Graph Edit Distance or Graph Edit Pseudo-Distance?

Francesc Serratosa(✉), Xavier Cortés, and Carlos-Francisco Moreno

Universitat Rovira i Virgili, Tarragona, Catalonia, Spain
{francesc.serratosa,xavier.cortes,
carlsofrancisco.moreno}@urv.cat

Abstract. Graph Edit Distance has been intensively used since its appearance in 1983. This distance is very appropriate if we want to compare a pair of attributed graphs from any domain and obtain not only a distance, but also the best correspondence between nodes of the involved graphs. In this paper, we want to analyse if the Graph Edit Distance can be really considered a distance or a pseudo-distance, since some restrictions of the distance function are not fulfilled. Distinguishing between both cases is important because the use of a distance is a restriction in some methods to return exact instead of approximate results. This occurs, for instance, in some graph retrieval techniques. Experimental validation shows that in most of the cases, it is not appropriate to denominate the Graph Edit Distance as a distance, but a pseudo-distance instead, since the triangle inequality is not fulfilled. Therefore, in these cases, the graph retrieval techniques not always return the optimal graph.

Keywords: Graph Edit Distance · Edit cost · Distance function

1 Introduction

Attributed graphs have been of crucial importance in pattern recognition throughout more than four decades [1, 2] since they have been used to model several kinds of problems. Interesting reviews of techniques and applications are [3–5]. If elements in pattern recognition are modelled through attributed graphs, error-tolerant graph-matching algorithms are needed that aim to compute a matching between nodes of two attributed graphs that minimizes some kind of objective function. To that aim, one of the most widely used methods to evaluate an error correcting graph isomorphism is the Graph Edit Distance [1, 2, 6].

Graph Edit Distance needs two main input parameters, which are the pair of attributed graphs to be compared and also other calibration parameters. These parameters have to be tuned in order to maximise a recognition ratio in a classification scenario or simply to minimise the Hamming distance between a ground-truth correspondence between nodes of both graphs and the obtained correspondence.

This research is supported by projects DPI2013-42458-P, TIN2013-47245-C2-2-R, and by Consejo Nacional de Ciencia y Tecnología (CONACyT México).

A. Robles-Kelly et al. (Eds.): S+SSPR 2016, LNCS 10029, pp. 530–540, 2016.
DOI: 10.1007/978-3-319-49055-7_47

Unfortunately, little research has been done to analyse if really the Graph Edit Distance is a distance or simply a dissimilarity function that could be classified as a pseudo-distance, since some distance restrictions are not fulfilled. Reference [7] is the only paper related on this idea, and it shows in which conditions of these calibration parameters the Graph Edit Distance is really a distance.

The importance of Graph Edit Distance being indeed a distance has an influence on some applications. As an example, in [8–10], authors present methods to retrieve graphs in a database. They suppose that given three graphs, the triangle inequality is fulfilled and thanks to this assumption, some comparisons are not needed to be performed. It turns out that if the Graph Edit Distance is not a distance, then the triangle inequality is not guaranteed, and then some graphs that would have to be explored are not considered, making the methods sub-optimal.

The aim of this paper is to empirically analyse if the distance definition is hold when the recognition ratio is maximised or the obtained correspondence is close to the ground truth. The outline of the paper is as follows; in Sect. 2, we define the attributed graphs and the Graph Edit Distance. In Sects. 3 and 4, we explain the restrictions that a distance needs to fulfil and we relate these restrictions on the specific case of the Graph Edit Distance. In Sects. 5 and 6, we show the experimental validation and conclude the paper.

2 Graphs and Graph Edit Distance

Let Δ_v and Δ_e denote the domains of possible values for attributed vertices and arcs, respectively. An attributed graph (over Δ_v and Δ_e) is defined by a tuple $G = (\Sigma_v, \Sigma_e, \gamma_v, \gamma_e)$, where $\Sigma_v = \{v_k \mid k = 1, \ldots, R\}$ is the set of vertices (or nodes), $\Sigma_e = \{e_{ij} | i, j \in \{1, \ldots, R\}\}$ is the set of edges (or arcs), $\gamma_v : \Sigma_v \to \Delta_v$ assigns attribute values to vertices and $\gamma_e : \Sigma_e \to \Delta_e$ assigns attribute values to edges.

Let $G^p = (\Sigma_v^p, \Sigma_e^p, \gamma_v^p, \gamma_e^p)$ and $G^q = (\Sigma_v^q, \Sigma_e^q, \gamma_v^q, \gamma_e^q)$ be two attributed graphs of order R^p and R^q. To allow maximum flexibility in the matching process, graphs can be extended with null nodes [1] to be of order $R^p + R^q$. We refer to null nodes of G^p and G^q by $\hat{\Sigma}_v^p \subseteq \Sigma_v^p$ and $\hat{\Sigma}_v^q \subseteq \Sigma_v^q$ respectively. Let T be a set of all possible correspondences between two node sets Σ_v^p and Σ_v^q. Correspondence $f^{p,q} : \Sigma_v^p \to \Sigma_v^q$, assigns each node of G^p to only one node of G^q. The correspondence between edges, denoted by $f_e^{p,q}$, is defined accordingly to the correspondence of their terminal nodes.

$$\begin{gathered} f_e^{p,q}\left(e_{ab}^p\right) = e_{ij}^q \Rightarrow f^{p,q}\left(v_a^p\right) = v_i^q \wedge f^{p,q}\left(v_b^p\right) = v_j^q \\ v_a^p, v_b^p \in \Sigma_v^p - \hat{\Sigma}_v^p \text{ and } v_i^q, v_j^q \in \Sigma_v^q - \hat{\Sigma}_v^q \end{gathered} \tag{1}$$

The basic idea behind the Graph Edit Distance is to define a dissimilarity measure between two graphs. This dissimilarity is defined as the minimum amount of distortion required to transform a graph into another. To this end, a number of distortion or edit operations, consisting of insertion, deletion and substitution of both nodes and edges are defined. Then, for every pair of graphs (G^p and G^q), there is a sequence of edit operations that transforms G^p into G^q. In general, several edit paths may exist between two given graphs and to evaluate which edit path is the best, edit cost functions are introduced. The basic idea is to assign a penalty cost to each edit operation according to the amount of distortion that it introduces into the transformation.

Each edit path can be related to an univocal correspondence $f^{p,q} \in T$ between the involved graphs. This way, each edit operation assigns a node of the first graph to a node of the second graph. Deletion and insertion operations are transformed to assignations of a non-null node of the first or second graph to a null node of the second or first graph. Substitutions simply indicate node-to-node assignations. Using this transformation, given two graphs, G^p and G^q, and a correspondence between their nodes, $f^{p,q}$, the graph edit cost is given by [1]:

$$
\begin{gathered}
EditCost(G^p, G^q, f^{p,q}) = \\
\sum_{\substack{v_a^p \in \Sigma_v^p - \hat{\Sigma}_v^p \\ v_i^q \in \Sigma_v^q - \hat{\Sigma}_v^q}} C_{ns}\left(v_a^p, v_i^q\right) + \sum_{\substack{e_{ab}^p \in \Sigma_e^p - \hat{\Sigma}_e^p \\ e_{ij}^q \in \Sigma_e^q - \hat{\Sigma}_e^q}} C_{es}\left(e_{ab}^p, e_{ij}^q\right) + \sum_{\substack{v_a^p \in \Sigma_v^p - \hat{\Sigma}_v^p \\ v_i^q \in \hat{\Sigma}_v^q}} C_{nd}\left(v_a^p, v_i^q\right) + \\
\sum_{\substack{v_a^p \in \hat{\Sigma}_v^p \\ v_i^q \in \Sigma_v^q - \hat{\Sigma}_v^q}} C_{ni}\left(v_a^p, v_i^q\right) + \sum_{\substack{e_{ab}^p \in \Sigma_e^p - \hat{\Sigma}_e^p \\ e_{ij}^q \in \hat{\Sigma}_e^q}} C_{ed}\left(e_{ab}^p, e_{ij}^q\right) + \sum_{\substack{e_{ab}^p \in \hat{\Sigma}_e^p \\ e_{ij}^q \in \Sigma_e^q - \hat{\Sigma}_e^q}} C_{ei}\left(e_{ab}^p, e_{ij}^q\right) \\
\text{being } f^{p,q}\left(v_a^p\right) = v_i^q \text{ and } f_e^{p,q}(e_{ai}^p) = e_{ij}^q
\end{gathered}
\tag{2}
$$

where C_{ns} is the cost of substituting node v_a^p of G^p by node $f^{p,q}\left(v_a^p\right)$ of G^q, C_{nd} is the cost of deleting node v_a^p of G^p and C_{ni} is the cost of inserting node v_i^q of G^q. Equivalently for edges, C_{es} is the cost of substituting edge e_{ab}^p of graph G^p by edge $f_e^{p,q}\left(e_{ab}^p\right)$ of G^q, C_{ed} is the cost of assigning edge e_{ab}^p of G^p to a non-existing edge of G^q and C_{ei} is the cost of assigning edge e_{ab}^q of G^q to a non-existing edge of G^p.

Finally, the Graph Edit Distance is defined as the minimum cost under any correspondence in T:

$$
GED(G^p, G^q) = \min_{f^{p,q} \in T} EditCost(G^p, G^q, f^{p,q}) \tag{3}
$$

Using this definition, the Graph Edit Distance essentially depends on $C_{ns}, C_{nd}, C_{ni}, C_{es}, C_{ed}$ and C_{ei} functions. Several definitions of these functions exist. Table 1 summarises the five different configurations that have been presented so far.

The first options [11–16] are the ones where the whole costs are defined as functions that depend on the involved attributes and also on either learned or general knowledge. Attributes are density functions instead of vectors of attributes. The second option makes the Graph Edit Distance to be directly related to the maximal common sub-graph. That is, in [17], authors demonstrate that computing the Graph Edit Distance is exactly the same than deducting the maximal common sub-graph. In the third

Table 1. Examples of Graph Edit Costs in the literature.

Reference	C_{ns}	C_{nd}	C_{ni}	C_{es}	C_{ed}	C_{ei}
[11–13, 15]	$d_n(v_a^p, v_i^q)$	$f_{nd}(v_a^p)$	$f_{nd}(v_i^q)$	$d_e(e_{ab}^p, e_{ij}^q)$	$f_{ed}(e_{ab}^p)$	$f_{ei}(e_{ij}^q)$
[17]	$0, \infty$	1	1	$0, \infty$	0	0
[18]	$d_n(v_a^p, v_i^q)$	K_n	K_n	$d_e(e_{ab}^p, e_{ij}^q)$	0	0
[1, 19, 20]	$d_n(v_a^p, v_i^q)$	K_n	K_n	$d_e(e_{ab}^p, e_{ij}^q)$	K_e	K_e
[23]	$0, K_{ns}$	K_n	K_n	$0, K_{es}$	K_e	K_e

option, [18], authors assume that the graphs are complete, and a non-existing edge is an edge with a "null" attribute. In this case, the cost of deleting and inserting an edge is encoded in the edge substitution cost. Inserting and deleting nodes have a constant cost, K_n. With this definition, authors describe several classes of costs that Eq. 3 deducts the same correspondence. The fourth option might be the most used one [1, 19, 20]. Substitution costs are defined as distances between vectors of attributes, usually the Euclidean distance. Insertion and deletion costs are constants, K_n and K_e, that have been manually tested or automatically learned [21, 22]. Finally, the last option is used in fingerprint recognition [23]. It is similar to the previous option, except from the substitution costs that are constants. Nodes represent minutiae and edges are the relations between them. If a specific distance between minutiae is lower than a threshold, then a zero is imposed as a substitution cost. Otherwise, this cost takes a constant value K_{ns}. The same happens with the edges that take a constant value K_{es}.

It is worth noting that for all of the cases except for the first one, the insertion and deletion costs on nodes are considered to be the same, K_n. The same happens for edges, K_e. Nevertheless, in the string edit distance, also known as Levenshtein distance [24], insertion and deletion costs might be considered different depending on the application. The most usual application is an automatic writing correction scenario, in which the possibility of missing a character is different than accidentally adding an extra character [25].

The optimal computation of the Graph Edit Distance is usually carried out by means of a tree search algorithm, which explores the space of all possible mappings of the nodes and edges of the first graph to the nodes and edges of the second graph. A widely used method is based on the A* algorithm, for instance [18]. Unfortunately, the computational complexity of this algorithm is exponential in the number of nodes of the involved graphs. This means that the running time may be non-admissible in some applications, even for reasonably small graphs. This is why bipartite graph matching [26–29] has appeared.

3 Restrictions on the Graph Edit Distance

A distance, also called a metric, is a function that defines a dissimilarity between elements of a set, such as x, y or z. The domain is $[0, \infty)$ and it holds the following restrictions for all elements in the set [30]:

$$\begin{aligned}
&(1)\ \text{Non-negativity: } dist(x,y) \geq 0.\\
&(2)\ \text{Identity of indiscernible elements: } dist(x,y) = 0 \Leftrightarrow x = y.\\
&(3)\ \text{Symmetry: } dist(x,y) = dist(y,x)\\
&(4)\ \text{Triangle inequality: } dist(x,y) \leq dist(x,z) + dist(z,y)
\end{aligned} \tag{4}$$

In some cases, there is a need to relax these restrictions and thus the resulting functions are not called distance but pseudo-distance, quasi-distance, meta-distance or semi-distance, depending on which restriction is violated and how it is violated [30]. In the case of the Graph Edit Distance, this relaxation comes from the imposition of a ground truth correspondence. This ground truth is deducted from an independent method and it is not related on the edit costs.

If we wish the Graph Edit Distance to be defined as a true distance function, it is needed to assure the whole edit operations in the ground truth correspondence fulfil the four properties in the following Eq. 5. In these equations, we suppose that the ground truth correspondence is $f^{p,q}$ such that $f^{p,q}\left(v_a^p\right) = v_i^q$ and $f^{p,q}\left(v_b^p\right) = v_j^q$.

$$\begin{aligned}
&(1)\ \text{Non-negativity: } \mathrm{C_{ns}} \geq 0 \text{ and } \mathrm{C_{es}} \geq 0.\\
&(2)\ \text{Identity of indiscernible elements:}\\
&\qquad C_{ns}\left(v_a^p, v_i^q\right) = 0 \Leftrightarrow \gamma_{\mathrm{v}}\left(v_a^p\right) = \gamma_{\mathrm{v}}\left(v_i^q\right)\\
&\qquad C_{es}\left(e_{ab}^p, e_{ij}^q\right) = 0 \Leftrightarrow \gamma_{\mathrm{e}}\left(e_{ab}^p\right) = \gamma_{\mathrm{e}}\left(e_{ij}^q\right)\\
&(3)\ \text{Symmetry:}\\
&\qquad \mathrm{C_{nd}}\left(v_a^p, v_i^q\right) = \mathrm{C_{ni}}\left(v_{a'}^p, v_{i'}^q\right) \Leftrightarrow \gamma_{\mathrm{v}}\left(v_a^p\right) = \gamma_{\mathrm{v}}\left(v_{i'}^q\right)\\
&\text{where } v_a^p \in \Sigma_v^p - \hat{\Sigma}_v^p,\ v_i^q \in \hat{\Sigma}_v^q,\ v_{a'}^p \in \hat{\Sigma}_v^p \text{ and } v_{i'}^q \in \Sigma_v^q - \hat{\Sigma}_v^q\\
&\qquad \mathrm{C_{ed}}\left(e_{ab}^p, e_{ij}^q\right) = \mathrm{C_{ei}}\left(e_{a'b'}^p, e_{i'j'}^q\right) \Leftrightarrow \gamma_{\mathrm{e}}\left(e_{ab}^p\right) = \gamma_{\mathrm{e}}\left(e_{i'j'}^q\right)\\
&\text{where } e_{ab}^p \in \Sigma_e^p - \hat{\Sigma}_e^p,\ e_{ij}^q \in \hat{\Sigma}_e^q,\ e_{a'b'}^p \in \hat{\Sigma}_e^p \text{ and } e_{i'j'}^q \in \Sigma_e^q - \hat{\Sigma}_e^q\\
&(4)\ \text{Triangle inequality:}\\
&\qquad \mathrm{C_{ns}}\left(v_a^p, v_i^q\right) \leq \mathrm{C_{nd}}\left(v_a^p, v_{i'}^q\right) + \mathrm{C_{ni}}\left(v_{a'}^p, v_i^q\right)\\
&\text{where } v_a^p \in \Sigma_v^p - \hat{\Sigma}_v^p,\ v_i^q \in \Sigma_v^q - \hat{\Sigma}_v^q\ v_{i'}^q \in \hat{\Sigma}_v^q \text{ and } v_{a'}^p \in \hat{\Sigma}_v^p\\
&\qquad \mathrm{C_{es}}\left(e_{ab}^p, e_{ij}^q\right) \leq \mathrm{C_{ed}}\left(e_{ab}^p, e_{i'j'}^q\right) + \mathrm{C_{ei}}\left(e_{a'b'}^p, e_{ij}^q\right)\\
&\text{where } e_{ab}^p \in \Sigma_e^p - \hat{\Sigma}_e^p,\ e_{ij}^q \in \Sigma_e^q - \hat{\Sigma}_e^q\ e_{i'j'}^q \in \hat{\Sigma}_e^q \text{ and } e_{a'b'}^p \in \hat{\Sigma}_e^p
\end{aligned} \tag{5}$$

For all cited references, functions in Table 1 are defined as distances, and constants as real positive numbers. For this reason, if the Graph Edit Distance cannot be defined as a true distance, it is due to the relations between these functions and constants. Considering the five options proposed in Table 1, it is really difficult to analyse the first option since being a distance or not depends on the specific distance values. We realise that the second and third ones do not hold the triangle inequality and therefore cannot be considered as distances. The fourth option is a distance only if it is guaranteed that the whole substitution operations in the edit path hold:

$$d_n\left(v_a^p, v_i^q\right) \leq 2 \cdot K_n \text{ and } d_e(e_{ab}^p, e_{ij}^q) \leq 2 \cdot K_e \tag{6}$$

That is, we only have to analyse if the triangle inequality of Eq. 5 is fulfilled. Finally, the last option is almost the same than the third one and it is a true distance if constant costs are defined such that,

$$K_{ns} \leq 2 \cdot K_n \text{ and } K_{es} \leq 2 \cdot K_e \tag{7}$$

Since the fourth option is both the most used and the one that can be defined as distance or not, depending on the costs, from now on, we concretise on this specific case.

4 Defining the Graph Edit Distance as a True Distance

Note that given a pair of graphs and an optimal correspondence (the one that minimise *EditCost* in Eq. 3), we can analyse if the used edit costs make the Graph Edit Distance to be a true distance or not. Moreover, each combination of edit costs generates a different optimal correspondence and a Graph Edit Distance value. For this reason, the problem of knowing which are the edit costs that make the Graph Edit Distance to be a true distance is a chicken egg problem. Given some edit costs, we need to compute the optimal correspondence to deduct if the four distance restrictions are violated (Eq. 5), but to deduct the proper edit costs, we need the optimal correspondence.

To solve this problem, we propose to use a ground truth correspondence. That is, given a pair of attributed graphs, and independently of the edit costs, a human or another method deducts which is the "best" correspondence. Thus, we consider that the Graph Edit Distance is a true distance if the four properties in Eq. 5 are fulfilled assuming $f^{p,q}$ is the ground truth correspondence.

Given an application that involves an attributed graph database of M graphs in which the computation of the Graph Edit Distance is needed, the same edit costs have to be used in the whole process and graphs. Thus, we generalise Eq. 6 considering that we have several graphs and also introducing the ground truth concept. We conclude that the Graph Edit Distance is a true distance given some specific insertion and deletion costs for nodes if the following equation holds,

$$\begin{gathered}\forall\, v_a^p \in \Sigma_v^p - \hat{\Sigma}_v^p \text{ given } p: 1..M \text{ and } a: 1..(\mathrm{R^p} + \mathrm{R^q}) \\ \text{such that } f^{p,q}\left(v_a^p\right) = v_i^q \,\&\, v_i^q \in \Sigma_v^q - \hat{\Sigma}_v^q \\ \text{leads to } d_n\left(v_a^p, v_i^q\right) \leq 2 \cdot K_n \\ \text{being } f^{p,q} \text{ the ground-truth correspondence}\end{gathered} \tag{8}$$

Similarly for the edges,

$$\begin{array}{l} \forall\, e^{p}_{ab} \in \Sigma^{p}_{e} - \hat{\Sigma}^{p}_{e} \text{ given } p:\ 1..M \text{ and } a,b:\ 1..(\mathrm{R^p}+\mathrm{R^q}) \\ \text{such that } f^{\mathrm{p,q}}_{\mathrm{e}}\left(e^{p}_{ab}\right) = e^{q}_{ij}\ \&\ e^{jq}_{i} \in \Sigma^{q}_{e} - \hat{\Sigma}^{q}_{e} \\ \text{leads to } d_e\left(e^{p}_{ab}, e^{q}_{ij}\right) \leq 2 \cdot K_e \end{array} \tag{9}$$

being $f^{\mathrm{p,q}}_{\mathrm{e}}$ the edge correspondence deducted from the ground-truth correspondence $f^{p,q}$.

5 Experimental Validation

We used four graph databases that are organised in registers such that each register is composed of a pair of graphs and a ground truth correspondence between their nodes. These databases were initially used to automatically learn insertion and deletion edit costs in [21, 22], and are publically available in [31]. These databases do not have attributes on the edges and therefore, we only analyse the insertion and deletion costs on nodes. Nonetheless, what can be deducted on nodes could be easily extrapolated to edges. Graphs in the first two databases, *Letter Low* and *Letter High*, represent hand written characters, which nodes have as only attribute the (x,y) position of the junctions of strokes in the character, and edges being the strokes. Graphs in *House Hotel* database and *Tarragona Rotation Zoom* database have been extracted from images. Their nodes represent salient points in the images with their attributes being the features obtained by the point extractor. Edges have been deducted by the Delaunay triangulation.

Table 2 shows the position of the quartiles, the mean and also half of the maximum values of the node substitution costs $d_n\left(v^{p}_{a}, v^{q}_{i}\right)$ given the whole correspondences. Clearly, if we want Eq. 8 to hold, the insertion and deletion costs have to be defined such that $K_n \geq \frac{1}{2}Max$.

Table 2. Average node substitution costs given the ground truth correspondences.

	Q1	Q2	Q3	Mean	½ Max
Letter low	0.08	0.12	0.17	0.20	1.68
Letter high	0.28	0.48	0.71	0.55	1.98
House-Hotel	2.82	4.00	5.29	4.08	5.75
Rotation Zoom	0	0	0.002	0.014	0.5

For the sake of clarification, Fig. 1 shows the histograms of $d_n\left(v^{p}_{a}, v^{q}_{i}\right)$ given all databases with the quartiles and the mean values.

We have used an error-tolerant graph-matching algorithm called Fast Bipartite [27] available in [32] to compute the automatically-deducted correspondence and the distance between the attributed graphs.

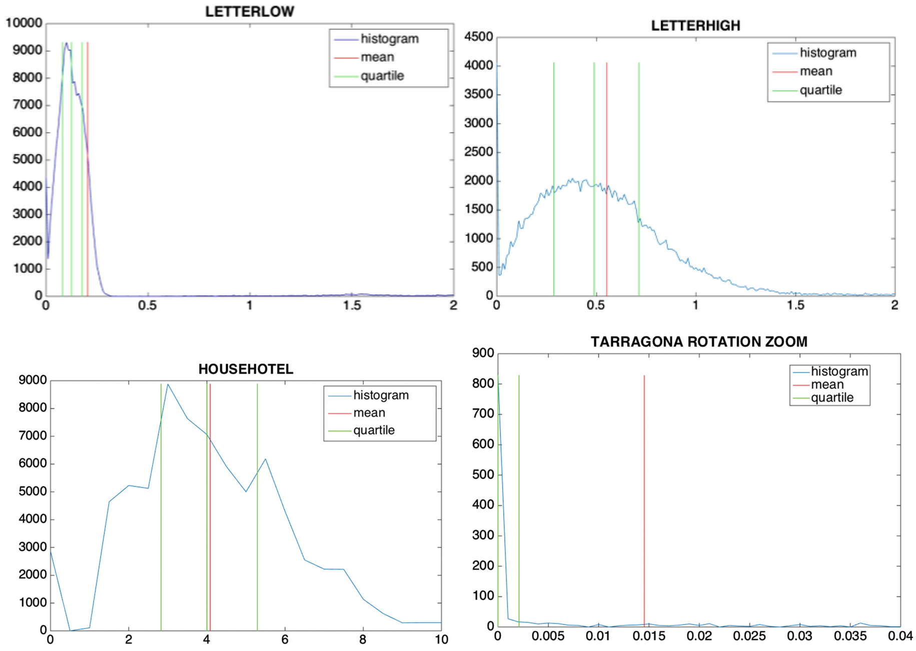

Fig. 1. Histogram of node substitution costs given the ground truth correspondences in the four databases. In green: the first three quartiles. In red: the mean values. (Color figure online)

Table 3 shows the average Hamming distance between the ground-truth correspondence and the automatically obtained correspondence when $K_n = Q1$, $K_n = Q2$, $K_n = Q3$, $K_n = Mean$, and $K_n = \frac{1}{2}Max$. Specific values are shown in Table 2. The Hamming distance is computed as the number of node mappings that are different between both correspondences. Therefore, the lower these values, the better the performance.

Table 3. Average Hamming distance.

	Q1	Q2	Q3	Mean	½ Max
Letter low	0.6	0.6	0.6	0.6	0.7
Letter high	0.9	0.8	0.9	0.9	1.2
House Hotel	0.61	0.71	0.78	0.72	0.80
Rotation Zoom	0.46	0.46	0.27	0.34	0.39

We realise that the lowest Hamming distances are achieved in the positions of the insertion and deletion edit costs such that the triangle inequality is not hold, since these lowest Hamming distances are achieved in the first three quartiles, which are always smaller than $\frac{1}{2}Max$.

Table 4. Classification ratio.

	Q1	Q2	Q3	Mean	½ Max
Letter low	0.97	0.97	0.97	0.97	0.93
Letter high	0.74	0.82	0.83	0.82	0.74
House Hotel	-	-	-	-	-
Rotation Zoom	0.2	0.2	0.35	0.3	0.1

Table 4 shows the classification ratio using the same conditions than the previous experiments. To compute the classification ratio, we have used the reference and test set of each database and the 1-Nearest Neighbour classification algorithm. Recall that the *House Hotel* database does not have classes. It seems as the classification ratio performs similar to the Hamming distance. That is, the best values are achieved when the insertion and deletion edit costs are smaller than $\frac{1}{2}Max$.

The relation of the recognition ratio with the Hamming distance between the ground truth and the obtained correspondences was explored in [22] while learning the edit costs. In that paper, it was empirically demonstrated that decreasing the Hamming distance leads to the recognition ratio to increase. We have validated this dependence again. Moreover, the experimental validation in that paper shows that the presented optimisation method converged to some negative insertion and deletion costs. Again, these values make the Graph Edit Distance not to be a truly defined distance.

Finally, in Table 5 we show the average runtime (in milliseconds) to compute one graph-to-graph comparison. We have used a Windows, I7 and Matlab. We appreciate there is no relation, in general, between the insertion and deletion edit costs and the runtime.

Table 5. Average runtime to match a pair of graphs.

	Q1	Q2	Q3	Mean	½ Max
Letter low	0.61	0.60	0.59	0.58	0.60
Letter high	0.63	0.59	0.59	0.59	0.64
House Hotel	4.8	5.1	5.4	5.1	5.5
Rotation Zoom	15	15	10	8	7

6 Conclusions

Graph Edit Distance is nowadays the most widely used function to compare two graphs and to obtain both a distance and a graph correspondence. This function does not only depend on a pair of graphs, but also on the definition of the insertion and deletion edit costs on nodes and edges. These costs are usually defined as constants, and depending on their definition, we can consider the Graph Edit Distance to be a true distance or not. The fact of not being a true distance can influence on the performance in some applications. Experimental validation has shown that the insertion and deletion costs that obtain the lowest Hamming distances between the ground truth correspondence and the optimal correspondence and also the highest classification ratios are the ones where the triangle inequality does not hold when considering the ground truth correspondence. For instance, the assumption that $EditCost(G^p, G^q, f^{p,q}) \leq \mathrm{GED}(\mathrm{G}^{\mathrm{p}}, \mathrm{G}^{\mathrm{t}}) + \mathrm{GED}(\mathrm{G}^{\mathrm{t}}, \mathrm{G}^{\mathrm{q}})$, where $f^{p,q}$ is the ground truth correspondence, a fact which is commonly assumed on some applications such as graph retrieval or learning processes. This is because in a properly tuned system, $EditCost(G^p, G^q, f^{p,q}) = \mathrm{GED}(\mathrm{G}^{\mathrm{p}}, \mathrm{G}^{\mathrm{q}})$.

References

1. Solé, A., Serratosa, F., Sanfeliu, A.: On the graph edit distance cost: properties and applications. Int. J. Pattern Recogn. Artif. Intell. **26**(5), 1260004 (2012). (21 pages)
2. Sanfeliu, A., Fu, K.-S.: A distance measure between attributed relational graphs for pattern recognition. IEEE Trans. Syst. Man Cybern. **13**(3), 353–362 (1983)
3. Conte, D., Foggia, P., Sansone, C., Vento, M.: Thirty years of graph matching in pattern recognition. Int. J. Pattern Recogn. Artif. Intell. **18**(3), 265–298 (2004)
4. Vento, M.: A long trip in the charming world of graphs for pattern recognition. Pattern Recogn. **48**, 291–301 (2015)
5. Foggia, P., Percannella, G., Vento, M.: Graph matching and learning in pattern recognition in the last 10 years. Int. J. Pattern Recogn. Artif. Intell. **28**, 1450001 (2013)
6. Gao, X., et al.: A survey of graph edit distance. Pattern Anal. Appl. **13**(1), 113–129 (2010)
7. Bunke, H., Allermann, G.: Inexact graph matching for structural pattern recognition. Pattern Recogn. Lett. **1**(4), 245–253 (1983)
8. He, L., et al.: Graph matching for object recognition and recovery. Pattern Recogn. Lett. **37** (7), 1557–1560 (2004)
9. Serratosa, F., Cortés, X., Solé, A.: Component retrieval based on a database of graphs for hand-written electronic-scheme digitalisation. Expert Syst. Appl. **40**, 2493–2502 (2013)
10. Berretti, S., Del Bimbo, A., Vicario, E.: Efficient matching and indexing of graph models in content-based retrieval. IEEE Trans. Pattern Anal. Mach. Intell. **23**(10), 1089–1105 (2001)
11. Wong, A., You, M.: Entropy and distance of random graphs with application to structural pattern recognition. Trans. Pattern Anal. Mach. Intell. **7**(5), 599–609 (1985)
12. Neuhaus, M., Bunke, H.: Automatic learning of cost functions for graph edit distance. Inf. Sci. **177**(1), 239–247 (2006)
13. Serratosa, F., Alquézar, R., Sanfeliu, A.: Function-described graphs for modelling objects represented by attributed graphs. Pattern Recogn. **36**(3), 781–798 (2003)
14. Serratosa, F., Alquézar, R., Sanfeliu, A.: Efficient algorithms for matching attributed graphs and function-described graphs. In: Proceedings of 15th International Conference on Pattern Recognition, ICPR 2000, Barcelona, Spain, vol. 2, pp. 871–876 (2000)
15. Sanfeliu, A., Serratosa, F., Alquézar, R.: Second-order random graphs for modelling sets of attributed graphs and their application to object learning and recognition. Int. J. Pattern Recogn. Artif. Intell. **18**(3), 375–396 (2004)
16. Alquézar, R., Sanfeliu, A., Serratosa, F.: Synthesis of function-described graphs. In: Amin, A., Dori, D., Pudil, P., Freeman, H. (eds.) SSPR'98 and SPR'98. LNCS, vol. 1451, pp. 112–121. Springer, Heidelberg (1998)
17. Bunke, H.: On a relation between graph edit distance and maximum common subgraph. Pattern Recogn. Lett. **18**(8), 689–694 (1998)

18. Bunke, H.: Error correcting graph matching: on the influence of the underlying cost function. Trans. Pattern Anal. Mach. Intell. **21**(9), 917–922 (1999)
19. Caetano, T., et al.: Learning graph matching. Trans. Pattern Anal. Mach. Intell. **31**(6), 1048–1058 (2009)
20. Ferrer, M., Serratosa, F., Riesen, K.: Improving bipartite graph matching by assessing the assignment confidence. Pattern Recogn. Lett. **65**, 29–36 (2015)
21. Cortés, X., Serratosa, F.: Learning graph matching substitution weights based on the ground truth node correspondence. Int. J. Pattern Recogn. Artif. Intell. **30**(2), 1650005 (2016). (22 pages)
22. Cortés, X., Serratosa, F.: Learning graph-matching edit-costs based on the optimality of the Oracle's node correspondences. Pattern Recogn. Lett. **56**, 22–29 (2015)
23. Jain, A.K., Maltoni, D.: Handbook of Fingerprint Recognition. Springer, New York (2003)
24. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions and reversals. Sov. Phys. Dokl. Cybern. Control Theory **10**, 707–710 (1966)
25. Serratosa, F., Sanfeliu, A.: Signatures versus histograms: definitions, distances and algorithms. Pattern Recogn. **39**(5), 921–934 (2006)
26. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. Image Vis. Comput. **27**(7), 950–959 (2009)
27. Serratosa, F.: Fast computation of bipartite graph matching. Pattern Recogn. Lett. **45**, 244–250 (2014)
28. Serratosa, F.: Computation of graph edit distance: reasoning about optimality and speed-up. Image Vis. Comput. **40**, 38–48 (2015)
29. Serratosa, F.: Speeding up fast bipartite graph matching through a new cost matrix. Int. J. Pattern Recogn. Artif. Intell. **29**(2), 1550010 (2015)
30. Arkhangel'skii, A.V., Pontryagin, L.S.: General Topology I: Basic Concepts and Constructions Dimension Theory. Encyclopaedia of Mathematical Sciences. Springer, Heidelberg (1990)
31. http://deim.urv.cat/~francesc.serratosa/databases/
32. http://deim.urv.cat/~francesc.serratosa/SW