AceMap: A Novel Approach towards Displaying Relationship among Academic Literatures

Zhaowei Tan
Shanghai Jiao Tong University
dilevski_tan@sjtu.edu.cn

Changfeng Liu
Shanghai Jiao Tong University
stevenevets@sjtu.edu.cn

Yuning Mao
Shanghai Jiao Tong University
morningmoni@sjtu.edu.cn

Jiaming Shen
Shanghai Jiao Tong University
sjm940622@sjtu.edu.cn

Biao Wang
Shanghai Jiao Tong University
sweetmvp24@sjtu.edu.cn

Luoyi Fu
Shanghai Jiao Tong University
yiluofu@sjtu.edu.cn

Li Song
Shanghai Jiao Tong University
song_li@sjtu.edu.cn

Xinbing Wang
Shanghai Jiao Tong University
xwang8@sjtu.edu.cn

ABSTRACT

A large number of papers are published every year, which makes it difficult for researchers to grasp the relationship among the scientific literatures and the big picture of academic fields. Therefore, we build this academic system, AceMap, to analyze the academic big data, present the results through a novel "map" approach, and thus help the researchers better do their work.

Unlike existing academic systems which mainly adopt text-based methods, AceMap displays the information in a clear and intuitive way. Currently, AceMap contains the following functions: dynamic citation network display, paper clustering, academic genealogy and academic path finding. We design distributed network analysis algorithms, perform the algorithms in a Spark system and utilize modern visualization tools to present the results.

Categories and Subject Descriptors

H.3.5 [**Information Storage and Retrieval**]: Online Information Services—*web-based services*; G.2.2 [**Discrete Mathematics**]: Graph Theory—*Network problems*

Keywords

Academic big data, Visualization, Network analysis

1. INTRODUCTION

Activities on scientific research play a strategic supporting role in improving the social productivity forces as well as the global national strength. Countries around the world put great emphasis on the scientific research, investing money, people and other resources.

As researchers, we personally feel privileged to work in such a supportive environment. However, we sometimes also feel helpless – with a limited reading ability and time, we have to face thousands of new papers in the field of study, for scientific research is so active nowadays. It is simply impossible that we read every publication and make clear the relationship among them. In fact, all we want and need is just a small portion of the papers, either the most important or the most relevant ones. The rest are not as important or relevant, and we might ignore them and save our time to focus on the important part.

Therefore, we feel it urgent to build a system capable of analyzing the attributes of the papers and the relationship among them exclusively for researchers. Using this system, the scholars are able to clearly see the property of one paper and know which one to pinpoint after seeing the big picture in a field. In a word, we want to build a system that makes doing research much easier than before, and even better than before.

Currently, several research entities and companies have already developed some systems to support the academic activities, including Google Scholar [4], Web of Science [5] and dblp [12]. However, these systems primarily focus on textual contents of publications, namely the metadata of one specific paper, instead of displaying a global view of the whole academic area. In addition, they generally fail to provide the users with a straightforward way to comprehend the relationship among scientific literatures. In the meantime, some systems like AMiner [21] do the pioneering job of digging deep into the academic data, and display them in a modern way. But we try to realize some functions different from them, and focus more on map visualization and network analysis.

Inspired by geographic maps, we build this novel academic map, AceMap, to provide better services for researchers. In geographic map, a user can: (1) drag the map and see other regions adjacent to the current place displayed on screen; (2) zoom in or zoom out to see the region in different scale (note that the places displayed in various scale are different, e.g. only state capitals when we see the whole US while

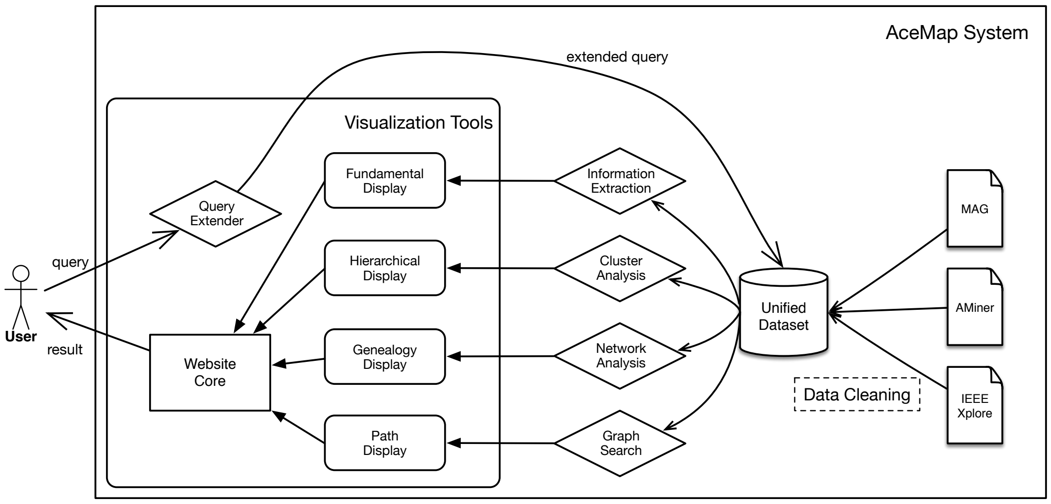


Figure 1: The Architecture of AceMap.

many of cities after we zoom in); (3) calculate and display the path between two places, e.g. the route from home to the school. Correspondingly, in this prototype version of the system AceMap, using the citation data collected from the Internet, we realize these functions among papers: (1) dynamically unfold and fold a paper's citations, checking what is "the region around this publication"; (2) cluster the papers in different levels, using algorithms to find community or field; (3) discover paths between two papers and display the paths. Besides, we also design algorithms to find the "genealogy" of a paper, i.e. given a single publication and its references, we try to answer this question: which ones among these references are of the most importance to generating this paper? Recursively, we try to traverse the whole citation network and generate a "family tree" of the paper, finding the ancestor with the original idea,.

Building an academic map is a tough task, for it is new and fresh. If we want to construct a geographic map, we can refer to the international standard and make a map that everybody with common sense can read and understand. For an academic map, since no one has ever made one, we have to explore ways to display useful information clearly, and make the users feel comfortable about the map.

We start from giving a system overview in **Section 2**, which gives a brief description of AceMap, introducing its functions and significances. Then we elaborate the details of how these functions are realized in **Section 3**. In **Section 4**, we provide the outlook of our system, after which some related work are presented in **Section 5**. Finally, we conclude this paper in **Section 6**.

2. SYSTEM OVERVIEW

2.1 System Architecture

The overall architecture of AceMap system is shown in Figure 1.

The first step to build AceMap is to get the data. We mainly use the Microsoft Academic Graph (MAG) [20]. It is a heterogeneous graph containing scientific publication records, citation relationships between those publications, as well as other information. After integrating this dataset with several others, such as AMiner dataset [21], we build a unified database to support the various functions.

Since the MAG dataset is pretty large, with tens of millions of publications and citation relationships, we extract a connected subgraph with 2,755,844 papers and 5,547,769 reference links between them. We use this relatively small dataset to perform our algorithms and build the prototype system.

When a user comes, he interacts with our system by website GUI. The server handles the input query using the unified database and sends the result back to the visualization tools, who are responsible for displaying the relationship among the academic literatures in a clear and intuitive way.

2.2 A Guided Tour

Since we have presented the AceMap's top level design, we now introduce what the users can do or achieve using AceMap system.

Different from the existing text-based methods, we implement a new method to display the topology of papers using D3.js [3], a JavaScript library for web-based data visualization. We use the JSON files to store the data and load it dynamically into the front end. By this approach and some novel algorithms in the back end, we present the relationship among academic literatures in a vivid and interactive way. We now list and elaborate the basic functions of our system, and their corresponding significances.

2.2.1 Fundamental Display

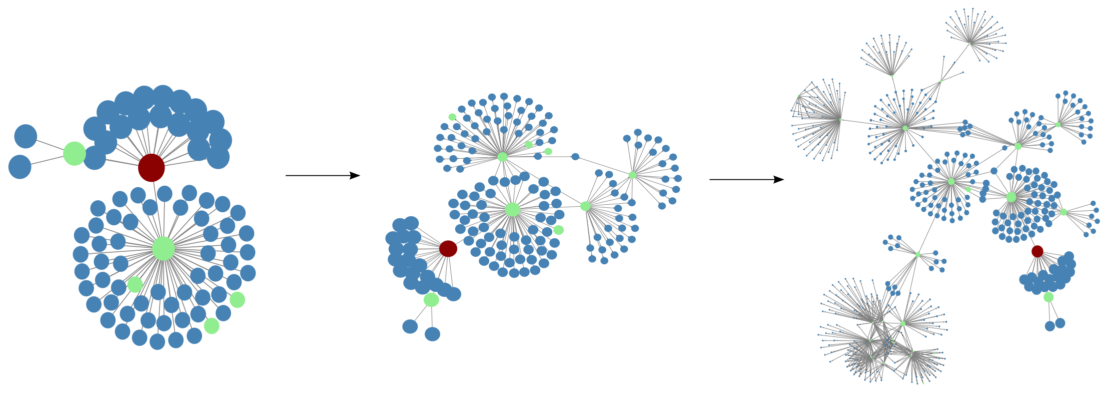


Figure 2: The graph shows the "ancestor" of the center paper. By clicking the nodes, networks of the references are generated, during which the user can explore the citation network of the center paper.

Function: In this interactive graph, we display a network centered on the paper that the user just clicked. Each node represents a paper, and each link represents a reference. In addition, an arrow mark is placed on the link to indicate the direction. The node is expandable as long as there is corresponding data in the database. When the user clicks on one node which is not expanded, it will be expanded to show its references. Similarly, the user can collapse a node in the same way. A dynamic process is illustrated in Figure 2. The user can hover on one paper to see the specifics of it such as title, publication venue, and publication year, which is both convenient and compact.

A slider offers the function of limiting the maximum hops between the papers displayed and the center one. Hence the user can heed the papers which are shorter in distance. In addition, by zooming in and out, the user is able to view the network at various scopes. He can either focus on the relationship among a few papers or have a broad view of the local clusters. Plus, we provide the user with the function of saving the network graph into SVG file, so the users could save it conveniently for further use.

Significance: In this view, the user can have an understanding of the topology of the reference network. With the help of the links between papers, the user can have a deeper understanding of the specific paper and the papers nearby. By clicking nodes, the user can see a dynamic network of the papers, during which he can explore the history and the process of the development of the papers. To sum up, the user can have an intuitive impression of the paper he is interested in and have a general idea of its "neighbors" and "ancestors".

2.2.2 *Hierarchical Display*

Function: Academic papers accumulate every year and therefore the network composed of these continuous papers and citation relationships among papers is becoming larger and more complicated. Here our focus is clustering these papers into several categories according to this network. We suppose each paper is a node and each citation from paper A to paper B is an edge from node A to node B in this network. So each citation from paper A to paper B suggests that A and B are closely related thus it is possible that they belong to one category. Based on this assumption, clustering this large network can be seen as a community detection problem.

First we use some cluster analysis algorithms to group the academic literatures at different levels. Then we display the clusters of academic literatures on the website in an interactive way. Each cluster is represented by a circle, whose diameter indicates the number of papers in this cluster. The user can click one of the clusters to see the sub-clusters of it and then continue digging deeper to see a smaller field or return to the previous level conveniently. We show the clustering in Figure 3 and Figure 4. The user can see information including the total number of academic literatures in it and a brief description when hovering on one cluster.

Significance: The interactive display of the papers' clusters provides the user with a straightforward way to have an intuitive idea of what the area he is focusing on is like, and what the whole field and the sub-fields are like. Moreover, he can realize the importance of one field and the relationship between fields by looking at the clustering result.

2.2.3 *Genealogy Display*

Function: Academic genealogy algorithm achieves two functions:

1) In a given citation network, when a root paper is chosen, we calculate the relative importance scores of related papers. One paper's relative importance score reflects the influence this paper has on the root paper.

2) By implementing the above algorithm in a distributed system, we can largely raise the computational efficiency, which enables user-oriented quick search functionality.

We define the score of one paper as its importance or influence to the center one. This attribute of the paper is shown

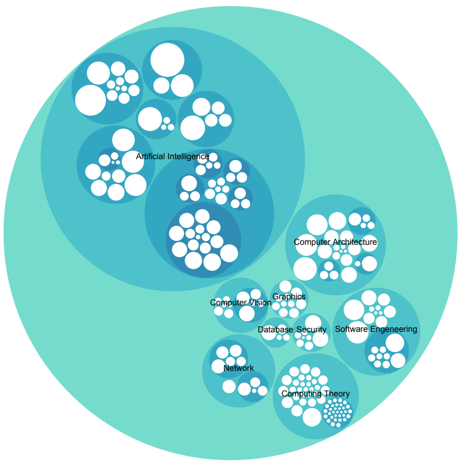


Figure 3: The cluster graph shows several fields of computer science.

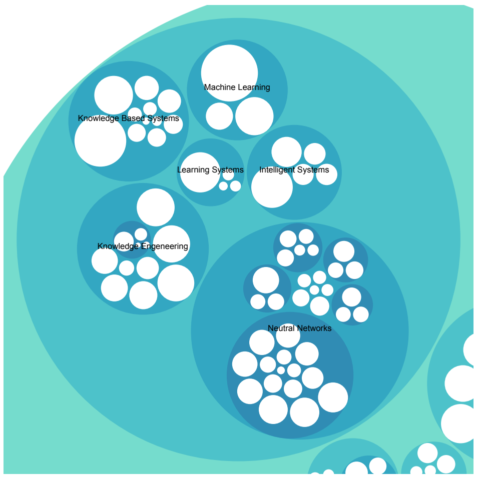


Figure 4: The effect after clicking the cluster "Artificial Intelligence".

vividly by the size of the node. Therefore, we are able to provide a slider at the top of the web page, which enables the user to filter the papers with relatively lower score if he wants to concentrate on the papers which are essential to the root paper. This feature is presented in Figure 5.

Significance: When a researcher finds a paper of interest (root paper), he might want more information about how the ideas in this paper are formed. In general, he will look for the direct or indirect citations and references of this paper. This is similar to finding its ancestors by constructing a genealogy. But there are two obstacles: 1) The number of citations is increasing exponentially as the citation layer accumulates. 2) Not all the cited papers have deep influences on the root paper – especially when used as background knowledge. Therefore, it is a rather heavy and inefficient job for researchers to find the important ancestors of the root paper in a large citation network.

Therefore, we propose novel algorithms to build an academic genealogy. Our algorithms quantify the influence to a certain paper, and our system presents the information flow in a citation network with data visualization techniques. Then the users can start from one paper and find the development skeleton of this field by omitting some insignificant papers.

2.2.4 Path Display

Function: The back end programs can be executed to find the path between two arbitrary papers in the citation network, and return the answer to the visualization tool. Therefor, the path from one paper to the center one (the paper that the user just clicked in the previous view, marked with a unique color) and the related papers (the papers on the path and the references of this paper) will be highlighted in different colors, while the opacities of other links and nodes are increased so as to avoid distraction. The users can thus clearly see the relationship between the two papers, and the Figure 6 and Figure 7 are two examples.

Significance: Through learning the academic paths between two papers, we can get the intuitively relationship between them, and thus draw the outline of the development in the related fields and find some important papers between among the field. Also, a certain degree of analysis can be conducted on the content correlation between papers, thus providing the researchers with an explicit direction and a convenient way to learn the relationship. With the help of the highlight of paths, the user can acquire plenty of information about papers' relationship.

3. IMPLEMENTATION DETAILS

3.1 Fundamental Display

This view is implemented based on a layout of D3.js named force. We add many functions to make the interface more user-friendly. Some major techniques we use are described as follows.

We load the whole data in the JSON file into some variables, say nodes and links, but do not display them at once. By using such variables we can calculate whether some nodes are connected or not. Whenever the user clicks on one node, the corresponding nodes will be added to the network displayed. We also judge whether the references of that clicked paper is already in the network or not, and handle the two situations separately.

In genealogy display, we calculate the scores of each paper and stored them in the JSON file for the use of filtering. When doing filtering, it is also intractable in that nodes and links are stored in different variables and changing according to the input of the user. We hence have to hide them respectively and carefully. We hide the nodes and links recursively so as to avert missing.

We also use asynchronous techniques since loading too much data at once is not realistic for the browser to handle. When the user has navigated to a node on the margin of the graph, which means no reference information of it is stored in the currently-loaded JSON file, the system can fetch the data automatically from the back end database and merge it into the existing network.

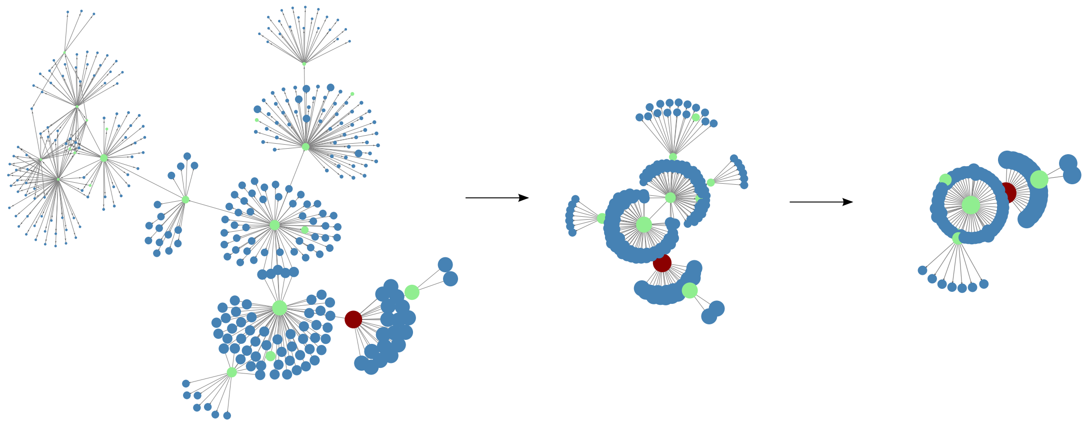


Figure 5: The graph displays the process of score filtering. The subfigure on the left is the whole network of papers. The one in the middle shows the effect after filtering papers with low scores. The one on the right is the result after increasing the threshold of filtering.

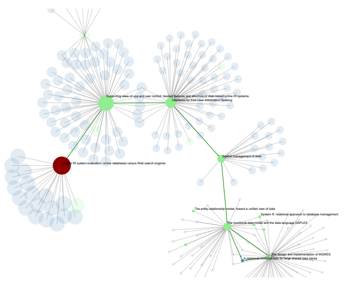


Figure 6: Finding multiple paths to the center paper.

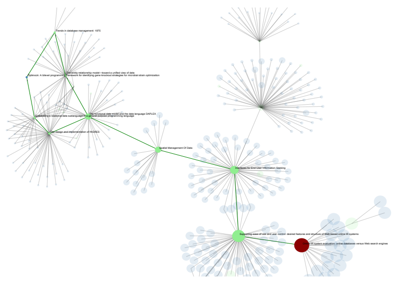


Figure 7: Finding path to the center paper in a complicated network.

3.2 Hierarchical Clustering

The implementation of the dynamic cluster graph is based on another basic layouts of D3.js named pack. We use predefined format in the JSON file so as to load the data into the browser efficiently. By using the cluster algorithm mentioned above, we store this relationship between cluster and its sub-clusters as parent and children in advance. We also calculate the size of each cluster in advance to save the running time. In addition, we add a callback function to navigate to the next detailed view, which will be introduced when the system detects that the user has explored to the deepest level.

We use the Label Propagation Algorithm (LPA) [18] to cluster the papers. LPA is an extremely fast algorithm because it has the advantage of nearly-linear running time. Another strength of LPA is that it needs small amount of a priori information about the network structure. As a result, it is widely used in large scale networks for community detection. So we use LPA through Apache Spark [2] for clustering. We leverage Apache Spark with 6 executors each with a memory of 30GB to implement a parallel version of LPA method. For our experiments, as mentioned before, we use a large academic network containing more than 2 million papers and more than 5 million citation relationships. Our system clusters these papers approximately into 100 communities within linear time.

Algorithm 1 Distributed cluster analysis algorithm

```
Require: graph$(V, E)$
1: $g = graph.mapV((vid, vdata) \rightarrow vid)$
2: $msg = g.aggregateMsg(sendMsg, mergeMsg)$
3: for $i = 1 : maxIterations$ do
4:     $g = g.joinV(msg)(vprog)$
5:     $oldMsg = msg$
6:     $msg = g.aggregateMsg(sendMsg, mergeMsg)$
7: end for
```

The basic idea of this algorithm is as follows. At the initial stage, each paper in the network is assigned to its own community. At each step, papers send their community affiliation messages marked by their own ID numbers to all neighbors and update their states to the mode community affiliation of incoming messages.

3.3 Academic Genealogy

3.3.1 The Concept of the Relevant Importance

How to define the influence that a paper makes to the selected paper? We use the concept of relative importance [22] to describe it. Relative importance is used to estimate the influence of other papers on the selected root paper in the citation network. Contrary to relative importance, global importance currently can be estimated in various algorithms, such as PageRank [16] and HITS [9]. Specifically, PageRank is based on the links of the entire network. After iterations, it calculates the score of every node, which can be achieved before the users input search contents. On the other hand, HITS requires that the users query beforehand, according to which the system will look for a subset of one related point, and subsequently scores will be graded within this subset. Essentially, this process is the global grading of the localized network. Let's take an example. Paper X cites paper Y and paper Z at the same time. Global importance of paper Y is lower than that of paper Z (paper Z is more popular in the whole citation network). However, as the main idea of paper X comes from paper Y and it cites paper Z just as background knowledge or for a mathematical tool. Then, if we set paper X as the root paper, paper Y shall get a score higher than paper Z.

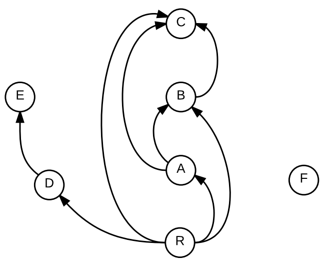


Figure 8: A simple citation network.

3.3.2 *Principle of the Algorithm*

To find the ancestors of the selected paper is one of the main objects of academic genealogy algorithm. Generally, ancestors refer to the the nodes where there exist routes from root paper to that papers in the citation network. For instance, in the Figure 8, all of A-E are ancestors of root paper R, for routes exist from R to A-E, while F is not an ancestor because the route does not exist accordingly. We define that a paper is a significant ancestor if it generates a whole field of study, propose a new problem, or invent a novel methodology (e.g., C in the Figure 8), and hence it might be directly cited by every descendant paper which is enlightened by it (A,B,R in the Figure 8). As for root paper R, it cite some important descendants of C (for instance, R is a refinement based on A). Meanwhile, A cites C anyway (A is in the domain generated by C). Therefore, between C and R, routes of different lengths exist. We can, according to the structure, give a higher score of relative importance to C, if every route denotes a certain degree of contribution from C to R (or rather, every route enhances the relative importance of C to R).

3.3.3 *Some Basic Assumptions*

As mentioned before, we assume more paths from the root paper R to another paper P may bring P a higher score in relative importance. Based on this method, here we give several basic assumptions on calculating the relative importance.

CLAIM 3.1. *A directed path from paper R to paper P means that P has some influence on R.*

CLAIM 3.2. *A paper would refer directly to the ancestors that have the most significant influence on it.*

If some papers leave strong impact on a selected paper, whether they are groundbreaking papers that open up the academic field of the selected paper or the direct idea sources that inspire the selected paper, the selected paper is extremely likely to refer to these papers because of academic integrity. Therefore, a paper that opens up a new field would be cited by most of the papers in this field.

CLAIM 3.3. *The shorter the path is, the more significant influence the end papers of the path would have on the start paper.*

Based on Claim 3.2, since the papers with the most significant influence are directly cited which means they have the shortest path length equaling one, we think that the shorter the path is, the more directly the end papers would leave impact on the start paper.

CLAIM 3.4. *After passing K_0 layers (K_0 is a relatively small natural number), the subsequent papers will have ignorable influence to the root paper.*

We consider that each point would allocate its own score to its citations equally. If the number of citations of every cited paper stays in a stable range, the score S will decrease exponentially along with the increasing of the length of path. Therefore, the papers, after passing K_0 layers, will have ignorable influence to the root paper, which means that if we choose a proper value K_0, we can obtain a relatively more precise outcome while reducing the computational cost effectively.

3.3.4 *Formula and Algorithms*

Based on the previous basic assumptions, we define the formula that calculates S as below:

$$S^n(u) = S^{n-1}(u) + \sum_{v \in B_u} \frac{S^{n-1}(v)}{N_v}$$

u, v represent some papers in the citation network; $S^n(u)$ represents u's score of relative importance after the n-th layer calculation; B_u is the set of papers that cite paper u but have not assigned their own scores to their reference papers including paper u; N_v represents how many papers v cites.

At the beginning, we assign 1 to the score of the root paper and 0 to other papers, and then we get the score through iterative calculation, which goes under the BFS algorithm. We take the length of the longest path from current paper to the root paper that has been scanned as the layer of current paper K. In this algorithm, we take $K_0 = 7$. When the calculation in the 7-th layer is over or there are no new papers to be graded, the iterative process stops.

After acquiring the citation network, we calculate the scores of relative importance. We claim that the citations of a paper contain both those important to it and those unimportant, such as the papers that introduce the background. Those papers that are cited directly and have great influence on the core papers, contain both groundbreaking papers in its domain and some lately published papers which bring some inspiration to the root paper. Those cited papers that are lately published, are likely to cite the same groundbreaking paper. Starting from the root paper, we distribute each paper's score to those which is cited by it as an increment to update scores of papers layer by layer. Since groundbreaking papers in those domains are cited by papers from different layers, they get higher scores. The citation paths from high score papers to low score papers display the development and the evolvement of a single paper. And the generation of various paths in a field provides a panorama of the whole field. The algorithm is described in Algorithm 2.

Algorithm 2 Academic genealogy scoring algorithm

```
Require: $G(V,E)$, $v_0$
 1: $Q \leftarrow \emptyset, S \leftarrow \emptyset$
 2: $v_0.score = 1$
 3: add $v_0 \rightarrow Q$
 4: add $v_0 \rightarrow S$
 5: while $Q \neq \emptyset$ and $K < K_0$ do
 6:     $v = Q.DeQueue()$
 7:     update $K$ as the length of current path from $u$ to $v$
 8:     $n = RefNum(v)$
 9:     for $\forall v'$ satisfies $e(v', v) \in E$ do
10:         $v'.score = v'.score + v.score/n$
11:         if $v' \notin S$ then
12:             add $v' \rightarrow Q$
13:             add $v' \rightarrow S$
14:         end if
15:     end for
16: end while
```

As the citation network G and the root paper v_0 are given, the algorithm calculates the relative importance scores of all papers by analyzing the citation graph. Q is the process queue and S is the set of papers that have contributed to the scores of their referred papers. First of all, we let Q and S be empty sets and make the score of root paper v_0 equal one. Then we add root paper v_0 to Q and S. After that, as long as there are papers waiting for updating scores and the calculation layer K is less than K_0, we continue the scoring process. We get a paper v from Q and update the value of K. For each referred paper of v, for example v', we divide the score of v evenly and add it to v'. If this is the first time that paper v' updates its score, we add v' to Q to update its referred papers' scores. We also add v' to S to avoid updating the score for one paper twice.

3.3.5 Distributed Algorithms

Because of the huge amount of data and the complexity of the network, the computational cost of the algorithm processing in a single computer is prohibitively high. Therefore, we apply our algorithm in distributed system to increase computing speed and ensure good user experience.

We adapt Algorithm 1 to a distributed version and implement it in Apache Spark. The program calculates the increment of scores layer by layer and updates the scores when the calculation of one layer is over. Because the calculation processes of different papers in the same layer are independent, we assign the calculation tasks of one layer to different executors which increases the speed of calculation greatly. The program reads the key information of each paper, such as paper ID, reference list and score before computing and then run the program on Apache Spark. After computation, we obtain scores of related papers and extract their metadata from the database. According to the requirement of the front end, we combine these two kinds of information into a JSON file and send it to the front end for visualizing the citation network. The distributed algorithm is described in Algorithm 3.

Algorithm 3 Distributed academic genealogy scoring algorithm

```
Require: $G(V,E)$, $v_0$
 1: $scores \leftarrow [v_0, 1]$, $visited \leftarrow v_0$
 2: $G = sc.paral(G)$
 3: $scores = sc.paral(scores)$
 4: $parentScores = sc.paral(scores)$
 5: $visited = sc.paral(visited)$
 6: while $parentScores$ is not empty do
 7:     $GAndScores = G.join(parentScores)$
 8:     for $\{parentPaper, [refPapers, score]\}$ in $GAndScores$ do
 9:         $meanContr = score/refPapers.num$
10:         for $refPaper$ in $refPapers$ do
11:             $[refPaper, meanContr] \rightarrow contri$
12:         end for
13:     end for
14:     $newScores = scores.union(contri).reduce(\text{add}())$
15:     $scores = newScores$
16:     $refPapers = GAndScores.refPaper()$
17:     $newVisited = visited.union(refPapers())$
18:     $newParentScores = scores.filter(\text{item in } refPapers \text{ but not in } visited)$
19:     $visited = newVisited$
20:     $parentScores = newParentScores$
21: end while
```

Table 1: The comparison between algorithms using different root noded.

Node ID	Basic time(s)	Spark time (s)	Times faster
00847E27	1261.594444	123.2824709	10.23336436
0084C210	833.61658	101.454792	8.216630909
0099CFC7	4688.261326	132.0533612	35.50277921
007CC96C	1179.102978	100.3532622	11.74952316

Here we use Spark to speed up the calculation of the scores. First, we change the queue into several layers in which there can be many nodes being computed at the same time, and reduce the dependence of the calculation procedure of each paper's score. We use the center paper to start the loop and set the score of it as 1. In one loop, we use several parent papers to give contribution to its reference papers in two parallel computations, one uses the map function and the other uses the reduce function. Using the map function, we map the "links and references" data, such as $(parent\_paper, \{[refer\_paper\_A, refer\_paper\_B, ...], score\})$, into "papers and contribution" data, such as

Table 2: The results of path finding.

Dataset	8087KB	141171KB
Preprocessing Time	< 1s	30s
Accesible Range	About 20 layers in 20s	Inside 5 layers
Path found	48	122
Speed	Seconds for 4 layer, 17s for 18 layers	Seconds for 4 layers, minutes for 5 layers

$[(refer\_paper\_A, average\_contrib)$, $(refer\_paper\_B, average\_contrib), ...]$. Using the reduce function, we sum up the original scores and contributions of each paper. In the next loop, we use the original parent papers' reference paper which have not been visited before, as the new parent papers. When there are no reference papers which have not been visited before, we can stop the loops.

The comparison of two algorithms are shown in Table 1.

3.4 Academic Paths

We take two classic methods, BFS and DFS respectively, to find the academic paths. The results of two experiments in two sub-datasets with different sizes are shown in Table 2. (the following are results of performing BFS, and if we use DFS the results are almost the same)

As you can tell, when the size of the dataset grows, the computational cost increases vastly. Therefore, considering the scale of the data, we use a parallel DFS algorithm to speed up the procedure of finding paths between two papers. We start the loops from the start paper. In a loop, we find out the reference papers for a group of parent papers and note down their paths, and this is executed as a parallel procedure. Because we have limited reference data, we can end the loop when we cannot find the reference papers any more.

Algorithm 4 Distributed path finding algorithm

```
Require: $a$,$b$  ▷ $a$ is the start paper and $b$ is the end paper
Require: $G(V,E)$
 1: $parent \leftarrow [a]$
 2: $parseLinks \leftarrow \emptyset$   ▷ $parseLinks$ is the link from the reference paper to its parent paper
 3: while $parent \neq \emptyset$ do
 4:     for $paper$ in $parent$ do
 5:         for $[parentPaper, refPaper]$ in $E$ do
 6:             if $parentPaper == paper$ then
 7:                 add $[refPaper, parentPaper] \rightarrow parseLinks$
 8:                 if $refPaper \neq b$ then
 9:                     add $refPaper \rightarrow child$
10:                 end if
11:             end if
12:         end for
13:     end for
14:     $parent = child$
15: end while
16: $tmp \leftarrow [b]$
17: while $tmp \neq \emptyset$ do
18:     $tmpChild \leftarrow \emptyset$
19:     for $paper$ in $tmp$ do
20:         for $[refPaper, parentPaper]$ in $parseLinks$ do
21:             if $refPaper == paper$ then
22:                 draw path $parentPaper \rightarrow refPaper$
23:                 add $parentPaper \rightarrow tmpChild$
24:             end if
25:         end for
26:     end for
27:     $tmp = tmpChild$
28: end while
```

4. OUTLOOKS

After we build this prototype system, we have some other functions that are being built or planned. These projects can roughly be divided into two categories. The first one is map improvement. We stick to the idea of building an academic map and try improving the quality, or enriching the functions of AceMap. Another part is some functions outside the scope of a map, and we also have some brilliant ideas to provide convenient services for the researchers.

4.1 Map Improvement

4.1.1 Map Objects Classification

First and foremost, we will classify different papers into several categories, and mark these classes with different colors or shapes in our map. When we look at a map, we can recognize that which place is a bridge, which place is a supermarket and which place is a tourist attraction. Different places hold different functions and statues inside a city. Similarly, papers play different roles. And we are eager to know whether a paper is a groundbreaking pioneer, a great follower, or just a useless one that you can simply ignore. This conception is depicted in the Figure 9.

This function is a great auxiliary to the current path and genealogy function. Imagine that when we find the genealogy of a paper, we can clearly see the roles in that tree – some papers might create a new research area, some follow them and finally the selected paper is generated. When we use the path function, we will be able to see a more clear relationship between publications, e.g. they both follow a great paper which generates a new research area, etc.

4.1.2 Integrated Database

Our current dataset depends largely on MAG dataset. However, this dataset has several problems. First, we cannot guarantee the data quality. Some papers lack the metadata and others are not scientific publications at all, e.g. some are online advertisements. Second, this dataset does not contain any abstract. Without the text information, we are not able to perform some particular algorithms, like LDA algorithm [6] to cluster our publications.

Therefore, a task to be done is to build a better, more complete database to support the current and the future functions. One choice is to use some other open source dataset

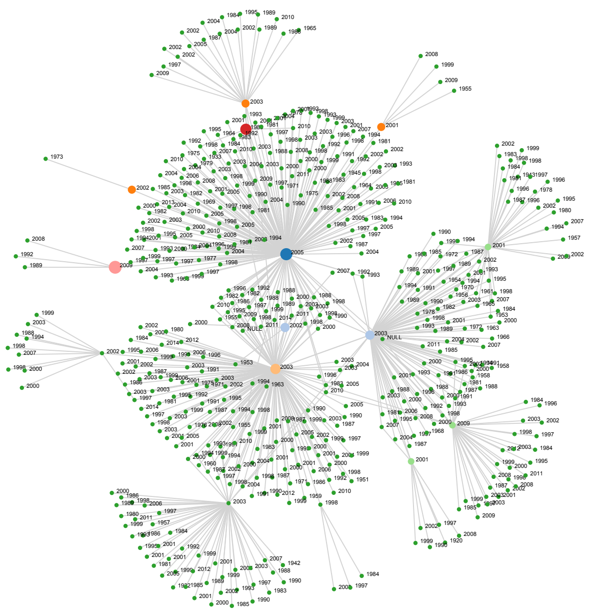


Figure 9: An example of the graph. Each node has a different size, indicating its importance. And the nodes are shown in different colors to indicate their status in the network.

with text information, like AMiner[21], while another is to write crawlers to collect information from the Internet.

4.1.3 *Enhanced Path Finding*

In a relative small network, using simple algorithms like DFS or BFS is fast enough to get the results. However, when the scale of the data increases to a certain level, current algorithms will no longer be suitable for the system. Therefore, new approaches will be taken to find paths more efficiently and more thoroughly.

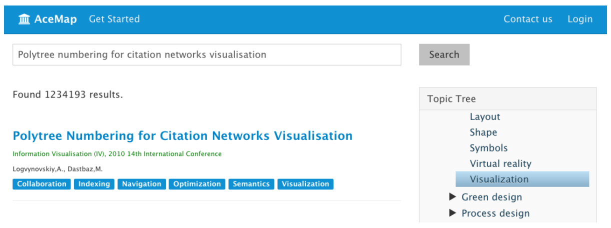
An intuitive idea is to explore important nodes first. We will try different centrality index to test their respective performance and take the best one.

4.2 Other Functions

4.2.1 *Integration with Search Engine*

Currently, we have already built a search engine at `acemap.sjtu.edu.cn`. The website (Figure 10) takes the advantage of Solr [1] framework and can return the results after typing in a query. After perfecting the website and the engine, we will integrate the map and the search engine together. i.e. when a user searches a paper and check the paper's detailed information, we will direct him to the map system and set that paper as the root paper. The user can then check its genealogy, citation network, etc.

Another merit of integrating search engine is that we can find a path in an easier way. The user can directly type in



AceMap Get Started Contact us Login

Polytree numbering for citation networks visualisation

Search

Found 1234193 results.

Polytree Numbering for Citation Networks Visualisation

Information Visualisation (IV), 2010 14th International Conference

Logvynovskiy,A., Dastbaz,M.

Collaboration Indexing Navigation Optimization Semantics Visualization

Topic Tree

- Layout
- Shape
- Symbols
- Virtual reality
- Visualization
- ▶ Green design
- ▶ Process design

Figure 10: A Snapshot of the Website.

two papers and get the path between two papers, instead of finding these two in the map.

4.2.2 *Academic Recommendation*

Existing search engines return the same results to a query. However, everyone wants something different. For instance, a novice who types in "networking" might want some surveys or introduction papers, while a professor who studies networking and types in the same keywords is more probably want some cutting-edge papers. Therefore, a customized result is preferred. We now allow users to log in and establish his own profile. In future, we will design some algorithms to collect the data, tag the users and customize the user's search results.

More interestingly, after a conference is hold or a transaction is published, we can notify the relevant users and recommend new papers to those who might be interested in those papers. This function will greatly save the researchers' time to keep track of a newly hold conference and search for useful publications.

4.2.3 *Analysis of Conferences & Transactions and Authors & Research Entities*

What is the relationship between two conferences, such as between WWW and SIGKDD? We want to answer this question in future. After performing extensive network analysis, we will show the results in an intuitive and clear way. This can be either done using the citation information, or text correlation. Alternatively, combining two together will definitely produce some interesting results and might provide some insights of a conference or a transaction.

Similarly, the techniques could also be used to analyze relationship among authors and research entities. A user will be able to find his relationship, by citation hops or content similarity, between Einstein and him with the help of our system.

5. RELATED WORK

5.1 Relevant Systems

There are existing academic searching systems such as Google Scholar [4], Web of Science [5] and dblp [12]. Google Scholar defines itself as the way to find scholarly articles across disciplines. Web of science aims to be today's premier research platform for information in the sciences, social sciences, arts, and humanities. The dblp computer science bibliography is the on-line reference for open bibliographic information on computer science journals and proceedings. However, all these systems are text-based. In addition, they focus more on the information of a specific paper. Though displaying some static figures as auxiliaries, they fail to provide the user with a straight-forward way to comprehend the relationship among academic literatures and to have a global view of the whole academic field.

There are also systems like VEGAS [19] and AMiner [21] that focus on the network analysis. VEGAS concentrates on summarizing the large citation graph according to the user's interest and finding the impact of a highly influential paper through summarization of citation network, and AMiner centers on the evaluation of the influence of researchers by analyzing social network, while we pay more attention to the history and development, or rather, the ancestors and the neighbors of papers which the users are interested in.

5.2 Clustering

The problem of community detection has drawn much attention in recent years. For a better analysis, many algorithms have been created. Some algorithms are based on the optimization of modularity [14] - a quality function that evaluates the performance of partitioning a network. One famous algorithm that aims at optimization of modularity is Louvain Algorithm [7]. The popular and widely used Louvain Algorithm still has some drawbacks since its performance degrades greatly as the size of the graph increases [11]. SLPA's [23] parallel implementation is too expensive because it needs each processor to cache the vast size of the whole network [10], though it demonstrates an excellent performance in detecting overlapping nodes and communities. Many other algorithms are computationally expensive. WalkTrap (WT) method in [17] takes the measure of calculating similarity between nodes based on random walks to detect communities. However, in many practical cases, WT has a high computing complexity. We use LPA [18] in our system. It has a nearly-linear running time, and it needs small amount of a priori information about the network structure.

5.3 Network Analysis

Network analysis, especially node ranking has been widely studied for some time. A well-known ranking algorithm for global network is PageRank [16], which is the core of Google search engine. PageRank calculates the scores of the nodes by imitating the random walk on the link network, which is based on the theory of Markov chain. Another famous ranking algorithm is HITS algorithm [9]. Unlike PageRank, HITS requires the user to choose a few nodes at first, and then it generates a set of nodes through a several hops from the chosen nodes. HITS ranks the nodes by calculate their scores of authority and hub.

When it comes to academic citation network, designing proper algorithm for paper ranking has become a focus of research. An advisable rank of papers not only helps the scholars become familiar with a new area quickly, but also marks the importance of different papers. Some paper ranking methods have been put forward. Related paper recommendation [15] builds a bipartite graph with related papers and leverages HITS to score the papers. Then it recommends the top-ranked papers to the user. Several variations of PageRank and HITS are introduced by Ekstrand et al. [8]. In this work, the original graph ranking algorithms are combined with collaborative filtering or content-based filtering to improve the efficiency and effectiveness. The authors compare different algorithms and find that CF with PageRank performs best.

However, node ranking algorithms, such as PageRank, HITS and their variations, all try to estimate the global importance of a single node in the whole network rather than measure the influence the single node has to a specific node. White and Smyth [22] put forward a concept of relative importance which describes how much influence a node leaves on a chosen node. That is, if you choose different root nodes in the same graph, you should get different relative importance ranking for a node. This paper gives two ways to calculate the relative importance, weighted paths and Markov chains.

All the work above pay most attention to the node ranking. In our perspective, visualising the directed graph with relative importance scores in a hierarchical way is of great importance to make the users grasp the development structure of this field. Politer Numbering for citation networks visualisation [13] gives a method of visualisation of the polytree-structure network. However, the real citation networks are not always a polytree. Therefore, we design academic genealogy algorithms to measure the relative importance in citation networks and visualising the resulting scores with D3.js in a user-friendly and interactive way.

6. CONCLUSION

In this paper, we present a novel academic system, AceMap, which aims to process the academic big data, analyze the citation network and visualize the relationship among papers to help researchers grasp the academic big picture more conveniently and more intuitively. First we give the overall structure of our AceMap system, and then present the front end display effects as well as their significances. After that, we focus on the back end details which support the front end display. We implement a cluster analysis algorithm which aggregates the papers into different clusters in different levels, propose an academic genealogy scoring algorithm to calculate the relative importance of papers to the root paper, and realize an academic path finding algorithm to find the paths between two given papers. For every algorithm, we implement it in a distributed manner in Spark system and greatly save the computational time. Last but not least, we describe a clear and attainable blueprint of our future system.

7. ACKNOWLEDGMENTS

The authors thank Jinguang Zhou, Jiajun Du, Xingbang Liu, Zheng Wu, Zheng Luo, Yang Lu, Yuchao Tao, Yunqi Guo, Wentian Bao and Chuqi Wang at SJTU for their help with processing the data, designing the algorithms and implementing the system.

8. REFERENCES

[1] Apache solr. `http://lucene.apache.org/solr/`. The Apache Software Foundation.

[2] Apache spark. `http://spark.apache.org/`. The Apache Software Foundation.

[3] D3.js. `http://d3js.org/`. Mike Bostock.

[4] Google scholar. https://scholar.google.com/. Google.

[5] Web of science. webofknowledge.com. Thomson Reuters.

[6] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[7] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

[8] M. D. Ekstrand, P. Kannan, J. A. Stemper, J. T. Butler, J. A. Konstan, and J. T. Riedl. Automatically building research reading lists. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 159–166. ACM, 2010.

[9] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

[10] K. Kuzmin, S. Y. Shah, and B. K. Szymanski. Parallel overlapping community detection with slpa. In *Social Computing (SocialCom), 2013 International Conference on*, pages 204–212. IEEE, 2013.

[11] A. Lancichinetti and S. Fortunato. Community detection algorithms: a comparative analysis. *Physical review E*, 80(5):056117, 2009.

[12] M. Ley. The dblp computer science bibliography: Evolution, research issues, perspectives. In *String Processing and Information Retrieval*, pages 1–10. Springer, 2002.

[13] A. Logvynovskiy and M. Dastbaz. Polytree numbering for citation networks visualisation. In *Information Visualisation (IV), 2010 14th International Conference*, pages 86–91. IEEE, 2010.

[14] M. E. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[15] M. Ohta, T. Hachiki, and A. Takasu. Related paper recommendation to support online-browsing of research papers. In *Applications of Digital Information and Web Technologies (ICADIWT), 2011 Fourth International Conference on the*, pages 130–136. IEEE, 2011.

[16] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: bringing order to the web. 1999.

[17] P. Pons and M. Latapy. Computing communities in large networks using random walks. In *Computer and Information Sciences-ISCIS 2005*, pages 284–293. Springer, 2005.

[18] U. N. Raghavan, R. Albert, and S. Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76(3):036106, 2007.

[19] L. Shi, H. Tong, J. Tang, and C. Lin. Vegas: Visual influence graph summarization on citation networks.

[20] A. Sinha, Z. Shen, Y. Song, H. Ma, D. Eide, B.-j. P. Hsu, and K. Wang. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th International Conference on World Wide Web Companion*, pages 243–246. International World Wide Web Conferences Steering Committee, 2015.

[21] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 990–998. ACM, 2008.

[22] S. White and P. Smyth. Algorithms for estimating relative importance in networks. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 266–275. ACM, 2003.

[23] J. Xie, B. K. Szymanski, and X. Liu. Slpa: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*, pages 344–349. IEEE, 2011.