Keywords-Driven Paper Recommendation Based on Undirected Paper Citation Graph

Hanwen Liu (hanqingliu2019@outlook.com)
Nanjing University of Science and Technology https://orcid.org/0000-0002-2905-370X

Research

Keywords: recommendation, pattern minin

Posted Date: July 16th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-144551/v2

Keywords-Driven Paper Recommendation Based on Undirected Paper Citation Graph

Hanwen Liu
School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, 210096, China
Hanqingliu2019@outlook.com

Abstract—Nowadays, a paper recommender system often recommends required papers to users based on their various retrieval demands. Typically, the recommender system analyzes the typed keywords of users and then returns papers covering the query keywords, in an efficient and economic manner. In practice, one paper may only contain partial keywords that users are interested in; thus, the recommender system needs to return users a set of papers that collectively cover all requested keywords. As we all know, the papers' keywords represent their research contents and topics, so the recommender system only considering keywords may generate a set of papers that belong to different research domains and are actually not correlated, which fails to satisfy users' requirements on deep and continuous research on a certain domain or topic. Furthermore, users hardly select a set of satisfactory papers from a mass of candidate papers as users manually discover and verify the correlation relationships and popularity among candidate papers. To return a set of satisfactory papers, we propose RSSP (recommend a set of satisfactory papers) approach based on an undirected paper citation graph, the keywords-driven and popularity-aware approach for paper recommendation, which integrates manuscript keywords, paper discovery and paper selection operations. In addition, we conduct large-scale experiments on the real-life Hep-Th dataset to further demonstrate the usefulness and efficiency of the RSSP. Furthermore, the experimental results can prove that the RSSP significantly saves users' time and effort in searching for satisfactory papers.

Index Terms—Paper Recommendation, Keyword Search, Popularity, Undirected Paper Citation Graph, Steiner Tree, Dynamic Programming

1. INTRODUCTION

With the increasing maturity of recommender system technology, users use some paper recommender websites (e.g., Google Scholar and Baidu Academic) to seek required paper. Furthermore, some the information filtering tools [1], such as search engines, also help users to search for papers. Generally, since a paper may contain partial keywords that users are requested, the recommender system needs to analyze users' requirement and return a set of papers that collectively contain all query keywords.

As formally depicted in Figure 1, the common process of paper recommendation [2] mainly consists of three phases: the first phase is manuscript keywords, where users analyze their research requirements and submit some query keywords, e.g., the keyword[1] is the *paper recommendation*, the keyword[2] is the *keyword search*, the keyword[3] is the *Steiner tree* and the keyword[4] is the *dynamic programming*; the second phase is paper discovery, where the recommender system automatically identifies diverse sets of candidate papers; the third phase is paper selection, where the recommender system recommends the candidate papers that contain query keywords to users. Frankly, these recommended papers that fail to satisfy users' requirements on deep and continuous research on certain content or topic as these papers may belong to the wide variety of research domains or aren't correlated.

In addition, the keywords search method [3] has long been popularized in searching for paper, but this method is hardly to find a set of satisfactory papers. That is because a set of satisfactory papers must

satisfy following requirements: one is collectively covering the users' requested keywords and having higher popularity; on the other hand, whether one candidate paper containing query keywords has a direct or indirect correlation relationship with other candidate paper containing diverse query keywords. In short, this work still needs further in-depth analyses and study. Therefore, none of existing recommender systems are directly applied to effectively recommend a set of satisfactory papers.

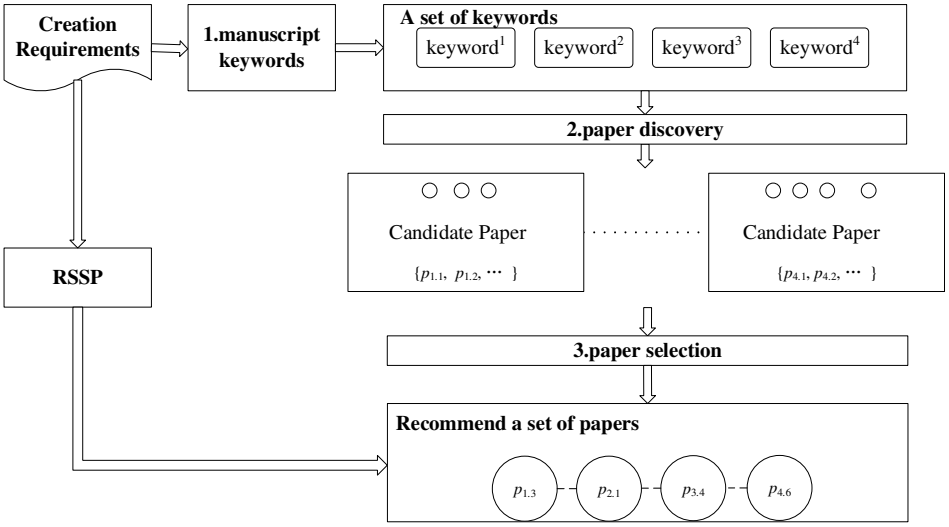


Figure 1. The RSSP recommendation process.

Here, we propose RSSP (recommend a set of satisfactory papers) approach assisting users in searching for a set of satisfactory papers, i.e., these papers are not only covering all requested keywords but also having higher popularity and correlation among papers. As shown in Figure 1, the RSSP integrates and automates the manuscript keywords, paper discovery and paper selection operations, offering a novel keywords-driven and popularity-aware approach for the paper recommendation. Frankly, our proposal helps users to search for multiple correlative and popular papers by entering a few keywords representing users' research content. Moreover, the RSSP runs on an undirected paper relationships graph, where paper is modeled as node and a connected edge represents whether there has correlation relationship among papers. In practice, the RSSP will return one or multiple subgraphs of the paper relationships graph according to a set of query keywords; the returned subgraphs include the keyword papers that cover the query keywords, the bridging papers (if any) needed however not specified by the requested keywords, and the composability and popularity of those papers.

In summary, we make the following main contributions:

(1) We propose a novel keywords-driven and popularity-aware paper recommendation approach for efficiently recommending a set of satisfactory papers, and making it easy to realize users' research goals. In other words, these recommended papers satisfy users' requirements on deep and continuous research on a certain domain or topic.

(2) We build an undirected paper citation graph, where paper is modeled as node and its edges are regarded as correlation relationships. Based on this graph, we are regarded the users' keywords query problem as the minimum group Steiner tree problem and obtain multiple solutions by utilizing the dynamic programming technique. According to the obtained multiple solutions, we will employ the popularity method to find an optimal solution.

(3) We conduct large-scale experiments on the Hep-Th dataset [4] to evaluate the usefulness and efficiency of the RSSP. Furthermore, the experimental results can prove that the RSSP significantly recommend a set of satisfactory papers to users.

The rest of paper is organized as follows: Section 2 demonstrates the research motivation of this paper, Section 3 defines an undirected paper citation graph, Section 4 formulates the research problem, Section 5 introduces how the RSSP answers the keywords query on the undirected paper citation graph, Section 6 evaluates the RSSP by using some experimental results, Section 7 reviews the related work, Section 8 further concludes this paper and points out the future research directions.

2. RESEARCH MOTIVATION

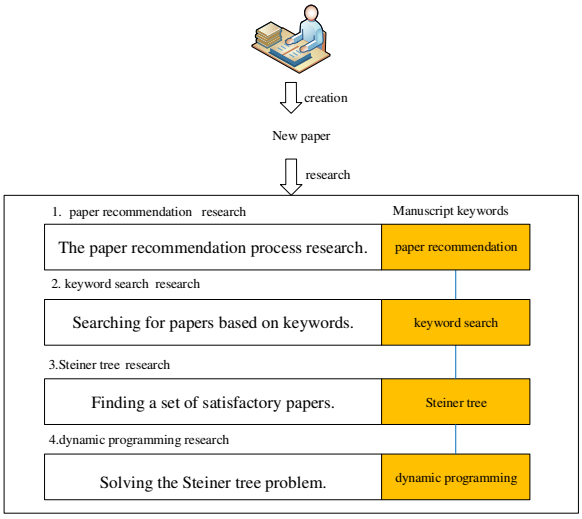


Figure 2. An example of paper research and creation task.

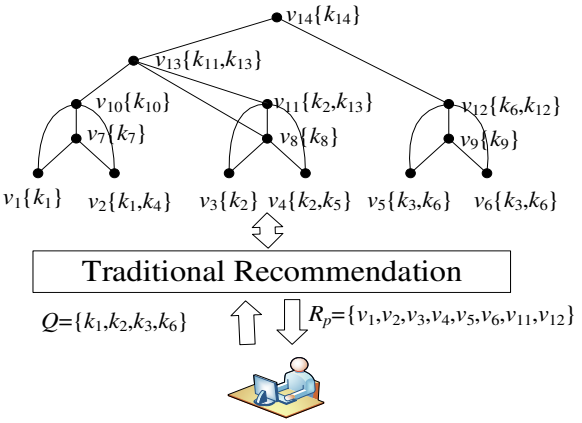


Figure 3. Tradition recommendation: an example.

In this section, using examples of Figure 2 and Figure 3 for demonstrating the research motivation of the paper. Figure 2 shows that a user needs to perform the following keywords research tasks in creating new papers: (1) *paper recommendation* (i.e., k_1) for paper recommendation process research; (2) *keyword search* (i.e., k_2)for keyword search research and applying it to the paper recommender process; (3) *Steiner tree* (i.e., k_3)for the Steiner algorithm [5-6] research and applying it to the keyword search ; (4) *dynamic programming* (i.e., k_6)for the dynamic programming [7-8] technique research and applying it to solve the 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. 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, …, v_{14}. Furthermore, the notation $v_{13}\{k_{11}, k_{13}\}$ indicates that node v_{13}

offers keywords k_{11} and k_{13}. Thus, given a query $Q = \{k_1, k_2, k_3, k_6\}$, the user easily to search 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 requested keywords; however, it is probable that his still has no idea of whether those papers can collectively finish his research task as the correlation relationships among different papers and the popularity of each paper are both transparent to users. In addition, each user must manually finds a set of satisfactory papers from massive candidate papers, and this process is very time-consuming and challenging. To tackle these issues, a novel keywords-driven and popularity-aware paper recommendation approach named RSSP is proposed in this paper, which will be detailed presentation in Section 5.

3.UNDIRECTED PAPER CITATION GRAPH

Many papers can form paper citation graph according to their citation relationships [9-10]. Moreover, the paper citation graph can provide valuable information about the papers' correlation relationships as the facts have proved that the papers' knowledge transitive. For example, a citation relationship $\{paper_1 \rightarrow paper_2\}$ indicates that not only $paper_2$ cites $paper_1$, but also $paper_1$ and $paper_2$ are correlative. Thus, such correlation relationships information supports an undirected paper citation graph to be built by digging deeper into citation relationships among papers in the paper citation graph. Such as another citation relationship $\{paper_2 \rightarrow paper_3\}$ in addition to $\{paper_1 \rightarrow paper_2\}$, we know that $paper_1$ is connected to $paper_2$ and $paper_2$ is connected to $paper_3$, so the undirected paper citation graph has the following correlation relationships: $\{paper_1 - paper_2 - paper_3\}$. As more citation relationships are mined and papers are included in the undirected paper citation graph, this graph will grow larger and denser, offering a solid base for recommending a set of satisfactory papers.

The RSSP 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*. Furthermore, each node contains multiple keywords representing the research content of paper, i.e., $k_1, \ldots k_m$. In the remainder of this paper, we will speak inter-changeably of a paper and its corresponding node in G_p, both denoted as *v*.

DEFINITION 2. Edges: For a pair of nodes (v_i, v_j), the G_p contains a corresponding edge $e(v_i, v_j)$. $e(v_i, v_j)$ denotes the correlation relationship among nodes v_i and v_j.

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. Honestly, paper database might contain papers that belong to different domains, e.g., recommender system and protein engineering; thus, it is possible that the paper database has multiple disconnected undirected paper citation graphs. In addition, users will not enter entirely irrelevant query keywords, i.e., each query keyword belongs to different research domains; frankly, we don't consider such keyword in our research.

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

4 PROBLEM FORMULATION

In my opinion, our proposal includes one key point: recommending a set of satisfactory papers. Specifically, given a keywords query Q (i.e., $Q = \{k_1, \ldots, k_l\}$), answering the query Q mainly consists of

two steps: (1) to find answer trees on an undirected paper citation graph G_p, denoted as *T(Q)*, where *T(Q)* is not only covering all query keywords but also having the fewest number of nodes (i.e., the higher correlation); (2) to obtain optimal answer trees $T_1(Q)$ based on *T(Q)*, where $T_1(Q)$ has the highest popularity.

An intuitive example is presented in Figure 4, where Figure 4 is a part of the G_p and it contains 14 nodes covering diverse keywords. Furthermore, the notation $v_1\{k_1\}$ indicates that node v_1 only offers a keyword k_1 in Figure 4, and the edge $e(v_1, v_{10})$ indicates that nodes v_1 and v_{10} have correlation relationship.

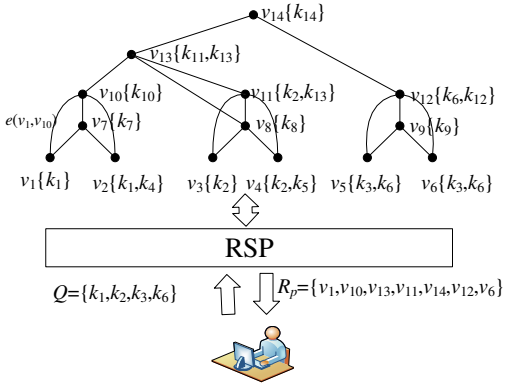


Figure 4. RSSP recommendation: an example.

In Figure 4, a user enters a set of keywords $\{k_1, k_2, k_3, k_6\}$ to search for papers. Here, v_1 and v_2 contain k_1, v_3, v_4 and v_{11} contain k_2, v_5 and v_6 contain k_3 and k_6, v_{12} contain k_6. Thus, given the $Q = \{k_1, k_2, k_3, k_6\}$, we are looking for an answer 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 $\{v_5, v_6, v_{12}\}$. This answer tree also connects nodes that do not cover any of query keywords, e.g., v_{10}, $v_{13}$and v_{14}. Honestly, such nodes v_{10}, v_{13} and v_{14} play an important role in supporting users to do continuous research. Therefore, the user obtains a Steiner tree, i.e., $R_p = \{v_1, v_{10}, v_{13}, v_{11}, v_{14}, v_{12}, v_6\}$. The 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 is a connected subgraph, T_P forms a Steiner tree in the G_p.

According to individual keywords *k* in a keywords query *Q*, we use the inverted indexes of Section 3 to identify multiple sets of nodes, denoted as $V_{p1}, \ldots V_{pl}$, where $V_{pn}(1 \le n \le l)$ at least contains the keyword k_n. Next, finding a group Steiner tree on the G_p, 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 $V_{pn}(1 \le n \le l)$ contains keyword $k_n$in the *Q*. When T_P is a Steiner tree and contains exactly one node of each group $V_{pn}(1 \le n \le l)$, the T_P forms a group Steiner tree in the G_p.

Here, we obtain multiple the group Steiner trees over the G_p according to the *Q*. Next, RSSP aims to find a minimum group Steiner tree (i.e., *T(Q)*) that not only covers the users' query keywords but also has the fewer nodes (i.e., the higher correlation). Furthermore, the *T(Q)* contains keyword nodes (i.e., the nodes containing the query keywords) and bridging nodes. In fact, the bridging nodes have a significant effect on the *T(Q)* as the bridging nodes are necessary nodes that connect the keyword nodes. Thus, the minimum group Steiner tree is defined as follows:

DEFINITION 6. Minimum Group Steiner Tree. Given a set of exact group Steiner trees, i.e., $T_{P1}, \dots, T_{Pm}$, when$(|T_{Pi}|) = min(|T_{P1}|, \dots |T_{Pm}|)$, the $T_{pi}(0 < i \leq m)$ is the minimum group Steiner tree in the G_p. Here, the $|T_{Pi}|$ represents the number of nodes in T_{Pi}.

Here, RSSP aims to define the minimum group Steiner tree. However, the method of finding answer tree (i.e., *T(Q)*) is the NP-complete problem in the G_p. Thus, the next section mainly introduces how to find answer trees *T(Q)* by using the DP (dynamic programming) technique [5-6] and how to induce optimal answer trees $T_1(Q)$ based on the *T(Q)*.

5 RSSP MECHANISMS

5.1Find Multiple Solutions

According to the above analysis, this subsection mainly discusses employing the DP technique to solve the MGST (the minimal group Steiner trees) problem. More specifically, the DP technique firstly breaks the MGST problem down into a series of simpler subproblems and solves each of subproblems only once, and then the corresponding results of subproblems are stored. Finally, the DP technique can effectively offer multiple solutions (i.e., *T(Q)*) by combining previously solved subproblems.

In this subsection, we are regarded all keywords entered by users as *K*, i.e., *K*=*Q*. In the DP model, we let the $T_{Pmin}(v, K')(K' \subseteq K)$ is a state and $w_{DP}(T_{Pmin}(v, K'))$ represents the number of nodes in $T_{Pmin}(v, K')$. Furthermore, the $T_{Pmin}(v, K')$ rooted at node *v* contains the users' query keywords K'. The state-transition equation of the DP model as follows:

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

$$w_{DP}\left(T_{Pmin}(v, K')\right) = \min\left(w_{DP}\left(T_{Pg}(v, K')\right), w_{DP}\left(T_{Pm}(v, K')\right)\right) \tag{2}$$

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

$$w_{DP}\left(T_{Pg}(v, K')\right) = \min_{\left(\begin{smallmatrix} K'_u \cap K' \neq K' \,\&\&\, K'_u \cap K' = K'_u \\ \| K'_u \cap K' \neq K'_u \,\&\&\, K'_u \cap K' = K' \end{smallmatrix}\right)}\left\{w_{DP}\left(T_{Pmin}(u, K'_u)\right) + w_{DP}\left(T_{Pmin}(v, K')\right)\right\} \tag{4}$$

$$w_{DP}\left(T_{Pm}(v, K')\right) = \min_{\left(\begin{smallmatrix} K'_u \subseteq K \,\&\&\, K' \subseteq K \,\&\&\, \\ K'_u \cap K' \neq K' \,\&\&\, K'_u \cap K' \neq K'_u \end{smallmatrix}\right)}\left\{\left|T_{Pmin}(v, K') + T_{Pmin}(u, K'_u)\right|\right\} \tag{5}$$

$$w_{DP}\left(T_{Pm}(v, K')\right) = \min_{K'_1 \cap K'_2 = \Phi}\left\{w_{DP}\left(T_{Pmin}(u, K'_1) \oplus T_{Pmin}(u, K'_2)\right)\right\} \tag{6}$$

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

Where $N(v)$ is a set of *v*'s neighbors in G_p, i.e., $u, v \in G_p(V_p, E_p)$, $u \in N(v)$ and $e(u, v) \in E_p$. Formula (1) indicates that the weight of tree is 1 in the DP model owing to only covering one keyword node. Formula (2) indicates that $T_{Pmin}(v, K')$ is obtained by using the following two operations, i.e., tree growth operation formally represented by Formula (3)-(4) and tree merging operation formally represented by Formula (5)-(6). In Figure 5. (a), the tree growth operation generates new $T_{Pmin}(v, K')$ by adding new node *u* (i.e., one of *v*'s neighbors) to $T_{Pmin}(v, K')$. Furthermore, Figure 5. (b) shows that the tree merging operation generates new $T_{Pmin}(v, K')$ by merging two trees both rooted at *v*.

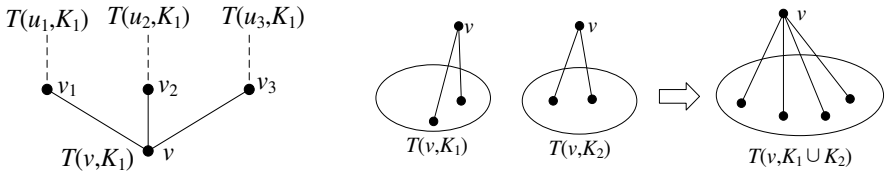


Figure 5. (a)Tree growth Figure 5. (b)Tree merging

Figure 5. Tree growth and tree merging operations.

Algorithm 1: MGST (G_p, K)

Input:

$G_p(V_p, E_p)$: a paper correlation graph.

$K = \{k_1, k_2, \ldots, k_l\}$: the users' query keywords.

Output:

$T_{Pmin}(v, K)$: a minimum group Steiner tree rooted at v.

```
1   Let Q^1 be a queue in ascending order of number of tree nodes
2   Q^1 = Φ
3   for each v ∈ V_p do
4    if v contains any nonempty keyword set K' ⊆ K
5        then enqueue T_Pmin(v, K') into Q^1
6    Min_count = ∞      // the number of nodes in the minimum group Steiner tree
7    while Q^1≠Φ do
8        dequeue Q^1 to T_Pmin(v, K')
9        if K'= K
10           then if w_DP(T_Pmin(v,K')) < Min_count
11                then Min_count = w_DP(T_Pmin(v,K'))
12                return T_Pmin(v, K')
13             else break
14       else
15           for each u∈N(v) do       // tree growth
16            if 1 + w_DP(T_Pmin(u,K'_u)) < w_DP(T_Pmin(v,K'))
17             then w_DP(T_Pmin(v,K'))=1 + w_DP(T_Pmin(u,K'_u))
18                  T_Pmin(v, K') = v + T_Pmin(u, K'_u)
19                  enqueue T_Pmin(v, K') into Q^1
20                  update Q^1
21            else
22             if (K'_u∩K'≠K'&&K'_u∩K'=K'_u||K'_u∩K'≠K'_u&&K'_u∩K'=K')    //tree growth
23                 if w_DP(T_Pmin(v,K'))+w_DP(T_Pmin(u,K'_u))<w_DP(T_Pmin(v,K'))
24                  then w_DP(T_Pmin(v,K'))=w_DP(T_Pmin(v,K'))+w_DP(T_Pmin(u,K'_u))
25                       enqueue T_Pmin(v, K') into Q^1
26                       update Q^1
27             else if (K'_u ⊆ K&&K' ⊆ K&&K'_u∩K' ≠ K'_u&&K'_u∩K'≠K')    //tree merging
28                  then T_Pmin(v, K') =T_Pmin(u, K'_u) +T_Pmin(v, K')
29                  enqueue T_Pmin(v, K') into Q^1
30                  update Q^1
31           K'_1 = K'
32           for each K'_2 in K s.t (K'_1∩K'_2 = Φ) do       //tree merging
33                if T_Pmin(v, K'_1) ⊕ T_Pmin (v, K'_2) < T_Pmin(v, K'_1 ∪ K'_2)
```

34 then $T_{Pmin}(v, K_1' \cup K_2') = T_{Pmin}(v, K_1') \oplus T_{Pmin}(v, K_2')$

35 if $K_1' \cup K_2' = K$

36 then if $w_{DP}\left(T_{Pmin}(v, K_1' \cup K_2')\right) <$ Min_count

37 then Min_count = $w_{DP}\left(T_{Pmin}(v, K_1' \cup K_2')\right)$

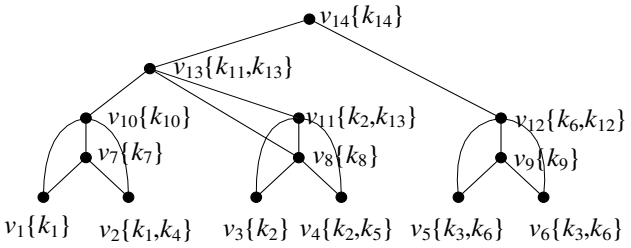
38 return $T_{Pmin}(v, K_1' \cup K_2')$

39 else break

40 else enqueue $T_{Pmin}(v, K_1' \cup K_2')$ into Q^1

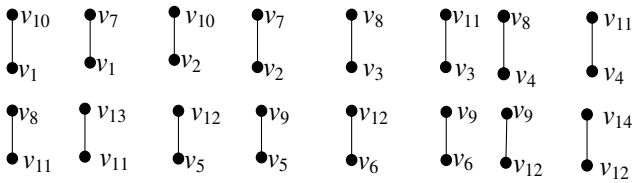
41 update Q^1

In Algorithm 1, we repeat the tree growth operation and the tree merging operation to obtain multiple answer trees (i.e., $T_{Pmin}(v, K)$). In fact, Algorithm 1 mainly maintains a queue Q^1, where the queue ranks generated trees according to the number of nodes and stores trees having the fewer nodes. Specifically, the lines 1-8 of Algorithm 1 is the pre-processing stage; the lines 9-13 denotes that a tree covering all queried keywords is inserted into the Q^1; the lines 14-26 executes the tree growth operation; the lines 27-41 executes the tree merging operation of tree.

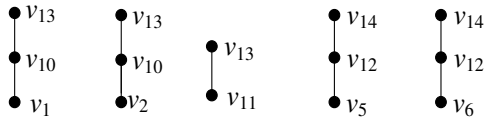


(a)

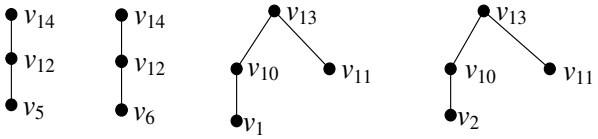
(b)



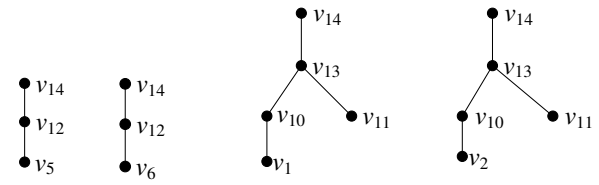
(c)



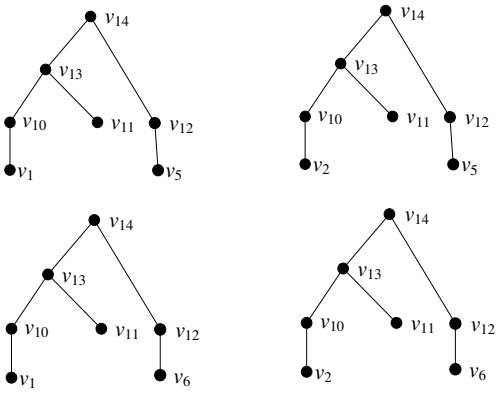
(d)



(e)



(f)



(g)

Figure 6. Minimal group Steiner tree generation process.

Next, an intuitive example of Figure 6 shows the execution process of Algorithm 1 in response to the query Q, i.e., K=$\{k_1, k_2, k_3, k_6\}$. The trees rooted at nodes containingk_1, k_2, k_3or 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 6(b). Since these eight trees only one node, the tree growth operation is performed in Figure 6(b). Fortunately, these nodes are both containing neighbor nodes in G_p, so edges connecting any one of nodes $v_1, \ldots v_6, v_{11}$and v_{12} are generated to the corresponding trees in Figure 6(c). The tree merging operation doesn't execute in the Figure 6(c) as this operation must increase new query keywords to generated trees. For example, while the tree $\{v_8, v_3\}$ and tree $\{v_8, v_{11}\}$ can generate a new tree rooted at node v_8, i.e., $\{v_8, v_3, v_{11}\}$, this operation is not the tree merging operation as this tree doesn't contain new query keywords. Thus, the tree growth operation is performed on Figure 6(c). Furthermore, some new generated trees are deleted as these trees are of no use, e.g., tree $\{v_8, v_{11}\}$ can generate new tree $\{v_8, v_{11}, v_{13}\}$, but the new tree contain more nodes and same query keywords (i.e., k_2). Therefore, five required trees are both retained in Figure 6(d). Next, we execute the tree merging operation in Figure 6(d) and Figure 6(f) and the tree growth operation in Figure 6(e). Finally, the user obtains four minimal group Steiner trees (i.e., *T(Q)*) in Figure 6(g).

Note that, we consider that the output results of the algorithm may be entire graph, e.g., the G_p; on the other hand, the worst-case scenario is that our algorithm fails to recommend papers to users.

5.2 Find Optimal Solutions

According to Algorithm 1, it is possible to return multiple qualified trees (i.e., *T(Q)*) based on the users' query keywords, e.g., the results of Figure 6 is obtained four answer trees. Generally, the number of paper citation frequency, the larger the better as the larger paper citation frequency often means the higher popularity of one paper. To further ease the heavy burden of users' research, we select an optimal answer tree with higher popularity (i.e., $T_1(Q)$) from the output result of Algorithm 1(i.e., *T(Q)*) and recommend $T_1(Q)$ to users. Thus, we employ the popularity method [12] to evaluate the *T(Q)*. Furthermore, the Paper Popularity (PP) [13] is defined as follows:

$$\mathrm{PP} = \sum_{v_i \in \mathrm{MST}, v_j \in \mathrm{G}_p} d\left(v_i, v_j\right) \tag{8}$$

Where, for any pair of nodes v_i and v_j, node v_i belongs to the *T(Q)* and node v_j belongs to the G_p.

$d(v_i, v_j)$= 1 if v_j cites v_i in paper citation graph (i.e., $v_i \rightarrow v_j$) and 0 otherwise.

In the PP model, we produce a ranking list in descending order according to the popularity values of each candidate answer trees. Finally, RSSP returns optimal solutions, i.e., $T_1(Q)$ having the highest popularity among candidate answer trees. Note that, we also consider that the recommendation result of the PP model could be all candidate answer trees, i.e., the $T(Q)$.

6 Experiments

To demonstrate the usefulness of the RSSP, large-scale experiments are designed and tested on existing paper citation dataset.

6.1 Experimental Environment

The experiments are conducted on a machine with Intel(R) Core(R) CPU @3.0GHz, 16GB RAM and Windows 10 @ 1809. The software configuration environment: Windows 10 @ 1809 and Python 3.6. In addition, a paper citation graph is extracted from the Hep-Th dataset [18]. And this graph covers 7304 papers and each node contains the keywords information.

Generally, an author allows creating up to 6 index terms in an article, so we create the query keywords with up to 6 in our research. Here, three sets of experiments (i.e., set A, set B and set C) are conducted. In set A, all keywords of one paper are used as a query Q. This scenario emulates that users exactly provide all 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 the query keywords. In set C, the query keywords are selected from two papers randomly, which further verifies the feasibility of Algorithm 1. Note that, we don't have to find optimal solutions in set C. Furthermore, each experiment is repeated 50 times and the average experiment results is adopted.

Frankly, we test the following evaluation criteria:

(1) Number of Nodes. Number of recommended papers, the smaller the better as fewer recommended papers often mean higher correlation among papers.

(2) Success Rate [8]. When the number of recommended papers is not larger than twice the number of query keywords, this recommendation result is successful.

(3) Average Paper Popularity (APP) [15]. APP defines as follows:

$$\mathrm{APP} = \frac{\sum_{z \in m}^{m} \mathrm{PP}_z}{\sum_{z \in m}^{m} n_z} \tag{10}$$

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

(4) Computation Time. The consumed time for generating optimal answer trees .

(5) Precision [14]. Precision is calculated as:

$$\mathrm{Precision} = \begin{cases} \dfrac{TP}{T_1(Q)} & \text{set A and set B} \\ \dfrac{TP}{T(Q)} & \text{set C} \end{cases} \tag{9}$$

where, the TP denotes a set of papers containing queried keywords.

(6) Recall. Recall is calculated as:

$$\mathrm{Recall} = \frac{1}{|p|} \sum_{p_a \in p} \frac{T(Q) \cap T_{p_a}}{T_{p_a}} \tag{11}$$

where $|p|$ is the number of papers, i.e., $|p|$=2 in set C, . T_{p_a} is the set of papers cited by p_a.

(7) F1-score. F1-Score is calculated as:

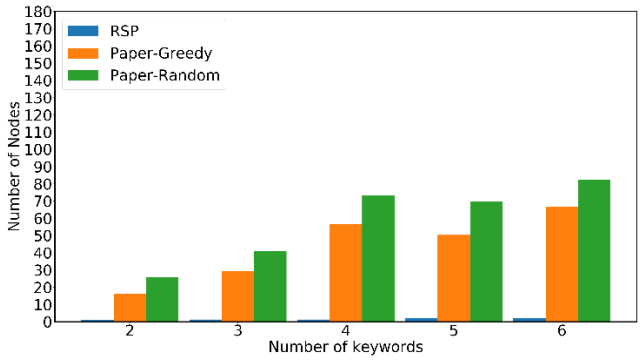
$$\text{F1-score} = \frac{2 * \text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}} \tag{12}$$

RSSP offers a new keyword-driven and popularity-aware approach for finding a set of satisfactory papers. To the best of our knowledge, fewer approaches address the keywords query issue by using the correlation relationships of papers. Thus, we compare RSSP with two straightforward approaches:

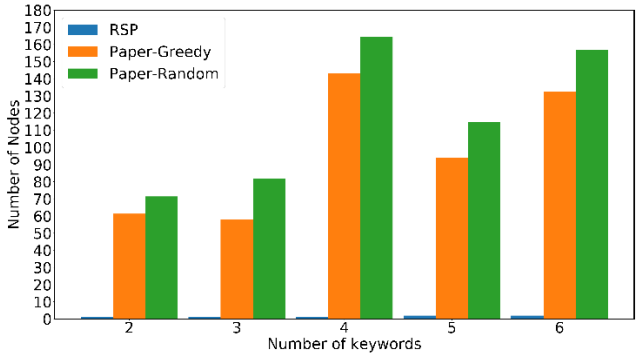
Baseline Approach 1. Paper-Random: This approach is randomly selecting a set of nodes that collectively cover all query keywords in the G_p. Next, the approach finds minimum Spanning trees that interconnect the selected nodes, and the trees contain the fewest number of nodes. Finally, we can obtain an optimal minimum Spanning tree by employing the popularity method.

Baseline Approach 2. Paper-Greedy: Likewise, the greedy approach is randomly selecting a set of nodes that collectively cover all query keywords in the G_p. Next, this approach regards the selected nodes as initial root nodes and continuously grows the trees until those nodes are interconnected. Furthermore, the greedy heuristic algorithm is applied in the trees grow process, which leads to first selecting neighbor nodes containing most queried keywords. Finally, we also use the popularity method for obtaining an optimal tree.

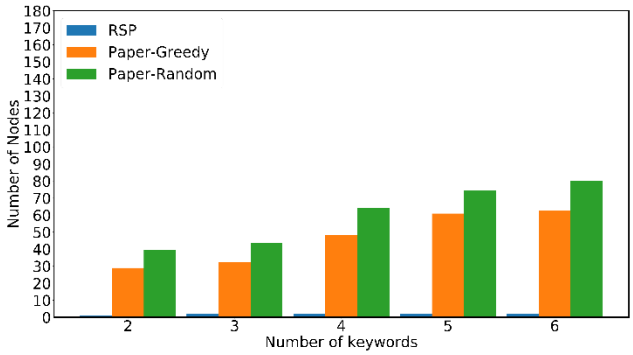
6.2 Experimental Results



(a) The number of returned nodes in set A.



(b) The number of returned nodes in set B



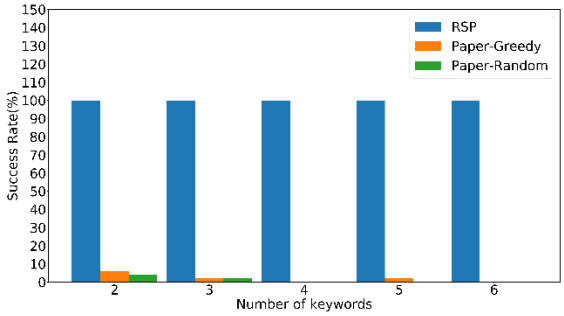
(c) The number of returned nodes in set C

Figure 7. The number of nodes recommended by three different approaches.

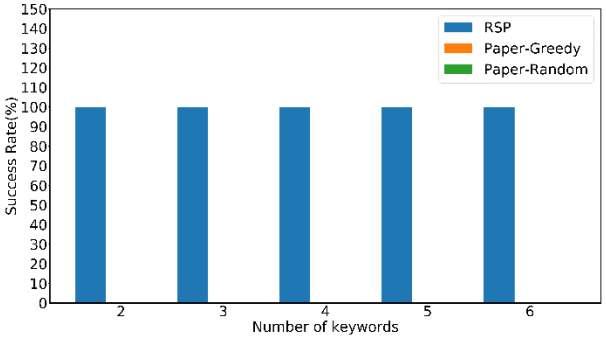
1) Profile-1: the number of nodes recommended by three different approaches .

In this profile, we compare the number of returned nodes in the RSSP with other approaches (i.e., Paper-Greedy and Paper-Random). As shown in Figure 7, the number of the users' query keywords is a range from 2 to 6. In Figure 7, the quantity of recommended papers in our approach increases with the number of queried keywords increasing, which is mainly because returned solutions including more papers can satisfy more query keywords requirements of users. For other two approaches, when the number of queried keywords equals to 6 in set A or the number of queried keywords equals to 4 in set B and set C, they obtain maximum paper quantity. In addition, these experiments results show that our proposal acquires a smaller number of recommended papers than other two approaches; of course, the smaller number of recommended papers guarantee higher correlation among papers. Therefore, the RSSP is superior to other two approaches (i.e., Paper-Greedy and Paper-Random).

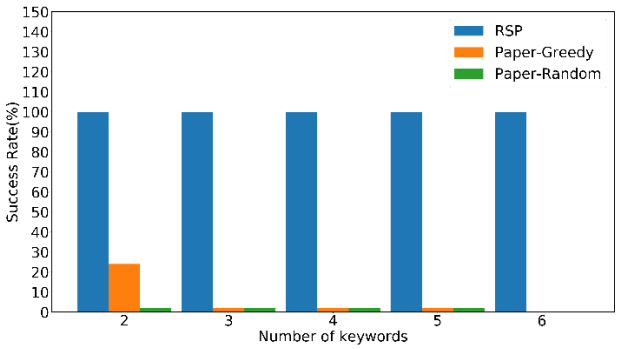
2) Profile-2: the success rate of three different approaches.



(a) The success rate in set A



(b) The success rate in set B

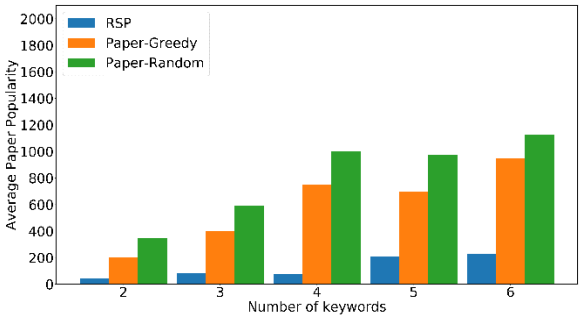


(c) The success rate in set C

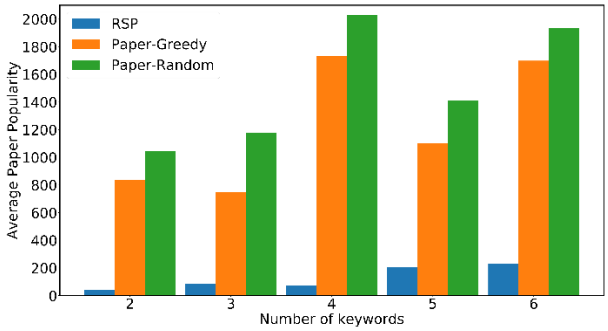
Figure 8. The success rate of three different approaches.

In this profile, we compare the success rates of three different approaches. As shown in Figure 8, the experiment results of the three different approaches are very different in the different experiment set. Facing to the different experiment scenarios (i.e., the set A, the set B and the set C), Figure 8 presents that our proposal can effectively answer the users' keywords query and the success rate is 100%. However, the Paper-Random and the Paper-Greedy are difficult to get successful solutions as the number of queried keywords increasing; especially, the success rates of these two approaches are both equal to 0 in the set B. Again, the experiments result present that our proposal can effectively acquire optimal solutions than other approaches.

3) Profile-3: the average paper popularity of three different approaches.



(a) The average paper popularity in set A

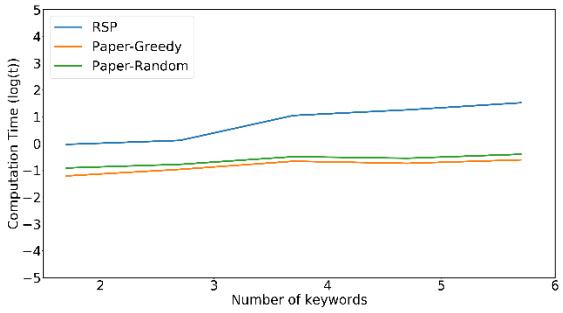


(b) The average paper popularity in set B

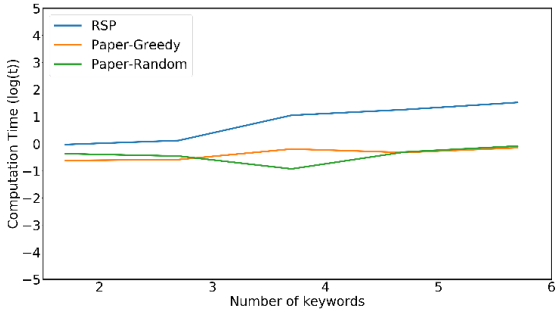
Figure 9. the average paper popularity of three different approaches.

In both the set A and the set B, we compare three different approaches by utilizing the average paper popularity. As shown in Figure 9, these experiment figures show that the average paper popularity of the Paper-Random and the Paper-Greedy are both larger than the RSSP, which is mainly because the quantity of recommended papers obtained by the Paper-Random and the Paper-Greedy are both in excess of our approach. Of course, the number of times the paper has been cited by others which more than twice.
In fact, the solutions of the Paper-Random and the Paper-Greedy will be seldom selected as these solutions take users a serious amount of time and energy to do some unnecessary research. In my opinion, Figure 9 presents that the average paper popularity of the RSSP is allowable and receivable in the case of satisfying the users' query keywords requirement.

4) Profile-4: the computation time of three different approaches.



(a) The computation time in set A

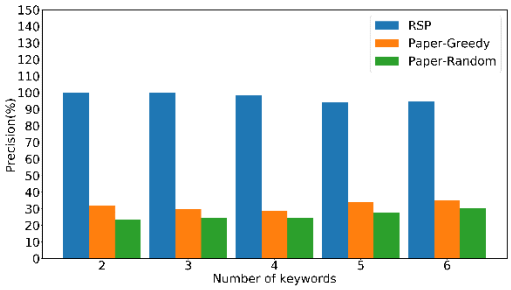


(b) The computation time in set B

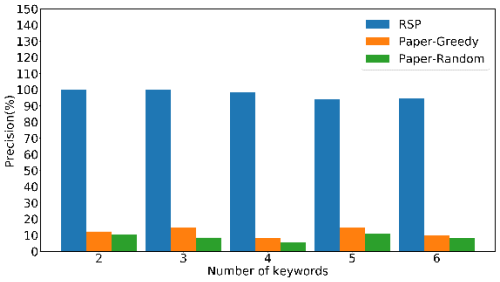
Figure 10. The computation time of three different approaches.

In this profile, we compare the time consuming of three different approaches in both the set A and the set B, respectively. Here, the computation time is regard as the time of finding optimal solutions (i.e., a set of satisfactory papers). As shown in Figure 10, the RSSP and the Paper-Greedy approaches spend more the computation time getting the optimal solutions as the number of query keywords increasing. Likewise, the Paper-Random only spend more the computation time with the number of query keywords increasing in the set A. In fact, the computation time of three approaches may increase exponentially with the number of query keywords increasing. In addition, since the Paper-Random and the Paper-Greedy all use extremely simple heuristic for selecting keyword nodes and bridging nodes, these two approaches spend fewer the computation time than the RSSP. Frankly, while our proposal needs to take a lot of time to select optimal solutions, the computation time of the RSSP 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 achieve their research goal.

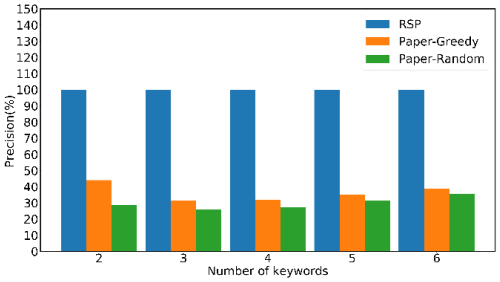
5) Profile-5: the precision values of three different approaches.



(a) The precision in set A



(b) The precision in set B

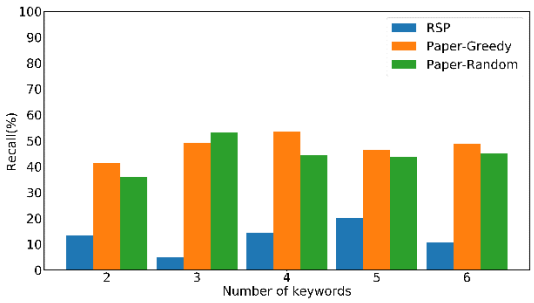


(c) The precision in set C

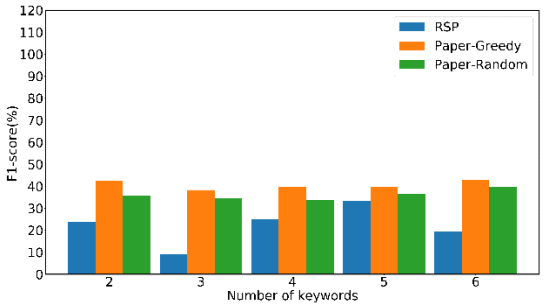
Figure 11. The precision of three different approaches.

As shown in Figure 11, these three figures present that the precision of the three different approaches in the different experiment set, respectively. Luckily, our proposal can accurately answer the users' keywords query and the precision is 100% in three different experiment set. For the Paper-Random and the Paper-Greedy, their precision values are a range from 10% to 45%. Therefore, whether users can accurately or randomly offer the query keywords, our approach can accurately answer the users' query and the RSSP can better meet the requirements of users than other two approach (i.e., Paper-Random and Paper-Greedy). Furthermore, these experiment results show further that users will spend fewer time and energy on realizing their research aim based on our approach.

6) Profile-6: the recall rate and F1-score values of three different approaches.



(a) The recall in set C



(b) The F1-score in set C

Figure 12. The recall rate and F1-score values of three different approaches.

In this profile, we compare the recall rate and F1-score of three different approaches in the set C, respectively. As shown in Figure 12, the recall rate values of the Paper-Random and the Paper-Greedy are a range from 39% to 54%, and the F1-score values of these two approaches are a range from 33% to 44%. Furthermore, the recall rate and F1-score values of the RSSP are a range from 4% to 21% and a range from 9% to 34%, respectively. Since the number of returned papers of the Paper-Random and the Paper-Greedy are in excess of the RSSP, and their experiment results in the set C may contain diverse trees (i.e., the minimal group Steiner trees), the recall rate and F1-score values of our proposal are smaller than other two approaches. In conclusion, the recall rate and F1-score values of the set C can directly verify the feasibility of our Steiner tree algorithm (i.e., Algorithm1).

7 RELATED WORK

The early work on paper recommendation mainly explored the use of collaborative filtering (CF) techniques; for example, the [16] mainly focused on the rating matrices in a paper citation network. Furthermore, content-based filtering (CBF) [17-18] approach attempted to retrieve papers with respect to textual content. However, the CF approach is generally limited by some problems, e.g., cold start problem and data sparsity problem [19]; furthermore, the CBF approach also suffers from the traditional information retrieval issues, such as semantic ambiguity problem [20].

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' contents and the transmission of knowledge. For example, the [21] proposed a graph-based PageRank-like recommendation approach that performs a biased random walk on a paper citation graph, and this approach further emphasized on the citation correlations. Furthermore, Wu et al. [22] thought that three different types of paper citation networks could be constructed based on citation relationships, i.e., directly connected network, coupling network and co-citation network. The [23] has

been proved that the co-citation relationships of co-citation network could be employed in a paper recommendation, e.g., if two papers are both cited by more same papers, these two papers have highly relevant and they are highly likely to be recommend simultaneously. Honestly, the citation relationships among papers can be formed in paper citation graph as most papers select their references based on the content similarity [24]. Thus, we use the citation relationships of paper citation graph for establishing the correlation relationships of our paper recommendation approach in this paper.

8 CONCLUSIONS

Whether a set of satisfy papers will be recommended to users is 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., the RSSP) to return a set of satisfy papers, i.e., these papers are not only covering the users' query keywords but also having higher correlation and popularity among papers. Specifically, the RSSP assists users to search a set of appropriate papers by typing the users' query keywords in an undirected paper citation graph G_p, and this recommendation results support users do deep and continuous research on a certain topic and domain. In addition, the experimental results further show that the usefulness and efficiency of our proposal.

Although our work shows desirable results, there are still some aspects worth further research and improvement: Since users cannot analyze their requirement in detail, e.g., the required experiment evaluation criteria, the recommended results may fail to satisfy academic creation requirements. Thus, how to supplement these research contents in paper recommendation process, which is further study and progress.

REFERENCE

[1] P. M. Herceg, T. B. Allison, R. S. Belvin, and E. Tzoukermann. Collaborative exploratory search for information filtering and large-scale information triage in Journal of the Association for Information Science and Technology, 395-409, 2018.

[2] S. Jieun, B. K. Seoung. Academic paper recommender system using multilevel simultaneous citation networks in Decision Support Systems, 24-33, 2018.

[3] Y. Jun, and D. Yong. Controllable keyword search scheme supporting multiple users in *Future Generation Computer Systems*, 433-442, 2018.

[4] J. Leskovec, J. Kleinberg and C. Faloutsos. Graphs over Time: Densification Laws, Shrinking Diameters and Possible Explanations on ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2005.

[5] J. Su, F. Lin, X. Zhou and X. Lu. Steiner tree based optimal resource caching scheme in fog computing," in *China Communications*, 161-168, 2015.

[6] V. F. Fedor, K. Petteri, L. Daniel, P. Fahad, and S. Saket. Parameterized Single-Exponential Time Polynomial Space Algorithm for Steiner Tree in *SIAM J. Discrete Math.,* 327–345, 2019.

[7] N. Löhndorf, and A. Shapiro. Modeling time-dependent randomness in stochastic dual dynamic programming in *SIAM J. Discrete Math.,* 327–345, 2019.

[8] L. Qi, Q. He, F. Chen, W. Dou, S. Wan, X. Zhang. and X. Xu. Finding All You Need: Web APIs Recommendation in Web of Things Through Keywords Search in IEEE Transactions on Computational Social Systems. 1-10, 2019.

[9] Z. Zhou, S. Chen, M. Hu, and Y, Liu. Visual ranking of academic influence via paper citation," in *Journal of Visual Languages & Computing*, 34-143, 2018.

[10] T. Zhang, H. Chi, and Z. Ouyang. Detecting Research Focus and Research Fronts in the Medical Big Data Field Using Co-word and Co-citation Analysis in *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, 313-320,2018.

[11] J. Zhou *et al*. A Generic Inverted Index Framework for Similarity Search on the GPU in *2018 IEEE 34th International*

Conference on Data Engineering (ICDE), 893-904, 2018.

[12] R. Zhou, S. Khemmarat, L. Gao, H. Wang. Boosting Video Popularity Through Recommendation Systems in Databases and Social Networks. ACM, 13-18, 2011.

[13] Q. He, J. Pei, L. Gao, D. Kifer, P. Mitra, L. Giles., "Context-aware Citation Recommendation" in Proceedings of the 19th International Conference on World Wide Web. ACM, 421-430, 2010.

[14] P.M. Chuan, L. H. Son, M. Ali, T. D. Khang, and N. Dey. Link prediction in co-authorship networks based on hybrid content similarity metric in *Applied Intelligence,* 2470–2486, 2018.

[15] H. Abdollahpouri, R. Burke, B. Mobasher. Controlling Popularity Bias in Learning-to-Rank Recommendation in Proceedings of the Eleventh ACM Conference on Recommender Systems. ACM, 42-46, 2017.

[16] S. McNee, I. Albert, D. Cosley, P. Gopalkrishnan, S. Lam, A. Rashid, J. Konstan, and J. Riedl. On the recommending of citations for research papers in *Proceedings of the 2002 ACM conference on Computer Supported Cooperative Work*, Pages 116-125, 2002.

[17] J. Tang and J. Zhang, A discriminative approach to topic-based citation recommendation *in Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Berlin, Germany: Springer-Verlag, 572-579, 2009.

[18] R. Yan *et al.* Guess what you will cite: Personalized citation recommendation based on users preference in *Asia Information Retrieval Sympo-sium*. Berlin, Germany: Springer-Verlag, 428-439, 2013.

[19] Z. Yang, B. Wu, K. Zheng, X. Wang, and L. Lei. A survey of collaborativeltering-based recommender systems for mobile Internet applications," *IEEE Access*, 3273-3287, 2016.

[20] L. C. Totti, P. Mitra, M. Ouzzani, and M. J. Zaki. A query-oriented approach for relevance in citation networks," *in Proc. 25th Int. Conf. Companion World Wide Web*, 401-406, 2016.

[21] 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*, 778-781, 2016.

[22] H. Wu, Y. Sun. On Status QUO of Citation Net- work Research and the Overview on its Devel- opment in Computer Applications and Software, 164-168, 2012.

[23] Y. Liang, Q. Li, T. Qian. Finding relevant papers based on citation relations in Web-Age Information Management, Springer Berlin Heidelberg, 403–414, 2011.

[24] D. R. Amancio, O. N. Oliveira, L. D. F. Costa. Three-feature Model to Reproduce the Topology of Citation Networks and the Effects from Authors' Visibili- ty on Their H-index in *Journal of Informatics*, 427-434, 2012.