International Journal of Database Theory and Application
Vol.8, No.3 (2015), pp.227-234
http://dx.doi.org/10.14257/ijdta.2015.8.3.19

Document Similarity Search Algorithm Based On Hierarchy Model

Zhu Ge*

Department of Information Science and Technology, Heilongjiang University, 74 Xue Fu Road, Harbin, China
zhuge@hlju.edu.cn

Abstract

Searching for similar documents from huge amounts of documents is an important and time consuming problem. Although the numerous precise models have been developed for the task, the traditional search algorithms are unable to meet the needs of users for quick search. Herein, a new document similarity calculation and search method with high efficiency is proposed. The calculation of the similarity is based on the total probability model and the efficient search is achieved via level n nodes and paths of citation graph. A special approach from the branch and bound limits the search scope and provide decision algorithm. With the increase in the number of documents, the efficiency of the proposed algorithm is dramatically promoted.

Keywords*: documents; similarity; search; level-n nodes; path*

1. Introduction

Many applications require a measure of "similarity" between objects. One obvious example is the "find-similar-document" query, on traditional text corpora or web [1]. Similarity search is widely used in recommender systems in library or web applications. For example, Google [2] provides related web pages via an advanced search. On the other hand, CiteSeer [3] can offer the users the similar papers with the currently browsed paper. Document similarity search is to find documents similar to a query document in a text corpus and return a ranked list of similar documents to users [4].The traditional algorithm first computes the similarity between each pair of documents, and then ranks them by relevance. This is a very time-consuming process, and needs to do a lot of Preprocessing and later maintenance work. When one faced large-scale document corpus, such a search approach, although with high precision, is unable to meet the quick search. Thus, to improve document similarity search efficiency is a great challenge.

The contents are as follows. Section two describes related work; section three introduces the citation graph, related definitions and calculation method of similarity; section four elaborates the similarity search algorithm; section five is the experiment, and finally gives the conclusion of the paper.

2. Related Work

Currently, The popular text retrieval models mainly include the content analysis, link analysis and a combination between the two.

Content analysis includes several models, such as boolean model [1], probabilistic model [5-7], vector space model [8-9], and Latent semantic analysis [10-11]. The probabilistic model and the vector space model are the most effective models and have been widely used for filtering, clustering and retrieval in IR field. The basic idea in a probabilistic model is to estimate the relevancy between a document and query. The *BM25* [5] model is a representative and successful probabilistic model and the okapi

ISSN: 2005-4270 IJDTA

system adopting this model has achieved high performance in *TREC* experiments. The vector space model creates a space in which documents are represented by vectors and relevance is measured by the similarity between each vector. In this model, the most popular similarity functions are the cosine measure [8].

The similarity between documents also can be calculated by analyzing links between documents. Some works focus on analysis of citation between documents. Most noteworthy from this field are the methods of *co-citation* [12], *co-coupling* [13], *SimRank* [14] and *CiteSeer* [3]. Co-citation means two papers, if often cited together by others, may discuss related topics. The two papers, if cite many papers in common, are considered to focus on the similar topic by co-coupling method. These methods have been used to cluster Web pages [15]. Analogously, *SimRank* method rank the similarity of different documents by the number of same words linked. *CiteSeer* combine both citation and contents to calculate the similarity between two documents and get better performance.

3. Document Similarity Calculation

3.1.Basic Graph Model

We model documents and relationships as an undirected graph $G = (V, E, W)$, where nodes in V represent documents; edges in E represent citation relationships between documents and weights in W represent similarity between documents. Figure 1 is an example of the graph.

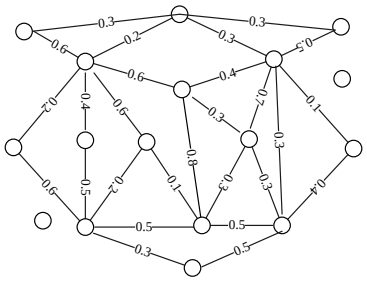

Figure 1. Citation Graph

3.2 Adjacent Nodes Similarity Calculation

In the graph, calculation of adjacent nodes similarity uses the vector space model. v_0 is a point in vector space ((k_1, w_1) , (k_2, w_2) , ... , (k_n, w_n)) ,wherein k_i represents a feature words, and w_i is the weight of the feature word k_i in document v_0. Calculation of w_i use the TF x IDF [1] algorithm. Adjacent nodes similarity calculation method adopts cosine measure and formula is as follows:

$$Sim_{adj}(v_i, v_j) = \cos(v_i, v_j) = \frac{v_i \bullet v_j}{|v_i||v_j|} = \frac{\sum_{k=1}^{n} w_{ik} \bullet w_{jk}}{\sqrt{\sum_{k=1}^{n} w_{ik}^2} \times \sqrt{\sum_{k=1}^{n} w_{jk}^2}} \quad (1)$$

3.1 Non Adjacent Nodes Similarity Calculation

In the graph, there is no direct citation relationship between non adjacent nodes, but if there is a path between them, it is still possible that there is some degree of correlation. When there is only one path between two nodes, due to indirect citation relationship, we use the multiplication rule of multi steps to reach, namely adjacent nodes similarity multiply on the path (see equation 2). But in most cases, there may be several paths between any two non adjacent nodes. The more paths between them exist, of the higher similarity the two nodes may be. In order to consider each path's contribution to similarity, we use the total probability model, and all the paths of the non adjacent nodes constitute similar complete event group. Under the total probability model, calculation of

the similarity between non adjacent nodes is to compute the sum of similarity of all paths (see equation 3). However, number of paths may be numerous and calculating the similarity between two nodes on the aforementioned total probability model is clearly unrealistic. Therefore, this paper introduces the concept of *level-n* nodes restricting the path, at the same time, and the search algorithm is based on *level-n* nodes.

Definition 1 (*level-n* nodes) *level-n* nodes are defined in graph *G*, extending outward from central node by *n* layers. The central node is defined as *level-0* node (namely query node); any nodes which adjoin to *level-0* are *level 1* nodes; if not *level-0* or *level-1* nodes, the nodes which adjoin to *level-1* are *level-2* nodes. More generally, for $n \geq 3$, if no less than level-n, the nodes which adjoin to *level-(n-1)* node are level-n nodes. Figure 2 is an illustration of *level-n* nodes defined on citation graph.

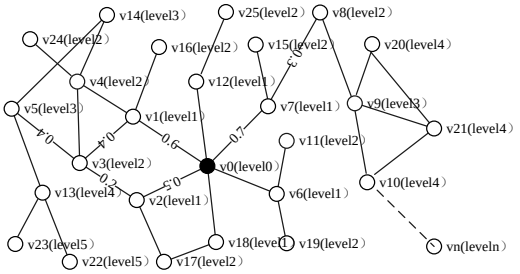

Figure 2. Level-n Nodes Illustrations

Definition 2 (path) For a given citation graph *G*, if there is a sequence of nodes of v_i, v_0, ... , v_n, v_j, such that the (v_i, v_0), (v_0, v_1) , … , (v_n, v_j) belong to the *E*(*G*), v_i, v_0 , … , v_n, v_j belong to different level of nodes and are sorted according to their levels, then we define there is a path between v_i to v_j.

Based on the definition of *level-n* nodes and path, calculation of non adjacent nodes similarity is as follows:

(1) When the number of paths between two non adjacent nodes is one, the calculation formula is as follows:

$$Sim_{disadj\_k}(v_i, v_j) = Sim_{adj}(v_i, v_0) \times Sim_{adj}(v_0, v_1) \times ... \times Sim_{adj}(v_n, v_j) \quad (2)$$

In it, $Sim_{adj}(v_i, v_j)$ represents the similarity between adjacent nodes v_i and v_j; $Sim_{disadj\_k}(v_i, v_j)$ represents the similarity between non adjacent nodes v_i and v_j. A sequence of nodes v_i, v_0, … , v_n, v_j is a path from v_i to v_j.

(2) When the number of paths between two non adjacent nodes is *n* (*n*>1), the calculation is based on the total probability formula as follows:

$$Sim_{disadj}(v_i, v_j) = \sum_{k=1}^{n} Sim_{disadj-k}(v_{i-1}, v_j) \times Sim_{disadj-k}(v_i, v_j) \quad (3)$$

In it, $Sim_{disadj}(v_i, v_j)$ represents the similarity between non adjacent nodes v_i and v_j; $Sim_{disadj\_k}(v_i, v_{j-1})$ represents the conditional similarity from v_i to v_j through path *k*; $Sim_{disadj\_k}(v_i, v_j)$ represents the similarity between non adjacent nodes v_i and v_j on the *k-th* path.

Through the formula 2 and formula 3, we know that computing the similarity between central node and *level-n* nodes can be achieved by using the similarity between central node and *level-(n-1)* nodes. Therefore, we save the results of every step to reduce computational cost.

4. Document Similarity Search Algorithm

4.1 Problem Description

In the citation graph, for a given query node (document), the search problem is how to retrieve the nodes which satisfying certain similarity with query node.

4.2 Search Algorithm

Based on the citation graph and calculation of similarity, this paper puts forward *Next-Level* algorithm for document similarity search. *Next-Level* is a hierarchical search algorithm. Its main idea is: for a certain query node *v* as the central node, the *Next-Level* uses breadth-first strategy scanning citation graph *G* by layers (direction to high-level nodes); scanning termination is determined by setting virtual node and threshold and finally returns a sorted nodes list based on the similarity.

4.2.1. Next-Level Search Algorithm: *Next-Level* search algorithm is described as follows:

Algorithm1：*Next-Level* Search algorithm

Input: query v, citation graph G = (V,E,W),sim_{user}
Output: similar document set C
1: Initialization level=1;
2: if v is isolated
3: C=NULL;
4: else
5: for(i=1; v_i∈ current level ;i++)
6: Compute(similarity(v,v_i));
7: if (Similarity(v,v_i)>sim_{user})
8: C=C+ v_i;
9: flag=Call(*Decide*);
10: If flag==true
11: level= level +1;
12: goto 5;
13: else return;
14: Sort(C);

In it, 2-3 lines judge query node v whether is an isolated node: if *v* is an isolated node, output *NULL*. 4-8 lines compute the similarity between *v* and v_i, while vi belongs to the level. If the similarity is greater than sim_{user} (meet user query conditions), $v_i \in C$. 9-13 lines judge whether to need to scan *G* outward. If true, back to 5 and continue to visit the next level nodes, otherwise, the algorithm ends. 14 line sorts the similar document set *C*.

4.2.2 Decide Decision Algorithm: Judging whether to visit the nodes of next level of *G* is accomplished by setting a virtual node to achieve. According to the formula 3 (calculation of similarity between non adjacent nodes), in the citation graph, the similarity between two non adjacent nodes depends on two conditions. The first one is the number of path between non adjacent nodes and the second one is weights passing through the every edge in path. According to the two conditions, between non-adjacent nodes, the more the paths are, the higher the similarity is, or the greater weight passing through the edges is, the higher the similarity is.

Based on the above two conditions, a virtual node is set up and made to be the same as next level nodes which are to be visited (Figure 3). First, the virtual node connects with all nodes which are currently level visited, and such the virtual node has the most paths between next level nodes and the query node inevitable. Secondly, the weight of edge between virtual node and the current level nodes is set as the max weight of the edge

between current level nodes and next level nodes. Therefore, virtual node must be the highest similarity with the query node in the next level nodes. If the similarity between virtual node and query node do not meet the user needs, the next level nodes are without visit.

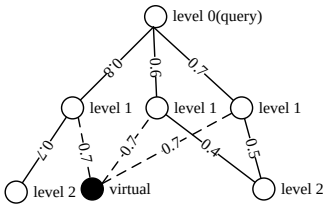

Figure 3. Virtual Node Illustration

Figure 3 is an illustration of setting virtual node, judging whether to visit level 2 nodes. Connecting with all level 1 nodes of *G* makes it a level 2 node. Set up the weights of edge between virtual node and level 1 nodes as the max weight of the edge between level 1 nodes and level 2 nodes, which in this case is 0.7. Obviously, similarity between virtual node and query node is more than any node of level 2. *Decide* decision algorithm is described as follows:

Algorithm2：***Decide*** **algorithm**

Input: query v, citation graph G = (V, E, W), sim_{user}
Output: Whether to visit nodes of next level.
1: Set up a virtual node and connect with all nodes of current level.
2: Initialization simvirtual=1;
3: Compute Similarity(virtual,v);
4: if Similarity(virtual,v)>sim_{user}
5: flag=true;
6: Else
7: flag=false;

In the *Next-Level* search algorithm, if the *Decide* algorithm is always called before visiting nodes of the next level, it will be more complex. In order to improve the search efficiency, we can adopt jumping, such as continuously visiting two level nodes for a judgment.

4.3 Analysis of Search Algorithm Efficiency

Next-Level algorithm cost is mainly consumed in the breadth-first search and computation of the similarity between query node and nodes at all levels. Because the essence of *Decide* algorithm is computing the similarity between nodes. In citation graph, assuming the number of nodes is *n* and the average number of nodes for each level is *i*, the search algorithm calls *Decide* algorithm every two levels. When the search is finished the augmented nodes level is *k*.

For search, the time complexity of visiting a node is $O(1)$ and average time complexity of visiting *k* levels nodes is $O(ki)$. For similarity, calculation between the query node and *k* level nodes depends on the number of paths between *k-1* level and *k* level. Because similarity between query node and *k-1* level nodes has been calculated before computing similarity between query node and *k* level nodes, the calculated results can be used directly. Assuming that the average number of paths of adjacent level nodes is m, equation 3 shows that the time complexity of computing similarity between query node and all nodes is $O(kmi)$. In *Decide* algorithm, set weights of virtual node by comparing with other weights, and its time complexity is $O(ki/2)$. Thus, time complexity of computing similarity between query node and virtual node is $O(ki/2)+ O((k-1)m)$. The whole time complexity of the algorithm is $O(kmi)$. It is because the average number of

nodes for each level is *i*, and the number of the path of adjacent level nodes is at most *i*, that is $m \leq i$. Thus, obviously when m=1, the algorithm takes the minimum value $O(ki)$, when $m=i$, the algorithm takes the maximum value $O(ki^2)$.

5. Experiments

5.1 Experimental Data and Environment

Experimental data come from *ACM* Digital Library, including from 20000 documents published in the international important academic conferences (*SIGMOD, VLDB, KDD, IEEE* etc.) and well-known international academic journals, with time span of 1980-2014 years.

Experimental environment are *Windows2003* Operating System, *Pentium(4)2.4HZ* Processor, *2G* Memory, *VC++6.0* Compiler and *Sql Server 2005* Database.

5.2 Evaluation Measure

We used *P@5, P@10,* average precision and efficiency to measure the performance. The precision at top *N* (*P@N*) results for one query is calculated as follows:

$$P@N = \frac{\text{Relevant} \cap \text{Retrieved}(N)}{\text{Retrieved}(N)} \quad (4)$$

In it, *Retrieved (N)* is the set of top *N* similar documents returned by our system, and Relevant is the set of the documents related to given query document.

The average precision (*MAP*) is average value for all queries, which is calculated as follows:

$$MAP = \frac{\sum_{i=1}^{n} P_i}{n} = \frac{\sum_{i=1}^{n} \frac{\text{Relevant} \cap \text{Retrieved}}{\text{Retrieved}}}{n} \quad (5)$$

In it, P_i is precision of query *i* and *n* is the number of queries submitted.

5.3 Experimental Methods and Results

We design a simple document similarity search system to test the effectiveness of our proposed retrieval model and use different models in our system, including the Okapi's *BM25* model, the Cosine measure, the *SimRank* measure and the proposed model. The parameters of various models use the classic parameter set and *Next-Level* algorithm calls *Decide* algorithm every two levels. During the experiment, 12 test persons with different research directions in the laboratory (information retrieval, graph mining, sensor networks, image processing, pattern recognition and network security) are invited to evaluate results of different query documents, and analysed coincidence degree between results and user needs. The experimental results are shown in Figure 4 and Table 1.

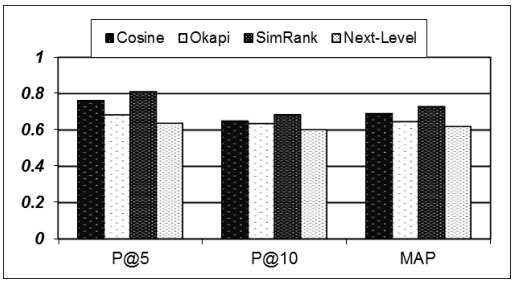

Figure 4. Performance Comparison for Different Retrieval Models

Table 1. Performance Values for Different Retrieval Models

	P@5	*P@10*	*MAP*
Cosine	0.762	0.649	0.69
Okapi	0.683	0.635	0.646
SimRank	0.81	0.684	0.729
Next-Level	0.637	0.602	0.619

Figure 4 and Table 1 give the performances for different models. As can be seen from Figure 4, *SimRank* performs better than the other three models, and the proposed model is slightly lower than the others.

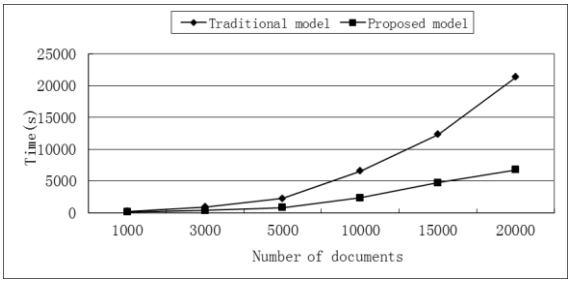

Figure 5. Efficiency Comparison of Four Retrieval Algorithm

Figure 5 shows a time comparison of *Next-Level* and traditional models (the traditional models generally calculate the similarity between documents one by one, and then rank them by relevance). It can be seen from the figure that the fewer the number of documents is, the less gap of search time is. But with the increase in the number of documents, the search time of the *Next-Level* algorithm was dramatically dropped compared to other models. According to the experimental results, although in terms of accuracy the *Next-Level* search algorithm is slightly lower than the other algorithms, the search efficiency is much higher than traditional search models. With the increase of document corpus, *Next-Level* search algorithm is efficient and feasible.

6. Conclusion

In summary, we present a new method of document similarity calculation. Based on the analysis of citation graph, we proposed the concept of the *level-n* nodes and path. From the views of the total probabilistic model and the aforementioned concepts, we proposed the formula for the calculation of the adjacent and non adjacent nodes similarity. In the search process, using a search technology combining breadth-first and pruning strategy reduces the scope of scanned documents and improves the efficiency of search. Theoretical analysis and experimental results show that though the accuracy of the proposed algorithm is slightly inferior to other retrieval models, the search efficiency is far superior to others. At the same time, the *Next-Level* algorithm is also applicable to other fields of similarity search.

Acknowledgements

Natural Science Foundation of Heilongjiang Province of China.

References

[1] R. Baeza- Yates, B. Ribeiro- Neto, "Modern Information Retrieval, Addison-Wesley Professional", **(2011)**; Boston.
[2] S. Brin and L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine", Seventh International World-Wide Web Conference **(1998)**; Brisbane, Australia.

[3] C. Lee Giles, K. D. Bollacker, S. Lawrence, "CiteSeer: An Automatic Citation Indexing system", Proceedings of the third ACM conference on Digital Library **(1998)**; Pittsburgh, America.
[4] X. J. Wan, Y. X. Peng, "A New Retrieval Model Based on TextTiling for Document Similarity Search", J.Comput.Sci.&Technol, vol. 4, no. 20, **(2005)**.
[5] S. Robertson, S. Walker, "Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval", Proc. of the 17th International ACM/SIGIR Conference on Research and Development in Information Retrieval, **(1994)**; New York, America.
[6] W. B. Croft, "Document representation in probabilistic models of information retrieval", Journal of the American Society for Information Science, vol. 6, no. 32, **(1981)**.
[7] L. A. Park, K. Ramamohanarao, "Efficient storage and retrieval of probabilistic latent semantic information for information retrieval", VLDB Journal, vol. 1, no. 18, **(2009)**.
[8] G. Salton, A. Wong, and C.S. Yang, "A vector space model for information retrieval, Communications of the ACM", vol. 11, no. 18, **(1975)**.
[9] Q. L. Guo, "The Similarity Computing of Documents Based on VSM", Proceedings of the 2nd international conference on Network-Based Information Systems, **(2008)**; Turin, Italy.
[10] S. Deerwester, G. W, Dumais, S. T, Furnas, T. K. Landauer and R. Harshman, "Indexing by latent semantic analysis", Journal of the American Society for Information Science, vol. 6, no. 41, **(1990)**.
[11] L.A.F. Park, K. Ramamohanarao, "Kernel latent semantic analysis using an information retrieval based kernel", The 18th ACM Conference on Information and Knowledge Management **(2009)**; Hong Kong.
[12] H. Small, "Co-citation in the scientific literature: A new measure of the relationship between two documents", Journal of the American Society for Information Science, vol. 24, **(1973)**.
[13] M. M. Kessler, "Bibliographic coupling between scientific papers", American Documentation, vol. 14, **(1963)**.
[14] G. Jeh, J. Widom, "*SimRank*: A Measure of Structural-Context Similarity", Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, **(2002)**; Edmonton, Canada.
[15] R. R. Larson, "Bibliometrics of the World-Wide Web: An exploratory analysis of the intellectual structure of cyberspace", Proceedings of the Annual Meeting of the American Society for Information Science**(1996)**; October, Baltimore, Maryland.

Author

Zhu Ge, Current position, grades: the Engineer of Department of Information Science and Technology, Heilongjiang University, China.

Scientific interest: His research interest fields include information retrieval, data mining.

Publications: more than 10 papers published in various journals.

