Evaluating Top-\(k\) Meta Path Queries on Large Heterogeneous Information Networks

Zichen Zhu, Reynold Cheng, Loc Do, Zhipeng Huang

The University of Hong Kong
Department of Computer Science
Hong Kong
{zczhu2, ckcheng, zhuang}@cs.hku.hk, dohaloc@gmail.com

Haoci Zhang

Columbia University
Department of Computer Science
New York, U.S.
zhanghaoci@gmail.com

Abstract—Heterogeneous information networks (HINs), which are typed graphs with labeled nodes and edges, have attracted tremendous interest from academia and industry. Given two HIN nodes \(s\) and \(t\), and a natural number \(k\), we study the discovery of the \(k\) most important paths in real time. The paths found can be used to support friend search, product recommendation, anomaly detection, and graph clustering. Although related algorithms have been proposed before, they were primarily designed to return the \(k\) shortest paths from unlabeled graphs. This leads to two problems: (1) there are often many shortest paths between \(s\) and \(t\), and so it is not easy to choose the \(k\) best ones; and (2) it is arguable whether a shortest path implies a more crucial one. To address these issues, we study the top-\(k\) meta path query for a HIN. A meta path abstracts multiple path instances into a high-level path pattern, thereby giving more insight between two nodes. We further study several ranking functions that evaluate the importance of meta paths based on frequency and rarity, rather than on path length. We propose a solution that seamlessly integrates these functions into an A* search framework. The connectivity experiment on ACM dataset shows that our proposed method outperforms state-of-the-art algorithms.

Index Terms—Heterogeneous Information Networks, top-\(k\), meta path

I. INTRODUCTION

Given a graph \(G\) and two nodes \(u\) and \(v\), a top-\(k\) shortest path query retrieves the \(k\) shortest paths between \(u\) and \(v\) [5]. This query, which enables the retrieval of relationship information between two graph nodes, has been studied extensively in various applications, including bibliographical, social, and road networks. The continuous growth of the sizes of these graphs and the need of online performance drives the development of efficient algorithms and data structures (e.g., [2], [9], [10], [13], [20]). The query is natively supported in graph database engines (e.g., Neo4j [3] and Pregel [11]).

In this paper, we ask the question: is top-\(k\) shortest path query the best way for retrieving relationship from a heterogeneous information network (or HIN)? An HIN is essentially a typed graph, whose nodes and edges are tagged with “type labels” to indicate their meanings. Due to its huge amount of information, HINs have recently raised a lot of interest [7], [15], [16], [18], [19]. Fig. 1 illustrates an ACM bibliographical network [15], where each node and edge has a type (e.g., Jiawei Han is an author; IEEE is an affiliation; Jiawei Han belong to IEEE). Suppose that we want to know how these two authors are related in this HIN. A simple way is to evaluate a top-1 shortest path query on it. However, there are plenty of (13) shortest paths between them. Should all these paths be returned to the query user (who may then get a query result with numerous paths)? Is it better to summarize the common properties of these paths (e.g., from these shortest paths, we learnt that the two researchers are close collaborators as they have co-authored many papers)?

To address the above questions, we study how to leverage the type information of a HIN to return the \(k\) best paths to a query user. Particularly, we propose the top-\(k\) meta path query, which computes the \(k\) best meta paths between nodes \(u\) and \(v\). A meta path, first proposed in [18], is essentially a sequence of node types and edge types. Two possible meta paths, derived from the shortest paths between Philip and Jiawei in the HIN of Fig. 1, are:

\[
\begin{align*}
M_1 : & \quad \text{Author} \xrightarrow{\text{belong to}} \text{Affiliation} \xrightarrow{\text{belong to}^{-1}} \text{Author} \\
M_2 : & \quad \text{Author} \xrightarrow{\text{write}} \text{Paper} \xrightarrow{\text{write}^{-1}} \text{Author}
\end{align*}
\]

where \(X \xrightarrow{R^{-1}} Y\) means the reverse direction of edge \(R\) (i.e., \(Y \leftarrow X\)). The above meta paths summaries the underlying paths between Philip and Jiawei. For example, \(M_1\) states that the two authors belong to the same affiliation (i.e., IEEE), while \(M_2\) is based on the fact they have co-authored a number of papers. A meta path abstracts the paths and provides important insights about paths. This allows a query user to
focus on the high-level relationship patterns, rather than on the detailed path instances. Meta paths have also been shown to be useful in node relevance computation [7], [15], [18], graph clustering [19], and recommendation [16].

To evaluate a top-$k$ meta path query, we need to decide the $k$ best meta paths. This is not trivial. Let us again consider the top-1 meta path query for (Philip Yu, Jiawei Han) in Fig. 1. Should we choose $M_1$ or $M_2$? Should we only consider the $k$ shortest meta paths? To answer the first question, notice that although $M_1$ and $M_3$ have the same length, $M_3$ is in fact more important than $M_1$. This is because Philip and Jiawei have co-authored 12 papers, and so $M_3$ summarizes 12 distinct paths between them. On the other hand, $M_1$ only represents one path (i.e., Philip Yu – IEEE – Jiawei Han). Moreover, we found that most authors in the HIN are registered IEEE members, so $M_1$ is not a very special relationship between these two researchers. On the other hand, it is relatively uncommon that two authors have co-authored in more than ten papers. Hence, in this scenario, $M_2$ is arguably a better candidate $M_1$ for the result of the top-1 meta path query.

For the second question (i.e., whether a shorter meta path is better), let us consider two other meta paths between Philip and Jiawei.

\[ M_3 : \text{Author} \xrightarrow{\text{write}} \text{Paper} \xrightarrow{\text{cite}^{-1}} \text{Paper} \xrightarrow{\text{write}^{-1}} \text{Author} \]

\[ M_4 : \text{Author} \xrightarrow{\text{write}} \text{Paper} \xrightarrow{\text{cite}} \text{Paper} \xrightarrow{\text{write}^{-1}} \text{Author} \]

which depict “citation” relationship (i.e., an author’s paper cites another author’s work). We found that $M_3$ and $M_4$ both represent more than 13 paths, and are much more than the single path that instantiates $M_1$. Also, $M_3$ and $M_4$ are less common among other authors than $M_1$. Hence, although $M_1$ has a shorter length than these two meta paths, it may not necessarily be ranked higher than $M_3$ and $M_4$.

Our contributions. From the above examples, we observe that in answering a top-$k$ meta path query, we should not only focus on the shortest meta paths, but should also consider other factors (e.g., number of paths pertaining to the meta paths, and the “uniqueness”, or rarity, of a meta path). We present an A* search algorithm framework, and propose an importance function, which captures several properties of a meta path (e.g., its number of instances (or “support”), degree of commonality, and meta path lengths) for determining its ranking with respect to the query. Because the importance function incorporates existing relevance measures for meta paths, our proposed solution is generic and supports these measures. We also study a new importance, which effectively captures the importance of a meta path. And we have tested our solutions and performed case studies on ACM. The experimental results show that our best solution is effective in answering top-$k$ meta path queries.

Organization. The rest of this paper is as follows. Section II discusses related work. Section III formally defines the top-$k$ meta path query. Section IV introduces some baselines and our solution framework. Section V gives more details of how to compute the importance function. Section VI presents our experimental results. We conclude in Section VII.

II. Related Work

Top-k shortest path query has attracted a lot of interest in the graph database community. These existing research works often design fast algorithms. For example, [5] improves the algorithm performance and reduces the space cost based on the seminal Yen’s algorithm [20]. Besides the classical top $k$-shortest path search, there are also other variants. In [9], an A* Prune algorithm is developed to tackle the $K$ Multiple-Constrained-Shortest-Path problem (KMCSP). The authors in [2] design an iterative bounding approach and two index structures, namely Partial Shortest Tree and Incremental Shortest Path Tree, to reduce the search space for the $K$ Shortest Paths Join (KSPJ). The work of [10] formalizes the $K$ Shortest Paths with Diversity (KSPD), and proposes a general framework to identify $k$-shortest paths, such that the paths are dissimilar with each other, and the total length of the paths is minimized. In addition, [13] proposes an approach to synthesize an approximate algorithm, which can be applied to the Single Source Shortest Path (SSSP) query.

HIN and meta paths. A solution [8] that generates meta paths is parametrized by the maximum length of a meta path $l$, which is hard to set, as it varies among different data sets. We propose a general framework to explore top-$k$ important meta paths between two given nodes in a HIN without specifying the maximum length. To avoid setting $l$, [12] proposes a Forward Stage-wise Path Generation (FSPG) method which adopts the Least-Angle Regression (LARS) [4] to discover meta paths according to the same example node pairs. However, this machine-learning-based solution involves lots of computation and examples, and so it is not suitable for top-$k$ meta paths queries. Apart from these examples-based methods, there is another approach [6] that explores top-$k$ sub-graphs for the whole graph based on a given pattern. However, few papers mentioned how to select the top-$k$ most representative meta paths for a given pair of nodes. [14] mentions a method to mine interesting meta paths in a complex HIN, but this method does not support the top-$k$ meta path query. There is a weight function for meta paths in [19], but it aims to train the weight according to several given meta paths, but it is also not clear how it can be applied in top-$k$ meta path query. There are other weight functions used in [16] and [21] for a meta path, which can then be applied in exploring top-$k$ meta paths, and we will compare these two methods in the paper.

III. Preliminaries

At first we introduce some terminologies and give a formal definition of the top-$k$ meta paths exploration problem.

Definition 1. Heterogeneous Information Network [18] A heterogeneous information network (HIN) is a directed graph $G = (V, E)$ with an object type mapping $\phi : V \rightarrow \mathcal{A}$ and a link type mapping $\psi : E \rightarrow \mathcal{R}$ subject to $|\mathcal{A}| > 1$ and $|\mathcal{R}| > 1$, where $V$ denotes the object set and $E \subseteq V \times V$ denotes the link set, and $\mathcal{A}$ denotes the object type and $\mathcal{R}$ denotes the link type.
Network schema is proposed to better understand meta-level description of a given heterogeneous information network, which can greatly simplify many complicated HINs.

**Definition 2. Network Schema [17]** The network schema is a meta template for a heterogeneous network \( G = (V, E) \) with the object type mapping \( \phi : V \rightarrow A \) and the link type mapping \( \psi : E \rightarrow R \), which is a directed graph defined over objects types \( A \), with edges as relations from \( R \), noted as \( T_G = (A, R) \).

**Definition 3. Meta Path [18]** A meta path \( P \) is path defined on the graph of network schema \( T_G = (A, R) \), and is denoted in the form of \( A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \cdots \xrightarrow{R_l} A_{l+1} \). If there is no ambiguity, we can short it as \( R = R_1 \circ R_2 \circ \cdots \circ R_l \) between \( A_1, A_2, \cdots , A_{l+1} \), where \( \circ \) denotes the concatenation operator on relations.

**Example 1.** A typical HIN is the bibliographic network of ACM and the corresponding network schema is shown in Fig. 2a, where there are six types of entities: paper, venue, author, term, affiliation and conference. Links exist between authors and papers denoting the writing or written-by relations, between papers and venue denoting the publishing or published-in relations, between papers and venues denoting the citing or cited-by relations. Fig. 2b lists 3 very common meta paths denoting the writing or written-by relations, between term, affiliation and conference. Links exist between authors and papers denoting the writing or written-by relations, between term, affiliation and conference.

Here we use lower-case letters(e.g., \( a_i \)) to denote objects in HIN while upper-case letters(e.g., \( A \)) to denote object types in \( T_G \). We say a concrete path \( p = (a_1, a_2, \ldots , a_{l+1}) \) between objects \( a_1 \) and \( a_{l+1} \) in network \( G \) follows the meta path \( P \), or say it is a path instance of the relevant path \( P \), which can be noted as \( p \in P \), if \( \forall a_i \in V, \phi(a_i) = A_i \) and \( \forall e_i = (a_i, a_{i+1}) \in E, \psi(e_i) = R_i \) in \( P \). We also note \( P^{-1} \) as the reverse meta path of \( P \), which defines an inverse relation of \( P \) in \( T_G \). If a meta path \( P \) is equal to \( P^{-1} \) such as \( APA \), this meta path \( P \) can be called a symmetric meta path. Two meta paths \( P_1 = (A_1, A_2, \cdots , A_l) \) and \( P_2 = (B_1, B_2, \cdots , B_k) \) are concatenable if and only if \( A_l \) is equal to \( B_1 \), and the concatenated path is written as \( P = P_1 \circ P_2 \), which equals to \( (A_1, A_2, \cdots , A_l, B_2, \cdots , B_k) \).

### IV. Querying Top-K Important Meta-Paths

In this section, we start with some baselines and then introduce a general framework to evaluate the importance of a meta path between two objects \( s, t \in V \) in a given heterogeneous information network \( G = (V, E) \). After that we discuss how we apply this model in A* searching.

**A. baselines**

**SMP.** \( k \)-Shortest Meta Paths (noted as SMP) is the simplest solution, since the length is one intrinsic characteristic of a meta path \( P \). There is also a proof, derived from [18], showing that too long meta paths are not necessarily useful in determining similarity between two objects in a HIN. A naive solution to SMP is to perform breadth-first search (BFS) from \( s \) to find all shortest paths connecting to \( t \), which are then used to construct \( k \)-shortest meta paths.

**SLV1 and SLV2.** A strength concept of a meta path \( P \) is proposed to evaluate its importance. For \( P \) of more than one relations, i.e., \( P = A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \cdots \xrightarrow{R_l} A_{l+1} \), the corresponding strength can be calculated as follows.

\[
\text{Strength}(P) = \prod_{i=1}^{l} \frac{1}{\sqrt{O(A_i R_i A_{i+1})}} \cdot I(A_i R_i A_{i+1})
\]

where \( O(A_i R_i A_{i+1}) \) is the average out-degree of \( A_i \)-type nodes, and \( I(A_i R_i A_{i+1}) \) is the average in-degree of \( A_{i+1} \)-type nodes on the relation type \( R_i \) from \( A_i \) to \( A_{i+1} \). Considering strength and length, [16], [21] propose Strength-and-Length-based Versioned (noted as SLV) importance function as follows.

\[
\text{SLV1} : I(P) = e^{\text{Strength}(P)} \cdot e^{-|P|}
\]
\[
\text{SLV2} : I(P) = \text{Strength}(P) \cdot |P|^{-1}
\]

**B. Importance Function of A Meta Path**

We posit that length, path instances, global and local relationships between the given node pairs are fundamental to evaluate importance of a meta path. Therefore, the meta path importance function can be generalized as follows:

\[
\mathcal{I}(\mathcal{S}_s,t) = \mathcal{S}_s(t) \ast \mathcal{R}_s(t) \ast \text{Penalty}(|P|) \;
\]

where \( \mathcal{P}_{s \rightarrow t} \) refers to the meta path set in which each one can connect the two nodes, \( s \) and \( t \). And in (4), for a meta path \( P, \mathcal{S}_s(t) \) refers to the support of this meta path between the two nodes \( s \) and \( t, \mathcal{R}_s(t) \) refers to the rarity between them.

The last part in (4), \( \text{Penalty}(|P|) \), is the penalty function of its length, \( |P| \). More details of the importance function will be discussed in the following sections.

**C. How to Make it Support A* Searching**

The proposed top-\( k \) important meta paths searching is performed by an A* manner, as shown in Algorithm 1. **MetaNode** is a class to represent current expanded meta path and store relevant information such as \( \mathcal{I}(P) \). The whole algorithm is based on a priority queue(noted as \( Q \)) of many
MetaNode instances, which is decreasingly sorted by the upper bound of its importance (noted as $\mathcal{I}(\mathcal{P})$) and the algorithm will end only if the minimum $\mathcal{I}(\mathcal{P})$ of explored top-k meta paths (noted as $mPaths$) is greater than $\mathcal{I}(\mathcal{P})$ of the first node in $Q$. To guarantee the A* algorithm work, the importance function $\mathcal{I}(\mathcal{P})$ should have the monotonic-decreasing-maximum property.

**Definition 4. Monotonic-Decreasing-Maximum Property**

The monotonic-decreasing-maximum property of the importance function $\mathcal{I}_{s,t}(\mathcal{P})$ is defined as follows: given a graph $G = (V, E)$, its corresponding schema $T_G = (A, R)$, and a pair of $(s, t)$ where $s, t \in V$, $\forall \mathcal{P} \in T_G$, $\forall R_i \in R$, $\mathcal{I}_{s,t}(\mathcal{P}) \geq \mathcal{I}_{s,t}(\mathcal{P} \circ R_i)$.

**Algorithm 1** Top-K Important Meta Paths Discovery

**Input:** Network $G$, two nodes $s, t \in G$ and the number $k$  
**Output:** Top-k Important Meta Paths between $s$ and $t$

1. **if** not isConnected($s, t, G$) **then**
2.  
3. **output**("No Meta Paths Found")
4. **return** 
5. **end if**
6. $D_{s,t} \leftarrow$ GetSimilarPairs($s, t, G$)
7. $root \leftarrow$ initMetaNode($s, G$)
8. $mPaths \leftarrow \{\}$
9. $Q \leftarrow$ initPriorityQueue(MetaNode.class)
10. $Q$.push(root)
11. **while** not $Q$.empty() **do**
12.  
13. **if** $|\mathcal{I}(N.mPath)| < |\mathcal{I}(mPaths[k-1])|$ **then**
14.  
15. **break**
16. **end if**
17. **new_mPaths \leftarrow expand(N, G)**
18. **for each** $mPath \in$ new_mPaths **do**
19.  
20. **if** $t$ and $s$ is connected through $mPath$ **then**
21.  
22. $Ipt \leftarrow \mathcal{I}(s, t, mPath)$
23. **updateMetaPaths(mPaths, Ipt, mPath)**
24. **end if**
25. $Q$.push(new MetaNode(mPath))
26. **end for**
27. **end while**
28. **return** $mPaths$

### V. Computing Importance

In this section, we will discuss how to compute each part of $\mathcal{I}_{s,t}(\mathcal{P})$ in detail.

#### A. Length Penalty Function

For a meta path $\mathcal{P}$, a length penalty function aims to diminish its importance when $\mathcal{P}$ is longer. There have been some functions to penalize the meta path’s length such as $e^{-|\mathcal{P}|}$ in [16] or $\frac{1}{|\mathcal{P}|}$ in [21]. [12] also used $\beta^{\mathcal{P}}$ as a penalty function, where $\beta$ is a decay factor ranging in the open interval $(0, 1)$, and this can be seen as a generalized version of $e^{-|\mathcal{P}|}$. We employ the $\beta^{\mathcal{P}}$ in this paper, as we can tune the $\beta$ to adjust how much we penalize the length.

#### B. Rarity Function

The rarity function is designed to evaluate how rare the meta path $\mathcal{P}$ is among similar pairs to $(s, t)$ in a given HIN $G = (V, E)$. And the similar pairs (noted by $D_{s,t}$) is defined as follows:

$$D_{s,t} = D_t \cup D_s$$

where

$$D_t = \{(s, v) | v \in V, \phi(v) \cap \phi(t) \neq \emptyset\}$$

$$D_s = \{(v, t) | v \in V, \phi(v) \cap \phi(s) \neq \emptyset\}$$

inspired by the inverse document frequency (IDF) in TF-IDF, we evaluate the rarity $\mathcal{P}$ in a similar way:

$$\mathcal{R}_{s,t}(\mathcal{P}) = \log \frac{|D_{s,t}|}{|\{(u, v) | u, v \in D_{s,t}, \mathcal{P} \in \mathbb{P}_{u \rightarrow v}\}|}$$

Because we do not exclude $s$ in $D_s$ or $t$ in $D_t$, it is easy to prove that $\forall \mathcal{P} \in \mathbb{P}_{s \rightarrow t}, \mathcal{R}_{s,t}(\mathcal{P}) \leq \log |D_{s,t}|$.

#### C. Support

To guarantee the monotonic-decreasing-maximum property of $\mathcal{I}_{s,t}(\mathcal{P})$, we should pay special attention to the support function. According to the definition of $\mathcal{I}_{s,t}(\mathcal{P})$ and the maximum rarity, we can have the following equation:

$$\mathcal{I}_{s,t}(\mathcal{P}) = \log |D_{s,t}| * \mathcal{R}_{s,t}(\mathcal{P}) * \text{Penalty}(\mathcal{P})$$

Since the penalty function is strict decreasing, we thus require $\mathcal{S}_{s,t}(\mathcal{P})$ have the monotonic-decreasing-maximum property to guarantee the same property of $\mathcal{I}_{s,t}(\mathcal{P})$.

#### D. Binary Support (BS)

A naive idea to design the support is a binary function, which returns a positive constant if $\mathcal{P} \in \mathbb{P}_{s \rightarrow t}$, and returns 0 if $\mathcal{P} \notin \mathbb{P}_{s \rightarrow t}$. The constant function always has the monotonic-decreasing-maximum property because it is bounded by a constant. For example, we set the constant as 1, and the corresponding binary support function is as shown as follows:

$$\mathcal{S}_{s,t}(\mathcal{P}) = \begin{cases} 1, & \mathcal{P} \in \mathbb{P}_{s \rightarrow t} \\ 0, & \mathcal{P} \notin \mathbb{P}_{s \rightarrow t} \end{cases}$$

$\mathcal{S}_{s,t}(\mathcal{P})$ can be seen as the constant value, 1, which means that $\mathcal{S}_{s,t}(\mathcal{P})$ can be applied into our A* algorithm. Therefore, we now have a complete importance function (Binary-Supported, noted as BS). This importance function can also be seen as the combination of the rarity and the penalty function, because the support remains the same for $\mathcal{P} \in \mathbb{P}_{s \rightarrow t}$.

#### E. MNI-based Support (MNIS)

Inspired from the Minimum Instance [1], we design another support function to reflect the frequency without violating the monotonic-decreasing-maximum property. The definition of Minimum Instance (short as MNI) is given as follows.

**Definition 5. Minimum Instances** The minimum instances of a meta path $\mathcal{P} = A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \cdots \xrightarrow{R_i} A_{i+1}$ between a given pair $(s, t)$ refers to the minimum instance number of each node type in that meta path $\mathcal{P}$ (not including two ends). If we note as $p_i$ the node $v$ in the $i$-th node type $(1 < i < l + 1)$ in
a path instance \( p \) of \( \mathcal{P} \), we then can get the following equation to calculate the MNI for a meta path \( \mathcal{P} \):

\[
MNI(\mathcal{P}) = \min_{1 \leq i < l + 1} |\{p_i | p \in \mathcal{P}\}|
\tag{10}
\]

The \( |\{p_i | p \in \mathcal{P}\}| \) in (10) refers to the instance number of the \( i \)-th node type in the meta path \( \mathcal{P} \). And obviously, MNI of a meta path \( \mathcal{P} \) has the monotonic-decreasing-maximum property because \( \min_{1 \leq i < l + 1} |\{p_i | p \in \mathcal{P}\}| \geq \min_{1 \leq i < l + 2} |\{p_i | p \in \mathcal{P} \circ R_i\}| \).

However, compared with one-to-many relation, such as \( \text{Paper} \xrightarrow{\text{publish}} \text{Venue} \), one-to-one relation such as \( \text{Paper} \xrightarrow{\text{mention}} \text{Topic} \) could always have smaller MNI because one paper can mention several topics but it can be only published in one venue. To eliminate the unbalance, we multiply MNI by another strength function:

\[
\text{Strength}(A \xrightarrow{R} B) = \frac{1}{\min\{O(A \xrightarrow{R} B), I(A \xrightarrow{R} B)\}}
\tag{11}
\]

Note that we only consider those \( A \)-type nodes and \( B \)-type nodes that have at least one relation \( A \xrightarrow{R} B \) when we compute the average out-degree and in-degree. Therefore, multiplying MNI by this function will keep the monotonic-decreasing-maximum property of the \( I(\mathcal{P}) \). And we can offline compute \( O(A \xrightarrow{R} B) \) and \( I(A \xrightarrow{R} B) \) of each relation for every strength function and reuse them to accelerate top-k meta paths query.

Synthesizing the above discussion, we can combine \( MNI_{s,i}(\mathcal{P}) \) with \( \text{Strength}(\mathcal{P}) \) and then construct the MNI support function in our framework:

\[
S_{s,i}(\mathcal{P}) = \text{Strength}(\mathcal{P}) \times MNI_{s,i}(\mathcal{P})
\tag{12}
\]

We thus have another version of importance function (strengthened-MNI Supported noted as MNIS) by using the strengthened MNI support and keeping other parts in \( I(\mathcal{P}) \) unchanged. By incorporating the strength, \( S_{s,i}(\mathcal{P} \circ R_i) \) of the next \textit{MetaNode} can be calculated by multiplying the current \( S_{s,i}(\mathcal{P}) \) and the strength of the expanding relation \( R_i \), which will reduce \( S_{s,i}(\mathcal{P}) \) significantly and thus A* searching will end much faster.

VI. RESULTS

We now discuss a case study and a connectivity experiment by comparing the performance of MNIS, BS with SMP, SLV1 and SLV2. These results can help us validate the effectiveness of MNIS. All the experiment programs are compiled with 4.6.3-version gcc and executed in Ubuntu 12.04.01 with 8 Intel(R) Core(TM) i7-3770 processors.

A. Data set

Our experiments employ the HIN of ACM data set [15]. The ACM data set contains 14 representative computer science conferences: KDD, SIGMOD, WWW, SIGIR, CIKM, SODA, STOC, SOSP, SPAA, SIGCOMM, MobiCom, ICML, COLT and VLDB. These conferences contain 196 venues proceedings. This dataset includes 12K papers, 17K authors, 1.8K affiliations and 1.5K frequent terms. Each paper is also labeled by a subject and we extract subjects for further use. After that, there are 1096K links remaining in this data set. The corresponding network schema is shown in Fig 2a.

B. Case Study

In this section, we first study the effectiveness of our approach on ACM data set through exploring the top-k relationship between Jiawei Han and Philip Yu with different methods. We will follow the definition of \( M_1, M_2, M_3, M_4 \) in Section I to simplify our discussion. We use \( k \)-shortest path to explore top 4 path instances between them and we found 13 path instances that have the same length. Among these 13 paths, 12 paths have the same meaning in meta level: they are coauthors for 12 papers, which can be expressed by \( M_2 \) (i.e., \( A \rightarrow P \leftarrow A \)), while the left instance can be captured by \( M_1 \).

<table>
<thead>
<tr>
<th>Rank</th>
<th>Meta Path ( \mathcal{P} )</th>
<th>( \mathcal{I}(\mathcal{P}) )</th>
<th>MNI</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( A \rightarrow P \leftarrow A )</td>
<td>0.154</td>
<td>12</td>
<td>5.8844</td>
</tr>
<tr>
<td>2</td>
<td>( A \rightarrow P \leftarrow P \leftarrow A )</td>
<td>2.4 \times 10^{-3}</td>
<td>14</td>
<td>4.3884</td>
</tr>
<tr>
<td>3</td>
<td>( A \rightarrow P \rightarrow A \leftarrow P )</td>
<td>2.1 \times 10^{-3}</td>
<td>13</td>
<td>4.1258</td>
</tr>
<tr>
<td>4</td>
<td>( A \rightarrow P \rightarrow A \leftarrow P \rightarrow A )</td>
<td>3.65 \times 10^{-4}</td>
<td>21</td>
<td>3.6735</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rank</th>
<th>Meta Path ( \mathcal{P} )</th>
<th>( \mathcal{I}(\mathcal{P}) ) (SLV1)</th>
<th>( \mathcal{I}(\mathcal{P}) ) (SLV2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( A \rightarrow P \leftarrow A )</td>
<td>1.2 \times 10^{-2}</td>
<td>0.139</td>
</tr>
<tr>
<td>2</td>
<td>( A \rightarrow F \leftarrow A )</td>
<td>4.86 \times 10^{-4}</td>
<td>0.135</td>
</tr>
<tr>
<td>3</td>
<td>( A \rightarrow P \rightarrow P \leftarrow A )</td>
<td>4.82 \times 10^{-4}</td>
<td>4.99 \times 10^{-2}</td>
</tr>
<tr>
<td>4</td>
<td>( A \rightarrow P \rightarrow P \leftarrow P \leftarrow A )</td>
<td>4.82 \times 10^{-4}</td>
<td>4.99 \times 10^{-2}</td>
</tr>
</tbody>
</table>

The top 4 relationships between Jiawei Han and Philip Yu (abbreviated as \( < J. \text{Han}, P. \text{Yu} > \)) by MNIS are shown in Table I. Because the top-4 rankings of SLV1 and SLV2 are the same, they are both put in Table II.

According to the result, all methods rank \( M_2 \) the first place but when it comes to the top 4 meta paths, much more difference emerged. In the result of MNIS, \( M_1 \) (short as \( A \rightarrow F \leftarrow A \)) cannot even rank into the top 4 important relationships. In addition to the ranking of \( M_1 \), from Table II we can also see that SLV1 and SLV2, these two methods do not consider the direction of the edge and thus the weights of \( M_3 \) (i.e., \( A \rightarrow P \leftarrow P \leftarrow A \)) and \( M_4 \) (i.e., \( A \rightarrow P \rightarrow P \leftarrow A \)) are the same, which means that SLV1 and SLV2 cannot differentiate two meta paths if they share the same edge types but some of them have different directions. Compared with SLV1 and SLV2, MNIS can differentiate \( M_3 \) and \( M_4 \). In a conclusion of this case study, only MNIS can rank \( M_2 \) as the top one and differentiate \( M_3 \) and \( M_4 \) at the same time.

C. Label-based Author Connectivity Analysis

In the original ACM data set, each paper is related to just one subject, and thus we extract these subjects as labels for those papers to evaluate the performance of the label-based author connectivity. We first sample 100 positive pairs and 100 negative pairs from all papers where a positive pair means that two papers are related to the same subject, and a negative pair...
means two papers are related to different subjects. Then, for the $j$-th pair ($1 \leq j \leq 100$) of two papers in positive pair, we compute the top 4 meta paths (noted by $\{P_{i,j} | 1 \leq i \leq 4\}$) between them according using different algorithms. For the $i$-th meta path $P_{i,j}$ obtained by the $j$-th pair, we now can count how many positive pairs can be connected by it (the number of connected positive pairs is noted as $p_{i,j}$) and how many negative pairs can be connected by it (the number of connected negative pairs is noted as $n_{i,j}$). The accuracy of the $i$-th meta path $P_{i}$ for the $j$-th pair can be calculated by $\frac{p_{i,j}}{p_{i,j} + n_{i,j}}$. Finally we evaluate the performance by the average accuracy of the top-$k$ meta paths, noted by $\text{Accuracy}@k$:

$$\text{Accuracy}@k = \frac{1}{k} \sum_{i=1}^{k} \frac{1}{100} \sum_{j=1}^{100} \frac{p_{i,j}}{p_{i,j} + n_{i,j}} \quad (k = 1..4)$$

The comparison result between MNIS ($\beta = 0.1$) and other baselines are shown in Fig. 3a, which reveals that MNIS outperforms other baselines in this task. Furthermore, from Fig. 3b, we can also see how different $\beta$ ($\beta = 0.05, 0.1, 0.15, 0.2$) influences on the $\text{Accuracy}@k$. The result shows that a larger $\beta$ generates a better result, and this is because a larger $\beta$ indicates a larger searching space which makes those important but longer meta paths can be explored.

VII. CONCLUSION

In this paper, we study the problem of discovering top-$k$ important meta paths between two nodes in a HIN. We propose a ranking function for meta paths based on frequency and rarity, and design an A* search algorithm for efficient top-$k$ search. The connectivity experiment on ACM shows that our proposed method outperforms state-of-the-art methods.

REFERENCES


