Tracing Related Scientific Papers by a Given Seed Paper using Parscit

Resmana Lim[1*], Indra Ruslan[2], Hansin Susatya[1], Adi Wibowo[2], Andreas Handojo[2], and Raymond Sutjiadi[3]

[1] Electrical Engineering Department, Petra Christian University, Surabaya, Indonesia
[2] Informatics Engineering Department, Petra Christian University, Surabaya, Indonesia
[3] Computer Engineering Department, Institut Informatika Indonesia, Surabaya, Indonesia
* Corresponding; Email: resmana@petra.ac.id

Abstract. The project developed a web site learning support for tracing scientific articles relating to a given input seed paper/article. The system will finds related articles that are listed on the references of the seed article. First, the reference list of an input seed article is extracted by utilizing Parscit citation extraction. Furthermore, the system searches the reference articles using Google Scholar and Mendeley API. Thus articles which are related to the seed article can be found. The system was built using PHP programming, it is utilizing Parscit modules, Google Scholar search and Mendeley.com API. Testing has been done by giving an input seed article. User will obtain the results of several articles related to the seed article.

Keywords: Paper Tracing, citation extraction, Parscit, Google Scholar, Mendeley.com

1. Introduction

To support literature review on a research, further reference search from an article/paper that we read is an important thing. When we read a scientific article, very often we are keen to discover more about the articles contained in the list of references. We can manually perform a search using Google scholar. But this will take substantial efforts when there are many references that we will find. Therefore we need a tool to find or download the references we find automatically given an input seed article.

An automated extraction of paper's bibliography metadata is a challenging task given the variety of different paper's layouts and formatting (citation style) of its reference strings. Several automated extraction approaches have been proposed which are using unsupervised methods and

template matching. It founds that the most promising approach is utilizing supervised sequence classification such as Hidden Markov Models (HMMs) [1] or Conditional Random Fields (CRFs) [2]. ParsCit [3] is a popular reference extraction system that uses heuristics approach to detect and segment references within a scientific paper. ParsCit uses CRF to assign labels to the tokens within each reference string. ParsCit is an open-source CRF-based citation parser that has been successfully used by CiteSeerx [4] and scientific papers harvester system [5].

Our project contributes to the purpose of enriching a scientific article (we call the seed paper) by identifying its bibliography (list of references) and connecting them to the original resources (full text if available) by utilizing google scholar and Mendeley API. The project uses Parscit, to process the bibliography in scientific articles. Parscit generate reference's meta-data such as author name, article title, year, etc. The meta-data is then used to perform queries to Google Scholar and Mendeley. The system seeks to connect the bibliography with its source full text using Google Scholar or Mendeley.

2. System Description

This web application is built by using PHP and HTML programming language and MySQL as database management system. The system uses Parscit as service provider to extract paper information data. All this components are run in a Linux based server. To accommodate function of parsing data to other data sources, PHP must be set to enable the cURL function.

In general, this system can be decomposed into parts of process as shown on Fig. 1. The first part is preprocessing. In this part, paper is prepared to be processed by Parscit by formatting PDF paper into TXT format. The second part is paper extraction process. This process is handled by Parscit to extract the paper meta-data such as: title, author(s), abstract, keywords. Parscit also do extract the list of references, which are used to make Google Scholar/Mendeley queries on the next phase. The third part is related paper searching process, where in this part, server makes query to other data sources (i.e. Google Scholar and Mendeley.com) to search related paper from references data. The last part is composing and presenting the information as the web output to the user.

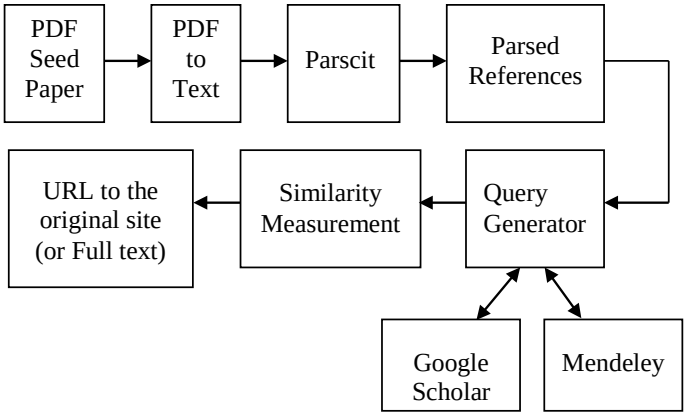


Fig. 1. System Block Diagram

As an input seed paper, user uploads a PDF paper into the system. Because Parscit can only process text based file, then before moving to the next part, unicode text must be extracted from PDF file. To do that, Apache PDFBox or similar application is used to extract unicode text from the input PDF file. The result from this process is a TXT file format.

This TXT file format is processed further by Parscit to grab paper meta-data (i.e. title, author, affiliation, abstract, etc.) and parsed its references/bibliography data. Before the meta-data extraction, Parscit must be trained first with some sample data using similar paper format and language to detect the sections of paper. Make sure that there are enough sample paper data to increase the accuracy of data extraction. The output from the Parscit extraction is an XML file format that already accommodate extracted information which is grouped by different XML tags as shown in Fig. 2.

```
<citation valid="true">
<authors>
<author>Adi Kusrianto</author>
</authors>
<title>Pengantar Desain Komunikasi Visual.</title>
<date>2007</date>
<publisher>Penerbit ANDI.</publisher>
<location>Yogyakarta:</location>
<marker>Kusrianto, 2007</marker>
<rawString>Kusrianto, Adi. (2007). Pengantar Desain Komunikasi Visual.
Yogyakarta: Penerbit ANDI.</rawString>
</citation>
```

Fig. 2. Part of XML Citation Data.

In the next step, each parsed reference data is forwarded to query generator. In this part, query generator searches related paper data from 2 data sources, Google Scholar and Mendeley. The system initiates HTTP query using Document Object Model (DOM) parsing method to Google Scholar to search related paper using title and author data that already classified by Parscit XML result above. This parsing method is used because Google Scholar does not provide Application Programming Interface (API). If similar data are found in Google Scholar, than system saves the result into database to be processed further in the next step.

For Mendeley, system makes the same query to search related paper information, but in easier way because Mendeley provides PHP API to query the searching result. If similar data are found in Mendeley, than system saves the result into database to be processed further in the next step.

After getting the result, in the next part, system runs similarity checking to compare Google Scholar and Mendeley results with parsed references data from Parscit. Similarity checking is done by comparing the author name and title data between each reference data and then calculating the similarity value using formula 1 and 2 respectively as used in [6] [7] .

$$\mathrm{AS} = \sum_{n} \frac{\mathrm{LS}}{\mathrm{LN}} \tag{1}$$

Where:

AS = final similarity score of author name between paper and reference data.

LS = number of matched characters between author name in paper and reference data.

LN = number of total author name characters.

$$Sr = 1 - \frac{||r1 - r2||}{||r1 + r2||} \tag{2}$$

Where:

Sr = efficiency metric as similarity indicator of title between paper and reference data.

r1 = vector of word sequence in paper data.

r2 = vector of word sequence in reference data.

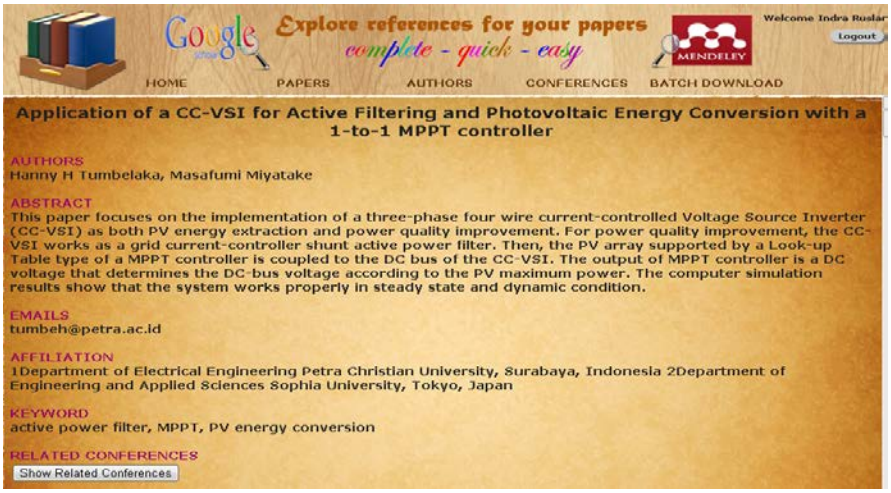
After similarity value has been known, system arranges the descending order of related paper data for each reference. This similarity checking process is to make sure that top listed is the most appropriate related paper.

As the output as shown in Fig. 3, system shows the result in form of web page format. Each reference data of a paper is added with an URL link to download the full text paper from the result of data query above.

Fig. 3. Result of Related Paper Query

3. Result and Discussion

For testing, we have already trained the Parscit to recognize the English paper optimally using the same writing format with paper that used as a sample for this testing. As the input, we use seed English paper in PDF format and gives the output as shown in Fig. 4.



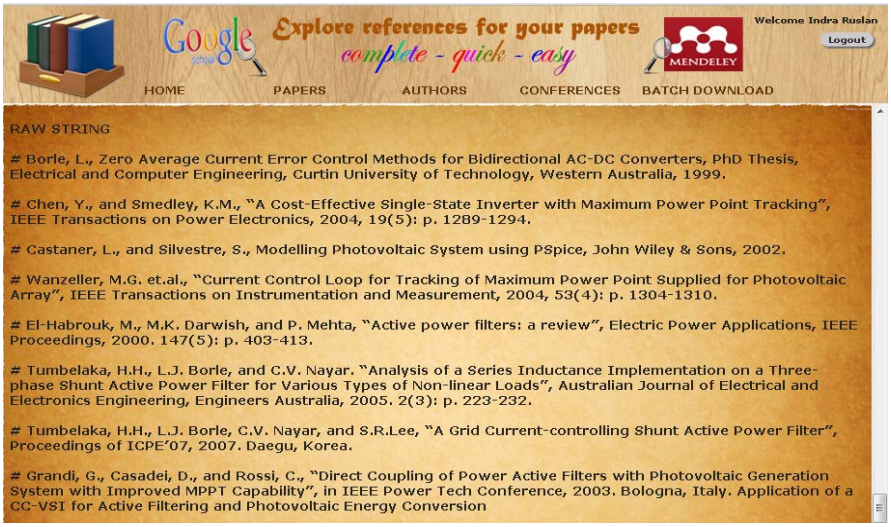


Fig. 4. Result of Parscit Extraction Process

From extraction result above, system can parse the reference data optimally. This reference data than processed further to Mendeley and Google Scholar to search related papers. As the result, system can list the related paper successfully as shown in Fig. 5.

Fig. 5. List of Reference Related Paper

4. Conclusion

We have developed a web site learning support for tracing scientific articles relating to a given input seed article. the system finds related articles that are listed on the references of the seed article so that user could track and download the paper easily.

Acknowledgements

The authors would like to thank the Indonesian Directorate General of Higher Education, which provided the funds for the research project.

References

[1] Hetzner, E., A simple method for citation metadata extraction using hidden markov models, in: *Proceedings of the 8th ACMIEEECS joint conference on Digital libraries JCDL 08*, vol. 18, no. 3, pp. 280–284, (2008)

[2] Peng, F., Mccallum, A., Information extraction from research papers using conditional random fields, Inf. Process. Manag., vol. 42, no. 4, pp. 963–979 (2006)

[3] Councill, I. G., Giles, C. L., Kan, M., ParsCit: An open-source CRF Reference String Parsing Package, in: Proceedings of LREC '08, pp. 661–667 (2008)

[4] Teregowda, P. B., Urgaonkar, B., Giles, C. L., Citeseerx: A cloud perspective, in: Proc. Of Second USENIX Work. Hot Topic in Cloud Computing (2010)

[5] Lim, R., Wibowo, A., Sutjiadi, R., Oktian, Y. E., Pengembangan Paper Citation Extraction Bahasa Indonesia Berbasis Parscit," in: *Seminar Nasional Teknologi Informasi (SNTI 2012)*, Jakarta (2012)

[6] Li, Y., Bandar, Z., McLean, D., O'Shea, J., A Method for Measuring Sentence Similarity and its Application to Conversational Agents, in: proc. of *FLAIRS Conf.* (2004)

[7] Liliana, L., Lim, R., & Kwan, E., Voice conversion application (VOCAL). In: *Uncertainty Reasoning and Knowledge Engineering (URKE), 2011 International Conference on*, Vol. 1, pp. 259-262, IEEE August (2011)