Research Paper

JIS

Journal of Information Science
1–18
© The Author(s) 2018
Reprints and permissions:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0165551518761011
journals.sagepub.com/home/jis
SAGE

DISC: Disambiguating homonyms using graph structural clustering

Ijaz Hussain
Department of Computer Science, COMSATS Institute of Information Technology, Pakistan

Sohail Asghar
Department of Computer Science, COMSATS Institute of Information Technology, Pakistan

Abstract
Author name ambiguity degrades information retrieval, database integration, search results and, more importantly, correct attributions in bibliographic databases. Some unresolved issues include how to ascertain the actual number of authors, how to improve the performance and how to make the method more effective in terms of representative clustering metrics (average cluster purity, average author purity, K-metric, pairwise precision, pairwise recall, pairwise-F1, cluster precision, cluster recall and cluster-F1). It is a non-trivial task to disambiguate authors using only the implicit bibliographic information. An effective method 'DISC' is proposed that uses graph community detection algorithm, feature vectors and graph operations to disambiguate homonyms. The citation data set is pre-processed and ambiguous author blocks are formed. A co-authors graph is constructed using authors and their co-author's relationships. A graph structural clustering 'gSkeletonClu' is applied to identify hubs, outliers and clusters of nodes in a co-author's graph. Homonyms are resolved by splitting these clusters of nodes across the hub if their feature vector similarity is less than a predefined threshold. DISC utilises only co-authors and titles that are available in almost all bibliographic databases. With little modifications, DISC can also be used for entity disambiguation. To validate the DISC performance, experiments are performed on two Arnetminer data sets and compared with five previous unsupervised methods. Despite using limited bibliographic metadata, DISC achieves on average K-metric, pairwise-F1, and cluster-F1 of 92%, 84% and 74%, respectively, using Arnetminer-S and 86%, 80% and 57%, respectively, using Arnetminer-L. About 77.5% and 73.2% clusters are within the range (ground truth clusters $\pm$ 3) in Arnetminer-S and Arnetminer-L, respectively.

Keywords
Author name disambiguation; community detection; graph structural clustering; homonyms

1. Introduction

Bibliographic databases such as DBLP,[1] MEDLINE,[2] Cite Seer,[3] arXiv,[4] MAS,[5] Google Scholar[6] and BDBComp[7] conserve bibliographic citations and provide several services, for example, receiving items associated with a particular author, multiple searches, browsing personalisation and building communities with certain educational fields [2]. Some of the main challenges highlighted in Han et al. [2] are to have high-quality content in digital libraries (DLs) come from several sources of errors such as lack of (the enforcement of) standards, imperfect citation-gathering software, disparate citation formats, data-entry errors, ambiguous author names, decentralised generation of content (i.e. by means of automatic harvesting) and abbreviations of publication venue, titles. Among these sources of errors, a great attention is paid to ambiguous author names from the research community due to its inherent difficulty [1,3,4]. Resolving author name ambiguity from citations is referred as author name disambiguation (AND). Author name ambiguity occurs when many authors share the same name. In this case, it is too difficult to be certain about the accuracy of retrieved results. This phenomenon of citation merger, the case of two or more persons having the same name (e.g. 'Wei Wang'), is also known as mixed citations [4].

Corresponding author:
Ijaz Hussain, Department of Computer Science, COMSATS Institute of Information Technology, Park Road, Islamabad 45550, Pakistan.
Email: ijazhussain7979@hotmail.com

Generally, ambiguity in author names is resolved by means of different publication attributes such as co-authors, titles, abstract, venue, affiliation and publication year [1,4]. However, every bibliographic database does not provide all these attributes; they provide only limited information and manual annotation is not possible at such a large scale. Furthermore, in recent years, large amounts of publication data are being created and accepted by bibliographic databases that make the name ambiguity problem even more severe than what it has been in the past [1].

When we search for an author named 'Wei Wang' in DBLP (a renowned bibliographic database), we get more than 86 authors having exactly the same names 'Wei Wang'. The same situation turns out in almost all major bibliographic databases [4]. This example is merely a representative of the magnitude of the mixed citation problem which motivated us. Additional motivation comes from the innumerable cases of Asian authors who have similar surnames, specifically in DBLP bibliographic database, which is loaded with such cases. Moreover, DBLP and other bibliographic databases are frequently not informative enough in their metadata and lack pieces of important information such as an author's affiliation, email and publication references.

Several AND methods have been proposed in the literature [2–27]. These methods have improved the situation somehow, but still, there is a space for improvement in current solutions. Supervised methods require a labelled data set for the training of the model and are less scalable due to the requirement of training thousands of models for each ambiguous author Han et al. [9], Ferreira et al. [10], Tran et al. [17] and Levin et al. [18]. The majority of the unsupervised AND methods assume that a number of ambiguous authors/clusters 'K' is known in advance [2,17,18,20]. Some techniques are not scalable when the number of ambiguous authors increases [3–5,17,25]. Others require hidden or extra information from the web or user feedback for disambiguation [11,13,25,27,28].

In this article, a novel method 'DISC: Disambiguating homonyms usIng graph Structural Clustering' is proposed to resolve homonyms. DISC is a graph-based method in which citation data set is pre-processed and ambiguous author blocks are created, and co-authors graph and feature vectors are constructed using the pre-processed citation data. Then, 'gSkeletonClu: Revealing Density-Based Clustering Structure from the Core-Connected Tree of a Network' is used to identify hubs, outliers and clusters of vertices (communities) from the co-author's graph [29]. DISC resolves homonyms by splitting these communities across the hub nodes if their title feature vector similarity is less than a predefined threshold and using graph operations. DISC uses only co-authors and titles to resolve the homonyms. Co-authors information is considered a more discriminating feature than other citation features [3,4,8,19]. However, distinct from existing techniques, we exploit the community detection algorithm in combination with feature vector similarity and graph operations to disambiguate homonyms. To the best of our knowledge, DISC algorithm is the first that uses a gSkeletonClu community detection algorithm and feature vector similarity to disambiguate homonyms.

DISC performance is evaluated by comparing it with three unsupervised graph-based and two non-graph-based AND methods using Arnetminer data sets [4,8,11]. DISC proved to be overall better than these methods from the perspective of frequently used clustering evaluation metrics despite the fact that DISC uses only the co-authors and titles. The main contributions of this work are followed:

- We design the AND task as co-author's graph that only uses implicit information of bibliographic databases. This work is motivated by the needs of a method that works for almost all bibliographic databases and does not require any explicit information. DISC does not require costly training data, hidden information about the number of clusters or expert knowledge of the domain.
- We propose a novel solution 'DISC' that exploits graph structural clustering and feature vector similarity to solve the name ambiguity. To the best of our knowledge, the identification of outliers in graph-based methods has been addressed for the first time. DISC has utilised a recently introduced community detection technique gSkeletonClu: a graph skeleton–based structural clustering algorithm for clustering to identify hub, outliers and clusters of nodes. The advantage of DISC is that it resolves homonyms using the output of clustering, feature vector similarity and graph operations.
- Experiments on Arnetminer – a real-world data set – are performed to empirically validate the effectiveness of DISC. The results show that DISC performance is overall better than three graph-based and two non-graph-based methods. This is a confirmatory contribution and indirectly proves our design and claim.

The rest of this article is structured as follows. 'Related work' outlines related works and critically reviews their techniques. In 'Proposed methodology of DISC', the proposed methodology is discussed in detail along with the definitions of graph structural clustering. We test this approach on real-world data sets from Arnetminer and benchmark it against other methods in 'Performance evaluation of DISC'. Finally, we conclude with a discussion of contributions and future research directions in 'Conclusion and future work'.

2. Related work

Three brief surveys about AND are presented in the literature that defined the basics of the AND problem and overviewed some representative AND methods [1,30,31]. According to Ferreira et al. [1], AND methods can be categorised as author assignment methods [9,10,15–17] and author grouping methods [3,4,7,8,11,20].

Onodera et al. [21] proposed a methodology for target author from the false homonym authors. They used publications features such as affiliation, title and co-authors for discriminating oeuvre of an author. Zhu et al. [22] proposed a hybrid name disambiguation framework that used the traditional information (co-authors) along with web page genre information. This framework consisted of two main steps – web page genre identification and re-clustering model. Dempster–Shafer theory (DST) was fused with Shannon entropy (SE) for AND in Wu et al. [20]. They used some high-level features like author affiliation, citation venue, co-authorship information and web correlation to calculate the similarities among citations. Subsequently, these features were combined using DST and SE. On the basis of this information, plausibility and belief of all authors were computed. Then, a matrix of pairwise correlation of papers was calculated. Each entry in this matrix was linked to a belief and a plausibility function. Finally, they applied the DST-based hierarchical agglomerative clustering for author disambiguation.

An algorithm that not only disambiguates author names but reconstructs the *h*-index of the individual authors was proposed in Schulz et al. [23]. They applied it to a large-scale Web of Science data set. They calculated the pairwise similarity between all papers on the basis of a number of shared co-authors, self-citations, common references and the number of papers citing both publications. They constructed a link between those publications that have similarity score greater than some predefined threshold and made clusters. Then, these connected clusters were merged, resulting in bigger clusters, which were the set of papers by a unique author. Information about the *h*-index is necessary for the working of this system.

INDi, a solution for the disambiguated DLs, was presented in De Carvalho et al. [24], which utilised similarity among bibliographic records and grouped the new records to authors with similar citation records in the DL or to new authors when the similarity evidence was not strong enough. Some particular heuristics were used for checking whether references of new citation records belong to pre-existing authors of the DL or if they belonged to new ones. This step prevents running the disambiguation process on the entire DL.

Liu et al. [16] used three-step clustering framework for name disambiguation. In the first step, they obtained clusters on the basis of common co-authors. Then, titles are used to make bigger clusters from the first step fragmented clusters. Finally, they fused clusters on the basis of venues. In Maguire [15], an ethnicity sensitive method that mainly comprises three parts was presented. In the first part, phonetic-based blocking for similar author signatures was done. Supervised machine learning–based linkage function was used that exploited the ethnicity sensitive information. Finally, hierarchical agglomerative clustering was done on the basis of a distance between two pairs of publications linkage function. A method named self-training associative name disambiguator (SAND) was proposed in Ferreira et al. [10] that consisted of three steps after pre-processing of the data set. In the first step, co-authors heuristic was used to find the clusters of author records that seem to be included in the same author group. Remaining clusters that were not used for training of the model were used for testing purposes. Finding the pure clusters in the first phase is a non-trivial task. Peng et al. [25] proposed an algorithm that used both web co-relation and author correlation based approach to measure similarities between publications. They basically used two assumptions, citations on the same web page and citations with same rarer authors belonged to the same author. They measured both types of correlations using the modified sigmoid function, cosine metric and name popularity metric.

Active name disambiguation for the name ambiguity (ADANA) problem was proposed in Wang et al. [11]. In this method, they modelled pairwise factor graph that can be used to integrate several types of features as well as user feedback into a unified model. They defined three types of feature functions, that is, document pair, correlation and constraint-based features. In document pair feature functions, they found known relationships from publications. In correlation feature function, they found some hidden feature with the help of known functions and in the constraint-based feature functions the user was involved in finding the unknown features. Finally, they exploited active selection of the user corrections in an interactive mode to improve the disambiguation performance after some preliminary clustering results. They exploited some additional information, such as affiliation and references, that is not present in every DL. In Santana et al. [12], a solution for incremental disambiguation was proposed, and in Shen et al. [13], a system that utilised user feedback for disambiguation was proposed. Momeni and Mayr [32] argued that a co-author's network alone is not sufficient for ambiguous author names and improved the performance using a community detection algorithm with varying resolutions for different names along with co-author's network. They also contributed an ambiguous author data set that can be used in future for AND research.

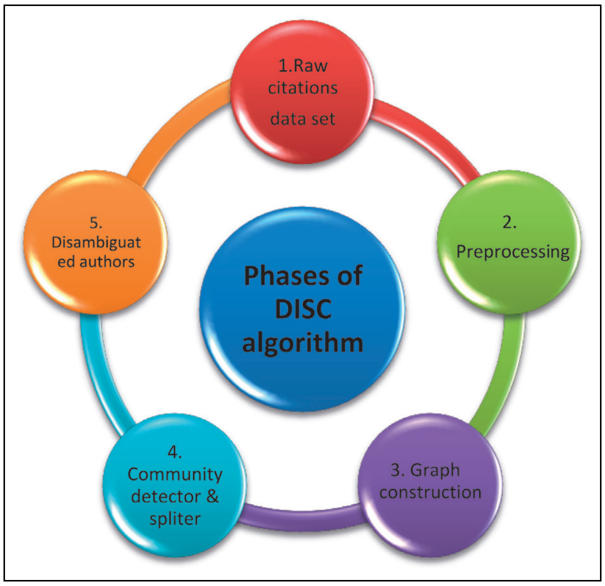


Figure 1. Phases of DISC algorithm.

As pointed out in this section that unsupervised methods required explicit features such as author affiliation, email address, web searches or a number of unknown ambiguous authors 'K' a priori, whereas supervised techniques required a lot of training data that is costly and difficult. In contrast to these techniques, DISC neither requires costly training data nor explicit bibliographic information. It uses only co-authors and titles to disambiguate homonyms.

3. Proposed methodology of DISC

In this section, we present some definitions that are used in structural clustering, particularly in gSkeletonClu. DISC algorithm is presented in subsequent paragraphs, whereas DISC architecture is shown in Figure 1.

3.1. Preliminaries

Let $G = (V, E, w)$ be a weighted undirected graph, where V is a set of vertices, E is set of edges and w is the weights assigned to the edges on the basis of the co-author relationship (e.g. Assigned 2 to the edge if co-authorship occurred two times in the citation's data set) in this graph G. The structure of a vertex can be described by its neighbourhood vertices. If two vertices share more neighbours then these are more similar and vice versa. When a vertex shares similar structure with one of its neighbouring vertices in a cluster then, their structural similarity will be high. A threshold ϵ is applied to the computed structural similarity when assigning cluster memberships. If a vertex share structural similarity with enough neighbours, it becomes a nucleus or seed for that cluster and is called a core vertex. Core vertices are a special class of vertices that have a minimum of μ neighbours with a structural similarity that exceeds the specified threshold ϵ. From core vertices, the clusters are grown. In this way, the parameters ϵ and μ determine the clustering of networks. For a given ϵ, the minimal size of a cluster is determined by μ.

3.1.1. Cluster. A cluster of a network G w.r.t. the given parameters ϵ and μ as all structure-connected clusters in G. Let $\varepsilon \in \Re$ and $\mu \in \aleph$. A clustering P of network $G = <V, E>$ w.r.t. ϵ and μ consists of all structure-connected clusters w.r.t. ϵ and μ in G, formally

$$Cluster_{\varepsilon,\mu}(P) \Leftrightarrow P = \left\{C \in V | Cluster_{\varepsilon,\mu}(C)\right\} \quad (1)$$

A vertex is either a member of a structure-connected cluster or it is an isolated vertex, that is, it does not belong to any of the structure-connected clusters. If a vertex is not a member of any structure-connected clusters, it is either a hub or an outlier, depending on its neighbourhood.

3.1.2. Hub. Let $\varepsilon \in \Re$ and $\mu \in \aleph$. For a given clustering P, that is, $Cluster_{\varepsilon,\mu}(P)$, if an isolated vertex $v \in V$ has neighbours belonging to two or more different clusters w.r.t. ϵ and μ, it is a hub (it bridges different clusters) w.r.t. ϵ and μ formally

$$HUB_{\varepsilon,\mu}(V) \Leftrightarrow \forall(C) \in P : v \notin C \; \exists(p,q) \in \Gamma(v) : \exists(X,Y) \in P : X \neq Y : \wedge p \in X \wedge q \in Y \qquad (2)$$

3.1.3. Outlier. For a given clustering P, that is, $Cluster_{\varepsilon,\mu}(P)$, an isolated vertex $v \in V$ is an outlier if and only if all its neighbours either belong to only one cluster or do not belong to any cluster

$$OUTLIER_{\varepsilon,\mu}(V) \Leftrightarrow \forall(C) \in P : v \notin C \neg \exists(p,q) \in \Gamma(v) : \exists(X,Y) \in P : X \neq Y : \wedge p \in X \wedge q \in Y \qquad (3)$$

We implemented gSkeletonClu algorithm [29] in Python to detect hubs, outliers and clusters (communities) in the co-author's graph. This step is given in 'DISC algorithm' in Figure 5 at line 4. Interested readers are requested to read Huang et al. [29] article for further details.

3.2. Proposed DISC algorithm

The raw citation data set is pre-processed and split into authors, titles and venues terms. Author names are further divided into first name and last name. Longer names that have a middle name are also divided into the same two parts. The initial and middle name becomes the first name in this strategy.

3.2.1. Author name blocking. The blocking stage is important as it affects the computations in the later stages of the name disambiguation algorithms. On et al. [33] defined four name blocking strategies – spelling-based heuristics, token-based, *n*-gram and sampling-based. However, we used spelling-based heuristic strategy that also has different heuristics for blocking such as the initial of the first name and the full last name (iFfL), the initial of the first name and the initial of the last name (iFiL) or full last name (fL) or some combination of these. However, in previous AND studies iFfL blocking strategy proved to be effective [8,10,27,32] and creates relatively small blocks as compared with other spelling-based heuristics such as iFiL or fL. For example, the three names 'Anubhav Gupta', 'Alok Gupta' and 'Aarti Gupta' all grouped into the same candidate block of 'A Gupta'. The blocking stage returns b number of blocks if given a number of authors, it is shown in Figure 2. The computation complexity is $O(a^2)$, for a authors if we do not use blocking, whereas blocking

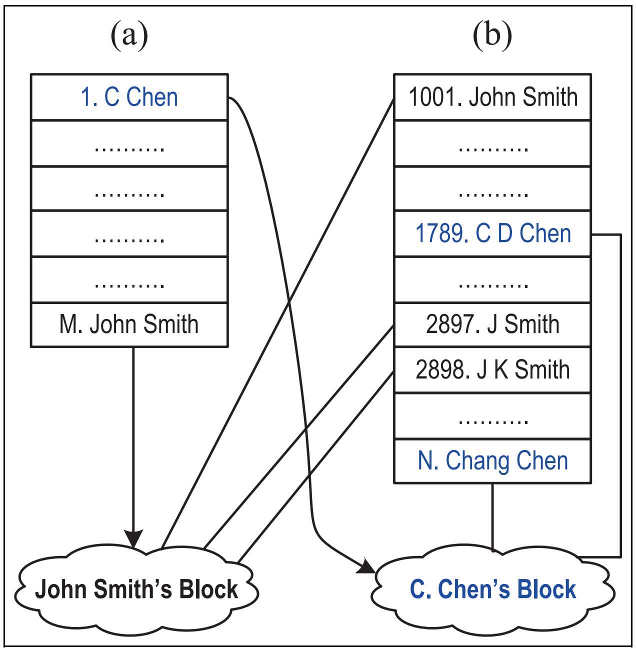


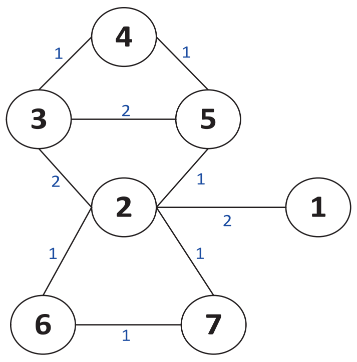
Figure 2. An example of ambiguous author blocking.

Citation No.	Bibliographic Citation Data
1	Wei Wang, S Huang :Disambiguation author names using spectral clustring: KDD, 2015:p 29-38
2	Wei Wang, C Chen :Graph pattern minning-A review: OIR, 2016:p 124-140
3	Wei Wang, Ajay Gupta, S. Huang :Non-linear control for hot air blower system: JUCS, 2012:p 78-87
4	Ajay Gupta, S. Huang, M Rehan:Community detection based graph author name disambiguation framework: KIR, 2014:p 329-339
5	Wei Wang, M Miller, Phillips S. Yu:POSHI:HeArt failure prediction tool: INMIC, 2016:p 295-308
6	C Chen, Wei Wang:Graph utilization in author name ambiguity problem:JUCS, 2014:p 319-329

(a)

Node Id	Node Name	Node Publications
1	C. Chen	P2,P6
2	Wei Wang	P1-P3, P5-P6
3	S. Huang	P1, P3-P4
4	M. Rehan	P4
5	Ajay Gupta	P3-P4
6	M. Miller	P5
7	Phillips S. Yu	P5

(b)



(c)

Figure 3. Co-author's graph of Wei Wang sample citations: (a) an example of small citation's data set, (b) information of nodes in the graph and (c) graph from the excerpt of bibliographic citations.

considerably reduces computational complexity to $O(B|S|)$. Where B is the number of blocks and S is the average size of blocks.

3.2.2. Graph construction. After blocking, we represent all authors and their co-author relations as a graph $G = (V, E, w)$. For instance, each vertex $v \in V$ represents an author $a_i \in A$, each edge $(v_1, v_2) \in E$ represents the co-author's relation and w is the number of co-authorship relations between two authors. In DISC, a vertex represents an author which has a distinct identity, a name (not necessarily distinct) and citations set where that author is present. Two co-authors are modelled using bidirectional edges between two authors (vertices). More specifically, in a citation, set of n authors is converted into a graph $G = (V, E, w)$. Authors are extracted from the citations data set and each distinct author a_i is mapped to a vertex $v \in V$. Then, for each citation, if there are a number of authors then there are $[a(a-1)/2]$ number of edges created in the graph G. For example, if a citation has three authors, then the number of edges would be (3*2)/2 = 3. Similarly, if a citation has four authors, then the number of edges would be (4*3)/2 = 6 and so on. Using this strategy for all citations, we build the whole graph of all the citations in the data set. A small working example of citation data set, vertices, edges, weights and the constructed graph is shown in Figure 3. As shown in the figure, there are seven authors (nodes), nine edges that are created from six citations and respective weights are shown on the edges that represent number of co-authorship relations between two authors. 'Wei Wang' is a node that is common to five citations and 'M. Rehan' is present only in citation 4. As there are two co-authorships between C. Chen and Wei Wang, so a weight of '2' is assigned to the edge between them in Figure 3.

3.2.3. Graph structural clustering. The aim of applying the gSkeletonClu community detection algorithm here is to only detect hubs, outliers and clusters of nodes in the co-author's graph [29]. The DISC algorithm then detects the potential

```
Data: Citations
Result: Feature Vectors
begin $FV \leftarrow 0$
while Citations do
    $Text \leftarrow get.Title(Citation)$
    $Tokens \leftarrow Text.Remove(Stopwords)$
    $Tokens \leftarrow Text.Stem(Tokens)$
    for $Token \in Tokens$ do
        if $Token \notin FV$ then
            $FV.Add(Token)$
            $Token \leftarrow Token + 1$
        else
            $Token \leftarrow Token + 1$
        end
    end
end
```

Figure 4. An implicit text feature vector constructor algorithm.

homonyms and then resolves these homonyms by applying graph operations that satisfy some predefined conditions. The use of gSkeletonClu is done here for four reasons: (1) it detects outliers that are not possible with existing graph-based methods; (2) it makes algorithm scalable as its computational complexity is $O(e)$, where e is the number of edges in co-authors graph; (3) in contrast to other popular community detection algorithms, it is useful to identify overlapping communities in the co-author's graph; and (4) no need to tune threshold parameters, it automatically computes these parameters.

In DISC, it is assumed that different homonyms have different communities in an academic social circle and different homonyms seldom work in the same institution or community [3,4]. So, they belong to different author communities. A community is generated from each citation in the co-author's graph and thus each community denotes the co-authorship for each citation that is the smallest social circle in the academic domain. For example, in 'citation 4', there are three authors – S. Huang, M. Rehan and Ajay Gupta, due to this citation three nodes 3, 4 and 5 and their relationships are constructed as shown in Figure 3(c), and when we add more citations to this graph, it becomes wider and bigger. Finally, when we construct the graph of all sample citations in Figure 3(a), it looks as shown in Figure 3. With the help of co-authors graph, it is possible to infer a social circle of an author from one's co-authors by finding shared communities.

3.2.4. Feature vector construction. After removing stop words and stemming with the help of the Porter [34] stemmer, we construct feature vectors of titles; pseudocode of this is given in Algorithm 1 (Figure 4). DISC assumes that an author linked with multiple social circles contains mixed information for several authors if the similarity that is calculated using Jaccard similarity between feature vectors of two clusters is less than 0.2. This threshold is chosen empirically by varying its value in small increments and finding the optimal value of it. DISC splits that node and its information into a number of nodes that are present in non-overlapping communities.

3.2.5. DISC algorithm. DISC considers each non-overlapping community emanating from the same node as a different social circle of an author if its feature vector similarity is less than 0.2. Complete pseudocode of DISC algorithm is given in Algorithm 2 (Figure 5), and its details are described in subsequent paragraphs.

In the co-author's graph, an edge represents a co-author's relationship between the two nodes (authors). The DISC algorithm is based on gSkeletonClu: Structural Clustering Algorithm for Networks to detect different non-overlapping communities in the co-author's graph [29]. In this algorithm, when we use gSkeletonClu on a co-author's graph it outputs communities (clusters of nodes), hub nodes and outliers. The community consists of a set of nodes from this co-author's graph. The node that is involved in many social circles in the co-author's graph is called a hub node. The hub node is the potential homonym that needs disambiguation. Outliers are nodes that only contribute one or a limited number of publications with hub node and have no other co-authors. For a formal definition of the hub, outlier and cluster, please refer to the 'Preliminaries'.

```
Data: Co-authors Graph (CG)
Result: Disambiguated Graph (DG)
1 begin
2 Authors ← PreprocessData (CitationsData)
3 CG ← BuildGraph (Authors)
4 [ClustersList, Hubs, Outliers] ← applygSkeletonClu (CG)
5 for hub ∈ Hubs do
6 HubCitations ← get.Citations (Hub)
7 $FV_{Hub}$ ← get.FV (Hub)
8 k, l ← 0
9 for cluster ∈ ClustersList do
10        clusterCitations ← 0
11        $FV_C$ ← get.FV (Cluster)
12        FV Similarity ← sim ($FV_C$, $FV_{Hub}$)
13        if FV Similarity < 0.2 then
14                for vertex ∈ cluster do
15                        clusterCitations ← clusterCitations ∪ getCitations (vertex)
16                end
17                $NS_k$Citations ← HubCitations ∩ clusterCitations
18                $NS_k$Name ← getName (Hub)
19                $NS_k$Id ← getLength (CG)
20                CG.InsertNewNode ($NS_k$Id, $NS_k$Name, $NS_k$Citations)
21                UpdateGraph (CG, cluster)
22                HubCitations ← HubCitations \ clusterCitations
23                k ← k + 1
24        else
25                cluster ← cluster + 1
26        end
27 end
28 for outlier ∈ outliers do
29        outlierCitations ← 0
30        $FV_O$ ← get.FV (outlier)
31        FV Similarity ← sim ($FV_O$, $FV_{Hub}$)
32        if FV Similarity < 0.2 then
33        $O_l$Citations ← HubCitations ∩ outlierCitations
34        $O_l$Name ← getName (Hub)
35        $O_l$Id ← getLength (CG)
36        CG.InsertNewNode ($O_l$Id, $O_l$Name, $O_l$Citations)
37        UpdateGraph (CG, outlier)
38        HubCitations ← HubCitations \ outlierCitations
39        l ← l + 1
40        else
41        outlier ← outlier + 1
42        end
43 end
44 end
45 end
```

Figure 5. Proposed DISC algorithm.

In a small co-author's graph example, detected communities, hub and outlier can be seen in Figure 6. There are two clusters of nodes (communities), one hub and one outlier. The first and second cluster consists of nodes $< 3, 4, 5 >$ and $< 6, 7 >$, respectively. The hub is node $< 2 >$ and the outlier is node $< 1 >$.

A hub node that has multiple non-overlapping communities contains mixed information for several homonyms. DISC splits the information about each author on the node containing the homonyms along the non-overlapping communities

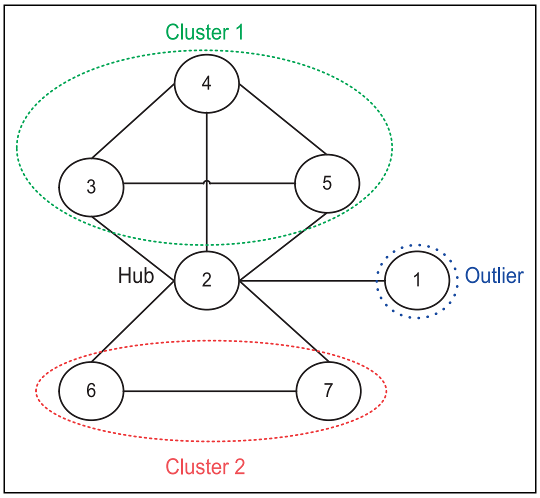


Figure 6. An example of detected communities, hub and outlier in Wei Wang's sample graph after clustering.

if its feature vector similarity is less than 0.2. This part starts from the line 7 of 'DISC algorithm', where the hub node publications list is retrieved. Similarly, publications of all cluster nodes (community) are retrieved and saved in a list (lines 13–15). Now, for each homonym, the intersection of hub publications and community publications are found. A new node is created in the co-authorship graph for each community detected that has the same name as that of the hub node and has an identity one more than in the current nodes of the co-authorship graph. This newly created node has the publication list that is found at line 16. Likewise, for all communities detected in the co-author's graph, new nodes are created.

Co-authors graph and hub node publications are updated after every new node is inserted into the graph. In graph update, some edges to the hub node to existing communities are removed and some new edges to newly created nodes are created (line 18–20). The same procedure is repeated for all outliers, as is done in the case of all communities (lines 27–41). This whole process is pictorially shown in Figure 7, where hub node $<2>$ is split into two new nodes $<$2A, 2B$>$, as there are two clusters of nodes (communities) that have feature vector similarity less than 0.2. In this way, the namesake's problem is solved using community detection algorithm.

3.2.6. DISC complexity analysis. DISC first step is ambiguous author blocking step that has a time complexity of $O(B|S|)$, where B is the number of blocks and S is the average size of blocks. Graph $G(V, E)$ can be constructed in $O(|V| + |E|)$.

According to Huang et al. [29], graph structural clustering can be done in $O(|E|)$. Text feature vectors and other graph comparisons are done only in blocks found in the first step. These steps take negligible time as compared with other steps, so we ignore time complexities of these steps. In the above three steps, the blocking step dominates the whole computational complexity. Hence, the DISC has a complexity of $O(B|S|)$.

4. Performance evaluation of DISC

In this section, the performance of DISC is compared with graph-based and non-graph-based methods using Arnetminer data set in the form of clustering evaluation metrics. All experiments were performed on the Intel® 64-bit Core™ i5-5200U 2 $\times$ 2.2 GHz processors with 8 GB of RAM.

4.1. Data set: Arnetminer

To evaluate the DISC performance, we used two data sets – Arnetminer-S and Arnetminer-L. Tables 1 and 2 illustrate the overall statistics of both the data sets.

These data sets are extracted from Arnetminer which was originally created by Wang et al. [11]. They manually checked, verified and labelled all these citation records. This collection has been used in many previous studies with slight variations [4,8,11] and is available online. In Arnetminer-S, we grouped all those ambiguous authors which have citation records less than or equal to 50 and in the Arnetminer-L more than 50 citation records. The small data set is composed of 950 citation records associated with 344 distinct authors belonging to 39 ambiguous groups, approximately

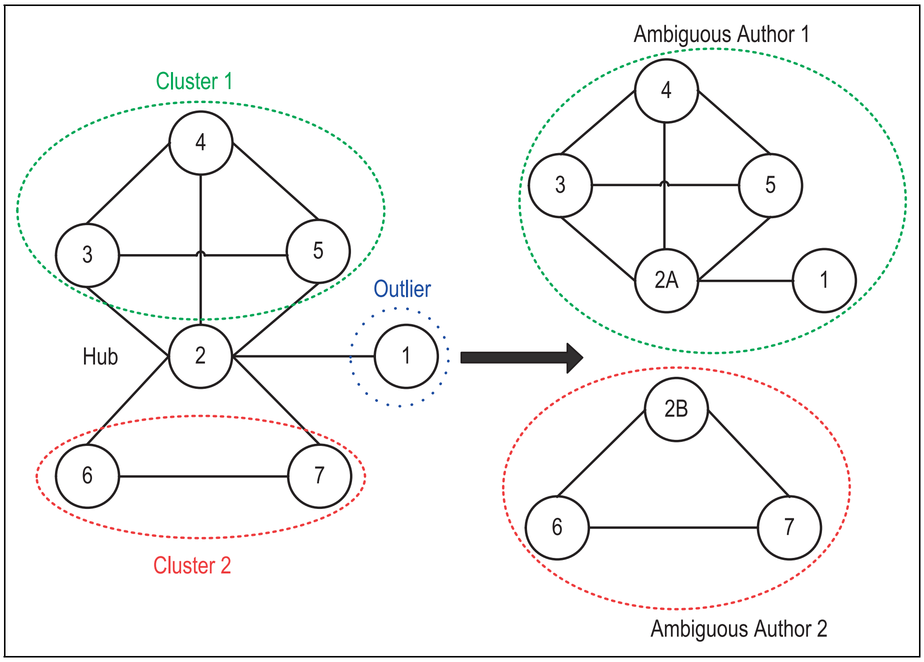


Figure 7. Example of homonyms resolution.

Table 1. The Arnetminer-S data set

Name	Auth.	Record	Name	Auth.	Record	Name	Auth.	Record
Ajay Gupta	8	31	B. Wilkinson	1	17	Bob Johnson	3	4
Bin Zhu	15	45	Cheng Chang	5	23	Michael Lang	4	16
Charles Smith	4	7	Paul Wang	6	15	Fei Su	4	37
Michael Siege	6	50	David Cooper	7	18	Robert Allen	8	21
Gang Luo	8	42	S. Huang	15	16	Hui Fang	8	42
J. Guo	9	12	Hui Yu	20	30	Xiaoyan Li	6	31
Yan Tang	11	31	Lei Fang	7	17	Ping Zhou	17	33
Xiaoming Wang	14	33	Yue Zhao	9	36	Lei Jin	6	16
Paul Brown	8	20	Peter Phillips	3	13	Thomas Taylor	3	4
David Nelson	10	15	F. Wang	14	15	Z. Wang	35	43
Hong Xie	6	10	J. Yin	7	18	Jianping Wang	5	35
John Hale	3	36	Kuo Zhang	4	16	M. Rahman	7	15
Michael Smith	16	27	Richard Taylor	11	27	Young Park	8	17

Auth. denotes number of distinct authors and record represents citations records associated with that author.

2.76 citation records per author, as shown in Table 1. Whereas Arnetminer-L is composed of 3189 citation records associated with 657 distinct authors belonging to 25 ambiguous groups, approximately 4.85 citation records per author, as given in Table 2. For instance, in Arnetminer-S number of 'Hiu Yu' variants is 20, while in Arnetminer-L number of 'Lei Wang' variants is 111 and so forth. Both data sets are publicly available and can be accessed https://github.com/kashak79/DISC_AND_Data_Set.

4.2. Evaluation metrics

We used nine well-known clustering metrics – namely, average cluster purity (ACP), average author purity (AAP), K-metric, pairwise precision (PP), pairwise recall (PR), pairwise-F1 (PF1), cluster precision (CP), cluster recall (CR) and cluster-F1 (CF1) – to measure the effectiveness of DISC [4,27,28].

Table 2. The Arnetminer-L data set

Name	Auth.	Record	Name	Auth.	Record	Name	Auth.	Record
Bin Li	58	178	Jie Yu	6	66	Rakesh Kumar	9	95
David Brown	24	60	Jose M. Garcia	2	83	Sanjay Jain	5	196
Eric Martin	5	84	Kai Zhang	23	65	Wei Xu	47	149
Feng Pan	14	70	Lei Chen	38	190	Wen Jao	10	482
Gang Chen	43	169	Lei Wang	111	299	X. Zhang	37	58
Hao Wang	45	158	Li Shen	8	65	Yang Yu	19	69
Ji Zhang	16	62	Lu Liu	16	57	Yong Chen	24	83
Jie Tang	6	66	Manuel Silva	4	73	Yu Zhang	71	221

Auth. denotes number of distinct authors and record represents citations records associated with that author.

K-metric is defined as the geometric mean of the ACP and the AAP. The purity of the system-generated clusters, as compared with the ground truth clusters is measured by the ACP, so it measures the amount of data misclassified into the wrong clusters by checking whether the generated clusters include only the publication records belonging to the reference clusters. The level of fragmentation of the system-generated clusters into multiple clusters is evaluated by AAP. ACP, AAP, and K-metric are expressed in equation (4)

$$\mathrm{ACP} = \frac{1}{N}\sum_{i=1}^{I}\sum_{j=1}^{J}\frac{{n_{ij}}^2}{n_i}, \quad \mathrm{AAP} = \frac{1}{N}\sum_{i=1}^{I}\sum_{j=1}^{J}\frac{{n_{ij}}^2}{n_j}, \quad \mathrm{K} = \sqrt{\mathrm{ACP} * \mathrm{AAP}} \tag{4}$$

Here, N denotes the size of the citations in the collection, J is the number of gold standard reference clusters manually generated and I is the number of clusters automatically generated by the DISC. Also, n_j is the number of elements in cluster j and n_{ij} is the number of elements belonging to both clusters i and j. PF1 is the harmonic mean of PP and PR. The fraction of citation records corresponding to the same author in the system-generated clusters are known as PP and fraction of citation records associated with the same author in the gold standard reference clusters are called as PR. The PP, PR and PF1 measures are expressed in equation (5), where $C(n; i)$ denotes the number of combinations of i elements from n elements. Other parameters including i, j, n_i and n_{ij} are defined as before in equation (4)

$$\mathrm{PP} = \frac{\sum_{r=1}^{R}\sum_{s=1}^{S} C(n_{rs}, 2)}{\sum_{r=1}^{R} C(n_r, 2)}, \quad \mathrm{PR} = \frac{\sum_{r=1}^{R}\sum_{s=1}^{S} C(n_{rs}, 2)}{\sum_{s=1}^{S} C(n_s, 2)}, \quad \mathrm{PF1} = \frac{2(\mathrm{PP} * \mathrm{PR})}{\mathrm{PP} + \mathrm{PR}} \tag{5}$$

CF1 is the harmonic mean of CP and CR, where CP is the fraction of the generated clusters that are equal to the reference clusters and CR is the fraction of correctly retrieved clusters from the reference clusters. The CP, CR and CF1 measures are given in equation (6)

$$\mathrm{CP} = \frac{I \cap J}{I}, \quad \mathrm{CR} = \frac{I \cap J}{J}, \quad \mathrm{CF1} = \frac{2(\mathrm{CP} * \mathrm{CR})}{\mathrm{CP} + \mathrm{CR}} \tag{6}$$

4.3. Comparison methods

Three graph-based and two non-graph-based AND methods which were related to our method are used to compare with DISC [4,8,11,35].

On graph-based name disambiguation called GHOST (GrapHical framewOrk for name diSambiguaTion) [8], we modelled the relationships among publications using undirected graphs. They solved name disambiguation by iteratively finding valid paths, computing similarities, clustering with the help of affinity propagation algorithm and, in the last, using user feedback as a complementary tool to enhance the performance. They used a number of ambiguous authors ('K') a priory and does not handle the sole author's cases. In ADANA [11], they made a pairwise factor graph that is used to integrate several types of features as well as user feedbacks into a unified model. They defined three types of feature functions. In document pair feature functions, they found, known relationships from publications. In correlation feature function, they extracted some hidden features with the help of known functions and in the constraint-based feature functions, the user is involved in finding the unknown features. In the last step, they exploited active selection of the user corrections in an interactive mode to improve the disambiguation performance after some preliminary clustering results. Shin et al. [4] proposed a framework which they called Graph Framework for Author Disambiguation (GFAD) in which Johnson cycle algorithm [36], and syntactic similarity was used to find non-overlapping cycles in a graph. To compare DISC with GFAD, we implemented the disambiguation algorithm as given in Shin et al. [4], with their proposed parameters. In Wang et al. [11], shortest valid path length, whereas in Shin et al. [4], largest non-overlapping cycles are used for clustering. Heuristic-based hierarchical clustering (HHC) is a heuristic-based unsupervised method that uses citation features for AND. HHC showed better results compared with other unsupervised and supervised methods [35]. We compared the performance of DISC with HHC using co-authors and titles heuristics which we call HHC-CT and HHC with all the features of publications (co-authors, titles and venue) which we denote with HHC-All. Both HHC-All and HHC-CT needs similarity thresholds for the paper title and publication venue. Therefore, we evaluated the performance of HHC by varying threshold values and then selected values on which the performances are maximised. We set the threshold values for paper title and publication venue as 0.2 and 0.3, respectively. They used hierarchical clustering for both HHC-CT and HHC-All. For further details on HHC, please refer to Cota et al. [35].

4.4. DISC performance evaluation

DISC performance is evaluated using nine well-known clustering metrics on both Arnetminer-S and Arnetminer-L data sets. Table 3 shows average ACP, AAP, K-metric, PP, PR, PF1, CP, CR and CF1 of DISC. It achieves on average an ACP of 99%, K and PF1 are 92% and 84%, respectively, and CF1 of 74% on Arnetminer-S, as depicted in Figure 8. Table 4 lists the complete results of each ambiguous author group in the Arnetminer-L using nine clustering metrics.

ACP and PP of DISC in the majority of cases are approximately equal to 100% but in some cases such as ‘X. Zhang’ and ‘Z. Wang’ is low. DISC achieves on average an ACP of 99%, K and PF1 of 86% and 80%, respectively, and CF1 of 57% on Arnetminer-L, as shown in Figure 9. In the case of ‘Sanjay Jain’, CF1 is as low as 7%. In this case, there are five ground truth clusters, but DISC found that there are 24 clusters. We carefully examine the citations of ‘Sanjay Jain’ and found that he has multiple affiliations and worked on multiple research domains in his career with an entirely different set of co-authors.

4.4.1. DISC comparison with graph-based methods. In Figure 10, DISC performance is compared with three graph-based methods – GHOST, ADANA and GFAD using Arnetminer-S. It shows overall better results than these methods and achieves 12%, 8% and 5% higher in K-metric; 32.1%, 27.6% and 4.2% higher in CF1; and 3.7%, 6.3% and 1.2% higher in PF1 as compared with GHOST, ADANA and GFAD, respectively. DISC performance is better as compared with methods due to the reason that GHOST, ADANA and GFAD are unable to detect outlier authors that only share one co-author with hub node, whereas DISC is able to identify these authors.

In Figure 11, DISC performance is compared with the same methods using Arnetminer-L data set and again it shows overall better results than these methods. It achieves 3.5%, 8% and 3.5% higher in K-metric; 10.5%, 8.8% and 5.3% higher in CF1; and 5%, 5% and 0% higher in PF1 as compared with GHOST, ADANA and GFAD, respectively.

4.4.2. DISC comparison with non-graph-based methods. In Figure 12, DISC performance is compared with two non-graph-based methods, HHC-CT and HHC-All using Arnetminer-S. DISC shows overall better results and achieves 7% and 16% higher in K-metric, 2.4 and same% higher in CF1 and 10.4% and 5.7% higher in PF1 as compared with HHC-CT and HHC-All, respectively. DISC performance is better as compared with HHC-CT and HHC-All due to two reasons. First, it detects outlier authors that only share one co-author with hub node, and second, it does not do wrong merges on the basis of venue similarity.

Complete details of ground truth clusters and DISC-generated clusters of Arnetminer-S are shown in Table 5, whereas totals of the ground truth clusters and DISC-generated clusters are summarised in Table 6. About 77.5% clusters are within the range (ground truth clusters $\pm$ 3) in Arnetminer-S, that’s why DISC shows better CF1 than these methods. Similar results are achieved in the case of Arnetminer-L, where 73.2% clusters are within the range (ground truth clusters $\pm$ 3).

We also observe that DISC performs better on Arnetminer-S than Arnetminer-L. We suspect that it may be due to the higher ratio of authors with only one citation record of 23.2% on the Arnetminer-S than that of 5.1% on the Arnetminer-l, which exerts a negative influence on the performance of DISC. DISC performance is better than all compared methods except PF1 that is same to GFAD on Arnetminer-L, despite the fact that GHOST and ADANA used the predefined number of clusters ‘K’.

Table 3. The performance evaluation of DISC for each ambiguous group on the Arnetminer-S

Name	ACP	AAP	K	PP	PR	PF1	CP	CR	CF1
Ajay Gupta	1	0.64	0.80	1	0.59	0.74	0.31	0.62	0.42
Barry Wilkinson	1	1	1	1	1	1	1	1	1
Bin Zhu	1	0.85	0.92	1	0.72	0.83	0.68	0.87	0.76
Bob Johnson	1	1	1	1	1	1	1	1	1
Charles Smith	1	1	1	1	1	1	1	1	1
Cheng Chang	1	0.79	0.89	1	0.70	0.82	0.38	0.60	0.46
David Cooper	1	0.92	0.96	1	0.90	0.95	0.75	0.86	0.80
David Nelson	1	0.93	0.97	1	0.88	0.93	0.82	0.90	0.86
F. Wang	1	1	1	1	1	1	1	1	1
Fei Su	0.95	0.71	0.83	0.98	0.77	0.86	0.25	0.50	0.33
Gang Luo	1	0.95	0.98	1	0.93	0.96	0.78	0.88	0.82
Hong Xie	1	0.87	0.93	1	0.50	0.67	0.75	0.86	0.80
Hui Fang	1	0.70	0.83	1	0.55	0.71	0.42	0.62	0.50
Hui Yu	1	0.97	0.98	1	0.95	0.97	0.91	0.95	0.93
J. Yin	0.95	0.52	0.70	1	0.20	0.33	0.91	0.95	0.93
J. Guo	1	0.92	0.96	1	0.67	0.80	0.82	0.90	0.86
Jianping Wang	1	0.63	0.80	1	0.56	0.72	0.50	0.80	0.62
John Hale	1	0.52	0.72	1	0.49	0.66	0.11	0.33	0.17
Kuo Zhang	1	0.91	0.95	1	0.91	0.95	0.60	0.75	0.67
Lei Fang	1	0.90	0.95	1	0.77	0.87	0.75	0.86	0.80
Lei Jin	1	1	1	1	1	1	1	1	1
M. Rahman	1	0.79	0.89	1	0.43	0.60	0.67	0.86	0.75
Michael Lang	1	0.73	0.86	1	0.56	0.72	0.50	0.75	0.60
Michael Siege	1	0.83	0.91	1	0.83	0.91	0.36	0.67	0.47
Michael Smith	1	0.76	0.87	1	0.35	0.52	0.62	0.81	0.70
Paul Brown	1	1	1	1	1	1	1	1	1
Paul Wang	1	0.91	0.95	1	0.90	0.95	0.71	0.83	0.77
Peter Phillips	1	0.77	0.88	1	0.74	0.85	0.20	0.33	0.25
Ping Zhou	1	0.97	0.98	1	0.98	0.99	0.89	0.94	0.91
Richard Taylor	1	0.73	0.86	1	0.49	0.65	0.64	0.82	0.72
Robert Allen	1	1	1	1	1	1	1	1	1
S. Huang	1	1	1	1	1	1	1	1	1
Thomas Taylor	1	1	1	1	1	1	1	1	1
Xiaoming Wang	0.97	0.86	0.92	1	0.94	0.97	0.62	0.71	0.67
Xiaoyan Li	1	0.94	0.97	1	0.92	0.96	0.71	0.83	0.77
Yan Tang	1	0.85	0.92	1	0.80	0.89	0.64	0.82	0.72
Young Park	1	0.85	0.92	1	0.77	0.87	0.70	0.88	0.78
Yue Zhou	1	0.66	0.81	1	0.52	0.68	0.46	0.67	0.55
Z. Wang	0.93	0.91	0.92	0.57	0.40	0.50	0.78	0.80	0.79

DISC: disambiguating homonyms using graph structural clustering; ACP: average cluster purity; AAP: average author purity; K: K-metric; PP: pairwise precision; PR: pairwise recall; PF1: pairwise-F1; CP: cluster precision; CR: cluster recall; CF1: cluster-F1.

DISC is unable to identify when two different authors with the same name have different co-authors with the same name which in AND literature are called very ambiguous authors [1]. However, analysis of both data sets shows that only 0.31% authors are very ambiguous authors, particularly some Asians names (Chinese and Korean).

To show the scalability of DISC, comparison of running time of all methods on Arnetminer-S and Arnetminer-L is shown in Table 7. All experiments were performed on a personal computer with Intel® Core™ i5-5200U CPU @ 2.20 GHz and 8 Gigabyte memory. All methods were implemented using Python. GFAD was the slowest among all the compared on both data sets. Its most time-consuming stage is the cycle finding in the co-author's graph. GHOST, HHC-CT and DISC are the fastest on Arnetminer-S and Arnetminer-L, respectively. The reason why HHC-CT outperforms in running time on Arnetminer-S is that there are simple comparisons between co-authors and title similarities of the papers. GHOST is performing equally well to DISC as it mostly compares two or three path length nodes in this graph, however, on larger path lengths such as Arnetminer-L GHOST takes more time than DISC. The running time of DISC was statistically tied with GHOST and HHC-CT on Arnetminer-S. DISC is about 11–14 times faster than GFAD and overall comparable than all other methods.

Table 4. The performance evaluation of DISC for each ambiguous group on the Arnetminer-L

Name	ACP	AAP	K	PP	PR	PF1	CP	CR	CF1
Bin Li	0.96	0.84	0.90	0.94	0.77	0.85	0.72	0.84	0.78
David Brown	1	0.83	0.91	1	0.82	0.90	0.69	0.92	0.79
Eric Martin	1	0.80	0.89	1	0.85	0.92	0.36	0.80	0.50
Feng Pan	1	0.66	0.81	1	0.54	0.70	0.45	0.71	0.56
Gang Chen	1	0.69	0.83	1	0.47	0.64	0.60	0.86	0.70
Hao Wang	1	0.69	0.83	1	0.55	0.71	0.46	0.71	0.56
Ji Zhang	1	0.87	0.93	1	0.92	0.96	0.62	0.81	0.70
Jie Tang	1	0.68	0.83	1	0.64	0.78	0.42	0.83	0.56
Jie Yu	1	0.88	0.94	1	0.87	0.93	0.56	0.83	0.67
Jose M. Garcia	1	0.61	0.78	1	0.58	0.74	0.14	0.50	0.22
Kai Zhang	1	0.84	0.91	1	0.82	0.90	0.55	0.74	0.63
Lei Chen	1	0.67	0.82	1	0.51	0.68	0.59	0.84	0.70
Lei Wang	0.94	0.86	0.90	0.85	0.78	0.82	0.73	0.84	0.78
Li Shen	1	0.74	0.86	1	0.73	0.84	0.40	0.75	0.52
Lu Liu	1	0.79	0.89	1	0.69	0.82	0.56	0.88	0.68
Manuel Silva	1	0.72	0.85	1	0.66	0.80	0.33	0.75	0.46
Rakesh Kumar	1	0.65	0.81	1	0.73	0.84	0.19	0.56	0.29
Sanjay Jain	1	0.73	0.86	1	0.80	0.89	0.04	0.20	0.07
Wei Xu	0.98	0.74	0.85	0.99	0.65	0.79	0.53	0.72	0.61
Wen Jao	0.99	0.89	0.94	0.99	0.90	0.94	0.24	0.70	0.36
X. Zhang	0.86	0.91	0.89	0.57	0.77	0.65	0.75	0.73	0.74
Yang Yu	1	0.76	0.87	1	0.63	0.77	0.52	0.79	0.62
Yong Chen	1	0.65	0.80	1	0.44	0.61	0.45	0.71	0.55
Yu Zhang	0.98	0.69	0.82	0.96	0.52	0.67	0.50	0.72	0.59

DISC: disambiguating homonyms using graph structural clustering; ACP: average cluster purity; AAP: average author purity; K: K-metric; PP: pairwise precision; PR: pairwise recall; PF1: pairwise-F1; CP: cluster precision; CR: cluster recall; CF1: cluster-F1.

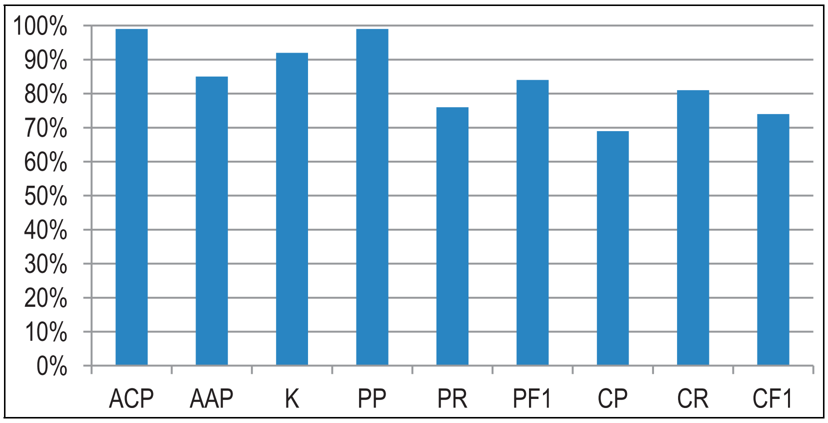


Figure 8. DISC average ACP, AAP, K-metric, PP, PR, PF1, CP, CR and CF1 on Arnetminer-S.

5. Conclusion and future work

Graph structural clustering–based community detection algorithms are not used, to the best of our knowledge, in the AND algorithms. In this article, we have proposed DISC which solves homonyms using only co-authors and titles that are surely present in almost all bibliographic data sets. DISC is a graph-based algorithm that neither requires a costly training data nor a priori information about how many ambiguous authors are there nor any web searches nor any expert knowledge. DISC pre-processes the bibliographic metadata and constructs the co-author's graph in which an author is represented by a vertex and co-author's relationship with an edge. Potential homonyms are those vertices that are hub nodes and can be split using the information of feature vectors of the titles of the publications of the clusters. We have shown how blocking step reduces the disambiguation time, how a graph structural clustering algorithm can be used to

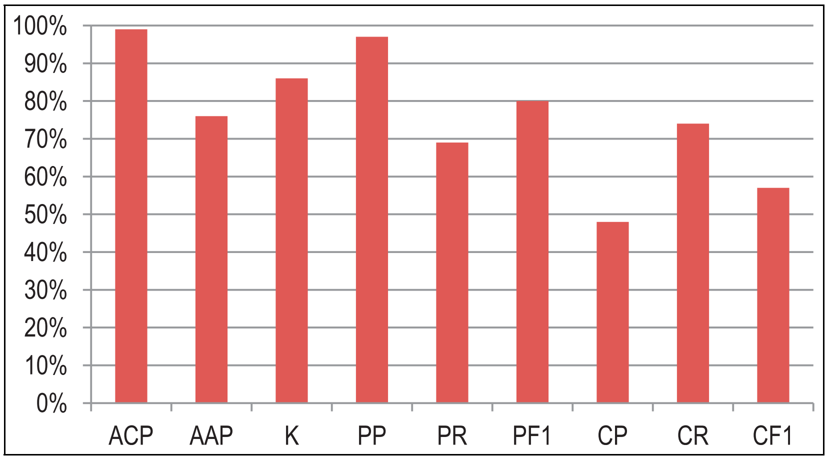


Figure 9. DISC average ACP, AAP, K-metric, PP, PR, PF1, CP, CR and CF1 on Arnetminer-L.

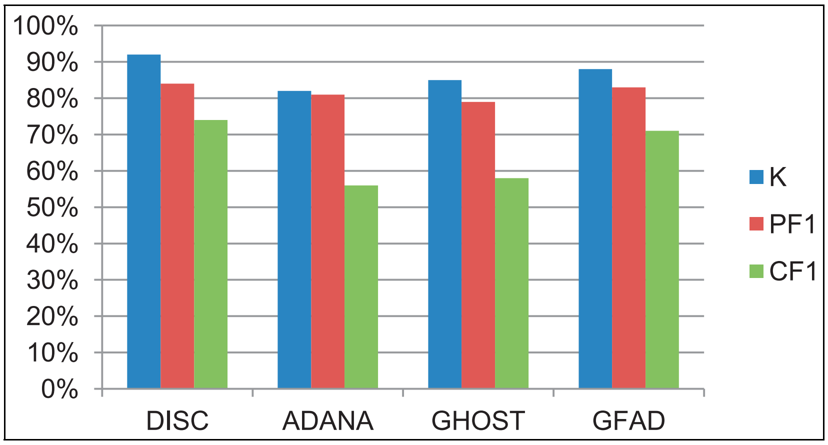


Figure 10. Average K-metric, PF1 and CF1 of the DISC, ADANA, GHOST and GFAD on Arnetminer-S.

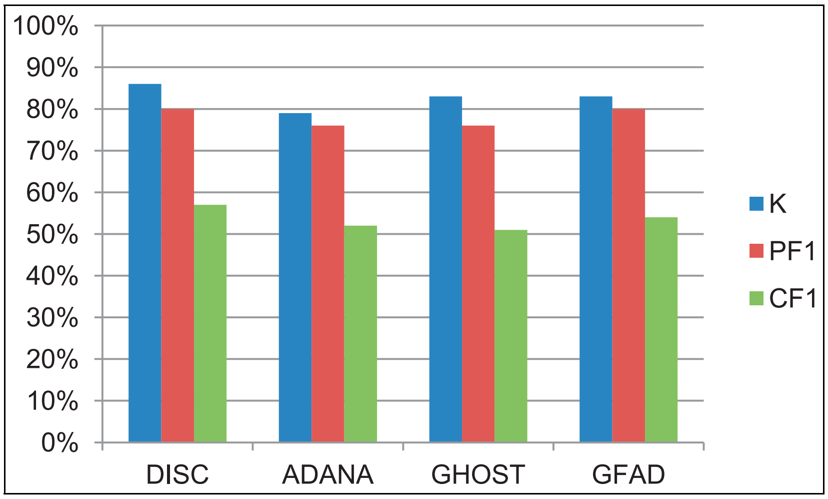


Figure 11. Average K-metric, PF1 and CF1 of the DISC, ADANA, GHOST and GFAD on Arnetminer-L.

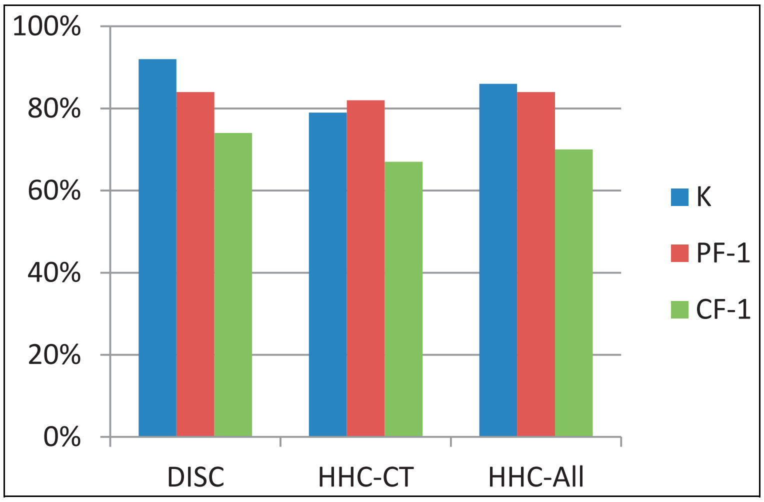


Figure 12. Average K-metric, PF1 and CF1 of the DISC, HHC-CT and HHC-All on Arnetminer-S.

Table 5. Comparisons of ground truth clusters and DISC-generated clusters on Arnetminer-S

Name	GTC	DGC	Name	GTC	DGC	Name	GTC	DGC
Ajay Gupta	8	12	B. Wilkinson	1	3	Bob Johnson	3	3
Bin Zhu	15	18	Cheng Chang	5	8	Michael Lang	4	3
Charles Smith	4	8	Paul Wang	6	6	Fei Su	4	2
Michael Siege	6	8	David Cooper	7	9	Robert Allen	8	6
Gang Luo	8	8	S. Huang	15	17	Hui Fang	8	10
J. Guo	9	11	Hui Yu	20	25	Xiaoyan Li	6	5
Yan Tang	11	8	Lei Fang	7	5	Ping Zhou	17	14
Xiaoming Wang	14	20	Yue Zhao	9	11	Lei Jin	6	11
Paul Brown	8	12	Peter Phillips	3	4	Thomas Taylor	3	8
David Nelson	10	14	F. Wang	14	12	Z. Wang	35	43
Hong Xie	6	8	J. Yin	7	9	Jianping Wang	5	8
John Hale	3	6	Kuo Zhang	4	7	M. Rahman	7	6
Michael Smith	16	19	Richard Taylor	11	14	Young Park	8	9

DISC: disambiguating homonyms using graph structural clustering; GTC: ground truth clusters; DGC: DISC-generated clusters.

Table 6. Summary of ground truth clusters and DISC-generated clusters

Data Set	Ground truth clusters	DISC-generated clusters
Arnetminer-S	341	410
Arnetminer-L	641	802

DISC: disambiguating homonyms using graph structural clustering.

detect and disambiguate homonyms and how title-based feature vector similarity can be used to reduce false splitting of the homonyms. DISC performance was tested on two Arnetminer data sets and it showed better results than three graph-based and two non-graph-based methods. DISC delivered a simple but effective and scalable solution to an author name ambiguity problem as it utilised the powers of graphs. However, it is unable to detect very ambiguous author cases where two different authors with the same name work with two different co-authors with the same name. In the future, we plan to analyse email address and affiliation of authors (if available) to overcome this limitation.

Table 7. Summary of all methods running time on both data sets (s)

Data set	DISC	ADANA	GHOST	GFAD	HHC-CT	HHC-All
Arnetminer-S	22	86	21	257	18	25
Arnetminer-L	38	124	57	548	40	64

DISC: disambiguating homonyms using graph structural clustering; ADANA: active name disambiguation for the name ambiguity.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship and/or publication of this article.

Notes

1. http://dblp.uni-trier.de
2. http://www.medline.com
3. http://citeseerx.ist.psu.edu
4. http://arxiv.org
5. http://academic.research.microsoft.com
6. http://scholar.google.com.pk
7. http://www.lbd.dcc.ufmg.br/bdbcomp

References

[1] Ferreira AA, Gonçalves MA and Laender AH. A brief survey of automatic methods for author name disambiguation. *SIGMOD Rec* 2012; 41(2): 15–26.

[2] Han H, Xu W, Zha H et al. A hierarchical naive Bayes mixture model for name disambiguation in author citations. In: *Proceedings of the 2005 ACM symposium on applied computing*, Santa Fe, NM, 13–17 March 2005, pp. 1065–1069. New York: ACM.

[3] Tang J, Fong AC, Wang B et al. A unified probabilistic framework for name disambiguation in digital library. *IEEE T Knowl Data En* 2012; 24(6): 975–987.

[4] Shin D, Kim T, Choi J et al. Author name disambiguation using a graph model with node splitting and merging based on bibliographic information. *Scientometrics* 2014; 100(1): 15–50.

[5] On BW, Elmacioglu E, Lee D et al. Improving grouped-entity resolution using quasi-cliques. In: *Proceedings of the 6th international conference on data mining (ICDM'06)*, Hong Kong, China, 18–22 December 2006, pp. 1008–1015. New York: IEEE.

[6] Torvik VI, Weeber M, Swanson DR et al. A probabilistic similarity metric for Medline records: a model for author name disambiguation. *J Assoc Inf Sci Tech* 2005; 56(2): 140–158.

[7] Bhattacharya I and Getoor L. Collective entity resolution in relational data. *ACM T Knowl Discov D* 2007; 1(1): 5.

[8] Fan X, Wang J, Pu X et al. On graph-based name disambiguation. *J Data Inf Qual* 2011; 2(2): 10.

[9] Han D, Liu S, Hu Y et al. ELM-based name disambiguation in bibliography. *World Wide Web* 2015; 18(2): 253–263.

[10] Ferreira AA, Veloso A, Gonçalves MA et al. Self-training author name disambiguation for information scarce scenarios. *J Assoc Inf Sci Tech* 2014; 65(6): 1257–1278.

[11] Wang X, Tang J, Cheng H et al. ADANA: active name disambiguation. In: *Proceedings of the 2011 IEEE 11th international conference on data mining*, Vancouver, BC, Canada, 11–14 December 2011, pp. 794–803. New York: IEEE.

[12] Santana AF, Gonçalves MA, Laender AH et al. Incremental author name disambiguation by exploiting domain-specific heuristics. *J Assoc Inf Sci Tech* 2017; 68: 931–945.

[13] Shen Q, Wu T, Yang H et al. NameClarifier: a visual analytics system for author name disambiguation. *IEEE T Vis Comput Gr* 2017; 23(1): 141–150.

[14] Liu Y, Li W, Huang Z et al. A fast method based on multiple clustering for name disambiguation in bibliographic citations. *J Assoc Inf Sci Tech* 2015; 66(3): 634–644.

[15] Maguire EJ. Ethnicity sensitive author disambiguation using semi-supervised learning. In: *Proceedings of the 7th international conference on knowledge engineering and semantic web (KESW 2016)*, Prague, 21–23 September 2016, vol. 649, pp. 272–287. Cham: Springer International Publishing.

[16] Liu W, Islamaj Doğan R, Kim S et al. Author name disambiguation for PubMed. *J Assoc Inf Sci Tech* 2014; 65(4): 765–781.

[17] Tran HN, Huynh T and Do T. Author name disambiguation by using deep neural network. In: *Proceedings of the Asian conference on intelligent information and database systems*, Bangkok, Thailand, 7–9 April 2014, pp. 123–132. Cham: Springer International Publishing.
[18] Levin M, Krawczyk S, Bethard S et al. Citation-based bootstrapping for large-scale author disambiguation. *J Assoc Inf Sci Tech* 2012; 63: 1030–1047.
[19] Kang IS, Na SH, Lee S et al. On co-authorship for author disambiguation. *Inform Process Manag* 2009; 45(1): 84–97.
[20] Wu H, Li B, Pei Y et al. Unsupervised author disambiguation using Dempster–Shafer theory. *Scientometrics* 2014; 101(3): 1955–1972.
[21] Onodera N, Iwasawa M, Midorikawa N et al. A method for eliminating articles by homonymous authors from the large number of articles retrieved by author search. *J Assoc Inf Sci Tech* 2011; 62(4): 677–690.
[22] Zhu J, Yang Y, Xie Q et al. Robust hybrid name disambiguation framework for large databases. *Scientometrics* 2014; 98(3): 2255–2274.
[23] Schulz C, Mazloumian A, Petersen AM et al. Exploiting citation networks for large-scale author name disambiguation. *EPJ Data Sci* 2014; 3(1): 11.
[24] De Carvalho AP, Ferreira AA, Laender AH et al. Incremental unsupervised name disambiguation in cleaned digital libraries. *J Inform Data Manag* 2011; 2(3): 289–304.
[25] Peng HT, Lu CY, Hsu W et al. Disambiguating authors in citations on the web and authorship correlations. *Expert Syst Appl* 2012; 39(12): 10521–10532.
[26] Tang L and Walsh JP. Bibliometric fingerprints: name disambiguation based on approximate structure equivalence of cognitive maps. *Scientometrics* 2010; 84(3): 763–784.
[27] Ferreira AA, Machado TM and Gonçalves MA. Improving author name disambiguation with user relevance feedback. *J Inform Data Manag* 2012; 3(3): 332–347.
[28] Pereira DA, Ribeiro-Neto B, Ziviani N et al. Using web information for author name disambiguation. In: *Proceedings of the 9th ACM/IEEE-CS joint conference on digital libraries*, Austin, TX, 15–19 June 2009, pp. 49–58. New York: ACM.
[29] Huang J, Sun H, Song Q et al. Revealing density-based clustering structure from the core-connected tree of a network. *IEEE T Knowl Data En* 2013; 25(8): 1876–1889.
[30] Elliott S. Survey of author name disambiguation: 2004 to 2010. *Libr Philos Pract* 2010, http://digitalcommons.unl.edu/libphilprac/473/ (2010, accessed October 2016).
[31] Smalheiser NR and Torvik VI. Author name disambiguation. *Annu Rev Inform Sci* 2009; 43(1): 1–43.
[32] Momeni F and Mayr P. Evaluating co-authorship networks in author name disambiguation for common names. In: *Proceedings of the international conference on theory and practice of digital libraries*, Hannover, 5–9 September 2016, pp. 386–391. Cham: Springer.
[33] On B-W, Lee D, Kang J et al. Comparative study of name disambiguation problem using a scalable blocking-based framework. In: *Proceedings of the 5th ACM/IEEE-CS joint conference on digital libraries, 2005 (JCDL'05)*, Denver, CO, 7–11 June 2005, pp. 344–353. New York: IEEE.
[34] Porter MF. An algorithm for suffix stripping. *Program* 1980; 14(3): 130–137.
[35] Cota RG, Ferreira AA, Nascimento C et al. An unsupervised heuristic-based hierarchical method for name disambiguation in bibliographic citations. *J Assoc Inf Sci Tech* 2010; 61(9): 1853–1870.
[36] Johnson DB. Finding all the elementary circuits of a directed graph. *SIAM J Comput* 1975; 4(1): 77–84.