ELSEVIER

Pattern Recognition Letters 18 (1997) 689–694

Pattern Recognition Letters

# On a relation between graph edit distance and maximum common subgraph

H. Bunke [1]

*Institut für Informatik und angewandte Mathematik, University of Bern, Neubrückstr. 10, CH-3012 Bern, Switzerland*

Received 14 April 1997; revised 8 July 1997

**Abstract**

In approximate, or error-correcting, graph matching one considers a set of graph edit operations, and defines the edit distance of two graphs $g_1$ and $g_2$ as the shortest (or least cost) sequence of edit operations that transform $g_1$ into $g_2$. A maximum common subgraph of two graphs $g_1$ and $g_2$ is a subgraph of both $g_1$ and $g_2$ such that there is no other subgraph of $g_1$ and $g_2$ with more nodes. Graph edit distance and maximum common subgraph are well known concepts that have various applications in pattern recognition and machine vision. In this paper a particular cost function for graph edit distance is introduced, and it is shown that under this cost function graph edit distance computation is equivalent to the maximum common subgraph problem. © 1997 Elsevier Science B.V.

*Keywords:* Approximate graph matching; Graph edit distance; Maximum common subgraph; Edit operation; Cost function

## 1. Introduction

Graphs are general and powerful data structures useful for the representation of various objects and concepts. In pattern recognition and machine vision, for example, graphs are often used to represent object models, which are known a priori and stored in a database, and also unknown objects, which are to be recognized. Using graphs as a representation formalism, the recognition problem turns into a graph matching problem. An input graph representing an unknown object is compared to the database in order to find the most similar model graph. Applications where this concept has been successfully applied include Chinese character recognition (Lu et al., 1991), schematic diagram interpretation (Lee et al., 1990), seal verification (Lee and Kim, 1989), shape analysis (Pearce et al., 1994), image recognition (Christmas et al., 1995), and 3-D object recognition (Cho and Kim, 1992; Wong, 1992).

Algorithms for graph matching include graph and subgraph isomorphism (Read and Corneil, 1977; Ullman, 1976). However, due to errors and distortions in the input data, approximate, or error-correcting, graph matching methods are needed in most applications (Shapiro and Haralick, 1981; Bunke and Allerman, 1983; Sanfeliu and Fu, 1983). Another way to cope with distorted input graphs is to use the maximum common subgraph in order

[1] E-mail: bunke@iam.unibe.ch.

0167-8655/97/$12.00 © 1997 Elsevier Science B.V. All rights reserved.
*PII* S0167-8655(97)00060-3

to measure graph similarity (Levinson, 1992; Horaud and Skordas, 1989). Subgraph isomorphism, error-correcting graph isomorphism and maximum common subgraph computation are NP-complete problems. Nevertheless, in many applications constraints and heuristics can be found that cut down the computational effort to a manageable size.

In this paper we first formally define error-correcting graph matching and graph edit distance. Then we introduce a special cost function for error-correcting graph matching and show that under this cost function the graph edit distance problem is equivalent to maximum common subgraph computation. A similar relation is known to hold between the string edit distance and the length of the longest common subsequence of two strings (Wagner and Fischer, 1974). The main contribution of the paper is the formal proof of the equivalence. But there are potential practical consequences of this result in the sense that any known algorithm for graph edit distance becomes applicable for computing maximum common subgraphs, and vice versa.

## 2. Graph edit distance and maximum common subgraph

Let $L$ be a finite alphabet of labels for nodes and edges.

**Definition 1.** A *graph* is a triple $g=(V,\alpha,\beta)$ where
- $V$ is the finite set of *vertices*,
- $\alpha: V \to L$ is the node labeling function,
- $\beta: V \times V \to L$ is the edge labeling function.

The set of *edges* $E$ is implicitly given by assuming that the graphs are fully connected, i.e., $E = V \times V$. In other words, there exists exactly one edge between any pair of nodes. This assumption is for notational convenience only and does not restrict generality. If it is necessary to model the situation where edges exist only between distinguished pairs of nodes, we can easily include a special *null* label in the set of labels, $L$. Edges are directed, i.e., edge $(x,y)$ originates at node $x \in V$ and terminates at node $y \in V$. Node and edge labels come from the same alphabet, for notational convenience. If node and edge labels need to be explicitly distinguished, the set $L$ can be partitioned into two disjoint subsets. If $V=\emptyset$ then $g$ is called the *empty graph.*

**Definition 2.** Let $g=(V,\alpha,\beta)$ and $g'=(V',\alpha',\beta')$ be two graphs; $g'$ is a *subgraph* of $g$, $g' \subseteq g$, if
- $V' \subseteq V$,
- $\alpha'(x)=\alpha(x)$ for all $x \in V'$,
- $\beta'((x,y))=\beta((x,y))$ for all $(x,y) \in V' \times V'$.

From Definition 2 it follows that, given a graph $g=(V,\alpha,\beta)$, any subset $V' \subseteq V$ of its vertices uniquely defines a subgraph. This subgraph is called the subgraph that is *induced* by $V'$.

**Definition 3.** Let $g_1=(V_1,\alpha_1,\beta_1)$ and $g_2=(V_2,\alpha_2,\beta_2)$ be two graphs. A *graph isomorphism* between $g_1$ and $g_2$ is a bijective mapping $f: V_1 \to V_2$ such that
- $\alpha_1(x)=\alpha_2(f(x))$ for all $x \in V_1$,
- $\beta_1((x,y))=\beta_2((f(x),f(y)))$ for all $(x,y) \in V_1 \times V_1$.

If $V_1=V_2=\emptyset$, then $f$ is called the *empty graph isomorphism.*

**Definition 4.** Let $g_1=(V_1,\alpha_1,\beta_1)$ and $g_2=(V_2,\alpha_2,\beta_2)$ be two graphs and $g_1' \subseteq g_1$, $g_2' \subseteq g_2$. If there exists a graph isomorphism between $g_1'$ and $g_2'$, then both $g_1'$ and $g_2'$ are called a *common subgraph* of $g_1$ and $g_2$.

**Definition 5.** Let $g_1$ and $g_2$ be two graphs. A graph $g$ is called a *maximum common subgraph* of $g_1$ and $g_2$ if

$g$ is a common subgraph of $g_1$ and $g_2$ and there exists no other common subgraph of $g_1$ and $g_2$ that has more nodes than $g$.

**Definition 6.** Let $g_1=(V_1,\alpha_1,\beta_1)$ and $g_2=(V_2,\alpha_2,\beta_2)$ be two graphs. An *error-correcting graph matching* (*ecgm*) from $g_1$ to $g_2$ is a bijective function $f:\hat{V}_1 \to \hat{V}_2$, where $\hat{V}_1 \subseteq V_1$ and $\hat{V}_2 \subseteq V_2$.

We say that node $x \in \hat{V}_1$ is *substituted* by node $y \in \hat{V}_2$ if $f(x)=y$. If $\alpha_1(x)=\alpha_2(f(x))$ then the substitution is called an *identical* substitution. Furthermore, any node from $V_1-\hat{V}_1$ is *deleted* from $g_1$, and any node from $V_2-\hat{V}_2$ *inserted* in $g_2$ under $f$. We will use $\hat{g}_1$ and $\hat{g}_2$ to denote the subgraphs of $g_1$ and $g_2$ that are induced by the sets $\hat{V}_1$ and $\hat{V}_2$, respectively.

The mapping $f$ *directly* implies an edit operation on each node in $g_1$ and $g_2$. I.e., nodes are substituted, deleted, or inserted, as described above. Additionally, the mapping $f$ *indirectly* implies edit operations on the edges of $g_1$ and $g_2$. If $f(x_1)=y_1$ and $f(x_2)=y_2$, then edge $(x_1,x_2)$ will be substituted by edge $(y_1,y_2)$. If a node $x$ is deleted from $g_1$, than any edge incident to $x$ is deleted, too. Similarly, if a node $x'$ is inserted in $g_2$, then any edge incident to $x'$ is inserted, too. Obviously, any *ecgm* $f$ can be understood as a set of edit operations (substitutions, deletions, and insertions of both nodes and edges) that transform a given graph $g_1$ into another graph $g_2$.

**Example 1.** A graphical representation of two graphs is given in Fig. 1. For those graphs, we have:
- $V_1=\{1,2,3\}$; $V_2=\{4,5,6\}$; $L=\{X,Y,Z,a,b,c,null\}$.
- $\alpha_1$: $1 \mapsto X$, $2 \mapsto X$, $3 \mapsto Y$.
- $\alpha_2$: $4 \mapsto X$, $5 \mapsto X$, $6 \mapsto Z$.
- $\beta_1$: $(1,2) \mapsto a$, $(1,3) \mapsto b$, $(2,3) \mapsto b$.
- $\beta_2$: $(4,5) \mapsto a$, $(4,6) \mapsto c$, $(5,6) \mapsto c$.

All other edges, including self-loops at vertices, are labeled with *null* and are not shown in Fig. 1.

A possible *ecgm* is $f:1 \mapsto 4$, $2 \mapsto 5$ with $\hat{V}_1=\{1,2\}$ and $\hat{V}_2=\{4,5\}$. Under this *ecgm*, nodes 1 and 2 are substituted by 4 and 5, respectively. Consequently, edge (1,2) is substituted by edge (4,5). Note that all these substitutions are identical substitutions in the sense that no label changes are involved. Under $f$, node 3 and edges (1,3) and (2,3) are deleted, and node 6 together with its incident edges (4,6) and (5,6) are inserted. There are, of course, many other *ecgm*'s from $g_1$ to $g_2$.

**Definition 7.** The *cost* of an *ecgm* $f$: $\hat{V}_1 \to \hat{V}_2$ from a graph $g_1=(V_1,\alpha_1,\beta_1)$ to a graph $g_2=(V_2,\alpha_2,\beta_2)$ is given by

$$c(f)=\sum_{x\in\hat{V}_1} c_{ns}(x)+\sum_{x\in V_1-\hat{V}_1} c_{nd}(x)+\sum_{x\in V_2-\hat{V}_2} c_{ni}(x)+\sum_{e\in\hat{E}_1} c_{es}(e)+\sum_{e\in E_1-\hat{E}_1} c_{ed}(e)+\sum_{e\in E_2-\hat{E}_2} c_{ei}(e),$$

where
- $c_{ns}(x)$ is the cost of substituting a node $x\in\hat{V}_1$ by $f(x)\in\hat{V}_2$,
- $c_{nd}(x)$ is the cost of deleting a node $x\in V_1-\hat{V}_1$ from $g_1$,

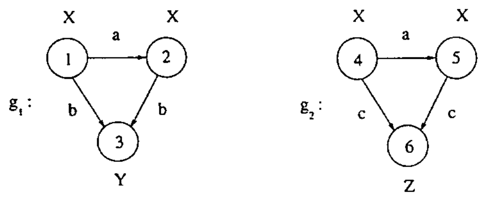

Fig. 1. Two graphs $g_1$ and $g_2$.

- $c_{ni}(x)$ is the cost of inserting a node $x \in V_2 - \hat{V}_2$ in $g_2$,
- $c_{ed}(e)$ is the cost of deleting an edge $e \in E_1 - \hat{E}_1$ from $g_1$,
- $c_{ei}(e)$ is the cost of inserting an edge $e \in E_2 - \hat{E}_2$ in $g_2$,
- $c_{es}(e)$ is the cost of substituting an edge $e = (x,y) \in \hat{E}_1$ by $e' = (f(x),f(y)) \in \hat{E}_2$.

All costs are non-negative real numbers.

In this definition, the shorthand notations $E_1, \hat{E}_1, E_2$, and $\hat{E}_2$ have been used for $V_1 \times V_1$, $\hat{V}_1 \times \hat{V}_1$, $V_2 \times V_2$, and $\hat{V}_2 \times \hat{V}_2$, respectively. Thus, the cost of an *ecgm* is simply obtained by summing the costs of all edit operations that are implied by the mapping $f$.

**Definition 8.** The *edit distance* $d(g_1,g_2)$ of two graphs $g_1$ and $g_2$ is the minimum cost taken over all *ecgm*'s from $g_1$ to $g_2$, i.e., $d(g_1,g_2) = \min\{c(f) \mid f \text{ is an } ecgm \text{ from } g_1 \text{ to } g_2\}$.

In practice, the costs $c_{ns}, \ldots, c_{es}$ introduced in Definition 7 are used to model the likelihood of errors or distortions that may corrupt ideal patterns. The more likely a certain distortion is to occur, the smaller is its cost. Concrete values for $c_{ns}, \ldots, c_{es}$ have to be chosen depending on the particular application. In the remainder of this paper, we consider a special cost function that is interesting from the theoretical point of view in that it allows us to establish a relation between graph edit distance and maximum common subgraph. This special cost function is such that the *ecgm* which corresponds to the graph edit distance between two graphs also defines a maximum common subgraph. This cost function is defined as follows:

$$c_{ns}(x) = \begin{Bmatrix} 0 & \text{if } \alpha_1(x) = \alpha_2(f(x)), \\ \infty & \text{otherwise,} \end{Bmatrix} \quad \text{for any } x \in \hat{V}_1,$$

$$c_{nd}(x) = 1 \quad \text{for any } x \in V_1 - \hat{V}_1,$$

$$c_{ni}(x) = 1 \quad \text{for any } x \in V_2 - \hat{V}_2,$$

$$c_{es}(e) = \begin{Bmatrix} 0 & \text{if } \beta_1((x,y)) = \beta_2((f(x),f(y))), \\ \infty & \text{otherwise,} \end{Bmatrix} \quad \text{for any } e = (x,y) \in \hat{E}_1,$$

$$c_{ed}(e) = 0 \quad \text{for any } e = (x,y) \in E_1 - \hat{E}_1,$$

$$c_{ec}(e) = 0 \quad \text{for any } e = (x,y) \in E_2 - \hat{E}_2.$$

Under this cost function, any node deletion and insertion has a cost equal to one. Identical node and edge substitutions have zero cost, while substitutions involving different labels have infinite cost. The insertion or deletion of an edge incident to a node that is also inserted or deleted, respectively, has no cost. Thus, for the considered cost function the equation given in Definition 7 reduces to

$$c(f) = \sum_{x \in \hat{V}_1} c_{ns}(x) + \sum_{x \in V_1 - \hat{V}_1} c_{nd}(x) + \sum_{x \in V_2 - \hat{V}_2} c_{ni}(x) + \sum_{e \in \hat{E}_1} c_{es}(e).$$

In the following we will be particularly interested in *ecgm*'s with minimum cost. As for any two graphs $g_1 = (V_1, \alpha_1, \beta_1)$ and $g_2 = (V_2, \alpha_2, \beta_2)$ there is always an *ecgm* $f$ with cost $c(f) = |V_1| + |V_2|$ (corresponding to the case where all nodes together with their incident edges are deleted from $g_1$, and all nodes with their incident edges are inserted in $g_2$), any edit operation with infinite cost will never be applied. Thus we may think of edit operations with infinite cost as non-admissible. In other words, we focus our attention on *ecgm*'s involving only insertions, deletions, identical node and edge substitutions, but no non-identical substitutions.

**Example 2.** For the *ecgm* $f$ discussed in Example 1, we have $c(f) = 2$ under the considered cost function. Obviously there is no other *ecgm* $f'$ from $g_1$ to $g_2$ with a cost smaller than $\infty$.

**Lemma 1.** *Let $f$ be an ecgm from a graph $g_1 = (V_1, \alpha_1, \beta_1)$ to a graph $g_2 = (V_2, \alpha_2, \beta_2)$ with $c(f) < \infty$. Then there exists a graph isomorphism between $\hat{g}_1$ and $\hat{g}_2$.*

**Proof.** Assume $\hat{V}_1 = \emptyset$. Then each node and edge in $g_1$ is deleted, and each node and edge in $g_2$ is inserted. Thus there exists the empty graph isomorphism between $\hat{g}_1$ and $\hat{g}_2$.

Now assume $\hat{V}_1 \neq \emptyset$. Because of $c(f) < \infty$, we know that no non-identical node and edge substitutions are implied by $f$. Therefore, for any $x \in \hat{V}_1$ we know that $\alpha_1(x) = \alpha_2(f(x))$. Also, for any pair $(x,y) \in \hat{V}_1 \times \hat{V}_1$, $\beta_1((x,y)) = \beta_2((f(x),f(y)))$. Thus there exists a graph isomorphism between $\hat{g}_1$ and $\hat{g}_2$. □

**Lemma 2.** *Let $f$ be the same as in Lemma 1. Then $c(f) = |V_1| + |V_2| - 2|\hat{V}_1|$.*

**Proof.** If $\hat{V}_1 = \emptyset$ then the cost of $f$ is $|V_1|$ – times the cost of a node deletion plus $|V_2|$ – times the cost of a node insertion. As $|\hat{V}_1| = 0$ the equation of Lemma 2 clearly holds. For $\hat{V}_1 \neq \emptyset$, any node from $\hat{V}_1$ is mapped to a node from $\hat{V}_2$ with zero cost. Also all edges from $\hat{V}_1 \times \hat{V}_1$ are mapped to the edges of $\hat{V}_2 \times \hat{V}_2$ with zero cost. The total remaining cost is $|V_1| - |\hat{V}_1|$ node deletions plus $|V_2| - |\hat{V}_2|$ node insertions. The equation of Lemma 2 follows as $|\hat{V}_1| = |\hat{V}_2|$. □

**Theorem 1.** *Let $g_1 = (V_1, \alpha_1, \beta_1)$ and $g_2 = (V_2, \alpha_2, \beta_2)$ be two graphs and $g = (V, \alpha, \beta)$ a maximum common subgraph of $g_1$ and $g_2$. Then $d(g_1, g_2) = |V_1| + |V_2| - 2|V|$.*

**Proof.** According to Definition 8, $d(g_1, g_2)$ is the minimum cost taken over all *ecgm*'s from $g_1$ and $g_2$. Because of Lemma 2 we know that the cost of any *ecgm f* is minimized by maximizing $|\hat{V}_1|$. And from Lemma 1 it follows that there is a graph isomorphism from $\hat{g}_1$, the graph induced by $\hat{V}_1$, to $\hat{g}_2$, the graph induced by $\hat{V}_2$. Consequently, $\hat{g}_1$ is a common subgraph of $g_1$ and $g_2$, and any maximum common subgraph $g$ minimizes the right-hand side of the equation in Theorem 1. □

## 3. Conclusions

In this paper, we have formally defined graph edit distance and introduced a special cost function under which graph edit distance computation is equivalent to the maximum common subgraph problem. That is, any function $f$ that minimizes the cost of mapping the nodes and edges of a graph $g_1$ to those of another graph $g_2$ is a graph isomorphism between a subgraph $\hat{g}_1$ of $g_1$ and a subgraph $\hat{g}_2$ of $g_2$, and both $\hat{g}_1$ and $\hat{g}_2$ are a maximum common subgraph of $g_1$ and $g_2$.

This result is not only interesting from the theoretical point of view, but it may be of practical relevance, too. An immediate consequence is that any algorithm that computes graph edit distance may be used for maximum common subgraph computation if it is run under the cost function introduced in this paper, and vice versa. Recently, some new algorithms for graph edit distance computation have been proposed (Bunke and Messmer, 1997). These algorithms make use of intensive preprocessing in order to reduce the computation time needed on-line. With the result derived in the present paper, these algorithms become applicable to the maximum common subgraph problem, too.

Error-correcting graph matching is a generalization of approximate string matching. A relation between string edit distance and the longest common subsequence of two strings, similar to the relation that has been derived for graphs in this paper, has been known for long (Wagner and Fischer, 1974). Recently, it has been shown that this relation holds true not only for the particular cost function given by Wagner and Fischer (1974), but for a whole class of cost functions (Rice et al., 1997). That is, there are infinitely many cost functions under which string edit distance computation is equivalent to finding the longest common subsequence of two strings. It is an interesting open question whether a similar class of cost functions exists for graph edit distance.

## References

Bunke, H., Allerman, G., 1983. Inexact graph matching for structural pattern recognition. Pattern Recognition Letters 1 (4), 245–253.

Bunke, H., Messmer, B., 1997. Recent advances in graph matching. Internat. J. Pattern Recognition Artif. Intell. 11 (1), 169–203.

Cho, C.J., Kim, J.J., 1992. Recognizing 3-D objects by forward checking constrained tree search. Pattern Recognition Letters 13 (8), 587–597.

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

Horaud, R., Skordas, T., 1989. Stereo correspondence through feature grouping and maximal cliques. IEEE Trans. Pattern Anal. Machine Intell. 11 (11), 1168–1180.

Lee, S., Kim, J.H., 1989. Attributed stroke graph matching for seal imprint verification. Pattern Recognition Letters 9, 137–145.

Lee, S.W., Kim, J.H., Groen, F.C.A., 1990. Translation-, rotation-, and scale invariant recognition of hand-drawn symbols in schematic diagrams. Internat. J. Pattern Recognition Artif. Intell. 4 (1), 1–15.

Levinson, R., 1992. Pattern associativity and the retrieval of semantic networks. Comput. Math. Appl. 23, 573–600.

Lu, S.W., Ren, Y., Suen, C.Y., 1991. Hierarchical attributed graph representation and recognition of handwritten Chinese characters. Pattern Recognition 24, 617–632.

Pearce, A., Caelli, T., Bischof, W.F., 1994. Rulegraphs for graph matching in pattern recognition. Pattern Recognition 27 (9), 1231–1246.

Read, R.C., Corneil, D.G., 1977. The graph isomorphism disease. J. Graph Theory 1, 339–363.

Rice, S., Bunke, H., Nartker, T., 1997. Classes of cost functions for string matching. Algorithmica 18 (2), 271–280.

Sanfeliu, A., Fu, K.S., 1983. A distance measure between attributed relational graphs for pattern recognition. IEEE Trans. Systems Man Cybernet. 13, 353–363.

Shapiro, L.G., Haralick, R.M., 1981. Structural descriptions and inexact matching. IEEE Trans. Pattern Anal. Machine Intell. 3, 504–519.

Ullman, J.R., 1976. An algorithm for subgraph isomorphism. J. ACM 23 (1), 31–42.

Wagner, R.A., Fischer, M.J., 1974. The string-to-string correction problem. J. ACM 21 (1), 168–173.

Wong, E.K., 1992. Model matching in robot vision by subgraph isomorphism. Pattern Recognition 25 (3), 287–304.