Multimedia Tools and Applications (2019) 78:10077–10096
https://doi.org/10.1007/s11042-018-6543-y

Summarizing database schema based on graph partition

Yingqi Wang [1] · Lianke Zhou [1] · Nianbin Wang [1]

Received: 4 September 2017 / Revised: 9 May 2018 / Accepted: 15 August 2018 /
Published online: 3 September 2018

Abstract

As the underlying database schemas become larger and more complex, it is difficult for casual users to understand the schemas and contents of databases. Therefore, it has become an essential task to summarize the database schemas. However, most prior approaches pay little attention to the topological characteristics between tables, ignore the effect of the user feedback, and fail to accurately predict the number of clusters in the output. This seriously limits their accuracy of schema summarization. To deal with the problems, we propose a new schema summarization method based on a graph partition mechanism. First, we introduce a novel strategy to construct a similarity matrix between tables, which is based on the topology compactness, content similarity and query logs. Then we provide a calculation formula for table importance and a detection scheme of the most important nodes in local areas. Both are used for selecting the initial cluster centers and predicting the number of clusters in the graph partition mechanism. Finally, we evaluate the proposed method over the database TPC-E, and results demonstrate that it achieves high performance in summarizing accuracy.

Keywords Graph partition · Relational database · Schema summarization · Spectral clustering

1 Introduction

The amount of available structured data grows rapidly [1, 10]. There are hundreds of interlinked tables in relational databases in enterprises, government agencies, and research organizations. In addition, the underlying database schema has become more complex and its scale has become larger than ever [3, 23, 24]. It is a substantial barrier for new users to learn about the structures, contents and features of these databases, and also a difficult and daunting task to retrieve the desired information [13, 26]. Schema summarization in relational databases is an effective and proven technique to improve the usability of databases. It aims to categorize the

✉ Lianke Zhou
zhoulianke@hrbeu.edu.cn

[1] College of Computer Science and Technology, Harbin Engineering University, No.145 Nantong Street, Nangang District, Harbin 150001, China

tables which have the similar topics into the same cluster. A user can get the basic knowledge about the information and data distribution in databases by using the subject clusters [8]. Therefore, schema summarization in databases has emerged as a hot research topic [12, 30].

In recent years, substantial research effort has been attracted in the database schema summarization and a number of methods have been developed [2, 9, 18]. Unfortunately, most of them neglect the effect of the user preferences and the topological property of tables in the entire spatial distribution. The efficiency and accuracy of the results can not be guaranteed. In addition, the existing studies fail to develop an effective algorithm for predicting the number of subject clusters and then it is necessary to provide a parameter's setting method. Overall, the quality of results is greatly influenced by human factors.

In reality, an enterprise database always has a huge number of inner-linked tables [27]. Users unfamiliar with the database must expend effort to comprehend the database schema before being able to interact with it. Here we give an example of application from real-world scenarios. A user who wishes to find out all the securities issued by the company AB Volvo and traded on New York Stock Exchange in the TPC-E database, has to study such a complex schema and filter away irrelevant information about AB Volvo, New York Stock Exchange and securities. This problem is aggravated for more complex schemas, especially when documentations are incomplete or missing. This has led to an urgent need for automatic methods to summarize the database schema called schema summarization which is an effective method to reduce the database schema complexity and provides a succinct overview of the entire schema for users, so that users can determine at a glance the type of information the database contains, and how the tables in the database are organized. Although the existing approaches preliminarily solve the problem above, very few of them analyze comprehensively the factors affecting the summarizing accuracy, thus the accuracy of the methods needs to be further improved. For instance, [28, 30] only consider the contents inside the databases but do not take into account the effect of the information in query logs during the process of summarization. Their average accuracies are only 52 and 64% respectively (see Section 5). At present, it is extremely desirable to have a method with high accuracy in this area. In fact, a more accurate summarization result can provide users with more accurate information to help them significantly reduce the time in understanding an unfamiliar system. After analysis, there are three important factors which are critical for the accuracy of summarization results: 1) structure; 2) contents; 3) query logs. This paper starts with the above three factors and puts forward the theoretical formulas, which provides users with a more accurate summarization method. It can help users to find relevant information in a large database quickly and precisely. This is our intention to study the approach GP-RDSS (**R**elational **D**atabase **S**chema **S**ummarization based on **G**raph **P**artition).

The core of our approach is the usage of a suitable graph partition mechanism-Spectral Clustering, which is a kind of classic method and has been applied in the social network field with tremendous success [17, 20]. A database is usually materialized as a schema graph that captures the tables and their relationships with nodes and edges [11]. With this representation, a graph partition method can be employed in the database schema summarization, thus the schema summarizing process is heavily affected by the topology potential and contents of tables. Furthermore, the effect of user preferences on the schema summarization is introduced for the first time in this paper to further improve the accuracy of the method.

The major contributions include the following:

(1) We propose GP-RDSS, a novel schema summarization method based on a graph partition mechanism and characteristics of relational databases.

(2) We design a comprehensive strategy of similarity matrix construction by simultaneously taking into consideration the topology compactness, content similarity, and user feedback.

(3) We present a formula to evaluate the table importance, which is achieved through analysis of the inherent structures and contents of tables and feedback information in the query logs. Then a scheme is proposed to precisely select the initial cluster centers in the graph partition mechanism.

(4) We conduct extensive experiments on the real data set TPC-E benchmark to verify the effectiveness of the method GP-RDSS.

2 Related works

Recently, schema summarization has been investigated extensively and some baseline solutions have already been proposed [4, 15, 19, 22, 25, 28, 30]. Existing studies can be broadly classified into three types [7]. The first one focuses on the ER model abstraction. It aims to cluster ER entities into abstract entities by calculating the semantic similarity between entities. More precisely, this type of methods can be grouped into two categories: entity clustering and entity abstraction. The entity clustering method partitions the entities into different classes and selects the most important entity to represent the entire class. The entity abstraction method utilizes an iterative algorithm. It deletes one or more least important nodes from the schema graph at each iteration until the number of nodes in graph reaches a certain threshold. An abstract conceptual model can be formed in this way. The second one focuses on the XML data. Generally, the XML data has no fixed schema. It is hard to use and comprehend. So schema summary is helpful for data browsing and query optimization. According to the structural character of XML data, it generates a condensed schema tree as the summarization for the hierarchical data model. Space limitations preclude discussing them in more detail, so we restrict the attention to the third one which summarizes schema on a relational database. Unlike above two data models, however, a relational database usually contains hundreds of tables and there isn't semantic information attached to the edges between tables. Therefore, the above methods are not applicable or satisfy completely in the relational schema summarization. Accordingly, it has led to an urgent need to design a summarization method for relational database schemas. Yu et al. formally define the notion of schema summary for the first time and develop a Balance-sum algorithm which can automatically generate the high-quality results according to the summary importance and summary coverage [30]. In order to improve the robustness and scalability of the summary system, Wu et al. design and realize a novel discovery system iDisc based on the above research and a multi-strategy learning framework [22]. Yang et al. put forward a new importance calculation method using a random walk model and information entropies of attributes, then they define a metric space over the tables [28]. On this basis, they construct a Weighted *k*-center algorithm to summarize the relational databases and the accuracy of the result is further improved. In order to make schema summarization methods avoid being impacted by high-degree tables and adapt to large-scale databases, Wang et al. introduce community detection methods to summarize database schema into multiple levels and effectively settle such problems [19]. The above methods help users get an overall picture of the database. However, the foreign/primary key relationships are not contained in the summary. And for this, Yang et al. introduce a novel concept which called summary graph to circumvent the limitations of above approaches [29]. Unlike previous summarization methods, Sampaio et al. propose a new model of relational schema summarization, based on the notion of context, which is well designed, structurally simple and complete [12]. Motivated by the above observation, Yuan et al. develop a three-step schema summarization method based on label propagation [31]. The approach first uses a kernel function to measure the table similarity by considering several relevant features and then exploits label propagation algorithm to automatically create a schema summary, which is more effective than the existing methods.

Unfortunately, these solutions suffer from several key limitations. First, they do not make full use of the structures and content characteristics of databases, and do not reflect the user preferences in the process of schema summarization at the same time. Moreover, the above methods do not provide an effective algorithm for predicting the number of subject clusters. Before performing the algorithm, users have to set the relevant parameters and the quality of results is influenced by human factors. In short, the solutions proposed so far are not effective and reliable enough for wide spread use. To address the above-mentioned issues, this paper presents a schema summarization method based on a graph partition mechanism, and the accuracy of the summarization results is significantly improved.

3 Overview of our approach

3.1 Preliminaries

Before going on, we need to introduce several notations in Table 1 and then give the definitions used in the paper.

Definition 1 (Schema graph) Given a database$D$, the schema graph of D is a directed graph $G(V, E)$, where:

- $V = \{v_1, \cdots, v_n | n > 1\}$ is a set of nodes, each v_i represents a table of the database.
- $E = \left\{e_{v_i \to v_j} | v_i, v_j \in V\right\}$ is a set of edges, each $e_{v_i \to v_j}$ represents the primary-foreign key relationship between tables represented by nodes v_i and v_j.

Table 1 Notations

$G(V, E)$	a schema graph
$C = \{C_1, C_2, \cdots, C_k\}$	a schema summarization with size k
D	a relational database
c_{ij}	the topology compactness between v_i and v_j
$\|v_i\|$	the number of tuples in v_i
ld_{ij}	the logical distance between v_i and v_j
σ	an impact factor
$\mathbf{C}$	the topology compactness matrix
$s_1(T_i, T_j)$	the names similarity between T_i and T_j
$T_i. A_m$	an attribute-column of T_i
$s_2(T_i, T_j)$	the content similarity between T_i and T_j
$\mathbf{S}$	the content similarity matrix
$\mathbf{A_D}$	the similarity matrix
$count(T_i, T_j)$	the number of occurrences of T_i and T_j in the same log record
$\mathbf{A_{Final}}$	the similarity matrix after adjustment
$I_f(T_i)$	the table importance of T_i
$\varphi(T_i)$	the topology potential of T_i

Fig. 1 shows the subset of the TPC-E schema

Definition 2 (Schema summarization) Given a database D with a set of tables T_1, T_2, $\cdots$, T_n, the summary of D with size k is a k-partition $C = \{C_1, C_2, \cdots, C_k\}$ over the tables in D, each partition $C_i \in C$ is composed of a set of tables which have the same topic, $\forall C_i$, $C_j \in C$, $C_i \cap C_j = \phi$, $(i \neq j)$ and $\cup_{i=1}^{k} C_i = \{T_1, T_2, \cdots, T_n\}$.

For example, part of the schema summarization results on TPC-E is illustrated in Fig. 2. Note that it contains 17 tables which can be divided into 4 clusters: Broker, Customer, Market, and Dimension. Tables in the same cluster apparently belong to the same topic.

3.2 Architecture

Schema graph representations are particularly natural when modeling databases. Inspired by this fact, we introduce an optimal graph partition idea in the research to put forward a schema summarization method GP-RDSS. The architecture of GP-RDSS is depicted in Fig. 3.

Input module. The inputs of GP-RDSS will be in three parts, the database itself (the schema graph and contents) and query logs.

Similarity matrix construction module. Based on the above inputs, we construct a similarity matrix with topology compactness and content similarity. In addition, we make some adjustments to the matrix by analyzing the query logs.

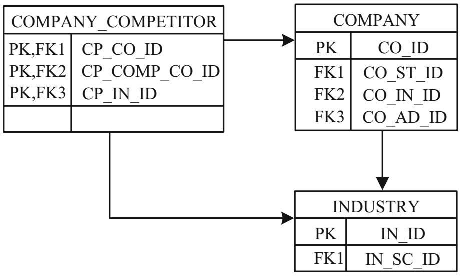
Initial cluster centers detection module. In this module, we define a calculation formula of table importance based on the research about the topology relationship, content similarity, and frequency-of-occurrence in a query log. And then we put forward a scheme to detect initial cluster centers.

Output module. Finally, it returns several subject clusters.

4 Graph partition-based Schema summarization

In generally, a relational database can be modeled as a schema graph or data graph. It provides researches with a formal model which is convenient for further study. In this paper, we model a relational database as a schema graph and categorize tables into different subject clusters on this model. Consequently, the problem of schema summarization can be solved effectively.

Fig. 1 Part of the schema graph of TPC-E



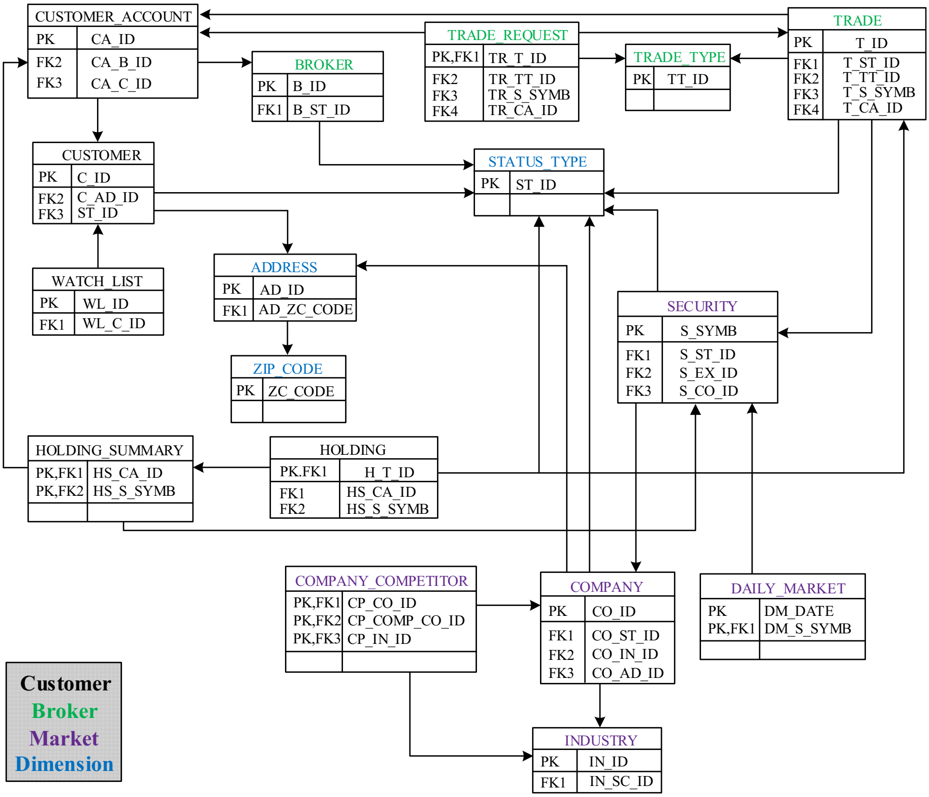


Fig. 2 Part of the schema summarization results of TPC-E

4.1 Construction of similarity matrix

Spectral clustering is known to be a sophisticated graph partition mechanism. It has been widely applied in many fields because of the good robustness and global optimization. Inspired by its success, we introduce the spectral clustering algorithm in the process of schema summarization. The core of this algorithm is to construct the initial matrix, that is the construction of similarity matrix in this paper. It is generally known that whether reasonable construction of the initial matrix directly affects the clustering quality. Similarly, the effective construction of similarity matrix is one of the key factors to improve the accuracy of the schema summarization results. Through the extensive investigation and analysis of influence factors of similarity between tables, this paper puts forward the construction strategy of the similarity matrix. The strategy considers not only the structure and content information, but also the influence from query logs. Therefore, it can calculate the similarity between tables more accurately, which becomes the basis for improving the accuracy of the schema summarization results.

4.1.1 Topology compactness

In this section, we exploit the topology potential to measure the topology compactness [5]. For a schema graph $G(V, E)$, the calculation formula of topology compactness between two nodes is shown in (1):

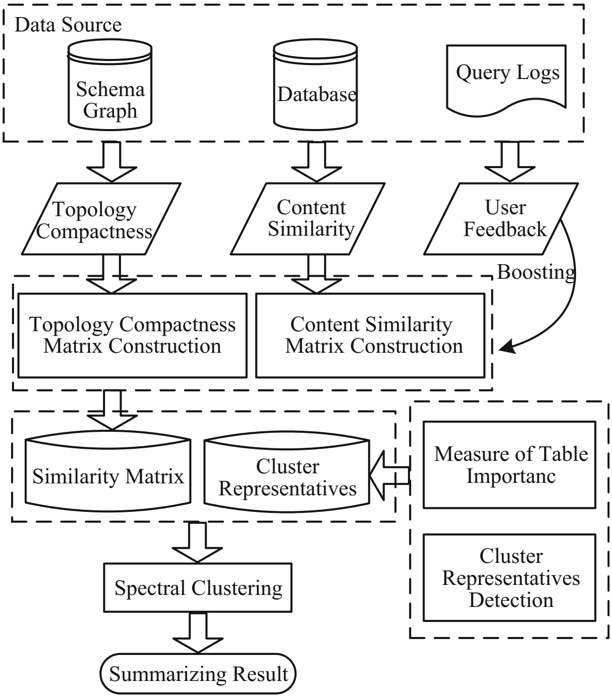


Fig. 3 GP-RDSS architecture

$$c_{ij}=\begin{cases}|v_i|\cdot|v_j|\cdot exp\left(-\left(ld_{ij}/\sigma\right)^2\right) & ld_{ij}\le\left\lfloor 3\sigma/\sqrt{2}\right\rfloor \quad i\ne j\\ 0 & ld_{ij}>\left\lfloor 3\sigma/\sqrt{2}\right\rfloor \quad i\ne j\\ 0 & i=j\end{cases} \tag{1}$$

Where $|v_i|$ is the number of tuples v_i contains. σ denotes a factor used to determine the influence sphere of nodes. ld_{ij} is the path length between v_i and v_j in G. Based on the properties of Gaussian formulae, if the ld_{ij} between nodes v_i and v_j is larger than $\left\lfloor 3\sigma/\sqrt{2}\right\rfloor$, the topology compactness between them is close unlimitedly to 0.

Note: If a path from v_i to v_j contains a physical table (i.e., a table that consists only of the primary keys of the participating tables) then we remove the additional relationship during the calculation processes of the logical distance. For example, let $|P|$ be the amount of physical tables on the path from v_i to v_j and fd_{ij} be the physical distance on the path, then $ld_{ij}=fd_{ij}-|P|$.

By the formula (1), the topology compactness matrix **C** is constructed.

$$\mathbf{C}=\begin{bmatrix}0 & c_{12} & \cdots & c_{1n}\\ c_{21} & 0 & \cdots & c_{2n}\\ \vdots & \vdots & \ddots & \vdots\\ c_{n1} & c_{n2} & \cdots & 0\end{bmatrix}$$

4.1.2 Content similarity

Having discussed the structural relationship between tables, we now turn to the discussion of the information such as tuples and attributes in tables which also affect the schema summarization.

The analysis needs to take into consideration both the names similarity and value similarity. Names similarity is one of the most critical influences. In particular, it consists of two parts, of which one is the table names similarity, and the other is the attribute names similarity. Vector Space Model is used to compute the names similarity [16]. First, a vector $\overrightarrow{N_i}$ for table T_i is constructed, the elements of vector $\overrightarrow{N_i}$ are table names and attribute names of table T_i. Then the names similarity is calculated by (2).

$$s_1\left(T_i, T_j\right) = Sim\left(\overrightarrow{N_i}, \overrightarrow{N_j}\right) = \overrightarrow{N_i}\cdot\overrightarrow{N_j}/\left(|\overrightarrow{N_i}|\cdot|\overrightarrow{N_j}|\right) \tag{2}$$

Next, we'll examine how content similarity is affected by the value similarity. The main steps are as follows:

1. Compute the content similarity between attribute-columns of two different tables with *Jaccard* function

$$J\left(T_i.A_m, T_j.A_n\right) = |T_i.A_m \cap T_j.A_n| / |T_i.A_m \cup T_j.A_n| \tag{3}$$

Where $T_i.\ A_m(T_j.\ A_n)$ is an arbitrary attribute-column of table $T_i(T_j)$.

2. Use a greedy strategy to test the matched attribute pairs [22]. Then assign different weights to attribute-column pairs according to $T_i.\ A_m$'s and $T_j.\ A_n$'s coefficients of variation (we defer a detailed discussion about the coefficient of variation to Section 4.2.1).

   a. Initialize the set Z to ϕ, $T_i.\ A(T_j.\ A)$ contains all attributes in $T_i(T_j)$.
   b. Look up and select the attribute pair $(T_i.\ A_m, T_j.\ A_n)$ with the maximum $J(T_i.\ A_m, T_j.\ A_n)$, $T_i.\ A_m \in T_i.\ A$ and $T_j.\ A_n \in T_j.\ A$.
   c. $Z \leftarrow Z \cup \{(T_i.\ A_m, T_j.\ A_n)\}$, then remove $T_i.\ A_m(T_j.\ A_n)$ from $T_i.\ A(T_j.\ A)$.
   d. Loop to the steps b and c until there are no more attribute pairs with *Jaccard* distance greater than 0.
3. Calculate the similarity using the method of weighted arithmetic mean.

$$s_2\left(T_i, T_j\right) = \begin{cases} \frac{1}{|Z|}\sum_{\left(T_i.A_m, T_j.A_n\right)\in Z} max\left\{T_i.A_m, T_j.A_n\right\}\cdot J\left(T_i.A_m, T_j.A_n\right) & i \neq j \\ 0 & i = j \end{cases} \tag{4}$$

$|Z|$ denotes the number of elements contained in the set Z, $T_i.\ A_m(T_j.\ A_n)$ is $T_i.\ A_m$'s ($T_j.\ A_n$'s) coefficients of variation.

$$s_{ij} = \alpha\cdot s_1\left(T_i, T_j\right) + (1-\alpha)\cdot s_2\left(T_i, T_j\right) \tag{5}$$

Motivated by the above analysis, the content similarity matrix **S** can be obtained:

$$\mathbf{S} = \begin{bmatrix} 0 & s_{12} & \cdots & s_{1n} \\ s_{21} & 0 & \cdots & s_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ s_{n1} & s_{n2} & \cdots & 0 \end{bmatrix}$$

Finally, we get the similarity matrix $\mathbf{A_D} = \beta\cdot\mathbf{C} + (1-\beta)\cdot\mathbf{S}$ by integrating the matrices **C** and **S**. α and β are balance factors to modulate the weights of influence factors. Here, we exploit the maximum entropy principles to assign them the same weight (1/2).

4.1.3 Adjustment of similarity matrix

The traditional approaches of schema summarization are exclusively focused on databases self-contained information. They categorize the tables with same topic into the same cluster, but do not take the query logs into consideration. Compared with traditional methods, we introduce the query logs into the process of schema summarization which can achieve a more meaningful and valuable result. We use the following boosting function (6) to adjust the result.

$$boost_{log}\left(T_i, T_j\right) = exp\left(log\left(count\left(T_i, T_j\right)\right)/log(max(count))\right) \tag{6}$$

where $count(T_i, T_j)$ makes statistics of tables T_i and T_j appearing in the same entry and $max(count)$ is its maximum.

According to the analyses above, we put forward the following strengthening function as shown in (7), if tables T_i and T_j appear in a log entry at the same time, the compactness score between them should be strengthened; otherwise, the compactness score will remain unchanged.

$$\mathbf{A}_{\mathbf{Final}} = \left[\mathbf{A}_{\mathbf{D}}\left(T_i, T_j\right)\cdot boost_{log}\left(T_i, T_j\right)\right]_{i,j=1}^{n} \tag{7}$$

4.2 Detection of cluster centers

To solve the problem that the number of clusters in the schema summarization results can not be predicted accurately, this section proposes an algorithm for detecting initial cluster centers.

4.2.1 Calculation method of table importance

During the process of detecting the initial cluster centers, the biggest challenge is to evaluate the importance of each table in a comprehensive and reasonable way. After calculating the importance of all the tables, we further design a detection schema to discover the most important tables in local areas as the initial center points of spectral clustering.

Since tables are in a complex topology, when calculating the importance of a table, we should consider not only its structure and content information, but also the impacts from other neighbor tables. Evidently, if the neighbor tables of T_i have high importance, the importance of T_i will be relatively high as well. In general, the table importance can be divided into two parts: the intrinsic importance $I_b(T_i)$ and the dependent importance $I_d(T_i)$ of the table, as shown in (8):

$$I_f(T_i) = I_b(T_i) + I_d(T_i) \tag{8}$$

The intrinsic importance $I_b(T_i)$ is closely related to the intrinsic properties of the table T_i, including the size of T_i, the information contained in T_i, the distribution of T_i in the database and the user feedback. The formula is shown as follows:

$$I_b(T_i) = \left(log|T_i| + \varphi(T_i) + \sum\nolimits_{T_i.A_m \in T_i.A} T_i.A_m\right)\cdot exp(tf(T_i)) \tag{9}$$

where $|T_i|$ is the number of tuples in table T_i and the importance of a table is proportional to its size. The topology potential $\varphi(T_i)$ of table T_i is defined as follows:

$$\varphi(T_i) = \begin{cases} \sum\limits_{j=1}^{n} |T_i|\cdot|T_j|\cdot exp\left(-\left(ld_{ij}/\sigma\right)^2\right) & ld_{ij} \le \left\lfloor 3\sigma/\sqrt{2}\right\rfloor \\ 0 & ld_{ij} > \left\lfloor 3\sigma/\sqrt{2}\right\rfloor \end{cases} \tag{10}$$

$T_i.A_m = S/\overline{A_m} \times 100\%$ is the coefficient of variation of attribute A_m in table T_i. It is a statistic which can be used to measure the extent of variation. Map the attribute values to a set of integers in ascending order. S is standard deviation and $\overline{A_m}$ is average of the mapped values.

$exp(tf(T_i))$ reflects the effect of user query on the table importance: $tf(T_i)$ is the number of occurrences of table T_i in query logs.

The latter part of (8) shows that the table importance is related to other neighbor tables:

$$I_d(T_i) = \sum_{j=1}^{n} \mathbf{A}_{\mathbf{Final}}\left(T_i, T_j\right)\cdot I_b\left(T_j\right) \tag{11}$$

Where $\mathbf{A}_{\mathbf{Final}}(T_i, T_j)$ is the similarity between the tables T_i and T_j. $I_b(T_j)$ is the intrinsic importance of table T_j.

Further normalization is carried out to get the final measure of table importance.

$$NI(T_i) = 2 \times \left(sigmoid\left(I_f(T_i)\right)-0.5\right) \tag{12}$$

Where $sigmoid(I_f(T_i)) = 1/(1 + exp\ (-I_f(T_i)))$

4.2.2 Detection algorithm

It is intuitively plausible that the important tables are more likely to be regarded as the initial cluster centers. Users can have a basic understanding about the clusters through these tables. However, it is

Algorithm 1 Initial Cluster Centers Detection Algorithm

```
Input: $G = (V, E)$;
Output: Set $R$;
Method:
Set $R \leftarrow \phi$, Set $I \leftarrow \phi$;
For $i = 1$ to $n$ do
    $I \leftarrow$ CalculateImportance($v_i$);
    $visited[v_i] = 0$;
EndFor
$Q.enqueue(DescSort(I))$;
$q_1 \leftarrow Q.denqueue()$;
$R \leftarrow q_1$, $visited[q_1] = 1$;
For each $q_i \in q_1.neighbors$ do
    $visited[q_i] = 1$;
EndFor
While(! $Q$ .isEmpty())
    $q \leftarrow Q.denqueue()$;
    For each $q_i \in q.neighbors$ do
        $visited[q_i] = 1$;
    EndFor
    If(! $visited[q]$) Then
        $R \leftarrow q$, $visited[q] = 1$;
    EndIf
EndWhile
Return $R$;
```

not reasonable to simply select the top-*k* important tables as the initial cluster centers. Because the above method is no longer applicable when two important tables are in the same cluster. In order to solve this problem, this paper proposes a schema to select the most important table in every local area as a cluster center [21]. The complete algorithm is presented as follows.

Step 1 initializes the sets *R* and *I*. In steps 2-5 of this algorithm, we calculate the importance of each node in the database schema graph according to (12), then in step 6 we sort the results in descending order and put them in the queue *Q*. In steps 7-8, the head element q_1 is dequeued from the queue *Q*, then we mark q_1 ‘visited’ and put it into the set *R*. Steps 9-11 mark q_1's neighbors ‘visited’. In steps 12-18, the current head element *q* is dequeued from the queue *Q* and its neighbors are marked. Determine whether *q* has been marked, if *q* has not been marked, we put it into the set *R* and mark it. Finally, the algorithm returns the set *R* as the output.

4.3 Schema summarization algorithm

After constructing the similarity matrix and detecting the initial cluster centers, we can summarize schema via spectral clustering. The detail of schema summarization is as shown in Algorithm 2.

Algorithm 2 Schema Summarization Algorithm based on Graph Partition

```
Input: $G = (V, E)$, where $V = \{v_1, \cdots, v_n\}$, $E = \{e_{v_i \to v_j} \mid v_i, v_j \in V\}$ and $\sigma$;
Output: $C = \{C_1, C_2, \cdots, C_k\}$;
Method:
1. Construct the similarity matrix $A_{Final}(v_i, v_j)$ via equation (7);
2. Calculate the importance of all the nodes with (12);
3. Detect all the local important nodes in $G$ by calling the Algorithm 1. Suppose we
   find $k$ local important nodes;
4. Calculate the first $k$ eigenvectors and eigenvalues of $A_{Final}(v_i, v_j)$;
5. Map each of the nodes in $V$ to $R^k$, and then the nodes in $R^k$ are divided into
   $C_1, C_2, \cdots, C_k$ by the $k$-means clustering.
```

In step 1, for every two tables in *G*, we need to calculate the similarity between them, and the corresponding time to construct the matrix $\mathbf{A}_{\mathbf{Final}}(v_i, v_j)$ is $O(n^2)$. The time complexity of step 2 is $O(n)$. Step 3 can be implemented based on the Algorithm 1, whose time complexity is $O(n)$. The overall time complexity for steps 4-5 is $O(n^3)$. In summary, the total time complexity of Algorithm 2 is $O(n^3)$, where *n* is the number of tables in the database schema graph *G*.

5 Experiments

We have systematically evaluated the schema summarization method GP-RDSS over TPC-E benchmark dataset [14]. We design and perform a series of experiments along different dimensions to verify the effectiveness of our method: First, the rationality and precision of the calculation formula for table importance in this paper are confirmed. Then we summary the database with GP-RDSS and examine the validity of the method by comparing with the predefined classification. Besides, we have validated that the

Table 2 Parameters of EGen for TPC-E

TPC-E	Number of Tables	Max (a)	Max (t)	Foreign keys
	33	24	162,137	45

detection scheme of initial cluster centers and user feedback play important roles in the schema summarization. At last, we compare our approach against the existing state-of-the-art approaches: Balance-sum [30] and the Weighted *k*-center [28]. The results strongly indicate that the method GP-RDSS achieve better effectiveness and the generated summaries are more helpful.

5.1 Data set and environment setup

TPC-E benchmark data set is provided by Transaction Processing Performance Council (TPC) to measure the performance of On Line Transaction Processing (OLTP) systems. It uses the data from the U.S. census and the New York Stock Exchange to generate people's names and company information. TPC-E consists of 33 tables that are grouped into four categories: Customer, Broker, Market, and Dimension. Category Customer is composed of customer-related information; Category Broker includes the data related to brokers; the data in Market is related to the exchanges, companies and securities; Dimension contains generic information. The TPC-E database contains 12 transactions which reflect the usage of the database, we use the transaction log as the query log for TPC-E. Moreover, we generate the TPC-E database using EGen_v3.14 which is a software package provided by TPC, with parameters as follows: Number of Customers = 1000, Scale factor = 1000, Initial trade days = 30. The characteristics of the database are shown in Table 2, where Max (a) is the maximum number of attributes in tables, Max (t) is the maximum number of tuples in tables.

We choose the TPC-E as our data set for three main reasons: First, it contains both data and schema information, which is complex enough for summaries. Second, the schema of TPC-E is strictly relational and it has the specified categories, which facilitates the evaluation of results. Last, it has transaction log, which can be used as query log in the experiments.

Our methods are all implemented in Java, and deployed on a PC with a 3.40GHz Intel®Core CPU and 4GB memory.

Table 3 Top-6 important tables based on I_s, I_e and I_f

Rank	$I_{s.}$Table	$I_{e.}$Table	$I_{f.}$Table
1	Trade	Trade	Trade
2	Trade_History	Security	Customer
3	Status_Type	Customer	Security
4	Daily_Market	Financial	Company
5	Holding_History	Holding	Financial
6	Customer	Customer_Account	Holding

Table 4 The P@6 of three methods: I_s, I_e and I_f

Method	I_s	I_e	I_f
P@6	50%	67%	83%

5.2 Experimental evaluation

5.2.1 Evaluation of table importance

We now compare our calculation method of table importance with the ones in [28, 30]. The I_f. Table rank, I_s. Table rank, and $I_{e.}$ Table rank are used to denote the ranking results obtained by above methods. Due to space limitations, we just show the Top-6 results in Table 3.

From the above comparison results, we note that, the importance calculation method I_s proposed in [30] measures the table importance mainly according to the size of the table. As listed in Table 3, tables Trade_History and Holding_History are ranked second and fifth respectively because of their larger scales. However, both tables are not very important and rarely get the attention of users during the transactional process. So the method I_s in [30] lacks rationality.

By contrast, the proposed method takes advantage of the topology relationship, content similarity and user feedback comprehensively, and gives a reasonable ranking result. For example, tables Trade and Customer are ranked in the top two because of their rich content, high topology potential and high frequency appeared in a query log. The result is what most humans want to get.

Above are the intuitive comparison and analysis of importance calculation methods. To further quantitatively compare these three methods, we then present a metric P@N for evaluation of the quality of calculation methods. P@N is the precision at the position N. We use the pre-defined table importance rank by human experts to define objective measures. The obtained experimental results are depicted in Table 4.

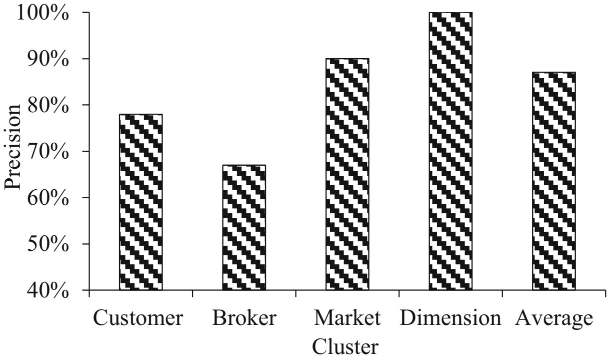


Fig. 4 Precision of GP-RDSS

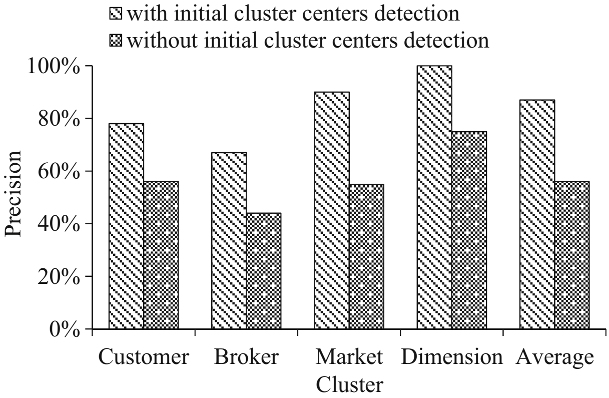


Fig. 5 Precision of GP-RDSS with/without cluster representatives detection

From the experimental results, it can be seen that the P@6 of method Balance-sum in [30] is obviously lower than the others. The major reason is that the factors it considers are seemingly too simplistic when calculating the table importance. Furthermore, we observe that the P@6 of GP-RDSS (83%) is little higher than that of Weighted *k*-center in [28] (67%). This is due to the fact that GP-RDSS further considers the effect of query logs on the table importance. Of course, the importance calculation method is not taken as the core in this paper. We aim at improving the quality of the schema summarization results, which will be demonstrated in more detail subsequently.

5.2.2 Effectiveness

In this section, the clustering results obtained by our method are compared with the summaries that have been pre-defined. In Fig. 4, the clusters of database are taken as *x*-axis and the clustering precision as *y*-axis. As we can see from Fig. 4, the precision of the schema

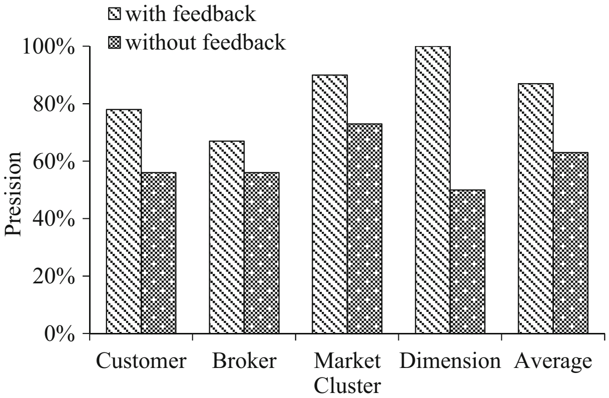


Fig. 6 Precision of GP-RDSS with/without user feedback

summarization generated by our approach varies for the different clusters, and the average precision is around 84%. We will perform a comprehensive analysis in the reminder of experiments.

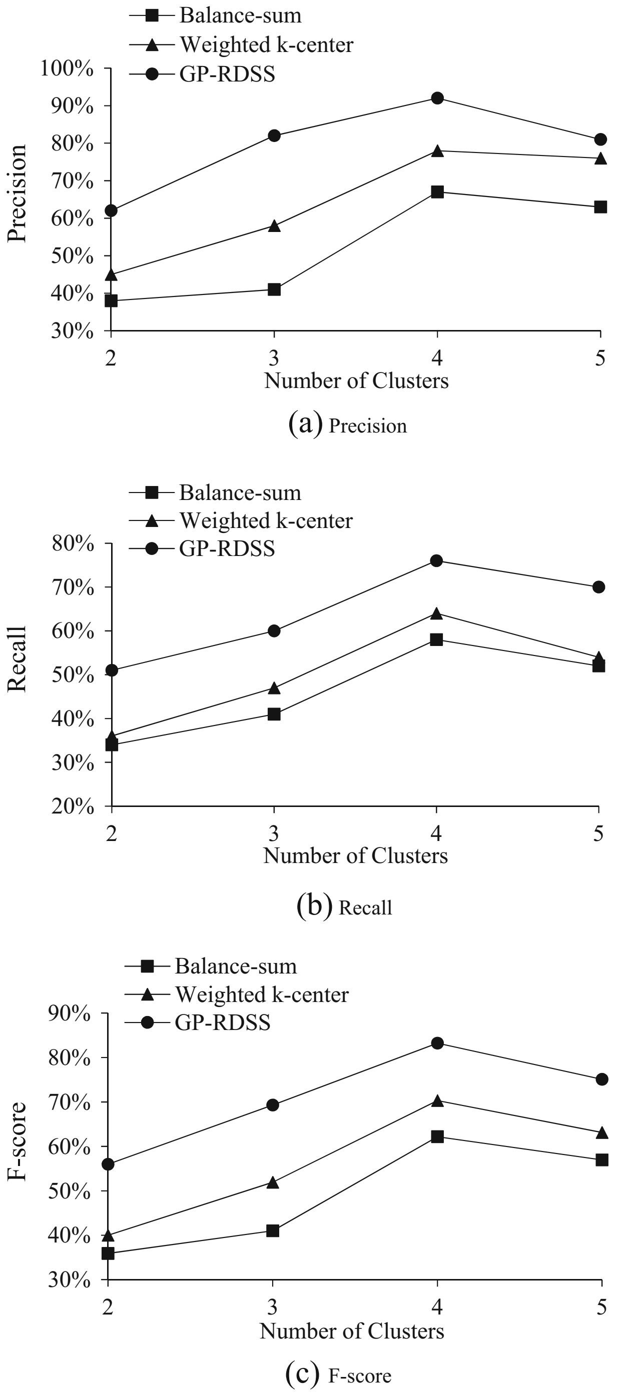
In addition, this paper puts forward a detection theme of initial cluster centers in Section 4.2. In order to verify its significant effects on improving the clustering precision, we conduct the following experiments. Fig. 5 illustrates the precision of results with and without the initial cluster centers detection. As expected, the average precision of GP-RDSS with cluster centers detection achieves 84.4% precision, which is about 47.6% higher than the other one. Overall, the approach with cluster centers detection achieves better summary performance, which gives us rich confidence that the cluster centers detection can improve the summarization efficiency. This is due to the fact that the reasonable detection of the cluster centers can dramatically improve the precision of the spectral clustering algorithm, and then the precision of the summarization results can be improved accordingly. In other words, the initial cluster centers detection enables the proposed method to group more data tables into correct categories.

Another important feature of our method is the introduction of the user feedback. In this experiment, we study how the quality of the summary is impacted by it. The results of the experiments are reported in Fig. 6. The x-axis represents the cluster, and the y-axis represents the precision of GP-RDSS. As we can see, the approach with feedback exhibits a much higher precision than the one without feedback. Specifically, for the cluster Market, the precision of GP-RDSS with feedback reaches 90.0%, which is nearly 23.3% higher than the other method. Clearly, GP-RDSS with feedback performs better. This can be attributed to the fact the measure of the similarity between two tables is more accurate, when taking the feedback into account. The similarity matrix is an important input of Algorithm 2 (Schema Summarization Algorithm), which has a direct impact on the output of the algorithm. From the above observations, we conclude that feedback plays a very important role in affecting the performance of GP-RDSS.

5.2.3 Experimental Comparision

In this section, we report the results quality of our method GP-RDSS in comparison with two most recent proposals on schema summarization: Balance-sum [30] and Weighted k-center [28]. The performance of these approaches is measured by the following standard metrics: $P = TP/(TP + FP)$, $R = TP/(TP + FN)$, and $F = 2PR/(R + P)$ [6], where TP represents the number of the tables which are similar and assigned to the same cluster. FP represents the number of the tables which are dissimilar to the same cluster. FN represents the number of the tables which are similar and assigned to different clusters. Fig. 7 demonstrates that GP-RDSS outperforms the current state-of-the-art technique on all aspects, including precision, recall, and F-score.

Figure 7a plots the precision of the summarization methods above. We notice that higher precision tends to be achieved when the number of clusters is larger. When the number of clusters is equal to 4, the precision reaches the maximum. When the number



(a) Precision

(b) Recall

(c) F-score

Fig. 7 Comparison of three schema summarization methods **a** Precision; **b** Recall; **c** F-score

of clusters is equal to 5, the precision starts to decrease. This is because there are only 4 categories in the TPC-E dataset. We also observe here that GP-RDSS always achieves more than 62.0% precision, which leads to about 37.8-63.2% higher than the other two approaches. For example, when the number of clusters is 4, the precision of GP-RDSS is approximately 17.9% higher than that of the Weighted *k*-center approach, and it is even higher when compared with the other approach Balance-sum. This is so because GP-RDSS integrates the information from many aspects to construct the similarity matrix, which can more accurately measure the similarity between the two tables, and thus make the clustering result more accurate.

To further evaluate the performance of our algorithms, we vary the number of clusters and evaluate the corresponding recall. Figure 7b illustrates the experimental results. For each number of clusters, our GP-RDSS approach obtains the maximum of recall. More precisely, when the number of clusters is 4, the recall of GP-RDSS is 76%, while the ones of Weighted *k*-center and Balance-sum are 64 and 58%. Therefore, GP-RDSS performs quite well compared with other two algorithms. The major reason of achieving high recalls is that the proposed method can discover the tables with indirect correlation by capturing the information in query logs, which can help to identify a more complete category and thus improve the recall. This comparison shows the significance of our proposed summarization method and reflects it is very effective and practicable.

Figure 7c shows the F-score achieved by various summarization algorithms. Due to space constraints, it won't be covered here. All these strongly indicate the effectiveness of GP-RDSS, because it synthesizes the structures, contents, and user feedback at the same time.

6 Conclusion and future work

This paper introduces a novel schema summarization method GP-RDSS based on graph partition. According to the best of our knowledge, this is the first attempt to summarize the database schema combining graph partition mechanism with user feedback. We construct the input matrix in spectral clustering by calculating topology compactness and content similarity in the databases. Besides, we use the statistical analysis of the information in query logs to modify the above matrix. The enhanced matrix can reflect the characteristics of the user preferences. Finally, we develop a comprehensive formula for calculating the table importance and detect the most important nodes in the local areas as the initial cluster centers. The final results can help users to understand and access the database quickly. We have implemented our approach on real data set TPC-E benchmark, and the results show that GP-RDSS achieves good quality to summarize relational schemas.

In the future work, we plan to apply our summarization method in the practical applications. We believe that better performance can be achieved in several domains like biological problems (e.g., protein interaction networks), Web application, e-business, social media (e.g., Twitter), Multimedia application, Multimedia databases and retrieval, etc. Because the basic idea of our method is a graph partition mechanism, which is very generic; while above information from different sources available to a user can also be naturally represented as graphs. In conclusion, the summarization method is useful whenever users wish to get a quick overview of a complex data set.

Acknowledgements This work is sponsored by the National Natural Science Foundation of China under Grant No. 61772152 and 61502037, and the Basic Research Project (No. JCKY2016206B001, JCKY2014206C002 and JCKY2017604C010).

References

1. Alborzi F, Chirkova R, Doyle J, Fathi Y (2015) Determining query readiness for structured data. In: 17th International Conference on Big Data Analytics and Knowledge Discovery, Valencia, Spain, 2015. pp 3-14
2. Beneventano D, Guerra F, Velegrakis Y (2017) Data exploration on large amount of relational data through keyword queries. In: 15th International Conference on High Performance Computing and Simulation, Genoa, Italy, 2017. pp 70-73
3. Bergamaschi S, Guerra F, Simonini G (2014) Keyword search over relational databases: Issues, approaches and open challenges. In: 2013 PROMISE Winter School: Bridging Between Information Retrieval and Databases, Bressanone, Italy, 2013. pp 54-73
4. Bergamaschi S, Ferrari D, Guerra F, Simonini G, Velegrakis Y (2016) Providing insight into data source topics. Journal on Data Semantics 5(4):211–228
5. Carlsson G (2009) Topology and data. Bull Am Math Soc 46(2):255–308
6. Dimitroff G, Georgiev G, Toloi L, Popov B (2014) Efficient F measure maximization via weighted maximum likelihood. Mach Learn 98(3):435–454
7. Kahng M, Navathe SB, Stasko JT, Chau DH (2016, 2016) Interactive browsing and navigation in relational databases. In: 42nd international conference on very large data bases. New Delhi, India:1017–1028
8. Kargar M, An A, Cercone N, Godfrey P, Szlichta J, Yu X (2015) Meaningful keyword search in relational databases with large and complex schema. In: 31st IEEE International Conference on Data Engineering, Seoul, Korea, 2015. pp 411-422
9. Kruse S, Hahn D, Walter M, Naumann F (2017) Metacrate: Organize and analyze millions of data profiles. In: 26th ACM International Conference on Information and Knowledge Management, Singapore, Singapore, 2017. pp 2483-2486
10. Liu D, Liu G, Zhao W, Hou Y (2017) Top-k keyword search with recursive semantics in relational databases. Int J Comput Sci Eng 14(4):359–369
11. Luo Y, Lin X, Wang W, Zhou X (2007) Spark: top-k keyword query in relational databases. In: SIGMOD 2007: ACM SIGMOD International Conference on Management of Data, Beijing, China, 2007. pp 115-126
12. Sampaio M, Quesado J, Barros S (2013) Relational schema summarization: A context-oriented approach. In: 16th East-European Conference on Advances in Databases and Information Systems, Poznan, Poland, 2013. pp 217-228
13. Taheriyan M, Knoblock CA, Szekely P, Ambite JL (2016) Learning the semantics of structured data sources. Journal of Web Semantics 37-38:152–169
14. TPCE. http://www.tpc.org/tpce/default.asp#top
15. Troullinou G, Kondylakis H, Daskalaki E, Plexousakis D (2015) RDF digest: Efficient summarization of RDF/S KBs. In: 12th European Semantic Web Conference, Portoroz, Slovenia, 2015. pp 119-134
16. Turney PD, Pantel P (2010) From frequency to meaning: vector space models of semantics. J Artif Intell Res 37(4):141–188
17. Van Gennip Y, Hunter B, Ahn R, Elliott P, Luh K, Halvorson M, Reid S, Valasik M, Wo J, Tita GE, Bertozzi AL, Brantingham PJ (2013) Community detection using spectral clustering on sparse geosocial data. SIAM J Appl Math 73(1):67–83
18. Wang N, Tian T (2016) Summarizing personal dataspace based on user interests. Int J Software Engineer Knowledge Engineer 26(5):691–713

19. Wang X, Zhou X, Wang S (2012) Summarizing large-scale database schema using community detection. J Comput Sci Technol 27(3):515–526
20. Wang X, Qian B, Davidson I (2014) On constrained spectral clustering and its applications. Data Min Knowl Disc 28(1):1–30
21. Wang Z, Chen Z, Zhao Y, Niu Q (2014) A novel local maximum potential point search algorithm for topology potential field. International Journal of Hybrid Information Technology 7(2):1–8
22. Wu W, Reinwald B, Sismanis Y, Manjrekar R (2008) Discovering topical structures of databases. In: 2008 ACM SIGMOD International Conference on Management of Data 2008, Vancouver, Canada, 2008. pp 1019-1030
23. Yan C, Zhang Y, Xu J, Dai F, Li L, Dai Q, Wu F (2014) A highly parallel framework for HEVC coding unit partitioning tree decision on many-core processors. IEEE Signal Process Lett 21(5):573–576
24. Yan C, Zhang Y, Xu J, Dai F, Zhang J, Dai Q, Wu F (2014) Efficient parallel framework for HEVC motion estimation on many-core processors. IEEE Trans Circuits Syst Video Technol 24(12):2077–2089
25. Yan N, Hasani S, Asudeh A, Li C (2016) Generating preview tables for entity graphs. In: 2016 ACM SIGMOD International Conference on Management of Data, San Francisco, United states, 2016. pp 1797-1811
26. Yan C, Xie H, Liu S, Yin J, Zhang Y, Dai Q (2018) Effective Uyghur language text detection in complex background images for traffic prompt identification. IEEE Trans Intell Transp Syst 19(1):220–229
27. Yan C, Xie H, Yang D, Yin J, Zhang Y, Dai Q (2018) Supervised hash coding with deep neural network for environment perception of intelligent vehicles. IEEE Trans Intell Transp Syst 19(1):284–295
28. Yang X, Procopiuc CM, Srivastava D (2009) Summarizing relational databases. Proceedings of the VLDB Endowment 2(1):634–645
29. Yang X, Procopiuc CM, Srivastava D (2011) Summary graphs for relational database schemas. Proceedings of the VLDB Endowment 4(11):899–910
30. Yu C, Jagadish HV (2006) Schema summarization. In: 32nd International Conference on Very Large Data Bases, Seoul, Korea, 2006. pp 319-330
31. Yuan X, Li X, Yu M, Cai X, Zhang Y, Wen Y (2014) Summarizing Relational Database Schema Based on Label Propagation. In: 16th Asia-Pacific Web Conference on Web Technologies and Applications, Changsha, China, 2014. pp 258-269

Yingqi Wang She is currently pursuing the Ph.D. degree in computer science and technology, Harbin Engineering University, Harbin, China. Her interests include: keyword query, dataspace and data mining.

Lianke Zhou received the PhD degree in computer science and technology from Harbin Institute of Technology, Harbin, China, in 2011. He is member of CCF. His interests include: Data Visualization, Dataspace, Distributed Computing.

Nianbin Wang received the PhD degree in computer science and technology from Harbin Institute of Technology, Harbin, China, in 2001. He is member of CCF, he has been a Professor with the Department of Computer Science and Technology, Harbin Engineering University. His interests include: Dataspace, Deep Learning, Data Integration.