# MyCites: An Intelligent Information System for Maintaining Citations

George Papadakis[1,2] and Georgios Paliouras[1]

[1] Institute of Informatics and Telecommunications,
National Centre for Scientific Research "Demokritos", Athens, Greece
[2] Department of Electrical and Computer Engineering,
National Technical University of Athens, Greece
{gpapad,paliourg}@iit.demokritos.gr

**Abstract.** The evaluation of their research work and its effect has always been one of scholars' greatest concerns. The use of citations for that purpose, as proposed by Eugene Garfield, is nowadays widely accepted as the most reliable method. However, gathering a scholar's citations constitutes a particularly laborious task, even in the current Internet era, as one needs to correctly combine information from miscellaneous sources. There exists therefore a need for automating this process. Numerous academic search engines try to cover this need, but none of them addresses successfully all related problems. In this paper we present an approach that facilitates to a great extent citation analysis by taking advantage of new algorithms to deal with these problems.

**Keywords:** information extraction, citation matching, name disambiguation, mixed citation problem, split citation problem, string distance metrics.

## 1 Introduction

In the last decade there has been a strong interest and considerable effort in developing on-line services that provide access to academic databases. These attempts have culminated in the development of search engines that specialize in scholarly literature. The most notable of them are *Scopus*[1]*, Web of Science (WoS)*[2] and *Google Scholar(GS)*[3], which are based on huge academic databases gathered from numerous sources. Some of these engines (e.g. Scopus and WoS) use structured sources, such as databases of publishers, in order to warrantee the precision of the provided information. Others (e.g. GS) emphasize on retrieving as much information as is available, automatically from unstructured data, such as Web sites. To the best of our knowledge there is currently no engine that addresses adequately both aspects of the problem.

One of the most valuable features of academic search engines is *citation analysis*, that is looking for papers that refer to a specific publication. In this way they automate

[1] http://www.scopus.com/scopus/home.url
[2] http://scientific.thomson.com/products/wos/
[3] http://scholar.google.com/

J. Darzentas et al. (Eds.): SETN 2008, LNAI 5138, pp. 371 – 376, 2008.

a laborious yet essential task of scholars, that of gathering citations in order to evaluate the influence of their research work. High recall engines, particularly GS, seem to gain in popularity, but there is still enough room for improvement, as they usually contain a relatively large portion of duplicate data and noise. The following problems are particularly important and hard to solve:

**Definition 1.** *Citation Matching* (CM) is the problem where, given two lists of publications, X and Y, the goal is to find for each x (∈ X) a set of $y_1$, $y_2$, ..., $y_n$ (∈ Y) such that both x and $y_i$ ($1 \leq i \leq n$) in fact pertain to the same publication. Among the main causes of CM are the lack of a fixed format for citations, the various names that are attributed to a single author and errors in the parsing software.

**Definition 2.** *Mixed Citation* (MC) [1] is the problem where, given a collection of publications, C, by an author, $a_i$, the goal is to accurately identify publications by another author $a_j$ in C, when $a_i$ and $a_j$ have identical name spellings.

**Definition 3.** *Split Citation* (SC) [2] is the problem where given two lists of author names and associated publications, X and Y, the goal is to find for each author name x (∈ X) a set of author names, $y_1$, $y_2$, ..., $y_n$ (∈ Y ) such that both x and $y_i$ ($1 \leq i \leq n$) are name variants of the same author.

We should point out that the MC and SC problems are so closely related to each other that are rarely succinctly distinguished. They are regarded as a single problem called *name disambiguation* or *name equivalence identification.* Along with the CM problem they belong to the broader *Identity Uncertainty Problem* or *Record Linkage*.

In this paper we propose new methods that deal with these problems and are embedded in a simple information system intended to automate the maintenance of citations for scholars and research groups through a user-friendly interface.

## 2 Related Work

Two teams have primarily worked on the SC and MC problems. The first one concentrated on the MC problem and tested supervised classification ([3]) and unsupervised clustering ([4],[5]) methods, concluding that the latter generally perform better, while not requiring processed datasets for training. Their clustering approaches presume though a predefined number of clusters, thus limiting their applicability. The second research group addresses both the MC ([1]) and SC ([1],[2]) problems, primarily concentrating on the scalability of their algorithms. The proposed matching methods are generally based on common features of publications: co-authors' names, paper and conference/journal title, with the first proving to be the most robust one.

As far as the CM problem is concerned, the term was initially coined in [6] and [7] by the creators of Citeseer, where they also presented four different methods based on simple string matching methods. In [8] another method is proposed, based on relational probability models (RPMs), while in [9] an innovative algorithm is presented based on conditional random fields (CRFs). It is worth noting that all of these methods were applied to the same dataset and are thus directly comparable, with the last one achieving the best performance.

Finally, [10] summarizes, categorizes and compares the most robust and efficient methods for string matching, that are at the heart of all the above-mentioned methods.

## 3 Application Use Cases

In this section we will analyze briefly the main functionality of the application that we have developed, based on the proposed approach. We do this by going through the steps that comprise a thorough search for a scholar.

1. *Fetch all publications that contain the given scholar in their author list.* This is done by issuing the appropriate query to the GS search engine, gathering and feeding the results to the wrapper we have developed for processing the returned HTML pages. The wrapper identifies the html tags that define the information of a single article and then the tags that encompass each attribute of that specific paper (title, authors, URL etc).
2. *Apply the Citation Matching algorithm for processing the gathered publications.* There is considerable noise in the form of duplicate articles in GS results and, therefore, a pre-processing stage that refines the data gathered by the initial query is indispensable. Otherwise the performance of the name disambiguation algorithm would be substantially degraded by the duplicates. Our method for solving this problem is presented in section 4.
3. *Present the user with the results of the CM algorithm for verification.* In this stage, the user is given the chance to amend potential mistakes or omissions of the CM algorithms. Specifically, the user is provided with all the necessary information (title, authors, URLs etc) so as to be able to judge whether two articles that were alleged to match are in fact different articles and thus have to be dissociated or whether two separate publications are duplicates and must be matched.
4. *Apply the Name Disambiguation algorithm.* This is the most critical step of each search as it entails the identification of all separate scholars that contribute to all papers maintained by our application, those already stored and those acquired during the current search. We address this problem by the algorithm presented in section 5.
5. *Present the user with the results of Name Disambiguation for verification.* The purpose of this step is to amend once again the results of the automatic processing so as to ensure that the data stored in the database is as accurate as possible. This is a critical step since potential errors that are not detected are perpetuated in subsequent runs of the name disambiguation algorithm, thus degrading its performance. By the end of this process, every piece of information concerning the separate scholars is stored.
6. *Search for citations.* Having completed the previous steps, the user can now search for the citations of a specific paper or start a new search for a different scholar. Every new search goes through the above steps before giving the user the chance to commence a new one.

## 4 Citation Matching

The most common forms of duplicate citations that appear in the results of GS and thus need to be addressed by our algorithm are the following:

1. *spelling mistakes*
2. *author's names concatenated with paper title (usually preceding it)*
3. *conference/journal title concatenated with paper title (usually following it)*
4. *title swapped with another information field*

In this context, our algorithm acts as follows:

1. It initially orders the retrieved papers by the number of citations, based on the assumption that among the multiple appearances of a single paper, the one with the most citations will probably contain the correct information, as it is highly unlikely that a paper is cited more frequently in a wrong way than in the right one. With this initial sorting we ensure that the more correct the information of a paper, the fewer times it is compared to another one.
2. It then checks the contents of the database before processing each paper, so as to avoid repeating the same process. We assume that the problem has been resolved for the stored papers, since the user has verified the stored data.
3. For each paper that is not stored in the database, its title is compared with that of its preceding ones, so as to find the most similar paper. The comparisons are done using the *SoftTFIDF string distance metric in combination with the Jaro metric* [10], which has proven the most suitable metric for the three first problems mentioned above, that is for matching strings that are sets of words (tokens).
4. If the best matching does not exceed a user-defined threshold, the algorithm checks whether the title has moved to another field of the paper's description (fourth case). This is done by forming a new string for each paper in the list, comprising the paper title, co-author names and conference/journal title. These strings are compared using again *SoftTFIDF with Jaro* and if their similarity exceeds another threshold, they are considered identical.

## 5 Name Disambiguation

In this section we introduce a new clustering method for solving simultaneously both the Mixed and the Split Citation problems. Our approach generally exploits the same article features as other methods proposed in the literature: co-authors' names, paper title, URLs of the papers and the name of the scholar. However, what differentiates our algorithm from the others is the range of its applicability. Our goal is to develop a method that given *any* dataset of citations identifies *all* separate scholars it contains *without any prior knowledge about them.* To achieve this goal the algorithm is based on the following basic principles:

1. Every author of a single paper corresponds to a separate scholar.
2. Every scholar can match with only one author of a single paper.

3. It is time and memory consuming to repeat the matching process for the authors of the already stored publications. After all, the data stored in the database has already been verified by the user and is thus assumed to be accurate.

Abiding by these principles our algorithm creates a graph of related authors, aiming to partition the graph in its connected components. In this context, a new node is initially added to the graph for each author of the **stored** papers. The nodes that correspond to the same scholar are then directly connected, in order to ensure that they are included in the same connected component after partitioning the graph.

Then, for each of the **new** papers, the most similar author name in the graph is found. The similarity is calculated by the *Jaro* string matching method. If this similarity exceeds the respective threshold, the value of the following formula is calculated for the data of the current two publications:

**Tot_sim = β * co-author similarity + γ * title similarity + δ * URL similarity**

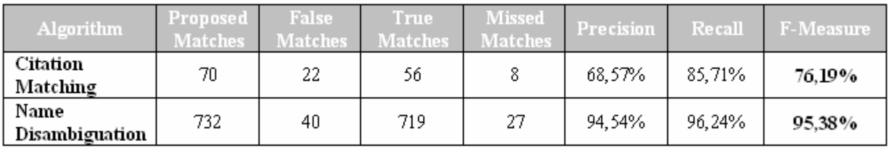
These similarities are calculated using again the *combined SoftTFIDF with Jaro metric*. If the value of Tot_sim exceeds another threshold, the two authors are considered identical and their nodes are connected with a new edge.

With the completion of this process, the graph is partitioned in its connected components, each of which contains all information about a unique scholar: all variants of a scholar's name, along with all the papers the scholar has authored.

## 6 Experiments and Results

In order to measure the performance of our algorithms, we need to test them over a fairly large dataset, covering various influential factors, such as the scientific field of the papers, the nationality of the authors, etc. However due to time limitations, we performed a limited set of initial experiments, using the work of one of the authors of this paper as a seed, moving to the work of those who cite his papers, and so on. Despite its limited nature, the dataset included authors of various nationalities, some of which (e.g. Korean names) amplify the mixed citation problem. The algorithms were evaluated with the use of standard information retrieval criteria: *precision*[4], *recall*[5] and *f-measure* (their harmonic mean), and the results are presented in Table 1.

**Table 1.** Evaluation of our algorithms



| Algorithm | Proposed Matches | False Matches | True Matches | Missed Matches | Precision | Recall | F-Measure |
|---|---|---|---|---|---|---|---|
| Citation Matching | 70 | 22 | 56 | 8 | 68,57% | 85,71% | **76,19%** |
| Name Disambiguation | 732 | 40 | 719 | 27 | 94,54% | 96,24% | **95,38%** |

These initial results seem promising, but we need to acknowledge that the sample for the citation matching algorithm is too limited to draw safe conclusions. Furthermore, the performance of the name disambiguation algorithm is degraded by the large

[4] *Precision = ( proposed matches – false matches ) / proposed matches*

[5] *Recall = ( true matches – missed matches ) / true matches*

portion of citing authors that appear only once in the dataset. Therefore, a large-scale experiment may lead to different results.

## 7 Conclusions

We presented a new approach for addressing important problems in using academic search engines for citation analysis, namely the citation matching and mixed and split citation problems. The proposed methods were successfully embedded in an information system that aims to facilitate the maintenance of citations for scholars. The methods were evaluated, giving initial encouraging results.

There is undoubtedly great potential in evolving our system. First of all, we plan to add support for additional major academic search engines, such as Scopus and WoS. Combining the contents of these on-line sources will significantly increase the comprehensiveness of our system, Furthermore, based on the aforementioned performance of our methods there is evidently enough room for improvement. We primarily need to refine our citation matching algorithm and generalize its applicability to other academic search engines.

## References

1. Lee, D., On, B.W., Kang, J., Park, S.: Effective and Scalable Solutions for Mixed and Split Citation Problems in Digital Libraries. In: Proceedings of the 2nd International Workshop on Information Quality in Information Systems, pp. 69–76 (2005)
2. On, B.W., Lee, D., Kang, J., Mitra, P.: 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, pp. 344–353 (2005)
3. Han, H., Giles, L., Zha, H., Li, C., Tsioutsiouliklis, K.: Two Supervised Learning Approaches for Name Disambiguation in Author Citations. In: Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 296–305 (2004)
4. Han, H., Xu, W., Zha, H., Giles, C.: A Hierarchical Naive Bayes Mixture Model for Name Disambiguation in Author Citations. In: Proceedings of the 2005 ACM Symposium on Applied Computing, pp. 1065–1069 (2005)
5. Han, H., Zha, H., Giles, C.: Name Disambiguation in Author Citations using a K-way Spectral Clustering Method. In: Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 334–343 (2005)
6. Giles, C., Bollacker, K., Lawrence, S.: Citeseer: An Automatic Citation Indexing system. In: Proceedings of the Third ACM Conference on Digital Libraries, pp. 89–98 (1998)
7. Lawrence, S., Giles, C., Bollacker, K.: Digital Libraries and Autonomous Citation Indexing. IEEE Computer Society 32(6), 67–71 (1999)
8. Pasula, H., Marthi, B., Milch, B., Russel, S., Shpitser, I.: Identity uncertainty and citation matching. In: Advances in Neural Information Processing Systems (NIPS), vol. 15 (2003)
9. Wellner, B., McCallum, A., Peng, F., Hay, M.: An Integrated, Conditional Model of Information Extraction and Coreference with Application to Citation Matching. In: Proceedings of the 20th conference on Uncertainty in artificial intelligence, pp. 593–601 (2004)
10. Cohen, W., Ravikumar, P., Fienberg, S.: A Comparison of String Distance Metrics for Name-Matching Tasks. In: Proceedings of International Joint Conferences on Artificial Intelligence (IJCAI 2003) Workshop on Information Integration on the Web (2003)