Hindawi
Complexity
Volume 2020, Article ID 2085638, 15 pages
https://doi.org/10.1155/2020/2085638

Research Article

Keywords-Driven and Popularity-Aware Paper Recommendation Based on Undirected Paper Citation Graph

Hanwen Liu, Huaizhen Kou, Chao Yan, and Lianyong Qi

Qufu Normal University, Rizhao 276826, China

Correspondence should be addressed to Lianyong Qi; lianyongqi@qfnu.edu.cn

Received 3 November 2019; Revised 12 December 2019; Accepted 7 February 2020; Published 24 April 2020

Academic Editor: Xiaopeng Zhao

Nowadays, scholar recommender systems often recommend academic papers based on users' personalized retrieval demands. Typically, a recommender system analyzes the keywords typed by a user and then returns his or her preferred papers, in an efficient and economic manner. In practice, one paper often contains partial keywords that a user is interested in. Therefore, the recommender system needs to return the user a set of papers that collectively covers all the queried keywords. However, existing recommender systems only use the exact keyword matching technique for recommendation decisions, while neglecting the correlation relationships among different papers. As a consequence, it may output a set of papers from multiple disciplines that are different from the user's real research field. In view of this shortcoming, we propose a keyword-driven and popularity-aware paper recommendation approach based on an undirected paper citation graph, named $PR_{keyword+pop}$. At last, we conduct large-scale experiments on the real-life Hep-Th dataset to further demonstrate the usefulness and feasibility of $PR_{keyword+pop}$. Experimental results prove the advantages of $PR_{keyword+pop}$ in searching for a set of satisfactory papers compared with other competitive approaches.

1. Introduction

With the increasing maturity of recommender systems [1], users are apt to employ existing academic paper recommender websites (e.g., Google Scholar and Baidu Academic) to search for their interested papers based on a set of keywords typed by the users. Generally, an academic paper often contains only partial keywords that a user is interested in. Therefore, a paper recommender system needs to analyze the user's search requirements to return a set of papers that collectively covers all the queried keywords.

Next, we use Figure 1 to introduce the common process of paper recommendation [2]. This process mainly consists of three phases. The first phase is entering keywords; users analyze their research requirements and enter all query keywords (e.g., k_1, k_2, k_3, and k_6) to a recommender system. The second phase is paper discovery [3]; the recommender system automatically identifies diverse sets of candidate papers. The third phase is paper selection [4, 5]; the recommender system recommends candidate papers containing query keywords to users. Frankly, the returned papers may fail to satisfy users' requirements on deep and continuous research on a certain content or topic as these papers may belong to the variety of research domains.

Keyword search methods [6, 7] have long been popularized in searching for papers, but these methods are hardly to find a set of satisfactory papers. In fact, a set of satisfactory papers must satisfy following requirements: on the one hand, these papers are collectively covering users' query keywords [8–10]; on the other hand, one candidate paper containing query keywords has direct or indirect correlation relationships [11] with other candidate papers containing diverse query keywords. In short, recommending a set of satisfactory papers still needs in-depth analyses and study [12].

To recommend a set of satisfactory papers, we propose $PR_{keyword+pop}$ (keywords-driven and popularity-aware paper recommendation) approach that assists users in searching for a set of satisfactory papers, i.e., these papers not only cover all queried keywords but also have higher

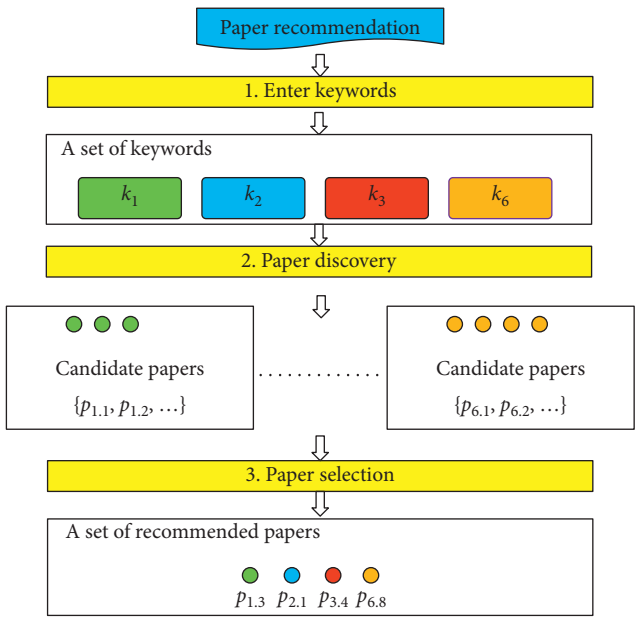


Figure 1: Paper recommendation process.

popularity and correlation among papers. Moreover, $PR_{keyword+pop}$ runs on an undirected paper relationships graph, where paper is modeled as node and a connected edge represents whether there has been correlation relationship among papers. In practice, $PR_{keyword+pop}$ may return one or multiple subgraphs of the paper relationships graph according to users' query keywords; the returned subgraphs include keywords papers covering query keywords, bridging papers (if any) needed however not specified by query keywords, and the composability and popularity of these recommended papers. Note that we speak interchangeably of a paper and its corresponding node in the remainder of this paper, both denoted as p/v.

In summary, we make the following contributions:

(1) We propose a novel keyword-driven and popularity-aware paper recommendation approach, which efficiently recommends a set of satisfactory papers.

(2) We build an undirected paper citation graph [13], and the users' keyword query problem is regarded as the Steiner tree problem. Finally, we employ papers' popularity to find optimal solutions.

(3) We conduct large-scale experiments on the Hep-Th dataset [14] to evaluate the usefulness and feasibility of $PR_{keyword+pop}$.

The rest of this paper is structured as follows: Section 2 demonstrates the research motivation, Section 3 defines an undirected paper citation graph, Section 4 formulates main research problems, Section 5 introduces how $PR_{keyword+pop}$ answers users' keyword query on the undirected paper citation graph, Section 6 evaluates $PR_{keyword+pop}$ by using experimental results, Section 7 reviews related works, and Section 8 further concludes this paper and points out the future research directions.

2. Research Motivation

In Section 2, we use examples of Figures 2 and 3 for demonstrating the research motivation. Figure 2 shows that a user needs to perform the following keywords research tasks [15] before his creation: (1) *paper recommendation* (i.e., k_1) for paper recommendation process research [16]; (2) *keyword search* (i.e., k_2) for keyword search research and applying it to paper recommendation process; (3) *Steiner tree* (i.e., k_3) for Steiner algorithm [17] research and applying it to keyword search; (4) *dynamic programming* (i.e., k_6) for dynamic programming technique research and applying it to solve Steiner tree problem. In Figure 2, the user obtains four corresponding keywords (i.e., $Q = \{k_1, k_2, k_3, k_6\}$) by the preliminary analysis of his research content [18]. Next, the user can search some corresponding papers from Figure 3.

Figure 3 is a part of an undirected paper citation graph and contains 14 nodes covering diverse keywords, i.e., $v_1, \ldots, v_{14}$. Furthermore, the notation $v_{13}\{k_{11}, k_{13}\}$ indicates that node v_{13} offers keywords k_{11} and k_{13}, and the edge $e(v_1, v_{10})$ indicates that nodes v_1 and v_{10} have a correlation relationship. Thus, given a query $Q = \{k_1, k_2, k_3, k_6\}$, the user easily searches a set of papers from Figure 3, i.e., $R_p = \{v_1, v_2, v_3, v_4, v_5, v_6, v_{11}, v_{12}\}$.

Even if this user fortunately obtains a set of papers covering all query keywords, however, it is possible that he is still having no idea of whether these papers can finish his creation as correlation relationships among these papers are both transparent to him. In fact, each user must manually find a set of required papers from massive candidate papers [19, 20]; worse still, this process is very time consuming and challenging. To tackle these issues, we propose a novel keyword-driven and popularity-aware paper recommendation approach, named $PR_{keyword+pop}$, which will be a detailed presentation in Section 5.

3. Undirected Paper Citation Graph

The citation relationships of paper citation graph [21] can sufficiently attest the correlation among papers' research content. If we use the paper citation graph in our proposal, the direction of knowledge information will be considered flowed in one direction. In fact, the knowledge information can be bidirectionally transferred in the paper citation graph. Thus, we use undirected citation relationships to denote the papers' correlations. For example, an undirected citation relationship $\{p_1 — p_2\}$ indicates the correlation among papers p_1 and p_2. As more citation relationships are mined and papers are included in an undirected paper citation graph, this graph will grow larger and denser [22], offering a solid base for recommending a set of satisfactory papers.

$PR_{keyword+pop}$ runs any undirected paper citation graph that fulfills requirements specified by the following definitions:

Definition 1 (nodes). For each paper, an undirected paper citation graph G_p has a corresponding node v. Each node contains multiple keywords (i.e., $k_1, \ldots, k_m$) representing

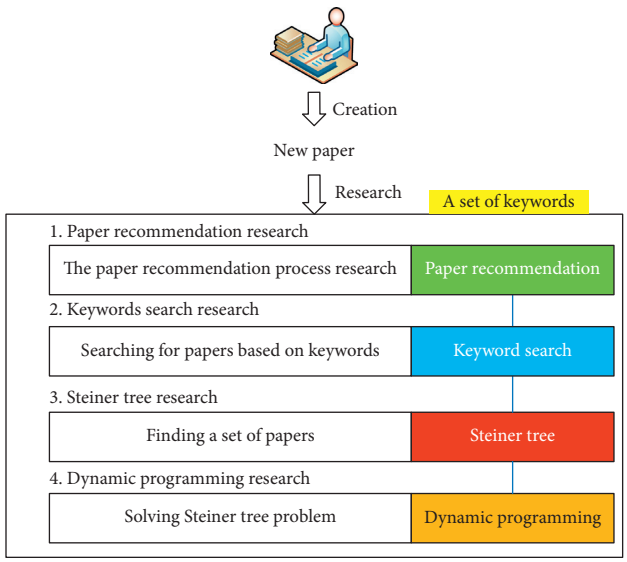


Figure 2: An example of paper's research and creation task.

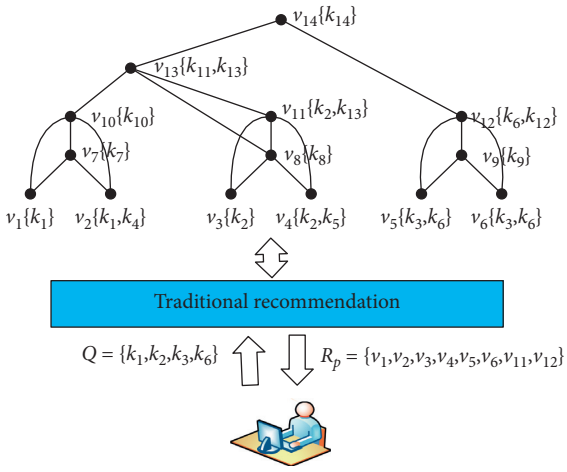


Figure 3: An example of the traditional paper recommendation approach.

the research contents of paper. Furthermore, V_p denotes a set of nodes of G_p.

Definition 2 (edges). For a pair of nodes (v_i, v_j), G_p contains a corresponding edge $e(v_i, v_j)$. $e(v_i, v_j)$ denotes the correlation among nodes v_i and v_j. Furthermore, E_p denotes a set of edges of the G_p.

Definition 3 (undirected paper citation graph). An undirected paper citation graph is expressed as $G_p(V_p, E_p)$, where V_p and E_p denote its sets of nodes and edges, respectively.

According to Definition 2, relevant papers in same domain are connected, either directly or indirectly, forming a connected undirected paper citation graph. Note that if users enter entirely irrelevant query keywords (e.g., privacy-preserving [23–25] and protein engineering), we will fail to recommend a set of satisfactory papers to users.

To answer users' query, $PR_{keyword+pop}$ prebuilds an inverted index $S(K)$ [17] on G_p, i.e., if nodes contain same query keywords, nodes are stored in common inverted index. For example, nodes v_3, v_4, and v_{11} are both containing keyword k_2 in Figure 3, the $S(k_2) = \{v_3, v_4, v_{11}\}$. This way, given an individual keyword k, $PR_{keyword+pop}$ can easily find all papers that perform the research of keyword k.

4. Problem Formulation

In fact, our proposal includes a key point, recommending a set of satisfactory papers. Specifically, answering a keyword query Q mainly consists of two steps: (1) to find Steiner trees based on an undirected paper citation graph G_p, denoted as $T(Q)$, where $T(Q)$ not only covers all query keywords but also has the fewest number of nodes (i.e., the higher correlation); (2) to obtain optimal Steiner trees $T_1(Q)$ based on $T(Q)$, where $T_1(Q)$ has the highest popularity (i.e., the more trust [26]). To better clarify our paper, we summarize the symbols in Table 1.

Likewise, our proposal recommends papers to the user based on the undirected paper citation graph of Figure 3 and the query keywords of Figure 2 (i.e., $\{k_1, k_2, k_3, k_6\}$). Here, nodes v_1 and v_2 contain query keyword k_1; nodes v_3, v_4, and v_{11} contain query keyword k_2; nodes v_5 and v_6 contain query keywords k_3 and k_6; node v_{12} contains query keyword k_6. Thus, given $Q = \{k_1, k_2, k_3, k_6\}$, we are looking for a Steiner tree that connects one node from $\{v_1, v_2\}$, one node from $\{v_3, v_4, v_{11}\}$, one node from $\{v_5, v_6\}$, and one node from $\{v_5, v_6, v_{12}\}$. Furthermore, the Steiner tree also connects nodes that do not cover any query keywords, e.g., nodes v_{10}, v_{13}, and v_{14}. Therefore, the Steiner tree of Figure 4 (i.e., $R_p = \{v_1, v_{10}, v_{13}, v_{11}, v_{14}, v_{12}, v_6\}$) can satisfy users' requirements on deep and continuous research on a certain content or topic.

Thus, Steiner tree is defined as follows.

Definition 4 (Steiner tree). Given an undirected paper citation graph $G_p(V_p, E_p)$ and a set of nodes $V'_p \subseteq V_p$. When T_P covers all nodes of V'_p and it is a connected subgraph, T_P forms a Steiner tree.

Given a query keyword k in $Q = \{k_1, \ldots, k_l\}$, we use the inverted indexes of Section 3 for identifying multiple sets of nodes, denoted as $V_{p1}, \ldots, V_{pl}$, where $V_{pn}(1 \le n \le l)$ at least contains keyword k_n. Next, we need to find a group Steiner tree, and the group Steiner tree is formally defined as follows.

Definition 5 (group Steiner tree). Given the $G_p(V_p, E_p)$ and multiple sets of nodes $V_{p1}, \ldots, V_{pl} \subseteq V_p$, where each group $V_{pn}(1 \le n \le l)$ contains the query keyword k_n. When T_P is a Steiner tree and it contains exactly one node of each group $V_{pn}(1 \le n \le l)$, T_P forms a group Steiner tree.

Firstly, we may obtain multiple group Steiner trees according to the Q. Next, $PR_{keyword+pop}$ aims to find minimum group Steiner trees that not only cover the users' query keywords but also have the higher correlation (i.e., the fewer nodes). Thus, a minimum group Steiner tree is defined as follows.

Table 1: Symbol definitions.

Symbol	Definition
p/v	A paper
$k_1, \ldots, k_m$	A node contains keywords
$e(v_i, v_j)$	An edge of a pair of nodes (v_i, v_j)
Q	Users' query keywords
K	A set of query keywords
V_p	A set of nodes (papers)
E_p	A set of edges
$G_p(V_p, E_p)$	Undirected paper citation graph
$S(K)$	An inverted index
$T(Q)$	Minimum group Steiner trees
$T_1(Q)$	Optimal solutions
Q^1	A queue in ascending order of number of tree nodes
$T_{P\min}(v, K)$	Minimum group Steiner trees rooted at v
R_p	Paper recommendation results

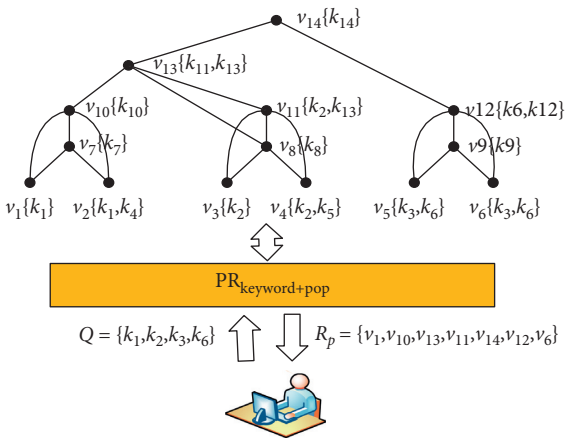


Figure 4: An example of $PR_{keyword+pop}$.

Definition 6 (minimum group Steiner tree). Given a set of exact group Steiner trees, i.e., $T_{P1}, \ldots, T_{Pm}$, when $(|T_{Pi}|) = \min(|T_{P1}|, \ldots, |T_{Pm}|)$, $T_{pi}\,(0 < i \le m)$ is a minimum group Steiner tree. $|T_{Pi}|$ represents the number of nodes (papers) of T_{Pi}.

5. $PR_{keyword+pop}$ Approach

The basic step of $PR_{keyword+pop}$ is as follows (see Figure 5): first, we generate multiple minimum group Steiner trees (i.e., $T(Q)$) by employing the DP (dynamic programming) technique [17]; then, we generate optimal solutions (i.e., $T_1(Q)$) by employing the PP (paper popularity) method.

Step 1. Minimum group Steiner trees generation based on an undirected paper citation graph.

This section mainly discusses employing the DP technique to solve a MGST (minimal group Steiner trees) problem. Specifically, the DP technique firstly breaks up the MGST problem into a series of simpler subproblems; next,

Step 1: Minimum group Steiner trees generation based on an undirected paper citation graph. According to users' query keywords, we generate multiple minimum group Steiner trees by employing the DP (dynamic programming) technique.

Step 2: Optimal solution generation based on minimum group Steiner trees. Based on minimum group Steiner trees, we generate optimal solutions by employing the PP (paper popularity) method.

Figure 5: Concrete process of $PR_{keyword+pop}$.

each of the same subproblems is solved only once and the corresponding results are stored; finally, multiple solutions are effectively provided via combining the stored results, i.e., $T(Q)$.

In this section, we treat all query keywords as K, i.e., $K = Q$. In the DP model, $T_{P\min}(v, K')\,(K' \subseteq K)$ rooted at node v is a state and it contains the users' query keywords K'. Moreover, $w_{DP}(T_{P\min}(v, K'))$ represents the number of nodes in $T_{P\min}(v, K')$. The state-transition equation of the DP model is as follows:

$$w_{DP}\left(\underset{\forall v \in V_k \subseteq V_p}{T_{P\min}}(v, K')\right) = 1, \quad \text{if} \left|\underset{\forall v \in V_k \subseteq V_p}{T_{P\min}}(v, K')\right| = 1, \tag{1}$$

$$\begin{aligned} w_{DP}(T_{P\min}(v, K')) = \min\big(&w_{DP}(T_{Pg}(v, K')),\\ &w_{DP}(T_{Pm}(v, K'))\big), \end{aligned} \tag{2}$$

$$w_{DP}(T_{Pg}(v, K')) = \min_{u \in N(v)} \left\{ w_{DP}\left(T_{P\min}\left(u, K_u^{'}\right)\right) + 1 \right\}, \tag{3}$$

$$\begin{aligned} w_{DP}(T_{Pg}(v, K')) = &\min_{\left(\begin{array}{c} K'_u \cap K' \neq K' \&\& K'_u \cap K' = K'_u \\ \| K'_u \cap K' \neq K'_u \&\& K'_u \cap K' = K' \end{array}\right)} \\ &\cdot \{ w_{DP}(T_{P\min}(u, K'_u)) \\ &+ w_{DP}(T_{P\min}(v, K')) \}, \end{aligned} \tag{4}$$

$$\begin{aligned} w_{DP}(T_{Pm}(v, K')) = &\min_{\left(\begin{array}{c} K'_u \subseteq K \&\& K' \subseteq K \&\& \\ K'_u \cap K' \neq K'_v \&\& K'_u \cap K' \neq K'_u \end{array}\right)} \\ &\cdot \{ | T_{P\min}(v, K') + T_{P\min}(u, K'_u) | \}, \end{aligned} \tag{5}$$

$$\begin{aligned} w_{DP}(T_{Pm}(v, K')) = \min_{K'_1 \cap K'_2 = \Phi} &\{ w_{DP}(T_{P\min}(v, K'_1) \\ &\oplus T_{P\min}(v, K'_2)) \}, \end{aligned} \tag{6}$$

$$T(Q) = T_{P\min}(v, K) = \underset{\forall v \in V_k \subseteq V_p}{T_{P\min}}(v, K'), \quad \text{if } K' = K, \tag{7}$$

where $N(v)$ is a set of v's neighbors in $G_p(V_p, E_p)$, i.e., $u, v \in G_p$, $u \in N(v)$ and $e(u, v) \in E_p$. Formula (1) indicates the weight of a tree is 1 in the DP model owing to only covering one keyword node [27]. Formula (2) indicates that $T_{P\min}(v, K')$ is obtained by using the following two operations: tree growth operation (i.e., formulas (3) and (4)) and tree merging operation (i.e., formulas (5) and (6)). In Figure 6(a), tree growth operation generates new $T_{P\min}(v, K')$ by adding new node u (i.e., one of v's neighbors) to $T_{P\min}(v, K')$. In Figure 6(b), tree merging operation generates new $T_{P\min}(v, K')$ by merging two trees that are both having same root node. The pseudocode of these two operations is specified more formally in Algorithm 1 and Algorithm 2, respectively.

In Step 1, we repeat tree growth operation and tree merging operation to obtain a queue Q^1. The pseudocode of obtaining Q^1 and $T(Q)$ is specified more formally in Steiner tree algorithm (Algorithm 3).

Next, an intuitive example of Figure 7 shows the $T(Q)$ generation processes according to $K = \{k_1, k_2, k_3, k_6\}$. The trees rooted at nodes containing k_1, k_2, k_3, or k_6 are enqueued firstly, i.e., $v_1, v_2, v_3, v_4, v_5, v_6, v_{11}$, and v_{12} are added in Figure 7(b). Since these eight trees only contain one node, the tree growth operation is performed in Figure 7(b). Fortunately, these nodes are both containing neighbor nodes, so trees connecting any one of nodes are generated in Figure 7(c). Next, tree growth operation is performed on Figure 7(c). For example, trees $\{v_8, v_3\}$ and $\{v_8, v_{11}\}$ can generate a new tree rooted at node v_8, i.e., $\{v_8, v_3, v_{11}\}$, but this operation is not tree merging operation as this tree does not contain new query keywords. Furthermore, some new generated trees are deleted as these trees are of no use, e.g., while tree $\{v_8, v_{11}\}$ can generate new tree $\{v_8, v_{11}, v_{13}\}$, the new tree contains not only same query keywords k_2 but also more nodes. Therefore, five required trees are both retained in Figure 7(d). Next, we execute tree merging operations in Figures 7(d) and 7(f) and tree growth operation in Figure 7(e). Finally, the user obtains four minimal group Steiner trees in Figure 7(g).

Note that we consider the output results of Steiner trees algorithm may be entire graph, i.e., $G_p(V_p, E_p)$; furthermore, the worst-case scenario is that our algorithm fails to recommend papers to users.

Step 2. Optimal solutions generation based on the minimum group Steiner trees.

According to the abovementioned algorithm, it is possible to return multiple qualified candidates, e.g., the output result of Figure 7. To ease the heavy burden of users' paper selection decisions, we will select optimal solutions (i.e., $T_1(Q)$) from the output results of Step 1. Generally, a higher citation frequency of papers often means a higher popularity of the papers. Thus, we use the PP (paper popularity) [28] method for selecting $T_1(Q)$ as follows:

$$\mathrm{PP} = \sum_{v_i \in \mathrm{MGST}, v_j \in G_p} d(v_i, v_j), \tag{8}$$

where nodes v_i and v_j belong to $T(Q)$ and G_p, respectively. $d(v_i, v_j) = 1$ if v_j cites v_i in paper citation graph (i.e., $v_i \longrightarrow v_j$) and 0 otherwise.

Finally, we produce a ranking list in descending order according to the popularity of each candidate. Thus, $\mathrm{PR}_{\mathrm{keyword+pop}}$ returns $T_1(Q)$ having the highest popularity among candidates. Note that we consider the recommendation result of $\mathrm{PR}_{\mathrm{keyword+pop}}$ could be $T(Q)$ as all candidates have same popularity.

6. Experiments

To demonstrate the usefulness of $\mathrm{PR}_{\mathrm{keyword+pop}}$, large-scale experiments are designed and tested.

6.1. Experimental Settings. Paper citation graph is extracted from the Hep-Th dataset [14], where the graph covers 8721 papers and each paper contains keyword information.

Generally, an author is allowed creating up to 6 index terms (i.e., keywords) in an article, so we create query keywords with up to 6 in our research. Here, we firstly set a series of experiments, i.e., set A, set B, and set C. In set A, all keywords of a paper are used as a query Q. This scenario emulates that users exactly provide query keywords for their research content. In set B, the query keywords are selected from different papers (in excess of one paper) randomly. This scenario emulates that users randomly provide query keywords. In set C, query keywords are selected from two papers randomly, which further verifies the feasibility of the Steiner trees algorithm. Here, we do not execute the PP method in set C. In addition, each experiment set is repeated 50 times and the average experimental results are adopted.

Currently, we conduct the following experimental evaluation:

(1) Number of nodes: the less amount of recommended papers in a tree, (i.e., the higher correlation of the tree), the better of recommendation approach.

(2) Success rate [17]: the number of recommended papers is smaller than twice the number of query keywords, and the recommendation result is successful.

(3) Average paper popularity (APP) [29]: the APP is defined as follows:

$$\mathrm{APP} = \frac{\sum_{z \in m}^{m} \mathrm{PP}_z}{\sum_{z \in m}^{m} n_z}, \tag{9}$$

where m is the number of $T_1(Q)$ and n_z is the number of nodes in $T_1(Q)$.

(4) Computation time: the consumed time for generating $T_1(Q)$ in sets A and B and $T(Q)$ in set C, respectively.

(5) Precision [30]: precision is calculated as follows:

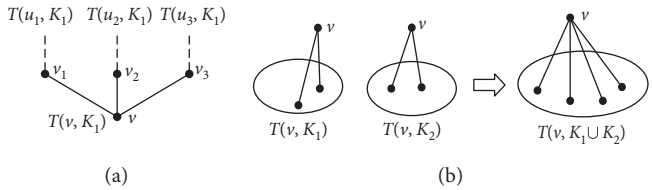


Figure 6: Tree operations. (a) Tree growth operation. (b) Tree merging operation.

Input: $K = \{k_1, k_2, \ldots, k_l\}$
Output: Q^1
(1) **For** each $u \in N(v)$ **do**
(2) **If** $1 + w_{\mathrm{DP}}(T_{P\min}(u, K'_u)) < w_{\mathrm{DP}}(T_{P\min}(v, K'))$
(3) $w_{\mathrm{DP}}(T_{P\min}(v, K')) = 1 + w_{\mathrm{DP}}(T_{P\min}(u, K'_u))$
(4) $T_{P\min}(v, K') = v + T_{P\min}(v, K'_u)$
(5) enqueue $T_{P\min}(v, K')$ into Q^1
(6) update Q^1
(7) **End If**
(8) **If** $(K \cap K \neq K' \&\& K'_u \| K'_u \cap K' \neq K'_u \&\& K'_u \cap K' \neq K')$
(9) $w_{\mathrm{DP}}(T_{P\min}(v, K')) + w_{\mathrm{DP}}(T_{P\min}(u, K'_u)) < w_{\mathrm{DP}}(T_{P\min}(u, K'_u))$
(10) $w_{\mathrm{DP}}(T_{P\min}(v, K')) = w_{\mathrm{DP}}(T_{P\min}(v, K')) + w_{\mathrm{DP}}(T_{P\min}(u, K'_u))$
(11) enqueue $(T_{P\min}(v, K'))$ into Q^1
(12) update Q^1
(13) **End If**
(14) **Return** Q^1
(15) **End For**

Algorithm 1: Tree growth.

Input: $K = \{k_1, k_2, \ldots, k_l\}$
Output: Q^1
(1) **For** each $u \in N(v)$ **do**
(2) **If** $(K'_u \subseteq K \&\& K' \subseteq K \&\& K'_u \cap K' \neq K'_u \&\& K'_u \cup K' \neq K')$
(3) $T_{P\min}(v, K') = T_{P\min}(u, K'_u) + T_{P\min}(v, K'_v)$
(4) enqueue $T_{P\min}(v, K')$ into Q^1
(5) update Q^1
(6) **End If**
(7) **Return** Q^1
(8) **End For**
(9) $K'_1 = K'$
(10) **For** each K'_2 is contained in K s.t $(K'_1 \cap K'_2 = \Phi)$ **do**
(11) **If** $T_{P\min}(v, K'_1) \oplus T_{P\min}(v, K'_2) < T_{P\min}(v, K'_1 \cup K'_2)$
(12) $T_{P\min}(v, K'_1 \cup K'_2) = T_{P\min}(v, K'_1) \oplus T_{P\min}(v, K'_2)$
(13) enqueue $T_{P\min}(v, K'_1 \cup K'_2)$ into Q^1
(14) update Q^1
(15) **End If**
(16) **Return** Q^1
(17) **End For**

Algorithm 2: Tree merging.

Input: $K = \{k_1, k_2, \ldots, k_l\}$
Output: $Q^1$and $T(Q)$
(1) **Let** $Q^1 = \Phi$
(2) **For** each $v \in V_p$ **do**
(3) **If** v contains any nonempty keyword set $K\prime \subseteq K$
(4) enqueue $T_{P\min}(v, K\prime)$ into Q^1
(5) **End If**
(6) **End for**
(7) Min_cou = ∞//the number of nodes
(8) **While** $Q^1 \neq \Phi$ **do**
(9) dequeue Q^1 to $T_{P\min}(v, K\prime)$
(10) **If** $K\prime = K$
(11) **If** $w_{DP}(T_{P\min}(v, K\prime)) <$ Min_cou
(12) Min_cou = $w_{DP}(T_{P\min}(v, K\prime))$
(13) End If
(14) Break
(15) **End If**
(16) **Else** tree growth
(17) **Else** tree merging
(18) **Return** Q^1
(19) **Return** $T(Q)$
(20) **End While**

Algorithm 3: Steiner trees algorithm: MGST (G_p, K).

$$\text{Precision} = \begin{cases} \dfrac{\text{TP}}{T_1(Q)}, & \text{set A and set B,} \\ \dfrac{\text{TP}}{T(Q)}, & \text{set C,} \end{cases} \tag{10}$$

where TP denotes a set of papers containing query keywords.

(6) Recall [30]: recall is calculated as follows:

$$\text{Recall} = \frac{1}{|p|} \sum_{p_a \in p} \frac{T(Q) \cap T_{p_a}}{T_{p_a}}, \quad \text{set C,} \tag{11}$$

where $|p| = 2$ in set C. T_{p_a} is a set of papers cited by p_a.

(7) F1 score: F1 score is calculated as follows:

$$\text{F1-score} = \frac{2 * \text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}}, \quad \text{set C.} \tag{12}$$

To the best of our knowledge, some of approaches address the papers recommendation issue by using papers' relationships. Thus, we compare $PR_{keyword+pop}$ with four approaches that are adapted from [17, 31, 32].

Baseline 1 (Paper-Random [17]): this approach randomly selects a set of nodes that collectively cover all query keywords. Next, the approach finds minimum Spanning trees that interconnect the selected nodes. Finally, we can obtain optimal minimum Spanning trees by executing the PP method.

Baseline 2 (Paper-Greedy [17]): likewise, the approach is randomly selecting a set of nodes that collectively cover all query keywords. Next, the approach regards the selected nodes as initial root nodes and continuously grows trees until these nodes are interconnected. Furthermore, the greedy heuristic algorithm is applied in the tree grow process. Finally, we also use the PP method for obtaining optimal solutions.

Baseline 3 (Random Walk (RW) [32]): RW runs on 2-layer graph, i.e., the undirected paper citation graph and the built paper-keywords graph. In addition, each query only uses users' entered keywords: $q = [0, q_W]$, and this approach only executes the keywords query of set C.

Baseline 4 (Random Walk Restart (RWR) [31]): RWR runs on same 2-layer graph. Furthermore, this approach only executes the query keywords of set C, i.e., $q = [0, q_W]$. Here, if the state vector of RWR has been growing linearly in the experiments, the approach achieves linear convergence.

The experiments are conducted on a machine with Intel(R) Core(R) CPU @3.0 GHz, 16 GB RAM and Windows 10 @ 1809. The software configuration environment: Windows 10 @ 1809 and Python 3.6.

6.2. Experimental Results

6.2.1. Profile 1: The Number of Recommended Nodes of Different Approaches. In this profile, we contrast the number of returned papers of $PR_{keyword+pop}$ with two approaches (i.e., *Paper-Greedy* and *Paper-Random*). As shown in Figure 8, the number of the users' query keywords ranges

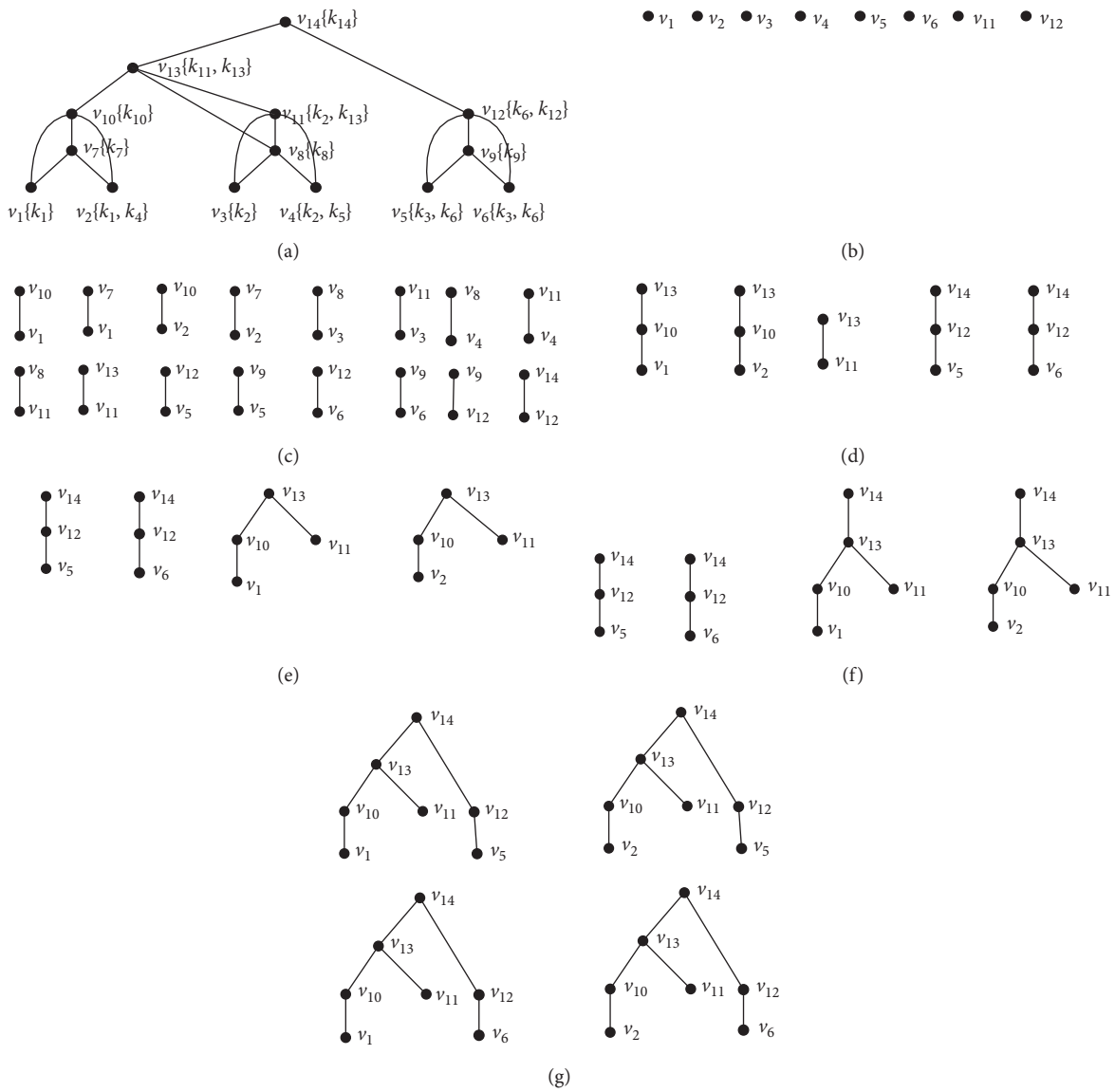


Figure 7: An example of $T(Q)$ generation process.

from 2 to 6. Furthermore, the quantity of recommended papers in our approach increases with the number of query keywords increasing, which is because the returned solutions including more papers can satisfy more query keywords requirements of users. For *Paper-Greedy* and *Paper-Random*, when the number of query keywords equals to 6 in sets A and C, or the number of queried keywords equals to 4 in set B, and they obtain maximum papers quantity. In addition, these experiments results show that our proposal acquires a smaller number of recommended papers than these two approaches. As the smaller number of recommended papers can guarantee higher correlation among papers, $\mathrm{PR}_{\mathrm{keyword+pop}}$ is superior to *Paper-Greedy* and *Paper-Random*.

6.2.2. Profile 2: The Success Rate of Different Approaches. In the profile, we compare the success rates of different approaches. As shown in Figure 9, the experiment results of the different approaches are very different in different experiment sets. Facing to the different experiment scenarios, Figure 9 presents that our proposal can effectively answer the users' keyword query and the success rate is 100%. However, *Paper-Random* and *Paper-Greedy* are difficult to get successful solutions as the number of the users' query keywords increasing; especially, the success rates of these two approaches are both equal to 0 in set B. Again, the experiment results present that our proposal can effectively acquire solutions than *Paper-Greedy* and *Paper-Random*.

6.2.3. Profile 3: The Average Paper Popularity of Different Approaches. In both sets A and B, we compare different approaches by utilizing the average paper popularity. As shown in Figure 10, these experiment figures show that the average paper popularity of *Paper-Random* and *Paper-Greedy* are both larger than $\mathrm{PR}_{\mathrm{keyword+pop}}$. That is because

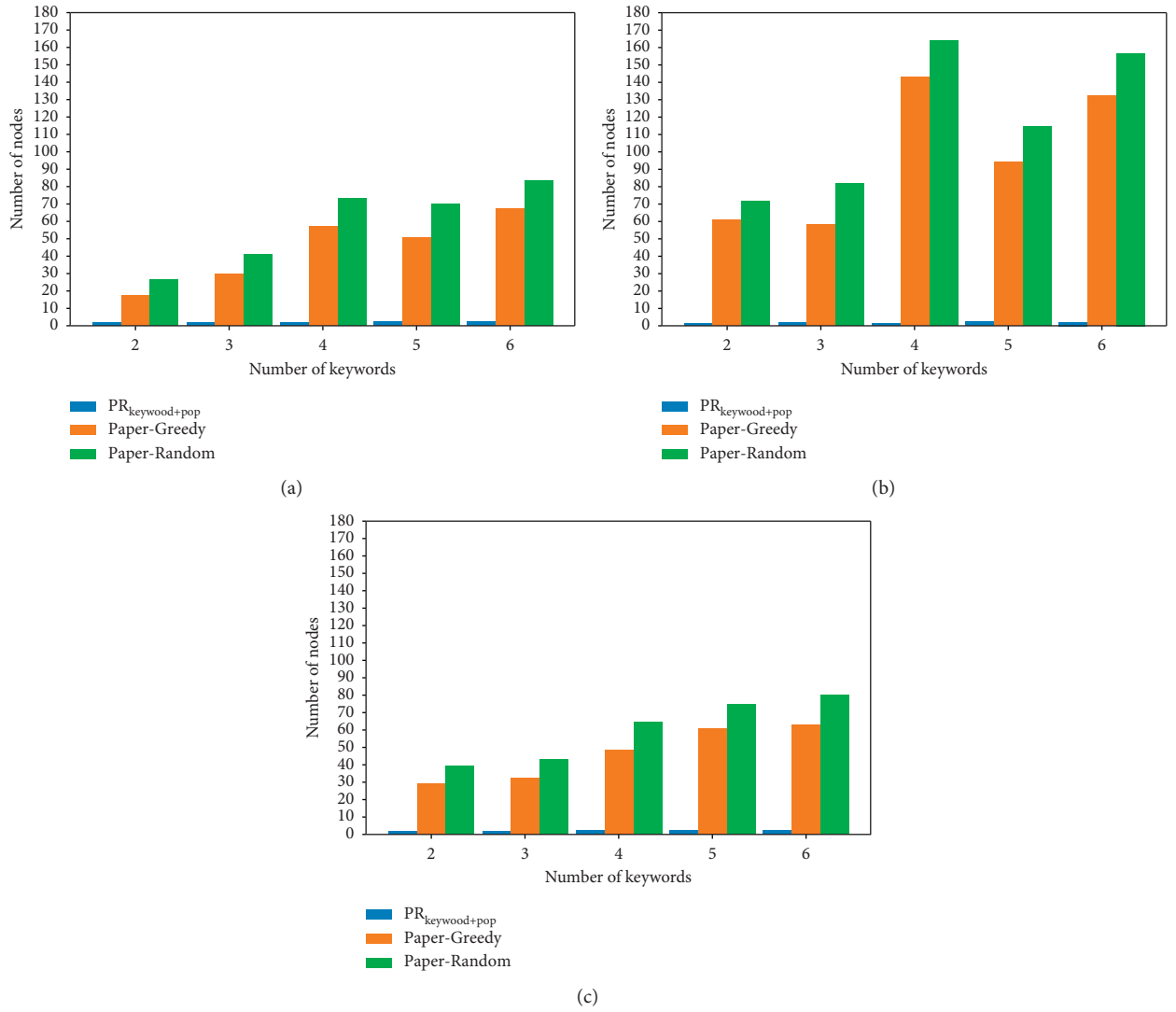


Figure 8: The number of recommended nodes of different approaches: (a) set A, (b) set B, and (c) set C.

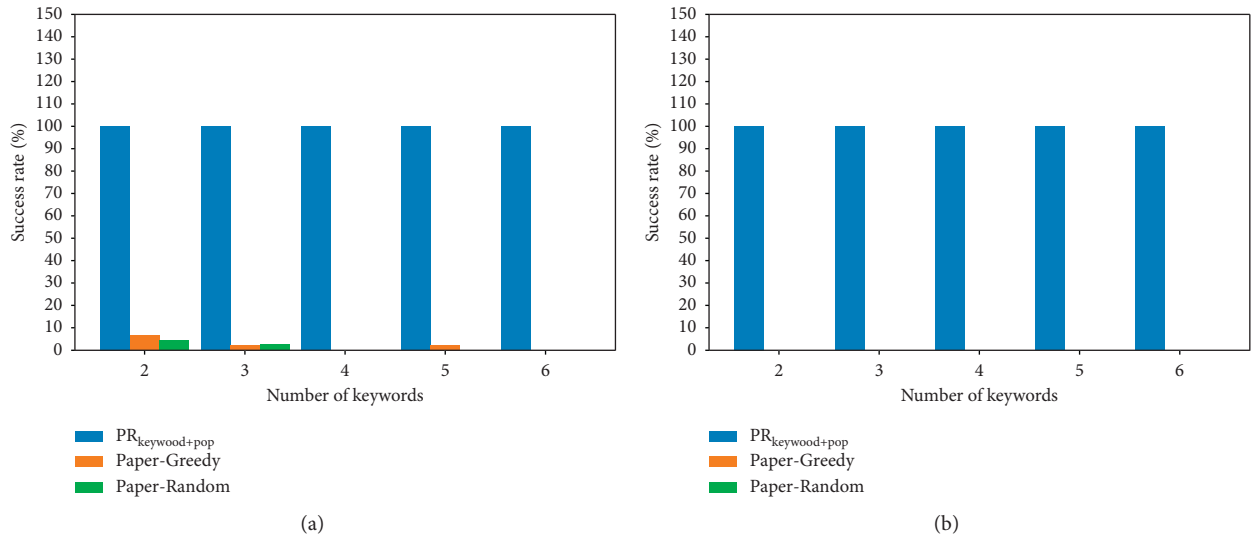
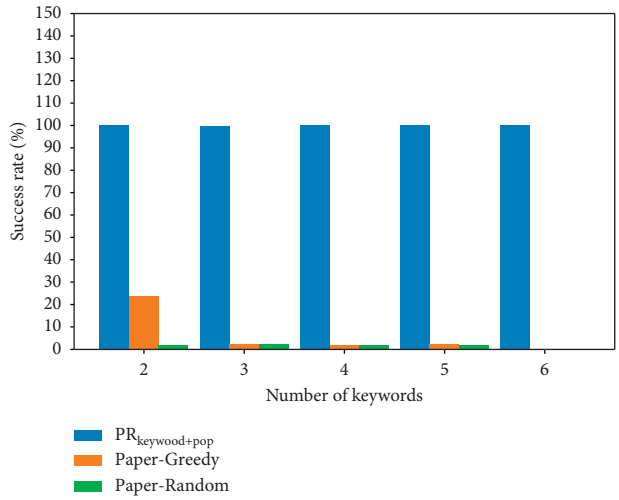
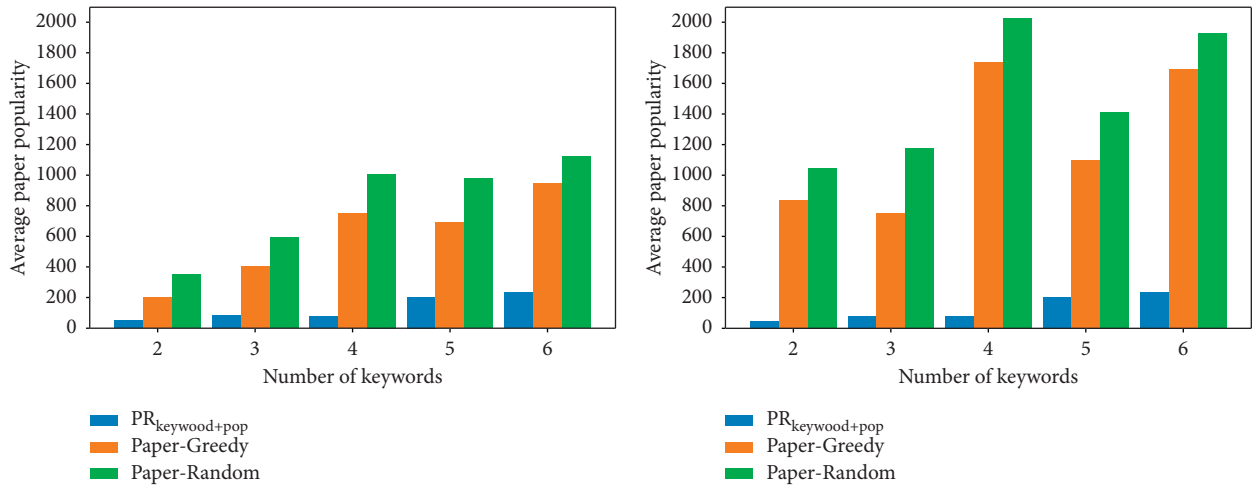


Figure 9: Continued.



(c)

Figure 9: The success rate of different approaches: (a) set A, (b) set B, and (c) set C.



(a) (b)

Figure 10: The average paper popularity of different approaches: (a) set A and (b) set B.

the number of recommended papers obtained by *Paper-Random* and *Paper-Greedy* are both in excess of our approach; moreover, each recommended paper is cited more than once. In practice, the solutions of *Paper-Random* and *Paper-Greedy* will be seldom selected as these solutions take users a serious amount of time and energy to do some unnecessary research studies. In my opinion, Figure 10 presents that the average paper popularity of $PR_{keyword+pop}$ is allowable and receivable in the case of satisfying users' query keywords requirements.

6.2.4. Profile 4: The Computation Time of Different Approaches. In Profile 4, we contrast the time consumption of different recommendation approaches. As shown in Figure 11, $PR_{keyword+pop}$, *Paper-Random*, and *Paper-Greedy* spend more time getting solutions with the number of the users' query keywords increasing. Furthermore, we only calculate the time of RW and RWR in set C, and their time is a constant value. As *Paper-Random* and *Paper-Greedy* use extremely simple heuristic for selecting papers, these two approaches spend fewer time than $PR_{keyword+pop}$ in most cases. In addition, RW and RWR are both spending a lot of time than our proposal as these two approaches need to do a significant amount of iterative operations and matrix operations in experiments. While our proposal takes time to obtain solutions, the time consumption of $PR_{keyword+pop}$ is allowable and receivable in most really cases for users. That is because this is the price to pay if users take fewer time and energy to effectively achieve their research goal.

6.2.5. Profile 5: The Precision of Different Approaches. As shown in Figure 12, these three figures present that the

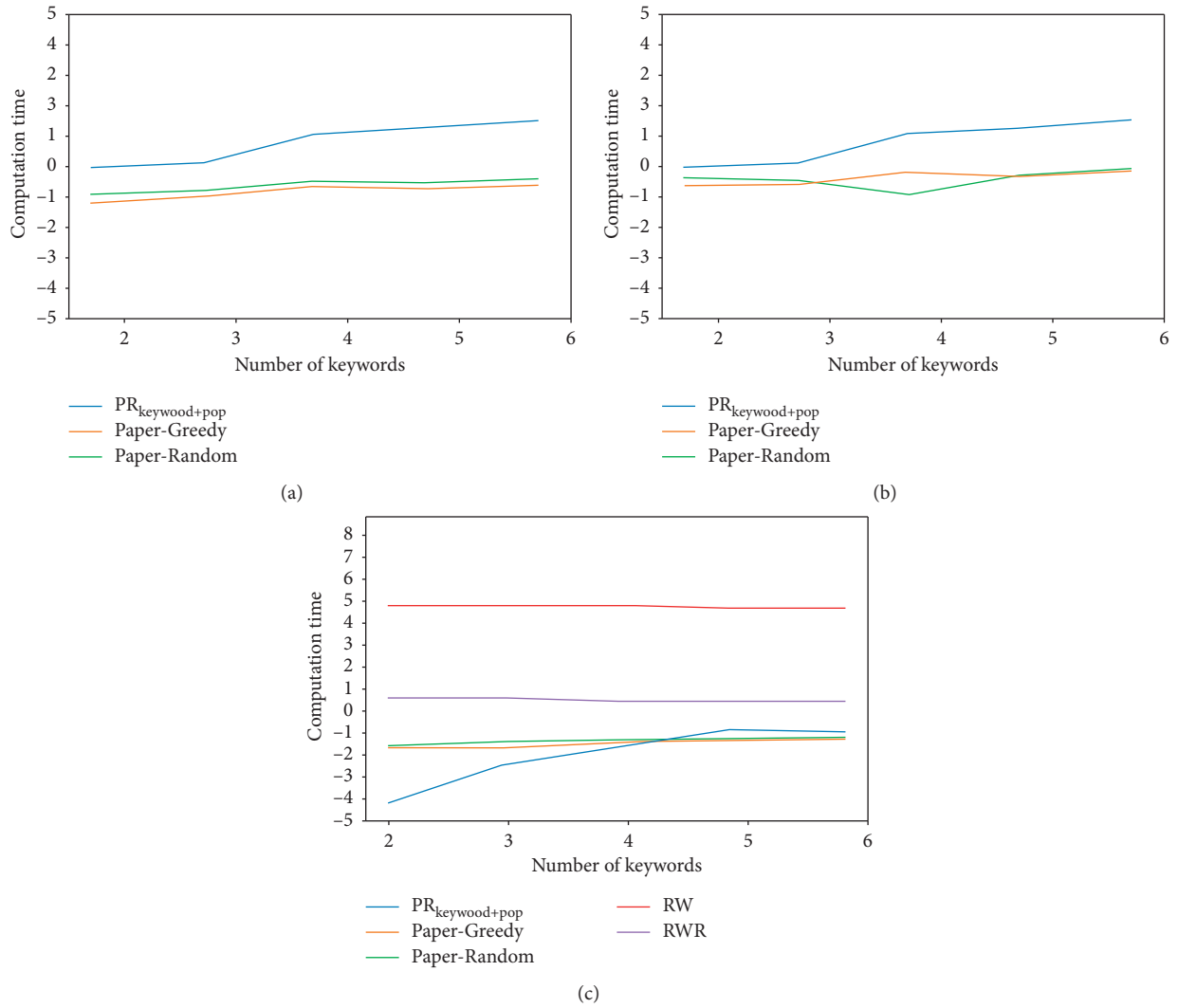


Figure 11: The computation time of different approaches: (a) set A, (b) set B, and (c) set C.

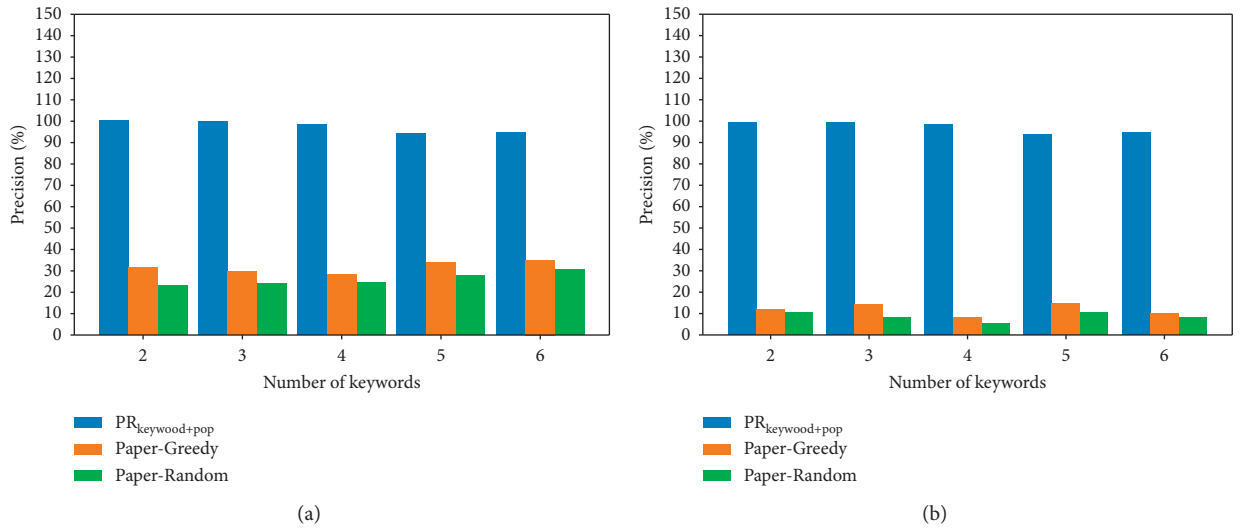
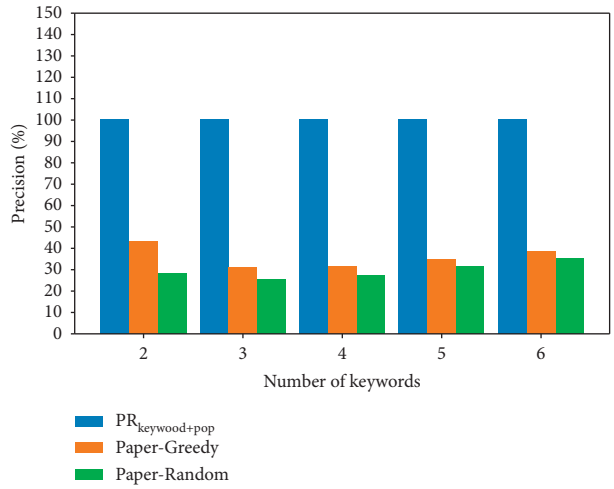


Figure 12: Continued.



(c)

Figure 12: The precision of different approaches: (a) set A, (b) Tset B, and (c) set C.

precision of different approaches, respectively. Luckily, our proposal can accurately answer the users' keyword query, and the precision of three different experiment sets are both 100%.

For *Paper-Random* and *Paper-Greedy*, their precision ranges from 10% to 45%. Therefore, whether users can accurately or randomly offer query keywords, our approach can accurately answer the users' keyword query, and the recommended results better satisfy users' query requirements. Furthermore, these experiment results show further that users may spend fewer time and energy on realizing their research aim.

6.2.6. Profile 6: The Recall Rate and F1 Score of Different Approaches. In this profile, we firstly contrast the recall rate of different approaches in set C. According to Figure 8, the number of recommended papers of our approach is not exceeding 30, so the number of recommended papers of RW and RWR are 10, 20, and 30, respectively; the recall rate of RW and RWR take the average value among the three. In Figure 13(a), the recall rate of $PR_{keyword+pop}$ ranges from 4% to 21%; the recall rate of *Paper-Random* and *Paper-Greedy* range from 39% to 54%; and the recall rate of RW and RWR are less than 9.5%. In addition, we also compare the F1 score of our approach with *Paper-Random* and *Paper-Greedy*. In Figure 13(b), the F1 score of our proposal ranges from 9% to 34% and the F1 score of *Paper-Random* and *Paper-Greedy* range from 33% to 44%. As the number of returned papers of *Paper-Random* and *Paper-Greedy* are in excess of $PR_{keyword+pop}$, the recall rate and the F1 score of our proposal are less than these two approaches. Furthermore, when the number of query keywords is not equal to 3, the recall rate of RW and RWR are both less than our approach. In conclusion, the recall rate and F1 score of Figure 13 can directly verify the feasibility of our proposal.

7. Related Work

Currently, recommender techniques play vital roles in many research areas. Furthermore, recommendation methods can be mainly classified into three categories: collaborative filtering (CF), content-based filtering (CBF), and graph-based approaches.

7.1. Collaborative Filtering. The early work on paper recommendation mainly explored the use of collaborative filtering (CF) techniques. For example, McNee et al. [33] mainly focused on the rating matrices in paper citation networks. In addition, Pennock [34] proposed a personality diagnosis method based on a Bayesian network as their considered rating frequency of other users made a difference to user's ratings of items. Furthermore, McNee et al. [33] combined CF method with the cited frequency of papers to recommend papers, which was because they [33] considered that the number of citations of a paper had a vital effect on papers' ratings. In addition, if there were interactions between users and items in implicit collaborative filtering, it was recorded as 1, otherwise 0. However, 1 or 0 did not indicate positive or negative factors between users and items that generated the interaction [35]. According to users' query keywords, CF approaches can effectively recommend papers to users, but these approaches are generally limited by some problems, e.g., the cold start problem and the data sparsity problem [36].

7.2. Content-Based Filtering. To further ameliorate paper recommendation approach, some researchers further explored content-based filtering (CBF) approaches. Generally, CBF [37] approaches attempted to retrieve papers with respect to textual content and it was not using rating relationships. For example, Alzoghbi et al. [38] examined the preferences among papers by their proposed two different

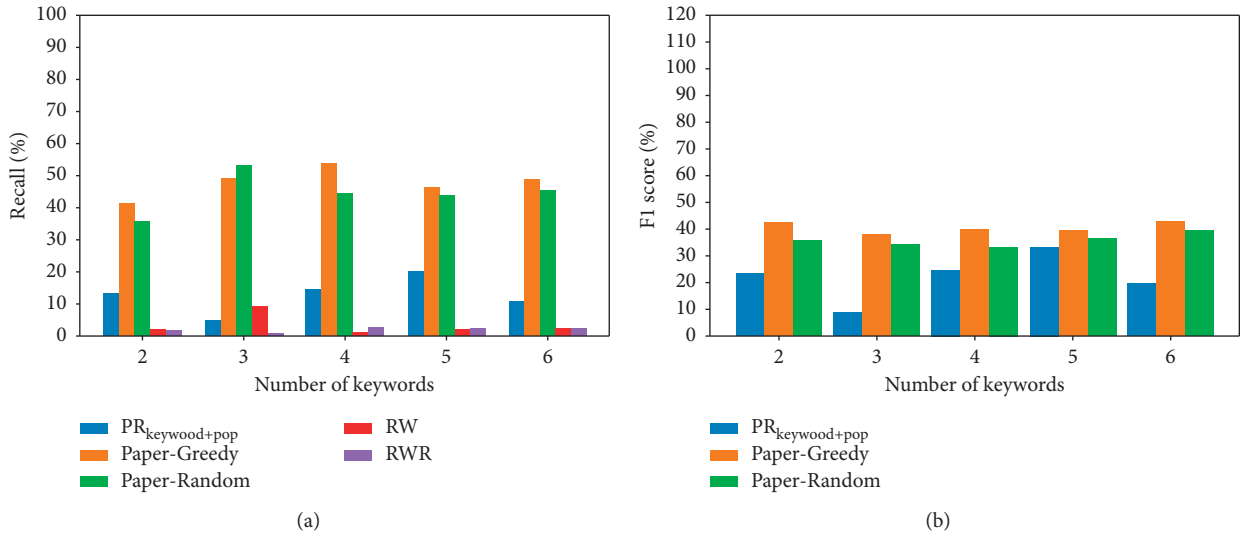


Figure 13: The recall rate and F1-score values of different approaches. (a) The recall in set C. (b) The F1 score in set C.

validation mechanisms and recommended interesting papers to users. Furthermore, Wang and Blei [39] combined a topic model with the collaborating filter to propose paper recommendation approach, named CTR. The CTR firstly used LDA to find latent topics for papers, and this approach inferred user-item relations by using matrix factorization. In fact, the CBF approach suffers from traditional information retrieval issues, e.g., the semantic ambiguity problem. Furthermore, gathering and dealing with the relevance information of papers is often time consuming.

7.3. Graph Model. Currently, the papers' relationships can further reflect the future research trends of paper recommendation, which is mainly because the correlation relationships among papers can indicate the correlation of papers' research contents. For example, Meng et al. [31] regarded authors, papers, topics, and keywords as nodes and their relationships as edges, and the approach recommended academic papers by executing the random walk on a four-layer heterogeneous graph. Furthermore, Gori and Pucci [32] proposed the graph-based PageRank-like recommendation approach that performed the biased random walk on paper citation graphs, and the approach further emphasized on the correlations among citations. In addition, Wu and Sun [40] thought that three different types of paper citation networks could be constructed based on papers' citation relationships, i.e., directly connected network, coupling network, and cocitation network. Liang et al. [41] have proved that the cocitation relationships of cocitation network could be employed in paper recommendation, e.g., if two papers were both cited by more same papers, these two papers had high relevance and were highly likely to be recommended simultaneously.

In fact, the correlation relationships [42] could be formed in a paper citation graph as most papers selected their references based on the content similarity. Thus, we use the paper citation graph for establishing an undirected paper citation graph. On the undirected paper citation graph, our proposal (i.e., $PR_{keyword+pop}$) efficiently recommends a set of satisfactory papers to users. Finally, extensive experiments results validate the usefulness and feasibility of $PR_{keyword+pop}$ approach.

8. Conclusions and Future Work

Whether a set of satisfactory papers will be recommended to users is very important paper discovery and paper selection tasks, which is known as paper recommendation problem. Here, we propose a novel keywords-driven and popularity-aware approach (i.e., $PR_{keyword+pop}$) to return a set of satisfactory papers, i.e., these papers not only collectively cover users' query keywords but also have higher correlation and popularity among papers. Furthermore, these recommendation results support users in doing deep and continuous research on a certain topic or domain. In addition, the experiment results further show the usefulness and feasibility of our proposal.

Although our work shows desirable results, there are still some aspects worth further research and improvement. Since users cannot analyze research requirements in detail, e.g., the required data types [43–45], the recommended results may fail to return satisfactory results. Furthermore, we may face the sparsity problem of the existing paper citation graph. Hence, the abovementioned research contents are to further study and progress.

Data Availability

The experiment dataset Hep-Th used to support the findings of this study has been deposited in "http://snap.stanford.edu/data/cit-HepTh.html."

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China (Grant no. 2017YFB1001800), National Natural Science Foundation of China (Grant no. 61872219), and Natural Science Foundation of Shandong Province (Grant no. ZR2019MF001).

References

[1] L. Qi, X. Zhang, W. Dou, and Q. Ni, "A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2616–2624, 2017.

[2] S. Jieun and B. K. Seoung, "Academic paper recommender system using multilevel simultaneous citation networks," *Decision Support Systems*, vol. 105, pp. 24–33, 2018.

[3] Y. Wang, Z. Cai, Z.-H. Zhan, Y.-J. Gong, and X. Tong, "An optimization and auction-based incentive mechanism to maximize social welfare for mobile crowdsourcing," *IEEE Transactions on Computational Social Systems*, vol. 6, no. 3, pp. 414–429, 2019.

[4] L. Qi, X. Zhang, W. Dou, C. Hu, C. Yang, and J. Chen, "A two-stage locality-sensitive hashing based approach for privacy-preserving mobile service recommendation in cross-platform edge environment," *Future Generation Computer Systems*, vol. 88, pp. 636–643, 2018.

[5] Y. Lin, X. Wang, F. Hao et al., "Dynamic control of fraud information spreading in mobile social networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–14, 2019, In press.

[6] X. Zhou, B. Wu, and Q. Jin, "Analysis of user network and correlation for community discovery based on topic-aware similarity and behavioral influence," *IEEE Transactions on Human-Machine Systems*, vol. 48, no. 6, pp. 559–571, 2018.

[7] D. Wu, H. Wang, and R. Seidu, "Smart data driven quality prediction for urban water source management," *Future Generation Computer Systems*, vol. 107, pp. 418–432, 2020.

[8] R. K. Singh, G. Srivastava, and A. Sharma, "Revisiting the purpose of selling: toward a model of responsible selling," *Journal of Nonprofit & Public Sector Marketing*, vol. 31, no. 2, pp. 184–200, 2019.

[9] G. Srivastava, "Impact of CSR on company's reputation and brand image," *Global Journal of Enterprise Information System*, vol. 11, no. 1, pp. 8–13, 2019.

[10] H. Huang, H. Yin, G. Min, J. Zhang, Y. Wu, and X. Zhang, "Energy-aware dual-path geographic routing to bypass routing holes in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 17, no. 6, pp. 1339–1352, 2018.

[11] H. Liu, H. Kou, C. Yan, and L. Qi, "Link prediction in paper citation network to construct paper correlation graph," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, p. 233, 2019.

[12] C. Huang, G. Min, Y. Wu, Y. Ying, K. Pei, and Z. Xiang, "Time series anomaly detection for trustworthy services in cloud computing systems," *IEEE Transactions on Big Data*, p. 1, 2017, In press.

[13] X. Wang, L. T. Yang, H. Li, M. Lin, J. Han, and H. Li, "NQA," *ACM Transactions on Embedded Computing Systems*, vol. 18, no. 4, pp. 1–21, 2019.

[14] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over time: densification laws, shrinking diameters and possible explanations," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, Chicago, IL, USA, 2005.

[15] B. Zhao, Y. Wang, Y. Li, Y. Gao, and X. Tong, "Task allocation model based on worker friend relationship for mobile crowdsourcing," *Sensors*, vol. 19, no. 4, p. 921, 2019.

[16] X. Wang, L. T. Yang, Y. Wang, X. Liu, Q. Zhang, and M. J. Deen, "A distributed tensor-train decomposition method for cyber-physical-social services," *ACM Transactions on Cyber-Physical Systems*, vol. 3, no. 4, pp. 1–15, 2019.

[17] L. Qi, Q. He, F. Chen et al., "Finding all you need: web APIs recommendation in web of things through keywords search," *IEEE Transactions on Computational Social Systems*, vol. 6, no. 5, pp. 1063–1072, 2019.

[18] X. Zhou, W. Liang, K. I.-K. Wang, R. Huang, and Q. Jin, "Academic influence aware and multidimensional network analysis for research collaboration navigation based on scholarly big data," *IEEE Transactions on Emerging Topics in Computing*, p. 1, 2018.

[19] W. Gong, L. Qi, and Y. Xu, "Privacy-aware multidimensional mobile service quality prediction and recommendation in distributed fog environment," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 3075849, 8 pages, 2018.

[20] L. Qi, Y. Chen, Y. Yuan, S. Fu, X. Zhang, and X. Xu, "A QoS-aware virtual machine scheduling method for energy conservation in cloud-based cyber-physical systems," *World Wide Web*, vol. 23, no. 2, pp. 1275–1297, 2019.

[21] Z. Zhou, S. Chen, M. Hu, and Y. Liu, "Visual ranking of academic influence via paper citation," *Journal of Visual Languages & Computing*, vol. 48, pp. 34–143, 2018.

[22] X. Wang, W. Wang, L. T. Yang, S. Liao, D. Yin, and M. J. Deen, "A distributed HOSVD method with its incremental computation for big data in cyber-physical-social systems," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 2, pp. 481–492, 2018.

[23] H. Wang, S. Ma, H. Dai, M. Imrand, and T. Wang, "Blockchain-based data privacy management with nudge theory in open banking," *Future Generation Computer Systems*, 2019.

[24] Y. Ma, Y. Wu, J. Ge, and J. Li, "An architecture for accountable anonymous access in the internet-of-things network," *IEEE Access*, vol. 6, pp. 14451–14461, 2018.

[25] Y. Wang, Z. Cai, G. Yin, Y. Gao, X. Tong, and G. Wu, "An incentive mechanism with privacy protection in mobile crowdsourcing systems," *Computer Networks*, vol. 102, pp. 157–171, 2016.

[26] Y. Wang, Z. Cai, X. Tong, Y. Gao, and G. Yin, "Truthful incentive mechanism with location privacy-preserving for mobile crowdsourcing systems," *Computer Networks*, vol. 135, pp. 32–43, 2018.

[27] X. Zhang, W. Dou, Q. He et al., "A generic framework for fast tree isolation-based ensemble anomaly analysis," in *Proceedings of the IEEE 33rd International Conference on Data Engineering (ICDE 2017)*, pp. 983–994, San Diego, CA, USA, April 2017.

[28] Q. He, J. Pei, L. Gao, D. Kifer, P. Mitra, and L. Giles, "Context-aware citation recommendation," in *Proceedings of the 19th International Conference on World Wide Web*, pp. 421–430, New York, NY, USA, 2010.

[29] H. Abdollahpouri, R. Burke, and B. Mobasher, "Controlling popularity bias in learning-to-rank recommendation," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pp. 42–46, Como, Italy, August 2017.

[30] Y. Wang, G. Yin, Z. Cai, Y. Dong, and H. Dong, "A trust-based probabilistic recommendation model for social networks," *Journal of Network and Computer Applications*, vol. 55, pp. 59–67, 2015.

[31] F. Meng, D. Gao1, W. Li, X. Sun, and Y. Hou, "A unified graph model for personalized query-oriented reference paper recommendation," in *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, pp. 1509–1512, San Francisco, CA, USA, October 2013.

[32] M. Gori and A. Pucci, "Research paper recommender systems: a random-walk based approach," in *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pp. 778–781, Omaha, NE, USA, October 2016.

[33] S. M. McNee, I. Albert, D. Cosley et al., "On the recommending of citations for research papers," in *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work*, pp. 116–125, New Orleans, LA, USA, November 2002.

[34] D. M. Pennock, "Collaborative filtering by personality diagnosis: a hybrid memory-and model-based approach," in *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, pp. 473–480, Cambridge, MA, USA, 2000.

[35] M. Yu, Y. Hu, X. Li et al., "Paper recommendation with item-Level collaborative memory network," in *Knowledge Science, Engineering and Management*, pp. 141–152, Springer International Publishing, Berlin, Germany, 2019.

[36] Y. Xu, L. Qi, W. Dou, and J. Yu, "Privacy-preserving and scalable service recommendation based on SimHash in A distributed cloud environment," *Complexity*, vol. 2017, Article ID 3437854, 9 pages, 2017.

[37] J. Tang and J. Zhang, "A discriminative approach to topic-based citation recommendation," in *Advances in Knowledge Discovery and Data Mining*, pp. 572–579, Springer-Verlag, Berlin, Germany, 2009.

[38] A. Alzoghbi, V. A. A. Ayala, P. M. Fischer, and G. Lausen, "Learningto-rank in research paper CBF recommendation: leveraging irrelevant papers," in *Proceedings of the 3rd Workshop New Trends Content-Based Recommender Systems*, pp. 43–46, Boston, MA, USA, 2016.

[39] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 448–456, San Diego, CA, USA, August 2011.

[40] H. Wu and Y. Sun, "On status QUO of citation net-work research and the overview on its development," *Computer Applications and Software*, pp. 164–168, 2012.

[41] Y. Liang, Q. Li, and T. Qian, "Finding relevant papers based on citation relations," in *Web-Age Information Management*, pp. 403–414, Springer, Berlin, Germany, 2011.

[42] H. Liu, H. Kou, X. Chi, and L. Qi, "Combining time, keywords and authors information to construct papers correlation graph," in *Proceedings of the 31st International Conference on Software Engineering and Knowledge Engineering (SEKE 2019)*, Lisbon, Portugal, July 2019.

[43] H. Wang, C. Guo, and S. Cheng, "LoC—a new financial loan management system based on smart contracts," *Future Generation Computer Systems*, vol. 100, pp. 648–655, 2019.

[44] A. Dwivedi, G. Srivastava, S. Dhar, and R. Singh, "A decentralized privacy-preserving healthcare blockchain for iot," *Sensors*, vol. 19, no. 2, p. 326, 2019.

[45] H. Wang, S. Ma, and H.-N. Dai, "A rhombic dodecahedron topology for human-centric banking big data," *IEEE Transactions on Computational Social Systems*, vol. 6, no. 5, pp. 1095–1105, 2019.