On Graph-Based Name Disambiguation

XIAOMING FAN, JIANYONG WANG, XU PU,
LIZHU ZHOU, and BING LV, Tsinghua University

名字歧义
Name ambiguity stems from the fact that many people or objects share identical names in the real world. Such name ambiguity decreases the performance of document retrieval, Web search, information integration, and may cause confusion in other applications. Due to the same name spellings and lack of information, it is a nontrivial task to distinguish them accurately. In this article, we focus on investigating the problem in digital libraries to distinguish publications written by authors with identical names. We present an effective framework named GHOST (abbreviation for GrapHical framewOrk for name diSambiguaTion), to solve the problem systematically. We devise a novel similarity metric, and utilize only one type of attribute (i.e., coauthorship) in GHOST. Given the similarity matrix, intermediate results are grouped into clusters with a recently introduced powerful clustering algorithm called *Affinity Propagation*. In addition, as a complementary technique, user feedback can be used to enhance the performance. We evaluated the framework on the real DBLP and PubMed datasets, and the experimental results show that GHOST can achieve both high *precision* and *recall*.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications—*Data mining*; H.2.m [**Database Management**]: Miscellaneous; H.3.7 [**Information Storage and Retrieval**]: Digital Libraries

General Terms: Algorithms, Experimentation

Additional Key Words and Phrases: Name disambiguation, similarity, clustering, graph

ACM Reference Format:
Fan, X., Wang, J., Pu, X., Zhou, L., and Lv, B. 2010. On graph-based name disambiguation. ACM J. Data Inform. Quality 2, 2, Article 10 (February 2011), 23 pages.
DOI = 10.1145/1891879.1891883 http://doi.acm.org/10.1145/1891879.1891883

1. INTRODUCTION

Due to personalized styles and contextual requirements, name variation, identical names for different persons, and name misspellings are not uncommon in various information sources (e.g, Web documents, bibliographies, and so on). As it is impossible

This work was supported in part by National Natural Science Foundation of China under grant No. 60833003, the National Basic Research Program of China (973 Program) under grant No. 2011CB302206, an HP Labs Innovation Research Program award, an Okawa Foundation Research Grant, and the Program for New Century Excellent Talents in University under grant No. NCET-07-0491, State Education Ministry of China.
Authors' address: X. Fan, Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, P.R. China; email: xmfan1983@gmail.com; J. Wang, Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, P.R. China; email: jianyong@tsinghua.edu.cn; X. Pu, Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, P.R. China; email: bertpu@gmail.com; L. Zhou, Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, P.R. China; email: dcszlz@tsinghua.edu.cn; B. Lv, Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, P.R. China; email: alvb05@gmail.com.

DOI 10.1145/1891879.1891883 http://doi.acm.org/10.1145/1891879.1891883

Table I. Different Authors with Identical Names

Name	Present Affiliation	# publications
Wei Wang	University of North Carolina at Chapel Hill	57
	Fudan University, China	31
	The University of New South Wales, Australia	19
	The State University of New York at Buffalo	5
Jim Smith	University of the West of England	13
	University of Newcastle upon Tyne	15
	The University of Melbourne, Australia	2
	University of Wisconsin	1
Lei Wang	Nanyang Technological University, Singapore	14
	Zhejiang University, China	8
	University of Texas at Dallas	7
	Harvard University	2
Jing Zhang	Tsinghua University, China	5
	Tsinghua University, China	1
	Tsinghua University, China	2
	Carnegie Mellon University	5

to assign a unique identifier to every name, it may cause confusion (also known as *name ambiguity*) in many cases if people use names to identify specified persons. In general, the problems caused by name ambiguity can be classified into two categories.

The first popular case is the so-called *reference disambiguation* [Kalashnikov and Mehrotra 2006], which aims to group different variations of an entity into a cluster so that publications in the cluster refer to the same entity. It is also known as *entity resolution* [Chen et al. 2007; On et al. 2006], *record linkage* [Kalashnikov and Mehrotra 2006; Winkler 1999], *duplicate detection* [Bilenko and Mooney 2003; Sarawagi and Bhamidipaty 2002], *reference reconciliation* [Dong et al. 2005], and *object matching* [Bilenko et al. 2003; Chaudhuri et al. 2003; Lawrence et al. 1999; McCallum et al. 2000; Pasula et al. 2002]. Much work has been conducted to detect whether two different names are duplicates of one another. Take person names as an example. The names of "David S. Johnson," "David Johnson," "D. S. Johson," and "D. Johnson" may appear in multiple publications. It is not a trivial task for a reader to know how many different authors are behind the four names. Prior work has achieved encouraging research results on the problem.

找到同一个人的所有不同指代

The second case, which is opposite to the preceding one and is the focus of this article, is called *name disambiguation*. Say we are looking for all publications authored by "Lei Wang" in a digital library. Without loss of generality, we take DBLP and CiteSeer as examples. When the query is submitted, about 151 publications are retrieved in DBLP and 123 in CiteSeer. As far as we know, there are no fewer than 39 different persons with this identical name, and it is difficult to differentiate the publications written by a person one really cares about from the remaining ones belonging to other persons who happen to share the same author name. As an example, Table I shows four names and for each name we list four distinct authors, their present affiliations, and the corresponding number of publications.

区分同一个名字的不同人

We need to point out that in essence both reference disambiguation and name disambiguation are subproblems of entity resolution and some techniques proposed for reference disambiguation could also be used for name disambiguation. However, there are still some differences between these two problems. First, name disambiguation is performed with respect to a given name, which is search-oriented, while reference disambiguation is usually a batch process. Second, as the names of different authors are identical in the problem setting of name disambiguation, some techniques (e.g., the ones based on textual similarity) proposed for reference disambiguation may not work well.

Name ambiguity of the second case also brings confusion when we try to merge multiple tables from different data sources into a single table (i.e., the so-called information integration), and it may decrease the performance of information retrieval and Web search. This is the reason why many database and knowledge management applications take name disambiguation as the first step of data cleaning. Thus the objective of name disambiguation is to group objects (e.g., publications) into clusters so that the elements in each cluster belong to the same entity (e.g., author), while all the objects of the same entity are grouped in the same cluster.

In this article we present a graph-based framework, GHOST, to address the problem. As a five-step framework, GHOST first tries to exploit the relationships/connections among every pair of publications via a graphical model which is quite different from those adopted in prior work, and the following four steps, *valid path selection*, *similarity computation*, *name clustering*, and *user feedback*, are then performed serially.

Here we summarize the contributions of this article as follows.

— We propose a graph-based framework, GHOST, for name disambiguation, which utilizes the attribute of coauthorship only while excluding all other attributes such as email, venue, publication title, and author affiliation.
— We propose a simple formula to compute similarity between any pair of nodes in a coauthorship graph. It is very intuitive, and achieves as good performance as some complicated approaches proposed in prior work.
— We adopt for GHOST a recently introduced clustering strategy named *Affinity Propagation* based on message-passing techniques. We conducted an empirical study by using it to cluster the intermediate results, and the experimental results show it works very well.

The remainder of this article is organized as follows. In Section 2, we present an overview of related work. In Section 3, we formalize the problem from the perspective of graphical model. The details of GHOST are discussed in Section 4. The overall framework is then empirically evaluated in Section 5, and the implications to research and practice are presented in Section 6. We conclude the article in Section 7.

2. RELATED WORK

As we stated before that name disambiguation can be seen as a subproblem of entity resolution, the technologies proposed for *reference disambiguation* can be potentially exploited to solve name disambiguation. Generally speaking, given a set of publications written by authors with similar spellings, almost all solutions proposed for reference disambiguation exploit multiple attributes of these publications, such as title, abstract, venue, authors' emails, and affiliations, and try to define the so-called *feature-based similarity* [Fellegi and Sunter 1969; Hernández and Stolfo 1995; Winkler 1999], including edit distance between two similar but not identical names, relevance of the titles' keywords, where IR techniques are utilized to facilitate modeling. Then similarity functions are employed to determine the probability that any pair of publications are authored by the same person. This elementary method based on textual similarity is employed in Kalashnikov and Mehrotra [2006], On et al. [2006], On and Lee [2007], Bekkerman and McCallum [2005], Lee et al. [2005], and Han et al. [2004, 2005]. Furthermore, a variety of methods are proposed to explore the interobject relational similarity via various graphical models. For example, Minkov et al. [2006] model the email data as a labeled directed graph and try to devise a type of relational similarity metric based on lazy graph walk, which can be used to disambiguate personal names in email. Chen et al. [2007] propose an adaptive graphical approach to entity resolution. It defines the similarity between two nodes in the graph as a combination of the traditional feature-based similarity and their connection strength, while the connection strength between nodes u and v is simply computed as the sum of connection strengths of all simple paths no longer than a user-specified length L. Differently from the approach of Chen et al. [2007], in this work the similarity between two nodes does not take into account the feature-based similarity, and the connection strength between nodes u and v is modeled using an *Ohm's Law*-like formula defined over a carefully selected subset of valid paths. Kalashnikov and Mehrotra [2006] make use of connecting paths to propose a notion of *connection strength* which captures how strongly two entities are connected to each other through *legal paths*. Differently from them, On et al. [2006] apply a recently proposed graph mining technique, *Quasi-Clique* [Pei et al. 2005], to exploit contextual information in addition to syntactic similarity.

In the setting of name disambiguation, we suppose that the names to be disambiguated have identical spellings, and textual similarity of names becomes useless. In addition, with the growth of dataset, it is exhausting to collect enough information such as emails and affiliations manually, and such kind of auxiliary information is not always available and may also have variants. Thus, common and similar attributes/features are not available in different contexts. As a result, it is obviously difficult to make good judgements based on such limited information, and it still deserves some efforts to study the problem of name disambiguation.

To the best of our knowledge, the so-called *object distinction* and *citation matching* studied in Yin et al. [2007], Zhang et al. [2007], Han et al. [2004, 2005], and Lee et al. [2005] are the problems most similar to ours. Traditionally, clustering methods appear to be a natural solution for such problems. By and large, related approaches make efforts in two aspects to achieve potential enhancement. The first is to measure similarities between every pair of names using different models and the second is to choose applicable clustering strategies to group publications into clusters. The performance of previous work appears to be satisfactory in specified contexts and on some datasets, but they also have several limitations.

For example, Han et al. [2004] propose two supervised clustering methods based on naive Bayes and support vector machines, respectively. Firstly it learns a specific model for each author's name from training data and uses the model to determine by whom a new-coming publication is written. Clearly, the approach relies on training sets which require much labor to construct manually. Even worse, the method becomes impractical to train thousands of models in a large-scale digital library. Zhang et al. [2007] utilize six types of constraints (including title, abstract, and references) and devise a probabilistic model for semisupervised name disambiguation. This model is based on hidden Markov random fields proposed in Basu et al. [2004]. As we can observe, the weights of different linkages are user-dependent and vulnerable to modifications. DISTINCT in Yin et al. [2007] is one of the state-of-the-art algorithms to solve a totally identical problem to ours. The method combines two complementary measures for relational similarity, that is, set resemblance of neighbor tuples and random walk probability. Similarly to solutions for reference disambiguation, DISTINCT utilizes several different types of linkages, including title, venue, publisher, year, authors' affiliation, etc. Then supervised learning with constructed training sets is applied to determine the weights of different types of linkages. Finally the agglomerative hierarchical clustering is employed to get results as the last step.

Differently from the preceding approaches, we recently proposed an effective graph-based framework for solving the name disambiguation problem, and a two-page poster paper has described its preliminary ideas [Fan et al. 2008]. In this manuscript, we give a detailed, complete, and in-depth description of our graph-based name disambiguation solution. Specifically, we introduce in detail the graphical view of

Table II. Notations

Name	Description
$\mathcal{D}$	a database of academic publications
$\vert\mathcal{D}\vert$	the number of publications in $\mathcal{D}$
r	the author name which needs to resolve
R	the set of publications with the author name r
$\vert R\vert$	the number of publications in R
P_r^i	the ith publication in R (coauthored by r_i, $1 \le i \le \vert R\vert$)
r_i	the author whose name is r in P_r^i
k	the number of distinct authors with name r ($k \le \vert R\vert$)
c_j	set of publications written by the jth distinct author ($1 \le j \le k$)

the input database, the valid path selection method, the similarity metric between two nodes, the clustering framework, and some user feedback strategies. We also present a convincing experimental study which shows that although our new graph-based framework uses the coauthorship information only, it achieves comparable performance with the state-of-the-art approach, DISTINCT [Yin et al. 2007].

3. PROBLEM FORMULATION

We have given an informal introduction to the concept of name disambiguation, and in this section, we formalize our problem definition in the setting of an academic publication record database. Let $\mathcal{D}$ be a database which contains a set of academic publications. Each publication has a set of attributes, such as a list of authors, a title, and a venue (e.g., a conference or a journal). We suppose r is an author name whose publications are to be distinguished, and denote the subset of $\mathcal{D}$ which contains the publications coauthored by any person whose name is r by R. Let the number of publications in R be $|R|$, the number of distinct authors with name r be k, the ith publication in R be P_r^i $(1 \le i \le |R|)$, the author whose name is r in publication P_r^i be r_i $(1 \le i \le |R|)$. The notations used in this article are summarized in Table II.

After introducing some basic notations, we give the formal definition of the problem studied in this article as follows. Given a database $\mathcal{D}$ consisting of $|\mathcal{D}|$ publications and an author name r, our task is to automatically group the publications authored by any person with a name of r (i.e., publications in R) into k disjoint clusters (where k is the number of distinct authors with a name r) $\mathcal{C} = \{c_1, \ldots, c_k\}$ such that each publication within a cluster is authored by the same person and the publications of the same person are not split across clusters.

4. THE GHOST FRAMEWORK

In this section we introduce the GHOST framework step by step, which consists of five subtasks, that is, graphical view of the database, valid path selection, similarity computation, name clustering, and user feedback.

4.1 Graphical View of the Database

In general, it is a natural idea to represent the database $\mathcal{D}$ as a graph $G = \{V, E\}$, where each node $v \in V$ represents a distinct author name (or an instance of the queried author name r in a certain publication), and an undirected edge represents a coauthorship. According to the preceding definition, even multiple different names correspond to the same person, we do not merge them in the graph, for example, if "Philip S. Yu" and "Philip Yu" correspond to the same author, they will be represented as two different nodes in the graph. Each edge between node v_i and node v_j has a label $S(i, j)$, which denotes the complete set of publications coauthored by both v_i and v_j. This means that we collect all publications coauthored by v_i and v_j in order to denote a unique edge. For

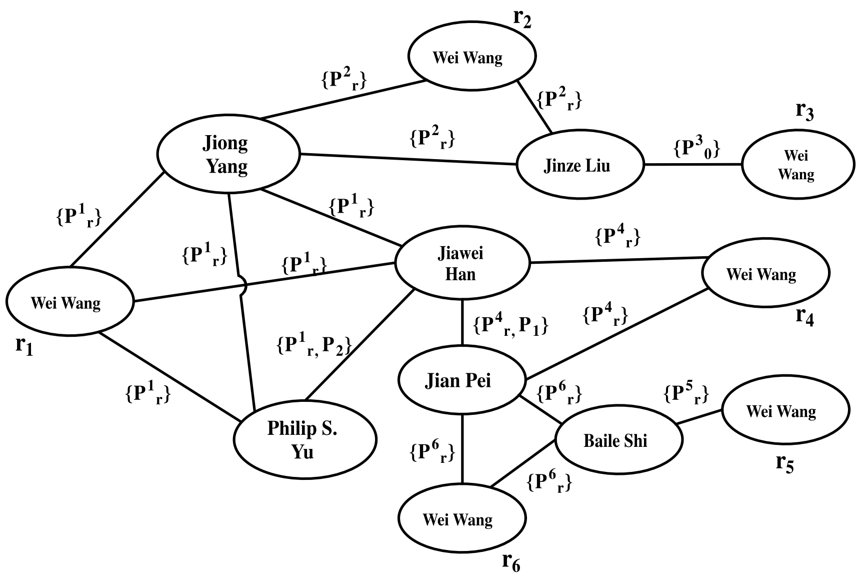

Fig. 1. Graphical view of the example database with respect to the name of "Wei Wang."

convenience, only the nonempty sets are taken into account. Here we give an example to illustrate the graphical model more clearly. Assume that a tiny database containing several papers is available as shown in Example 1, and the name to resolve is "Wei Wang". Note that in Example 1 P_j as opposed to P_r^i represents a publication that does not mention author name of "Wei Wang". Since our method does not utilize the paper titles, they are omitted in this example.

Example 1.

P_r^1: Jiong Yang, Wei Wang, Philip S. Yu, Jiawei Han. SIGMOD'02.
P_1: Jiawei Han, Jian Pei, Guozhu Dong, Ke Wang. SIGMOD'01.
P_r^2: Jinze Liu, Wei Wang, Jiong Yang. SIGKDD'04.
P_2: Charu C. Aggarwal, Jiawei Han, Jianyong Wang, Philip S. Yu. VLDB'03.
P_r^3: Jinze Liu, Wei Wang. ICDM'03.
P_r^4: Jian Pei, Jiawei Han, Wei Wang. CIKM'02.
P_r^5: Peng Wang, Haixun Wang, Xiaochen Wu, Wei Wang, Baile Shi. ICDM'05.
P_r^6: Chen Wang, Wei Wang, Jian Pei, Yongtai Zhu, Baile Shi. SIGKDD'04.

The graph generated from Example 1 is shown in Figure 1. For simplicity, other authors (e.g., Jianyong Wang and Charu Aggarwal) who never coauthored with *r* (i.e., "Wei Wang") or only coauthored one publication (e.g., Peng Wang, Haixun Wang, and Xiaochen Wu in P_r^5, Chen Wang and Yongtai Zhu in P_r^6) are omitted. There are six nodes labeled with "Wei Wang" in Figure 1, each of which represents an instance of "Wei Wang" in one of the publications coauthored by "Wei Wang". Two authors are linked by an edge if they have coauthored a paper, and we ignore other types of attributes and their corresponding links. For example, in Figure 1, there is an edge between the nodes of "Jiawei Han" and "Philip S. Yu", and the edge label of $\{P_r^1, P_2\}$ means Jiawei Han and Philip S. Yu are coauthors of publications P_r^1 and P_2.

4.2 Valid Path Selection

The design philosophy of GHOST is based on the observation that the research interests of a researcher usually do not change too frequently during a short period of time, and in particular, he/she would stay in the same institution for a relatively long time. We can infer that a researcher usually has a relatively stable set of coauthors (i.e., author community) during a certain period of time. Also, we expect that different persons with the same name seldom work in the same institution and thus they should have quite different author communities. In addition, it is a natural idea that the coauthors usually focus on some close research topics and share some common research interests. If the same name which needs to be disambiguated (e.g., "Wei Wang") appears in both author lists of two publications and corresponds to the same person, these two publications are expected to share some coauthors either directly or by some other intermediate publications. On the other hand, if they are not written by the same person, their coauthor sets likely do not overlap or there are few paths which link them. GHOST tries to unearth some valuable hidden information based on both short and long coauthorship linkages in order to compute the similarity of two authors with identical name. Thus, the first task of GHOST is to retrieve paths linking two nodes in the graph view.

A naïve way to compute the similarity between two nodes in the coauthorship graph is based on the length of their shortest path. However, it may neglect valuable information which can be derived from other paths linking the two nodes in the graph. For example, if five paths of length 3 link two nodes of v_i and v_j which need to be disambiguated, they should be regarded to be more similar to each other than another two nodes of v_i and v_k which are linked by just one path of length 3 or even shorter. Therefore, more precise similarity computing strategy is required, and it is imperative for GHOST to exploit all simple paths, where a path is said to be *simple* if it does not contain any duplicate nodes. Because we assign equal weight to edges, breadth-first search appears to be a natural strategy to find out all loopless paths. As we know, however, the runtime complexity of breadth-first search is $O(|V| + |E|)$, thus we need to propose a faster search strategy to reduce the time complexity. But before we go further, let us first investigate a substructure of the coauthorship graph shown in Figure 1, which will lead to the definition of *valid path*.

4.2.1 Valid Path. Suppose we want to find paths linking nodes of r_1 and r_6 in Figure 1, the shortest path is obviously "Wei Wang — Jiawei Han — Jian Pei — Wei Wang". The two intermediate nodes representing "Jiawei Han" and "Jian Pei" connect the starting node and ending node. In addition, we can easily figure out another 19 paths which link nodes of r_1 and r_6 from the graph structure. This is because there are a total of 5 simple subpaths linking "Wei Wang(i.e., r_1)" and "Jiawei Han", 2 simple subpaths linking "Jiawei Han" and "Jian Pei", and 2 simple subpaths linking "Jian Pei" and "Wei Wang (i.e., r_6)". Take the 5 simple subpaths connecting "Wei Wang(i.e., r_1)" and "Jiawei Han" as an example. They form a substructure which is shown in Figure 2. We can derive the other four simple subpaths shown as follows besides the shortest subpath introduced before (i.e., "Wei Wang — Jiawei Han").

(1) "Wei Wang — Jiong Yang — Jiawei Han"
(2) "Wei Wang — Philip S. Yu — Jiawei Han"
(3) "Wei Wang — Jiong Yang — Philip S. Yu — Jiawei Han"
(4) "Wei Wang — Philip S. Yu — Jiong Yang — Jiawei Han"

An intuitive observation from the previous example is that there is some redundancy among all the five simple subpaths (e.g., the two paths of "Wei Wang — Jiong

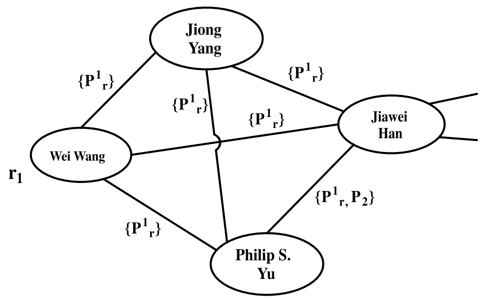

Fig. 2. A substructure of the coauthorship graph.

Yang — Philip S. Yu — Jiawei Han" and "Wei Wang — Philip S. Yu — Jiong Yang — Jiawei Han"). Therefore, GHOST should discriminate different paths based on the graph structure and eliminate those redundant subpaths that make no sense in similarity computation. Besides, the coauthorship graph could be so huge, resulting in high runtime complexity to find all simple paths. For this purpose, GHOST calls for a criterion to judge whether a path deserves to be searched for. Before we present a generalized definition of a valid path, it is necessary to present the following lemma first.

LEMMA 4.1. *Consider two nodes of v_1 and v_2 both of which connect to another node v_0 via sets of publications of S_{01} and S_{02}, respectively. If S_{01} and S_{02} share a common publication, which is denoted by P_c, we come to the conclusion that v_1 and v_2 also connect via a publication set containing at least one publication of P_c.*

PROOF. Because S_{01} and S_{02} share a common element P_c, we know that v_0 and v_1 have coauthored the publication P_c, so have v_0 and v_2. This implies that v_0 , v_1, and v_2 are all in the author list of P_c. So the lemma holds. □

In the following, we will develop the valid path selection criterion. For each of the intermediate vertices V in a specific path, the vertex itself and its two adjacent vertices will form a triangle-like structure if there exists an edge between the two adjacent vertices, taking V_k for example as illustrated in Figure 3. If no publication is shared by the publication sets $S_{(k-1)k}$, $S_{k(k+1)}$ and $S_{(k-1)(k+1)}$, V_k is a valid vertex in the specific path. If any two publication sets of the three edges contain a same publication, denoted by P_c, we can infer from Lemma 4.1 that the third set must also contain P_c. In this case, if there are at least two sets with cardinality no smaller than two, V_k is also considered to be valid as the three authors are coauthored frequently. However, when two of the three sets both consist of only P_c, we consider V_k as an invalid vertex because the path of $V_s \xrightarrow{S_{s1}} V_1 \xrightarrow{S_{12}} V_2 \cdots V_{k-1} \xrightarrow{S_{(k-1)k}} V_k \xrightarrow{S_{k(k+1)}} V_{k+1} \cdots V_{n-1} \xrightarrow{S_{(n-1)n}} V_n \xrightarrow{S_{ne}} V_e$ makes redundant contribution of path $V_s \xrightarrow{S_{s1}} V_1 \xrightarrow{S_{12}} V_2 \cdots V_{k-1} \xrightarrow{S_{(k-1)(k+1)}} V_{k+1} \cdots V_{n-1} \xrightarrow{S_{(n-1)n}} V_n \xrightarrow{S_{ne}} V_e$ to the similarity between V_s and V_e.

We now give the formal definition of an invalid vertex in a path and a valid path.

Definition 1 (*Invalid Vertex in a Path*). Given a path $P = V_s \xrightarrow{S_{s1}} V_1 \xrightarrow{S_{12}} V_2 \cdots V_{n-1} \xrightarrow{S_{(n-1)n}} V_n \xrightarrow{S_{ne}} V_e$, if $\exists V_k \in \{V_1, V_2, \cdots, V_{n-1}, V_n\}$ such that at least one of the following three conditions holds, V_k is called *an invalid vertex* in path P.

(1) $S_{(k-1)k} = S_{k(k+1)}$ and $|S_{(k-1)k}| = 1$
(2) $S_{(k-1)k} = S_{(k-1)(k+1)}$ and $|S_{(k-1)k}| = 1$
(3) $S_{k(k+1)} = S_{(k-1)(k+1)}$ and $|S_{k(k+1)}| = 1$

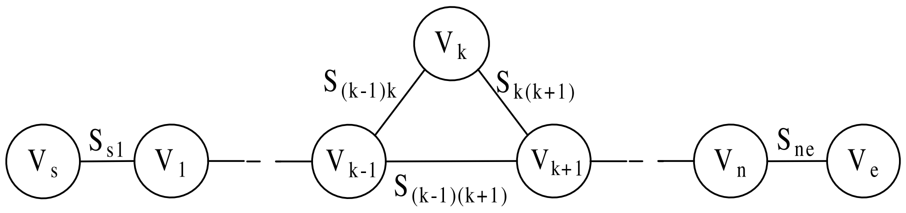

Fig. 3. A triangle-like structure.

Note that an invalid vertex V in a path P could be valid in other paths, since the validity of a vertex with respect to a given path depends on its adjacent vertices in the path.

Definition 2 (*Invalid Path*). A path P is an invalid path if and only if P contains at least one invalid vertex.

Definition 3 (*Valid Path*). A path P is said to be valid if it contains no invalid vertex.

While searching in the graph, GHOST will eliminate all the invalid paths. What GHOST needs to do is to check the intermediate vertices in turn for each path and eliminate the path once the first invalid vertex is found in it.

4.2.2 The Search Strategy. As we state at the beginning of Section 4.2, the runtime complexity of the breadth-first search strategy is $O(|V| + |E|)$, that is, the size of the adjacency-list representation of graph G. Suppose the set of names to resolve is R, it would result in a total time of $\frac{|R|(|R|-1)}{2} \times O(|V| + |E|)$, that is, $O(|R|^2 \times (|V| + |E|))$, and thus a more efficient search strategy is required to reduce the time complexity. We will see from next subsection that the longer a path is, the less contribution it makes to the total similarity. So, in practice the path search process should terminate when it reaches a predefined path length limitation (denoted by L). In other words, only the valid paths with no more than $(L - 1)$ intermediate nodes would be saved.

Here we present a dynamic programming algorithm which solves the problem by combining the solutions to independent subproblems. In general, dynamic programming solves every subproblem just once and then saves its answer, avoiding the overhead of computing the answer to the subproblems once again. In practice $L = 4$ is enough for most ambiguous names, and we start from each node $v \in R$ and take $\frac{L}{2} = 2$ steps forward, saving subpaths into memory. For any pair of nodes $v_i, v_j \in R$, we check whether there exists any intermediate node shared by any two subpaths starting from v_i and v_j, respectively, and if the answer is yes, two subpaths would be merged to form a valid path linking v_i and v_j.

Given a graph $G(V, E)$ and the set of nodes $R \subseteq V$ representing names to resolve, several notations and structures are explained as follows.

(1) *subpath*(r) denotes the set of subpaths starting from any node $r \in R$. For example, the value of *subpath*(r_6) in Figure 1 is {Wei Wang — Jian Pei, Wei Wang — Jian Pei — Jiawei Han, Wei Wang — Baile Shi, Wei Wang — Baile Shi — Jian Pei}.
(2) *path*(r_i, r_j) denotes the set of paths connecting nodes r_i and r_j, where $r_i, r_j \in R$. It is the output of Algorithm 1.
(3) I denotes the set of nodes in V-R which are reachable from at least one node in R. For example, the value of I in Figure 1 is the set of all the nodes in Figure 1 except the six "Wei Wang"s.

Algorithm 1. Valid Path Search

```
$Q \leftarrow \emptyset$;
forall $r \in R$ do
    ENQUEUE($Q$, $r$);
    while $Q \neq \emptyset$ do
        $u \leftarrow DEQUEUE$(Q);
        forall $v \in Adj[u]$ do
            forall $s \in step(r, v)$ do
                if $s.length = 1$ then
                    if $v \notin I$ then
                        insert $v$ into $I$;
                    if $r \notin source(v)$ then
                        insert $r$ into $source(v)$;
                    $path$="$r$—$v$";
                    insert $path$ into $subpath(r)$;
                    ENQUEUE($Q$, $v$);
                else if $s.length = 2$ then
                    $path$="$r$—$s.intermediate$—$v$" ;
                    if path is simple and valid then
                        if $v \notin I$ then
                            insert $v$ into $I$ ;
                        if $r \notin source(v)$ then
                            insert $r$ into $source(v)$ ;
                        insert $path$ into $subpath(r)$ ;
forall $v \in I$ do
    forall pair $(r_i, r_j)$, such that $r_i, r_j \in source(v)$ do
        forall $path_i \in subpath(r_i)$ such that $path_i$ ends with $v$ do
            forall $path_j \in subpath(r_j)$ such that $path_j$ ends with $v$ do
                $path \leftarrow$ merge $path_i$ and $path_j$ ;
                if path is valid then
                    insert $path$ into $path(r_i, r_j)$;
forall pair $(r_i, r_j)$, such that $r_i, r_j \in R$ do
    Output $path(r_i, r_j)$;
```

(4) *source*(v) denotes a set of nodes in R which are reachable from v within two steps where v is a node in I. For example, the value of *source(Philip S. Yu)* in Figure 1 is $\{r_1, r_2, r_4\}$.

(5) *step*(r, v) denotes a set of paths with length 1 or 2 that begin with $r \in R$ and end with $v \in V$. For each path $p \in step(r, v)$, *p.length* denotes the length of the path and *p.intermediate* denotes the intermediate node if the length of the path is 2. For example, the value of *step*(r_1, *Jiawei Han*) in Figure 1 is {{*length=1, intermediate=NULL*}, {*length=2, intermediate=Jiong Yang*}, {*length=2, intermediate=Philip S. Yu*}}.

We can see the details in Algorithm 1. The algorithm can be divided into three parts: lines 1 to 23, lines 24 to 30, and lines 31 to 32. The first part searches for all the simple and valid paths whose length is less than or equal to two for each of the nodes to be resolved. The second part merges the subpaths obtained from the first part. For each intermediate node v belonging to I, each pair of subpaths that end with v will be merged to a full path. The full path is inserted into $path(r_i, r_j)$ if it is valid. The last part outputs $path(r_i, r_j)$.

It is a nontrivial task to conduct the runtime complexity analysis of the algorithm, because it depends on the structure of any path starting from every node $r \in R$. However, provided some statistics we can estimate the worst-case complexity. Assume that we know the average number of edges pointing to a node x in the graph G is ξ, then on average there are ξ^2 subpaths of length 2 pointing to node x. In the worst case, there can be ξ^4 paths of length 4, $2 \times \xi^3$ paths of length 3, and ξ^2 paths of length 2, linking any given two nodes in R. Thus, in the worst case there are $O(\xi^4)$ simple and valid paths linking two nodes in R. Since the search operation in line 11 is implemented by a map structure in C++, it takes us at most $O(log(|R|))$ time. Thus we can estimate the worst-case runtime of the first procedure ranging from line 2 to line 31 to be $|R| \times \xi^2 \times (log(\xi^2) + log(|R|))$. However, for the second procedure ranging from line 32 to line 43, aggregate analysis is adopted to estimate the complexity. That is because the total number of paths would not exceed $O(\xi^4)$ no matter how complex the paths' structures are. Thus the algorithm would run in time at most $O(|R| \times \xi^2 \times (log(\xi^2) + log(|R|)) + \xi^4)$. We can view ξ as a constant number(in our experiments it is about 3), thus the whole algorithm runs in time $O(|R| \times log(|R|))$. The runtime results of this step for 9 ambiguous names are shown in details in Section 5.2, we can see that for all names the runtime for this step is less than 2 seconds.

4.3 Similarity Computation

4.3.1 Similarity Function. The length of the shortest path in a graph can be utilized to measure the proximity between two nodes. Furthermore, Kalashnikov and Mehrotra [2006] and Yin et al. [2007] propose *connection strength* to capture how strongly two nodes are connected to each other through relationships. Differently from all metrics introduced in prior work, we try to measure the similarity between two nodes by a simple approach in GHOST. The *similarity* denotes the confidence of two nodes corresponding to the same author.

Given two nodes in the coauthorship graph, we can use the path selection approach introduced in the preceding section to find all valid paths, which will be used to compute the similarity of the two nodes based on the following heuristics.

— The shortest path is the most indicative one among all valid paths in similarity computation, and the shorter it is, the more similar two nodes may be.
— The more paths there exist between two nodes, the more "similar" the two nodes may be.

A good similarity function should reflect the preceding two points. They remind us of Ohm's Law which is used to calculate the total resistance of a parallel circuit. More precisely, in the subject of *principle of circuit*, the reciprocal of the total resistance (also called effective resistance) of the circuit is equal to the sum of the reciprocals of the resistances of each component. In other words, the total inductance is equal to the sum of all inductances of each component. Notice that the total resistance is less than the smallest resistance, and the more resistance you add to a parallel circuit the lower the total resistance becomes.

Inspired by the design philosophy in calculating the resistance of a parallel circuit, we propose a similar proximity function for GHOST. The length of each path is compared to a resistance, and the proximity between two nodes is inversely proportional to the "total resistance." In GHOST, similarity(or proximity) is set to the negative "total distance." Formally, suppose we get *sum*(i, j) valid paths connecting authors r_i and r_j, then the similarity function can be described in the following mathematical formula. We have

$$Sim(r_i, r_j) = -1/\{\sum_{h=1}^{sum(i,j)} \frac{1}{l_h}\}, \tag{1}$$

where l_h denotes the number of intermediate nodes along the hth path.[1] The value of $Sim(r_i, r_j)$ ranges from $-\infty$ to 0 and is obviously a symmetric measure, that is, $Sim(r_i, r_j) = Sim(r_j, r_i)$. The similarity is equal to 0 only when there exist an infinite number of valid paths connecting two specified nodes, which is the ideal condition. On the other hand, for the sake of simplicity, we set the similarity between two isolated nodes not to $-\infty$ but to a fixed negative value (e.g., -10), leading to no change of final results.

Formula 1 works well in most cases. Travers and Milgram [1969] conducted a famous experiment in 1969 and their result brings some critical information to our problem. They found that in a large population, every individual is on an average six "steps" away from one another, even in the case when two individuals are randomly selected. The experiment is also associated with the famous phrase "six degrees of separation." This phenomenon indicates that although we have introduced a predefined path length limitation, the valid path selection process in GHOST can still find most valid paths. On the other hand, this notion of six degrees of separation indicates that every pair of authors could be highly likely connected to each other when we constrain the problem in the active academic community (or social network), even if they are different persons. Thus it suggests that the contributions made by relatively long paths need to be weakened sharply. As a result, we need to modify formula (1) in order to reflect this point. Here a simple example as shown in Figure 4 can also be used to illustrate the necessity to modify formula (1). Say there are four valid paths each having four intermediate nodes as shown in the upper part of Figure 4, the similarity between "CN_s" and "CN_e" is calculated as follows according to formula (1).

$$Sim(CN_s, CN_e) = -1/\{\frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4}\} = -1$$

Similarly, the similarity of the starting and ending nodes depicted in the lower graph of Figure 4 is computed to be -1 too, according to formula (1). This means formula (1) is unable to differentiate the two cases shown in Figure 4. But in general, two authors connected to each other by a coauthor are more likely to be the same

[1]Note that l_h can be zero if r_i and r_j represent two different authors who ever coauthored the same paper. However, this rarely happens in the real world and in this case it is impossible to differentiate r_i from r_j based on only the coauthor information. In our framework we will merge the two nodes of r_i and r_j into one node when this happens.

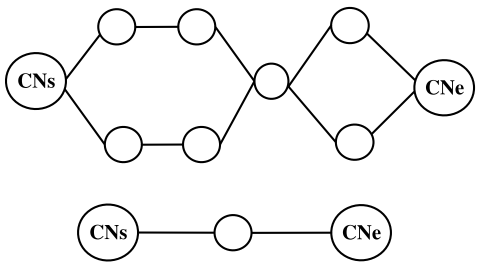

Fig. 4. Two types of connection for comparison.

person than another two authors connected by multiple longer paths. We therefore modify formula (1) to the following form. We have

$$Sim(r_i, r_j) = -1/\{\sum_{h=1}^{sum(i,j)} \frac{1}{f(l_h)}\}, \tag{2}$$

where $f(l_h)$ is a monotonic increasing function, and satisfies the constraint: $f(l_h) \geq l_h$. Polynomial in one indeterminate and exponential functions are good choices. Nevertheless we require that the corresponding value of the similarity function decrease slowly when the independent variable is relatively small, and sharply when it is large. This constraint determines that an exponential function is an ideal model. In our experiments, we set $f(l_h) = (\alpha)^{(l_h-1)}$, where $\alpha = 2.1$.

4.3.2 Discussion. We must note here that it is not novel to employ coauthorship to define closeness between any two objects, which has been adopted by several previous work, such as *relational similarity* in Bhattacharya and Getoor [2007] and interobject *connection strength* in Kalashnikov and Mehrotra [2006]. Nevertheless, the work of Bhattacharya and Getoor [2007] considers only collaboration paths of length three and assigns equal weights to paths of different lengths. There is no more systematic description on this topic. While it seems more complicated for the method proposed in Kalashnikov and Mehrotra [2006] to relate weights to connection strengths, a group of equations was proposed and an iterative solution for a nonlinear programming problem was called for to determine the weights. Relatively speaking, as we state earlier, the similarity measure motivated by Ohm's Law seems very intuitive, and requires less computation. As one step of GHOST, it helps achieve good performance which outperforms (or is at least comparable to) some state-of-the-art algorithms.

4.4 Name Clustering

In preceding sections, we developed methods for valid path selection and similarity computation between any two nodes in a coauthorship graph, which can be used to compute the similarity matrix for a set of publications that need to be disambiguated. Given a similarity matrix, GHOST tries to group "different" authors into clusters so that each cluster corresponds to a distinct author. Similarly to traditional clustering algorithms, our objective is to produce high-quality clusters with high intraclass similarity and low interclass similarity.

Our clustering problem has several features. First, the similarity measure defined in formula (2) means that the corresponding distance (which is defined as the negative similarity here) metric does not satisfy the triangle inequality (namely, $-Sim(r_i, r_k) < -Sim(r_i, r_j) - Sim(r_j, r_k)$ does not hold). One such situation is that when author r_j is a coauthor with author r_i and author r_k separately, but author r_i has never been a coauthor with author r_k. Second, the number of clusters is completely unknown. Third, the ideal clustering results are unique and unchangeable, and therefore it is easy to evaluate the quality of clustering results. Many algorithms have been devised to solve different clustering problems within different backgrounds. K-means and hierarchical clustering algorithms are two popular ones. K-means is sensitive to the initial choice and outliers, and needs the user to specify the number of clusters, thus, K-means is not suitable for our problem. Agglomerative hierarchical clustering method is adopted in DISTINCT [Yin et al. 2007], which presents a relatively sophisticated metric to measure the similarity between two clusters.

Differently from any prior clustering methods used in related work, we adopt a powerful new clustering algorithm called *Affinity Propagation* (AP for short) for GHOST. Frey and Dueck [2007] introduced it first based on the message-passing techniques from the perspective of max-sum algorithm in a factor graph. Unlike most other clustering methodologies, AP does not need the user to specify a fixed number of potential cluster centers in advance, but instead chooses to simultaneously consider all data points as potential cluster centers (also known as *exemplars*). From this perspective, the AP algorithm is quite suitable for clustering in the name disambiguation problem setting. Following we introduce its main ideas, and for more details please refer to Frey and Dueck [2007, 2008] and Brusco and Köhn [2008].

First, besides a similarity matrix with $n \times n$ elements, AP also takes as input a self-similarity $Sim(k, k)$ for each data point k, which is named "preference." The larger is the preference of point k, the more likely it is chosen as an exemplar. Thus the final number of clusters is influenced by the values of input exemplars (and the message-passing procedure). Without any prior knowledge about the underlying clusters, AP recommends that preferences be set to a global (shared) value, the median of all the input similarities in order to produce a moderate number of clusters.

Next, every pair of data points exchanges two kinds of messages. The *responsibility* $r(i, k)$ sent from point i to candidate point k indicates how well-suited point k is as the exemplar for point i in contrast to other potential exemplars, while the *availability* $a(i, k)$ sent from candidate point k to point i indicates how much support point k has received from other points for being an exemplar. The responsibility and availability are set to new values through the following formulae (3) and (4) at the beginning of each iteration until exemplars emerge.

$$r(i,k) = s(i,k) - \max_{k' \, s.t. k' \neq k} \{a(i,k') + s(i,k')\} \tag{3}$$

$$a(i,k) = \min\{0, r(k,k) + \sum_{i' \, s.t. i' \notin \{i,k\}} \max\{0, r(i',k)\}\} \tag{4}$$

In formula (3), the term $s(i, k)$ is the similarity value between point i and point k. At any point during the procedure of affinity propagation, extra computation is utilized to check whether changes in the messages fall below a threshold or the local decision stays constant for some number of iterations. Finally, for point i, the value of k that maximizes $a(i, k) + r(i, k)$ identifies the exemplar for point i.

Because of its simplicity, general applicability, and distinguished performance, Affinity Propagation has been applied to image categorization [Dueck and Frey 2007], face clustering [Frey and Dueck 2007], treatment portfolios construction [Dueck and Frey 2008], and gene expression [Leone et al. 2007]. GHOST tries to draw on AP to group authors with identical names into disjoint clusters by taking as input the values of similarities generated in the way described in preceding subsections. As our

Table III. An Example of Dense Author: Wei Wang

Affiliation	Title	Coauthors
UNC-Chapel Hill	ApproxMAP: Approximate Mining of Consensus Sequential Patterns	H.C. Kum, Jian Pei, Dean Duncan
	Mining long sequential patterns in a noisy environment	Jiong Yang, Philip S. Yu, Jiawei Han
	Clustering by pattern similarity in large data sets	Haixun Wang, Jiong Yang, Philip S. Yu
Fudan University	Scalable mining of large disk-based graph databases	Chen Wang, Jian Pei, Yongtai Zhu, Baile Shi
	Mining sequential patterns with with constraints in large databases	Jian Pei, Jiawei Han
	LOCI: Load Shedding through Class-Preserving Data Acquisition	Peng Wang, Haixun Wang, Baile shi, Philip S. Yu

experimental study will show, AP can bring significant performance improvement to our framework while achieving both high precision and recall.

4.5 User Feedback

Until now, we have presented a four-step version of GHOST which can provide satisfactory performance in most cases. One possible extension is to provide an option to incorporate some user feedback in order to further improve the performance. As far as we know, there is no reported study which uses user feedback in the problem settings of reference disambiguation and name disambiguation. Nevertheless the reason why it is imperative to utilize user feedback can be observed obviously in the following case.

Let us focus on the name of “Wei Wang”. There are at least 208 papers written by authors with this name, which belong to 50 distinct authors. After the four-step processing, we would find that many papers written by two distinct authors, one is affiliated with University of North Carolina at Chapel Hill but another is from Fudan University of China, are grouped into one cluster. It leads to very low-quality result as almost half of the papers coauthored with “Wei Wang” belong to these two authors. We investigated their respective coauthors and papers’ topics, interestingly, we found that there are at least four direct coauthors, “Jiawei Han”, “Philip S. Yu”, “Jian Pei”, and “Haixun Wang”, shared by both of them, and their research interests are by and large both in data mining and database systems, as we can see in Table III. Apart from direct coauthors, they share considerable coauthor community overlap through linkages starting from the common direct coauthors, which leads to high similarities between any pair of nodes. Here we adopt two notions to handle it. When we resolve a name and find that there is at least one direct coauthor shared by two distinct authors, any author with the name is referred to as a “*dense author*,” otherwise it is referred to as a “*sparse author*.” To deal with the low performance caused by dense authors, user feedback is adopted to enhance the performance.

User feedback is widely used in the area of information retrieval to reflect users’ opinions on the initial results and improve query reformulation. More precisely, once user feedback is received, we can adjust the parameters in the steps of valid path selection and clustering (corresponding to Sections 4.2 and 4.4, respectively) to solve the problem of poor performance caused by dense authors. In general, three strategies are available to get a satisfactory evaluation in response to user feedback.

— Decrease the number of valid paths. In GHOST, it is supposed initially that all valid paths with length no more than L will be found out, but we realize it would produce high similarities between any pair of dense authors. Therefore, it is reasonable to retrieve only the K shortest paths which will be used to calculate the value of similarity.

— Increase the depth while searching for valid paths. It helps unearth underlying and relatively long relationships, and would not cause overly high similarities considering that we have decreased the number of valid paths.
— Adjust the value of *preferences* mentioned in the clustering process of Affinity Propagation. In most cases authors are sparse ones, the default preference is set to be the median of all similarities and it will result in a moderate number of clusters, which is close to the real number of distinct authors. However, for dense authors, the default value would produce some big cluster that should be split into two or more. Thus we need to adjust the value of preferences for Affinity Propagation clustering.

As we mentioned earlier, dense authors are rare cases, thus user feedback is seldom used in GHOST, however, it does provide a way to improve algorithm performance further by incorporating user feedback for dense authors. In next section, we will evaluate different strategies and their combinations as the responses to user feedback for a typical dense author, Wei Wang.

5. EXPERIMENTAL RESULTS

5.1 Test Environment and Datasets

In this section, we report our empirical study to test the effectiveness of the proposed name disambiguation framework, GHOST. We conducted experiments on two datasets. One is the famous computer science bibliography database, DBLP [DBLP 2010], another one is the PubMed dataset [PubMed 2010], which is a famous digital archive of biomedical and life sciences publications. In our experiments we used a subset of publications for DBLP and PubMed published in recent years, which consists of a total of 564153 publications corresponding to 417923 authors for DBLP and 202959 publications corresponding to 765190 authors for PubMed, respectively. Similarly to Yin et al. [2007], publications whose author identities cannot be found are removed, so are those with only one author, each of which would result in an isolated “island” in the graph. In addition, we collected those venues where any publications with ambiguous authors were published and constructed a venue list, and then publications whose venues are not in the list were also eliminated. Thus, the number of papers and the number of authors left vary with the name we chose to resolve.

To evaluate the performance of GHOST, we need to provide labeled sets of publications for comparison. First, we checked the email addresses of the specified authors in every publication. Those email addresses shared by two or more authors imply that these authors correspond to the same person. Next, we examined the specified authors' homepages which are available on the Internet. Generally speaking, papers written by one person can be found through the publication list on his or her homepage. If this does not work, which means that some publications cannot be correctly labeled, we then turned to affiliation information or sent query emails to available authors to ask them to help point out their own publications. We got 12 labeled authors for DBLP dataset and 8 labeled authors for PubMed dataset to test the differences between the true clusters and clusters generated by GHOST. Note that although we tried our best to send emails to many authors of the PubMed publications, we only got feedback from 8 authors, and unfortunately among these authors only one author name (i.e., David J Hunter) is ambiguous.

Similarly to Yin et al. [2007], we measure the performance of GHOST by precision, recall, and f-score in the context of pairwise comparison based on traditional information retrieval measures. Suppose the set of real clusters is C_{std} and the set of clusters generated by GHOST is C_{ghost}. Let TP (true positive) be the number of pairs of publications that are in the same cluster in both C_{std} and C_{ghost}, FP (false positive) be the

Table IV. Runtime Evaluation of GHOST for 12 Authors (DBLP, Without User Feedback)

Name	# Vertices in Graph	# Edges in Graph	Building Graph(s)	Valid Path Selection(s)	Clustering (s)	Total Time(s)
Cheng Chang	22700	49165	0.578	0.001	0.125	0.704
Hui Fang	9607	22974	0.234	0.001	0.109	0.344
Ajay Gupta	45603	113844	1.14	0.031	0.109	1.28
Rakesh Kumar	45363	126132	1.296	0.032	0.125	1.453
Yi Li	69076	173510	1.703	0.016	0.188	1.907
Bing Liu	60601	159735	1.688	0.094	0.609	2.391
Jim Smith	19503	43690	0.485	0.001	0.11	0.596
Michael Wagner	58647	136859	1.5	0.016	0.141	1.657
Lei Wang	105545	295114	2.672	0.031	0.547	3.25
Wei Wang	120926	334740	3.625	1.547	3.204	8.376
Bin Yu	45347	113733	0.921	0.001	0.234	1.156
Jing Zhang	93444	244397	2.406	0.078	0.25	2.734
(Average)	**58030**	**151158**	**1.521**	**0.154**	**0.479**	**2.154**

number of pairs of publications in the same cluster in C_{ghost} but not in C_{std}, and FN (false negative) be the number of pairs of publications in the same cluster in C_{std} but not in C_{ghost}. The two measures, *precision* and *recall*, are defined as

$$precison = \frac{TP}{TP + FP}, \qquad recall = \frac{TP}{TP + FN}, \tag{5}$$

while *f-score* is the harmonic mean of precision and recall, and is defined as follows.

$$f\text{−}score = \frac{2 \times precision \times recall}{precision + recall} \tag{6}$$

5.2 Runtime Analysis of GHOST

GHOST consists of five steps, among which graph construction, valid path selection, and clustering are the most time-consuming steps. In this subsection we list in Table IV the runtime it would take to accomplish each of the three steps for 9 ambiguous authors in DBLP. All experiments were carried out on a machine with 2.00 GHz Intel(R) Core(TM)2 Duo CPU and 2Gigabyte memory. Note that the fourth column depicts the time it takes to build a corresponding graph, and the number of vertices and edges in the graph $G(V, E)$ are presented in columns 2 and 3. Column 5 demonstrates the time needed to find all valid paths, while the sixth column shows the runtime for the clustering step. Similarly, we also tested the runtime efficiency of each step for the 8 authors in PubMed, and the results are shown in Table V. From Tables IV and V we see that GHOST takes several seconds on average to disambiguate an author name, thus it is quite efficient.

5.3 Performance Evaluation of GHOST

The performance results on name disambiguation of the 12 authors in DBLP dataset are shown in Table VI, together with comparisons in terms of some undirect but necessary measures, such as the number of publications, the number of distinct persons, and the number of estimated clusters (i.e., persons). The last row in Table VI shows the average values of precision, recall, and f-score for all 12 authors. Similarly, we also tested GHOST's performance with the 8 manually labeled authors in PubMed. The results are shown in Table VII.

For sparse authors, we set L in the experiments, the path length limit, to be 4, and selected a fixed negative value of -10 to replace $-\infty$ (i.e., the similarity value between

Table V. Runtime Evaluation of GHOST for 8 Authors (PubMed, Without User Feedback)

Name	# Vertices in Graph	# Edges in Graph	Building Graph(s)	Valid Path Selection(s)	Clustering (s)	Total Time(s)
Paul M Ridker	12525	80971	0.391	0.016	0.078	0.485
George Perry	9259	32356	0.250	0.001	0.062	0.313
R Buhl	3557	16943	0.141	0.093	0.094	0.328
Julie E Buring	13507	84506	0.469	0.015	0.078	0.562
George L Bakris	5892	34277	0.265	0.032	0.188	0.485
A Méjean	396	1547	0.031	0.001	0.078	0.110
David J Hunter	19887	134994	0.687	0.032	0.109	0.828
JoAnn E Manson	9497	47542	0.344	0.001	0.141	0.486
(Average)	**9315**	**54142**	**0.322**	**0.024**	**0.104**	**0.450**

Table VI. Name Disambiguation Quality Evaluation for 12 Authors (DBLP, Without User Feedback)

Name	# Papers	# Authors	# Estimated Clusters	Result Evaluation		
				precision	recall	f-score
Cheng Chang	14	4	4	1.00	1.00	1.00
Hui Fang	20	3	3	1.00	1.00	1.00
Ajay Gupta	16	4	5	1.00	0.93	0.96
Rakesh Kumar	36	2	2	1.00	1.00	1.00
Yi Li	41	19	20	0.98	0.93	0.95
Bing Liu	89	6	18	0.85	0.27	0.41
Jim Smith	20	3	5	1.00	0.86	0.92
Michael Wagner	37	10	13	1.00	0.54	0.70
Lei Wang	95	39	39	0.99	1.00	0.99
Wei Wang	141	14	13	0.51	0.75	0.61
Bin Yu	53	11	17	0.96	0.68	0.79
Jing Zhang	60	25	25	1.00	1.00	1.00
(Average)	—	—	—	**0.941**	**0.83**	**0.86**1

Table VII. Name Disambiguation Quality Evaluation for 8 Authors (PubMed, Without User Feedback)

Name	# Papers	# Authors	# Estimated Clusters	Result Evaluation		
				precision	recall	f-score
Paul M Ridker	18	1	1	1.00	1.00	1.00
George Perry	14	1	1	1.00	1.00	1.00
R Buhl	33	1	1	1.00	1.00	1.00
Julie E Buring	17	1	1	1.00	1.00	1.00
George L Bakris	19	1	1	1.00	1.00	1.00
A Méjean	33	1	2	1.00	0.94	0.97
David J Hunter	26	3	6	1.00	0.89	0.94
JoAnn E Manson	16	1	2	1.00	0.88	0.93
(Average)	—	—	—	**1.00**	**0.964**	**0.98**

two isolated nodes).[2] We must note that a choice of any value (for the smallest possible similarity value) which is smaller than -10 would not affect the accuracy of clustering results. In general, GHOST shows good performance to successfully disambiguate almost all the names in DBLP and PubMed datasets, with an average f-score of 0.861 for DBLP and an average f-score of 0.98 for PubMed. Note that all the scores in Tables VI and VII were obtained without any user feedback, and we will test the effectiveness of user feedback for dense authors like "Wei Wang" in Section 5.4.

[2]Other values for path length limit can also be used in order for GHOST to achieve better performance for certain names.

Table VIII. Effectiveness Evaluation of User Feedback for "Wei Wang" (DBLP)

# Papers	# Authors	Feedback Response	# Clusters	Result Evaluation		
				precision	recall	f-score
141	14	No Feedback	13	0.51	0.75	0.61
		(2 edges, all paths)	23	1.00	0.53	0.69
		(3 edges, 4 shortest)	19	0.95	0.57	0.71
		(4 edges, 5 shortest)	13	0.79	0.83	0.81
		(4 edges, 3 shortest)	13	0.85	0.91	0.88
		(4 edges, 2 shortest)	13	0.92	0.95	0.94

5.4 Effectiveness Evaluation of User Feedback

As we described in Section 4.5, for dense authors we can make use of user feedback to enhance the performance. It is a natural idea to decrease the number of valid paths and increase the depth of valid paths in order to differentiate the dense authors. Several strategies and their combinations were evaluated in our experiments, and we present the experimental results for the name of "Wei Wang" in Table VIII, where (2 edges, all paths) denotes that any valid path is retrieved as long as its number of edges is no greater than two, while (3 edges, 4 shortest) denotes that only the top 4 shortest valid paths with a length no longer than 3 are taken into consideration. We can see significant improvement of performance can be achieved by incorporating the user feedback.

5.5 GHOST vs. DISTINCT

We also compared GHOST with one of the state-of-the-art name disambiguation algorithms, DISTINCT, using both the DBLP and PubMed databases. To make the comparison between GHOST and DISTINCT more fair, we changed DISTINCT to consider coauthorship only. In the experiments, we adopted 12 names from DBLP database which have been used in evaluating DISTINCT algorithm [Yin et al. 2007]. Each of these 12 names has at least two distinct authors. As we do not have the labeled data for the name of "Joseph Hellerstein", we did not use it in the comparison. The f-score comparison results of the two algorithms for DBLP database are shown in Table IX. We can see that GHOST has comparable performance with DISTINCT for all these names except "Wei Wang" and "Bing Liu". For some dense authors like "Wei Wang", we could exploit user feedback to improve the algorithm performance. For example, by using user feedback, GHOST can achieve a f-score of 0.94, which is much better than DISTINCT. The performance of GHOST for the name of "Bing Liu" can be improved by adopting the agglomerative hierarchical clustering paradigm, which can be seen from Table XI. For database PubMed, Table X shows the comparison results. We see that GHOST has the same or better performance than DISTINCT for all the names from the PubMed database.

5.6 Discussion

From the preceding experimental results we see that GHOST uses less information (i.e., only the coauthorship information), but achieves very good performance in comparison with the state-of-the-art name disambiguation algorithm, DISTINCT. As DISTINCT adopts the agglomerative hierarchical clustering framework, we also tested GHOST under the same clustering framework in order to identify which factor contributes more between the new similarity measure and the new clustering method. We found that with a well-tuned threshold the agglomerative hierarchical clustering method can achieve almost the same clustering performance as the affinity propagation clustering algorithm. For example, among the 20 names used in our experiment (from the DBLP and PubMed datasets) both the affinity propagation and

Table IX. Performance Comparison (DBLP, GHOST Without User Feedback vs. DISTINCT Using Only the Coauthorship Attribute)

Name	GHOST (without user feedback)			DISTINCT (coauthorship only)		
	precision	recall	f-score	precision	recall	f-score
Cheng Chang	1.00	1.00	**1.00**	1.00	1.00	**1.00**
Hui Fang	1.00	1.00	**1.00**	1.00	1.00	**1.00**
Ajay Gupta	1.00	0.93	**0.96**	1.00	0.93	**0.96**
rakesh kumar	1.00	1.00	**1.00**	1.00	1.00	**1.00**
Yi Li	0.98	0.93	0.95	1.00	0.93	**0.97**
Bing Liu	0.85	0.27	0.41	1.00	0.51	**0.67**
Jim Smith	1.00	0.86	**0.92**	1.00	0.81	0.89
Michael Wagner	1.00	0.54	**0.70**	1.00	0.44	0.61
Lei Wang	0.99	1.00	0.99	1.00	1.00	**1.00**
Wei Wang	0.51	0.75	0.61	0.99	0.76	**0.86**
Bin Yu	0.96	0.68	**0.79**	1.00	0.64	0.78
Jing Zhang	1.00	1.00	**1.00**	1.00	0.97	0.98

Table X. Performance Comparison (PubMed, GHOST Without User Feedback vs. DISTINCT Using Only the Coauthorship Attribute)

Name	GHOST (without user feedback)			DISTINCT (coauthorship only)		
	precision	recall	f-score	precision	recall	f-score
Paul M Ridker	1.00	1.00	**1.00**	1.00	1.00	**1.00**
George Perry	1.00	1.00	**1.00**	1.00	1.00	**1.00**
R Buhl	1.00	0.93	**0.96**	1.00	0.54	0.70
Julie E Buring	1.00	1.00	**1.00**	1.00	1.00	**1.00**
George L Bakris	1.00	1.00	**1.00**	1.00	0.79	0.88
A Méjean	1.00	0.94	**0.97**	1.00	0.88	0.94
David J Hunter	1.00	0.89	**0.94**	1.00	0.75	0.85
JoAnn E Manson	1.00	0.88	**0.93**	1.00	0.46	0.63

Table XI. Performance Comparison of the Two Clustering Algorithms (Without User Feedback)

Name	Agglomerative Hierarchical Clustering			Affinity Propagation Clustering		
	precision	recall	f-score	precision	recall	f-score
Yi Li	1.00	0.93	**0.97**	0.98	0.93	0.95
Bing Liu	0.89	0.85	**0.87**	0.85	0.27	0.41
Lei Wang	0.86	1.00	0.92	0.99	1.00	**0.99**
Wei Wang	0.36	0.96	0.53	0.51	0.75	**0.61**
Bin Yu	1.00	0.68	**0.81**	0.96	0.68	0.79
George L Bakris	1.00	0.79	0.88	1.00	1.00	**1.00**

agglomerative hierarchical clustering approaches have the same clustering results for 14 names. The six names for which the two clustering algorithms have different results are listed in Table XI. We see that for names of "Lei Wang", "Wei Wang", and "George L. Bakris" affinity propagation is the winning clustering algorithm, while for the remaining three names agglomerative hierarchical clustering has better performance. This indicates that most of the performance gain in comparison with DISTINCT stems from the new similarity measure.

However, we must point out that GHOST has some disadvantages. First, because GHOST tries to calculate the similarity between authors based on the linkages in the coauthorship graph, a large amount of information about publications is required. Otherwise, inadequate relationship may result in low similarity of two nodes, which may cause the two nodes to be separated into two disjoint clusters. Second, GHOST is prone to distinguish authors based on their research community or experience. In other words, it works based on the hypothesis that the coauthors of an ambiguous author also collaborate with each other frequently. If one ever moved from a lab to another and there is little cooperation between such two research teams, he/she is likely to be recognized as two different researchers by GHOST. The chance would become higher if one ever changed his/her research areas or just extended them to new fields. Third, GHOST is solely based on the coauthorship information, and all the isolated nodes are removed from the coauthorship graph structure, thus it cannot handle any publications with only one author. If we do not remove any isolated nodes from the graph structure, the f-measure will be deteriorated. In this sense, DISTINCT should do better job than GHOST.

6. IMPLICATIONS TO RESEARCH AND PRACTICE

In this article we propose to use a graph structure to model the coauthorship relationship and introduce an effective graph clustering approach to name disambiguation in a digital library setting. As graph data can be used to naturally model various complex relationships among entities and their attributes, the proposed graph-based framework, GHOST, can be seen as a very promising paradigm to solve the name disambiguation problem and other entity resolution problems. Recently we have explored a similar paradigm to disambiguate and tag people names in Web search [Jiang et al. 2009]. In Jiang et al. [2009], we first extracted various tags (e.g., email address, phone number, organization name, and people name), built a graph model for these extracted tags, and applied a two-stage graph clustering method to disambiguate names in Web people search. The performance study in Jiang et al. [2009] validates the high utility of the graph-based approach too. Some potential research problems in this graph-based framework for name disambiguation include graph modeling of the entity relationships and similarity calculation among any two nodes in the graph. If the constructed graph is large, designing highly efficient graph-based name disambiguation algorithms can be also very challenging. In addition, in real life a bibliographic database usually changes over time (e.g., some new publications may be added into the database), thus to make GHOST support incremental updates could also be a very practical problem.

Although the name disambiguation problem studied in this article is search-like, that is, each time when we run the GHOST algorithm a particular name to be resolved needs to be specified, this does not mean the GHOST algorithm cannot be applied to an entire dataset. We can collect all the names in the dataset, and run the algorithm and resolve each name in batch. In fact, the GHOST algorithm has been applied in this way to solve the name ambiguity problem in the CDBLP database.[3] According to the feedback from the CDBLP research group, they found that for many Chinese author names, GHOST could generate more clusters than necessary (i.e., the set of publications with respect to the same author could be split into several clusters). To solve this problem, they further extracted and used the affiliation information of each author to refine the result of GHOST. The experience from the CDBLP group suggests that as GHOST only uses the coauthorship information, we may use it as a first step in disambiguating names for a given dataset, and then explore more auxiliary attributes (e.g., affiliation and email address) to improve the output of GHOST.

7. CONCLUSION

In this article, we have explored the problem of name disambiguation. We develop an effective five-step framework, GHOST, which employs only one type of relationship, namely, coauthorship. In GHOST, we propose a novel similarity metric and adopt the affinity propagation algorithm to group intermediate results into clusters. User feedback can also be used as an option to improve the performance if the publications belong to dense authors. There are several distinguished advantages of GHOST over the

[3]http://www.cdblp.cn

existing approaches. First, the similarity metric requires only local and simple computation, and the procedure of searching for valid paths is more efficient than random graph walk and probability propagation. Second, GHOST is free of any constructed training sets and learning models. Third, in most cases, the clustering algorithm works automatically and does not need manual supervision, although its performance may suffer for some rare dense authors. We have conducted extensive experiments on the real DBLP and PubMed databases, and the results demonstrate that GHOST can achieve both high precision and recall in comparison with the-state-of-the-art approach, DISTINCT.

REFERENCES

BASU, S., BILENKO, M., AND MOONEY, R. J. 2004. A probabilistic framework for semi-supervised clustering. In *Proceedings of the International SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'04)*. 59–68.

BEKKERMAN, R. AND MCCALLUM, A. 2005. Disambiguating web appearances of people in a social network. In *Proceedings of the International World Wide Web Conference (WWW'05)*. 463–470.

BHATTACHARYA, I. AND GETOOR, L. 2007. Collective entity resolution in relational data. *ACM Trans. Knowl. Discov. Data 1*, 1, 1–36.

BILENKO, M. AND MOONEY, R. J. 2003. Adaptive duplicate detection using learnable string similarity measures. In *Proceedings of the International SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'03)*. 39–48.

BILENKO, M., MOONEY, R., COHEN, W., RAVIKUMAR, P., AND FIENBERG, S. 2003. Adaptive name matching in information integration. *IEEE Intell. Syst 18,* 5, 16–23.

BRUSCO, M. J. AND KÖHN, H.-F. 2008. Comment on "clustering by passing messages between data points". *Sci. 319*, 726c.

CHAUDHURI, S., GANJAM, K., GANTI, V., AND MOTWANI, R. 2003. Robust and efficient fuzzy match for online data cleaning. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 313–324.

CHEN, Z., KALASHNIKOV, D. V., AND MEHROTRA, S. 2007. Adaptive graphical approach to entity resolution. In *Proceedings of the ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'07)*. 204–213.

DBLP. 2010. bibliography. http://www.informatik.uni-trier.de/∼ley/db/.

DONG, X., HALEVY, A., AND MADHAVAN, J. 2005. Reference reconciliation in complex information spaces. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 85–96.

DUECK, D. AND FREY, B. J. 2007. Non-metric affinity propagation for unsupervised image categorization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'07)*. 1–8.

DUECK, D. AND FREY, B. J. 2008. Constructing treatment portfolios using affinity propagation. In *Proceedings of the Annual Conference on Research in Computational Molecular Biology (RECOMB'08)*. 360–371.

FAN, X., WANG, J., LV, B., ZHOU, L., AND HU, W. 2008. GHOST: An effective graph-based framework for name distinction. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM'08)*. 1449–1450.

FELLEGI, I. P. AND SUNTER, A. B. 1969. A theory for record linkage. *J. Amer. Statist. Assoc. 64,* 328, 1183–1210.

FREY, B. J. AND DUECK, D. 2007. Clustering by passing messages between data points. *Sci. 315*, 972–976.

FREY, B. J. AND DUECK, D. 2008. Response to comment on "clustering by passing messages between data points". *Sci. 319*, 726d.

HAN, H., GILES, L., ZHA, H., LI, C., AND TSIOUTSIOULIKLIS, K. 2004. Two supervised learning approaches for name disambiguation in author citations. In *Proceedings of the ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'04)*. 296–305.

HAN, H., ZHA, H., AND GILES, C. L. 2005. Name disambiguation in author citations using a k-way spectral clustering method. In *Proceedings of the ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'05)*. 334–343.

HERNÁNDEZ, M. A. AND STOLFO, S. J. 1995. The merge/purge problem for large databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 127–138.

JIANG, L., WANG, J., AN, N., WANG, S., ZHAN, J., AND LI, L. 2009. GRAPE: A graph-based framework for disambiguating people appearances in web search. In *Proceedings of the IEEE International Conference on Data Mining (ICDM'09)*. 199–208.

KALASHNIKOV, D. V. AND MEHROTRA, S. 2006. Domain-independent data cleaning via analysis of entity-relationship graph. *ACM Trans. Datab. Syst. 31,* 2, 716–767.

LAWRENCE, S., GILES, C. L., AND BOLLACKER, K. D. 1999. Autonomous citation matching. In *Proceedings of the AGENTS'99 Conference*. 392–393.

LEE, D., ON, B.-W., KANG, J., AND PARK, S. 2005. Effective and scalable solutions for mixed and split citation problems in digital libraries. In *Proceedings of the International Workshop on Information Quality in Information Systems (IQIS'05)*. 69–76.

LEONE, M., SUMEDHA, AND WEIGT, M. 2007. Clustering by soft-constraint affinity propagation: Applications to gene-expression data. *Bioinf. 23,* 20, 2708–2715.

MCCALLUM, A., NIGAM, K., AND UNGAR, L. H. 2000. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proceedings of the SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'00)*. 169–178.

MINKOV, E., COHEN, W. W., AND NG, A. Y. 2006. Contextual search and name disambiguation in email using graphs. In *Proceedings of the Annual ACM SIGIR Conference on Research and Development in Information Retrieval*. 27–34.

ON, B.-W. AND LEE, D. 2007. Scalable name disambiguation using multi-level graph partition. In *Proceedings of the SIAM International Conference on Data Mining (SDM'07)*. 575–580.

ON, B.-W., ELMACIOGLU, E., LEE, D., KANG, J., AND PEI, J. 2006. Improving grouped-entity resolution using quasi-cliques. In *Proceedings of the IEEE International Conference on Data Mining (ICDM'06)*. 1008–1015.

PASULA, H., MARTHI, B., MILCH, B., RUSSELL, S., AND SHPITSER, I. 2002. Identity uncertainty and citation matching. In *Proceedings of the Conference on Advances in Neural Information Processing Systems (NIPS'02)*.

PEI, J., JIANG, D., AND ZHANG, A. 2005. On mining cross-graph quasi-cliques. In *Proceedings of the International SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'05)*. 228–238.

PUBMED. 2010. bibliography. http://www.ncbi.nlm.nih.gov/pubmed/

SARAWAGI, S. AND BHAMIDIPATY, A. 2002. Interactive deduplication using active learning. In *Proceedings of the International SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'02)*. 269–278.

TRAVERS, J. AND MILGRAM, S. 1969. An experimental study of the small world problem. *Sociometry 32,* 4, 425–443.

WINKLER, W. 1999. The state of record linkage and current research problems. Tech. rep., Statistical Research Division, U.S. Bureau of the Census.

YIN, X., HAN, J., AND YU, P. S. 2007. Object distinction: Distinguishing objects with identical names. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE'07)*. 1242–1246.

ZHANG, D., TANG, J., LI, J., AND WANG, K. 2007. A constraint-based probabilistic framework for name disambiguation. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM'07)*. 1019–1022.

Received August 2008; revised August 2010; accepted September 2010