Error Correcting Graph Matching: On the Influence of the Underlying Cost Function

H. Bunke, Member, IEEE Computer Society

Abstract—This paper investigates the influence of the cost function on the optimal match between two graphs. It is shown that, for a given cost function, there are an infinite number of other cost functions that lead, for any given pair of graphs, to the same optimal error correcting matching. Furthermore, it is shown that well-known concepts from graph theory, such as graph isomorphism, subgraph isomorphism, and maximum common subgraph, are special cases of optimal error correcting graph matching under particular cost functions.

Index Terms—Graph, subgraph, maximum common subgraph, graph isomorphism, subgraph isomorphism, graph matching, error correcting graph matching, cost function, edit operation, graph edit distance.

1 INTRODUCTION

Graphs are a general and powerful data structure 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 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.

In a graph representation, the nodes typically represent objects or parts of objects, while the edges describe relations between objects or object parts. For example, a node may represent a line segment, a closed region, or a surface patch of a 3D object, while an edge describes relationships such as collinearity or parallelism for straight lines or spatial adjacency for image regions and surface patches. Graphs have some interesting invariance properties. For example, 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 very well-suited to model complex objects in terms of parts and their relations, make them very attractive for various applications. Graph representations and graph matching have been successfully applied to a large number of problems in computer vision and pattern recognition. Examples include character recognition [1], [2], [3], schematic diagram interpretation [4], shape analysis [5], image registration [6], and 3D object recognition [7].

Algorithms for graph matching include graph and subgraph isomorphism [8]. However, due to errors and distortions in the input data, approximate, or error-correcting, graph matching methods are needed in most applications [9], [10], [11]. Another way to cope with distorted input graphs is to use the maximum common subgraph in order to measure graph similarity [12]. 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. Moreover, randomized algorithms, such as probabilistic relaxation, neural networks, simulated annealing and genetic algorithms, can be used to reduce the required computation time.

Error correcting graph matching is a generalization of string matching, or string edit distance computation [13]. In order to measure the similarity of two graphs, graph edit operations are introduced, for example, the deletion, insertion, and substitution of nodes and edges. Often, each of these edit operations is assigned a certain cost. The costs are application dependent and reflect the likelihood of graph distortions. The more likely a certain distortion is to occur, the smaller is its cost. Concrete algorithms for error correcting graph matching are based on tree search using A*-like heuristic evaluation functions to prune the search space [9], [10], [11], probabilistic relaxation [6], neural networks [15], simulated annealing [14], and genetic algorithms [3].

In error correcting graph matching, the cost function, i.e., the costs that are assigned to the individual edit operations, have an important influence on the matching results. Two graphs that are similar under one particular cost function may be no longer similar under another cost function. Similarly, the optimal correspondence of the nodes and edges of two graphs may significantly change if the underlying cost function changes. In this paper, the influence of the cost function on error correcting graph matching is studied. It will be shown that, for any given cost function, there is an infinity of other cost functions that result in the same error correcting matching, i.e., correspondence of nodes and edges, for any two given graphs. Under certain cost functions, the optimal error correcting matching, if it exists, always corresponds to an isomorphism, a subgraph isomorphism, or a maximum common subgraph of the two graphs under consideration. All these properties are independent of the underlying algorithm that implements error correcting graph matching.

2 BASIC DEFINITIONS AND NOTATION

Some of the following definitions are taken from [16]. 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 nodes, \( \alpha : V \rightarrow L \) is the node labeling function, and \( \beta : V \times V \rightarrow L \) is the edge labeling function.

The set of edges \( E \) is implicitly given by assuming that our 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 doesn’t restrict generality. If it is necessary to model the situation where edges exist only between distinguished pairs of nodes, we include a special label \( \text{null} \) in the set of labels. For any two given graphs, \( L \). Edges are directed, i.e., edge \((x, y)\) originates at node \( x \in V \) and terminates at node \( y \in V \). An undirected graph is obtained as a special case if \( \beta((x, y)) = \beta(y, x) \) for any \( x, y \in V \). 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 \( 1) V' \subseteq V \), \( 2) \alpha'(x) = \alpha(x) \) for all \( x \in V' \), and \( 3) \beta'(y, z) = \beta((y, z)) \) for all \( (y, z) \in V' \times V' \).
Example 1. A graphical representation of two graphs is given in Fig. 1. For these graphs, we have:

\[ V_1 = \{1, 2, 3\}; V_2 = \{4, 5, 6\}; L = \{X, Y, Z, a, b, \text{null}\} \]

\[ \alpha_1 : 1 \rightarrow X, 2 \rightarrow X, 3 \rightarrow Y; \alpha_2 : 4 \rightarrow X, 5 \rightarrow X, 6 \rightarrow Z \]

\[ \beta_1 : (1, 2) \rightarrow a, (1, 3) \rightarrow b; (2, 3) \rightarrow b; \beta_2 : (4, 5) \rightarrow a, (4, 6) \rightarrow b, (5, 6) \rightarrow b \]

All other edges are labeled with \text{null} and are not shown in Fig. 1. An example of an \textit{ecgm} is \( f : 1 \rightarrow 4, 2 \rightarrow 5 \) with \( V_1 = \{1, 2\} \) and \( V_2 = \{4, 5\} \). Under this \textit{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, i.e., there are no label changes 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), is inserted. There are, of course, many other \textit{ecgms} from \( g_1 \) to \( g_2 \).

Definition 7. The cost of an \textit{ecgm} \( f : V_1 \rightarrow V_2 \) 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

\[ \gamma(f) = \sum_{x \in V_1 - V_2} c_{nd}(x) + \sum_{x \in V_1 - V_2} c_{si}(x) + \sum_{x \in V_1} c_{ns}(x) + \sum_{e \in E_2} c_{es}(e), \]

where \( c_{nd}(x) \) is the cost of deleting a node \( x \in V_1 - V_2 \) from \( g_1 \), \( c_{si}(x) \) is the cost of inserting a node \( x \in V_2 - V_2 \) in \( g_2 \), \( c_{ns}(x) \) is the cost of substituting a node \( x \in V_1 \) by \( f(x) \) in \( g_2 \), and \( c_{es}(e) \) is the cost of substituting an edge \( e = (x, y) \in V_1 \times V_1 \) by \( e' = (f(x), f(y)) \) in \( V_2 \times V_2 \). All costs are nonnegative real numbers.

The costs introduced in Definition 7 are often used to model the likelihood of errors and distortions that may corrupt ideal patterns. Typically, the more likely a certain distortion is to occur, the lower is its cost. Concrete values of \( c_{nd}, c_{si}, c_{ns}, \) and \( c_{es} \) have to be chosen dependent on the particular application. According to Definition 7, no individual costs for the deletion of edges from \((V_1 - V_1) \times (V_1 - V_2), (V_1 - V_1) \times V_2, V_1 \times (V_2 - V_2), (V_2 - V_2) \times (V_2 - V_2), (V_2 - V_2) \times V_2, V_2 \times (V_2 - V_2) \) are taken into account. The reason is that these operations are automatically implied by the deletion of nodes from \((V_1 - V_1)\) and the insertion of nodes \((V_2 - V_2)\), respectively. Thus, it is assumed that their costs are included in the costs of the corresponding node deletions and insertions. In other words, the cost of a node deletion (insertion) includes not only the cost of deleting (inserting) a node, but also the deletion (insertion) of the edges that connect it to the other nodes of the graph. Note that the deletion of edges from \( V_1 \times V_1 \) and the insertion of edges in \( V_2 \times V_2 \) is modeled via edge substitutions (replacing label \( \neq \) null by null, or replacing null by \( l \neq \) null).

In the remainder of this paper, it will be assumed that the costs \( c_{nd}(x), c_{si}(x), \) and \( c_{ns}(x) \) don’t depend on node \( x \) but depend on edge \( e \). In other words, \( c_{nd}(x) \) will be the same for any node \( x \in V_1 - V_2 \). Similarly, \( c_{si}(x) \) and \( c_{ns}(x) \) will be the same for any node \( x \in V_2 - V_1 \). Then, according to Definition 7, \( c_{es}(e) \) will be the same for any edge \( e \in V_1 \times V_1 \). The graph \( G = (V, L, c_{es}) \) will be called a cost function. If the cost function \( G \) is to be explicitly mentioned, the notation \( \gamma_G(f) \) will be used instead of \( \gamma(f) \).

Unless otherwise stated, it will be assumed that the cost of an identical node or edge substitution is zero, while the cost of any other edit operation is greater than zero.

Definition 8. Let \( f \) be an \textit{ecgm} from \( g_1 \) to \( g_2 \) and \( C \) a cost function. We will call \( f \) a minimal \textit{ecgm} \( \textit{under} \ C \) if there is no other \textit{ecgm} \( f' \) from \( g_1 \) to \( g_2 \) with \( \gamma_C(f') \leq \gamma_C(f) \).

The cost of an optimal \textit{ecgm} \( \textit{under} \ C \) is called the edit distance between \( g_1 \) and \( g_2 \). For a given cost function \( C \), there are usually several optimal \textit{ecgms} from a graph \( g_1 \) to another graph \( g_2 \).

Example 2. Consider cost function

\[ C = (c_{nd}, c_{si}, c_{ns}, c_{es}) = (1, 1, 1, 2) \]

Then, the \textit{ecgm} \( f \) given in Example 1 has cost \( \gamma_C(f) = 2 \).
In order to further illustrate the formal concepts introduced in this section, we consider an example from 3D shape analysis. A 3D wedge and one of its possible graph representations are shown in Fig. 2a. In the graph, nodes represent planar surface patches with the size and shape of a surface patch as labels (for the sake of clarity, not shown in Fig. 2b), and edges represent the angle enclosed by the surface normal vectors between any two surface patches. The graph representation of the wedge shown in Fig. 2a is not unique as the nodes and edges may be listed in any order. However, all these representation are isomorphic to each other. Thus, graph isomorphism is useful to check if two objects are identical. Clearly, the graph is Fig. 2b is independent of any particular viewpoint. Taking a 3D (range) image of the wedge from a particular point in 3D space will result in a graph that is a subgraph of the graph shown in Fig. 2b. For an example, see Fig. 2c, d. This example shows that subgraph isomorphism is a suitable concept to identify instances of the wedge in an input image.

In order to derive the graph representation shown in Fig. 2d from an input image, suitable segmentation procedures must be applied. However, due to noise and distortions, there will be errors in the segmentation results. In order to model (and correct) these errors, graph edit operations with associated costs are introduced. For example, the cost of an edge substitution may be equal to the angular difference between a pair of surface normals of the undistorted wedge (Fig. 2d) and its noisy version (Fig. 2e). Finding an optimal error correcting graph matching means finding an optimal (i.e., least cost) registration between the graph of an undistorted object and its distorted version. In our example in Fig. 2d, e the optimal registration, i.e., mapping, is given by 7′ → 7, 8′ → 8, 9′ → 9, with a total cost of 10 (caused by different edge labels). In general, the optimal registration between a graph and its distorted version crucially depends on the edit operations and their cost. This influence will be studied in greater detail in the remainder of this paper.

**3 EQUIVALENCE OF COST FUNCTIONS**

Consider two different cost functions $C = (c_{nd}, c_{ni}, c_{ns}, c_{es})$ and $C' = (c'_{nd}, c'_{ni}, c'_{ns}, c'_{es})$ where $C'$ is a scaled version of $C$, i.e., $c_{nd} = \alpha c'_{nd}$, $c_{ni} = \alpha c'_{ni}$, $c_{ns} = \alpha c'_{ns}$, and $c_{es} = \alpha c'_{es}$ for some $\alpha > 0$. Then, one would intuitively expect that, for any two graphs, any $ecgm f$ that is optimal under $C$ is also optimal under $C'$ and vice versa. Just the absolute costs of the two optimal $ecgms$ would differ by a factor of $\alpha$, i.e., $\gamma_C(f) = \alpha \gamma_{C'}(f)$. In this section, it will be shown that optimal $ecgms$ under a cost function $C$ are optimal under another cost function $C'$ not only if $C'$ is a scaled version of $C$, but for a much larger class of cost functions $C'$.

**Lemma 1.** Let $g_1$ and $g_2$ be two graphs, $C = (c_{nd}, c_{ni}, c_{ns}, c_{es})$ a cost function, and $f$ an $ecgm$ from $g_1$ to $g_2$. Then,

$$
\gamma_C(f) = (|V_1| - |V_1'|) \cdot c_{nd} + (|V_2| - |V_2'|) \cdot c_{ni} + N_f \cdot c_{ns} + E_f \cdot c_{es}
$$

(3.1)

where $N_f \leq |V_1|$ is the number of nonidentically substituted nodes from $V_1$, and $E_f \leq |V_1|(|V_1'| - 1)$ is the number of nonidentically substituted edges from $V_1 \times V_1$.

**Proof.** This lemma immediately follows from Definition 7. There are $|V_1| - |V_1'|$ node deletions, each involving cost $c_{nd}$ and $|V_2| - |V_2'|$ node insertions, each involving cost $c_{ni}$. Furthermore, each nonidentical node substitution has cost $c_{ns}$ and each nonidentical edge substitution has cost $c_{es}$. □

Fig. 2. (a) Wedge (the dashed line represents a hidden edge); (b) graph representation of (a) (see text); (c) an image of the wedge; (d) graph representation of (c); (e) graph of a distorted version of the wedge.
Theorem 1. Let \( g_1 \) and \( g_2 \) be two graphs and \( f \) an \( ecdnm \) from \( g_1 \) to \( g_2 \). Furthermore, let \( C = (c_{in}, c_{nd}, c_{ns}, c_{es}) \) and \( C' = (c'_{in}, c'_{nd}, c'_{ns}, c'_{es}) \) be two cost functions such that

\[
\frac{c_{in} + c_{nd}}{c_{ns}} = \frac{c'_{in} + c'_{nd}}{c'_{ns}} \tag{3.2}
\]

and

\[
\frac{c_{es}}{c_{ns}} = \frac{c'_{es}}{c'_{ns}}. \tag{3.3}
\]

The \( ecdnm f \) is an optimal \( ecdnm \) under cost function \( C \) if and only if \( f \) is an optimal \( ecdnm \) under cost function \( C' \).

Proof. Rewriting (3.1) and substituting \( |V_1| = |V_2| \) yields

\[
\gamma_C(f) = |V_1| \cdot c_{nd} + |V_2| \cdot c_{ns} - \left( |V_1| \frac{c_{in} + c_{nd}}{c_{ns}} N_f + E_f c_{es} \right). \tag{3.4}
\]

An optimal \( ecdnm f \) is one that minimizes the right-hand side of this equation. As all costs in (3.4) are greater than zero, minimizing the right hand side of (3.4) means maximizing the expression

\[
\frac{c_{in} + c_{nd}}{c_{ns}} N_f + E_f c_{es} = c_{es} c_{ns} N_f + E_f c_{es} c_{ns}. 
\]

Obviously, this expression doesn’t depend on the costs \( c_{nd}, c_{ns} \), and \( c_{es} \) individually, but only on the two ratios \( (c_{in} + c_{nd})/c_{ns} \) and \( c_{es}/c_{ns} \). Therefore, if \( f \) is an optimal \( ecdnm \) under cost function \( C \), it will also be an optimal \( ecdnm \) under any other cost function \( C' \) as long as (3.2) and (3.3) hold true.

Theorem 2. Let \( g_1, g_2, f, C \), and \( C' \) be the same as in Theorem 1. Then,

\[
\gamma_C(f) = \frac{1}{c_{ns}} \left(|V_1| (c_{in} c_{nd} - c_{nd} c_{ns}) + |V_2| (c'_{in} c_{ns} - c_{in} c_{ns}) + \gamma_C(f) c_{es}\right). \tag{3.5}
\]

Proof. Rewriting (3.1) yields

\[
N_f = \frac{1}{c_{ns}} \gamma_C(f) - |V_1| c_{nd} - |V_2| c_{ns} + |V_1| (c_{in} + c_{nd}) - E_f c_{es}. \tag{3.5}
\]

For cost function \( C' \) we get, similarly to (3.1),

\[
\gamma_C(f) = |V_1| c'_{nd} + |V_2| c'_{ns} - |V_1| (c'_{in} + c'_{nd}) + E_f c'_{es} + |V_2| c_{ns}. \tag{3.6}
\]

Substituting (3.5) into (3.6) yields

\[
\gamma_C(f) = \frac{1}{c_{ns}} \left( |V_1| (c_{in} c_{nd} - c_{nd} c_{ns}) + |V_2| (c'_{in} c_{ns} - c_{in} c_{ns}) - |V_1| (c_{in} c_{nd} - c_{nd} c_{ns}) - c_{in} c_{ns} + E_f (c'_{in} c_{ns} - c_{in} c_{ns}) + \gamma_C(f) c_{es}\right). 
\]

Now, the equation of Theorem 2 follows because of (3.2) and (3.3).

From Theorem 1, we can conclude that any two cost functions \( C \) and \( C' \) satisfying (3.2) and (3.3), respectively, are equivalent in the sense that any optimal \( ecdnm \) under the one cost function is an optimal \( ecdnm \) under the other. From Theorem 2, we know that the graph edit distance under such two cost functions will be different in general. However, given the edit distance for one cost function, we can easily get the edit distance under the other function. In particular, given an algorithm for graph edit distance computation designed for a particular cost function \( C \), we can use the same algorithm for any other cost function \( C' \) for which (3.2) and (3.3) are satisfied.

4 ERROR CORRECTING GRAPH MATCHING AND MAXIMUM COMMON SUBGRAPH

Theorem 3. Let \( g_1, g_2, \) and \( g_3 \) be graphs such that both \( g_1 \leq g_2 \) and \( g_2 \leq g_3 \) are a maximum common subgraph of \( g_1 \) and \( g_2 \). Furthermore, let \( f : V_1 \to V_2 \) be an isomorphism between \( g_1 \) and \( g_2 \) and \( C = (c_{in}, c_{nd}, c_{ns}, c_{es}) \) a cost function with \( c_{nd} + c_{ns} \leq c_{es} \) and \( c_{in} + c_{nd} \leq c_{es} \). Then, \( f \) is an optimal \( ecdnm \) from \( g_1 \) to \( g_2 \).

Proof. Assume that \( f \) is not an optimal \( ecdnm \). Then, there must exist another function \( f' : V_1 \to V_2 \) such that \( \gamma_C(f') < \gamma_C(f) \).

The function \( f' \) is not an isomorphism between two maximum subgraphs of \( g_1 \) and \( g_2 \). Hence, \( f' \) must follow either case 1 or 2 discussed below.

Case 1: \( f' \) is an isomorphism between two common subgraphs of \( g_1 \) and \( g_2 \), but not between two maximum common subgraphs. In this case, we have \( |V_1'| = |V_2'| < |V_1| = |V_2| \) and \( \gamma_C(f') = (|V_1' - V_1|) c_{nd} + (|V_2' - V_2|) c_{es} \). Recall that \( \gamma_C(f) = (|V_1 - V_1|) c_{nd} + (|V_2 - V_2|) c_{es} \). Because \( |V_1'| < |V_1| \) and \( |V_2'| < |V_2| \), it follows that \( \gamma_C(f) < \gamma_C(f') \), which is a contradiction to our assumption.

Case 2: \( f' \) is not an isomorphism. Assume there are 0 \( \leq \) \( m \leq |V_1'| \) nodes from \( V_1' \) that are mapped with zero cost to a node of \( V_2' \). (Mapping a node \( x \) with zero cost means that \( \beta(x) = \beta(f(x)) \) and, for every pair of edges \((x, y), (y, x) \in V_1' \times V_1' \), we have \( \beta(x, y) = \beta(f(x, y)) \) and \( \beta(y, x) = \beta(f(y, x)) \)). For each of the remaining \( |V_1' - m \) nodes \( x \), we observe \( \beta(x) \neq \beta(f(x)) \) or there exists an edge \( (x, y) \in V_1' \) with \( \beta(x, y) \neq \beta(f(x, y)) \) or \( \beta(y, x) \neq \beta(f(y, x)) \). Thus,

\[
\gamma_C(f') \geq \gamma_C(f) \text{ or } \text{min}_{c_{ns}, c_{es}} \left( \gamma_C(f) c_{es} + |V_1| c_{nd} + |V_2| c_{ns} - (|V_1| c_{in} + c_{nd}) + (|V_2| c_{in} + c_{nd})\right) \text{ or } \gamma_C(f) \geq \gamma_C(f') \text{ holds in Case 2, which is again a contradiction to our assumption.}
\]

This theorem tells us that any \( ecdnm \) that corresponds to the maximum common subgraph of two graphs is optimal if the cost function satisfies the conditions \( c_{nd} + c_{ns} \leq c_{es} \) and \( c_{in} + c_{nd} \leq c_{es} \). On the other hand, not any optimal \( ecdnm \) necessarily corresponds to a maximum common subgraph. This can be concluded from the two graphs shown in Fig. 1 and the cost function \( C \) given in Example 2. Obviously, \( C \) satisfies the condition of Theorem 3 and the function \( f \) of Example 1 corresponds to the maximum common subgraph of \( g_1 \) and \( g_2 \). However, the function \( f' \) given in Example 2 is also optimal, but doesn’t correspond to a maximum common subgraph of \( g_1 \) and \( g_2 \). The next theorem states that, under slightly more restricted cost functions, any optimal \( ecdnm \) necessarily corresponds to a maximum common subgraph.

Theorem 4. Let \( g_1 \) and \( g_2 \) be graphs and \( C = (c_{in}, c_{nd}, c_{ns}, c_{es}) \) a cost function with \( c_{nd} + c_{ns} \leq c_{es} \) and \( c_{in} + c_{nd} \leq c_{es} \). Then, for any optimal \( ecdnm f : V_1 \to V_2 \), both \( g_1 \) and \( g_2 \) are a maximum common subgraph of \( g_1 \) and \( g_2 \).

Proof. The proof is similar to that of Theorem 3. Assume \( f \) is optimal but doesn’t correspond to a maximum common subgraph. If \( f \) is not an isomorphism, then there must exist a node \( x \in V_1 \) or an edge \((x, y) \in V_1 \times V_1 \) such that \( \beta(x) \neq \beta(f(x)) \) or \( \beta(x, y) \neq \beta(f(x, y)) \). Now, the mapping \( f' : V_1 \to V_2 \) with \( f'(z) = f(z) \) for \( z \notin V_1 - \{x\} \) has a lower cost than \( f \), i.e., \( \gamma_C(f') < \gamma_C(f) \), which contradicts our assumption. The case where \( f \) is an isomorphism but doesn’t
In this subsection, we’ll consider edit operations with infinity cost and optimal error correcting graph matching is equivalent to the maximum common subgraph of two graphs. This result is a generalization of a similar relation that was shown for the particular cost function $c_{\text{nd}} = c_{\text{ni}} = 1$, $c_{\text{es}} = c_{\text{es}} = \infty$ [16]. It can also be regarded as a generalization of a well-known result in string matching stating that string edit distance and longest common subsequence are equivalent under a special cost function [13], [17].

Note that graph isomorphism and subgraph isomorphism are special cases of maximum common subgraph where, according to Definition 4, $g_1^* = g_1$ and $g_2^* = g_2$ (graph isomorphism) or $g_1^* = g_1$ and $g_2 \subset g_2$ ($g_2^* \neq g_2$) (subgraph isomorphism). Thus, any algorithm that implements optimal eegm can be used for graph isomorphism, subgraph isomorphism, or maximum common subgraph detection if it is run under a cost function that satisfies the conditions of Theorem 4. The resulting cost of the optimal eegm is $\gamma_C(f) = 0$, $\gamma_C(f) = ([V_1] - [V_2])c_{\text{nd}}$, and $\gamma_C(f) = ([V_1] - [V_2])c_{\text{nd}} + ([V_1] - [V_2])c_{\text{ni}}$ for graph isomorphism, subgraph isomorphism, and maximum common subgraph, respectively.

5 MORE GENERAL COST FUNCTIONS

In the previous sections, the cost of identical substitutions was assumed to be zero, while all other edit operations had a cost greater than zero. Now, cases will be considered where any edit operation other than a nonidentical substitution may have a nonnegative real number including zero or infinity as cost. It is still assumed that the cost of identical substitutions is zero.

5.1 Edit Operations with Zero Cost

Case 5.1.1: $c_{\text{nd}} + c_{\text{ni}} = 0$.

If, additionally, $c_{\text{es}} + c_{\text{es}} = 0$, then any function $f : \hat{V}_1 \rightarrow \hat{V}_2$ from any subset $V_1 \subseteq V_1$ to any subset $V_2 \subseteq V_2$ is an optimal eegm with cost equal to zero. Otherwise, if $c_{\text{es}} + c_{\text{es}} > 0$, then there still exists an optimal eegm with zero cost for any pair of graphs. In particular, if $V_1 = \emptyset$, then $\gamma_C(f) = 0$. If there exist common subgraphs $\hat{g}_1$ and $\hat{g}_2$ of $g_1$ and $g_2$ with $\hat{g}_1 \subseteq g_1$ and $\hat{g}_2 \subseteq g_2$, then any isomorphism from $\hat{g}_1$ to $\hat{g}_2$, including the one that corresponds to the maximum common subgraph, is an optimal eegm with $\gamma_C(f) = 0$. (The proof is similar to that of Theorem 3.)

Case 5.1.2: $c_{\text{nd}} + c_{\text{ni}} > 0$.

If $c_{\text{es}} + c_{\text{es}} = 0$ and $|V_1| < |V_2|$, then any mapping $f : \hat{V}_1 \rightarrow \hat{V}_2$ from $V_1$ to any subset $\hat{V}_2 \subseteq V_2$ is optimal and $\gamma_C(f) = ([V_1] - [V_2])c_{\text{nd}}$. Similarly, if $c_{\text{es}} + c_{\text{es}} = 0$ and $|V_1| \geq |V_2|$, then any function $f : \hat{V}_1 \rightarrow \hat{V}_2$ from any subset $\hat{V}_1 \subseteq V_1$ to $V_2$ is optimal and $\gamma_C(f) = ([V_1] - [V_2])c_{\text{nd}}$. If $c_{\text{es}} = 0$ and $c_{\text{es}} > 0$, the optimal eegm is the one that maximizes the expression $|V_1||c_{\text{nd}} + c_{\text{es}}| - E_jc_{\text{es}}$. Otherwise, if $c_{\text{es}} > 0$ and $c_{\text{ns}} = 0$, the expression $|V_1||c_{\text{nd}} + c_{\text{es}}| - E_jc_{\text{es}}$ will be maximized (see (3.1)).

5.2 Edit Operations with Infinity Cost

In this subsection, we’ll consider edit operations with infinity cost and ask for optimal eegms $f$ with a cost $\gamma_C(f)$ smaller than infinity. Thus, edit operations with infinity cost may be regarded as forbidden operations.

Case 5.2.1: $c_{\text{nd}} = c_{\text{ni}} = \infty$.

If $c_{\text{es}} = c_{\text{es}} = \infty$, an eegm $g$ with $\gamma_C(f) < \infty$ exists only if $g_1$ and $g_2$ are isomorphic to each other. Any isomorphism $f$ between $g_1$ and $g_2$ is optimal and $\gamma_C(f) = 0$. If $0 \leq c_{\text{es}}, c_{\text{es}} < \infty$, then an eegm $f$ with $\gamma_C(f) < \infty$ exists only if $|V_1| = |V_2| = \{V_1|V_2\}$. Optimal eegms are those that minimize $N_jc_{\text{es}} + E_jc_{\text{es}}$. If $c_{\text{es}} = \infty$ and $0 \leq c_{\text{es}} < \infty$, then any eegm $f$ with $\gamma_C(f) < \infty$ must satisfy the condition $\beta(x) = \beta(f(x))$ for all $x \in V_1$, and optimal eegms $f$ are those that minimize $E_jc_{\text{es}}$. Similarly, if $c_{\text{es}} = \infty$ and $0 \leq c_{\text{es}} < \infty$, then, for any eegm $f$ with $\gamma_C(f) < \infty$, the condition $\beta(x,y) = \beta(f(x), f(y))$ must hold for all $(x,y) \in V_1 \times V_1$. Optimal eegms are those that minimize $N_jc_{\text{es}}$.

Case 5.2.2: $c_{\text{nd}} + c_{\text{ni}} = \infty$, $c_{\text{es}} \neq c_{\text{es}}$.

If $c_{\text{nd}} = \infty$ and $0 \leq c_{\text{es}} < \infty$, then an eegm $f$ with $\gamma_C(f) < \infty$ exists only if $|V_1| < |V_2|$. In this case, if $c_{\text{es}} = c_{\text{es}} = \infty$, then a necessary and sufficient condition for the existence of an eegm $f$ with $\gamma_C(f) < \infty$ is the existence of a subgraph isomorphism from $g_1$ to $g_2$. The cost of such a subgraph isomorphism is $(|V_1| - |V_2|)c_{\text{es}}$. Similarly, if $0 \leq c_{\text{es}} < \infty$, then the relation $|V_1| \leq |V_2|$ must hold. If $c_{\text{es}} = c_{\text{es}} = \infty$, then an eegm $f$ with $\gamma_C(f)$ exists if and only if there is a subgraph isomorphism from $g_2$ to $g_1$. Any such subgraph isomorphism $f$ has a cost equal to $(|V_1| - |V_2|)c_{\text{es}}$. The cases $1) c_{\text{es}}, c_{\text{es}} < \infty, 2) c_{\text{es}} = \infty, c_{\text{es}} < \infty, 3) c_{\text{es}} < \infty, c_{\text{es}} = \infty$ are similar to Case 5.2.1 for either situation $c_{\text{es}} = \infty$ and $0 \leq c_{\text{es}} < \infty$, or $0 \leq c_{\text{es}} < \infty$ and $c_{\text{es}} = \infty$.

Case 5.2.3: $c_{\text{es}}, c_{\text{es}} < \infty$.

Here, an optimal eegm $f$ with $\gamma_C(f) < \infty$ always exists. For $c_{\text{es}} = c_{\text{es}} = \infty$, see Theorems 3 and 4. If $c_{\text{es}} = \infty$ and $0 \leq c_{\text{es}} < \infty$, or $0 \leq c_{\text{es}} < \infty$ and $c_{\text{es}} = \infty$, the optimal eegm is the mapping that maximizes $|V_1|(c_{\text{es}} + c_{\text{es}}) - E_jc_{\text{es}}$ or $|V_1|(c_{\text{es}} + c_{\text{es}}) - N_jc_{\text{es}}$, respectively, provided that, similar to Case 5.2.1, $\beta(x) = \beta(f(x))$ for all $x \in V_1$ or $\beta(x,y) = \beta(f(x), f(y))$ for all $(x,y) \in V_1 \times V_1$. In either case, Theorems 3 and 4 apply as well.

6 CONCLUSIONS

In this paper, error correcting graph matching has been formally introduced. Then, the influence of the cost function (i.e., the costs of the individual graph edit operations) on error correcting graph matching has been studied. It has been shown that, for each cost function $C$, an infinity of other cost functions exists that are equivalent to $C$ in that they lead to the same optimal error correcting matchings for any two given graphs. An optimal error correcting graph matching just depends on certain ratios of edit costs, but not on all of the different edit costs individually. For two equivalent cost functions $C$ and $C'$, the corresponding optimal error correcting graph matchings have a different cost in general, but, given the cost under the one cost function, it can be easily converted into the cost under the other cost function. Furthermore, it has been shown that standard concepts from graph theory, such as graph isomorphism, subgraph isomorphism, and maximum common subgraph, are special cases of error correcting graph matching under particular cost functions. Thus, any algorithm that implements error correcting graph matching can be used for the computation of graph isomorphism, subgraph isomorphism, and maximum common subgraph if it is run under an appropriate cost function. Conversely, for certain cost functions, algorithms for graph isomorphism, subgraph isomorphisms, or maximum common subgraph detection can be used to implement error correcting graph matching.

The main contributions of this paper are theoretical results on how error correcting graph matching depends on the underlying cost function. However, these results may also be useful for more practical problems, for example, the design of new graph matching algorithms. For the special case of string matching, the time and space complexity of edit distance computation in the general case is $O(n^2)$, where $n$ is the length of the strings involved. However, for special classes of cost functions, it was shown that only $O(n)$
space is needed [17]. Using the results derived in this paper, efficient algorithms for graph matching designed for just one special individual cost function become applicable to an infinite number of other, equivalent cost functions.

Another potential application is the automatic inference of edit costs. This problem is of crucial importance for any practical graph matching application, but it hasn’t been solved yet. As a matter of fact, edit costs for a particular application are typically designed by hand rather than through automatic inference based on a set of sample patterns. Using the edit cost model introduced in Definition 7, inference is a four-dimensional problem (four independent values $c_{nd}, c_{ni}, c_{ns}$, and $c_{es}$ need to be found). However, utilizing the results derived in this paper, the problem is reduced to a two-dimensional one. For example, $c_{nd}$ and $c_{ni}$ can be arbitrarily defined and only $c_{ns}$ and $c_{es}$ remain to be inferred. This greatly simplifies the problem.

Last, but not least, the results presented in this paper hold true for any particular algorithm that implements error correcting graph matching. They may be regarded as a theoretical basis and may be useful for a better understanding of error correcting graph matching in general.

REFERENCES