K-Loop Free Assignment in Conference Review Systems

Longhua Guo,*† Jie Wu,† Wei Chang,‡ Jun Wu,* Jianhua Li*
*School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China
†Department of Computer and Information Sciences, Temple University, Philadelphia, USA
‡Department of Computer Science, Saint Joseph's University, Philadelphia, USA
Email: staring@sjtu.edu.cn, jiewu@temple.edu, wchang@sju.edu, junwuhn@sjtu.edu.cn, lijh888@sjtu.edu.cn

***Abstract*—Peer review is a common process for evaluating paper submissions and selecting high-quality papers at academic conferences. A significant task is assigning submissions to appropriate reviewers. Considering the constraints of reviewers, papers, and conflicts of interest, retrieval-based methods and assignment-based methods were proposed in previous works. However, an author could also be a reviewer in the conference. The loops between authors and reviewers may cause cooperative cheating. In this paper, two algorithms are proposed for a *k*-loop free assignment, which ensures the loop length is no less than *k*. Inspired by the existing works, the first algorithm assigns reviewers to maximize the summation of suitability scores, temporarily ignoring the *k*-loop free constraint. Afterward, the loops are detected and adjusted based on mergers. The second algorithm generates *k*-loop free assignments within the nodes that are both reviewers and authors. The other assignments are generated using linear programming. Extensive experiments show the effectiveness of the proposed methods.**

***Index Terms*—Loop-free, paper-reviewer assignment, loop detection, merger.**

I. Introduction

As the most common practice in evaluating papers, peer review systems attache a great significance to selecting high-quality papers in a conference [1]. Conference organizers are faced with the challenge of managing the peer-review process. The organizers receive submissions from authors and identify competent reviewers to evaluate the manuscript's quality. Paper-reviewer assignment (PRA) is an important task that assigns the submitted papers to suitable reviewers [2]. However, the quantity of submitted papers and program committees are quite large in some conferences. Manual PRA is really time consuming. Cheating is one of the drawbacks of manual PRA. For example, a submitted paper is assigned by chairs to reviewers who have close relationships with the authors. What's more, in some systems (e.g., EasyChair [3]), reviewers are asked to bid on the preferred papers in order to improve the assignment quality. The reviewers may not have the time or patience to go through all the papers' titles and abstracts, which results in a bias of the bids. This cheating drawback also exists for a bid-based system.

Automatic PRA systems are proposed and implemented to avoid the above drawback, e.g. Infocom [1] and the NIPS conference review system [4]. Considering reviewers' maximum workloads and the quantity of reviewers that each paper requires, the PRAs try to find an optimized assignment subject to the constraints [1]. Apart from the above two constraints, the papers are not allowed to be assigned to the reviewers with close relationships as a Conflict of Interest (COI) forms. This includes co-author relationships, colleague relationships, advisor-advisee relationships [5].

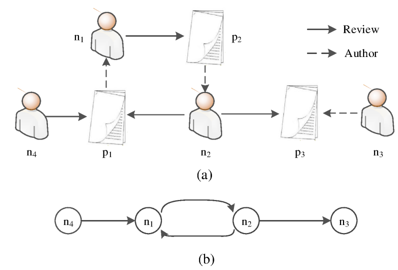


Fig. 1. A motivating example.

The objective in most systems is to pursue the optimal assignment, such as the suitability scores for maximization between papers and assigned reviewers. However, many people have two roles in a conference. They are authors who submitted papers to the conference and are also severed as reviewers to qualify the assigned papers. As a result, the authors and reviewers, who are working in similar academic areas, are more likely to have academic links not limited to the above COIs. In the worst case, authors of two different papers are reviewers and are assigned each other's paper. They may cooperate and give higher scores than they should. Such reviewer-paper loops increase the cheating risk and should be avoided in the automatic assignment.

Take Fig. 1 as an example. In Fig. 1(a), the solid lines represent review relationship. For instance, paper p_1 is reviewed by n_2 and n_4. The dotted lines represent the author's relationship. The author of p_1 is n_1. n_1 and n_2 serve as both authors and reviewers. n_3 is an author only, while n_4 is a reviewer only. The relationship between those four nodes can be modeled via Fig 1(b). There is a loop between n_1 and n_2.

To address the above problem, we add the loop constraint into the COI and propose a *k*-loop free assignment method. With the growth of the length of the loop, the hardness of cooperative cheating increases. Large loops are allowed, but their lengths must be greater than a pre-defined threshold, *k*. The larger *k* is, the smaller the risk of cooperative cheating is. This paper systematically models the PRA problem to avoid *k*-loops between reviewers and authors. Our objective is to maximize the summation of suitability scores subject to a set of constraints, including the workload capacity of reviewers, the required reviewer quantity of each paper, and the COIs. In the first algorithm, the papers are assigned using linear programming, temporarily ignoring the loop-free constraint. A directed graph is generated based on the assignment results. The loops are detected and adjusted based on mergers to meet the *k*-loop free constraint. The small loops are merged to generate a loop with length more than *k*. The second algorithm generates a *k*-loop free assignment for nodes which may have loops. The remaining papers and reviewers are assigned using linear programming afterwards. Our main contributions are summarized as follows.

- A loop detection algorithm is designed. Two *k*-loop free assignment algorithms are proposed subjected to the constraints.
- Extensive experiments on two real world datasets show the effectiveness of our method.

The remainder of the paper is organized as follows. Section II describes the related work. Section III presents the paper-reviewer assignment model and formulates the problem. The details of the proposed algorithms are given to generate the *k*-loop free assignment in Sections IV and V. Experiments on two real-world datasets are conducted to evaluate the proposed methods in Section VI. Finally, we draw our conclusion in Section VII.

II. Related work

In the paper-reviewer assignment, two categories are divided including retrieval-based methods and assignment-based methods. Retrieval-based methods focus on solving this problem using information retrieval and machine learning technologies. As a widely used system in both the machine learning and computer vision communities, the Toronto recommender system [4] provided an overview of the system and a summary of the learning models. Reviewers' bids, which represent their interest degree to a specific set of papers, are taken into consideration. Assessed expertise scores are utilized to predict missing scores. The term frequency-inverse document frequency algorithm was utilized in [6] to calculate the similarity score between papers and reviewers. [7] used a topic model to exploit reviewers' expertises. [8] considered various domain-specific constraints and proposed some specific matching algorithms to optimize the assignment procedure. [9] utilized the logistic model for expert recommendation and determination. [10] recommended a group of experts to submissions based on the topic distributions to maximize the weighted coverage.

The assignment-based methods take several constraints into consideration, such as the maximal workload of reviewers, required quantity for each submission, and conflicts of interests. The INFOCOM Review Assignment System [1] computed the suitability score between a submission and a reviewer using Latent Semantic Indexing. The total suitability score is maximized with the constraints using linear programming. The quality of each assignment pair is individually considered, however, it may turn out that an interdisciplinary paper is reviewed by a group of reviewers with too narrow of an expertise. [8] constructed a convex cost network and transformed the assignment problem to an equivalent minimum cost flow problem with various constraints. [11] automatically identified reviewers using a relative-rank particle-swarm propagation algorithm in a co-authorship network. The goal of [12] is suitability, the reviewer's bid preference, and expertise degree. [2] took the suitable score, interest trend, and authority of a reviewer into consideration. The three objectives are balanced with different weights and optimized using integer linear programming. [5] maximized the total number of distinct covered topics of the papers. A greedy algorithm is utilized which gives a 1/3-factor approximation. Considering the weights of topics, [10] maximized the topic coverage scores using a stage-deepening greedy algorithm. The Branch-and-Bound algorithm contributes to decreasing the search space. However, the above methods may cause loops in the assignment, which increases the cooperation cheat risks. In this paper, the *k*-loop free assignment algorithm is proposed with the constraints.

III. Model and problem formulation

The assignments between authors and reviewers can be modeled as a directed graph, *G*. $G = (V, E)$, where *V* represents the node set and *E* represents the edge set in the graph. g represents a subgraph in *G*, $g \in G$. The directed edge from node *i* to *j* is represented as $\{i, j\}$. Considering the reviewers who also submitted a paper in the conference, nodes can be classified into three categories: *A, B*, and *C*. *A* represents the reviewer-only nodes whose in-degree equals 0 and *B* represents the nodes who serve as both the author and reviewer. *C* represents the author-only nodes whose out-degree is 0. The numbers of each kind of nodes are N_a, N_b, and N_c respectively. $|V| = N_a + N_b + N_c$. *E* represents the assignments from the reviewer nodes to the author nodes. $\{i, j\}$ represents that the paper, written by *j*, is reviewed by *i*. $l(g)$ represents the loop length of subgraph *g*.

Definition 1: **Loop**. A loop is a set of directed edges which form a circle. The loop length is the minimal number of edges to form the loop in a graph.

Definition 2: ***k*-loop free assignment**. The minimal loop length in *G* is larger than *k*.

The objective of the problem is to maximize the summation of suitability scores between papers and reviewers. The problem is formulated as follows.

$$
\begin{aligned}
\max_{\alpha(r,p)} & \sum_{r} \sum_{p} s(r,p)\alpha(r,p) \\
s.t. & \sum_{p} \alpha(r,p) \le w, \forall r \in A \cup B \\
& \sum_{r} \alpha(r,p) = q, \forall p \in B \cup C \\
& \alpha(r,p) \in \{0,1\}, \forall r \in A \cup B, \forall p \in B \cup C \\
& l(g) \ge k, \forall g \in G
\end{aligned}
\tag{1}
$$

where $s(r,p)$ represents the suitable score between a reviewer, *r*, and a paper, *p*. $\alpha(r,p)$ is a binary variable of the assignment. $\alpha(r,p)=0$ if a reviewer, *r*, has a COI with submission *p*. w represents the maximal workload capacity for each reviewer. q represents the number of required reviewers for each paper. The length of loops in *G* is larger than *k*.

The problem is subject to the following three constraints: (1) Paper Demand Constraint: any paper must be reviewed by a given number of reviewers, (2) Reviewer Workload Constraint: each reviewer can review up to a certain number of papers, and (3) COI Constraint: there are no COI, including *k*-loop free constraint, between any pair of authors and their assigned reviewers.

IV. Merger based assignment algorithm

In this section, we propose an algorithm to generate a *k*-loop free assignment. First, the papers are assigned to maximize the total suitability scores [1] using linear programming, temporarily ignoring the loop free constraint. However, such assignments may conflict with the *k*-loop free constraint. Afterwards, the loops are detected based on topological sorting and the Depth-First Search (DFS) algorithm. The loops with $l(g)<k$ are merged with other loops to generate a *k*-loop free assignment.

A. Loop detection

Based on the assignment results, the directed graph is constructed. First, we check whether loops exist in the graph using topological sorting. The goal of the topological sorting algorithm is to find a linear ordering of vertices such that for any edge *(i,j)* in *E*, *i* precedes *j* in the ordering. For a graph which is not a directed acyclic graph, there will not be a topological sorting. Topological sorting is efficient for distinguishing whether there is a loop in the assignment [13].

At the beginning, the in-degree of each node is calculated. The first vertex is always a vertex whose in-degree is 0. The edges from this vertex are deleted in each iteration and the in-degree of the nodes are updated. If there are no vertices remaining, it shows there are no loops in the graph. As shown in Fig. 2(a), the subgraph with n_1 and n_2 is a directed acyclic graph with no loops. n_1 and n_2 are removed using two iterations. For the entire graph, n_1, n_2, and n_7 are removed. The other 9 nodes remain. As a result, loops exist in the graph.

For the remaining graph with loops, calculating the loop length and detecting the vertexes in each loop are solved using DFS. DFS-based algorithms are utilized to generate a tree and detect the back edge in the loop. The route from the root node to the leaf with the back edge forms a loop. For the connected graph, only one tree is generated to detect the back edge. For the unconnected graph, different parts are detected individually and generated to different trees. Stacks in the recursive procedure can be utilized to implement the algorithm. As it is acceptable if the length of the loop is more than *k*, the hight of the trees are limited to *k*. When the (*k*+1)-th level nodes are added to the tree, the top level father nodes can be deleted to decrease the storage space.

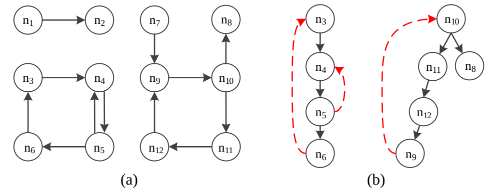


Fig. 2. An example for loop detection.

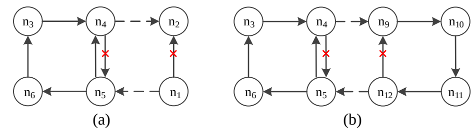


Fig. 3. An example for the merger-based algorithm.

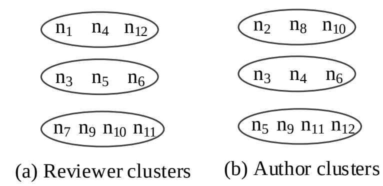


Fig. 4. An example for the clustering result.

As shown in Fig. 2(b), the red dotted directed edges represent the back edges. One back edge reflects one loop. If k is set as 5, three loops are conflicted to the loop constraint, including: $n_3\rightarrow n_4\rightarrow n_5\rightarrow n_6\rightarrow n_3$ as *loop*1, $n_4\rightarrow n_5\rightarrow n_4$ as *loop*2, and $n_{10}\rightarrow n_{11}\rightarrow n_{12}\rightarrow n_9\rightarrow n_{10}$ as *loop*3. If k is set as 3, only $n_4 \rightarrow n_5 \rightarrow n_4$ needs adjustment.

B. Merger based adjustment

To adjust the assignment, the loops are merged into large loops to meet the constraint. Assuming $\{n_i,n_j\}\in g_1$, $\{n_p,n_q\}\in g_2$, $\{n_i,n_j\}$ is a part of the loop whose length equals $l(g_1)$ and $\{n_p,n_q\}$ is a part of the loop whose length equals $l(g_2)$. We merge these two loops through breaking the edges $\{n_i,n_j\}$, $\{n_p,n_q\}$ and creating new edges as $\{n_i,n_q\}$, $\{n_p,n_j\}$. A larger loop is generated with length $l(g_1)+l(g_2)$. Such operations make no difference to $w(n_i),w(n_p)$ and $q(n_j),q(n_q)$, so the other assignments will not be in conflict. If $l(g_1)+l(g_2)\geq k$, the merger based adjustment is finished. A subgraph with no loop can regarded as a special loop with an infinite length. However, the created edges may lead to new loops. To avoid such cases, the connected graph with the new edges are detected to check if new loop conflicts are generated.

As shown in Fig. 3(a), as $\{n_4,n_5\}$ is a common edge for two loops, if $\{n_4,n_5\}$ exchanges endpoints with $\{n_1,n_2\}$, two loops will disappear. In Fig. 3(b), if $\{n_4,n_5\}$ exchanges endpoints with $\{n_{12},n_9\}$, the three loops will merge into two loops with lengths of 6 and 8, respectively.

To maximize the summation of suitable scores, it is important to choose proper edge pairs to break and create new edges. The merger of the loops will make a reduction to the total suitable scores, which is defined as merger cost *c*. If we merge these two loops through breaking the edges $\{n_i,n_j\}$, $\{n_p,n_q\}$ and creating new edges as $\{n_i,n_q\}$, $\{n_p,n_j\}$,

$$c = s(n_i, n_j) + s(n_p, n_q) - s(n_i, n_q) - s(n_p, n_j) \quad (2)$$

However, all edges in the graph can be the candidate edge that operates with each edge in the target loop. The enumeration-based algorithms spend a large amount of search space and computation costs. To simply the search, we utilize the cluster-based algorithm to filter similar reviewer pairs and similar paper pairs, which reduces the searching space. If the pairs n_i, n_p and n_j, n_q are similar, $s(n_i, n_j) - s(n_i, n_q)$ and $s(n_p, n_q) - s(n_p, n_j)$ will be reduced. As a result, the merger cost will be low. The suitability scores between reviewers and submissions represent the similarity between nodes, whose reciprocals can be regarded as the distance between two nodes.

As a classical partitioning technique of clustering, a simple and fast algorithm for the *K*-means clustering algorithm [14] is utilized to cluster users into several clusters. Based on the node types, the nodes are divided into reviewer-related clusters and author-related clusters. In each cluster, the distance between nodes are short, which means that they are similar to each other. The cluster contributes to the search space reduction. For $\{n_i, n_j\}$ in a loop, the $\{n_p, n_q\}$ is selected as a candidate when n_i, n_p and n_j, n_q are in the same cluster, respectively. If there are no new loops with length larger than *k* when $\{n_i, n_q\}$ and $\{n_p, n_j\}$ are created, $\{n_i, n_j\}$ and $\{n_p, n_q\}$ are regarded as an exchange pair to merge the loops. In the worst case, there are no such edge pairs that satisfy the cluster-based selection requirement. The edge $\{n_p, n_q\}$ is selected as a candidate when n_i, n_p or n_j, n_q are in the same cluster. The merger cost is calculated for each exchange pair. For loops in the graph, several exchange pairs are selected. We greedily select the pair with the minimal merger cost in each iteration until all the loops satisfy the *k*-loop free constraint.

For example, as shown in Fig. 4, the reviewer-related nodes and author-related nodes are divided into 3 clusters, respectively. For $\{n_3, n_4\}$, n_5 and n_6 are in the same reviewer cluster with n_3, and n_3 and n_6 are in the same author cluster with n_6. No edges are selected as candidates for the merger. For $\{n_4, n_5\}$, n_4 and n_{12} are in the same reviewer cluster, and n_5 and n_9 are in the same author cluster. No extra loops are created when $\{n_4, n_9\}$ and $\{n_{12}, n_5\}$ are connected. $\{n_4, n_5\}$ and $\{n_{12}, n_9\}$ is an exchange pair that merge the loops. As a result, Fig. 3(b) is a good option for the merger.

V. Shortest path based algorithm

A. k-*loop free assignment*

In the *k*-loop free assignment, the key point is to guarantee that *B* nodes, which servers as authors and reviewers, can conform to the loop constraint. Given the suitable score matrix between nodes in *B*, the reviewers are assigned based on suitable scores using a greedy algorithm. Initially, the directed edge with the maximal score is selected. In an iteration, if $\{n_i, n_j\}$ is selected, the other nodes' shortest path to n_i and n_j are detected. For example, if the path length from n_i to n_k is no greater than *k*, $s(n_i, n_k)$ are updated to 0. The workloads and reviewer numbers of n_i, n_j are calculated. If they reach the threshold, the scores related to n_i, n_j are updated to 0. The process will stop when no extra edge can be added.

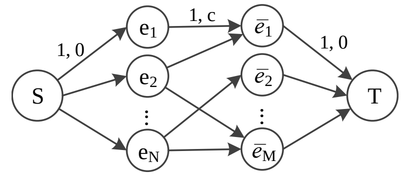


Fig. 5. The optimization process using network flow algorithm.

The assignment only takes nodes in *B* into consideration. However, not all the *B* nodes are assigned enough quantity reviewers. In addition, the nodes in *P* and *R* are not matched. The objective for this step is to maximize the summation of similarities for the rest of the assignment. The remaining workload and required quantity of reviewers are updated based on the existing assignment. The suitable scores between *B* nodes are set as 0, so no assignment will generate between the *B* nodes. This optimal assignment problem can be solved using linear programming.

B. *optimal result using network flow algorithm*

In above subsection, the *k*-loop free assignment is generated. However, the result is not an optimal solution considering the summation of suitable scores. Section V has shown that exchanging the end points of edge pairs is a good way to adjust the assignment. The edges are divided into two parts: the edges with start and end points all within *B* as part I and other edges as part II. The exchange pair consists of two edges from each part. Such an operation will not generate new loops or effect the workloads of the reviewers. The exchange cost for each exchange pair is calculated afterwards.

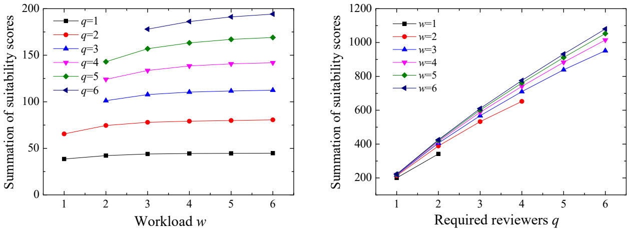
However, the exchange pairs many have common edges. One edge can only be adjusted once. A network flow model is utilized to optimize the assignment result. The problem can convex to a minimal cost maximum flow problem. As shown in Fig. 5, *S* and *T* the represent virtual source and terminal. Edges in $\{e_1, e_2, ..., e_N\}$ belong to Part I and edges in $\{\bar{e}_1, \bar{e}_2, ..., \bar{e}_M\}$ belong to Part II. The two parameters above each edge represents the capability and cost. The capability of all edges is limited to 1.

VI. Experiment Evaluations

A. *Datasets*

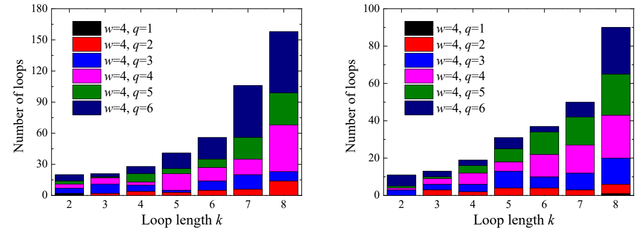
The first dataset is the SIGIR dataset, provided by Karimzadehgan and Zhai [15]. 73 papers and 189 reviewers from SIGIR 2007 are selected. 25 major subtopics are utilized to model the papers' and reviewers' academic areas. The experts assign relevant subtopic aspects to all the papers and reviewers through reading the abstracts and the expertise profiles. According to the subtopic information, we calculated the suitable score matrix between papers and reviewers.

The second dataset is the NIPS dataset, collected by Mimno and McCallum [7]. The information from 148 papers and 364 reviewers from the NISP 2006 conference are collected. In the



(a) SIGIR dataset (b) NIPS dataset

Fig. 6. The relationship between the objective and w, q.



(a) SIGIR dataset (b) NIPS dataset

Fig. 7. The relationship between N_l and q.

dataset, the reviewers are asked to provide a suitable ground truth between papers and reviewers. Four level-suitable scores are provided in the dataset from 0 to 3, where 0 is the least suitable and 3 represents the most suitable. In the dataset, 650 ground truth scores are provided. Based on the provided scores, the scores in the paper-reviewer matrix are calculated using matrix factorization [16], which is commonly used and infers the unknown scores in a recommendation system.

B. The loop detection results

Based on the suitable score matrix, the papers are assigned to reviewers regardless of the loop-free constraints using the linear programming algorithm. As shown in Fig. 6, the summation of suitable scores of the assignments are related to the w and q. In some cases, the workload and quantity of the reviewers are not large enough to meet the requirement. There is not a feasible solution, such as a case with w=2 and q=6. So, these points are absent in the figure. With the growth of q, more assignments are required, and the objective is almost increased linearly with the same w. If a larger w is allowed, the objective is optimized with the same q.

Afterwards, the loops are detected in the assignment graph. In the real world, the cooperation cheat risks decrease with the growth of the loop length. We calculated the number of loops with lengths from 2 to 8.

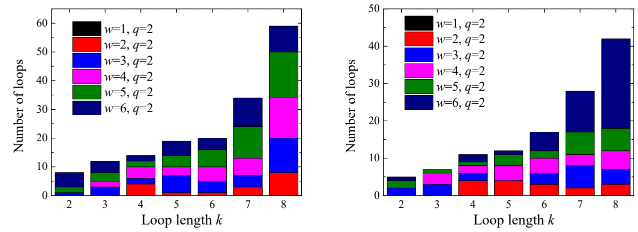
As shown in Fig. 7, the workload is set as 4, and required reviewers for each paper are different, which is set as 1 to 6. With the increase of the loop length, more loops are detected on the average. As a result, more loops need adjustment based on merger with the increase of k. What's more, with the increase of q, more assignment edges are generated, which also leads to a loop quantity increase on average.

As shown in Fig. 8, the required reviewers for each paper are set to 2, and the workload for each reviewer is different, set as 1 to 6. With increase of loop length k, more loops are detected in average.

N_b are nodes who sever as reviewers and authors. The assignments between N_b may lead to loops. The larger N_b is, the more loops are detected on average as shown in Fig. 9.

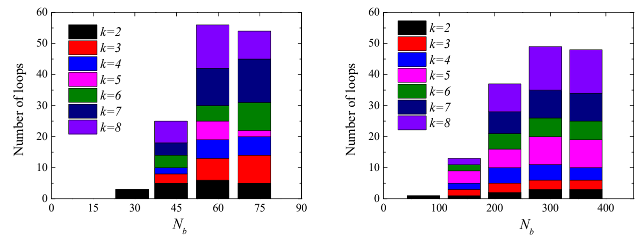
C. k-loop free assignment result

We first conduct a detailed simulation on the merger-based algorithm. With the detected loops, the merger-based algorithm is utilized to generate the k-loop free assignment. We conduct our experiments to evaluate the performance of the proposed assignment algorithm considering different variables,



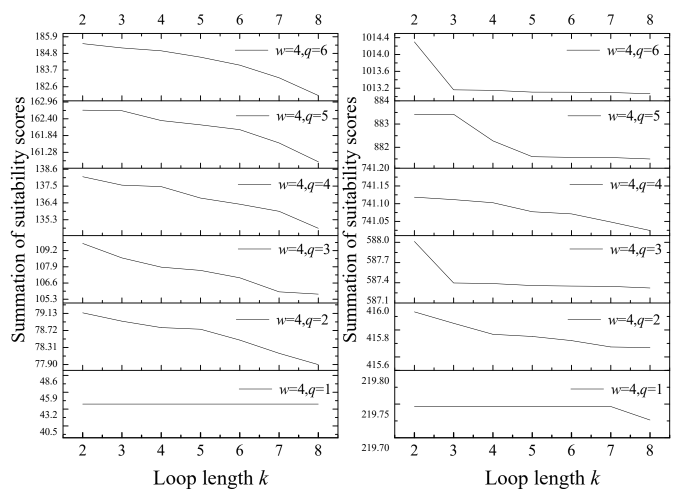
(a) SIGIR dataset (b) NIPS dataset

Fig. 8. The relationship between N_l and w.



(a) SIGIR dataset (b) NIPS dataset

Fig. 9. The relationship between N_l and N_b.

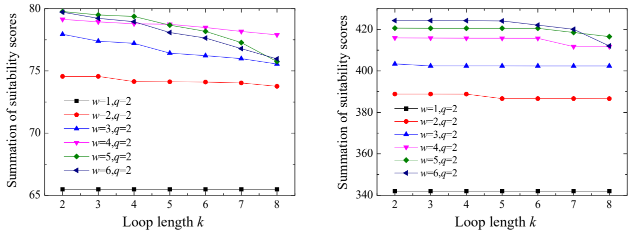


(a) SIGIR dataset (b) NIPS dataset

Fig. 10. The relationship between the objective and q.

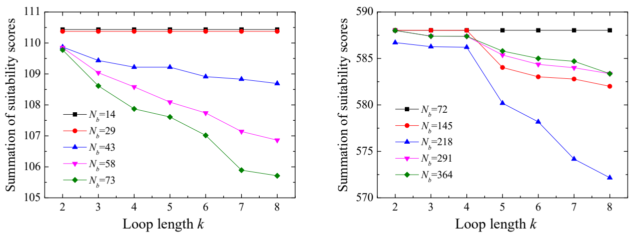
including k, w, q, and N_b. As shown in Fig. 10, if k is larger, more loops are detected and merged. The objective is decreased because of the merger cost. If no more loops need to be merged, the objective remains the same as shown in the lowest graph in Fig. 10(a).

As shown in Fig. 11, the required reviewers for each paper are set to 2, and the workload for each reviewer is different,



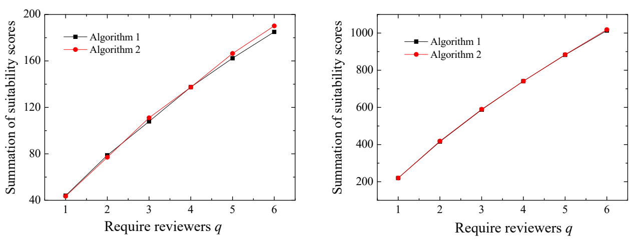
(a) SIGIR dataset (b) NIPS dataset

Fig. 11. The relationship between the objective and w.



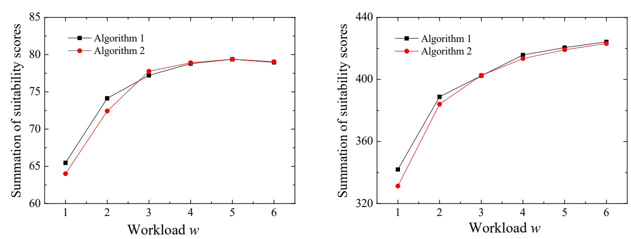
(a) SIGIR dataset (b) NIPS dataset

Fig. 12. The relationship between the objective and N_b.



(a) SIGIR dataset (b) NIPS dataset

Fig. 13. The comparison results between the two algorithms.



(a) SIGIR dataset (b) NIPS dataset

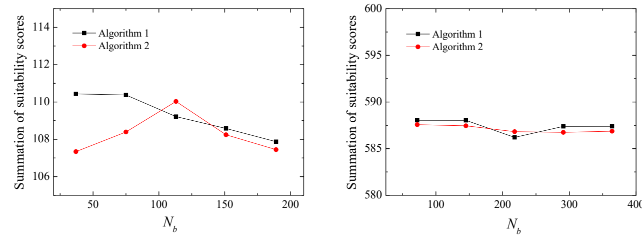
Fig. 14. The comparison results between the two algorithms.

set as 1 to 6. With increase of loop length k, the objective is decreased. N_b are nodes that serve as reviewers and authors, and the assignments between N_b may lead to loops. As shown in Fig. 12, the larger N_b is, the quicker the objectives decrease.

What's more, the comparison results between the two algorithms are shown in Fig. 13-15. In the simulation, k is set at 4. To keep consistent with the detailed simulation on Algorithm 1, w is set at 4 in Fig. 13. The summation of similarity scores increase with the growth of q. In Fig. 13, q is set at 2. In Fig. 15, w and q are set at 4 and 3. In general, Algorithm 1 shows a better performance than Algorithm 2, especially in the cases with a small number of loops.

VII. Conclusion

In this paper, we propose a generalized framework for k-loop free assignment. The condition of the loop-free assignment is discussed, which is useful for conference organizers who determine the number of reviewers and the threshold k. Two k-loop free assignment algorithms are proposed and are subject to the constraints. Extensive experiments on two datasets show the effectiveness of our proposed algorithms.



(a) SIGIR dataset (b) NIPS dataset

Fig. 15. The comparison results between the two algorithms.

VIII. Acknowledgments

This work is supported in part by the National Science Foundation of China (NSFC) grants 61571300, 61401273, and 61431008. This work of the first author was done during his stay as a visitor scholar at Temple University.

References

[1] B. Li and Y. T. Hou, "The new automated ieee infocom review assignment system," *IEEE Network*, vol. 30, no. 5, pp. 18–24, 2016.

[2] J. Jin, Q. Geng, Q. Zhao, and L. Zhang, "Integrating the trend of research interest for reviewer assignment," in *Proceedings of the 26th WWW Conference*, 2017, pp. 1233–1241.

[3] http://easychair.org.

[4] L. Charlin and R. Zemel, "The toronto paper matching system: an automated paper-reviewer assignment system," 2013.

[5] C. Long, R. C.-W. Wong, Y. Peng, and L. Ye, "On good and fair paper-reviewer assignment," in *Proceedings of the 13th ICDM Conference*, 2013, pp. 1145–1150.

[6] S. Hettich and M. J. Pazzani, "Mining for proposal reviewers: lessons learned at the national science foundation," in *Proceedings of the 12th ACM SIGKDD Conference*, 2006, pp. 862–871.

[7] D. Mimno and A. McCallum, "Expertise modeling for matching papers with reviewers," in *Proceedings of the 13th ACM SIGKDD Conference*, 2007, pp. 500–509.

[8] W. Tang, J. Tang, and C. Tan, "Expertise matching via constraint-based optimization," in *Proceedings of the IEEE WI-IAT Conference*, vol. 1, 2010, pp. 34–41.

[9] Y. Fang, L. Si, and A. P. Mathur, "Discriminative models of integrating document evidence and document-candidate associations for expert search," in *Proceedings of the 33rd ACM SIGIR Conference*, 2010, pp. 683–690.

[10] N. M. Kou, U. L. Hou, N. Mamoulis, and Z. Gong, "Weighted coverage based reviewer assignment," in *Proceedings of the 2015 ACM SIGMOD Conference*, 2015, pp. 2031–2046.

[11] M. A. Rodriguez and J. Bollen, "An algorithm to determine peer-reviewers," in *Proceedings of the 17th ACM CIKM Conference*, 2008, pp. 319–328.

[12] N. Di Mauro, T. Basile, and S. Ferilli, "Grape: An expert review assignment component for scientific conference management systems," *Innovations in applied artificial intelligence*, pp. 53–59, 2005.

[13] A. B. Kahn, "Topological sorting of large networks," *Communications of the ACM*, vol. 5, no. 11, pp. 558–562, 1962.

[14] A. Vattani, "K-means requires exponentially many iterations even in the plane," *Discrete & Computational Geometry*, vol. 45, no. 4, pp. 596–616, 2011.

[15] M. Karimzadehgan and C. Zhai, "Constrained multi-aspect expertise matching for committee review assignment," in *Proceedings of the 18th ACM CIKM Conference*, 2009, pp. 1697–1700.

[16] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, 2009.