Graph Based K-Nearest Neighbor Minutiae Clustering for Fingerprint Recognition

Vaishali Pawar
Information Technology Department
NDMVP S's KBT College of Engineering
Nashik, Maharashtra, India

Mukesh Zaveri
Computer Engineering Department
SVNIT
Surat, Gujarat, India

***Abstract*—The graph is an efficient data structure to represent multi-dimensional data and their complex relations. Pattern matching and data mining are the two important fields of computer science. Pattern matching finds a particular pattern in the given input where as data mining deals with selecting specific data from the huge databases. This work contributes towards the combination of graph theory, pattern recognition and graph based databases. A variety of graph based techniques have been proposed as a powerful tool for pattern representation and classification in the past years. For a longer time graphs remained computationally expensive tool. But recently the graph based structural pattern recognition and image processing is becoming popular. The computational complexity of the graph based methods is becoming feasible due to high end new generations of the computers and the research advancements. In this work we have implemented graph based fingerprint recognition algorithm. The fingerprints are represented as attributed relational graphs. In the pattern recognition phase graph matching is applied. This study focuses on the clustering of graph databases prior to graph matching. When the structural feature set size of the data grows longer, graph matching becomes expensive. The clustering of graph databases drastically reduce the graph matching candidates.**

I. Introduction

Fingerprints play distinguishing role in biometrics. They give unique identification to the individual. They are permanent and non changing character pattern. The fingerprint recognition consists of retrieving specific fingerprints from database. The characteristics or features of a fingerprint are extracted and used in identification process.

In the past years many statistical and structural approaches were explored for fingerprint classification. Statistical methods adopt the extraction of various mathematical characteristics to classify the patterns [1]. The structural approaches make use of syntactic [2] or structural constructs of the patterns [3] for the recognition methods. In the syntactic approach fingerprints are represented as grammars and the graph parsing techniques are used to classify the graphs. However some hybrid approaches are also explored which combine the structural and statistical methods [4], [5].

The graphs play an important role in the structural and syntactic methods of fingerprint classification. In these methods the fingerprint patterns are represented by graphs or production rules in the grammar. Graph isomorphism, sub graph isomorphism or grammar parsing is used to classify the fingerprints to their respective classes. Graph isomorphism and sub graph isomorphism approaches must be able to incorporate errors and distortions for the possibility of noisy fingerprint data. The syntactic approach [6] defines grammar in terms of terminal and non terminal symbols and production rules. The complex attributes considered in syntactic patterns make them difficult to build the parsing methods. They have high computational complexity [6]. The graph based methods mainly use relational attributes of the features to build the graphs. The fingerprint applications mostly make use of attributed relational graphs to denote the relation between various features.

The combination of fingerprint features and corresponding graph model used, plays crucial role in selection of the classification algorithm.

The effective graph prototypes are described for representing "core" and "deltas" of the fingerprints in [7], [8]. The graph derived from the fingerprint is classified to its corresponding class by using inexact graph matching [20]. The use of inexact graph matching methods make the algorithm robust to absorb distortions in the fingerprints.

The fingerprint graph comparison procedure can be optimized by reorganization of the graph data in a preprocessing step. In structural pattern recognition, various arrangements of the graph data are discussed. As per the need of any application, its graph representation, distortion acceptance model, data organization and matching policy is to be employed. The graph database clustering effectively reduces the number of searches in the graph data. Various graph clustering policies like a hierarchical [9], indexed [10][11] , hashed [12] databases have been suggested. Ultimate goal is to reduce the overall graph matching time by limiting the search area.

A decomposition based hierarchical approach avoids matching each model graph individually onto the input graph [13]. The model graph is recursively decomposed offline into smaller subgraphs. At run-time, these subgraphs are matched onto the input graph and all detected subgraph isomorphisms are combined to form subgraph isomorphisms for complete model graphs. The main advantage of this scheme is that subgraphs that appear multiple times in the same or in different model graphs must be matched only once onto the input. Consequently, the corresponding subgraph isomorphism detection process will be more efficient than the sequential matching of the input graph with each of the models. When all model graphs are highly similar, the method becomes

978-1-4799-5151-2/14/$31.00 ©2014 IEEE

independent of the size of the database. and algorithm's best case complexity becomes $O(IM)$, while worst case complexity goes to $O(I^M M^2)$ [13]. For model graphs with some common substructures the algorithm is sublinearly dependent on the size of the database. Then the algorithm's best case takes $O(NIM)$ amount of time, while worst case takes $O(NI^M M^2)$ amount of time, where N is total number of model graphs in the database, I is the number of vertices in the input graph and M is number of vertices of the model graph [13]. For large model graphs the algorithm is faster than traditional algorithms due to reappearing substructures that naturally evolve in large graphs. But for unlabeled, highly connected graphs, the algorithm performs poorly [13].

Median graph is an important concept used in graph matching. It acts as a representative for a set of graphs by extracting the essential information from them into a single prototype. Given a set of graphs S, the median is derived from S which has the smallest sum of distances (SOD)[14] to all the graphs in the set. Computing the median graph is an NP Complete problem [15]. The algorithm uses the concept of dissimilarity distance to compare the two graphs. If the the median itself belongs to the set of graphs, the set median is obtained, otherwise the generalized median is obtained. The algorithm guarantees that a good solution will be found [14]. The efficiency of the algorithm depends on reduction of the set of node labels and on a search strategy aiming at exploring the most likely median graph candidates.

The graph can also be stored by means of hashing and indexing [12]. Each graph has a numerical key associated with it. A hash function maps this key onto the array of manageable size. The address thus calculated for an object is called as the hash code of the object. Database construction consists of computing such a hash-code for each object to be stored in the database. Several objects may have the same address. In such case a list of objects will be associated with each address. So the database becomes a table of lists, where each list stores a list of objects with same hash code. Indexing is the process of computing hash code for the unknown object for finding whether such object exists in the hash table or not. By using all the mathematical calculations based on Laplacian transform, [12] presented the graph hashing method. The method characterizes the graph by using its mathematical and computational properties [12]. The Laplacian matrix is a richer graph description than the well known adjacency matrix. However the indexing procedure has some limitations [12]. The decomposition of a graph into subgraphs is not optimal and it does not take into account the geometrical and structural constraints. Also it can not deal with images containing more than one object.

Even if graphs are good for feature representation, they are harder to manipulate and operate than feature vectors. Some attempts have been made to combine the best of the graph and the vector domains [15], [16] in order to get the advantages of both. The median graph can be seen as the representative of a set of graphs. These methods generate approximate median graph using real databases containing large graphs. The approximate K-Means based algorithm computes the generalized median graph [17] for graph clustering. The median graph is defined as the minimization of

$$\bar{g} = arg \min_{g \in U} \sum_{i=1}^{n} d(g, g_i) \tag{1}$$

The algorithm shows the efficiency in correctly classifying graphs into sets of clusters [17] in content-based image retrieval and analysis of DNA structures[17].

II. The graph based fingerprint recognition

A graph $G = (V, E)$ is a set of vertices V and edges E. V is the set of vertices and $E \subseteq (V \times V)$ is the set of edges of graph G. The order or size of a graph G is defined as the number of vertices of G and it is represented as $|V|$. If two vertices in G, say $u, v \in V$, are connected by an edge $e \in E$, this is denoted by $e = (u, v)$ and the two vertices are said to be adjacent to each other. Edges can be directed or undirected. Graph matching can be applied to directed and undirected graphs both. A directed graph $G = (V, E)$ is called complete when there is always an edge $(u_1, u_2) \in E = V \times V$ between any two vertices u_1, u_2 in the graph. When graphs are used to represent objects, images or fingerprints, vertices usually represent regions or features of the object or images, and edges between them represent the relations between the features.

Graph vertices and edges can contain large complex information. When this information is a label or attributes of vertices and edges, the graph is called an attributed graph. Formally an attributed graph is a graph with six-tuple $G = (V, E, u, v, L_V, L_E)$, where
o V is a finite set of nodes,
o $E \in V \times V$ is a finite set of edges,
o u is a labeling function for vertices, $V \rightarrow L_V$
o v is a labeling function for edges, $E \rightarrow L_E$
o L_V is a set of node labels, and
o L_E is a set of edge labels

Graph Isomorphism

The two graphs G and G' can be compared by determining a mapping function f which associates nodes and edges of G with nodes and edges of G' and vice versa. A graph isomorphism between two graphs G and G' is given if there exists a bijective mapping f from the nodes of G to the nodes of G' such that the structure of the edges as well as all node and edge labels are preserved under f. More precisely: A bijective function $f : V \rightarrow V'$ is a graph isomorphism from a graph $G = (V, E, u, v, L_V, L_E)$ to a graph $G' = (V', E', u', v', L_V, L_E)$ if:

1. $u(v) = u'(f(v))$ for all $v \in V$
2. for any edge $e = (v_1, v_2) \in E$ there exists an edge $e' = (f(v_1), f(v_2)) \in E'$ such that $v(e) = v(e')$; for any $e' = (v'_1, v'_2) \in E'$ there exists an edge

$e = (f^{-1}(v1'), f^{-1}(v2')) \in E$ such that $v(e') = v(e)$.

Subgraph Isomorphism

Formally, two graphs are isomorphic to each other if they are equal. Practically, for most applications this mapping is too much restrictive. In practice, graph isomorphism is of limited use and more relaxing concepts are necessary. One such approach is subgraph isomorphism. An injective function $f : V \rightarrow V'$ is a subgraph isomorphism from a graph G to a graph G' if there exists a subgraph $G_s \subseteq G'$ such that f is a graph isomorphism from G to G_s.

In this work we are presenting a new algorithm for automated graph based fingerprint representation and matching. We aim to achieve the promising results by clustering the graph feature set and then applying the graph isomorphism [20]. The algorithm proves robust for noisy images as well. We extract various structural features like ridges, end points, bifurcations of the fingerprint. These end points and bifurcations are called as minutiae. Each fingerprint is represented as an attributed relational graph [19] where these features are represented as nodes of the graph. The edges of the graph represent relation attributes between the vertices of the graph. In this way the graph structure captures the structural relationships within the fingerprint graph. The algorithm is implemented and tested using a database of real fingerprint images. We further extend our work for clustering features and thus modify the graph database itself. Most of the fingerprint recognition algorithms deal with large databases. So comparing the fingerprint with all images in the database takes long time and needs lots of computations. The novelty of algorithm lies in the following considerations.

1. It uses graph as fingerprint data structure.
2. It optimizes fingerprint search by clustering fingerprint graph feature nodes.

III. Method 1: Graph based fingerprint recognition

This is a graph based approach to compare structural fingerprint features called as minutiae. The Method 0 implements graph based algorithm given by [18]. The graph based representation of the features of the fingerprint make the procedure more flexible in nature. Each minutiae is represented as a vertex of the graph. We are using attributed relational graph. Each minutiae is labeled with its own attributes and neighborhood features. An edge between two vertices indicates that they share neighborhood information between them. The graph isomorphism algorithm [10], [11] decides similarity or dissimilarity between the patterns being compared.

The overall method involves following steps.

1. Fingerprint image acquisition
2. Preprocessing
3. Features Extraction
4. Features representation as a graph
5. Graph isomorphism algorithm to classify the fingerprint.

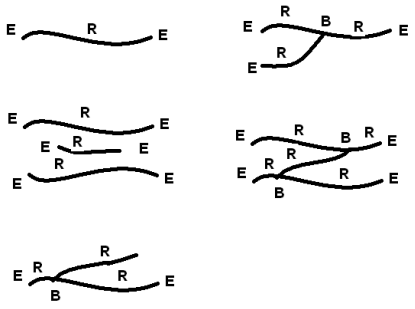

Fig. 1. Ridges, end points and bifurcations

We are using ridge (R), ridge ending (E) and ridge bifurcation (B) minutiae for fingerprint recognition. The figure 1 shows some typical examples of ridges and features extracted.

If the feature extraction phase detects 'n' minutiae for the fingerprint, an attributed relational graph with 'n' nodes is formed. The graph isomorphism is applied between the input fingerprint graph G_I and database fingerprint graphs G_D. The algorithm returns true when the input graph G_I is isomorphic to the graph G_D in the fingerprint database. The graph isomorphism is performed by using linear comparison of each of the vertex from the sample fingerprint images.

IV. Method 2: Graph based fingerprint recognition by K-NN clustering of the minutiae

When the fingerprint minutiae are more in numbers, comparing each one of them becomes very expensive for the machine. So in the second stage we clustered the graph data by using K-NN clustering based on Euclidean distance between the vertices of the graph. A features template for each fingerprint is build after clustering the neighborhood relative vertices in similar classes. The advantage of this type of clustering is that the graph comparison is first done with the cluster features template of the fingerprint instead of direct comparison of the vertices of the graph. If the cluster features match with the input fingerprint graph then only graph isomorphism is performed. Otherwise the next cluster is visited in a hope to get matching candidate in it. The algorithm involves following steps.

1. Fingerprint image acquisition
2. Preprocessing
3. Features Extraction
4. Features representation as a graph
5. KNN based graph node clustering
6. Graph isomorphism algorithm to classify the fingerprint.

A. The K-Nearest Neighbor clustering

The goal of K-NN clustering method is to simply cluster the minutiae that are K nearest neighbor in the fingerprint space. The Euclidean distance metric is chosen to determine

the closeness between the minutiae (vertices). The Euclidean distance is defined as the K-NN distance because of the structural neighborhood properties of the minutiae (or vertices). The Euclidean distance is given by:

$$d(x,y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_n - y_n)^2} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \quad (2)$$

The K-NN clustering proves to be the best due to its lesser execution time and good clustering accuracy.

We use the K-nearest neighbor algorithm (K-NN) for clustering the fingerprint graph nodes based on Euclidean distance between them. The K-nearest neighbor algorithm is the simplest clustering algorithm.

Generally a node is classified by a majority votes of its neighbors. The nodes are assigned to the class that is closest amongst its K nearest neighbors, where K is a small positive integer. If K = 1, then the node is assigned to the class of its nearest neighbor.

1. Each node within the feature set has a class label in the set, $Class = c_1, ..., c_n$.
2. The K-nearest neighbors are found by calculating the distance matrix where K is the number of neighbors.
3. The K-closest nodes are analyzed to determine which class label is the most common among the set.
4. The most common class label is then assigned to the node being analyzed.

After graph node clustering the graph based fingerprint matching is performed.

The identification of the fingerprint is performed in this module. It reads each fingerprint clustered graph template from the database linearly and compares it with the clustered input fingerprint graph template. If the initial graph template matches then only the total graph isomorphism [18] is applied. The graph matching is continued till the input fingerprint is identified otherwise no match is found.

V. Simulation Results

For the fingerprint detection we have tested the results on real life data set with 150 images. In fingerprint recognition we implemented following algorithms
Method 1 : Fingerprint recognition with graph matching [18]
Method 2 : Fingerprint recognition with K-NN clustering of the features and graph matching

Following are the results of those tests. We notice that the graph clustering is reducing the graph matching time drastically than the time taken by graph matching without clustering. The figure 2 shows sample database used. The features extraction phase extracts the minutiae in the fingerprint. It identifies ridges, endpoints and bifurcations present in the image. An

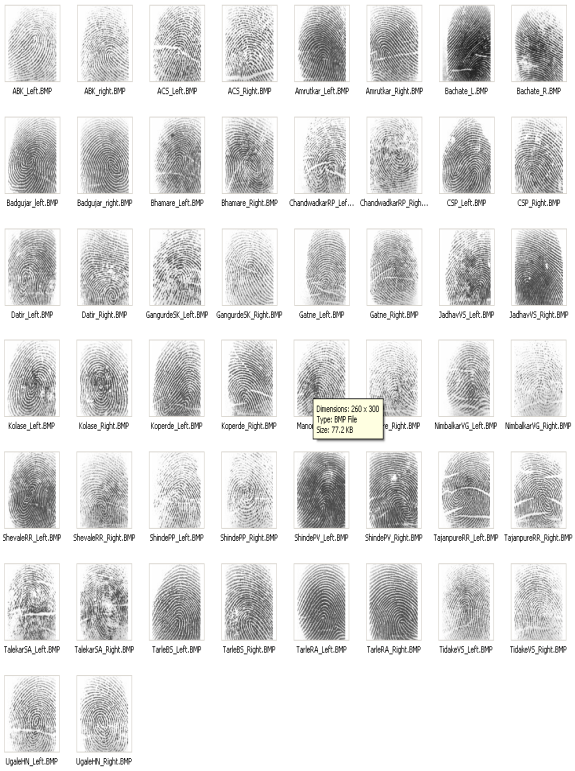

Fig. 2. Real life database of fingerprints used for testing

attributed relational graph is build for each fingerprint. These features are stored as a characteristic template for each image. The figure 3 shows the features extracted from the image. Each minutia is identified by its (x, y) position, angle, and its type as end point, ridge, bifurcation etc. Each minutia is a vertex and the features are attributes of the vertex.

```
<FingerprintTemplate Version="2" OriginalDpi="96" OriginalWidth="3
  <Minutia X="781" Y="828" Direction="151" Type="Ending" />
  <Minutia X="615" Y="193" Direction="77" Type="Ending" />
  <Minutia X="609" Y="1021" Direction="178" Type="Ending" />
  <Minutia X="1151" Y="89" Direction="205" Type="Ending" />
  <Minutia X="802" Y="859" Direction="96" Type="Ending" />
  <Minutia X="1333" Y="1109" Direction="205" Type="Ending" />
  <Minutia X="1474" Y="141" Direction="192" Type="Bifurcation" />
  <Minutia X="583" Y="1062" Direction="77" Type="Bifurcation" />
  <Minutia X="1089" Y="885" Direction="223" Type="Ending" />
  <Minutia X="885" Y="99" Direction="192" Type="Bifurcation" />
  <Minutia X="625" Y="1094" Direction="64" Type="Bifurcation" />
  <Minutia X="833" Y="552" Direction="215" Type="Ending" />
  <Minutia X="1427" Y="62" Direction="178" Type="Bifurcation" />
  <Minutia X="854" Y="641" Direction="192" Type="Ending" />
  <Minutia X="1078" Y="859" Direction="242" Type="Ending" />
  <Minutia X="562" Y="1062" Direction="64" Type="Bifurcation" />
  <Minutia X="1047" Y="516" Direction="50" Type="Ending" />
  <Minutia X="651" Y="266" Direction="64" Type="Bifurcation" />
  <Minutia X="1047" Y="729" Direction="205" Type="Bifurcation" />
  <Minutia X="1281" Y="120" Direction="168" Type="Bifurcation" />
```

Fig. 3. Minutiae features extracted like ridges, end points, bifurcations

The following graph 4 shows the time difference for graph

processing without clustering(Method 1) and graph processing with KNN clustering of features(Method 2).

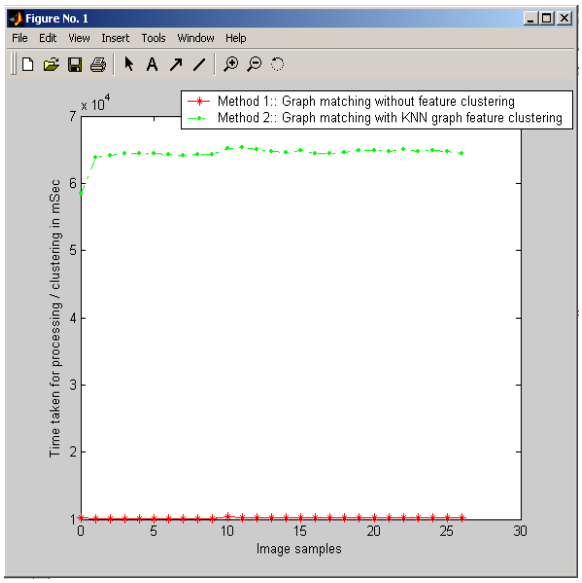

Fig. 4. The processing time comparison: Method 1 and Method 2

Similarly following graph 5 shows the time difference for graph matching without clustering(Method 1) and graph matching with KNN clustering of the features(Method 2).

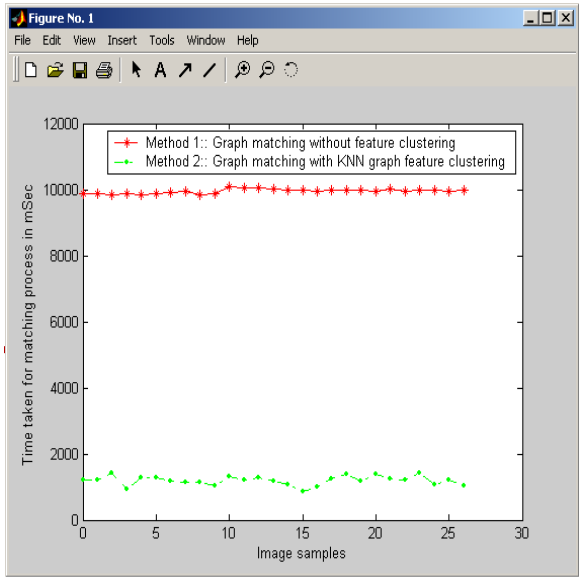

Fig. 5. The matching time comparison: Method 1 and Method 2

VI. Conclusion

In this work we have proposed and implemented graph clustering and matching algorithms. The fingerprint recognition is explored for the said area. The fingerprints are preprocessed and structural features of the minutiae are extracted. The fingerprint is represented as an attributed graph by relating all the features with each other. When a query is set for fingerprint matching graph isomorphism is applied. For the graph of size $'n'$ vertices the algorithm performs $'n*n'$ comparisons. When size of $'n'$ becomes larger, graph matching proves expensive. So we have also applied Euclidean distance based K-NN clustering to the feature set. It clusters the feature vertices in similar groups based on their neighborhood relationships. Because of the clustering, the algorithm compares cluster properties before comparing contents of the cluster individually. The approach reduces approximately one third of the time taken by comparison without clustering. We have tested the results on real time data set available with 150 fingerprints. The observation is that graph clustering techniques reduce time needed for the graph matching purpose. The results of graph matching remain almost same as the size of graph we are considering is restricted due to memory limits of the processing machine. These techniques increase the processing time though. In that case it is expected that clustering of the databases is to be done off line prior to graph matching process for best results as a combination.

In the future work we intend to analyze and test the results on larger data sets along with detail performance evaluation of the algorithms. Definitely graph based algorithms have their own pros and cons. The selection of appropriate graphs for specific applications with appropriate graph matching algorithm is the key. Its always a challenge to achieve a general platform for various applications and their needs.

References

[1] A.K. Jain, S. Prabhakar, and L. Hong, A Multichannel Approach to Fingerprint Classification, IEEE Trans. on PAMI,21 (4) 348-358, 1999

[2] K. Rao and K. Balck, Type Classification of Fingerprints: a Syntactic Approach, IEEE Trans. on PAMI, 2 (3) 223-231,1980

[3] A. Lumini, D. Maio, and D. Maltoni, Inexact graph matching for Fingerprint Classification, Machine Graphics and Vision, 8 (2) 241-248, 1999

[4] A. Serrau, G.L. Marcialis, H. Bunke, and F. Roli, An experimental comparison of fingerprint classification methods using graphs, 5th Int. Work. on Graph-based Representations in Pattern Recognition GbR05, L. Brun and M. Vento Eds., Springer LNCS 3434, pp. 281-290, 2005

[5] R. Cappelli, D. Maio, and D. Maltoni, A Multi-Classifier Approach to Fingerprint Classification, Patt. Anal. and Appl.,5 (2) 136-144, 2002

[6] K. Rao and K. Balck, Type Classification of Fingerprints: a Syntactic Approach, IEEE Trans. on PAMI, 2 (3) 223-231,1980

[7] M. Neuhaus and H. Bunke, A Graph Matching Based Approach to Fingerprint Classification Using Directional Variance, Proc. of 5th Int. Conf. on Audio- and Video-Based Biometric Person Authentication AVBPA05, T. Kanade, A.K. Jain, N. Ratha Eds., Springer LNCS 3546, pp. 191-200, 2005

[8] M. Neuhaus and H. Bunke, Graph-Based Multiple Classifier System - A Data Levels Fusion Approach, Proc. of 13-th Int. Conf. on Image Analysis and Processing ICIAP05, F. Roli and S. Vitulano Eds., Springer LNCS 3617, pp.479-486, 2005

[9] L. Shapiro, R. Haralick, Organization of relational models for scene analysis, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 3, pp. 595-602, 1982

[10] J. Llados, E. Marti, and J.J. Villanueva, Symbol recognition by error-tolerant subgraph matching between region adjacency graphs, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 10,pp. 1137-1143,2001

[11] G. Sanchez, J. Llados, and K. Tombre, An error correction graph grammar to recognize textured symbols, LNCS, Graphics Recognition: Algorithms and Applications, Vol. 2390, pp. 128-138, 2002

[12] H. Sossa and R. Horaud, Model indexing: The graph-hashing approach, Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 811-814,1992

[13] Bruno T. Messmer and Horst Bunke, Efficient Subgraph Isomorphism Detection: A Decomposition Approach, IEEE transactions on Knowledge and Data Engineering, Vol. 12, No. 2, pp.307-323, 2000
[14] A. Hlaoui and S. Wang, A New Median Graph Algorithm, Graph Based Representations in Pattern Recognition LNCS, Vol. 2726, pp. 225-234, 2003
[15] Miquel Ferrer and Horst Bunke, An Iterative Algorithm for Approximate Median Graph Computation, International Conference on Pattern Recognition, pp. 1562-1565, 2010
[16] K. Riesen and H. Bunke, Dissimilarity Based Vector Space Embedding of Graphs Using Prototype Reduction Schemes , Machine Learning and Data Mining in Pattern Recognition, LNCS, Vol. 5632, pp. 617-631, 2009
[17] A. Hlaoui and S. Wang, Approximate Graph Matching and Computing Median Graph for Graph Clustering, The international workshop on multidisciplinary image, video, and audio retrieval and mining, Sherbrooke, Canada, 2004
[18] D. K Isenor and S. G. Zaky, Fingerprint identification using graph matching, Pattern Recognition, Vol. 19, No. 2, pp. 113-122, 1986.
[19] L. Changhua, Y. Bing, X. Weixin, Online hand-sketched graphics recognition based on attributed relational graph matching, 3rd World Congress on Intelligent Control and Automation,pp. 2549 - 2553. 2000
[20] Pawar, V.S.; Zaveri, M.A., "Graph based pattern matching," Fuzzy Systems and Knowledge Discovery (FSKD), 2011 Eighth International Conference on , vol.2, no., pp.1022,1026, 26-28 July 2011