Hindawi
Security and Communication Networks
Volume 2021, Article ID 1737850, 11 pages
https://doi.org/10.1155/2021/1737850

Research Article

Coauthorship Network Mining for Scholar Communication and Collaboration Path Recommendation

Weiting Zhao,[1] **Zheng Zou**,[1] **Zidong Wei**,[1] **Wenwen Gong**,[2] **Chao Yan**,[1] **and Ashish Kr Luhach**[3]

[1]*School of Computer Science, Qufu Normal University, Jining, China*
[2]*College of Information and Electrical Engineering, China Agricultural University, Beijing, China*
[3]*Department of Electrical and Communications Engineering, The PNG University of Technology, Lae, Papua New Guinea*

Correspondence should be addressed to Ashish Kr Luhach; ashish.kumar@pnguot.ac.pg

Received 20 August 2021; Revised 9 October 2021; Accepted 16 October 2021; Published 5 November 2021

Academic Editor: Hao Wang

With the increasing penetration of interdisciplinary subjects, it is more difficult for researchers to complete a paper individually, showing that the division of labor can improve the level and efficiency of scientific research. Thus, collaboration among multiple scholars has become a trend in academic research. However, because the numbers of scholars and papers are increasing and cooperation between scholars has become more frequent in recent years, it is an increasingly challenging task to discover useful knowledge resources for researchers. Against the background of big data, how to help scholars quickly find interested target collaborators, encourage them to participate more actively in academic communication, and create high-quality achievements in scientific research has become a significant problem. Considering this challenge, this article proposes a framework of coauthorship strength, author contribution, and search (CCS , taking the first letter of the keyword), which is based on the coauthorship feature of Google Academics. In CSS, we combined the search algorithm to select the optimal connection path to help scholars find interested target scholars efficiently and to better solve practical application problems. Finally, our proposal is evaluated by a set of experiments based on a real-world dataset. Experimental results of our approach show better search outcomes compared to other competitive approaches.

1. Introduction

With the development of Internet information technology, academic communication and cooperation are no longer restricted by geographical location. Scholars from different research institutions and different countries can conveniently engage in academic communication. At the same time, with the increasing penetration of interdisciplinary subjects, it is more difficult for researchers to complete a paper individually, and the division of labor can improve the level and efficiency of scientific research. Thus, collaborative research between scholars has become the trend in academic investigation [1]. In academia, coauthors jointly publish papers, a practice that can be regarded as reliably representing scientific cooperation. Generally, papers coauthored by multiple institutions have a higher number of citations than papers published by one research institute [2]. The reason for this is that, through formal or informal personal interactions, researchers share knowledge, exchange ideas, and jointly ensure the accuracy of research results, providing a scientific advantage.

However, considering the example, suppose that scholar A has cooperated with many scholars, excluding scholar B. Scholar A inadvertently reads the scientific research achievements of scholar B and wants to exchange ideas and enter discussions with scholar B. At this time, intermediary scholars can make connections. However, the choice of intermediary scholars has also changed. Generally, the greater the coauthorship intensity is, the closer the communication is, and the easier it is to establish connections with other scholars. Therefore, establishing a coauthor network and selecting proper intermediary scholars to help other researchers connect with each other will be useful and meaningful. However, currently, research faces the two following challenges:

(1) The current scholarly contributions mainly use single indicators, for example, the number of coauthors, which lacks a comprehensive method to calculate coauthorship strength. This may lead to an assessment of the results of the collaboration that are not sufficiently accurate.
(2) Most of the existing studies did not combine related search algorithms for experimental verification, which leads to a lack of application in actual scenarios. This drawback is prone to utilize theoretical methods of calculating coauthor strength that may not be suitable for practical applications.

Considering the above challenges, we propose a coauthorship strength, author contribution, and search framework (CCS) based on the Google Academic Platform. This new approach not only considers multiple coauthorship indicators to comprehensively calculate coauthorship strength but also uses the Dijkstra search algorithm to apply it to the actual coauthorship scene. Therefore, it can use a more comprehensive coauthorship strength calculation method to choose more suitable intermediaries, which can establish connections between scholars and solve the existing research shortcomings.

In summary, our scientific contributions in this study are fourfold:

(1) We obtain the real dataset of the Google Academic Platform using crawler technology to build a coauthorship network.
(2) We take into consideration the number of coauthors, the number of times they have collaborated, the number of citations of the paper, and scholarly contributions of the same paper to ensure an accurate measurement of coauthorship.
(3) We combined a search algorithm to select the optimal connection path between scholars in the form of intermediaries to help scholars find interested target scholars efficiently and to be more effective at solving practical application problems.
(4) A wide range of experiments are enacted according to the real dataset from the Google Academic Platform. Compared with other solutions, the reported experimental results show that our solution has better performance.

The remainder of the paper is organized as follows. The recent literature is investigated in Section 2 to review the current research status in the field. In Section 3, we introduce the coauthorship calculation method and search framework in detail. Evaluations of the experiment are presented in Section 4. In Section 5, we summarize the article and indicate prospective future work.

2. Related Work

In 2001, Newman et al. [3] first studied coauthorship networks. They pointed out that the coauthorship strength between two authors is not constant because the number of authors is inversely proportional to the strength of coauthorship, so a method for calculating the edge weight of coauthorship networks based on the number of coauthors is proposed. However, this method does not take into account the influence of the papers; thus, Hrisch et al. [4] proposed the h index based on the frequency of citations and the number of documents. Scholars from the Chinese Academy of Sciences have performed a more in-depth study of scientific research cooperation [5]. They analyzed four existing weighting models and found that none distinguished the different coauthorship strength between authors based on signature order. Therefore, they suggested introducing factors such as interpersonal relationships, author discipline, and organization affiliation into the calculation of coauthorship strength. Han et al. [6] believed that the cooperation of two authors could be regarded as one author supporting the scientific work of another and proposed a method that uses author cooperation support analysis to calculate the strength of cooperation in a coauthorship network. Empirical research on real large-scale datasets shows that support measures are meaningful. As the data scale continues to grow, the ranking of academic entities is becoming an increasingly compelling task. Therefore, Amjad et al. [7] considered the number of papers, the number of citations, and whether the scholar was the first author; they also proposed a scholarly ranking algorithm based on mutual influence and citation exclusivity. Practice has shown that the proposed method has produced substantial results.

However, existing research on coauthorships only proposes a theoretical method to calculate coauthorship strength, which has not been applied to actual scenarios. According to the Google Academic Platform, scholars can effectively find interested scholars through the existing coauthorship intermediaries capacity, yet there is no complete and effective practical plan. Based on this situation, we propose a coauthorship strength, author contribution, and search framework (CCS) based on the Google Academic Platform. It not only considers multiple coauthorship indicators to comprehensively calculate coauthorship strength but also uses a search algorithm to solve the application problem in the actual coauthorship scene.

3. Motivation

Figure 1 gives a concrete example to illustrate the motivation of this article. Suppose that scholar A has cooperated with many scholars, with the exception of scholar B. Scholar A inadvertently reads the scientific research achievements of scholar B and wants to exchange ideas and engage with scholar B.

In this situation, scholar A is defined as the source scholar, scholar B is defined as the target scholar, and scholar A finds scholar B of interest. We find that there is not just one social relationship between these scholars, meaning that there are many possible contact schemes. For example, scholar A can regard scholar C as an intermediary and connect with scholar B indirectly. Another option for scholar A is through scholar D. Both schemes are

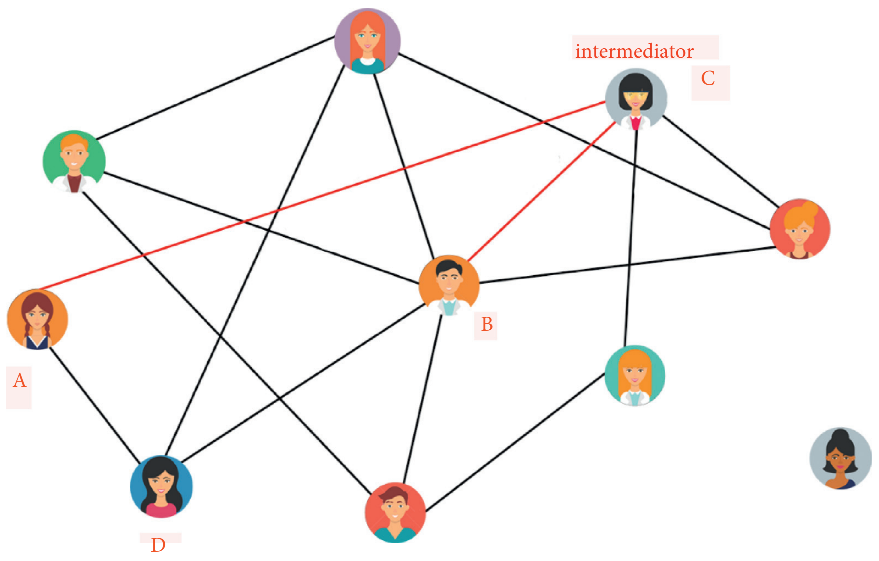

Figure 1: Coauthorship relationship based on the Google Academic Platform.

theoretically feasible, but because the coauthorship strength between scholars is often different, a popular understanding is that two people with a stronger relationship may contact each other more frequently. Choosing scholars with stronger coauthorship relationships as intermediaries will greatly help source scholars get in touch with target scholars and, at the same time, reduce the cost of time and other costs.

To solve this problem, we propose a coauthorship strength, author contribution, and search framework based on the Google Academic Platform, namely, CCS. Our idea is shown in Figure 2, which consists of the four following steps.

Step 1. Crawler technology obtains data. Python was combined with network packet capture and HTML packet capture to obtain real scholarly data on the Google Academic Platform.

Step 2. Construct coauthorship network. The nodes represent scholars, and the edges represent coauthorship.

Step 3. Calculate the coauthorship strength. Based on the calculation of the number of coauthors, the number of times they have cooperated, the number of citations, and the author contributions of different contributors, the coauthorship relationship is comprehensively measured.

Step 4. Use the shortest path algorithm to search. The Dijkstra and the Bellman–Ford algorithms are used to search for intermediaries and establish the connection paths between the source scholar and the target scholar.

4. Coauthorship Strength and Search Framework

4.1. Coauthorship Strength Calculation Model. Scientific research is the purposeful creation of knowledge or the arrangement of knowledge based on existing knowledge. With the increase in scholarly connections, the phenomenon of academic cooperation is becoming increasingly popular. Beaver et al. [8] proposed that the spirit of encouraging cooperation should be materialized, and the contributions of different authors should be distinguished in academic evaluations. Therefore, we establish a coauthorship strength model from different perspectives, such as the number of coauthors, the number of times they have cooperated, the number of citations of the coauthored paper, and scholarly contributions, to measure the coauthorship relationship accurately.

4.1.1. Coauthorship Index Calculation. There are many ways to calculate the edge rights in coauthorship networks. For example, according to the number of times scholars have collaborated, this method assumes that the coauthorship strength between coauthors in the same paper is equal; this may not be the same in practice, however, so this method has limitations. Therefore, Bormer et al. [5] proposed a new method to calculate coauthorship network edge weights, which not only considers the number of coauthors and the number of times they have cooperated but also takes into account the coauthorship effect, which is expressed by the number of citations. The formula is as follows:

$$w_{ij} = \frac{\left(1 + c_p\right)}{n_p\left(n_p - 1\right)}, \tag{1}$$

where n_p represents the number of coauthors of document p, c_p represents the total number of citations of document p, and w_{ij} represents the edge weight between the two nodes of author i and author j.

4.1.2. The Solution of Contribution Degree. Li et al. [9] evaluated the core authors in intelligence research and assessed the authors' contribution rate ranking method, which is mainly used to assign the weight of each author in the coauthored literature in descending order; nevertheless, the author who ranks first plays a leading role. Tang et al.

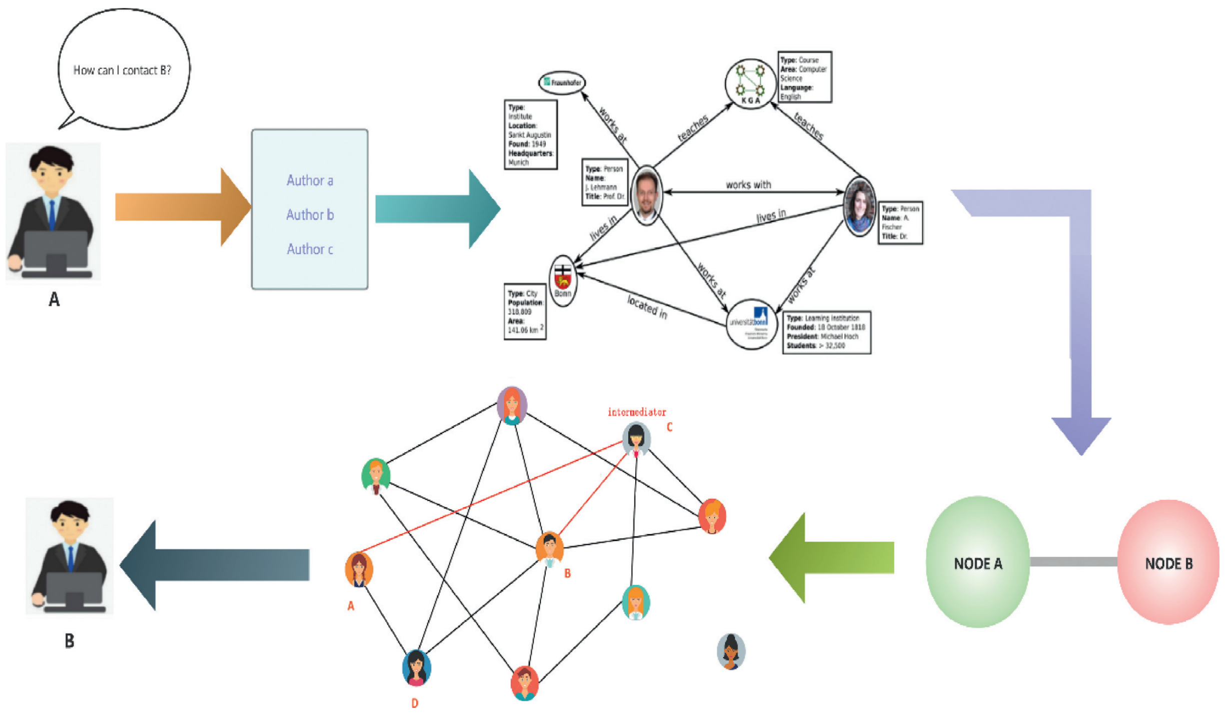

Figure 2: A coauthorship strength, author contribution, and search framework based on the Google Academic Platform.

[10] proposed the author contribution rate grade distribution method that uses an inverse proportional function to express the author contribution value and rank, as in the following formula:

$$W_i = \frac{1}{i\sum_{i=1}^{N} 1/i}, \tag{2}$$

where N represents the number of coauthors and i represents the rank of the authors.

Ling et al. [11] proposed a method for calculating inventor contribution based on the order of patent signatures, combined with the analysis of the topological characteristics of the patent inventor cooperation network, from the influence of inventors qualitatively and quantitatively, to measure personal innovation ability and domain cooperation ability, as in the two following formulas:

$$P_i = \frac{N - i + 1}{N}, \tag{3}$$

$$W_i = \frac{P_i}{\sum_{i=1}^{N} P_i}, \tag{4}$$

where N represents the number of coauthors, i represents the rank of the author, and P_i represents the contribution of the i-th inventor.

Sukhwan et al. [12] proposed a citation-based author contribution measurement method that is independent of the order of the authors in a publication and captures the importance of the first and last authors. They posit that the number of citations by researchers is the degree of recognition in the academic field and an indispensable basis for the measurement of research quality, as in the following formula:

$$w_k = \frac{c_k}{\sum_{k=1}^{N} c_k}, \tag{5}$$

where $k\,i$ is the position of the author in the signature, N is the total number of authors, and c_k is the number of citations of the k-th author.

The authors' contribution calculated by this method is relative to their citations count, not their position in a particular byline. This allows the calculation of signatures that are deliberately sorted alphabetically.

Zuckerman et al. [13] believed that, based on a long-term tradition, there must be a corresponding author at the end of the signature. As the most important author in the authors list, half of the total credits can be given to the last author, as in the following formula:

$$w_k = \begin{cases} 0.5, & k = A, \\ \dfrac{1}{2(A-1)}, & k = 1, \ldots, A-1, \end{cases} \tag{6}$$

where k is the position of the author in the signature and A is the total number of authors.

The Global Nature Index [9] proposed the $1/N$ evaluation model, which assumes that the contribution of each author in the same article is the same. However, the author signature order reflects the different scholarly contributions to a certain extent. Therefore, Du et al. [14] revised the estimation formula as follows:

$$P_{ij} = \begin{cases} 1 - \sum_{i=2}^{N} P_{ij}, \\ \dfrac{1}{\text{Order}_j(i) + (N-1)}, \end{cases} \tag{7}$$

where P_{ij} is the contribution value of author i with $\text{Order}_{(i)}$ to paper j and N represents the total number of authors.

It is generally believed that scholars in different signature orders have made different contributions to the scientific research results. The academic community is accustomed to placing the names of scholars with the greatest contributions first. Therefore, to distinguish the degree of knowledge exchange between scholars in the same coauthored paper, this article also considers adding the authors' contribution to more accurately measure scholarly coauthorship strength. The formula proposed in this article is as follows:

$$W_{ij} = \begin{cases} P_{ij} - \sum_{i=2}^{N} W_{ij}, \\ P_{ij} * \left(\dfrac{1 + c_p}{n_P(n_P - 1)}\right), \end{cases} \tag{8}$$

where n_p represents the number of coauthors of document p, c_p represents the total number of citations of document p, and w_{ij} represents the edge weight between the two nodes of author i and author j. P_{ij} is the contribution value of author i to paper j, and N represents the total number of authors.

4.2. Optimal Path Search Algorithm. The shortest path problem aims to solve how a search can minimize the sum of the weights of the edges. The Dijkstra algorithm, heuristic search algorithm, and Bellman–Ford algorithm commonly use shortest path algorithms [15]. The coauthorship strength, author contribution, and search framework proposed in this article selects scholars with stronger coauthorship strength as the intermediaries, which is more likely to increase the probability that source scholars successfully find target scholars. Since the edge weight of the coauthorship network is defined as the inverse of coauthorship strength, we need to select the edge with the greater coauthorship strength, that is, the smaller the weight that should be selected during each search, in which case the shortest path search algorithm is applicable.

4.2.1. Dijkstra Search. The Dijkstra search algorithm was first proposed by the Dutch computer scientist E. W. Dijkstra [16]. The algorithm searches for the node closest to the starting point each time, determines the length of the path from the starting node to the node, and then checks whether the shortest path length from the vertex to the end node has decreased [17]. In concrete, the pseudocode of this method is specified formally in Algorithm 1.

The Dijkstra algorithm is the most basic and most widely used algorithm for finding the shortest path. When finding the shortest path from a certain node (source point) in the network to the rest of the nodes, the classic Dijkstra algorithm divides the nodes in the network into three parts: unmarked nodes, temporarily marked nodes, and shortest path nodes (permanently marked nodes). At the beginning of the algorithm, the source point is initialized as the shortest path node, and the rest are unmarked nodes. During the execution of the algorithm, each time from the shortest path node to the neighboring node, the neighboring node of the nonshortest path node is modified to a temporarily labeled node, as well as the right to judge. After the value is updated, the node with the smallest weight from all the temporary marked nodes is extracted and then modified as the shortest path node and used as the next expansion source, and then the previous steps are repeated. When all nodes have expanded sources, the algorithm ends. More details can be found in Algorithm 1.

4.2.2. Bellman–Ford Search. The Bellman–Ford algorithm was invented by American mathematicians Chad Bellman and Lester Ford Jr [18]. The algorithm repeatedly judges each edge in the graph by an iterative method so that the estimated value of the shortest path from the starting node to other vertices gradually approximates its shortest distance [17]. The steps of the Bellman–Ford algorithm are as follows:

(1) Initialize the shortest distance from all points to the starting point to infinity and the distance from the starting point to itself to be zero
(2) Traverse each edge in the edge set array E and perform relaxation operations
(3) Check in turn whether the two vertices of each edge in the edge set array E converge

To solve the shortest path problem between two given nodes in the graph, the Dijkstra search algorithm and the Bellman–Ford search algorithm are better solutions. They are widely used in various fields, such as routing algorithms in computer networks, intelligent robot path-finding problems, and navigation of traffic routes [19]. They are also effective in searching for the optimal connection path between the two scholars by coauthorship strength proposed in this article.

In this article, we have made improvements when applying the Dijkstra algorithm and the Bellman–Ford search algorithm. After using the number of coauthors, the number of citations of the paper, and the contributions of scholars to find the coauthoring strength, we take the reciprocal of the coauthoring strength as the weight of the coauthored edges. Considering that, in real life, scholars with greater coauthoring strength are selected as intermediaries, the closer the coauthoring relationship, the higher the success rate of successfully introducing scholars who do not know each other. However, when applying the shortest path length algorithm to search, each time the edge with the smallest weight connected to the current node is selected; therefore, we take the reciprocal of the joint strength as the weight.

```
Inputs:
  s: start of path, e: end of path
  G: undirected weighted graph composed of all nodes
Output:
path: shortest path from node start to node end
dist: shortest path length from node start to node end
    for $v_i \in G - \{s\}$ do
        $\text{dist}[s, v_i] = w(s, v_i)$
          if $\text{dist}[s, v_j] + w_{j,i} < \text{dist}[s, v_i]$ then
          $\text{dist}[s, v_i] = w_j + \text{dist}[s, v_j]$
    if $v_j == e$ then return $\text{dist}[s, e]$, path
```

Algorithm 1: Dijkstra algorithm.

```
Inputs:
s: start of path
e: end of path
G: undirected weighted graph composed of all nodes
Output:
path: shortest path from node start to node end
dist: shortest path length from node start to node end
for $i = 1$ to $|G.V| - 1$
        for each edge$(u, v) \in\in G.E$
        RELAX$(u, v, w)$
for each edge$(u, v) \in\in G.E$
        if $v.d > u.d + w(u, v)$
          return FALSE
return TRUE
```

Algorithm 2: Bellman–Ford algorithm.

5. Experiments

5.1. Experimental Configuration. A dataset from crawler technology is used for feasibility validation purposes. We crawled 5,201 papers and 11,191 authors' data from the Google Academic Platform. The numbers of coauthors and paper citations can be obtained directly from the personal pages and the number of cooperation times can be calculated from the dataset. For experimental comparison, four competition methods were implemented and tested; the running configuration included hardware settings (2.40 GHz CPU, 16 GB RAM) and software settings (Windows 10 and Python 3.7).

The four comparison solutions for comparison are the following:

(1) Reverse [10]: a method that uses an inverse proportional function to express author contributions
(2) Forward [11]: a calculation method based on the forward sequence of signatures.
(3) Cite [12]: a citation-based author contribution method that emphasizes the first and last authors
(4) Last [13]: a method that considers the last author to be the most important author

Evaluation metrics include the following:

(1) The shortest path length: smaller is better for measuring the path length (the sum of weights) when finding the target scholar
(2) Computational memory: smaller is better for measuring the memory usage of the algorithm when finding the target scholar
(3) Computation time: smaller is better for measuring the time consumed of the algorithm when finding the target scholar

5.2. Results Comparison

5.2.1. The Shortest Path Length (SPL). In this test profile, we compare 5 related methods to find SPL from the source scholar to the target scholar. In the experimental setting, the x-axis represents the number of nodes (the number of scholars), and the y-axis represents SPL when using different coauthor strength calculation methods. The experiment's comparison is reported in Figure 3.

Figure 3 shows the CCS that we proposed has the smallest shortest path length compared to the other methods. Since the weight of the edge is the reciprocal of

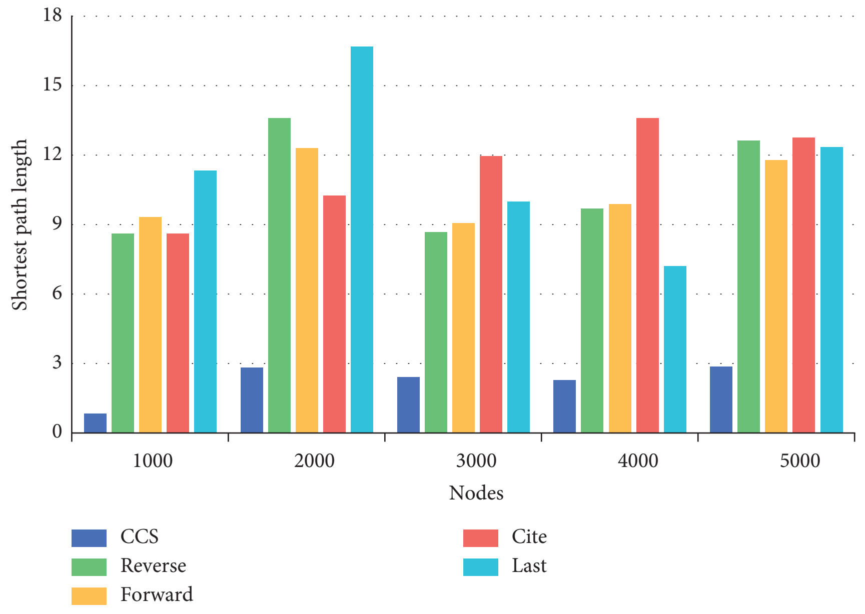

Figure 3: Comparison of the shortest path length with the number of nodes.

coauthorship strength, using the shortest path search algorithm to select a node with a smaller weight actually represents choosing a scholar with a closer coauthorship strength as an intermediary.

As the number of nodes increases from 1,000 to 5,000, SPL of CCS that we proposed is always stable within 3, and the magnitude of the change is small. Meanwhile, the four methods of comparison are basically above 6, especially when the number of nodes is 2000; Last reached 15 during the search. According to this dataset, CCS had the best results on the smallest shortest path length, Reverse and Forward had a similar effect, and Last needed the longest shortest path length.

This means that CCSs are more inclined to look for scholars who have greater coauthorship strength because the greater the coauthorship strength, the smaller the weight and the shorter SPL. Through intermediary scholars who have greater coauthorship intensity, the contact probability of source scholars and target scholars will increase. Therefore, the CCS proposed in this article performs best in terms of the smallest shortest path length.

5.2.2. Computation Memory (CM). In this test profile, we measure and compare the memory usage of five related methods when applying the Dijkstra search algorithm to search for target scholars, where the number of text inputs ranges from 1,000 to 5,000. Because different coauthorship strength calculation methods obtain different weights, there is a gap in memory usage when searching from the source scholar to the target scholar. Concrete comparison results are demonstrated in Figure 4.

Concretely, Reverse, Forward, Cite, and Last do not perform well in terms of memory usage. Reverse and Forward do not consider the number of coauthors or the number of papers cited. Cites are based on citations, ignoring the position of scholars in the order of specific signatures. Last only considers the order of signatures, assuming that the last scholar is the most important. In other words, these four algorithms cannot guarantee a comprehensive measurement of the strength of coauthorship.

In contrast, the CCS has better memory performance than the former four coauthorship strength algorithms. As the number of nodes continues to increase, memory usage is small. When the number of nodes is 1,000, the memory occupies 57 MB, and when the number of nodes is 5,000, the increase is approximately 4919 MB; thus, the effect is better.

5.2.3. Computation Time (CT). In this experiment, we test the efficiency and compare the time taken of five related methods. In the experimental setting, the x-axis represents the number of nodes (the number of scholars), the y-axis represents the time taken when using different coauthorship strength calculation methods, and the number of nodes = {1,000, 2,000, 3,000, 4,000, 5,000}. The consumed computational time of the five algorithms is presented in Figure 5.

Experimental data show that the time costs of the five algorithms all increase with the growth of nodes, as more author data often require additional search time. Furthermore, CCS runs more quickly than the former four coauthorship strength algorithms. Reverse and Forward methods require more computational time when applying the Dijkstra search algorithm to search for target scholars. Among these, Last consumes the most time, which has been validated by the data reported in Figure 5.

It should be noted that when the number of nodes is less than 4,000, the time taken by the five methods increases relatively smoothly. However, when the number of nodes is more than 4,000, the time taken will increase rapidly. CCS occupies the least time close to 30 s when the number of nodes is 1,000, while the most time taken is close to 120 s

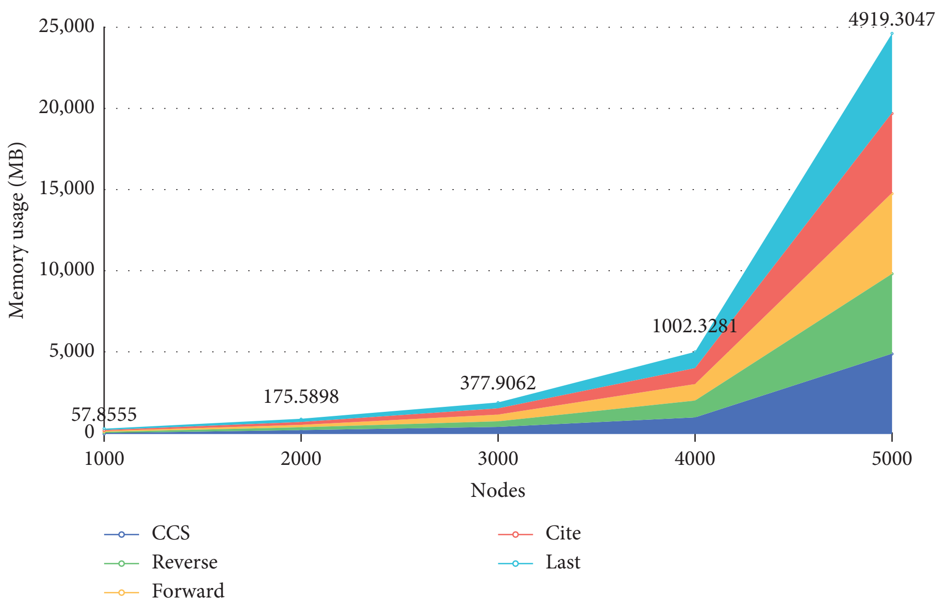

Figure 4: Comparison of the memory usage with the number of nodes.

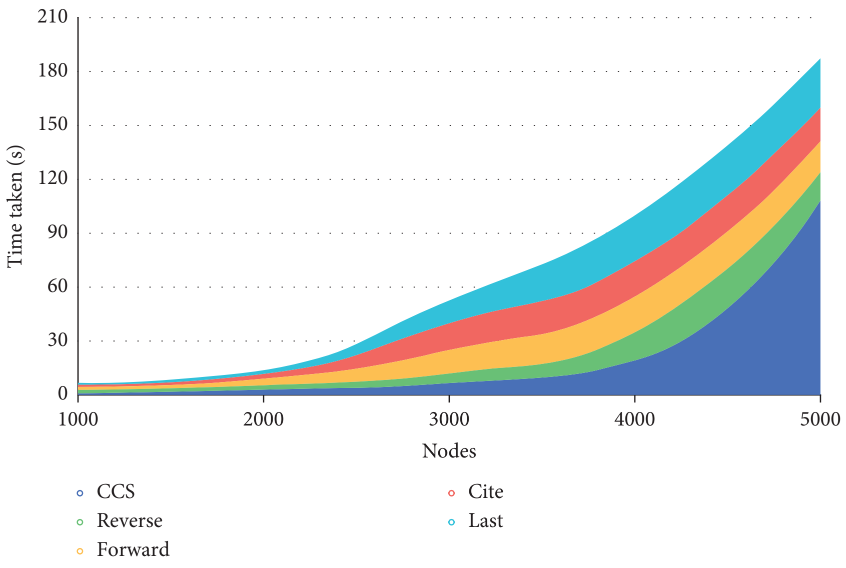

Figure 5: Comparison of the time taken with the number of nodes.

when the number of nodes reaches 5,000. From Figures 4 and 5, a comprehensive conclusion that our CCS has good performance in the areas of memory usage and time taken could be drawn.

In addition, we also verified the feasibility by calculating the number of intermediary scholars needed to find the target scholars shown in Figure 6. It can be seen that, for this dataset, applying the CCS that we have proposed to calculate the coauthorship strength, the number of intermediaries required for searching is one or two more intermediary scholars compared to those required by the other four methods. However, combined with the comprehensive analysis of the shortest path length, time, and memory consumption, the CCS is more inclined to find intermediary scholars with greater coauthor strength, which makes the probability of contacting the target scholar higher.

5.3. Further Discussions. In this test, we also used the Dijkstra algorithm and Bellman–Ford algorithm to conduct search experiments. The experimental comparison is reported in Figure 7.

Figure 7 shows that as the number of nodes in the dataset increases, that is, as the coauthorship network of scholars continues to increase, when using the Dijkstra and Bellman–Ford algorithms to search for intermediary scholars to find target scholars of interest, the required time and memory usage continue to increase, which is in line with

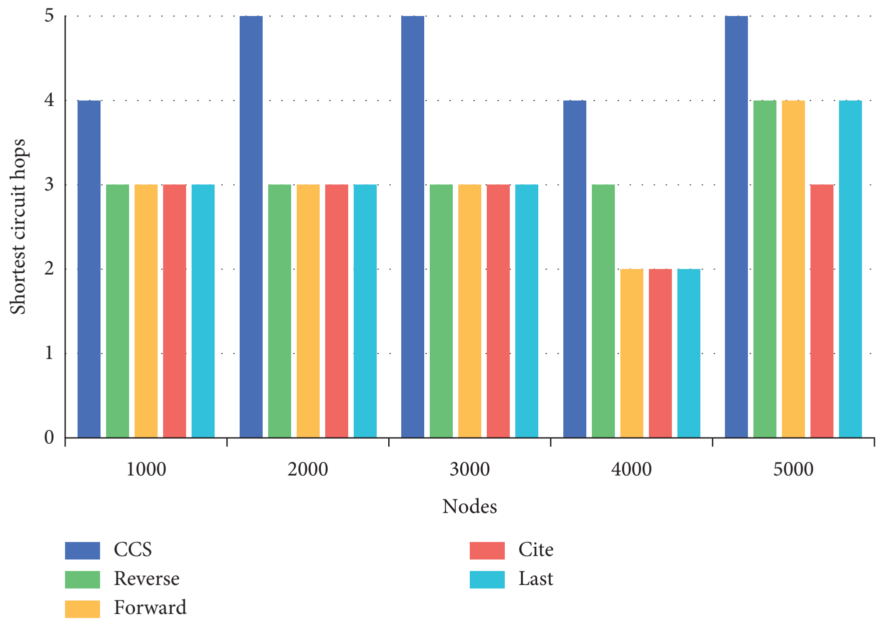

Figure 6: Comparison of the intermediate nodes with the number of nodes.

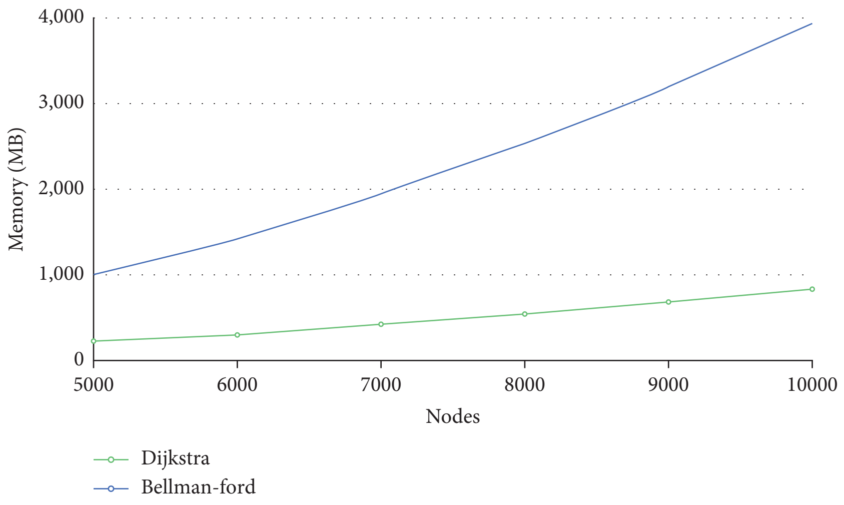

Figure 7: Memory usage of the Dijkstra algorithm and Bellman–Ford algorithm with the number of nodes.

reality. It can also be observed that memory usage of Bellman–Ford algorithm is always higher than that of the Dijkstra algorithm and increases at a very rapid rate; when the number of nodes is 5000, the memory required by the algorithm is 1000 MB, which is much larger than that of the Dijkstra algorithm. Compared with the Dijkstra search algorithm, the Bellman–Ford search algorithm, based on the coauthorship strength calculation method that we have proposed, is more effective in terms of memory usage.

6. Conclusion

With the development of Internet information technology, academic communication and cooperation are no longer restricted by geographical location. Nevertheless, it is an increasingly challenging task to discover useful knowledge resources. How to help scholars quickly find interested target collaborators, encourage them to participate more actively, and create higher-quality achievements has become a significant problem.

Considering this challenge, we propose a coauthorship strength, author contribution, and search framework in this article, based on the Google Academic Platform, which obtains real scholarly data from Google Scholar through crawlers and establishes a scholarly coauthored network. In this way, we take into consideration multiple indicators to ensure accurate measurement of coauthorship. Moreover, we combined it with a search algorithm to better solve practical application problems.

Finally, we validate the advantages of the CCS framework that we proposed through a set of experiments using real-world data from the Google Academic Platform. As a result, we find that the coauthorship model proposed in this article is more likely to choose scholars with stronger coauthorship intermediaries. In practice, intermediary scholars who are more closely connected can improve the probability that source scholars can quickly find target scholars.

In the future, we will introduce more academic indicators, such as user trust [20–22] and time context [23–28]. In addition, computational cost or time cost is a key concern when the dataset to be processed is big [29–39]. Therefore, we will further optimize our proposed method to accommodate the big data applicable scenarios.

Data Availability

The dataset can be accessed at https://www.aminer.cn/data/?nav=openData.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] T. Wan, S. Yue, and W. Liao, "Privacy-preserving incentive mechanism for mobile crowdsensing," *Security and Communication Networks*, vol. 2021, Article ID 4804758, 17 pages, 2021.

[2] H. Liu, J. Guo, J. Li, F. Mei, and H. He, "An A~ algorithm based on random walk," *Journal of Civel Aviation University of China*, vol. 35, no. 6, pp. 61–64, 2017.

[3] D. Zou, "Evolution analysis of scientific research co authorship network in the field of computer science [J/OL]," *Knowledge Management Forum*, vol. 1, no. 2, pp. 130–135, 2016.

[4] R. Xie, X. Li, X. Han, and S. Shi, "Author influence evaluation index construction based on weighted citation frequency and signature order," *Information Science*, vol. 36, no. 8, pp. 90–93+111, 2018.

[5] L. Zhu and J. Yu, "Research on the weighted model of Co-author relationship network," *Library and Information Service*, vol. 54, no. 12, pp. 69–73, 2010.

[6] Y. Han, B. Zhou, J. Pei, and Y Jia, "Understanding importance of collaborations in Co-authorship networks: a supportiveness analysis approach," in *Proceedings of the 2009 SIAM International Conference on Data Mining*, Sparks, Nevada, May 2009.

[7] T. Amjad, A. Daud, D. Che, and A. Akram, "MuICE: mutual influence and citation exclusivity author rank," *Information Processing & Management*, vol. 52, no. 3, pp. 374–386, 2016.

[8] D. B. Beaver and R. Rosen, "Studies in scientific collaboration: Part Il--Professionalization and the natural history of modern scientific co- authorship," *Scientometrics*, no. 1, pp. 231–245, 1979.

[9] L. Li and Z. Zhang, "Research on the impact evaluation method of core authors in information research," *Journal of Information*, vol. 29, no. 10, pp. 80–83, 2010.

[10] Q. Tang and Y. Wang, "Core author evaluation and collaborative network research based on field contribution value," *Information Theory and Practice*, vol. 38, no. 1, pp. 85–89, 2015.

[11] Y. Ling, "Research on patent inventor influence evaluation based on contribution degree and cooperation network analysis," *Journal of Agricultural Library and Information Science*, vol. 30, no. 9, pp. 27–32, 2018.

[12] S. Jung and W. C. Yoon, "Citation-based author contribution measure for byline-independency," in *Proceedings of the 2019 IEEE International Conference on Big Data (Big Data)*, pp. 6086–6088, IEEE, Los Angeles, CA, USA, December 2019.

[13] H. A. Zuckerman, "Patterns of name ordering among authors of scientific papers: a study of social symbolism and its ambiguity," *American Journal of Sociology*, vol. 74, no. 3, pp. 276–291, 1968.

[14] https://www.novopro.cn/articles/2015%2007221217.html1811.

[15] https://wiki.mbalib.com/wiki/Dijkstra%E7%AE%97%E6%B3%95.

[16] S. Idwan and W. Etaiwi, "Dijkstra algorithm heuristic approach for large graph," *Journal of Applied Sciences*, vol. 11, no. 12, pp. 2255–2259, 2011.

[17] W. Han, "Fixed order. An improvement of Bellman-Ford algorithm," *Journal of Harbin Institute of Technology*, vol. 46, no. 11, pp. 58–62, 2014.

[18] Y. Cao and J. Ma, "Optimization of traditional Chinese medicine delivery route based on improved bellman-ford algorithm," *Journal of Hebei North University (Natural Science Edition)*, vol. 36, no. 3, pp. 18–21, 2020.

[19] W. Zhao, Z. Gong, W. Wang, and S. fan, "Comparative analysis of several classical shortest path algorithms," *Journal of Chifeng University (Natural Science Edition)*, vol. 34, no. 12, pp. 47–49, 2018.

[20] F. Wang, H. Zhu, G. Srivastava, S. Li, M. R. Khosravi, and L. Qi, "Robust collaborative filtering recommendation with user-item-trust records," *IEEE Transactions on Computational Social Systems*, pp. 1–11, 2021.

[21] W. Zhang, Z. Li, and X. Chen, "Quality-Aware user recruitment based on federated learning in mobile crowd sensing," *Tsinghua Science and Technology*, vol. 26, no. 6, pp. 869–877, 2021.

[22] H. Kou, H. Liu, Y. Duan et al., "Building trust/distrust relationships on signed social service network through privacy-aware link prediction process," *Applied Soft Computing*, vol. 100, Article ID 106942, 2021.

[23] X. Yang, X. Jia, M. Yuan, and D.-M. Yan, "Real-time facial pose estimation and tracking by coarse-to-fine iterative optimization," *Tsinghua Science and Technology*, vol. 25, no. 5, pp. 690–700, 2020.

[24] L. Qi, R. Wang, C. Hu, S. Li, Q. He, and X. Xu, "Time-aware distributed service recommendation with privacy-preservation," *Information Sciences*, vol. 480, pp. 354–364, 2019.

[25] P. Nitu, J. Coelho, and P. Madiraju, "Improvising personalized travel recommendation system with recency effects," *Big Data Mining and Analytics*, vol. 4, no. 3, pp. 139–154, 2021.

[26] X. Xu, Z. Fang, J. Zhang et al., "Edge content caching with deep spatiotemporal residual network for IoV in smart city," *ACM Transactions on Sensor Networks*, vol. 17, no. 3, pp. 1–33, 2021.

[27] Y. Jin, W. Guo, and Y. Zhang, "A time-aware dynamic service quality prediction approach for services," *Tsinghua Science and Technology*, vol. 25, no. 02, pp. 227–238, 2020.

[28] M. S. Mahmud, J. Z. Huang, S. Salloum, T. Z. Emara, and K. Sadatdiynov, "A survey of data partitioning and sampling methods to support big data analysis," *Big Data Mining and Analytics*, vol. 3, no. 2, pp. 85–101, 2020.

[29] X. Xu, Q. Huang, H. Zhu et al., "Secure service offloading for internet of vehicles in SDN-enabled mobile edge computing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 6, pp. 3720–3729, 2021.

[30] R. Bi, Q. Liu, J. Ren, and G. Tan, "Utility aware offloading for mobile-edge computing," *Tsinghua Science and Technology*, vol. 26, no. 2, pp. 239–250, 2021.

[31] Z. Tong, F. Ye, M. Yan, H. Liu, and S. Basodi, "A survey on algorithms for intelligent computing and smart city applications," *Big Data Mining and Analytics*, vol. 4, no. 3, pp. 155–172, 2021.

[32] X. Xu, Q. Huang, Y. Zhang, S. Li, L. Qi, and W. Dou, "An LSH-based offloading method for IoMT services in integrated cloud-edge environment," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 16, no. 3s, pp. 1–19, 2021.

[33] J. Guo, H. Liang, S. Ai, C. Lu, H. Hua, and J. Cao, "Improved approximate minimum degree ordering method and its application for electrical power network analysis and computation," *Tsinghua Science and Technology*, vol. 26, no. 4, pp. 464–474, 2021.

[34] Y. Bie and Y. Yang, "A multitask multiview neural network for end-to-end aspect-based sentiment analysis," *Big Data Mining and Analytics*, vol. 4, no. 3, pp. 195–207, 2021.

[35] X. Xu, X. Zhang, X. Liu, J. Jiang, L. Qi, and M. Z. A. Bhuiyan, "Adaptive computation offloading with edge for 5G-envisioned internet of connected vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 8, pp. 5213–5222, 2021.

[36] J. Mabrouki, M. Azrour, D. Dhiba, Y. Farhaoui, and S. E. Hajjaji, "IoT-based data logger for weather monitoring using arduino-based wireless sensor networks with remote graphical application and alerts," *Big Data Mining and Analytics*, vol. 4, no. 1, pp. 25–32, 2021.

[37] X. Xu, Q. Huang, X. Yin, M. Abbasi, M. R. Khosravi, and L. Qi, "Intelligent offloading for collaborative smart city services in edge computing," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 7919–7927, 2020.

[38] J. Cai, Z. Huang, L. Liao, J. Luo, and W.-X. Liu, "APPM: adaptive parallel processing mechanism for service function chains," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1540–1555, 2021.

[39] J. Luo, J. Li, L. Jiao, and J. Cai, "On the effective parallelization and near-optimal deployment of service function chains," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 5, pp. 1238–1255, 2021.