Pattern Recognition Letters 45 (2014) 244–250

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

# Fast computation of Bipartite graph matching ☆

Francesc Serratosa *

*Universitat Rovira i Virgili Tarragona, Catalonia, Spain*

ARTICLE INFO

*Article history:*
Received 13 December 2013
Available online 1 May 2014

*Keywords:*
Graph Edit Distance
Bipartite graph matching
Munkres algorithm

ABSTRACT

We present a new algorithm to compute the Graph Edit Distance in a sub-optimal way. We demonstrate that the distance value is exactly the same than the one obtained by the algorithm called Bipartite but with a reduced run time. The only restriction we impose is that the edit costs have to be defined such that the Graph Edit Distance can be really defined as a distance function, that is, the cost of insertion plus deletion of nodes (or arcs) have to be lower or equal than the cost of substitution of nodes (or arcs). Empirical validation shows that higher is the order of the graphs, higher is the obtained Speed up.



## 1. Introduction

Attributed Graphs have been of crucial importance in pattern recognition throughout more than 3 decades [1]. Graphs have been used to model several kinds of problems in some pattern recognition fields such as object recognition [2,3], scene view alignment [4–7], multiple object alignment [8,9], object characterization [10,11], among a great amount of other applications. Interesting reviews of techniques and applications are [12–14]. 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. Unfortunately, the time and space complexity to compute the minimum of these objective functions is very high. For this reason, a lot of research has been done on trying to reduce as much as possible the run time of the graph-matching algorithms through sub-optimal techniques. Since its presentation, Bipartite algorithm [15] has been considered one of the best graph-matching algorithms due to it obtains a sub-optimal distance value almost near to the optimal one but with a considerable decrease on the run time.

This paper presents a variant of the Bipartite algorithm [15] that obtains exactly the same distance value but with a reduced run time. Experimental validation shows a Speed up of 5 on well-known databases and higher Speed up on synthetically generated graphs. In fact, higher is the order of the graphs, higher is also the Speed up of our algorithm. This property is interesting since in the next years, we will see a need on representing the objects (social nets, scenes, proteins...) on larger structures.

The outline of the paper is as follows, in the next section, we define the attributed graphs and the graph-edit distance. On Section 3, we explain how to compute the Graph Edit Distance and we concretize on the Bipartite algorithm. Finally, on Section 4, we present our new method and we schematically show our algorithm. On Section 5, we show the experimental validation and we finish the article with some conclusions.

## 2. Attributed graphs and Graph Edit Distance

In this section, we first define the Attributed Graphs, Cliques and Graph matching and then we explain the Graph Edit Distance.

### 2.1. Definitions

Attributed Graph and Cliques: Let $\mathbf{\Delta_v}$ and $\mathbf{\Delta_e}$ denote the domains of possible values for attributed vertices and arcs, respectively. An attributed graph (over $\mathbf{\Delta_v}$ and $\mathbf{\Delta_e}$) is defined by a tuple $\mathbf{G} = (\mathbf{\Sigma_v}, \mathbf{\Sigma_e}, \gamma_\mathbf{v}, \gamma_\mathbf{e})$, where $\mathbf{\Sigma_v} = \{\mathbf{v_a}|\mathbf{a} = 1, \ldots, \mathbf{n}\}$ is the set of vertices (or nodes), $\mathbf{\Sigma_e} = \{\mathbf{e_{ab}}|\mathbf{a}, \mathbf{b} \in 1, \ldots, \mathbf{n}\}$ is the set of arcs (or edges), $\gamma_\mathbf{v} : \mathbf{\Sigma_v} \to \mathbf{\Delta_v}$ assigns attribute values to vertices and $\gamma_\mathbf{e} : \mathbf{\Sigma_e} \to \mathbf{\Delta_e}$ assigns attribute values to arcs. The order of graph $\mathbf{G}$ is $\mathbf{n}$.

We define a clique $\mathbf{K_a}$ on an attributed graph $\mathbf{G}$ as a local structure composed of a node and its outgoing edges $\mathbf{K_a} = (\mathbf{v_a}, \{\mathbf{e_{ab}}|\mathbf{b} \in 1, \ldots, \mathbf{n}\}, \gamma_\mathbf{v}, \gamma_\mathbf{e})$.

Error correcting graph isomorphism between graphs: 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 initial order n and m. To allow maximum flexibility in the matching process, graphs are extended with null nodes [16] to be of order $n + m$. We will 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. We assume null nodes have

☆ This paper has been recommended for acceptance by R. Davies.

* Tel.: +34 558507; fax: +34 977559710.
*E-mail address:* francesc.serratosa@urv.cat

http://dx.doi.org/10.1016/j.patrec.2014.04.015
0167-8655/

indices $a \in [n+1,\ldots,n+m\}$ and $i \in [m+1,\ldots,n+m\}$ for graphs $G^p$ and $G^q$, respectively. Let T be a set of all possible bijections between two vertex sets $\Sigma_v^p$ and $\Sigma_v^q$. Bijection $f^{p,q}: \Sigma_v^p \rightarrow \Sigma_v^q$, assigns one vertex of $G^p$ to only one vertex of $G^q$. The bijection between arcs, denoted by $f_e^{p,q}$, is defined accordingly to the bijection of their terminal nodes. In other words:

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

We define the non-existent or null edges by $\hat{\Sigma}_e^p \subseteq \Sigma_e^p$ and $\hat{\Sigma}_e^q \subseteq \Sigma_e^q$.

### 2.2. Graph Edit Distance between two graphs

Graph Edit Distance [17–19] is the most used method to solve the error-tolerant graph matching. It is based on defining a distance between attributed graphs through the minimum modifications that are required to transform one attributed graph into the other. To do so, it is needed to define these modifications, which are called edit operations. Basically, six different edit operations have been defined: insertion, deletion and substitution of both nodes and edges. In this way, for every pair of attributed graphs ($G^p$ and $G^q$), there is an edit path $editPath(G^p, G^q) = (\varepsilon_1, \ldots, \varepsilon_k)$ (where each $\varepsilon_i$ denotes an edit operation) that transforms one graph into the other. Given two graphs, there is a set of edit paths, which we name $\vartheta$, that each of them transforms one of the graphs into the other. Fig. 1 shows the edit path that transforms $G^p$ into $G^q$. It is composed of the following 5 edit operations: Delete edge, Delete node, Insert node, Insert edge and Substitute Node. The substitution operation is needed since the attributes of both nodes are different.

Edit cost functions have been introduced to quantitatively evaluate which edit path is the best. The aim of these functions is to assign a penalty cost to each edit operation according to the amount of modification that it introduces in the transformation sequence.

Given two attributed graphs to be compared, we can define a graph bijection $f^{p,q} \in T$ between them and also we can relate this bijection with the edit path, $editPath(G^p, G^q) \in \vartheta$. To do so, we can assume the edit operation called Substitution simply represents node-to-node assignations. Moreover, the edit operations Deletion and Insertion are transformed to mappings of non-null nodes of the first or second graph to null nodes of the second or first graph. Using this transformation, given two graphs, $G^p$ and $G^q$, and a bijection between their nodes, $f^{p,q}$, the graph edit cost is given by (Definition 7 of [20]):

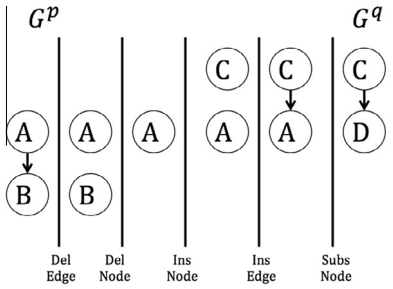


**Fig. 1.** One of the edit paths that transforms $G^p$ into $G^q$.

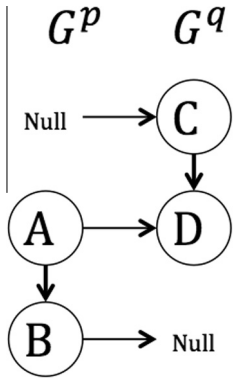


**Fig. 2.** Labelling $f^{p,q}$ between $G^p$ and $G^q$ given the edit path of Fig. 2.

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

Using this definition, the Graph Edit Distance directly depends on parameters or functions $C_{vs}$, $C_{vd}$, $C_{vi}$, $C_{es}$, $C_{ed}$ and $C_{ei}$. Several definitions of these functions exist, if we focus first on the definition of functions $C_{vs}$ and $C_{es}$, the most common approaches are the following. The first and simplest approach considers $C_{vs}(v_a^p, v_i^q) = K_{vs}$ if $dist(\gamma_v^p(v_a^p), \gamma_v^q(v_i^q)) > Threshold$ otherwise $C_{vs} = 0$, $dist(\cdot)$ is defined as a distance function over the domain of the attributes. Specific examples of this cost can be found in fingerprint verification [21] where $C_{vs} \in \{0, 1\}$ or in [20,22]. The second and most frequently used approach corresponds to the case where $C_{vs}(v_a^p, v_b^q, \theta_v) \in \mathbb{R}$. In this case, node substitution cost depends on the attributes of the nodes and possibly on some other parameters $\theta_v$ as shown in [23,24,4], among others. Similar approaches can be used to define $C_{es}$. With regard to $C_{vd}, C_{vi}, C_{ed}$ and $C_{ei}$, these functions usually simply assign a constant cost. However, they can also depend on node or edge attributes [16,25,26].

$$EditCost(G^p, G^q, f^{p,q}) = \sum_{\substack{v_a^p \in \Sigma_v^p - \hat{\Sigma}_v^p \\ v_i^q \in \Sigma_v^q - \hat{\Sigma}_v^q}} C_{vs}(v_a^p, v_i^q) + \sum_{\substack{v_a^p \in \Sigma_v^p - \hat{\Sigma}_v^p \\ v_i^q \in \hat{\Sigma}_v^q}} C_{vd}(v_a^p, v_i^q) + \sum_{\substack{v_a^p \in \hat{\Sigma}_v^p \\ v_i^q \in \Sigma_v^q - \hat{\Sigma}_v^q}} C_{vi}(v_a^p, v_i^q) + \sum_{\substack{e_{ab}^p \in \Sigma_e^p - \hat{\Sigma}_e^p \\ e_{ij}^q \in \Sigma_e^q - \hat{\Sigma}_e^q}} C_{es}(e_{ab}^p, e_{ij}^q) + \sum_{\substack{e_{ab}^p \in \Sigma_e^p - \hat{\Sigma}_e^p \\ e_{ij}^q \in \hat{\Sigma}_e^q}} C_{ed}(e_{ab}^p, e_{ij}^q) + \sum_{\substack{e_{ab}^p \in \hat{\Sigma}_e^p \\ e_{ij}^q \in \Sigma_e^q - \hat{\Sigma}_e^q}} C_{ei}(e_{ab}^p, e_{ij}^q)$$
$$f^{p,q}(v_a^p) = v_i^q \text{ and } f_e^{p,q}(e_{ai}^p) = e_{ij}^q \quad (2)$$

where the edit costs are: $C_{vs}$: Cost of substituting node $v_a^p$ of $G^p$ for node $f^{p,q}(v_a^p)$ of $G^q$. $C_{vd}$: Cost of deleting node $v_a^p$ of $G^p$. $C_{vi}$: Cost of inserting node $v_i^q$ of $G^q$. And for edges, $C_{es}$: Cost of substituting edge $e_{ab}^p$ of graph $G^p$ for edge $f_e^{p,q}(e_{ab}^p)$ of $G^q$. $C_{ed}$: Cost of assigning edge $e_{ab}^p$ of $G^p$ to a non-existing edge of $G^q$. $C_{ei}$: Cost of assigning edge $e_{ab}^q$ of $G^q$ to a non-existing edge of $G^p$. The cost of mapping two null nodes or two null arcs is always defined to be zero. For this reason, we have not considered this case in the equation. Fig. 2 shows the obtained labelling given the edit path presented in Fig. 1. The cost of this labelling is: $EditCost(G^p, G^q, f^{p,q}) = C_{ed} + C_{vd} + C_{vi} + C_{ei} + C_{vs}$.

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

We say the optimal bijection, $f^{p,q*}$, is the one that obtains the minimum cost,

$$f^{p,q*} = \text{argmin}_{f^{p,q} \in T} EditCost(G^p, G^q, f^{p,q}) \quad (4)$$

We define the distance and the optimal bijection between two cliques in a similar way as the distance between two graphs since they are local structures of graphs. We name the cost of substituting clique $K_a^p$ by $K_i^q$ as $C_{a,i}$. The cost of deleting clique $K_a^p$ as $C_{a,\varepsilon}$ and the cost of inserting clique $K_i^q$ as $C_{\varepsilon,i}$.