On Perspective-Aware Top-*k* Similarity Search in Multi-relational Networks*

Yinglong Zhang[1,2], Cuiping Li[1], Hong Chen[1], and Likun Sheng[2]

[1] Key Lab of Data Engineering and Knowledge Engineering of MOE, and
Department of Computer Science, Renmin University of China, China
zhang_yinglong@126.com,cuiping_li@263.net

[2] JiangXi Agricultural University, China

Abstract. It is fundamental to compute the most "*similar*" k nodes w.r.t. a given query node in networks; it serves as primitive operator for tasks such as social recommendation, link prediction, and web searching. Existing approaches to this problem do not consider types of relationships (edges) between two nodes. However, in real networks there exist different kinds of relationships. These kinds of network are called multi-relational networks, in which, different relationships can be modeled by different graphs. From different perspectives, the relationships of the objects are reflected by these different graphs. Since the link-based similarity measure is determined by the structure of the corresponding graph, similarity scores among nodes of the same network are different w.r.t. different perspectives. In this paper, we propose a new type of query, *perspective-aware top-k similarity query*, to provide more insightful results for users. We efficiently obtain all top-k similar nodes to a given node simultaneously from all perspectives of the network. To accelerate the query processing, several optimization strategies are proposed. Our solutions are validated by performing extensive experiments.

Keywords: Random walk, Multi-relational network, Graph, Proximity.

1 Introduction

Recent years have seen an astounding growth of networks in a wide spectrum of application domains, ranging from sensor and communication networks to biological and social networks [1]. At the same time, a number of important real world applications (e.g. link prediction in social networks, collaborative filtering in recommender networks, fraud detection, and personalized graph search techniques) rely on querying the most "similar" k nodes to a given query node. The measure of "similarity" between two nodes is the proximity between two nodes

* This work was supported by National Basic Research Program of China (973 Program)(No. 2012CB316205), NSFC under the grant No.61272137, 61033010, 61202114, and NSSFC (No: 12&ZD220). It was partially done when the authors worked in SA Center for Big Data Research in RUC. This Center is funded by a Chinese National 111 Project Attracting.

S.S. Bhowmick et al. (Eds.): DASFAA 2014, Part II, LNCS 8422, pp. 171–187, 2014.

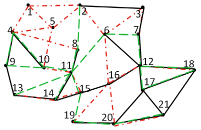


Fig. 1. A coauthor network G

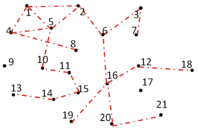


Fig. 2. The graph G_{DB} from perspective of DB

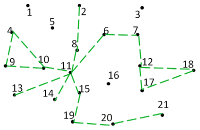


Fig. 3. The graph G_{DM} from perspective of DM

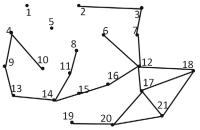


Fig. 4. The graph G_{IR} from perspective of IR

based on the paths connecting them. For example, random walk with restart (RWR) [2], Personalized PageRank (PPR) [3], SimRank [4], and hitting time [5] are all such kinds of measures. These measures are computed based on the structure of graphs.

The question, computation of the most "similar" k nodes to a given query node, has been studied in these researches [2,6,7,8]. Although their works are excellent, they did not consider the query **under a specific viewpoint**. A query, top k similar authors w.r.t. *Jiawei Han* **in the database field**, is more interesting and useful than the query, that without the viewpoint, for people who are interested in the research of database.

Actually, as mentioned in [9,10,11,12,13], there may exist different kinds of relationships between any two nodes in real networks. For example, in a typical social network, there always exist various relationships between individuals, such as friendships, business relationships, and common interest relationships [10]. So different relationships can be modeled by different graphs in multi-relational networks. And these different graphs reflect relationships among objects from different perspectives. Correspondingly, the top k similarity query based on these graphs will return different answers.

Here an example is given:

Example 1. A network G of coauthor relationships is showed in Figure 1. In the figure, relationships are extracted based on the publish information from database (DB), data mining (DM), and information retrieval (IR) fields. Relationships of coauthor in different fields are denoted by different colors (DB, DM, and IR are denoted by red, green, and black edges respectively in Figure 1). The graph showed in figure 2 is modeled based on coauthor relationships of DB field. Similarly, Figure 3 and Figure 4 show the graphs from DM and IR perspective respectively. $G = G_{DB} \bigcup G_{DM} \bigcup G_{IR}$ is also considered as the graph from perspective of DB or DM or IR. Obviously, the corresponding structure from different perspective is different for G. The graph G' in figure 5 reflects the coauthor relationship among authors without considering the specific research field, on which the traditional top-k query is performed. G' is the corresponding simple graph of G.

Given a query node q, if we want to know the most "similar" nodes w.r.t. q from DB perspective, the result will be determined by the graph in figure 2 (rather than G or G'). Given the query node 4, Table 1 shows its top-3 similar nodes from different perspectives. The result of the traditional query based on G' (without considering a specific viewpoint) is (5, 10, 1), while the result is

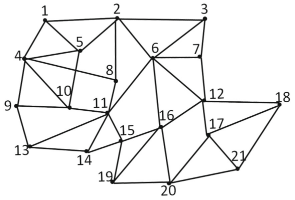


Fig. 5. The graph G' without considering perspectives in the network G

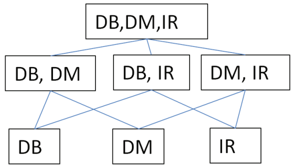


Fig. 6. All Perspectives

(1, 5, 8) from perspective DB, from perspective DM result is (10, 9, 11), and from perspective DB or DM or IR the result is (10, 9, 5). □

From the example, the perspective-aware top-k search provides more insightful information to users. It is used in a lot of applications. For example, in e-commerce activities, if product a is frequently co-purchased with product b, then we construct a product co-purchasing network which contains an edge (a, b). For a young customer who bought a product a, recommending top-k most similar products w.r.t a from the perspective of young people to the customer is a targeted marketing effort in contrast with that without considering the viewpoint. Sometimes people desire to query the most "similar" k nodes w.r.t. a query node from different perspectives rather than under a specific viewpoint. For the network G of coauthor relationship in figure 1, its perspectives and the relationships among them are showed in figure 6. It is more interesting and useful how the result of the query varies as the perspectives change from bottom to top along the relationship showed in figure 6. For instance, the corresponding results of the query w.r.t. node 5 are showed in table 1 when the perspectives changed along DB→(DB DM),(DB IR)→(DB DM IR). The corresponding query results are almost same although perspectives are different. This information is interesting and it motivates us to think that the person (node 5) may be a pure database researcher. The assumption can be further verified by comparing G_{DB} (figure 2) with G (figure 1): the person (node 5) collaborates merely with other researchers in database field, and most of his coauthors also collaborate with other researchers in database field. Therefore, by computing the query from different perspectives, we can explore the relationship between query node and perspectives.

From the above discussion, the advantages of perspective-aware top-k search are the following:

Table 1. Top 3 query from different perspective on G using RWR measure

Query node	Perspective	Top-3 nodes	Query node	Perspective	Top-3 nodes
4	DB	1, 5, 8	5	DB	4, 1, 2
	DM	10, 9, 11		DB DM	4, 1, 2
	DB DM IR	10, 9, 5		DB IR	4, 1, 2
	G′	5,10,1		DB DM IR	4, 1, 2

- We can retrieve the most "similar" k nodes to a given query node from any specific viewpoint. Some results of the query can not be achieved by the traditional top-k query.
- We can discover the relationship between query node and perspectives. By exploring results of the query from different perspectives, we can find how the results change with perspectives. These are useful to comprehend both the query node and corresponding results for users.

In the paper we choose RWR as our proximity measure. RWR is a given node's personalized view of the importance of nodes on the graph. This is compliant with our problem: given a query node we want to find k most similar nodes based on the view of the query node.

To the best of our knowledge, our work is the first one to propose the perspective-aware top-k query in multi-relational networks. Due to the complexity for the computation of similarity and the huge size of the graphs, the challenge of the problem is whether we can traverse once to efficiently obtain all top-k nodes about all perspectives simultaneously to the query node. To address the challenge, we design a concise structure of graphs which contains information of all perspectives, then we accelerate speed of the query by merely searching the neighborhood of the query node.

The contributions of this paper are summarized as below:

1. We define a new type of query, perspective-aware top-k query, in multi-relational networks. The query can provide more meaningful and rich information than the traditional top-k query.
2. RWR is adopted as the measure of similarity. To accelerate the query processing, the corresponding bounding of proximity is given.
3. We propose an efficient query processing algorithm. By designing a concise data structure of graphs and with the help of boundings of the proximity, we merely traverse once the neighborhood of a query node to obtain all its top-k nodes simultaneously from all perspectives. Also, we can achieve top-k nodes from any specific perspective.

Related Work. Recently there are several works [2,6,7,8] based on link-based similarity measures to compute the most "similar" k nodes to a given query node. Theses algorithms are excellent but they did not consider the situations that perspective-aware top-k query.

Graph OLAP [14,15] provide a tool which can view and analyze graph data *from different perspectives*. The idea of Graph OLAP inspired our works. However Graph OLAP is fundamentally different from our problem. Vertex-specific attributes are considered as the dimensions of a network for Graph OLAP. We consider features of whole graph as perspectives. From different perspectives the structure of graph is different.

As dicussed in [11], the multi-relational network is not new. Some researches about multi-relational networks mainly focus on the community mining [9,10,12]). [13] gave the basis for multidimensional network analysis.

2 Problem Formulation

Multi-relational networks are modeled by multigraphs. For the sake of simplicity, we only consider undirected multigraphs and these can be easily extended to directed multigraphs. In the paper, all discussions are based on the following model and definitions.

A multigraph is denoted as $G =< V, E, \widetilde{F} >$ where V is a set of nodes; E is a set of labeled edges; $\widetilde{F}$ is a set of base perspectives: $\widetilde{F} = \{f_1, f_2, ..., f_m\}$. $(u, v, f) \in E$ ($u, v \in V$ and $f \in \widetilde{F}$) means there is a relationship between u and v from perspective f. Each pair of nodes in G is connected by at most $|\widetilde{F}|$ possible edges.

Definition 1. *From any perspective* F*($F \subseteq \widetilde{F}$ and $F \neq \varnothing$), the corresponding graph is an edge-induced subgraph $G(S)$ where $S = \{(u, v, f)|f \in F \wedge (u, v, f) \in E\}$. The subgraph $G(S)$ is called* ***perspective graph*** *of F.*
If $|F| = 1$, the corresponding subgraph is a ***base perspective graph****. The graph G is called* ***top perspective graph****.*

For a base perspective $f \in \widetilde{F}$, the corresponding base perspective graph is denoted by G_f. Based on definition 1 we conclude that $G = \bigcup_{f \in \widetilde{F}} G_f$.

Given a query node q and a number k, from perspective of F ($F \subseteq \widetilde{F}$ and $F \neq \varnothing$), the result of query, *the top-k similarity nodes of* q , is $T_k(q) = \{t_1, \ldots, t_k\}$ iif similarity score $P(q, t_i) \geq P(q, t)$ ($\forall t \in V(G(S))/T_k(q)$) on the graph $G(S)$ where $S = \{(u, v, f)|f \in F \wedge (u, v, f) \in E\}$.

Problem statement (On Perspective-Aware Top-k Similarity Search): Given a query node q and a number k, return all lists of top-k similar nodes of q from all the different perspectives.

From above statements and analyses, the number of corresponding perspective graphs is $2^m - 1$ where m is the number of base perspectives. So the size of results of the query is $2^m - 1$ from all different perspectives.

In practice, the set of base perspectives $\widetilde{F}$ is determined by domain experts.

3 Proximity Measure

A multigraph is consider as a weighted graph where weight A_{uv} is the number of edges (u, v). So similarity measures based on random walk can be defined in multigraphs which are represented by weighted graphs. RWR is same as PPR when the preference set of PPR contains merely one node q. According to the work [3], the RWR score between q and v, denoted by $r(q, v)$, is:

$$r(q, v) = \sum_{t : q \sim v} P(t) c (1 - c)^{l(t)} \tag{1}$$

where $c \in (0, 1)$ is called a constant decay factor, the summation is taken over all paths t (paths that may contain cycles) starting at q to **random walk** and ending at v, the term $P(t)$ is the probability of traveling t, and $l(t)$ is the length of path t.

Random Walks on Multigraphs: random walking on a multigraph is considered as random walking on the corresponding weighted graph where weight A_{ij} is the number of edges (i, j). Given a multigraph $G(V, E)$, A is its adjacency matrix, where A_{ij} is the number of edges (i, j) if edge $(i, j) \in E$ otherwise $A_{ij} = 0$. $d_i = \sum_i A_{ij}$ is the degree of node i on the multigraph.

Based on the work [16], the transition probability of from node i to node j is:

$$p'(i, j) = A_{ij}/d_i \quad . \tag{2}$$

So given any path $t : (w_1, w_2, ..., w_n)$, the probability of random surfer traveling t, *P(t)*, is $\prod_{i=1}^{n-1} \frac{A_{w_i w_{i+1}}}{d_{w_i}}$.

4 Naïve Method

As discussed in the previous section, each perspective graph is considered as a weighted graph. Given a query node q, the naïve method of perspective-aware top-k search consists in the computation of the similarity scores between query node and other nodes on each perspective graph respectively.

The naïve method is an inefficient method due to heavy overheads in both time and space. The time of fast executing a top-k query on a single graph is $O(n^2)$ and expensive [2]. Given a query node, we must execute the top-k query on each perspective graph and need to store the $2^m - 1$ perspective graphs adopting the method described in the work [2].

5 Top-*k* Algorithm

The naïve method is infeasible in practice because of heavy overheads in both time and space. So at this section we devise a concise data structure and give the bounding of the proximity to address the challenge. With aid of the data structure and bounding, we propose a new method to obtain all lists of top-k similarity nodes w.r.t a query node by merely searching the neighborhood of the query node.

5.1 Data Structure of Graph with All Perspectives Information

The goal of the data structure is: starting from a query node we traverse once the multigraph to compute RWR scores about all perspectives simultaneously, avoiding storing and traversing each perspective graph separately.

Given an edge we distinguish which perspective graph the edge belongs to. Since the size of base perspectives is m, we adopt m bits to denote the perspective graph the edge belongs to. Iff an edge (a, b) only belongs to a base perspective graph of G_{f_i}, ith bit of the bits is one and the rest is zero. If an edge (a, b) belongs to several base perspective graphs: $(a, b) \in \bigcap_{i=i_1}^{i_t} G_{f_i}$, each corresponding ith $(i = i_l, 1 \leq l \leq t)$ bit of the bits is one and the rest is zero. So edges (src, dst)

are denoted by a **triplet** $(src, dst, perspectiveFlag)$, where src and dst are nodes in the multigraph, and ***perspectiveFlag*** is the bits.

Analogously, we also adopt m bits to represent perspective graph $\bigcup_{i=i_1}^{i_t} G_{f_i}$, each corresponding ith $(i_1 \leq i \leq i_t)$ bit of the bits is one and the rest value of the bits is zero. The bits is denoted by ***persIdent*** which represents corresponding perspective graph.

Therefore, given an edge (a, b, e), any perspective graph $\bigcup_{i=i_1}^{i_t} G_{f_i}$ and corresponding value of $persIdent$ is p we have:

$$\begin{array}{l} (a, b, e) \in \bigcup_{i=i_1}^{i_t} G_{f_i}, \text{ if } (e \text{ BITAND } p) \text{ != } 0 \\ (a, b, e) \notin \bigcup_{i=i_1}^{i_t} G_{f_i}, \text{ otherwise} \end{array} \quad (3)$$

, where $BITAND$ is bitwise AND operator. The weight of the edge is the number of non-zero bit in $(e$ BITAND $p)$ on the perspective graph $\bigcup_{i=i_1}^{i_t} G_{f_i}$.

A graph G contains information of all perspective graphs when each edge of G is represented by the **triplet** format.

5.2 Bounding RWR

Using Eq.(1), we must traverse all paths which start from q and end at v to obtain the similarity. However to obtain all the paths is time consuming. At the same time, $P(t)(1-c)^{l(t)}$ decreases exponentially with increasing of $l(t)$. This means when a random-walk path is more longer it contributes less to value of $r(p, v)$ in Eq.(1). Based on the observation, the following formula is utilized to approximate $r(q, v)$:

$$r_d(q, v) = \sum_{\substack{t:q\sim v \\ l(t) \leq d}} P(t)c(1-c)^{l(t)} \quad . \quad (4)$$

Obviously $r_d(q, v) \leq r(q, v)$ and $r(q, v) = \lim_{d \to \infty} r_d(q, v)$. It is unpractice to accurately compute $r(q, v)$. Therefor we compute $r_z(q, v)$ instead of $r(q, v)$:

$$|r_z(q, v) - r(q, v)| < \varepsilon \quad (5)$$

where ε controls the accuracy of $r_z(q, v)$ in estimating $r(q, v)$, and z is the minimum value that satisfies the inequation.

We fast compute $r_{d+1}(q, v)$ from $r_d(q, v)$ by the following iteration :

$$r_{d+1}(q, v) = r_d(q, v) + c(1-c)^{d+1} \sum_{\substack{t:q\sim v \\ l(t)=d+1}} P(t) \quad . \quad (6)$$

Using Eq.(6), we efficiently compute $r_{d+1}(q, v)$ expanding one step from paths, whose length is d, when $r_d(q, v)$ has been obtained. The summation $\sum_{\substack{t:q\sim v \\ l(t)=d+1}} P(t)$ is computed by the algorithms 2 at section 5.

E.q. (6) is the lower bound of $r(q, v)$. It was shown in [17], that at dth iteration the upper bound of RWR is

$$r_d(q, v) + \varepsilon_d \quad (7)$$

, where $\varepsilon_d = (1-c)^{d+1}$. The upper bound is very coarse because it is obtained in the extreme case that $\sum_{\substack{t:q\sim v\\ l(t)=i}} P(t)$, which is the probability that a surfer at q can reach v at the ith step, is 1. In most cases, $\sum_{\substack{t:q\sim v\\ l(t)=i}} P(t)$ is far less than 1. To attain more tight upper bound we assume that at the ith $(i \geq d+1)$ step there is only one path along which a surfer at q can reach v and the probability of the path is estimated by a large value, which is less than 1.

Upper bound of RWR is introduced by the following proposition:

For all paths $t : (w_1, w_2, ..., w_n)$ which are obtained by breadth-first traversing from w_1, at dth iteration the maximum transition probability is $p_d = MAX\{p'(w_d, w_{d+1})\}$ where $p'(w_d, w_{d+1})$ is the transition probability from node w_d to node w_{d+1}.

Proposition 1. *At dth iteration the upper bound of RWR is:*

$$r_d(q,v) + \varepsilon_d \tag{8}$$

, where $\varepsilon_d = (1-c)^{d+1}\prod_{i=1}^{d} p_i$ *and* $p_i = MAX\{p'(w_i, w_{i+1})\}$.

Proof. According to Eq.(1): $r(q,v) = \sum_{t:q\sim v} P(t)c(1-c)^{l(t)} = \sum_{\substack{t:q\sim v\\ l(t)\leq d}} P(t)c(1-c)^{l(t)} + \sum^{\infty}_{\substack{t:q\sim v\\ l(t)\geq d+1}} P(t)c(1-c)^{l(t)} = r_d(q,v) + \sum^{\infty}_{\substack{t:q\sim v\\ l(t)\geq d+1}} P(t)c(1-c)^{l(t)}$. For any path $t =< w_1, \ldots, w_{n-1}, w_n >$ which is obtained by breadth-first traversing from w_1 where $q = w_1$ and $v = w_n$, $P(t) = \prod_{i=1}^{n-1} p'(w_i, w_{i+1}) \leq \prod_{i=1}^{d} p_i \prod_{i=d+1}^{n-1} p'(w_i, w_{i+1}) \leq \prod_{i=1}^{d} p_i$ at dth iteration because the transition probability $p'(w_i, w_{i+1}) \leq 1$.

Thus at dth iteration: $\sum^{\infty}_{\substack{t:q\sim v\\ l(t)=d+1}} P(t)c(1-c)^{l(t)} \leq \sum^{\infty}_{\substack{t:q\sim v\\ l(t)=d+1}} ((\prod_{i=1}^{d} p_i)c(1-c)^{l(t)}) = c(\prod_{i=1}^{d} p_i)(\sum^{\infty}_{\substack{t:q\sim v\\ l(t)=d+1}} (1-c)^{l(t)}) = (1-c)^{d+1}\prod_{i=1}^{d} p_i = \varepsilon_d$ according to $\sum^{\infty}_{\substack{t:q\sim v\\ l(t)=d+1}} (1-c)^{l(t)} = \frac{(1-c)^{d+1}}{c}$. □

$r_d(q,v)$ and $r_d(q,v)+\varepsilon_d$ is lower bound and upper bound of $r(q,v)$ respectively at dth iteration. Given two nodes v and v', $r(q,v) < r(q,v')$ if $r_d(q,v) + \varepsilon_d < r_d(q,v')$. So using bounding of RWR we accelerate the top-k query in the paper.

5.3 On Perspective-Aware Top-*k* Similarity Search

Given a set of base perspectives $\widetilde{F} = \{f_1, f_2, ..., f_m\}$ and a multigraph $G =< V, E >$ which is represented by the data structure described in section 5.1, starting at a query node q, we do a breadth-first traverse to visit remaining nodes. At dth iteration, when a node v is visited, which perspective graphs the node belongs to is judged. Then we compute $r_d(q,v)$ and its upper value on each corresponding perspective graph. Each node v is associated with a list to store the values of $r_d(q,v)$ and its upper values.

After dth iteration, we then find a set of k nodes with the highest scores of lower bounds. Let T_k be the kth largest score on the corresponding perspective graph which the query node belongs to. We terminate the query and obtain the final result of the top-k query on the corresponding perspective graph based on following theorem:

Theorem 1. *At dth iteration, R is a set of k nodes with the highest scores r_d of lower bounds w.r.t. the query node q on any respective graph G', T_k is the kth largest r_d, and P is a set of nodes which already are visited by traversing on the G'. R is the exact theoretical top-k set w.r.t q on G' if one of following conditions is true:*

- *the value of its upper bound is less than T_k for any node $p \in P \setminus R$ and $T_k > \varepsilon_d$*
- *$\varepsilon_d \leq \varepsilon$.*

Proof. For any $p \in P \setminus R$, $r(q,p) < T_k$ and p is not in the set of the top-k nodes because its upper bound value is less than T_K. For any $v \in V(G') \setminus P$, v is still not visited so $r_d(q,v) = 0$. According to proposition 1, $r(q,v) \leq r_d(q,v) + \varepsilon_d = \varepsilon_d < T_k$ and v is not in the set of the top-k nodes. Therefore R is the result.

If $\varepsilon_d \leq \varepsilon$, according to inequality (5), $r_z(q,v)(\forall v \in P)$ is achieved and considered as final value of $r(q,v)$. While $r_z(q,v')$ $(\forall v' \in V(G') \setminus P)$ is estimated as 0. So R is the result. □

Our method merely considers the neighborhood of the query node and avoids searching the whole graph.

5.4 Optimization Strategies

By relaxing terminating condition of traversing we improve the query speed at the expense of accuracy. In contrast to theorem 1, we terminate the query on the corresponding respective graph if $T_k > \varepsilon_d$ or $\varepsilon_d \leq \varepsilon$ is true. Although the new conditions do not guarantee accuracy in theory, in practice results of the query are almost accurate due to ε_d approaches 0 drastically as the increasing of d.

On the other hand, based on the general idea of the algorithm, we must sort the visited nodes for trying to obtain top-k nodes on each corresponding perspective graphs at each iteration. These would lead to overhead. So we adopt following strategy: at each iteration we merely try to obtain top-k nodes on **top perspective graph** until the corresponding T_k is greater than ε_d before we try to obtain top-k on other perspective graphs.

The bound of RWR accelerates the top-k query. We desire a more tight bound of RWR with the purpose of getting more faster response time. As discussed in subsection 5.2, for all paths $t : (w_1, w_2, ..., w_n)$ we can achieve transition probability $p'(w_i, w_{i+1}) \quad (1 \leqslant i \leqslant d-1)$ at i iteration, and we approximate the transition probability $p'(w_i, w_{i+1}) \quad (d \leqslant i \leqslant n-1)$ by average value, $\widetilde{p}$, of values $p'(w_i, w_{i+1}) \quad (1 \leqslant i \leqslant d-1)$. Then at dth iteration, the upper bound of RWR is:

$$r_d(q,v) + \varepsilon_d \tag{9}$$

, where $\varepsilon_d = \frac{c\widetilde{p}(1-c)^{d+1}}{1-\widetilde{p}(1-c)} \prod_{i=1}^{d} p_i$.

Algorithm 1. Top-k similarity queries from different perspectives

```
   input  : Graph $g$,$c$,$v$, $k$
   output: $2^m - 1$ top-$k$ ranking lists of $v$ corresponding to each perspective
 1 Set $pathProb$←{1.0,1.0,...,1.0};
 2 push pair ($v$, $pathProb$) into queue $que$;
 3 push ($-1$, $pathPob$) into queue $que$;
 4 $degree$←$g.getDegree(v)$;$i$←1;
 5 for $m \leftarrow 1$ to $sizeOfPerspectives$ do
 6     if $degree[m] \neq 0$ then
 7         $actualSize$←$actualSize + 1$;
 8         $perspective[m]$←1;
 9 while $obtained.size() < actualSize$ do
10     ($currentNode$, $pathProb$)←$que.front()$;
11     $que.pop()$;
12     if $currentNode == -1$ then
13         $i$←$i + 1$;
14         $rwrScore$ ←calRWR($g$,$que$,$queTemp$,$c$);
15         if $obtained.has(top\ perspective)\ is\ false$ then
16             sort $rwrScore$[top perspective] to obtain top k nodes;
17             if $its\ T_k > \varepsilon_i\ or\ \varepsilon_i < \varepsilon$ then
18                 $obtained.insert$(top perspective)
19         if $obtained.has(top\ perspective)\ is\ true$ then
20             try to obtain top-$k$ on other perspective graphs, and insert a identify of a graph into
               $obtained$ if the top-$k$ result obtained on corresponding graph ;
21         push ($-1$, $pathProb$) into $que$;
22         clear $queTemp$;
23         continue;
24     else
25         walkToNeighbors($g$,$currentNode$,$pathProb$,
26         $perspective$,$queTemp$);                                  // update $queTemp$
27 return $result$;
```

5.5 The Details of the Algorithm

In this subsection we examine the details of the algorithm adopting optimization strategies.

Algorithm 1 describes the main framework of the algorithm. Starting at a query node v we do a breadth-first traversal to obtain perspective-aware top-k nodes w.r.t v on a graph G.

In the algorithm, the current visiting node is allocated a list *pathProb* where each entry of the list is the probability of paths from the query node to the visiting node on corresponding perspective graph. In line 1, we first initialize each entry of *pathProb* to be one. In lines 5∼8 we judge which perspective graph the query node belongs to. *actualSize* is the total size of perspective graphs the query node belongs to.

In line 12, if current node popped from queue is -1 : we call method *calRWR* (line 16) to compute RWR scores of the nodes visited at ith iteration and update the queue, at each iteration we merely try to obtain top-k nodes on the **top perspective graph** until the corresponding T_k is greater than ε_i or $\varepsilon_i \leq \varepsilon$, then we try to obtain top-k nodes on the others perspective graphs (lines 15∼20). If the result is achieved on a corresponding graph, then the identity of the graph is inserted into *obtained*.

Algorithm 2. walkToNeighbors

```
input  : Graph $g$,$currentNode$,$pathProb$,$perspective$
output: $queTemp$
// update $queTemp$
1 $i \leftarrow currentNode$; $degree \leftarrow g.getDegree(i)$;
2 foreach a neighbors j of i do
3     for $m' \leftarrow 1$ to $sizeOfPerspectives$ do
4         if $perspective[m'] \neq 0$ and (eflag(j) & m') and $pathProb[m'] \neq 0$ then
5             $probValue \leftarrow \frac{pathProb[m'] \times A_{ij}}{degree[m']}$;
6             $queTemp[j][m'] \leftarrow queTemp[j][m'] + probValue$;        // update $queTemp$
7 return;
```

If the current node is not -1 we call method *walkToNeighbors* (line 24) to visit its neighbors and calculate probability of paths from query node to the neighbors.

Algorithm 3. calRWR

```
input  : $g$,$que$,$queTemp$,$v$,$c$
output: $que$, $rwrScore$
// update $que$, $rwrScore$
foreach element i of queTemp do
    foreach element j of queTemp[i] do
        $rwrScore[j][i] \leftarrow rwrScore[j][i] + queTemp[i][j] \times c \times (1-c)^{step}$;
        $temp[j] \leftarrow queTemp[i][j]$;
    push $(i, temp)$ into $que$;
    clear temp;
return;
```

Algorithm 2 is the method *walkToNeighbors* mentioned above. The conditions (line 4) are **key factors** that we can traverse once on the graph to simultaneously compute RWR scores about all perspectives starting from the query node. The condition $perspective[m'] \neq 0$ is tested to judge whether or not the query node belongs to corresponding perspective graph whose identifier is m'. The condition $eflag(j)$ & m' refers to Eq.(3). The last condition, $pathProb[m'] \neq 0$, is true means there exits at least one path from query node to node i on the perspective graph of m'. We compute the probability of paths that start at the query node and via the current node end at its neighbors on perspective graph of m'. Then we accumulate the probability of the paths whose length is current iteration number (line 6). In a word the algorithm compute the summation $\sum_{\substack{t:q \sim v \\ l(t)=k+1}} P(t)$ of the Eq.(6) on each corresponding perspective graph the query node belongs to. And *queTemp* contains all nodes visited at current iteration and the probability of paths from query node to those nodes.

Algorithm 3 (the method $calRWR$ in the algorithm 1) compute RWR score between query node and each node of $queTemp$ on each corresponding respective graph based on Eq.(6), and then update the queue.

Time complexity of the algorithm is $max\{O(DNM), O(DN'log_2N')\}$ where D is maximum iterations, N is average number of visited nodes at each iteration, M is total number of perspective graphs which the query node belongs to, and N' is the average number of all visited nodes.

6 Experimental Study

In this section, we report our experimental studies to evaluate the effectiveness and efficiency of the proposed perspective-aware top-k query. We implemented all experiments on a PC with i3-550 CPU, 4G main memory, running windows 7 operating system. All algorithms are implemented in C++. The default values of our parameters are: c = 0.2, and $\varepsilon = 10^{-6}$. In the experiments the accurate method, which is described at section 5.3 and adopts ε_d in Eq.(7), is used to test effectiveness of our query, the method adopting ε_d in Eq.(8) and the method adopting the optimization strategies are denoted as RWR-approxity1 and RWR-approxity2 respectively .

6.1 Experimental Data Sets

Table 2. Major conferences chosen for constructing the co-authorship network

Area	Conferences
DB	SIGMOD, PVLDB, VLDB, PODS, ICDE, EDBT
DM	KDD, ICDM, SDM, PAKDD, PKDD
IR	SIGIR, WWW, CIKM, ECIR, WSDM
AI	IJCAI, AAAI, ICML, ML, CVPR, ECML

We conduct our experiments on two real-world data sets. The **DBLP**[1] Bibliography data is downloaded in September, 2012. Four research areas are considered as base perspectives: database (DB), data mining (DM), information retrieval (IR) and artificial intelligence (AI). We construct the network based on publication information from major conferences in the four research areas which are showed in table 2. The number of nodes and edges in the network is 38,412 and 110,486 respectively. The **IMDB**[2] data was extracted from the Internet Movies Data Base (IMDB). Movies contained in the data were released at the time between 1990 and 2000. We construct the network as following: we choose eight types of genres as base perspectives, including Action, Animation, Comedy, Drama, Documentary, Romance, Crime and Adventure. Assuming T is any one of the eight genres, one of the relationships of two movies is T if the two movies have same genre T and they also have a same actor/actress or a same writer or a same director at least. There are 51,532 vertices and 2,220,321 edges in the network.

[1] http://www.informatik.uni-trier.de/~ley/db/index.html

[2] http://www.imdb.com/interfaces

Table 3. Top-5 similar query from different perspectives on DBLP

Query author	Perspective	Top-5 authors
Jennifer Widom	IR	Robert Ikeda Semih Salihoglu,Glen Jeh Beverly Yang,Hector Garcia-Molina
	DM	Glen Jeh
	DB DM IR AI	Robert Ikeda Semih Salihoglu,Glen Jeh Hector Garcia-Molina,Jeffrey D. Ullman
	without perspectives	Hector Garcia-Molina Jeffrey D. Ullman,Shivnath Babu Robert Ikeda,Arvind Arasu
Jiawei Han	IR	Tim Weninger Xin Jin,Jiebo Luo Yizhou Sun,Ding Zhou
	DB DM IR AI	Zhenhui Li Xiaofei He,Deng Cai Jian Pei,Xifeng Yan
	without perspectives	Xifeng Yan Jian Pei,Yizhou Sun Hong Cheng,Philip S. Yu
Jim Gray	DB * * *	Alexander S. Szalay Peter Z. Kunszt,Ani Thakar Betty Salzberg,Michael Stonebraker

We also generate a series of synthetic data sets to evaluate the performance. All the top-k queries are repeated 200 times and the reported values are average values.

6.2 Effectiveness Evaluation

We evaluate the effectiveness of perspective-aware top-k query by comparing it with the traditional query. The difference between our query and the traditional query lies in structure of networks. Although the principle of their proximity measure is the same, our query contains viewpoints while traditional query does not.

In table 3 top-5 similar authors w.r.t given authors based on RWR from different perspectives are showed. And in the table the corresponding query results of *without perspectives* actually are the results obtained by the traditional query that without considering any specific viewpoint.

From perspective of DM, the similar authors w.r.t *Jennifer Widom* are *Glen Jeh*. However *Glen Jeh* is not in the top-5 candidates list that without considering any specific viewpoint (table 3). From perspective of (DB DM IR AI) the corresponding first three similar authors all collaborated with *Jennifer Widom* in two different fields. In contrast, the authors in the results of the traditional query merely collaborated with *Jennifer Widom* in DB field. The top-5 similar authors from perspective of IR are important for peoples interested in IR because the first three authors collaborate with *Jennifer Widom* in IR field whereas no one in the result of the traditional query collaborate with *Jennifer Widom* in IR field.

There is similar situation for querying *Jiawei Han* as showed in table 3. From perspective (DB DM IR AI) the corresponding first three similar authors collaborated with *Jiawei Han* in the all four field whereas the authors in the result of traditional query collaborated with *Jiawei Han* in three fields at most.

Let the order of basic perspectives is (DB DM IR AI), * means the corresponding base perspective exist or does not exist and 0 means the corresponding base perspective does not exist. From table 3 we conclude *Jim Gray* focus on the research of only one field DB because the top-5 candidate list is almost same from a group of perspectives (DB * * *).

Therefore our perspective-aware top-k query can provide more meaningful and insightful results in contrast to traditional query.

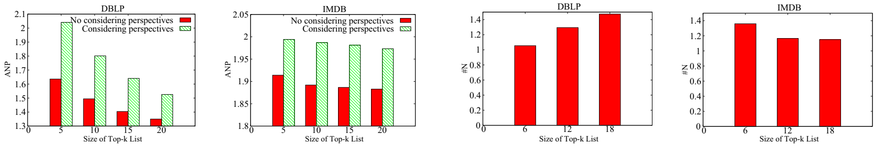
Examples mentioned above are based on only several authors, so we further evaluate the effectiveness of the perspective aware top-k query randomly choosing 200 query nodes. The two real data sets are used in this subsection. For IMDB, we choose first four types of genres as base perspectives. For the query node q, R is its top-k result on the top perspective graph.

Let $NP(q,p)$ denote the $\underline{N}$umber of $\underline{P}$espective relationships between q and p. For example, $NP(q,p) = 3$ means q and p co-published papers in 3 different area in DBLP. Given a metric $ANP = \frac{\sum_{p \in R} NP(q,p)}{|R|}$. We test whether results of our query reflect more perspectives information than the results of the traditional query by comparing ANP of our query on top perspective graph with ANP of the traditional query. The larger ANP is, the more perspective information our query can reflect. As illustrated in figure 7(a), our query considers more perspectives information than the traditional query does.

Given a query node q, $R' \quad (R' \neq R)$ is its top-k result on any perspective graph. Then we evaluate whether the new query can provide rich and insightful results by the following metric: $\#N = |\{R'| \quad (|R'| - |R \cap R'|) \geq \frac{1}{3}|R'|\}|$. $\#N$ is the number of perspective graphs, on which at least one third nodes in the results are different with the nodes in the results on top perspective graph for a given query node. For example showed in table 3, the nodes in query results from perspective IR almost are different with the nodes in the results on the top perspective graph for *Jiawei Han*, then we know who are most similar to *Jiawei Han* in IR field while these peoples are not contained in the results on top perspective graph. The experimental result based on 200 query nodes is showed in figure 7(b) and it verified the effectiveness of the new query.

6.3 Efficiency Evaluation

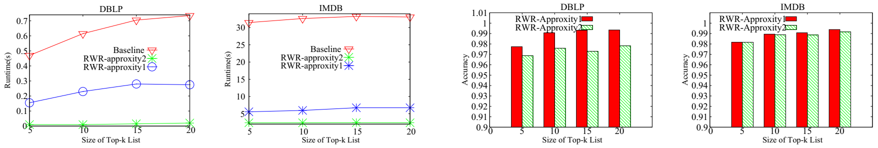
In this section, we evaluate the efficiency of our perspective-aware top-k query. First we assess the query time of our method in different situations. Then We use P@k (Precision at k) to measure the accuracy of top-k lists based on approximate mehtod by comparing it with the accurate top-k lists. At last we evaluate the efficiency of bounding of our proximities on synthetic data because bounding of RWR is adopted to accelerating speed of the query. Figure 8(a) shows the query time of the top-k query for different k values on the two real networks, where we choose the first four types of genres as base perspectives for IMDB. As illustrated in figure 8(a) approximate methods are much faster than the accuracy method,



(a) ANP of new query on top perspective graph vs. ANP of traditional query.

(b) Average #N.

Fig. 7. Effective of the new query



(a) Runtime of accurate method vs. runtime of approximate method.

(b) Accuracy ratio.

Fig. 8. Efficiency of the new query

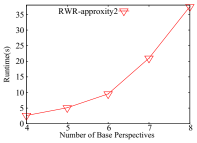


Fig. 9. Query time vs. number of base perspectives on IMDB

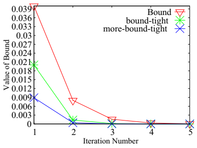


Fig. 10. Bounds on scale-free graph

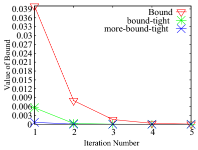


Fig. 11. Bounds on Erdős Rényi graph

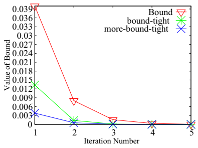


Fig. 12. Bounds on random regular graph

while approximate method achieves a very high precision (>96.5%) as showed in figure 8(b) .

We also test how the number of base perspective affects the runtime of the RWR-approxity2 method on IMDB data set. As showed in figure 9 the runtime becomes large as the number of base perspective increases. Our method is efficient for the data set with a small number of base perspectives (<8). Our future work will focus on the top-k query when the number of base perspectives is large (≥8).

ε_d in Eq.(8) and (9) are denoted as *bound-tight* and *more-bound-tight* respectively. The bound of PPR (Eq.(4) in [17]) is a baseline and denoted as *bound* to compare with our bounds, where RWR is same as PPR because its preference set of PPR is itself for a query node q.

For simplicity we evaluate the efficiency of bounding of our proximities on simple graphs. Three type synthetic graphs that are used to test the bounds are scale-free graph [18], Erdős Rényi graph [19] and random regular graph [19] respectively. The average degree of the three graph is 6, 3 and 5 respectively. And their number of nodes all is 1000. As analyzed in theorem 1, ε_d can quicken the speed of the query if ε_d decreases drastically as iteration number increases. Figures from 10 to 12 show the results of bounds on the three synthetic graphs. The results show that bounds ε_d sharply decline as iteration number increases although the types of graphs are different. Our method is efficient because ε_d becomes very small after 3th iteration based on the results.

7 Conclusions

We have proposed a novel and practical perspective-aware top-k query in multi-relational networks. We not only achieve the most "similar" k nodes to a given query node from any specific viewpoint, but also can observe how the results change with perspectives to full understand query node and the results. With aid of the concise data structure of graphs and bounding of RWR, starting from a query node we can traverse once on the graphs and merely search the neighborhood of the query node to obtain all top-k nodes about all perspectives. Then we accelerated speed of the query by adopting several optimization strategies including tighter bounding of proximity. At last we showed the effectiveness and efficiency at the section of experimental study.

References

1. Zhao, P., Li, X., Xin, D., Han, J.: Graph cube: On warehousing and olap multidimensional networks. In: SIGMOD Conference, pp. 853–864 (2011)
2. Fujiwara, Y., Nakatsuji, M., Onizuka, M., Kitsuregawa, M.: Fast and exact top-k search for random walk with restart. PVLDB 5(5), 442–453 (2012)
3. Jeh, G., Widom, J.: Scaling personalized web search. In: WWW, pp. 271–279 (2003)
4. Jeh, G., Widom, J.: Simrank: A measure of structural-context similarity. In: KDD, pp. 538–543 (2002)
5. Sarkar, P., Moore, A.W., Prakash, A.: Fast incremental proximity search in large graphs. In: ICML, pp. 896–903 (2008)
6. Sarkar, P., Moore, A.W.: Fast nearest-neighbor search in disk-resident graphs. In: KDD, pp. 513–522 (2010)
7. Avrachenkov, K., Litvak, N., Nemirovsky, D.: Quick detection of top-k personalized pagerank lists. In: WAW, pp. 50–61 (2011)
8. Sun, Y., Han, J., Yan, X., Yu, P.S., Wu, T.: Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. PVLDB 4(11), 992–1003 (2011)
9. Ahn, Y.Y., Bagrow, J.P., Lehmann, S.: Link communities reveal multiscale complexity in networks. Nature 466(7307), 761–764 (2010)
10. Cai, D., Shao, Z., He, X., Yan, X., Han, J.: Community mining from multi-relational networks. In: Jorge, A.M., Torgo, L., Brazdil, P.B., Camacho, R., Gama, J. (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, pp. 445–452. Springer, Heidelberg (2005)
11. Rodriguez, M.A., Shinavier, J.: Exposing multi-relational networks to single-relational network analysis algorithms. J. Informetrics 4(1), 29–41 (2010)
12. Berlingerio, M., Coscia, M., Giannotti, F.: Finding and characterizing communities in multidimensional networks. In: ASONAM 2011, pp. 490–494 (2011)

13. Berlingerio, M., Coscia, M., Giannotti, F., Monreale, A., Pedreschi, D.: Foundations of multidimensional network analysis. In: ASONAM 2011, pp. 485–489 (2011)
14. Zhao, P., Han, J., Sun, Y.: P-rank: A comprehensive structural similarity measure over information networks. In: CIKM, pp. 553–562 (2009)
15. Chen, C., Yan, X., Zhu, F., Han, J., Yu, P.: Graph olap: A multi-dimensional framework for graph data analysis. Knowledge and Information Systems 21, 41–63 (2009)
16. Bollobás, B.: Modern Graph Theory. Springer (1998)
17. Sun, L., Cheng, R., Li, X., Cheung, D.W., Han, J.: On link-based similarity join. PVLDB 4(11), 714–725 (2011)
18. Barabasi, A.L., Albert, R.: Emergence of scaling in random networks. Science 286, 509–512 (1999)
19. Bollobás, B.: Random Graphs. Cambridge University Press (2001)