Pattern Recognition 72 (2017) 266–284

Contents lists available at ScienceDirect

Pattern Recognition

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

Fuzzy generalized median graphs computation: Application to content-based document retrieval

Ramzi Chaieb[a,e], Karim Kalti[b,e], Muhammad Muzzamil Luqman[c], Mickaël Coustaty[c], Jean-Marc Ogier[c], Najoua Essoukri Ben Amara[d,e,*]

[a] National Engineering School of Tunis, Tunis El Manar University, Tunis, Tunisia
[b] Faculty of Sciences of Monastir, Monastir University, Monastir, Tunisia
[c] L3i Laboratory, University of La Rochelle, La Rochelle, France
[d] National Engineering School of Sousse, Sousse University, Sousse, Tunisia
[e] LATIS – Laboratory of Advanced Technology and Intelligent Systems, ENISo, Sousse University, Sousse, Tunisia

ARTICLE INFO

Article history:
Received 28 November 2016
Revised 20 April 2017
Accepted 27 July 2017
Available online 29 July 2017

Keywords:
Fuzzy attributed relational graph
Graph embedding
Fuzzy set median graph
Fuzzy generalized median graph
Similarity measure
Document image retrieval

ABSTRACT

Fuzzy median graph is an important new concept that can represent a set of fuzzy graphs by a representative fuzzy graph prototype. However, the computation of a fuzzy median graph remains a computationally expensive task. In this paper, we propose a new approximate algorithm for the computation of the Fuzzy Generalized Median Graph (FGMG) based on Fuzzy Attributed Relational Graph (FARG) embedding in a suitable vector space in order to capture the maximum information in graphs and to improve the accuracy and speed of document image retrieval processing. In this study, we focus on the application of FGMGs to the Content-based Document Retrieval (CBDR) problem. Experiments on real and synthetic databases containing a large number of FARGs with large sizes show that a CBDR using the FGMG as a dataset representative yields better results than an exhaustive and sequential retrieval in terms of gains in accuracy and time processing.

1. Introduction

During the recent years, the widespread use of digital devices (smartphones, cameras, tablet computers, etc.) and the tremendous growth in digitizing various collections of documents (invoices, books, historical documents, etc.) have resulted in the creation of large databases of document images. Browsing and retrieving from such large-scale document images is more than ever a challenging problem in data mining research. There is still a growing need for efficient and fast methods for searching information in these huge databases. Traditional approaches for Document Image Retrieval (DIR) are text-based [1–3]. However, fully text-based approaches are not practical enough for DIR applications due to many limitations such as time consumption, subjectivity of the annotator, ambiguity (same words but not in the same place in a document image), etc. Therefore, Content-based Document Retrieval (CBDR) approaches have been proposed to overcome the shortcomings of the text-based retrieval [4–10]. Most document image content representation methods are based on the vector space model of information. One of the most important advantages of this representation model is the computation time and memory consumption. However, this popular method of document representation does not capture important structural information, such as the location of an object within the document image or the spatial relationships between document image regions. Also, vector based representations suffer from the constraint of fixed dimensionality of feature vectors [11]. Therefore, structural signature represents some interesting alternatives to construct faith documents representation, through the use of document image regions and their topological relationships. Indeed, graph-based document image representation model can preserve document image structural and spatial information and furthermore it doesn't require a predefined size of the representation. It provides a rich and holistic description of the layout and content of the analyzed document images. It was shown to outperform the traditional vector representation for several applications in many fields [11,12]. However, graph-based representations are usually computationally more expensive than vector-based representations as they require exponential time and space due to the NP completeness of the problem. For further read-

* Corresponding author.
E-mail addresses: ramzi.chaieb@hotmail.com (R. Chaieb), karim.kalti@gmail.com (K. Kalti), muhammad_muzzamil.luqman@univ-lr.fr (M.M. Luqman), mickael.coustaty@univ-lr.fr (M. Coustaty), jean-marc.ogier@univ-lr.fr (J.-M. Ogier), najoua.benamara@eniso.rnu.tn, benamaranajwa@gmail.com (N. Essoukri Ben Amara).

http://dx.doi.org/10.1016/j.patcog.2017.07.030
0031-3203/

ing on graph-based representations and applications we refer the interested reader to [11,12].

Due to many particularities of document images (for example: noise and degradation, overlapping layouts, presence of handwriting, etc.), eventual segmentation errors may occur after the segmentation of document regions. Segmentation results are highly dependent on the performance of the segmentation algorithm which might be unstable over various document images. Therefore, the use of fuzzy graph-based description (instead of a classic graph-based representation which is a rigid description) could be helpful to add flexibility against these errors. Moreover, a document image content may generally be submitted to variations that introduce some vagueness or uncertainty in the way to describe the information. Therefore, representing the content by fuzzy graphs allows to capture the maximum information from a document image with a certain error-tolerance. Structural and visual features should be represented by fuzzy concepts, such as "Near" and "Far", "Big" and "Small", etc. These concepts are described with the use of the fuzzy set theory. A crisp description can be represented as a special case of a fuzzy description. For further reading on fuzzy graphs and its applications we refer the interested reader to [13–15].

Generally, the CBDR process is composed of three main phases: extracting features, structuring feature space and retrieving. The first two phases are usually performed off-line. In this paper, we assume that all document images are represented by Fuzzy Attributed Relational Graphs (FARGs). We will focus on the second phase which aims at organizing the input fuzzy graphs into an efficient data structure in order to improve and accelerate retrieval results. The proposed approach avoids the sequential search in a FARG database by direct access to a reduced set containing the FARGs most similar to a query FARG. Therefore, we used the concept of median graph for the structuring phase of a CBDR system. Median graph aims at representing an input set of graphs with no constraint on the set size. It is frequently used to indicate the graph that best captures the information presented in all input graphs. In other words, the median graph of a given set of graphs is the graph that minimizes the sum of distances to all other graphs in this set. Median graphs have wide-spread applications in diverse fields such as pattern recognition, classification, image analysis, etc. [16–19]. In recent years, more and more research efforts have been devoted to median graph problem. A brief review of some strategies for median graph construction will be outlined in the related work section.

In this study, we propose a new algorithm for the computation of the Fuzzy Generalized Median Graph (FGMG) in order to improve the speed and accuracy of retrieval processing. The first contribution of this paper consists in representing query and database document images by FARGs. One motivation for the FARG representation is to keep advantages from both the graph and the vector domains (power of representation of graphs and easiness manipulation of the vector representations). Besides, FARGs allow to represent document images with a fuzzy approach which is similar to human perception [9,10,13]. FARGs provide both syntactic and semantic information. Syntactic information is held by the layout of the graph (nodes and edges), while semantic information is expressed by attributes associated to nodes and edges in the graph. Describing images by exploiting these two informations will reduce the semantic gap between low-level features and high-level concepts and therefore improve the retrieval results. Also, the FARG representation is very useful for reducing the effect of possible segmentation errors which may be occurred after the segmentation phase [13]. The second contribution of this paper is a new algorithm for the computation of the FGMG based on FARG embedding into a vector space. In the context of dealing with a large mass of document image datasets, it is not trivial to perform an exhaustive and sequential comparison of the query with all document images in the database due to the high computational complexity requirements. Thus, we have developed a new FGMG computation algorithm in order to contribute to the structuration of the FARG space. The third contribution of this paper is a new FARG embedding method in order to reduce the computation time of the FGMG. FGMG computation is based on computing the distance between every pair of FARGs. Since all the possible combinations of FARGs need to be explored, the computation of the FGMG will be therefore exponential in the number and size of input FARGs. Embedding FARGs into a vector space solves this problem since FARGs are represented by feature vectors [14,20,21,22]. Thanks to this new method, we are able to keep power of representation of FARGs while manipulating the vector representation of the FARGs.

The remaining of this paper is organized as follows. The next section gives some preliminaries for the new FGMG algorithm and a detailed presentation of the concept of the median graph. Section 3 presents a literature review of related work. Section 4 describes the FGMG computation algorithm in detail. Section 5 introduces an example of application of FGMGs to the CBDR problem. Experimental results are evaluated and discussed in Section 6. Finally, concluding remarks and future work are given in the last section of this paper.

2. Fuzzy median graph theory

In this section, we present some preliminaries for the new FGMG algorithm, including the definition of the Fuzzy Attributed Relational Graph (FARG) and those of a Fuzzy Set Median Graph (FSMG) and of a Fuzzy Generalized Median Graph (FGMG).

2.1. Fuzzy attributed relational graph

A FARG is a graph whose nodes (also called vertices) and edges (also called arcs) are both represented with fuzzy attributes. Given a finite fuzzy attribute set A, a FARG G can be defined as a quadruple (N, E, μ, υ) where N is a non-empty finite set of nodes, $E \subseteq N \times N$ is the set of edges, $\mu: N \rightarrow A$ is a function that associates a fuzzy attribute value in A to each node, and $\upsilon: E \rightarrow A$ is a function that associates a fuzzy attribute value in A to each edge. In this paper, FARG nodes represent document image regions and FARG edges represent spatial relationships between the regions.

2.2. Fuzzy median graph

Given a set of graphs, the median graph is frequently used to indicate the graph that best represents the set. In other words, the median graph is the graph that best captures the information presented in all graphs. Intuitively, the median graph is located in the center of the given graph set. Basically, two different definitions for median graphs have been presented: the set median graph and the generalized median graph. One difference between them is in the search space of graphs where the median is looked for. The generalized median graph is usually constructed from a larger set of graphs.

2.2.1. Fuzzy set median graph

Let U be the set of FARGs that can be constructed using a given set of attributes A. Given a set $S = \{G_1, G_2, \ldots, G_i\} \subseteq U$, the Fuzzy Set Median Graph (FSMG) of S is defined as the FARG $\hat{G} \in U$ that minimizes the Sum Of Distances (SOD) to all FARGs in S.

$$\hat{G} = \underset{G \in S}{\operatorname{argmin}} \sum_{i=1}^{|S|} d(G, G_i) = \underset{G \in S}{\operatorname{argmin}}\, SOD(G) \tag{1}$$

where $d(G, G_i)$ denotes a distance or a dissimilarity measure between a candidate median FARG G and a FARG $G_i \in S$.

2.2.2. Fuzzy generalized median graph

The FSMG is only computed from the set $S \subseteq U$ of FARGs in question. Another alternative to represent all FARGs in U is to compute the FGMG. It is defined as follows:

$$\bar{G} = \underset{G \in U}{\operatorname{argmin}} \sum_{i=1}^{|S|} d(G, G_i) = \underset{G \in U}{\operatorname{argmin}}\, SOD(G) \tag{2}$$

The FGMG is computed from a larger set of potential fuzzy median graphs. Therefore, it is usually a better representative of a set of FARGs than the FSMG [23–27] and gives a more accurate description of the FARG set. However, the computational complexity of finding the FGMG is significantly higher than that of the FSMG because the search space is extended to the whole set U.

2.3. Distance between two FARGs

As shown in Eqs. (1) and (2), a distance measure $d(G,G_i)$ between a candidate median FARG G and each FARG $G_i \in S$ must be computed. The concept of FARG similarity is of great importance in the computation of FSMG and FGMG. Different distance measures may be used to calculate the SOD for the fuzzy median graph. In this study, we improved the tree-based graph matching distance that we proposed in [13] to measure the similarity between two FARGs. Basically, the tree-based graph matching distance $d(G_1,G_2)$ between two FARGs G_1 and G_2 takes into account the total node and edge distances between G_1 and G_2 and their associated weights. The total node and edge distances are computed using Eqs. (3) and (4), respectively.

$$TND = \sum_{i=1}^{N} d_{n_i}(q_n, s_n) \tag{3}$$

$$TED = \sum_{i=1}^{W} d_{e_i}(q_e, s_e) \tag{4}$$

where $d_{n_i}(q_n, s_n)$ and $d_{e_i}(q_e, s_e)$ are the distance between a pair of nodes and the distance between a pair of edges, respectively. N and W are the number of matched node pairs and the number of matched edge pairs, respectively.

The total matching distance $d(G_1,G_2)$ between G_1 and G_2 is defined as follows:

$$d(G_1, G_2) = w_{G_1} \times TND \; + \; w_{G_2} \times TED \tag{5}$$

where w_{G_1} and w_{G_2} are properly selected weights of G_1 and G_2. Depending on the values of w_{G_1} and w_{G_2} different matching strategies can be applied:

- $w_{G_1} > w_{G_2}$: Matching is done by focusing on regions content more than their position in the document image.
- $w_{G_1} < w_{G_2}$: Matching is done by focusing on structural similarities rather than on visual similarities.
- $w_{G_1} = w_{G_2}$: Visual and structural similarities are taken into consideration for matching with the same importance degree.

3. Related work

Recently, the median graph approaches have gained great attention of many researchers in document image retrieval field. For instance, Hlaoui and Wang [16] proposed an approximate algorithm for computing the generalized median graph from a set of graphs. The proposed algorithm allows the extension of standard algorithms such as k-means to graph clustering in order to bridge the gap between statistical and structural representations. The experimental evaluation on a synthetic image database shows the effective use of the proposed algorithm in correctly classifying graphs into sets of clusters. However, the proposed algorithm still suffers from the large complexity of its preparation step which includes the determination of the size of the median graph and the reduction of the set of possible nodes used in the search step. In addition, all experiments were done with synthetic data and graphs of a relatively small size (5–10 nodes per graph) and with a low number of labels. Jouili and Tabbone [17] introduced a prototype-based clustering algorithm to cluster a set of graphs. The proposed algorithm detects automatically the number of classes in the graph database using the concept of the median graph and a given threshold. Besides, the proposed algorithm allows the multi-assignment of one graph, i.e. one graph can be assigned to more than one cluster. In subsequent work, Jouili et al. [18] proposed a graph clustering algorithm by adapting the mean-shift algorithm into the domain of graphs. They used the notion of a set median and a generalized median graph to implement the shifting operation instead of the mean in the classical mean-shift clustering. The proposed algorithm is experimentally evaluated on three datasets using two validation indices and a comparison with the k-means algorithm is provided. It is a deterministic and non-parametric algorithm that does not require a-priori knowledge of the number of clusters. However, it largely depends on the bandwidth selection during clustering. Ferrer et al. [28] proposed an exact algorithm using a distance based on the maximum common subgraph for the exact median graph computation. They have applied the proposed exact algorithm to a set of real data using a database of graphs representing molecules. The obtained results show that the proposed algorithm outperforms the previous existing exact algorithms using synthetic data. Nevertheless, the exact computation of the median graph has an exponential complexity. In addition, the application of the exact algorithm to real databases is still limited. To overcome these drawbacks, the authors [23,24,25,29] proposed iterative algorithms for approximate median graph computation. For instance, they used spectral-based approaches [19,30,31] and genetic algorithms [29] to solve the problem. Although the spectral-based approaches allow to synthesize an approximation of the median graph with linear complexity with respect to the number of graphs, these approaches suffer from two main limitations. The first limitation is that the number of nodes of the graphs must be equal. The second limitation of these approaches is that they can only be used with weighted graphs. The authors also applied heuristic functions in order to reduce the complexity of the graph distance computation and the size of the search space. In [25,32,33], they used graph embedding methods in order to keep the representational power of graphs while being able to operate with vector space representations. Further, a comparative study between the set median and the generalized median approaches is made. Using two standard clustering performance measures (the Rand index and the Dunn index), the experimental results show that the generalized median graph approach yields better performance than the set median graph approach. The proposed algorithms provide some remarkable improvements over other exact algorithms for the median graph computation. However, all these algorithms can only be applied to restricted sets of graphs, regarding either the type or the size of the graphs. In a graph classification context, some recent studies have been dedicated to the computing of graph prototypes. As an example, Raveaux et al. [26] proposed an approach for graph prototypes extraction using a graph based genetic algorithm. The experiments have been carried out using three real databases and one synthetic database. Experimental results show that the generalized median graphs outperform the set median graphs. In 2015, Musmanno and Ribeiro [27] proposed two heuristics for solving the generalized median graph problem: a greedy adaptive algorithm and a Greedy Randomized Adaptive Search Procedure (GRASP) heuristic. They used the graph edit distance to measure the similarity between two

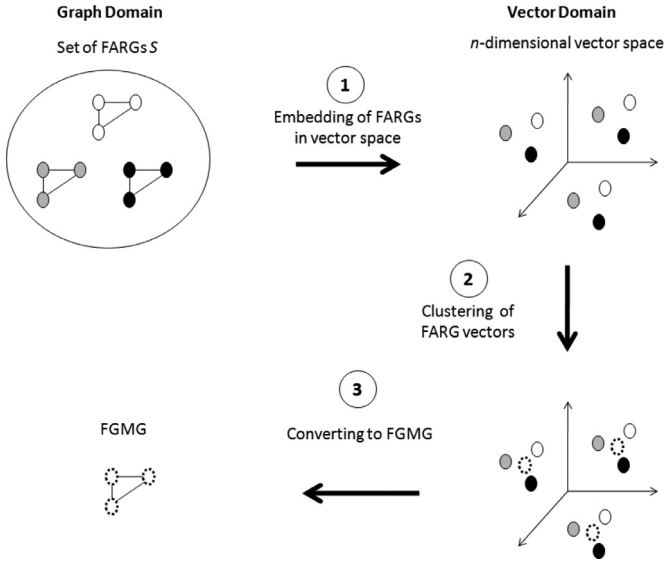

Fig. 1. The three main steps of the proposed FGMG computation algorithm.

graphs. Experimental results demonstrate that the generalized median graph gives better results than those provided by the median graph.

As a conclusion, the generalized median graph usually yields better performance than the set median graph and it can be used in a large number of applications such as classification, retrieval and pattern recognition. While the generalized median graph has a large number of advantages, its main disadvantage is that its computation is exponential both in the number of input graphs and their size. A number of exact algorithms for the median graph computation have been reported in the past [28]. The major drawback of such algorithms is their computational complexity which is exponential in the number of nodes of the involved graphs. Consequently, exact algorithms cannot deal with large graphs. Therefore, their application has been restricted to small problems involving sets of graphs with no more than 25 vertices altogether. As the computational cost of these exact algorithms is very high, a set of approximate algorithms have also been proposed in the past based on different approaches such as spectral-based approaches [19,30,31], genetic algorithms [26,29] and greedy-based algorithms [27]. Such approximate algorithms apply some heuristics in order to reduce the complexity of the graph distance computation and the size of the search space. However, all these algorithms are very limited in their application. They can only be applied to restricted sets of graphs concerning either the number, the size or the type of the graphs. None of them have been applied using fuzzy graphs. In this paper, we propose a new approximate algorithm for the computation of the FGMG. The proposed algorithm provides some remarkable improvements over previous algorithms for the median graph computation. First, it deals with FARGs in order to model the vagueness and/or uncertainty associated with the attributes of document image regions and their relationships. This solves the problem of "all-or-nothing" representation that leads to unsatisfactory results in several situations. Further, as we will show later in the experiments, this new algorithm is applicable to a large number of FARGs which have no constraints regarding the number of nodes and edges. Furthermore, it is based on a new FARG embedding method in order to get the main advantages of both the vector and graph representations. Finally, we will show that this new algorithm is able to obtain a good approximation for the FGMG by testing its applicability to the CBDR problem.

4. Fuzzy generalized median graphs computation

In this section, we present a detailed description of the proposed FGMG computation algorithm. First, we briefly introduce the overview of the three main steps of the proposed algorithm. Then, we describe each of these steps in detail.

4.1. General schema

Fig. 1 shows the three main steps of the proposed FGMG computation algorithm. Let be $S=\{G_1, G_2, \ldots, G_m\}$ a set of m FARGs. First, all FARGs in S are embedded into an n-dimensional vector space. Each FARG is mapped to a set of points in $\mathbb{R}^n$. Then, the vectors obtained in the first step are clustered into K clusters using a clustering method such as K-Means, Fuzzy C-Means (FCM), etc. After this, the weight of each cluster is computed and becomes a new attribute of node in FGMG. Finally, the obtained clusters are represented as the nodes and edges of the obtained FGMG.

4.2. Fuzzy generalized median graph algorithm

Fuzzy Generalized Median Graph Algorithm:
Inputs: Datasets containing a large number of FARGs
Outputs: FGMG for each dataset
Step 1: Embedding of FARGs in a vector space
Each FARG is represented by a set of vectors (one vector per node).
Step 2: Clustering of the FARG vectors
- Clustering the FARG vectors using a clustering algorithm such as FCM, K-Means, etc.
- Computing the weight of each cluster.

Step 3: Converting to FGMG
- Each cluster becomes a node of the FGMG.
- Add each obtained weight of each cluster as an attribute of node in FGMG.

The three main steps are further explained in detail in the following subsections.

4.2.1. FARG embedding in a vector space

The first step consists of embedding all FARGs in S into a suitable vector space. The vectorial representation of the corresponding FARGs aims at combining advantages from both the FARG and the vector representations. Each FARG G_i ($i \in \{1, 2, \ldots, m\}$) is encoded by equal size vectors (one vector per node). We denote by FV_{ij} a vector representation which corresponds to a node j in FARG G_i. A matrix M_i containing all the vectors FV_{ij} associated to all nodes in the FARG G_i is then constructed. The obtained vectors can be graphically represented by points in a suitable vector space. Finally, all FARGs in S are mapped to appropriate points in the vector space. More formally, for a given FARG $G=(N, E, \mu, \upsilon)$, FARG embedding can be defined as a function Φ, which maps the FARG G_i from graph space GS to a matrix M_i into n-dimensional vector space $\mathbb{R}^n$. It is given as:

$$\Phi : GS \rightarrow \mathbb{R}^n$$

$$G_i \rightarrow \Phi(G_i) = M_i = \begin{pmatrix} FV_{i1} \\ FV_{i2} \\ \vdots \\ FV_{ij_i} \end{pmatrix} \quad (6)$$

where J_i is the number of nodes in G_i.

A block diagram of FARG embedding is presented in Fig. 2. As input, it accepts a collection of m FARGs $\{G_1, G_2, \ldots, G_e, \ldots, G_m\}$ where the e^{th} FARG is denoted by $G_e = (N_e, E_e, \mu_e, \upsilon_e)$. As output, it produces a set of m matrices, given by $\{M_1, M_2, \ldots, M_e, \ldots, M_m\}$. Each matrix contains J_i ($i \in \{1, 2, \ldots, m\}$) equal size feature

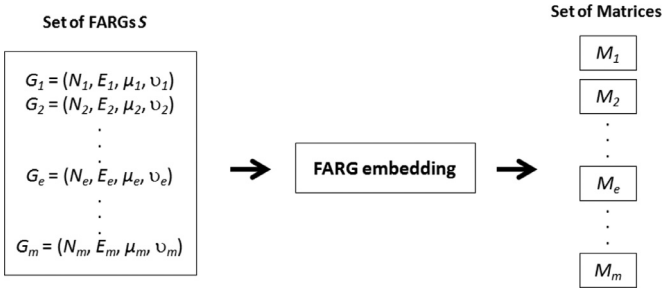

Fig. 2. Overview of FARG embedding in vector space.

vectors{FV_{i1}, FV_{i2}, ..., FV_{iJi}}. The e^{th} input FARG G_e is embedded into a matrix M_e containing J_e feature vectors:

$$G_e \rightarrow \Phi(G_e) = M_e = \begin{pmatrix} FV_{i1} \\ FV_{i2} \\ \vdots \\ FV_{iJ_e} \end{pmatrix} \tag{7}$$

Each vector representation FV_{ij} is composed of two main parts: fuzzy intrinsic features of node j in FARG G_i and fuzzy relationship information between this node and other nodes in G_i. Each FV_{ij} can be graphically represented by a point in n-dimensional vector space $\mathbb{R}^n$. It is given as:

$$FV_{ij} = \begin{pmatrix} f_{FV_{ij}}{}^{(1)} & f_{FV_{ij}}{}^{(2)} & \cdots & f_{FV_{ij}}{}^{(n)} \end{pmatrix} \tag{8}$$

where n is the total number of features in FV_{ij}.

Therefore, each matrix M_i ($i \in \{1, 2, ..., m\}$) can be expressed in terms of features as follows:

$$M_i = \begin{pmatrix} FV_{i1} \\ FV_{i2} \\ \vdots \\ FV_{iJ_i} \end{pmatrix} = \begin{pmatrix} f_{FV_{i1}}{}^{(1)} & f_{FV_{i1}}{}^{(2)} & \cdots & f_{FV_{i1}}{}^{(n)} \\ f_{FV_{i2}}{}^{(1)} & f_{FV_{i2}}{}^{(2)} & \cdots & f_{FV_{i2}}{}^{(n)} \\ \vdots & \ddots & & \vdots \\ f_{FV_{iJ_i}}{}^{(1)} & f_{FV_{iJ_i}}{}^{(2)} & \cdots & f_{FV_{iJ_i}}{}^{(n)} \end{pmatrix} \tag{9}$$

Fig. 3 depicts an example of embedding three FARGs G_1, G_2 and G_3 into an n-dimensional vector space $\mathbb{R}^n$. M_1, M_2 and M_3 are three matrix containing all the feature vectors associated to G_1, G_2 and G_3, respectively.

4.2.2. Embedded fuzzy graphs clustering

Once all FARGs G_i have been embedded into a vector space, the corresponding FARG points can then be used as inputs to a clustering algorithm to cluster these points into K partitions (clusters). First, the optimal K for a given set of FARGs is calculated. Then, a FCM algorithm is used for clustering. Finally, a weight measure is associated to each cluster.

Automatic computation of the optimal K:

Given a set $S = \{G_1, G_2, ..., G_m\}$ of m FARGs. The number of nodes in a FARG G_i is calculated as follows:

$$NN_i = Card\left(\{FV_{ij}\}\right) \tag{10}$$

Let K_S be the optimal K for the set S and $S_{NN} = \{NN_1, NN_2, ..., NN_m\}$ be the set of the numbers of nodes associated with S. K_S can be automatically calculated from the set of underlying FARGs by using one of the following possible measures:

- Mean:

K_S is the mean value of the numbers of nodes associated with S. For the former, the obtained value of K_S is rounded to the nearest integer value.

$$K_S = \frac{1}{m} \times \sum_{i=1}^{m} NN_i \tag{11}$$

- Maximum:

K_S is equal to the maximum number of nodes that can be found in a FARG G_i.

$$K_S = \underset{NN_i \in S_{NN}}{\arg\max} \{NN_i\};\ i \in \{1, 2, \ldots, m\} \tag{12}$$

- Minimum:

K_S is equal to the minimum number of nodes that can be found in a FARG G_i.

$$K_S = \underset{NN_i \in S_{NN}}{\arg\min} \{NN_i\};\ i \in \{1, 2, \ldots, m\} \tag{13}$$

- Median:

K_S is equal to the median number of nodes that can be found in a FARG G_i.

$$K_S = \underset{NN_i \in S_{NN}}{\arg\text{med}} \{NN_i\};\ i \in \{1, 2, \ldots, m\} \tag{14}$$

Clustering:

We used the FCM algorithm to group the points obtained in the first step into K_S subsets. The FCM algorithm is well explored in the literature and has been proved to have good clustering performance on a wide range of datasets that are uncertain, overlapped and hard to cluster [34,35]. The FCM algorithm introduces the fuzziness for the membership of each data point to all clusters with different membership grades between 0 and 1. The sum of the membership grades for each data point must be equal to 1. The FCM algorithm is composed of the following steps [34,35]:

FCM algorithm:
Step 1: Initialize the membership matrix $U = [u_{ij}]$, $U^{(0)}$
Step 2: At k-step: calculate the centers vectors $C^{(k)} = [c_j]$ with $U^{(k)}$

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}{}^m \times x_i}{\sum_{i=1}^{N} u_{ij}{}^m} \tag{15}$$

Step 3: Update $U^{(k)}$, $U^{(k+1)}$

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{x_i - c_j}{x_i - c_k}\right)^{\frac{2}{m-1}}} \tag{16}$$

Step 4: If $||U^{(k+1)} - U^{(k)}|| < \varepsilon$ then STOP; otherwise return to step 2.
where m is any real number greater than 1, u_{ij} is the degree of membership of x_i in the cluster j, x_i is the ith of d-dimensional measured data and c_j is the d-dimension center of the cluster.

Computing the weight of each cluster:

Let S_C be the obtained set of clusters. For each obtained cluster C_i ($i \in \{1, 2, ..., K_S\}$), the weight associated to C_i is calculated using the following formula:

$$W_{C_i} = \frac{NN_{C_i}}{\sum_{i=1}^{K_S} NN_{C_i}} \tag{17}$$

where NN_{C_i} is the number of points associated to the cluster C_i.

An example of clustering of FARG vectors is presented in Fig. 4.

4.2.3. Converting to FGMG

After clustering the FARG vectors, the final step is to represent each obtained cluster C_i ($i \in \{1, 2, ..., K_S\}$) in the previous step as a node of the FGMG. Since each feature vector FV_{ij} is composed of two main parts: fuzzy intrinsic features and fuzzy spatial information associated to a node j in FARG G_i, each obtained cluster C_i is composed of fuzzy intrinsic features and fuzzy spatial information associated to a node i in the FGMG. The obtained FGMG is a complete graph. The spatial relationship between each pair of nodes in

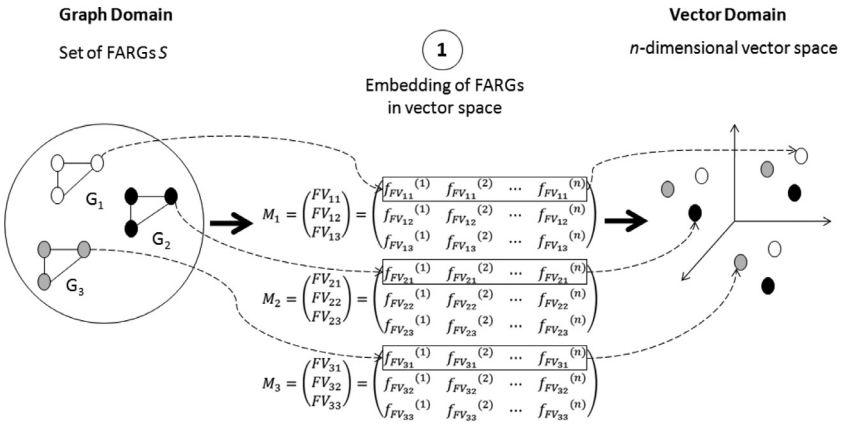

Fig. 3. A detailed example of the first step (FARG embedding in a vector space).

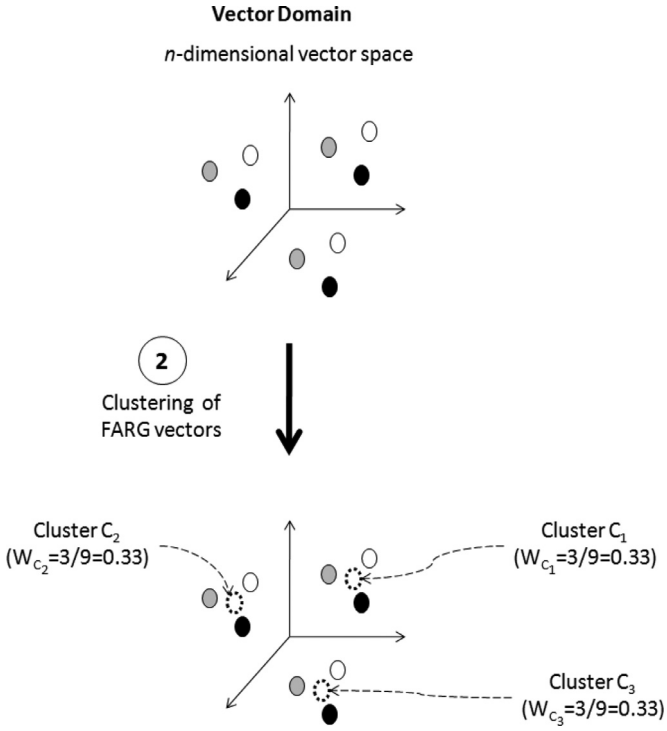

Fig. 4. A detailed example of the second step (Clustering of FARG vectors).

the obtained FGMG is determined using the obtained spatial informations in each cluster C_i. The obtained FGMG is considered as an approximation of the fuzzy median graph of the set S. Finally, each obtained weight W_{C_i} is added as an attribute of node in FGMG as shown in Fig. 5.

5. Application of fuzzy generalized median graphs to content based document retrieval

FGMGs can be used in many document analysis fields such as CBDR, document image classification, pattern recognition, computer vision, etc. In this paper, we focus only on the application of FGMGs to CBDR. First, we start by presenting the overall schema of the proposed CBDR approach using FGMGs. Then, we will present a comparison between using or not FGMGs for CBDR. Finally, we will define two evaluation measures in order to compare the performance of a CBDR using the proposed FGMG algorithm with a classical CBDR using one-to-one matching in terms of performance gain and time-processing gain.

Fig. 6 shows an overview of all the steps of the proposed CBDR approach using FGMGs. The proposed approach is composed of two main phases: offline indexing phase and online retrieval phase. The first phase starts with segmenting each document image into a number of regions. After this, each document image is represented by a FARG whose nodes represent the segmented regions and edges represent the spatial relations between these regions. A detailed description of node and edge attributes will be presented in the experimental section. Then, the obtained FARGs are indexed into a FARG database. Finally, a FGMG is computed for each document image dataset and a FGMG database containing all the obtained FGMGs is constructed. In the online retrieval phase, three types of query specification are used. These types allow the user firstly to specify a query by selecting a number of regions of interest (ROIs) in a document image, choosing an example document image or drawing a sketch of a document image with a graphic editing tool. Then, a FARG representing the user's query $FARG^q$ is generated and compared to each FGMG $FGMG_i$ ($i \in \{1, 2, \ldots, N\}$) in the FGMG database. After this, the FARGs associated to FGMGs that have the smallest distance $d(FARG^q, FGMG_i)$ to $FARG^q$ are selected and compared to $FARG^q$. Finally, the most similar FARGs belonging to the selected FARGs are retrieved and displayed to the user. An example of a CBDR using FGMGs is illustrated in Fig. 7.

In order to compare the performance of a CBDR using the proposed FGMG algorithm with a classical CBDR using one-to-one matching, we defined two evaluation measures: Accuracy Gain (AG) and Time-processing Gain (TG). These two measures are expressed as percentages as follows:

$$AG = \left[1 - \left(\frac{A_{FGMG}}{A_{OTO}}\right)\right] \times 100 \tag{18}$$

$$TG = \left[1 - \left(\frac{T_{FGMG}}{T_{OTO}}\right)\right] \times 100 \tag{19}$$

where A_{FGMG} and T_{FGMG} are the accuracy and the time-processing of a CBDR using the proposed FGMG algorithm, respectively. Similarly, A_{OTO} and T_{OTO} are the accuracy and the time-processing of a CBDR using one-to-one matching, respectively.

6. Experimental results and analysis

This section is devoted to the experimental evaluation of the proposed FGMG algorithm. We present in this section three experiments using two synthetic databases and one real database of

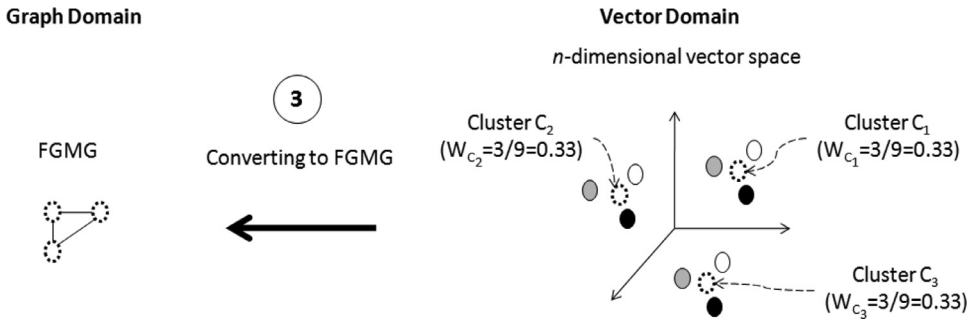

Fig. 5. A detailed example of the third step (Converting to FGMG).

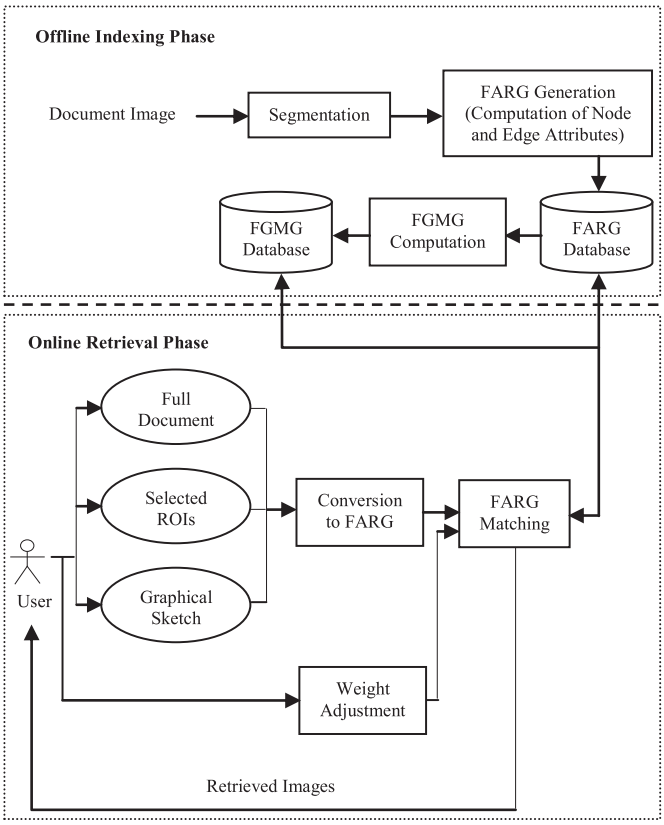

Fig. 6. Block diagram of the proposed CBDR approach using FGMGs.

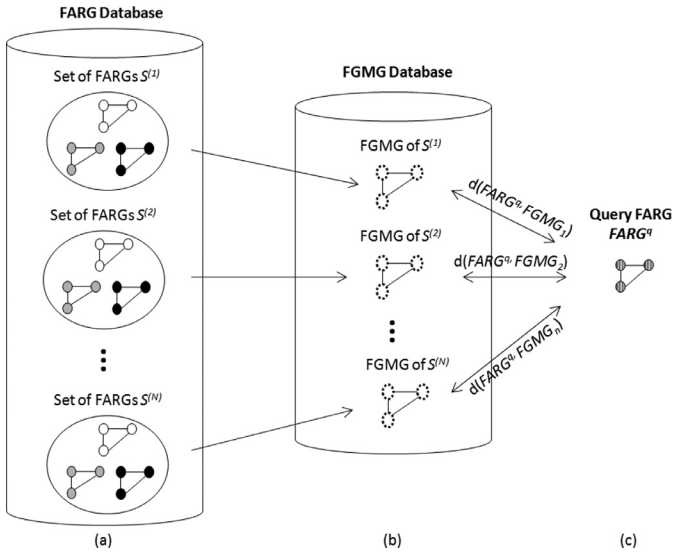

Fig. 7. An example of a CBDR using FGMGs.

FARGs representing synthetic and real document images, respectively. Our experiments are organized in a three step methodology. First, a simple example of the FGMG computation is presented in details in order to clarify the proposed algorithm. Second, we have investigated the influence of the number of nodes in a FARG and the number of FARGs in a dataset values on the runtime processing of the FGMG computation. Finally, an evaluation of the CBDR performance that can be reached with or without using FGMG as prototype is carried out. The two retrieval scenarios have been experimentally compared according to several criteria on both synthetic and real databases.

6.1. Databases

The experiments described in this section have been carried out on two synthetic databases and one real database which are detailed as follows:

6.1.1. Databases description

6.1.1.1. Synthetic database (Base A). This database is composed of three datasets of document images. Each dataset contains a large number of FARGs that represent real-world document images (dataset 1: journals, dataset 2: book pages and dataset 3: invoices). Each dataset template corresponds to a real-world template of a document image. Each dataset template is composed of a number of two types of regions: constant region and variable region. A constant region must contain at most one subregion belonging to one of two types: “Text” or “Non-Text”, whereas a variable region may contain at most one subregion belonging to one of two types: “Text” or “Non-Text”. Figs. 8–10 illustrate the templates of the three datasets, respectively.

- *Dataset 1: Journals*

As shown in Fig. 8, each document image is composed of eight regions: four “Non-Text” regions in the left and four “Text” regions in the right.

- *Dataset 2: Book pages*

As shown in Fig. 9, each document image is composed of five regions: one “Non-Text” region in the left, three “Text” regions in the middle and one “Non-Text” region in the right.

- *Dataset 3: Invoices*

As shown in Fig. 10, each document image is composed of five regions: two “Non-Text” regions and one “Text” region in the up, one “Text” region in the middle and one “Text” region in the bottom.

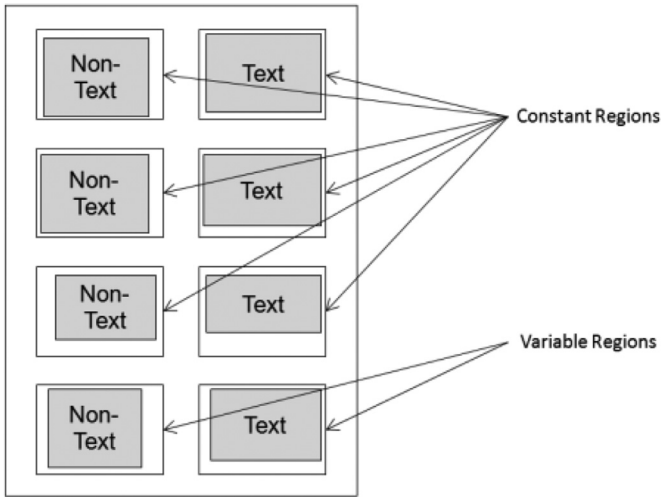

Fig. 8. An example of a document image belonging to dataset 1.

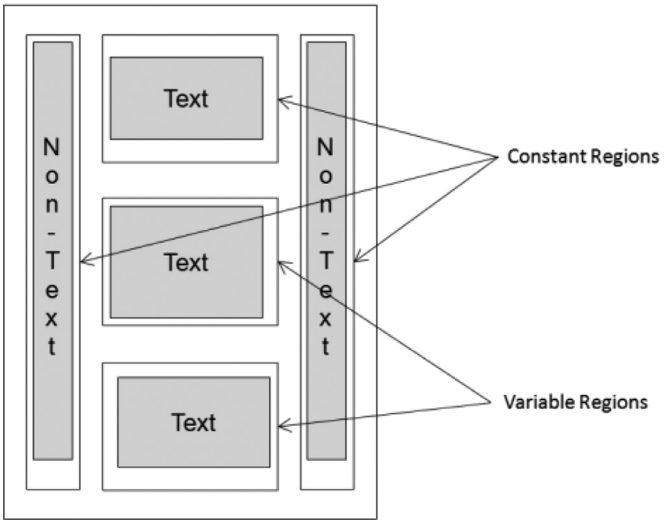

Fig. 9. An example of a document image belonging to dataset 2.

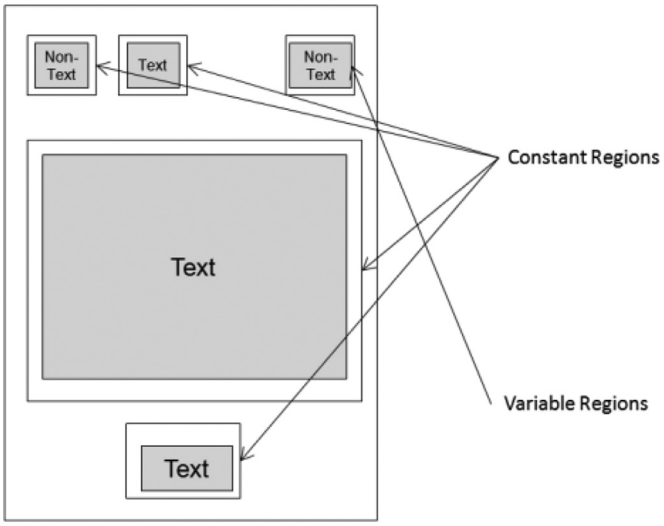

Fig. 10. An example of a document image belonging to dataset 3.

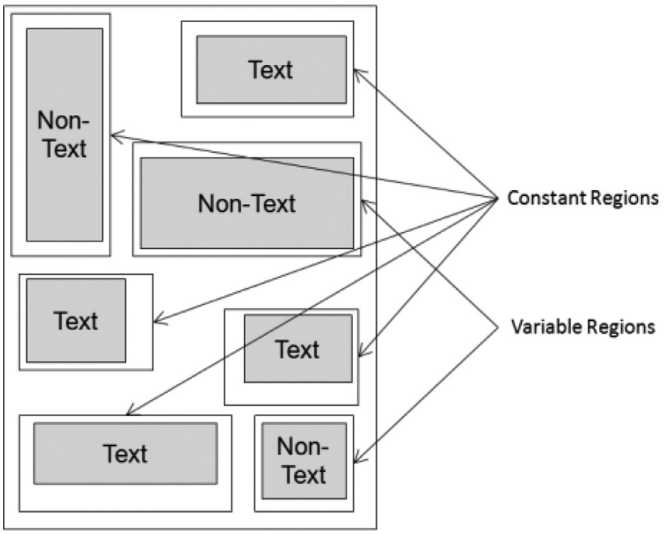

Fig. 11. An example of a document image belonging to a dataset of Base *B*.

6.1.1.2. Synthetic database (Base B). This second database is composed of a number of datasets of document images. This number can be specified by the user before generating the database. Each dataset contains a large number of FARGs that do not necessarily correspond to real-world templates of document images. A dataset template is randomly generated for each dataset of document images. Each dataset template is composed of a randomly generated number of two types of regions: constant region and variable region. A constant region must contain at most one subregion belonging to one of two types: "Text" or "Non-Text", whereas a variable region may contain at most one subregion belonging to one of two types: "Text" or "Non-Text". Fig. 11 illustrates an example of a template of a document image dataset. In this example, each document image belonging to a dataset of Base *B* is composed of seven regions: four "Text" regions and three "Non-Text" regions.

6.1.1.3. Real database (Base C). This third database is domain specific. It is derived from the Tobacco-800 dataset [36,37]. Tobacco-800 is a public subset of the Illinois Institute of Technology Complex Document Information Processing (IIT-CDIP) Test Collection [36,37]. It is a realistic and complex dataset for document analysis and retrieval. It consists of 1290 real-world documents. The image resolutions range from 150 to 300 DPIs. We tested our approach using a total of 15 document images across 3 classes from the Tobacco-800 dataset. Each class contains 5 document images with similar structures and contents. Similarly to the two previous databases, each document image in *Base C* contains a number of constant and variable regions. Three examples from the three different classes are shown in Figs. 12–14.

- *Dataset 1*

As shown in Fig. 12, each document image is composed of seven regions: four "Text" regions and three "Non-Text" regions.

- *Dataset 2*

As shown in Fig. 13, each document image is composed of six regions: four "Text" regions and two "Non-Text" regions.

- *Dataset 3*

As shown in Fig. 14, each document image is composed of eleven regions: five "Text" regions and six "Non-Text" regions.

It is important to note that all regions in the input document images were manually cropped using a ground truthing editor:

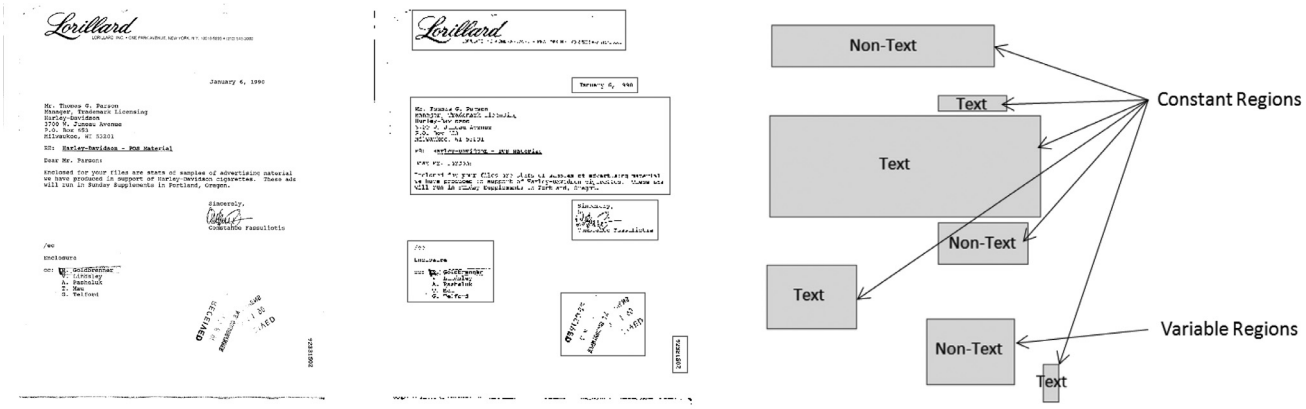

Fig. 12. An example of a document image belonging to dataset 1: (a) Input document image, (b) Selected regions, (c) Ground truth.

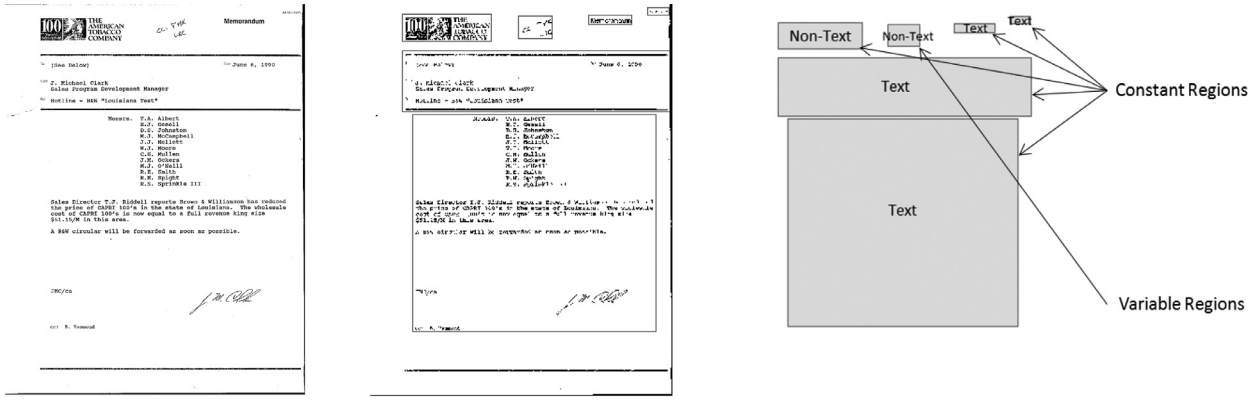

Fig. 13. An example of a document image belonging to dataset 2: (a) Input document image, (b) Selected regions, (c) Ground truth.

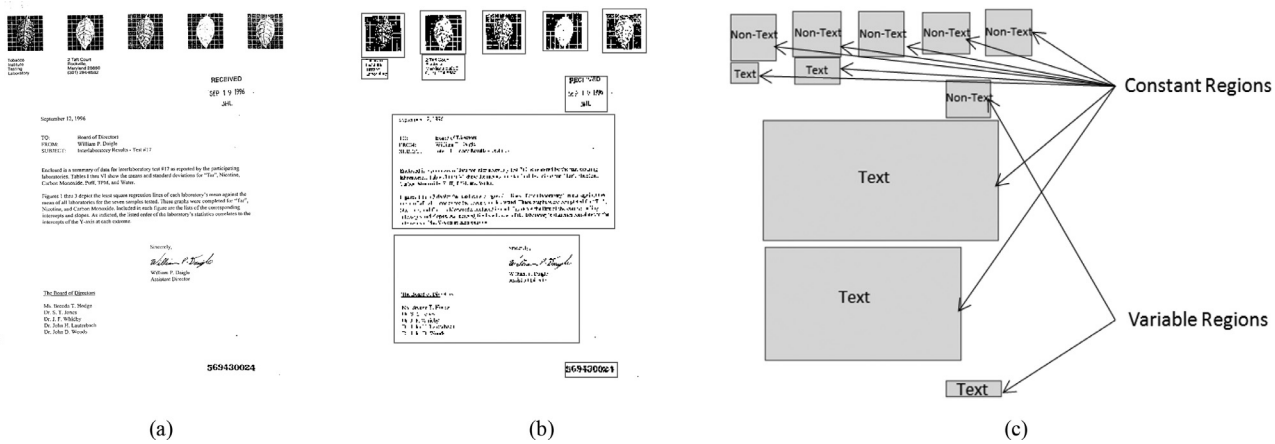

Fig. 14. An example of a document image belonging to dataset 3: (a) Input document image, (b) Selected regions, (c) Ground truth.

Groundtruthing Environment for Document Images (GEDI) [38]. After selecting rectangular regions in each document image and assigning them to a number of pre-defined attributes, GEDI generates an XML schema representing the nature and the location of each region in a document image. Finally, each FARG representing a document image is constructed using its correspondant XML schema.

6.1.2. FARG attributes

Let G be a FARG representing a document image in one of previous databases. Node and edge attributes are given as follows:

6.1.2.1. Node attributes. Node attributes represent intrinsic informations of a region. In this paper, two node attributes are used: *Type* and *Area*. More formally, for each node n_i in G, the set of node attributes is given as α_i:

$$\alpha_i = \{\alpha_{i1},\ \alpha_{i2}\} \tag{20}$$

where α_{i1} and α_{i2} represent the *Type* and *Area* attributes associated with the node n_i, respectively.

As shown in Figs. 15 and 16, a trapezoidal fuzzy membership function is defined for each node attribute. The sum of the values

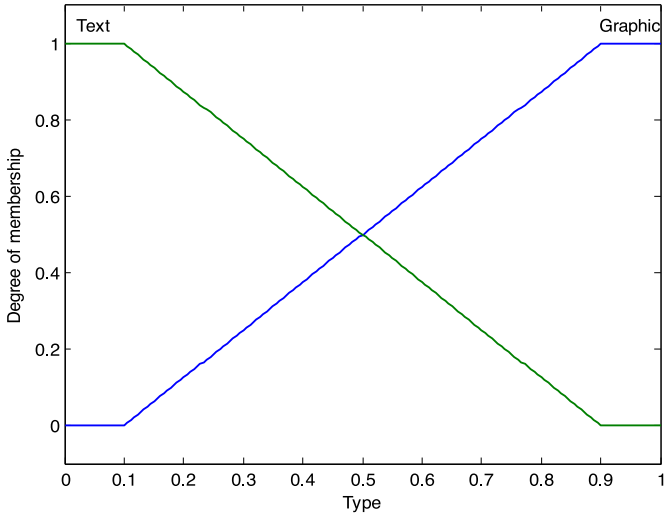

Fig. 15. Trapezoidal fuzzy membership function of *Type* attribute.

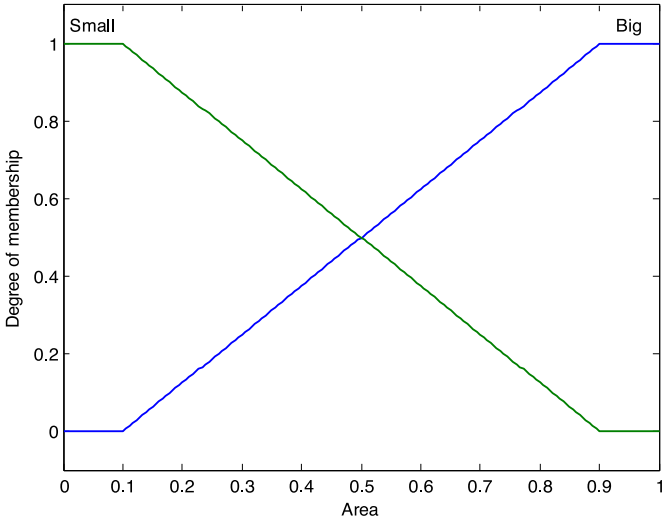

Fig. 16. Trapezoidal fuzzy membership function of *Area* attribute.

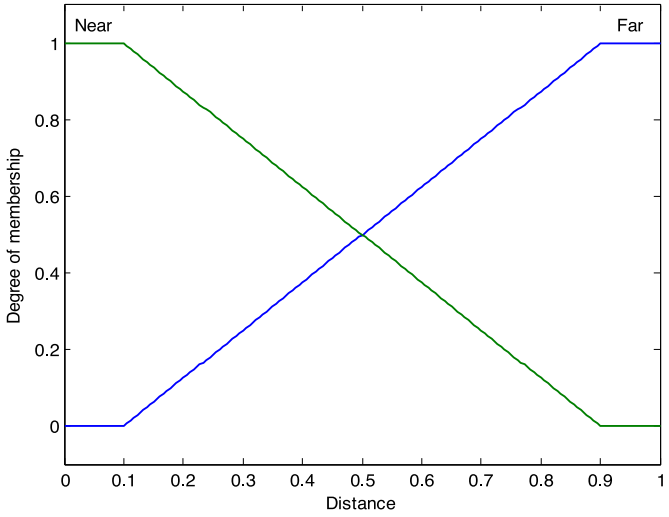

Fig. 17. Trapezoidal fuzzy membership function of *Distance* attribute.

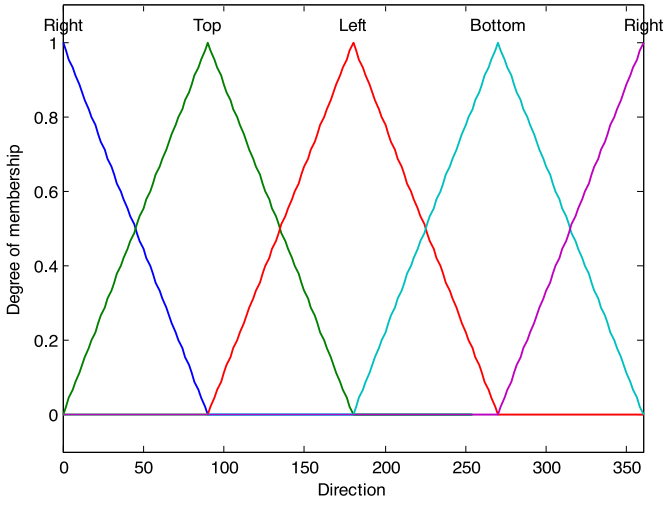

Fig. 18. Trapezoidal fuzzy membership function of *Direction* attribute.

of the membership functions corresponding to a node attribute is equal to 1. *Type* attribute is represented as a vector $\mu_{\alpha_{i1}}$:

$$\mu_{\alpha_{i1}} = \left[\mu_{\alpha_{i1}}{}^{(1)},\ \mu_{\alpha_{i1}}{}^{(2)}\right] \tag{21}$$

$$\sum_{j=1}^{2} \mu_{\alpha_{i1}}{}^{(j)} = 1 \tag{22}$$

where $\mu_{\alpha_{i1}}{}^{(1)}$ and $\mu_{\alpha_{i1}}{}^{(2)}$ are the membership degrees associated with α_{i1}. $\mu_{\alpha_{i1}}{}^{(1)}$ and $\mu_{\alpha_{i1}}{}^{(2)}$ represent the percentage of the *Type* attribute to be "Text" or "Non-Text", respectively.

Similarly, *Area* attribute is represented as a vector $\mu_{\alpha_{i2}}$:

$$\mu_{\alpha_{i2}} = \left[\mu_{\alpha_{i2}}{}^{(1)},\ \mu_{\alpha_{i2}}{}^{(2)}\right] \tag{23}$$

$$\sum_{j=1}^{2} \mu_{\alpha_{i2}}{}^{(j)} = 1 \tag{24}$$

where $\mu_{\alpha_{i2}}{}^{(1)}$ and $\mu_{\alpha_{i2}}{}^{(2)}$ are the membership degrees associated with α_{i2}. $\mu_{\alpha_{i2}}{}^{(1)}$ and $\mu_{\alpha_{i2}}{}^{(2)}$ represent the percentage of the *Area* attribute to be "Big" or "Small", respectively.

6.1.2.2. Edge attributes. Edge attributes represent the spatial position informations of a region. In this paper, two edge attributes are used: *Distance* and *Direction*. More formally, for each edge e_i in G, the set of edge attributes is given as β_i:

$$\beta_i = \{\beta_{i1},\ \beta_{i2}\} \tag{25}$$

where β_{i1} and β_{i2} represent the *Distance* and *Direction* attributes associated with the edge e_i, respectively.

As shown in Figs. 17 and 18, trapezoidal and triangular fuzzy membership functions are defined for *Distance* attribute and *Direction* attribute, respectively. The sum of the values of the membership functions corresponding to an edge attribute is equal to 1. *Distance* attribute is represented as a vector $\mu_{\beta_{i1}}$:

$$\mu_{\beta_{i1}} = \left[\mu_{\beta_{i1}}{}^{(1)},\ \mu_{\beta_{i1}}{}^{(2)}\right] \tag{26}$$

$$\sum_{j=1}^{2} \mu_{\beta_{i1}}{}^{(j)} = 1 \tag{27}$$

where $\mu_{\beta_{i1}}{}^{(1)}$ and $\mu_{\beta_{i1}}{}^{(2)}$ are the membership degrees associated with β_{i1}. $\mu_{\beta_{i1}}{}^{(1)}$ and $\mu_{\beta_{i1}}{}^{(2)}$ represent the percentage of the *Distance* attribute to be "Far" or "Near", respectively.

Table 1
A brief summary of each of the three databases.

	Database						
	Base A			Base B	Base C		
	Dataset 1	Dataset 2	Dataset 3	Specified number of datasets	Dataset 1	Dataset 2	Dataset 3
Synthetic / Real Document images	Synthetic Journals	Synthetic Book pages	Synthetic Invoices	Synthetic Randomly generated	Real Tobacco-800 dataset	Real Tobacco-800 dataset	Real Tobacco-800 dataset
No. of FARGs	Specified number	Specified number	Specified number	Specified number for each dataset	5	5	5
No. of "Text" regions in each FARG	4	3	3	Randomly generated	4	4	5
No. of "Non-Text" regions in each FARG	4	2	2	Randomly generated	3	2	6
No. of constant regions in each FARG	6	3	4	Randomly generated	6	5	9
No. of variable regions in each FARG	2	2	1	Randomly generated	1	1	2

Similarly, *Direction* attribute is represented as a vector $\mu_{\beta_{i2}}$:

$$\mu_{\beta_{i2}} = \left[\mu_{\beta_{i2}}{}^{(1)},\ \mu_{\beta_{i2}}{}^{(2)}, \mu_{\beta_{i2}}{}^{(3)},\ \mu_{\beta_{i2}}{}^{(4)}\right] \tag{28}$$

$$\sum_{j=1}^{4} \mu_{\beta_{i2}}{}^{(j)} = 1 \tag{29}$$

where $\mu_{\beta_{i2}}{}^{(1)}$, $\mu_{\beta_{i2}}{}^{(2)}$, $\mu_{\beta_{i2}}{}^{(3)}$ and $\mu_{\beta_{i2}}{}^{(4)}$ are the membership degrees associated with β_{i2}. $\mu_{\beta_{i2}}{}^{(1)}$, $\mu_{\beta_{i2}}{}^{(2)}$, $\mu_{\beta_{i2}}{}^{(3)}$ and $\mu_{\beta_{i2}}{}^{(4)}$ represent the percentage of the Direction attribute to be "Right", "Left", "Top" or "Bottom", respectively.

In order to synthesize the descriptions of all the databases, we provide in Table 1 a brief summary of each of the three databases we have presented.

6.2. Experimentations

For the experiments, the CBDR domain is used as the application area by using two synthetic databases and one real database of document images.

Experiment 1: An example of the FGMG computation

In this experiment, we present a simple example to illustrate how the proposed FGMG algorithm works. As shown in Fig. 19, suppose that we have three input FARGs G_1, G_2 and G_3.

The three steps of the FGMG computation are explained in detail as following:

Step 1: FARG Embedding in a Vector Space

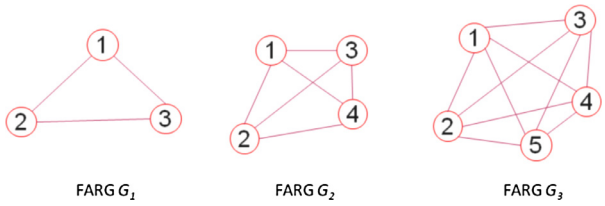

Fig. 19. Three input FARGs G_1, G_2 and G_3.

The first step consists of embedding all FARGs G_1, G_2 and G_3 into a suitable vector space. Let FV_{ij} be the obtained feature vector which corresponds to the node j in FARG G_i. Three matrix M_1, M_2 and M_3 containing all the feature vectors FV_{ij} of all nodes in G_1, G_2 and G_3 are constructed as follows:

$$M_1 = \begin{pmatrix} FV_{11} \\ FV_{12} \\ FV_{13} \end{pmatrix} \tag{30}$$

$$M_1 = \begin{pmatrix} \mu_{\alpha_{11}}{}^{(1)} & \mu_{\alpha_{11}}{}^{(2)} & \mu_{\alpha_{12}}{}^{(1)} & \mu_{\alpha_{12}}{}^{(2)} & \mu_{\beta_{11}}{}^{(1)} & \mu_{\beta_{11}}{}^{(2)} & \mu_{\beta_{12}}{}^{(1)} & \mu_{\beta_{12}}{}^{(2)} & \mu_{\beta_{12}}{}^{(3)} & \mu_{\beta_{12}}{}^{(4)} \\ \mu_{\alpha_{21}}{}^{(1)} & \mu_{\alpha_{21}}{}^{(2)} & \mu_{\alpha_{22}}{}^{(1)} & \mu_{\alpha_{22}}{}^{(2)} & \mu_{\beta_{21}}{}^{(1)} & \mu_{\beta_{21}}{}^{(2)} & \mu_{\beta_{22}}{}^{(1)} & \mu_{\beta_{22}}{}^{(2)} & \mu_{\beta_{22}}{}^{(3)} & \mu_{\beta_{22}}{}^{(4)} \\ \mu_{\alpha_{31}}{}^{(1)} & \mu_{\alpha_{31}}{}^{(2)} & \mu_{\alpha_{32}}{}^{(1)} & \mu_{\alpha_{32}}{}^{(2)} & \mu_{\beta_{31}}{}^{(1)} & \mu_{\beta_{31}}{}^{(2)} & \mu_{\beta_{32}}{}^{(1)} & \mu_{\beta_{32}}{}^{(2)} & \mu_{\beta_{32}}{}^{(3)} & \mu_{\beta_{32}}{}^{(4)} \end{pmatrix} \tag{31}$$

$$M_2 = \begin{pmatrix} FV_{21} \\ FV_{22} \\ FV_{23} \\ FV_{24} \end{pmatrix} \tag{32}$$

$$M_2 = \begin{pmatrix} \mu_{\alpha_{11}}{}^{(1)} & \mu_{\alpha_{11}}{}^{(2)} & \mu_{\alpha_{12}}{}^{(1)} & \mu_{\alpha_{12}}{}^{(2)} & \mu_{\beta_{11}}{}^{(1)} & \mu_{\beta_{11}}{}^{(2)} & \mu_{\beta_{12}}{}^{(1)} & \mu_{\beta_{12}}{}^{(2)} & \mu_{\beta_{12}}{}^{(3)} & \mu_{\beta_{12}}{}^{(4)} \\ \mu_{\alpha_{21}}{}^{(1)} & \mu_{\alpha_{21}}{}^{(2)} & \mu_{\alpha_{22}}{}^{(1)} & \mu_{\alpha_{22}}{}^{(2)} & \mu_{\beta_{21}}{}^{(1)} & \mu_{\beta_{21}}{}^{(2)} & \mu_{\beta_{22}}{}^{(1)} & \mu_{\beta_{22}}{}^{(2)} & \mu_{\beta_{22}}{}^{(3)} & \mu_{\beta_{22}}{}^{(4)} \\ \mu_{\alpha_{31}}{}^{(1)} & \mu_{\alpha_{31}}{}^{(2)} & \mu_{\alpha_{31}}{}^{(2)} & \mu_{\alpha_{32}}{}^{(2)} & \mu_{\beta_{31}}{}^{(1)} & \mu_{\beta_{31}}{}^{(2)} & \mu_{\beta_{32}}{}^{(1)} & \mu_{\beta_{32}}{}^{(2)} & \mu_{\beta_{32}}{}^{(3)} & \mu_{\beta_{32}}{}^{(4)} \\ \mu_{\alpha_{41}}{}^{(1)} & \mu_{\alpha_{41}}{}^{(2)} & \mu_{\alpha_{42}}{}^{(1)} & \mu_{\alpha_{42}}{}^{(2)} & \mu_{\beta_{41}}{}^{(1)} & \mu_{\beta_{41}}{}^{(2)} & \mu_{\beta_{42}}{}^{(1)} & \mu_{\beta_{42}}{}^{(2)} & \mu_{\beta_{42}}{}^{(3)} & \mu_{\beta_{42}}{}^{(4)} \end{pmatrix} \tag{33}$$

$$M_3 = \begin{pmatrix} FV_{31} \\ FV_{32} \\ FV_{33} \\ FV_{34} \\ FV_{35} \end{pmatrix} \tag{34}$$

$$M_3 = \begin{pmatrix} \mu_{\alpha_{11}}{}^{(1)} & \mu_{\alpha_{11}}{}^{(2)} & \mu_{\alpha_{12}}{}^{(1)} & \mu_{\alpha_{12}}{}^{(2)} & \mu_{\beta_{11}}{}^{(1)} & \mu_{\beta_{11}}{}^{(2)} & \mu_{\beta_{12}}{}^{(1)} & \mu_{\beta_{12}}{}^{(2)} & \mu_{\beta_{12}}{}^{(3)} & \mu_{\beta_{12}}{}^{(4)} \\ \mu_{\alpha_{21}}{}^{(1)} & \mu_{\alpha_{21}}{}^{(2)} & \mu_{\alpha_{22}}{}^{(1)} & \mu_{\alpha_{22}}{}^{(2)} & \mu_{\beta_{21}}{}^{(1)} & \mu_{\beta_{21}}{}^{(2)} & \mu_{\beta_{22}}{}^{(1)} & \mu_{\beta_{22}}{}^{(2)} & \mu_{\beta_{22}}{}^{(3)} & \mu_{\beta_{22}}{}^{(4)} \\ \mu_{\alpha_{31}}{}^{(1)} & \mu_{\alpha_{31}}{}^{(2)} & \mu_{\alpha_{32}}{}^{(1)} & \mu_{\alpha_{32}}{}^{(2)} & \mu_{\beta_{31}}{}^{(1)} & \mu_{\beta 31}{}^{(2)} & \mu_{\beta_{32}}{}^{(1)} & \mu_{\beta_{32}}{}^{(2)} & \mu_{\beta_{32}}{}^{(3)} & \mu_{\beta_{32}}{}^{(4)} \\ \mu_{\alpha_{41}}{}^{(1)} & \mu_{\alpha_{41}}{}^{(2)} & \mu_{\alpha_{42}}{}^{(1)} & \mu_{\alpha_{42}}{}^{(2)} & \mu_{\beta_{41}}{}^{(1)} & \mu_{\beta_{41}}{}^{(2)} & \mu_{\beta_{42}}{}^{(1)} & \mu_{\beta_{42}}{}^{(2)} & \mu_{\beta_{42}}{}^{(3)} & \mu_{\beta_{42}}{}^{(4)} \\ \mu_{\alpha_{51}}{}^{(1)} & \mu_{\alpha_{51}}{}^{(2)} & \mu_{\alpha_{52}}{}^{(1)} & \mu_{\alpha_{52}}{}^{(2)} & \mu_{\beta_{51}}{}^{(1)} & \mu_{\beta_{51}}{}^{(2)} & \mu_{\beta_{52}}{}^{(1)} & \mu_{\beta_{52}}{}^{(2)} & \mu_{\beta_{52}}{}^{(3)} & \mu_{\beta_{52}}{}^{(4)} \end{pmatrix} \tag{35}$$

Step 2: Embedded Fuzzy Graphs Clustering

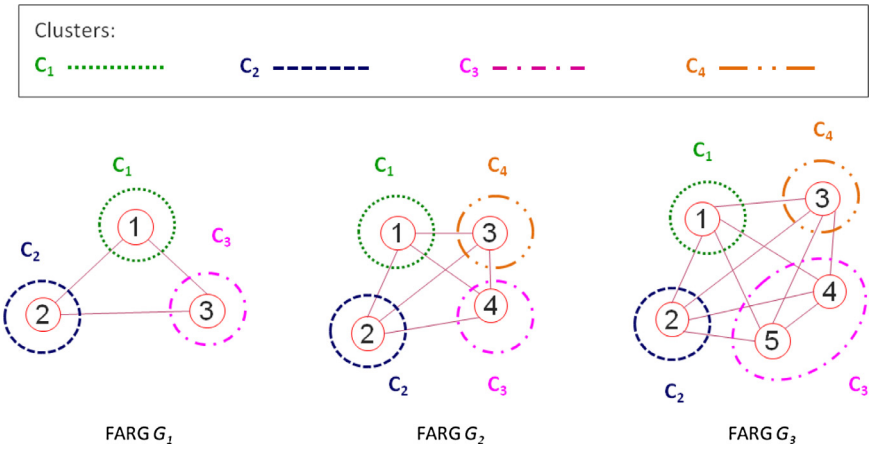

Fig. 20. Obtained clusters using FCM algorithm.

Table 2
Obtained weight of each cluster.

Cluster	No. of associated nodes	No. of nodes	Weight
C_1	3	12	$W_{C_1} = 3/12 = 0.25$
C_2	3	12	$W_{C_2} = 3/12 = 0.25$
C_3	4	12	$W_{C_3} = 4/12 = 0.33$
C_4	2	12	$W_{C_4} = 2/12 = 0.16$

The second step consists of clustering the obtained feature vectors (FV_{11}, FV_{12}, FV_{13}, FV_{21}, FV_{22}, FV_{23}, FV_{24}, FV_{31}, FV_{32}, FV_{33}, FV_{34} and FV_{35}) into K clusters using the FCM algorithm. In this example, we used the Eq. (11) to calculate K. The obtained value of K is 4. Let $S_C = \{C_1, C_2, C_3, C_4\}$ be the obtained set of clusters. As shown in Fig. 20, three feature vectors (FV_{11}, FV_{21} and FV_{31}) are associated to C_1, three feature vectors (FV_{12}, FV_{22} and FV_{32}) are associated to C_2, four feature vectors (FV_{13}, FV_{24}, FV_{34} and FV_{35}) are associated to C_3 and two feature vectors (FV_{23} and FV_{33}) are associated to C_4.

As shown is Eq. (36), we denote by M_{FGMG} the matrix containing all the obtained clusters in S_C.

$$M_{FGMG} = \begin{pmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \end{pmatrix} \tag{36}$$

$$M_{FGMG} = \begin{pmatrix} \mu_{\alpha_{11}}{}^{(1)} & \mu_{\alpha_{11}}{}^{(2)} & \mu_{\alpha_{12}}{}^{(1)} & \mu_{\alpha_{12}}{}^{(2)} & \mu_{\beta_{11}}{}^{(1)} & \mu_{\beta_{11}}{}^{(2)} & \mu_{\beta_{12}}{}^{(1)} & \mu_{\beta_{12}}{}^{(2)} & \mu_{\beta_{12}}{}^{(3)} & \mu_{\beta_{12}}{}^{(4)} \\ \mu_{\alpha_{21}}{}^{(1)} & \mu_{\alpha_{21}}{}^{(2)} & \mu_{\alpha_{22}}{}^{(1)} & \mu_{\alpha_{22}}{}^{(2)} & \mu_{\beta_{21}}{}^{(1)} & \mu_{\beta_{21}}{}^{(2)} & \mu_{\beta_{22}}{}^{(1)} & \mu_{\beta_{22}}{}^{(2)} & \mu_{\beta_{22}}{}^{(3)} & \mu_{\beta_{22}}{}^{(4)} \\ \mu_{\alpha_{31}}{}^{(1)} & \mu_{\alpha_{31}}{}^{(2)} & \mu_{\alpha_{32}}{}^{(1)} & \mu_{\alpha_{32}}{}^{(2)} & \mu_{\beta_{31}}{}^{(1)} & \mu_{\beta_{31}}{}^{(2)} & \mu_{\beta_{32}}{}^{(1)} & \mu_{\beta_{32}}{}^{(2)} & \mu_{\beta_{32}}{}^{(3)} & \mu_{\beta_{32}}{}^{(4)} \\ \mu_{\alpha_{41}}{}^{(1)} & \mu_{\alpha_{41}}{}^{(2)} & \mu_{\alpha_{42}}{}^{(1)} & \mu_{\alpha_{42}}{}^{(2)} & \mu_{\beta_{41}}{}^{(1)} & \mu_{\beta_{41}}{}^{(2)} & \mu_{\beta_{42}}{}^{(1)} & \mu_{\beta_{42}}{}^{(2)} & \mu_{\beta_{42}}{}^{(3)} & \mu_{\beta_{42}}{}^{(4)} \end{pmatrix} \tag{37}$$

Finally, we used the Eq. (17) to calculate the weight of each cluster C_i ($i \in \{1, 2, 3, 4\}$). Table 2 summarizes the obtained weights W_{C_i} ($i \in \{1, 2, 3, 4\}$).

Step 3: Converting to FGMG

In the final step, a FGMG composed of all clusters C_i ($i \in \{1, 2, 3, 4\}$) obtained in the previous step is created as shown in Fig. 21. Each obtained weight W_{C_i} ($i \in \{1, 2, 3, 4\}$) is then added as an attribute of node in the FGMG.

Experiment 2: Computation time of a FGMG

This experiment was intended to quantitatively evaluate the scalability of the proposed FGMG algorithm in terms of computation time with respect to the size of the input FARGs. To this end, experiment 2 was performed on different sized datasets from the three databases: *Base A*, *Base B* and *Base C*. A closer look is given to the impact of dataset sizes on time complexity of the FGMG computation. Therefore, the processing time is benchmarked though the number of nodes and the number of FARGs in each dataset. Tables 3–9 present the basic characteristics of the three databases used to compute the FGMG and the runtime processing values obtained for different number of FARGs in *Base A*, *Base B* and *Base C*.

- *Base A:*
- *Base B:*
- *Base C:*

Table 3
Runtime performance of the proposed FGMG algorithm on Base *A* comprising 3000 FARGs.

	Dataset 1	Dataset 2	Dataset 3
No. of FARGs	1000	1000	1000
Maximum no. of nodes in each FARG	8	5	5
No. of constant nodes in each FARG	6	3	4
No. of variable nodes in each FARG	2	2	1
Maximum no. of nodes in all FARGs	8000	5000	5000
Minimum no. of nodes in all FARGs	6000	3000	4000
No. of nodes in all FARGs	6997	3999	4521
No. of variable nodes in all FARGs	997	999	521
No. of nodes in the FGMG	7	4	5
Computation time of generation (s)	25.34	17.88	20.01
Computation time of the FGMG (s)	1.14	0.20	0.26

Table 4
Runtime performance of the proposed FGMG algorithm on Base *A* comprising 9000 FARGs.

	Dataset 1	Dataset 2	Dataset 3
No. of FARGs	3000	3000	3000
Maximum no. of nodes in each FARG	8	5	5
No. of constant nodes in each FARG	6	3	4
No. of variable nodes in each FARG	2	2	1
Maximum no. of nodes in all FARGs	24,000	15,000	15,000
Minimum no. of nodes in all FARGs	18,000	9000	12,000
No. of nodes in all FARGs	21,025	11,998	13,479
No. of variable nodes in all FARGs	3025	2998	1479
No. of nodes in the FGMG	7	4	4
Computation time of generation (s)	85.20	63.01	70.35
Computation time of the FGMG (s)	3.56	0.34	0.36

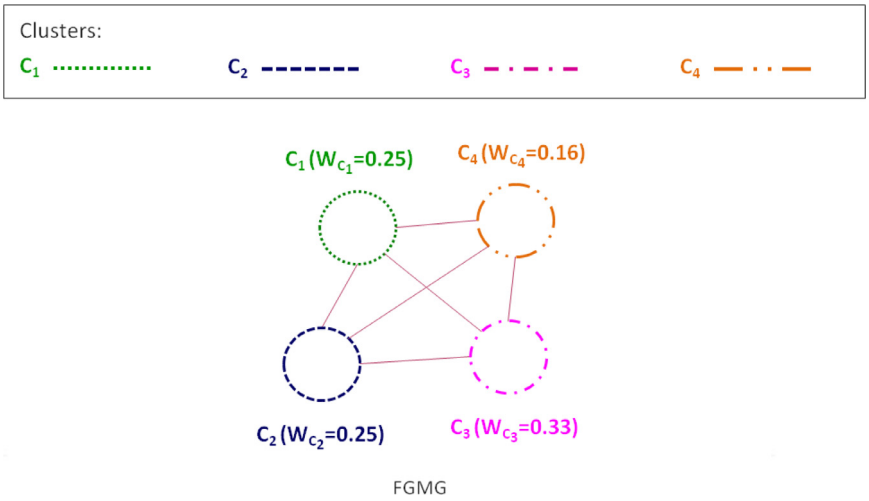

Fig. 21. The obtained FGMG (the weight belonging to each cluster is given in parenthesis).

Table 5
Runtime performance of the proposed FGMG algorithm on Base *A* comprising 15,000 FARGs.

	Dataset 1	Dataset 2	Dataset 3
No. of FARGs	5000	5000	5000
Maximum no. of nodes in each FARG	8	5	5
No. of constant nodes in each FARG	6	3	4
No. of variable nodes in each FARG	2	2	1
Maximum no. of nodes in all FARGs	40,000	25,000	25,000
Minimum no. of nodes in all FARGs	30,000	15,000	20,000
No. of nodes in all FARGs	34,981	20,033	22,495
No. of variable nodes in all FARGs	4981	5033	2495
No. of nodes in the FGMG	7	4	4
Computation time of generation (s)	373.32	273.74	316.43
Computation time of the FGMG (s)	13.14	1.37	1.51

Table 6
Runtime performance of the proposed FGMG algorithm on Base *B* comprising 1500 FARGs.

	Dataset 1	Dataset 2	Dataset 3
No. of FARGs	500	500	500
Maximum no. of nodes in each FARG	100	100	100
Maximum no. of constant nodes in each FARG	91	91	91
Maximum no. of variable nodes in each FARG	9	9	9
Maximum no. of nodes in all FARGs	50,000	50,000	50,000
Maximum no. of constant nodes in all FARGs	45,500	45,500	45,500
Maximum no. of variable nodes in all FARGs	4500	4500	4500
No. of nodes in all FARGs	8473	12,662	17,397
No. of nodes in the FGMG	17	25	35
Computation time of generation (s)	20.44	36.11	57.62
Computation time of the FGMG (s)	3.15	6.76	14.44

Table 7
Runtime performance of the proposed FGMG algorithm on Base *B* comprising 3000 FARGs.

	Dataset 1	Dataset 2	Dataset 3
No. of FARGs	1000	1000	1000
Maximum no. of nodes in each FARG	100	100	100
Maximum no. of constant nodes in each FARG	91	91	91
Maximum no. of variable nodes in each FARG	9	9	9
Maximum no. of nodes in all FARGs	100,000	100,000	100,000
Maximum no. of constant nodes in all FARGs	91,000	91,000	91,000
Maximum no. of variable nodes in all FARGs	9000	9000	9000
No. of nodes in all FARGs	26,496	37,174	48,032
No. of nodes in the FGMG	26	37	48
Computation time of generation (s)	83.59	130.66	198.01
Computation time of the FGMG (s)	16.92	33.46	55.13

Table 8
Runtime performance of the proposed FGMG algorithm on Base *B* comprising 9000 FARGs.

	Dataset 1	Dataset 2	Dataset 3
No. of FARGs	3000	3000	3000
Maximum no. of nodes in each FARG	100	100	100
Maximum # of constant nodes in each FARG	91	91	91
Maximum no. of variable nodes in each FARG	9	9	9
Maximum no. of nodes in all FARGs	300,000	300,000	300,000
Maximum no. of constant nodes in all FARGs	273,000	273,000	273,000
Maximum no. of variable nodes in all FARGs	27,000	27,000	27,000
No. of nodes in all FARGs	191,691	149,088	45,929
No. of nodes in the FGMG	64	50	15
Computation time of generation (s)	1123	757.97	139.71
Computation time of the FGMG (s)	342.03	203.99	20.12

Table 9
Runtime performance of the proposed FGMG algorithm on Base *C* comprising 15 FARGs.

	Dataset 1	Dataset 2	Dataset 3
No. of FARGs	5	5	5
Maximum no. of nodes in each FARG	7	6	11
No. of constant nodes in each FARG	6	5	9
No. of variable nodes in each FARG	1	1	2
Maximum no. of nodes in all FARGs	35	30	55
Minimum no. of nodes in all FARGs	30	25	45
No. of nodes in all FARGs	31	28	50
No. of variable nodes in all FARGs	1	3	5
No. of nodes in the FGMG	6	6	10
Computation time of generation (s)	4.05	3.81	5.69
Computation time of the FGMG (s)	0.01	0.01	0.02

Tables 3–9 present some interesting results of the FGMG computation. For instance, we can observe that the number of nodes of all the FARGs used to compute the FGMG in *Base A* and *Base B* range from 3999 to 191,691, while the computation times of the FGMG range from 0.20 to 342.03 s. Such results indicate that the FGMG algorithm is able to handle large datasets containing a big number of large FARGs in reasonable computation times. The main reason for this is the usefulness and applicability of FARG embedding to the FGMG computation. It is important to notice that previously existing methods to compute the median graph could only be applied to small datasets due to their high computational requirements. For instance, the genetic approach [29] could only be applied to a number of nodes between 400 and 1000 (from 100 to

Table 10
Obtained gain in terms of performance and computation time (Dataset 1 of Base *A* comprising 1000 FARGs).

		Query number					Average
		1	2	3	4	5	
CBDR using sequential matching	**No. of Relevant Retrieved Images**	906	919	912	956	945	927.6
	No. of Relevant Images	1000	1000	1000	1000	1000	1000
	Accuracy (%)	90.60	91.90	91.20	95.60	94.50	92.76
	Time processing (s)	16.94	17.13	17.81	17.78	17.79	17.49
CBDR using FGMG	**No. of Relevant Retrieved Images**	1000	1000	1000	1000	1000	1000
	No. of Relevant Images	1000	1000	1000	1000	1000	1000
	Accuracy (%)	100	100	100	100	100	100
	Time processing (s)	6.15	6.02	5.86	6.23	6.17	6.08
Accuracy gain (%)							7.24
Gain in computation time (%)							65.20

Table 11
Obtained gain in terms of performance and computation time (Dataset 2 of Base *A* comprising 1000 FARGs).

		Query Number					Average
		1	2	3	4	5	
CBDR using sequential matching	**No. of Relevant Retrieved Images**	778	825	833	774	746	791.2
	No. of Relevant Images	1000	1000	1000	1000	1000	1000
	Accuracy (%)	77.80	82.50	83.30	77.40	74.60	79.12
	Time processing (s)	17.76	18.76	17.80	17.91	16.94	17.84
CBDR using FGMG	**No. of Relevant Retrieved Images**	1000	1000	1000	1000	1000	1000
	No. of Relevant Images	1000	1000	1000	1000	1000	1000
	Accuracy (%)	100	100	100	100	100	100
	Time processing (s)	5.77	5.81	6.30	5.65	5.54	5.81
Accuracy gain (%)							20.88
Gain in computation time (%)							67.39

Table 12
Obtained gain in terms of performance and computation time (Dataset 3 of Base *A* comprising 1000 FARGs).

		Query number					Average
		1	2	3	4	5	
CBDR using sequential matching	**No. of Relevant Retrieved Images**	556	575	593	626	453	560.6
	No. of Relevant Images	1000	1000	1000	1000	1000	1000
	Accuracy (%)	55.60	57.50	59.30	62.60	45.30	56.06
	Time processing (s)	18.44	23.34	18.42	17.95	17.85	19.20
CBDR using FGMG	**No. of Relevant Retrieved Images**	1000	1000	1000	1000	1000	1000
	No. of Relevant Images	1000	1000	1000	1000	1000	1000
	Accuracy (%)	100	100	100	100	100	100
	Time processing (s)	10.79	6.30	5.90	5.91	6.01	6.98
Accuracy gain (%)							43.94
Gain in computation time (%)							63.63

300 nodes per graph), while the computation times of the median graph range from 13 to 3000 s.

Experiment 3: Merits of FGMGs in CBDR

The third experiment which has been performed presents an example of an application of the proposed FGMG algorithm to the CBDR problem. This experiment aims at comparing the accuracy and time complexity of a CBDR with or without using FGMG as prototype. It is important to notice that each CBDR operation (query generation and retrieving process) was run five times for each dataset in *Base A* and *Base B* in order to better evaluate the retrieval results of these two synthetic databases. Tables 10–18 give examples of the retrieval results for each query and the average values obtained from these queries.

- *Base A:*

The obtained retrieval results from *Base A* are reported in Tables 19–21 and illustrated in Figs. 22 and 23.

- *Base B:*

The obtained retrieval results from *Base B* are reported in Tables 22–24 and illustrated in Figs. 24 and 25.

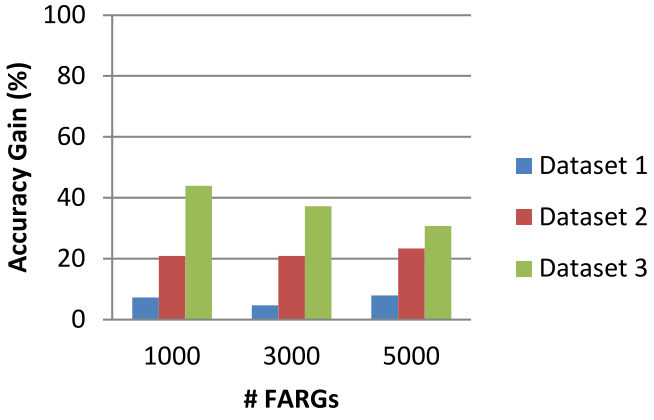

Fig. 22. Accuracy gain (*Base A* comprising 1000 to 5000 FARGs).

- *Base C:*

The obtained retrieval results from *Base C* are reported in Table 25 and illustrated in Fig. 26.

Table 13
Obtained gain in terms of performance and computation time (Dataset 1 of Base *A* comprising 3000 FARGs).

		Query number					Average
		1	2	3	4	5	
CBDR using sequential matching	**No. of Relevant Retrieved Images**	2857	2826	2863	2850	2903	2859.8
	No. of Relevant Images	3000	3000	3000	3000	3000	3000
	Accuracy (%)	95.23	94.20	95.43	95.00	96.77	95.33
	Time processing (s)	120.99	122.54	122.58	121.88	129.62	123.52
CBDR using FGMG	**No. of Relevant Retrieved Images**	3000	3000	3000	3000	3000	3000
	No. of Relevant Images	3000	3000	3000	3000	3000	3000
	Accuracy (%)	100	100	100	100	100	100
	Time processing (s)	41.56	41.43	42.46	41.45	42.77	41.94
Accuracy gain (%)							4.67
Gain in computation time (%)							66.05

Table 14
Obtained gain in terms of performance and computation time (Dataset 2 of Base *A* comprising 3000 FARGs).

		Query number					Average
		1	2	3	4	5	
CBDR using sequential matching	**No. of Relevant Retrieved Images**	2607	2032	2477	2402	2347	2373
	No. of Relevant Images	3000	3000	3000	3000	3000	3000
	Accuracy (%)	86.90	67.73	82.56	80.06	78.23	79.10
	Time processing (s)	118.24	119.61	119.53	119.64	231.59	141.72
CBDR using FGMG	**No. of Relevant Retrieved Images**	3000	3000	3000	3000	3000	3000
	No. of Relevant Images	3000	3000	3000	3000	3000	3000
	Accuracy (%)	100	100	100	100	100	100
	Time processing (s)	39.33	39.25	39.21	40.08	89.04	49.38
Accuracy gain (%)							20.90
Gain in computation time (%)							65.15

Table 15
Obtained gain in terms of performance and computation time (Dataset 3 of Base *A* comprising 3000 FARGs).

		Query number					Average
		1	2	3	4	5	
CBDR using sequential matching	**No. of Relevant Retrieved Images**	1652	1646	1601	1828	1874	1720.2
	No. of Relevant Images	3000	3000	3000	3000	3000	3000
	Accuracy (%)	55.06	54.86	53.36	60.93	62.46	57.34
	Time processing (s)	276.98	275.14	273.16	272.53	282.34	276.03
CBDR using FGMG	**No. of Relevant Retrieved Images**	1689	3000	3000	3000	3000	2737.8
	No. of Relevant Images	3000	3000	3000	3000	3000	3000
	Accuracy (%)	56.30	100	100	100	100	91.26
	Time processing (s)	186.33	90.57	90.20	90.79	91.02	109.78
Accuracy gain (%)							37.17
Gain in computation time (%)							60.23

Table 16
Obtained gain in terms of performance and computation time (Dataset 1 of Base *A* comprising 5000 FARGs).

		Query number					Average
		1	2	3	4	5	
CBDR using sequential matching	**No. of Relevant Retrieved Images**	4732	4585	4633	4546	4539	4607
	No. of Relevant Images	5000	5000	5000	5000	5000	5000
	Accuracy (%)	94.64	91.70	92.66	90.92	90.78	92.14
	Time processing (s)	303.94	301.78	302.34	300.53	314.18	304.55
CBDR using FGMG	**No. of Relevant Retrieved Images**	5000	5000	5000	5000	5000	5000
	No. of Relevant Images	5000	5000	5000	5000	5000	5000
	Accuracy (%)	100	100	100	100	100	100
	Time processing (s)	102.87	101.25	101.51	102.05	104.70	102.48
Accuracy gain (%)							7.86
Gain in computation time (%)							66.35

We performed CBDR experiments using two approaches: a CBDR using the FGMG as a dataset representative and an exhaustive and sequential CBDR. The first approach presents the advantage that the number of comparisons between FARGs is greatly reduced, since each query FARG is compared only to a small number of FARGs, while with the second approach the query FARG is compared with every FARG of all datasets. The obtained results show that a CBDR using the FGMG as a dataset representative yields better results than an exhaustive and sequential retrieval in terms of accuracy and computation time. With these results in hand, we can conclude that we obtain good approximations of the FGMG that can be effectively computed by the FGMG computation algorithm proposed in this paper. In addition, we have applied the FGMG computation to synthetic and real databases containing a

Table 17
Obtained gain in terms of performance and computation time (Dataset 2 of Base *A* comprising 5000 FARGs).

		Query number					Average
		1	2	3	4	5	
CBDR using sequential matching	**No. of Relevant Retrieved Images**	3901	3955	3303	4217	3802	3835.6
	No. of Relevant Images	5000	5000	5000	5000	5000	5000
	Accuracy (%)	78.02	79.10	66.06	84.34	76.04	76.71
	Time processing (s)	300.15	305.31	302.59	306.18	302.58	303.36
CBDR using FGMG	**No. of Relevant Retrieved Images**	5000	5000	5000	5000	5000	5000
	No. of Relevant Images	5000	5000	5000	5000	5000	5000
	Accuracy (%)	100	100	100	100	100	100
	Time processing (s)	100.93	101.06	100.26	101.84	99.67	100.75
Accuracy gain (%)							23.29
Gain in computation time (%)							66.79

Table 18
Obtained gain in terms of performance and computation time (Dataset 3 of Base *A* comprising 5000 FARGs).

		Query number					Average
		1	2	3	4	5	
CBDR using sequential matching	**No. of Relevant Retrieved Images**	2558	2787	3063	2941	2971	2864
	No. of Relevant Images	5000	5000	5000	5000	5000	5000
	Accuracy (%)	51.16	55.74	61.26	58.82	59.42	57.28
	Time processing (s)	341.77	306.37	300.42	300.37	308.39	311.46
CBDR using FGMG	**No. of Relevant Retrieved Images**	2560	5000	3104	5000	5000	4132.8
	No. of Relevant Images	5000	5000	5000	5000	5000	5000
	Accuracy (%)	51.20	100	62.08	100	100	82.66
	Time processing (s)	201.68	101.64	202.99	99.88	100.35	141.31
Accuracy gain (%)							30.70
Gain in computation time (%)							54.63

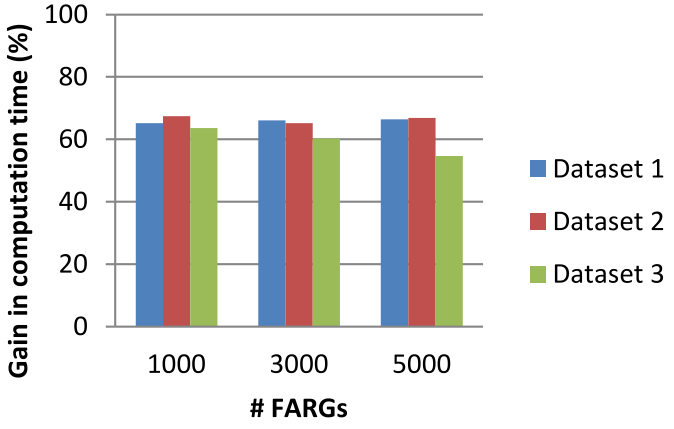

Fig. 23. Gain in computation time (*Base A* comprising 1000 to 5000 FARGs).

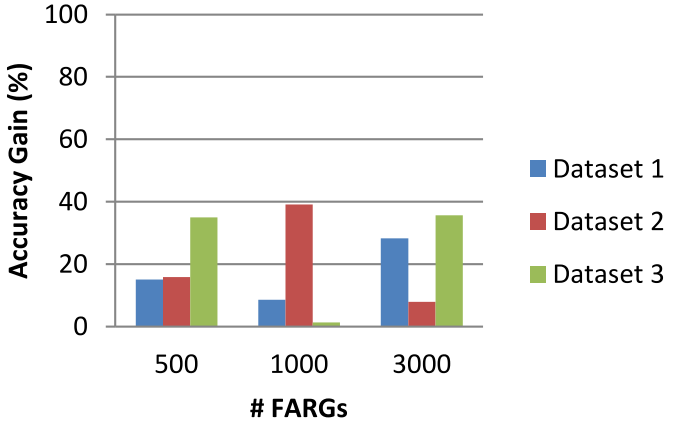

Fig. 24. Accuracy gain (*Base B* comprising 500 to 3000 FARGs).

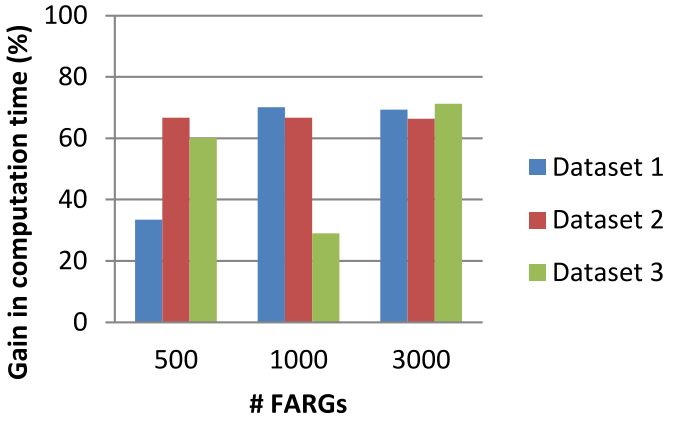

Fig. 25. Gain in computation time (*Base B* comprising 500 to 3000 FARGs).

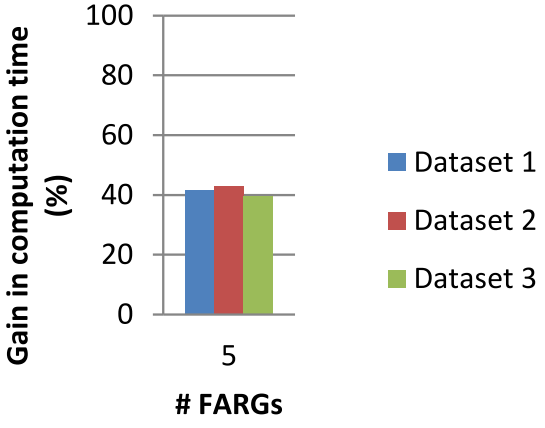

Fig. 26. Gain in computation time (*Base C* comprising 15 FARGs).

Table 19
Obtained performance and computation time according to the input number of FARGs in Dataset 1 of Base *A*.

		No. of FARGs in each dataset		
		1000	3000	5000
CBDR using sequential matching	**Accuracy (%)**	92.76	95.33	92.14
	Time processing (s)	17.49	123.52	304.55
CBDR using FGMG	**Accuracy (%)**	100	100	100
	Time processing (s)	6.08	41.94	102.48
Accuracy gain (%)		7.24	4.67	7.86
Gain in computation time (%)		65.20	66.05	66.35

Table 20
Obtained performance and computation time according to the input number of FARGs in Dataset 2 of Base *A*.

		No. of FARGs in each dataset		
		1000	3000	5000
CBDR using sequential matching	**Accuracy (%)**	79.12	79.10	76.71
	Time processing (s)	17.84	141.72	303.36
CBDR using FGMG	**Accuracy (%)**	100	100	100
	Time processing (s)	5.81	49.38	100.75
Accuracy gain (%)		20.88	20.90	23.29
Gain in computation time (%)		67.39	65.15	66.79

Table 21
Obtained performance and computation time according to the input number of FARGs in Dataset 3 of Base *A*.

		No. of FARGs in each dataset		
		1000	3000	5000
CBDR using sequential matching	**Accuracy (%)**	56.06	57.34	57.28
	Time processing (s)	19.20	276.03	311.46
CBDR using FGMG	**Accuracy (%)**	100	91.26	82.66
	Time processing (s)	6.98	109.78	141.31
Accuracy gain (%)		43.94	37.17	30.70
Gain in computation time (%)		63.63	60.23	54.63

Table 22
Obtained performance and computation time according to the input number of FARGs in Dataset 1 of Base *B*.

		No. of FARGs in each dataset		
		1000+	500	3000
CBDR using sequential matching	**Accuracy (%)**	91.40	45.20	71.77
	Time processing (s)	39.54	9.53	713.56
CBDR using FGMG	**Accuracy (%)**	100	53.20	100
	Time processing (s)	11.79	6.34	218.57
Accuracy gain (%)		8.60	15.04	28.23
Gain in computation time (%)		70.17	33.47	69.37

Table 23
Obtained performance and computation time according to the input number of FARGs in Dataset 2 of Base *B*.

		No. of FARGs in each dataset		
		1000	500	3000
CBDR using sequential matching	**Accuracy (%)**	60.90	84.20	92.10
	Time processing (s)	60.27	13.11	927.01
CBDR using FGMG	**Accuracy (%)**	100	100	100
	Time processing (s)	20.04	4.37	312.01
Accuracy gain (%)		39.10	15.80	7.90
Gain in computation time (%)		66.74	66.66	66.34

Table 24
Obtained performance and computation time according to the input number of FARGs in Dataset 3 of Base *B*.

		No. of FARGs in each dataset		
		1000	500	3000
CBDR using sequential matching	**Accuracy (%)**	44.30	65.00	64.37
	Time processing (s)	78.81	17.70	349.45
CBDR using FGMG	**Accuracy (%)**	44.90	100	100
	Time processing (s)	55.96	7.06	100.60
Accuracy gain (%)		1.34	35.00	35.63
Gain in computation time (%)		28.99	60.09	71.21

Table 25
Obtained performance and computation time for each dataset in Base *C*.

		Dataset 1	Dataset 2	Dataset 3
CBDR using sequential matching	**Accuracy (%)**	100	100	100
	Time processing (s)	0.10	0.10	0.11
CBDR using FGMG	**Accuracy (%)**	100	100	100
	Time processing (s)	0.05	0.05	0.06
Accuracy gain (%)		0	0	0
Gain in computation time (%)		41.56	42.98	39.70

large number of FARGs with no constraints regarding the number of nodes and edges. The FARG representation allows node and edge features to be represented by fuzzy concepts such as “Near” and “Far”, “Big” and “Small”, etc. Up to now, some of the existing algorithms could only be applied to very limited sets of labeled graphs with no fuzzy attributes in neither the nodes nor the edges. All input FARGs have been embedded into a suitable vector space in order to improve the accuracy and speed of document image retrieval processing. Also, different weights have been associated to FGMG nodes in the final step of the proposed FGMG computation algorithm in order to rather focus on the most important FGMG nodes in the retrieval process. As a future work, more visual (e.g. color and texture) and structural (e.g. inclusion relationship) attributes might be incorporated in the proposed FARG representation. Such new attributes may lead to more accurate FARG representations of document images and more computation time and memory consumption. In addition, the obtained results open the possibility of extending the application of the proposed FGMG algorithm to other real applications where a representative of a set of graphs is needed.

7. Conclusion

In this paper, we have proposed a new algorithm for the computation of the FGMG on large-scale document image databases. This algorithm is based on embedding of input FARGs into an n-dimensional vector space. The vectorial representation of input FARGs aims to keep advantages from both the graph and the vector domains (power of representation of graphs and easiness manipulation of the vector representations). The proposed algorithm is composed of three main steps. It starts by mapping each FARG from graph space to a set of feature vectors into a suitable vector space. The obtained vectors can be graphically represented by points in the vector space. The second step consists of clustering the obtained points into a number of clusters using a clustering method. In this study, we have used the FCM algorithm for clustering. Then, we computed the weights of the obtained clusters in order to rather focus on the most important clusters in the retrieval process. Each obtained weight becomes a new attribute of node in FGMG. Finally, the obtained clusters become the nodes of the FGMG. The proposed algorithm can be applied to many appli-

cations where a representative of a set could be needed. In this paper, we studied its applicability to the CBDR problem. Experiments on two synthetic databases and one real database containing a large number of FARGs with large sizes demonstrated that our algorithm is able to obtain good approximations of the FGMG. The experimental results prove the scalability and effectiveness of the proposed algorithm in terms of gains in accuracy and time processing.

In spite of the advantages of the FGMG construction we have introduced, we believe there are still some improvements that can be further developed in the future. A future research line is to extend the FARG representation in order to be able to deal with FARGs with more visual and structural attributes. Further, in spite of the good properties of the proposed similarity function between two FARGs, it would be desirable to extend the obtained results to other similarity functions. Furthermore, the new FGMG computation algorithm potentially allows the use of the FGMG in other graph-based applications in pattern recognition and machine learning that require to compute a FGMG, for instance, classification and clustering using big real databases.

References

[1] S. Pattanaik, D. Bhalke, Beginners to content based image retrieval, Ijsret. Org. 1 (2012) 40–44.
[2] G.F. Ahmed, R. Barskar, A study on different image retrieval techniques in image processing, Int. J. Soft Comput. Eng. 1 (2011) 247–251.
[3] R. Datta, D. Joshi, J. Li, J.Z. Wang, Image retrieval: ideas, Influences, and trends of the new age, ACM Comput. Surv. 40 (2008) 1–60.
[4] N.V. Hoàng, V. Gouet-Brunet, M. Rukoz, M. Manouvrier, Embedding spatial information into image content description for scene retrieval, Pattern Recogn. 43 (2010) 3013–3024.
[5] E.G. Karakasis, A. Amanatiadis, A. Gasteratos, S.A. Chatzichristofis, Image moment invariants as local features for content based image retrieval using the Bag-of-Visual-Words model, Pattern Recognit. Lett. 55 (2015) 22–27.
[6] J. Kumar, P. Ye, D. Doermann, Structural similarity for document image classification and retrieval, Pattern Recognit. Lett. 43 (2014) 119–126.
[7] L. Liu, Y. Lu, C.Y. Suen, Near-duplicate document image matching: a graphical perspective, Pattern Recognit. 47 (2014) 1653–1663.
[8] Y. Liu, D. Zhang, G. Lu, W.-Y. Ma, A survey of content-based image retrieval with high-level semantics, Pattern Recognit. 40 (2007) 262–282.
[9] N. Singhai, S.K. Shandilya, A survey on: content based image retrieval systems, Int. J. Comput. Appl. 4 (2010) 22–26.
[10] Y. Mistry, D.T. Ingole, Survey on content based image retrieval systems, Int. J. Innov. Res. Comput. Commun. Eng. 2 (2013) 1827–1836.
[11] K. Riesen, H. Bunke, Graph classification based on vector space embedding, IJPRAI 23 (2009) 1053–1081.
[12] D. Conte, P. Foggia, C. Sansone, M. Vento, Thirty years of graph matching in pattern recognition, Int. J. Pattern Recognit. Artif. Intell. 18 (2004) 265–298.
[13] R. Chaieb, K. Kalti, N. Essoukri Ben Amara, Interactive content-based Document Retrieval using fuzzy attributed relational graph matching, in: 2015 13th Int. Conf. Doc. Anal. Recognit., IEEE, 2015, pp. 921–925.
[14] M.M. Luqman, J.Y. Ramel, J. Lladós, T. Brouard, Fuzzy multilevel graph embedding, Pattern Recognit. 46 (2013) 551–565.
[15] S. Philipp-Foliguet, J. Gony, P.-H. Gosselin, FReBIR: an image retrieval system based on fuzzy region matching, Comput. Vis. Image Underst. 113 (2009) 693–707.
[16] A. Hlaoui, S. Wang, Median graph computation for graph clustering, Soft Comput. 10 (2006) 47–53.
[17] S. Jouili, S. Tabbone, in: A Hypergraph-Based Model for Graph Clustering: Application to Image Indexing, Springer, Berlin Heidelberg, 2009, pp. 360–368.
[18] S. Jouili, S. Tabbone, V. Lacroix, Median Graph Shift: A New Clustering Algorithm for Graph Domain, (2010) 1051–4651.
[19] M. Ferrer, E. Valveny, F. Serratosa, Spectral median graphs applied to graphical symbol recognition, in: Proc. 11th Iberoam. Conf. Prog. Pattern Recognition, Image Anal. Appl., Springer-Verlag, 2006, pp. 774–783.
[20] J. Gibert, E. Valveny, H. Bunke, Embedding of graphs with discrete attributes via label frequencies, IJPRAI 27 (2013).
[21] J. Gibert, E. Valveny, H. Bunke, L. Brun, in: Graph Clustering through Attribute Statistics Based Embedding, Springer, Berlin Heidelberg, 2013, pp. 302–309.
[22] J. Gibert, E. Valveny, H. Bunke, Graph embedding in vector spaces by node attribute statistics, Pattern Recogn. 45 (2012) 3072–3083.
[23] M. Ferrer, E. Valveny, F. Serratosa, I. Bardají, H. Bunke, in: Graph-Based k-Means Clustering: A Comparison of the Set Median Versus the Generalized Median Graph, Springer, Berlin Heidelberg, 2009, pp. 342–350.
[24] M. Ferrer, H. Bunke, An iterative algorithm for approximate median graph computation, in: 2010 20th Int. Conf. Pattern Recognit., IEEE, 2010, pp. 1562–1565.
[25] M. Ferrer, E. Valveny, F. Serratosa, K. Riesen, H. Bunke, Generalized median graph computation by means of graph embedding in vector spaces, Pattern Recogn. 43 (2010) 1642–1655.
[26] R. Raveaux, S. Adam, P. Héroux, É. Trupin, Learning graph prototypes for shape recognition, Comput. Vis. Image Underst. 115 (2011) 905–918.
[27] L.M. Musmanno, C.C. Ribeiro, Heuristics for the generalized median graph problem, Eur. J. Oper. Res. 1 (2016) 1–10.
[28] M. Ferrer, E. Valveny, F. Serratosa, Median graph: a new exact algorithm using a distance based on the maximum common subgraph, Pattern Recognit. Lett. 30 (2009) 579–588.
[29] M. Ferrer, E. Valveny, F. Serratosa, Median graphs: a genetic approach based on new theoretical properties, Pattern Recogn. 42 (2009) 2003–2012.
[30] M. Ferrer, F. Serratosa, A. Sanfeliu, Synthesis of median spectral graph, in: J.S. Marques, N.P. de la Blanca, P. Pina (Eds.), Pattern Recognit. Image Anal. Second Iber. Conf. IbPRIA 2005, Estoril, Port. June 7–9, 2005, Proceedings, Part {II}, Eds., Springer, 2005, pp. 139–146.
[31] M. Ferrer, F. Serratosa, E. Valveny, Evaluation of spectral-based methods for median graph computation, in: Pattern Recognit. Image Anal., Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 580–587.
[32] M. Ferrer, D. Karatzas, E. Valveny, I. Bardaji, H. Bunke, A generic framework for median graph computation based on a recursive embedding approach, Comput. Vis. Image Underst. 115 (2011) 919–928.
[33] M. Ferrer, I. Bardají, E. Valveny, D. Karatzas, H. Bunke, Median graph computation by means of graph embedding into vector spaces, in: Graph Embed. Pattern Anal., Springer New York, New York, NY, 2013, pp. 45–71.
[34] R. Suganya, R. Shanthi, Fuzzy C-means Algorithm-A review, Int. J. Sci. Res. Publ. 2 (2012) 2250–3153. www.ijsrp.org.
[35] N. Grover, A study of various fuzzy clustering algorithms, 5013 (2014) 177–181.
[36] G. Agam, S. Argamon, O. Frieder, D. Grossman, D. Lewis, The complex document image processing ({CDIP}) test collection project, Illinois Inst. Technol. (2006). http://ir.iit.edu/projects/CDIP.html.
[37] D. Lewis, G. Agam, S. Argamon, O. Frieder, D. Grossman, J. Heard, Building a test collection for complex document information processing, in: Proc. 29th Annu. Int. ACM SIGIR Conf. Res. Dev. Inf. Retr. - SIGIR '06, New York, USA, ACM Press, New York, 2006, p. 665.
[38] D. Doermann, E. Zotkina, H. Li, GEDI – A groundtruthing environment for document images, Ninth IAPR International Work. Doc. Anal. Syst. (DAS 2010), 2010.

Ramzi Chaieb was born in 1984 in Monastir, Tunisia. He obtained the Engineering Degree in Computer Engineering in 2012 and the Master Degree in Intelligent and Communicating Systems in 2013, from the National Engineering School of Sousse, Tunisia. Currently, he is preparing his Doctorate degree. He is also a teaching assistant at the Computer Science Department of Faculty of Science of Monastir, University of Monastir, Tunisia. His current research interests include Pattern Recognition, Content Based Document Retrieval, Document Image Analysis/Recognition, Fuzzy Graph Representation, Fuzzy Graph Matching, Graph Embedding and Fuzzy Median Graph.

Karim Kalti is an assistant professor in the Department of Computer Science of the Faculty of Science of Monastir, and member of research unit Sage (Advanced System in Electrical Engineering) team signals, image and document National Engineering School of Sousse, University of Sousse, Tunisia. His research interests include Computer vision, pattern recognition, medical imaging and data retrieval. He is a member of IEEE and his main results have been published in international journals and conferences.

Muhammad Muzzamil Luqman received his Ph.D. degree in Computer Science from François-Rabelais University of Tours, France and Autonoma University of Barcelona, Spain in 2012. He got his masters from François-Rabelais University of Tours, France in 2008. In 2004 he was awarded gold medal and academic roll of honor by Government College University Lahore, Pakistan for achieving distinction in Bachelors of Computer Science (honors). Currently Luqman is a teaching and research assistant at François-Rabelais University of Tours, France. His main research interests include Structural Pattern Recognition, Machine Learning, Document Image Analysis/Recognition and Graphics Recognition.

Mickaël Coustaty was born on December 3, 1984. He received the M.S. degree from the University of La Rochelle, La Rochelle, France, after working on symbol recognition using structural signature. He got his Ph.D. degree from the University of La Rochelle in 2011. His thesis work focused on object recognition in noisy and complex old documents in order to describe and to index them. His main research interests include image analysis and signal processing.

Jean-Marc Ogier received his Ph.D. degree in computer science from the University of Rouen, France, in 1994. During this period (1991–1994), he worked on graphic recognition for Matra Ms&I Company. From 1994 to 2000, he was an associate professor at the University of Rennes 1 during a first period (1994–1998) and at the University of Rouen from 1998 to 2001. Now full professor at the university of la Rochelle, Pr Ogier is the head of URL laboratory which gathers more than 120 members and works mainly of Document Analysis and Content Management. Author of more than 230 publications / communications, he managed several French and European projects dealing with historical document analysis, either with public institutions, or with private companies. Pr Ogier was Deputy Director of the GDR I3 of the French National Research Centre (CNRS) between 2005 and 2013. He was also Chair of the Technical Committee 10 (Graphic Recognition) of the International Association for Pattern Recognition (IAPR) from 2010 to 2015, and is the representative member of France at the governing board of the IAPR. Jean-Marc Ogier has been the general chair of the program chair of several international scientific events dealing with document analysis (DAS, ICDAR, GREC, ...) At last he is also Vice rector of the university of La Rochelle and president of VALCONUM association, which is an association aiming at fostering relations between industries and research organizations.

Najoua Essoukri Ben Amara received the B.Sc., M.S., Ph.D. and HDR degrees in Electrical Engineering, Signal Processing, System Analysis and Pattern Recognition from the National School of Engineers of Tunis, University El Manar, Tunisia, in 1985, 1986, 1999, 2004 respectively. She became a senior lecturer in July 2004 and a full Professor in October 2009 in Electrical Engineering at the National School of Engineers of Sousse-ENISo, University of Sousse, Tunisia. She is a founding member and director of the research group SAGE-Systèmes Avancés en Genie Electrique. From July 2008 to July 2011, she was the director of the ENISo. She has a wide national and international visibility as she chaired several national committees and chaired/cochaired various international scientific conferences. She acts in many program committees and belongs to several scientific ones. She was the coordinator of several European projects (Euromed $3+3$ and Tempus). She initiated multiple collaborations with international research laboratories and socioeconomic organizations. Since 2011 she is the President of the "Tunisian Association of Innovative Techniques of Sousse". Her areas of research include pattern recognition, document analysis, multimodal biometric, medical image processing, computer vision, with application to the segmentation of documents, biometric recognition, individuals, detection / monitoring multi-object where she has authored and co(authored) over 100 articles in various national and international journals and conference proceedings.