Wang X, Zhou X, Wang S. Summarizing large-scale database schema using community detection. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 27(3): 515–526 May 2012. DOI 10.1007/s11390-012-1240-1

Summarizing Large-Scale Database Schema Using Community Detection

Xue Wang[1] (王 雪), Xuan Zhou[2] (周 烜), and Shan Wang[1,2] (王 珊), *Senior Member, CCF, Member, ACM*

[1]*School of Information, Renmin University of China, Beijing 100872, China*

[2]*Key Laboratory of Data Engineering and Knowledge Engineering, Renmin University of China, Beijing 100872, China*

E-mail: xuew@ruc.edu.cn; xuan.zhou.mail@gmail.com; swang@ruc.edu.cn

Received September 4, 2011; revised December 31, 2011.

Abstract Schema summarization on large-scale databases is a challenge. In a typical large database schema, a great proportion of the tables are closely connected through a few high degree tables. It is thus difficult to separate these tables into clusters that represent different topics. Moreover, as a schema can be very big, the schema summary needs to be structured into multiple levels, to further improve the usability. In this paper, we introduce a new schema summarization approach utilizing the techniques of community detection in social networks. Our approach contains three steps. First, we use a community detection algorithm to divide a database schema into subject groups, each representing a specific subject. Second, we cluster the subject groups into abstract domains to form a multi-level navigation structure. Third, we discover representative tables in each cluster to label the schema summary. We evaluate our approach on Freebase, a real world large-scale database. The results show that our approach can identify subject groups precisely. The generated abstract schema layers are very helpful for users to explore database.

Keywords schema, summarization, large scale, community detection

1 Introduction

Recently, there has been a trend of collecting knowledge data into open shared databases on the Web. Typical examples include Freebase and DBPedia. Both contain thousands of tables and concepts. While these databases are intended to be used by normal Web users, their complex schemas make them difficult to query and explore. Furthermore, the documentation and metadata of these databases are often incomplete, imprecise or simply missing, making the database more difficult to use.

Schema summarization is an effective technique to improve the usability of a complex database. It generates a number of abstract levels over a database schema, to shield users from the complexity of the underlying schema structure. The abstract levels contain the "important" schema elements and their relationships. Using the abstractions, users can have an overview of the schema structure and easily navigate to the part they are interested in. For instance, Fig.1 shows a small fraction of the Freebase schema, which contains 20 tables. These tables apparently belong to two topic groups. One is about American football. The other is about amusement ride. A user can first look at the groups and determine which one contains the desired information. Then he/she can zoom into the right group to browse the tables. The level of topic groups can significantly shorten the information seeking process.

There have been a number of recent proposals about how to conduct schema summarization automatically. Various experiments have been conducted to verify the effectiveness of these approaches on small and medium databases. However, these approaches show limited usage on large-scale databases.

First, in a typical large-scale database, there are usually a number of tables of high connectivity, such as the *person* table and the *location* table in Freebase. Each of the high-degree tables is referenced by a large number of the other tables. As a result, the tables in the entire database are connected closely around these

Regular Paper

This work is partly supported by the "HGJ" National Science and Technology Major Project of China under Grant No. 2010ZX01042-001-002, the National Natural Science Foundation of China under Grant No. 61070054, the National High Technology Research and Development 863 Program of China under Grant No. 2009AA01Z149, the Research Funds of Renmin University of China under Grant No. 10XNI018 and the Postgraduate Science & Research Funds of Renmin University of China under Grant No. 12XNH177.

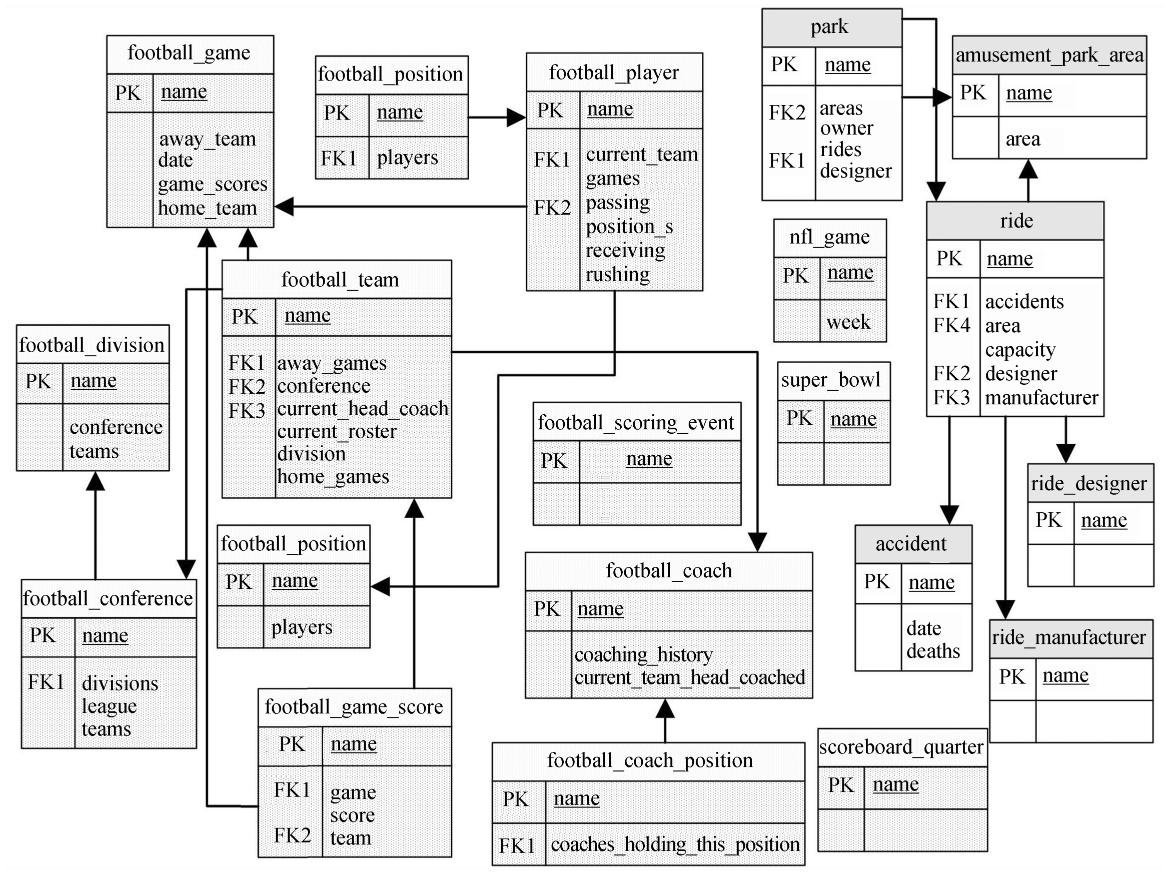

Fig.1. Part of the schema graph of Freebase.

high degree tables and are difficult to be separated into meaningful clusters using existing schema summarization algorithms. Such high-degree tables cannot be simply viewed as noise, as they usually contain highly important information. On the contrary, they need to be placed properly in the schema summary.

Second, most existing approaches of schema summarization use foreign-key relationships as the main information to generate table clusters. However, a shared open database are usually integrated from different data sources or contributed by different users. Foreign keys between the tables from different sources are sparse. We need additional information apart from foreign keys to generate good schema summaries.

Third, in a database with thousands of tables, a simple summarization can only reduce the knowledge base from thousands of tables to hundreds of groups, which are still too many to navigate for normal users. Further abstraction needs to be performed to create a multi-level summary of the schema.

In this paper, we apply the technique of community detection in social networks to schema summarization. We show this new approach can generate good schema summaries on large scaled databases. Our contributions can be summarized as follows:

• We propose a community detection based algorithm to partition a database schema into subject groups, such that different subject groups present different specific topics. As the community detection techniques of social networks, especially the ones based on modularity[1-2], can handle high-degree nodes properly, it helps us avoid the impact of high-degree tables.

• We propose a clustering algorithm to automatically generate more abstract topic groups over the subject groups, so as to form a multi-level navigation structure in the schema summary.

• To maximize the accuracy, we propose to use three types of information in schema summarization. They include foreign key relationship, subclass relationship and overlap of data instances.

• We evaluate our schema summarization techniques over Freebase. The results demonstrate that our algorithm can identify subject clusters with a high degree of accuracy and outperform the existing approaches significantly.

In the rest of this paper, Section 2 provides an overview of our approach. Section 3 introduces how to apply community detection to identify subject groups.

Section 4 presents the algorithm for generating abstract schema summary. Section 5 discusses discovering representatives for subject groups and clusters. Section 6 presents experimental results. Section 7 introduces the related work, and we conclude the paper in the last section. We use Freebase as the running example throughout this paper.

2 Big Picture

In this section, we use Freebase as the example to illustrate the process of schema summarization and the issues it is faced with. Fig.1 shows a subset of the Freebase schema, which contains 20 tables and 16 foreign key relationships among the tables. Each table in the schema consists of multiple attributes, where the primary keys and the foreign keys are marked with PK (Primary Key) and FK (Foreign Key). The schema can be looked as a graph, where the nodes represent the tables and the edges represent the foreign key relationships.

It is obvious that the tables in the schema can be divided into two groups (which are in different backgrounds), which provide information about two separate subjects — American football and amusement ride. We call these groups *subject groups*. The goal of schema summarization is to automatically identify such subject groups. These subject groups together constitute a schema summary. To explore the database, a user could first look at the schema summary. After identifying interesting subject, the user can zoom into the subject group to find more detailed information.

A large database can contain hundreds of subjects, such that a scan of the subjects can be time consuming too. To further improve the usability, schema summarization can cluster the subject groups into more abstract topic groups. For example, the American football group in Fig.1 can be merged with several other subject groups about sport, such as the baseball and basketball subject groups, to form a sport group. We call these abstract groups *abstract clusters*. The abstract clusters and the subject groups then constitute a multi-layer navigation structure for users to explore the database schema.

For users to understand the subject groups and the abstract clusters, we need to label them with representative terms, so that users can have an overview about the contained information by each subject group or abstract cluster without looking at the tables. To label subject groups, we identify the most important tables in each subject group and use the table names as the labels. For example, in the American football group in Fig.1, the most important tables are *football_team* and *football_game*. Intuitively, we can use "football" as the label. In the amusement ride group, the important tables include *park* and *ride*. Then we can use "park ride" as the label. To label abstract clusters, we identify the most important subject groups in each cluster and use the labels of these important subject groups as the labels of the cluster.

The subject groups, the abstract clusters and the labels together form the schema summary of the database. When users explore the database, they can first look at the abstract clusters. If they are interested in a cluster, they can expand the cluster and view what subject groups it includes. If they find interesting subject group, they can expand it into detailed tables. This is much faster and easier than browsing through the thousands of tables in the database.

The major issue of schema summarization on a large schema is the high-degree nodes in the schema graph. Fig.2 shows the distribution of the node degree in the Freebase schema graph. We can see that a small number of tables in Freebase are connected to more than 100 edges. These high-degree nodes play as hubs in the schema graph, which connect most of the nodes together. As a result, a normal graph clustering algorithm may not be able to separate the nodes into the different subject groups or abstract clusters.

In the following, we introduce how the community

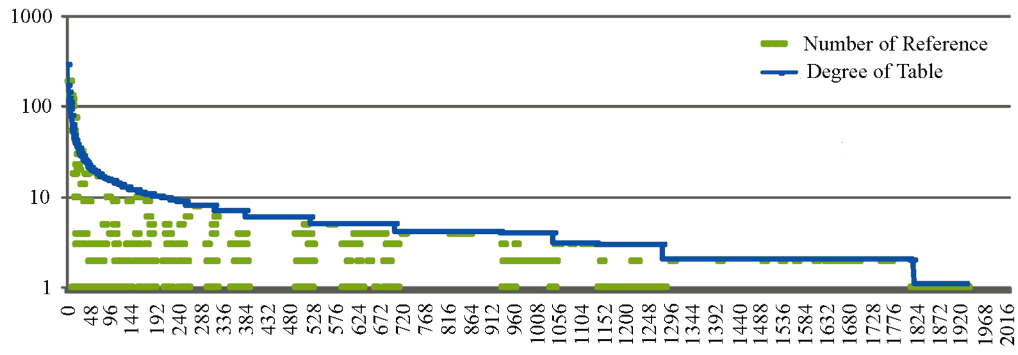

Fig.2. Degree of tables in Freebase.

detection techniques can be used to automatically discover the subject groups and the abstract clusters in a large database schema.

3 Identifying Subject Groups

The intuition applied by most schema summarization approaches is that tables about the same subject are often closely connected by foreign keys, while tables about different subjects are less connected. Therefore, existing approaches usually perform clustering on the schema graphs to generate schema summaries. As mentioned previously, in a large scaled database, there are always a few high-degree tables that are connected to a large number of many foreign keys, such that most clustering approaches do not work well. To address this issue, we resort to the community detection techniques used in social network mining.

Before presenting our algorithm of subject group detection, we first discuss how to weight the foreign keys in the schema graph.

3.1 Weighting Foreign Keys

In a shared database, such as Freebase, there are usually two types of foreign keys. One represents reference relationships. For instance, the *director* attribute in the *film* table is a reference foreign key, which refers to the primary key of the *director* table. The other type of foreign key represents the subclass relationships. For instance, the primary key of the *actor* table is also a foreign key of the *person* table, representing that actor is subclass of person (see Fig.3). As these two types of foreign keys carry two types of information, they need to be treated differently in schema summarization.

person	
PK	name
	date_of_birth
	place_of_birth
	nationality
	religion
	gender
	parents
	children

actor	
PK	name
	date_of_birth
	place_of_birth
	nationality
	religion
	gender
	parents
	children
	film
	dubbing_performances
	netflix_id
	nytimes_id

Fig.3. Subclass relationship between tables.

Intuitively, reference relationships represent closer semantic connection, as they are used more frequently in information seeking. Subclass relationships appear less effective in schema summarization, as a super class can contain many subclasses that are spread across different domains. Our approach chooses to assign different weights to the reference relationships and the subclass relationships. For reference relationships, we always assign them with the weight 1. The weight of a subclass relationship is computed using the following formula:

$$w = \arg\max_{0<x<1}(Q_x/C_x). \qquad (1)$$

In (1), Q_x is the modularity of the resulting schema summary, whose definition will be introduced subsequently. It measures how good the resulting subject groups are separated from each other. C_x is the number of the resulting subject groups.

3.2 Identifying Subject Groups

Due to the presence of high-degree tables in a large scaled database schema, the clustering algorithms adopted by existing schema summarization techniques do not work. We observe that high degree nodes are very common in social networks too. The techniques of community detection in social networks are designed intentionally to avoid the negative impact of high degree nodes. This inspires us to apply the community detection techniques to detect subject groups in database schemas.

There are a large number of community detection algorithms in the literature[3]. Most of the algorithms build upon a common principle, that is, the connectivity within a community should be high, and the connectivity between communities should be low. Applying this principle, we can naturally exclude the high degree nodes from communities, as the inclusion of a high-degree node in a community tends to increase the community's external degree. The measure of modularity[1] is a typical application of this principle, which has been widely used in community detection. Therefore, in this paper, we use a modularity-based algorithm[2] to identify subject groups in schema graphs. Other typical algorithms for community detection, such as the N-cut graph partitioning approach[4] and the spectral partitioning approach[5], are also applicable. As it is not an objective of this paper to compare different approaches of community detection, we schedule the evaluation of those approaches in our future work.

The modularity function was first introduced in [1]. It is used to measure the quality of a particular division of a network into communities. The function is defined as:

$$Q = \sum_{i}^{C}(e_{ii} - a_i^2), \qquad (2)$$

where e_{ii} is the fraction of edges with both end vertices in the same community i, and $a_i = \sum_j e_{ij}$ which corresponds to the fraction of edges with at least one end vertex in community i. Using this quality function, the influence of the high-degree nodes can be minimized in community detection. This is because the a_is of the high-degree nodes tend to be very big. Therefore, we apply the same function to measure the quality of the subject groups identified from a database schema.

To maximize the modularity, we can use either the divisive method or the clustering method. The former works by removing network elements (edges and vertices) that are positioned between communities[6]. For instance, the seminal algorithm by Girvan and Newman[1] progressively removes the edges of a network with the highest betweeness until communities emerge as disconnected components. The latter[2], on the contrary, repeatedly joins communities together in pairs until getting an approximate maximum modularity.

In our approach, we implement a greedy clustering community detection algorithm. We first view each node in the schema graph as a subject group. Then, we join the subject group pairs wisely until the modularity function is maximized. At each step, we evaluate each pair of subject groups, and merge the pair that results in the greatest increase in Q. The change in modularity Q upon joining two subject groups is given by ΔQ[2]:

$$\Delta Q(C_i, C_j) = e_{ij} + e_{ji} - 2a_i a_j, \tag{3}$$

where C_i and C_j are subject groups. Fig.4 shows our greedy algorithm of subject group identification, in which the *Join*() function is used to combine two subject groups into one.

```
Input: A schema graph $G(V, E)$.
Output: Subject groups $C = \{C_1, C_2, \ldots C_m\}$.
GreedyDiscSG($G = (V, E)$)
1.   $C = \{C_1, C_2, \ldots, C_n\}$, $C_i = \{R_i | R_i \in V\}$
2.   $numberofSG = n$;
3.   Do
4.       $isJoined$ = false;
5.       for $i = 1$ to $numberofSG$
6.           for $j = 1$ to $numberofSG$
7.               $\Delta Q = \Delta Q(C_i, C_j)$ s.t. $\Delta Q(C_i, C_j) =$
                     $\max_{i,j}(\Delta Q)$;
8.           end for
9.       end for
10.      if $\Delta Q > 0$
11.          $isJoined$ = true;
12.          $Join(C_i, C_j)$;
13.          $numberofSG--$;
14.      end if
15.  while $isJoined$
16.  return $C$;
```

Fig.4. Greedy algorithm for identifying subject groups.

It is worth mentioning that we have assigned different weights to the edges in a schema graph. Therefore, e_{ij} in (3) represents the fraction of the aggregated weight on the edges rather than the number of edges.

There are a number of techniques to optimize the community detection algorithms based on modularity[2,6]. As our goal is to evaluate how a generic community detection approach can help in schema summarization, we do not apply any optimization in this paper. For comparison, we implement a divisive algorithm for community detection. The comparison results will be given in Section 6.

4 Generating Abstract Schema Levels

For a large-scale database, there can be hundreds of subject groups. These subject groups need to be clustered to generate a higher abstract level for users to understand the structure of database easily. In this section, we introduce how to aggregate subject groups into abstract clusters to form a hierarchical summary.

4.1 Subject Group Similarity

We observe that the data of two relevant subject groups usually intersect. For example, in the subject groups of baseball and Olympics, there are a significant number of persons who are both baseball players and participants of Olympics. Such intersection enables us to measure how close two subjects are in semantics.

A straight forward approach is to use the fraction of intersected tuples to measure the similarity between two subject groups. This can be inaccurate. As some tables are more important and more representative than the others within a subject group, the intersection on more important table should count more in principle. Therefore, we formally define the intersection measure as follows.

Definition 1. *Let R and S be two subject groups with respect to database D. The similarity of R and S is*:

$$Sim(R, S) = P(R \cap S)/(P(R) + P(S) - P(R \cap S)), \tag{4}$$

where

$$P(R) = \sum_{T_i \in R, T_j \in D} I(T_i) \times I(T_j), \tag{5}$$

$$P(R \cap S) = \sum_{T_i \in R, T_j \in S} I(T_i) \times I(T_j), \tag{6}$$

T_i, T_j present tables, $I(T_i)$ presents importance (discussed in Section 5) of table T_i.

4.2 Neutralizing Impact of High-Degree Tables

As we discussed before, tables of high degree usually bring negative impact to the clustering of a schema graph. The same applies to the generation of the abstract levels, as the data in high-degree tables intersects more frequently with the other data than the data in low-degree tables. For example, the person table stores the basic information about persons in Freebase, which is repeated in many other subject groups. Such intersection is not very helpful in the clustering of subject groups.

We therefore choose to remove the impact of the high degree tables. Basically, we select the top 5 tables in the database with highest degree as the high-degree tables, and simply ignore them during the clustering process. In most cases, the subject groups containing high-degree tables will be put into a single cluster in the clustering result. This actually complies with our intuition, as the groups containing high-degree tables are usually about general concepts.

4.3 Hierarchical Clustering Algorithm

To generate abstract clusters, we propose an agglomerative hierarchical clustering algorithm. This algorithm starts by putting every single subject group in a single cluster. In each successive iteration, the closest pair of clusters is identified and merged. The process continues until the end condition is satisfied.

In the clustering, the distance between two clusters C_i and C_j is defined as:

$$Dist(C_i, C_j) = \max(1 - Sim(R, S)), \tag{7}$$

where R, S are subject groups satisfying $R \in C_i$ and $S \in C_j$. The complete algorithm is presented in Fig.5. The input is a schema graph G, a set of subject groups S with high degree tables removed and the number of desired clusters k. The output of this algorithm is a set of abstract clusters C. The computational complexity of the algorithm is $O((|S| - k)|S|)$, where $|S|$ is the number of input subject groups and k is the cluster number.

```
HierarchicalCLUS($G = (S, E), k$)
$C = \{C_1, C_2, \ldots, C_m\}$: current clustering;
1.  $C_i = \{S_i\}$; /*each subject group is a single cluster*/
2.  for $i = |S|$ to $k$ /*iteration*/
3.      find $C_p, C_q \in C$ with a minimum distance;
4.      merge cluster $C_p$ into $C_q$;
5.      delete $C_p$ from $C$;
6.      for each $C_j$ in $C$
7.          compute distance between $C_j$ and $C_q$;
8.      endfor
9.  endfor
10. return ($C$);
```

Fig.5. Hierarchical clustering algorithm.

5 Representatives of Cluster and Subject Group

For each identified subject group, we choose the most important tables and use the table names as the label of the subject group. With the labels, users can understand the contents of each subject group easily. We implement the importance function introduced in [7] to measure the importance of tables. Using this measure, each table is first given an initial importance as:

$$IC(R) = \log|R| + \sum_{R.A \ in \ R} H(R.A),$$

where $IC(R)$ represents the initial importance of the table R, $|R|$ is the number of tuples in R, and $R.A$ is an attribute of R. $H(R.A)$ presents the entropy of the attribute A, which is defined as:

$$H(R.A) = -\sum_{a_i \in R.A} P(a_i)\log(P(a_i)),$$

where $\{a_1, a_2 \ldots a_k\}$ are the possible values of the attribute $R.A$. $P(a_i)$ is the fraction of tuples in R that is given value a_i on $R.A$.

Then, an $n \times n$ probability matrix $\boldsymbol{\Pi}$ is defined to represent the "transfer information" between tables. Each element $\boldsymbol{\Pi}[R, S]$ is defined as:

$$\boldsymbol{\Pi}[R,S] = \sum_{R.A-S.B} \frac{H(R.A)}{\log|R| + \sum_{R.A'} q_{A'} \times H(R.A')},$$

where R, S present two tables, $R.A$-$S.B$ represents a join edge from Table R to Table S, and q_A denotes the total number of join edges involving attribute $R.A$. The outer sum covers all the edges between R and S and the inner sum covers all attributes of table R. Therefore, a diagonal element $\boldsymbol{\Pi}[R, R]$ is defined as:

$$\boldsymbol{\Pi}[R,R] = 1 - \sum_{R \neq S} \boldsymbol{\Pi}[R,S].$$

Finally, the importance vector $\boldsymbol{I}$ of schema graph G is computed by the following iterative process. It starts with an vector $\boldsymbol{V}_0 = (IC(R_1), IC(R_2), \ldots, IC(R_n))$, and repeatedly computes $\boldsymbol{V}_{i+1} = \boldsymbol{V}_i \times \boldsymbol{\Pi}$ until $\text{dist}(\boldsymbol{V}_i, \boldsymbol{V}_{i+1}) \leqslant \varepsilon$ (where dist is defined by the L_∞-metric). Setting $\varepsilon = 0$ means that the process stops when the stationary distribution is reached. The importance of a table R is measured by $I(R)$ in the resulting vector.

To choose the labels of abstract clusters, we use a different approach. As there can be a hierarchy in a large database, the hierarchical information can be used to generate more meaningful labels. For example, the subject group about American football in Fig.1 can be merged with the baseball group, the basketball group and some others to form a cluster about sports. A hierarchy can be built based on the subclass relationship between tables in this cluster. We present the hierarchy of the sport cluster in Fig.6. We can see from this figure, the subject groups constitute a tree with the sports group as the root. Hence, the sports group can be chosen as the representative of the whole cluster, and the term "sports" can be used as the label.

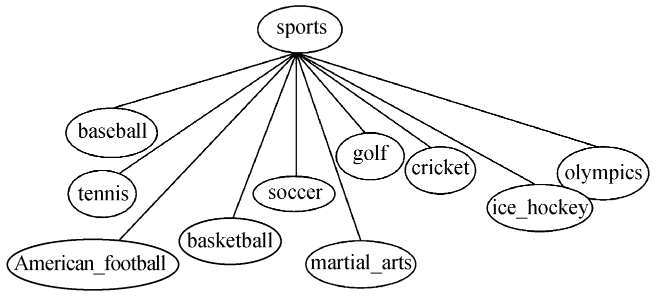

Fig.6. Hierarchy of the sports cluster.

If the hierarchical structural graph of a cluster is not a tree but a forest, the roots of all the trees in the forest can be chosen as the representatives of the cluster. If there are too many roots, we measure the importance of each root using the average importance of the tables in each subject group and choose the most important roots as the representatives.

6 Experimental Evaluation

In this section, we evaluate our schema summarization methods over Freebase dataset.

6.1 Experimental Setup

Dataset. Freebase is a shared database on the Web, which consists of more than 2 000 tables and more than 300 million entities. The contents of Freebase are either integrated from some open data sources, such as IMDB, or contributed by ordinary Web users. While the schema of Freebase is not strictly relational, it can be transformed to a relational schema easily.

Types and Properties. The data objects in Freebase are categorized into *types*[8]. Each *type* is associated with a set of *properties*. For example, music artist is a particular *type*, whose *properties* include album, instruments, etc. From the perspective of the relational model, *types* correspond to relations or tables, and *properties* correspond to attributes. Each *type* in Freebase has an *id* property, which is used as a unique identifier of the objects of that type. This *id* property can be viewed as the primary key of the corresponding table.

Expected Types. Given a *type* of Freebase, some of its *properties* are used as references to other *types*. For example, *place_of_birth* is a *property* of the type *person* that refers to the type *location*. In Freebase, if a *property* is a reference to a certain *type*, that *type* is marked as the *expected type* of the *property*. In our example, *location* is the *expected type* of *place_of_birth*. Therefore, we treat the *expected type* relationships as the foreign key relationships in relational model.

Ground Truth. In Freebase, each table is manually annotated with a *domain*. Therefore the *domain*s can be regarded as a manual schema summarization of Freebase. In our experiments, we use *domain*s as the ground truth to evaluate the effectiveness of schema summarization.

Data Cleaning. To prepare the data and the schema for our experiments, we conducted data cleaning. We dropped tables which do not appear in the schema. We also dropped the reference *properties* whose *expected types* do not exist. For instance, there are a significant number of *properties* referring to the *type* called integer. While integer is a common data type, it is not explicitly defined in the Freebase schema and should not be regarded as a foreign key.

We conducted three sets of experiments altogether. First, we evaluated the effectiveness of our subject group identification algorithm. Second, we evaluated the effectiveness of our complete schema summarization approach. Third, we compared our approach against two most recent proposals on schema summarization[7,9].

Our programs were all implemented in C++, and deployed on a PC with a 2GHZ Core2 Duo CPU and a 2 G RAM.

6.2 Effectiveness of Subject Group Identification

In Table 1, we compare the subject groups generated by our approach against the ground truth — the set of manually created *domain*s in Freebase. To demonstrate the representativeness of our greedy algorithm, we compare it against a divisive algorithm proposed in [1], which repeatedly removes the edges with the largest betweeness until it achieves the maximum modularity. We denote our algorithm by "Greedy" and the divisive algorithm by "Divisive".

In Table 1, the first column lists a subset of the domains in Freebase. The second column shows the number of tables in each *domain*. The third column gives two types of numbers. In each record of column 3, the one outside the bracket is the number of subject groups generated by our approach which can be regarded as sub-categories of the corresponding *domain*. The one inside the bracket is the number of tables in the subject groups. As shown in Table 1, the results of the "Divisive" algorithm are slightly more accurate than the results of the "Greedy" algorithm. For each *domain* of Freebase, the "Divisive" algorithm can normally generate more subject groups than the "Greedy" algorithm. This result is concordant with the conclusion in [2].

Table 1. Subject Groups

Domain (Ideal Subject Group)	No. Tables	No. Matched Subject Groups (No. Tables)		Accuracy		No. Isolated Nodes
		Greedy	Divisive	Greedy	Divisive	
American_football	23	1(20)	4(20)	86.7%	86.7%	3
amusement_parks	15	2(15)	1(15)	100.0%	100.0%	0
astronomy	63	2(62)	13(62)	98.4%	98.4%	1
bicycles	3	1(3)	1(3)	100.0%	100.0%	0
biology	47	4(43)	10(44)	91.5%	93.6%	0
business	54	7(47)	18(49)	87.0%	90.7%	2
education	28	2(28)	5(28)	100.0%	100.0%	0
engineering	24	3(21)	7(20)	87.5%	83.3%	3
event	15	4(10)	6(11)	66.7%	73.3%	0
film	52	1(49)	5(49)	94.2%	94.2%	1
food	40	7(37)	9(38)	92.5%	95.0%	2
Freebase	110	8(69)	13(71)	62.7%	64.5%	38
government	33	2(30)	6(31)	91.0%	94.0%	0
location	150	3(145)	7(147)	96.7%	98.0%	2
measurement_unit	91	1(79)	4(83)	87.8%	91.2%	0
media_common	20	5(14)	7(17)	70.0%	85.0%	0
organization	24	2(23)	4(22)	95.8%	91.7%	0
people	26	3(16)	6(18)	61.5%	69.2%	1
sports	46	4(41)	9(42)	89.1%	91.3%	0
symbols	19	1(19)	3(19)	100.0%	100.0%	0
time	13	1(8)	4(10)	61.5%	76.9%	0
Average	**42.7**	**3(37.5)**	**7(38.0)**	**87.0%**	**89.4%**	**2.5**

However, as the "Greedy" algorithm runs much faster than the "Divisive" algorithm and its accuracy is not significantly worse than the "Divisive" algorithm, we believe it is a more practical approach in real world.

We can see that the subject groups generated by our approach have a smaller granularity than the *domains*, such that multiple subject groups can be classified into a single *domain*. For instance, as shown in Table 2, the subject groups of Disney ride and amusement park are joined into a single *domain* named amusement_parks. The third column shows the accuracies. Each accuracy value is calculated as the percentage of tables in the subject groups that belong to the corresponding *domain*.

Table 2. Domain Amusement Parks

Domain	Subject Group	No. Tables
amusement_parks	Disney ride	3
amusement_parks	amusement parks	12

Table 1 is only for illustrating the results of subject group identification. As there are more than 80 *domains* in Freebase, not all of them can be presented in Table 1. The complete statistics of accuracy are shown in Fig.7, which will be discussed in more detail subsequently. As we can see from the result, the accuracy of the subject group detection varies for different *domains*, and the average accuracy is around 86%.

According to our observation, the majority of the inaccurate cases are due to the inability of our approach to separate two closely connected subject groups. Table 3 gives an example. The beer group and the game group are actually two subject groups belonging to different *domains*. Our algorithm was not able to separate them, because there are too many foreign keys between these two groups. This can be understood, as people like to drink beer when watching or playing games. Nevertheless, such cases are not common in our results, and our approach is in general quite accurate.

6.3 Effectiveness of Schema Summarization

After the subject groups were generated, we performed the algorithm in Section 4 to generate the abstract clusters and the labeling method in Section 5 to construct the final schema summary. Fig.8 shows a

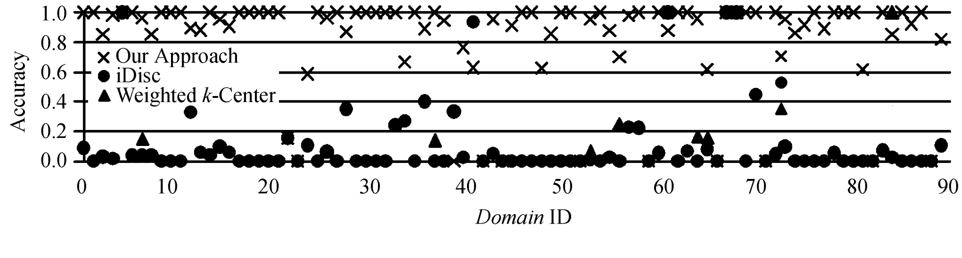

Fig.7. Clustering accuracy (compared with *domain* in Freebase).

Table 3. Subject Group game&beer

Table	Domain
beer	food
beer_container	food
beer_containment	food
beer_country_region	food
beer_production	food
beer_style_category	food
beer_style	food
brewery_brand_of_beer	food
game	games
game_designer	games
game_expansion	games
game_genre	games
game_publisher	games
playing_card_deck_type	games
playing_card_game	games
playing_card_game_play_direction	games

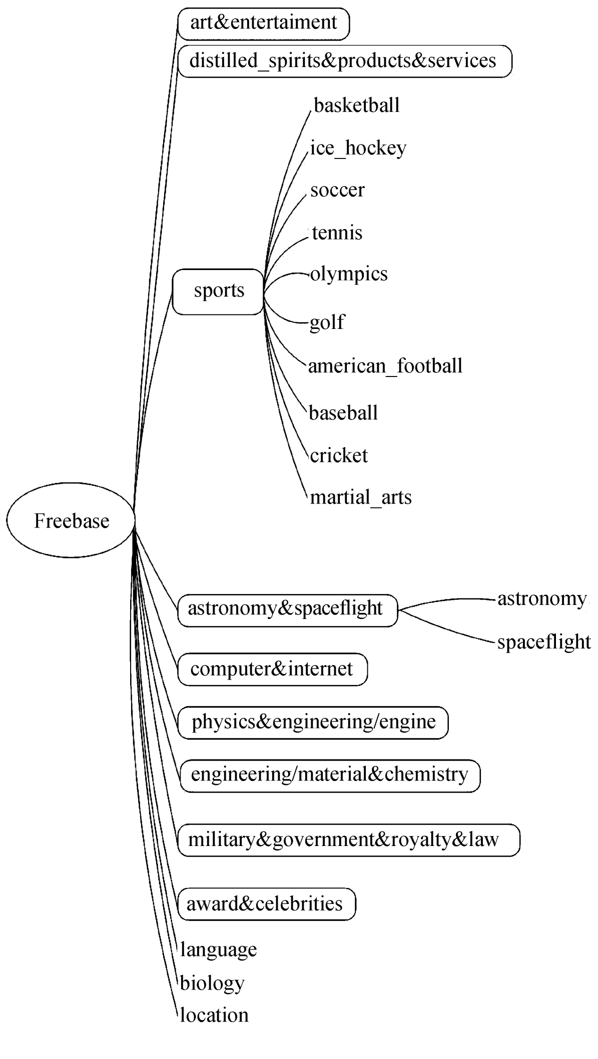

Fig.8. Schema summarization of Freebase.

part of the final summary. We can see that the summary consists of two levels. The first level contains the abstract clusters.

Each abstract cluster consists of a number of subject groups. For instance, the sport cluster contains the subject groups such as baseball, basketball, American football. Each cluster is represented by its most important subjects, which are selected using the labeling method in Section 5. The second level of schema summary contains the subject groups, which are labeled with the names of the most representative tables.

The results show that the subject groups are not uniformly distributed to the abstract clusters. Some clusters contain more than 20 subject groups. Some contain only one subject group. This is mainly determined by the sizes of the intersection between subject groups.

6.4 Comparison with the State-of-the-Art Approaches

We compared our approach against the schema summarization approaches introduced in [7] and [9]. The approach in [7] applies a weighted k-center clustering algorithm[10] to perform schema summarization. It does not distinguish between subject groups and abstract clusters, but generates a single layer of clusters. It also uses the foreign key relationships to quantify the distance between clusters, and the weights on foreign keys are calculated using information theory. We denote this approach by weighted k-center in our evaluation results. To minimize the bias of a single clustering algorithm, the approach in [9] uses a voting scheme to aggregate the effects of multiple clustering algorithms. It also generates a single layer of clusters as the schema summary. We denote this approach by iDisc.

In our experiments, we applied three base clustering algorithms B_1, B_2 and B_3 to iDisc. B_1, B_2 and B_3 are exactly the algorithms used in the evaluation of iDisc[9]. B_1 and B_2 are two hierarchical agglomerative clustering algorithms based on different distance matrixes. B_3 is a community discovery algorithm[11-12] designed for networks. We present the accuracies of subject group identification of the three approaches in Figs. 7 and 9. The accuracies were computed using the same method of Table 1. To ensure fairness, we tuned the parameters of weighted k-center and iDisc, such that they generated the same number of clusters as our subject group identification approach. The x-axis in Fig.7 represents the IDs of *domains* of Freebase, and the y-axis represents the accuracy of the clusters/subject groups. As we can see, our approach exhibits a much higher accuracy than iDisc and weighted k-center. The low clustering accuracy of iDisc and weighted k-center is mainly caused by the effect of the high degree tables in Freebase. Those high degree tables connect most of the tables in Freebase through short paths, such that neither weighted k-center nor iDisc partitions them into different clusters.

As a result, the majority of the tables were assigned to a small number of clusters. Table 4 shows some example clusters generated by weighted k-center. We can see that the film cluster contains 1 699 tables, while the tv_program cluster contains only 1 table. Fig.10 gives a more detailed distribution of the size of the subject groups generated by the three approaches. The x-axis represents the IDs of subject groups and the y-axis represents the number of tables in each group. We can see that the result distributions of weighted k-center and iDisc are much more skewed than the distribution of our approach. In the cases of weighted k-center and iDisc, there are a small proportion of the groups that contain a large proportion of the tables. Such schema summaries do not appear usable.

Table 4. Weighted k-Center Clustering Over Freebase

Cluster ID	Center Table	No. Tables
1	film	1 699
2	book_edition	251
3	music_composition	1
4	tv_program	1
5	computer_videogame	1

While weighted k-center and iDisc can only generate a single layer of clusters, we can tune their granularity. So that the scales of their results are comparable to the abstract clusters generated by our approach. In Fig.9, we compare the accuracies of the abstract clusters generated by the three approaches. In Fig.9, the x-axis represents the manually generated categories for Freebase and the y-axis represents the clustering accuracy. There are in total 9 manually created categories in Freebase. As we can see from Fig.9, our approach still outperforms weighted k-center and iDisc significantly. The major reason is again the effects of the high degree tables.

In summary, the approaches of weighted k-center and iDisc do not appear to be able to generate good schema summaries on a large database schema that contains high degree tables. In contrast, the schema summary generated by our approach is significantly better.

7 Related Work

The problem on conceptual model abstraction[13-17] is similar to that of schema summarization. The approaches on conceptual model abstraction can be classified into two basic types: entity clustering and entity abstraction. Entity clustering partitions entities into categories and picks the most "important" entities to represent the categories. Entity abstraction approaches find the least "important" entities first and delete them from the current conceptual model to form a more abstract conceptual model. All these methods rely heavily on the semantic annotations on the relationships between entities, which do not necessarily exist in a database schema. Therefore, these approaches are not applicable to schema summarization in general cases.

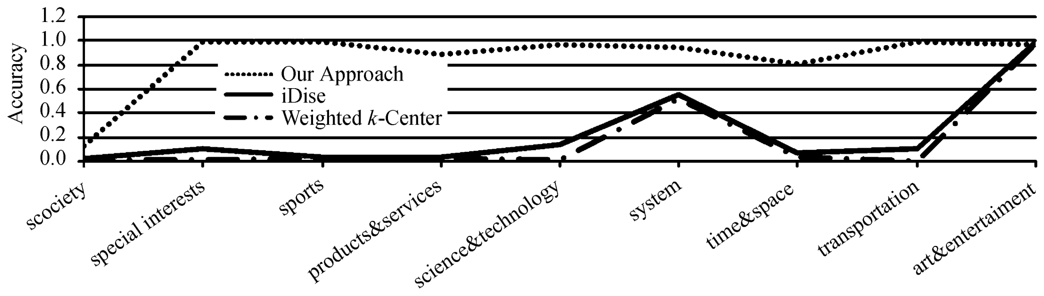

Fig.9. Clustering accuracy (compared with manual hierarchy category in Freebase).

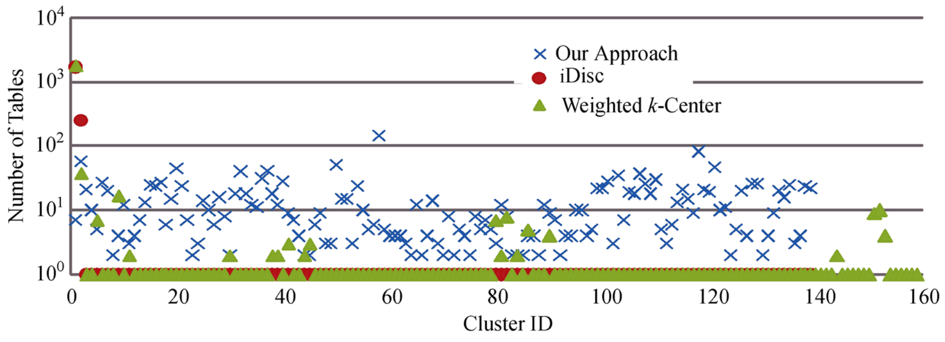

Fig.10. Number of tables in each cluster.

In [17], the authors present an algorithm to find abstract schema elements that can balance "importance" and "coverage". This paper introduced the notion of schema summary for first time. Its algorithm can be used on both XML's and relational databases' schemas. However, it is shown that the approaches in [7] and [9] perform better than that of [18].

In [7], the authors first view the schemas of relational databases as graphs, and apply graph clustering to generate schema summaries. Their approach first calculates the "importance" of each table using a information content and random walk model[19], and uses the most important tables as the centers of clusters. Then it applies a weighted k-center cluster algorithm to cluster the tables.

In [9] the authors describe the iDisc system for generating topical structures of databases. Their approach uses multiple base clustering algorithms[20] to cluster the tables in databases, and applies a meta clustering algorithm to aggregate the clustering results into topic structures. The most important table of each topic cluster will be picked out to present each topic.

As a large scaled database can contain thousands of tables, it is common that there are a few high degree tables to connect the majority of tables closely together. As a result, it is difficult for the approaches of [7, 9, 18] to run successfully. In contrast, as shown in our experimental results, our approaches based on community detection can get rid of the impact of high degree tables and generate good schema summaries. In addition, our approach uses extra information to perform schema summarization, such as the subclass relationships and the intersection of data instances. Such information is not utilized by the previous approaches.

To identify representative tables in a subject group or a cluster, an alternative approach is to utilize the approach of influence maximization, which was introduced by P. Domingos and M. Richardson in [21-22]. Given a social network graph, a specific influence cascade model and a small number k, the influence maximization problem is to find k vertices in the graph, such that the expected number of vertices influenced by the k vertices is maximized. The optimization influence maximization problem is proved to be NP-hard[23]. To optimize the efficiency of schema summarization, we adopted a light weighted approach to identify representative tables (Section 5). In our future work, we plan to compare the effectiveness of influence maximization against that of our approach, and explore how to use influence maximization in schema summarization.

8 Conclusions

In this paper, we use the community detection techniques of social network to identify subject groups of large-scale database. Based on these subject groups, we designed a new algorithm for automatically generating schema summary. We evaluated our approach over Freebase, and demonstrated that our approach can generate schema summaries of good quality and it outperforms the existing approaches significantly.

References

[1] Newman M E J, Girvan M. Finding and evaluating community structure in networks. *Physical Review E*, 2004, 69(2): 026113.

[2] Newman M E J, Fast algorithm for detecting community structure in networks. *Physical Review E*, 2004, 69(6): 066133.

[3] Papadopoulos S, Kompatsiaris Y, Vakali A, Spyridonos P. Community detection in social media. *Data Mining and Knowledge Discovery*, 2012, 24(3): 515-554.

[4] Shi J, Malik . Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000, 22(8): 888-905.

[5] Luxburg U. A tutorial on spectral clustering. *Statistics and Computing*, 2007, 17(4): 395-416.

[6] Rahn E, Bernstein P A. A survey of approaches to automatic schema matching. *J. Very Large Data Base*, 2001, 10(4): 334-350.

[7] Yang X, Procopiuc C M, Srivastava D. Summarizing relational databases. *PVLDB*, 2009, 2(1): 634-645.

[8] www.freebase.com, September 2011.

[9] Wu W, Reinwald B, Sismannis Y, Manjrekar B. Discovering topical structures of databases. In *Proc. SIGMOD 2008*, June 2008, pp.1019-1030.

[10] Dyer M E, Fireze A M. A simple heuristic for the p-center problem. *Operations Research Letters*, 1985, 3(6): 285-288.

[11] Clauset A, Newman M E J, Moore C. Finding community structure in very large networks. *Physical Review E*, 2004, 70(6): 066111.

[12] Lancichinetti A, Fortunato S. Community detection algorithms: A comparative analysis. *Physical Review E*, 2009, 80(5): 056117.

[13] Campbell L J, Halpin T A, Proper H A. Conceptual schemas with abstractions making flat conceptual schemas more comprehensible. *Data & Knowledge Engineering*, 1996, 20(1): 39-85.

[14] Feldman P, Miller D. Entity model clustering: Structuring a data model by abstraction. *The Computer Journal*, 1986, 29(4): 348-360.

[15] Teorey T, Wei G, Bolton D, Koenig J. ER model clustering as an aid for user communication and documentation in database design. *Communications of the ACM*, 1989, 32(8): 975-987.

[16] Huffman S B, Zoeller R V. A rule-based system tool for automated ER model clustering. In *Proc. the 8th International Conference on Entity-Relationship Approach to Database Design and Querying*, Oct. 1990, pp.221-236.

[17] Campbell L J, Halpin T A, Proper H A. CA ERwin data modeler, www.ca.com.

[18] Yu C, Jagadish H V. Schema summarization. In *Proc. the 32nd International Conference on Very Large Data Bases*, Sep. 2006, pp.319-330.

[19] Motwani R, Raghavan P. Randomized Algorithms. Cambridge Univ. Press, 1995.

[20] Han J, Kamber M. Data Mining: Concepts and Techniques (2nd edition). Morgan Kaufmann, 2006.

[21] Domingos P, Richardson M. Mining the network value of customers. In *Proc. the 7th ACM SIGKDD*, Aug. 2001, pp.57-66.

[22] Richardson M, Domingos P. Mining knowledge-sharing sites for viral marketing. In *Proc. the 8th ACM SIGKDD*, July 2002, pp.61-70.

[23] Kempe D, Kleinberg J M, Tardos E. Maximizing the spread of in?uence through a social network. In *Proc. the 9th ACM SIGKDD*, Aug. 2003, pp.137-146.

Xue Wang received the M.S. degree in computer science from Lanzhou University in 2005. She is a Ph.D. candidate at Renmin University of China. Her research interests include databases and information retrieval.

Xuan Zhou obtained his Ph.D. degree from the National University of Singapore in 2005. He was a researcher at the L3S Research Center, Germany, from 2005 to 2008, and a researcher at CSIRO, Australia, from 2008 to 2010. Since 2010, he has been an associate professor at the Renmin University of China. His search interests include database system and information management. He has contributed to a number of research and industrial projects in European Union, Australia and China.

Shan Wang received the M.S. degree in computer science from Renmin University of China in 1981. She is a professor and doctoral supervisor at Renmin University of China. Her research interests include high performance databases, knowledge systems, data warehouses and big data management. She is a senior member of CCF and a member of ACM.