Database Development by Learning from Sample for Handwritten Chinese Character Recognition

Ying Ren and Si Wei Lu
Department of Computer Science Memorial University of Newfoundland
St. John's, Newfoundland, Canada A1C 5S7

ABSTRACT

For handwritten Chinese character recognition, a well-organized model database is very essential to effectively deal with the large character set. Because written characters have a great variety of character shapes, styles, position of radicals, and directions or lengths of strokes, a simplest way to achieve recognition accuracy is to develop models from written samples. However, this may increase the number of models for each character in the database and slow down the search process of the database. In this paper, Chinese characters are represented by hierarchical attributed graphs (HAGR), a graph synthesis algorithm is introduced to build models for new characters and to update the models in a database to represent more handwriting variations. With this approach, only one model each for most of the characters is required.

1 Introduction

The database techniques have been widely applied in Chinese information processing. Most of the applications have been aimed on input encoding, display, electronic publishing, information interchange, word processing, machine translation of Chinese as well as Chinese character recognition. Handwritten Chinese characters varies greatly in shapes, radicals and strokes. Therefore, to develop a database which covers most variation of each character is very critical to improve the recognition speed and rate. Basically, the approaches for model database development can be divided into two main streams: human approaches and machine approaches. In most systems, the models are derived with the human approaches. However, human analyzers process data very slow and make errors very easily. The errors are very difficult to be checked and recovered. Hence, machine approaches become more attractive because they provide significantly more advantages on speed and accuracy than human approaches. Handwritten Chinese characters are non-rigid objects and for each Chinese character, it is very difficult to determine how many handwritten variations due to the geometric diversity of character shapes produced by writers with the different writing habits and styles. Thus, the best way is to develop models by learning form the samples. Learning from the samples has two advantages: (1) knowledge accumulation, the recognition rate of a Chinese character is increased, when more handwritten samples of this character are learned; (2) condense representation, as only the useful features should be extracted from the samples, learning from the samples provides the possibility to use few models to cover the variation of each character.

Based on HAGR [1], we propose a new machine approach to develop models in the database. During the learning process, users introduce into the system many samples of the same character and the identity of the character. The HAGs of samples of the input character are synthesized. From the synthesized results, a model HAG is built for the character in the database. Using a graph synthesis, for most character, only one model need to be built for each of them and storage required is significant reduced. This approach is better than the existing approach [2-4] which require, for single character, many models to cover the variations included in its samples.

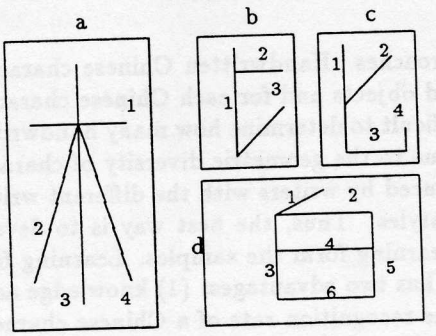
2 HAGR of handwritten Chinese characters

A hierarchical attributed graph is a two level graph. The high level describes radicals and relations between radicals, and the low level describes strokes and the relations between the strokes in a radical.

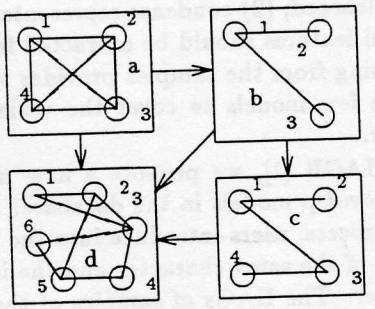
Definition 1: A radical attributed graph $G_b = (V_{b1}, E_b)$ is an attributed graph which represents a radical in a character. (1) V_b is a set of attributed vertices representing the strokes in the radical, with two attributes to describe the type (dot or line) and

the direction (H, RD, V, or LD). (2) E_b is the set of attributed edges representing the relations between the strokes such as T-, L-, or X-joints.

Definition 2: A hierarchical attributed graph is an attributed graph $H = (V_H, A_H)$ which represents a Chinese character. The vertex represents a radical and the arc represents the spatial relation between two adjacent radicals (LR, TB, and BC). Fig 1 show the HAGR of a Chinese character.



(1) Input character



(2) HAGR for the character

Figure 1. An input character and its HAGR

Hierarchical attributed graph(HAG) for the input character:

$H = (V_H, A_H)$, $V_H = \{v_a, v_b, v_c, v_d\}$,
$A_H = \{e_{ab}, e_{ad}, e_{bc}, e_{bd}, e_{cd}\}$,
$v_a =< (number\_of\_strokes, 4) >, \cdots\cdots$
$v_d =< (number\_of\_strokes, 6) >$,
$e_{ab} =< (spatial\_relation, left\_right) >, \cdots\cdots$,
$e_{cd} =< (spatial\_relation, top\_bottom) >$.

Radical attributed graph(RAG) for :

$G_a = (V_a, E_a)$, $V_a = \{v_1, v_2, v_3, v_4\}$
$E_a = \{e_{12}, e_{13}, e_{14}, e_{23}, e_{24}, e_{34}\}$,
$v_1 =< (type, line), (orientation, vertical) >$,
$v_2 =< (type, line), (orientation, horizontal) > \cdots\cdots$
$e_{12} =< (relation, T\_junction) >$,
$e_{13} =< (relation, X\_junction) > \cdots\cdots$

Definition 3: A radical adjacency matrix(RAM) is an adjacency matrix with diagonal entries describing the attributes of vertices and non-diagonal entries describing the edges in a radical attributed graph.

Definition 4: A character adjacency matrix(CAM) is an attributed adjacency matrix with diagonal entries describing the attributes of vertices and non-diagonal entries describing the arcs in a hierarchical attributed graph.

Definition 5: For a character with several radicals, any other character that shares at least one or more common radicals is defined as co-radical character of this character.

Definition 6: Radical-derived characters of a radical are the characters containing the radical.

Definition 7: A radical is defined as learned radical, if its model exists in the model database.

3 Table network organization of model database

The samples of an input character (a character will be learned) are well selected by a teacher. Only positive samples are learned by the system. If there exists a model of the input character in the database, it may be modified to include the samples by graph synthesis. In order to determine whether a model of the input character exists in the database, the database has to be searched with the HAGs of these samples. If there is no model matched by any HAG of samples, the HAGs will be inserted in the database as new models.

Moreover, the inconsistency for common radicals of radical-derived characters would be produced in the database. All radical-derived characters with a common radical in the database have to be updated to include the new sample RAG. It is very difficult to find all these radical-derived characters without traversing all the models in the database. To overcome these difficulties, several new strategies are proposed: (1) Sample acquisition: during the learning phase, the teacher should provide the character code and the radical codes of component radicals for the input character. (2) Table network organization: provides direct and fast accessing and updating of the database. (3) Integrating algorithm: use graph synthesis to update the model database so that the inconsistencies and the redundancies can be avoided.

The network consists of two hosts: the host for characters and the host for radicals. At the host for characters, there are three kinds of tables: character table, MHAG subtables, and component RAG subtables. Character table is a linked list which stores the models of characters. At the host for radicals, there are three kinds of tables: radical table, index subtables, and MRAG subtables. These tables store the information about all the radicals in the database. Figure 2. Illustrates the overall table network organization of model database.

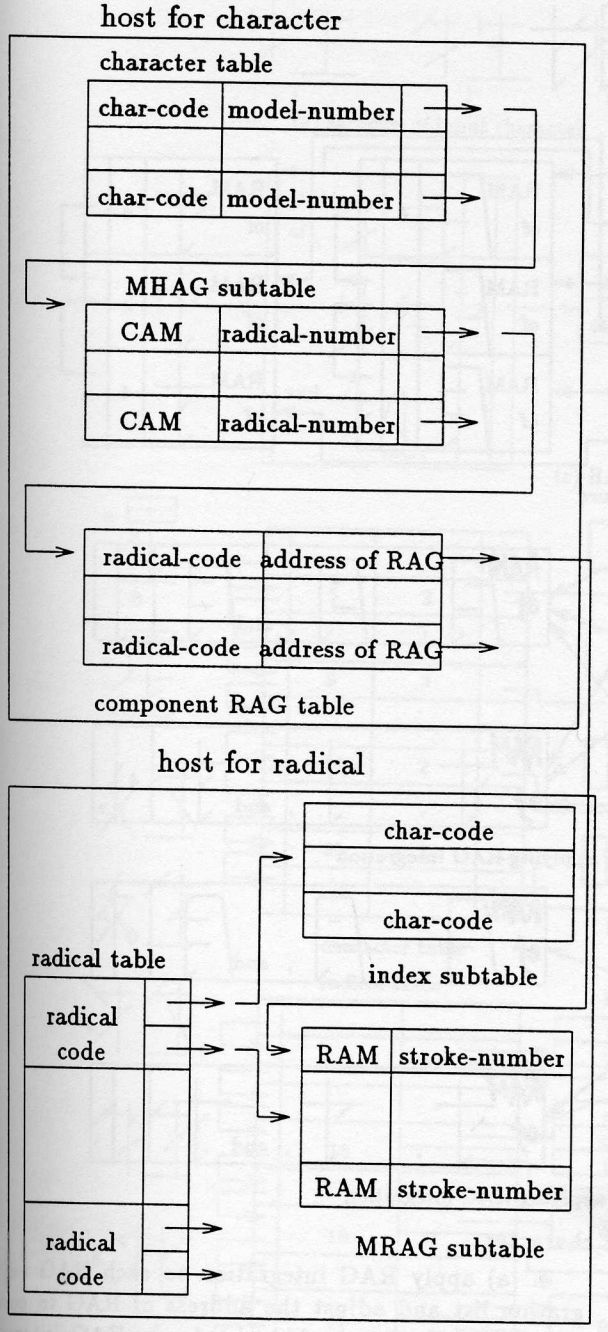


Figure 2. Table network organizaton of model database.

4 RAG and HAG integrations

RAG integration and HAG integration are proposed to synthesize radical attributed graphs and hierarchical attributed graphs. The RAGs to be synthesized are stored in a list called the RAG integration list (see Fig. 3). If two RAGs in the RAG integration list can be synthesized, they are replaced by the convergent RAG. This process is continued until there are no two RAGs that can be integrated in the RAG integration list. The HAG integration is designed to synthesize the hierarchical attributed graphs. HAGs are stored in the HAG integration structure which consists of a HAG integration subtable, component RAG subtables, and RAG integration lists. A HAG integration subtable is similar to a MHAG subtable. Each record of the component RAG subtable consists of two fields: a radical-code and an address of RAG which is stored in a RAG integration list. The conditions that two HAGs in a HAG integration subtable can be integrated are: (1) they have the same CAM, and (2) both of their component RAG subtables are the same. Under these conditions, both HAGs are replaced by their convergent HAG. Similarly, the process continued until no two HAGs can be synthesized in the HAG integration subtable.

5 Modification of models

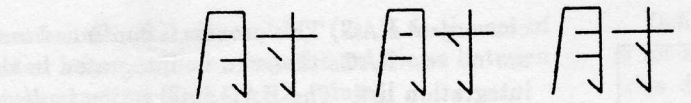
With the table network organization, RAG and HAG integrations, the model database can be updated quickly by learning from samples. The procedure consists of transformation and updating.

5.1 Transformation

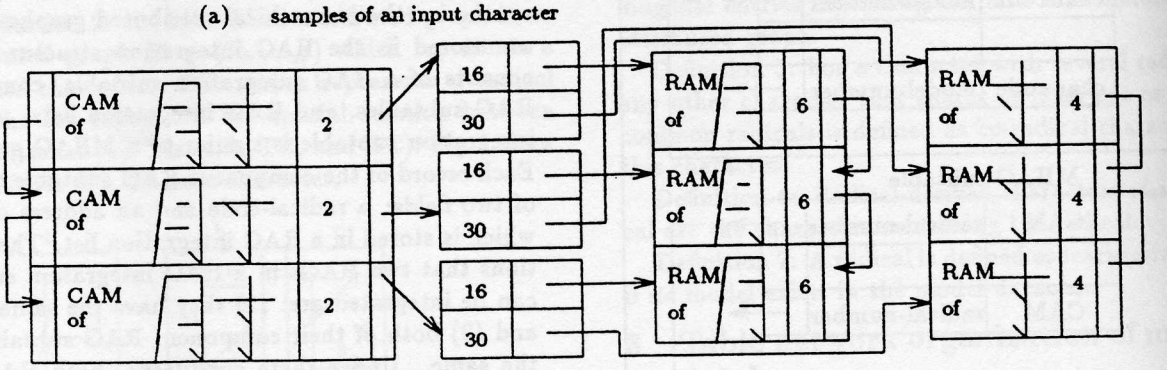
For each sample, there is a sample HAG (SHAG) and several sample RAGs (SRAGs) which correspond to the radicals in the sample. In the transformation phase, integration of RAG and HAG is performed to reduce the number of SHAGs and SRAGs. For each component radical of the input character, its RAGs of the samples, (SRAGs) are obtained and stored in its corresponding RAG integration list. Then the SHAGs are constructed and stored in its corresponding HAG integration structure. RAG integration is applied on each RAG integration list in the HAG integration structure such that no two SRAGs in the list can be further synthesized. At last, the HAG integration is applied on the HAG integration structure such that no two HAGs in the structure can be further synthesized. Figure 3 illustrates an example of transformation.

5.2 Updating

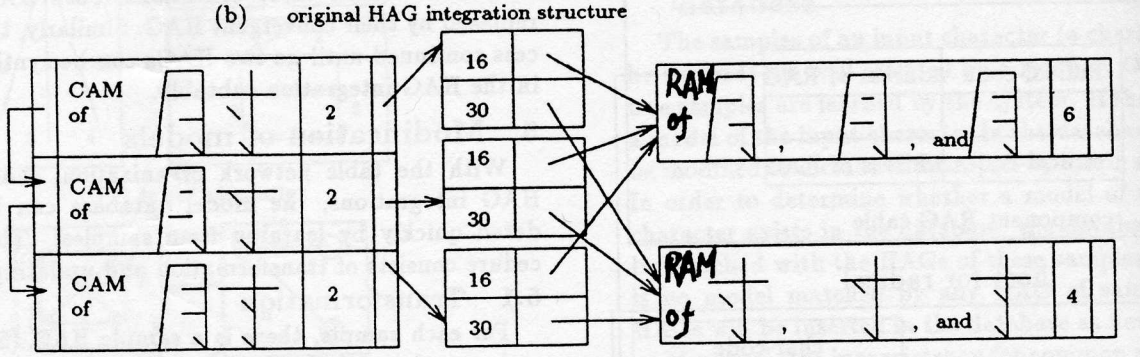
There are two cases of updating. 1) An input character has been learned before. Its models exist in the database. 2) An input character has never been learned. Therefore, no model for the character in the database. New models for the input character have to be inserted in the database. In both cases, for the learned radicals of the input character, the models of their radical-derived characters in the database should be modified according to the RAG integration lists of these radicals. Figure 4 shows the updating for the first case.



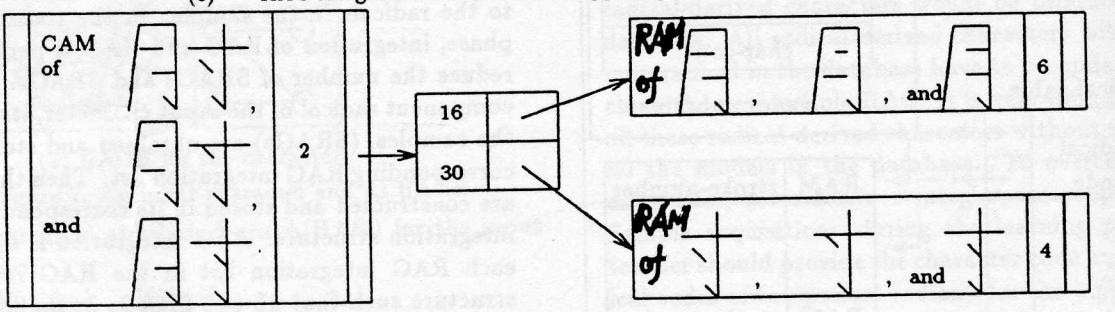
(a) samples of an input character



(b) original HAG integration structure



(c) HAG integration structure after applying RAG integration



(d) HAG integration structure after applying HAG integration

Figure 3. An example of transformation for an input character

Case 1. The input character has been learned before. All its component radicals are the learned radicals. The updating procedure for this case is:

1. Make a copy of the MHAG table of the found models from the host for characters, and a copy of each component RAG subtable corresponding to each model.

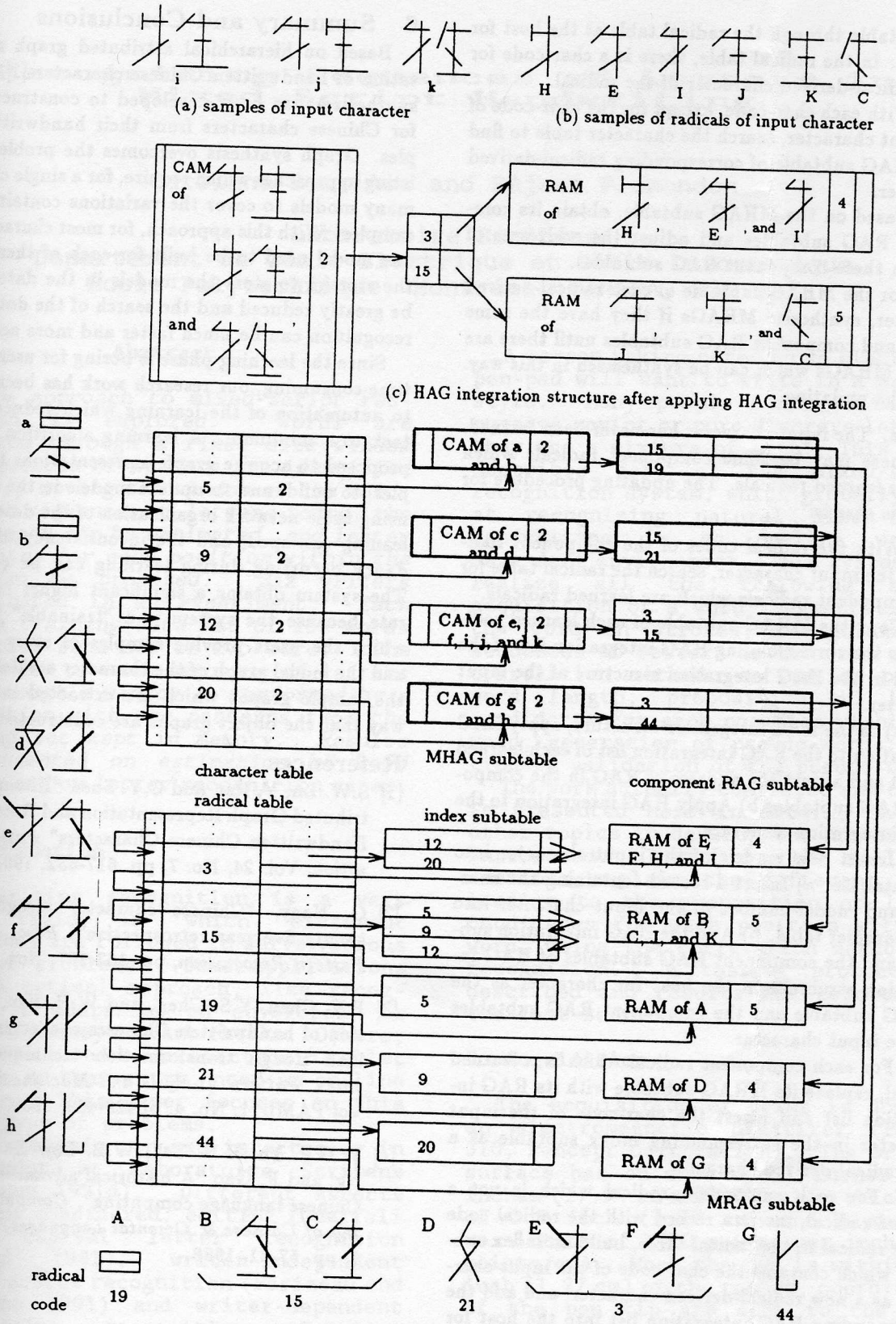
2. For each component radical, make a copy for its MRAG subtable in the host for radicals.

3. Update the HAG integration structure obtained in the transformation phase.

4. a) apply RAG integration to each RAG int gration list and adjust the address of RAG in ea component RAG subtable. b) Apply HAG integr tion to the HAG integration structure.

5. a) Replace the MHAG subtable and the comp nent RAG subtables of the input character at the h for characters with the HAG integration subtable a the component RAG subtables of the HAG integ tion structure. b) For each component radical, repl its MRAG subtable in the host for radicals with RAG integration list.

6. According to the radical code of each com nent radical which is a learned radical, obtain its



(d) Final updating results for the input character

Figure 4. An example for the updating the table network.

dex subtable through the radical table at the host for radicals. In the radical table, there is a char-code for each radical-derived character of the radical.

7. With each char-code, except for the char-code of the input character, search the character table to find the MHAG subtable of corresponding radical-derived character.

8. Based on the MHAG subtable, obtain its component RAG subtables and adjust the addresses of RAG in these component RAG subtables.

9. For the MHAG subtable of each radical-derived character, synthesize MHAGs if they have the same CAMs and component RAG subtables until there are no two MHAGs which can be synthesized in this way. Exist the updating.

Case 2. The input character has never been learned. There may be some component radicals which are learned radicals. The updating procedure for this case is:

1. With the radical codes of the component radicals of the input character, search the radical table for the component radicals which are learned radicals.

2. Copy the MRAG subtable of each learned radical into the corresponding RAG integration list of the radical in the HAG integration structure of the input character.

3. a) In the HAG integration structure, apply RAG integration to the RAG integration list of each learned radical and adjust the address of RAG in the component RAG subtable. b) Apply HAG integration to the HAG integration structure.

4. Insert new models of the input character into the database. a) Insert a record containing the char-code and model-number of the input character into the character table. b) Add the HAG integration subtable and the component RAG subtables of HAG integration structure in the host for character as the MHAG subtable and the component RAG subtables for the input character.

5. For each component radical which is a learned radical, replace its MRAG subtable with its RAG integration list and insert the char-code of the input character in the corresponding index subtable as a new radical-derived character.

6. For each component radical which is not a learned radical, insert a record with the radical code of the radical in the radical table, build an index subtable which contains the char-code of the input character as a new radical-derived character, and add the corresponding RAG integration list into the host for radicals as its MRAG subtable.

7 - 10. Same as steps 6-9, inclusive in case 1.

6 Summary and Conclusions

Based on hierarchical attributed graph representation of handwritten Chinese characters[1], a graph synthesizer has been developed to construct models for Chinese characters from their handwritten samples. Graph synthesis overcomes the problem of existing approaches which require, for a single character, many models to cover the variations contained in its samples. With this approach, for most character, only one model need to be built for each of them. Thus, the storage to store the models in the database can be greatly reduced and the search of the database for recognition can be much faster and more accurate.

Since the learning phase is boring for users and the time-consuming, our research work has been devoted to automation of the learning which reduces human task to a minimum. A learning algorithm has been proposed to acquire graph representations from samples, to build, and to update models in the database, using table network organization of the database, the leaning can sped, and the inconsistency and redundancy occurring during learning can be conquered. The system obtains a significant higher recognition rate because the system is a "trainable" system in which the users provide samples of every character and the model graph of the character are derived from the sample graphs which are extracted in the same way that the object graphs are constructed.

References

[1] S.W. Lu, Y. Ren and C.Y. Suen "Hierarchical Attributed Graph Representation and Recognition of Handwritten Chinese Characters" *Pattern Recognition*, Vol. 24, No. 7, pp. 617-632, 1991.

[2] G. Nagy, "Chinese character recognition: A twenty-five-year retrospective", *Proc. Intl. Conf. Pattern Recognition*, pp. 163-167, Nov. 1988.

[3] P.N. Chen, Y.S. Chen, and W.H. Hsu, "Recognition of handwritten Chineses characters by modified Hough transformation techniques", *IEEE trans. Pattern Analysis and Machine Intelligence* Vol PAMI-11, No. 4, pp429-439, 1989.

[4] W.C.P. Yu, H.H. Teh, W.B. Low X. Yan, T.M. Ng, and F. Gao "A historical advancement of the Chinese language computing", *Computer Processing of Chinese & Oriental Languages*, Vol. 4, No 1, pp. 57-81, 1988.