Results Clustering for Keyword Search over Relational Database

Shuxin Yang
School of Information Engineering,Jiangxi University of Science and Technology,Ganzhou 341000, China
Email: 670774377@qq.com
Lanying Shi
School of Information Engineering,Jiangxi University of Science and Technology,Ganzhou 341000, China
Email: shilanying1207@163.com

***Abstract*—Keyword Search over Relational Database (KSORD) has been a hot research topic in the field of the database. The existing prototype systems present the results to user in a linear list. The user has to browse individually. Therefore, it is still very difficult to find the information users really need. To solve this problem, this study is carried out on results clustering for Keyword Search over Relational Database. Learning from the concept of vector in physics, this study proposes a new model of result tree, which is called result-tree characteristic vector. This study also proposes a new clustering strategy based on result-tree characteristic vector. It firstly gets the result-tree characteristic information, and describes the joint tuple tree using vector representation, and then classifies the retrieval results according to the corresponding vector representation. The experimental results verify the feasibility and effectiveness of the clustering strategy in this study and manifest that the method in this study can efficiently help users navigate through and improve the users' browsing efficiency.**

***Index Terms*—relational database; joint tuple tree; result tree characteristic vector; clustering;**

I. Introduction

With the continuous development of database technology, more and more users need to access the database online, which makes KSORD become a hot research topic in field of database. KSORD can enable users to get information in the database conveniently by inputting keywords, without being familiar with the knowledge of underlying database mode and the structured query language, just like using Baidu, Google to query WEB. Because of its great user friendliness and convenience, KSORD is welcomed by the majority of Internet users. The existing prototype systems respectively put forward different modeling methods or different search algorithm to get the query results, and then rank the search results according to various scoring function, at last, present the results to users in a linear list. However, due to the huge amounts of information in relational database, a set of keyword query often gets tens of thousands of results containing the query keywords, so the linear list, produced by search engine, is generally long, and lacks a clear organizational structure. Users have to browse individually, therefore, it is still very difficult to find information they really want. In order to solve this problem, many scholars improved the result ranking algorithm. However, in most cases, the query posed by users does not clearly express their needs, especially when users are not familiar with the field they search. Therefore, only improving the rank algorithm is not enough. But, result clustering allows users browse search results conveniently and quickly. Result clustering is not only a necessary step for allowing users to quickly get the required information, but also an effective method to improve the retrieval performance. Aiming to solve this issue, this study carries out a research of result clustering on KSORD. On the basis of in-depth study and analysis of the clustering algorithms in the field of information retrieval, this study proposes a clustering strategy based on the result-tree characteristic vector.

Manuscript received September 16, 2013; revised October 5, 2013; accepted October 9, 2013. Copyright credit,
Project number: 20122BAB211035 and GJJ12349.
Corresponding author: Lanying Shi, Tel: +86 18103832616, Email address: shilanying1207@163.com

II. Related Work

The first research work related to keyword search over relational databases is shown in literature [1], in which the database was modeled as a graph. The tuples are represented as nodes in the graph and the relationships between tuples are represented as edges in the graph. Some ground-breaking research work were published in 2002, such as BANKS[2], DBXplorer[3] and DISCOVER[4]. Since then, at the top of academic international conference, including VLDB, SIGMOD and ICDE, many studies have emerged, such as IR-Style[5], SEEKER[6] and Spark[7] and so on. These systems support arbitrary keywords query, but the solutions to the problem are different, mainly involved in the aspects of data model, search algorithm and scoring function. In terms of the data model, there are mainly two types, data

doi:10.4304/jsw.8.12.3188-3193

graph and schema graph. Some research such as BANKS, BLINKS[8] and [9,12] are based on data graph, while some others, including DBXplorer and DISCOVER, are based on schema graph. In the method based on data graph, search algorithm is respectively different, such as BANKS adopts the backward search algorithm; BANKS-II poses bidirectional search algorithm on the basis of improving the backward search algorithm, so as to improve search efficiency. BLINKS presents backward search strategy based on price equilibrium extend, so as to shorten the processing time. In addition, concerning the score function, the existing researches mainly focus on structure compactness, correlation with the content, and the combination of the content and the structure. Such as DBXplorer and DISCOVER are based on the structure compactness, SEEKER is based on the correlation with content, and literature [13] is based on the combination of the content and structure.

However, the existing studies focus mainly on the data model, search algorithm and results ranking algorithm. The research aiming to solve the results clustering problem on KSORD is fairly rare. The studies [14,15] involved in results clustering on KSORD. Unfortunately, they have some limits, such as in [15], the method of clustering is based on the pattern of result tree. It firstly enumerates all the possible patterns of the results and encodes all the traversal trees of each pattern result, then selects the minimum be the corresponding code of each pattern result. This process is rather time-consuming. Besides, it classifies the results according to the pattern class, which would lead to a fact that some pattern classes do not include any result. In the meantime, there are a large number of outstanding research achievements concerning result clustering over text document and XML, but due to the structural characteristic of relational database, traditional text document and XML search results clustering method can not be applied to relational database query results. Therefore, the result clustering problem on keyword search over relational database still has a large room for research. The application of result clustering strategy in keyword search over relational databases system is rather important, which can classify the results into different clusters, and present the results to user in hierarchical structure. The user can directly look over the information of the category they are interested in, which not only can greatly improve the user query efficiency, but also can enhance the interaction between search engines and user.

III. Materials and Methods

A. Definition

This study models relational database as an undirected graph. Related definition is following.

Definition 1(Data graph) A relational database is modeled as an undirected data graph $G(V,E)$. V is a node set. E is an edge set. For each node in V, there is a one-to-one mapping tuple in the database. For each edge in E, there is a one-to-one mapping primary key/foreign key relationship between two tuples in the database.

Definition 2 (Result tree) For each query keywords $Q(k_1,k_2,...,k_n)$, where k_i $(1 \leqslant i \leqslant n)$ is the keyword inputted by user. The result is a set of minimum joint tuple tree (result tree for short), which contains all the keywords inputted by user, without redundant node and redundant edge. In other words, a result tree must be "minimum", which has no subtree that is also contains all of query keywords.

B. Result Tree Characteristic Vector

This study proposes a new idea of modeling joint tuple tree. It takes advantage of the structural characteristic of relational database. In the data graph, node type corresponds to the database table, and edge type corresponds to the primary key/foreign key relationship. The node characteristics include node type and the number of each type. The edge characteristics include edge type and the number of each type. Abstract node characteristics and edge characteristics, and organize those characteristics in vector. In the following text, it is collectively referred to as the result tree characteristic vector.

In physics, a vector is a variable which has both magnitude and direction. From the definition, it can be learnt that the vector has two attributes. One attribute describes size, and the other attribute describes direction. Similarly, the joint tuple tree also contains two type elements, namely, node and edge. Node corresponds to tuple in relational database. Node contains content information and belongs to content attribute. Edge corresponds to the primary key/foreign key relationship in relational database. Edge conveys structure information, belongs to structure attribute.

Definition 3(Result tree characteristic vector) Given a data graph G, if it contains m node types and n edge types, then all the joint tuple trees retrieved from the system based on the method of data graph can be presented in vector $[CA_1, CA_2, \ldots, CA_m, SA_1, SA_2, \ldots, SA_n]$, CA_i $(i=1,2,\ldots,m)$ represent content attribute, SA_j $(j=1,2,\ldots,n)$ represent structure attribute, m is the number of note types, and it determines the dimension of the content attribute. n is the number of edge types, and it determines the dimension of the structure attribute. Variable m and n together determine the dimension of the vector. The value of CA_i $(i=1,2,\ldots,m)$ represent the number of nodes of each type. The value of SA_j $(j=1,2,\ldots,m)$ represent the number of edges of each type.

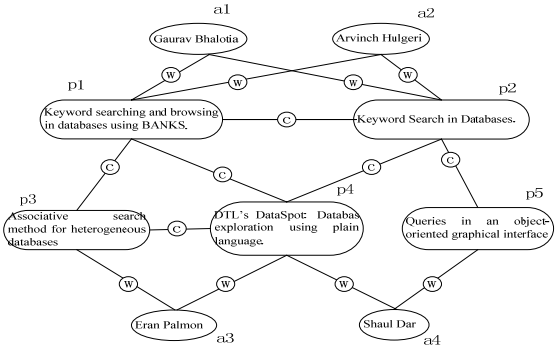

Figure 1: Sub-graph of database graph

Take Fig.1 for example, it is a sub-graph of database graph converted from DBLP. DBLP data set is saved in XML. It is a data set which describes the citation relationships among papers. Four tables can be got after converting DBLP into relational database by using the XML parser. They are “author, paper, write (author-paper), cite (paper-paper)”, and then model it as a data graph.

In Fig.1, there are two kinds of nodes, corresponding to two type tuples respectively coming from the author table and the paper table. There are two kinds of edges. The edges connecting author and paper convey the "writing" relationship, the edges connecting paper and paper convey the "cite" relationship.

According to the definition of result tree characteristic vector, all the search results based on data graph can be expressed as [*A*, *P*, *W*, *C*]. *A* and *P* respectively represent two different types of nodes, and the value of *A* and *P* respectively represent the number of each type notes; *W* and *C* respectively represent two different types of edges. The value of *W* represents the number of primary key/foreign key connecting author and paper. The value of *C* represents the number of primary key/foreign key connecting paper and paper. Namely: *A* and *P* belong to content attribute, describing the number of nodes in the graph, *W* and *C* belong to structure attribute, describing the number of edges in the graph.

C. Result Clustering Method

Take DBLP as an example, assume that the user inputs a set of query keywords $Q(k_1, k_2, ..., k_n)$, the joint tuple trees returned by system is shown in Fig.2, the specific node label of *A*, *P* in the graph is different, the result of each tree represents a type of results.

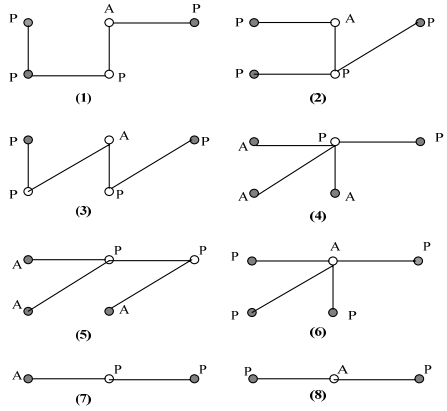

Figure 2: Result Tree

Now, describe the joint tuple trees using vector representation. What can be got is as following:

Result Tree1: [*A*,*P*,*W*,*C*]=[1,4,2,2]
Result Tree2: [*A*,*P*,*W*,*C*]=[1,4,2,2]
Result Tree3: [*A*,*P*,*W*,*C*]=[1,4,2,2]
Result Tree4: [*A*,*P*,*W*,*C*]=[3,2,3,1]
Result Tree5: [*A*,*P*,*W*,*C*]=[3,2,3,1]
Result Tree6: [*A*,*P*,*W*,*C*]=[1,4,4,0]
Result Tree7: [*A*,*P*,*W*,*C*]=[1,2,1,1]
Result Tree8: [*A*,*P*,*W*,*C*]=[1,2,2,0]

Intuitively, classify the results in different clusters according to whether they have the same vector representation. The following is to analyze whether it is reasonable.

Analysis: Result tree1,2,3 have the same vector representation of [*A*, *P*, *W*, *C*]=[1,4,2,2]. All of those three result trees contain one author information and four paper information, describing the “write” relationship between the author and the papers and the “cite” relationship among papers. The author participated in two papers’ writing, and there are 2 cite relationship among the four papers. Result tree 4, 5 have the same vector representation of [*A*, *P*, *W*, *C*]=[3,2,3,1], both of them contain three author information and two paper information, describing the “write” relationship between 3 authors and 2 papers and the “cite” relationship between two papers. Three authors participate in paper writing, and there is one “cite” relationship between the two papers. In addition, it can be noticed that result tree 4,6 have a great similarity in structure, but they have different number of each note type, which means the amount of information they contain is different, and can not be classified in one cluster. Besides, result tree 2,6 contain the equal amount of information, but have different number of each edge type, which means the content they transfer is different, and can not be classified in one cluster. The relationship between result tree 7,8 is similar to that of result tree 2,6, and they can not be classified in one cluster either. The result tree characteristic vector can well reflect those slight differences.

Conclusion: from the above analysis, the results with the same vector representation have the same node type and the same number of each type node. While the results’ topology is connected in different ways, but they contain the same edge type and the same number of each type edge. Namely, the amount of information they contain is approximately equal, and the content they transfer is similar. Therefore, to classify them into one cluster is reasonable. The pseudo-code of results clustering is shown in table 1. NOTES: *Map* is a storage space for storing the value of *A,P,W,C* of each result tree *rt*. *Cluster*[] denotes cluster array. Function *Label* is used to calculate the value of *A,P,W,C* of a result tree. Variable *Id* is the No. of each cluster.

TABLE 1:
THE PSEUDO-CODE OF RESULTS CLUSTERING

```
Input: the top-k result trees rts_list
Output: Results in different clusters
1   Map ← empty;
2   For each result tree rt in result-trees rts_list {
3       [A,P,W,C]←Label (rt);
4       If find [A,P,W,C] in Map
5           {  Id ← get Id of [A,P,W,C] in Map;
6              Add result tree rt to Cluster[Id];   }
7       Else
8           {  Add [A,P,W,C] in Map;
9              Id ← get Id of [A,P,W,C] in Map;
10             Add result tree rt to Cluster[Id];   } }
```

IV. EXPERIMENT

A. Experiment Setup

In this study, the experiments were conducted on an Intel (R) Core (TM) 2 i3-330M computer with 2GB of RAM running Microsoft Windows 7, and the algorithm were implemented in C++ in the development environment of Microsoft Visual C++6.0. Take DBLP as the test data and MYSQL as the database.

B. Experimental Results

1). Results Output and Rank

The search algorithm in this study is inspired by [12], taking advantage of building index to support search on large data graph. Index table records the shortest path between any two nodes, which can be connected within P_{max} edges. Note that when the length of the path between two nodes is larger than the threshold the maximum path length P_{max}, the answer will be less meaningful. In order to reduce index space and produce more accurate and compact result, this experiment let the threshold of the path length P_{max} be 3. When user inputs a set of query keywords, the system retrieves all the paths containing the given keywords, and if there are same node in different paths, merge them, then produce a set of minimum joint tuple tree containing all the keywords.

Score the result trees according to the score function in [13], and number them in serial number from 1 to k (k equals the k of top-k). The higher the score is, the smaller the No. is. Before clustering, these result trees are ranked in ascending order of the No. The rank of these result trees after clustering is following: the sequence of different clusters is determined by the smallest No. of the result tree in each cluster. In the same cluster, the result trees are ranked according to the arrangement of the size of the No.

2). User Interface

When a user inputs a set of query keywords $Q(k_1, k_2, ..., k_n)$ = (keyword, search, relational), and wants to see the answers of Top - k (k = 200), the result of the implementation is as shown in Fig.3-5.

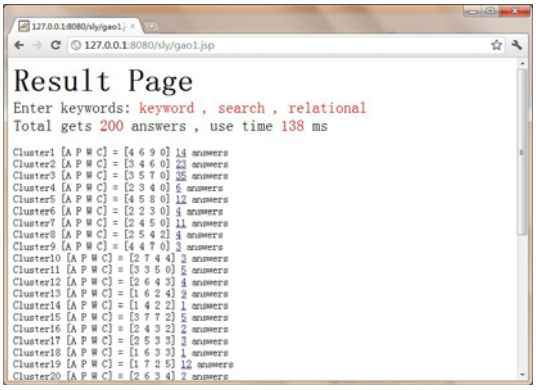

Figure 3: The result after clustering

Figure 4: The answer list of a specific cluster

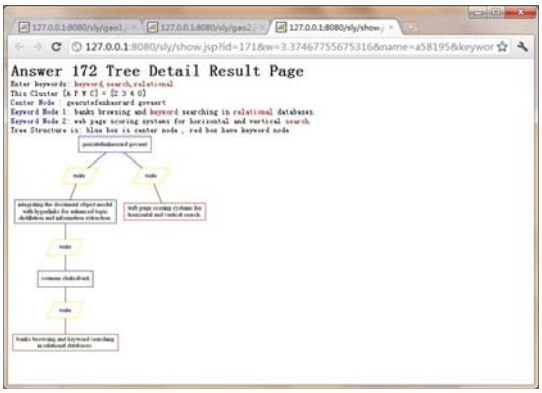

Figure 5: The structure information of an answer

Fig.3 shows the user interface after result clustering, which provides heuristic information and enables user to grasp the overall distribution of the answers and the class information of each cluster in short time. It supports user to do further retrieval for interesting category, to help user find the answer as soon as possible. Fig.4 is the result list of one specific cluster. Each answer contains the information of keywords nodes. As is shown in Fig.4, the No. of the result trees in one cluster is discontinuous, which can be explained by the rank and cluster strategy above. The benefit is that it can not only provide the sequence of the result trees in each cluster, but also provide the result trees' sequence in the whole result trees, which is not available in [14,15]. Fig.5 is the structure information of an answer, which is very straightforward and easy to understand.

3). Experiment Comparison

In this part, the method in [15] was implemented and compared with the method in this study. Six group experiments were carried out on both of the two systems, and top-k answers were output. Let k be 20, 60, 100, 140, 180, 220 successively each time. Take the average values of 6 experiments as standard. Compare the time consumed and the number of clusters after clustering. The results are shown in Fig.6 and Fig.7. NOTES: S-_CBR is the method in [15], and RTCV (result tree characteristic vector) denotes the method in this study.

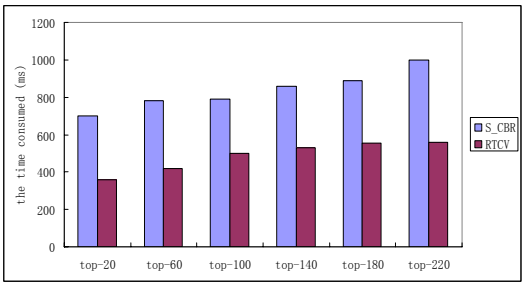

Figure 6: The time consumed

The parameter "top-*k*" is taken on the horizontal axis and the parameter "time consumed" is taken on the longitudinal axis in Fig.6. The experiment results shown in Fig.6 manifests the method in [15] is more time-consuming, since it needs to encode all the possible traversal trees of each result tree in the process of clustering. The method in this study just needs to calculate the characteristic vector of each result tree, and the characteristic vector of each result tree is unique, which is convenient and time-saving.

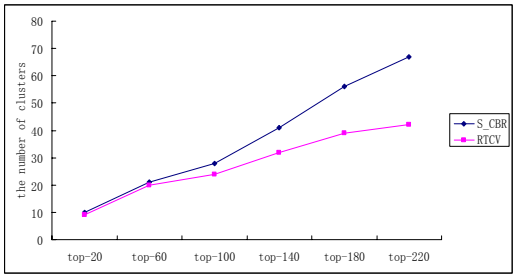

Figure 7: The effect of clustering

The parameter "top-*k*" is taken on the horizontal axis and the parameter "number of clusters" is taken on the longitudinal axis in Fig.7. The experiment results shown in Fig.7 manifests that the larger the value of *k* is, the more obvious the effect of clustering is. Namely, given a certain number of answers, the fewer the number of clusters is, the better the clustering effect is, and vice versa. Use numbers to specify, given 100 answers, if it get 70 clusters after clustering, the effect is not desirable. But, if it get 30 clusters after clustering, the effect is acceptable. On this basis, the effects of two methods are both acceptable, but when the value of *k* increases, the cluster effect of the method in this study is more obvious. The reason is that in [15], it requires that results in one cluster must have the exactly schema, which is hard to meet. In this method, it starts from the structure characteristics of the result tree, focuses on the type of nodes and edges of the result tree, which reflects connection relation between tuples, ignoring the specific topological structure of the result tree to achieve a better clustering effect.

Experimental result not only shows the feasibility and rationality of the clustering strategy presented by this study, but also verifies the rationality of the concept of result-tree characteristic vector, since it integrates the node information (content information) and edge information (structure information) of the result tree. It ensures that the result trees in one cluster satisfying that: (1) They contain the same node type, and each type has the same number of nodes. (2) They contain the same edge type, and each type has the same number of edges. (3) They contain the equal amount and similar content of the information. This result clustering method for KSORD, to a certain extent, improves the results show, helps users navigate and enhances browsing efficiency, and enables users to quickly grasp the result information and distribution of the retrieval results as a whole.

V. Conclusion

This study discussed the deficiency and importance of study on results cluster for keyword search over relational database, and it proposed the concept that the result tree characteristic vector. It is a novel method of modeling the joint tuple tree, which is used for the measure of similarity. It plays a key role in producing higher quality of clustering, and be used in all the systems of KSORD. It takes advantage of the structure characteristic of DBLP, and can be applied flexibly in other relational databases, such as IMDB. It provides a new method for solving the problem of result clustering on keyword search over relational database.

Acknowledgment

This research is supported by the National Natural Science Fund (No.41362015), Key Project of Education Department of Jiangxi Province (No.12770), Science and Technology Project of Education Department of Jiangxi Province (No.GJJ12349, GJJ13411, GJJ13415), and Youth Science Foundation Project of Science and Technology Department of Jiangxi (No.20122BAB211035).

References

[1] Goldman R, Shivakumar N, Venkatasubramanian S, Garcia-Molina H. Proximity search in databases. In: Gupta A, Shmueli O, Widom J, eds. Proc. of the 24th Int'l Conf. on Very Large Data Bases (VLDB'98). New York: Morgan Kaufmann Publishers, pp. 26-37, 1998.

[2] G.Bhalotia, A.Hulgeri, C.Nakhe, and S.Chakrabarti. Keyword searching and browsing in databases using banks.In ICDE, pp. 431-440, 2002. doi: 10.1109/ICDE.2002.994756

[3] S. Agrawal, S. Chaudhuri, and G. Das. Dbxplorer: A system for keyword-based search over relational databases. In ICDE, pp. 5-16, 2002. doi: 10.1109/ICDE.2002.994693

[4] V.Hristidis and Y.Papakonstantinou. DISCOVER: Keyword search in relational databases. In Proc. of VLDB'02, pp. 670-681, 2002.

[5] V. Hristidis, L. Gravano, and Y. Papakonstantinou. Eficient IR-style keyword search over relational databases. In Proc.of VLDB'03, pp. 850-861, 2003.

[6] Wen JJ, Wang S. SEEKER: Keyword-Based information retrieval over relational databases. Journal of Software, pp. 1270-1281, 2005.

[7] Y. Luo, X. Lin, W. Wang, and X. Zhou. SPARK: top-k keyword query in relational databases. In SIGMOD, pp. 115-126, 2007. doi: 10.1145/1247480.1247495

[8] H.He, H.Wang, J.Yang,and P.S.Yu. Blinks: ranked keyword searches on graphs. In Proc. Of SIGMOD '07, pp. 305-316, 2007. doi: 10.1145/1247480.1247516

[9] Yubin Guo, Liankuan Zhang, Fengren Lin,and Ximing Li.A Solution for Privacy-Preserving Data Manipulation and Query on NoSQL Database. Journal of Computers,Vol.8, pp.1427-1432, 2013. doi:10.4304/jcp.8.6.1427-1432
[10] Xiaoming Wang, Duobao Yuan.A query verification scheme for dynamic outsourced databases.Journal of Computers, Vol.7, pp.156-160, 2012. doi:10.4304/jcp.7.1.156-160
[11] Dengyin Zhang, Xuefeng Lin, Hui Zhang.An Improved Cluster-Based Cooperative Spectrum Sensing Algorithm. Journal of Computers, Vol.8, pp.2678-2681, 2013. doi:10.4304/jcp.8.10.2678-2681
[12] G. Li, J. Feng, X. Zhou, and J. Wang. Providing Built-in Keyword Search Capabilities in RDBMS. VLDB, pp.1-19, 2011. doi: 10.1007/s00778-010-0188-4
[13] Yang Shuxin and Shi Lanying. A Ranking Strategy Based on the Information of Content and Structure on KSORD. In Journal of intelligence, pp.127-131, 2013.
[14] Wang S, Peng ZH, Zhang J, Qin L, Wang S, Yu JX, Ding BL. NUITS: A novel user interface for efficient keyword search over databases. In: Dayal U, Whang KY, Lomet DB, Alonso G, Lohman GM, Kersten ML, Cha SK, Kim YK, eds. Proc. of the 30th Int'l Conf. on Very Large Data Bases (VLDB 2006). Seoul: Morgan Kaufmann Publishers, pp.1143-1146, 2006.
[15] Peng ZH, Zhang J, Wang S. S-CBR: Presenting results of keyword search over databases based on database schema. Journal of Software, pp.323-337, 2008.

Shuxin Yang(1978-). He was born in Jiujiang City of Jiangxi Province. He received PhD in the area of research on workflow technology from TongJi University in 2009, Shanghai, China.

He works in Jiangxi University of Science and Technology, as an Associate Professor. The location is No. 156 Hongqi Road, Zhanggong District, Ganzhou, Jiangxi, China. His several papers were published in Core journals and international conferences, and some of them were indexed in EI Compendex. Such as, Monitor permission control for process in workflow system was published in Computor Integrated Manufacturing Systems, Vol.13 No.11, Nov.2007. Current and previous research interests include data management and information retrieval.

Dr. Yang is a member of China Computer Federation, and director of Computer Department, and a reviewer of the research and application of computer.

Lanying Shi (1986-). She was born in Kaifeng City of Henan Province. She graduated with Bachelor's of Electrical engineering and its automation from Changchun Institute of Technology, Changchun, China, 2009. She is a Master Degree Candidate. Her current research interests include keyword search on relational database and information security.

Notice of Retraction

The following article has been retracted by the Editorial Board and the Publisher of Journal of Software:

"Determining the Physiological Age of Plant Organs on Basis of Clustering Analysis"
by Dayan Shangguan and Xinyuan Huang
Journal of Software
Volume 8, Number 12, November 2013, Pages: 3194-3199
doi:10.4304/jsw.8.12.3194-3199

This paper has been retracted at the request of the Authors. The authors, Dayan Shangguan and Xinyuan Huang, decided to retract their paper on the basis of the following considerations:

"According to the latest research results, there is a certain difference between the conclusion and experiment data analysis and some minor errors are found in the above paper, so further study needs to be done for modification."