Due to the way in which the association graph is constructed this means that either u or u' must have at most one incoming edge. But t and t' are trees, so both u and u' have at most one incoming edge, namely the one that originates from the parent. $\square$

The directed association graph can be used to reduce a tree matching problem into subproblems using a divide-and-conquer approach. We call the maximum (weight) common subtree rooted at (v, v') a solution to the maximum (weight) common subtree problem applied to two subtrees of t and t'. In particular, the solution is constrained to the subtrees of t and t' rooted at v and v' respectively. This solution is further constrained by the condition that v and v' are roots of the matched subtrees.

With the maximum rooted common subtree problem for each child of (v, v') at hand, the maximum isomorphism rooted at (v, v') can be reduced to a maximum weight bipartite match problem between the set U of the children of v and the set U' of the children of v'.

Let $B = (U \cup U', E)$ be a bipartite graph with partitions U and U' and $w : E \rightarrow \mathbb{R}$ be a weight function on the edges of B. A bipartite match is a set of non-adjacent edges of B. The maximum weight bipartite match is the set of non-adjacent edges with maximum total weight. The search for a maximum weight bipartite match is a well known linear programming problem with several very efficient polynomial time algorithms to solve it [59].

The two partitions V and V' of the bipartite match consist of the children of v and v' respectively. The weight of the match between $u \in V$ and $u' \in V'$ is the sum of the matched weights of the maximum isomorphism rooted at (u, u'). In the case of a un-weighted tree this is the cardinality of the isomorphism. This structure provides us with a one-to-one relationship between matches in the bipartite graph and the children of (v, v') in the association graph. The solution of the bipartite matching problem identifies a set of children of (v, v') that satisfy the constraint of matching one node of t to no more than one node of t'. Furthermore, among such sets is the one that guarantees the maximum total weight of the isomorphism rooted at (v, v'). See [63] for a similar approach applied to the subtree problem.

The maximum isomorphism between t and t' is a maximum isomorphism rooted at (v, v'), where either v or v' is the root of t or t' respectively. This reduces the isomorphism problem to $n + m$ rooted isomorphism problems, where n and m are the cardinalities of t and t'. Furthermore, since there are $n \times m$ nodes in the association graph, the problem is reduced to a set of $n \times m$ maximum bipartite matching problems, each of which can be solved with known polynomial time algorithms. In what follows, we will extend this approach to the minimum weighted tree-edit problem and present an evolutionary method to conquer the subproblems.

5.3 Inexact Tree Matching

We want to extend the algorithm described in the previous section to develop an error-tolerant method for locating tree isomorphisms. There is a strong connection between the computation of the maximum common subtree and the tree edit-distance. Bunke [14] showed that, under certain constraints applied to the edit-costs, locating the maximum common subgraph problem and computing the minimum graph edit-distance are computationally equivalent to one another.

This is not directly true for trees, because of the added constraint that a tree must be connected. However, extending the concept to the common edited subtree, we can use common substructures to find the minimum-cost edited tree-isomorphism.

5.3.1 Editing the Transitive Closure of a Tree

Given a tree $t = (V, E)$, we define the *closure* $\Omega(t) = (V, E_\Omega)$ to be a directed acyclic graph with the same node set and with edges satisfying

$$u \rightsquigarrow v \text{ in } \Omega(t) \iff u \dashrightarrow v \text{ in } t. \tag{5.5}$$

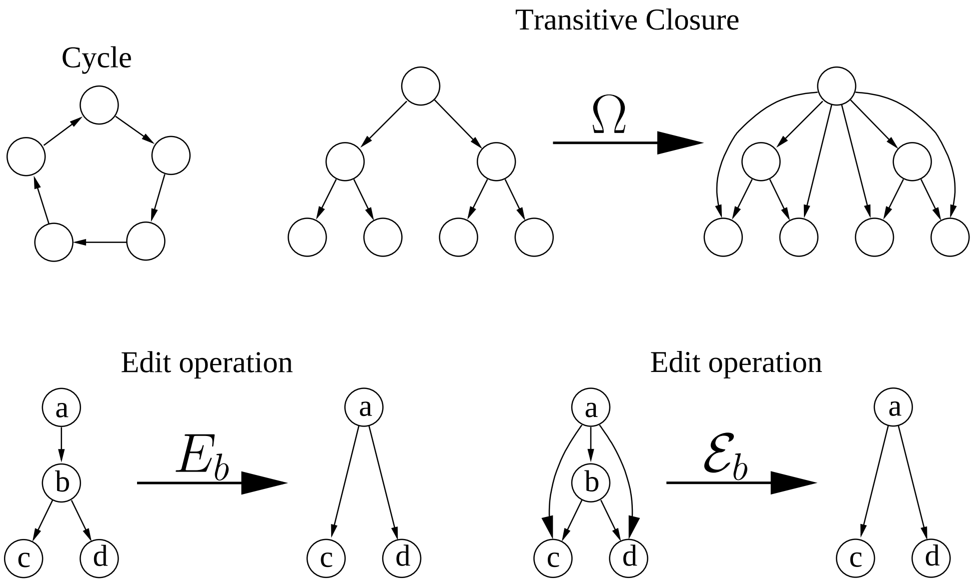


Figure 5.1: Terminology on directed graphs.

Clearly, t and $\Omega(t)$ are subject to the same order relation $\mathcal{O}$ between their nodes. Furthermore, we have $u \rightsquigarrow v$ in $\Omega(t) \iff u\mathcal{O}v$, i.e. the edge set of $\Omega(t)$ is a complete description of $\mathcal{O}$.

For each node v of t, we can define an edit operation E_v on the tree and an edit operation $\mathcal{E}_v$ on the closure $\Omega(t)$ of the tree t (see Figure 5.1). In both cases the edit operation removes the node v, all the incoming edges, and all the outgoing edges.

We show that the transitive closure operation and the node removal operation commute. That is, we have

Lemma 1 $\mathcal{E}_v(\Omega(t)) = \Omega(E_v(t))$.

Proof. If a node is in $\mathcal{E}_v(\Omega(t))$ it is clearly also in $\Omega(E_v(t))$. What remains to be shown is that an edge (a, b) is in $\mathcal{E}_v(\Omega(t))$ if and only if it is in $\Omega(E_v(t))$.

If (a, b) is in $\Omega(E_v(t))$ 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 $\mathcal{O}$, there must be a path from a to b in t as well. This implies that (a, b) is in $\Omega(t)$. Since neither a nor b is v, the operation $\mathcal{E}_v$ will not delete (a, b). Thus (a, b) is in $\mathcal{E}_v(\Omega(t))$.

If (a, b) is in $\mathcal{E}_v(\Omega(t))$, then it is also in $\Omega(t)$, because $\mathcal{E}_v(\Omega(t))$ is obtained from $\Omega(t)$ by simply removing a node and some edges. This implies that there is a path from a to b in t and also, as long as neither a nor b are v, there is a path from a to b in $E_v(t)$ as well. Thus (a, b) is in $\Omega(E_v(t))$. Since (a, b) is in $\mathcal{E}_v(\Omega(t))$, both a and b must be nodes in $\mathcal{E}_v(\Omega(t))$ and, thus, neither can be v. $\square$

Furthermore, the transitive closure operation clearly commutes with node relabeling as well, since one acts only on weights and the other acts only on node connectivity.

We say that two nodes a and b of tree t are *independent* if there is no path from a to b or from b to a, that is if neither is a descendent of the other in t. We call a subtree s of $\Omega(t)$ *obtainable* if for each node v of s there cannot be two children a and b so that the edge (a, b) is in $\Omega(t)$. In other words, s is obtainable if and only if every pair of siblings in s are independent.

We can now prove the following:

Theorem 1 *A tree $\hat{t}$ can be obtained from a tree t with an edit sequence composed of only node removal and node relabeling operations if and only if $\hat{t}$ is an obtainable subtree of* $\Omega(t)$.

Proof. Let us assume that there is an edit sequence $\{E_{v_i}\}$ that transforms t into $\hat{t}$, then, by virtue of the Lemma 1, the dual edit sequence $\{\mathcal{E}_{v_i}\}$ transforms $\Omega(t)$ into $\Omega(\hat{t})$. By construction we have that $\hat{t}$ is a subtree of $\Omega(\hat{t})$ and $\Omega(\hat{t})$ is a subgraph of $\Omega(t)$, thus $\hat{t}$ is a subtree of $\Omega(t)$. Furthermore, since the node-removal operations respect the hierarchy, $\hat{t}$ is an obtainable subtree of

$\Omega(t)$.

To prove the converse, assume that $\hat{t}$ is an obtainable subtree of $\Omega(t)$. If (a, b) is an edge of $\hat{t}$, then it is an edge on $\Omega(t)$ as well, 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 in 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 a descendant of a, an ancestor of a, or a itself. This means that w has to be in the edited path. Were it not so, then w would have 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, q is an ancestor of w in t, but q and w are siblings in $\hat{t}$ against the hypothesis that $\hat{t}$ is obtainable.
$\square$

By virtue of Theorem 1, every common edited isomorphism between tree t and tree t' induces a consistent subtree of both $\Omega(t)$ and $\Omega(t')$. Since minimizing the edit-cost and maximizing the utility are equivalent, the set of correspondences of the minimum-cost edited tree-isomorphism can be found by searching for the consistent subtree with maximum utility. As a consequence, finding a minimum-cost edit-sequence is equivalent to finding a maximum utility common obtainable subtree of $\Omega(t)$ and $\Omega(t')$.

5.3.2 Cliques and Common Obtainable Subtrees

In this section we show that the directed association graph induces a divide-and-conquer approach to edited tree matching as well. Given two trees t and t' to be matched, we create the directed association graph of the transitive closures $\Omega(t)$ and $\Omega(t')$ and we search for an obtainable matching tree in the graph. That is, we seek a tree in the graph that corresponds to two obtainable trees in the transitive closures $\Omega(t)$ and $\Omega(t')$. Any such tree having maximum utility induces the optimal set of node-correspondence between t and t'.

By analogy to what we did for the exact matching case, we divide the problem into a maximum common obtainable subtree rooted at (v, w), for each node (v, w) of the association graph. We show that, given the utility of the maximum common consistent subtree rooted at each child of (v, w) in the association graph, we can transform the rooted subtree problem into a *maximum weighted clique problem.* A *clique* of a graph $G = (V, E)$ is a complete, or fully connected, subgraph of G. A *maximum (unweighted) clique* is a clique with maximum node cardinality among all cliques of G, while a *maximum weighted clique* of a weighted graph G is a clique with maximum total weight among all cliques of G. The search for a clique with maximum weight is a well-known NP-hard problem. Solving this problem for each node in the association graph and looking for the one with maximum utility, we can find the solution to the minimum-cost edit-sequence problem and hence find the edit-distance.

Let us assume that we know the utility of the subtree for every child of the node (v, w) in the association graph. We want to find the set of *independent* siblings with greatest total utility. Let us construct an undirected graph whose nodes consist of the children of (v, w) in the association graph. We connect the two nodes (p, q) and (r, s) if and only if p and r are *independent* in t, and q and s are *independent* in t'. Furthermore, we assign to each association node (a, b) a weight equal to the utility of the maximum common obtainable subtree rooted at (a, b). The maximum weight clique of this graph will be the set of mutually independent siblings with maximum total weight. Let W be the weight of this clique, The utility of the maximum common obtainable subtree rooted at (v, w) will be

$$U^{(u,w)} = W + r_v + r_w - m_{vw}, \tag{5.6}$$

where r_v and r_w are the costs of removing nodes v and w respectively, while m_{vw} is the cost of matching v to w. Furthermore, the nodes of the clique will be the children of (v, w) in the maximum common consistent subtree.

5.4 Heuristics for Maximum Weighted Clique

We have transformed an inexact tree-matching problem into a series of maximum weighted clique problems. That is, we transformed one NP-hard problem into multiple NP-hard problems. The observation underlying this approach is the fact that the max clique problem is, on average, a relatively easy one and a large number of approaches and very powerful heuristics exist to solve it or to approximate it. Furthermore, since the seminal paper by Barrow and Burstall [5], transforming matching problems into max-clique problems has become a standard technique.

The approach we will adopt to solve each single instance of the max weight clique problem is an evolutionary one introduced by Bomze, Pelillo and Stix [9]. This approach is based on a continuous formulation of the combinatorial problem and transforms it into a symmetric quadratic programming problem.

In 1965, Motzkin and Strauss [54] showed that the (unweighted) maximum clique problem can be reduced to a quadratic programming problem on the n-dimensional simplex $\Delta = \{\mathbf{x} \in \mathbb{R}^n | x_i \geq 0 \text{ for all } i = 1 \ldots n, \sum_i x_i = 1\}$, where x_i are the components of vector $\mathbf{x}$. More precisely, let $G = (V, E)$ be a graph where V is the node set and E is the edge set, and let $C \subseteq V$ be a maximum clique of G, then the vector $\mathbf{x}^* = \{x_i^* = 1/\#C \text{ if } i \in C, 0 \text{ otherwise}\}$ maximizes in Δ the function $g(\mathbf{x}) = \mathbf{x}^T A \mathbf{x}$, where A is the adjacency matrix of G. Furthermore, given a set $S \subseteq V$, we define the *characteristic* vector $\mathbf{x}^S$

$$x_i^S = \begin{cases} 1/\#S & \text{if } i \in S \\ 0 & \text{otherwise.} \end{cases}$$

With this definition, S is a maximum (maximal) clique if and only if $g(\mathbf{x}^S)$ is a global (local) maximum for the function g.

Gibbons *et al.* [28] generalized this result to the weighted clique case. In their formulation the association graph is substituted with a matrix $\bar{A} = (\bar{a}_{ij})_{i,j \in V}$ related to the weights and connectivity of the graph by the relation

$$\bar{a}_{ij} = \begin{cases} 1/w_i & \text{if } i = j \\ k_{ij} \geq \frac{\bar{a}_{ii} + \bar{a}_{jj}}{2} & \text{if } (i,j) \notin E \\ 0 & \text{otherwise.} \end{cases} \tag{5.7}$$

Let us consider a weighted graph $G = (V, E, w)$, where V is the set of nodes, E the set of edges, and $w : V \to \mathbb{R}$ a weight function that assigns a weight to each node. Gibbons *et al.* proved that, given a set $S \subseteq V$ and its *characteristic* vector $\mathbf{x}^S$ defined as

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

then S is a maximum (maximal) weight clique if and only if $\mathbf{x}^S$ is a global (local) minimizer for the quadratic form $\mathbf{x}^T \bar{A} \mathbf{x}$. Furthermore, the weight of the clique S is $w(S) = \frac{1}{\mathbf{x}^{S\,T} \bar{A} \mathbf{x}^S}$.

Unfortunately, under this formulation, the minima are not necessarily isolated. As a result, when we have more than one clique with the same maximal weight, any convex linear combinations of their characteristic vectors will give the same maximal value. This implies that, if we find a minimizer $\mathbf{x}^*$ we can derive the weight of the clique. However, we might not be able to determine the nodes that constitute the clique.

Bomze, Pelillo and Stix [9] introduce a regularization factor to the quadratic programming method that generates an equivalent problem with isolated solutions. The new quadratic program minimizes $\mathbf{x}^T C\mathbf{x}$ in the simplex, where the matrix $C = (c_{ij})_{i,j\in V}$ is defined as

$$c_{ij} = \begin{cases} \frac{1}{2w_i} & \text{if } i = j \\ k_{ij} \geq c_{ii} + c_{jj} & \text{if } (i,j) \notin E, i \neq j \\ 0 & \text{otherwise.} \end{cases} \tag{5.8}$$

Once again, S is a maximum (maximal) weighted clique if and only if $\mathbf{x}^S$ is a global (local) minimizer for the quadratic program.

To solve the quadratic problem we transform it into the equivalent problem of maximizing $\mathbf{x}^T(\gamma\mathbf{e}\mathbf{e}^T - C)\mathbf{x}$, where $\mathbf{e} = (1, \cdots, 1)^T$ is the vector with every component equal to 1 and γ is a positive scaling constant.

To approximate the quadratic programming problem, we use relaxation labeling. Relaxation labeling is an evidence-combining process developed in the framework of constraint satisfaction. Its goal is to find a classification that assigns a label from a set $\Lambda = \{\lambda_1, \cdots, \lambda_m\}$ to a set of objects $O = \{o_1, \cdots, o_n\}$ that satisfies pairwise constraints and interactions between the objects and labels. The discrete assignment space is relaxed into a probability space $\Theta = (\Delta_m)^n$, where Δ_m is an m-dimensional simplex. Given a relaxed assignment $\mathbf{p}$, $p_i(\lambda)$ represents the probability that object $o_i \in O$ is classified with label $\lambda \in \Lambda$. The contraints to the possible assignmets are given in the form of mutual compatibility between pair of assignments. We indicate with $r_{ij}(\lambda, \mu)$ the degree of compatibility of assigning label λ to object o_i, given that object o_j is labeled μ. A relaxation labeling process takes as input the initial labeling assignment $\mathbf{p}^0$ and iteratively updates it taking into account the compatibility model. The evolution of the assignment is determined by the update rule

$$p_i^{t+1}(\lambda) = \frac{p_i^t(\lambda) q_i^t(\lambda)}{\sum_\mu p_i^t(\mu) q_i^t(\mu)}, \tag{5.9}$$

where the compatibility coefficient is $q_i(\lambda) = \sum_{j=1}^n \sum_{\mu=1}^m r_{ij}(\lambda, \mu) p_j(\mu)$.

Pelillo [61] showed that, when the compatibilities are symmetric, that is $r_{ij}(\lambda, \mu) = r_{ji}(\mu, \lambda)$, the function $A(\mathbf{p}) = \sum_{i,\lambda} p_i(\lambda) q_i(\lambda)$ is a Lyapunov function for the process, i.e. $A(\mathbf{p}^{t+1}) \geq A(\mathbf{p}^t)$, with equality if and only if $\mathbf{p}^t$ is a stationary point. Therefore, this process can be used to find local optima of a quadratic programming problem defined on Θ. By setting the number of objects equal to 1, the quadratic problem solved by the relaxation labeling process is

$$\begin{array}{l} \max \sum_i^m \sum_j^m p(\lambda_i) r(\lambda_i, \lambda_j) p(\lambda_j) \\ \text{subject to } p \in \Delta_m. \end{array} \tag{5.10}$$

By setting $r(\lambda_i, \lambda_j) = \gamma - c_{ij}$, this problem is equivalent to 5.8 above. Therefore, relaxation labeling can be used to approximate the maximum weighted clique problem and hence the maximum set of *independent* children of nodes v and v'. Each label λ_i of the labeling problem is in relation with a node (u_i, u_i') which is a child of (v, v') in the association graph. Upon convergence, the non-zero component of $\mathbf{p}^\infty$ are in correspondence with nodes of the children of (v, v') that from a clique of *independent* siblings with maximal weight. That is, we find a set S such that

$$(u_i, u_i') \in S \iff p^\infty(\lambda_i) > 0. \tag{5.11}$$

This set of correspondences is optimal subject to the fact that v is matched to v'. Hence, the weight of the match rooted at (v, v') is

$$W^{(v,v')} = r_v + r_{v'} - m_{vv'} + \sum_{(u,u')\in S} W^{(u,u')}. \tag{5.12}$$

In this way we can propagate the associations from the leaves of the directed association graph upwards, using the weight of the extracted cliques to initialize the compatibility matrix of every parent association. For a subproblem rooted at (u, v) the compatibility coefficients can be calculated knowing the weight of every isomorphism rooted at the descendants of u and v. Specifically, the compatibility coefficients are initialized as $r_{(u,v)}(a, a'b, b') = \gamma - c^{(u,v)}_{(a,a')(b,b')}$, where

$$c^{(u,v)}_{(a,a')(b,b')} = \begin{cases} \frac{1}{2W^{(a,a')}} & \text{if } (a,a') = (b,b') \\ c^{(u,v)}_{(a,a')(a,a')} + c^{(u,v)}_{(b,b')(b,b')} & \text{if } (a,a') \text{ and } (b,b') \text{ are } \textit{independent} \\ 0 & \text{otherwise.} \end{cases} \tag{5.13}$$

Once all the optimal associations have been calculated for all possible nodes of the association graph, we can determine the topmost match in the edited isomorphism by searching for the association with the maximum weight. Once the topmost match is found, we can construct the complete isomorphism by following the optimal matches obtained with relaxation labeling.

This optimization approach allows us to make use of problem specific information about the solutions. Usually, when solving the maximum clique problem, the label assignment probabilities are initialized with a uniform distribution so that the relaxation labeling process starts from a point 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, regardless of their weights. 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 large number of nodes and are, thus, less likely to give an optimum solution. Due to the nature of our problem decomposition, matches higher up in the hierarchy are more likely than matches lower down. For this reason, we initialize the assignment probabilities as $p^0(\lambda_i) = \frac{\hat{p}(\lambda_i)}{\sum_j \hat{p}(\lambda_j)}$, where

$$\hat{p}(\lambda_i) = \exp[-k(\text{depth}(u_i) + \text{depth}(u_i'))]. \qquad (5.14)$$

Here, k is a constant and depth(u_i) is the relative depth in the original tree t of node u_i with respect to node v.

5.5 Experimental Results

Our experimental evaluation is divided into two parts. We commence by evaluating the method for computing the tree edit-distance. Then, with the edit-distances to hand, we explore how they may be used to discover shape categories via pairwise clustering.

5.5.1 Shock Trees

The first set of experiments aims to establish the usefulness of the proposed edit-distance approach as a tool to compare shapes abstracted using shock trees. In these experiments we have used a database of shapes with very different topologies. Hence there is no simple way to locate the correspondences between the skeletal branches. Edit-distance was used to compute

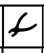
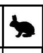
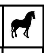
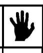
																									
	0.981	0.844	0.823	0.434	0.604	0.463	0.562	0.600	0.495	0.554	0.422	0.389	0.451	0.419	0.511	0.447	0.557	0.656	0.646	0.552	0.484	0.427	0.428	0.415	0.402
	0.844	0.981	0.736	0.548	0.685	0.559	0.683	0.720	0.552	0.509	0.381	0.533	0.550	0.538	0.621	0.553	0.629	0.744	0.748	0.613	0.534	0.447	0.474	0.517	0.434
	0.823	0.736	0.973	0.409	0.479	0.375	0.420	0.456	0.398	0.598	0.352	0.388	0.398	0.400	0.468	0.435	0.515	0.594	0.584	0.517	0.451	0.381	0.398	0.369	0.360
	0.446	0.543	0.437	1.000	0.569	0.764	0.554	0.643	0.637	0.475	0.651	0.425	0.475	0.473	0.436	0.525	0.559	0.559	0.550	0.411	0.428	0.395	0.406	0.414	0.373
	0.604	0.685	0.479	0.809	1.000	0.755	0.663	0.731	0.713	0.592	0.496	0.467	0.496	0.531	0.507	0.513	0.624	0.545	0.580	0.429	0.404	0.418	0.443	0.449	0.404
	0.404	0.486	0.352	0.790	0.718	1.000	0.563	0.572	0.504	0.390	0.813	0.355	0.393	0.425	0.405	0.446	0.468	0.491	0.476	0.328	0.314	0.340	0.382	0.381	0.339
	0.562	0.684	0.420	0.531	0.666	0.608	1.000	0.857	0.714	0.459	0.385	0.503	0.590	0.511	0.588	0.611	0.622	0.600	0.656	0.443	0.456	0.497	0.524	0.541	0.521
	0.571	0.650	0.415	0.643	0.750	0.572	0.857	1.000	0.790	0.584	0.505	0.527	0.545	0.512	0.580	0.593	0.670	0.580	0.585	0.462	0.456	0.515	0.514	0.480	0.503
	0.502	0.559	0.398	0.648	0.727	0.525	0.703	0.796	0.999	0.533	0.503	0.472	0.480	0.348	0.438	0.531	0.562	0.478	0.506	0.448	0.443	0.457	0.467	0.506	0.482
	0.554	0.606	0.612	0.475	0.592	0.390	0.459	0.531	0.554	0.999	0.415	0.411	0.443	0.319	0.434	0.402	0.405	0.492	0.404	0.468	0.443	0.459	0.419	0.400	0.408
	0.441	0.356	0.352	0.651	0.506	0.813	0.388	0.472	0.530	0.394	0.981	0.364	0.384	0.377	0.384	0.388	0.438	0.318	0.373	0.345	0.396	0.386	0.310	0.353	0.347
	0.450	0.536	0.388	0.399	0.415	0.355	0.506	0.516	0.483	0.411	0.367	0.896	0.853	0.692	0.600	0.593	0.584	0.617	0.615	0.458	0.576	0.611	0.741	0.687	0.744
	0.481	0.543	0.398	0.475	0.479	0.393	0.590	0.526	0.487	0.443	0.386	0.818	0.976	0.643	0.669	0.583	0.497	0.630	0.531	0.500	0.469	0.630	0.706	0.670	0.747
	0.393	0.475	0.400	0.456	0.486	0.425	0.581	0.477	0.404	0.367	0.370	0.756	0.559	0.848	0.755	0.788	0.784	0.616	0.626	0.539	0.396	0.608	0.534	0.572	0.524
	0.516	0.626	0.468	0.479	0.507	0.410	0.593	0.586	0.487	0.434	0.379	0.631	0.692	0.596	0.999	0.733	0.825	0.650	0.691	0.517	0.556	0.520	0.634	0.630	0.555
	0.470	0.553	0.432	0.510	0.495	0.463	0.616	0.627	0.531	0.385	0.392	0.572	0.583	0.605	0.702	0.998	0.747	0.678	0.670	0.519	0.473	0.557	0.606	0.484	0.569
	0.556	0.636	0.511	0.559	0.626	0.510	0.622	0.670	0.578	0.405	0.428	0.622	0.586	0.784	0.820	0.737	1.000	0.700	0.639	0.523	0.449	0.590	0.542	0.532	0.518
	0.651	0.711	0.594	0.537	0.559	0.495	0.600	0.606	0.489	0.393	0.371	0.585	0.607	0.631	0.635	0.678	0.692	1.000	0.632	0.695	0.522	0.553	0.517	0.537	0.552
	0.629	0.696	0.567	0.616	0.563	0.487	0.656	0.624	0.506	0.404	0.387	0.523	0.577	0.664	0.727	0.671	0.688	0.844	0.933	0.581	0.584	0.515	0.568	0.541	0.548
	0.499	0.613	0.481	0.466	0.429	0.328	0.477	0.488	0.464	0.468	0.319	0.499	0.511	0.534	0.537	0.558	0.538	0.707	0.610	0.998	0.556	0.610	0.539	0.492	0.546
	0.496	0.499	0.450	0.417	0.388	0.314	0.462	0.429	0.475	0.397	0.403	0.500	0.507	0.403	0.492	0.443	0.449	0.586	0.576	0.577	1.000	0.627	0.496	0.519	0.560
	0.395	0.447	0.368	0.395	0.441	0.340	0.495	0.488	0.445	0.497	0.371	0.663	0.629	0.466	0.602	0.612	0.573	0.532	0.544	0.554	0.693	0.992	0.711	0.686	0.699
	0.443	0.520	0.398	0.334	0.436	0.382	0.570	0.536	0.465	0.422	0.335	0.677	0.702	0.536	0.632	0.538	0.554	0.576	0.568	0.575	0.494	0.695	0.975	0.895	0.840
	0.431	0.450	0.391	0.409	0.425	0.380	0.520	0.520	0.480	0.445	0.378	0.685	0.668	0.573	0.629	0.546	0.543	0.505	0.586	0.577	0.572	0.718	0.846	0.981	0.771
	0.441	0.447	0.392	0.380	0.447	0.339	0.560	0.520	0.493	0.434	0.346	0.677	0.724	0.537	0.563	0.586	0.558	0.480	0.565	0.566	0.541	0.765	0.863	0.817	1.000

Figure 5.2: Pairwise similarities between shapes for the weighted shock trees.

both node correspondences and dissimilarity measures between shapes. We have compared the results obtained when the skeleton is weighted with the measure proposed in the previous chapter and when it is unweighted. In the weighted case the cost of adding or removing a node with weight w is equal

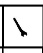
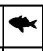
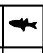
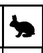
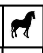
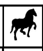
																									
	1.000	1.000	1000.	0.774	1.000	0.786	0.889	0.889	0.706	0.643	0.850	0.536	0.536	0.565	0.818	0.792	0.889	0.850	0.804	0.719	0.635	0.640	0.706	0.694	0.684
	1.000	1.000	1.000	0.774	1.000	0.786	0.889	0.889	0.706	0.643	0.729	0.536	0.536	0.565	0.818	0.792	0.889	0.850	0.804	0.719	0.635	0.640	0.706	0.694	0.684
	1.000	1.000	1.000	0.774	1.000	0.786	0.889	0.889	0.706	0.643	0.729	0.536	0.536	0.565	0.701	0.792	0.889	0.850	0.804	0.719	0.635	0.640	0.706	0.694	0.684
	0.774	0.774	0.774	1.000	0.774	0.833	0.833	0.833	0.676	0.714	0.800	0.714	0.714	0.500	0.773	0.625	0.694	0.667	0.875	0.688	0.615	0.620	0.676	0.667	0.658
	1.000	1.000	1.000	0.774	1.000	0.786	0.889	0.889	0.706	0.643	0.850	0.536	0.536	0.565	0.818	0.792	0.889	0.850	0.804	0.719	0.635	0.640	0.706	0.694	0.684
	0.786	0.786	0.786	0.833	0.786	1.000	0.722	0.722	0.618	0.643	0.700	0.643	0.643	0.667	0.682	0.667	0.722	0.700	0.750	0.625	0.577	0.580	0.618	0.611	0.605
	0.889	0.889	0.889	0.833	0.889	0.722	1.000	1.000	0.765	0.730	0.950	0.548	0.548	0.486	0.808	0.681	0.778	0.739	0.826	0.781	0.673	0.680	0.765	0.750	0.737
	0.889	0.889	0.889	0.833	0.889	0.722	1.000	1.000	0.765	0.730	0.950	0.548	0.548	0.486	0.808	0.681	0.778	0.739	0.826	0.781	0.673	0.680	0.765	0.75	0.737
	0.706	0.706	0.706	0.676	0.706	0.618	0.765	0.765	1.000	0.782	0.715	0.456	0.456	0.498	0.674	0.640	0.680	0.715	0.643	0.667	0.730	0.791	0.765	0.743	0.724
	0.643	0.643	0.643	0.714	0.643	0.643	0.730	0.730	0.782	1.000	0.857	0.500	0.500	0.619	0.568	0.542	0.639	0.600	0.688	0.938	0.714	0.724	0.586	0.635	0.744
	0.850	0.729	0.729	0.800	0.850	0.700	0.950	0.950	0.715	0.857	1.000	0.600	0.600	0.458	0.764	0.642	0.739	0.700	0.788	0.812	0.692	0.700	0.794	0.778	0.763
	0.536	0.536	0.536	0.714	0.536	0.643	0.548	0.548	0.456	0.500	0.600	1.000	1.000	0.619	0.487	0.387	0.456	0.429	0.589	0.469	0.549	0.669	0.651	0.635	0.620
	0.536	0.536	0.536	0.714	0.536	0.643	0.548	0.548	0.456	0.500	0.600	1.000	1.000	0.619	0.487	0.387	0.456	0.429	0.589	0.469	0.495	0.669	0.651	0.635	0.620
	0.565	0.565	0.565	0.500	0.565	0.667	0.486	0.486	0.498	0.619	0.458	0.619	0.619	1.000	0.523	0.500	0.583	0.550	0.521	0.438	0.609	0.678	0.640	0.625	0.612
	0.818	0.818	0.701	0.773	0.818	0.682	0.808	0.808	0.674	0.568	0.764	0.487	0.487	0.523	1.000	0.784	0.909	0.764	0.864	0.767	0.647	0.655	0.824	0.806	0.789
	0.792	0.792	0.792	0.625	0.792	0.667	0.681	0.681	0.640	0.542	0.642	0.387	0.387	0.500	0.784	1.000	0.875	0.917	0.729	0.656	0.548	0.617	0.711	0.694	0.680
	0.889	0.889	0.889	0.694	0.889	0.722	0.778	0.778	0.680	0.639	0.739	0.456	0.456	0.583	0.909	0.875	1.000	0.844	0.826	0.694	0.598	0.604	0.765	0.750	0.737
	0.850	0.850	0.850	0.667	0.850	0.700	0.739	0.739	0.715	0.600	0.700	0.429	0.429	0.550	0.764	0.917	0.844	1.000	0.788	0.731	0.623	0.700	0.715	0.622	0.687
	0.804	0.804	0.804	0.875	0.804	0.750	0.826	0.826	0.643	0.688	0.788	0.589	0.589	0.521	0.864	0.729	0.826	0.788	1.000	0.656	0.654	0.660	0.735	0.722	0.711
	0.719	0.719	0.719	0.688	0.719	0.625	0.781	0.781	0.667	0.938	0.812	0.469	0.469	0.438	0.767	0.656	0.694	0.731	0.656	1.000	0.757	0.666	0.667	0.708	0.633
	0.635	0.635	0.635	0.615	0.635	0.577	0.673	0.673	0.730	0.714	0.692	0.549	0.495	0.609	0.647	0.548	0.598	0.623	0.654	0.757	1.000	0.706	0.730	0.752	0.729
	0.640	0.640	0.640	0.620	0.640	0.580	0.680	0.680	0.791	0.724	0.700	0.669	0.669	0.678	0.655	0.617	0.604	0.700	0.660	0.666	0.706	1.000	0.840	0.860	0.880
	0.706	0.706	0.706	0.676	0.706	0.618	0.765	0.765	0.765	0.586	0.794	0.651	0.651	0.640	0.824	0.711	0.765	0.715	0.735	0.667	0.730	0.840	1.000	0.972	0.836
	0.694	0.694	0.694	0.667	0.694	0.611	0.750	0.750	0.743	0.635	0.778	0.635	0.635	0.625	0.806	0.694	0.750	0.622	0.722	0.708	0.752	0.860	0.972	1.000	0.811
	0.684	0.684	0.684	0.658	0.684	0.605	0.737	0.737	0.724	0.744	0.763	0.620	0.620	0.612	0.789	0.680	0.737	0.687	0.711	0.633	0.729	0.880	0.836	0.811	1.000

Figure 5.3: Pairwise similarities between shapes for the unweighted shock trees.

to the weight itself, while the cost of matching two nodes with weights w and w' respectively is $|w - w'|$. In the unweighted case the node-removal cost was set at 1, while the matching cost was set at 0.

Given the topological diversity of the shape skeletons, we have used a

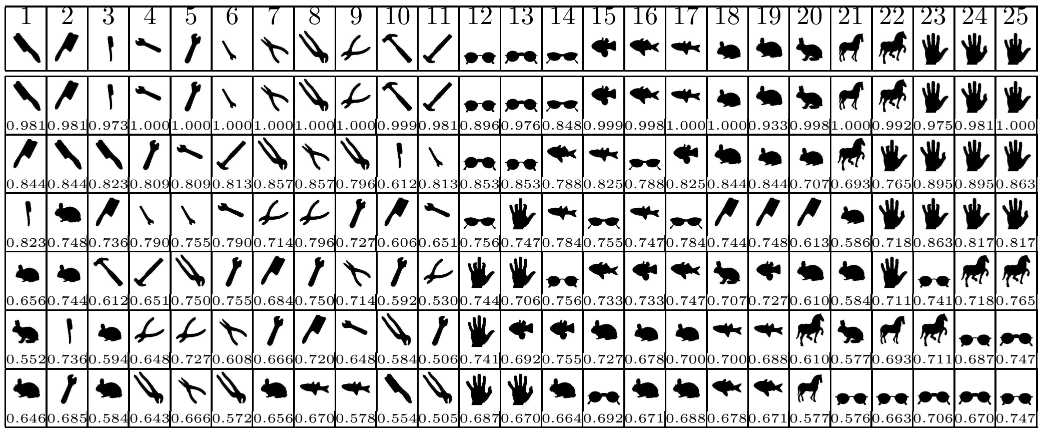


Figure 5.4: Top six matches for each shape for the weighted shock trees.

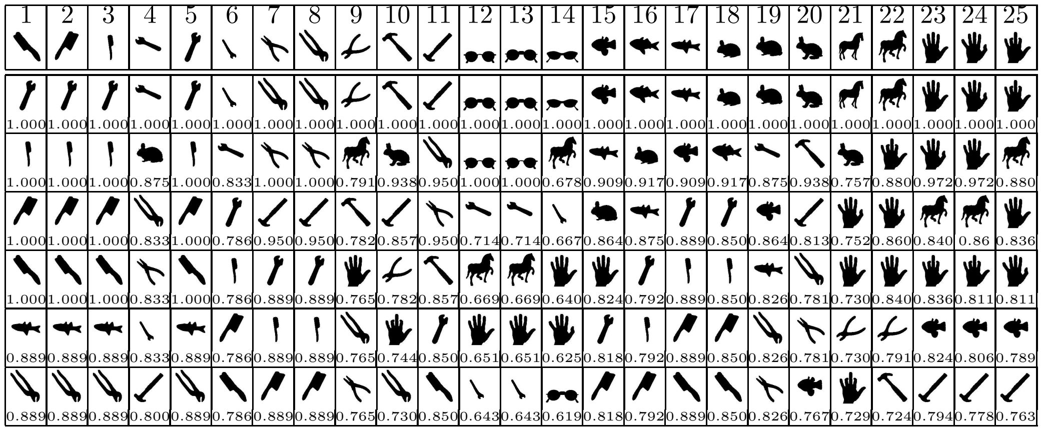


Figure 5.5: Top six matches for each shape for the unweighted shock trees.

more powerful representation than the simple one used for the experiments presented in the previous chapter. Here we follow Zucker, Siddiqi, and others, by labeling points on the skeleton using so-called shock labels [85]. This taxonomy of local differential structure, implies different classes associated with the behavior of the radius of the bitangent circle. The so-called shocks distinguish between the cases where the local bitangent 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 of a node in the tree is determined by the time of formation of the corresponding skeletal branch [81, 85]. The later the time of formation of the shock, the higher the corresponding node in the hierarchy.

The silhouettes used to generate the shock graphs at the basis of our experiments are shown in Figure 5.2. There are 25 different shapes. These include brushes, tools, spectacles, various animals and human hands. The figure is annotated with the pairwise similarity of the shapes. For the shapes indexed i and j, the similarity measure is defined as

$$S_{i,j} = 1 - \frac{1}{2} d_{i,j}, \tag{5.15}$$

where $d_{i,j}$ is the edit-distance between shapes i and j. In order to make the comparison independent of the size of the picture or of the shock-tree representation, the weight on each tree was previously normalized by dividing it by the sum of all the weights on every node of the tree. In this way the maximum possible edit-distance between two trees was 2.

For comparison purposes, Figure 5.3 reports the similarities between the unweighted shock trees. In this case the similarity between t_i and t_j is

$$S_{i,j} = \frac{1}{2} \frac{|V_i| + |V_j| - d_{i,j}}{2} \left(\frac{1}{|V_i|} + \frac{1}{|V_j|} \right), \tag{5.16}$$

where V_i is the node set of tree t_i and $d_{i,j}$ is the unattributed edit-distance between tree t_i and tree t_j.

In Figures 5.4 and 5.5 we show the six best matched shapes for each object from the database. The top row of the figures shows the shapes considered. The remaining rows, from top to bottom, show the six best matched shapes ordered according to similarity. Hence, the further down we go in each column, the poorer the match to the shape in the top position. Figure 5.4 shows the matches obtained when we associate the shape measure to the shock trees. In each case the first matched shape is the object under study. From the third row down errors begin to emerge. For instance, a monkey wrench (object 6) matches to a hammer (object 11), and a horse (object 22) matches to a hand (object 25). Although there are 6 such errors in the third row (objects 6, 10, 11, 14, 16, 22), several of these are associated with small differences in the similarity values. This is the case with object 6, where a monkey wrench is matched to a hammer. In both objects the dominant feature is the long handle. Additionally, for four of the objects the correct matches appear in the fourth (object 6, 16), fifth (object 14), or sixth (object 22) position. It is only the two hammers that pose a real problem. This is due to the fact that the handle, the main feature on both objects, shows variation in its differential properties. Specifically, object 10 bulges on the grip, creating a type one shock that splits the handle, whereas the handle of object 11 generates a single shock segment. The problem could be solved by allowing the edit-distance calculation to merge segments, as do Sebastian, Kimia and Klein in [72, 74], but this is beyond the scope of the present study.

Figure 5.5 displays the top matches obtained using unattributed shock trees. Here again the top match is a perfect fit in each case. However, the performance degrades more quickly as we go down the columns of the table. In fact, the first error emerges in the second row of the figure.

To visualize the pairwise relationships between the different shapes, we have performed multi-dimensional scaling on the set of pairwise similarities. Multi-dimensional scaling is a well known statistical technique for visualizing data which exists in the form of pairwise similarities rather than ordinal

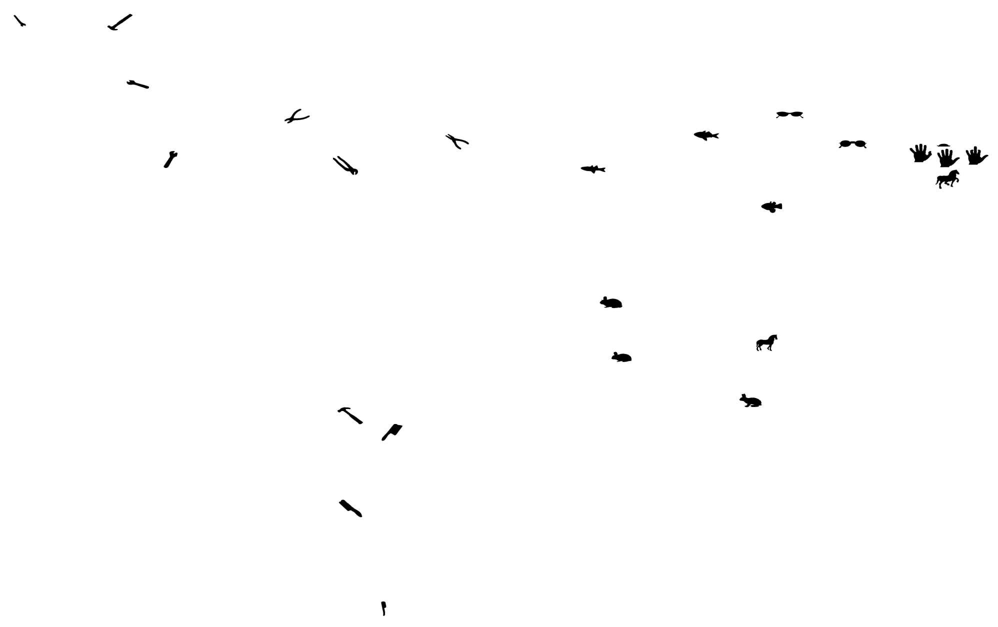


Figure 5.6: First and second principal components of the edit-distances of the shapes for the weighted shock trees.

values. Stated simply, the method involves embedding the objects associated with the pairwise distances in a low-dimensional space. This is done by performing principal components analysis on the matrix of pairwise similarities, and projecting the original objects into the resulting eigenspace. The objects are visualized by displaying their positions in the space spanned by the leading eigenvectors. The method has been widely exploited for data-analysis in the psychology literature. A comprehensive review can be found in the recent book by Cox and Cox [18]. Details of the procedure can be found in Appendix B.

The projections of the edit-distances onto the 2D space spanned by the two leading eigenvectors is shown in Figure 5.6 (where the skeleton is weighted with the border-length to shock length ratio) and Figure 5.7 (where it is not). When the skeleton is weighted with this ratio the MDS projection reveals the

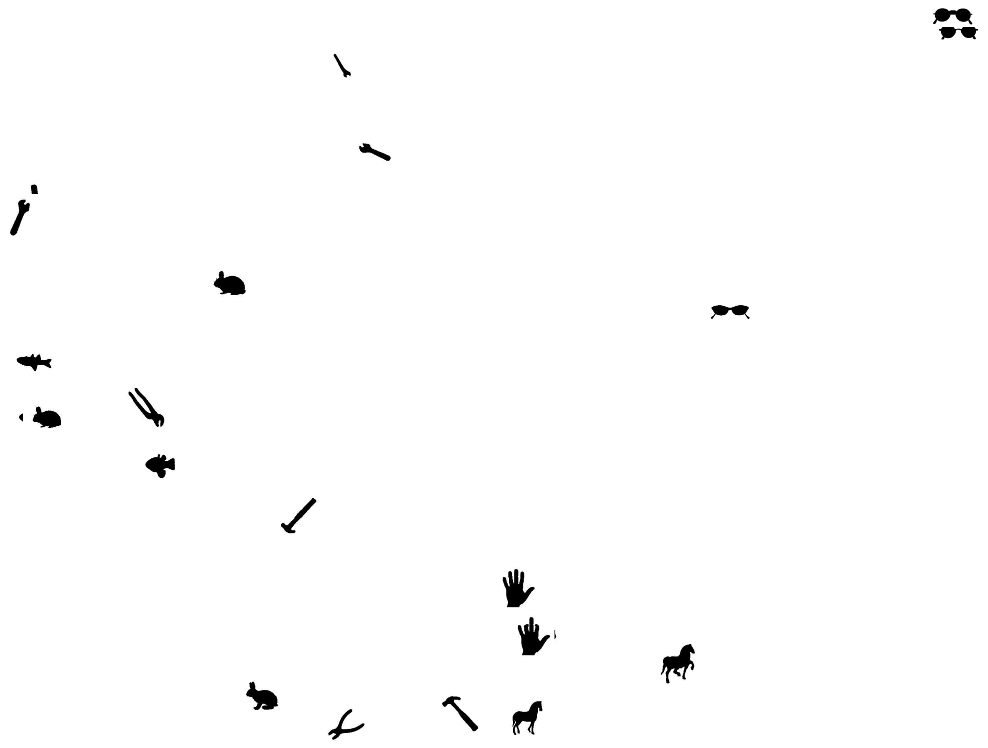
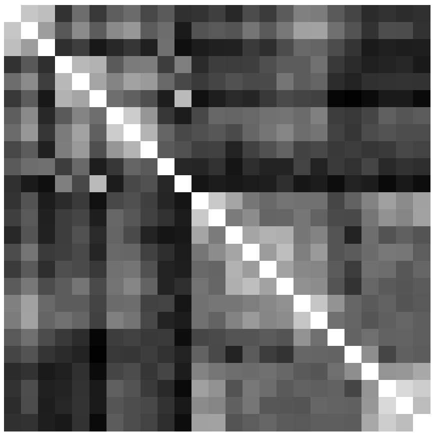


Figure 5.7: First and second principal components of the edit-distances of the shapes for the unweighted shock trees.

emergence of some class structure. However, the full shape-structure is not captured by the two leading eigenvectors. For instance, the hands, the fish, the tools and the brushes all appear close to each other. However, there is no clear delineation of the shape-classes. When the skeleton is not weighted using the above-mentioned measure the grouping of the shapes is even poorer, with only the spectacles forming a well separated group.

Encouraged by these results, we have performed a detailed pairwise clustering of the pattern of similarities. Here we use the method recently described by Robles-Kelly and Hancock [66]. Details of the clustering algorithm can be found in Appendix A. The initial and final matrices of pairwise distance are shown in Figure 5.8 for the measure-weighted skeleton and in Figure 5.9 for the unweighted skeleton. In the case of the weighted skeleton the initial pairwise similarity matrix shows a strong separation of the shape-groups, which is further re-enforced by the iterative clustering method. On the basis of the block structure of the final matrix of pairwise distances, we identify eight clusters. Figure 5.10a presents the clusters in order of extraction.

In other words, the hands, tools, spectacles and animals form clusters. However, there are shapes which leak between these clusters. The problems encountered above are due to the fact that certain shapes straddle the true shape-classes and cause cluster-merging. Figure 5.10b shows the result of


(a)

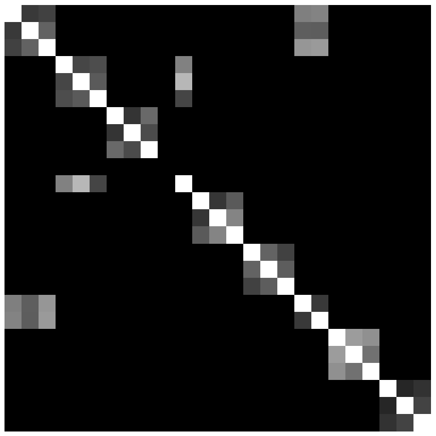
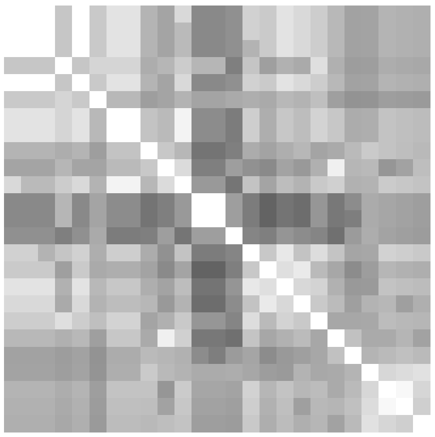
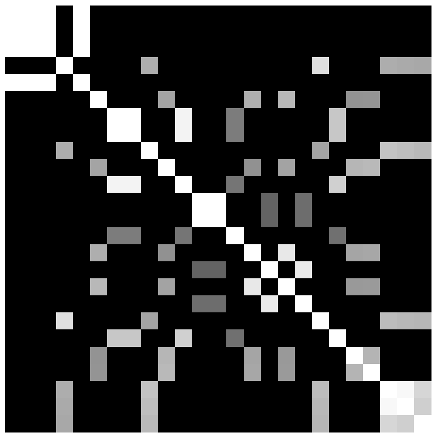

(b)

Figure 5.8: (a) Initial similarity matrix for the weighted tree edit-distances; (b) Final similarity matrix for the weighted tree edit-distances.



(a)



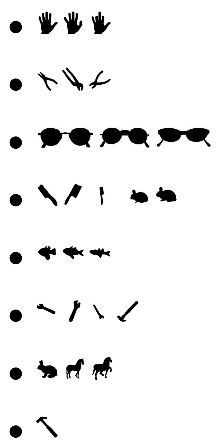
(b)

Figure 5.9: (a) Initial similarity matrix for the unweighted tree edit-distances; (b) Final similarity matrix for the unweighted tree edit-distances.

applying the clustering algorithm to a pruned set of 16 shapes.

This is a much better set of clusters, which reflects the true shape-classes in the data. We have repeated these clustering experiments with the unweighted skeletons. Here the initial pairwise similarity matrix contains less structure than in the weighted case, and iteration of the clustering algorithm results in a noisier set of final cluster membership indicators (Figure 5.3). In particular, the clusters extracted from unweighted shock trees do not appear to correlate well with the shape classes in the database (Figure 5.11a).

Clearly there is considerable merging and leakage between clusters. As illustrated in Figure 5.11b, the classification does not improve when the algorithm is applied to the reduced database.


a) Full database.

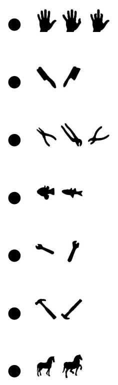
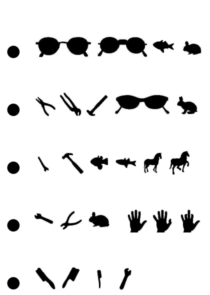

b) Partial database.

Figure 5.10: Clusters extracted from weighted edit-distance.


a) Full database.

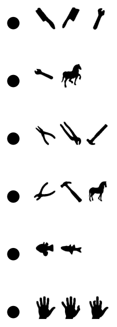

b) Partial database.

Figure 5.11: Clusters extracted from un-weighted edit-distance.

5.5.2 Quantitative Analysis

We now turn our attention to the properties of the weighted variant of our edit-distance approach when applied to a larger database of 150 shock trees. The database consists of 150 shapes divided into 10 shape classes containing 15 shapes each.

In Figure 5.12 we show the results of the application of multi-dimensional scaling to the edit-distances between the trees. Each label in the plot corresponds to a particular shape class. Label 1 identifies cars, label 2 dogs, 3 ducks, 4 fishes, 5 hands, 6 horses, 7 leaves, 8 men, 9 pliers, and, finally, label 10 is associated with screwdrivers. The plot clearly shows the difficulty of this clustering problem. The shape-groups are not well separated. Rather, there is a good deal of overlap between them. Furthermore, there are a considerable number of outliers.

To asses the ability of the clustering algorithm to separate the shape-classes, we performed experiments on an increasing number of shapes. We commenced with the 30 shapes from two shape-classes, and then increased the number of shape-classes under consideration up to a set of 120 shapes, i.e. 8 classes. Each experiment was performed four times with different choices of shape-groups in order to provide some error analysis. Figure 5.13 plots the proportion of shapes correctly classified as the number of shapes is increased. We can clearly see that the performance rapidly decreases as the number of shape-classes increases. Further improvements on these results will be presented in Chapter 7.

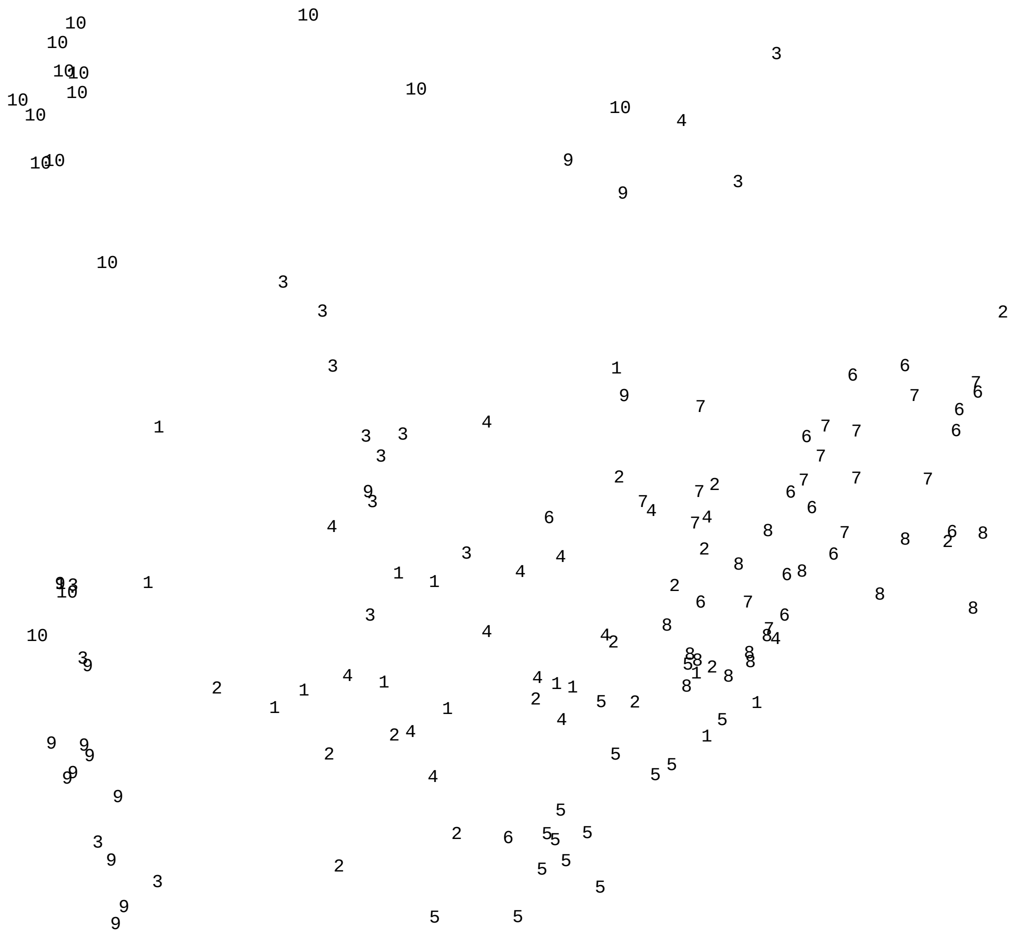


Figure 5.12: 2D multi-dimensional scaling of the pairwise distances of the shock graphs. The numbers correspond to the shape-classes.

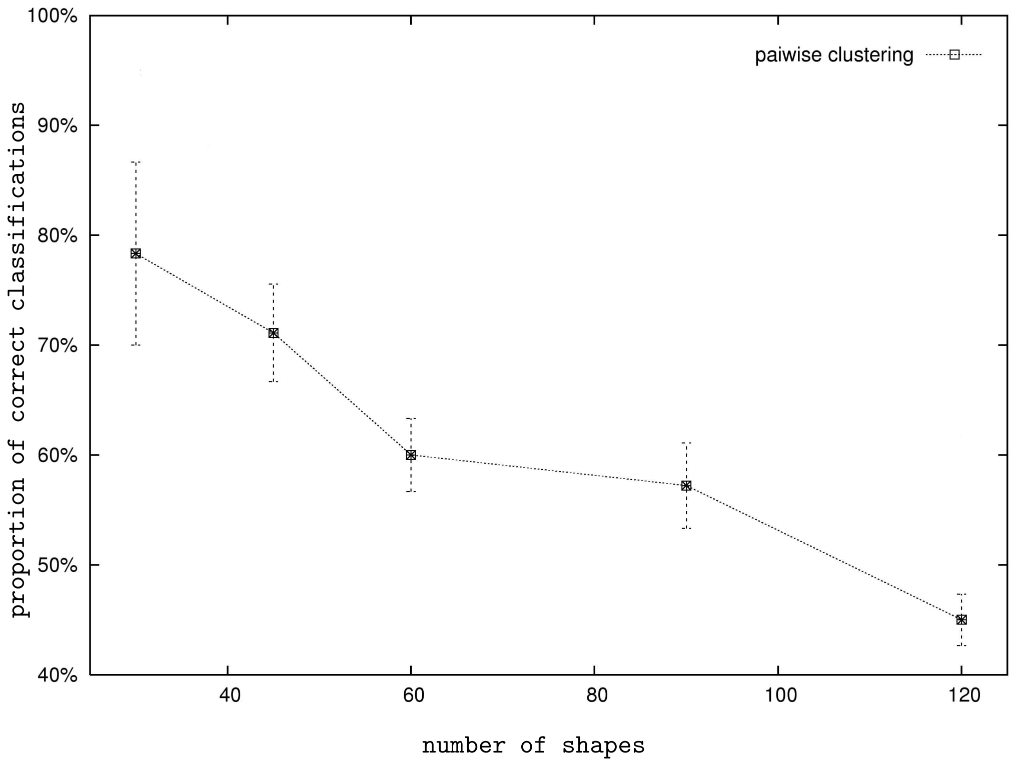


Figure 5.13: Proportion of correct classifications obtained with pairwise clustering of the edit-distances.

5.5.3 Sensitivity Study

To augment these real world experiments, we have performed a sensitivity analysis. The aim here is to characterize the behavior of our edit-distance algorithm when confronted with measurement errors resulting from noise, or jitter, on the weights and with structural errors resulting from node removal.

To test how well the method copes with structural modification we use it to match a randomly generated tree with modified versions of itself. To create these modified versions we removed an increasing fraction of nodes. Since we remove nodes only from one tree, the edited tree will have an exact match against the unedited version. Hence, we know the optimum value of the weight that should be attained by the maximum edited isomorphism. This is equal to the total weight of the edited tree.

By adding measurement errors or jitter to the weights, we test how well the method copes with a modification in the weight distribution. The measurement errors are distributed normally, with zero mean and controlled variance. Here we match the tree with noisy or jittered weights against its noise-free version. In this case we have no easy way of determining the optimal weight of the isomorphism, but we do expect a smooth drop in total weight with increasing noise variance.

We performed the experiments on trees with 10, 15, 20, 25, and 30 nodes. For each experimental run we used 11 randomly generated trees. The procedure for generating the random trees was as follows: we commenced with an empty tree (i.e. one with no nodes) and we iteratively added the required number of nodes. At each iteration nodes were added as children of one of the existing nodes. The parents were randomly selected with uniform probability from among the existing nodes. The weight of the newly added nodes was selected at random from an exponential distribution with mean 1. This procedure tends to generate trees in which the branch ratio is highest closest to the root. This is quite realistic in real-world situations, since shock trees tend to have this property.

The fraction of nodes removed was varied from 0% to 60%. In Figure 5.14 top left we show the ratio of the computed weighted edit-distance to the optimal value of the maximum isomorphism. Interestingly, for certain trees the relaxation algorithm failed to converge within the allotted number of iterations. Furthermore, the algorithm also failed to converge on the noise corrupted variants of these trees. In other cases, the algorithm exhibited particularly rapid convergence. Again, the variants of these trees also showed

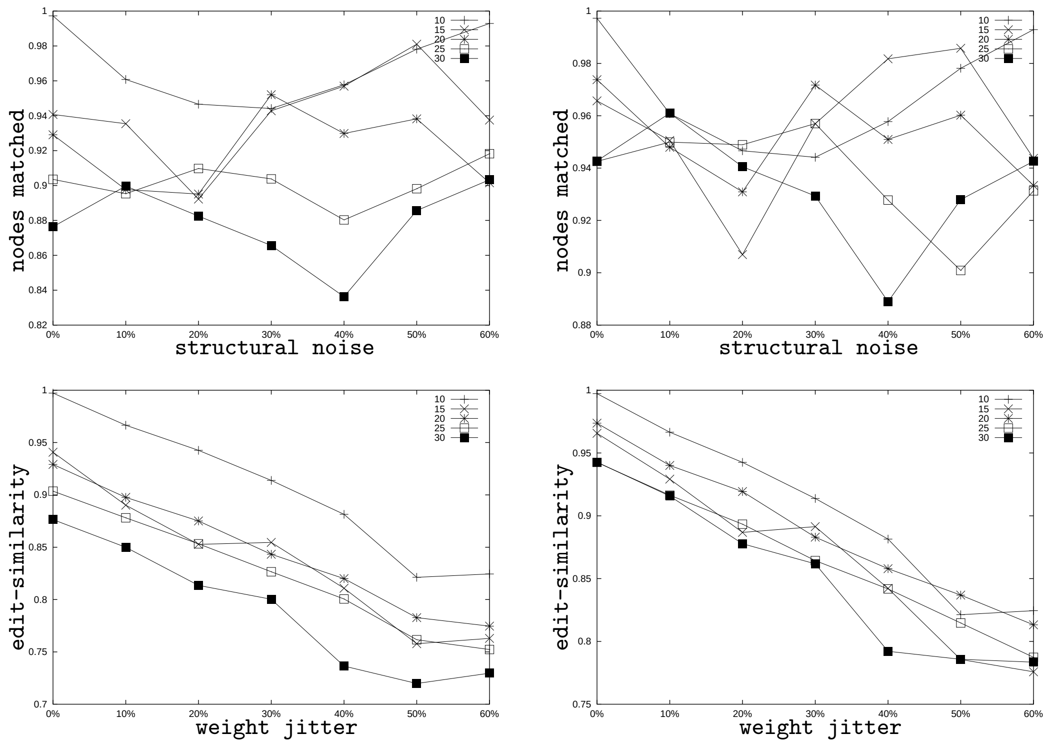


Figure 5.14: Sensitivity analysis: top-left node removal, top-right node removal without outliers, bottom-left weight jitter, bottom-right weight jitter without outliers.

rapid algorithm convergence. When the method fails to converge in an allocated number of iterations, we can still give a lower bound to the weight. However, this bound is substantially lower than the average value obtained when the algorithm does converge. The top right-hand graph of Figure 5.14 shows the proportion of matched nodes when we eliminate these convergence failures. The main conclusions that can be drawn from these two plots are as follows. First, the effect of increasing structural error is a systematic underestimation of the weighted edit-distance. Second, the different curves exhibit a minimum within the plot-range. The reason for this is that the matching problem becomes trivial as the trees are decimated to extinction.

The bottom row of Figure 5.14 shows the results obtained when measurement errors or jitter was added to the weights. Noise corrupted weights were obtained by adding random Gaussian noise with standard deviation ranging from 0 to 0.6. The bottom left-hand graph shows the result of this test. It is clear that the matched weight decreases almost linearly with the noise standard deviation. In these experiments, we encountered similar problems with algorithm convergence failure. Furthermore, the problematic trees were identical. This further supports the observation that the performance of the algorithm strongly depends on the randomized realization of the tree. The bottom right-hand plot shows the results of the jitter test with the convergence failures removed. Here we see a smaller variation in performance as the number of nodes increases.

5.6 Conclusions

In this chapter we have presented an algorithm for computing tree edit-distance. Adopting an optimization approach to tree matching, we show that any tree obtained with a sequence of node-removal or node-relabel operations 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 to 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 applied this algorithm to the problem of matching shock trees, demonstrating that the algorithm is able to match similar shapes together. We went on to perform pairwise clustering on the set of distances obtained. The combination of edit-distance and pairwise clustering, while capable of extracting the shape-classes present in the database, suffers from the high level of noise present. In fact, the approach is capable of extracting the shape-classes present in the database only when presented with a limited number of shapes. Indeed, too great a number of shapes saturates the shock-tree space, and the performance of the approach deteriorates.

To overcome the problem of underestimating edit-distance, in Chapter 6 we explore the use of the union structure to capture the modes of variation of graphs belonging to a particular class.

By measuring the edit-distance to the union structure, we make better estimates of the dissimilarity of trees. This work is taken a step further in Chapter 7, where we treat the tree union as a generative model for the distribution of tree structure and present an algorithm for learning a mixture of tree unions by minimizing description length.

Section III

Structural Archetype

Chapter 6

Structural Embedding Through Tree Union

In order to improve on the results presented in the previous chapter, we address the problem of how to organize shock trees into a shape-space where a) similar shapes are close to one another and b) the space is traversed in a relatively uniform manner as the shapes are gradually modified. In other words, the aim is to embed the trees in a vector-space where the dimensions correspond to principal modes of structural variation. In many ways this is a prerequisite to learning a representation for a set of graphs. This chapter deals with the problem of defining this structural representation, while the next one will deal with the problem of learning this representation from a set of samples.

There are a number of ways in which graph embedding can be achieved. The first, as we have seen in the previous chapter, is to compute the edit-distance between shock trees and to use multi-dimensional scaling to embed the individual graphs in a low-dimensional space. However, this approach does not necessarily result in a shape-space where the dimensions reflect the modes of structural variation of the shock trees. Furthermore, pair-wise distance algorithms consistently underestimate the distance between shapes belonging to different clusters. When two shapes are similar, the node-correspondences can be estimated reliably, but as shapes move farther apart in shape-space the estimation becomes less reliable. This is due to the fact that correspondences are chosen to minimize the distance between trees and, as the shock trees move farther apart, the advantage the "correct" correspondence has over alternative ones diminishes, until, eventually, a match which yields a lower distance is selected.

A second approach is to extract feature vectors from the graphs and to use these as a shape-space representation. A shape-space can be constructed from such vectors by performing modal analysis on their covariance matrix. However, this approach becomes problematic as soon as graphs of different sizes are used.

Here we take a different approach to the problem. We aim to embed trees in a pattern space by mapping them to vectors of fixed length. We do this as follows. We commence from a set of trees, and from this we construct a super-tree from which each tree may be obtained by the edit operations of node and edge removal. Hence, each tree is an edited subtree of the super-tree. The super-tree is constructed so that it minimizes the total edit-distance to the set of shock trees. To embed the individual shock trees in a vector-space we allow each node of the super-tree to represent a dimension of the space. Each shock tree is represented in this space by a vector which has non-zero components only in the directions corresponding to its constituent nodes. The non-zero components of the vectors are the weights of the nodes. In this space, the edit-distance between trees is the L_1 norm between their embedded vectors.

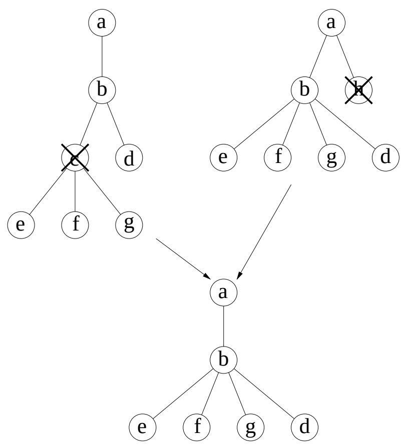


Figure 6.1: Edit-intersection of two trees.

In the previous chapter we have seen how the edit-distance between two trees is completely determined by the set of nodes that do not get removed by edit operations and, therefore, get matched. That is, in a sense, we are taking the intersection of the sets of nodes of the two structures. With this approach we match the trees by extracting a structure that can be obtained from our original trees by removing some nodes. We have seen how the edit-distance between two trees is related to this intersection structure (see Figure 6.1). We would like to extend the concept of edit-distance presented in the previous chapter to more than two trees so that we can compare a shape tree to a whole set T of trees. Moreover, we would like to determine how a new sample relates to a previous distribution of tree structures. Formally, we would like to find the match that minimizes the sum of the edit-distances between the new tree t^* and each tree $t \in T$, with the added constraint that if node a in the new tree t^* is matched to node b in a tree $t_1 \in T$ and to node c in another tree $t_2 \in T$, then b must be matched to c, i.e.

$$(a,b) \in M \wedge (a,c) \in M \Rightarrow (b,c) \in M, \tag{6.1}$$

where M is the "matches to" relation on nodes. One way of finding this match would be to find the maximum substructure that can be obtained from any tree in a set by removing appropriate nodes. Unfortunately, by discarding unmatched nodes, we are losing too much information. The main problem is that the set of common nodes becomes marginal and we lose information about how the nodes distribute in the various structures. To use Bunke's [14] analogy, the maximum common substructure gives us information about the mean of the set of trees, but it completely discards any information about how sample trees distribute around this mean. To overcome this limitation we can calculate a union of the nodes. This is a structure from which we can obtain any tree in our set by removing appropriate nodes, as opposed to the common substructure, which provides the intersection of the nodes.

Any such structure has the added advantage of implicitly creating an embedding space for our trees. Crucially, it guarantees that any node in any tree matches to a node in this structure. Assigning to each node a coordinate in a vector space X, we can associate to tree t the vector $x \in X$ so that $x_i = w_i$, where w_i is the weight of the node of t associated with coordinate i.

6.1 Embedding Space

Formally, we place the nodes of the union structure G in any arbitrary order. To each sample tree t we associate a pattern-vector $\vec{x}^t = (x_1, \cdots, x_n)^T \in \mathbb{R}^n$, where n is the number of nodes in the union G. The component x_i^t of vector $\vec{x}^t$ is:

$$x_i^t = \begin{cases} w_i^t & \text{if the tree has a node mapped to the } i\text{-th} \\ & \text{node of the sample} \\ 0 & \text{otherwise.} \end{cases} \tag{6.2}$$

In other words, we associate a pattern-vector $\vec{x}^t$ with the sample tree whose components are equal to the weight of the corresponding node in the union tree, if the node is present, and are zero otherwise.

6.2 Union of Two Trees

As was the case with edit-distance, the edit-union of two trees is completely determined by the set of matched nodes. Start with the two trees and iteratively merge nodes that are matched. Upon completion of this procedure, the result is a directed acyclic graph with multiple paths connecting various nodes (see Figure 6.2). This structure, thus, has more links than necessary and in order to obtain the original trees using node removal operations alone, the superfluous edges need to be removed. If in Figure 6.2 we eliminate the edges connecting node b to nodes e,f and g, we obtain a tree. Starting from this tree, we can obtain either one of the original trees by node removal operations alone. Furthermore, the order relation defined by this tree and the one defined by the unaltered structure are identical.

We would hope that such a structure would always be a tree, so that we can use the matching technique already described to compare a tree to a group of trees. Unfortunately, it is not always possible to find a tree such that we can edit it to obtain the original trees. An example is provided in Figure 6.3. In this figure α and β are subtrees. Because of the constraints posed by matching subtrees α and subtrees β, nodes b and b' cannot be matched and neither one can be a child of the other. The only alternative is to keep the two paths separate. In this way we can obtain the first tree by removing the node b' and the second tree by removing node b. Actually, removing the nodes is not enough: shortcutting edges need to be removed. However, once

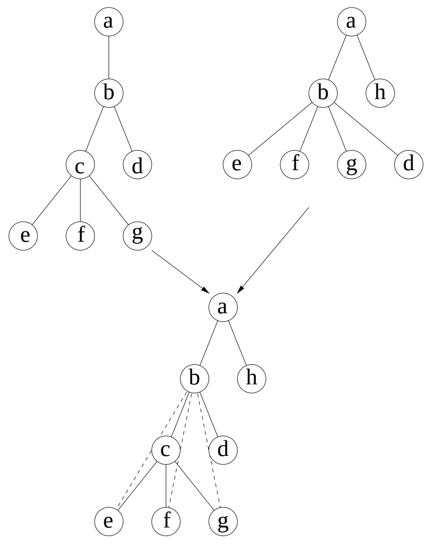


Figure 6.2: Edit-union of two trees.

again, the transitive closure of the union minus node b' is identical to the closure of the first tree.

6.3 Matching a Tree to a Union

As shown above, the union of two trees is, in general, a directed acyclic graph. Our approach can only match trees, and would fail on structures with multiple paths from one node a to a descendent node b, since it would count any match in the subtree rooted at b twice. Hence, we cannot directly use our approach to compare a tree to a tree-set or a distribution of trees.

Fortunately, we do not need to perform a match between two generic directed acyclic graphs. The reason for this is that in an edit-union we have multiple paths between node a and node b, but each tree can have only one; hence multiple paths are mutually exclusive. If we constrain our search to matching nodes in one path only, we can match any tree to the union, being still guaranteed not to count the same subtree multiple times. Interestingly,

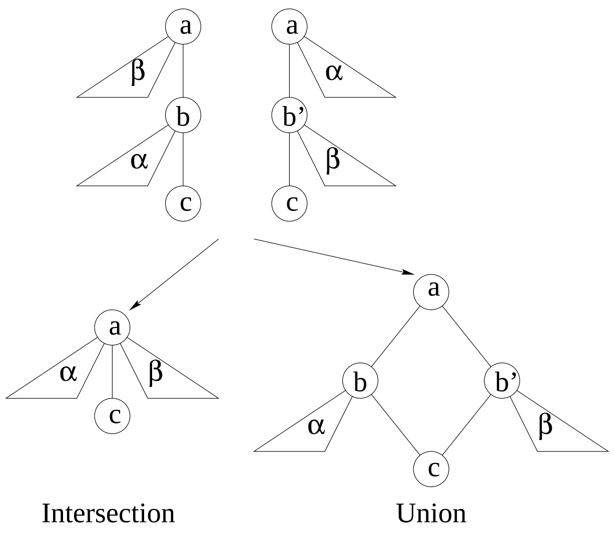


Figure 6.3: Edit-union is not always a tree.

this constraint can be merged with the *obtainability* constraint. We say that a match is *obtainable* if for each node v there cannot be two children a and b and a node c such that there is one path, possibly of length 0, from a to c and another from b to c. This constraint is reduced to the previously defined *obtainability* for trees when $c = b$, but it also makes it impossible for a and b to belong to two separate paths joining at c. Hence, from a node where multiple paths fork, we can extract matches from one path only.

We wish to find the match consistent with the *obtainability* constraint that minimizes the sum of the edit-distances between the new tree and each tree in the set. For this purpose we can maximize the sum of the utilities.

$$\mathcal{U}(M) = \sum_{t' \in T} \sum_{(u,v) \in M, u \in \mathcal{N}^{t'}} \left(r_u^t + r_v^{t'} - m_{uv}^{t'} \right). \tag{6.3}$$

Here $M \subset \mathcal{N}^t \times \mathcal{N}^{\mathcal{T}}$ is the set of matches between the nodes $\mathcal{N}^t$ of the tree t and the nodes $\mathcal{N}^{\mathcal{T}}$, where $\mathcal{T}$ is the union structure of the set of trees T. Furthermore, r_u^t is the cost of removing node u from tree t, $r_v^{t'}$ is the cost of removing node v from t', and $m_{u,v}^{t'}$ is the cost of matching node u of t

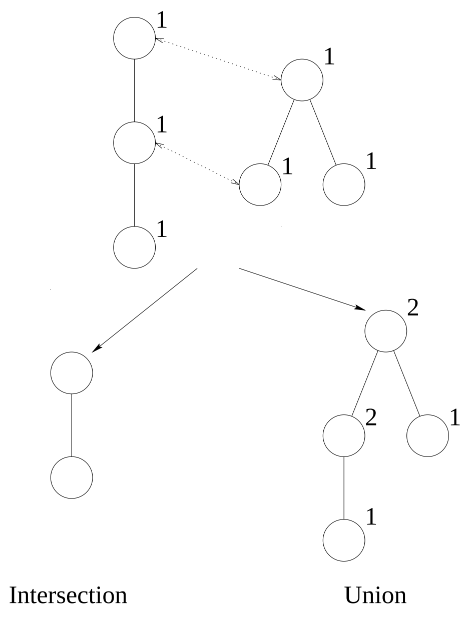


Figure 6.4: The weight of a node in the union account for every node mapped to that node.

to node v of t'. When the edit-distance is uniform (all weights are unity), the utility produced by a single match is equal to the number of trees that have an instance of that node (see Figure 6.4). On the other hand, when we use the weights described in Chapter 5 and the relative edit-costs, we have $r_u^t + r_v^{t'} - m_{uv}^{t'} = w_u + w_v^{t'} - |w_u - w_v^{t'}| = \min(w_u, w_v^{t'})$, where w_u is the weight associated to node u and $w_v^{t'}$ is the weight associated to node v in t'. By solving the modified weighted clique problems, we obtain the correspondence between the nodes in the new tree and the nodes in each tree in the set. Moreover, the edit-distance obtained is the sum of the distances from the new tree to each tree in the set T.

To be able to calculate this quantity, we keep the weights of the matched nodes for each node in the union structure. A way to do this is to assign to each node v in the union a vector $\vec{x}^v = (x_1, \ldots, x_m)^T \in \mathbb{R}^n$, where m is the number of trees in the set T. The component x_i^v of vector $\vec{x}^v$ is:

$$x_i^v = \begin{cases} w_v^{t_i} & \text{if node } v \text{ is present in tree } t_i \\ 0 & \text{otherwise.} \end{cases} \tag{6.4}$$

This representation also makes it easy to obtain the coordinate of each tree in the embedding space induced by the union: if $X = (\vec{x}^1|\cdots|\vec{x}^m)$ is the a matrix whose columns are equal to he vectors $\vec{x}^v$, the columns of X^T will correspond to the embedding vectors of the trees in T. This embedding is defined modulo reordering of the trees and of the nodes.

It is worth noting that this approach can be extended to match two union structures, as long as at most one has multiple paths to a node. To do this we iterate through each pair of weights drawn from the two sets. In other words, we define the utility as:

$$\mathcal{U}(M) = \sum_{t \in T_1, t' \in T_2} \sum_{(i,j) \in M} \left(r_u^t + r_v^{t'} - m_{uv}^{tt'} \right), \tag{6.5}$$

where $M \subset \mathcal{N}^{\mathcal{T}_1} \times \mathcal{N}^{\mathcal{T}_2}$ is the set of matches between the nodes of the union structures $\mathcal{T}_1$ and $\mathcal{T}_2$, r_u^t is the cost of removing node v from tree t, and $m_{uv}^{tt'}$ is the cost of matching node u of tree t in $\mathcal{T}_1$ to node v of t' in $\mathcal{T}_2$. The requirement that no more than one union has multiple paths to a node is necessary to avoid double counting.

6.4 Joining Multiple Trees

In the previous paragraph we have seen that it is possible to construct the edit union of a set of trees, and we have considered how a tree can be compared to this superstructure. We now want to show how to construct such a structure. Finding the super-structure that minimizes the total distance between the trees in a set is computationally infeasible, but we propose a suboptimal iterative approach which at each iteration extends the union by adding a new tree to it. This is done by matching the tree to the union and then using the matched nodes to construct the union of the two structures. That is, we select a new tree t^* and we match it against the current union $\mathcal{T}^{(t)}$, to obtain the updated union $\mathcal{T}^{(t+1)}$. We proceed in this way until we have joined every tree.

In order to increase the accuracy of the approximation, we wish to merge the trees with the smaller distance first. The reason for this is that the smaller the distance between two trees, the higher is our confidence regarding the extracted correspondences. We start with the full set of trees, merge them and replace them with the union. We reiterate this procedure until we obtain a single structure. At each iteration we select two trees t_1 and t_2 such that the distance $d(t_1, t_2)$ is minimal, merge the two tree and reinsert the union structure $\mathcal{T}_{1,2}$ in the set of trees to be merged. Unfortunately, since we have no guarantee that the edit-union is a tree, we might attempt to merge two graphs with multiple paths to a node, and this is something that our matching algorithm cannot cope with. For this reason, we discard any union that is not a tree and attempt to merge the next-best match. When no trees can be merged without duplicating paths, we randomly select one union and merge the remaining structures to it in random order. In this way we are guaranteed to merge at most one multi-path structure at each step.

6.5 Experimental Results

We evaluate the application of the new graph embedding approach on the problem of shock tree matching. First, we compare the embedding obtained using the union approach to 2D multi-dimensional scaling of the pairwise edit-distances. Then we compare the clusters obtained using the L_1 norm defined on the union with those obtained using pairwise edit-distances. The clusters are extracted using the algorithm presented in Appendix A.

By forcing the matches to be consistent across shapes, we enable the embedding to better capture the structural information present in the shapes, yielding better spatial distribution than that obtained with multi-dimensional scaling.

6.5.1 Embedding

We perform principal components analysis on the embedding space defined by the union . To do this we compute the covariance matrix for the pattern vectors, and project the pattern vectors onto the space spanned by the leading eigenvectors of the covariance matrix.

We run three experiments with 4, 5, and 9 shapes each. In each experiment the shapes belong to two or more distinct visual clusters. Let $T = \{t_1, \cdots, t_n\}$ be a set of n trees and $\mathcal{T}$ their union. We extract the embedding vectors $\vec{x}^{t_1}, \cdots, \vec{x}^{t_n}$ in the way described in Section 6.1. In order to avoid scaling effects due to the difference in the number of nodes, we normalize the embedding vectors so that they have L_1 norm equal to 1. That is, we calculate the new vectors

$$\vec{y}^i = \frac{\vec{x}^i}{\sum_j |x^i_j|}. \tag{6.6}$$

Then, we compute the mean pattern-vector

$$\vec{\hat{y}} = \tfrac{1}{n} \sum_{t \in T} \vec{y}^t \tag{6.7}$$

and covariance matrix

$$\Sigma = \frac{1}{n}\sum_{t\in T}(\vec{y}^t - \hat{\vec{y}})(\vec{y}^t - \hat{\vec{y}})^T. \tag{6.8}$$

Suppose that the eigenvectors (ordered by decreasing eigenvalue) are $\vec{e}_1, ...\vec{e}_n$. The leading l eigenvectors are used to form the columns of the matrix $E = (\vec{e}_1|\cdots|\vec{e}_l)$. We perform PCA on the sample trees by projecting the pattern vectors onto the leading eigenvectors of the covariance matrix. The projection of the pattern vector for the sample tree indexed t is $\vec{z}^t = E^T\vec{y}^t$. We confront the spatial distribution obtained with the 2 principal components of the union embedding with that obtained with 2D multi-dimensional scaling.

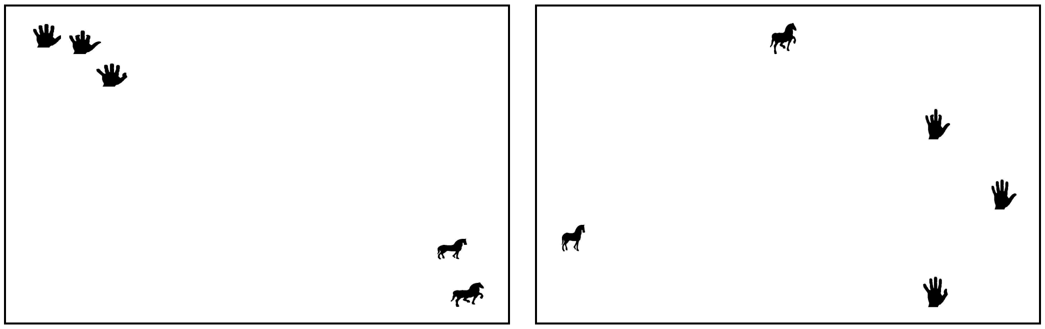


Figure 6.5: Left: embedding through union. Right: multi-dimensional scaling of pairwise distances.

Figure 6.5 shows a clear example where the embedding obtained through edit-union is better than that obtained through multi-dimensional scaling of the pairwise distances. In this case the pairwise distance algorithm consistently underestimates the distance between shapes belonging to different clusters. This is a general problem of pairwise matching. The method works very well when the shapes are close and the extracted correspondence is reliable, but as the shapes move further apart the advantage the correct correspondence has over alternative ones diminishes, until, eventually, another

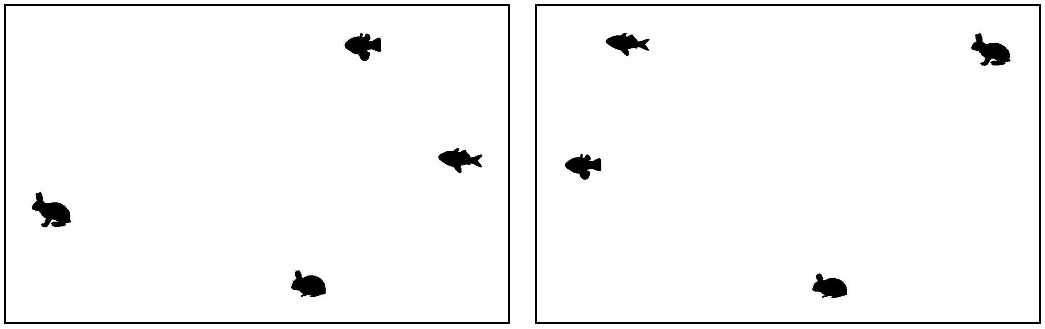


Figure 6.6: Left: embedding through union. Right: multi-dimensional scaling of pairwise distances.

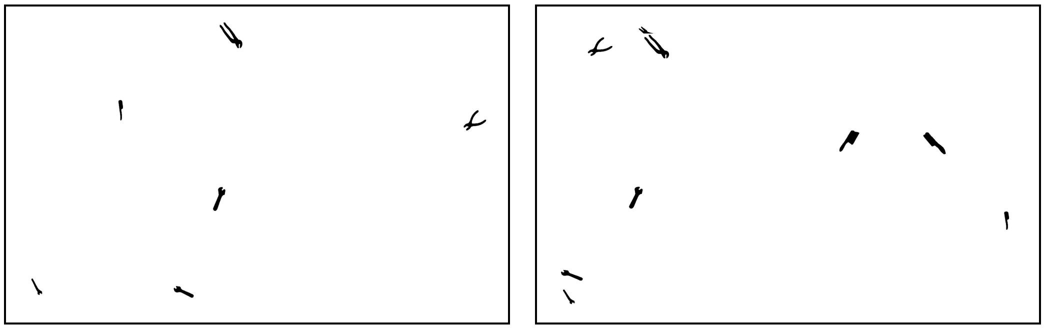


Figure 6.7: Left: embedding through union. Right: multi-dimensional scaling of pairwise distances.

match is selected, which reports a lower distance. The result of this is a consistent underestimation of the distance as the shapes move farther apart in shape-space. Figures 6.6 and 6.7 show examples where the distance in shape-space is not large enough to allow us to observe the described behavior, yet the embedding obtained through union fares well against the embedding obtained through multi-dimensional scaling of the pairwise edit-distances. In particular, Figure 6.7 shows a better ordering of the shapes, with brushes being so tightly packed that they overlap. It is interesting to note how the union embedding puts the monkey wrench (at the top) somewhere in between the pliers and the wrenches. The algorithm is able to consistently match the head to the heads of the wrenches, and the handles to the handles of the pliers.

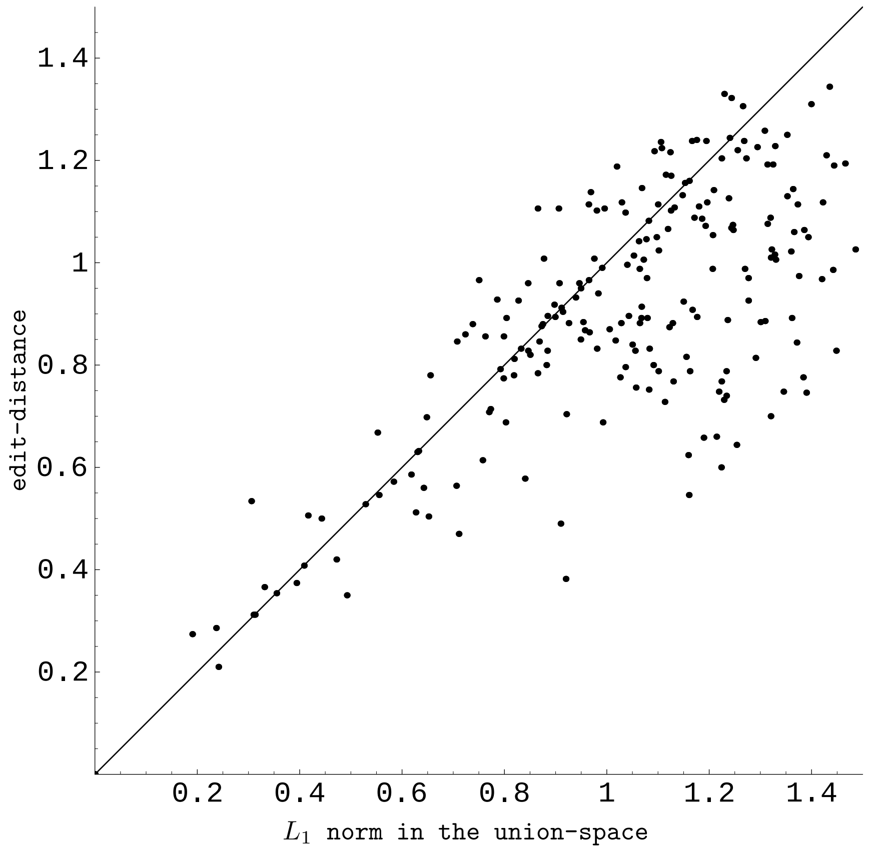


Figure 6.8: Edit-union vs pairwise distances.

Figure 6.8 plots the distances obtained through edit-union of weighted shock trees (x axis) versus the corresponding pairwise edit-distances (y axis). The plot clearly highlights that the pairwise distance approach tends to underestimate the distances between shapes.

6.5.2 Clustering

Figure 6.9 shows the results of applying the pairwise clustering presented in Appendix A to the distances obtained with edit-distance and to the dis-

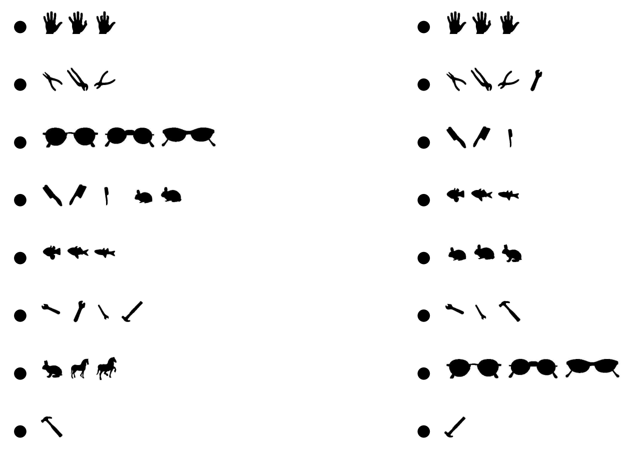

a) Weighted edit-distance. b) Union of attributed trees.

Figure 6.9: Clusters extracted from edit-distances versus those obtained from the L_1 norm defined on the union.

tances obtained through edit-union. When applied to the reduced database of 25 shapes, edit-union clearly outperforms the pairwise distance approach. Unfortunately the union approach does not scale, and fails as the database becomes larger and the number of shape classes present increases.

6.5.3 Synthetic Data

To augment these real world experiments, we have computed the union using synthetic data. The aim of the experiments is to characterize the ability of the approach to generate an embedding space for tree structures. To meet this goal we have randomly generated some prototype trees and, from each tree, we have generated five or ten structurally perturbed copies. The procedure for generating the random trees was the same as described in Section 5.5.3. We commenced with an empty tree (i.e. one with no nodes) and we iteratively added the required number of nodes. At each iteration nodes were added as children of one of the existing nodes. The parent was randomly selected

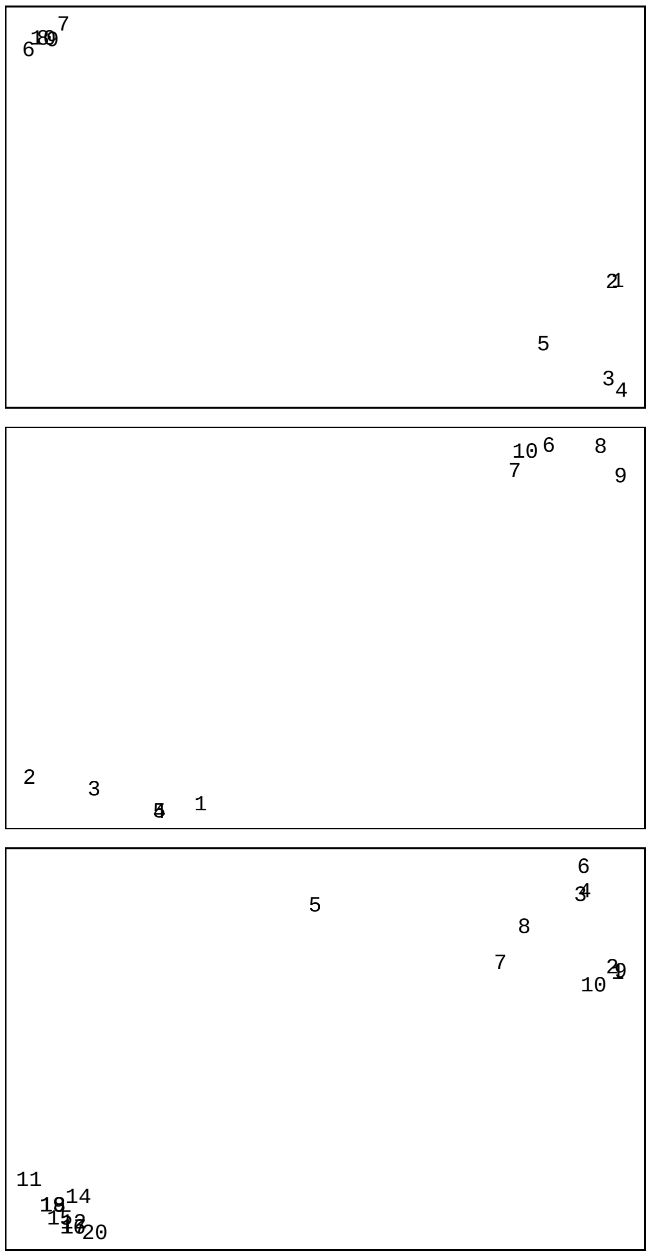


Figure 6.10: Synthetic clusters.

with uniform probability from among the existing nodes. The weight of the newly added nodes was selected at random from an exponential distribution with mean 1. To perturb the trees we simply added nodes using the same approach.

In our experiments the size of the prototype trees varied from 5 to 20 nodes. As we can see from Figure 6.10, the algorithm was able to clearly separate the clusters of trees generated by the same prototype. Figure 6.10 shows three experiments with synthetic data. The top and middle images are produced by embedding 5 structurally perturbed trees per prototype. Hence, trees 1 to 5 are perturbed copies of the first prototype, 6 to 10 of the second. The bottom image shows the result of the experiment with 10 structurally perturbed trees per prototype. Hence, trees 1 to 10 belong to one cluster and trees 11 to 20 to the other. In each image the clusters are well separated.

6.6 Conclusions

In this chapter we investigated a technique to extend the tree edit-distance framework to allow for the simultaneous matching of multiple tree structures. With this approach we can impose a consistency of node correspondences between matches, avoiding the underestimation of the distance typical of pairwise edit-distance approaches. Furthermore, through this method we obtain a “natural” embedding space of tree structures that can be used to analyze how tree representations vary in our problem domain.

In a set of experiments we apply this algorithm to match shock graphs. The results of these experiments are very encouraging, as they show that the algorithm is able to group similar shapes together in the generated embedding space. Yet the approach fails when confronted with larger databases containing several shape classes. This is due to the fact that the union is capable of capturing the structural variation present in a class, but has problems with multiple classes. The next chapter of the thesis presents a development of this result, in which the union is used to describe various single classes. Hence, the matching problem is cast as one of learning the structural representations that best describe a set of trees. We present an information theoretic approach for learning these representations.

Chapter 7

Classification Using a Probabilistic Mixture of Tree Unions

The previous chapter presented the tree union. This is a structural archetype that can be used to represent the ensemble of sample trees present in a set. However, the construction of the archetype was done in a goal-directed manner, without direct link to how the tree samples are distributed in the set. The aim in this chapter is to develop an information theoretic framework for the unsupervised learning of generative models of tree-structures from sets of examples. We pose the problem as that of learning a mixture of union-trees. Central to this approach is the realization that each tree union constitutes an archetype that represents a class of trees as well as the probability distribution of trees within the class. We work under conditions in which the node correspondences required to perform merges are unknown and must be located by minimizing tree edit-distance. Associated with each node of the union structure is a probability. This way the tree archetypes provide us with generative models. There are hence three quantities that must be estimated in order to construct this generative model. The first of these is the set of correspondences between the nodes in the training examples and the nodes in the estimated union structure. The second is the union structure itself. The third is the set of node probabilities.

We cast the estimation of these three quantities in an information theoretic setting. The problem is that of learning a mixture of union-trees to represent the classes of trees present in the training data. We use as our information criterion the description length for the union structure [64]. The entities to be described are the union structures themselves and the probabilities associated with their nodes, conditioned to the set of correspondences with the training samples. An important contribution of this research is the proof that the description length is related to the edit-distance between the union structure and the training examples. Our approach essentially involves three computational ingredients. First, we locate correspondences in such a way as to minimize the edit distance. Secondly, we construct the union structure using a set of tree merge operations that minimize the description length. Thirdly, we make maximum likelihood estimates of the node probabilities. It is important to note that the union model underpinning our method assumes node independence on the training samples. This is why the straightforward union approach described in the previous chapter does not work well with multiple classes. Using a mixture of unions we condition this independence to the class. This conditional independence assumption, while often unrealistic, is at the basis of the naive Bayes model [44] which has proven to be robust and effective for a wide range of classification problems.

7.1 Generative Tree Model

Consider the data set $\mathcal{D}$ consisting of sample trees $\mathcal{D} = \{t_1, t_2, \ldots, t_n\}$. Our aim is to cluster these trees, i.e. to perform unsupervised learning of the class structure of the data. We pose this problem as that of learning a mixture of generative class archetypes. Each class archetype is constructed by merging sets of sample trees together to form a set of union-trees. This merge process requires node correspondence information, and we work under conditions in which this information is unknown and must be inferred as part of the learning process. The class archetypes are generative models since they capture in an explicit manner the structural variations for the sample trees belonging to a particular class in a probabilistic manner. In Section 6.3 we will focus in more detail on how the class archetypes are learned, i.e. how the tree merge operations are effected. Now we detail the probabilistic ingredients of our model.

To commence, suppose that the set of class archetypes constituting the mixture model is denoted by $\mathcal{H} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_k\}$. For the class c, the tree model $\mathcal{T}_c$ is a structural archetype derived from the tree-union obtained by merging the set of trees $\mathcal{D}_c$ constituting the class. Associated with the archetype is a probability distribution which captures the variations in tree structure within the class. Hence, the learning process involves estimating the union structure and the parameters of the associated probability distribution for the class model $\mathcal{T}_c$.

The estimation of the required class models is effected using a simple greedy optimization method. The quantity to be optimized is the descriptor length for the sample data set $\mathcal{D}$. The parameters to be optimized include the structural archetype of the model $\mathcal{T}$ as well as the node correspondences $\mathcal{C}$ between the samples in the set $\mathcal{D}$ and the archetype. Hence, the inter-sample node correspondences are not assumed to be known a priori. Since the correspondences are uncertain, we must solve two interdependent optimization problems. These are the optimization of the union structure given a set of correspondences, and the optimization of the correspondences given the tree structure. These dual optimization steps are approximated by greedily merging similar tree-models.

We characterize uncertainties in the structure obtained by tree merge operations by assigning probabilities to nodes. We present two node probability models. The first model is used when the sample trees are unweighted, i.e. when the nodes have no attributes associated with them. In this case the node probability is estimated using the sample frequency of the nodes. The second model is used when the samples have continuous attributes, or weights, associated with them. By adopting an information theoretic approach we demonstrate that the tree edit-distance, and hence the costs for the edit operations used to merge trees, are related to the entropies associated with the node probabilities.

7.1.1 Probabilistic Framework

More formally, the basis of the proposed structural learning approach is a generative tree model which allows us to assign a probability distribution to a sample of hierarchical trees. Each hierarchical tree t is defined by a set of nodes $\mathcal{N}^t$, a tree-order relation $\mathcal{O}^t \subset \mathcal{N}^t \times \mathcal{N}^t$ between the nodes, and, in the case of weighted trees, a weight set $W^t = \{w_i^t | i \in \mathcal{N}^t\}$ where w_i^t is the weight associated with node i of tree t.

Our aim is to construct a generative model for a class of trees $\mathcal{D}_c \subset \mathcal{D}$. The structural component of this model $\mathcal{T}_c$ consists of a set of nodes $\mathcal{N}_c$ and an associated tree order relation $\mathcal{O}_c \subset \mathcal{N}_c \times \mathcal{N}_c$. Additionally, there is a set

$\Theta_c = \{\theta_i^c, i \in \mathcal{N}_c\}$ of sampling probabilities θ_i^c for each node $i \in \mathcal{N}_c$. Hence the model is the triple $\mathcal{T}_c = (\mathcal{N}_c, \mathcal{O}_c, \Theta_c)$. A sample from this model is a hierarchical tree $t = (\mathcal{N}^t, \mathcal{O}^t)$ with node set $\mathcal{N}^t \subset \mathcal{N}_c$ and a node hierarchy $\mathcal{O}^t$ that is the restriction to $\mathcal{N}^t$ of $\mathcal{O}_c$. In other words, the sample tree is just a subtree of the class archetype, and it can be obtained using a simple set of edit operations that prune the archetype.

To develop our generative model, we make a number of simplifying assumptions. First, we drop the class index c to simplify notation. Second, we assume that the set of nodes for the union structure $\mathcal{T}$ spans each of the trees encountered in the set of sample trees $\mathcal{D}$, i.e. $\mathcal{N} = \bigcup_{t \in \mathcal{D}} \mathcal{N}^t$. Third, we assume that the sampling error acts only on nodes, while the hierarchical relations are always sampled correctly. That is, if nodes i and j satisfy the relation $i\mathcal{O}j$, then node i will be an ancestor of node j in each tree-sample that has both nodes.

Our assumptions imply that two nodes will always satisfy the same hierarchical relation whenever they are both present in a sample tree. A consequence of this assumption is that the structure of a sample tree is completely determined by the restriction of the order relation $\mathcal{O}$ to the nodes observed in the sample tree. The sampling process is equivalent to the application of a set of node removal operations to the archetypical structure $\mathcal{T} = (\mathcal{N}, \mathcal{O}, \Theta)$, and this makes the archetype a union of the set of all possible tree samples.

The following subsections, we describe how to construct a node probability distribution over the set of trees when the nodes are weighted and when they are unweighted.

Unweighted model

To define a probability distribution over the union structure $\mathcal{T}$, we require the correspondences between the nodes in each sample tree t and the nodes in the class-model $\mathcal{T}$. Therefore, we define a map $\mathcal{C} : \mathcal{N}^t \rightarrow \mathcal{N}$ from the set $\mathcal{N}^t$ of the nodes of t to the nodes of the class model $\mathcal{T}$. The mapping induces a sample correspondence for each node $i \in \mathcal{N}$. The correspondence probability for the node i is

$$\phi(i|t, \mathcal{T}, \mathcal{C}) = \begin{cases} \theta_i & \text{if there exists } j \in \mathcal{N}^t \text{ such that } \mathcal{C}(j) = i \\ 1 - \theta_i & \text{otherwise.} \end{cases} \tag{7.1}$$

The probability of sampling the tree t from the model $\mathcal{T}$ given the set of correspondences $\mathcal{C}$ is

$$\Phi(t|\mathcal{T}, \mathcal{C}) = \begin{cases} \prod_{i \in \mathcal{N}^t} \phi(i|t, \mathcal{T}, \mathcal{C}) & \text{if } \forall v, w \in \mathcal{N}^t, v \dashrightarrow w \iff \mathcal{C}(v) \dashrightarrow \mathcal{C}(w) \\ 0 & \text{otherwise.} \end{cases} \tag{7.2}$$

That is, the order relation is respected as a hard constraint on the choice of the correspondences.

Weighted model

When the nodes of the sample trees have weights associated with them, we would expect the sampling likelihood to reflect the distribution of weights. Hence, the simple probability distribution described above, based on uniform sample node probability, is not sufficient because it does not take into account the weight distribution. To overcome this shortcoming, in addition to the set of sampling probabilities Θ, we associate with the union model a weight distribution function. Here we assume that the weight distribution is a rectified Gaussian. For the node i of the union tree the weight probability distribution is given by

$$p(w_j|\mathcal{C}(j)=i) = \begin{cases} \frac{1}{\theta_i \sigma_i \sqrt{2\pi}} \exp\left(-\frac{1}{2}\frac{(w_j-\mu_i)^2}{\sigma_i{}^2}\right) & \text{if } w_j \geq 0 \\ 0 & \text{otherwise,} \end{cases} \tag{7.3}$$

where the weight distribution has mode μ_i and standard deviation σ_i. The sampling probability is the integral of the distribution over positive weights, i.e.

$$\theta_i = \int_0^\infty \frac{\exp\left(-\frac{1}{2}\frac{(w-\mu_i)^2}{\sigma_i{}^2}\right)}{\sigma_i\sqrt{2\pi}} dw = 1 - \operatorname{erfc}(\tau_i), \tag{7.4}$$

where $\tau_i = \mu_i/\sigma_i$ and erfc is the complementary error function

$$\operatorname{erfc}(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} \exp(-\frac{1}{2}s^2)\, ds. \tag{7.5}$$

In the weighted case, the complete structural model is represented by the tuple $\mathcal{T} = (\mathcal{N}, \mathcal{O}, \bar{\tau}, \bar{\sigma})$, where $\bar{\tau} = \{\tau_i; i \in \mathcal{N}\}$ and $\bar{\sigma} = \{\sigma_i; i \in \mathcal{N}\}$ are sets of node parameters. Taking into account the correspondences, the probability for node i induced by the mapping is

$$\phi(i|t,\mathcal{T},\mathcal{C}) = \begin{cases} \theta_i p(w_j|\mathcal{C}(j)=i) & \text{if there exists } j \in \mathcal{N}^t \text{ such that } \mathcal{C}(j)=i \\ 1-\theta_i & \text{otherwise.} \end{cases} \tag{7.6}$$

Using this node probability, we can compute the tree sample probability $\Phi(t|\mathcal{T},\mathcal{C})$ in the same way as in the unweighted case.

7.1.2 Estimating Node Parameters

Using either the weighted or the unweighted node model, we can compute the log-likelihood of the sample data $\mathcal{D}$ given the tree-union model $\mathcal{T}$ and the correspondence mapping function $\mathcal{C}$. Assuming that the sampling process acts independently on the nodes of the structure, the log-likelihood function is

$$\mathcal{L}(\mathcal{D}|\mathcal{T},\mathcal{C}) = \sum_{t\in\mathcal{D}} \sum_{i\in\mathcal{N}^t} \ln\left[\phi(i|t,\mathcal{T},\mathcal{C})\right]. \tag{7.7}$$

Our ultimate aim is to optimize the log-likelihood with respect to the correspondence map $\mathcal{C}$ and the tree-union model $\mathcal{T}$. These variables, however, are not independent, since they both depend on the node-set $\mathcal{N}$. A variation in the actual identity and number of the nodes does not change the log-likelihood. Hence the dependency on the node-set can be lifted by simply assuming that the node set is the image of the correspondence map $\mathrm{Im}(\mathcal{C})$. As we will see later, the reason for this is that those nodes that remain unmapped do not affect the maximization process.

We defer details of how we estimate the correspondence map $\mathcal{C}$ and the order relation $\mathcal{O}$ to later sections of the chapter. However, assuming estimates of them are to hand, we can make maximum likelihood estimates of the selected node model. That is, the set of sampling probabilities Θ in the unweighted case, and the node parameters $\bar{\tau}$ and $\bar{\sigma}$ in the weighted case.

To proceed, let $K_i = \{j \in \mathcal{N}^t | t \in \mathcal{D}, C(j) = i\}$ be the set of nodes in the different trees for which $\mathcal{C}$ maps a node to i and let $p_i = |K_i|$ be the number of trees satisfying this condition. Further, let n_i be the number of trees in $\mathcal{D}$ for which $\mathcal{C}$ results in no mapping to the node i. For the *unweighted* node-model, the sampling probability θ_i that maximizes the likelihood function is $\theta_i = \frac{p_i}{m}$, where $m = n_i + p_i$ is the total number of tree samples. When these optimal sampling probabilities are substituted into the log-likelihood function, we have

$$\hat{\mathcal{L}}(\mathcal{D}|\mathcal{T},\mathcal{C}) = \sum_{i\in\mathcal{N}} m\left[\frac{n_i}{m}\log\left(\frac{n_i}{m}\right) + \left(1-\frac{n_i}{m}\right)\log\left(1-\frac{n_i}{m}\right)\right] = -\sum_{i\in\mathcal{N}} mI(\theta_i), \tag{7.8}$$

where $I(\theta_i) = -\left[\theta_i \log(\theta_i) + (1-\theta_i)\log(1-\theta_i)\right]$ is the entropy of the sampling distribution for node i. This equation holds provided that there exists an order relation that is respected by every hierarchical tree in the sample set $\mathcal{D}$. If this is not the case then the log-likelihood function takes on the value $-\infty$.

Using the *weighted* node model, we are unable to express the log-likelihood function as the sum of the per node entropies. However, we can express it as the sum of the per-node log-likelihood functions:

$$\begin{aligned}\mathcal{L}(\mathcal{D}|\mathcal{T},\mathcal{C}) &= \textstyle\sum_{i\in\mathcal{N}} \phi(i|t,\mathcal{T},\mathcal{C}) \\ &= \textstyle\sum_{i\in\mathcal{N}} n_i \log(\operatorname{erfc}(\tau^i)) - \frac{p_i}{2}\log(2\pi\sigma_i^2) - \frac{1}{2}\sum_{j\in K_i}\left(\frac{w_j^t}{\sigma_i} - \tau_i\right)^2 \\ &= \textstyle\sum_{i\in\mathcal{N}} \log\left(\operatorname{erfc}(\tau_i)^{n_i}(2\pi\sigma_i^2)^{-\frac{p_i}{2}}\exp\left[-\frac{1}{2}\sum_{j\in K_i}\left(\frac{w_j^t}{\sigma_i}-\tau_i\right)^2\right]\right).\end{aligned} \tag{7.9}$$

To estimate the parameters of the weight distribution, we take the derivatives of the log-likelihood function with respect to σ_i and τ_i and set them to zero. As a result

$$\sigma_i = -\frac{\tau_i}{2}\overline{W} + \sqrt{\left(\frac{\tau_i}{2}\overline{W}\right)^2 + \overline{W^2}} \tag{7.10}$$

$$n^i \operatorname{erfc}'(\tau_i) + p_i \operatorname{erfc}(\tau_i)\left(\frac{\overline{W}}{\sigma_i} - \tau_i\right) = 0, \tag{7.11}$$

where $\overline{W} = \frac{1}{p_i}\sum_{j\in K_i} w_j^t$ and $\overline{W^2} = \frac{1}{p_i}\sum_{j\in K_i}\left(w_j^t\right)^2$.

It is clear that when $n_i = 0$ then the likelihood function is maximized by $\sigma_i = \sqrt{\overline{W^2} - \overline{W}^2}$ and $\tau_i = \frac{\overline{W}}{\sigma_i}$. When $n_i > 0$, we maximize the log likelihood by setting ${\tau_i}^0 = \operatorname{erfc}^{-1}\left(\frac{n_i}{n_i+p_i}\right)$, and iterating the recurrence:

$$\sigma_i^{(k)} = -\frac{\tau_i^{(k)}}{2}\overline{W} + \sqrt{\left(\frac{\tau_i^{(k)}}{2}\overline{W}\right)^2 + \overline{W^2}} \tag{7.12}$$

$$\tau_i^{(k+1)} = \tau_i^{(k)} - \frac{f(\tau_i^{(k)}, \sigma_i^{(k)})}{\frac{d}{d\tau_i^{(k)}} f(\tau_i^{(k)}, \sigma_i^{(k)})}, \tag{7.13}$$

where $f(\tau_i, \sigma_i) = n_i \operatorname{erfc}'(\tau_i) + p_i \operatorname{erfc}(\tau_i) \left(\frac{\overline{W}}{\sigma_i} - \tau_i \right)$.

7.2 Mixture Model

We now commence our discussion of how to estimate the order relation $\mathcal{O}$ for the tree union $\mathcal{T}$, and the set of correspondences $\mathcal{C}$ needed to merge the sample trees to form the tree union. We pose the problem as that of fitting a mixture of tree unions to the set of sample trees. Each tree union may be used to represent the distribution of the trees that belong to a single class $\mathcal{D}_c$. The defining characteristic of the class is the fact that the nodes present in the sample trees satisfy a single order relation $\mathcal{O}_c$. However, the sample set $\mathcal{D}$ may have a complex class structure, and it may be necessary to describe it using multiple tree unions. Under these conditions the unsupervised learning process must allow for multiple classes. We represent the distribution of sample trees using a mixture model and applying it to separate union structures. Suppose that there are k tree unions, that the tree union for the class c is denoted by $\mathcal{T}_c$, and that the mixing proportion for this tree-union is α_c. The mixture model for the distribution of sample trees is

$$P(t|\bar{\mathcal{T}}, \mathcal{C}) = \sum_{c=1}^{k} \alpha_c \prod_{t \in \mathcal{D}} \prod_{i \in \mathcal{N}^t} \phi(i|t, \mathcal{T}_c, \mathcal{C}). \tag{7.14}$$

The expected log-likelihood function for the mixture model over the sample-set $\mathcal{D}$ is:

$$\mathcal{L}(\mathcal{D}|\bar{\mathcal{T}}, \mathcal{C}, \bar{z}) = \sum_{t \in \mathcal{D}} \sum_{i \in \mathcal{N}^t} \sum_{c=1}^{k} z_c^t \alpha_c \ln \phi(i|t, \mathcal{T}_c, \mathcal{C}), \tag{7.15}$$

where z_c^t is an indicator variable that takes on the value 1 if tree t belongs to the mixture component c and is 0 otherwise.

We require an information criterion that can be used to select the set of tree-merge operations over the sample set $\mathcal{D}$ that results in the optimal set of tree unions. It is well known that the maximum likelihood criterion cannot be used directly to estimate the number of mixture components, since the maximum of the likelihood function is a monotonic function on the number of components. In order to overcome this problem, we use the Minimum Description Length (MDL) principle [64], which asserts that the model that best describes a set of data is the one that minimizes the combined cost of encoding the model and encoding the error between the model and the data. The MDL principle allows us to select from a family of possibilities the model that is most parsimonious and that best approximates the underlying data.

More formally, the expected description length of a data set $\mathcal{D}$ generated by an estimate $\mathcal{H}$ of the true, or underlying, model $\mathcal{H}^*$ is

$$\begin{aligned}E\left[\mathrm{LL}(\mathcal{D},\mathcal{H})\right] = -\int P(\mathcal{D}|\mathcal{H}^*)\log\left[P(\mathcal{D}|\mathcal{H})P(\mathcal{H})\right]\,d\mathcal{D} = \\ -\frac{1}{P(\mathcal{H}^*)}\int P(\mathcal{D},\mathcal{H}^*)\log\left[P(\mathcal{D},\mathcal{H})\right]\,d\mathcal{D} = \\ -\frac{1}{P(\mathcal{H}^*)}\left[\int P(\mathcal{D},\mathcal{H}^*)\log\left(P(\mathcal{D},\mathcal{H}^*)\right)\,d\mathcal{D} + \int P(\mathcal{D},\mathcal{H}^*)\log\left(\frac{P(\mathcal{D},\mathcal{H})}{P(\mathcal{D},\mathcal{H}^*)}\right)\,d\mathcal{D}\right] = \\ \frac{1}{P(\mathcal{H}^*)}\left[I(P(\mathcal{D},\mathcal{H}^*)) + KL(P(\mathcal{D},\mathcal{H}^*),P(\mathcal{D},\mathcal{H}))\right],\end{aligned} \quad (7.16)$$

where

$$I(P(\mathcal{D},\mathcal{H}^*)) = -\int P(\mathcal{D},\mathcal{H}^*)\log\left(P(\mathcal{D},\mathcal{H}^*)\right)\,d\mathcal{D} \quad (7.17)$$

is the entropy of the joint probability of the data and the underlying model $\mathcal{H}^*$, and

$$KL(P(\mathcal{D},\mathcal{H}^*),P(\mathcal{D},\mathcal{H})) = -\int P(\mathcal{D},\mathcal{H}^*)\log\left(\frac{P(\mathcal{D},\mathcal{H})}{P(\mathcal{D},\mathcal{H}^*)}\right)\,d\mathcal{D} \quad (7.18)$$

is the Kullback-Leiber divergence between the joint probabilities using the underlying model $\mathcal{H}^*$ and the estimated model $\mathcal{H}$. This quantity is clearly minimized when $\mathcal{H} = \mathcal{H}^*$, and hence $P(\mathcal{D}, \mathcal{H}) = P(\mathcal{D}, \mathcal{H}^*)$. Under these conditions, $KL(P(\mathcal{D}, \mathcal{H}^*), P(\mathcal{D}, \mathcal{H})) = 0$ and $E[LL(\mathcal{D}, \mathcal{H})] = I(P(\mathcal{D}, \mathcal{H}))$. In other words, the description length associated with the maximum likelihood set of parameters is simply the expected value of the negative log likelihood, i.e. the Shannon entropy.

Our model is described by the set of mixing proportions $\bar{\alpha}$ and the set of union structures $\mathcal{H} = \{\mathcal{T}_1, \ldots, \mathcal{T}_c, \ldots, \mathcal{T}_k\}$. The union structure $\mathcal{T}_c = \{\mathcal{N}_c, \mathcal{O}_c, \Theta_c\}$ for the mixture component indexed c consists of a set of nodes $\mathcal{N}_c$, a set of order relations $\mathcal{O}_c$ and a set of node probabilities $\Theta_c = \{\theta_i^c, i \in \mathcal{N}_c\}$, where θ_i^c is the probability for the node n in the union-tree indexed c. To describe or encode the fit of the model to the data, for each tree sample t we use the indicator variables $\bar{z}_c^t$ which indicates from which tree model the sample was drawn. Additionally, for each node in the model, we need to describe or encode whether or not the node was present in the sample and, in the case of the weighted model, we need to specify the value of the sampled weight within a given precision.

In the following subsections, we analyze the description length criterion for the unweighted and weighted node models.

7.2.1 Unweighted Samples

As noted above, the cost incurred in describing or encoding the model $\bar{\mathcal{T}}$ is $-\log\left[P(\bar{\mathcal{T}})\right]$, while the cost of describing the data $\mathcal{D}$ using that model is $-\log\left[P(\mathcal{D}|\bar{\mathcal{T}})\right]$. If we make the dependence on the correspondences $\mathcal{C}$ explicit, we see that the description length is

$$LL(\mathcal{D}|\mathcal{T}) = -\mathcal{L}(\mathcal{D}|\bar{\mathcal{T}}, \mathcal{C}). \tag{7.19}$$

Asymptotically, the cost of describing the set of mixing components $\bar{\alpha} = \{\alpha_c; c = 1, ..., k\}$ and the set of indicator variables $\bar{z} = \{z^t_c | t \in \mathcal{D}, c = 1, ..., k\}$ is bounded by $mI(\bar{\alpha})$, where m is the number of samples in $\mathcal{D}$ and $I(\bar{\alpha}) = -\sum_{c=1}^{k} \alpha_c \log(\alpha_c)$ is the entropy of the mixture distribution $\bar{\alpha}$. The cost of describing the structure of a union model is proportional to the number of nodes contained within it, while the cost of describing the sampling probability θ^c_i of node i for model c and the existence of this node in each of the $m\alpha_c$ samples generated by union c is asymptotically equal to $m\alpha_c I(\theta^c_i)$. Here $I(\theta^c_i) = -\theta^c_i \log(\theta^c_i) - (1 - \theta^c_i) \log(1 - \theta^c_i)$ is the entropy associated with the node sampling probability. Hence, given a model $\mathcal{H}$ consisting of k tree-unions, where the component $\mathcal{T}_c$ has d_c nodes and a mixing proportion α_c, the descriptor length for the model, conditioned to the set of correspondences $\mathcal{C}$ is:

$$\mathrm{LL}(\mathcal{D}|\mathcal{H}, \mathcal{C}) = mI(\bar{\alpha}) + \sum_{c=1}^{k} \sum_{i=1}^{d_c} \left[m\alpha_c I(\theta^c_i) + l \right], \tag{7.20}$$

where l is the description length per node of the tree-union structure, which we set to 1. Given that $m_c = \sum_{t \in \mathcal{D}_c} z^t_c$ is the number of trees mapped to model c, the sampling frequency α_c is estimated using $\alpha_c = \frac{m_c}{m}$. Furthermore, given that $p^c_i = |\{j \in \mathcal{N}^t | t \in \mathcal{D}, \mathcal{C}(j) = i\}|$ is the number of nodes from all the sample trees in $\mathcal{D}$ that are mapped by $\mathcal{C}$ to node i of model c, the node probability θ^c_i is estimated using $\theta^c_i = \frac{p^c_i}{m_c}$.

7.2.2 Weighted Samples

We now turn our attention to the case of weighted trees. To pursue our analysis of the description length criterion, we encode the weight distribution as a histogram: we divide the weight space of the samples associated with node i of union-tree c into buckets of width $k\sigma^c_i$. As a result, the probability that a weight falls in a bucket centered at x is, for infinitesimally small k

$$b_c^i(x) = k \frac{\exp\left[-\frac{1}{2}\left(\frac{x}{\sigma_i^c} - \tau_i^c\right)^2\right]}{\theta_i^c\sqrt{2\pi}}. \tag{7.21}$$

Again, the asymptotic cost of describing the node parameters τ_i^c and σ_i^c and, at the same time, describing within the specified precision the $n\alpha_c$ samples associated to node i in union c, is

$$\mathrm{LL}_c^i(\mathcal{D}|\mathcal{T}_c, \mathcal{C}) = -(m\alpha_c - p_i)\log(1-\theta_i^b) - \sum_{j=1}^{p_i} \log\left(b_c^i(w_j^i)\right), \tag{7.22}$$

where $\theta_i^c = 1 - \mathrm{erfc}(\tau_i)$ is the sampling probability for node i and p_i is the number of times the correspondence $\mathcal{C}$ maps a sample node to i. Hence $(m\alpha_c - p_i)$ is the number of times node i has not been sampled according to the correspondence map $\mathcal{C}$. As a result

$$\mathrm{LL}(\mathcal{D}|\mathcal{H}, \mathcal{C}) = mI(\bar{\alpha}) + \sum_{c=1}^{k} \sum_{i \in \mathcal{N}_c} \left[\mathrm{LL}_c^i(\mathcal{D}|\mathcal{T}_c, \mathcal{C}) + l\right]. \tag{7.23}$$

7.3 Learning the Mixture

With the description length criterion to hand, our aim is to locate tree-merge operations that give rise to the set of tree unions that optimally partition the training data $\mathcal{D}$ into non-overlapping classes. Unfortunately, locating the global minimum of the description length in this way is an intractable combinatorial problem. Moreover, we can not make use of the Expectation-Maximization algorithm. This is because the complexity of the maximization step grows exponentially with the number of sample trees. Hence, we resort to a local search technique, which allows us to limit the complexity of the maximization step. The approach is as follows:

- Commence with an overly-specific model. Use one structural model per sample-tree, where each model is equiprobable and structurally identical to the respective sample tree and each node has sample probability equal to 1.

- Iteratively generalize the model by merging pairs of tree-unions. Choose the candidates for merging in such a way that they maximally decrease the description length.

- Stop the algorithm when there are no merges remaining that can decrease the description length.

This algorithm bears some resemblance with the spanning tree clustering algorithm [39]. Both algorithms iteratively merge samples or clusters that satisfy a minimum distance or a maximum similarity criterion. The main difference is that, in our algorithm, the similarity matrix cannot be assumed to be fixed, as is the case with the spanning tree algorithm. Rather, it changes after each merge to reflect the changes in the joint model. These changes in the distance matrix will limit the amount of chaining allowed in the clusters. This is due to the fact that the tree unions describing the two clusters that are merged are replaced by a single union which must be able to describe the variation present in both clusters. As a result, its mean must be placed in pattern-space somewhere between the means of the two tree unions. This implies that the distance to the remaining clusters must vary. Regardless of these differences, our algorithm is still guaranteed to converge to a local minimum with at most a linear number of merges.

The main requirement of our description length minimization algorithm is that we can optimally merge two tree models. That is, we can find a structure from which it is possible to sample every tree previously assigned to the two models. From Equations 7.20 and 7.23 we see that the descriptor length is linear with respect to the contribution from each component of the mixture. In fact, the description cost of the component c and of the $n\alpha_c$ data samples assigned to it is

$$\mathrm{LL}_c(\mathcal{D}|\mathcal{T}_c, \mathcal{C}) = \sum_{i \in \mathcal{N}_c} \left[m\alpha_c I(\theta_i^c) + l \right] \tag{7.24}$$

for the *unweighted* node-model, and

$$\mathrm{LL}_c(\mathcal{D}|\mathcal{T}_c, \mathcal{C}) = -\sum_{i \in \mathcal{N}_c} (m\alpha_c - p_i) \log(1 - \theta_i^b) - \sum_{i \in \mathcal{N}_c} \sum_{j=1}^{p_i} \log\left(b_c^i(w_j^i) \right) \tag{7.25}$$

for the *weighted* node model. The total description cost, thus, becomes:

$$\mathrm{LL}(\mathcal{D}|\hat{\mathcal{T}}, \mathcal{C}) = mI(\bar{\alpha}) + \sum_{c=1}^{k} \mathrm{LL}_c(\mathcal{D}|\mathcal{T}_c, \mathcal{C}). \tag{7.26}$$

Furthermore, the description length per component $LL_c(\mathcal{D}|\mathcal{T}_c, \mathcal{C})$ is linear in the number of model nodes. This allows us to pose the minimization of the description length as a linear optimization problem with a combinatorial constraint. In particular, as we will show in the next section, we can pose the model-merging problem as an instance of a particular minimum edit-distance problem.

Given two tree models $\mathcal{T}_1$ and $\mathcal{T}_2$, we wish to construct a union $\hat{\mathcal{T}}$ whose structure respects the hierarchical constraints present in both $\mathcal{T}_1$ and $\mathcal{T}_2$, and which minimizes the quantity $\mathrm{LL}(\hat{\mathcal{T}})$. Since the trees $\mathcal{T}_1$ and $\mathcal{T}_2$ already assign node correspondences $\mathcal{C}_1$ and $\mathcal{C}_2$ from the data samples to the model, we can simply find a map $\mathcal{M}$ from the nodes in $\mathcal{T}_1$ and $\mathcal{T}_2$ to $\hat{\mathcal{T}}$ and transitively extend the correspondences from the samples to the final model $\hat{\mathcal{T}}$ in such a way that, given two nodes $v \in \mathcal{N}_1$ and $v' \in \mathcal{N}_2$, then $\hat{\mathcal{C}}(v) = \hat{\mathcal{C}}(v') \Leftrightarrow v' = \mathcal{M}(v)$.

Posed as the merge of two structures, the correspondence problem is reduced to one of finding the set of nodes in $\mathcal{T}_1$ and $\mathcal{T}_2$ that are common to both trees. Starting with the two structures, we merge the set of nodes that reduces the description length by the largest amount while still satisfying the hierarchical constraint. That is to say, we merge nodes u and v of $\mathcal{T}_1$ with node u' and v' of $\mathcal{T}_2$ respectively if and only if $u \dashrightarrow v \Leftrightarrow u' \dashrightarrow v'$, where $a \dashrightarrow b$ indicates that a is an ancestor of b.

Let m_1 and m_2 be the number of tree samples from $\mathcal{D}$ that are respectively assigned to $\mathcal{T}_1$ and $\mathcal{T}_2$. Further, let p_v and $p_{v'}$ be the number of times the nodes v and v' in $\mathcal{T}_1$ and $\mathcal{T}_2$ are respectively in correspondence with nodes of trees in the data sample $\mathcal{D}$. With the *unweighted* model, if the two nodes are not merged then the sampling probabilities are $\theta_v = \frac{p_v}{m_1+m_2}$ and $\theta_{v'} = \frac{p'_v}{m_1+m_2}$ respectively, while the sampling probability of the merged node is $\theta_{(vv')} = \frac{p_v+p_{v'}}{m_1+m_2}$. Hence, the advantage in description length obtained by merging the nodes v and v' is

$$\mathcal{A}(v, v') = (m_1 + m_2)\left[I(\theta_v) + I(\theta_{v'}) - I(\theta_{(vv')})\right] + l. \qquad (7.27)$$

When using the *weighted* model, we can still estimate the sampling probabilities using the formulae $\theta_v = 1 - \mathrm{erfc}(\tau_v)$, $\theta_{v'} = 1 - \mathrm{erfc}(\tau_{v'})$, and $\theta_{(vv')} = 1 - \mathrm{erfc}(\tau_{(vv')})$, where the node parameters τ_v, $\tau_{v'}$, and $\tau_{(vv')}$ are estimated by fitting the node weight model to the unmerged and merged node data. The combined cost incurred in describing these models is $\mathrm{LL}^{v'}(\mathcal{D}|\mathcal{T}_c, \mathcal{C}) + \mathrm{LL}^{v'}(\mathcal{D}|\mathcal{T}_c, \mathcal{C}) + 2l$. Hence, in the *weighted* case the advantage in description length obtained by merging the nodes v and v' is:

$$\mathcal{A}(v, v') = \mathrm{LL}^{v}(\mathcal{D}|\mathcal{T}_c, \mathcal{C}) + \mathrm{LL}^{v'}(\mathcal{D}|\mathcal{T}_c, \mathcal{C}) - \mathrm{LL}^{(vv')}(\mathcal{D}|\mathcal{T}_c, \mathcal{C}) + l. \qquad (7.28)$$

In both the unweighted and the weighted cases, the set of merges $\mathcal{M}$ that minimizes the description length of the combined tree union also maximizes

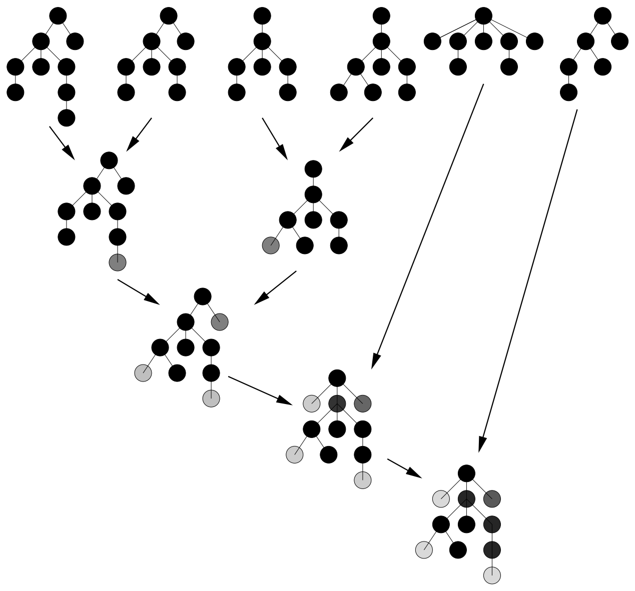


Figure 7.1: Merging sample trees into a single tree-model.

the advantage function

$$\mathcal{A}(\mathcal{M}) = \sum_{(v,v')\in\mathcal{M}} \mathcal{A}(v,v'). \tag{7.29}$$

At the end of the node merging operation we are left with a set of nodes that respects the original partial order relation defined by all the hierarchies in the sample trees. Unfortunately, as stated in the previous chapter, the hierarchy of the merged model is not guaranteed to be a tree order. An illustration of this problem is provided in Figure 6.3. When the merged model does not provide a tree order, we simply reject the merge and search for another pair of trees to be merged.

We initialize our algorithm by calculating the description length of a model in which there is one mixing component per tree sample in $\mathcal{D}$. For each pair of initial mixture components we calculate the union and the description length of the merged structure. From the set of potential merges, we can identify the one which reduces the descriptor cost by the greatest amount.

The mixing proportion for this optimal merge is equal to the sum of the proportions of the individual unions. At this point we calculate the union and the description cost that result from merging the newly obtained model with each of the remaining components. We iterate the algorithm until no more merges that reduce the description length can be found.

To conclude this section, Figure 7.1 illustrates an example merge of 6 sample trees. The figure shows the structural archetype of the merged models after each stage. The shading of the nodes represents their sampling probabilities: the higher the probability the darker the node.

7.4 Tree Edit-Distance

As noted earlier, the advantage in description length is related to the edit-distance between tree structures. This is an important observation. One of the difficulties with graph edit-distance [68, 25] is that there is no clear methodology for assigning costs to edit operations. By contrast, in the work reported here the description length changes associated with tree merge operations are determined by the node probabilities, and these in turn may be estimated from the available sample of trees. By establishing a link between tree edit-distance and description length, we provide a means by which edit costs may be estimated.

Pivotal to our approach is the observation that locating the correspondences that result in the optimal tree merge is equivalent to computing the edit-distance. The reason for this is that both processes share the same hierarchical constraints and the same objective function. Hence, the set of common nodes obtained through the edit-distance approach is equal to the set of nodes that must be optimally merged to form the tree unions.

We can identify the cost associated with node removal and node matching operations by equating the utility $\mathcal{U}$ defined in Equation 5.3 with the advantage in description length $\mathcal{A}$. The costs that allow the *unweighted* problem to be posed as an edit-distance problem are $r_v = (m_1 + m_2)I(\theta_v) + l$ for the removal of node v, and $m_{(vv')} = (m_1 + m_2)I(\theta_{(vv')}) + l$ for matching node v with node v'. In the *weighted* case, the corresponding edit costs are $r_v = \mathrm{LL}^v(\mathcal{D}|\mathcal{T}_c, \mathcal{C}) + l$ for the removal of node v, and $m_{(vv')} = \mathrm{LL}^{(vv')}(\mathcal{D}|\mathcal{T}_c, \mathcal{C}) + l$ for matching node v with node v'.

Hence, our union-tree approach can be viewed as a means of learning tree-edit costs. This has been a longstanding problem since Fu and his co-workers introduced the idea of graph edit-distance in the early 1980's [68, 25].

7.5 Experimental Results

We illustrate the usefulness of the tree-clustering algorithm on sets of shock trees. Our experiments are divided into three parts. We commence by illustrating qualitative examples of the clusters obtained with the two variants of our algorithm. The results suggest that the weighted version is the most effective. We then focus in more detail on some of the quantitative properties of the weighted version. Finally, we carry out a sensitivity analysis on synthetic data.

7.5.1 Clustering

To illustrate the clustering process, we begin with a study on a small database of 25 shapes. In order to assess the quality of the method, we compare the clusters defined by the components of the mixture with those obtained with the methods described in the previous chapters. In the previous methods the

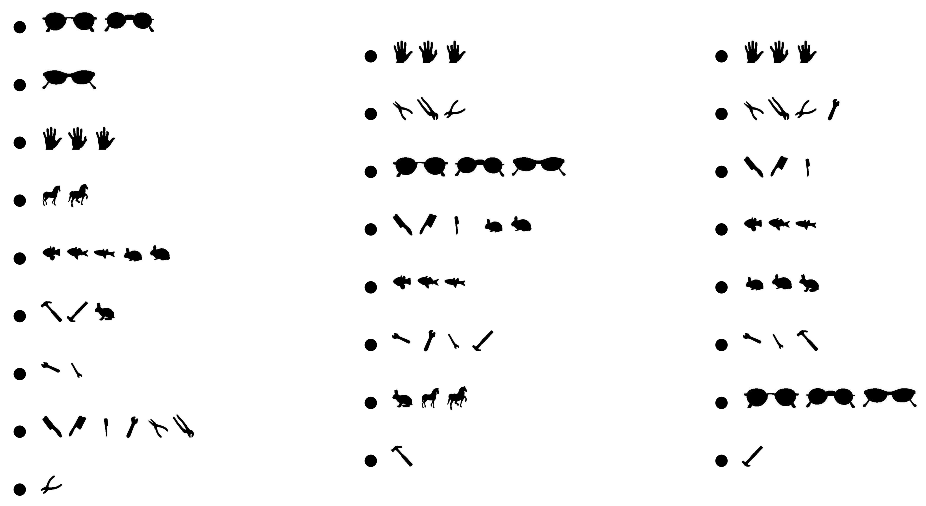
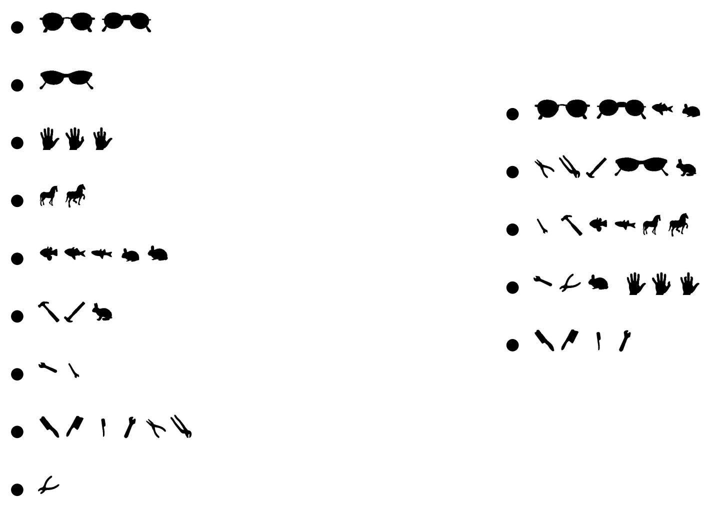

a) Mixture of tree unions b) Weighted edit-distance c) L_1 norm in union-space

Figure 7.2: Comparison of three clustering approaches. a) Mixture of trees b) pairwise clustering of edit-distance c) pairwise clustering of the Union-induced L_1 norm.

clusters are obtained by applying the pairwise clustering algorithm described in Appendix A to the set of distances obtained with plain edit-distance and to the set obtained by embedding the trees in a union structure.

Figure 7.2 shows the clusters extracted from the database of 25 shapes. The first column shows the clusters extracted through the mixture of tree unions approach applied to unweighted trees. The second column displays the clusters extracted from the weighted edit-distances between shock trees. Here the structural information is enhanced with the measure described in Chapter 4. The third column shows the clusters extracted from the distances obtained by embedding the shock trees in a single tree-union. Although there is some merge and leakage, the clusters extracted with the mixture of tree unions compare favorably with those obtained using the alternative clustering algorithms, even though these are based on data enhanced with geometrical information.



a) Mixture of tree models b) Pairwise clustering from edit-distance

Figure 7.3: Comparison of clusters obtained from non-attributed edit-distance and mixture of trees.

Figure 7.3 compares the results obtained through mixture of tree unions, and pairwise clustering of edit-distances, applied to unweighted trees. The clusters obtained using the mixture of tree unions are shown on the left, while those obtained through the pairwise clustering of unweighted edit-distances are shown on the right. These results suggest that the mixture of tree-unions method outperforms pairwise clustering of edit-distance on purely structural data.

7.5.2 Quantitative Analysis

We now turn our attention to the properties of the weighted variant of our mixture-of-tree-unions clustering method when applied to a larger database of 150 trees. The database consists of 150 shapes divided into 10 shape classes containing 15 shapes each.

Figure 7.4 recalls the multi-dimensional scaling applied to the weighted

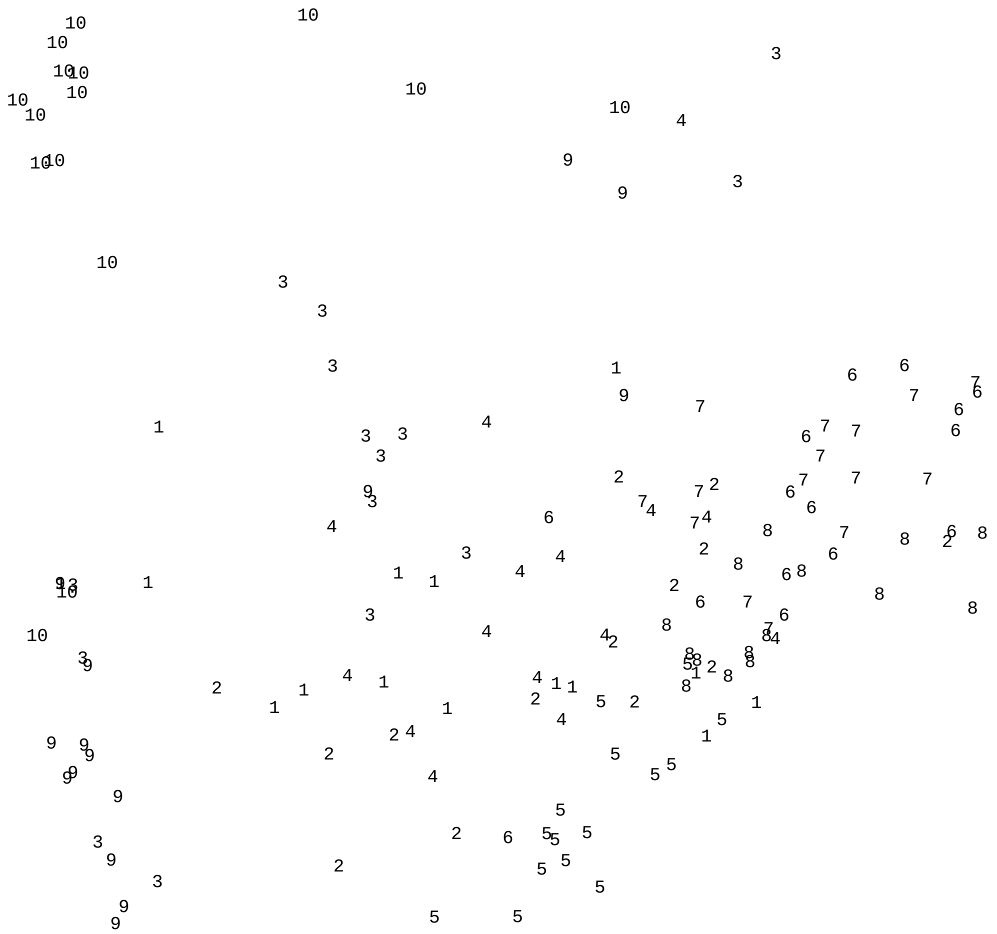


Figure 7.4: 2D multi-dimensional scaling of the pairwise distances of the shock graphs. The numbers correspond to the shape classes.

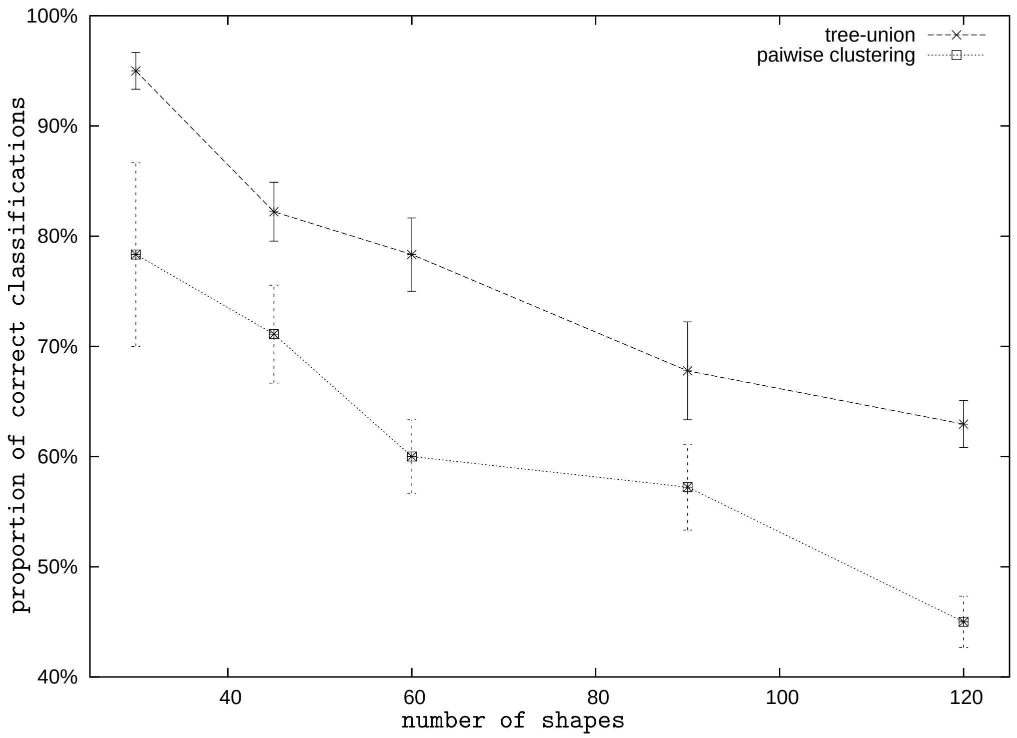


Figure 7.5: Proportion of correct classifications obtained with the mixture of tree versus those obtained with pairwise clustering.

edit-distances of between the shock graphs in this database.

To asses the ability of the clustering algorithm to separate the shape classes, we performed experiments on an increasing number of shapes. We commenced with 30 shapes from two shape classes, and then increased the number of shape classes under consideration up to a set of 120 shapes, forming 8 classes. Each experiment was performed four times with different choices of shape groups in order to provide some error analysis. Figure 7.5 plots the proportion of shapes correctly classified as the number of shapes is increased. The dashed line plots the result obtained using the mixture of weighted tree unions, while the dotted line displays the results obtained with pairwise clustering of the weighted edit-distances between the shapes. The mixture of tree unions clearly outperforms the pairwise clustering algorithm.

We now turn our attention to the results of applying PCA to the union-trees, as described in Section 6.5.1. Figure 7.6 displays the first two principal components of the distribution of sample trees within six of the extracted shape classes. In most cases there appears to be a tightly packed central cluster with a few shapes scattered farther away. This separation is linked to substantial variations in the structure of the shock trees. For example, in the shape-space formed by the class of pliers the outlier is the only pair of pliers with the prongs closed. In the case of shape-space for the horse class, the outliers appear to be the cart-horses while the inliers are the ponies.

7.5.3 Synthetic Data

To augment these real world experiments, we have fitted the mixture of weighted tree unions to synthetically generated data. Our aim here has been to characterize the sensitivity of the algorithm to cluster merging. We have randomly generated a number of unweighted prototype trees and, from each tree, we have generated structurally perturbed copies. The procedure for generating the random trees was the one described in Section 5.5.3. The trees were perturbed by randomly adding the required number of nodes.

The sample of trees used in our study was controlled by increasing the number of prototypes, and increasing the degree of structural perturbation to which they were subjected. We tested the performance of the mixture of weighted tree unions on samples generated from 2, 3, and 4 prototypes of 10 nodes each. The amount of perturbation or noise was increased from an initial 10% to a maximum of 50% of the total number of nodes. Figure 7.7 plots the fraction of pairs of trees that are correctly classified as belonging to the same or different clusters as the noise is increased. From this plot, it is

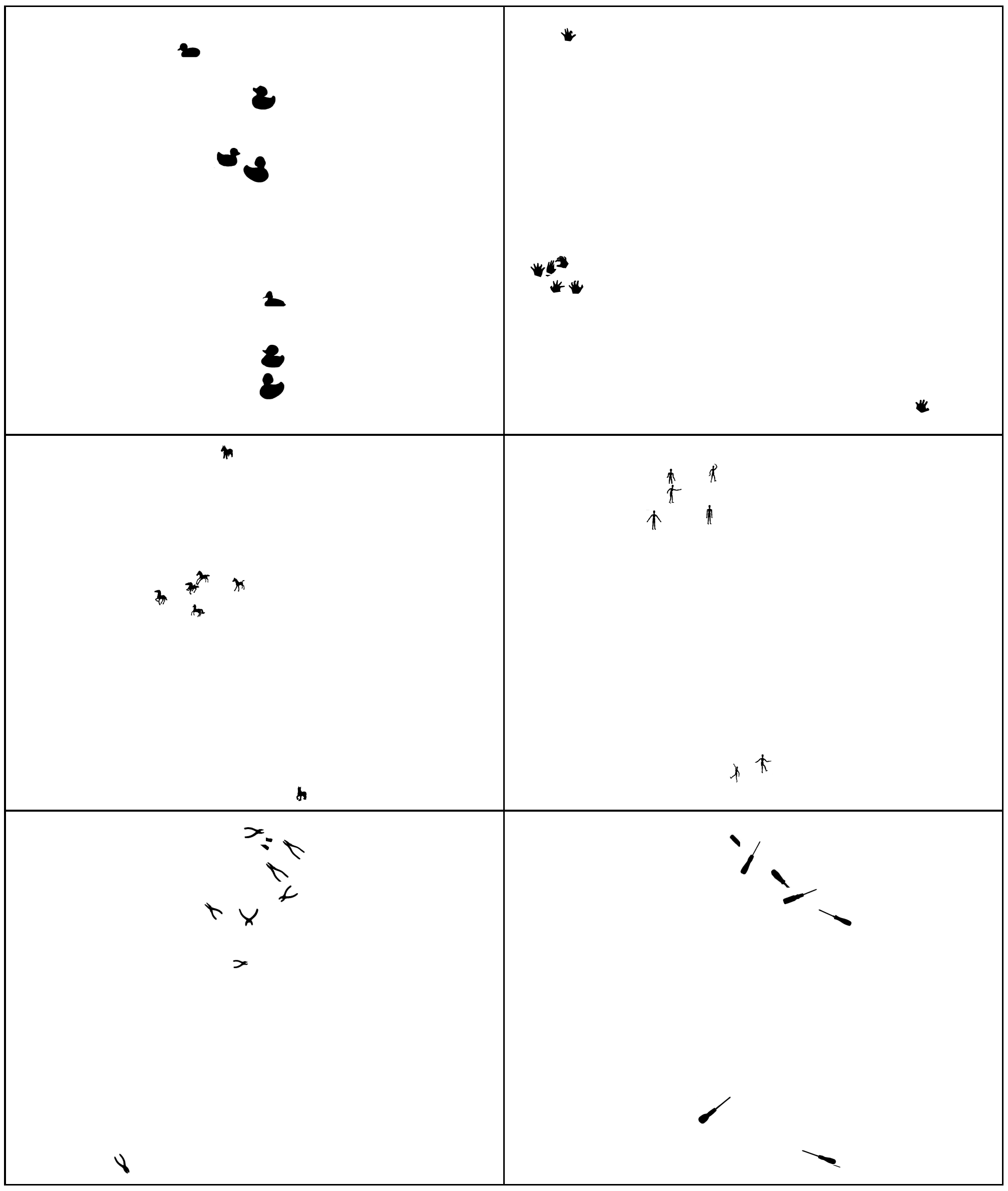


Figure 7.6: Principal components analysis of the union embedding of the clusters.

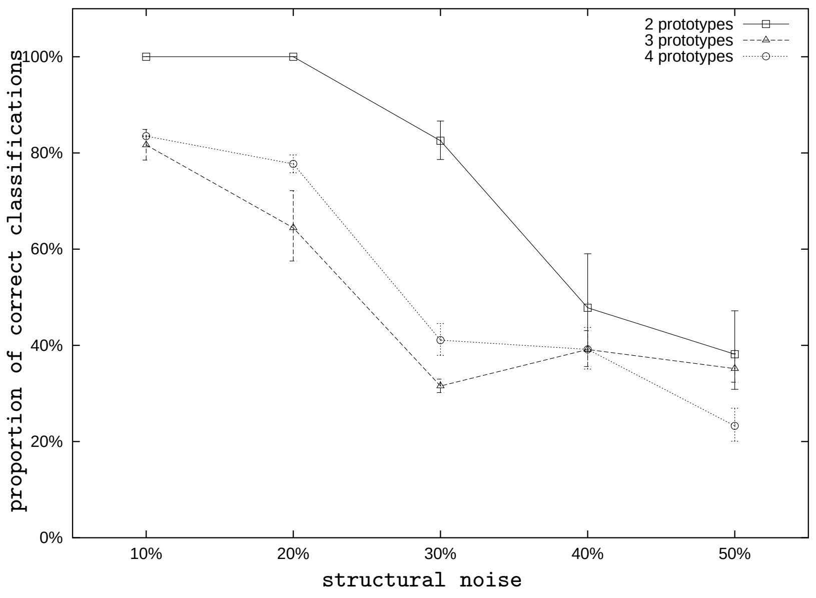


Figure 7.7: Percentage of correct classifications under increasing structural noise.

clear that the method works well with compact and well separated clusters. However, the algorithm undergoes a sudden drop in performance when the structural variability of the class reaches 40% of the total number of nodes of the prototypes. Furthermore, when a greater number of prototypes are used, the distance between the clusters becomes smaller and, consequently the classes become harder to separate.

7.6 Conclusions

In this chapter we present an information theoretic framework for clustering trees and for learning a generative model of the variation in tree structure. The problem is posed as that of learning a mixture of tree unions. We demonstrate how the three sets of operations needed to learn the generative model, namely node correspondence, tree merging and node probability estimation, can each be cast in terms of minimizing a description length criterion. We provide variants of the algorithm that can be applied to samples of both weighted and unweighted trees. Moreover, we illustrate the relationship between classical tree edit-distance, and the node entropy in our model. We illustrate the method in relation to the problem of learning shape-classes from sets of shock trees.

There are clearly a number of ways in which this work may be extended. First, we have concentrated on trees, and there is scope for generalizing the method to graphs. Second, the method only accommodates node probabilities, and an important priority would be to incorporate structures with weighted edges and to allow for edge-probabilities. Third, the optimization process is extremely simplistic, and prone to convergence to local optima. Hence, there is a need to investigate the use of more sophisticated methods such as mean field annealing or evolutionary search. Finally, we have explored only a single application of the method, and more extensive testing is needed. Studies aimed at addressing these points are underway, and will be reported on in future studies.

Chapter 8

Conclusions

The overall goal of this thesis was to learn the class structure of information abstracted in terms of hierarchical trees. To this end, we a) developed an algorithm to approximate tree edit-distance, b) developed a structural archetype capable of capturing the modes of variation of a set of trees, and c) presented an information-theoretic approach to learning these structural archetypes from tree samples. The proposed techniques were analyzed on the problem of unsupervised classification of shapes abstracted in terms of their shock trees.

While the skeletal abstraction of shape was chosen mainly as a experimental vehicle, nonetheless we made some contributions to the fields of skeleton extraction and its graph representation.

8.1 Skeleton Extraction

First, we presented a skeletonization method that corrects curvature effects in the Hamilton-Jacobi framework. Our approach addresses a shortcoming of the Hamilton-Jacobi method for skeleton extraction, namely its sensitivity to high curvature due to curvature-related error terms in the differential analysis. To overcome this problem, we have presented an analysis which takes into account variations of density due to boundary curvature. This yields a skeletonization algorithm that is both better localized and less susceptible to boundary noise than the Hamilton-Jacobi method. Despite the improvements, our analysis of the effects of boundary noise show that noise still affects the extraction algorithm. This is due to the intrinsic sensitivity of the skeletal representation, and is hence present in every extraction algorithm. This intrinsic sensitivity is compounded with a higher incidence of discretization error associated with high frequency in the boundary features. To counter this we smooth the boundary by approximating a diffusion operator. However, in degenerate cases, this could lead to the creation of spurious branches. There is clearly room for improvement in the way we handle boundary noise. One promising direction of investigation is the use of interpolation techniques to compute the flux with sub-pixel precision.

8.2 Shape Measure

A second contribution is in the analysis of the ability of the ratio of boundary-length to skeleton-length as a means to gauge the similarity between shapes abstracted through skeletal representations. The contribution is as follows. Although this ratio that has been known for some time, it has not been used for shape comparison. The novelty of our work resides in the use of the measure to characterize shape similarity and in the analysis of its behavior as the shapes are deformed and the skeleton undergoes a topological transition. The shape-measure has a number of interesting properties that allows it to distinguish between structurally similar shapes. In particular, the measure

a) changes smoothly through topological transitions of the skeleton, b) is able to distinguish between ligature and non-ligature points and to weight them accordingly, and c) exhibits invariance under “bending”. The use of this measure is particularly interesting for its simplicity. For instance, it does not require explicit boundary comparison. Moreover, it can be computed directly using the presented skeleton extraction algorithm. From a theoretical perspective, the contribution is to demonstrate the relationship of the measure to the divergence, and to illustrate a number of important properties that it possesses. Clearly, this measure alone in not enough to completely discriminate arbitrary shapes. For example, the experiments clearly show that the measure has problems with articulated objects. However, it can be a part of a richer description of the shape and can be used for a fast first-tier discrimination between shape groups. Furthermore, the analysis carried out on the measure provides a template for other shape measures. In particular, the requirement that any shape measure must vary in a continuous way when the skeleton undergoes a topological transition is a very important one.

8.3 Edit-Distance

The presented approach to shape recognition and classification using a skeletal representation requires several components. After extracting the skeleton and labeling the branches with some measure of shape-similarity, we need a way to estimate the global similarity of two shapes abstracted in terms of shock trees. We opted to cast the correspondence problem in terms of edit-distance. First, we transformed the tree edit-distance problem into a series of maximum weight clique problems. Then we adopted an optimization approach to approximate the optimal set of correspondence. We have done this by casting the clique problems in a continuous setting by using the Motzkin-Strauss theorem and then using relaxation labeling to find an approximate solution to the continuous problem. There are a number of ways in which this research can be improved upon. The method can be extended to graphs, and different approximation algorithms for the maximum weighted clique problem can be studied.

With the edit-distances to hand, we showed how pairwise clustering can be applied to the set of edit-distances in order to perform unsupervised classification of the graphs. The combination of edit-distance and pairwise clustering proved to suffer from the high level of noise present that is typical of skeletal representations. In fact, the approach was capable of extracting the shape-classes present in the database only when presented with a limited number of shapes. Indeed, too great a number of shapes saturates the shock-tree space and compromises the performance of the approach. The main reason for this is that pairwise graph matching does not enforce node consistency across different matches. This results in a consistent underestimation of the distance of strongly dissimilar shapes.

8.4 Class Archetypes for Graphs

To overcome this problem, we developed a technique to extend the tree edit-distance framework to allow for the simultaneous matching of multiple tree structures. This was done by merging all the trees into a union structure. Using this approach we can impose node correspondence consistency between matches, avoiding underestimation of the distance, typical of pairwise edit-distance approaches. Furthermore, the union provides a “natural” embedding space for tree structures that can be used to analyze how tree representations vary in our problem domain. There are two problems with the direct use of union structures for performing unsupervised classification from sample trees. First, while the archetype is capable of capturing the modes of variation present in a single shape class, the approach fails when confronted with large numbers of shape classes. The second problem is that purely structural methods do not provide a principled approach to what in effect is a learning problem.

Central to our search for a more principled approach to the unsupervised learning of tree-classes is the realization that each tree union can be used to represent the probability distribution of trees within a single class, thus providing us with a generative model of trees. This is done by representing the process of sampling from the distribution in terms of edit-distance. The learning problem is, hence, posed as that of learning a mixture of tree unions. We demonstrated how the three sets of operations needed to learn the generative model, namely node correspondence, tree merging and node probability estimation, can each be cast in terms of minimizing a description length criterion.

There are clearly a number of ways in which this work may be extended. First, we have concentrated on trees, and there is scope for generalizing the method to graphs. Second, the method only accommodates node probabilities, and an important priority would be to incorporate structures with weighted edges and to allow for edge-probabilities. Third, the optimization process is extremely simplistic, and prone to convergence to local optima. Hence, there is a need to investigate the use of more sophisticated methods such as mean field annealing or evolutionary search. In particular, we are interested in reestimating the current set of extracted correspondences after a number of tree merges. Finally, we have explored only a single application of the method, and more extensive testing is needed. Studies aimed at addressing these points are underway, and will be reported on in future studies.

Appendix A: Pairwise Clustering

The process of pairwise clustering is somewhat different to the more familiar one of central clustering. Whereas central clustering aims to characterize cluster-membership using the cluster mean and variance, in pairwise clustering it is the relational similarity of pairs of objects which is used to establish cluster membership. Although less well studied than central clustering, there has recently been renewed interest in pairwise clustering aimed at placing the method on a more principled footing using techniques such as mean-field annealing [36].

To commence, we require some formalism. We are interested in grouping a set of graphs $\mathcal{G} = \{G_1, ..., G_{|M|}\}$ whose index set is M. The set of graphs is characterized using a matrix of pairwise similarity weights. The elements of this weight matrix are computed using the approximate tree edit distance $d_{i,j}$ between the shock trees indexed i and j.

Similarity Matrix

We adopt the following picture of the graph clustering process. The picture revolves around the idea that the graphs can be embedded in an n-dimensional space $\mathcal{R}^n$. Here we treat the embedding space as a latent representation. Hence we are not concerned with a specific embedding procedure. However, a number of concrete possibilities exist. For instance, features could be extracted from the graph adjacency structure and subjected to principal components analysis, or the pattern of pairwise distances could be subjected to multi-dimensional scaling. In this way each graph would become a point in the embedding space. We assume that for each distinct cluster of graphs the embedded position vectors follow a spherically symmetric Gaussian distribution. For the cluster with index ω, the covariance matrix is $\sigma_\omega I_n$ where n is the $n \times n$ identity matrix. Suppose that $x_{i\omega}$ and $x_{j\omega}$ represent the embedded position vectors for the graphs G_i and G_j, and that the graphs both belong to the cluster indexed ω. The difference in position between the graphs, i.e. $x_{i\omega} - x_{j\omega}$ will be drawn from the normal distribution $\mathcal{N}(0, 4\sigma_\omega^2 I)$. As a result the distance measure

$$\left\| \frac{x_{i\omega} - x_{j\omega}}{2\sigma_\omega} \right\| = \frac{d_{ij}^2}{4\sigma_\omega^2} \approx \chi_n^2 \tag{1}$$

will follow a χ^2 distribution with n degrees of freedom.

Given a distance d_{ij} between two points i and j, we can estimate the probability that the two points belong to the same cluster ω' using the χ^2 distribution, provided that we know the cluster variance $\sigma_{\omega'}^2$. The estimated probability is:

$$P\{i \in \omega' \text{ and } j \in \omega'\} = P\left\{ \chi_n^2 > \frac{d_{ij}^2}{4\sigma_{\omega'}^2} \right\}. \tag{2}$$

Using this simple model, we can define the similarity matrix W setting its coefficients W_{ij} to the probability that the graphs i and j belong to the same cluster. In other words:

$$W_{ij} = P\left\{ \chi_n^2 > \frac{d_{ij}^2}{4\sigma_{\omega'}^2} \right\}. \tag{3}$$

Clustering

The aim in graph-clustering is to update a set of similarity weights which partition the set of graphs into disjoint subsets. Let S_ω represent the index-set of the cluster of graphs indexed ω. Since the different clusters are disjoint $S_{\omega'} \cap S_{\omega''} = \emptyset$ if $\omega' \neq \omega''$.

Here, we are interested in using matrix factorization methods to locate the clusters. One way of viewing this is to search for the permutation matrix which re-orders the elements of W into non-overlapping blocks. However, when the elements of the matrix W are non-binary in nature, then this is not a straightforward task. However, Sarkar and Boyer [69] have shown how the same-sign eigenvectors of the matrix of similarity-weights can be used for clustering. Using the Rayleigh-Ritz theorem, they observe that the scalar quantity $\mathbf{v}^t W \mathbf{v}$, where W is the weighted adjacency matrix, is maximized when $\mathbf{v}$ is the leading eigenvector of W. Moreover, each of the subdominant eigenvectors corresponds to a disjoint cluster. We confine our attention to the same-sign eigenvectors (i.e. those whose corresponding eigenvalues are real and positive, and whose components are either all positive or are all negative in sign). If a component of a same-sign eigenvector is non-zero, then the corresponding node belongs to the cluster associated with the eigenmodes of the similarity weight matrix. The eigenvalues $\lambda_1, \lambda_2....$ of W are the solutions of the equation $|W - \lambda I| = 0$ where I is the $|M| \times |M|$ identity matrix. The corresponding eigenvectors $\mathbf{v}_{\lambda_1}, \mathbf{v}_{\lambda_2}, ...$ are found by solving the equation $W\mathbf{v}_{\lambda_i} = \lambda_i \mathbf{v}_{\lambda_i}$. Let the set of same-sign eigenvectors be represented by $\Omega = \{\omega | \lambda_\omega > 0 \wedge [(\mathbf{v}^*_\omega(i) > 0 \forall i) \vee \mathbf{v}^*_\omega(i) < 0 \forall i])\}$. Since the same-sign eigenvectors are orthogonal, this means that there is only one value of ω for which $\mathbf{v}^*_\omega(i) \neq 0$. In other words, each node i is associated with a unique cluster. We denote the set of nodes assigned to the cluster with modal index

ω as $S_\omega = \{i|\mathbf{v}^*_\omega(i) \neq 0\}$.

Maximum Likelihood Framework

We are interested in exploiting the factorization property of Sarkar and Boyer [69] to develop a maximum likelihood method for updating the similarity-weight matrix W. We commence by facto-rising the likelihood-function over the set of modal clusters of the similarity-weight matrix. Since the set of modal clusters are disjoint we can write:

$$P(W) = \prod_{\omega \in \Omega} P(\Phi_\omega), \tag{4}$$

where $P(\Phi_\omega)$ is the probability distribution for the set of similarity-weights belonging to the modal-cluster indexed ω. To model the component probability distributions, we introduce a cluster membership indicator $s_{i\omega}$ which models the degree of affinity of the graph indexed i to the cluster with modal index ω.

Using these variables, we develop a model of probability distribution for the similarity-weights associated with the individual clusters. We assume that the distribution can be factorized over the set of pairwise associations $\Phi_\omega = S_\omega \times S_\omega - \{(i,i)|i \in M\}$ with each cluster and write

$$P(\Phi_\omega) = \prod_{(i,j)\in\Phi_\omega} P(W_{i,j}). \tag{5}$$

To model the probability distribution for the individual link-weights, we adopt the Bernoulli distribution

$$p(W_{i,j}) = W_{i,j}^{s_{i\omega}s_{j\omega}}(1 - W_{i,j})^{1-s_{i\omega}s_{j\omega}}. \tag{6}$$

This distribution takes on its largest values either when the similarity weight W_{ij} is unity and $s_{i\omega} = s_{j\omega} = 1$, or when the similarity-weight $W_{i,j} = 0$ and $s_{i\omega} = s_{j\omega} = 0$.

With these ingredients the log-likelihood function for the observed pattern of similarity-weights is:

$$\mathcal{L} = \sum_{\omega\in\Omega} \sum_{(i,j)\in\Phi_\omega} \Big\{ s_{i\omega}s_{j\omega} \ln W_{ij} + (1 - s_{i\omega}s_{j\omega}) \ln(1 - W_{i,j}) \Big\}. \tag{7}$$

Posed in this way the structure of the log-likelihood function has two features which are reminiscent of the expectation-maximization algorithm. First, the modes of the link-weight matrix play the role of mixing components. The product of cluster-membership variables $s_{i\omega}s_{j\omega}$ plays the role of an *a posteriori* measurement probability. Second, the similarity-weights are the parameters which must be estimated. However, there are important differences. The most important of these is that the modal clusters are disjoint. As a result there is no mixing between them.

Based on this observation, we will exploit an EM-like process to update the similarity-weights and the cluster-membership variables. In the "M" step we will locate maximum likelihood similarity-weights. In the "E" step we will use the revised similarity-weight matrix to update the modal clusters. To this end we index the similarity-weights and cluster memberships with iteration number and aim to optimize the quantity

$$Q(W^{(n+1)}|W^{(n)}) = \sum_{\omega\in\Omega} \sum_{(i,j)\in\Phi_\omega} \Big\{ s_{i\omega}^{(n)} s_{j\omega}^{(n)} \ln \frac{W_{ij}^{(n+1)}}{1 - W_{ij}^{(n+1)}} + \ln(1 - W_{i,j}^{(n+1)}) \Big\}. \tag{8}$$

The revised similarity-weights are indexed at iteration $n+1$ while the cluster-memberships are indexed at iteration n.

Expectation

To update the cluster-membership variables we have used a gradient-based method. We have computed the derivatives of the expected log-likelihood function with respect to the cluster-membership variable

$$\frac{\partial Q(W^{(n+1)}|W^{(n)})}{\partial s_{i\omega}^{(n+1)}} = \sum_{j \in S_\omega} s_{j\omega}^{(n)} \ln \frac{W_{ij}^{(n+1)}}{1 - W_{ij}^{(n+1)}}. \tag{9}$$

Since the associated saddle-point equations are not tractable in closed form, we use the soft-assign ansatz to update the cluster membership assignment variables. As a result the update equation for the cluster membership indicator variables is:

$$s_{i\omega}^{(n+1)} = \frac{\prod_{j \in S_\omega} \left\{ \frac{W_{i,j}^{(n+1)}}{1-W_{ij}^{(n+1)}} \right\}^{s_{j\omega}^{(n)}}}{\sum_{i \in S_\omega} \prod_{j \in S_\omega} \left\{ \frac{W_{ij}^{(n+1)}}{1-W_{ij}^{(n+1)}} \right\}^{s_{j\omega}^{(n)}}}. \tag{10}$$

We initialize the cluster membership variables using the same sign eigenvectors and set

$$s_{iw}^{(0)} = \frac{|\mathbf{v}_{\omega_0}^*(i)|}{\sum_{i \in S_{\omega_0}} |\mathbf{v}_{\omega_0}^*(i)|}. \tag{11}$$

Maximization

Once the revised cluster membership variables are to hand, we can apply the maximization step of the algorithm to update the similarity-weight matrix. The updated similarity-weights are found by computing the derivatives of the expected log-likelihood function

$$\frac{\partial Q(W^{(n+1)}|W^{(n)})}{\partial W_{ij}^{(n+1)}} = \sum_{\omega \in \Omega} \left\{ s_{i\omega}^{(n)} s_{j\omega}^{(n)} \frac{1}{W_{ij}^{(n+1)}(1 - W_{ij}^{(n+1)})} - \frac{1}{1 - W_{ij}^{(n+1)}} \right\} \tag{12}$$

and solving the saddle-point equations

$$\frac{\partial Q(W^{(n+1)}|W^{(n)})}{\partial W_{ij}^{(n+1)}} = 0. \tag{13}$$

As a result the updated link-weights are given by

$$W_{ij}^{(n+1)} = \frac{1}{|\Omega|} \sum_{\omega \in \Omega} s_{i\omega}^{(n)} s_{j\omega}^{(n)}. \tag{14}$$

In other words, the similarity-weight for the pair of nodes (i, j) is simply the average of the product of individual node cluster memberships. Since each graph is associated with a unique cluster, this means that the updated similarity-weight matrix is composed of non-overlapping blocks. Moreover, the similarity-weights are are guaranteed to be in the interval $[0, 1]$.

Algorithm Description

Finally, to summarize, the iterative steps of the algorithm are as follows:

1. *Initialization*: Compute the eigenvectors of the initial current link-weight matrix $W^{(0)}$. Each same-sign eigenvector whose eigenvalue is positive is used to seed a different component of the mixture model.

2. *Expectation*: Compute the updated cluster-membership variables using the E-step (Equation (11)).

3. *Maximization*: Update the link-weights using the M-step to compute the updated link weight matrix $W^{(n)}$ (Equation (14)).

4. Repeat steps (2) and (3) until convergence is reached.

Appendix B: Multi-Dimensional Scaling

Multi-dimensional scaling(MDS)[15] is a procedure which allows data specified in terms of a matrix of pairwise distances to be embedded in a Euclidean space. The classical multi-dimensional scaling method was proposed by Torgenson[89] and Gower[32]. Here we intend to use this method to embed shock trees in a low-dimensional space.

Suppose that $d_{i1,i2}$ is the edit-distance between the shock trees indexed $i1$ and $i2$. The first step of MDS is to calculate a matrix T whose element with row r and column c is given by

$$T_{rc} = -\frac{1}{2}[d_{rc}^2 - \hat{d}_{r.}^2 - \hat{d}_{.c}^2 + \hat{d}_{..}^2], \tag{15}$$

where

$$\hat{d}_{r.} = \frac{1}{N}\sum_{c=1}^{N} d_{rc} \tag{16}$$

is the average dissimilarity value over the rth row, $\hat{d}_{.c}$ is the similarly defined average value over the cth column, and

$$\hat{d}_{..} = \frac{1}{N^2}\sum_{r=1}^{N}\sum_{c=1}^{N} d_{r,c} \tag{17}$$

is the average similarity value over all rows and columns of the similarity matrix T.

We subject the matrix T to an eigenvector analysis to obtain a matrix of embedding co-ordinates X. If the rank of T is $k, k \leq N$, then we will have k non-zero eigenvalues. We arrange these k non-zero eigenvalues in descending order, i.e. $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_k > 0$. The corresponding ordered eigenvectors are denoted by $\vec{e_i}$ where λ_i is the ith eigenvalue. The embedding co-ordinate system for the shock trees is

$$X = [\vec{f_1}, \vec{f_2}, \ldots, \vec{f_k}], \tag{18}$$

where $\vec{f_i} = \sqrt{\lambda_i}\vec{e_i}$ are the scaled eigenvectors. The vector embedding the shock tree indexed i in an l-dimensional Euclidean space is:

$$\vec{x}_i = (X_{i,1}, X_{i,2}, \cdots, X_{i,l})^T. \tag{19}$$

Bibliography

[1] C. Arcelli and G. Sanniti di Baja, A width-independent fast thinning algorithm. *IEEE Trans. Pattern Anal. Machine Intell.*, 7(4):463–474, 1985.

[2] C. Arcelli and G. Sanniti di Baja, Ridge points in euclidean distance maps. *Pattern Recognition Letters*, 13:237–243, 1992.

[3] J. August, A. Tannenbaum, and S. W. Zucker, On the evolution of the skeleton. In *Proc. Int. Conf. Computer Vision*, pp. 315–322, 1999.

[4] J. August, K. Siddiqi, and S. W. Zucker, Ligature instabilities in the perceptual organization of shape. *Computer Vision and Image Understanding*, 76(3):231–243, 1999.

[5] H. G. Barrow and R. M. Burstall, Subgraph isomorphism, matching relational structures and maximal cliques. *Information Processing Letters*, 4:83–84, 1976.

[6] M. Bartoli, M. Pelillo, K. Siddiqi, and S. W. Zucker, Attributed tree homomorphism using association graphs. In *Proc. IEEE Int. Conf. on Pattern Recognition*, pp. 2133–2136, 2000.

[7] H. Blum, Biological shape and visual science (part I). *Journal of theoretical Biology*, 38:205–287, 1973.

[8] H. Blum and R. N. Nagel, Shape description using weighted symmetric axis features. *Pattern Recognition*, 10:167–180, 1978.

[9] I. M. Bomze, M. Pelillo, and V. Stix, Approximating the maximum weight clique using replicator dynamics. *IEEE Trans. on Neural Networks*, 11(6):1228–1241, 2000.

[10] G. Borgefors, G. Ramella, and G. Sanniti di Baja, Multi-scale skeletons via permanence ranking. In *Advances in Visual Form Analysis*, pp. 31–42. World Scientific, 1997.

[11] S. Bouix and K. Siddiqi, Divergence-based medial surfaces. In *European Conference on Computer Vision*, Vol 1, pp. 603–618. Springer, 2000. LNCS 1842.

[12] H. Bunke and G. Allermann, Inexact graph matching for structural pattern recognition. *Pattern Recognition Letters*, 1:245–253, 1983.

[13] H. Bunke, P. Foggia, C. Guidobaldi, and M. Vento, Graph clustering using the weighted minimum common supergraph. In *4th IAPR-TC15 Workshop on Graph-based Representations in Pattern Recognition*, Springer-Verlag Berlin, LNCS 2727, pp. 235–246, 2003.

[14] H. Bunke and A. Kandel, Mean and maximum common subgraph of two graphs. *Pattern Recognition Letters*, 21:163–168, 2000.

[15] C. Chatfield and A. J. Collins, *Introduction to multivariate analysis.* Chapman & Hall, 1980.

[16] J. Kittler, W. J. Christmas, and M. Petrou, Structural matching in computer vision using probabilistic relaxation. *IEEE Trans. Pattern Anal. Machine Intell.*, 17(8):749–764, 1995.

[17] T. F. Cootes, C. J. Taylor, and D. H. Cooper, Active shape models - their training and application. *Computer Vision and Image Understanding*, 61:38–59, 1995.

[18] T. F. Cox and M. A. A. Cox, *Multidimensional Scaling.* Chapman & Hall, 1994.

[19] J. Crank and P. Nicolson, A practical method for numerical evaluation of solutions of partial differential equations of the heat conduction type. In *Proc. Cambridge Philos. Soc.*, Vol. 43, pp. 50–67, 1947.

[20] A. D. J. Cross, R. C. Wilson, and E. R. Hancock. Inexact graph matching using genetic search. *Pattern Recognition*, 30(6):953–970, 1997.

[21] C. Cyr and B. Kimia, 3D Object Recognition Using Shape Similarity-Based Aspect Graph. *Proc. Int. Conf. Computer Vision*, pp. 254–261, 2001.

[22] S. J. Dickinson, A. P. Pentland, and A. Rosenfeld, 3-D shape recovery using distributed aspect matching, *IEEE Trans. Pattern Anal. Machine Intell.*, 14(2):174–198, 1992.

[23] P. Dimitrov, J. N. Damon, and K. Siddiqi, Flux Invariants for Shape. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, Vol. 1, pp. 835–841, 2003.

[24] R. Englert and R. Glantz, Towards the clustering of graphs. In *2nd IAPR-TC-15 Workshop on Graph-Based Representations*, pp. 125–134, 1999.

[25] M. A. Eshera and K.-S. Fu, An image understanding system using attributed symbolic representation and inexact graph-matching, *IEEE Trans. Pattern Anal. Machine Intell.*, 8:604–618, 1986.

[26] N. Friedman and D. Koller, Being Bayesian about Network Structure. *Machine Learning*, 50:95–126, 2003.

[27] L. Getoor, N. Friedman, D. Koller, and B. Taskar, Learning Probabilistic models of relational structure. *J. Machine Learning Research*, 3:679–707, 2002.

[28] L. E. Gibbons, D. W. Hearn, P. M. Pardalos, and M. V. Ramana, Continuous characterizations of the maximum clique problem. *Math. Oper. Res.*, 22:754–768, 1997.

[29] P. J. Giblin and B. B. Kimia, On the local form and transitions of symmetry sets, medial axes, and shocks. In *Proc. Int. Conf. Computer Vision*, pp. 385–391, 1999.

[30] S. Gold and A. Rangarajan, A graduated assignment algorithm for graph matching, *IEEE Trans. Pattern Anal. Machine Intell.*, 18:377–387, 1996.

[31] P. Golland and E. L. Grimson, Fixed topology skeletons. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, Vol. 1, pp. 10–17, 2000.

[32] J. C. Gower, Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, 23:325–328, 1964.

[33] T. Heap and D. Hogg, Wormholes in shape space: tracking through discontinuous changes in shape, In *Proc. Int. Conf. Computer Vision*, pp. 344–349, 1998.

[34] D. Heckerman, D. Geiger, and D. M. Chickering, Learning Bayesian networks: the combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995.

[35] G. R. Hjaltason and H. Samet, Properties of embedding methods for similarity searching in metric spaces. *IEEE Trans. Pattern Anal. Machine Intell.*, 25:530–549, 2003.

[36] T. Hofmann and M. Buhmann, Pairwise data clustering by deterministic annealing. *IEEE Trans. Pattern Anal. Machine Intell.*, 19(1):1–14, 1997.

[37] S. Ioffe and D. A. Forsyth, Human Tracking with Mixtures of Trees. In *Proc. Int. Conf. Computer Vision*, Vol. I, pp. 690–695, 2001.

[38] X. Jiang, A. Muenger, and H. Bunke, Computing the generalized mean of a set of graphs. In *Workshop on Graph-based Representations, GbR'99*, pp. 115–124, 2000.

[39] S. C. Johnson, Hierarchical clustering schemes. *Psychometrika*, Vol. 32(3), pp. 241–254, 1967.

[40] Y. Keselman, A. Shokoufandeh, M. F. Demirci, and S. Dickinson, Many-to-many graph matching via metric embedding. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, Vol. 1, pp. 850–857, 2003.

[41] B. B. Kimia, A. R. Tannenbaum, and S. W. Zucker, On the evolution of curves via a function of curvature, I: the classical case. *J. Mathematical Analysis Applications*, 163(2):438–458, 1992.

[42] B. B. Kimia, A. R. Tannenbaum, and S. W. Zucker, Shapes, shocks, and deforamtions I. *Int. J. Computer Vision*, 15:189–224, 1995.

[43] P. Klein, S. Tirthapura, D. Sharvit, and B. B. Kimia, A tree-edit-distance algorithm for comparing simple, closed shapes. In *ACM-SIAM Symp. on Discrete Algorithms*, pp. 696–704, 2000.

[44] P. Langley, W. Iba, and K. Thompson, An analysis of Bayesian classifiers. In *Proc. 10th Nat. Conf. on Artificial Intelligence*, AAAI Press, pp. 223–228, 1992.

[45] F. Leymarie and M. D. Levine, Simulating the grassfire transform using an active contour model. *IEEE Trans. Pattern Anal. Machine Intell.*14(1):56–75, 1992.

[46] N. Linial, E. London ,and Y. Rabinovich, The geometry of graphs and some of its applications. In Proc. 35th Annual Symp. on Foundations of Computer Science, pp. 169–175, 1994.

[47] T. Liu and D. Geiger, Approximate tree matching and shape similarity. In *Proc. Int. Conf. Computer Vision*, pp. 456–462, 1999.

[48] M. A. Lozano and F. Escolano, EM Algorithm for Clustering an Ensemble of Graphs with Comb Matching. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, Springer-Verlag Berlin, LNCS 2683, pp. 52–67, 2003.

[49] B.Luo, R.C.Wilson, and E.R.Hancock, Spectral Embedding of Graphs. *Pattern Recognition*, 36:2213–2233, 2003.

[50] D. Macrini, A. Shokoufandeh, S. Dickinson, K. Siddiqi, and S. W. Zucker, View-Based 3-D Object Recognition using Shock Graphs. In *Proc. IEEE Int. Conf. on Pattern Recognition*, pp. 24–28, 2002.

[51] M. Meilă, Learning with Mixtures of Trees. PhD thesis, MIT, 1999.

[52] B. Moayer and K.-S. Fu, A tree system approach for fingerprint pattern recognition. *IEEE Trans. Pattern Anal. Machine Intell.*, 8(3):376–387, 1986.

[53] F. Mokhtarian and A. K. Mackworth, A theory of multiscale, curvature based shape representation for planar curves. *IEEE Trans. Pattern Anal. Machine Intell.*, 14:789–805, 1992.

[54] T. S. Motzkin and E. G. Straus, Maxima for graphs and a new proof of a theorem of Turán. *Canadian Journal of Mathematics*, 17:533–540, 1965.

[55] A. Munger, H. Bunke, and X. Jiang, Combinatorial search vs. genetic algorithms: A case study based on the generalized median graph problem. *Pattern Recognition Letters*, 20(11-13):1271–1279, 1999.

[56] R. L. Ogniewicz, A multiscale mat from Voronoi diagrams: the skeleton-space and its application to shape description and decomposition. In *Aspects of Visual Form Processing*, pp. 430–439. World Scientific, 1994.

[57] R. L. Ogniewicz and O. Kübler, Hierarchic Voronoi skeletons. *Pattern Recognition*, 28(3):343–359, 1995.

[58] S. J. Osher and J. A. Sethian, Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations. *J. of Computational Physics*, 79:12–49, 1988.

[59] C. H. Papadimitriou and K. J. Steiglitz, *Combinatorial optimization: algorithms and complexity*. Prentice-Hall, Englewood Cliffs, NJ, 1982.

[60] M. Pelillo, Replicator equations, maximal cliques, and graph isomorphism. *Neural Computation*, 11:1935–1955, 1999.

[61] M. Pelillo, The dynamics of relaxation labeling process. *J. Math. Imaging Vision*, 7:309–323, 1997.

[62] M. Pelillo, K. Siddiqi, and S. W. Zucker, Matching hierarchical structures using association graphs. *IEEE Trans. Pattern Anal. Machine Intell.*, 21(11):1105–1120, 1999.

[63] S. W. Reyner, An analysis of a good algorithm for the subtree problem. *SIAM Journal on Computing*, 6:730–732, 1977.

[64] J. Rissanen, Stochastic complexity and modeling. *Annals of Statistics*, 14:1080–1100, 1986.

[65] S. Rizzi, Genetic operators for hierarchical graph clustering. *Pattern Recognition Letters*, 19:1293–1300, 1998.

[66] A. Robles-Kelly and E. R. Hancock, A maximum likelihood framework for iterative eigendecomposition. In *Proc. Int. Conf. Computer Vision*, Vol. I, pp. 654–661, 2001.

[67] H. Samet, Distance transform for images represented by quadtrees. *IEEE Trans. Pattern Anal. Machine Intell.*, 4(3):298–303, 1982.

[68] A. Sanfeliu and K.-S. Fu, A distance measure between attributed relational graphs for pattern recognition. *IEEE Transactions on Systems, Man and Cybernetics*, 13:353–362, 1983.

[69] S. Sarkar and K. L. Boyer, Quantitative measures of change based on feature organization: Eigenvalues and eigenvectors. *Computer Vision and Image Understanding*, 71(1):110–136, 1998.

[70] M. Schmitt, Some examples of algorithms analysis in computational geometry by means of mathematical morphological techniques. In *Geometry and Robotics*, 1989. LNCS 391.

[71] S. Sclaroff and A. P. Pentland, Modal matching for correspondence and recognition, *IEEE Trans. Pattern Anal. Machine Intell.*, 17:545–661, 1995.

[72] T. S. Sebastian, P. N. Klein, and B. B. Kimia, Recognition of shapes by editing shock graphs. In *Proc. Int. Conf. Computer Vision*, Vol. 1, pp. 755–762, 2001.

[73] T. S. Sebastian, P. N. Klein, and B. B. Kimia, Shock-based indexing into large shape databases. In *European Conference on Computer Vision*, Vol. 3, pp. 731–746, 2002.

[74] T. B. Sebastian, P. N. Klein, and B. B. Kimia, Recognition of shapes by editing their shock graphs. *IEEE Trans. Pattern Anal. Machine Intell.*, to appear, 2004.

[75] J. Segen, Learning graph models of shape. In *Proc. 5th Int. Conf. on Machine Learning*, pp. 29–25, 1988.

[76] K. Sengupta and K. L. Boyer, Organizing large structural modelbases. *IEEE Trans. Pattern Anal. Machine Intell.*, 17(4), 1995.

[77] K. Sengupta and K. L. Boyer, Modelbase partitioning using property matrix spectra. *Computer Vision and Image Understanding*, 70(2), 1998.

[78] D. Shaked and A. M. Bruckstein, Pruning medial axes. *Computer Vision and Image Understanding*, 69(2):156–169, 1998.

[79] L. G. Shapiro and R. M. Haralick, Relational models for scene analysis. *IEEE Trans. Pattern Anal. Machine Intell.*, 4:595–602, 82.

[80] D. Sharvit, J. Chan, H. Tek, and B. B. Kimia, Symmetry-based indexing of image database. *J. Visual Communication and Image Representation*, 9(4):366–380, 1998.

[81] A. Shokoufandeh, S. J. Dickinson, K. Siddiqi, and S. W. Zucker, Indexing using a spectral encoding of topological structure. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, Vol. 2, pp. 2491–2497, 1999.

[82] K. Siddiqi, S. Bouix, A. Tannenbaum, and S. W. Zucker, The Hamilton-Jacobi Skeleton. In *Proc. Int. Conf. Computer Vision*, Vol. 2, pp. 828–834, 1999.

[83] K. Siddiqi, S. Bouix, A. Tannenbaum, and S. W. Zucker, Hamilton-Jacobi Skeletons. *Int. J. Computer Vision*, 3:215–231, 2002.

[84] K. Siddiqi and B. B. Kimia, A shock grammar for recognition. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pp. 507–513, 1996.

[85] K. Siddiqi, A. Shokoufandeh, S. J. Dickinson, and S. W. Zucker, Shock graphs and shape matching. *Int. J. Computer Vision*, 35(1):13–32, 1999.

[86] K-C Tai, The tree-to-tree correction problem, *J. of the ACM*, 26:422–433, 1979.

[87] H. Tek and B. B. Kimia, Symmetry maps of free-form curve segments via wave propagation. In *Proc. Int. Conf. Computer Vision*, Vol. 1, pp. 362–369, 1999.

[88] S. Tirthapura, D. Sharvit, P. Klein, and B. B. Kimia, Indexing based on edit-distance matching of shape graphs. In *SPIE International Symposium on Voice, Video, and Data Communications*, pp. 25–36, 1998.

[89] W. S. Torgerson, Multidimensional scaling I: theory and method. *Psychometrika*, 17:401–419, 1952.

[90] W. H. Tsai and K.-S. Fu, Error-correcting isomorphism of attributed relational graphs for pattern analysis. *IEEE Transactions on Systems, Man and Cybernetics*, 9:757–768, 1979.

[91] J. Turner, Generalized matrix functions and the graph isomorphism problem. *SIAM Journal on Applied Mathematics*, 16(3):520–526, 1968.

[92] J. R. Ullmann, An algorithm for subgraph isomorphism. *J. of the Association for Computing Machinery*, 23(1):31–42, 1976.

[93] S. Umeyama, An eigendecomposition approach to weighted graph matching problems. *IEEE Trans. Pattern Anal. Machine Intell.*, 10(5):695–703, 1988.

[94] J. T. L. Wang, K. Zhang, and G. Chirn, The approximate graph matching problem. In *Proc. IEEE Int. Conf. on Pattern Recognition*, pp. 284–288, 1994.

[95] M. L. Williams, R. C. Wilson, and E. R. Hancock, Deterministic search for relational graph matching. *Pattern Recognition*, 32:1255–1271, 1999.

[96] R. C. Wilson and E. R. Hancock, Structural matching by discrete relaxation. *IEEE Trans. Pattern Anal. Machine Intell.*, 19(6):634–648, 1997.

[97] A. K. C. Wong and M. You, Entropy and distance of random graphs with application to structural pattern recognition. *IEEE Trans. Pattern Anal. Machine Intell.*, 7:599–609, 1985.

[98] K. Zhang, A constrained edit distance between unordered labeled trees. *Algorithmica*, 15:205–222, 1996.

[99] K. Zhang and D. Shasha, Simple fast algorithms for the editing distance between trees and related problems. *SIAM J. of Comp.*, 18:1245–1262, 1989.

[100] K. Zhang, R. Statman, and D. Shasha, On the editing distance between unorderes labeled trees. *Information Processing Letters*, 42:133–139, 1992.

[101] S. C. Zhu and A. L. Yuille, FORMS: a flexible object recognition and modelling system. *Int. J. Computer Vision*, 20(3):187–212, 1996.