Ranking of Object Summaries

Georgios John Fakas, Zhi Cai

Department of Computing and Mathematics,
Manchester Metropolitan University,
Manchester, UK.
{g.fakas, z.cai}@mmu.ac.uk

***Abstract*— A previously proposed Keyword Search paradigm produces, as a query result, a ranked list of Object Summaries (OSs); each OS summarizes all data held in a relational database about a particular Data Subject (DS). This paper further investigates the ranking of OSs and their tuples as to facilitate (1) the top-*k* ranking of OSs and also (2) the generation of partial size-*l* OSs (i.e. comprised of the *l* most important tuples).**
Therefore, a global Importance score for each tuple of the database (denoted as $Im(t_i)$) is investigated and quantified. For this purpose, *ValueRank* (an extension of ObjectRank) is introduced which facilitates the estimation of scores for arbitrary databases (in contrast to PageRank-style techniques that are only effective on bibliographic databases). In addition, a variation of Combined functions are investigated for assigning an Importance score to an OS (denoted as *Im(OS)*) and a local Importance score of their tuples (denoted as $Im(OS, t_i)$).
Preliminary Experimental evaluation on DBLP and Northwind Databases is presented.

I. Introduction

The success of the Web Keyword Search (W-KwS) paradigm has encouraged the emerge of the Keyword Search paradigm in Relational databases (R-KwS) [1, 4, 6]. Keyword Search paradigms have been very successful so far because they allow users to extract effectively and efficiently useful information using only a set of keywords. The R-KwS paradigm is very useful when trying to combine keywords, e.g. "Faloutsos Papadias" which will return papers co-authored by Faloutsos and Papadias. In contrast, the R-KwS paradigm is not very effective when trying to extract information about a particular DS, e.g. for "Faloutsos". The diagram below illustrates the result for **Keyword Query** Q1="Faloutsos" on the DBPL database (used in [2]). Namely a (ranked) set of Authors tuples containing the Faloutsos keyword; which are the Author tuples corresponding to the three brothers. It is apparent that the results of the R-KwS paradigm fail to provide comprehensive information to users about the Faloutsos brothers, e.g. a complete list of their publications and other corresponding details.

Author, Id: 557432, Name: Christos **Faloutsos**
Author, Id: 611200, Name: Michalis **Faloutsos**
Author, Id: 558418, Name: Petros **Faloutsos**

Fig. 1. Q1 on DBLP using R-KwS

The novel Keyword Search paradigm proposed by Fakas [3] introduces OSs, where an OS summarises data held in a database about a particular DS. This paradigm resembles more the W-KwS rather than R-KwS. Therefore, users with W-KwS experience will potentially find it friendlier and also closer to their expectations. For instance, the result for Q1 will be a ranked set of OSs - one per brother that includes all data held in the database for each brother. This result evidently provides a more complete set of information per brother. Fig. 2 illustrates the OS for Christos Faloutsos (the complete set of papers was omitted because of lack of space; please visit demo at http://mudfoot.doc.stu.mmu.ac.uk/research/ksdbos/).

Author: Christos **Faloutsos**
The QBIC Project: Querying Images by Content, Using Color, Texture, and Shape. *Storage and Retrieval for Image and Video Databases (SPIE), 1993*
An Efficient Pictorial Database System for PSQL. *IEEE Trans. Software Eng. 1988*
.
.
.
KDD-2002 Workshop Report Fractals and Self-similarity in Data Mining: Issues and Approaches. *SIGKDD Explorations, 2002*
Reminiscences on Influential Papers, *SIGMOD Record, 2000*
Total 144 papers

Fig. 2. OS for Christos Faloutsos

From the results of the above Example, we make the following observations: (1) some of the OSs may be very large in size, for instance, Christos Faloutsos has (co-) authored many papers. This is not only unfriendly to users that would like a quick glance at a first stage until they realise which Faloutsos they are really interested in but also expensive to produce. Therefore, the presentation of a partial OS of size-*l* may be adequate for users either because this will include the complete answer or because it can assist to find the DS users are looking for (and then proceed by requesting a complete OS). (2) Similarly, the number of OSs may be large. In this case the number of OSs is only three but searching for "Papadopulos" or "index" keywords the results may be hundreds of OSs. Evidently, the effective and efficient ranking and more precisely the top-*k* ranking of OSs and the top size-*l* ranking of their tuples are necessary. Fig. 3 illustrates Q1 with k=3 and l=10 on the DBLP database.

Challenges

Ranking database's tuples and estimating global Importance scores of tuples (denoted $Im(t_i)$) is a challenging problem; since techniques such as PageRank and ObjectRank

[2] can only be applied on bibliographic databases. Therefore in this paper, ValueRank is introduced that also incorporates tuples' values and as a consequence can be applied in any type of database.

Ranking OSs is another challenging problem since existing ranking semantics of traditional R-KwS are completely inappropriate for OS ranking. As in R-KwS, a result of a small size has generally a higher ranking semantic than another result of a larger size [1, 4, 5]. In contrast, an OS containing many well connected tuples should have certainly greater importance. In this paper, a ranking paradigm is proposed that ranks OS descending their Importance scores, (denoted as *Im(OS)*) that considers (1) each comprising tuple's local Importance score (denoted as $Im(OS, t_i)$) and (2) the *size* of the OS (denoted as |OS|); where $Im(OS, t_i)$ is a function of (1) tuple's global Importance scores ($Im(t_i)$) and (2) tuples' Affinity scores (denoted as $Af(t_i)$).

Author: Christos **Faloutsos**
On Power-law Relationships of the Internet Topology. *SIGCOMM, 1999*
The QBIC Project: Querying Images by Content, Using Color, Texture, and Shape. *Storage and Retrieval for Image and Video Databases (SPIE), 1993*
Efficient and Effective Querying by Image Content. *J. Intell. Inf. Syst., 1994*
10 tuples (Total 144 papers)

Author: Michalis **Faloutsos**
On Power-law Relationships of the Internet Topology. *SIGCOMM, 1999*
QoSMIC: Quality of Service Sensitive Multicast Internet Protocol. *SIGCOMM, 1998*
Aggregated Multicast with Inter-Group Tree Sharing. *Networked Group Communication, 2001*
10 tuples (Total 14 papers)

Author: Petros **Faloutsos**
On Power-law Relationships of the Internet Topology, *SIGCOMM, 1999*
Composable controllers for physics-based character animation. *SIGGRAPH, 2001*
The virtual stuntman: dynamic characters with a repertoire of autonomous motor skills. *Computers & Graphics. 2001*
10 tuples (Total 9 papers)

Fig. 3. Q1 on DBLP using top-*3* and size-*10* OSs

Contributions:

- The proposition of ValueRank, an extension of ObjectRank, that facilitates the global ranking of tuples of any arbitrary databases rather than only bibliographic. Preliminary results reveal that ValueRank provides better results on non-bibliographic databases.
- The effective size-*l* ranking of OS tuples.
- The effective top-*k* ranking of OSs.

Paper Organization

The rest of the paper is structured as follows. Section II describes preliminaries of OSs and their semantics. Section III presents global Importance score and ValueRank whilst Section IV presents local Importance score and how they can be used in top-*k* and size-*l* ranking of OSs. Section V presents initial experimental results. Section VI concludes the paper.

II. Preliminaries of the Proposed Keyword Search Paradigm

In the context of Fakas Keyword Search paradigm [3], the result of a Keyword Query is a set of OSs; where an OS is a tree comprised of tuples, with the t^{DS} tuple as the root node and t^{DS}'s neighboring tuples the child nodes. An OS is generated for each tuple (t^{DS}) found in the database that contains the keyword(s) as part of an attribute's value. For each R^{DS} (where $t^{DS} \in R^{DS}$ includes a keyword), a Data Subject Schema Graph (G^{DS}) is generated; namely a Directed Labelled Tree that captures a subset of the schema with R^{DS} as a root. Affinity measures of relations in G^{DS} (denoted as $Af(R_i)$) are quantified and annotated on the G^{DS}. The Affinity of a Relation R_i to R^{DS} can be calculated with the following formula:

$$\textbf{(1)} \quad Af(R_i) = \sum_j m_j w_j * Af(R_{Parent}) \qquad (1)$$

where j ranges over a set of metrics ($m_1,..,m_n$) and their corresponding weights ($w_1,..,w_n$) and $Af(R_{Parent})$ is the Affinity of the R_i's Parent to R^{DS}. Affinity metrics between R_i to R^{DS} include (1) their distance to R^{DS} and (2) their connectivity properties on both the database schema and the data-graph. (Refer to [3] for more details). Provided an Affinity threshold θ, a subset of G^{DS} can be produced (denoted as $G^{DS}(\theta)$). Finally, by traversing the $G^{DS}(\theta)$ we can generate OSs. E.g. from Author $G^{DS}(0.7)$, we can produce the OSs of Fig. 2 and 3.

A. Top-k and Size-l Keyword Queries

A **top-*k* Keyword Query** is a set of keywords and a value for k provided by the user; e.g. Q1 with k=3. The result of this top-k Keyword Query will include a partially ranked set of OSs; namely the k(=3) most important OSs ranked descending their *Im(OS)*. Similarly, a **size-*l* Keyword Query** is a set of keywords and a value for l; e.g. Q1 with l=10. The result of this query will include the complete ranked set of OSs where each OS is comprised only by a maximum size-l(=10) tuples; namely the l tuples with the highest $Im(OS, t_i)$. Analogously, a **top-*k* and size-*l* Keyword Query** is a set of keywords together with k and l (Fig. 3). The result of this Query is the analogues combination of the first two types of queries.

III. Global Ranking of Tuples ($Im(t_i)$): ValueRank

For bibliographic databases such as DBLP, the ObjectRank [2], a PageRank-style approach, is considered to be the most effective. In contrast, for trading databases such as Northwind or TPC-H, PageRank-style approaches although they give some indication of the important nodes, they completely ignore the values of their tuples. For instance, although a particular customer C_1 has many orders, a customer C_2 with fewer orders may be significantly more important if his orders are of bigger value. Therefore we observe, that in such databases, we must rank OSs based on the values of some of

their tuples. We propose and investigate a more general solution, i.e. ValueRank that can address arbitrary databases. Nevertheless, we plan to experiment with both techniques (ValuesRank and ObjectRank), where $Im(t_i)$ can be the normalised value of these scores.

A. ObejctRank Preliminaries

ObjectRank [2] is an extension of PageRank and introduces the concept of *Authority Transfer Rates* between the tuples of each relation of the database. More precisely, the database is modelled as a labelled Data Graph $D(V_D, E_D)$ whilst its schema structure is described by the Schema Graph $G(V_G, E_G)$. From the Schema Graph $G(V_G, E_G)$ we create the corresponding Authority Transfer Schema Graph $G^A(V_G, E^A)$ to reflect the authority flow through the edges of the graph (see Fig. 4). More precisely, for each edge $e_G=(v_i \rightarrow v_j)$ of the E_G two Authority Transfer Edges are created, i.e. $e^f_G=(v_i \rightarrow v_j)$ and $e^b_G=(v_j \rightarrow v_i)$. Finally, from the Data Graph D and Authority Transfer Schema Graph G^A, the Authority Transfer Data Graph $D^A(V_D, E^A_D)$ can be derived as follows: for each edge of E_D the D^A has two edges $e^f=(v_i \rightarrow v_j)$ and $e^b=(v_j \rightarrow v_i)$ which are annotated with the corresponding Authority Transfer Rates $\alpha(e^f)$ and $\alpha(e^b)$. Where $\alpha(e^f)=\alpha(e^f_G)/OutDeg(u, e^f_G)$ if $OutDeg(u, e^f_G)>0$ or $\alpha(e^f)=0$ otherwise; $OutDeg(u, e^f_G)$ is the number of outgoing edges from u, of type e^f_G ($\alpha(e^b)$ is defined accordingly).

Instead of using the whole V_D as a Base Set we can use an arbitrary subset S of nodes, hence increasing the authority associated with them. In the case of ObjectRank, S can be the set of tuples that include the keywords. Let **r** denote the vector with ObjectRank r_i of each node v_i, then **r** can be calculated:

$$\mathbf{(2)}\quad \mathbf{r} = d\mathbf{A}\mathbf{r} + (1-d)\frac{\mathbf{s}}{|S|}$$

where $A_{ij}=\alpha(e)$ if there is an edge $e=(v_i \rightarrow v_j)$ in E^A_D and 0 otherwise, d controls the Base Set importance and $\mathbf{s}=[s_1,\ldots,s_n]^T$ is the Base Set vector for S, i.e. $s_i=1$ if v_i belongs to S and $s_i=0$ otherwise.

B. ValueRank

ValueRank is an extension of ObjectRank where the Base Set S and *Authority Transfer Rates* consider tuples' values. The Base Set S includes nodes whose values are considered to have significant influence on other nodes authority. For instance in the Northwind database, all tuples from $R_{OrderDetails}$, $R_{Product}$ since their values Product.Price and OrderDetails.Price*OrderDetails.Quantity respectively influence global authority. In addition, the *Authority Transfer Rate* between Customers, Orders, OrderDetails etc. can be a function of these (normalised) values. For instance, consider Customer C_1 with two Orders of values \$4 and \$5 and C_2 with only one Order of value \$1,000, then the *Authority Transfer Rate* between Customers and Orders can be a function of these values and therefore C_2 would obtain higher ValueRank.

The s_i value of a node v_i that belongs to S is a normalised value that describes the comparative importance of the node and is a function of v_i's attributes' values. The s_i of a node v_i in S can be calculated with the formula:

$$\mathbf{(3)}\quad s_i=\alpha+\beta\cdot f(v_i)$$

where α and β are tuning constants such that that $\alpha+\beta\leq 1$ and $f(v_i)$ is a normalisation function of the values of v_i (s_i produces values in the range [0, 1] rather than just 1 as in the case of ObjectRank). The tuning constants allow a minimum value of s_i in case v_i is 0 (or close to 0) and control of the impact of the value of v_i in general. Also, notice that when $\alpha=1$ and $\beta=0$ then $s_i=1$; i.e. the same value of s_is as in ObjectRank. For example, for a tuple v_i in $R_{OrderDetails}$, $s_i=0.1+0.9*f(OrderDetails.Price*OrderDetails.Quantity)$. It is, however, possible that s_i to be a function of neighbouring nodes attributes, e.g. s_i for a tuple of Orders $s_i=f(\Sigma OrderDetails.Price*\ OrderDetails.Quantity)$.

We also combine v_i's values with Authority Transfer Edges, therefore have more dynamic transfer rates. The intuition is that different attributes' values of a particular tuple may influence different edges of the tuple. For instance, for the $R_{Orders} \rightarrow R_{Shippers}$ edge, the Authority Transfer Edge is a function of Orders.Freight whilst for the $R_{Orderss} \rightarrow R_{Customers}$ edge is a function of total Order values (i.e. UnitPrice*Quantity, hence s_i value is adequate). The Authority Transfer Edges, either forward or backward denoted as $a(e)$, can be calculated with the formula:

$$\mathbf{(4)}\quad \alpha(e)=\gamma+\delta\cdot f(v_i \rightarrow v_j)$$

where γ and δ are tuning constants such that that $\gamma+\delta\leq 1$ and $f(v_i \rightarrow v_j)$ is a normalisation function of the values of v_i and v_j. Fig. 4 illustrates the G^A for the Northwind database. Similarly to ObjectRank, the selection of *Authority Transfer Rates*, S and tuning constants can be experimental.

The ValueRank r_i of a node v_i can be calculated by Formula 2; where s_i and $\alpha(e)$ are calculated by Formulas 3 and 4 respectively.

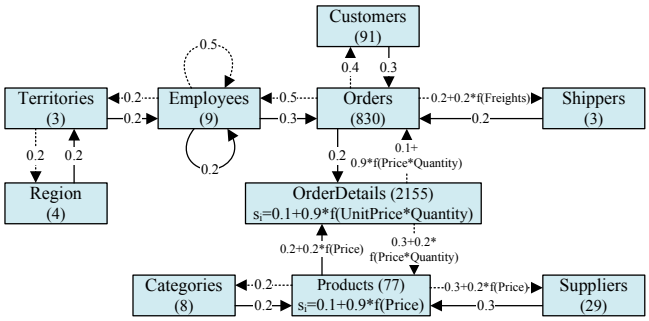

Fig. 4. The G^A for the Northwind database (Cardinality)

IV. Local Ranking of Tuples ($Im(OS, t_i)$)

The local Importance of each tuple t_i of an OS (namely $Im(OS, t_i)$) can be calculated with:

$$\mathbf{(5)}\quad Im(OS, t_i)= Im(t_i)^{\alpha} * Af(t_i)^{\beta}$$

where $Im(t_i)$ is the global Importance of t_i (e.g. its ValueRank or ObjectRank), $Af(t_i)$ is the Affinity of t_i to the t^{DS} (namely the Affinity of the Relation R_{ti} it belongs to to R^{DS};

denoted also as $Af(R_{ti})$), α and β are tuning constants. The product of $Im(t_i)$ with $Af_R(t_i)$ actually reduces the *Importance* contribution of each tuple towards the overall *Im(OS)*. This was necessary as discrimination of tuples with different Affinity is considered necessary; for instance, recall that tuples with small Affinity may or may not be included in an OS (depending on threshold value); therefore the values of thresholds should not impact significantly *Im(OS)*.

A. Top-k Ranking of OS

We treat an OS as a document comprising of |OS| tuples; where each tuple is associated with a local Importance score $Im(OS, t_i)$. The Importance of an OS *Im(OS)* should consider (1) $Im(OS, t_i)$ of each tuple and (2) the size of OS. Therefore, the following formula can be used:

$$\textbf{(6)}\quad Im(OS)=\frac{\sum Im(OS,t_i)}{log(|OS|)+1}$$

The size of the OS is depressed with a log; notice that excluding the log will result to the average of $Im(OS, t_i)$ which is not desired as big OSs may be equalised with small OSs. The following variations of the above formula will also be investigated:

$$\textbf{(6.1)}\quad Im(OS)=\Sigma Im(OS, t_i)$$

$$\textbf{(6.2)}\quad Im(OS)=Im(t^{DS})$$

$$\textbf{(6.3)}\quad Im(OS)=\frac{\sum Im(OS,t_i)}{|OS|}$$

Formula 6.1 disregards the size of the OS. The intuition is that the more tuples and the more important tuples an OS contains the higher ranking gets. Formula 6.2 considers only the global Importance of the t^{DS} tuple. This is a simple to implement and very cheap to execute solution. This is because it ignores the rest OS tuples' Importance and Affinity and therefore *Im(OS)* ranking can be achieved without realising OSs. As it is described in the previous section, the $Im(t_i)$ is calculated from Importance transfer from neighbour tuples and therefore $Im(t^{DS})$ represents to some extent its neighbours. Finally, the last formula averages the local Importance scores of OSs' tuples.

B. Size-l Ranking of OS

As mentioned in the Introduction, for usability and efficiency reasons (since the size of an OS may be large), OSs may also be presented partially to users (rather than complete) containing only the *l* most important tuples (i.e. size-*l*). For this purpose, $Im(OS, t_i)$ (Formula 5) can be used.

V. Experimental evaluation

The proposed ranking paradigms will be evaluated with three databases (DBLP, Northwind and TPC-H). So far, we have only produced ObjectRank and ValueRank results for the DBLP and Northwind databases. As DBLP database can only be ranked with ObjectRank and this has already been well examined by [2], we will concentrate on the Northwind database results.

The results below show ObjectRank and ValueRank scores for several tuples of the database produced for d=0.85 and the G^A of Fig 4. For the ObjectRank, however, no Base Set was used and for all edges we have $\alpha(e)=\gamma$. The results interestingly show that ValueRank gives better comparative ranking than ObjectRank and also make the following general observation: ObjectRanks have bigger correlation with the total amount of Orders, OrderDetails etc. whilst ValueRanks with the total value of Orders, Freight etc.

Tuple ID	ObjectRank	ValueRank	Total Orders	{UnitPrice*Quantity, Freight†, Price††}
Employee 4	**0.617**	**0.896**	**156**	**250187.4**
Employee 3	0.515	0.752	127	213051.3
….				
Shipper 2	**1.000**	**0.841**	**326**	**28244.8**†
….				
Product 38	0.195	**1.000**	24	**149984.2**
Product 59	**0.397**	0.903	**54**	76296
….				
Customer SAVEA	**0.094**	0.104	**31**	115673.39
Customer QUICK	0.089	**0.114**	28	**117483.39**
….				
Supplier 18	0.034	**0.060**	2	**281.5**††
Supplier 7	**0.043**	0.042	**5**	177.85††
….				

Fig 5. Samples of normalised ObjectRank and ValueRank scores (maximum values per relation are indicated in bold)

VI. Conclusions and Future Work

This paper presents work in progress of the top-*k* and size-*l* ranking of OSs in the context of Fakas Keyword Search in relational databases. More precisely, ValueRank (an extension of ObjectRank) is introduced in order to estimate the global Importance of tuples. To the best of the authors' knowledge this is the first PageRank-style attempt to rank tuples considering also their values. In addition, a variation of Combined functions are proposed for assigning an Importance score to an OS and a local Importance score of their tuples. Preliminary experimental evaluation results on DBLP and Northwind Databases are presented.

A direction of future work concerns the efficient top-*k* and size-*l* ranking of OSs. This is a challenging problem as local Importance scores are not monotonic (e.g. a tuple's global Importance may increase whilst its Affinity decrease). For this purpose, hashing and indexing techniques will be investigated.

References

[1] B. Aditya, G. Bhalotia, S. Chakrabarti, A. Hulgeri, C. Nakhe, P. S. Sudarshan, "BANKS: Browsing and keyword searching in relational databases", *In VLDB*, 2002.

[2] A. Balmin, V. Hristidis, and Y. Papakonstantinou, "Objectrank: Authority-based keyword search in databases", *In VLDB,* 2004.

[3] G. Fakas, "Automated Generation of Object Summaries from Relational Databases: A Novel Keyword Searching Paradigm", DBRank Workshop 2008, ICDE 2008.

[4] V. Hristidis, and Y. Papakonstantinou, "DISCOVER: Keyword search in relational databases", *In VLDB*, 2002.

[5] V. Hristidis, L. Gravano, and Y. Papakonstantinou, "Efficient IR-style keyword search over relational databases", *In VLDB*, 2003.

[6] A. Markowetz., Y. Yang, and D. Papadias. "Keyword search on relational data streams", In SIGMOD, 2007.