An Edit Distance Between Graph Correspondences

Carlos Francisco Moreno-García[1], Francesc Serratosa[2(✉)], and Xiaoyi Jiang[3]

[1] School of Computer Science and Digital Media, The Robert Gordon University, Aberdeen, UK
c.moreno-garcia@rgu.ac.uk
[2] Department of Computer Science and Mathematics, Universitat Rovira i Virgili, Tarragona, Spain
francesc.serratosa@urv.cat
[3] Department of Mathematics and Computer Science, University of Münster, Münster, Germany
xjiang@uni-muenster.de

Abstract. The Hamming Distance has been largely used to calculate the dissimilarity of a pair of correspondences (also known as labellings or matchings) between two structures (i.e. sets of points, strings or graphs). Although it has the advantage of being simple in computation, it does not consider the structures that the correspondences relate. In this paper, we propose a new distance between a pair of graph correspondences based on the concept of the edit distance, called Correspondence Edit Distance. This distance takes into consideration not only the mapped elements of the correspondences, but also the attributes on the nodes and edges of the graphs being mapped. In addition to its definition, we also present an efficient procedure for computing the correspondence edit distance in a special case. In the experimental validation, the results delivered using the Correspondence Edit Distance are contrasted against the ones of the Hamming Distance in a case of finding the weighted means between a pair of graph correspondences.

Keywords: Graph correspondence · Hamming distance · Edit distance · Weighted mean

1 Introduction

A graph correspondence (or simply referred as a correspondence) is defined as a bijective function which designates a set of element-to-element mappings between the nodes of a pair of graphs. It can be generated either manually or automatically, with the purpose of finding the similarity between these two graphs. In the case that a

This research is supported by projectsTIN2016-77836-C2-1-R, ColRobTransp MINECO DPI2016-78957-R AEI/FEDER EU and by Consejo Nacional de Ciencia y Tecnologías (CONACyT México).

P. Foggia et al. (Eds.): GbRPR 2017, LNCS 10310, pp. 232–241, 2017.
DOI: 10.1007/978-3-319-58961-9_21

correspondence is obtained through an automatic method; the process is most commonly done through an optimisation process called error-tolerant graph matching. Several graph matching methods have been proposed in recent years [1–3] and therefore, it is possible to generate more than one correspondence between a single pair of graphs. In these scenarios, it may be interesting to know how different the generated correspondences are with respect to a ground truth correspondence, or also to analyse how different two correspondences are, and thus the requirement of a specifically designed distance between correspondences. So far in literature, the most commonly used distance between correspondences is the Hamming Distance (HD), which measures the number of mappings that are different between two correspondences. This distance has been used either to measure the accuracy of graph matching algorithms [4, 5] or to perform classification [6]. Nonetheless, the HD falls short on truly representing the dissimilarity between a pair of correspondences.

To justify this claim, consider the following toy example. Assume that three separate parties (human experts or automatic systems) deduce respectively three correspondences f^1, f^2 and f^3 between two graphs G and G' as shown in Fig. 1 (numbers in nodes represent their attribute). Notice that if the HD is used to calculate the dissimilarity between these correspondences, the result is $HD(f^1, f^2) = 2$ and $HD(f^1, f^3) = 2$, implying that both f^2 and f^3 are equally dissimilar with respect to f^1. Nonetheless, if we consider the cost of matching nodes on G and G' as the Euclidean distance between the attributes, then it can be seen that $Cost(f^1) = 1+0+1+1 = 3$, $Cost(f^2) = 1+0+1+3 = 5$ and $Cost(f^3) = 6+5+1+1 = 13$. Notice that the HD fails at reflecting that the cost difference between f^1 and f^3 is larger than between f^1 and f^2.

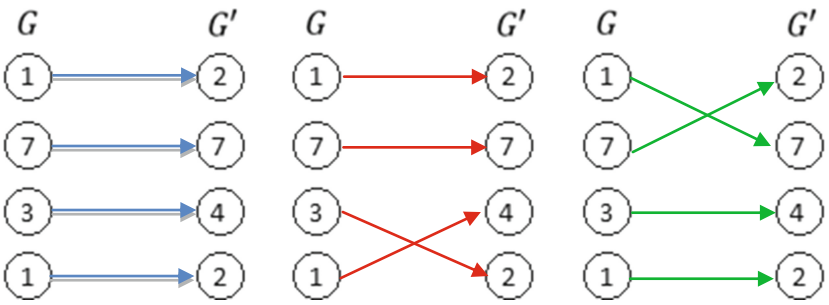

Fig. 1. A first example of two correspondences f^1 and f^2 between two graphs.

The rest of the paper is structured as follows. The next section briefly introduces the basic definitions. In Sect. 3, we present the newly proposed distance between a pair of correspondences. In Sect. 4, we contrast the new distance against the Hamming distance in the case of finding the weighted mean correspondences. Finally, Sect. 5 is reserved for conclusions and further work.

2 Basic Definitions

Let us represent an attributed graph as a four-tuple $G = (V, E, \gamma, \mu)$, where elements $v_i \in \Sigma$ represent the set of nodes, elements $e_i \in E$ represent the set of edges, and γ and μ are functions that assign a set of attributes to each node or edge respectively. Such graph may contain a specific kind of nodes called "null nodes", which are an additional set of nodes which have differentiated attributes (i.e. distinct values to the range of the original attribute values). Moreover, given a pair of graphs $G = (V, E, \gamma, \mu)$, and $G' = (V', E', \gamma', \mu')$, of the same order n (naturally or due to the presence of null nodes), we define the set T of all possible correspondences, such that each correspondence in T maps all nodes of G to nodes in G', $f : V \rightarrow V'$ in a bijective manner. Let f^1 and f^2 denote two arbitrarily selected correspondences in T. We can calculate how similar these two correspondences are through the Hamming distance (HD) between f^1 and f^2

$$HD\left(f^1, f^2\right) = \sum_{i=1}^{n} \left(1 - \partial\left(v'_a, v'_b\right)\right) \tag{1}$$

Where a and b are defined such that $f^1(v_i) = v'_a$ and $f^2(v_i) = v'_b$, and ∂ is the well-known Kronecker Delta function

$$\partial(x, y) = \begin{cases} 0 \, if \, x \neq y \\ 1 \, if \, x = y \end{cases} \tag{2}$$

One of the most widely used frameworks to evaluate the distance between two data structures is the edit distance. This concept has been concretised in the literature as string edit distance [7], tree edit distance [8] and graph edit distance [9–11]. The edit distance is defined as the minimum amount of required operations that transform one object into the other. To this end, several distortions or edit operations, consisting of insertion, deletion and substitution of elements are defined. Edit cost functions are introduced to quantitatively evaluate the edit operations. The basic idea is to assign a penalty cost to each edit operation considering the amount of distortion that it introduces in the transformation. Substitutions simply indicate element-to-element mappings. Deletions are transformed to assignments of a non-null element of the first structure to a null element of the second structure. Insertions are transformed to assignments of a non-null element of the second structure to a null element of the first structure. Given two graphs G and G' and a correspondence f between them, the edit cost would be

$$Graph\_EditCost\left(G, G', f\right) = \sum_{v_i \in V} DV\left(v_i, v'_a\right) + \sum_{e_{ij} \in E} DE\left(e_{ij}, e'_{ab}\right) \tag{3}$$

where $f(v_i) = v'_a$, $f\left(v_j\right) = v'_b$, and DV and DE the distances between nodes and edges respectively. In the case that one of the nodes is a null node, then $DV\left(v_i, v'_a\right) = K_v$, which is the assigned penalty cost for nodes. Similarly for edges, $DE\left(e_{ij}, e'_{ab}\right) = K_e$ in

case that one of the edges is a "null edge" (i.e. non-existing edge). If both nodes and adjacent edges are null, these functions return a zero. In the case that both nodes or both edges are non-null, these functions are application dependent. For instance, if the attributes of the nodes and edges are in $\mathbb{R}^n$, it is usual to apply the Euclidean distance or the weighted Euclidean distance.

Thus, the graph edit distance (GED) is defined as the minimum cost under any bijection in T

$$GED(G, G') = \min_{f \in T} \{Graph\_EditCost(G, G', f)\} \tag{4}$$

Several algorithms have been presented in the literature to compute the GED in an exact or an approximate From this vast pool of options, one of the most widely used algorithms to calculate the GED based on the local substructures [12–14] of the graphs is the bipartite graph matching(BP) framework [15–19].

3 Correspondence Edit Distance

In this section, we present a first step towards a concretisation of an edit distance for correspondences, which we have called Correspondence Edit Distance (CED). In contrast to the HD, the CED aims to consider both the attributes and the local substructure of the nodes mapped by the correspondences. Given G and $G^{'}$ and two correspondences f^1 and f^2 between them, the elements to be considered by the CED must be the elements within the correspondence (mappings) within f^1 and f^2. To that aim, correspondences f^1 and f^2 are defined as sets of mappings $f^1 = \{m_1^1, \ldots, m_i^1, \ldots, m_n^1\}$ and $f^2 = \{m_1^2, \ldots, m_a^2, \ldots, m_n^2\}$, where $m_i^1 = (v_i, f^1(v_i))$ and $m_a^2 = (v_a, f^2(v_a))$. This means that we do not intend to compute the distance between G and $G^{'}$, but rather the distance between f^1 and f^2 while also considering the attributes of graphs G and $G^{'}$.

Figure 2 (left) shows an illustrative example of our proposal using two graphs with no edges, four nodes each (in both graphs, the fourth node is a null node marked as ϕ and ϕ') and two correspondences between them: f^1 (blue) composed of m_1^1, m_2^1, m_3^1 and m_4^1, and f^2 (red) composed of m_1^2, m_2^2, m_3^2 and m_4^2. Notice that m_4^1 and m_4^2 map the null node of G, and thus will be onwards referred as "null mappings". Figure 2 (right) shows a bijective function $h = \{h_1, h_2, h_3, h_4\}$ (green) between f^1 and f^2. Then, the cost of h is calculated as the sum of distances between all mapping-to-mapping relations in h. For this example, the cost yielded by the mappings in h_1 is zero, given the two mappings are the same. For the rest of cases, depending on the attributes and the penalty costs K_v, K_e, the substitution costs would be calculated for the mappings involved.

Notice that for the CED it is important to first define a bijective function $h \in H$ between mappings, where H is the set of all possible bijections between a pair of correspondences. Given such a bijective function h, the edit cost function *Corr_EditCost* is defined in terms of the distances between mappings

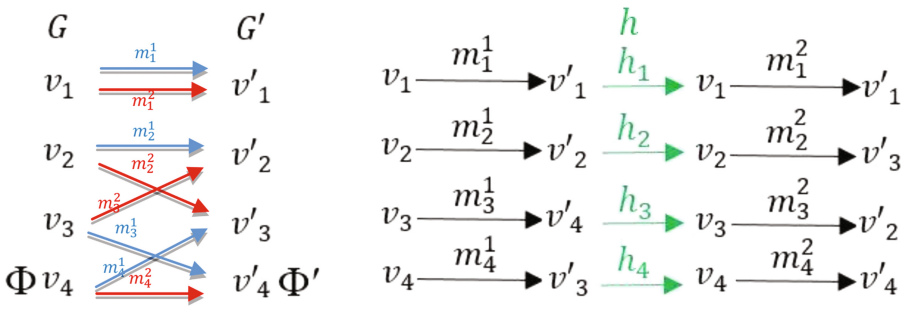

Fig. 2. Left: Two graphs G and $G^{'}$ and two correspondences f^1 and f^2 between them. Right: A bijective function h between f^1 and f^2. (Color figure online)

$$Corr\_EditCost\left(G, G^{'}, f^1, f^2, h\right) = \sum_{m_i^1 \in f^1} DM\left(G, G^{'}, m_i^1, h(m_i^1)\right) \tag{5}$$

where DM is the distance (cost) between two mappings related by h. Then, the CED is defined in a similar way as the GED, that is

$$CED\left(G, G^{'}, f^1, f^2\right) = \min_{h \in H}\left\{Corr\_EditCost\left(G, G^{'}, f^1, f^2, h\right)\right\} \tag{6}$$

Due to the combinatorial nature, the computation of CED is not easy in general. In the following we thus consider a special case which enables an efficient CED computation. If the aim of defining h is to relate the mappings which may resemble the most, then the most straightforward solution is to set all mapping-to-mapping relations in h as $h_j : m_j^1 \rightarrow m_j^2$. Figure 2 shows an example of this solution. In this case, the DM (Eq. 5) becomes the distance between the local substructures DS of the nodes being mapped, that is

$$DM\left(G, G^{'}, m_i^1, m_i^2\right) = DS\left(G^{'}, f^1(v_i), f^2(v_i)\right) \tag{7}$$

Notice that a key difference between *Graph_EditCost* (Eq. 3) and *Corr_EditCost* (Eq. 5) is that in the first case, the distance functions DV and DE are defined between the nodes and adjacent edges of G and G', while in the second case, the distance between local substructures DS is obtained between nodes and adjacent edges on the same graph G'. In other words, to compute DS it is only necessary to compute the distance (cost) between the local substructure being mapped by f^1 in $G^{'}$ and the local substructures being mapped by f^2 in the same $G^{'}$.

For this special case, the computation of the CED is presented in Algorithm 1. If the i^{th} pair of mappings of f^1 and f^2 is equal, then it is excluded from the CED calculation. Moreover, the exclusion also prevails for the cases that two null mappings are paired, or that the two mappings refer to a null node (ϕ) in G'.

Algorithm 1. *Correspondence Edit Distance*
Input: G', f^1, f^2
Output: CED
Begin
CED=0
$for\ j = 1{:}n$
$if\ [f^1(i) \neq f^2(\mathrm{i})] \wedge \neg \{[f^1(i)) = \phi \wedge f^2(i) = \phi] \vee [f^1(i) = \phi \wedge f^2(i) = \phi]\}$
$CED = CED + DS(G', f^1(i), f^2(i))$
$end\ if$
$end\ for$
End Algorithm

4 Validation

To demonstrate in the most practical way that the use of either the HD or the CED produces different outcomes, we propose to use the scenario of calculating the weighted mean between a pair of correspondences. The concept of the weighted mean between two elements x and y has been largely used on data structures such as strings [20], graphs [21] and data clusters [22] to find an element z such that

$$Dist(x, y) = Dist(x, z) + Dist(y, z) \tag{8}$$

In practice, the weighted mean is used to implement methods that approximate towards the generalised median [23] of a set of strings [24–26], graphs [27], data clusters [28] or correspondences [29], as well as to define frameworks such as the

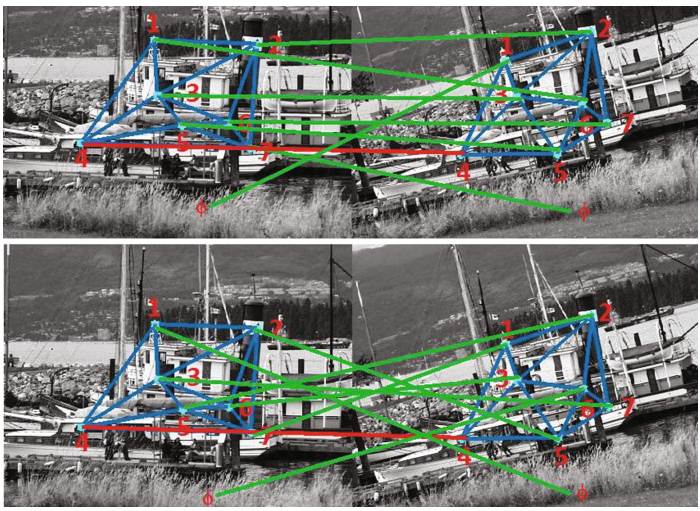

Fig. 3. Correspondences f^1 (top) and f^2 (bottom) between the graphs. (Color figure online)

"consensus" calculation between a set of correspondences, where the aim is to find the most accurate representative prototype from a pool of set-of-points correspondences or graph correspondences [30–33].

Using the first two images of the "BOAT" sequence in the "Tarragona Rotation Zoom" database [6], we randomly select 7 out of the 50 original nodes provided. A node represents a salient point in the image and the normalised SURF features [34] are its attribute. Afterwards, a graph is constructed using these nodes with edges conformed through the Delaunay triangulation. Two correspondences f^1 and f^2 are generated using two different matching algorithms. Notice that since graphs have been enlarged with a null node each to create mutually bijective correspondences, both have a total of eight mappings, with 7 of them being different one from the other (green lines) and one being equal (red line). The result of this process is shown in Fig. 3.

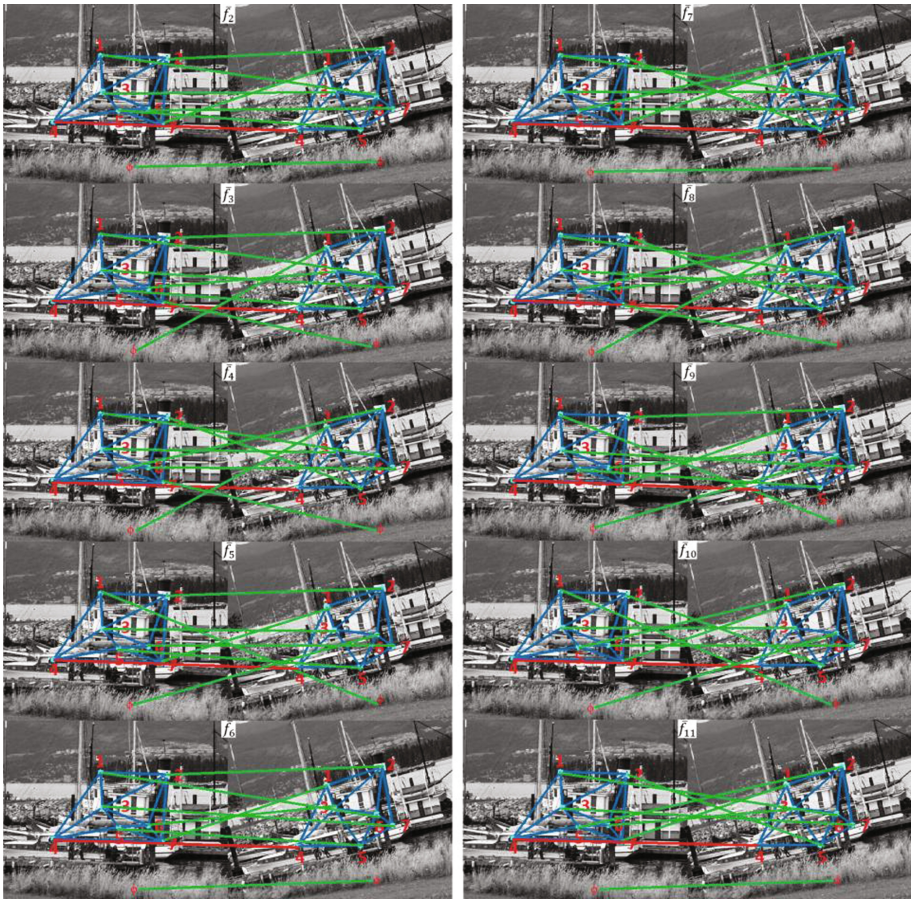

Fig. 4. All weighted means between f^1 and f^2 excluding the first and last one.

To find all weighted mean correspondences, we have implemented an A^* search algorithm which generates all possible correspondences between the two graphs and selects the ones that hold

$$Dist\left(f^1, f^2\right) = Dist\left(f^1, \bar{f}\right) + Dist\left(f^2, \bar{f}\right) \tag{9}$$

Using either HD or CED, the algorithm obtains the same weighted means, but with a different numerical value. For this test, the algorithm found the set of correspondences $\mathcal{W} = \bar{f}_1, \ldots \bar{f}_{12}$, as weighted means, where two of them are the original f_1 and f_2, thus $\bar{f}_1 = f^1$ and $\bar{f}_{12} = f^2$. Figure 4 shows the correspondences $\bar{f}_2, \ldots \bar{f}_{11}$, in $\mathcal{W}$.

Figure 5 shows the distance value using HD (+) or CED (O) ($K_V = K_E = 0.2$) between each of the 12 weighted means towards f_1, normalised by the distance between f_1 and f_2, that is

$$\alpha_i = \frac{Dist(f_1, \bar{f}_i)}{Dist(f_1, f_2)},\ 1 \leq i \leq 12 \tag{10}$$

Notice that using the HD for the weighted means in $\mathcal{W}$ achieves seven different distance values, with repetitions such as $\alpha_3 = \alpha_4 = \alpha_5 = 0.\bar{3}$ and $\alpha_8 = \alpha_9 = \alpha_{10} = 0.\bar{6}$. Conversely, all weighted means in $\mathcal{W}$ deliver different distance values when the CED is used. The main conclusion drawn from this validation is that CED can deliver more diverse distance values than HD since it considers the attributes of the nodes and edges of the graphs being mapped. This characteristic allows to find better distributed weighted means when intending to use algorithms that aim at approximating towards the generalised median.

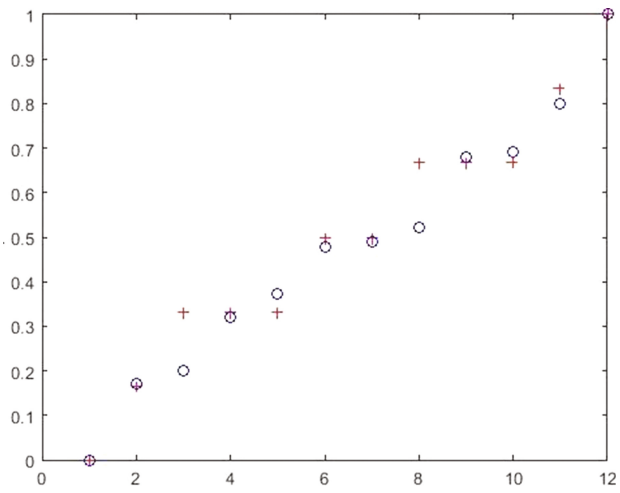

Fig. 5. Normalised distances of the 12 weighted means considering HD (+) and CED (O). The horizontal axis represents the different weighted means $\bar{f}_i$; $1 \leq i \leq 12$.

5 Conclusion and Further Work

In this paper, we present a first approach towards a new distance between a pair of correspondences called Correspondence Edit Distance (CED), based on the well-known concept of the edit distance. In contrast to the classic HD, CED is defined through the attributes of the nodes and their local substructure from the graphs being mapped. This characteristic allows more flexibility and versatility in cases such as obtaining the weighted mean correspondences for their use in algorithms that approach towards the generalised median or the consensus correspondence. In a near future, we intend to present an algorithm to calculate the generalised median correspondence through the use of the CED.

References

1. Conte, D., Foggia, P., Sansone, C., Vento, M.: Thirty years of graph matching. Int. J. Pattern Recognit. Artif. Intell. **18**(3), 265–298 (2004)
2. Foggia, P., Percannella, G., Vento, M.: Graph matching and learning in pattern recognition on the last ten years. Int. J. Pattern Recognit. Artif. Intell. **28**(1), 1450001 (2014). (40 pages)
3. Vento, M.: A long trip in the charming world of graphs for pattern recognition. Pattern Recognit. **48**(2), 291–301 (2015)
4. Caetano, T.S., McAuley, J.J., Cheng, L., Le, Q.V., Smola, A.J.: Learning graph matching. IEEE Trans. Pattern Anal. Mach. Intell. **31**(6), 1048–1058 (2009)
5. Zhou, F., De la Torre, F.: Factorized graph matching. IEEE Trans. Pattern Anal. Mach. Intell. **38**(9), 1774–1789 (2016)
6. Moreno-García, C.F., Cortés, X., Serratosa, F.: A graph repository for learning error-tolerant graph matching. In: Robles-Kelly, A., Loog, M., Biggio, B., Escolano, F., Wilson, R. (eds.) S+SSPR 2016. LNCS, vol. 10029, pp. 519–529. Springer, Cham (2016). doi:10.1007/978-3-319-49055-7_46
7. Wagner, R.A., Fischer, M.J.: The string-to-string correction problem. J. ACM **21**(1), 168–173 (1974)
8. Bille, P.: A survey on tree edit distance and related problems. Theoret. Comput. Sci. **337**(9), 217–239 (2005)
9. 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)
10. Gao, X., Xiao, B., Tao, D., Li, X.: A survey of graph edit distance. Pattern Anal. Appl. **13**(1), 113–129 (2010)
11. Solé-Ribalta, A., Serratosa, F., Sanfeliu, A.: On the graph edit distance cost: properties and applications. Int. J. Pattern Recognit Artif Intell. **26**(5), 1260004 (2012). (24 pages)
12. Cortés, X., Serratosa, F., Moreno-García, C.F.: On the influence of node centralities on graph edit distance for graph classification. In: Liu, C.-L., Luo, B., Kropatsch, W.G., Cheng, J. (eds.) GbRPR 2015. LNCS, vol. 9069, pp. 231–241. Springer, Cham (2015). doi:10.1007/978-3-319-18224-7_23
13. Serratosa, F., Cortés, X.: Graph edit distance: moving from global to local structure to solve the graph-matching problem. Pattern Recognit. Lett. **65**, 204–210 (2015)

14. Cortés, X., Serratosa, F., Riesen, K.: On the relevance of local neighbourhoods for greedy graph edit distance. In: Robles-Kelly, A., Loog, M., Biggio, B., Escolano, F., Wilson, R. (eds.) S+SSPR 2016. LNCS, vol. 10029, pp. 121–131. Springer, Cham (2016). doi:10.1007/978-3-319-49055-7_11
15. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. Image Vis. Comput. **27**(7), 950–959 (2009)
16. Serratosa, F.: Fast computation of bipartite graph matching. Pattern Recognit. Lett. **45**, 244–250 (2014)
17. Serratosa, F.: Computation of graph edit distance: reasoning about optimality and speed-up. Image Vis. Comput. **40**, 38–48 (2015)
18. Serratosa, F.: Speeding up fast bipartite graph matching trough a new cost matrix. Int. J. Pattern Recognit. Artif. Intell. **29**(2), 1550010 (2015). (17 pages)
19. Sanroma, G., Penate-Sanchez, A., Alquezar, R., Serratosa, F., Moreno-Noguer, F., Andrade-Cetto, J., Gonzalez, M.A.: MSClique: multiple structure discovery through the maximum weighted clique problem. PLoS ONE **11**(1), e0145846 (2016)
20. Bunke, H., Jiang, X., Abegglen, K., Kandel, A.: On the weighted mean of a pair of strings. Pattern Anal. Appl. **5**(1), 23–30 (2002)
21. Bunke, H., Günter, S.: Weighted mean of a pair of graphs. Computing **67**(3), 209–224 (2001)
22. Franek, L., Jiang, X., He, C.: Weighted mean of a pair of clusterings. Pattern Anal. Appl. **17** (1), 153–166 (2014)
23. Jiang, X., Münger, A., Bunke, H.: On median graphs: properties, algorithms, and applications. IEEE Trans. Pattern Anal. Mach. Intell. **23**(10), 1144–1151 (2001)
24. Jiang, X., Abegglen, K., Bunke, H., Csirik, J.: Dynamic computation of generalised median strings. Pattern Anal. Appl. **6**(3), 185–193 (2003)
25. Jiang, X., Wentker, J., Ferrer, M.: Generalized median string computation by means of string embedding in vector spaces. Pattern Recognit. Lett. **33**(7), 842–852 (2012)
26. Franek, L., Jiang, X.: Evolutionary weighted mean based framework for generalized median computation with application to strings. In: Gimel'farb, G., et al. (eds.) S+SSPR 2012. LNCS, vol. 7626, pp. 70–78. Springer, Heidelberg (2012). doi:10.1007/978-3-642-34166-3_8
27. Ferrer, M., Valveny, E., Serratosa, F., Riesen, K., Bunke, H.: Generalized median graph computation by means of graph embedding in vector spaces. Pattern Recognit. **43**(4), 1642–1655 (2010)
28. Franek, L., Jiang, X.: Ensemble clustering by means of clustering embedding in vector spaces. Pattern Recognit. **47**(2), 833–842 (2014)
29. Moreno-García, C.F., Serratosa, F., Cortés, X.: Generalised median of a set of correspondences based on the hamming distance. In: Robles-Kelly, A., Loog, M., Biggio, B., Escolano, F., Wilson, R. (eds.) S+SSPR 2016. LNCS, vol. 10029, pp. 507–518. Springer, Cham (2016). doi:10.1007/978-3-319-49055-7_45
30. Moreno-García, C.F., Serratosa, F.: Correspondence consensus of two sets of correspondences through optimisation functions. Pattern Anal. Appl. **20**(1), 201–213 (2015)
31. Moreno-García, C.F., Serratosa, F.: Online learning the consensus of multiple correspondences between sets. Knowl. Based Syst. **90**, 49–57 (2015)
32. Moreno-García, C.F., Serratosa, F.: Consensus of multiple correspondences between sets of elements. Comput. Vis. Image Underst. **142**, 50–64 (2016)
33. Moreno-García, C.F., Serratosa, F.: Obtaining the consensus of multiple correspondences between graphs through online learning. Pattern Recognit. Lett. (2016)
34. Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.: Speeded-up robust features (SURF). Comput. Vis. Image Underst. **110**(3), 346–359 (2008)