Graph Matching: Theoretical Foundations, Algorithms, and Applications

Horst Bunke
Department of Computer Science
University of Bern, Neubrückstr. 10, CH-3012 Bern, Switzerland
Email: bunke@iam.unibe.ch

Abstract

Graphs are a powerful and versatile tool useful in various subfields of science and engineering. In many applications, for example, in pattern recognition and computer vision, it is required to measure the similarity of objects. When graphs are used for the representation of structured objects, then the problem of measuring object similarity turns into the problem of computing the similarity of graphs, which is also known as graph matching. In this paper, similarity measures on graphs and related algorithms will be reviewed. Applications of graph matching will be demonstrated giving examples from the fields of pattern recognition and computer vision. Also recent theoretical work showing various relations between different similarity measures will be discussed.

1 Introduction

Graphs are a general and powerful data structure for the representation of objects and concepts. In a graph representation, the nodes typically represent objects or parts of objects, while the edges describe relations between objects or object parts. Graphs have some interesting invariance properties. For instance, if a graph, which is drawn on paper, is translated, rotated, or transformed into its mirror image, it is still the same graph in the mathematical sense. These invariance properties, as well as the fact that graphs are well-suited to model objects in terms of parts and their relations, make them very attractive for various applications.

In applications such as pattern recognition and computer vision, object similarity is an important issue. Given a database of known objects and a query, the task is to retrieve one or several objects from the database that are similar to the query. If graphs are used for object representation this problem turns into determining the similarity of graphs, which is generally referred to as *graph matching*.

Standard concepts in graph matching include graph isomorphism, subgraph isomorphism, and maximum common subgraph. However, in real world applications we can't always expect a perfect match between the input and one of the graphs in the database. Therefore, what is needed is an algorithm for error-tolerant matching, or equivalently, a method that computes a measure of similarity between two given graphs. In this paper we review recent work in the area of graph matching. Basic concepts are introduced in Section 2. Then in Section 3 theoretical foundations of graph matching are presented. Various algorithms for graph matching are introduced in Section 4. Applications are described in Section 5, and a discussion and conclusions are given in Section 6.

2 Basic Concepts in Graph Matching

In this paper we consider directed and labeled graphs, which are sometimes synonymously referred to as (attributed) relational graphs, or relational structures. Such a graph consists of a finite number of nodes, or vertices, and a finite number of directed edges. A finite number of labels are usually associated to the nodes and edges. (Labels are also called attributes sometimes.) If we delete some nodes from a graph g, together with their incident edges, we obtain a *subgraph* $g' \subseteq g$. A graph *isomorphism* from a graph g to a graph g' is a bijective mapping from the nodes of g to the nodes of g' that preserves all labels and the structure of the edges. Similarly, a *subgraph isomorphism* from g to g' is an isomorphism from g to a subgraph of g'. Another important concept in graph matching is *maximum common subgraph*. A maximum common subgraph of two graphs, g and g', is a graph g'' that is a subgraph of both g and g' and has, among all possible subgraphs of g and g', the maximum number of nodes. Notice that the maximum common subgraph of two graphs is usually not unique.

Graph isomorphism is a useful concept to find out if two objects are the same, up to invariance properties inherent to the underlying graph representation. Similarly, subgraph isomorphism can be used to find out if one object is part of another object, or if one object is present in a group of objects. Maximum common subgraph can be used to measure the similarity of objects even if there exists no graph or subgraph isomorphism between the corresponding graphs. Clearly, the larger the maximum common subgraph of two graphs is, the greater is their similarity.

Real world objects are usually affected by noise such that the graph representation of identical objects may not exactly match. Therefore it is necessary to integrate some degree of error tolerance into the graph matching process. A powerful alternative to maximum common subgraph computation is *error-tolerant graph matching* using *graph edit distance*. In its most general form, a graph *edit operation* is either a deletion, insertion, or substitution (i.e. label change). Edit operations can be applied to nodes as well as to edges. The edit distance of two graphs, g and g', is defined as the shortest sequence of edit operations that transform g into g'. Obviously, the shorter this sequence is the more similar are the two graphs. Thus edit distance is suitable to measure the similarity of graphs. The sequence of edit operations that transform g into g' implies an error-correcting mapping from the nodes of g to the nodes of g'.

In practical applications, some edit operations may have more importance than others. Hence, very often *costs* are assigned to the individual edit operations. Typically the more likely an edit operation is to occur the smaller is its cost. An assignment of costs to the individual edit operations is often called a *cost function*. Given a set of edit operations together with their costs, graph edit distance computation in its most general form means to find a sequence of edit operations that transform, with minimum cost, one of the given graphs into the other.

Actually, graph isomorphism, subgraph isomorphism, and maximum common subgraph detection are all special instances of graph edit distance computation under special cost functions [7]. Also the well-known problem of weighted graph matching [2, 50] can be regarded a special case of graph edit distance. Algorithms for graph matching, including graph edit distance computation, will be discussed in Section 4 of this paper. For a more formal treatment of the concepts introduced in this section see [5].

3 Theoretical Foundations

Relationships between error-tolerant graph matching using graph edit distance and the well-known concept of maximum common subgraph were studied recently [4]. The main result of this paper is that, for a particular class of cost functions, maximum common subgraph and graph edit distance computation are equivalent to each other. In particular, the maximum common subgraph g'' of two graphs, g and g', and their edit distance, $d(g,g')$, are related with each other through the simple equation

$$d(g,g') = |g| + |g'| - 2|g''| \qquad (1)$$

where $|g|$,$|g'|$ and $|g''|$ denote the number of nodes of g, g' and g'', respectively.
Hence any algorithm for maximum common subgraph computation can be used for graph edit distance computation and vice versa, as long as the cost function satisfies the conditions stated in [4].

In close relation with this result, a new graph similarity measure, $\delta(g_1, g_2)$, based on the maximum common subgraph was proposed in [6]:

$$\delta(g,g') = 1 - \frac{|mcs(g,g')|}{max(|g|,|g'|)} \qquad (2)$$

In this equation $mcs(g,g')$ denotes the maximum common subgraph of g and g' and $|g|$ stands for the number of nodes of g, similarly to eq.(1). This similarity measure is a metric. Thus it may be useful for applications where properties such as reflexivity or triangular inequality are desired.

An in-depth study of the influence of the underlying cost function on graph edit distance computation was presented in [7]. The main result of this study is that, for any cost function, there exist infinitely many other, equivalent cost functions that lead to the same optimal sequence of edit operations for transforming two given graphs into each other. Moreover, given the edit distance $d(g,g')$ under one particular cost function, the edit distance $d'(g,g')$ under any other cost function from the same equivalence class is just a linear function of $d(g,g')$. From the practical point of view, this result tells us that any particular graph matching algorithm designed for a special cost function can be used for infinitely many other cost functions as well, i.e., all other cost functions from the same equivalence class.

A novel concept, the minimum common supergraph of two graphs, was recently introduced [8]. A supergraph g of two graphs, g' and g'', is a graph that contains both g' and g'' as subgraphs. The minimum common supergraph of g' and g'' is a graph that is a supergraph of both g' and g'' and has, among all those supergraphs, the minimum number of nodes. It has been shown that the computation of the minimum common supergraph can be solved through computation of the maximum common subgraph. Similarly to eq. (1) there is a relation between the minimum common supergraph of two graphs and their edit distance [8]. While maximum common subgraph can be regarded a kind of intersection operator on graphs, minimum common supergraph can be interpreted as graph union. This observation may be an interesting starting point for investigating graph operators with algebraic properties.

4 Graph Matching Algorithms

All results presented in the previous section of this paper are independent of the algorithm that is actually used for graph matching. A wide spectrum of graph matching algorithms with different characteristics have become available meanwhile. The standard algorithm for graph and subgraph isomorphism detection is the one by Ullman [49]. Maximum common subgraph detection has been addressed in [17, 23, 34]. Classical methods for error-tolerant graph matching can be found in [14, 42, 43, 48, 55]. Most of these algorithms are particular versions of the A* search procedure, i.e., they rely on some kind of tree search incorporating various heuristic lookahead techniques in order to prune the search space.

These methods are guaranteed to find the optimal solution but require exponential time and space due to the NP-completeness of the problem. Suboptimal, or approximative methods, on the other hand, are polynomially bounded in the number of computation steps but may fail to find the optimal solution. For example, in [10, 54] probabilistic relaxation schemes are described. Other approaches are based on neural networks such as the Hopfield network [15] or the Kohonen map [57]. Also genetic algorithms have been proposed recently [12, 52]. In [51] an approximate method based on maximum flow is introduced. However, all of these approximate methods may get tracked in local minima and miss the optimal solution. Approaches to the weighted graph matching problem using Eigenvalues and linear programming, have been proposed in [50] and [2], respectively. As a special case, the matching of trees has been addressed in a series of papers recently [9, 33, 35, 53].

In the remainder of this section we briefly review three optimal graph matching methods that were proposed recently. In [27, 29] a new method is described for matching a graph g against a database of model graphs $g_1, \ldots, g_n$ in order to find the model g_i with the smallest edit distance $d(g, g_i)$ to g. The basic assumption is that the models in the database are not completely dissimilar. Instead, it is supposed that there are graphs $s_j's$ that occur simultaneously as subgraphs in several of the $g_i's$, or multiple times in the same g_i. Under a naive procedure, we will match g sequentially with each of the $g_i's$. However, because of common subgraphs s_j shared by several models g_i, the $s_j's$ will be matched with g multiple times. This clearly implies some redundancy.

In the approach described in [27, 29] the model graphs $g_1, \ldots, g_n$ are preprocessed generating a symbolic data structure, called network of models. This network is a compact representation of the models in the sense that multiple occurrences of the same subgraph s_j are represented only once. Consequently, such subgraphs will be matched only once with the input. Hence the computational effort will be reduced. A further enhancement of the computational efficiency of the method is achieved by a lookahead procedure. This lookahead procedure returns an estimation of the future matching cost. It is precise and can be efficiently computed based on the network. In [27, 32] the same procedure is applied not to graph edit distance computation, but subgraph and graph isomorphism detection.

In [27, 31] an even faster algorithm for graph and subgraph isomorphism detection was described. It is based on an intensive preprocessing step in which a database of model graphs is converted into a decision tree. At run time, the input graph is classified by the decision tree and all model graphs for which there exists a subgraph isomorphism from the input are detected. If we neglect the time needed for preprocessing, the computational complexity of the new subgraph isomorphism algorithm is only quadratic in the number of input graph vertices. In particular, it is independent of the number of model graphs and the number of edges in any of the graphs. However, the decision tree that is constructed in the preprocessing step is of exponential size in terms of the number of vertices of the model graphs. The actual implementation described by the authors is able to cope with a single graph in the database of up to 22 nodes, or up to 30 models in the database consisting of up to 11 nodes each.

Recently the decision tree method was extended from exact graph and subgraph isomorphism detection to error-tolerant graph matching [30]. Actually, there are different possible approaches. In one approach, error correction is considered at the time of the creation of the decision tree. That is, for each model graph a set of distorted copies are created and compiled into the decision tree. The number of distorted copies depends on the maximal admissible error. At run time, the decision tree is used to classify the unknown input graph in the same way as in case of exact subgraph isomorphism detection. The time complexity of this procedure at run time is only quadratic in the number of input graph nodes. However, the size of the decision tree is exponential in the number of vertices of the model graphs and in the degree of distortion that is to be considered. Therefore, this approach is limited to (very) small graphs.

In the second approach, the error corrections are considered at run time only. That is, the decision tree for a set of model graphs does not incorporate any information about possible errors. Hence, the decision tree compilation step is identical to the original preprocessing step and, consequently, the size of the decision tree is exponential only in the size of the model graphs. At run time, a set of distorted copies of the input graph are constructed such that all possible error corrections up to a certain error threshold are considered. Each graph in this set is then classified by the decision tree. The run time complexity of this method is $O(\vartheta n^{2(\vartheta+1)})$ where n is the number of nodes in the input graph and ϑ is a threshold that defines the maximum number of admissible edit operations.

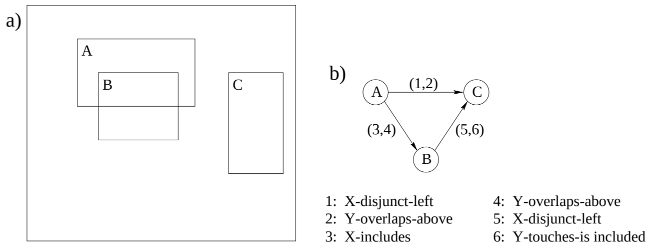

Figure 1:
a) an image where the objects are represented through their bounding boxes
b) graph representation of a)

5 Applications

A large number of applications of graph matching have been described in the literature. One of the earliest was in the field of chemical structure analysis [40]. More recently, graph matching has been applied to case-based reasoning [3, 36], machine learning [11, 16, 28], planning [41], semantic networks [13], conceptual graph [26], and monitoring of computer networks [47]. Furthermore it was used in the context of visual languages and programming by graph transformations [37, 39]. Numerous applications from the areas of pattern recognition and machine vision have been reported. They include recognition of graphical symbols [21, 22], character recognition [25, 38], shape analysis [9, 24, 35], three-dimensional object recognition [56], and others.

In the rest of this section we briefly sketch an application of graph matching to image and video indexing [44, 45]. The system under consideration is based on indexing by qualitative spatial relationships. For this purpose, the relational calculus proposed in [1] has been extended into two dimensions. Any object of interest in an image is represented by its bounding box, which is described, in turn, by a node in the underlying graph representation. The spatial relations between two objects are left-of, touches, overlaps, includes a.s.o. There are 13 relations in both the *x*- and *y*-direction, resulting in a total of 169 possible relations between two different objects in an image. Each graph representing an image is fully connected, i.e., there is an edge between any pair of nodes. An example of this kind of graph representation is shown in Fig. 1.

The transformation of the images in the database into their graph representation is accomplished in a semi-automatic fashion, where only the first frame of a video clip needs full manual processing. Once all objects of interest have been manually extracted and labeled in the first image, an automatic tracking procedure is started, which is based on the assumption that objects change only slightly from one image to the next. Retrieval of images from the database is by pictorial example. Given a query image, the user interactively defines the bounding boxes of the objects of interest and labels them on the screen. This information can be easily converted into the corresponding graph representation.

Given the graph representation of the query and the images in the database, the task of image retrieval is cast as a graph matching problem. Various matching paradigms, including maximum common subgraph detection, have been implemented. In the context of the considered application, the maximum common subgraph between the query Q and an image I in the database is particularly interesting as it represents the largest collection of objects present in Q and I that have compatible labels and maintain the same spatial relations to each other in both images.

Standard algorithms for maximum common subgraph detection are based on maximal cliques [23, 34] and tree search [17]. In the system under consideration, an extension of the decision tree based subgraph isomorphism detection algorithm proposed in [31] was adopted. This algorithm converts, in an off-line phase, the image database into a decision tree. Given a query graph, the time needed to traverse the decision tree is O($2^n n^3$), where n is the number of nodes in the query graph. (Notice that the time complexity is independent of the size of the database.) Obviously, the complexity of this procedure is significantly higher than O(n^2), which is needed for subgraph isomorphism detection [31], indicating that maximum common subgraph detection is a task more complicated than subgraph isomorphism detection. Nevertheless, the O($2^n n^3$) complexity favourably compares with O($L(nm)^n$), which is needed by the method described in [23] (where L is the number of graphs in the database and m is the number of nodes of a graph in the database). A potential drawback of the proposed algorithm for maximum common subgraph detection is the space complexity, which is exponential in the size of the database. But there are pruning strategies available for cutting down the space requirements [45].

The proposed graph matching procedures have been tested on a real video database [44]. The clips in this database vary in length from 4 to 20 seconds, and contain between 12 and 19 objects each. The shortest clip contains 71 changes to object relationships, while the longest has 402 changes. Table 1 shows the time (in milliseconds) required for the maximum common subgraph decision tree algorithm to search a database of 10 clips with a total of 5956 images. For the purpose of comparison, not only the time needed by the new decision tree procedure, but also the time required by Ullman's algorithm [49], and an A* procedure for subgraph isomorphism detection is recorded. The numbers given in the table are values averaged over several queries containing between 4 and 11 nodes each. From Table 1, the high execution speed of the new decision tree based maximum common subgraph procedure becomes evident. On the other hand we must remember the large space requirements of this method. Nevertheless, the method seems applicable to real world problems. For more details and further experi-

Algorithm	Mean	Minimum	Maximum	σ
Ullman	393.2	252	607	113.1
A*-based	617.1	362	861	178.2
Decision tree based MCS	16.6	6	23	6.5

Table 1: Performance evaluation of different graph matching algorithms

ments the reader is refereed to [45].

An extension of the decision tree based subgraph matching procedure to the case where the query consists of a whole sequence of images is described in [46].

6 Discussion and Conclusions

In this paper we have reviewed recent developments in graph matching. It can be concluded that graphs are a versatile and flexible representation formalism suitable for a wide range of problems in intelligent information processing, including the areas of pattern recognition and computer vision. A wide spectrum of graph matching algorithms have become available meanwhile. They range from deterministic approaches, suitable for finding optimal solutions to problems involving graphs with a limited number of nodes and edges, to approximate methods that are applicable to large-scale problems.

The graph matching algorithms reviewed in this paper are very general. In fact, there are no problem dependent assumptions included. The nodes and edges of a graph may represent anything, and there are no restrictions on the node and edge labels. The distortion model used in graph edit distance computation includes the deletion, insertion, and substitution of both nodes and edges. Hence it is powerful enough to model any type of error that may be introduced to a graph.

Adapting a graph matching algorithm to a particular task requires the solution of two concrete problems. First, a suitable graph representation of the objects of the problem domain has to be found. Secondly, appropriate error correction, i.e. edit operations together with their costs, have to be defined. For the solution of both problems, domain specific knowledge must be utilized whenever it is meaningful.

There are a number of open problems in graph matching that deserve further research. It is conjectured that there are many applications in pattern recognition and computer vision where the full representational power of graphs may not be needed. Restricting the focus on special subclasses of graphs may result in more efficient matching procedures. For example, restricted classes of graphs, where the isomorphism can be solved in polynomial time, have been reported in [58]; see also the references in this paper. Additional classes of graphs have been discovered recently. In [18, 20] so-called ordered graphs have been investigated. It was shown that the isomorphism problem for ordered graphs can be solved in O(nm) time, where n and m represent the number of edges of the two graphs. A special form of subgraph isomorphism for these graphs has been considered in [19]. Under the assumption that the degree of some distinguished vertices is preserved under the subgraph isomorphism mapping, it was shown that the subgraph isomorphism problem is solvable in quadratic time as well. This clearly demonstrates that restricting the focus on special subclasses of graphs may lead to more efficient matching procedures. Most of the works referenced here were motivated by graph theoretical considerations. In future work it will be interesting to search for other special classes of graphs with a lower matching complexity from a more application oriented point of view, paying particular attention to classes of graphs that are relevant to pattern recognition and computer vision.

Other promising areas of future research include the automatic inference of edit costs from a set of sample graphs, and the combination of optimal and approximate graph matching methods.

Acknowledgement: The author wants to thank Dr. X. Jiang for continuous collaboration and intensive exchange of ideas.

References

[1] Allen, J.F. (1990). “Maintaining knowledge about temporal intervals”, JACM 26, 1983, 832-843.

[2] Almohamed, H. (1993). “A linear programming approach for the weighted graph matching problem”, IEEE Trans. PAMI 15, 522-525.

[3] Börner, K., Pippig, E., Tammer, E., and Coulon, C. (1996). “Structural similarity and adaption”, in I.Smith and B.Faltings (eds.): Advances in Case-based Reasoning, LNCS 1168, Springer, pp. 58–75.

[4] Bunke, H. (1997). “On a relation between graph edit distance and maximum common subgraph”, Pattern Recognition Letters, Vol. 18, pp. 689–694.

[5] Bunke, H. (1998). “Error-tolerant graph matching: a formal framework and algorithms”, in A.Amin, D.Dori, P.Pudil, and H.Freeman (eds.): Advances in Pattern Recognition, LNCS 1451, Springer Verlag, pp. 1–14.

[6] Bunke, H. and Shearer, K. (1998). “A graph distance metric based on maximal common subgraph”, Pattern Recognition Letters, Vol. 19, Nos. 3-4, pp. 255–259.

[7] Bunke, H. (1999), Error correcting graph matching: On the influence of the underlying cost function, IEEE Trans. PAMI 21: 917–922.

[8] Bunke,H. , Jiang,X. and Kandel,A. (2000). “On the minimum common subgraph of two graphs”, to appear in Computing.

[9] Cantoni, V. et al. (1998). “2-D object recognition by multiscale tree matching”, Pattern Recognition 31, 1443-1455.

[10] Christmas, W.J., Kittler, J., and Petrou, M. (1995). “Structural matching in computer vision using probabilistic relaxation”, IEEE Trans. PAMI 8, pp. 749–764.

[11] Cook, D.J. and Holder, L.B. (1994). “Substructure discovery using minimum description length and background knowledge”, Journal of Artificial Intelligence Research, pp. 231–255.

[12] Cross, A., Wilson, R., and Hancock, E. (1996). “ Genetic search for structural matching”, In B. Buxton, R. Cipolla (eds.): Computer Vision - ECCV ’96, LNCS 1064, Springer Verlag, pp. 514–525.

[13] Ehrig, H. (1992). “Introduction to graph grammars with applications to semantic networks”, Computers and Mathematics with Applications Vol. 23, pp. 557–572.

[14] Eshera, M.A. and Fu, K.S. (1984). “A graph distance measure for image analysis”, IEEE Trans. SMC 14, pp. 398–408.

[15] Feng, J., Laumy, M., and Dhome, M. (1994). “Inexact matching using neural networks”, In E.S. Gelsema and L.N. Kanal (eds): Pattern Recognition in Practice IV: Multiple Paradigms, Comparative Studies and Hybrid Systems, pp. 177–184. North-Holland.

[16] Fisher, D.H. (1990). “Knowledge acquisition via incremental conceptual clustering”, in J.W.Shavlik and T.G.Dietterich (eds.): Readings in Machine Learning, pp. 267–283, Morgan Kaufmann.

[17] McGregor, J. (1982). “Backtrack search algorithms and the maximal common subgraph problem”, Software-Practice and Experience, Vol. 12, pp. 23–34.

[18] Jiang, X. and Bunke, H. (1998a). “On the coding of ordered graphs”, Computing, Vol. 61, No. 1, pp. 23–38.

[19] Jiang, X. and Bunke, H. (1998b). “Marked subgraph isomorphism of ordered graphs”, in A.Amin, D.Dori, P.Pudil, and H.Freeman (eds.): Advances in Pattern Recognition, LNCS 1451, Springer Verlag, pp. 122–131.

[20] Jiang, X. and Bunke, H. (1999a). “Optimal quadratic-time isomorphism of ordered graphs”, Pattern Recognition 32, pp. 1273–1283.

[21] Jiang, X., Münger, A., and Bunke, H. (1999c). “Synthesis of representative graphical symbols by generalized median graph computation”, Proc. of 3rd IAPR Workshop on Graphics Recognition, Jaipur.

[22] Lee, S.W., Kim, J.H., and Groen, F.C.A. (1990). “Translation- rotation- and scale invariant recognition of hand-drawn symbols in schematic diagrams”, Int. Journal of Pattern Recognition and Artificial Intelligence, Vol. 4, pp 1–15.

[23] Levi, G. (1972). “A note on the derivation of maximal common subgraphs of two directed or undirected graphs”, Calcolo, Vol. 9, pp. 341–354.

[24] Lourens, T. (1998). “A biologically plausible model for corner-based object recognition from color images”, PhD thesis, University of Groningen, The Netherlands.

[25] Lu, S.W., Ren, Y., and Suen, C.Y. (1991). “Hierarchical attributed graph representation and recognition of handwritten Chinese characters”, Pattern Recognition 24, pp. 617–632.

[26] Maher, P. (1993). “A similarity measure for conceptual graphs”, Int. Journal of Intelligent Systems, Vol. 8, pp. 819–837.

[27] Messmer, B.T. (1995). “Efficient graph matching algorithms for preprocessed model graphs”, PhD thesis, University of Bern, Switzerland.

[28] Messmer, B.T. and Bunke, H. (1996). “Automatic learning and recognition of graphical symboles in engineering drawings”, in K.Tombre and R.Kasturi (eds): Graphics Recognition, Lecture Notes in Computer Science 1072, pp. 123-134, Springer Verlag.

[29] Messmer, B.T. and Bunke, H. (1998a). “A new algorithm for error tolerant subgraph isomorphism”, IEEE Trans. PAMI 20, pp. 493–505.

[30] Messmer, B.T. and Bunke, H. (1998b). “Error-correcting graph isomorphism using decision trees”, Int. Journal of Pattern Recognition and Art. Intelligence, Vol. 12, pp. 721–742.

[31] Messmer, B.T. and Bunke, H. (1999a). “A decision tree approach to graph and subgraph isomorphism detection”, Pattern Recognition 32, 1999, pp. 1979-1998.

[32] Messmer, B.T. and Bunke, H. (2000). “Efficient subgraph isomorphism detection - a decompostion approach”, to appear in IEEE Trans. on Data and Knowledge Engineering.

[33] Oflazer, K. , (1997). “Error-tolerant retrieval of trees”, IEEE Trans. PAMI 19, 1997, 1376-1380.

[34] Pelillo, M. (1998). “A unifying framework for relational structure matching”, Proc. 14th ICPR, Brisbane, 1998.

[35] Pelillo, M., Siddiqi, K. and Zucker, S. (1999). “Matching hierarchical structures using associated graphs”, IEEE Trans. PAMI 21, 1999, 1105-1120.

[36] Poole, J. (1993). “Similarity in legal case based reasoning as degree of matching in conceptual graphs”, in MM.Richter, S.Wess, K.-D.Althoff, and F.Maurer (eds.): Preproceedings: First European Workshop on Case-Based Reasoning, pp. 54–58.

[37] Rekers, J. and Schürr, A. (1997). “Defining and parsing visual languages with layered graph grammars”, Journal of Visual Languages and Computing, Vol. 8, pp. 27–55.

[38] Rocha, J. and Pavlidis, T. (1994). “A shape analysis model with applications to a character recognition system”, IEEE Trans. PAMI, pp. 393–404.

[39] Rodgers, P.J. and King, P.J.H. (1997). “A graph - rewriting visual language for database programming”, Journal of Visual Languages and Computing, Vol. 8, pp. 641–674.

[40] Rouvray, D.H. and Balaban, A.T. (1979). “Chemical applications of graph theory”, in R.J.Wilson and L.W.Beineke (eds.): Applications of Graph Theory, pp. 177–221, Academic Press.

[41] Sanders, K., Kettler, B., and Hendler, J. (1997). “The case for graph-structured representations”, in D.Leake and E.Plaza (eds.): Case-Based Reasoning Research and Development, Lecture Notes in Computer Science, Vol. 1266, Springer, pp. 245–254.

[42] Sanfeliu, A. and Fu, K.S. (1983). “A distance measure between attributed relational graphs for pattern recognition”, IEEE Trans. SMC, Vol. 13, pp. 353–363.

[43] Shapiro, L.G. and Haralick, R.M. (1981). “Structural descriptions and inexact matching”, IEEE Trans. PAMI, Vol. 3, pp. 504–519.

[44] Shearer, K.R. (1998). “Indexing and retrieval of video using spatial reasoning techniques”, PhD thesis, Curtin University of Technology, Perth, Australia.

[45] Shearer, K., Bunke, H. and Venkatesh, S. (2000). “Video indexing and similarity retrieval by largest common subgraph detection using decision trees”, to appear in Pattern Recognition.

[46] Shearer, K., Bunke, H. and Venkatesh, S. (2000). “Video sequence matching via decision tree path following”, submitted.

[47] Shoubridge, P., Krarne, M., and Ray, D. (1999). “Detection of abnormal change in dynamic networks”, Proc. of IDC’99, Adelaide, pp. 557–562.

[48] Tsai, W.H. and Fu, K.S. (1979). “Error-correcting isomorphisms of attributed relational graphs for pattern recognition”, IEEE Trans. SMC 9, pp. 757–768.

[49] Ullman, J.R. (1976). “An algorithm for subgraph isomorphism”, Journal of the Association for Computing Machinery, Vol. 23, No. 1, pp. 31–42.

[50] Umeyama, S. (1988). “An Eigendecomposition approach to weighted graph matching problems”, IEEE Trans. PAMI 10, 695–703.

[51] Wang, I., Zhang, K., and Chirn, G. (1994). “The approximate graph matching problem”, Proc. of 12th ICPR, pp. 284–288, Jerusalem.

[52] Wang, Y.-K., Fan, K.-C., and Horng, J.-T. (1997). “Genetic-based search for error-correcting graph isomorphism”, IEEE Trans. SMC 27, No. 4, pp. 588–597.

[53] Wang, J. et al. (1998). “An algorithm for finding the largest approximately commen substructure of two trees”, IEEE Trans. PAMI 20, 1998, 889-895.

[54] Wilson, R. and Hancock, E. (1994). “Graph matching by discrete relaxation”, In E.S. Gelsema and L.N. Kanal (eds): Pattern Recognition in Practice IV: Multiple Paradigms, Comparative Studies and Hybrid Systems, pp. 165–176. North-Holland.

[55] Wong, E.K. (1990). “Three-dimensional object recognition by attributed graphs”, In H.Bunke and A.Sanfeliu (eds.): Syntactic and Structural Pattern Recognition-Theory and Applications, pp. 381–414. World Scientific.

[56] Wong, E.K. (1992). “Model matching in robot vision by subgraph isomorphism”, Pattern Recognition, Vol. 25, pp. 287–304.

[57] Xu, L. and Oja, E. (1990). “Improved simulated annealing, Boltzmann machine, and attributed graph matching”, In L.Almeida (ed): LNCS 412, pp. 151–161. Springer Verlag.

[58] Yamazaki, K. et al. (1999). “Isomorphism for graphs with bounded distance width”, Algorithmica 24, 1999, 103-127.