國 立 交 通 大 學

資訊管理研究所

博士論文

以知識流探勘與文件推薦提供知識支援

Knowledge Flow Mining and Document Recommendation for Knowledge Support

研 究 生： 賴 錦 慧

指導教授： 劉 敦 仁 博士

中 華 民 國 九 十 八 年 十二 月

以知識流探勘與文件推薦提供知識支援

Knowledge Flow Mining and Document Recommendation for Knowledge Support

研　究　生：賴錦慧　　　　Student: Chin-Hui Lai

指導教授：劉敦仁　　　　Advisor: Dr. Duen-Ren Liu

國立交通大學

資訊管理研究所

博士論文

A Dissertation

Submitted to Institute of Information Management

College of Management

National Chiao Tung University

in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy in Information Management

December 2009

Hsinchu, Taiwan, the Republic of China

中華民國　九十八　年　十二　月

以知識流探勘與文件推薦提供知識支援

研究生：賴錦慧　　　　　　　　　　　　　　指導教授：劉敦仁 博士

國立交通大學資訊管理研究所

摘要

知識是獲得與維持組織競爭優勢的重要來源。在不斷變動的商業環境中，組織必須使用有效的方法來保留知識、分享知識和知識再利用，以協助知識工作者尋找工作相關的資訊。因此，要如何從工作者過去的工作記錄中，發掘與建構知識流（Knowledge Flow）是一個重要的議題。建立知識流模型的目的是在於，了解知識工作者的工作需求與參考知識的方式，進而提供適性化的知識支援。此外，組織中的知識是透過知識流的遞送與累積，而且知識工作者具備不同領域的知識，他們會參與以工作為基礎的群體，並進行合作，以滿足工作的需求。

本研究首先提出以知識流模型為基礎之混合式推薦方法，其整合知識流探勘、序列規則探勘，以及協同式過濾技術來推薦工作知識。這些以知識流為基礎的推薦方法包含二個階段：知識流探勘階段與知識流推薦階段。知識流探勘階段能藉由分析工作者的知識參考行為（資訊需求），以發掘工作者的知識流；而知識流推薦階段則利用所提出的混合式推薦方法，主動地提供相關知識給工作者。因此，根據工作者對於知識文件的喜好與知識參考行為，本研究方法能預測工作者感興趣的主題，進而推薦工作相關的知識文件給工作者。在實驗中，我們利用某研究單位實驗室的真實資料，來評估本研究之混合式方法的推薦效果，並與傳統的協同式過濾方法做比較。最後，實驗結果顯示，工作者對於知識文件的偏好與知識參考行為，可以有效地改善推薦品質並促進組織內的知識分享。

此外，為了協助群體學習與分享工作相關知識，針對以工作任務為基礎之群體，我們提出整合資訊檢索與資料探勘技術之演算法，發掘與建構群體知識流（Group-based Knowledge Flow）。群體知識流可利用有向性之知識圖來表示，藉此呈現一群工作需求相近工作者的知識參考行為（或知識流），而從知識圖中所發現的頻繁知識參考路徑，可以代表群體使用者的頻繁知識流。為了驗證方法的效能，我們實作一個群體知識流探勘之雛型系統。在一個重視協同合作與團隊合作的環境中，透過群體知識流探勘的方法與系統，可以加強組織學習，以及知識的管理、分享與再利用。

關鍵字： 知識流、知識流探勘、知識分享、文件推薦、協同式過濾、序列規則探勘、推薦系統、群體知識流、知識圖、資料探勘、資訊檢索.

Knowledge Flow Mining and Document Recommendation For Knowledge Support

Student: Chin-Hui Lai **Advisor: Dr. Duen-Ren Liu**

Institute of Information Management
National Chiao Tung University

Abstract

Knowledge is a critical resource that organizations use to gain and maintain competitive advantages. In the constantly changing business environment, organizations must exploit effective and efficient methods of preserving, sharing and reusing knowledge in order to help knowledge workers find task-relevant information. Hence, an important issue is how to discover and model the knowledge flow (KF) of workers from their historical work records. The objectives of a knowledge flow model are to understand knowledge workers' task-needs and the ways they reference documents, and then provide adaptive knowledge support. Additionally, knowledge is circulated and accumulated by knowledge flows (KFs) in the organization to support workers' task needs. Because workers accumulate knowledge of different domains, they may cooperate and participate in several task-based groups to satisfy their needs.

This work first proposes hybrid recommendation methods based on the knowledge flow model, which integrates KF mining, sequential rule mining and collaborative filtering techniques to recommend codified knowledge. These KF-based recommendation methods involve two phases: a KF mining phase and a KF-based recommendation phase. The KF mining phase identifies each worker's knowledge flow by analyzing his/her knowledge referencing behavior (information needs), while the KF-based recommendation phase utilizes the proposed hybrid methods to proactively provide relevant codified knowledge for the worker. Therefore, the proposed methods use workers' preferences for codified knowledge as well as their knowledge referencing behavior to predict their topics of interest and recommend task-related knowledge. Using data collected from a research institute laboratory, experiments are conducted to evaluate the performance of the proposed hybrid methods and compare them with the traditional CF method. Finally, the results of experiments demonstrate that utilizing the document preferences and knowledge referencing behavior of workers can effectively improve the quality of recommendations and facilitate efficient knowledge sharing.

Moreover, to support group-based learning and share task-related knowledge, we propose an algorithm that integrates information retrieval and data mining techniques to mine and construct group-based KFs (GKFs) for task-based groups. A GKF is expressed as a directed knowledge graph which represents the knowledge referencing behavior, or knowledge flow, of a group of workers with similar task needs. The frequent knowledge referencing path is identified from the knowledge graph to indicate the frequent knowledge flow of the workers. To demonstrate the efficacy of the proposed method, we implement a prototype of the GKF mining system. Our GKF mining method and system can enhance organizational learning and facilitate knowledge management, sharing, and reuse in an environment where collaboration and teamwork are essential.

Keywords: *Knowledge Flow, Knowledge Flow Mining, Knowledge Sharing, Document Recommendation, Collaborative Filtering, Sequential Rule Mining, Recommender System, Group-based Knowledge Flow, Knowledge Graph, Data Mining, Information Retrieval.*

致　謝

論文即將完成，也代表著即將完成現階段的任務，揮別漫長的博士班生涯，並邁向人生的另一個旅程、另一個開始。在過去的日子裡，一直很期盼能快快完成學業，然而在此刻來臨時，心中卻開始覺得依依不捨了。雖然還有許多夢想與目標尚未達成，未來的我仍會秉持自己的信念與堅持，繼續努力與成長。

在博士班期間，非常感謝指導教授劉敦仁老師，他是一位個性好相處、時時關懷與體貼學生、做研究認真的好老師。感謝劉老師對錦慧論文的細心指導，也教導我做研究的方法與態度，並培養我獨立思考與做研究的能力。在忙碌的研究之餘，老師也常會帶我們出去踏青、辦生日 party、並和我們一起唱歌一起同樂，讓我們舒解研究上的壓力，也讓研究室無時無刻充滿歡樂的氣氛。能夠順利畢業，還要感謝口試委員羅濟群老師、李永銘老師、李瑞庭老師與許秉瑜老師，在論文審查口試期間，給與許多寶貴的建議與指正，讓論文能更加完善。

感謝所有交大資管所老師的教導，讓我學習不同領域的知識。感謝實驗室的學長姐，在研究上與生活上的協助與指導；感謝可愛的學弟妹們，常常與你們一起玩樂，讓實驗室總是充滿快樂與歡笑，也謝謝你們幫忙我分擔許多雜務；感謝博士班的同儕，一起分享研究上的心得與經驗，舒解研究上的壓力；謝謝所有交大朋友們對我的照顧、關心與協助，並大方地與我分享生活哲學和有趣的事情，讓我的生活變得多采多姿。

最後，感謝我最摯愛的家人與親戚長輩們，謝謝你們不斷地鼓勵與支持，得以讓我無後顧之憂順利完成學位，也讓我有無限的力量與勇氣，去迎接人生的下一階段，去面對未來更多的困難與挑戰，真的很謝謝你們，我永遠愛你們。

賴錦慧
于新竹-交通大學
2009/12/08

Contents

List of Figures

List of Tables

Chapter 1. Introduction

1.1 Research Background and Motivation

Organizational knowledge can be used to create core competitive advantages and achieve commercial success in a constantly changing business environment. Hence, organizations need to adopt appropriate strategies to preserve, share and reuse such a valuable asset, as well as to support knowledge workers effectively [42, 44]. Knowledge and expertise are generally codified in textual documents, e.g., papers, manuals and reports, and preserved in a knowledge database. This codified knowledge is then circulated in an organization to support workers engaged in management and operational activities [12]. Because most of these activities are knowledge-intensive tasks, the effectiveness of knowledge management depends on providing task-relevant documents to meet the information needs of knowledge workers.

In task-based business environments, knowledge management systems (KMSs) can facilitate the preservation, reuse and sharing of knowledge. Moreover, workers may need to obtain task-relevant knowledge to complete a knowledge-intensive task by referencing codified knowledge (documents); For example, based on a task's specifications and the process-context of the task, the *KnowMore* system [1] provides context-aware knowledge retrieval and delivery to support workers' procedural activities. The task-based *K-support* system [39, 58] adaptively provides knowledge support to meet a worker's dynamic information needs by analyzing his/her access behavior or relevance feedback on documents. To help knowledge workers complete multiple tasks, *TaskTracer* [19] was developed to monitor workers' activities and help them rapidly locate and reuse processes employed previously. However, previous research on task-based knowledge support did not analyze and utilize the flow of knowledge among various types of codified knowledge (documents) to provide effective recommendations about task-relevant documents.

Knowledge flow (KF) research focuses on how KF can transmit, share, and accumulate knowledge when it passes from one team member/process to another. In a workflow situation, work knowledge may flow among workers in an organization, while process knowledge may flow among various tasks [61-62, 64]. Thus, KF reflects the level of knowledge cooperation between workers or processes and influences the effectiveness of teamwork/workflow. Zhuge [61] proposed a management mechanism for realizing ordered knowledge sharing, and integrated the knowledge flow with the workflow to assist people working in a complex and knowledge intensive environment. Also, KF plays an important role in academic research, as researchers often devise novel concepts based on previous research reported in the literature [63]. However, to the best of our knowledge, there is no systematic method that can flexibly identify KF in order to understand the information needs of workers. Furthermore, conventional KF approaches do not analyze knowledge flow from the perspective of information needs and recommend relevant documents based on the discovered KF.

Knowledge workers normally have various task needs over time. Moreover, they may need to obtain task-relevant knowledge to complete a task by referencing several types of codified knowledge (documents); and the knowledge in one document may prompt a worker to reference another related document. Based on a worker's referencing behavior, KF can be used to describe the evolution of information needs, preferences, and knowledge accumulated for a specific task. From the perspective of information needs, some knowledge in a KF may have a higher priority for accomplishing a task. For example, before taking a Data Mining course, a student must take courses in Statistics and Database Systems, which represent the fundamental knowledge of Data Mining. Thus, these two courses are significant and have a high priority for the student. Additionally, academic knowledge may flow between different courses and thereby help students accumulate more knowledge. Similarly, the codified knowledge for a task also has different referencing priorities and ordering based on its perceived importance. In other words, important basic knowledge about a task should be referenced first. Therefore, KF can be utilized to provide effective recommendations about task-relevant knowledge to suit workers' information needs for tasks. This issue has not been addressed by previous research.

In task-based business environments, large amounts of such codified knowledge are circulated and accumulated in an organization to support knowledge workers engaged in diverse tasks and activities. Knowledge workers may cooperate with each other to accomplish a specific task. During the collaboration phase, task knowledge can be transmitted, shared and accumulated from one team member/process to another. Knowledge flows (KFs) can be used to represent the long-term evolution of workers' information needs [36]. Based on those needs, the knowledge flow-based document recommendation method proactively delivers task-relevant topics and documents to the workers.

To work more efficiently, workers who have task-related knowledge, expertise and experience may join a task-based group and collaborate to perform a task. The workers can share task-related knowledge delivered by their knowledge flows (KF) during the collaboration. In addition, workers in the same group may have similar referencing behavior and techniques for learning knowledge. Each group may require knowledge of different topic domains to accomplish its tasks and goals. Because the information needs of workers or groups may change over time, modeling the knowledge referencing behavior of a group of workers is difficult. Obviously, recognizing those needs, delivering knowledge during the collaboration, and facilitating knowledge sharing/reuse are important issues that must be addressed in a knowledge intensive organization. However, to the best of our knowledge, there is no appropriate approach for analyzing and constructing KFs from the perspective of a group's information needs; and very little research effort has been expended on KF mining for task-based groups.

1.2 Research Objectives

According to the research motivation, the major research objectives are listed below.

- Mining the knowledge flow for each knowledge worker and a group of workers;
- Identifying and analyzing topics of interest, major referencing behavior patterns, and the long-term evolution of workers' information needs;
- Providing knowledge support adaptively based on the referencing behavior of workers;
- Effectively recommending task-relevant knowledge to suit workers' information needs for tasks;
- Enhancing organizational learning and task collaboration;
- Facilitating knowledge dissemination, sharing and reusing among workers in the context of collaboration and teamwork;

1.3 The Approaches Based on Knowledge Flow

In an attempt to resolve the limitations of previous research, we first propose KF-based recommendation methods for recommending task-related codified knowledge. To adaptively provide relevant knowledge, collaborative filtering (CF), the most frequently used method, predicts a target worker's preference(s) based on the opinions of similar workers. However, the target worker's referencing behavior may change over the period of the task's execution, because his/her information needs may vary. Traditional CF methods only consider workers' preferences for codified knowledge. They neglect the effect of the time factor, i.e., workers' referencing behavior for knowledge over time. To fill this research gap, we propose a KF-based sequential rule method (KSR) that recommends codified knowledge by utilizing the KF-based sequential rules. However, the method is based on the target worker's referencing behavior without considering the opinions of his/her neighbors who may have similar preference for documents. Therefore, to take advantage of the merits of typical CF and KSR methods, we propose hybrid recommendation methods that combine CF and KSR methods to enhance the quality of document recommendation. The hybrid methods consider workers' preferences for codified knowledge, as well as their knowledge referencing behavior, in order to predict topics of interest and recommend task-related knowledge.

The proposed hybrid methods consist of two phases: a KF mining phase and a KF-based recommendation phase. To determine a knowledge worker's referencing behavior, the KF mining phase analyzes his/her historical work records to identify the knowledge flow, i.e., the target worker's information needs. Then, the KF-based recommendation phase selects and recommends documents based on the document preferences and KF-based sequential rules derived from the target worker's neighbors. In other words, the proposed methods trace a worker's information needs by analyzing his/her knowledge referencing behavior for a task over time, and also proactively provide relevant codified knowledge for the worker based on the KFs of the worker's neighbors.

According to the KF mining approach [36], we extend it and propose algorithms that integrate information retrieval and data mining techniques for mining and constructing the group-based knowledge flows (GKFs). Specifically, we discover a group's KF from the KFs of the participating workers. First, based on the workers' logs, we analyze each worker's referencing behavior when acquiring task-related knowledge, and then construct his/her KF. Workers who have similar KFs are clustered into the same group by a clustering method, and the resulting group is regarded as a working group. Because workers in the same group may adopt different behavior when referencing task-related knowledge, we design GKF mining algorithms to discover the frequent referencing behavior of a group of workers. Second, we apply the concepts of graph theory to visualize the GKF as a knowledge graph in which a vertex and an edge indicate, respectively, a topic domain and a direct flow relation between two topic domains. From the knowledge graph, frequent knowledge paths (patterns) can be identified based on the edge frequencies in the graph. The paths represent the worker's frequent knowledge referencing behavior and important knowledge flows in the group. Finally, to demonstrate the efficacy of our proposed method, we implement a prototype system for mining the GKF of a group of workers. The system provides useful functions that allow users to simplify the complexity of KF mining and visualize KFs graphically.

1.4 Organization of the proposal

The remainder of this proposal is organized as follows. Chapter 2 provides a brief overview of related works. In Chapter 3, we describe the knowledge flow model, the overview of knowledge flow-based research and the knowledge flow mining phase. The knowledge flow-based recommendation framework is illustrated in Chapter 4. The group-based knowledge flow mining methods are illustrated in Chapter 5. According to these methods, we propose a prototype system for mining the group-based knowledge flow. Finally, in Chapter 6, we summarize our conclusions and consider future research directions.

Chapter 2. Related Work

In this chapter, we discuss the background of our research, including knowledge flow, information retrieval and task-based knowledge support, document clustering methods, dynamic programming algorithm, rule-based recommendations, collaborative filtering and process mining.

2.1 Knowledge Flow

Knowledge can flow among people and processes to facilitate knowledge sharing and reuse. The concept of knowledge flow has been applied in various domains, e.g., scientific research, communities of practice, teamwork, industry, and organizations [33, 63]. Scholarly articles represent the major medium for disseminating knowledge among scientists to inspire new ideas [8, 63]. A citation implies that there is knowledge flow between the citing article and the cited article. Such citations form a knowledge flow network that enables knowledge to flow between different scientific projects to promote interdisciplinary research and scientific development.

KM enhances the effectiveness of teamwork by accumulating and sharing knowledge among team members to facilitate peer-to-peer knowledge sharing [61]. To improve the efficiency of teamwork, Zhuge [62] proposed a pattern-based approach that combines codification and personalization strategies to design an effective knowledge flow network. Kim *et al.* [33] proposed a knowledge flow model combined with a process-oriented approach to capture, store, and transfer knowledge. KF in weblogs (blogs) is a communication pattern where the post of one blogger links to that of another blogger to exchange knowledge [8]. Similarly, knowledge flow in communities of practice helps members share their knowledge and experience about a specific domain to complete their tasks [46].

2.2 Information Retrieval and Task-based Knowledge Support

Information retrieval (IR) facilitates access to specific items of information [10, 21]. The vector space model [48] is typically used to represent documents as vectors of index terms, where the weights of the terms are measured by the *tf-idf* approach. *tf* denotes the occurrence frequency of a particular term in the document, while *idf* denotes the inverse document frequency of the term. Terms with higher *tf-idf* weights are used as discriminating terms to filter out common terms. The weight of a term i in a document j, denoted by $w_{i,j}$, is expressed as follows:

$$w_{i,j} = tf_{i,j} \times idf_i = tf_{i,j} \times (\log_2 \frac{N}{n} + 1), \qquad (1)$$

where $tf_{i,j}$ is the frequency of term i in document j, idf_i is measured by $(\log_2 N/n) + 1$, N is the total number of documents in the collection, and n is the number of documents in which term i occurs at least once.

Information retrieval techniques coupled with workflow management systems (WfMS) have been used to support proactive delivery of task-specific knowledge based on the context of tasks within a process [2]. For example, the *KnowMore* system [1] provides context-aware delivery of task-specific knowledge. The *Kabiria* system assists knowledge workers with knowledge-based document retrieval by considering the operational context of task-associated procedures [9].

Information filtering with a similarity-based approach is often used to locate knowledge items relevant to the task-at-hand. The discriminating terms of a task are usually extracted from a knowledge item/task to form a task profile, which is used to model a worker's information needs. Holz *et al.* [27] proposed a similarity-based approach to organize desktop documents and proactively deliver task-specific information. Liu *et al.* [39] proposed a *K-Support* system to provide effective task support for a task-based working environment.

2.3 Document Clustering Methods

Document clustering or unsupervised document classification methods are used in many applications. Most methods apply pre-processing steps to the document set and represent each document as a vector of index terms. To cluster similar documents, the similarity between documents is usually measured by the cosine measure [10, 57], which computes the cosine of the angle between their corresponding feature vectors. Two documents are considered similar if the cosine similarity value is high. The cosine similarity of two documents, X and Y, is $simcos(X, Y) = \frac{\vec{X} \cdot \vec{Y}}{\|\vec{X}\|\|\vec{Y}\|}$, where $\vec{X}$ and $\vec{Y}$ are the feature vectors of X and Y respectively.

Documents within a cluster are very similar, while documents in different clusters are very dissimilar.

Agglomerative hierarchical clustering [30, 32] is a popular document clustering method. In this work, we use the single-link clustering method [20, 29] to cluster codified knowledge (documents). Initially, each document is regarded as a cluster. Next, the single-link method computes the similarity between two clusters, which is equal to the greatest similarity between any document in one cluster and any document in the other cluster. Then, based on the similarity measurement, the two most similar clusters are merged to form a new cluster. The merging process continues until all documents have been merged into one cluster at the top of a hierarchy, or a pre-specified threshold is satisfied [29].

2.3.1 The CLIQUE Clustering Method

We also apply the CLIQUE clustering method [6, 29] to derive worker groups. CLIQUE starts with the definition of a unit-elementary rectangular cell in a subspace and uses a bottom-up approach to find units whose densities exceed a threshold. The algorithm has four key steps. First, 1-dimensional units are determined by dividing intervals into equal-width bins (a grid). Next, candidate k-dimensional units are generated from (k-1)-dimensional dense units, which involves self-joining of k-1 units that have common k-2 dimensions (Apriori-reasoning). Finally, all the subspaces are sorted by their coverage and those with less coverage are pruned. Therefore, a cluster is defined as a maximal set of connected dense units.

2.3.2 Clustering Quality

A good clustering method generates clusters that are cohesive and isolated from other clusters. For this reason, the measurement of clustering quality takes both inter-cluster similarity and intra-cluster similarity into account [16]. Let C be a set of clusters. The inter-cluster similarity between two clusters C_i and C_j, $similarity_A(C_i, C_j)$, is defined as the average of all pairwise similarities between the documents in C_i and C_j; and the intra-cluster similarity within a cluster C_i, $similarity_A(C_i, C_i)$, is defined as the average of all pairwise similarities between documents in C_i. On the basis of the cohesion and isolation of C, the quality measure of C , $CQ(C)$, is defined as:

$$CQ(C) = \frac{1}{|C|} \sum_{C_i \in C} \frac{similarity_A(C_i, \overline{C_i})}{similarity_A(C_i, C_i)}, \text{ where } \overline{C_i} = \cup_{i \neq j} C_j . \tag{2}$$

Note that the smaller the value of CQ(C), the better the quality of the derived set of clusters, C, will be.

2.4 Dynamic Programming Algorithm for Sequence Alignment

In this work, each worker's knowledge flow is represented as a sequence. We use sequence alignment techniques to analyze the similarity of workers' knowledge flows, which corresponds to a sequence alignment problem. Such techniques are used to compare or align strings in many application domains, such as biology, speech recognition, and web session clustering. A number of methods can be used for sequence alignment, e.g., the sequence alignment method (SAM) [14, 24] and dynamic programming. SAM, also called the string edit distance method [35], considers the sequential order of elements in a sequence and then measures the similarity/dissimilarity of sequences. The measurements reflect the operations necessary to equalize the sequences by computing the costs of deleting and inserting unique elements as well as the costs of reordering common elements [24, 41]. In addition, Charter *et al.* [14] proposed a dynamic programming algorithm that solves the sequence alignment problem efficiently.

The algorithm consists of three steps: initialization, *FindScore* and *FindPath* [14, 43]. The first step creates a dynamic programming matrix with N+1 columns and M+1 rows, where N and M correspond to the sizes of the sequences to be aligned. One sequence is placed at the top of the matrix and the other is placed on the left-hand side of the matrix. There is a gap at the end of each sequence to allow calculation of the alignment score. The *FindScore* step calculates the two-dimensional alignment score of sequences. If two aligned sequences have an identical matching in the same column, the column is given a positive score s (e.g., +1 or +2); but if the values in a column are mismatches, the score s is zero or negative (e.g., 0, -1 or -2). In addition, if a column contains a gap, it is given a penalty score w (e.g., 0, -1 or -2). Therefore, starting from the bottom right-hand corner, each position in the dynamic programming matrix is given the maximal score M_{ij}. For each position in the matrix, M_{ij} is defined as follows:

$$M_{ij} = Maximum\{(M_{i-1,j-1} + s_{ij}), (M_{i,j-1} + w), (M_{i-1,j} + w)\}, \tag{3}$$

where i is the row number, j is the column number, s_{ij} is the match/mismatch score, and w is the penalty score. The third step, *FindPath,* determines the actual KF alignment that derives the maximal score. It traverses the matrix from the destination point (top left-hand corner) to the starting point (bottom right-hand corner) to find an optimal alignment path in order to determine the maximal alignment score δ. We calculate the flow similarity based on the maximal alignment score. The details are given in Section 4.2.

2.5 Rule-based Recommendations

Association rule mining [3-4, 59] is a widely used data mining technique that generates recommendations in recommender systems. An association rule describes the relationships between items, such as products, documents, or movies, based on patterns of co-occurrence across transactions. The Apriori algorithm [3-4] is usually employed to identify such rules. Two measures, support and confidence, are used to indicate the quality of an association rule [3]. The discovered rules should satisfy two user-defined requirements, namely minimum support and minimum confidence.

To improve the quality of traditional CF, Cho *et al.* [15] proposed a sequential rule-based recommendation method that considers the evolution of customers' purchase sequences. Transactions are clustered into a set of q transaction clusters, $C=\{C_1,C_2,\ldots,C_q\}$, where each C_j is a subset of transactions. Each customer's transactions over l periods are then transformed into transaction clusters as a behavior locus, $L_i = <C_{i,T-l-1},\ldots C_{i,T-1}, C_{i,T}>$, where $C_{i,T-k} \in C$, k=1,2,…,l-1, $l \geqq 2$. Finally, sequential purchase patterns are extracted from the behavior locus of customers by time-based association rule mining to keep track of customers' preferences during l periods, with T as the current (latest) period. A sequential rule is expressed in the form $C_{T-l+1}, \ldots, C_{T-1} \Rightarrow C_T$, where C_T represents the customers' purchase behavior in period T. If a target customer's purchase behavior prior to period T was similar to the conditional part of the rule, then it is predicted that his/her purchase behavior in period T will be C_T. Accordingly, C_T is used to recommend products to the target customer in T.

2.6 Collaborative Filtering Recommendation

Collaborative filtering (CF) is a well-known approach for recommender systems: GroupLens [34], Ringo [51], Siteseer [47], and Knowledge Pump [22]. CF recommends items, e.g., products, movies, and documents, based on the preferences of people who have the same or similar interests to those of the target user [11, 38, 40]. The CF approach involves two steps: neighborhood formation and prediction. The neighborhood of a target user is selected according to his/her similarity to other users, and is computed by Pearson correlation coefficient or the cosine measure. Either the k-NN (nearest neighbor) approach or a threshold-based approach is used to choose *n* users that are most similar to the target user. Here, we use the k-NN approach. In the prediction step, the predicted rating is calculated from the aggregated weights of the selected *n* nearest neighbors' ratings, as shown in Eq. (4):

$$P_{u,j} = \bar{r}_u + \frac{\sum_{i=1}^{n} w(u,i)\left(r_{i,j} - \bar{r}_i\right)}{\sum_{i=1}^{n} \left|w(u,i)\right|}, \tag{4}$$

where $P_{u,j}$ denotes the prediction rating of item *j* for the target user *u;* $\bar{r}_u$ and $\bar{r}_i$ are the average ratings of user *u* and user *i,* respectively; *w(u,i)* is the similarity between target user *u* and user *i*; $r_{i,j}$ is the rating of user *i* for item *j*; and *n* is the number of users in the neighborhood.

Similar to the PCF method, the item-based collaborative filtering (ICF) algorithm [37, 40, 50] analyzes the relationships between items (e.g., documents) first, rather than the relationships between users. Then, the item relationships are used to compute recommendations for workers indirectly by finding items that are similar to other items the worker has accessed previously. Thus, the prediction for an item *j* for a user *u* is calculated by the weighted sum of the ratings given by the user for items similar to *j* and weighted by the item similarity, as shown in Eq. (5).

$$p_{u,j} = \frac{\sum_{m=1}^{n} w(j,m) \times r_{j,m}}{\sum_{m=1}^{n} \left|w(j,m)\right|}, \tag{5}$$

where $p_{u,j}$ represents the predicted rating of item *j* for user *u*; *w*(*j*,*m*) is the similarity between two items *j* and *m*; and $r_{j,m}$ denotes the rating of user *u* for item *m*. A number of methods can be used to determine the similarity between items e.g., the cosine-based similarity, correlation-based similarity, and adjusted cosine similarity methods. Since the adjusted cosine similarity method performs better than the others [50], we use it as the similarity measure for the ICF method. The adjusted cosine similarity between two items *i* and *j* is given by Eq. (6).

$$sim(i,j) = \frac{\sum_{u \in U}(r_{u,i} - \bar{r}_u)(r_{u,j} - \bar{r}_u)}{\sqrt{\sum_{u \in U}(r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_{u \in U}(r_{u,j} - \bar{r}_u)^2}}, \qquad (6)$$

where $r_{u,i}$ / $r_{u,j}$ is the rating of item *i*/*j* given by user *u*; and $\bar{r}_u$ is the average item rating of user *u*.

2.7 Process Mining

In a workflow system, a process mining technique is used to extract the description of a structural process from a set of real process executions [54]. It then infers the relations between the tasks/activities and generates a process model from event-based data (log data) automatically [7, 53, 55-56]. The relations between processes (tasks/activities) are defined as casual relations and parallel relations, and are modeled by a directed graph [7, 23] or an instance graph [56]. Because a workflow log contains information about workflow processes, a loop may occur in a process. Most process mining algorithms assume that loops do not exist [23, 56]. However, some algorithms have been proposed to handle the problem of process loops [18, 54]. For example, Agrawal, *et al.*'s algorithm [7] builds a general directed graph with cycles for mining process models from the logs of executed processes. The algorithm labels multiple instances of the same activity with different identifies to differentiate them in the workflow graph. Vertices with different instances of the same activity form an equivalent set and can be merged to form one vertex. A directed edge is added if there is an edge between two vertices of different equivalent sets.

Process mining is used in various applications. Discovering frequently occurring temporal patterns in process instances facilitates intelligent and automatic extraction of useful knowledge to support business decision-making [7, 28]. Similarly, data mining techniques are exploited in workflow management contexts to mine frequent workflow execution patterns [23]. The frequent patterns represent blocks of activities that have been scheduled together more frequently during the execution of a process. The sequence of activities within a process, the time required to complete it, the execution cost and the reliability of the process can be predicted by using the process path mining technique [13]. Based on the process patterns and process paths, unexpected and useful knowledge about the process is extracted to help the user make appropriate decisions. In addition, combining the concepts of process mining and social network analysis is useful for mining social networks from event logs [52].

Another benefit of process mining is that it is useful for discovering how people and/or procedures work [54]. In this work, we use process mining to analyze the relations between knowledge topics in a knowledge flow and model the referencing behavior of a group of workers. We design algorithms for mining the group-based knowledge flow (GKF) and construct a GKF as a directed knowledge graph. In such graphs, frequent knowledge paths can be derived to represent the most common referencing behavior of the group.

Chapter 3. The Overview of Knowledge Flow-Based Research

3.1 Knowledge Flow Model

In a knowledge-intensive and task-based environment, workers may need to access a large number of documents (codified knowledge) to accomplish a task. From the perspective of information needs, a worker's knowledge flow (KF) represents the evolution of his/her information needs and preferences during a task's execution. Workers' KFs are identified by analyzing their knowledge referencing behavior based on their historical work logs, which contain information about previously executed tasks, task-related documents and when the documents were accessed.

A KF consists of two levels: a codified level and a topic level, as shown in Fig. 1. The knowledge in the codified-level indicates the knowledge flow between documents based on the access time. In most situations, the knowledge obtained from one document prompts a knowledge worker to access the next relevant document (codified knowledge). Hence, the task-related documents are sorted by their access time to obtain a document sequence as the codified-level KF.

Documents with similar concepts can be grouped together automatically to form a topic-level abstraction of knowledge. Note that each topic may contain several task-related documents. The codified-level KF can be abstracted to form a topic-level KF, which represents the transitions between various topics. Since the task knowledge in the topic level may flow among topics, it could prompt the worker(s) to retrieve knowledge from the next related topic. Formally, we define knowledge flow as follows.

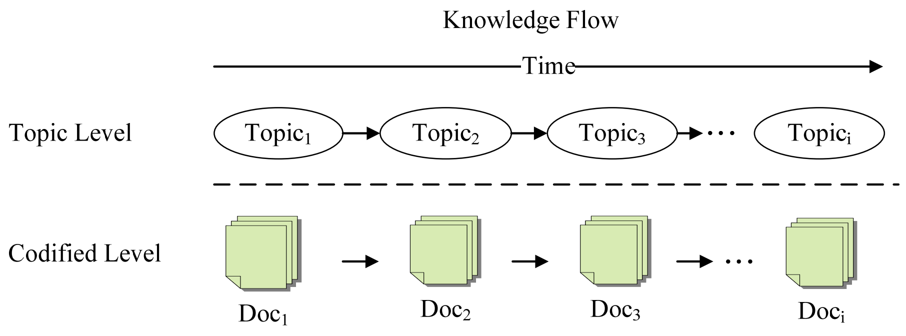


Fig. 1: The two levels of a knowledge flow

Definition 1: Knowledge Flow (KF)

Let a worker's knowledge flow be $KFlow_w^v = \{TKF_w^v, CKF_w^v\}$, where TKF_w^v is the topic-level KF of the worker *w* for a task *v*, and CKF_w^v is his/her codified-level KF for the task *v*.

Definition 2: Codified-Level KF

A codified-level KF is a time-ordered sequence arranged according to the access times of the documents it contains. Thus, it is defined as $CKF_w^v = < d_{w,}^{t_1}, d_w^{t_2}, \cdots, d_w^{t_f} >$ and $t_1 < t_2 < \cdots < t_f$, where $d_w^{t_j}$ denotes the document that the worker *w* accessed at time t_j for a specific task *v*. Each document can be represented by a document profile, which is an *n*-dimensional vector containing weighted terms that indicate the key content of the document.

Definition 3: Topic-Level KF

A topic-level KF is a time-ordered topic sequence derived by mapping documents in the codified-level KF to corresponding topics. Thus, it is defined as $TKF_w^v = < TP_w^{t_1}, TP_w^{t_2}, \cdots, TP_w^{t_f} >$, $t_1 < t_2 < \cdots < t_f$, where $TP_w^{t_j}$ denotes the corresponding topic of the document that worker *w* accessed at time t_j for a specific task *v*. Each topic is represented by a topic profile, which is an *n*-dimensional vector containing weighted terms that indicate the key content of the topic.

3.2 The Framework of Knowledge Flow-based Approaches

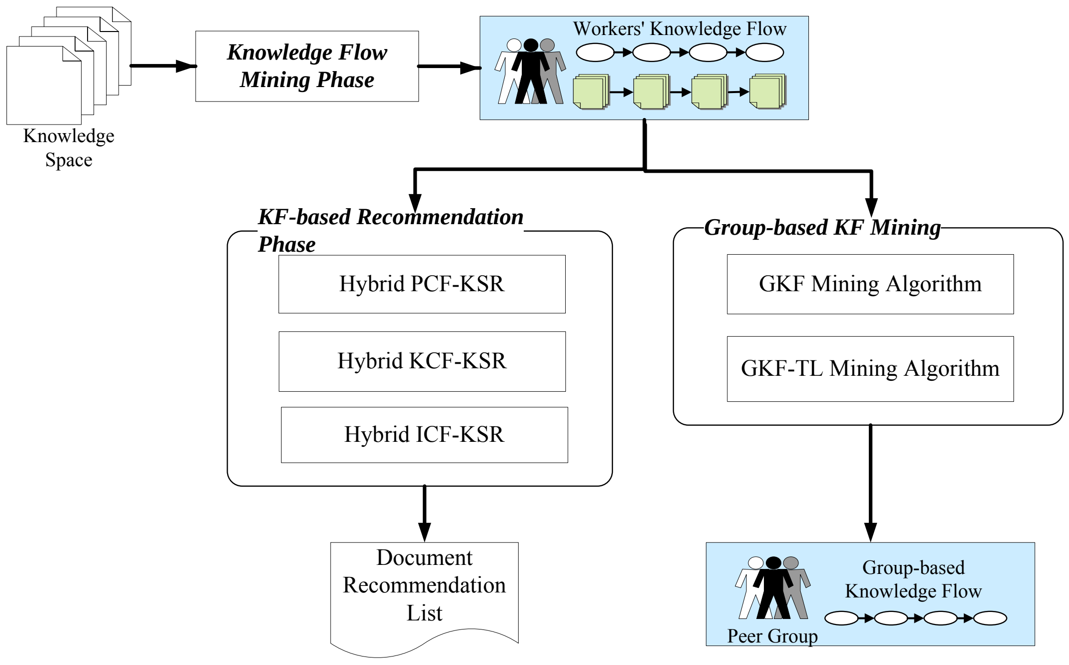


Fig. 2: The overview of knowledge flow-based research

Fig. 2 illustrates the overview of our research which is knowledge flow-based approaches for providing knowledge support. According to the definition of knowledge flow, the knowledge flow mining is used to identify both topic-level and codified-level KF of each knowledge worker based on their log data which consists of the access behavior of task-related documents. Then, based on the discovered KF, our research is divided into two parts: KF-based recommendation phase and group-based KF mining methods.

The KF-based recommendation phase selects and recommends documents based on document preferences and knowledge flows derived from the target worker's neighbors. In other words, the proposed recommendation methods trace a worker's information needs by analyzing his/her knowledge referencing behavior for a task over time, and also proactively provide relevant codified knowledge for the worker based on the KFs of the worker's neighbors.

According to the KF mining approach [36], we propose the group-based KF mining algorithms that integrate information retrieval and data mining techniques for mining and constructing the group-based knowledge flows (GKFs). Specifically, we discover a group's KF from the KFs of the participating workers and identify the frequent referencing behavior of a group of workers. Then, the concepts of graph theory are applied to visualize the GKF as a knowledge graph. The paths on such graph represent the workers' frequent knowledge referencing behavior and important knowledge flows in the group. Section 3.3 describes the details of knowledge flow mining first. Then, the two parts of our research are based on the mining results and are illustrated in Chapter 4 and Chapter 5.

3.3 Knowledge Flow Mining Phase

The objective of the knowledge flow (KF) mining phase is to identify the KF of each knowledge worker. In this Section, we describe how the KF mining method identifies KFs from workers' log. This phase consists of three steps: document profiling, document clustering and KF extraction. In the first step, each document is represented as a document profile, which is an *n*-dimensional vector comprised of significant terms and their weights. Then, based on the document profiles, documents with higher similarity measures are grouped in clusters by the hierarchical clustering method. In the third step, topic-level and codified-level KFs are generated from the document clustering results. A topic-level KF is expressed as a sequence of topics referenced by a worker, while a codified-level KF is represented as a sequence of codified knowledge accessed by a worker. Further details are given in the following subsections.

3.3.1 Document Profiling and Document Clustering

Two profiles, a document profile and a topic profile, are used to represent a worker's KF. A document profile can be represented as an *n*-dimensional vector composed of terms and their respective weights derived by the normalized *tf-idf* approach based on Eq. (1). Based on the term weights, terms with higher values are selected as discriminative terms to describe the characteristics of a document. The document profile of d_j is comprised of these discriminative terms. Let the document profile be $DP_j =< dt_{1j} : dtw_{1j}, dt_{2j} : dtw_{2j}, \cdots, dt_{nj} : dtw_{nj} >$, where dt_{ij} is the term i in d_j and dtw_{ij} is the degree of importance of a term i to the document d_j, which is derived by the normalized *tf-idf* approach. The document profiles are used to measure the similarity of the documents.

We adopt the single-link hierarchical clustering method [29] to group documents with similar profiles into clusters by using the cosine measure to calculate the similarity between the profiles of two documents. The single-link method computes the cluster similarity between two clusters C_r and C_t by $\max_{d_i \in C_r, d_j \in C_t} \left\{ simcos\left(d_i, d_j\right) \right\}$ [60], and then merges the two most similar clusters into a single cluster. The similarity computation and cluster combination steps are repeated until the similarity of the most similar pair of clusters is no greater than a pre-specified threshold value. Different clustering results can be obtained by setting different threshold values. We adjust the threshold value systematically and use the quality measure described in Section 2.3.2 to evaluate each clustering result. Then, we take the one with the best quality measure as our clustering result. Note that a cluster represents a topic set and has a topic profile (derived from the document cluster) that describes the features of the topic.

Topic Profile

Documents in the same cluster contain similar content and form a topic set. The key features of the cluster are described by a topic profile, which is derived from the profiles of documents that belong to the cluster. Let $TP_x = < tt_{1x} : ttw_{1x}, tt_{2x} : ttw_{2x}, \cdots, tt_{nx} : dtw_{nx} >$ be the profile of a topic (cluster) x, where tt_{ix} is a topic term and ttw_{ix} is the weight of the topic term. In addition, let D_x be the set of documents in cluster x. The weight of a topic term is determined by Eq. (7) as follows:

$$ttw_{ix} = \frac{\sum_{j \in D_x} dtw_{ij}}{\left|D_x\right|}, \tag{7}$$

where dtw_{ij} is the weight of term i in document j, and $|D_x|$ is the number of documents in cluster x. The weight of a topic term is obtained from the average weight of the terms in the document set.

3.3.2 Knowledge Flow Extraction

In this section, we describe the method used to extract a worker's KF from his/her data log when performing a task. We define a task as a unit of work, which denotes either a previously executed (i.e., historical) task or the current task. When performing a task in a knowledge-intensive and task-based environment, a worker usually requires a large amount of task-related knowledge to accomplish the task. By analyzing a worker's referencing behavior for a specific task, the corresponding knowledge flow of the task is derived by the knowledge flow extraction method. Note that if a worker performs more than one task, more than one knowledge flow will be extracted. For a specific task, the method derives two kinds of KF, *codified-level KF* and *topic-level KF*, to represent the worker's information needs for the task.

Codified-Level Knowledge Flow

The codified-level KF is extracted from the documents recorded in the worker's work log. In most situations, workers are motivated to access a document about a specific task because of knowledge derived from other documents. The documents are arranged according to the times they were accessed, and a document sequence, i.e., a codified-level KF, is obtained. The order of documents in the sequence is subjective, since it is determined by the worker. In other words, each worker has his/her own codified-level KF, which represents his/her knowledge accumulation process for a specific task at the codified level.

Topic-Level Knowledge Flow

The topic-level KF is derived by mapping documents in the codified-level KF of a specific task into corresponding clusters and is represented by a topic sequence. In the previous step, documents with similar content were grouped into clusters. We use the document clustering results to map the documents in the codified-level KF into topics (clusters) in order to compile the topic-level KF. Since the codified-level KF is the basis of the topic-level KF, the knowledge in the latter is an abstraction of the former, and indicates how knowledge flows among various topics. A topic in the topic-level KF may be duplicated because the worker may read about the same topic frequently to obtain essential knowledge while executing a task.

Chapter 4. Knowledge Flow-based Recommendation Framework

The proposed recommendation methods are illustrated in Fig. 3. Our methods consist of two phases, a knowledge flow mining phase and a KF-based recommendation phase. The first phase identifies the worker's knowledge flow from the large amount of knowledge in the worker's log. Then, the second phase recommends codified knowledge to the target worker by using the proposed recommendation methods.

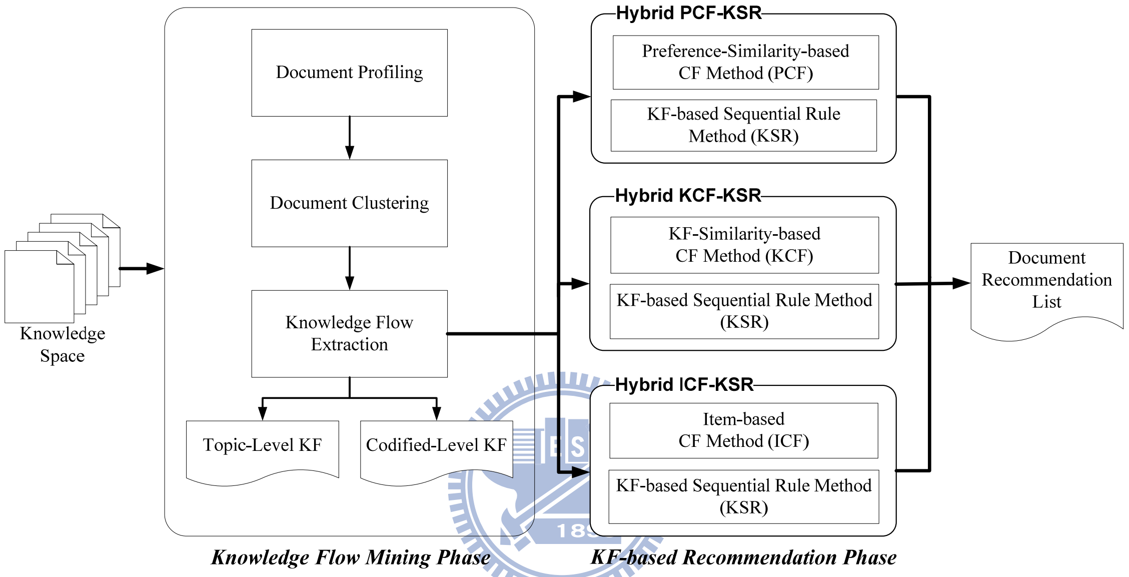

Knowledge Space
Document Profiling
Document Clustering
Knowledge Flow Extraction
Topic-Level KF
Codified-Level KF
Knowledge Flow Mining Phase
Hybrid PCF-KSR
Preference-Similarity-based CF Method (PCF)
KF-based Sequential Rule Method (KSR)
Hybrid KCF-KSR
KF-Similarity-based CF Method (KCF)
KF-based Sequential Rule Method (KSR)
Hybrid ICF-KSR
Item-based CF Method (ICF)
KF-based Sequential Rule Method (KSR)
KF-based Recommendation Phase
Document Recommendation List

Fig. 3: Document recommendation based on knowledge flows

In the knowledge flow mining phase, KFs are identified from the task requirements and the referencing behavior of workers recorded in their logs. As tasks are performed at various times, each knowledge worker requires different kinds of knowledge to achieve a goal or complete a task. Further details about this phase are given in Section 3.3.

The proposed hybrid recommendation methods combine a KF-based sequential rule (KSR) method with a user-based/item-based collaborative filtering (CF). The KSR method is regarded as the core process of the proposed hybrid methods. In the KSR method, workers with similar KFs to that of the target worker are deemed neighbors of the target worker and their knowledge referencing behavior patterns are identified by a sequential rule mining method. Based on the discovered sequential rules and the neighbors' KFs, relevant topics and codified knowledge are recommended to the target worker to support the task-at-hand. Moreover, by considering workers' preferences for codified knowledge, the CF method makes recommendations to the target worker based on the opinions of similar workers. Three approaches are used to find similar workers to the target worker. The preference-similarity-based CF method (PCF) chooses workers with similar preferences, while the KF-similarity-based CF method (KCF) chooses workers with similar KFs. Different from these two user-based methods, the item-based CF method predicts a document rating based on its similar documents that have been rated by a target user. To adaptively and proactively recommend codified knowledge, we consider workers' referencing behavior as well as their preferences for codified knowledge. Therefore, three hybrid recommendation methods are used in the KF-based recommendation phase: 1) a hybrid of PCF and KSR (PCF-KSR), 2) a hybrid of KCF and KSR (KCF-KSR) and 3) a hybrid of ICF and KSR (ICF-KSR). Further details are given in the following subsections.

4.1 Knowledge Flow-based Recommendation Phase

In this work, we propose three hybrid recommendation methods based on knowledge flow (KF), which is a sequence of codified knowledge (documents) or topics referenced by a worker during a task's execution. KF represents a worker's information needs and the evolution of knowledge requirements, and is identified by analyzing a worker's work log. To support workers effectively, our methods consider workers' preferences as well as their referencing behavior in order to recommend task-related knowledge. During the recommendation phase, the user-based collaborative filtering (CF) is used to predict a target worker's preferences based on the opinions of similar workers, while the item-based collaborative filtering [50] is used to predict a document based on the targets worker's interests on its similar items (documents). However, the limitation of these traditional CF methods is that they only consider workers' preferences for codified knowledge and neglect workers' referencing behavior. A worker's referencing behavior may change during the task's execution to suit his/her current information needs. To address this issue, we propose a KF-based sequential rule method that improves the recommendation quality by tracking workers' referencing behavior based on sequential rules. However, this method does not consider the opinions of the target worker's neighbors who have similar preferences for documents. To overcome the limitations of CF and KF-based sequential rule methods, we combine the advantages of the two approaches and propose three hybrid recommendation methods that integrate KF mining, KF-based sequential rule mining and CF techniques to enhance the quality of recommendations.

The KF-based recommendation phase consists of three hybrid recommendation methods: 1) PCF and KSR (PCF-KSR), 2) KCF and KSR (KCF-KSR) and 3) ICF and KSR (ICF-KSR), as shown in Fig. 3. We note that PCF denotes the preference-similarity based CF method; KCF denotes the KF-similarity based CF method; ICF denotes the item-based CF method; and KSR denotes the KF-based sequential rule method. To adaptively recommend documents, both the PCF method and the KCF method select neighbors based on the similarity of preferences, while the ICF method chooses similar documents for a document based on their preferences given by a target user. The three methods differ in the way they compute the similarity between workers' preferences to select the target worker's neighbors. The PCF method (traditional CF) uses preference ratings to compute the similarity, while the KCF method uses workers' KFs to derive the similarity. The ICF method applies similarity measure to evaluate the similarity between two items (i.e., documents), rather than the similarity between two workers. The proposed KSR method traces workers' knowledge referencing behavior by using the KF-based sequential rules. The proposed hybrid recommendation methods take advantage of the merits of the KSR, PCF, KCF and ICF methods.

4.2 Identifying Similar Workers Based on their Knowledge Flows

To find a target worker's neighbors, his/her topic-level KF is compared with those of other workers to compute the similarity of their KFs. The resulting similarity measure indicates whether the KF referencing behavior of two workers is similar. In this work, we regard each knowledge flow as a sequence. Since comparing knowledge flows is very similar to aligning sequences, the sequence alignment method (SAM) [24] and the dynamic programming approach [14, 43] can be used to measure the similarity of two KF sequences.

To determine which of the two methods would be more appropriate for comparing workers' knowledge flows, we applied both methods in our experiments and found that dynamic programming is better than SAM. Therefore, we employ the dynamic programming algorithm [14, 43] to measure the similarity of workers' knowledge flows.

Unlike the sequence alignment problem, a worker's KF contains task-related documents. Thus, we have to consider the sequential order of topics in a knowledge flow, as well as the worker's aggregated profile, which accumulates the task-related documents based on the times they were accessed during the task's execution. We propose a hybrid similarity measure, comprised of the KF alignment similarity and the aggregated profile similarity, to evaluate the similarity of two workers' KFs, as shown in Eq. (8).

$$sim(TKF_i^v, TKF_j^l) = \alpha \times sim_a(TKF_i^v, TKF_j^l) + (1-\alpha) \times sim_P(AP_i^v, AP_j^l), \qquad (8)$$

where $sim_a(TKF_i^v, TKF_j^l)$ represents the KF alignment similarity between worker i and worker j who execute task v and task l respectivel TKF_i^v / TKF_j^l is the topic-level KF of worker *i/j* for task *v/l*; $sim_p(AP_i^v, AP_j^l)$ represents the aggregated profile similarity of two workers' KF AP_i^v AP_j^l is the aggregated profile of worker *i/j* for task *v/l*; and α is a parameter used to adjust the relative importance of the two types of similarity.

The KF alignment similarity is based on the topic sequence and topic coverage, while the aggregated profile similarity is based on the aggregated profiles derived from the profiles of referenced documents in the KFs. Note that the KF alignment similarity considers the topic sequence in the KF without considering the content of workers' profiles; while the aggregated profile similarity considers the content of profiles without considering the topic sequence in the KF. By linearly combining these two similarities, we can balance the tradeoff between KF alignment and the aggregated profile. We discuss the rationale behind these two similarity measures next.

4.2.1 KF Alignment Similarity

The KF alignment similarity is comprised of two parts: the KF alignment score, which measures the topics in sequence; and the join coefficient, which estimates the topic's coverage in two compared topic-level KFs. We modify the sequence alignment method [14] to derive the KF alignment score. In addition to computing the sequence alignment score, we estimate the overlap of the topics in two compared topic-level KFs by using the join coefficient. The rationale is that if the topic overlap is high, the KF alignment similarity of the two compared KFs will also be high. In other words, the two compared KFs will be very similar. The KF alignment similarity, $sim_a(TKF_i^v, TKF_j^l)$, is defined as follows:

$$sim_a(TKF_i^v, TKF_j^l) = Norm(\eta) \times \frac{2 \times \left|TPS_i^v \cap TPS_j^l\right|}{\left|TPS_i^v\right| + \left|TPS_j^l\right|}, \tag{9}$$

where TKF_i^v / TKF_j^l denotes the topic-level KF of worker *i*/ worker *j* for task *v*/ task *l*; η is the KF alignment score; *Norm* is a normalization function used to transform the value of η into a number between 0 and 1; TPS_i^v and TPS_j^l are the sets of topics in TKF_i^v and TKF_j^l respectively; $TPS_i^v \cap TPS_j^l$ is the intersection of topics common to TKF_i^v and TKF_j^l; and $\left|TPS_i^v\right|$ and $\left|TPS_j^l\right|$ represent the number of topics in TKF_i^v and TKF_j^l respectively. The KF alignment score, which is based on the sequence alignment method [43], is defined in Eq. (10):

$$\eta = \frac{\delta}{m_s \times \xi}, \tag{10}$$

where δ is the maximal alignment score derived by the dynamic programming approach, m_s is the identical matching score (+2), and ξ is the length of the aligned KF. To obtain the maximal alignment score δ, we set the matching score m_s, the mismatching score m_d and the gap penalty score m_g to +2, -1 and -2 respectively in the dynamic programming approach [14] discussed in Section 2.4. The maximum value of η is 1 if the two compared KFs are exactly the same. On the other hand, the value of η is negative if most of topics in the two compared KFs do not match. Thus, the value of η may range from a negative value to 1. To alter the range of the KF alignment score, the value of η is transformed into a value in the range [0, 1] by the normalization function. The normalized KF alignment score *Norm*(η) is then used to calculate the KF alignment similarity.

4.2.2 Aggregated Profile Similarity

The aggregated profile similarity, defined as $sim_p(AP_i^v, AP_j^l)$, computes the similarity of two workers' KFs based on their aggregated profiles, which are derived from the profiles of documents they have referenced; AP_i^v and AP_j^l are the respective vectors of the aggregated profiles of workers *i*/ *j* for task *v*/ *l*. We use the cosine formula to calculate the similarity between two aggregated profiles. The value of the similarity score ranges from 0 to 1. The aggregated profile of a worker *i* for task *v* is defined as

$$AP_i^v = \sum_{t=1}^{T} tw_{t,T} \times DP_t^v \,, \tag{11}$$

where $tw_{t,T}$ is the time weight of the document referenced at time *t* in the KF; *T* is the index of the times the worker accessed the most recent documents in his KF; and DP_t^v is the profile of the document referenced by worker *i* at time *t* for task *v*. The aggregation process considers the time decay effect of the documents. Each document profile is assigned a time weight according to the time it was referenced. Thus, higher time weights are given to documents referenced in the recent past. The time weight of each document profile is defined as $tw_{t,T} = \frac{t - St}{T - St}$, where *St* is the start time of the worker's KF.

4.3 KF-based Sequential Rule Method

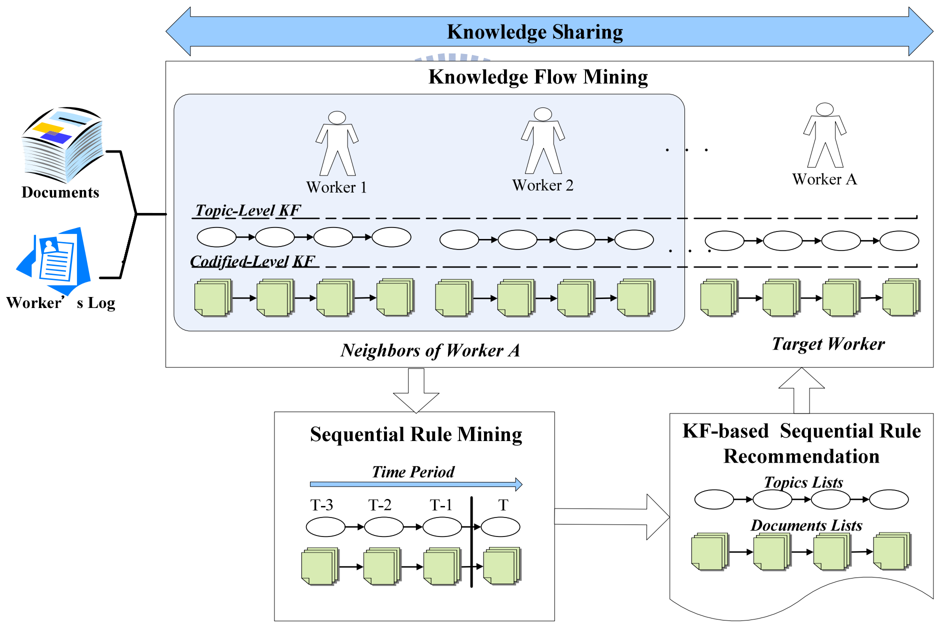


Fig. 4: An overview of the KSR method

The KF-based sequential rule method (KSR) considers the referencing behavior of neighbors whose KFs were very similar before time *T*, and then recommends documents at time *T* for the target worker. Fig. 4 provides an overview of the KSR method. To determine the similarity of various topic-level KFs, the target worker's KF is compared with those of other workers by measuring their KF similarity, as discussed in Section 4.2. Workers with similar KFs to that of the target worker are regarded as the latter's neighbors and their topic-level KFs are used to discover frequent knowledge referencing behavior by applying sequential rule mining to the target worker's referencing behavior. The discovered sequential rules with high degrees of rule matching are selected to recommend topics at time *T*. Documents belonging to the recommended topics have a high priority of being recommended. The KSR recommendation method involves four steps: identifying similar workers, mining their knowledge referencing behavior, identifying the target worker's knowledge referencing behavior, and document recommendation.

4.3.1 Mining Knowledge Referencing Behavior

Knowledge workers with similar referencing behavior (high similarities) of the target worker are regarded as neighbors of the target worker. We modify the association rule mining method [3-4] and sequential pattern mining method [5] to discover topic-level sequential rules from the neighbors' topic-level KFs. The extracted rules can be used to keep track of the referenced topics among workers with similar referencing behavior. Let R_y be a sequential rule, as defined in Eq. (12).

$$\begin{gathered} R_y\text{: } g_{y,T-s},\ldots, g_{y,T-1} \Rightarrow g_{y,T} \quad (Support_y,\ Confidence_y) \\ \text{where } g_{y,T-f} \in TPS;\ f = 0 \text{ to } s;\ \text{and } TPS \text{ is a set of all topics} \end{gathered} \tag{12}$$

The conditional part of the sequential rule is $<g_{y,T-s},\ldots,g_{y,T-1}>$, and the consequent part is $g_{y,T}$. The items that appear in the rules are topics extracted from the neighbors' topic-level KFs (*TKF*). The support and confidence values, $Support_y$ and $Confidence_y$, are used to evaluate the importance of rule R_y. We use the support and confidence scores to measure the degree of match between the referencing behavior and the conditional part of a rule for a target worker, as illustrated in the third step. Note that if the knowledge referencing behavior of the target worker is similar to the conditional part of R_y, then the topic predicted for him/her at *T* will be $g_{y,T}$.

4.3.2 Identifying the Knowledge Referencing Behavior of the Target Worker

This step identifies the target worker's knowledge referencing behavior by matching his/her KF with the sequential rules discovered in the previous step. Specifically, the rules are matched with the topic-level KF of the target worker to predict the topics required at time *T*. We set a knowledge window on the KF before time *T*. The size of the window is determined by the user. Let $KW_u =< TP_u^{T-s}, TP_u^{T-s+1} \cdots, TP_u^{T-1} >$ be the knowledge window for the topic-level KF of a target worker *u* before time *T*. Note that TP_u^{T-f} is the topic referenced by *u* at time *T-f, f=1…s*. The knowledge window KW_u covers several topics previously referenced by the target worker and arranged in time order. The steps of sequential rule matching are as follows.

Step 1. Set a knowledge window KW_u.

The reference time of topics in the window may range from *T-s* to *T-1*, where *s* is the window size determined by the worker. The referencing behavior within the knowledge window is then compared with the sequential rules extracted from the KFs of the target worker's neighbors (Step 3).

Step 2. Generate topic subsequences and compare them with the knowledge window

All generated rules are compared with the given knowledge window to obtain the matching scores of rules. A sequential rule may partially or fully match a knowledge window. To identify sequential rules that match the target worker's referencing behavior, we consider all partial matches of the rules. Therefore, all possible topic subsequences are generated from the conditional part of the rule first.

The topic subsequences are enumerated according to the topic order in the conditional part of a rule. Let $TS_y^k =< TP_y^{k_1}, ..., TP_y^{k_i}, ...TP_y^{k_m} >$ be a topic subsequence in the conditional part of a sequential rule *y,* and let $TP_y^{k_i}$ be a topic with the index position k_i in the sequence TS_y^k. In addition, let KW_u be a knowledge window in a worker's KF, and let TP_u^{hj} be a topic with the index position h_j in the sequence KW_u. Then, each topic subsequence of a rule is examined by checking whether it exists in the knowledge window.

Instead of using identical matches, all the topics in a topic subsequence are compared with those in the knowledge window by using topic similarities to determine their matches. The characteristics of a KF are different from those of a general sequence, because a topic in a KF is composed of abstract knowledge concepts. Rather than using the identical match method, we use the topic similarity, i.e., *simcos*($TP_y^{k_i}, TP_u^{h_j}$), to determine if two topics match. That is, they match if their similarity is greater than the user-specified threshold θ.

We define a similarity matching score to compare a topic subsequence with a knowledge window. A topic subsequence TS_y^k matches the knowledge window KW_u, if their corresponding topic similarities are larger than the user defined threshold, i.e. $simcos(TP_y^{k_1}, TP_u^{h_1}) > \theta$, $simcos(TP_y^{k_2}, TP_u^{h_2}) > \theta$, ..., $simcos(TP_y^{k_m}, TP_u^{h_m}) > \theta$, where integers $k_1 < k_2 < \ldots < k_m$, $h_1 < h_2 < \ldots < h_m$, and θ is the user-defined threshold. The similarity matching score is the summation of the topic similarities, as defined in Eq. (13).

$$SM_{TS_y^k, KW_u} = \sum_{i=1}^{m} simcos(TP_y^{k_i}, TP_u^{h_i}), \qquad (13)$$

Step 3. Find the matching degree of a sequential rule.

Given the similarity matching scores of all topic subsequences extracted from a sequential rule, we choose the subsequence with the highest score to compute the matching degree of the rule. The matching degree is defined as follows:

$$RMD_{R_y, KW_u} = \max_{k=1..q}\left\{SM_{TS_y^k, KW_u}\right\} \times Support_y \times Confidence_y, \qquad (14)$$

where RMD_{R_y, KW_u} is the matching degree of rule R_y and KW_u of the target worker u; $\max_{k=1..q}\left\{SM_{TS_y^k, KW_u}\right\}$ is the highest similarity matching score of all topic subsequences of sequential rule y; and k from 1 to q is all topic subsequences of sequential rule y; The matching degree is used to identify the sequential rules qualified to recommend topics at time T.

Step 4. Choose sequential rules for recommendation

A sequential rule with a high matching degree means that the referencing behavior of the target worker matches the conditional part of the rule, so the consequent part of the rule can be selected as a predicted topic for the target worker at time T. Hence, the Top-N approach can be used to derive a set of predicted topics by selecting N rules with the highest matching degree scores.

4.3.3 Document Recommendation

The KSR method predicts a document rating based on sequential rules derived from the KFs of a target worker's neighbors. Let KNB_u^v be a set of neighbors of target worker u for a task *v*, selected according to the KF similarity (using Eq. (8)). The sequential rules derived from KNB^{v}_{u} with high degrees of rule matching are selected to recommend topics for the target worker at time *T*. However, the referencing behavior of some workers in KNB^{v}_{u} may not match the selected sequential rules. Therefore, we apply the sequential rule matching method discussed in Section 4.3.2 to compare the KFs of workers in KNB^{v}_{u} with the selected sequential rules. If a worker's KF matches a selected sequential rule, that worker's referencing behavior conforms to the sequential rule, and can therefore be used to make recommendations based on the selected sequential rules. The reason for checking the KFs of workers in KNB^{v}_{u} is to identify neighbors whose referencing behavior conforms to the selected sequential rule.

For a task *v*, let $KNBR^{v}_{u}$ denote the neighbors in KNB^{v}_{u} whose KFs are very similar to the target worker's KF and whose referencing behavior matches the selected sequential rules. In addition, let *RTS* be a set of recommended topics derived from the consequent parts of the recommended sequential rules; τ be a recommended topic, where $\tau \in RTS$; and the topic of a document *d* be τ. Based on the KFs of the neighbors in $KNBR^{v}_{u}$, the predicted rating of a document *d* belonging to the recommended topic τ for the target worker *u* is calculated by Eq. (15):

$$\hat{p}^{v}_{u,d,\tau} = \bar{r}^{v}_{u,\tau} + \frac{\sum_{x^{l} \in KNBR^{v}_{u}} sim(TKF^{v}_{u}, TKF^{l}_{x}) \times (r^{l}_{x,d,\tau} - \bar{r}^{l}_{x,\tau})}{\sum_{x^{l} \in KNBR^{v}_{u}} \left| sim(TKF^{v}_{u}, TKF^{l}_{x}) \right|}, \qquad (15)$$

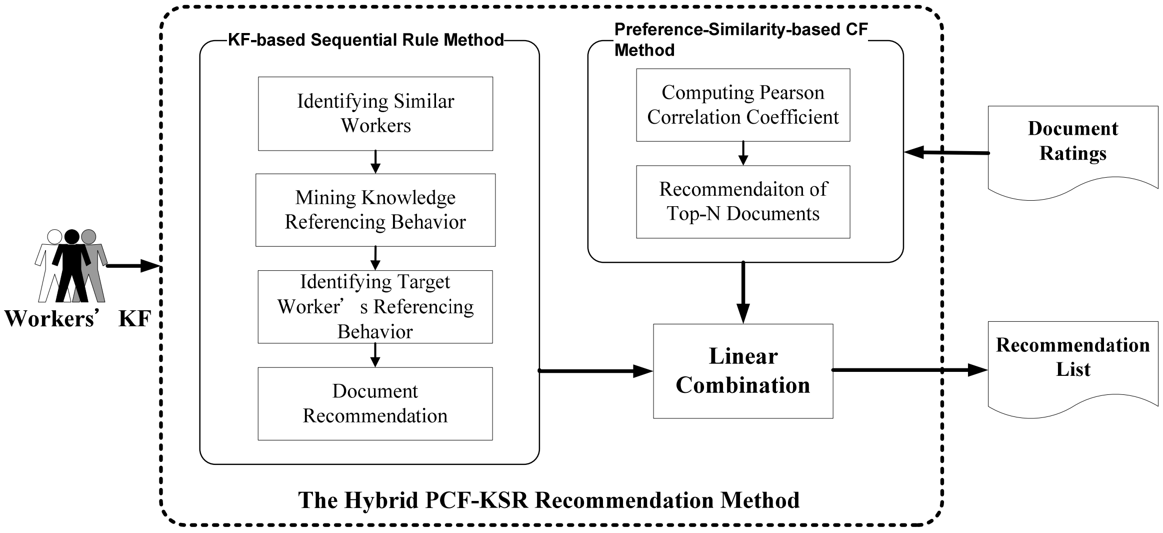
where $\bar{r}^{v}_{u,\tau} / \bar{r}^{l}_{x,\tau}$ is the topic rating of the target worker *u*/worker *x* for task *v*/ *l*, derived from the worker's average rating of documents in the recommended topic τ; $TKF^{v}_{u} / TKF^{l}_{x}$ is the topic-level KF of the target worker *u*/ worker *x* for task *k*/ task *l*; $r^{l}_{x,d,\tau}$ is the rating given by worker *x* for a document *d* belonging to the recommended topic τ in task τ; and $sim(TKF^{v}_{u}, TKF^{l}_{x})$ is the KF similarity of worker *u* and worker *x*, derived by Eq. (8). If the target worker *u* does not rate any documents in τ, then $\bar{r}^{v}_{u,\tau}$ is replaced by the average rating of all his/her documents. Meanwhile, if the target worker's neighbors do not rate any documents in τ, the predicted rating of document *d* is derived by the average rating of the target worker's documents.

To recommend task-related documents to a target worker, it is necessary to collect data with explicit ratings. Many recommender systems and recommendation methods use such ratings to represent users' preferences. Similarly, our recommendation methods use knowledge workers' document ratings to predict other documents that may be useful to a target worker's task, as shown in Eq. (15). Each knowledge worker gives explicit ratings to the documents referenced during the task's execution, while documents related to different tasks are re-rated by different workers. The ratings are used to gauge a worker's perceptions about the usefulness and relevance of documents for a specific task. The stronger the worker's perceptions of the usefulness or relevance of a document for the task at hand, the higher the rating he/she will give the document. Such ratings are subjective because they are based on the worker's perspective. Moreover, since a document may be referenced by different workers as they execute their specific tasks, it will be given different ratings based on how the workers perceive its usefulness and relevance to their tasks.

The sequential rules with high matching scores are selected to recommend topics. In other words, topics with high scores in the consequent part of a rule are recommended to the target worker at time *T*. The KSR method predicts ratings for documents that belong to the recommended topics and gives them a high priority for recommendation. Unlike traditional methods, KSR recommends documents to the target worker based on the selected sequential rules and the document ratings. Note that the KSR method does not consider the similarity of workers' preferences when calculating the predicted rating of a document.

4.4 The Hybrid PCF-KSR Method

The hybrid PCF-KSR recommendation method linearly combines the preference-similarity-based CF method (PCF) with the KSR method to recommend documents to a target worker, as shown in Fig. 5. The PCF method is the traditional CF method that makes recommendations according to workers' preferences for codified knowledge. To recommend a document, the neighbors of a target worker are selected based on the similarities of the workers' preference ratings. Pearson's correlation coefficient is used to find similar workers based on the document rating vectors. Then, PCF-KSR predicts the rating of a document by linearly combining the predicted ratings calculated by the two methods. One part of the rating is derived by the PCF method based on the document ratings and the preferences of the target worker's neighbors. The other part is derived by the KSR method described in Section 4.3. Because a worker's knowledge flow may change over time, the hybrid method considers the worker's preference for documents as well as topic changes in his/her KF to make recommendations adaptively.



KF-based Sequential Rule Method
Identifying Similar Workers
Mining Knowledge Referencing Behavior
Identifying Target Worker's Referencing Behavior
Document Recommendation
Preference-Similarity-based CF Method
Computing Pearson Correlation Coefficient
Recommendaiton of Top-N Documents
Document Ratings
Workers' KF
Linear Combination
Recommendation List
The Hybrid PCF-KSR Recommendation Method

Fig. 5: The framework of the hybrid PCF-KSR method

The predicted rating of a document d for a worker u executing a task v is derived by combining the PCF and KSR methods, as defined in Eq. (16):

$$\hat{p}_{u,d}^{v} = \beta_{PCF-KSR} \times \left[\bar{r}_u^v + \frac{\sum_{x^l \in PNB_u^v} PSim(u^v, x^l) \times (r_{x,d}^l - \bar{r}_x^l)}{\sum_{x^l \in PNB_u^v} \left| PSim(u^v, x^l) \right|} \right] + (1 - \beta_{PCF-KSR}) \times \hat{p}_{u,v,d}^{KSR}, \qquad (16)$$

where $\bar{r}_u^v$ / $\bar{r}_x^l$ is the average rating of documents for task v / task l given by the target worker u / worker x; $PSim(u^v, x^l)$ is the similarity between the target worker u for task v and the neighbor worker x for task l, derived by Pearson's correlation coefficient; PNB_u^v is the set of neighbors of the target worker u for task v, selected by $PSim(u^v, x^l)$; $r_{x,d}^l$ is the rating of a document d for task l given by worker x; $\hat{p}_{u,v,d}^{KSR}$ is the predicted rating of a document d for the target worker u engaged in task v based on the KSR method; and $\beta_{PCF\text{-}KSR}$ is the weighting used to adjust the relative importance of the PCF method and KSR method.

According to Eq. (16), a document in a recommended topic has a higher priority for recommendation than documents that are not in the recommended topics, based on their predicted ratings derived by the KSR method. Documents with high predicted ratings are used to compile a recommendation list, from which the top-N documents are chosen and recommended to the target worker.

4.5 The Hybrid KCF-KSR Method

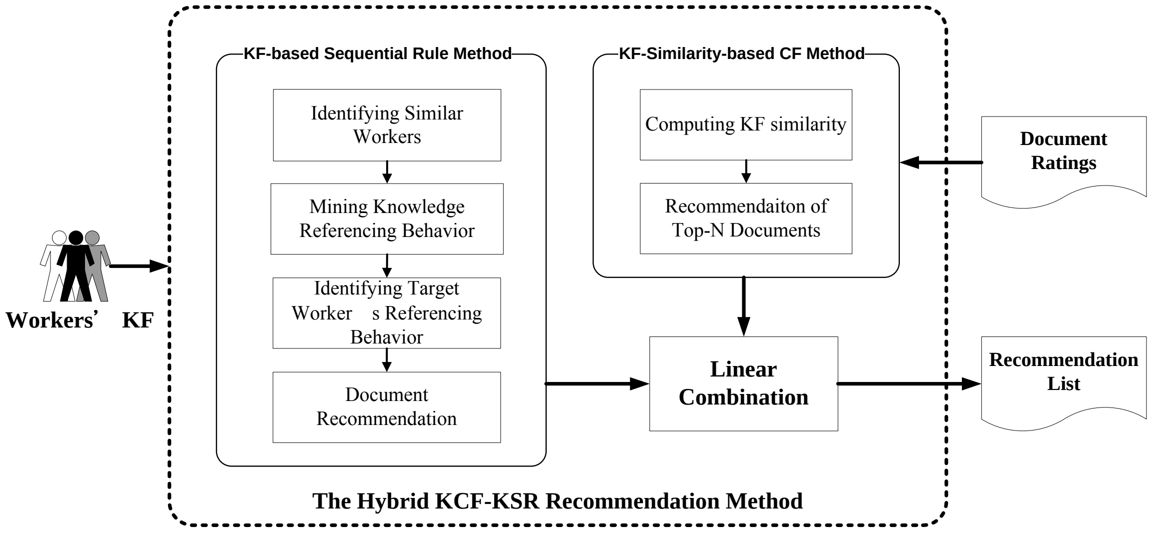

KF-based Sequential Rule Method
Identifying Similar Workers
Mining Knowledge Referencing Behavior
Identifying Target Worker s Referencing Behavior
Document Recommendation
KF-Similarity-based CF Method
Computing KF similarity
Recommendaiton of Top-N Documents
Document Ratings
Workers' KF
Linear Combination
Recommendation List
The Hybrid KCF-KSR Recommendation Method

Fig. 6: The framework of the hybrid KCF-KSR method

The hybrid KCF-KSR method linearly combines the KF-similarity-based CF method (KCF) with the KSR method to recommend documents to a target worker, as shown in Fig. 5. The KCF method is based on the referencing behavior of neighbors with similar KFs, while the PCF method is based on the similarity of preference ratings derived by Pearson correlation coefficient. Like the PCF-KSR method, the predicted rating of a document is also derived by integrating two parts of the ratings. One part is obtained by the KCF method, while the other is obtained by the KSR method described in Section 4.3.

The hybrid KCF-KSR method predicts the rating of a document d for worker u engaged in task v by Eq. (17), and then determines which documents should be recommended.

$$\hat{p}^{v}_{u,d} = \beta_{KCF-KSR} \times \left[\bar{r}^{v}_{u} + \frac{\sum_{x^{l} \in KNB^{v}_{u}} sim(TKF^{v}_{u}, TKF^{l}_{x}) \times (r^{l}_{x,d} - \bar{r}^{l}_{x})}{\sum_{x^{l} \in KNB^{v}_{u}} \left| sim(TKF^{v}_{u}, TKF^{l}_{x}) \right|} \right] + (1 - \beta_{KCF-KSR}) \times \hat{p}^{KSR}_{u,v,d} \quad , \qquad (17)$$

where $\bar{r}^{v}_{u} / \bar{r}^{l}_{x}$ is the average rating of documents given by the target worker u / worker x engaged in task v/ l; $r^{l}_{x,d}$ is the rating of a document d for task l given by worker x; TKF^{v}_{u}/ TKF^{l}_{x} denotes the topic-level KF of the target worker u/ worker x for task k/ task l; $sim(TKF^{v}_{u}, TKF^{l}_{x})$ is the KF similarity of worker u and worker x, derived by Eq. (8); KNB^{v}_{u} is the set of neighbors of the target worker u for task v, selected according to their KF similarity scores; $\hat{p}^{KSR}_{u,v,d}$ is the predicted rating of a document *d* based on the KSR method; and $\beta_{KCF\text{-}KSR}$ is the weighting used to adjust the relative importance of the KCF method and the KSR method.

According to Eq. (17), a document in a recommended topic has a higher priority for recommendation than those documents that are not in the recommended topic. The KCF-KSR method considers the KF similarity of two workers, their preferences for documents, and topic sequences in the KF when making recommendations.

4.6 The Hybrid ICF-KSR Method

The hybrid ICF-KSR recommendation method linearly combines the item-based CF method (ICF) with the KSR method to recommend documents to a target worker, as shown in Fig. 7. The ICF method is the traditional item-based CF method [50] described in Section 2.6. The similar documents (neighbors) of a target document are selected based on the adjusted cosine similarities of the documents (Eq. (6)). Then, the predicted rating of the target document is computed by taking the weighted average of the target worker's ratings for similar documents (Eq. (5)).

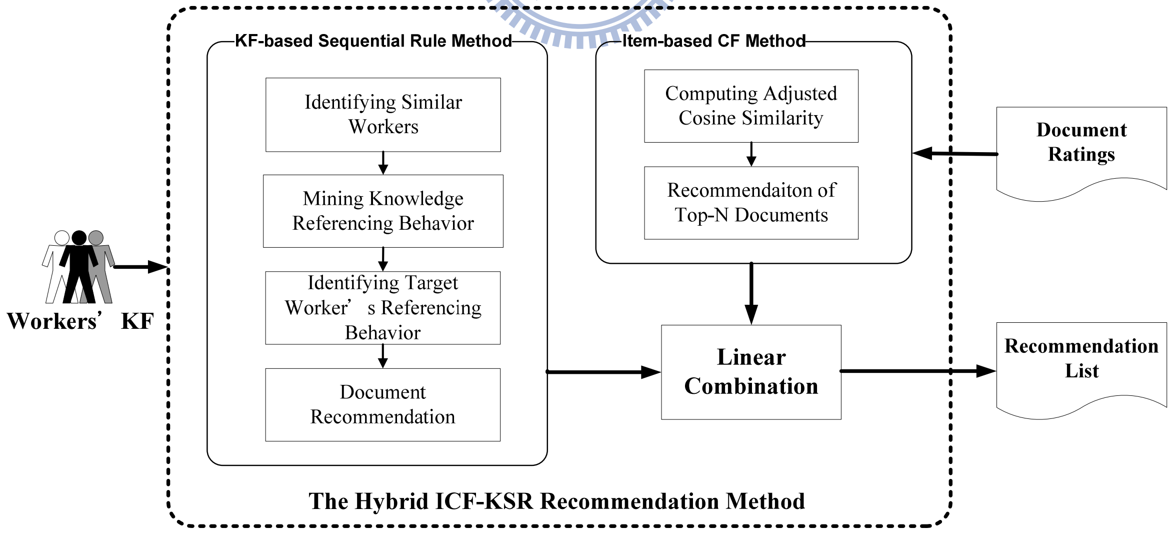

KF-based Sequential Rule Method
Identifying Similar Workers
Mining Knowledge Referencing Behavior
Identifying Target Worker' s Referencing Behavior
Document Recommendation
Item-based CF Method
Computing Adjusted Cosine Similarity
Recommendaiton of Top-N Documents
Document Ratings
Workers' KF
Linear Combination
Recommendation List
The Hybrid ICF-KSR Recommendation Method

Fig. 7: The framework of the hybrid ICF-KSR method

The ICF method does not consider workers' referencing behavior when they perform tasks. To address this issue, we propose the hybrid ICF-KSR method, which integrates traditional item-based collaborative filtering and the KSR method to recommend documents that may meet workers' information needs. The ICF-KSR approach predicts the rating of a document by linearly combining the predicted ratings calculated by the two methods. One part of the rating is derived by the ICF method based on the target worker's ratings for documents similar to the target document. The other part is derived by the KSR method described in Section 4.3. A worker's knowledge flow may change over time. Thus, to make recommendations adaptively, the hybrid method considers documents similar to the target document, the worker's perceptions about the usefulness of the documents, and the topic sequences in his/her KF.

The hybrid ICF-KSR method predicts a rating for a document *d* for worker *u* performing a task *v* by using Eq. (18), and then determines the documents that should be recommended.

$$\hat{p}^{v}_{u,d} = \beta_{ICF-KSR} \times \left[\frac{\sum_{i \in I_d} ACSim(d,i) \times r^{v}_{u,i}}{\sum_{i \in I_d} \left| ACSim(d,i) \right|} \right] + (1 - \beta_{ICF-KSR}) \times \hat{p}^{KSR}_{u,v,d} \quad , \qquad (18)$$

where $r^{v}_{u,i}$ is the rating of the usefulness of a document *i* given by worker *u* for task *v*; $ACSim(d,i)$ is the adjusted cosine similarity between document *d* and document *i*; I_d is the set of documents similar to document *d*, selected according to their adjusted cosine similarities; $\hat{p}^{KSR}_{u,v,d}$ is the predicted rating of document *d* for the target worker *u* engaged in task *v* based on the KSR method; and $\beta_{ICF-KSR}$ is the weighting used to adjust the relative importance of the ICF method and the KSR method. According to Eq. (18), a document in a recommended topic has a higher priority for recommendation than documents that are not in the recommended topic.

In Section 4.7 and 4.8, we conduct experiments to compare and evaluate the recommendation quality for the hybrid PCF-KSR, KCF-KSR and ICF-KSR methods, and then have some discussions about these experimental results. Next, we will describe the experiment setup in Section 4.7, discuss the experiment results and evaluations in Section 4.8, and have some discussions in Section 4.9.

4.7 Experiment Setup

To demonstrate that knowledge flows can support the recommendation of task-relevant knowledge (documents) to knowledge workers, experiments were conducted on a dataset from a real application domain, namely, research tasks in the laboratory of a research institute.

The dataset contained information about the access behavior of each knowledge worker engaged in performing a specific task, e.g., writing a research paper or conducting a research project. To accomplish their tasks, the workers needed various documents (research papers). Besides the documents, other information, such as when the documents were referenced and the document ratings, is necessary for implementing our methods. Since it is difficult to obtain such a dataset, using the real application domain restricts the sample size of the data in our experiments.

The dataset is based on the referencing behavior of 14 knowledge workers in a research laboratory and 424 research papers used to evaluate the proposed methods. Specifically, it contains information about the content of the documents, the times they were referenced, and the document ratings given by workers. For each worker, the documents and the times at which they were referenced are used to identify the worker's referencing behavior when performing a task.

The document rating, which is given by a worker and on a scale of 1 to 5, indicates whether a document is perceived as useful and relevant to a task. A high rating, i.e., 4 or 5, indicates that the document is perceived as useful and relevant to the task at hand; while a low rating, i.e., 1 or 2, suggests that the document is deemed not useful. If a document has been referenced by a worker without being assigned a rating value, it is given a default rating of 3.

In our experiment, the dataset is divided according to the time order of the documents accessed by knowledge workers as follows: 70% for training and 30% for testing. The testing set contains documents with access time more close to the current time period. The training set is used to generate recommendation lists, while the test set is used to verify the quality of the recommendations. In the experiments, we evaluate and compare the performance of traditional CF methods and our KF-based recommendation methods, namely the hybrid PCF-KSR method, the hybrid KCF-KSR method, and the hybrid ICF-KSR method.

We use the Mean Absolute Error (MAE), which is widely used in recommender systems [11, 25-26, 51], to evaluate the quality of recommendations derived by our methods. MAE measures the average absolute deviation between a predicted rating and the user's true rating [50], as shown in Eq. (19).

$$MAE = \frac{\sum_{i \in Z, i=1}^{n} |p_i - q_i|}{n}, \qquad (19)$$

where MAE is the mean absolute error; Z is the test set of a target worker, which consists of n predicted documents; p_i is the predicted rating of document i; and q_i is the real rating of document i. The lower the MAE, the more accurate the method will be. The advantages of this measurement are that its computation is simple and easy to understand and it has well studied statistical properties for testing the significance of a difference.

4.8 Experiment Results

We conduct several experiments to measure the quality of recommendations derived by our methods. To generate topic-level KFs, the documents in the data set are grouped into clusters by the single-link hierarchical clustering method described in Section 3.3.1. To determine the threshold value that yields the best clustering result, we adjust the threshold value systematically in decrements of 0.05 ranging from 0.5 to 0.2 to generate different clustering results, each of which is evaluated by using the quality measure defined in Section 2.3.2. The cluster with the best quality measure generated by setting the threshold value at 0.3 is selected as our clustering result; it contains 8 clusters. Based on the clustering results, topic-level KFs are generated by mapping documents from the codified-level KFs into their corresponding clusters for each knowledge worker. Finally, by considering the topic-level and codified-level KFs, the hybrid PCF-KSR and KCF-KSR methods recommend task-related documents to users. In the following sub-sections, we discuss the experiment results.

4.8.1 Evaluation of the hybrid PCF-KSR Method

In this experiment, we evaluate the performance of the hybrid PCF-KSR method. The parameters, α and $\beta_{PCF\text{-}KSR}$, may affect the quality of the recommendations; α is used to calculate the KF similarity (Eq. (8)), while $\beta_{PCF\text{-}KSR}$ is used to predict a document's rating. We set various values for these parameters and determine the settings that yield the best recommendation performance. The experiment was conducted by systematically adjusting the values of α in increments of 0.1, and the optimal value (i.e., the lowest MAE value) was chosen as the best setting. Based on the experiment results, we set $\alpha = 0.3$ in all the following experiments.

We evaluate how the $\beta_{PCF\text{-}KSR}$ values and the number of neighbors, k, affect the recommendation quality, as shown in Fig. 8. The parameter $\beta_{PCF\text{-}KSR}$, whose value ranges from 0.1 to 1, represents the relative importance of the PCF method and KSR method in Eq. (16). The experiment was conducted using various numbers of neighbors (parameter k) to derive the predicted ratings. Fig. 8 shows that the lowest MAE value generally occurs when $\beta_{PCF\text{-}KSR}$ is 0.5.

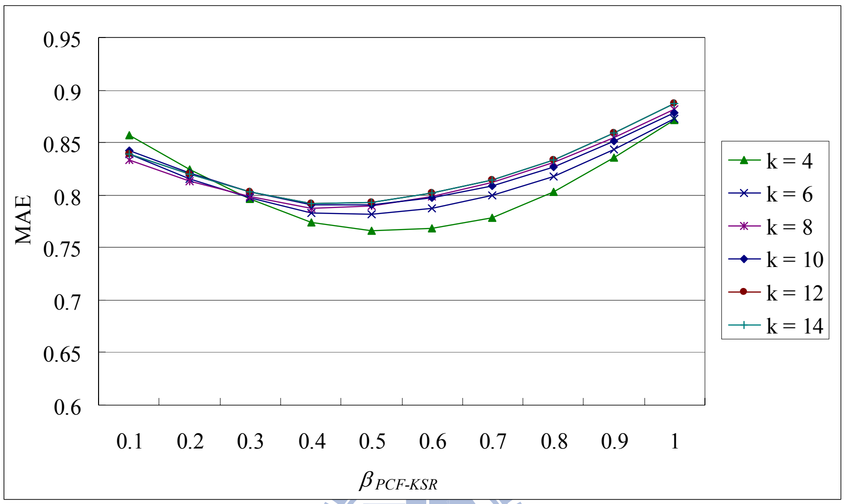


Fig. 8: The performance of the hybrid PCF-KSR method with various k and $\beta_{PCF\text{-}KSR}$ values

Fig. 9 compares the hybrid PCF-KSR method with the traditional CF method (PCF method). The predicted rating of a document is derived in two parts by the PCF method and the KSR method respectively. The part derived by the PCF method is based on the document ratings of the target worker's neighbors, while the other part is derived by the KSR method based on documents in the recommended topics and sequential rules generated from the KFs of the target worker's neighbors. If a document is in the recommended topic, the KSR part of PCF-KSR can be used to adjust the predicted rating of the document. Therefore, the PCF-KSR method ensures that documents in the recommended topics have a high priority for recommendation to the target worker. In the experiment, we set $\alpha = 0.3$ and $\beta_{PCF\text{-}KSR} = 0.5$, and select the top-5 sequential rules with high rule matching scores. The experiment results show that the PCF-KSR method outperforms the traditional CF method (PCF method) under various numbers of neighbors (parameter k). That is, the KSR method improves the recommendation quality of the PCF method. In other words, the PCF-KSR method is effective in recommending documents to the target worker, and it improves on the quality of the recommendations derived by the PCF method alone.

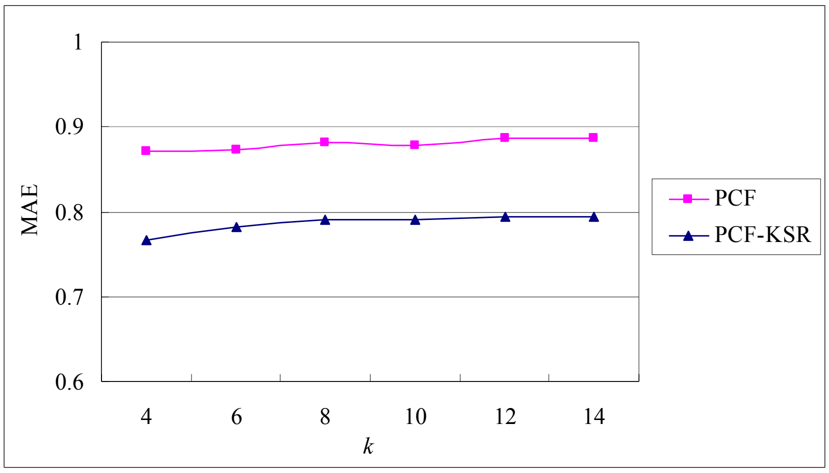


Fig. 9: Comparison of the hybrid PCF-KSR and PCF methods under different *k*

4.8.2 Evaluation of the hybrid KCF-KSR Method

Similar to the evaluation of the hybrid PCF-KSR method, we first determine the value of $\beta_{KCF\text{-}KSR}$ for the KCF-KSR method. The $\beta_{KCF\text{-}KSR}$ parameter, whose value ranges from 0.1 to 1, represents the relative importance of the KCF method and the KSR method. We set α=0.3 when calculating the KF similarity. The results show that the smallest value of MAE usually occurs when $\beta_{KCF\text{-}KSR}$ = 0.5 for different the numbers of neighbors (*k*). Thus, in this experiment, $\beta_{KCF\text{-}KSR}$ is set at 0.5 for the KCF-KSR method.

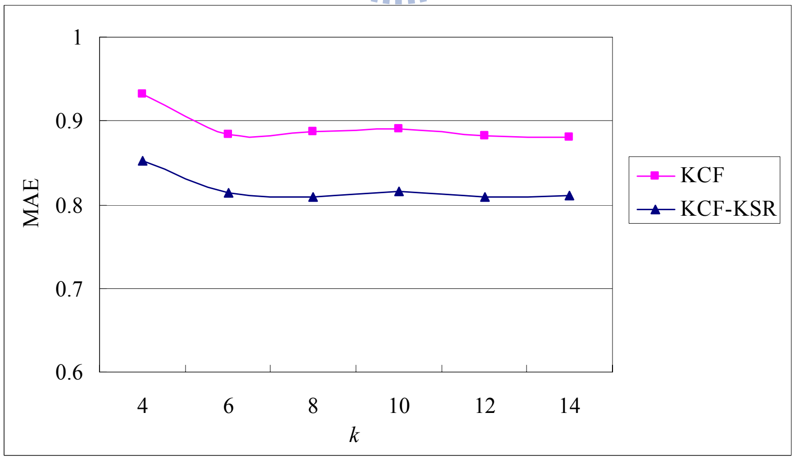


Fig. 10: Comparison of the hybrid KCF-KSR and KF methods under different *k*

To evaluate the performance of the KCF-KSR method, we compare it with the KF-similarity-based CF method (KCF) by setting $\beta_{KCF\text{-}KSR}$ at 1, as shown in Fig. 10. Note that when $\beta_{KCF\text{-}KSR}$ = 1, the predicted rating of a document is derived totally by the KCF method, which only uses the document ratings of the target worker's neighbors with similar KFs to make recommendations. The experiment results demonstrate that the hybrid KCF-KSR outperforms the KCF method. In other words, considering workers' knowledge referencing behavior can enhance the quality of recommendations.

4.8.3 Evaluation of the hybrid ICF-KSR Method

This experiment evaluates the performances of ICF and ICF-KSR methods. Once again we have to determine the value of the $\beta_{ICF\text{-}KSR}$ parameter in the range 0.1 to 1 to represent the relative weights of the ICF method and the KSR method. The results show that the smallest value of MAE usually occurs when $\beta_{ICF\text{-}KSR}$ = 0.4 under various number of neighbors (*k*). Relatively, KSR is more important than ICF in the hybrid ICF-KSR method because the weight of KSR is higher than that of ICF. Thus, $\beta_{ICF\text{-}KSR}$ is set at 0.4 for the ICF-KSR method in this experiment.

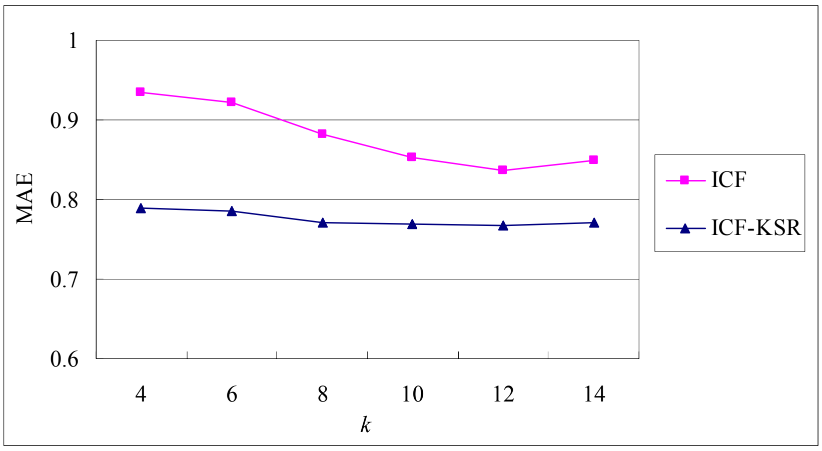


Fig. 11: Comparison of the hybrid ICF-KSR and KF methods under different *k*

To assess the impact of considering workers' referencing behavior on the ICF-KSR method, we compare it with the ICF method by setting $\beta_{ICF\text{-}KSR}$ at 1, as shown in Fig. 11. Setting $\beta_{KCF\text{-}KSR}$ = 1 means that the predicted rating of a document is derived totally by the ICF method, which only utilizes the adjusted cosine similarity measures between documents to make recommendations. The hybrid ICF-KSR method takes this issue into account. Fig. 11 demonstrates that the hybrid ICF-KSR method performs better than the ICF method under various numbers of neighbors (parameter *k*). The experiment results show that considering workers' knowledge referencing behavior under the KSR method improves the recommendation quality of the ICF method.

4.8.4 Comparison of All Methods

To evaluate the recommendation performances of the different methods, we compare the three individual methods (the PCF, KCF and ICF methods) and the three hybrid methods (the PCF-KSR, KCF-KSR and ICF-KSR methods), as shown in Fig. 12.

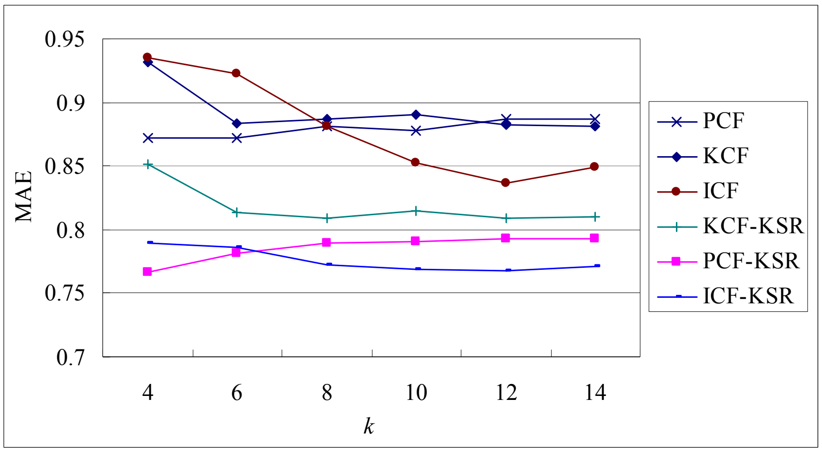


Fig. 12: The performances of the compared methods under different *k*

When the number of neighbors, *k*, is less than 8, the PCF method yields the lowest MAE values, while the ICF method yields the highest values. However, when the value of *k* is more than 8, the ICF method outperforms the KCF and PCF methods. The recommendation performances of the PCF method and the KCF methods are very close.

In this experiment, we also compare the hybrid PCF-KSR, the hybrid KCF-KSR and the hybrid ICF-KSR methods, under various *k* (the number of neighbors). To obtain the MAE values of these methods, we set α=0.3, $\beta_{PCF\text{-}KSR}$ =0.5, $\beta_{KCF\text{-}KSR}$ =0.5 and $\beta_{ICF\text{-}KSR}$ =0.4. The results show that the hybrid ICF-KSR method generally outperforms the PCF-KSR and KCF-KSR methods, while the PCF-KSR method performs better than the KCF-KSR method.

To examine the differences between the KF-based methods and the traditional CF method, we performed a statistical hypothesis test, the paired *t*-test, under various *k*. The results show that the differences are statistically significant at the 0.01 level. Here, we only report the results of the *t*-test under $k = 8$. The mean, standard deviation (SD), and *p*-value of MAE for each pair of recommendation methods are listed in Table 1. The proposed hybrid methods, i.e., PCF-KSR, KCF-KSR and ICF-KSR, have smaller mean and generally smaller standard deviation scores than their individual methods. In terms of the *p*-value, the differences between the proposed hybrid methods and the individual CF-based methods are statistically significant.

Table 1: The *t*-test results for various recommendation methods with $k = 8$

Recommendation Method	Mean	SD	*t*-test
PCF-KSR	0.7898	0.7189	$p = 0.0006$ (<0.01)
PCF	0.8814	0.7244	
KCF-KSR	0.8086	0.7581	$p = 0.0006$ (<0.01)
KCF	0.8865	0.7836	
ICF-KSR	0.7718	0.6880	$p = 0.0045$ (<0.01)
ICF	0.8814	0.6829	

From the above results, it is clear that the hybrid methods perform better than their individual methods. That is, the hybrid PCF-KSR, KCF-KSR and ICF-KSR methods perform better than PCF, KCF and ICF methods alone. The results show that the KF-based approaches can enhance the recommendation quality of traditional CF methods.

4.9 Discussion

The comparison of KSR, PCF, KCF and ICF methods are listed in Table 2. There are five major differences among these four methods, including tracking workers' referencing behavior, the effect of time factor, considering topic preferences, similarity computation methods and the document preferences of neighbors. Each method has its own advantages and limitations of making recommendations in different domains. To complement the merits of two methods, we propose three hybrid recommendation methods based on the KSR method.

The KF-based sequential rule (KSR) method improves the recommendation quality by considering the topic preferences and tracking workers' referencing behavior based on sequential rules, i.e., the information needs over time. It chooses neighbors whose KFs are very similar to the target worker's KF and whose referencing behavior matches the selected sequential rules. However, it does not consider the opinions of the target worker's neighbors who have similar preferences for documents, but PCF does. To solve this limitation, PCF method (traditional CF) and the KSR method are linearly combined as PCF-KSR method to improve the recommendation quality. Similar to the PCF method, the KCF method uses KF similarity to choose neighbors of the target worker, while the PCF uses Pearson's correlation coefficient to select neighbors with similar opinions. Thus, based on the KSR method, a hybrid of KCF and KSR as KCF-KSR method are proposed. In addition, both the PCF method and the KCF method select neighbors based on the similarity of preferences, while the ICF method chooses similar documents for a document based on their preferences given by a target user. Thus, the KSR method is combined with ICF method as ICF-KSR method which recommends documents from both user and item perspectives. Note that, each hybrid method linearly combines the recommendation lists from two individual methods. Because hybrid methods have complementary features derived from the merits of their combined methods, they generally outperform those individual methods in our experiments.

Table 2: The differences of all methods

Methods / **Influences**	**KSR**	**PCF**	**KCF**	**ICF**	**PCF-KSR**	**KCF -KSR**	**ICF -KSR**
Tracking workers' referencing behavior	Yes	No	No	No	Yes	Yes	Yes
Time factor	Yes	No	Yes	No	Yes	Yes	Yes
Considering topic preferences	Yes	No	No	No	Yes	Yes	Yes
The document preferences of neighbors	No	Yes	Yes	Yes	Yes	Yes	Yes
Similarity computation method	KF Similarity	Pearson's Correlation Coefficient	KF Similarity	Adjusted Cosine Similarity	Pearson's Correlation Coefficient / KF similarity	KF Similarity	Adjusted Cosine Similarity / KF similarity

Because each method has different features, it should be applied on an appropriate dataset or a suitable context to obtain the best performance. Our proposed methods are appropriate for a dataset where documents are clustered as various topic domains and the access behavior of workers over time are recorded. In addition, the CF methods have cold-start problem causing by new items and the sparsity problem. If there are new items that have fewer ratings given by users in a dataset, the CF methods cannot correctly make recommendations based on insufficient preference data, i.e., ratings on items. Similarly, a dataset with fewer preference ratings also causes the inaccurate recommendations. Moreover, the CF methods do not predict items based on their content similarity. To solve these problems and improve the recommendation quality, we will consider the content similarity of items in recommendation methods in our future work.

The contribution of this work is that our recommendation methods can proactively provide task-related knowledge based on knowledge flow. The experiment results demonstrate that the proposed KF-based hybrid methods, i.e., the PCF-KSR, KCF-KSR and ICF-KSR methods, improve the quality of document recommendation and outperform traditional CF methods. The three hybrid methods also perform better than the individual methods, i.e., the PCF, KCF, and ICF methods. Therefore, we discover that our proposed methods indeed improve the recommendation quality and obtain better performance than the traditional CF methods. In addition, providing topic knowledge to workers is helpful to support their tasks.

This study has some limitations. First, our experiments were conducted using a real application domain, i.e., research tasks in a research institute's laboratory. The domain restricted the sample size of the data and the number of participants in the experiments, since it is difficult to obtain a dataset that contains information that can be used for knowledge flow mining. Because of this limitation, in our future work, we will evaluate the proposed approach on other application domains involving larger numbers of workers, tasks and documents. Second, our evaluation focused on verifying the effectiveness of the proposed approach for recommending codified knowledge (documents) based on knowledge flows, rather than on user satisfaction or the system's usability. A study of user satisfaction or usability would add further insights into our system's ability to recommend task-relevant knowledge. In addition, the ratings given by people with different roles (e.g., professors and students) may have different influences on the recommendations. For example, it could be assumed that the rating given by a professor is more trustworthy than that given by a student. We will consider this issue in our future work.

Chapter 5. Group-based Knowledge Flow Mining Methods

A knowledge flow (KF) represents a knowledge worker's long-term information needs and accumulated task-related knowledge when he/she performs a task. In a previous work, we proposed a KF mining method to obtain each worker's KF from his/her work log [36]. We also presented document recommendation methods to support workers' in the execution of tasks and facilitate knowledge sharing in an organization. In the context of collaboration, workers usually have similar referencing behavior patterns, in which they share common topics or documents they find useful, or they reference task-related knowledge in a similar order. To model the common referencing behavior of a group, we propose a method for mining a group-based knowledge flow (GKF) from the KFs of a group of workers.

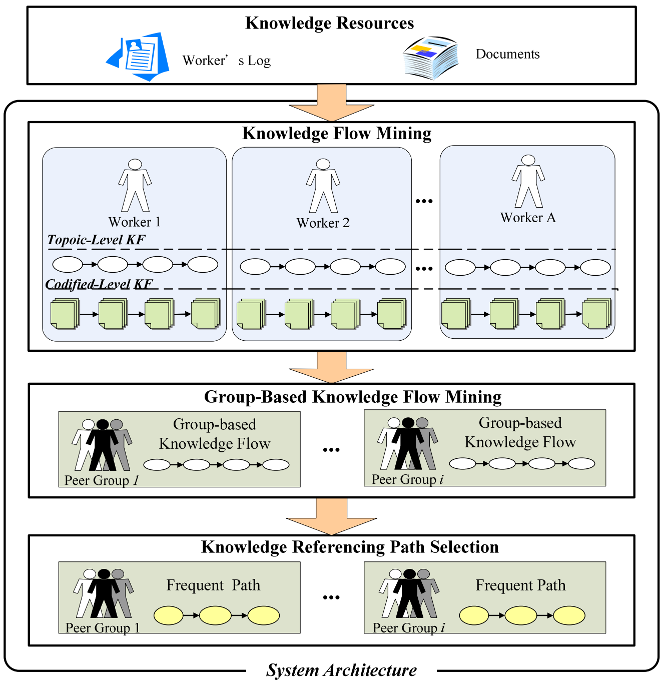


Fig. 13: An overview of mining group-based knowledge flows

Fig. 13 provides an overview of the proposed method for mining GKFs. Based on the workers' KFs, workers with similar topic-level KFs are clustered together to form a task-based group. Members of the group have task-related knowledge or similar referencing behavior in terms of the topics of interest and the order the topics were referenced in their KFs. To identify similar referencing behavior from the KFs, we propose KF mining algorithms based on process mining and graph theory to discover a group's knowledge flow. The algorithms identify common information needs and referencing patterns from the KFs of a group of workers, and then build a group-based knowledge flow (GKF) model. Then, a frequent knowledge path is identified from the model to represent the referencing (learning) patterns of the group and to support novices in learning a group's knowledge. In this work, we focus on two issues: 1) how to construct a group-based knowledge flow (GKF) model for a group of knowledge workers with similar KFs; and 2) how to identify frequent referencing patterns (paths) from the GKF model.

In the remainder of this Chapter, we detail the steps of the proposed group-based KF mining algorithm.

5.1 The group-based knowledge flow mining process

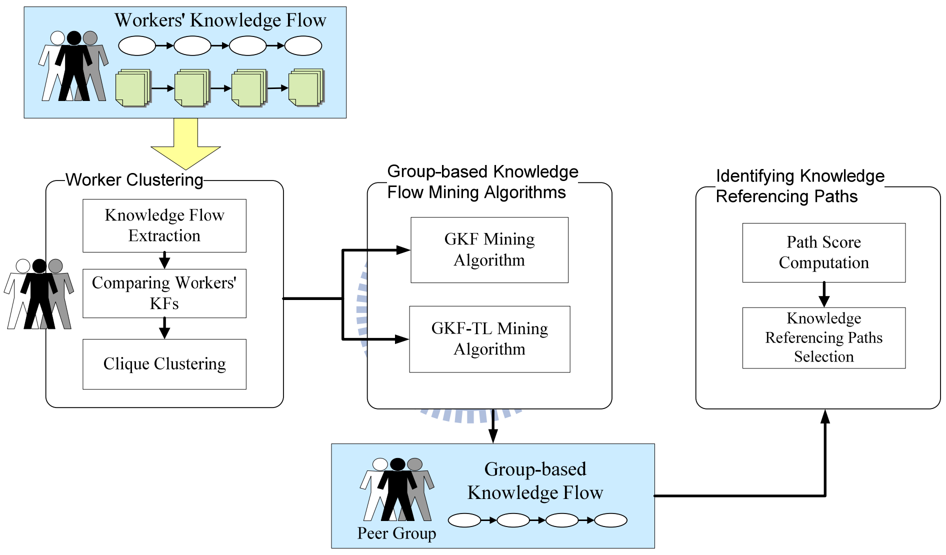

Workers' Knowledge Flow
Worker Clustering
Knowledge Flow Extraction
Comparing Workers' KFs
Clique Clustering
Group-based Knowledge Flow Mining Algorithms
GKF Mining Algorithm
GKF-TL Mining Algorithm
Identifying Knowledge Referencing Paths
Path Score Computation
Knowledge Referencing Paths Selection
Group-based Knowledge Flow
Peer Group

Fig. 14: The procedure of the proposed GKF mining method

The proposed method comprises three phases: worker clustering, group-based knowledge flow (GKF) mining, and identifying knowledge-referencing paths, as shown in Fig. 14. Based on the extracted KFs, the worker clustering step clusters workers with similar KFs as an interest group because they have similar information needs and task-related knowledge to fulfill a task. Given the KFs of the workers, we formalize the GKF model to represent the group's information needs by applying the proposed GKF mining algorithms. The GKF is represented by a directed acyclic graph comprised of vertices and edges. Each vertex denotes a topic in a KF, while each directed edge represents the referencing order of two topics. A GKF contains several knowledge referencing paths, which indicate the referencing behavior patterns of the group of workers. To identify frequent referencing behavior from the GKF model, we determine the frequency of each path. Then, we choose the paths with scores higher than a user-specified threshold as frequent knowledge referencing paths for the group.

5.2 Clustering Similar Workers Based on their Knowledge Flows

To find a target worker's neighbors, his/her topic-level KF is compared with those of other workers to compute the similarity of their KFs. The resulting similarity measure indicates whether the KF referencing behavior of two workers is similar. Since the KFs are sequences, the sequence alignment method [14, 43], which computes the cost of aligning two sequences, can be used to measure the similarity of two KF sequences. Based on this concept, we propose a hybrid similarity measure, comprised of the KF alignment similarity and the aggregated profile similarity, to evaluate the similarity of two workers' KFs, as shown in Eq. (8).

As mentioned earlier, workers with similar KFs are clustered together because they have similar task knowledge and referencing behavior. In this work, we use the CLIQUE clustering method [6, 29] to cluster knowledge workers based on a similarity matrix of their KFs. Each entry in a similarity matrix represents the degree of KF similarity between two workers, derived by Eq. (8). Based on the matrix, the CLIQUE clustering method is exploited to group workers with similar KFs. Workers in the same cluster are highly connected with each other because they have similar referencing behavior and information needs in topic domains. To identify each group's GKF, we apply our group-based knowledge mining method to process the clustering results.

5.3 Definition of Group-based Knowledge Flows

The group-based knowledge flow (GKF) represents the information needs and common referencing behavior of a group of workers. Based on GKF, workers can share their task knowledge to complete the target task. Moreover, managers can comprehend the information needs of workers and groups to provide knowledge support adaptively.

We use graph theory to model a GKF. A GKF graph models the relations between topics, the direction of the knowledge flow and the frequent knowledge paths to describe a group's information needs and referencing behavior. Next, we define the components of the GKF model and the features of the GKF graph, and then propose our GKF mining algorithms.

Definition 4: Knowledge Graph

A knowledge graph is defined as $G = (V, E)$, where V is a finite set of vertices, and E is a finite set of directed edges connecting two topics. Each vertex in V denotes a topic in the knowledge domain, and each edge in E denotes the knowledge flow from one topic to the other topic.

Example: Given a directed knowledge graph comprised of two vertices (topics) v_x and v_y and an edge $e_{x,y}$, the edge is used to connect vertices v_x to v_y directly, as shown in Fig. 15. In addition, v_x is said to be an adjacent predecessor of v_y, while v_y is said to be an adjacent successor of v_x.

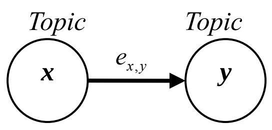


Fig. 15: An example of a directed graph

Definition 5: Knowledge Sub-graph

Given a knowledge graph $G = (V, E)$, a knowledge sub-graph of G is a graph $G' = (V', E')$, where V' and E' are subsets of V and E respectively, i.e., $V' \subset V$ and $E' \subset E$.

A GKF graph represents the referencing behavior of a group of workers as a directed knowledge graph, which consists of a finite set of vertices and edges, defined as follows.

Definition 6: Group-based Knowledge Flow (GKF)

As mentioned earlier, a GKF is derived from the KFs of workers who are in the same cluster and therefore have similar information needs. A GKF is defined as $GKF = \{G, W, TKF\}$, where G is a directed knowledge graph; $W = \{w_i \mid \forall i, i = 1 \cdots n\}$ is a set of n workers who have similar KFs; and $TKFS = \{TKF_j \mid \forall j, j = 1 \cdots n\}$ is a set of topic-level KFs of the workers in W.

The properties of TKF and the directed knowledge graph G are defined as follows.

Definition 7: Flow Relation and Direct Flow Relation

In a flow relation of a topic-level KF (TKF), topic x is followed by topic y, denoted by $x > y$, if topic x was accessed before topic y in the TKF. A topic x is followed directly by another topic y if there does not exist a distinct topic such that x is followed by z and z is followed by y. Thus, the relation between topics x and y is a direct flow relation, defined as $x \rightarrow y$.

Definition 8: Path

Given a directed graph G, if there is a path from a vertex v_x to another vertex v_y, the path is denoted as $v_x \sim> v_y$.

Definition 9: Topic Cycle

Let a flow relation $x > y$ appear in a TKF and a flow relation $y > x$ also appear in another TKF. The relations are represented by their corresponding paths, $v_x \sim> v_y$ and $v_y \sim> v_x$, on the graph of the GKF. Such relations form a topic cycle between the vertices of v_x (topic x) and v_y (topic y) in the GKF.

Definition 10: Topic Loop

Let x be a duplicate topic in a TKF and let two flow relations $x > y$ and $y > x$ appear in the TKF. These relations are represented by their corresponding paths, $v_x \sim> v_y$ and $v_y \sim> v_x$, on the graph of GKF. Such relations form a topic loop between the vertices of v_x (topic x) and v_y (topic y) in the GKF.

Definition 11: Strongly Connected Component (SCC)

A strongly connected component is a maximal strongly connected sub-graph in which every vertex is reachable from every other vertex in the sub-graph.

Definition 12: Knowledge Referencing Path

Given a directed graph $G = (V, E)$ of a GKF, if there is a path from a start vertex to an end vertex, it is a knowledge referencing path. Such a path is defined as $p = \{s, d, V_p, E_p\}$, where s is a start vertex, d is an end vertex, and V_p is a set of topics on the path p. E_p is a set of edges, where each edge is an ordered pair (v_i, v_j); v_i and $v_j \in V_p$, $v_i \neq v_j$ and v_i is an adjacent predecessor of v_j.

Definition 13: Frequent Referencing Path

Given a set of referencing paths derived from the graph of the GKF, a path *p* is said to be frequent if its path score, which is calculated based on the frequency count of edges on the path, is greater than a certain threshold. A frequent referencing path indicates that workers accessed task-related knowledge in a particular topic order frequently.

Problem Statement: Given the TKFs of a group of workers, the GKF mining algorithms finds the GKF from the KFs. The GKF is represented by a directed graph, which is used to model the referencing behavior of a group of workers.

5.4 GKF Mining Algorithm (without considering duplicate topics)

To derive a GKF model from a set of KFs, we propose two algorithms: one for cases where there are no duplicate topics in a KF; and the other for cases where there are duplicate topics. Both algorithms, which are based on graph theory, model a group's information needs as a group-based knowledge flow. The referencing path of a GKF details the order in which topics are accessed when workers search for task-related knowledge. In the following, we present a GKF mining algorithm for cases without duplicate topics.

We assume that a topic in a TKF appears just once in this algorithm. That is, there is no duplicate topic in each TKF; hence, there will not be a topic loop in the GKF. However, the order of topics in different TKFs may vary, so topic cycles, which form strongly connected components, may appear in the graph *G*.

In a strongly connected component (SCC), where each vertex is reachable from every other vertex, it is difficult to determine the ordering relation among the vertices. To resolve the problem, the algorithm applies the *Topic_Relation_Identification* procedure to identify the vertex relation in the SCC. The relation, which can be classified as either a parallel relation or a sequential relation to characterize the topic relations in the GKF, represents part of the topic ordering in workers' referencing behavior.

The GKF mining algorithm discovers frequent referencing of topics from the TKFs of a group of workers. To discover frequent referencing behavior patterns, which are modeled as frequent edges or frequent referencing paths on the GKF graph, the algorithm use the edge deletion procedure to remove infrequent edges whose weights are no greater than a user specified threshold. A start vertex and an end vertex are added to the discovered graph to indicate the start and end of the referencing behavior paths of the workers. Note that a topic is represented as a vertex on the graph. It would be odd to generate a GKF in which topic references were incomplete; that is, where a topic reference does not originate at the start vertex or reach the end vertex. The algorithm ensures that every topic can be referenced successfully from the start vertex to the end vertex. Thus, an infrequent edge can only be deleted if its removal does not make any vertex unreachable from the start vertex or to the end vertex.

```
GKF mining algorithm
Input: A set of $n$ workers in $W$ and their KFs, $TKFS = \{TKF_w | w=1…n\}$;
Output: $GKF = \{G, W, TKFS\}$;

A directed graph $G = \{V, E\}$, where $V=\phi$ and $E=\phi$ ;
Add a start vertex $s$ and an end vertex $d$ to $V$;
For each $TKF_w$ in $TKFS$ {
   Add each topic $v_y$ to $V$ according to the sequence order in $TKF_w$;
   Add an edge between the start vertex and the first topic in $TKF_w$ in $E$;
   Add an edge between the last topic in $TKF_w$ and the end vertex in $E$;
   For each vertex $v_x \in V$ and $v_x \rightarrow v_y$ in $TKF_w$ {
          Add an edge between vertex $v_x$ and $v_y$ in $E$; }
   Update the frequency of each edge in $E$;
}
Identify the strongly connected components (SCC) from $G$;
For each SCC $G_s$, where $G_s = (V_s, E_s)$, $V_s \in V$ and $E_s \in E$
    Topic_Relation_Identification($TKFS$, $G$, $G_s$);
Calculate the weights of all the edges in $E$;
Transform the graph $G$ into a new graph $G_N$ by mapping each SCC in $G$ as a vertex $v_{Gs}$ in
$G_N$ and mapping edges connected to $G_s$ in $G$ as edges connected to $v_{Gs}$ in $G_N$, where $G_N$
$= (V_N, E_N)$;
$L$ = Topological Sorting $(V_N, E_N)$;
$P$ = Edge Deletion $(L, G, G_N)$;
```

Fig. 16: The algorithm for mining a GKF when TKFs do not contain duplicate topics

Several knowledge paths may exist on a GKF graph. The paths represent the group's frequent referencing behavior when learning/referencing knowledge. Thus, the discovered graph can be used to inform a group of workers about topics of interest and the referencing behavior related to those topics.

The steps of the proposed algorithm are shown in Fig. 16. To generate a GKF model for a specific group (task), a set of TKFs is taken as the algorithm's input, and the graph of the *GKF* is the output result. In the GKF graph, a topic domain in a TKF is represented as a knowledge vertex, and each flow that directly orders the knowledge between two topics is represented as an edge. For example, given a TKF <A, B, E, C>, the four topics A, B, E and C are represented as four knowledge vertices, i.e., v_A, v_B, v_E and v_C, respectively; and the direct flows between two knowledge vertices are represented as three directed edges, i.e., $e_{A,B}$, $e_{B,E}$, and $e_{E,C}$, in the graph of G. Note that an edge is used to order the flow between two topics directly, e.g., the edge $e_{A,B}$ orders the flow from topic A to topic B. In contrast, if two topics have no direct flow relation, no edge exists between them. In the same example, there is no flow relation between topic A and topic E , so an edge $e_{A,E}$ does not exist.

The algorithm for building the GKF model involves several steps. First, a start vertex s and an end vertex d are added to the directed graph. Second, each topic in a TKF is regarded as a vertex and is added to a vertex set V if it does not exist in V already. Then, to connect the vertices in V, the edges related to the inserted vertex are added to the edge set E as follows. Let $x \rightarrow y$ be a direct flow relation from topic x to topic y, which denotes that topic x is followed immediately by topic y in a TKF_w. When adding the edge $e_{x,y}$ to E, the algorithm has to check two additional conditions for the edge to connect the starting/ending vertex with other vertexes. First, if the vertex y is the first vertex in a TKF, the edge $e_{s,y}$ from the starting vertex s to the vertex y is added to E; then, if the vertex y is the last topic in the TKF, the edge $e_{y,d}$ from the vertex y to the ending vertex d is added to E. When adding an edge to E, the algorithm counts the frequency of the edge. Adding all the vertices and their related edges to V and E respectively yields the initial graph of the GKF model.

Example of Creating the GKF Graph

This example illustrates how to build a GKF graph by using the GKF algorithm without considering duplicate topics in a TKF. Five workers who have similar TKFs form a group. Their topic-level KFs are listed in Table 3.

The topic domains in each topic-level KF (TKF) are arranged as a topic sequence according to the times they were referenced. Based on the TKF of each worker, the proposed algorithm derives the group's GKF, which is represented by a directed graph, as shown in Fig.

17. The topic domains, including the start and end vertices are represented by circles; an edge is represented by an arrow, which indicates the direction of knowledge flow from one knowledge vertex to another; and the number on each edge is the edge's frequency count.

Table 3: Five workers and their TKFs

Worker	Topic-level KF (TKF)
John	<A, B, C, D, E>
Mary	<A, C, G, F, D, E>
Lisa	<B, A, C, E>
Tom	<A, B, C, D>
Bob	<C, B, G, F, D>

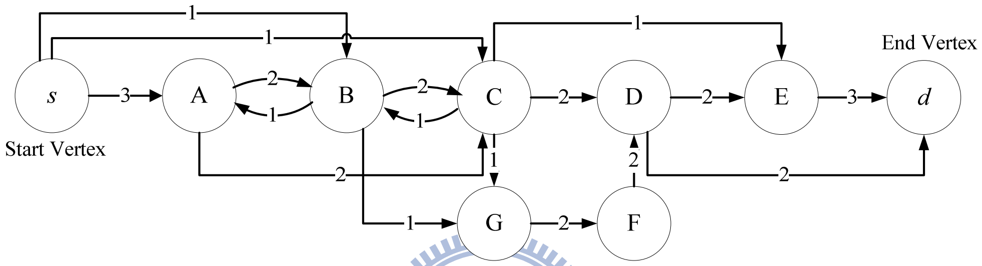


Fig. 17: The initial graph of the GKF model

In the initial graph, a strongly connected component (SCC) may be evident when some vertices appear in reverse order in any two TKFs. A strongly connected component G_s is a maximal strongly connected sub-graph that contains a path from each vertex to every other vertex in G_s. Because the vertices in a connected component are strongly connected, it is difficult to determine the ordering relationships between them. Even so, such relationships can be used to represent the characteristics of a TKF and they are important for modeling workers' referencing behavior. Thus, we use a procedure called *Topic_Relation_Identification* to determine the relationships among vertices in any strongly connected component.

In an SCC, two kinds of relations can be identified, namely, parallel and sequential relations. Any two vertices in an SCC indicate that two topics, x and y, may be referenced by different TKFs with the ordering $x > y$ and $y > x$. This ordering is an example of a parallel relation, where either v_x ~> v_y or v_y~>v_x would be appropriate; thus, there is no strict ordering between v_x and v_y. The referencing order of the vertices is not obvious, and the knowledge items represented by the vertices may be referenced simultaneously. As the vertices in an SCC are not in a specific order, conventional workflow mining methods consider the association between the vertices as a parallel relation. However, in contrast to such methods, a sequential relation pattern (SRP) rather than a parallel relation pattern (PRP) may be extracted if most of the referencing behavior in the SCC fits the SRP. That is, the SRP represents the most frequent knowledge referencing pattern in the SCC.

We explain how to recognize the above relations in Section 5.4.1, and how to evaluate, the weight of each edge when measuring the importance of a flow in the GKF in Section 5.4.2. Then, we transform the initial graph of the GKF into a new directed acyclic graph G_N in which a strongly connected component G_s is regarded as a vertex in Section 5.4.3.

After graph transformation, the topological sorting and edge deletion procedures are applied on G_N to remove any infrequent edges. An infrequent edge indicates that only a few workers in the group adopt a particular reference behavior pattern. Since such patterns are not representative of the group's general referencing behavior, they can be removed. The topological sorting procedure is used to sort all vertices in V_N in topological order, as discussed in Section 5.4.4. Based on the sorting result, the edge deletion procedure (described in Section 5.4.5) checks all the edges and removes infrequent and unqualified edges from E_N and E. After edge deletion, the graph G represents the group-based knowledge flow.

5.4.1 Topic Relation Identification

The topic relation identification procedure determines the relations between vertices in a strongly connected component, as shown in Fig. 18. Let the strongly connected component $G_s = (V_s, E_s)$, where V_s is a vertex set and E_s is an edge set. Parallel and sequential relations can be discovered from a strongly connected component $G_s = (V_s, E_s)$ based on the frequency count of knowledge flow sequences (*KFSs*). To determine and rebuild the relationships between vertices in V_s, all possible non-duplicate *KFSs* of length $|V_s|$, which contain all vertices in V_s, are identified from G_s. The derived *KFSs* are then compared with a non-duplicate sequence, i.e., SQ_w, in a TKF_w, which contains a set of vertices that are common to both V_s and the vertex set of $V(TKF_w)$, i.e., $V(SQ_w) = \{V_s \cap V(TKF_w)\}$. $V(SQ_w)$ / $V(TKF_w)$ denotes the set of vertices in the sequence SQ_w / TKF_w. When the sequence SQ_w is a subsequence of a *KFS*, the frequency count of the *KFS* is increased. Next, all the *KFSs* are sorted in descending order of their frequencies and the top-2 frequent *KFSs* are selected to elicit the relations of vertices in V_s. The preceding pseudo node v_γ and the succeeding pseudo node v_ρ of G_s are also added to V.

```
Topic_Relation_Identification ($TKF$, $G$, $G_s$) {
  Identify all possible non-duplicate flow sequences of length | $V_s$ | from $G_s$, where $KFS$ =
      {$KFS_x$ | $x$= 1..$n$};
  //Identify a sequence of vertices in $V_s$ from a TKF and compare it with sequences in $KFS$
  For each $TKF_w$ {
     Identify a non-duplicate sequence $SQ_w$ in $TKF_w$ that contains the common vertices in
         $V_s$ and $TKF_w$, i.e., $V(SQ_w) = \{V_s \cap V(TKF_w)\}$;
     Compare $SQ_w$ with each $KFS_x$ in $KFS$. If $SQ_w$ is a subsequence of $KFS_x$, increase the
         frequency count of $KFS_x$ , i.e., $f_{KFSx}$;
  }
  Sort all $KFS_x$ and select top-2 frequent flow sequences $KFS_a$ and $KFS_b$;
  Add a preceding pseudo node $v_\gamma$ and a succeeding pseudo node $v_\rho$ of $G_s$ to $V$;
  If ($|f_{KFSa} - f_{KFSb}| \le \varepsilon$) {    //parallel relation (and/or split)
      For each edge $e_{i,j}$ in $E_s$ {
        If ($v_i \rightarrow v_j$ exists in a $TKF_w$ and $v_j > v_i$ exists in another $TKF_y$)
            Remove the edge $e_{i,j}$ from $E$ and $E_s$;
      }
     For each vertex $v_i$ in $V_s$ {
        For each adjacent predecessor $v_k$ of $v_i$, where $v_k \in V$ and $v_k \notin V_s$ {
           Replace the edges $e_{k,i}$ with the edges $e_{k,\gamma}$ and $e_{\gamma,i}$, and update their frequency
              counts; }
        For each adjacent successor $v_l$ of $v_i$, where $v_l \in V$ and $\notin V_s$ {
             Replace the edges $e_{i,l}$ with the edges $e_{i,\rho}$ and $e_{\rho,l}$, and update their frequency
             counts; }
     }
  }
  else {    //sequential relation
     If ($f_{KFSa} > f_{KFSb}$) or ($f_{KFSb} > f_{KFSa}$)
         Let $KFS_y$ be the most frequent flow sequence;
     Let $v_i$/ $v_j$ be the first/ last vertex in $KFS_y$;
     Remove all edges in $E_s$ from $E_s$ and $E$;
     For each $v_g \rightarrow v_h$ in $KFS_y$ {add edges $e_{g,h}$ to $E_s$ and $E$};
     For each vertex $v_f$ in $V_s$ {
         For each adjacent predecessor $v_k$ of $v_f$, where $v_k \in V$ and $v_k \notin V_s$ {
              Replace edge $e_{k,f}$ with edges $e_{k,\gamma}$ and $e_{\gamma,i}$, and update their frequency counts; }
         For each adjacent successor $v_l$ of $v_f$, where $v_l \in V$ and $v_l \notin V_s$ {
              Replace edge $e_{f,l}$ with edges $e_{j,\rho}$ and $e_{\rho,l}$, and update their frequency counts; }
     }
  }
  Return $G$;
}
```

Fig. 18: The topic relation identification procedure

If the difference in the frequency counts of the selected *KFSs* is no greater than a user-specified threshold ε, the order of the vertices in V_s is not significant. In this case, the vertex relation is defined as parallel. For example, let us consider a strongly connected component where vertex v_x, vertex v_y and vertex v_z are in V_s; and let the user-specified threshold $\varepsilon = 2$. When the frequency counts of two *KFSs* $<v_x, v_y, v_z>$ and $<v_z, v_y, v_x>$ are 7 and 6 respectively, the relation between vertex v_x, vertex v_y and vertex v_z is parallel because the difference in their frequency counts is no greater than the threshold. However, if the difference is greater than a user-specified threshold, the *KFS* with the largest frequency count can be used to represent the relationship of vertices in V_s based on the majority principle. The ordering of these vertices is defined as a sequential relation. Next, we explain how to identify the order of vertices in a strongly connected component, i.e., parallel relations and sequential relations.

Identifying Parallel Relations in an SCC

For parallel relations, the order of the vertices in V_s is not important. The *Topic_Relation_Identification* procedure checks each edge in E_s for each TKF. Let $e_{i,j}$ be an edge in E_s that connects vertex v_i to vertex v_j directly. If this direct flow relation $v_i \rightarrow v_j$ appears in a TKF and a flow relation $v_j > v_i$ exists in another TKF, the edge $e_{i,j}$ is removed from E and E_s, and the relation between vertex v_i and vertex v_j is regarded as parallel. That is, there is no specific ordering between vertex v_i and vertex v_j, and their corresponding topics can be referenced in any order.

After adding a preceding pseudo node v_γ and a succeeding pseudo node v_ρ to G, the edges connected to the vertices in V_s are redirected through the pseudo nodes. To connect a vertex in V to the pseudo nodes, each adjacent predecessor v_k of v_i, where $v_k \notin V_s$ and $v_i \in V_s$, and each adjacent successor v_l of v_i, where $v_l \notin V_s$ and $v_i \in V_s$, are examined. For vertex v_k, if edge $e_{k,i}$, which connects vertex v_k to vertex v_i , exists in E, it is removed. Then, the edges $e_{k,\gamma}$ and $e_{\gamma,i}$ are added to E and their frequency counts are calculated. If the two edges already exist in E, their frequency counts are simply updated. Briefly, the edge $e_{k,i}$ is replaced by edges $e_{k,\gamma}$ and $e_{\gamma,i}$ to make a connection with vertex v_k and vertex v_i through the pseudo node v_γ. Similarly, for a vertex v_l, if edge $e_{i,l}$ exists in E, it is removed. Then, the edges $e_{i,\rho}$ and $e_{\rho,l}$, are added to E and their frequency counts are calculated. If the edges already exist in E, their frequency counts are simply updated.

Example of Identifying Parallel Relations

Fig. 17, there is a strongly connected component G_s comprised of V_s = {A, B, C} and E_s = {$e_{A,B}$, $e_{B,A}$, $e_{B,C}$, $e_{C,B}$, $e_{A,C}$}. Let the threshold ε be 1. The graph of the GKF after topic relation identification is shown in Fig. 19.

Based on the *Topic_Relation_Identification* procedure, two pseudo nodes, γ and ρ, are added to G. Then, the edges in E_s are examined to determine which ones should be removed. Three non-duplicate sequences are discovered in G_s, i.e., <A, B, C>, <A, C, B> and <B, A, C>; their frequency counts are 2, 1 and 1 respectively. Because the difference in the frequency counts of the top-2 sequences is equal to 1, the relation between vertex v_A, vertex v_C, and vertex v_B is regarded as parallel, and the edges $e_{A,B}$, $e_{B,A}$, $e_{B,C}$ and $e_{C,B}$ are removed from the graph.

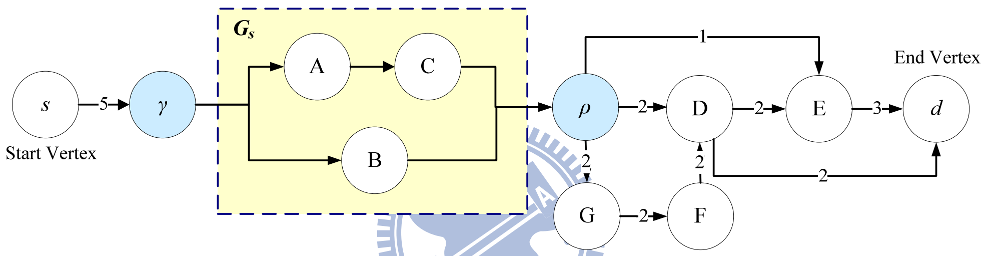


Fig. 19: A parallel relation in a GKF graph

Meanwhile, the relation between vertex v_A and v_C is regarded as sequence because A → C exists in one TKF, but there is no flow relation, i.e., C > A, in any other TKF. Thus, $e_{A,C}$ is not removed from the graph. The incoming edges of vertex v_A, vertex v_B and vertex v_C are changed to make connections through pseudo node v_γ. Similarly, the outgoing edges of vertex v_A, vertex v_B and vertex v_C are changed to make connections through pseudo node v_ρ. Then, the frequency counts of these edges are updated, as shown in Fig. 19.

Identifying Sequential Relations in an SCC

If the difference between the frequency-counts of the selected top-2 KFSs is greater than a user-specified threshold, the ordering of the vertices in the KFSs is regarded as a sequential relation. That is, based on the majority principle w.r.t. knowledge referencing behavior discussed earlier, the vertices in V_s follow the ordering of the *KFS* with the highest frequency. Let KFS_y be the knowledge flow sequence with the highest frequency count; and let v_i and v_j

be, respectively, the first and last vertices in the sequential order of KFS_y. All the edges in E_s are removed from E_s and E. Then, for each direct flow relation $v_g \rightarrow v_h$ in KFS_y, an edge $e_{g,h}$ is added to E_s and E. Similarly, the edges connected to the vertices in V_s are redirected through the pseudo nodes.

For each adjacent predecessor v_k of v_f, where $v_k \in V$, $v_k \notin V_s$, and $v_f \in V_s$, the edges $e_{k,\gamma}$ and $e_{\gamma,i}$ are added to E, and their frequency counts are calculated. If the edges already exist in E, their frequency counts are simply updated. The edge $e_{k,f}$, which connects vertex v_k to vertex v_f, is removed from E and replaced by the connections from v_k to v_γ and from v_γ to v_i, the first vertex of KFS_x. That is, the edge $e_{k,f}$ is replaced by edges $e_{k,\gamma}$ and $e_{\gamma,i}$, which connect with vertex v_k and vertex v_i respectively through the pseudo node v_γ. Similarly, for each adjacent successor v_l of v_f, where $v_l \in V$ and $v_l \notin V_s$, and $v_f \in V_s$, we use the same method to establish connections from the last vertex in KFS_x to the vertex v_l through the pseudo node v_ρ. The connection from v_f to v_l is replaced by the connections from the last vertex of KFS_x, i.e., v_j, to the pseudo node v_ρ and from v_ρ to v_l.

Example of Identifying Sequential Relations

Table 4: The TKFs of seven knowledge workers

Worker	Topic-level KF (TKF)
W1	<A, F, B, C, D, H>
W2	<A, G, B, C, D, I>
W3	<F, B, C, D, H>
W4	<A, F, C, D, B, K, H>
W5	<F, C, D, B, K, H>
W6	<A, G, B, C, K, H>
W7	< F, B, C, D>

Table 4 lists the knowledge flows of a group of seven workers. The GKF mining algorithm, described in Section 5.4, is used to generate the graph of the group-based KF and a strongly connected component with vertices v_B, v_C, and v_D is identified from the GKF graph. Then, the *Topic_Relation_Identification* procedure is applied to determine the relation between those vertices. As shown in Fig. 20, the relation is sequential with the ordering v_B, v_C, and v_D. In addition, the edges connected to any vertex in V_s are changed. For example, the edge $e_{B,K}$ is changed to edge $e_{D,\rho}$ and edge $e_{\rho,K}$ such that there is a path from vertex v_B to vertex v_K via the pseudo node v_ρ.

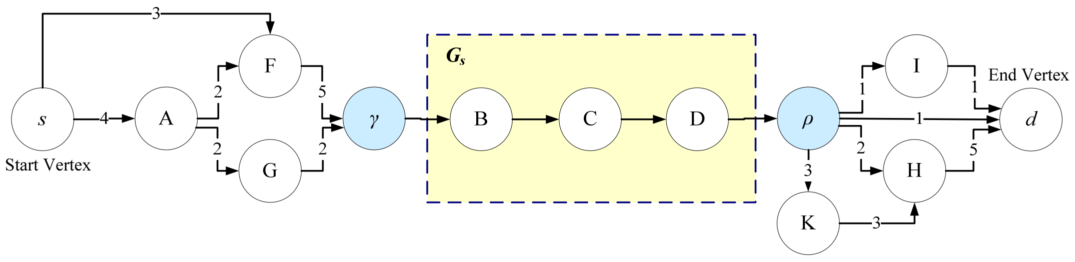


Fig. 20: A sequential relation in a GKF graph

5.4.2 Measuring the Importance of an Edge

Our objective is to derive the referencing behavior of a group of workers by constructing a frequent knowledge path in a GKF graph. However, some infrequent edges in the graph may not be suitable for building the path. To measure the importance of each edge in a graph, the frequency count of each edge is normalized by the maximum edge frequency in *E*. The weighting function measures the importance of an edge in a GKF model, as defined in Eq. (20).

$$we_{x,y} = \frac{f_{x,y}}{max\{f_{i,j} \mid \forall i,j,e_{i,j} \in E\}}, \qquad (20)$$

where $we_{x,y}$, which ranges from 0 to 1, is the weight of the edge $e_{x,y}$ that represents a direct flow from vertex v_x to vertex v_y; $f_{x,y}$ is the frequency of the edge $e_{x,y}$; *E* is the edge set of the graph; and the denominator is a maximum function that derives the frequency count of the most frequent edge in the graph. The more frequently an edge occurs, the more important it is deemed to be. The most frequent edge represents the frequent referencing behavior of most members of the group. Thus, it is suitable for describing the group's referencing behavior.

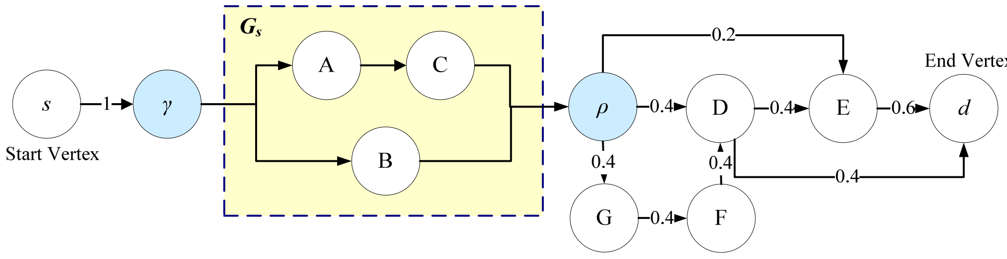


Fig. 21: The edge weights in a GKF graph

Example

The weight of each edge in Fig. 19 is calculated by using the edge weighting method. The edge is then labeled with the weight to indicate its importance in the graph, as shown in Fig. 21.

5.4.3 Graph Transformation

To simplify a strongly connected component in a graph, the proposed algorithm transforms the original GKF graph into a new graph G_N. After the transformation, the graph G_s is regarded as a vertex v_{Gs} in G_N. We create two pseudo nodes, v_γ and v_ρ , to represent, respectively, the split operator and the join operator of G_s. In addition, the incoming/ outgoing edges of $G_{s,}$ which connect to the pseudo nodes v_γ (the split operator) /v_ρ (the join operator), are merged to form a new edge whose weight is also updated. The weight of the incoming edge of $v_{Gs,}$ which combines the incoming edges of $G_{s,}$ is derived by combining the edge weights of the incoming edges of the node v_γ. Similarly, the weight of the outgoing edge of v_{Gs} is derived by combining the edge weights of the outgoing edges of the node v_ρ.

Example of Graph Transformation

We transform the graph G_s in Fig. 21 into a new graph for further analysis, as shown in Fig. 22. To simplify the strongly connected component, all the vertices in G_s are wrapped as a vertex v_{Gs} in the new graph. The incoming edges and outgoing edges of any vertex in G_s and the weights of those edges are adjusted. In Fig. 21, edge $e_{\gamma,A}$ and edge $e_{\gamma,B}$ are merged to form a new edge $e_{\gamma,Gs}$ in Fig. 22 and their edge frequencies are combined as 1. In the same way, edge $e_{C,\rho}$ and edge $e_{B,\rho}$ are combined to form an edge $e_{Gs,\rho}$.

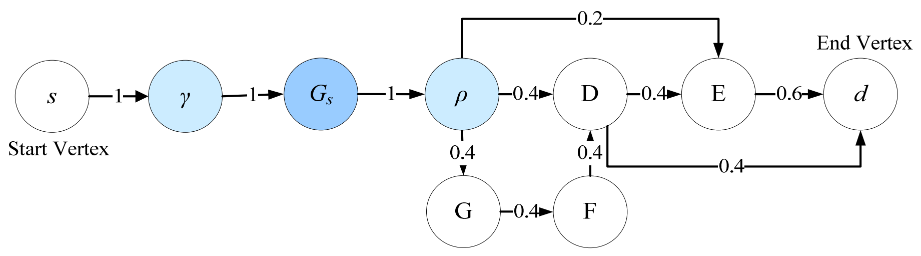


Fig. 22: The result of graph transformation

5.4.4 Topological Sorting

The frequent referencing behavior of a group of workers is derived by mining the group's knowledge flow from a GKF graph. The workers may reference topics in a different order when performing tasks, but some referencing behavior is more frequent because the majority of workers in the group reference topics in the same order. In the GKF graph, a frequent knowledge path from the start vertex to the end vertex represents the workers' frequent referencing behavior. For any vertex v_i on the path, vertex v_i is reachable from the start vertex and the end vertex is reachable from vertex v_i. Note that a path with infrequent edges denotes an infrequent referencing behavior pattern.

To derive a group's frequent referencing behavior, a topological sorting procedure is used to sort all vertices in the graph, after which infrequent edges whose weights are no greater than a specified threshold are deleted. In graph theory, topological sorting [17, 31] is a very efficient way to arrange the vertices of a directed acyclic graph in topological order in linear time. The key property of the topological order is that, for any two vertices x and y, if x is a predecessor of y in the graph then x precedes y in the topological order.

In this work, we use topological sorting to arrange all vertices in $G_{N,}$ which is a directed acyclic graph before the edge deletion procedure is applied. Then, the edge-deletion procedure examines the vertices in topological order to identify the infrequent incoming edges of each vertex that should be removed. However, before removing an infrequent edge, the procedure needs to ensure that each vertex in the GKF satisfies two criteria. First, any vertex v_i on a knowledge path must be reachable from the start vertex and the end vertex must be reachable from vertex v_i. Second, removing the edges of a vertex v_i does not affect the path from the start vertex to the preceding vertices of v_i in the topological order. In other words, topological ordering guarantees that 1) a predecessor will be processed before a successor; and 2) the predecessor's reachability (i.e., from the start vertex to v_i) will not be affected by its successors. Thus, when an infrequent edge of any vertex v_i in G is removed, there is no need to verify the reachability of the predecessors of vertex v_i from the start vertex. On the other hand, the path from the predecessors of vertex v_i to the end vertex will be affected by removing an infrequent edge of v_i; therefore, the predecessors should be examined again to ensure that they can still reach the end vertex.

Example

In Fig. 22, all the vertices are sorted in topological order, and the resulting list is <s, γ, G_s, ρ, G, F, D, E, d>. According to the list, v_s is the first vertex to be checked, v_{Gs} is the second vertex and so on. The algorithm examines all the vertices in topological order and removes infrequent edges from the graph G_N via the edge deletion procedure.

5.4.5 Using the Edge Deletion Procedure to Remove Infrequent Edges

Based on the results of topological sorting of V_N, the edge deletion procedure examines the vertices and determines which incoming edges should be removed from them. It then removes infrequent edges whose weight is no greater than a user-specified threshold, as shown in Fig. 23. The inputs of this procedure are the sorted list L derived by topological sorting and the edge set E_N of the GKF graph. The algorithm checks the incoming edges of each vertex in ascending order of their weights, and those whose weights are no greater than a user-specified threshold η are candidates for removal. If an edge is removed, it means that the knowledge referencing behavior between two vertices (topics) is infrequent among the group of workers.

```
Edge Deletion ($L$, $G$, $G_N$) {
  $Q = \phi$ ; // the checked set of vertices
  For each vertex $v_y$ in $G_N$ , according to the vertex's order in the sorted list $L$ {
        For each incoming edge $e_{x,y}$ of $v_y$ , according to its weight in ascending order {
             If (the weight of edge $e_{x,y}$ < threshold $\theta$) {
                Remove the edge $e_{x,y}$ from $E$ and $E_N$;
               If (no path $p_{s,y}$ exists from the start vertex $s$ to vertex $v_y$ in $G_N$) or (there
                 exists a vertex $v_j$, $v_j \in Q$ and no path $p_{j,d}$ exists from vertex $v_j$ to the end
                 vertex $d$)
                   Add the edge $e_{x,y}$ to $E$ and $E_N$;
             }
        }
        Add vertex $v_y$ to $Q$;
  }
}
```

Fig. 23: The edge deletion procedure

However, an infrequent edge should only be deleted from the graph if removing it would not make any vertex unreachable. Let Q be the set of vertices that have been checked in topological order to remove their infrequent incoming edges. For a vertex v_y, if one of its incoming edges is removed and there is no other path from the start vertex to v_y, the removed edge should be returned to the edge sets E and E_N. In addition, the vertices checked before v_y should be reexamined to ensure that there is a path from a checked vertex v_i in Q to the end vertex. If removing an edge violates the above condition, the edge should be returned to the edge sets E and E_N.

Because of the characteristics of topological sorting, the edge deletion procedure ensures that 1) any vertex in the graph G_N can be reached from the start vertex; and 2) removing an edge of a vertex does not affect any path from the start vertex to the predecessors of the vertex. In other words, there exists at least one path from each vertex to the end vertex. Moreover, we can obtain several frequent knowledge paths from the GKF graph to help workers learn the group's knowledge. The following example explains how to remove an edge from the GKF graph.

Example of Removing Infrequent Edges

In Fig. 22, let vertex v_E be the examined vertex and let the user-specified threshold be 0.3. The vertex v_E has two incoming edges: $e_{\rho,E}$ with weight 0.2 and $e_{D,E}$ with weight 0.4. The edge $e_{\rho,E}$ qualifies for removal, because its weight is no greater than 0.3 and removing it would not make any vertex unreachable. Fig. 24 shows the resulting graph, which represents the GKF of the group. The graph is used to visualize the knowledge flows among the frequent topics and model the referencing behavior of the group.

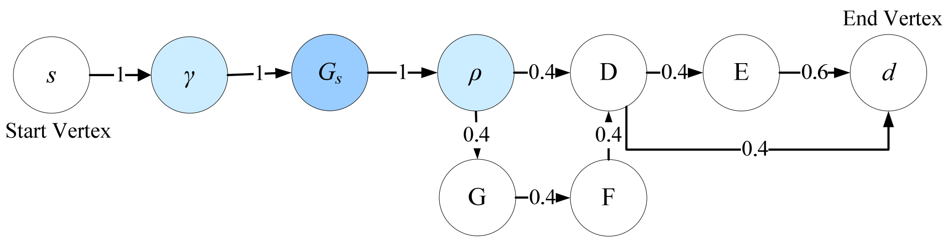


Fig. 24: The final graph G_N of the GKF model

The edge deletion procedure has several properties. We define and prove the associated lemmas below.

Lemma 1: Let v_s be the start vertex in a graph, G_N, of a group-based knowledge flow. For any vertex v_h in G_N, there exists a path $P_{s,h}$ from vertex v_s to v_h.

Proof: In the edge deletion procedure, removal of an incoming edge from a vertex v_h depends on the weight of the edge. All vertices in G_N are visited in topological order and their incoming edges are examined. For any vertex v_h, an incoming edge should be removed if its weight is no greater than a user-specified threshold. However, if removing an edge from v_h also removes the path $P_{s,h}$ from G_N, that edge should be returned to the vertex.

When deleting an incoming edge of a vertex, the edge deletion procedure ensures that 1) there is a path $P_{s,h}$ from the start vertex v_s to vertex v_h; and 2) removing an incoming edge from a successor of v_h does not affect the path $P_{s,h}$. The proof is as follows. Let a vertex v_k be a succeeding vertex of v_h in the topological order. Based on the topological order, the edge deletion procedure processes the vertex v_h before vertex v_k and there exists a path $P_{s,h}$. Assume that a path $P_{s,h}$ does not exist from v_s to v_h, because an incoming edge of v_k has been deleted. Thus, a path must have existed from vertex v_s through v_k to v_h before the edge was deleted. Consequently, v_k must be a predecessor of v_h. However, this statement contradicts the algorithm's processing of vertices in topological order. That is, v_k is a succeeding vertex of v_h and the path $P_{s,h}$ exists in G_N. Thus, removing an incoming edge from a succeeding vertex of v_h does not affect the path $P_{s,h}$. According to the algorithm and the above explanation, for any vertex v_h in G_N, there exists a path $P_{s,h}$ from vertex v_s to v_h.

Lemma 2: Let v_d be an end vertex in the graph of the group-based knowledge flow G_N. For any vertex v_h in G_N, there exists a path $P_{h,d}$ from vertex v_h to v_d.

Proof: Let vertex v_k be the succeeding vertex of the vertex v_h. Removing an incoming edge of vertex v_k will affect the reachability of the end vertex v_d from vertex v_h. When the edge deletion procedure removes an incoming edge of vertex v_k, it has to check whether the path $P_{h,d}$ from vertex v_h to the end vertex v_d exists. If it does not exist, the incoming edge should not be removed. Therefore, the procedure ensures that a path $P_{h,d}$ exists from vertex v_h to the end vertex v_d.

Lemma 3: Let $G_N = \{V_N, E_N\}$ be the directed graph of a group-based knowledge flow. All vertices in V_N can be visited by traversing vertices from the start vertex v_s to the end vertex v_d. Then, for any vertex v_h in V, there exists a path from v_s to v_d through v_h.

Proof: According to Lemma 2 and Lemma 3, for any vertex v_h in V_N, there exists a path $P_{s,h}$

from the start vertex v_s to v_h and a path $P_{v,d}$ from v_h to end vertex v_d. Therefore, there exists a path from v_s to v_d through v_h.

Lemma 4: For any infrequent edge $e_{h,k}$ on an infrequent path of G_N, either the path from the start vertex v_s to vertex v_k or the path from the vertex v_h to the end vertex v_d must pass through the edge $e_{h,k}$.

Proof: Let vertex v_h be a predecessor of vertex v_k in the topological order, and let $e_{h,k}$ be an infrequent edge from vertex v_h to vertex v_k in G_N. Assume that there exist two paths, one from start vertex v_s to vertex v_k and the other from vertex v_h to the end vertex $v_{d,}$ neither of which passes through the edge $e_{h,k}$. Our algorithm removes any infrequent edge if doing so will not make any vertex unreachable. Thus, the algorithm will remove the edge $e_{h,k}$. However, this contradicts the statement that $e_{h,k}$ exists in G_N. Consequently, for any infrequent edge $e_{h,k}$ of an infrequent path of G_N, either the path from the start vertex v_s to vertex v_k or the path from the vertex v_h to the end vertex v_d must pass through the edge $e_{h,k}$.

The vertex V_{GS} in graph G_N represents a corresponding strongly connected component G_S in G. All vertices in G_S with parallel relations or sequential relations are reachable. Lemmas 2, 3, 4 and 5 also hold for G.

5.5 The GKF Mining Algorithm for Dealing with Topic Loops

The GKF mining algorithm for dealing with topic loops (GKF-TL) is based on the GKF algorithm introduced in Section 5.4, which assumes there are no topic loops in workers' KFs when it generates the graph of the group-based KF. A topic loop means that a specific topic appears repeatedly in a TKF because it is referenced by a worker several times. This may happen because the worker needs the knowledge at different times during a task's execution. For example, given a worker's topic-level KF <A, B, A, C, D>, if topic A is referenced twice, it is appears as a topic loop in the corresponding graph of the TKF. Because the loop problem in a workflow mining domain is difficult to resolve, no matter what the application domain, many researchers ignore the problem [23, 56]. Agrawal et al. [7] proposed an algorithm for workflow systems that builds a general directed graph with cycles for mining process models from workflow logs. The algorithm gives activities different labels to differentiate them in a workflow instance. The problem of dealing with topic loops in TKFs is analogous to that of workflow systems. Thus, we adopt the above approach to solve the loop problem. Specifically, we propose an algorithm that considers duplicate topics (topic loops) in each TKF to build a directed graph for modeling the referencing behavior of a group of workers.

The GKF-TL algorithm differs from the GKF algorithm. First, it identifies duplicated topics in a TKF and gives them different labels in order to solve the loop problem. For example, given a KF <B, A, B, C, B>, because topic B appears three times, it is transformed into three instances, i.e., B1, B2 and B3, such that the original KF becomes <B1, A, B2, C, B3>.

After infrequent edges have been removed from the graph *G,* it is transformed into a new graph G_T as follows. The vertices with different instances of the same topic form an equivalent set and can be merged to make one vertex. For a topic *TP* in a TKF, each vertex in the equivalent set of *TP* is an instance of the topic. Then, a directed edge is added to the new graph G_T if there is an edge between two vertices of different equivalent sets in graph *G*. Initially, the merging process is applied to vertices of each equivalent set in *G* when a strongly connected component is not involved. To merge vertices involving a strongly connected component G_s, the steps are as follows.

Let vertices v_i/ v_j be instances in the equivalent sets Q_a / Q_b, and let v_k be an another instance in Q_a as well as a vertex in a strongly connected component, i.e., $v_k \in G_s$, where v_γ and v_ρ are two pseudo nodes of G_s. Note that because v_k and v_i are instances of the same topic, they are in the same equivalent set and are thus merged to form one vertex. In addition, v_i is in G_s, since v_k is in G_s. Generally, the vertices of an equivalent set Q_a in *G* are combined as a vertex v_a in the new graph G_T, while the vertices of an equivalent set Q_b are merged to form one vertex v_b. For a strongly connected component G_s with pseudo nodes v_γ and v_ρ, if a directed edge $e_{i,j}$ between v_i and v_j exists in *G*, a directed edge $e_{\rho,b}$ is added to the new graph G_T. Similarly, if a directed edge $e_{j,i}$ exists in *G*, a directed edge $e_{b,\gamma}$ is added to graph G_T.

Next, we consider how to combine vertices involving two strongly connected components. Let v_k / v_l be vertices in strongly connected components G_a / G_b; $v_{\gamma a}$ and $v_{\rho a}$ be pseudo vertices that connect with graph G_a; $v_{\gamma b}$ and $v_{\rho b}$ be pseudo vertices that connect with G_b; and Q_a / Q_b be the corresponding equivalent sets of vertices in G_a / G_b. In addition, let vertex v_i and v_k (resp. v_j and v_l) be instances of the equivalent sets Q_a (resp. Q_b). Vertices in Q_a / Q_b are merged as vertex v_a / v_b. Because v_k / v_l is in G_a / G_b, v_i / v_j also belongs to G_a / G_b; however, some edges need to be adjusted. If there is a directed edge $e_{i,j}$ from v_i to v_j in graph G, an edge $e_{\rho a,\gamma b}$ with the same direction as edge $e_{i,j}$ is added to the new graph G_T. Similarly, if a directed edge $e_{j,i}$ exists in graph G, a directed edge $e_{\rho b,\gamma a}$ is added to G_T. These new added edges are used to merge two equivalent sets in different strongly connected components and make a connection between them. Note that the weights of the edges are updated during the merging process.

Note that we assume the instances of a topic exist in at most one strongly connected component after the vertices of each equivalent set have been merged to form one vertex. We defer consideration of the case where the same topic belongs to more than one strongly connected component to a future work. Next, we provide an example of implementing the GKF-TL algorithm.

5.5.1 Applying the GKF Mining Algorithm for Dealing with Topic loops

The following example considers a group of four workers with similar KFs. Their topic-level KFs (TKFs) are listed in Table 5. Each element in a TKF is used to represent a topic domain. Thus, the elements in a TKF are arranged as a topic sequence based on the times they were referenced. As a topic may appear more than once in a specific KF, because the worker needs the knowledge at different times, we apply the GKF-TL mining algorithm to deal with topic loops.

Table 5: The TKFs of four workers

Worker	Topic-level KF (TKF)	TKF’
John	<A, B, A, C, D, F>	<A1, B1, A2, C, D, F>
Mary	<B, A, B, C, D>	<B1, A1, B2, C, D>
Lisa	<B, A, D, F>	<B1, A1, D, F>
Tom	<A, B, A, E, G, D>	<A1, B1, A2, E, G, D>

In Table 5, a topic that appears more than once in a TKF is labeled as a different instance of the topic, and a TKF with duplicate topics is transformed into a TKF’. Then, the algorithm uses TKF’ to build the initial graph of the GKF model. In this example, we set the user-specified thresholds for topic relation identification and edge deletion as $\varepsilon = 1$ and $\theta =$

0.3 respectively. The initial graph derived before graph transformation is shown in Fig. 25. A strongly connected component is discovered in the initial graph. To resolve the vertex relation problem in the strongly connected component, the algorithm applies the topic relation identification procedure detailed in Fig. 18. The vertex relation in the strongly connected component is shown in G_s in Fig. 25. The number on each edge represents the edge's weight. Recall that the weight is derived by Eq. (20) to indicate the importance of the edge.

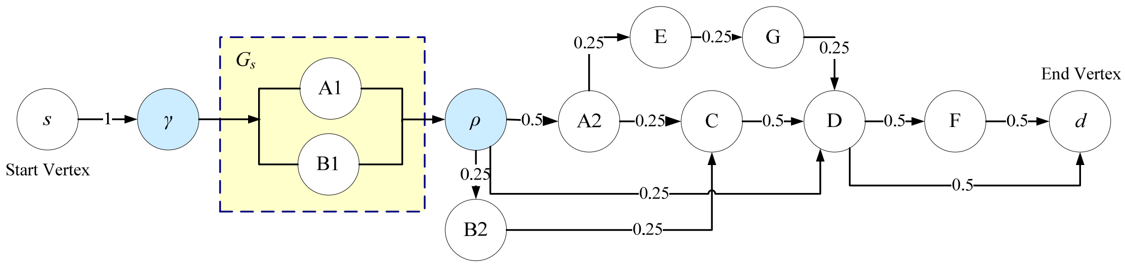


Fig. 25: The initial graph of the GKF model with topic loops

Fig. 26 shows the result of removing the infrequent edges from the graph in Fig. 25. The sub-graph G_s in the initial graph is transformed into a vertex v_{Gs}.; and the edge that connects a vertex in G_s with another vertex, i.e., $e_{\rho,D}$, is removed because its weight is no greater than 0.3.

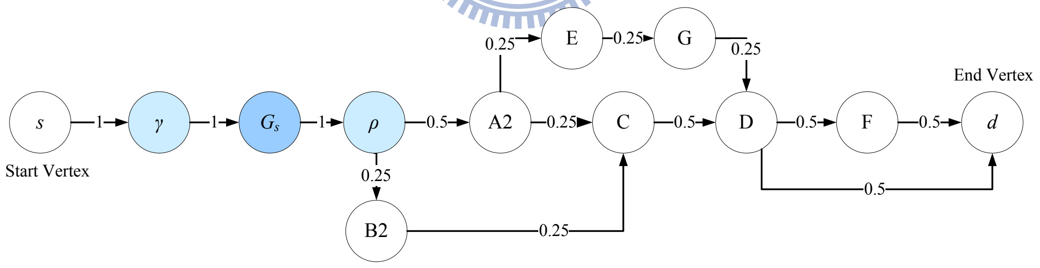


Fig. 26: The graph of the GKF model with topic loops

Finally, the algorithm merges vertices that are different instances of the same topic into one vertex. For example, in Fig. 25, vertices v_{B1} and v_{B2} are different instances of the same topic, so they are merged to form the vertex v_B. Moreover, the edge $e_{\rho,B2}$ is replaced by an edge connecting v_ρ to v_γ; and the edge $e_{B2,C}$ is changed to edge $e_{\rho,c}$. The vertices v_{A1} and v_{A2} are two instances of topic A; hence they are merged to form vertex v_A, and their edges are changed accordingly. Fig. 27 shows the final GKF graph, which considers the duplicate topics in each worker's TKF. To illustrate all knowledge paths in the graph, the vertex v_{Gs} is converted into the original graph G_s.

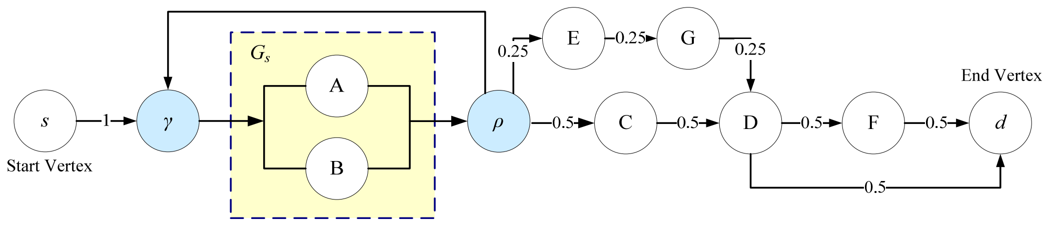


Fig. 27: The final GKF graph, which considers the duplicate topics in each worker's TKF

5.6 Identifying Knowledge Referencing Paths in a GKF Graph

We have developed a method for identifying frequent knowledge paths from the GKF graph to describe the information needs of a group of workers, i.e. their knowledge referencing behavior. A knowledge path, which represents the knowledge referencing behavior of a group of workers, consists of several vertices and edges that can be traversed from the start vertex to the end vertex. To identify a frequent knowledge path, a path score derived from the weights of the edges on a path is used to evaluate each path and indicate its importance, as defined in Eq. (21).

$$ps_i = Min\{we_{x,y} \mid \forall e_{x,y} \in path_i\}\,, \qquad (21)$$

where ps_i is the path score of the path i; and $we_{x,y}$ is the weight of edge $e_{x,y}$, which belongs to the path i and represents a direct flow relation between vertex x and vertex y. Based the weights of all the edges on a specific path, a path score is derived from the minimal weight among the edges to indicate the path's level of importance. Note that the edge weight derived by Eq. (20) denotes the importance of the direct flow in a GKF. A large edge weight means that the referencing flow between topics is highly significant for the group of workers.

Paths with scores higher than a user-specified threshold are regarded as frequent knowledge paths in the GKF and are selected for the group. Specifically, such knowledge paths (patterns) are used to represent the frequent knowledge referencing behavior of workers and important knowledge flows. The discovered paths will be important references for workers, while the frequent knowledge paths also will help novices learn group-related knowledge. The following example illustrates the computation of the path score.

5.7 The Prototype System for Mining Group-based Knowledge Flows

In this Chapter, we develop a prototype system to demonstrate the proposed methods for mining group-based knowledge flows (GKFs), which are generally difficult to formalize. To address the problem, our system provides a mining function and modules to identify GKFs easily and effectively. In addition, a GKF is modeled as a graph to represent the referenced topics, the directions of knowledge flows, and the knowledge referencing paths (patterns) for a group of workers with similar KFs. The referencing paths with scores higher than a user-specified threshold are identified to represent the frequent knowledge referencing patterns of the group. We describe the real-world dataset used in our system in Section 5.7.1, present the implementation of our prototype system in Section 5.7.2 and discuss the contributions of this work in Section 5.7.3.

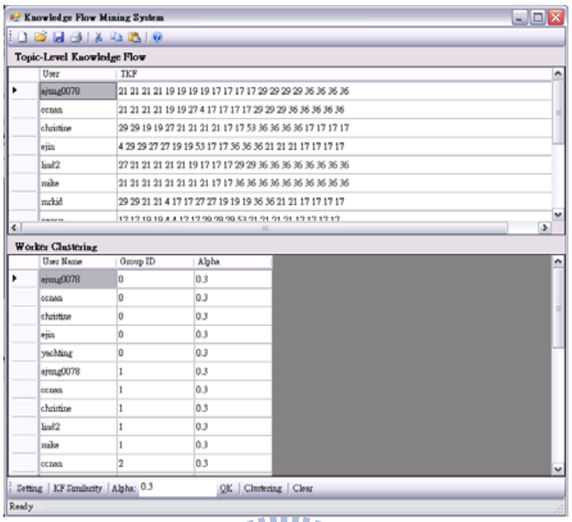
5.7.1 Dataset

We use a dataset from a research laboratory in a research institute. It contains information about 14 knowledge workers, 424 research documents, and a usage log that records the times documents were accessed and the workers' document preferences. Each worker may perform a number of tasks, e.g., conducting a research project and writing research papers, and the research documents are the codified knowledge needed to perform the tasks. Because a worker' information needs may change over time, the access time of documents can be used to track changes in his/her information needs for a specific task, and his/her knowledge referencing behavior can be identified.

5.7.2 System Implementation

To implement our prototype system for group-based KF mining, we use Microsoft Visual Studio 2005 (with C#) to develop the system and Microsoft SQL Server 2005 as the database system to storing the dataset. Because the dataset contains workers' logs, it should be preprocessed to generate each worker's codified-level KF and topic-level KF. To obtain the KF, documents in the dataset are grouped into eight clusters by using a single-link clustering method. Based on the clustering results, a topic-level KF is generated by mapping the codified knowledge into its corresponding clusters for each knowledge worker. Then, the two types of KF, the topic-level KF and the codified-level KF, are derived to describe the information needs of a worker. We use such KFs to build a prototype system to demonstrate the method for mining the knowledge flows of a group of workers.

Our system has two major functions: worker clustering and group-based knowledge flow mining. The former identifies a group’s knowledge flow, and the latter uses a directed acyclic graph to present the mining results. An interface that can visualize the KF is necessary. Note that our system can be applied in any knowledge intensive organization to help workers obtain and learn knowledge. Next, we describe the system in detail.



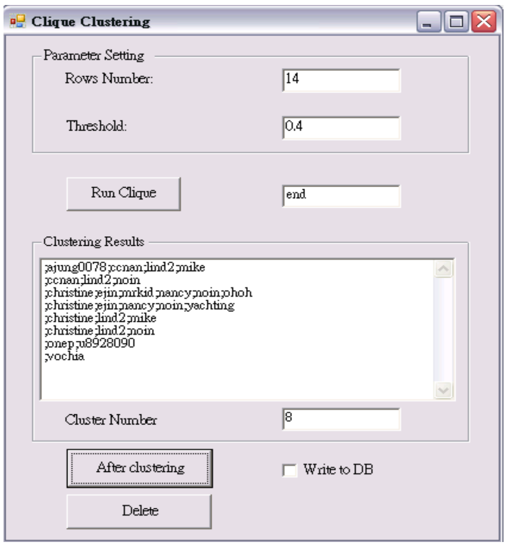
User	TKF
ajung0078	21 21 21 21 19 19 19 19 17 17 17 17 29 29 29 29 36 36 36 36
ccnan	21 21 21 21 19 19 27 4 17 17 17 17 29 29 29 36 36 36 36 36
christine	29 29 19 19 27 21 21 21 21 17 17 53 36 36 36 36 17 17 17 17
ejin	4 29 29 27 27 19 19 53 17 17 36 36 36 21 21 21 17 17 17 17
lind2	27 21 21 21 21 21 19 17 17 17 29 29 36 36 36 36 36 36 36 36
mike	21 21 21 21 21 21 21 21 17 17 36 36 36 36 36 36 36 36 36 36
mrkid	29 29 21 21 4 17 17 27 27 19 19 19 36 36 21 21 17 17 17 17
[illegible]	17 17 19 19 4 4 17 17 29 29 29 53 21 21 21 21 17 17 17 17

User Name	Group ID	Alpha
ajung0078	0	0.3
ccnan	0	0.3
christine	0	0.3
ejin	0	0.3
yachting	0	0.3
ajung0078	1	0.3
ccnan	1	0.3
christine	1	0.3
lind2	1	0.3
mike	1	0.3
ccnan	2	0.3

Alpha: 0.3

Fig. 28: The main frame of the KF mining system

The knowledge flow mining system is comprised of three modules: the main module, the CLIQUE clustering module and the GKF model. Each module has functions to help the user (a manager/worker) build a knowledge flow easily. Fig. 28 shows the main frame of the system, which provides essential functions for building the GKF model, e.g., the system settings, the KF alignment similarity and clustering functions. The system setting is used to initialize the system environment, e.g., database selection. The KF similarity function calculates the similarity between two workers’ knowledge preferences based on their knowledge flows and creates a similarity matrix of the workers. The parameter alpha adjusts the relative importance of the KF alignment similarity and the aggregated profile similarity on a scale of 0 to 1, as shown in Eq. (8). The user can specify the value of alpha and use the KF similarity function to create a KF similarity matrix based on the specified value. Then, the CLIQUE clustering method uses the similarity matrix to cluster workers who have similar KFs. The system also provides an interface to show the topic-level KFs of all workers and the results of worker clustering. To simplify the presentation of the KFs, we use a number to represent a topic domain that consists of topic-related terms.



Parameter Setting
Rows Number: 14
Threshold: 0.4

end

Clustering Results

```
;ajung0078;ccnan;lind2;mike
;ccnan;lind2;noin
;christine;ejin;mrkid;nancy;noin;ohoh
;christine;ejin;nancy;noin;yachting
;christine;lind2;mike
;christine;lind2;noin
;onep;u8928090
;vochia
```

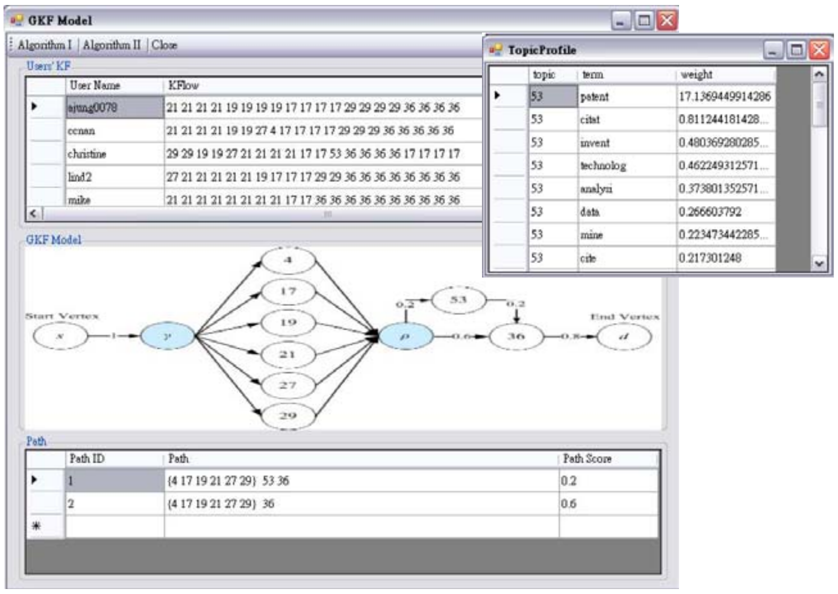
Cluster Number: 8

Write to DB: [ ]

Fig. 29: The CLIQUE clustering module

Fig. 29 shows the CLIQUE clustering module. Before using the module, we have to set two parameters: the number of rows in the KF similarity matrix and the clustering threshold. The number of rows is used to determine the number of times clustering is performed using the CLIQUE clustering method, while the threshold is used to cluster workers whose similarity scores are higher than a certain value. Then, the clustering result is displayed on the system interface. For example, to perform clustering, the value of alpha is set at 0.3, the number of rows of the KF similarity matrix is 14 and the similarity threshold is set at 0.4. Each group is comprised of several workers, and each worker belongs to several task-based groups based on the KF similarities. After clustering similar workers, the system stores the clustering results in the database for further utilization and analysis.

Next, using the proposed algorithm, the system builds a group-based knowledge flow (GKF) for a group of workers, as shown in Fig. 30. All the workers in a cluster have similar KFs, which are used to generate a GKF graph to characterize the referencing behavior of the group. In the graph, each circle is a topic domain represented by a number, while each directed edge indicates the flow of knowledge between two topics. The topic domain contains a topic profile, which consists of several representative terms and their term weights. Fig. 30 shows the profile of topic domain 53 in a small window. The listed terms represent the knowledge of the topic.



	User Name	KFlow
▶	ajung0078	21 21 21 21 19 19 19 19 17 17 17 17 29 29 29 29 36 36 36 36
	ccnan	21 21 21 21 19 19 27 4 17 17 17 17 29 29 29 36 36 36 36 36
	christine	29 29 19 19 27 21 21 21 21 17 17 53 36 36 36 36 17 17 17 17
	lind2	27 21 21 21 21 21 19 17 17 17 29 29 36 36 36 36 36 36 36 36
	mike	21 21 21 21 21 21 21 21 17 17 36 36 36 36 36 36 36 36 36 36

	topic	term	weight
▶	53	patent	17.1369449914286
	53	citat	0.811244181428...
	53	invent	0.480369280285...
	53	technolog	0.462249312571...
	53	analyzi	0.373801352571...
	53	data	0.266603792
	53	mine	0.223473442285...
	53	cite	0.217301248

	Path ID	Path	Path Score
▶	1	{4 17 19 21 27 29} 53 36	0.2
	2	{4 17 19 21 27 29} 36	0.6
*			

Fig. 30: The GKF graph and knowledge referencing paths for a specific group

In addition, the number on an arrow indicates the importance of a flow relation in this group's topics. From the GKF graph, we observe that 6 topics, i.e., 4, 17, 19, 21, 27, and 29, can be referenced in parallel. That is, there is no specific order among the topics accessed by this group of workers. Moreover, the task-related knowledge may flow through 2 paths from the start vertex to the end vertex. In Fig. 30, the listed paths, which consist of several relevant topics and directed edges, are the knowledge referencing paths of this group. The paths with scores larger than a user-specified threshold are frequent referencing behavior patterns. The paths can be regarded as knowledge references for workers to share needed task knowledge.

5.7.3 Discussion

GKF mining by task-based groups has several advantages in a knowledge intensive organization. A GKF represents the flow and delivery of knowledge when workers in the same group perform a task. It can be used to identify topics of interest, major referencing behavior patterns, and the long-term evolution of the group's information needs; and it allows task knowledge to be circulated and delivered efficiently among workers. If a novice joins the group, the GKF can provide a reference for learning group-based knowledge. The frequent knowledge paths in a GKF help a worker learn task-related knowledge, overcome obstacles encountered in a new domain, and enhance his/her learning efficiency. Moreover, based on the GKF, a manager can determine who has task-related knowledge and who satisfies a task's requirements, and then assign appropriate workers accordingly. In addition, through the GKF, an organization can realize the frequent referencing behavior and the information needs of a group of workers, and actively provide knowledge support for them. The GKF can also enhance organizational learning, as well as facilitate knowledge sharing and reuse in the context of collaboration and teamwork.

In this work, we propose a recommendation framework based on the discovered knowledge flow for each knowledge worker, as described in Chapter 4. Such method analyzes workers' referencing behavior and provides task-related documents to fulfill workers' tasks. Because teamwork in an organization is common, we also develop a group-based knowledge flow mining algorithm that analyzes workers' information needs from a group perspective and model the referencing behavior of a group as a knowledge graph. In our future work, we will apply the recommendation techniques on the group-based knowledge flow to provide knowledge support for workers in a teamwork environment.

Chapter 6. Conclusions and Future works

6.1 Summary

Knowledge is both abstract and dynamic. A worker's knowledge flow (KF) comprises a great deal of working knowledge that is difficult to acquire from an organizational knowledge base. In this dissertation, we have considered how to identify the knowledge flow of knowledge workers, and how to provide knowledge support based on KFs effectively. To the best of our knowledge, no existing approach focuses on providing relevant knowledge proactively based on KFs.

We propose KF-based recommendation methods, namely hybrid PCF-KSR, KCF-KSR and ICF-KSR methods, to proactively recommend codified knowledge for knowledge workers and enhance the quality of recommendations. These methods use KF-based sequential rule (KSR) method to recommend topics by considering workers' knowledge referencing behavior; and then adjust the predicted rating of documents belonging to the recommended topic. Moreover, they consider workers' preferences for codified knowledge, as well as their knowledge referencing behavior to predict topics of interest and recommend task-related knowledge. The collaborative filtering (CF) method, which is widely used to predict a target worker's preferences based on the opinions of similar workers, only considers workers' preferences for codified knowledge, but it neglects workers' referencing behavior for knowledge.

In the experiments, we evaluate the quality of recommendations derived by the proposed methods under various parameters and compare it with that of the traditional user-based/item-based CF method. The experiment results show that the proposed methods improve the quality of document recommendation and outperform the traditional CF methods. Additionally, using KF mining and sequential rule mining techniques enhances the performance of recommendation methods and increases the accuracy of recommendations. The KF-based recommendation methods provide knowledge support adaptively based on the referencing behavior of workers with similar KFs, and also facilitate knowledge sharing among such workers.

Furthermore, we have proposed the group-based KF mining method to identify the KFs of groups of workers. Such groups may be interest groups or communities, where the workers have very similar KFs. A group may comprise many workers with similar KFs, and a worker may join many groups simultaneously according to his/her information needs. Even though workers are in the same group, their KFs will differ in some respects. To discover the KF of a group of workers, we design algorithms that can analyze the workers information needs in their KFs to generate a GKF model. The model is then used to represent the information needs, the direction of knowledge flows, and possible paths for referencing task knowledge for a group of workers. Based on the model, we can identify representative paths as common behavior patterns for the group. Thus, the patterns can be regarded as learning references to help new members of a group. Finally, we implement a prototype system to demonstrate the efficacy of the proposed algorithms. Our system not only derives the KF for a group of workers, but also visualizes the mining results for further analysis.

6.2 Future Works

In our current work, a KF is simply regarded as a set of topics/codified knowledge objects arranged in a time sequence. However, a KF may have a complicated order structure with AND/OR, JOIN and SPLIT operations. In our future work, we will investigate a complex KF mining technique to model workers' KFs with an order structure that includes such operations. Moreover, the discovered topic is regarded as an abstraction of topic-related documents. Auto-summarization techniques [45, 49] can be applied to extract the theme of a topic by summarizing the documents' contents. In a future work, we will investigate the use of such techniques to derive knowledge flows based on theme information. In addition, the domain restricted the sample size of the data and the number of participants in the experiments, since it is difficult to obtain a dataset that contains information that can be used for knowledge flow mining. We will evaluate the proposed approach on other application domains involving larger numbers of workers, tasks and documents. Moreover, the method of generating topic subsequences for identifying the target worker's knowledge referencing behavior is computationally expensive, especially for the large datasets. A more efficient method will be investigated in the future.

Additionally, we will develop a recommendation method based on the GKF, so that workers can cooperate and share their knowledge with other group members to accomplish a task. Moreover, different working groups in an organization may provide knowledge support for one another. To facilitate knowledge sharing in a group or among groups, we will investigate recommendation methods that provide task knowledge to workers and groups proactively. The effectiveness of a recommendation method depends to a large extent on how much workers trust one another. This factor is important because the level of trust may determine whether or not a worker is willing to share knowledge with others. Through group recommendation methods, task-related knowledge can be shared effectively to enhance the work efficiency of all knowledge workers.

References

[1] A. Abecker, A. Bernardi, K. Hinkelmann, O. Kuhn, and M. Sintek, "Context-Aware, Proactive Delivery of Task-Specific Information: The KnowMore Project," *Information Systems Frontiers,* vol. 2, no. 3, pp. 253-276, 2000.

[2] A. Abecker, A. Bernardi, H. Maus, M. Sintek, and C. Wenzel, "Information Supply for Business Processes: Coupling Workflow with Document Analysis and Information Retrieval," *Knowledge-Based Systems,* vol. 13, pp. 271-284, 2000.

[3] R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules Between Sets of Items in Large Databases," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 207-216, 1993.

[4] R. Agrawal, and R. Srikant, "Fast Algorithms for Mining Association Rules in Large Databases," *Proceedings of the 20th International Conference on Very Large Data Bases*, pp. 487-499, 1994.

[5] R. Agrawal, and R. Srikant, "Mining Sequential Patterns," *Proceedings of the Eleventh International Conference on Data Engineering*, pp. 3-14, 1995.

[6] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications," in *Proceedings of the International Conference on Management of Data (ACM SIGMOD)*, pp. 94-105, 1998.

[7] R. Agrawal, D. Gunopulos, F. Leymann, and G. Boblingen, "Mining Process Models from Workflow Logs," in *6th International Conference on Extending Database Technology (EDBT'98)*, Valencia, Spain, pp. 469-483, 1998.

[8] A. Anjewierden, R. de Hoog, R. Brussee, and L. Efimova, "Detecting Knowledge Flows in Weblogs," in *13th International Conference on Conceptual Structures (ICCS)*, pp. 1-12, 2005.

[9] C. Augusto, F. Maria Grazia, and P. Silvano, "Knowledge-based Document Retrieval in Office Environments: the Kabiria System," *ACM Transactions on Information Systems,* vol. 13, no. 3, pp. 237-268, 1995.

[10] R. Baeza-Yates, and B. Ribeiro-Neto, *Modern Inofrmation Retrieval*, Boston: Addison-Wesley, 1999.

[11] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering," in *Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence*, pp. 43-52, 1998.

[12] J. S. Brown, and P. Duguid, *The Social Life of Information*, Boston, MA, USA: Harvard Business School Press, 2002.

[13] J. Cardoso, and M. Lenic, "Web Process and Workflow Path Mining Using the Multimethod Approach," *International Journal of Business Intelligence and Data*

Mining, vol. 1, no. 3, pp. 304-328, 2006.

[14] K. Charter, J. Schaeffer, and D. Szafron, "Sequence Alignment Using FastLSA," in *Proceedings of the 2000 International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences (METMBS,2000)*, pp. 239-245, 2000.

[15] Y. B. Cho, Y. H. Cho, and S. H. Kim, "Mining Changes in Customer Buying Behavior for Collaborative Recommendations," *Expert Systems with Applications,* vol. 28, no. 2, pp. 359-369, 2005.

[16] S. L. Chuang, and L. F. Chien, "A Practical Web-based Approach to Generating Topic Hierarchy for Text Segments," in *Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management (CIKM)*, Washington, D.C., USA, pp. 127-136, 2004.

[17] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*: MIT Press, 2001.

[18] A. K. A. de Medeiros, B. F. van Dongen, W. M. P. van der Aalst, and A. Weijters, "Process Mining: Extending the a-algorithm to Mine Short Loops," *BETA Working Paper Series, WP,* vol. 113, 2004.

[19] A. N. Dragunov, T. G. Dietterich, K. Johnsrude, M. McLaughlin, L. Li, and J. L. Herlocker, "TaskTracer: a Desktop Environment to Support Multi-tasking Knowledge Workers," in *Proceedings of the 10th international conference on Intelligent User Interfaces*, San Diego, California, USA, pp. 75-82, 2005.

[20] R. C. Dubes, and A. K. Jain, *Algorithms for Clustering Data*: Prentice Hall, Inc., 1988.

[21] R. Feldman, and J. Sanger, *The Text Mining Handbook: Advanced Approaches to Analyzing Unstructured Data*, New York, USA: Cambridge University Press, 2007.

[22] N. Glance, D. Arregui, and M. Dardenne, "Knowledge Pump: Community-centered Collaborative Filtering," in *Proceedings of the Fifth DELOS Workshop on Filtering and Collaborative Filtering*, pp. 83-88, 1998.

[23] G. Greco, A. Guzzo, G. Manco, and D. Sacca, "Mining and Reasoning on Workflows," *IEEE Transactions on Knowledge and Data Engineering,* vol. 17, no. 4, pp. 519-534, 2005.

[24] B. Hay, G. Wets, and K. Vanhoof, "Clustering Navigation Patterns on a Website Using a Sequence Alignment Method," in *Proceedings of 17th International Joint Conference on Artificial Intelligence (IJCAI)*, Seattle, Washington, pp. 1-6, 2001.

[25] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An Algorithmic Framework for Performing Collaborative Filtering," in *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Berkeley, California, United States, pp. 230-237, 1999.

[26] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating Collaborative Filtering Recommender Systems," *ACM Transactions on Information Systems (TOIS),* vol. 22, no. 1, pp. 5-53, 2004.

[27] H. Holz, H. Maus, A. Bernardi, and O. Rostanin, “A Lightweight Approach for Proactive, Task-Specific Iinformation Delivery,” *Proceedings of the 5th International Conference on Knowledge Management (I-Know)*, 2005.

[28] S. Y. Hwang, C. P. Wei, and W. S. Yang, “Discovery of Temporal Patterns From Process Instances,” *Computers in Industry,* vol. 53, no. 3, pp. 345-364, 2004.

[29] A. K. Jain, M. N. Murty, and P. J. Flynn, “Data Clustering: a Review,” *ACM Computing Surveys (CSUR),* vol. 31, no. 3, pp. 264-323, 1999.

[30] S. C. Johnson, “Hierarchical Clustering Schemes,” *Psychometrika,* vol. 32, no. 3, pp. 241-254, 1967.

[31] A. B. Kahn, “Topological sorting of large networks,” *Communications of the ACM,* vol. 5, no. 11, pp. 558-562, 1962.

[32] L. Kaufman, and P. J. Rousseeuw, "Finding Groups in Data. an Introduction to Cluster Analysis," *Wiley Series in Probability and Mathematical Statistics. Applied Probability and Statistics, New York: Wiley*, 1990.

[33] S. Kim, H. Hwang, and E. Suh, “A Process-based Approach to Knowledge-flow Analysis: a Case Study of a Manufacturing Firm,” *Knowledge and Process Managment,* vol. 10, no. 4, pp. 260-276, 2003.

[34] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl, “GroupLens: Applying Collaborative Filtering to Usenet News,” *Communications of the ACM,* vol. 40, no. 3, pp. 77-87, 1997.

[35] J. B. Kruskal, “An Overview of Sequence Comparison: Time Warps, String Edits, and Macromolecules,” *SIAM Review,* vol. 25, no. 2, pp. 201-237, 1983.

[36] C.-H. Lai, and D.-R. Liu, “Integrating Knowledge Flow Mining and Collaborative Filtering to Support Document Recommendation,” *Journal of Systems and Software,* vol. 82, pp. 2023-2037, 2009.

[37] G. Linden, B. Smith, and J. York, “Amazon.com Recommendations: Item-to-Item Collaborative Filtering,” *IEEE INTERNET COMPUTING*, pp. 76-80, 2003.

[38] D.-R. Liu, and Y.-Y. Shih, “Hybrid Approaches to Product Recommendation Based on Customer Lifetime Value and Purchase Preferences,” *Journal of Systems and Software,* vol. 77, no. 2, pp. 181-191, 2005.

[39] D.-R. Liu, I.-C. Wu, and K.-S. Yang, “Task-based K-Support System: Disseminating and Sharing Task-Relevant Knowledge,” *Expert Systems with Applications,* vol. 29, no. 2, pp. 408-423, 2005.

[40] D.-R. Liu, C.-H. Lai, and C.-W. Huang, “Document Recommendation for Knowledge Sharing in Personal Folder Environments,” *Journal of Systems and Software,* vol. 81, no. 8, pp. 1377-1388, 2008.

[41] H. Mannila, and P. Ronkainen, “Similarity of Event Sequences,” in *Proceedings of the Fourth International Workshop on Temporal Representation and Reasoning*, Florida, USA, pp. 136-139, 1997.

[42] I. Nonaka, and H. Takeuchi, *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*: Oxford University Press, 1995.

[43] S. G. Oguducu, and M. T. Ozsu, "Incremental Click-stream Tree Model: Learning From New Users for Web Page Prediction," *Distributed and Parallel Databases,* vol. 19, no. 1, pp. 5-27, 2006.

[44] M. Polanyi, "The Tacit Dimension," Doubleday New York, 1966.

[45] D. R. Radev, H. Jing, M. Stys, and D. Tam, "Centroid-based Summarization of Multiple Documents," *Information Processing & Management,* vol. 40, no. 6, pp. 919-938, 2004.

[46] O. M. Rodriguez, A. I. Martinez, J. Favela, A. Vizcaino, and M. Piattini, "Understanding and Supporting Knowledge Flows in a Community of Software Developers," in *International Workshop on Groupware (CRIWG)*, pp. 52-66, 2004.

[47] J. Rucker, and M. J. Polanco, "Siteseer: Personalized Navigation for the Web," *Communications of the ACM,* vol. 40, no. 3, pp. 73-76, 1997.

[48] G. Salton, and C. Buckley, "Term-weighting Approaches in Automatic Text Retrieval," *Information Processing & Management,* vol. 24, no. 5, pp. 513-523, 1988.

[49] G. Salton, A. Singhal, M. Mitra, and C. Buckley, "Automatic Text Structuring and Summarization," *Information Processing & Management,* vol. 33, no. 2, pp. 193-207, 1997.

[50] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, "Item-based Collaborative Filtering Recommendation Algorithms," in *Proceedings of the 10th international conference on World Wide Web*, Hong Kong, pp. 285-295, 2001.

[51] U. Shardanand, and P. Maes, "Social Information Filtering: Algorithms for Automating "Word of Mouth"," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '95)*, Denver, Colorado, United States, pp. 210-217, 1995.

[52] W. M. P. van der Aalst, and M. Song, "Mining Social Networks: Uncovering Interaction Patterns in Business Processes," *Lecture Notes in Computer Science,* vol. 3080, pp. 244-260, 2004.

[53] W. M. P. van der Aalst, and A. Weijters, "Process Mining: a Research Agenda," *Computers in Industry,* vol. 53, no. 3, pp. 231-244, 2004.

[54] W. M. P. van der Aalst, T. Weijters, and L. Maruster, "Workflow Mining: Discovering Process Models From Event Logs," *IEEE Transactions on Knowledge and Data Engineering,* vol. 16, no. 9, pp. 1128-1142, 2004.

[55] W. M. P. van der Aalst, V. Rubin, B. F. van Dongen, E. Kindler, and C. W. Gunther, "Process Mining: A Two-Step Approach Using Transition Systems and Regions," *BPM Center Report BPM-06-30*, Department of Technology Management, Eindhoven University of Technology 2006.

[56] B. F. van Dongen, and W. M. P. van der Aalst, "Multi-Phase Process Mining:

Building Instance Graphs," in *International Conference on Conceptual Modeling (ER 2004)*, pp. 362-376, 2004.

[57] C. J. Van RijsBergen, *Information Retrieval*, London: Butterworths, 1979.

[58] I. C. Wu, D. R. Liu, and W. H. Chen, "Task-stage Knowledge Support: Coupling User Information Needs with Stage Identification," *IEEE International Conference on Information Reuse and Integration (IRI, 2005)*, pp. 19-24, 2005.

[59] H. Yun, D. Ha, B. Hwang, and K. Ho Ryu, "Mining Association Rules on Significant Rare Data Using Relative Support," *The Journal of Systems & Software,* vol. 67, no. 3, pp. 181-191, 2003.

[60] Y. Zhao, G. Karypis, and U. Fayyad, "Hierarchical Clustering Algorithms for Document Datasets," *Data Mining and Knowledge Discovery,* vol. 10, no. 2, pp. 141-168, 2005.

[61] H. Zhuge, "A Knowledge Flow Model for Peer-to-peer Team Knowledge Sharing and Management," *Expert Systems with Applications,* vol. 23, no. 1, pp. 23-30, 2002.

[62] H. Zhuge, "Knowledge Flow Network Planning and Simulation," *Decision Support Systems,* vol. 42, no. 2, pp. 571-592, 2006.

[63] H. Zhuge, "Discovery of Knowledge Flow in Science," *Communications of the ACM,* vol. 49, no. 5, pp. 101-107, 2006.

[64] H. Zhuge, and W. Guo, "Virtual Knowledge Service Market--For Effective Knowledge Flow within Knowledge Grid," *Journal of Systems and Software,* vol. 80, no. 11, pp. 1833-1842, 2007.