TopSeer: A Novel Scholar Search Engine based on Community Detection in Citation Network

Ming Zhong   Jinghua Cao

College of Information Engineering, Shenzhen University, Shenzhen, China. 518060
mingz@szu.edu.cn

Abstract

There has been considerable interest in development of scholar search engines. A disadvantage of current scholar search engines is that they have not explored the relationship among theses fully. A novel scholar search engine is proposed here to fill this gap, which can detect different topics hidden in a large volume of theses to help researchers find their interested theses easily. A new community detection algorithm in citation networks is proposed here to achieve detecting topics. This algorithm has time complexity O(c*n) in a sparse citation network, where n is the number of nodes, and c is related to the average degree of nodes and the initial number of communities. Initial experiments show that this algorithm produces effective result on a standard test dataset.

Keywords: Scholar Search Engine, Community Detection, Citation Network

1. Introduction

Since one of the first web search engines World Wide Web Worm (WWWW) [1] was born in 1994, both the research and application areas of search engine have gained rapid development. Scholar search is one of the most interesting areas of search engines. Current scholar search engines include Google Scholar [2], CiteSeer [3], and other autonomous citation indexing systems. Usually these scholar search engines provide most of the advantages of traditional citation indexes (e.g. Wos and Scopus [4]), such as literature retrieval by following citation links, and the ranking of papers based on the number of citations. Besides these, they also have some advantages over traditional citation indexes, including the ability to create more up-to-date databases which are not limited to a preselected set of journals. However these functions are far to satisfy researchers' need, especially for a research rookie. Given a large volume of theses, there exist different topics. One topic may be corresponding to a research direction. Detecting these topics can help researchers easily find theses they have interest in.

We propose TopSeer, which is a scholar search engine based on our proposed algorithm for detecting communities in citation networks. TopSeer provides many advantages over current scholar search engines, such as searching topics which are generated automatically by community detection, and showing query results not just by relevance but also by topics.

This paper is organized as follows: in section 2, we introduce background knowledge about community detection in citation networks; in section 3, we introduce TopSeer's architecture; in section 4,

Proceedings of the 11th Joint Conference on Information Sciences (2008)
Published by Atlantis Press

we introduce our community detection algorithm and show the initial experiment result; in section 5, we give conclusion and future work description.

2. Background Knowledge

References contained in academic articles are used to give credit to previous work in the literature and provide a link between the "citing" and "cited" articles [5]. With citations contained in articles, a citation network is made up. Within this network, an article is a node, and the citation between two articles is an edge, and the direction of an edge represents the "citing" and "cited" relation. One important property of citation networks is called community structure. Figure 1 [6] shows an example of a network that has three communities. Generally speaking, a community of a network is a set of vertices within which vertex-vertex connections are dense, but between which connections are less dense.

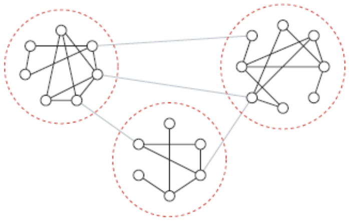


Fig.1: A network with three communities

In a citation network, articles sharing a common topic should have more connections than articles from other topics, so detecting topics is to find communities in the network.

Different kinds of methods are designed to detect communities. Traditional one is hierarchical clustering. Figure 2 depicts hierarchical clustering. First select n nodes that are not connected, and proceed to add edges that are most related to these nodes until all nodes are connected. Finally a clustering tree is formed, and a community is represented by an internal node in the tree.

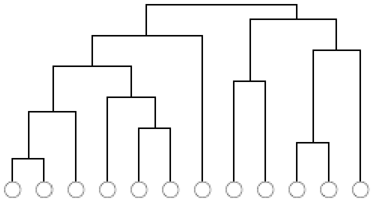


Fig. 2: A clustering tree with each internal node representing a community [7]

[7] have introduced a divisive approach which includes the removal of the edges depending on their betweenness values. It uses the Network Modularity Q to get an optimized division of the network with O(m^3) time complexity, where m is the number of edges. [8] proposed a fast clustering algorithm with O(n^2) time complexity on sparse graph by using a greedy strategy to get a maximal Q by merging pairs of nodes iteratively until it becomes negative, where n is the number of nodes. Recently [9] proposed an algorithm called ComTector which is more efficient for the community detection in large-scale networks based on the nature of overlapping communities in the real world, and its running time is O(CTri2), where C is the number of the detected communities and Tri is the number of the triangles in the given network for the worst case. All these current algorithms are either computational expensive or lack of accuracy.

In this paper, we proposed a new method to detect hidden communities. Our algorithm is fast compared to other algorithms, while producing effective results. The algorithm will be introduced in section 4.

3. The Architecture of TopSeer

This section introduces the architecture of TopSeer. The total image of TopSeer's modules is in Figure 3. It is composed of five layers. The crawler layer contains one or more crawlers to download articles available on the web. The storage layer contains RDBMS storing articles and indexes of metadata of articles. The process layer contains procedures to extract metadata, build up citation network and analyze it to attain topic information. The cache layer contains caches to accelerate the speed of responding to a user query. The query layer serves as a user interface for answering a user's query. Key modules are introduced below.

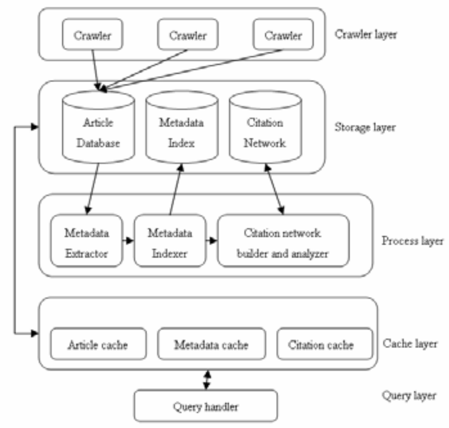


Fig. 3: The architecture of TopSeer

3.1. Crawler

A crawler is to search for public accesses of literatures or articles available on the web. There are several ways to locate a position of an academic article, e.g. searching for pages that contain words such as "journal", "paper", "publisher". Documents with postfix like ".pdf", ".ps" are also indicated that they are academic articles. Another way is to have agreement with some publishers to subscribe their journals. All downloaded articles are stored and ready for Article Information Extractor.

3.2. Article Metadata Extractor

Article Metadata Extractor is used to extract metadata (e.g. title, author names, abstract) of downloaded articles. Many methods have been proposed to extract metadata of a thesis including machine learning methods such as Conditional Random Field and Hidden Markov Model. Among these methods, CRF (conditional random field) is our choice. This method has been demonstrated well in article metadata extraction [10]. Meanwhile, we also use this method to extract metadata within each citation in the reference part of a thesis, such as cited paper's title, authors and published date.

After extracting the metadata of a citation, we have to check the article which the citation refers to has been exist in our database. That is to judge whether two citations refer to the same article. We assume that if two citations have same authors, same title and same published date, they are the same.

All the extracted metadata is passed to the metadata indexer, citation network builder and analyzer, for building up indexes for metadata and the citation network.

.

3.3. Citation Network Builder and Analyzer

The citation network builder is responsible for building and updating the citation network. Because the citation network is usually sparse in reality, we use adjacent matrix to store the citation network. When a new article is added into the citation network, we can quickly figure out the topic that it belongs to by calculating the probability of how much it belongs to a certain topic. Our community detection algorithm is implemented in the network analyzer. The analyzer is also responsible for updating the communities found in the citation network every one month, because the network is growing and communities are changing.

4. Community Detection Algorithm

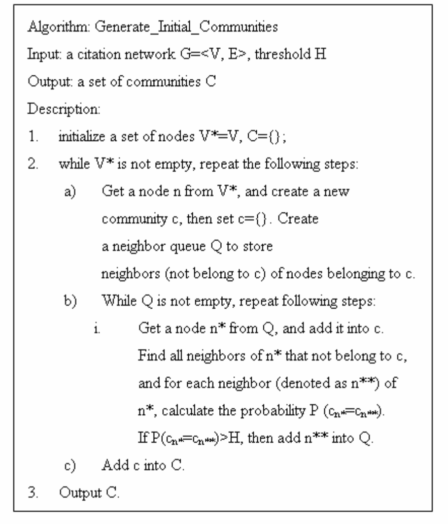
Here we describe our algorithm. It has two stages. The first stage is to generate an initial set of communities, and the second stage is to refine the initial community set through an iterative process.

4.1. Stage One

To generate an initial set of communities, we propose a method to calculate the probability of two nodes belonging to the same community. To our observation, if two nodes belong to the same community, they may have many common neighbors. Let *u* and *v* be two nodes connected in the network, and *Neighbor*(*u*)={a_1, a_2, …, a_p}, *Neighbor*(*v*)={b_1, b_2, …, b_q}, where a_i (i=1, 2, …, p) is a neighbor of *u*, and b_i (i=1, 2, …, q) is a neighbor of *v*. Let *w*={c_1, c_2, …, c_r} be the set of common neighbors among *u* and *v*. Then

$$P(c_u = c_v) = \left(\frac{r+1}{p} + \frac{r+1}{q} \right) / 2$$

is the probability of *u* and *v* that they belong to the same community. c_u represents the community *u* belongs to, and c_v represents the community *v* belongs to. We define a threshold such that if the probability exceeds the threshold, we conceive two nodes belong to the same community. The algorithm to generate an initial set of communities is depicted in Figure 4.



```
Algorithm: Generate_Initial_Communities
Input: a citation network G=<V, E>, threshold H
Output: a set of communities C
Description:
1.  initialize a set of nodes V*=V, C={};
2.  while V* is not empty, repeat the following steps:
    a)  Get a node n from V*, and create a new
        community c, then set c={}. Create
        a neighbor queue Q to store
        neighbors (not belong to c) of nodes belonging to c.
    b)  While Q is not empty, repeat following steps:
        i.  Get a node n* from Q, and add it into c.
            Find all neighbors of n* that not belong to c,
            and for each neighbor (denoted as n**) of
            n*, calculate the probability P ($c_{n*}=c_{n**}$).
            If P($c_{n*}=c_{n**}$)>H, then add n** into Q.
    c)  Add c into C.
3.  Output C.
```

Fig. 4: Algorithm to generate initial set of communities

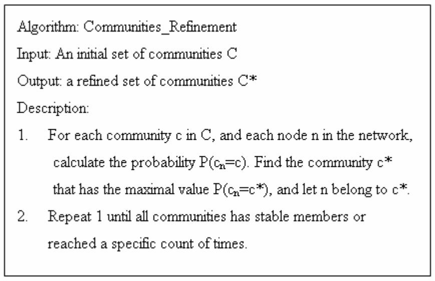
This algorithm checks every two connected nodes if their probability of belonging to the same community is bigger than the threshold, and finally output a segmentation of nodes in the network which is corresponding to an initial set of communities. It has O(m*d) time complexity, where m is the number of edges and d is the average degree of each node.

4.2. Stage Two

The set of communities attained in stage one needs further refinement, because some communities in that set may only contain very few nodes. The reason for this is that a node's neighbors are too sparse. To refine the initial set, we need a method to calculate how much a node belongs to a community. Let *u* be a node, c be a community, and then

$$P(c_u = c) = \frac{\sum_{v \in Neighbor(u)} \delta(c_v, c)}{|Neighbor(u)|}$$

is the probability that *u* belongs to community c. *Neighbor*(*u*) is the set of neighbors of *u*. $\delta(c_v, c) = 1$, when $c_v = c$, otherwise $\delta(c_v, c) = 0$. This method is based on the idea that if a node has more edges connecting to nodes belonging to a certain community, this node is more likely belong to this community. The refinement algorithm is depicted in Figure 5.



```
Algorithm: Communities_Refinement
Input: An initial set of communities C
Output: a refined set of communities C*
Description:
1.  For each community c in C, and each node n in the network,
      calculate the probability P($c_n$=c). Find the community c*
      that has the maximal value P($c_n$=c*), and let n belong to c*.
2.  Repeat 1 until all communities has stable members or
    reached a specific count of times.
```

Fig. 5: Algorithm to refine initial set of communities

The algorithm runs in an iterative way, and has O(n*t*r) time complexity, where n is the number of nodes, and t is the number of communities in the initial set C, and r is the number of iterative times. Usually both t and r are small compared to n. Usually the citation network is sparse, so our whole community detection algorithm's time complexity is O(n*(d+t*r)), where (d+t*r) is small compared a large n. So it is faster than other detection algorithms whose time complexity is O(n^2).

4.3. Initial Experiment

We have tested our community detection algorithm on a well-known graph from the social networks literature. This is the "karate club" network of Zachary [11], which was studied previously by a number of others in community detection context. The network represents the pattern of friendships amongst the members of a karate club at a US university, constructed from ethnographic observations by Zachary over a period of two years in the early 1970s [6]. During the period of study, the club split into two as a result of a dispute between two factions.

In Figure 6, we show the result of communities of the karate club network using our algorithm described above. In this case, we set the threshold to 0.5, and it works well. If finds the known split of the network into two groups nearly perfectly. Only one vertex (vertex 10) is classified wrongly, because it is on the border between two communities.

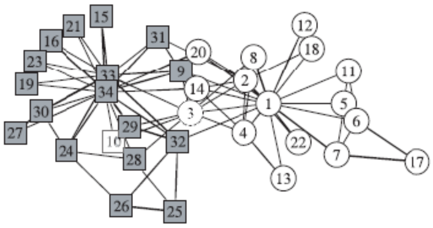


Fig. 6: The two communities into which the club split during the course of the study are indicated by the squares and circles, while the dark grey and white show the communities of the network found by our algorithm.

5. Conclusion

Current scholar search engines do not mine the relation among theses fully, and we proposed a novel scholar search engine based on community detection in citation networks to help researchers find theses fitting their research interest easily. Meanwhile, we designed a community detection algorithm to detect topics within volumes of articles. Initial experiment shows that it produced effective result. Because of the complexity and volume of work to achieve functions of TopSeer, it is still being developing.

6. Future Work

We proposed the design of TopSeer, but it still need tests from users to justify its usefulness. Besides the citation network of articles, we can explore the network of authors to find research communities. We may figure out a better way to calculate the probability of how much two nodes belong to the same community.

7. Acknowledgement

The work reported in this paper is supported in part by the National Natural Science Foundation of China under Grant No. 60673122, the Foundation of Shenzhen technology plan under Grant No. 200731.

8. References

[1] Oliver A. McBryan, “GENVL and WWWW: Tools for Taming the Web,” In M. Verleysen, editor, *First International Conference on the World Wide Web*, May 25-26-27, CERN, Geneva (Switzerland), 1994.

[2] http://scholar.google.com/

[3] K. D. Bollacker, S. Lawrence, C. Lee, “CiteSeer: An autonomous web agent for automatic retrieval and identification of interesting publications,” *In Proceedings of 2nd International ACM Conference on Autonomous Agents*, ACM Press, pp.116-123, 1998.

[4] C. LaGuardia, “E-views and reviews: Scopus vs. Web of Science,” *Library J.(online)*, January 15, 2005.

[5] C. L. Giles, K. Bollacker, S. Lawrence, “CiteSeer: An Automatic Citation Indexing System,” *Digital Libraries*, ACM Press, New York, pp.88-98, 1998.

[6] M. Newman, “Detecting community structure in networks,” *The European Physical Journal B-Condensed Matter*, 38:321-330, 2004.

[7] G. Michelle, M. E. J. Newman, “Community structure in social and biological networks,” *PNAS*, 99(12):7821-7826, June, 2002.

[8] A. Clauset, M. Newman, C. Moore, “Finding community structure in very large networks,” *Physical Review E*, 70(06611), 2004.

[9] N. Du, B. Wu, X. Pei, B. Wang, L. Xu, “Community Detection in Large-Scale Social Networks,” *Joint 9th WEBKDD and 1st SNA-KDD Workshop’07*, August 12, 2007.

[10] J. Lafferty, A. McCallum, F. Pereira, “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data,” *International Conference on Machine Learning (ICML)*, 2001.

[11] W. W. Zachary, “An information flow model for conflict and fission in small groups,” *Journal of Anthropological Research* 33, 452-473, 1977.