KSII TRANSACTIONS ON INTERNET AND INFORMATION SYSTEMS VOL. 13, NO. 12, Dec. 2019 5888

ValueRank: Keyword Search of Object Summaries Considering Values

Cai Zhi[1], Lan Xu[1], Su Xing[1*], Lang Kun[2], Cao, Yang[1]
[1]Department of Computer Science, Beijing University
of Technology,100124 - China
[e-mails: caiz,lanxury,xingsu,caoyang@bjut.edu.cn]
[2]China Three Gorges International Corporation, 100008 – China
[e-mails: lang_kun@ctg.com.cn]
*Corresponding author: Su Xing

Received March 30, 2018; revised April 27, 2019; accepted May 26, 2019;
published December 31, 2019

Abstract

Abstract: The Relational ranking method applies authority-based ranking in relational dataset that can be modeled as graphs considering also their tuples' values. Authority directions from tuples that contain the given keywords and transfer to their corresponding neighboring nodes in accordance with their values and semantic connections. From our previous work, ObjectRank extends to ValueRank that also takes into account the value of tuples in authority transfer flows. In a maked difference from ObjectRank, which only considers authority flows through relationships, it is only valid in the bibliographic databases e.g. DBLP dataset, ValueRank facilitates the estimation of importance for any databases, e.g. trading databases, etc. A relational keyword search paradigm Object Summary (denote as OS) is proposed recently, given a set of keywords, a group of Object Summaries as its query result. An OS is a multilevel-tree data structure, in which node (namely the tuple with keywords) is OS's root node, and the surrounding nodes are the summary of all data on the graph. But, some of these trees have a very large in total number of tuples, size-*l* OSs are the OS snippets, have also been investigated using ValueRank.We evaluated the real bibliographical dataset and Microsoft business databases to verify of our proposed approach.

Keywords: Relational databases, Keyword Search, Object Summary, Rankings

This work is supported by the Beijing Natural Science Foundation (No. 4172004), the National Natural Science of Foundation of China (No. 61703013), Key Project of Beijing Municipal Education Commission (No. KM201810005023, KM201810005024), The Construction Project for National Engineering Laboratory for Industrial Big-data Application Technology (No. 312000522303).

http://doi.org/10.3837/tiis.2019.12.006 **ISSN : 1976-7277**

1. Introduction

Keyword Search is very remarkable because users just need to use only one set of keywords to get useful information from the web (such as Google). It shows that the result of W-KwS is the ranked set which considering the importance of each tuple contains keywords in this set. Follow the result, there have some interesting discoveries. (1)Each result from a query is accompanied by a snippet[1,17,18], which is a brief summary, which sometimes may be included in the complete result. (2) The corresponding web pages with the keyword(s) (e.g. their personal web pages) can potentially provide meaningful and ample information about the designated subject.

The use of keyword search paradigms in relational databases is due to the favorable outcome of the W-KWS paradigm [2,3,4,20]. The relevant ranking paradigm takes into account the importance of which weights the flow through relationships. A great tool, Pagerank[4] can rank the global importance of web pages, which proves Google's success. Keyword search in the database has its own unique characteristics, making the Pagerank model invalid. That is to say, each database infers that the semantics of different attributes are different and characteristic. In the database data graph, different attributes are represented by different relationships and attributes' values, which is different from the Web where all edges are hyperlinked. ObjectRank[3] has some appropriate extensions and modifications to PageRank. For instance, in a bibliographic database (e.g. DBLP), under normal circumstances, an author with many citations is more important than another author with fewer citations.

However, ObjectRank ignores tuples' attribute worth, which can affect the global importance. For example, the value of a customer with a high total purchase price should be higher than others same as these number of orders but lower overall purchase price. Respond to this limitation, Given this limitation, ValueRank is proposed in this paper, which can also consider values. Similarly, for ObjectRank, you can use patterns and specify the way to flow permissions across database graph nodes, which consider tuple values.

Because the methods of PageRank, ObjectRank [7] and other techniques can only be used bibliographic database., it is a challenging problem for sorting database tuples and estimating tuples' global importance scores (represented as $Im(t_i)$). Therefore, ValueRank is introduced in this paper that also takes account of tuples value and so that it can be used to any class of database. In [6], ValueRank has been introduced with only trading databases (such as Northwind database) and has no evaluation results. In this paper, a new definition of ValueRank is defined and evaluation results verify this ValueRank produces more excellently or effectively ranking results than ObjectRank on general databases, such as DBLP databases.

From our previous work, the new keyword search paradigm proposed by [5,19] brings a concept of OSs, where all tuples from dataset about particular subject. More precisely, as we described in abstract, an OS is a multilevel-tree data structure, whose root is a tuple including keywords (e.g. Author tuple "Peter Chen", denoted t^{DS}) and the descendant nodes[5] are its connecting (i.e. Neighboring) itmes (containing other additional semantic meaning such as his papers, year of publication, etc.). But we find that some OSs' size may be very large, which is not only unfriendly to users because they want to glance at the moment and find out which "Faloutsos" they are really want to browse, but also the production cost is also high. Evidently, the effective and efficient size-l generation of OSs is necessary[6]. The exact concept of object summary is described in the following section.

We highlight our contribution of this paper as follows: (1) the introduction of a new ranking method, which extends our previous work ValueRank[6] algorithm to a widely usage not only in commercial or trading domain, but also in all numerical or normalizable datasets. (2) based on the concept of Object Summary, we also propose a novel greedy algorithm (namely *k*-LASP) for the size-*l* generation of OSs.

The following is the rest of the structure of this paper. Section 2 presents the background and related work of this paper. Section 3 introduces ValueRank. Section 4 provides a greedy algorithm *k*-LASP. Whereas Section 5 provides our evaluation results. Finally, Section 6 conclusions and future work are discussed in it.

2. Related work and research background

2.1 Object Summary

In the research filed of the new keyword search, a keyword query is a set of keywords[5,19]. In other words, the result of the query is a set of OSs. It should combine graphs and SQL to construct OSs. The fundamental principle is based on the fact that the relations, which includes information about DSs and the relations linked around R^{DS}s contain additional information about the particular DS. For each R^{DS}, a Data Subject Schema Graph (G^{DS}) is generated automatically, this is a directed labeled tree that finds a subset of the database schema with R^{DS} which is the root. (**Fig. 1** illustrates the schemata of DBLP and **Fig. 2** illustrates respective G^{DS}s from DBLP databases). The G^{DS} is a "Treealization" of the schema, examples of such replications are the relationships Paper (Cited by), Paper (Cites) and Co-Author on Author G^{DS} (see G^{DS}s in **Fig. 2**). In G^{DS}, affinity measures of relations (denoted $Af(R_i)$) are investigated, quantified and annotated, aiming to create a good OS, it's difficult to select the relations from G^{DS} which have the highest Affinity with the R^{DS} that need to be traversed[12]. The Affinity of a relation R_i to R^{DS} can be calculated with the following formula:

$$Affiniy(R_i) = \sum_j m_j \cdot w_j \cdot Affinity(R_{parent}) \tag{1}$$

where j denotes the ranges of metrics ($m_1, m_2, ..., m_n$), with weights ($w_1, w_2, ..., w_n$) respectively, $Affintiy(R_{Parent})$ (≤ 1) is the Affinity of the R_i's parent to R^{DS}. Affinity metrics between R_i to R^{DS} include (1) their distance and (2) their connection properties on database schema and data-graph(see [5] for details). Provided an Affinity threshold θ, we can get a subset of G^{DS} denoted as $G^{DS}(\theta)$. Finally, we can generate the object summaries by traversing the graph $G^{DS}(\theta)$. More precisely, it can user a BFS search for the corresponding $G^{DS}(\theta)$, its initial root is the t^{DS} tuple of the OS tree [5].

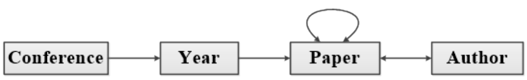


Fig. 1. The DBLP Database Schema

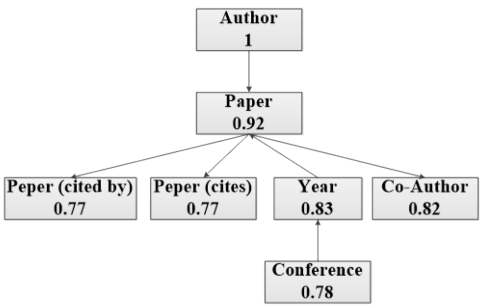


Fig. 2. The DBLP Author(Annotated with Affinity)

In order to weaken the contribution of each tuple's global importance, the score of local importance for each tuple t_i in an object summary (namely *Im*(OS, t_i)) can be generated from the formula:

$$\mathrm{Im}(OS, t_i) = \mathrm{Im}(t_i) \cdot Affnity(t_i) \tag{2}$$

Im(t_i) denoted as the score of global importance of t_i in the database. The global importance was calculated by ValueRank which is an importance ranking system (see section 3). Other tuples' importance ranking systems can be investigated such as [7,9,10] etc. Note that IR-style techniques [13,14,15,16] are completely inappropriate for ranking OS tuples, because they miss important tuples without the keyword(s). We mentioned that an object summary usually contains the given keywords only once (i.e. t^{DS}), therefore IR-style techniques can't rank the remaining tuples of the OS effectively. So, it can use Formula 1 to calculate the Affinity (alternatively, *Affinity*(R_i)s can be manually set by domain experts) and then use Formula 2 to calculate the local importance. For example, consider tuple t_i is the paper named "Efficient and Effective Querying by Image Content" with *Im*(t_i)=21.74 and *Af*(t_i)=*Affinity*(R_{Paper})=0.92 (see Affinity scores annotated on Author G^{DS} of **Fig. 2**) then *Im*(OS, t_i) = 21.74 × 0.92=20.

Distinguishing tuples with different Affinity relation score is considered crucial. For example, comparing Paper tuple "Efficient..." with the global importance score 21.74 and Year tuple "1988" with the global importance score 21.64 (i.e. almost equal scores), their local importance becomes 20 (calculate by 21.74 × 0.92) and 18 (calculate by 21.64 × 0.83) respectively. It is also recalled that due to the threshold θ, while tuples with less affinity relation scores may not be selected into an object summary.

2.2 size-l Object Summary

According to [11], a size-*l* object summary is a set of *l* nodes, i.e. given an complete object summary and an integer *l*, a candidate size-*l* object summary is any subset of the object summary consisting of *l* nodes (tuples are connected, while rooted at the tuple that including the given keywords). The result of size-*l* object summary meets the following two criteria. (1) All *l* tuples are connected with the t^{DS} of the multilevel-tree and (2) the importance scores *Im*(OS, size-*l*) is maximum, namely max($\sum$*Im*(OS, t_i)). The first criterion is to ensure that it can include self-descriptive semantics of keywords in the size-*l* object summary. Authos of [11] argued that an appropriately size-*l* object summary should be an independent, meaningful introduction to the most important node of a particular data subject, and it is easy for users to understand it without any redundant information. Thus, connecting nodes with n^{DS} constraint guarantees that the size-*l* remains independent. For instance, consider the path R_{Author}→R_{Paper}→$R_{(Co\text{-})Author}$ (in DBLP database), even if a paper's local importance is not as high as the co-author, then it cannot only choose the co-author and exclude the paper. It is rational to exclude the semantic association by excluding the paper tuple between the authors which are the co-authors of this paper in this case.

Also, note that because of criterion (1)The *l* tuples with the top importance scores will not be included in a size-*l* object summary. E.g., we consider the path R_{Author}→R_{Paper} →R_{Year}→$R_{Conference}$ with corresponding tuples with scores 0.9, 0.2, 0.7, 0.6, then, the Conference tuple, although it has bigger importance score than that of Paper, may be excluded from the size-*l* object summary whilst Paper may remain. Also, the *Im*(OS, size-*l*) does not represent the maximum importance of *l* tuples but the maximum summation of the *l* connected to t^{DS} tuple.

2.3 Rest of the Related Work

Recently, documentation summarizing techniques have aroused extensive research interest [1,18]. Web fragment is an example of a document summary, a search result used by web based keyword search for quick preview. They can be static (for example, consisting of the first few words of a document or descriptive metadata) or query biased (for example, consisting of sentences containing multiple keywords) [18]. Applying these technologies directly to the database, especially the OS, are still ineffective (e.g. relational associations and semantics of displayed tuples will be ignored). For example, papers authored by Chen (although the keyword “Chen” is not be included) importance is similar to their authors and citations, and this is ignored by the document summarization. On the other hand, general idea is the entity summarization in the semantic knowledge graph, it is similar to ours.More accurate concepts are given in [21], If a semantic knowledge graph and an entity represented by a node q graph, then the summary of q is a subset of the size l graph, where nodes surround the node q.[21].

RELIN [22] is another related research, which uses random walks on a graph to describe entity’s features.Different from such document summarization studies or existing works, our proposed Object Summary generation approach is for each standalone data subject, we use tuples to further explaining and supporting tuples that including the querying keywords, in order to distinguish each other from the results, while its relational tuple ranking methodology is an authority-transfer based approach considering their corresponding ‘values’ in relational datasets, specially for keyword search in relational databases.

A similar approach that using OSs to search semantics in web was proposed in [23] namely information unit.That is, the result of web keyword search is a document consisting of a group of linked web pages containing all the keywords, rather than a physical document. The Sphere Search proposed in [24] is a keyword search for heterogeneous data in semi-structured, none-structured, and structured data. These works are searching for associations of nodes that contain the keywords to adopt and provide the semantics of relational keyword search. Moreover, ranking algorithms, keyword search, and value based analysis etc techniques have been widely studied in cloud computing [25], fog computing[26], dig data etc. approaches.

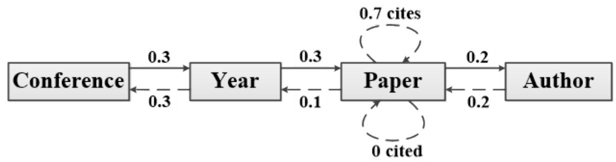
3. ValueRank

PageRank-style (such as ObjectRank[8]) are considered the most effective approaches for databases with relationship edges associating with authority flow semantics. But for trading databases, like Northwind, PageRank-style gives more references to important nodes (i.e. with high score), but ignores tuples’ values.For examples, there are two customers that are namely C_1 (has 100 orders) and C_2 (has 5 orders), but if C_2’s orders have high total order price, C_2 may be more important than C_1. As a result, it can observe that in such a database, it has to rank OSs according to some of its tuples’ values. This paper proposes and investigates a more versatile solution, namely ValueRank that can be applicable to any databases.

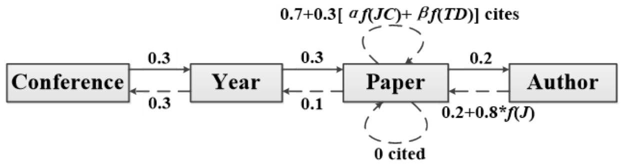
The nature of ValueRank is based on the concept of ObjectRank[8], when calculating the authority transfer rate, basic set ect, also take values into account, where the Basic Set and Authority Transfer Rate consider not only the number of tuples’ link or linked but also values. The Basic Set is the set of tuples, where the values of these nodes are deemed to have a significant impact on the authority of other nodes. E.g. All tuples in Paper and Year have influence on other tuples in DBLP database. Furthermore, the Authority-based Rate from Paper to Year and so on can be taken as a functional relationship of these values (normalised).

For example, consider Paper P_1 (published in 2016) with one reference which is published in 1996 and P_2 (published in 2017) with one reference which is published in 2016. The Authority Transfer Rate between Papers and Years can be a function of these values, therefore according to this function, it can be calculated that P_2 would obtain higher ValueRank.

More exactly, the dataset is modeled as a data-graph whilst the Schema Graph describes its schema structure. The corresponding Authority-based Transfer Schema is be created from $G(V_G, E_G)$, it affects the authority-based flow through the edges of the graph (e.g. **Fig. 3**). Further, for each edge $e_G = (v_i \rightarrow v_j)$ of the E_G, two Authority Transfer Edge can be created, that are D (the Data Graph) and G^A (defined from Authority-based Transfer Schema Graph), $D^A(V_D, E_D{}^A)$ is the corresponding Authority-based Transfer Data Graph could be derived as: for each edge of the E_D, the D^A has two edges, i.e. in edge and outgoing edge, respectively $e^f = (v_i \rightarrow v_j)$, $e^b = (v_j \rightarrow v_i)$ which are represented by the Authority-based Transfer Rates $a(e^f)$ and $a(e^b)$ correspondingly, where $a(e^f) = \mathbf{a}(e_G^f)/OutDeg(u, e_G^f)$ if $OutDeg(u, e_G^f) > 0$ ($OutDeg(u, e_G^f)$ is the total number of outgoing edges from u of type e_G^f) or $a(e^f) = 0$ otherwise ($a(e^b)$ is defined accordingly).



(a) The G^A for the DBLP database(ObjectRank)



(b) The G^A for the DBLP database(ValueRank)

Fig. 3. The G^As for the DBLP database

Instead of using the whole V_D, it can use any subset S of nodes as the Base Set, which can increase the authority associated with them.S is a subset of the tuple containing the keywords.

A node v_i 's value s_i describe the relative score of a node, and s_i can be calculated by a function with the normalized attributes values of v_i. The s_i of a node v_i in S can be defined with the equation:

$$\mathbf{s}_i = \boldsymbol{a} \cdot \boldsymbol{f}(\boldsymbol{v}_i) \tag{3}$$

where a is a tuning constant and $a \leq 1$. $function(v_i)$ is a normalizing function of the value of v_i and $0 \leq f(v_i) \leq 1$. S_i is in the range [0, 1] rather than just 0 or 1 as in ObjectRank. For example, for a tuple v_i in $R_{OrderDetails}$, $s_i = function$(OrderDetails.Price * OrderDetails.Quantity). s_i may be a function of the attributes of neighbouring nodes. For instance, for a tuple of Orders, $s_i = function(\sum OrderDetails.Price * OrderDetails.Quantity)$. It has more dynamic transfer rates if v_i's values combine with Authority-based Transfer Edges. The intuition is that a tuple 's different restriction values may an impact on its different edges. The Authority Transfer Edges can be denoted as $a(e)$' whether forward or backward , $a(e)$' can be calculated by the following formula:

$$\boldsymbol{a}(\boldsymbol{e})' = \boldsymbol{\beta} + \boldsymbol{\gamma} \times \boldsymbol{f}(\boldsymbol{v}_i \rightarrow \boldsymbol{v}_j) \tag{4}$$

where β and γ are tuning constants, so $\beta + \gamma \leq 1$, $f(v_i \rightarrow v_j)$ is a normalizing function of v_i and v_j and its values is in the range [0, 1] . **Fig. 4** illustrates the *graph* for the Microsoft Northwind database. Similarly to ObjectRank calculations, the Authority-based Transfer Rates, Basic Set S and tuning constants are experimented as variables.

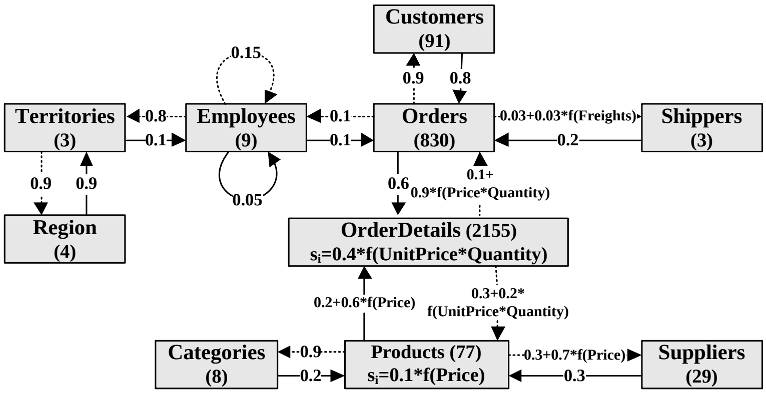


Fig. 4. The *graph* for the Microsoft Northwind dataset

Basic Set S including nodes, while whose *Jaccard coefficient* (Jc) are considering to have significant affect on the their connecting tuples authorities. E.g., in the DBLP dataset, the corresponding Authority Transfer Schema Gragh G^A(i.e. **Fig. 3**) is created. For the *Paper* → *Paper*, a paper that is cited by important paper and their Jaccard coefficient (Jc) is high, then it will be clearly important. For an author, the papers of his main areas should obtain higher ValueRank. If A_1 has three papers named P_1, P_2, P_3 respectively, P_1 cites P_2 and P_3, then get Jaccard coefficients of P_1 with P_2 and P_1 with P_3, P_1 with P_2, has higher Jaccard coefficient than P_1 with P_3(namely s_1 and s_2), so it can obtain the Jaccard value of $A_1 \rightarrow P_1$ is s_1. The Jaccard value $J(v_{i \rightarrow} v_j)$ of the Authority Transfer Edge can be calculated:

$$J(v_i \rightarrow v_j) = \max[(A-e)(j,:)] \tag{5}$$

in which, A is an $n \times n$ matrix with $A_{ij}=Jc(n_i, n_j)$ ($Jc(n_i, n_j)$ is the Jaccard coefficient of n_i with n_j), $n_i, n_j \in R_{Paper}(n_{Author})$ (i.e. all papers of an author). e is an $n \times n$ unit matrix and $max[(A)(i, :)]$ is the maximum score in line i of matrix A. J_i produces values in the range [0,1), J_i fails to reach 1 because there are no two papers that are exactly alike. $a(e)$' is calculated by Formulas 4 where $f(v_i \rightarrow v_j)$ is a normalization function of $J(v_i \rightarrow v_j)$.

Now, this paper also proposes the Time Decrement (namely TD) for the *Paper* → *Paper*. More precisely, the rate with $TD(v_i \rightarrow v_j)$ of a paper v_i flows to its a cited paper v_j can be calculated by

$$TD(v_i \rightarrow v_j) = \frac{\dfrac{1}{A_{v_j}+b}}{\sum_{v_j \in p_{v_i}} \dfrac{1}{A_{v_j}+b}} \tag{6}$$

where p_{v_i} is a set of the paper that cited by paper v_i, is the "age" that v_j cited by v_i (calculated by $A_{v_j} = y_{v_i} - y_{v_j} + 1$, y_{v_i} is the year of the publication of paper v_i) and b is a tuning constant, it can adjust the transfer flow rate with different cited papers of different ages. It will not obtain a large weight of the cited paper with younger age, i.e. b will obtain the smaller value for the paper aging fast whereas the larger value for the paper aging slow. Regarding to tuning constant b, when $b = 5$, for instance, a paper was published in 1989 named "*A Knowledge Level Analysis of Belief Revision*" (denote as P_A) in Computer Science research field, it cites 2 papers: one of them was published in 1988 named "*Investigations into a Theory of Knowledge Base Revision*" (P_B), and another was published in 1986 year named "*Learning at the Knowledge Level*" (P_C). So that P_C's age is 4 and P_B's age is 2, then it can use the Formula 6 to calculate their corresponding $TD(P_A \rightarrow P_B)$ and $TD(P_A \rightarrow P_c)$ are 0.562 and 0.438 respectively. $a(e)$ is extension of $a(e)'$ can be calculate by

$$a(e) = \beta + \gamma(\sum_{s=1}^{n} a_s f_s(v_i \rightarrow v_j)) \tag{7}$$

where a_s is the score that required considering factor in transfer rate like J and TD, $f_s(v_i \rightarrow v_j)$ is its corresponding normalization function and $\sum_{s=1}^{n} a_s = 1$. **Fig. 3(b)** illustrates the G^A for the DBLP database.

Let r denote the vector with ValueRank r_i of a node v_i, then r can be calculated:

$$r = dAr + (1-d)\frac{s}{|S|} \tag{8}$$

where $A_{ij} = a(e)$ if there is an edge $e = (v_i \rightarrow v_j)$ in E_D^A or 0 otherwise, d control the Base Set importance and $s = [s_1, \ldots, s_n]^T$ is the Base Set vector for S, s_i and $a(e)$ are calculated by Formulas 3 and 7 respectively.

Table 1 gives ValueRank scores that were produced by the graph for the Microsoft trading Northwind database and $d = 0.85$ (i.e. the default setting: $d = 0.85$ as described in Section 5). Whereas ObjectRanks were generated by the corresponding "ObjectRank version" of this G^A (i.e. denoted as G^{A3}), namely *basic sets* were not used and it has $a(e) = \beta$ for all edges (see Section 5 for details). The results in [6], it interestingly shows that ValueRank provides a better comparison scores than that from ObjectRank, and we also get the following observations: In the Northwind database, ObjectRank is highly correlated with the total number of *Order_Details*, *Orders*, and so on. But ValueRank is highly correlated with the summing value of *Freight*, *Orders* and so on.

For example, Cus_SA has 31 total number of *Orders*, thus whose ObjectRank score (0.70) is higher than that of Cus_QU (0.62), on the other hand, Cus_QU has higher values (considering *Orders*), thus in the view of ValueRank, Cus_QU (0.69) is greater than Cus_SA (0.65). Moreover, Prod_59 has a greater total number of *Orders* (i.e. 54), while it has higher score in ObjectRank than that of Prod_38, however, by considering their corresponding values, the results of ValueRank scores are more likely balance the conditions of number and values.

Table 1. Selected examples in Microsoft trading dataset (ObjectRank against ValueRank scores)

Relation/ID	V.R.	O.R.	Number of Orders	Corresponding Values
Cus SA	0.65	**0.70**	**31**	115,673
Cus QU	**0.69**	0.62	28	**117,483**
...				
Ship 1	0.20	0.36	249	16,185
Ship 2	**0.27**	**0.47**	**326**	**28,244**
...				
Prod. 38	**1.00**	0.49	24	**149,984**

Prod. 59	0.50	**1.00**	**54**	76,296
...				
Emp. 4	**0.38**	**0.38**	**156**	**250,187**
Emp. 3	0.35	0.30	127	213,051
...				
Sup. 18	**0.04**	0.09	2	**281**
Sup. 7	0.02	**0.13**	**5**	178
...				

4. Greedy Algorithm: k-LASP

Note that the cost of dynamic programming algorithm[11] will be high when the required l is huge, so this paper proposes the following one greedy algorithm exploit accordingly interesting properties of OS for more efficiency. Although it provides approximate results in section 5.

Meanwhile, we define a greedy algorithm named k-LASP (k-Largest Averaged Score Path) in this paper, it is the extension of LASP [12] that uses a Priority Queue (PQ) to build the size-l OS by expanding on the current tuple with the largest averaged score path. But we have to update all remaining nodes when it selects a path (or a node) to the size-l object summaries on the size-l generations. Note that the cost of LASP algorithm will be high if the scale of |OS| is very large. This paper presents k-LASP, i.e. the largest averaged score path of k nodes. It has to calculate $w(t_i)$ of each node and its corresponding average $w(t_i)$ score with its n-1 ($n = max(k, length\text{-}1)$) grandparent nodes (donated as $AP_k(t_i)$) of the path from the t_i to the root. The corresponding $AP_k(t_i)$ of each node t_i on the size-l OS generation can be calculated by:

$$AP_k(t_i) = \frac{v_i + \sum_{j=1}^{n-1} w(R_j)}{n} \tag{9}$$

where $n = max(k$, physical length), R_is are(is) t_i's grandparent nodes(or node) that have been not selected to size-l OS, $w(R_i)$ is its corresponding score. More precisely (see Algorithm 1), the input of the algorithm are l (the size of tuples returned, i.e. the size of output)，t^{DS} (It can be regarded as the keyword tuple of search) and G^{DS} includes information about DSs and the relations linked t^{DS} , the t^{DS} contains the additional particular DS's information. Firstly, the initial OS (i.e. complete OS) with the $AP_k(t_i)$ calculated by Equation 9(line 1) is generated, The original value of each tuple in complete OS is calculated based on ValueRank (Equation 8). It use the PQ to select the largest AP score node and add its corresponding path p_i to size-l object summaries (lines 3 and 4). Then remove the nodes of p_i from OS and PQ, the OS tree become a forest, the parents of all roots of the forest are the nodes of p_i, the affected nodes v_i of this forest need to update its corresponding $AP(v_i)$ (lines 6-8). Finally, as long as the selected nodes are smaller than the required l, the process will be repeated. **Fig. 5** illustrates this algorithm using the example of 3-LASP, the t^{DS} is node t_1 in **Fig. 5**，**Fig. 5(a)** shows the complete OS generated by using t_1 as input, t_6 is the largest value in deQueue(PQ), so the path p_1 is $t_1 \rightarrow t_6$, we add first two nodes of p_1 to size-10 OS(line 3-4), now the number of | size-10 OS | is 2, so we remove t_1 and t_6 from the OS and PQ, for each descendant node t_i (number n, n = max(k-1, physical length)) of nodes in p_1, upadate descendant node t_i's value $AP_k(t_i)$ on the OS tree and PQ(line 6-8). **Fig. 5(b)** illustrates that t_9 is the largest value in deQueue(PQ), so p_2 is $t_3 \rightarrow t_9$, | size-10 OS | is 4, do line 6-8 again, so continue, **Fig. 5(d)** shows the result of this example of size-10 OS.

Algorithm 1: *k*-LASP Algorithm

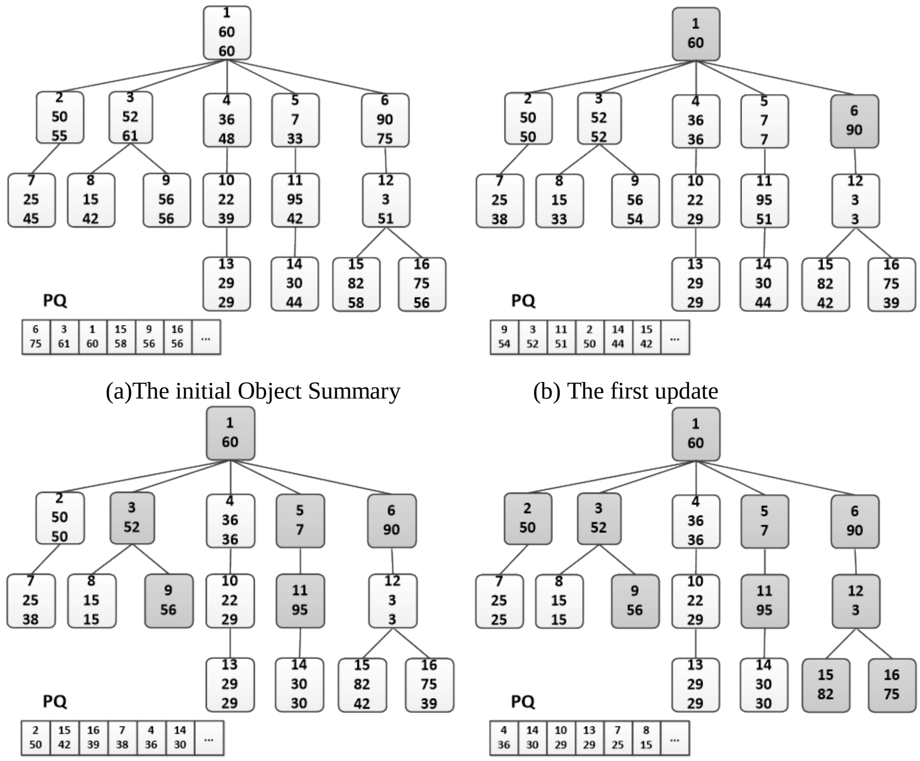
***k*-LASP (l, t^{DS}, G^{DS})**

Input: l, t^{DS}

Output: size-l OS

```
generate the initial OS and initial PQ    //the initial OS is the complete OS. PQ is the priority queue with OS's leaf nodes.
  calculate AP(t_i).
while (|size-l Object Summary| <l) do
   p_i = path from delete(PriorityQueue)          //the largest value from PQ
   add 1st (l-|size-l Object Summary|) nodes of p_i to size-l Object Summary
   if (|size-l Object Summary| < l) then
      Delete selected path p_i from the Object Summary tree and Priority Queue
      For each descendant node t_i (number n, n = max(k-1, physical length)) of nodes
         in p_i do
         update AP_k(t_i) on the OS tree and PQ
return size-l Object Summary                      //i.e. the output size-l OS
```

In the worst case, k-LASP costs $O(l(bk+n\log_2 n))$ to get size-l object summaries, where the number of l is the size of object summaries, variable b is the total number of tuples in the complete OS tree, n is nodes of the complete OS, every k-LASP chooses nodes to add to size-l OS, it costs $O(bk)$, i.e. the value of AP of descendant nodes in b paths need to be updated, and up to k nodes' AP are updated in each path on OS tree, sorting algorithm costs $O(n\log_2 n)$, So in the worst case, it needs to update l times, So the time complexity of k-LASP is $O(l(bk+n\log_2 n))$.



(a)The initial Object Summary (b) The first update

(c) The size-*5* or size-*6* OS (d) The size-*10* OS

Fig. 5. The 2-LASP Algorithm: Under construction size-*5* OSs and their corresponding PQs. (Shaded nodes are the selected ones).

5. Evaluations

We conduct our evaluation on two aspects, i.e. effectiveness and efficiency. We compare the results generated from our proposed *k*-LASP algorithm with different variables. It evaluates the scores (i.e.global importance) with both ValueRank and ObjectRank. Regarding to the simulation setup, initially, we investigate and study the effectiveness of ValueRank through our selected evaluators. Then, we comparatively investigate the performance results from both of the ObjectRank and ValueRank. Finally, we analyzed the quality of the object summaries emerged from the greedy heuristics algorithm *k*-LASP.

It used two databases in this paper: bibliography and trading, there are 2,959,511 and 3,209 tuples in the DBLP, MS Northwind databases. They take about 500MB and 1MB of disk space. With ObjectRank scores i.e. global importance[8] and ValueRank, it generating the *global importance* for each tuples of the and Northwind trading databases separately. Cold cache and a PC with an i5-4590 3.30 GHz (Intel-Core) processor and 8GB of memory were used in experiments.

5.1 Effectiveness

The effectiveness of ValueRank is thoroughly investigated comparatively with ObjectRank against evaluators. As the Northwind trading database and DBLP database have schema (comprising of many relationships, restrictions, and attributes), this paper uses them for evaluation. They have more understandable instances to present and evaluate the techniques easily. It measures the affect of d and transfer rate in different graphs. It imitates and extends the setting parameters used to evaluate ObjectRanks[3]. To be more exact, in [3], the affect of variable d is investigated (where d = 0.85, 0.99, 0.10, 0.85 is set as default).

Meanwhile, 3 groups of different graphs for the Northwind database and four different graphs for the DBLP database. Namely, for the Northwind database, the default graph1 of **Fig. 4**, For the DBLP database, this paper proposes two factors (i.e. Jc and TD), the Equation 7 becomes $\alpha(e)$ =0.7+0.3($a_1 f_s(Jc)$ + $a_2 f_s(TD)$), so the corresponding $G^{A\,I}$ is the G^A of **Fig. 3(b)** with a_1 = 0.5 and a_2 = 0.5, $G^{A\,II}$ is the G^A of **Fig. 3(b)** with a_1 = 0.1 and a_2 = 0.9, and $G^{A\,III}$ with a_1 = 0.9 and a_2 = 0.1. However, G^{AIV} had all α (consequently s_i = 0) and set to 0, hence producing ObjectRank values. **Table 2** illustrates the variables of graphs and default settings， **Table 3** illustrates the evaluation of ValueRank's effectiveness.

Table 2. Experimental variable and default settings

Parameter	**Range**
Graph	$G^1, G^2, G^3, G^A\ ,^{I}G^A\ ,^{II}G^A\ ,^{III}G^{AIV}$
$d(d_1, d_2, d_3)$	0.85, 0.10, 0.99

Table 3. Evaluation of ValueRank effectiveness

	Graph1- d_1	***Graph1- d_2***	***Graph1- d_3***	***Graph2- d_1***	***Graph3- d_1***
Effectiveness	7.8	8.1	7.8	7.8	3

For the objective of the evaluation of ValueRank's quality, ObjectRank[3] was used to conduct a similar evaluation survey. Namely, in our University, five professors and researchers are participated in this survey. Select lists of 10 tuples randomly, they are compared and ranked by every participant, afterwards give a score of 1 to 10. For each tuple, it also provides a set of descriptive details and statistical data. Generally, evaluators gives better scores on Graph1 and Graph2, with different settings of variable *d*, on the other hand, as we see from the results, the last group of settings, i.e. Graph2-d_1, did not satisfy evaluators, comparing with the rest groups, which is due to without considering values.

ValueRank also gives better comparative ranking than ObjectRank in the DBLP database, for instance, R^{G1}, R^{G2}, $R^{G3}$and R^{G4} are the corresponding ObjectRank (G^{AIV}), VauleRank (G^{AI}), VauleRank(G^{AII}) and VauleRank (G^{AIII})'s rank in all Author tuples (341,623) respectively. Author A_1's R^{G1} is 4, but R^{G2} is 2. The cause of the rank going up is that Author A_1's papers mainly concentrated in the direction of the Database, but some of his papers are not or little relationships to the direction of the Database, the number of these papers is n_{ur} and the number of his all papers is named n_{sum}, then we can get a ratio r_i calculated by n_{ur} / n_{sum} , the bigger of r_i, his the value of rank will drop more. On the contrary, Author A_2's R^{G2} is higher than R^{G1}, because the field of his papers is relatively concentrated, he majors in computer science and technology. Authors' R^{G3} and R^{G4} are changed by corresponding required which compared with R^{G2}. R^{G3} emphasis *TD* more and R^{G4} emphasis *Jc* more. The changes of the tuples of Papers' rankings are alike to Authors'. Paper A_1's R^{G2} is higher than R^{G3}, because *Jc* is higher than others, i.e. this paper has strong relevance with cited papers and *TD* is also higher, i.e. this paper is younger, it makes better qualified for users. Similarly, the *TD* should be paid more attention to, then the Paper B_1 and B_2 have corresponding changes. And, we pay more attention to the *Jc*, then the Paper C_1 and C_2 have corresponding changes. The result (**Table 4**) illustrates the impact of G^A on tuples ranking on the DBLP database.

Table 4. **S**amples of ObjectRank and ValueRank scores in DBLP database

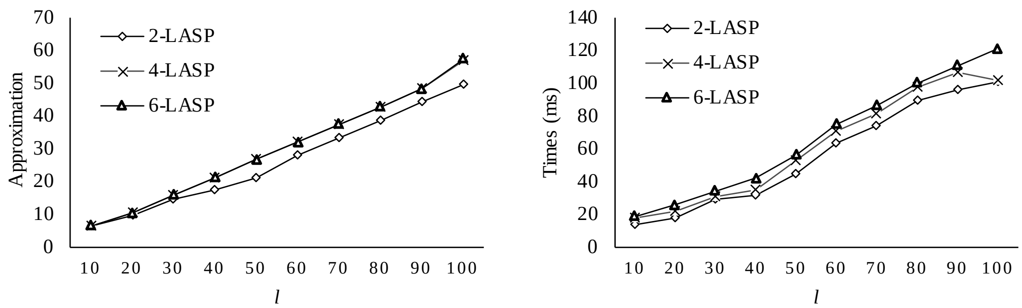
Tuple ID	R^{G1}	R^{G2}	R^{G3}	R^{G4}
Author A_1	97,763	210,913	--	--
Author A_2	4	2	--	--
Author B_1	--	45	47	--
Author B_2	--	37,187	35,196	--
Author C_1	--	777	--	765
Author C_2	--	934	--	925
…				
Paper A_1	37	8	--	--
Paper A_2	454	3896	--	--
Paper B_1	--	13	11	--
Paper B_2	--	8	12	--
Paper C_1	--	12	--	6
Paper C_2	--	15	--	23

5.2 Efficiency

In this subsection, we mainly focusing on comparing the overall importance of the size-*l* OSs generated by the greedy method (i.e. our proposed *k*-LASP algorothm). For details, the results of **Fig. 6.(a)** show the approximate quality under the default settings, namely holistic importance of the achieved object summary importance (i.e. *Im*(size-*l*)). Meanwhile, the average results for 10 random object summaries are shown. The result shows that the scores of 4-LASP and 6-LASP are always higher than the 2-LASP. This is exactly what we expect, the node with lower score may be considered because its ancestor nodes have higher scores. In other words, the node with higher score may not be considered because its ancestor nodes have lower scores. For instance, *Im*(OS, P_1) = 0.4 (P_1 is a paper tuple 'On Total Functions...') and one of its children is a Year tuple Y_1 with Im(OS, Y_1) = 1.2, Im(OS, P_2) = 0.9 (P_2 is a paper tuple 'A deterministic...') and one of its children is a Year tuple Y_2 with Im(OS, Y_2) = 1.0, it will choose the tuple Y_1 and P_1 by traditional method, but actually more Paper tuples may want to be known, so Y_2's score (= (1.0+0.9)/2 = 0.95) is more than other tuples' scores (like Y_1's = (1.2+0.4)/2 = 0.8) by 2-LASP., the data subject graph is as same as **Fig. 2** with the setting θ=0.7, so the results of 4-LASP and 6-LASP have the same importance. Since the running cost of the blind search algorithm is very high, we have not given any optimization results.

On the other hand, we also considering the total run-time performance of our proppsed algorithm with different coefficient k in **Fig. 6.(b)**. Again the same object summaries are used as in **Fig. 6.(a)** (i.e. the same 10 object summaries) and generate the global importance of the tuple with the default settings.

Fig. 6.(b) shows the costs of our algorithm using different k values to calculate size-l OSs from OSs with different l values, excluding the time required to generate the OS for the algorithm. We can see that with the increases of k, the cost is increases, so the cost of 2-LASP is the lowest.



(a)Approximation Quality on DBLP (b) Efficiency on DBLP

Fig. 6. Approximation Quality and Efficiency on DBLP(Aver(|OS| = 1116)

6. Conclusion and Future Works

In this paper, based on our previous work, we initially extended the ValueRank approach, which also taking the values from none business dataset, e.g. DBLP database into account, to further providing precise authority transfer flow when calculating their neighboring relations. Meanwhile, we also provided a novel faster object summary generation algorithm, i.e. k-LASP algorithm, which not only the single average score per path or pre pare, but also considering the k-LASP from the root. The evaluation show that our proposed methods have significant results in relational keyword searches.

As a further work, we will extend our proposed techniques to more complicated relational dataset, e.g. XML, OWL, and also further investigate the spatio-temporal dataset while taking location and time as values combining keyword search.

References

[1] Turpin, A., Tsegay, Y., Hawking, D., & Williams, H. E, "Fast generation of result snippets in web search," in *Proc. of International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM*, pp.127-134, 2007. Article (CrossRef Link)

[2] Aditya, B., Bhalotia, G., Chakrabarti, S., Nakhe, C., Nakhe, C., & Parag, P., et al., "BANKS: browsing and keyword searching in relational databases," in *Proc. of International Conference on Very Large Data Bases, VLDB*, pp.1083-1086, 2002.

[3] Bhalotia, G., Hulgeri, A., Nakhe, C., Chakrabarti, S., & Sudarshan, S., "Keyword searching and browsing in databases using BANKS," in *Proc. of International Conference on Data Engineering, 2002. Proceedings IEEE*, pp.431-440, 2002. Article (CrossRef Link)

[4] Hristidis, V., & Papakonstantinou, Y., "Discover: keyword search in relational databases," *Vldb*, Vol. 26 No. 2, pp. 670-681, 2002. Article (CrossRef Link)

[5] Fakas, G. J., "Automated generation of object summaries from relational databases: A novel keyword searching paradigm," in *Proc. of IEEE, International Conference on Data Engineering , DBRank Workshop, IEEE Computer Society*, pp.564-567, 2008. Article (CrossRef Link)

[6] Fakas, G. J., Cai, Z., "Ranking of object summaries," *IEEE International Conference on Data Engineering, DBRank Workshop*, pp.1580-1583, 2009. Article (CrossRef Link)

[7] Sehgal, U., Kaur, K., & Kumar, P., "Notice of Violation of IEEE Publication Principles The Anatomy of a Large-Scale Hyper Textual Web Search Engine," in *Proc. of International Conference on Computer & Electrical Engineering, IEEE Computer Society*, Vol.2, pp.491-495, 2009. Article (CrossRef Link)

[8] Hwang, H., Hristidis, V., & Papakonstantinou, Y., "Objectrank: authority-based keyword search in databases," *ACM Transactions on Database Systems (TODS)*, vol. 33, no. 1, 2008. Article (CrossRef Link)

[9] Varadarajan, R., Hristidis, V., & Raschid, L., "Explaining and Reformulating Authority Flow Queries," in *Proc. of IEEE, International Conference on Data Engineering, IEEE*, pp.883-892, 2007. Article (CrossRef Link)

[10] Huang, X. F., "Tuplerank and implicit relationship discovery in relational databases," *Lecture Notes in Computer Science*, 2762, 445-457, 2003. Article (CrossRef Link)

[11] Fakas, G. J., Cai, Z., & Mamoulis, N., "Versatile size-l object summaries for relational keyword search," in *Proc. of IEEE Transactions on Knowledge & Data Engineering*, Vol.26 No.4, 1026-1038, 2014. Article (CrossRef Link)

[12] Fakas, G., Cai, Z., & Mamoulis, N., "Diverse and Proportional Size-l Object Summaries for Keyword Search," in *Proc. of ACM SIGMOD International Conference*, pp.363-375, 2015. Article (CrossRef Link)

[13] Hristidis, V., Papakonstantinou, Y., & Gravano, L, "Efficient IR-Style Keyword Search over Relational Databases," in *Proc. of 2003 VLDB Conference*, pp.850-861, 2003. Article (CrossRef Link)

[14] Liu, F., Yu, C., Chowdhury, A., & Chowdhury, A., "Effective keyword search in relational databases," in *Proc. of ACM SIGMOD International Conference on Management of Data, ACM*, pp.563-574, 2006. Article (CrossRef Link)

[15] Yi Luo, Wei Wang, Xuemin Lin, & Xiaofang Zhou., "Spark2: top-k keyword query in relational databases," in *Proc. of Knowledge and Data Engineering, IEEE Transactions on*, Vol.23 No.12, pp.763-1780, 2011. Article (CrossRef Link)

[16] Yu, B., Li, G., Sollins, K., & Tung, A. K. H., "Effective keyword-based selection of relational databases," in *Proc. of ACM SIGMOD International Conference on Management of Data, ACM*, pp.139-150, 2007. Article (CrossRef Link)

[17] Huang, Y., Liu, Z., & Chen, Y., "Query biased snippet generation in XML search," in *Proc. of ACM SIGMOD International Conference on Management of Data, ACM*, pp.315-326, 2008. Article (CrossRef Link)

[18] Tombros, Anastasios, & Sanderson, Mark., "Advantages of query biased summaries in information retrieval," in *Proc. of SIGIR '98: Proceedings of the, International ACM SIGIR Conference on Research and Development in Information Retrieval, August 24-28 1998, Melbourne, Australia*, pp.2-10, 1998. Article (CrossRef Link)

[19] Fakas, G. J., "A novel keyword search paradigm in relational databases: object summaries," *Data & Knowledge Engineering*, Vol.70 No.2, pp.208-229, 2011. Article (CrossRef Link)

[20] Markowetz, A., Yang, Y., & Papadias, D., "Keyword search over relational tables and streams," *Acm Transactions on Database Systems*, Vol.34 No.3, pp.1-51, 2009. Article (CrossRef Link)

[21] Sydow, M., Pikuła, M., & Schenkel, R., "The notion of diversity in graphical entity summarisation on semantic knowledge graphs," *Journal of Intelligent Information Systems*, Vol.41 No.2, pp.109-149, 2013. Article (CrossRef Link)

[22] Cheng, G., Tran, T., & Qu, Y., "RELIN: Relatedness and Informativeness-Based Centrality for Entity Summarization," *The Semantic Web - ISWC 2011 -, International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, DBLP*, Vol.7031, pp.114-129, 2011. Article (CrossRef Link)

[23] Li, W. S. , K. Selçuk Candan, Vu, Q. , & Agrawal, D., "Retrieving and organizing web pages by "Information Unit," in *Proc. of International World Wide Web Conference*, pp. 230-244, 2001. Article (CrossRef Link)

[24] Graupmann, J., Schenkel, R., Weikum, G., Böhm, K., Jensen, C. S., & Haas, L. M., et al., "The spheresearch engine for unified ranked retrieval of heterogeneous xml and web documents," 2005.

[25] Vinueza Naranjo, Paola & Shojafar, Mohammad & Vaca-Cardenas, Leticia & Canali, Claudia & Lancellotti, Riccardo & Baccarelli, Enzo, "Data Over SmartGrid -A Fog Computing Perspective," 2016.

[26] Baccarelli, Enzo & Scarpiniti, Michele & Vinueza Naranjo, Paola & Vaca-Cardenas, Leticia, "Fog of Social IoT: When the Fog Becomes Social," *IEEE Network*, 32(4), 68-80, 2018. Article (CrossRef Link)

Cai Zhi: is an associate professor in the College of Computer Science, Beijing University of Technology, China. He obtained his M.Sc. in 2007 from the School of Computer Science in the University of Manchester and his Ph.D. in 2011 from the Department of Computing and Mathematics of the Manchester Metropolitan University, U.K. His research interests include Information Retrieval, Ranking in Relational Databases, Keyword Search, Intelligent Transportation Systems.

Lan Xu: received her Master of Engineering in Computer Science of Technology from Beijing University of Technology, Beijing, P.R. China, in 2018. Her research interests include relational databases and content retrieval.

Su Xing: is a lecturer in the Faculty of Informatics, Beijing University of Technology in Beijing, China. He received his B.Sc in the school of software engineering from Beijing University of Technology in 2007. He received his M.Sc and PhD in computer science from University of Wollongong, Australia in 2012 and 2015. His research interests include distributed artificial intelligence, multi-agent systems, disaster management and service-oriented computing.

Lang Kun: is a manager of CTGI, she received her Master of Science Degree in Business School, the University of Manchester, in 2011. Her research interests include relational databases and content retrieval.

Cao Yang: received his Ph.D. of Computer Science from Beijing University of Technology, Beijing, P.R. China, in 2019. His research interests include data mining, relational databases and content retrieval.