Proceedings of the Ninth International Conference on Machine Learning and Cybernetics, Qingdao, 11-14 July 2010

WEB LOG DATA CLUSTERING FOR A MULTI-AGENT RECOMMENDATION SYSTEM

JOÃO C. XAVIER-JUNIOR [1], ALEX A. FREITAS [2], ANNE M. P. CANUTO [3], LUIZ M. G. GONÇALVES [1]

[1] Computing and Automation Engineering Department, Federal University of Rio Grande do Norte, Natal, Brazil
[2] School of Computing, University of Kent, Canterbury, Kent, UK.
[3] Informatics and Applied Mathematics Department, Federal University of Rio Grande do Norte, Natal, Brazil
E-MAIL: jcxavier01@gmail.com, A.A.Freitas@kent.ac.uk, anne@dimap.ufrn.br, lmarcos@dca.ufrn.br

Abstract:

In this paper, we propose an automatic way of recommending information to be visualized by users. The list of information to be recommended is generated based on the web logs of the users stored by the system in a multi relational database. This system is a web-based multi-agent system which provides geographical information and monitors the actions of the users by generating logs. Frequently, the system joins the relational web log data and runs a clustering algorithm in order to recommend a list of most accessed information up to that moment to registered users who log in the system.

Keywords:

Relational clustering; Web log; Multi-agent system.

1. Introduction

As web-based Geographic Information Systems (GIS) can provide a large amount of data, sometimes new registered users need a friendly guide that is able to recommend a list of the most accessed geographical information. In this case, the access and use of the system can become easier. Nowadays, many web-based systems provide a very useful guide for their users.

When using web-based systems, it is important to store web log data and this storage in relational databases is a normal procedure. However, the processing of this information for recommendation purposes is not commonly found in the literature. Despite the increase in volume of data stored in relational databases, very little effort have been done to propose techniques to work with multiple relations data [1]. In a dataset stored in a relational database with one-to-many associations between records, each table record can form many associations with others records from other tables, making the processing of these data a difficult task.

There has been a growing interest in clustering multi relational data in the literature [8, 9]. Clustering is an unsupervised learning method that partitions a set of patterns into groups (clusters) [11]. As a consequence, some clustering methods have been proposed, most of which can produce good results by using attributes and relationships simultaneously. However, the trade-off for the above advantage is that clustering algorithms need to explore far more information, so their scalability and efficiency are reduced.

In this paper we propose an automatic way of recommending information to be accessed by users. A list of information recommendations is created based on the web logs of the users stored in a multi relational database. In order to achieve good results in system usability (identification of relevant information to be visualized), the relational tables that store the logs of the users were used for clustering purposes. On top of that, three scenarios were chosen to evaluate the effectiveness of the clustering algorithm. In all three scenarios, we use an agglomerative hierarchical clustering algorithm, since it is a well-known type of clustering algorithm. Nevertheless, any clustering algorithm could be used for the aforementioned scenarios.

This paper is divided into 7 sections and structured as follows. Section 2 analyzes works related to this paper. Section 3 presents the system architecture designed for retrieving geographic information and for monitoring the actions of the users. Section 4 formulates the relational clustering problem. Section 5 describes the clustering algorithm, the similarity measure and the cluster validity indices used for validating the results. Section 6 presents three different scenarios and the experimental results. Finally, Section 7 reports the conclusions of the authors.

2. Related Works

The use of intelligent agents in the development of web-based GIS has brought several benefits to these systems, such as improvement in services related to spatial data. Services of search, retrieval and online analysis of geographical data have become more easily available to users. In this context, some works have been proposed in the literature where different types of agents are employed to improve usability of GIS software, as well as the access to geographical data and services [4]. In addition to that, other works have proposed the use of intelligent agents to monitor end-users and maintain their profiles within the system [3, 12]. For instance, this GIS uses intelligent agents to monitor and store the actions of the users within the system. It also uses agents to mine web log data (actions of the users).

Web mining involves a wide range of applications which aim to discover and extract knowledge from data. Web mining also provides an important mechanism to make data access more efficient and adequate to users [10]. In web usage (log files) mining, for instance, the main task is to discover the activities (preferences) of the users while they are browsing and navigating through the web. The goal of understanding the preferences of the users is to enhance the quality of the service and to personalize this service [7]. In this area of data mining, several techniques can be applied such as classification and clustering.

The main aim of clustering algorithms, for instance, is to use attribute information to group instances (records) that have similar attribute values. However, when working with relational data there are more types of information available that need to be used to distinguish groupings [12]. Clustering in multi-relational data has been studied in some works [2, 12, 13]. When computing similarity between two instances, they have to consider not only the attributes vales, but also the inter relationship between the instances. As similarity measure plays an important role on clustering different objects (instances), some measures have been proposed by researches [5]. In addition, other works have been produced to show the results of clustering only attribute information and both attribute and inter relationships information [2, 12].

Unlike the aforementioned works, this work relates intelligent agents and multi-relational clustering in the web-based GIS context. In order to do this, it presents a way of offering recommended information for registered users of a web-based GIS system. Within the system, intelligent agents monitor the action of the users (accessed information) and store them in a relational database. They are also responsible for join the logs stored in different tables, and also for clustering the data. As a result of this procedure, a list of most accessed information is generated to be recommended to users who log in the system. This list can be a very useful kind of guide for users who do not know what geographical information can be visualized in the system.

3. The System Architecture

In order to combine the concepts of multi-agent systems applied for monitoring, filtering and retrieval of geographical information in a system, a general architecture has been proposed for the system, which is illustrated in Figure 1. This architecture is composed by three types of agents (Interface, Monitoring and Clustering).

During the interaction with the system, the actions of the users are monitored and stored. By storing the log data of the users, the system is able to cluster them and generate an information recommendation list of the most accessed geographical information by the users.

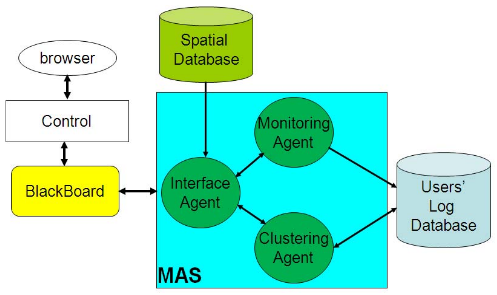

Figure 1. The system architecture.

After registering the users have access to the geographical information. They can browse the main page and select the objects (diving spots, types of fauna and flora, types of sediments and water features) available in map format. When browsing it, they can see a list of information suggestions which is provided as a guide. After selecting one of the objects, the control model writes the information selected by the user on the Blackboard (message exchange channel) which is read by the Interface agent.

The Interface agent accesses the Spatial Database and plots the map according to the user's selection. If the user selects any particular point of the map and visualizes its attribute information, the Interface agent passes the entire log to the Monitoring agent that is responsible for storing it in the tables of the database.

Periodically, the Clustering agent runs the clustering procedure that generates the list of most accessed geographical information. After that, the list is passed to the Interface agent to be shown to the users.

3.1. Dataset

The system stores geographical information and only registered users can access it. After the user selects the information to be visualized, the system plots a map of the selected information. Then the user can access the information contained in the map by choosing a single point or a group of points in the map and clicking on the view button located above the map. By doing that the user confirms to the system that the information was visualized and the system can now record it under his/her profile (type of user, type of information, point or points, and the attributes belonging to each visualized information.

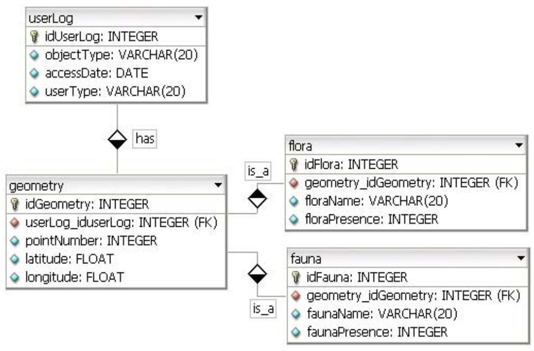

Figure 2. Subset of tables.

Figure 2 shows a subset of tables that store the logs of the users. The information stored generates one or many records in these tables. If a user has visualized fauna information, for instance, the system stores the user information (type of user), the date of access, the point or points selected, and also the attributes belonging to each type of information (e.g. *faunaName*).

By storing the logs of the users, the system will be able to join the tables and cluster the data. By doing that it will be able to understand how the geographical information (type of information, points and coordinates) has been accessed by the users. It will probably indicate, for instance, that the biologists are only interested in fauna information. However, it can also show that some biologists are also interested in sediments information.

4. Problem Formulation

In the multi-type relational data clustering problem, there are some collections of multi-type objects and their relationships. We represent them as a set of data objects $X=\{X_1,\ldots, X_m\}$ and a set of relationships $R=\{R_{intra}, R_{inter}\}$. X includes m subsets, which are $X_1=\{x_{11}, x_{12}, \ldots, x_{1n1}\}$, .., $X_m=\{x_{m1}, x_{m2},\ldots, x_{mnm}\}$, that refer to m different types of objects, and each type of objects X_i has its own attributes $X_i.A$. R includes two parts: R_{intra} and R_{inter}. $R_{intra}=\{R_1,\ldots,R_m\}$ refers to intra-relationships in each object type, and R_{inter} refers to interrelationships between different object types.

Some relational data do not have intra-relationships, when this is the case, only inter relationships are taken into account. This is the case of the data used in this work shown in Figure 2. The inter relationships in this relational structure are *one-to-many* and there is no intra relationship within the tables. Given X and R, one can cluster X into k clusters and this process is called multi-type relational data clustering.

Using the subset tables shown in Figure 2 as an example, there are four different types of objects (tables). In this case, X can be defined as X = *{userLog, geometry, fauna, flora}*. Each table has its own attributes, for example, userLog ("userType", "objectType", "accessDate"), geometry ("pointNumber", "latitude", "longitude"), fauna ("faunaName", "faunaPresence"), flora ("floraName", "floraPresence").

Relationships between different types of objects, such as "*has*" and "*is_a*" are inter-relationships and can be obtained from the data. In our case, there is no intra-relationship in the same type of object.

5. Clustering Algorithm

In this paper, we use the well-known agglomerative hierarchical clustering algorithm [11]. It generates clusters by iteratively merging similar objects into larger and larger clusters. Initially it uses each object as a cluster, and computes the similarity between each pair of objects. Then it keeps merging the most similar pairs of clusters, until no clusters are similar enough or a certain number of clusters remain. Furthermore, the similarity measure was implemented as a module of the agglomerative hierarchical clustering algorithm.

5.1. Similarity Measures

As the relational tables (see Figure 2) used to store the log data of the users have both numeric and categorical attributes, the chosen similarity measure needs to cope with both types of attributes. In this paper, we use a similarity measure similar to one proposed in [8, 9]. Our similarity measure is defined as follows.

Suppose x_{ij} and x_{ik} are two objects in X_i, the similarity between x_{ij} and x_{ik} takes both numerical and categorical attributes into account. Sim_A is the sum of differences of standardized numeric and categorical attributes, which is defined as:

$$Sim_A\left(x_{ij},x_{ik}\right)=\sum_{r=1}^{p}\left|x_{ij}^{r}-x_{ik}^{r}\right|+\sum_{r=p+1}^{N}\delta\left(x_{ij}^{r},x_{ik}^{r}\right) \quad (1)$$

where x_{ij}^{r},x_{ik}^{r} is the rth attribute of x_{ij}^{r},x_{ik}^{r} , N is the number of attributes, p is number of numeric attributes and

(*N-p*) is the number of categorical attributes. The difference function possible results for δ(*a*,*b*) are 0 or 1. If *a* and *b* are equals 0, otherwise, 1.

The numeric attribute values were normalized by using the min-max normalization. Note that as the tables (see Figure 2) have no intra relationships, equation (1) is just part of formula used in [8, 9], since the latter works use intra-relationships.

5.2. Cluster Validity Measures

In order to measure the results of a clustering algorithm, some validity indices have been proposed in the field of data mining [11]. These indices are used to measure the "goodness" of a clustering result comparing it to other results obtained by other clustering algorithms, or the same algorithm but varying different parameters.

In this paper, we choose an internal validity index for measuring the clustering results, since our dataset has no class information. The Davies-Bouldin (DB) Index [6] was choose in this case. The DB index is based on similarity measure of a cluster (S_{ij}) whose bases are the dispersion measure of a cluster (b_i) and the cluster dissimilarity measure (d_{ij}). The similarity measure of a pair of clusters (Sij) can be defined as follows:

$$S_{ij} = \frac{b_i + b_j}{d_{ij}}, \quad d_{ij} = d(v_i, v_j),$$

$$b_i = \frac{1}{|C_i|} \sum_{x \in c_i} d(x, v_i) \qquad (2)$$

Then the DB index can be defined as:

$$DB = \frac{1}{n_c} \sum_{i=1}^{n_c} S_i \text{ , where}$$

$$S_i = \max_{j=1..nc, i \neq j} (S_{ij}), \; i = 1..n_c \qquad (3)$$

The Davies-Bouldin index measures the average similarity between each cluster and its most similar one. As the clusters need to be compact and separated, the smaller the DB index value, the better the clustering partition.

6. Experimental Results

In this section, we describe in detail the different scenarios used to evaluate the clustering partitioning using the similarity measure defined in equation (1).

As the majority of the log data of the users is stored in two tables (fauna and flora), we decided to work only with those two types of objects (tables), plus we joined tables *userLog* with *geometry* and called it as object type 1, and called tables fauna and flora as object types 2 and 3, respectively. The number of records for objects type 1, 2 and are 177, 75 and 38, respectively.

Scenario 1: We extracted data from the database via SQL command (Select + left join) and saved it in a worksheet format file. This file contains all attributes (specified in the SQL command) which belong to object types 1, 2 and 3. As a result of using SQL command to extract data from a database, some instances may have null values. When this was the case, the null value was changed by "*N/A*". This procedure assures the absence of missing values in our data.

The resulting table (result of SQL command) has the following attributes: *Scenario1 = {userType, objectType, pointNumber, floraName, floraPresence, faunaName, faunaPresence}*. The *faunaPresence* and *floraPresence* attributes were treated as numeric and take the value 1 if there is fauna or flora presence at a particular map point or 0, otherwise. The *faunaName* and *floraName* attributes are categorical and store the names of fauna and flora found in a particular region.

The *pointNumber* attribute is numeric and stores the position of each point in the region map. The *userType* and *objectType* attributes are categorical and store the following values: "Biologist, Geologist, Oceanographer, Environmental Manager and Tourist" and "fauna, flora, diving spots, type of sediment and water features", respectively.

Two parameters of the agglomerative hierarchical clustering were considered in our experiments, namely: type of link used to measure the distance between two clusters and the number of *k* (clusters) output by the algorithm. The number of *k* varied from 2 to 20, and the links used were *singlelink*, *completelink* and *averagelink* (closest, largest and average distance between two items of two clusters).

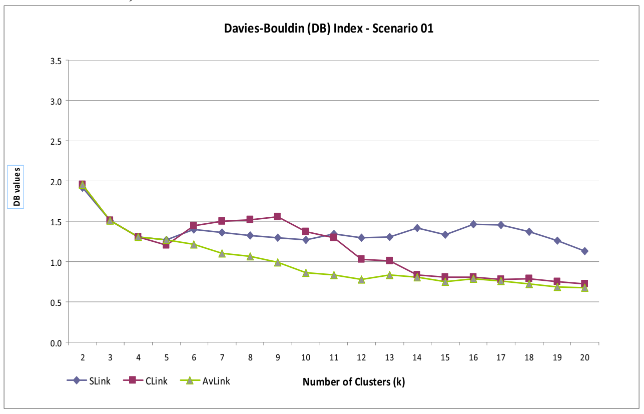

Figure 3. The experiments results for scenario 1.

As it can be seen in Figure 3, when using *AverageLink* the values were lower (better) than when using *Complete* and *SingleLink*. In fact, *SingleLink* results were considerably high (worst results). *CompleteLink* results, at the end, were quite similar to *AverageLink* ones. However, they were quite high at the beginning.

Scenario 2: The approach used in scenario 2 was completely different from the one used in scenario 1. Instead of clustering objects types 1, 2 and 3 in a single large table (worksheet approach), we clustered each original table separately and clustered the results together in a second stage. This relational clustering approach is similar to the method proposed in [8, 9]. It avoids null values (one Select command per table) and has been adjusted to cope with our dataset particularities.

More precisely, in the first stage, we used the agglomerative hierarchical clustering algorithm and the similarity measure equation (2) (distance function) to cluster objects of type 1 (*userType, objectType, and pointNumber*), objects of type 2 (*faunaName, faunaPresence and pointNumber*) and objects of type 3 (*floraName, floraPresence and pointNumber*) separately.

In this stage, the link type and number of *k* parameters were varied in the experiments. The value of *k* varied from 2 to 20 for object type 1, from 2 to 6 for object type 2. The object type 3 had k = 2. These different ranges are due to the number of different values for their categorical attributes (6 types of fauna and 2 types of flora).

The similarity between two clusters of different objects is determined by the inter-relationships between them, which is defined as:

$$Sim(C_{ip}, C_{jq}) = \left\{ \frac{|R_{\text{inter}}(C_{ip}, C_{jq})| * |X_i| * |X_j|}{|C_{ip}| * |C_{jq}| * |R_{\text{inter}}(X_i, X_j)|}, i \neq j \right. \quad (4)$$

where C_{ip}, C_{jq} are two clusters of X_i and X_j, which are in the results of the first stage, $|C_{ip}|$ and $|C_{jq}|$ are the number of instances in C_{ip} and C_{jq}, $|R_{inter}(C_{ip},C_{jq})|$ is the number of inter-relationships between C_{ip} and C_{jq}, $|X_i|$ and $|X_j|$ are the number of clusters in X_i and X_j, $|R_{inter}(X_i,X_j)|$ is the number of inter-relationship between X_i and X_j.

As there is no relationship between tables fauna and flora (objects of type 2 and 3), the resulting clusters of these two tables were merged with the clusters of the main table. In other words, objects of type 1 with object of types 2 and object of type 1 with objects of types 3.

For the experiments of scenario 2, we varied the value of *k* from 2 to 20 for object type 1, and set *k=4* and *k=2* for object types 2 and 3 respectively. The *k* values for object types 2 and 3 were chosen according to the DB values obtained for each one separately.

As it can be seen in Figure 4, the DB index values for *AverageLink* were lower compared to *SingleLink* and *CompleteLink* values. Again *SingleLink* vales were higher than any other values. *CompleteLink* values began very high compared to *AverageLink* and *SingleLink* ones, but they decreased toward the end and finished with similar values as those of *SingleLink*. As it could be seen in both scenarios, the DB index values for *AverageLink* were the lowest. For this reason, only *AverageLink* values will be used for further comparison between scenarios 1 and 2.

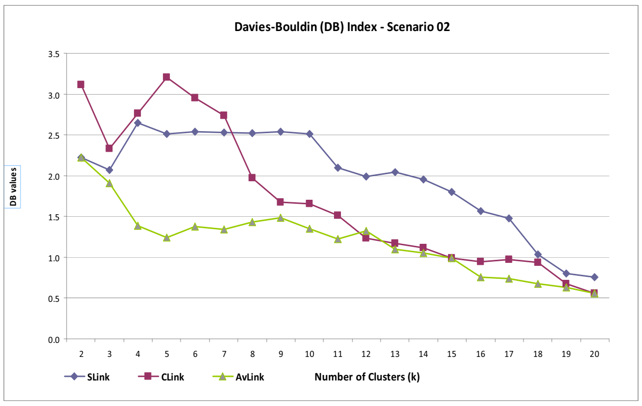

Figure 4. The experiments results for scenario 2.

6.1. Comparing the 3 Different Scenarios

Figure 5 shows the DB index values (*AverageLink*) for the proposed scenarios 1 and 2. As it can be seen in the graph, the values for both scenarios began very high (1.95 and 2.22), but they decreased below 1.5 when *k* was equal to 4 and kept decreasing until reaching points just above 0.5. Scenario 2 had the lowest DB index value at the end (*k=20*). However, Scenario 1 had better values (lower) in the beginning of the experiments until *k=15*.

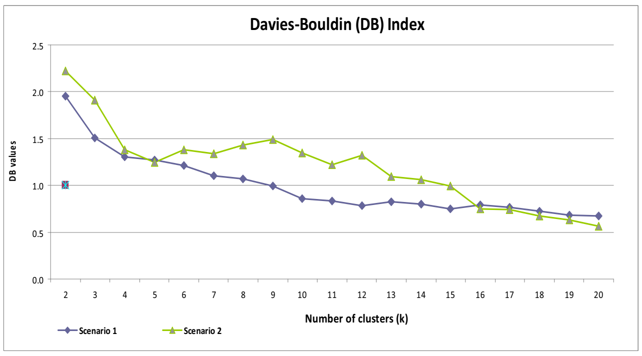

Figure 5. The DB index values for the scenarios.

After computing the cluster validity values for the results obtained in scenario 2, we could see that the DB index values for scenario 2 are smaller than the ones for scenario 1 when k is greater than 15. It indicates that the clusters merged by using the inter-relationships are very compacted (compactness criteria).

7. Conclusion

The main contribution of this paper is to propose a new variant of a relational clustering method to create a recommendation list of most accessed geographical information. In order to do so, we developed a system environment where users can visualize geographical information by accessing them through a web-based GIS and have their actions monitored and stored by the system.

Secondly, by using data stored in relational tables (web log data of users), we compared the effectiveness of a new approach of clustering multi-type relational data proposed in [8, 9] with other old technique (all attributes in one large table).

Based on the results obtained from the analyzed scenarios (1 and 2), we can conclude that the proposed relational clustering method approach had significant positive results on clustering our dataset. Although the entire clustering process is made in two stages (not straightforward), the outcome clustering results were satisfactory and can be easily used for information recommendation purposes.

Finally, as future work, an investigation needs to be performed on different similarity measures, especially the ones suitable for categorical attributes. As the one used in this work was very simple on dealing with categorical attribute values.

Acknowledgements

This work has the financial support of CAPES and CNPq (Brazilian Research Councils), under processes numbers BEX 2481/09-0 and 550810/2007-2.

References

[1] R. Alfred, and D. Kazakov. "A Clustering Approach to Generalized Pattern Identification Based on Multi-instanced Objects with DARA". *Advances in Databases and Information Systems (ADBIS)*, Varna, Bulgaria, September 29-October 3, 2007, pp. 38-49.

[2] I. Bhattacharya, and L. Getoor. "Relational Clustering for Multi-type Entity Resolution". *in Proceedings of the Fourth International Work-shop on Multi-Relational Data Mining (MRDM'05)*, Chicago, USA, August 21, 2005.

[3] A. Canuto, and M. Gomes. "Carcara: A Multi-agent System for Web Mining using Adjustable User Profile and Dynamic grouping". *IEEE/WIC/ACM International Conference on Intelligent Agent Technology(IAT'06)*, Hong Kong, China, December 18-22, 2006, pp. 187-190.

[4] D. Chao, and He Lili. "Multi-agent framework for distributed geographic information services", in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMC)*, Vol. 6, IEEE, pp. 5554–5559.

[5] J. Lee, Y. Lee, and M. Park. "Clustering with Domain Value Dissimilarity for Categorical Data". *Industrial Conference on Data Mining (ICDM 2009)*, Leipzig, Germany, July 20-22, 2009. LNAI 5633, pp. 310-324.

[6] D. L. Davies, and D. W. Bouldin. "Cluster Separation Measure". IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 1, No. 2, pp. 95-104, 1979.

[7] M. Eirinaki, and M. Vazirgiannis, "Web mining for web personalization," *ACM Transactions on Internet Technology*, Vol. 3, No. 1, February 2003, pp. 1-27.

[8] Y. Gao, D. Liu, C. Sun, and H. Liu. "A Two-Stage Clustering Algorithm for Multi-type Relational Data". *Ninth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, Phuket, Thailand, August 6-8, 2008.

[9] Y. Gao, H. Qi, D. Liu, and H. Liu. "Semi-Supervised K-means Clustering for Multi-Type Relational Data". *Seventh International Conference on Machine Learning and Cybernetics*, Kunming, China, July 12-15, 2008.

[10] R. Iváncsy, and I. Vajk. "Frequent Pattern Mining in Web Log Data". Acta Polytechnica Hungarica, Vol. 3, No. 1, pp. 77-90, 2006.

[11] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review", *ACM Compute Survey*, 1999, 31, pp. 264-323.

[12] J. Neville, M. Adler, and D. Jensen. "Clustering Relational Data Using Attribute and Link Information". *in Proceedings of the Text Mining and Link Analysis Workshop, 18th International Joint Conference on Artificial Intelligence*, Acapulco, Mexico, August 9-15, 2003.

[13] X. Yin, J. Han, and P. S. Yu, "Cross -Relational Clustering with User's Guidance", *Eleventh International Conference Knowledge Discovery and Data Mining – KDD'05*, Chicago, Illinois, USA, August 21-24, 2005.