International Journal of Information Technologies and Systems Approach
Volume 16 • Issue 2

Discovery of User Groups Densely Connecting Virtual and Physical Worlds in Event-Based Social Networks

Tianming Lan, School of Information Management, Jiangxi University of Finance and Economics, China & Fujian Key Laboratory of Big Data Application and Intellectualization for Tea Industry, College of Mathematics and Computer, Wuyi University, China*

https://orcid.org/0000-0002-6251-3499

Lei Guo, Fujian Key Laboratory of Big Data Application and Intellectualization for Tea Industry, College of Mathematics and Computer, Wuyi University, China

ABSTRACT

An essential task of the event-based social network (EBSN) platform is to recommend events to user groups. Usually, users are more willing to participate in events and interest groups with their friends, forming a particularly closely connected user group. However, such groups do not explicitly exist in EBSN. Therefore, studying how to discover groups composed of users who frequently participate in events and interest groups in EBSN has essential theoretical and practical significance. This article proposes the problem of discovering maximum k fully connected user groups. To address this issue, this article designs and implements three algorithms: a search algorithm based on Max-miner (MMBS), a search algorithm based on two vectors (TVBS) and enumeration tree, and a divide-and-conquer parallel search algorithm (DCPS). The authors conducted experiments on real datasets. The comparison of experimental results of these three algorithms on datasets from different cities shows that the DCPS algorithm and TVBS algorithm significantly accelerate their computational time when the minimum support rate is low. The time consumption of DCPS algorithm can reach one tenth or even lower than that of MMBS algorithm.

KEYWORDS

divide-and-conquer, EBSN, enumeration tree, group discovery, Max-miner

1. DISCOVERY OF USER GROUPS DENSELY CONNECTING VIRTUAL AND PHYSICAL WORLDS IN EVENT-BASED SOCIAL NETWORKS

Event-Based Social Network (EBSN) is a new type of social network that can provide online social services and offline social activities (Liu et al., 2012). Online social services typically include users joining online interest groups, sharing comments and photos, etc. EBSN provides many offline social

DOI: 10.4018/IJITSA.327004 *Corresponding Author

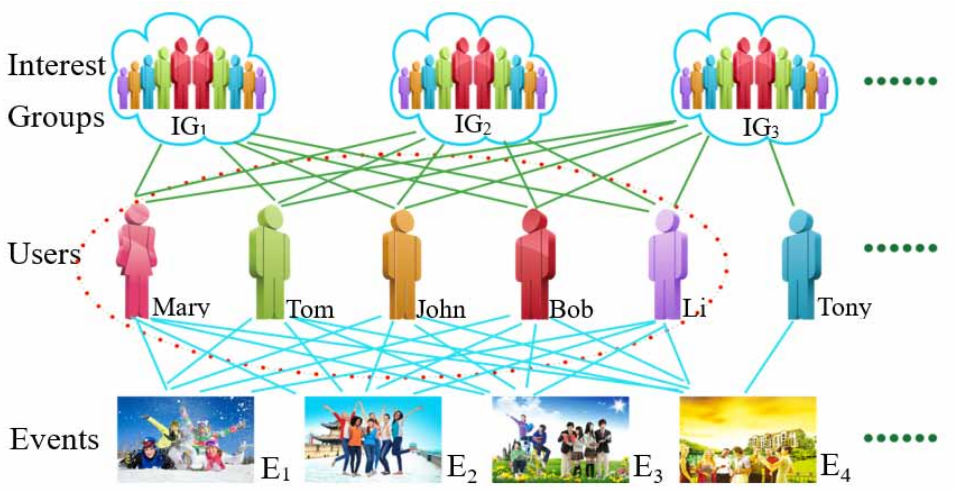
activities, such as reading, dancing, swimming, etc. The services provided by EBSN have attracted a large number of users and are increasingly favored by them. Typical EBSN platforms include Meetup, Plancast, Doubanevent, and more (Lan et al., 2022).

In EBSN, users usually participate in events and interest groups with their familiar friends. These people influence each other when deciding to participate in events or interest groups, forming a particular group. Usually, members of such groups significantly affect each other and are particularly closely connected, and they will participate in multiple interest groups and events together. Treating such groups as a whole for event recommendation would have a better effect than only considering individual user event recommendations. Kim et al. (2020) proposed that group users maintain close connections on social networking platforms and geographical locations, which can facilitate event recommendation, friend recommendation, and geographic data analysis. In addition to considering group members' preference similarity, online and offline interactions between members can also be considered, utilizing various information to discover groups (Liao et al., 2021). Wang et al. (2017) believe that appropriate aggregation of online and offline user groups can better understand user behavior and its underlying organizational principles comprehensively. Trinh et al. (2020) defined group activity and user loyalty and proposed a method to measure group activity. The article believes that the close relationship between users and groups determines group activity, and user loyalty is a critical factor in maintaining group activity. The close online and offline connections between users and groups promote the development of the group. The above research indicates that identifying user groups with close online and offline relationships is meaningful and deserves our in-depth study.

The EBSN recommendation system effectively improves the efficiency of recommendation and prediction by recommending events and predicting group event participation for closely connected groups online and offline. However, EBSN does not directly provide such groups, and finding suitable and effective group discovery methods to quickly and accurately discover such groups is the purpose of this study.

In Figure 1, Mary, Tom, John, Bob, and Li jointly participated in multiple interest groups and events, forming a particularly closely connected group of users. However, they also participated in the event and interest groups together with Tony, and among these six user groups, Tony and other users were not closely connected, which is not the group we want to discover.

Figure 1. A closely connected group in EBSN



There are two main methods for obtaining groups in EBSN. One is to cluster users who participate in the same event or interest group or who have high similarity in EBSN. Another type is to divide communities based on user similarity.

Liao et al. (2019) removed inactive users from their interest groups and treated the remaining users as groups. Liao et al. (2006) formed groups of users who frequently participate in events together. Yuan et al. (2014) and Vinh et al. (2019) formed a group of users who participate in a certain event. Du et al. (2019) defined that users with preference similarity higher than a specific threshold form user groups. Jeong, et al. (2019), Ji et al. (2018), and Liao et al. (2020) identified users from the same interest group as user groups. Purushotham et al. (2016) proposed using offline social groups participating in events at specific locations as user groups. Trinh et al. (2021) searched for users and their friends and recommended events to them. The group obtained by this method may not have close connections among its members and may not necessarily participate in the next event together.

Yin et al. (2016) generated communities based on network proximity, spatiotemporal co-occurrence, and semantic similarity of user preferences. Communities can also be viewed as user groups. Khrouf and Troncy (2018) considered users, events, and related attributes, established a ternary group graph model, and clustered events based on topics on its established ternary group graph model. Finally, all participants in the clustering event were formed into user groups. Li et al. (2020) proposed a community detection algorithm based on social impact. Groups obtained based on user similarity clustering usually have multiple group members and high partitioning complexity, which is not conducive to improving recommendation accuracy. This method also cannot identify closely related groups.

The above research provides an important reference and basis for discovering closely connected groups in EBSN. However, they do not cluster users from the perspective of close online and offline connections. The difficulty and challenge of this research is to adopt a strategy to search the dataset to obtain closely connected user groups.

This paper proposes the concept of the maximum k fully connected user group and the problem of discovering the maximum k fully connected user group. The maximum k fully connected user group is a closely related user group in EBSN mentioned earlier in this article. Discovering the maximum k fully connected user groups is an entirely new problem. To solve this problem, we naturally propose three search algorithms based on the idea of width first search: the Max-miner-based search algorithm (MMBS), the two-vectors-based search algorithm (TVBS), and the divide and conquer parallel search algorithm (DCPS). In the MMBS algorithm, we use Max-miner to mine user participation events and interest groups separately in EBSN that meet the conditions. We intersect the online maximum frequent user set and the offline maximum frequent user set in pairs and then filter out non-maximum k fully connected user groups to obtain the results. In the TVBS algorithm, we first use a vertical data representation format to represent users in a two-vector format based on their participation events and interest group records. The two-vector format of vertical data representation can facilitate the calculation of thresholds. Secondly, when conducting searches, we adopted pruning methods, which can save search time and avoid invalid searches. The TVBS algorithm can quickly search for results, while the DCPS algorithm decomposes the search task into multiple independent subtasks. Each subtask is searched using the TVBS algorithm, and all subtasks are completed in parallel to obtain results. The algorithm uses the ideas of divide and conquer and parallelism.

The main contributions of this paper include the following:

(1) We defined the concept of maximum k fully connected user groups and proposed the problem of discovering maximum k fully connected user groups.
(2) We propose a search algorithm based on Max-miner, a search algorithm based on two vectors, and a divide-and-conquer parallel search algorithm. The bidirectional volume we constructed reduces the computational complexity of generating the maximum k fully connected user group, significantly improving search efficiency.

(3) Tests on real datasets have confirmed that our proposed MMBS, TVBS, and DCPS algorithms can effectively discover the maximum k fully connected user groups.

The rest of this article is arranged as follows: Section 2 introduces the relevant work. In Section 3. the problem and related definitions are described. In Section 4, the proposed algorithm is introduced. A detailed experimental evaluation is conducted in Section 5 and conclusions are provided in Section 6.

2. RELATED WORK

This section will briefly introduce the relevant work from three aspects.

2.1. Maximum Frequent Itemset Mining Method

The maximum frequent itemset refers to the frequent itemset that meets the condition of no superset in the frequent itemset. Most existing maximum frequent itemset mining algorithms use breadth-first search or depth-first search. The Apriori and FP-growth algorithms are classic for frequent item set mining and representative algorithms for breadth-first or depth-first search. Many studies mine the maximum frequent itemset by extending the Apriori or FP-growth algorithms.

The Apriori algorithm proposed by Agrawal & Srikant (1994) utilizes an iterative method of layer-by-layer search to identify frequent item sets in the database. Lin et al. (2002) proposed a new algorithm for mining the maximum frequent itemset, Max-miner, which combines bottom-up and top-down searches and can prune candidate sets in the search. Liu et al. (2018) proposed an improved algorithm based on Apriori, which sorts the maximum frequent itemsets discovered based on confidence intervals.

Han et al. (2000) proposed a novel frequent pattern tree (FP-tree) structure for storing compressed, extended prefix tree structures and developed an effective FP-tree-based mining method FP-growth for mining complete sets of frequent patterns. Grahne et al. (2003) extended the FP-growth method called FP-Max. Ji et al. (Ji et al., 2005) adopted an FP-tree storage structure and a top-down search strategy, effectively improving the mining efficiency of the maximum frequent itemset.

The Apriori and FP-growth algorithms have problems mining the maximum frequent itemsets, such as low efficiency, high memory consumption, difficulty adapting to the processing of dense datasets, and affecting the efficiency of big data value mining.

Gouda et al. (2001) proposed an algorithm for mining the maximum frequent itemsets based on a backtracking depth-first search. Burdick et al. (2001) proposed a new algorithm for mining the maximum frequent itemset, whose search strategy can significantly improve mining performance. Zhang et al. (2021) proposed a multi-objective optimization algorithm to mine the maximum frequent itemsets in high-dimensional data. Hamid et al. (2021) introduced a new parallel algorithm to generate the maximum frequent itemset. Zhang et al. (2021) proposed Right Extension (RHSE) algorithm, which can accurately locate all MFIs. Khafaji et al. (2021) proposed an algorithm for extracting the maximum frequent itemset from Arabic text datasets. The algorithm starts at the selected initial boundary in the search space and then moves up and down to generate the required maximum frequent itemset. Fatemi et al. (2021) introduced an approximation algorithm that transforms the problem into a clustering problem. Extract all the maximum frequent itemsets from the clustered itemsets using MAFIA. Chen et al. (2020) proposed a two-step algorithm APFI-MAX, which approximately mines probabilistic maximum frequency itemsets.

The maximum frequent itemset mining method cannot consider the interaction between user participation events and interest groups. Therefore, these methods cannot be directly used to solve the problem of discovering maximum k fully connected user groups.

2.2. Maximum Frequent Subgraph Mining Method

Graph mining refers to mining helpful information from data using graph structures. Frequent subgraph mining aims to find all subgraphs in a large image that appear more than or equal to a given frequency threshold (Nguyen et al., 2022). Maximum frequent subgraphs are a fundamental pattern discovered in graph sets and mining them has become a hot topic in graph mining. There are currently some mature algorithms for mining maximum frequent subgraphs.

Kuramochi and Karypis (2001) proposed an algorithm to mine all frequently connected subgraphs by following a pattern growth step-by-step method and proposed candidate generation and frequency counting optimization methods that introduce practical specification markers. Yan and Han (2002) proposed the gSpan algorithm that follows the depth-first pattern growth method. Huan et al. (2004) proposed a new algorithm that only mines the maximum frequent subgraphs, which may exponentially reduce the size of the output set. Wu et al. (2018) proposed an algorithm for mining uncertain maximum frequent subgraphs based on adjacency matrix and weight. Salem et al. (2021) proposed a reverse search algorithm for mining frequent and maximum frequent subgraphs in a given graph set. Li (2020) constructed a three-dimensional hierarchical search tree based on the graph to be searched and designed a hierarchical search algorithm to obtain subgraphs.

The maximum frequent subgraph mining algorithm is used to discover the maximum frequent subgraphs that appear in the graph set, but EBSN does not have a graph set, so the maximum frequent subgraph mining algorithm cannot be directly used to discover the maximum k fully connected user group.

2.3. Subgroup Discovery Method

Currently, many subgroup discovery algorithms use different search strategies for subgroup discovery (Padillo et al., 2017; Lemmerich et al., 2016; Belfodil et al., 2019; Deng et al., 2020; De et al., 2018; Lucas et al., 2018). Herrera et al. (2011) divided subgroup discovery algorithms into three types: classification-based extension, association rule mining extension, and evolutionary algorithm. Ventura et al. (2018) added an appropriately designed algorithm that can run in big data environments based on the previous (Herrera et al., 2011). Atzmueller (2015) divided subgroup discovery algorithms into exhaustive search and heuristic algorithms. Helal (2016) classified subgroup discovery algorithms into exhaustive search, heuristic, and evolutionary algorithms. Padillo, et al. (2017) proposed two exhaustive search algorithms: AprioriKSD-OE and PFP-SD-OE. PFP-SD-OE uses efficient data structures to mine subgroups on big data. Lemmerich, et al. (2016) proposed a new method for fast exhaustive subgroup discovery with numerical objectives.

Belodil et al. (2019) proposed an efficient and parameterless branch and a bound algorithm based on the greedy strategy called FSSD, which uses multiple optimization strategies to obtain subgroups effectively in a short period. Deng et al. (2020) proposed a dual subgroup pattern mining algorithm based on classical beam search, which uses the interest of information theory to find node subgroup pairs with attractive or high edge density. Meeng et al. (2021) compared the results of three search strategies: traditional beam search, complete search, and subgroup discovery based on coverage subgroup selection.

De et al. (2018) improved the performance of evolutionary algorithms in high-dimensional subgroup discovery tasks by biasing the initial population towards smaller individuals. Lucas et al. (2018) proposed an evolutionary method for mining diverse and more informative subgroups focused on high-dimensional datasets.

Heuristic algorithms have a faster search speed than exhaustive search strategies, but they only explore a portion of the search space and cannot guarantee the best results. Evolutionary algorithms mimic the principles of natural evolution and follow the process of natural evolution to form searches, making them one of the most widely used search algorithms. However, the subgroup discovery method does not perform group discovery based on the closeness of users' connections, so the maximum k fully connected user group discovery cannot directly use the subgroup discovery method.

3. RELATED DEFINITIONS AND PROBLEM DESCRIPTIONS

EBSN consists of online and offline social networks, which describe the relationship between user participation in interest groups and user participation in events. Let $U=\{u_1,u_2,\ldots u_i\}$ be the user set, $E=\{e_1,e_2,\ldots e_j\}$ be the event set, and $IG=\{ig_1,ig_2,\ldots ig_k\}$ be the interest group set. The relevant definitions in this article are as follows.

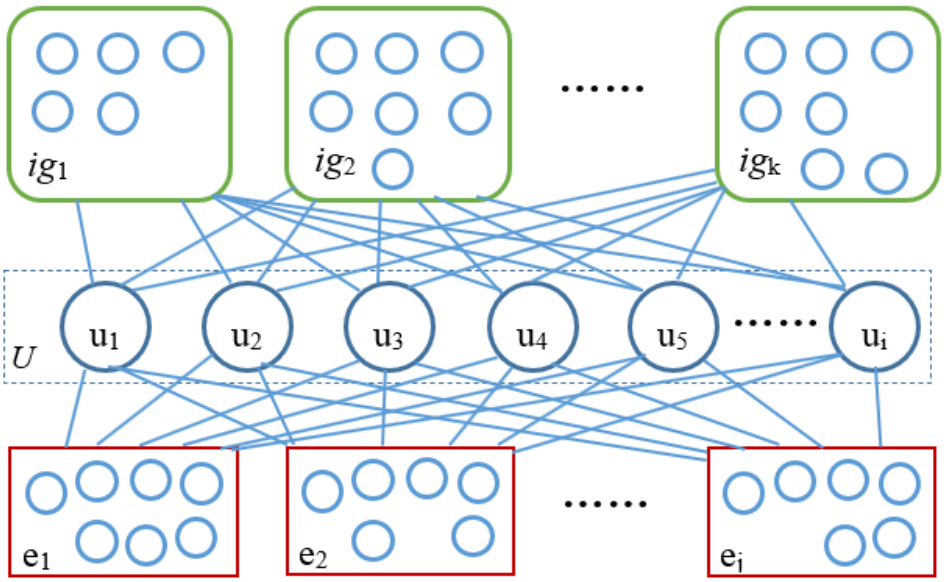
Definition 1 (EBSN Heterogeneous Social Network): EBSN heterogeneous social network is represented as $N^{het}=(N^{on}, N^{off})$, where N^{on} is the online social network, represented as $N^{on}=(U, IG, Edge^{on})$, where U and IG are network nodes, referring to users and interest groups, and $Edge^{on}$ is the edge between users and interest groups, indicating that users participate in interest groups. There is no edge between users or interest groups. N^{off} is an offline social network, represented by: $N^{off}=(U, E, Edge^{off})$, where U and E are network nodes referring to users and events. $Edge^{off}$ is the edge between users and events, meaning that users participate in events. There is no edge between users or between events. Online and offline social networks share a common set of users, so EBSN heterogeneous networks can be represented as $N^{het}=(U, IG, E, Edge^{on}, Edge^{off})$.

Definition 2 (online k fully connected user group): Given the EBSN online social network $N^{on}=(U, IG, Edge^{on})$, as shown in Figure 2, assuming the existence of a subset of users (referred to as U_s), where all users collectively participate in at least k identical interest groups, that is, each user in U_s and each interest group in k interest groups have an edge. Therefore, the users of U_s and k interest groups form a fully connected structure, and we call U_s the online k fully connected user group.

Definition 3 (offline k fully connected user group): Given the offline social network $N^{off}=(U, E, Edge^{off})$ of EBSN, as shown in Figure 2, assuming the existence of a subset of users (referred to as U_s), all users participate in at least k events together, that is, each user and k events have an edge. Therefore, the users of U_s and k events form a fully connected structure, and we refer to U_s as the offline k fully connected user group.

Definition 4 (k fully connected user group): Given the EBSN heterogeneous social network $N^{het}=(U, IG, E, Edge^{on}, Edge^{off})$, as shown in Figure 2, assuming the existence of U_s, all users in this set jointly participate in at least k_1 events and k_2 interest groups, with k being the minimum support

Figure 2. The fully connected structure formed by users, interest groups, and events



and $k= k_1+k_2$, a fully connected structure is formed between U_s ' users, k_1 events, and k_2 interest groups. We call U_s a k fully connected user group.

Definition 5 (maximum k fully connected user groups): Given the EBSN heterogeneous social network $N^{het}=(U, IG, E, Edge^{on}, Edge^{off})$, assuming the existence of a subset of users U_s, being a k fully connected user group, k is the minimum support and $k=k_1+k_2$, if a user is added to the user subset U_s, all users in the set no longer meet the condition of jointly participating in k_1 events or k_2 interest groups, that is, the network formed between the user subset and k_1 events and k_2 interest groups is no longer a fully connected structure, U_s is referred to as a maximum k fully connected user group.

Property 1: The proper subset of any maximum k fully connected user group is not the maximum k fully connected user group.

Property 2: Any subset of the maximum k-fully connected user group is a k-fully connected user group.

Definition 6 (maximum k fully connected user group found issues): Given the EBSN heterogeneous social network $N^{het} =(U, IG, E, Edge^{on}, Edge^{off})$ and the minimum support $k=k_1+k_2$, Discovering all the maximum k fully connected user groups from EBSN heterogeneous social networks is the problem of discovering the maximum k fully connected user groups.

Definition 7 (User Interaction Subgroup): Divide users in the user set who have reached a support level for the number of interest groups and events with a certain user u into a user subset, which is called the user interaction subgroup of user u.

4. MAXIMUM K FULLY CONNECTED USER GROUP DISCOVERY ALGORITHM

Mining the maximum k fully connected user group (G_{mk}) in EBSN requires considering both online and offline social relationships, making it challenging to establish unique data structures such as FP-Tree to mine G_{mk}. The existing group acquisition methods cannot effectively obtain G_{mk}. Therefore, we consider using the width first search algorithm and propose a search algorithm (MMBS) based on the maximum frequent itemset mining algorithm Max-miner to obtain G_{mk}. Then, based on the characteristics of EBSN heterogeneous social networks, a two-vectors-based search algorithm (TVBS) is proposed. Then, a divide and rule parallel search algorithm (DCPS) is proposed based on the divide and conquer and parallel thought.

4.1. Max-Miner-Based Search Algorithm

The Max-miner algorithm is a classic algorithm proposed by Bayardo et al. (Bayardo, 1998) for mining the maximum frequent itemset. It is the most representative width-first search algorithm and has proven superior to the classic width-first search algorithm Apriori. The maximum frequent itemset mining algorithm is also a group discovery algorithm. In the online and offline networks of EBSN, users participate in multiple interest groups and events. The user participation in interest groups or events is considered a database transaction set. Each transaction is a record of the user participating in an interest group or an event, and the items in the transaction are the users. The event and interest group can be represented as $e=\{ u_1,u_2,\ldots u_1 \}$, $ig=\{ u_1,u_2,\ldots u_m \}$, mining the maximum frequent itemset is mining the maximum frequent user set.

The Max-miner algorithm can quickly mine the maximum frequent user set in both the online and offline networks of EBSN. However, the maximum frequent user set is not the maximum k fully connected user group. It is necessary to intersect the elements in the obtained online and offline maximum frequent user sets one by one to obtain candidate k-fully connected user groups and then filter the candidate k-fully connected user groups to obtain the maximum k-fully connected user group.

The algorithm is as follows:

Algorithm 1. MMBS(E, IG,k_1,k_2).
Input:E, IG, k_1, k_2
Output:S_{gmk}

```
1:begin
2:S_off =max_miner(E, k_1)
3:S_on =max_miner(IG, k_2)
4:For item_off in S_off
    For item_on in S_on
          S_w= item_off Ç item_on
5:S_gmk = remove from S_w any itemset with a proper superset in S_w or
|itemset|<3
6:End
```

In Algorithm 1, Steps 2 and 3 obtain the offline maximum frequent user set and the online maximum frequent user set, Step 4 intersects the offline maximum frequent user set and the online maximum frequent user set, and Step 5 filters out non-G_{mk} items in S_w. This paper stipulates that G_{mk}'s group members must be at least three people.

4.2. Two-Vectors-Based Search Algorithm

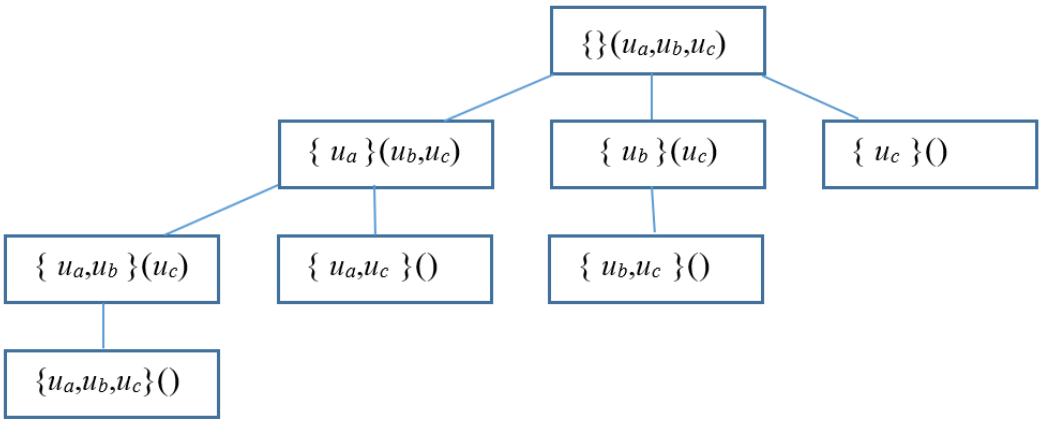
The width first search algorithm is a commonly used method for group discovery. Based on the idea of width-first search, we propose a width-first search algorithm for discovering G_{mk}, called the TVBS algorithm. The TVBS algorithm searches for all subsets of users in the user set that meet the support criteria, identifies the users who jointly participate in the number of interest groups and events in the user subset that meet the support criteria, and then finds all G_{mk}s through loops and pruning. It combines user participation information from EBSN during the search process and can quickly and accurately search for all G_{mk}s in the user set.

The enumeration tree is a method proposed by Bayardo et al. (Bayardo, 1998). It arranges all items appearing in a database in lexicographic order and logically organizes them into an enumeration tree. Layer 0 is the root node. The kth layer contains all the k-item sets. Each node in the tree consists of two parts, namely the head of the node and the tail of the node. The head is the set of items represented by the node itself. The tail is a collection of items that nodes can extend. As shown in Figure 3, assuming that the set of all items is $\{u_a, u_b, u_c\}$, in the figure, "{}" represents the head, and "()" represents the tail. The TVBS algorithm searches through the enumeration tree of the dataset.

4.2.1. Two Vector Data Formats for Users and Groups

To solve the G_{mk} problem, it is necessary to identify a group of users who have reached the threshold of participating in events and interest groups together and cluster these users. During the search

Figure 3. Enumeration tree over three items



process, it is also necessary to identify the events and interest groups that these users jointly participate in. The dot product of vectors composed of multiple 0s and 1s effectively calculates the number of intersections. In contrast, the Hadamard product of vectors effectively finds the events and interest groups that users participate in together.

Let vector $\boldsymbol{V_1}$ be $[v_{11},v_{12},\ldots,v_{1n}]$, vector $\boldsymbol{V_2}$ be$[v_{21},v_{22},\ldots,v_{2n}]$, and the dot product formula of the two vectors is as follows:

$$\boldsymbol{V}_1 \bullet \boldsymbol{V}_2 = v_{11} \times v_{21} + v_{12} \times v_{22} + \ldots + v_{1n} \times v_{2n} \tag{1}$$

The Hadamard product formula for two vectors is as follows:

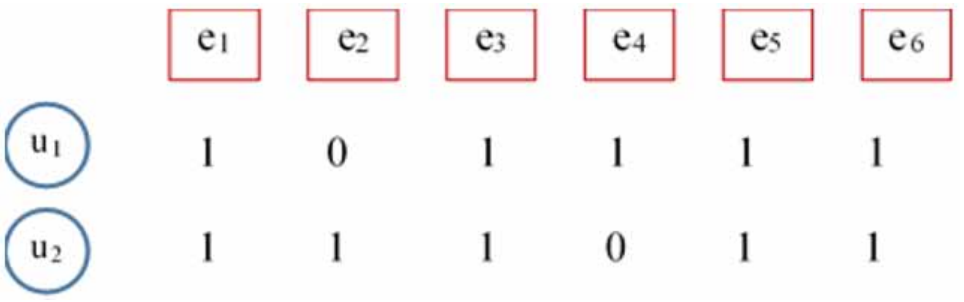
$$\boldsymbol{V}_1 \bigcirc \boldsymbol{V}_2 = [v_{11} \times v_{21}, v_{12} \times v_{22}, \ldots, v_{1n} \times v_{2n}] \tag{2}$$

As shown in Figure 4, the number of events participated by users u_1 and u_2 is taken as the dimension of the user participation event vector. The dimensions of the events and vectors correspond one-to-one, and the corresponding dimensions of the vectors are set to 1 and 0 based on whether the user participates in the event. To obtain the number of shared participation events for users u_1 and u_2, simply dot product the corresponding participation event vectors of the two users. To obtain the common participation event vectors of users u_1 and u_2, do the Hadamard product of the corresponding participation event vectors of the two users.

The TVBS proposed in this paper requires the user's participation event vectors v_u^e and the user's participation interest group vectors v_u^{ig}, which can be obtained according to the above method. The acquisition algorithm is as follows:

Algorithm 2. GetUserParticipateVector(*U, E, IG*).
Input:*U, E, IG*
Output: v_u^e , v_u^{ig}
1: Begin
2: *j= len(E)*
3: Number the event in *E* from 1 to *j*
4: Scan *E* to obtain the participation events of each user
5: Set the participation event vector *ve u* for each user based on their participation
6: k =len(IG)
7: Number the interesting group in *IG* from 1 to *k*
8: Scan *IG* to obtain the participation interesting groups of each user

Figure 4. Construct the user's participation event vector



9: Set the participation interesting group vector *vig u* for each user based on their participation
10: Return v_u^e, v_u^{ig}
11: End

4.2.2. Search Process of TVBS Algorithm

The search process of the TVBS algorithm is as follows:

First, identify all users who meet the support threshold conditions and treat them as a set of U_s.
Second, an enumeration tree is constructed based on the user set U_s, with nodes consisting of two parts: the head $h(n)$ and the tail $t(n)$. Initialize the root node of the enumeration tree so that the head $h(n)$ of the root node is { } and the tail $t(n)$ is U_s.
Third, starting from the root node, generate the child nodes of the current node. For all the items in $t(n)$, add one of the items in sequence and the original items in $h(n)$ to form the head $h(x)$ of the child node. The items in $t(n)$ that come after that item are the child node's tail $t(x)$. Nodes that do not meet the support threshold are pruned. When $t(n)$ is empty, the node's head $h(n)$ forms the G_{mk}.
Fourth, when $t(n)$ is not empty, repeat the third step for the obtained child nodes.

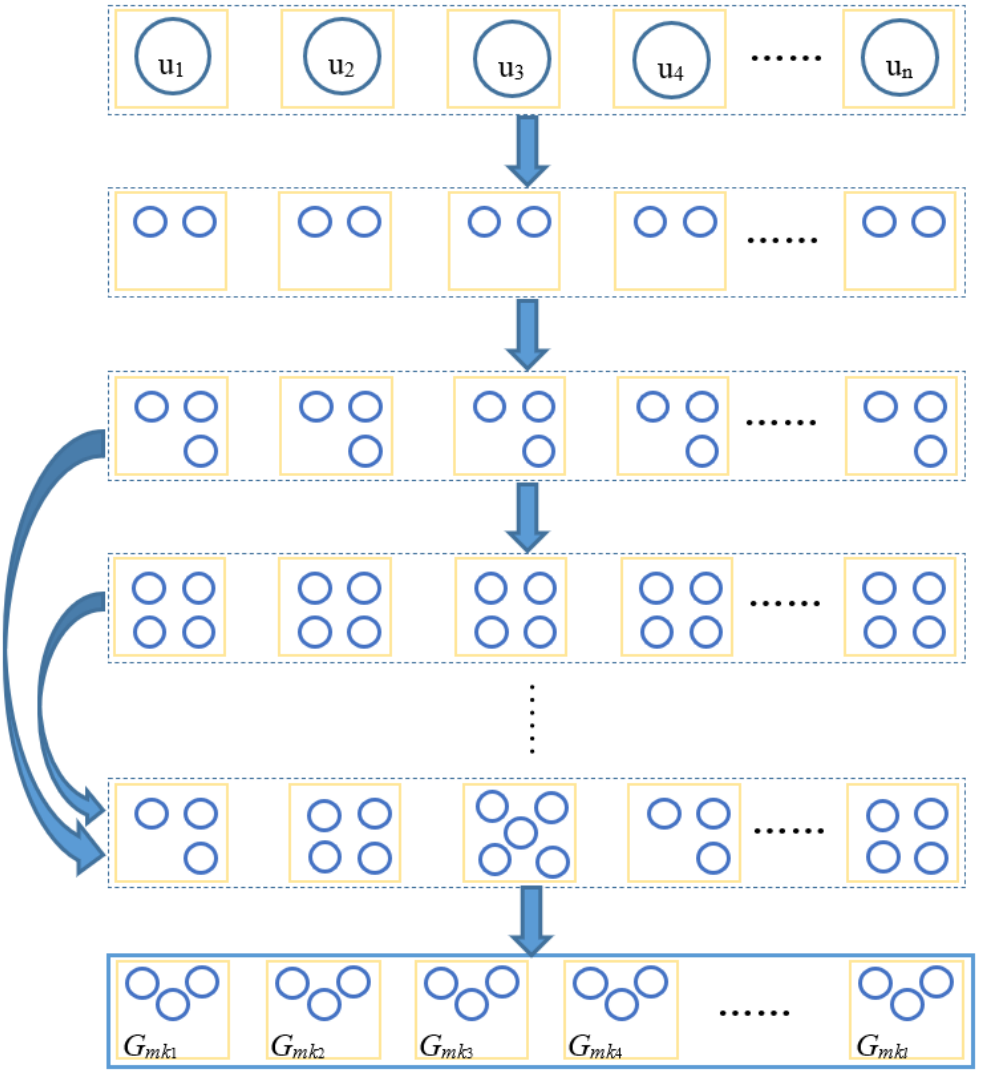
Pruning during the search process can significantly reduce search volume and improve search efficiency. We set a threshold k to filter users and reduce the number of search nodes. If the number of co-participating events and co-participating interest groups of a certain group and other groups do not meet the threshold condition, filtering is performed. We also use property 1 in definition 5 for pruning. Figure 5 shows the search process for the maximum k fully connected user group.
The TVBS algorithm is described as follows:

Algorithm 3. TVBS (U, v_u^e, v_u^{ig}, k_1, k_2).
Input: U, v_u^e, v_u^{ig}, k_1, k_2
Output: S_{gmk}
1: Begin
2: Sc = { }
3: S_{gmk} = {Gen_Initial_Candidate_Groups (U, Sc, v_u^e, v_u^{ig}, k_1, k_2)}
4: While Sc is non-empty do
Set of Candidate Groups Sc_{new} = { }
for each $g \in Sc$ do
$S_{gmk} = S_{gmk} \cup$ { Gen_Child_Nodes (g,Sc_{new})}
$Sc = Sc_{new}$
remove from S_{gmk} any itemset with a propersuperset in S_{gmk} or | itemset |<3
remove from Sc any group g such that h(g) $\cup$t(g) has a superset in S_{gmk}
5: Return S_{gmk}

In the algorithm, Step 3 identifies users who meet the support threshold, Step 4 generates sub-nodes for nodes in the enumeration tree, generates G_{mk} and filters. The algorithm description of the functions Gen_Initial_Candidate_Groups and Gen_Child_Nodes () used in the algorithm can be found in Algorithm 4 and Algorithm 5:

Algorithm 4. Gen_Initial_Candidate_Groups(U, S_c, v_u^e, v_u^{ig}, k_1, k_2).
Input: U, S_c, v_u^e, v_u^{ig}, k_1, k_2
Output: v_u^e, v_u^{ig} the itemset in S_{gmk1} containing the greatest item
1: Begin

Figure 5. Search process for maximum k fully connected user groups



2: Scan U if $v_{u_i}^{ig} \bullet v_1^{ig} >= k_2$ & $v_{u_i}^{e} \bullet v_1^{e} >= k_1$: $S_{gmk1} \neg u_i$
3: Impose an ordering on the items in S_{gmk1}
4: For each item u_i in S_{gmk1} other than the greatest item do
let g be a new candidate with $h(g)$ = { u_i }and $t(g)$ = { u_j | u_j follows u_i in the ordering}

$$v_{h(g)}^{ig} = v_{u_i}^{ig}$$

$$v_{h(g)}^{e} = v_{u_i}^{e}$$

$S_c = S_c \cup \{g\}$
5: Return the itemset in S_{gmk1} containing the greatest item
6: End

Among the above algorithms, Step 2 prunes by setting support threshold, *vig 1*and *ve 1*represent the value of 1 for each dimension of the vector. Step 4 establishes nodes of the enumeration tree and generates two vectors of the nodes.

Algorithm 5. Gen_Child_Nodes(g,S_c).

Input: g, S_c

Output: v_u^e, v_u^{ig} h(g)∪{u_m} or h(g)

1: Begin

2: Remove any item u_i from $t(g)$ if $v_{u_i}^{ig} \bullet v_{h(g)}^{ig} < k_2 \,||\, v_{u_i}^{e} \bullet v_{h(g)}^{e} < k_1$:

3: Reorder the items in $t(g)$

4: For each $u_i \in t(g)$ other than the greatest do
let g' be a new candidate with $h(g') = h(g) \cup \{u_i\}$and $t(g') = \{u_j | u_j \in t(g)$ and j follows i in $t(g)\}$

$v_{h(g')}^{ig} = v_{u_i}^{ig}$ ¡ $v_{h(g)}^{ig}$

$v_{h(g')}^{e} = v_{u_i}^{e}$ ¡ $v_{h(g)}^{e}$

$S_c = S_c \cup \{g'\}$

5: Return $h(g) \cup \{u_m\}$ where u_m is the greatest item in $t(g)$ or return $h(g)$ if $t(g)$ is empty.

6: End

Among the above algorithms, Step 2 implements pruning and Step 4 generates sub-nodes and sub-node vectors.

The search process of the TVBS algorithm draws inspiration from the idea of the Max-miner algorithm, which can ensure the correctness and effectiveness of the algorithm.

The TVBS algorithm can quickly search for all the maximum k fully connected user groups by sequentially searching for all users in the user set. The pruning process the algorithm uses limits the search space but does not affect the search results. However, as the number of users in the user set increases, the temporal and spatial performance of the algorithm deteriorates, and the problem of discovering the maximum k fully connected user groups becomes complex. Therefore, we propose a divide-and-conquer parallel search algorithm based on the TVBS algorithm.

4.3. Divide and Conquer Parallel Search Algorithm

The maximum k fully connected user group forms a fully connected structure between users, interest groups, and events. We found that the maximum k fully connected user group containing user u_i must be in the user interaction subgroup of user u_i. Therefore, by dividing the user set and identifying the interaction subgroups of each user in the user set, the TVBS algorithm can search for the maximum k fully connected user group in the user interaction subgroups.

The user interaction subgroups are independent so they can be searched in parallel within the user interaction subgroups. Because the size of the user interaction subgroup is much smaller than the user set, the search complexity is significantly reduced. The divide and conquer parallel search algorithm combines the characteristics of fully connected structures and the heterogeneous network characteristics of EBSN and adopts the idea of divide and conquer parallel for search, thus having more efficient search performance.

The process of the divide and conquer parallel search algorithm is as follows: first, the user set is segmented to obtain each user's interaction subgroup, and then the TVBS algorithm is called in parallel to mine to get multiple sets of maximum k fully connected user groups on each user interaction subgroup. Finally, the union of multiple sets of maximum k fully connected user groups is filtered to obtain the maximum k fully connected user group set.

The process of segmenting the user set to obtain user interaction subgroups is as follows:

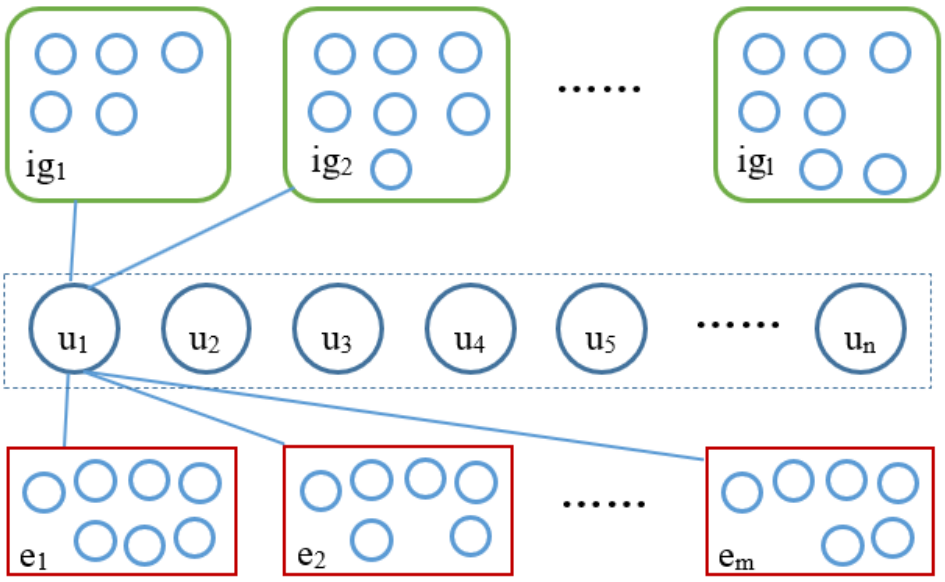
Scan sets *IG* and *E* to obtain the participating interest group set U^{ig} and participating event set U^{e} for each user. The elements in U^{ig} are *uig i*= $\{ig_1,ig_2,\ldots ig_m\}$, and the elements in U^{e} are *ue i* = $\{e_1,e_2,\ldots e_n\}$. Filter out users in the two sets whose number of participating events and interest groups has not reached the threshold. Intersect the two sets and each user in the new set has two records *uig* and*ue i*. Obtain the interest groups that users participate in *uig i* and the events that users participate in *ue i*. Scan *IG* to obtain the user set UIG_{ui} that participates in the interest groups with user u_i. Scan *E* to obtain the user set UE_{ui} that participates in the events with user u_i. Intersect and filter these two sets to obtain the interaction subgroup UIS_{ui} of user u_i.

Figure 6 shows the acquisition of interaction subgroups for user u_1. First, obtain the set of participating interest groups and participating events for user u_1, and then obtain the participating users for each interest group in the set of participating interest groups for user u_1 and merge these users into one set. Next, use the same method to obtain the set of participating users in the event set of user u_1. Intersect these two sets to obtain a set of users who have interacted with user u_1 online and offline. Select all users in this set who have participated in k_2 interest groups and k_1 events with user u_1. These users form the interaction subgroup of user u_1.

The algorithm for obtaining user interaction subgroups is described as follows:

Algorithm 6. Get_User_Interaction_Subgroup(*IG*, *E*, k_1, k_2).
Input: *IG*, *E*, k_1, k_2
Output:User interaction subgroup set S_u^i
1: Begin
2: Scan *IG* obtain User participation in interest group records U^{ig}
3: Scan *E* obtain User participation event recording U^{e}
4: $US_{on} \leftarrow$ scan R_u^{ig} if $|u_i^{ig}| >= k_2$, $US_{off} \leftarrow$ scan R_u^e if $|u_i^e| >= k_1$
5: $U_s = US_{on} \cap US_{off}$
6: $S_u^i = \{\}$
7: For u_i in U_s

Figure 6. Obtain user interaction subgroups



8: For ig in u_i^{ig}
$UIG_{ui} = UIG_{ui} \cup ig$
9: For e in $ue\ i$
$UE_{ui} = UE_{ui} \cup e$
10: $UIA_{ui} \leftarrow UIG_{ui} \cap UE_{ui}$
11: For u_j in UIA_{ui}
if $v_{u_i}^{ig} \bullet v_{u_j}^{ig} >= k_2 \& v_{u_i}^{e} \bullet v_{u_j}^{e} >= k_1$: $UIS_{ui} \leftarrow u_j$
12: $UIS_{ui} \leftarrow u_i$, $S_u^i \leftarrow S_u^i \cup UIS_{ui}$
13: Return S_u^i
14: End

In the algorithm, Steps 4 and 5 obtain the set of users who interact with the user u_i, Step 10 obtains the set of users who interact with the user u_i, and Step 11 filters out users who do not meet the conditions to find user interaction subgroups.

After obtaining the user interaction subgroups of all users in the user set, a recursive search algorithm is executed concurrently with the user and their interaction subgroups as parameters to obtain the maximum k fully connected user group. The DCPS algorithm is described as follows:

Algorithm 7. DCPS(S_u^i, v_u^{ig}, v_u^e, k_1, k_2).
Input: S_u^i, v_u^{ig}, v_u^e, k_1, k_2
Output: S_{gmk}
1:Begin
2: $p=pool\ (25)$
3: For $item \in S_u^i$
4: $res=p.apply\_async(\text{TVBS}, args=(item, v_u^e, v_u^{ig}, k_1, k_2))$
5: $res\_l.append(res)$
6: $p.join()$
7: Remove from S_{gmk} any itemset with a propersuperset in S_{gmk} or | itemset |<3
8: Return S_{gmk}
9: End

In the DCPS algorithm, Step 2 initializes the number of concurrent processes in the process pool, Steps 4-6 dynamically create processes, where Step 4 establishes the process, Step 5 adds the established process to the process pool, and Step 6 executes the process concurrently. Step 7: Filter groups that do not meet the support threshold.

5. EXPERIMENTS AND RESULT ANALYSIS

5.1. Experimental Setup

We validated and evaluated the performance of MMBS, TVBS, and DCPS algorithms through experimental analysis. We compared the time performance of three algorithms. This article uses Meetup EBSN1 API to obtain partial public datasets of the website from December 1, 2016, to December 31, 2016, including data from four cities: Chicago (CHI), Los Angeles (LA), San Diego (SD), and Washington (WDC). We conducted experiments on datasets of these four cities to validate the MMBS, TVBS, and DCPS algorithms proposed in this article. All experiments were conducted on Lenovo's DeepNex deep learning platform, equipped with Intel Xeon processors, including 20 processor cores and 64GB of DDR4 memory. The MMBS, TVBS, and DCPS algorithms are implemented in Python. The characteristics of the datasets for the four cities are shown in Table 1:

Table 1. Dataset features

	CHI	LA	SD	WDC
$\|E\|$	1495	1413	2278	866
$\|U\|$	3285	1467	2698	2759
$\|IG\|$	6017	7553	5738	6760
$\| R_U^E \|$	33808	11285	26282	17983
$\| R_U^{IG} \|$	38137	22442	27287	37953
$\|E_{\text{Unum>=3}}\|$	1182	955	1746	718
$\|U_{\text{Enum>=3}}\|$	2490	1030	2134	1874
$\|IG_{\text{Unum>=3}}\|$	2126	2124	1848	2423
$\|U_{\text{IGnum>=3}}\|$	3050	1371	2400	2634
$\| E_A^U \|$	22.6	8.0	11.5	20.8
$\| IG_A^U \|$	6.3	3.0	4.8	5.6
$\| U_A^E \|$	10.3	7.7	9.7	6.5
$\| U_A^{IG} \|$	11.6	15.3	10.1	13.8

In Table 1,$|E|$ represents the number of events,$|U|$ represents the number of users, $|IG|$ represents the number of interest groups, $\| R_U^E \|$ represents the number of records of users participating in events, $\| R_U^{IG} \|$ represents the number of records of users participating in interest groups, $|E_{\text{Unum>=3}}|$ represents the number of events with more than 3 participants, $|U_{\text{Enum>=3}}|$ represents the number of users participating in events with more than 3 participants, $|IG_{\text{Unum>=3}}|$ represents the number of interest groups with more than 3 participants, $|U_{\text{IGnum>=3}}|$ represents the number of users participating in interest groups with more than 2 participants, $\| E_A^U \|$ represents the average number of participants in the event, $\| IG_A^U \|$ represents the average number of participants in the interest group, $\| U_A^E \|$ represents the average number of users participating in the event, and $\| U_A^{IG} \|$ represents the average number of users participating in the interest group.

5.2. Experimental Results and Analysis

In the DCPS algorithm, selecting the number of parallel threads in the parallel threads pool is necessary. Setting the number of parallel threads too much or too little can affect the algorithm's performance. Therefore, given the minimum support of 31 in the Los Angeles offline network dataset, we set the number of parallel threads to 10, 15, 20, 25, 30, and 35. The results of running the algorithm are shown in Figure 7. When the number of parallel threads is 25, the DCPS algorithm achieves the best performance. Therefore, in the subsequent experiments of the DCPS algorithm, we set the number of parallel threads in the DCPS algorithm to 25.

We conducted experiments on MMBS, TVBS, and DCPS algorithms on the Los Angeles and Washington datasets, respectively. Figure 8 (a) shows the results of the three algorithms with different support values on the Los Angeles dataset, and Figure 8 (b) shows the results on the Washington dataset. The 30+2 of the horizontal axis in the figure represents support k_1+k_2, so $k= k_1+k_2=30+2=32$. The representation of support in the following figure is the same as here. The values of support k_1 and k_2 consider the average number of participants in the event and interest group and the average number of users participating in the event and interest group.

The memory consumption results of the three methods are shown in Table 2 and Table 3.

We conducted experiments on three algorithms on datasets from Chicago and San Diego. Figures 9 (a) and 9 (b) show the results on these two datasets, respectively.

The memory consumption results are shown in Table 4 and Table 5.

From Figures 8 and 9, it can be observed that the performance of the MMBS algorithm deteriorates rapidly with the decrease in support. The MMBS algorithm only discovers the maximum k fully connected user group based on the frequency of user co-occurrence. It does not utilize the information

Figure 7. Time performance of different parallel threads

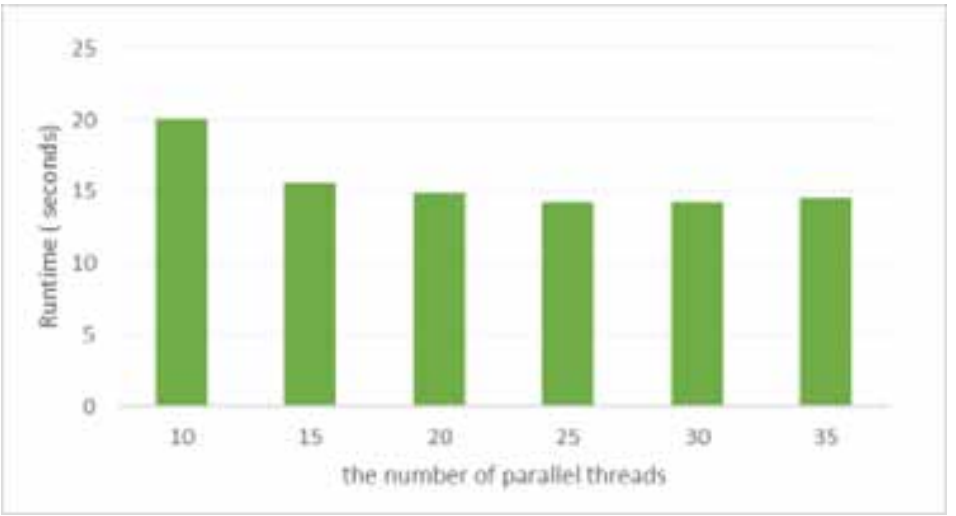
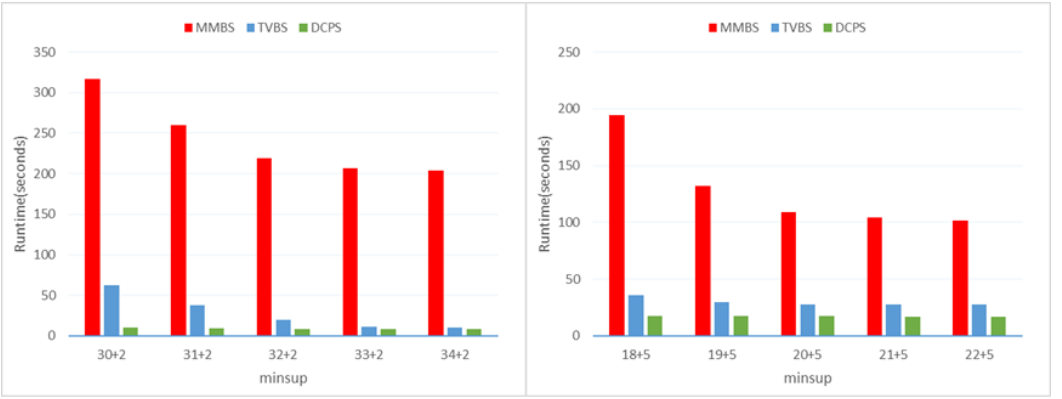


Figure 8. Time performance



(a) Los Angeles (b) Washington

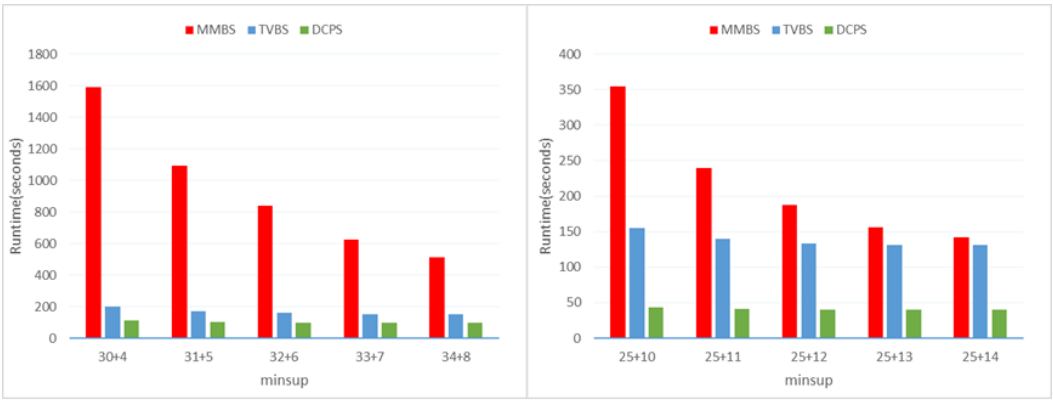
Table 2. Memory consumption

Los_Angeles	30+2	30+3	30+4	30+5	30+6
MMBS	0.1091	0.1073	0.1069	0.1064	0.1061
TVBS	1.7688	1.2447	1.2357	1.2355	1.2351
DCPS	32.674	32.6731	32.6427	32.6124	32.612

Table 3. Memory consumption

Washington	18+5	19+5	20+5	21+5	22+5
MMBS	0.0863	0.0854	0.0853	0.0852	0.0852
TVBS	0.6967	0.6965	0.6964	0.6964	0.6964
DCPS	19.1336	19.1323	19.1321	19.1284	19.0909

Figure 9. Time performance



(a) Chicago (b) San Diego

Table 4. Memory consumption

Chicago	30+4	31+5	32+6	33+7	34+8
MMBS	0.0831	0.0829	0.0826	0.0825	0.0822
TVBS	0.8732	0.8728	0.8724	0.872	0.7747
DCPS	17.0706	17.06	17.0692	17.069	17.0688

Table 5. Memory consumption

San_Diego	25+10	25+11	25+12	25+13	25+14
MMBS	0.0927	0.0924	0.0923	0.0922	0.0919
TVBS	0.7287	0.7285	0.7284	0.7282	0.7281
DCPS	19.451	19.44	19.433	19.421	19.4125

contained in heterogeneous networks or fully connected structures. Therefore, with reduced support, time consumption becomes significant. The overall performance of the DCPS algorithm is better than that of the TVBS algorithm. When the support is high, the performance of the TVBS algorithm will be better than that of the DCPS algorithm. It is because when the support level is high, the number of users to search for decreases, and the maximum number of fully connected user groups also decreases. The DCPS algorithm needs to dynamically establish processes during the search process, which consumes time and memory. Therefore, when the number of users that need to be searched is relatively small, the performance is not as good as the TVBS algorithm.

Tables 2, 3, 4, and 5 indicate that MMBS has the lowest memory consumption, followed by TVBS. DCPS has the highest memory consumption because the search process of TVBS requires user participation in event vectors and interest group vectors, which will occupy a large amount of content space. In contrast, the DCPS algorithm requires establishing multiple processes simultaneously, which takes up more memory than the TVBS algorithm.

The experimental results show that the three algorithms proposed in this article can effectively mine Gmk in the dataset. On the one hand, when processing the same dataset under the same minimum support threshold conditions, the time cost of all three algorithms increases with data volume. The TVBS and DCPS algorithms have good time performance, and the advantage becomes more evident with a larger data volume. We considered three scenarios in the experiment. 1) Fixed the participation threshold for interest groups, only adjusting the participation threshold for events. 2) Fixed the participation threshold for events, only adjusting the participation threshold for interest groups. 3) Simultaneously adjust the interest groups' participation threshold and events' participation threshold. The above three situations comprehensively consider the variation of the k value. All three cases have the same trend. Compared to the MMBS algorithm, the TVBS and DCPS algorithms consume less time, and the advantages of DCPS algorithms are more prominent. On the other hand, the TVBS algorithm and the DCPS algorithm only need to traverse the database once, and when processing the same number of transaction datasets, the execution time of the algorithm changes relatively smoothly as the support count increases. The experiment proves that the three algorithms proposed in this article can fully utilize computers' performance and memory space and efficiently complete Gmk discovery of large-scale data.

Kim et al. (2020) have verified that user groups with close connections on social networking platforms and geographical locations can promote event recommendations, friend recommendations, and more. Therefore, this article did not validate the practical application effect of G_{mk}.

The TVBS algorithm is an effective method for searching user groups in EBSN, which fully utilizes the network structure characteristics of EBSN and combines different entity relationships in EBSN. This algorithm can be applied to search for groups in other heterogeneous social networks such as LBSN.

6. CONCLUSION

This article defines concepts of maximum k fully connected user groups for user relationship networks in EBSN. On this basis, this article proposes the problem of discovering the maximum k fully connected user groups. In order to effectively discover the maximum k fully connected user group, this article proposes the MMBS algorithm, TVBS algorithm, and DCPS algorithm. Experiments conducted on real datasets have found that the MMBS algorithm can quickly discover the maximum k fully connected user group with high support, but as support decreases, the time consumption of the algorithm increases sharply. The TVBS algorithm and DCPS algorithm have better time performance in searching for the maximum k fully connected user group with lower support, and the DCPS algorithm has better time performance than the TVBS algorithm. The memory consumption is the opposite, with the MMBS algorithm having the minor memory consumption, followed by the TVBS algorithm, and the DCPS algorithm having the most memory consumption.

Due to a large number of events and interest groups in event sets and interest group sets in EBSN, generating user event participation vectors and interest group participation vectors requires a large amount of memory, which is not conducive to promoting the algorithm. Therefore, future work will focus on obtaining the maximum k fully connected user groups in time segments, finding approach to group decision-making (Haseli, 2021), studying how users interact with each other, how user groups interact with each other, and how user groups interact with each other. On this basis, we will study the group recommendation model in EBSN.

ACKNOWLEDGEMENT

This work was supported by National Natural Science Foundation of China (No. 61772245), Natural Science Foundation of Fujian Province (No. 2021J011147, No.2021j011144), Nanping Science and Technology Plan Project (No. N2021J007)

REFERENCES

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules, *Proceeding of the 20th Int. Conf. Very Large Data bases, VLDB*, (pp. 487-499). IEEE.

Atzmueller, M. (2015). Subgroup discovery. *Wiley Interdisciplinary Reviews. Data Mining and Knowledge Discovery*, *5*(1), 35–49. doi:10.1002/widm.1144

Bayardo, R. Jr., J. (1998). Efficiently mining long patterns from databases, *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, (pp. 85-93). ACM. doi:10.1145/276304.276313

Belfodil, A., Belfodil, A., Bendimerad, A., Lamarre, P., Robardet, C., Kaytoue, M., & Plantevit, M. (2019). Fssd-a fast and efficient algorithm for subgroup set discovery, *Proceedings of the 2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA).* IEEE. doi:10.1109/DSAA.2019.00023

Burdick, D., Calimlim, M., & Gehrke, J. (2001). Mafia: A maximal frequent itemset algorithm for transactional databases, *Proceedings of the 17th International Conference on Data Engineering*. IEEE. doi:10.1109/ICDE.2001.914857

Chen, S., Nie, L., Tao, X., Li, Z., & Zhao, L. (2020). Approximation of probabilistic maximal frequent itemset mining over uncertain sensed data. *IEEE Access : Practical Innovations, Open Solutions*, *8*, 97529–97539. doi:10.1109/ACCESS.2020.2997409

De, A., Torreão, V., & Vimieiro, R. (2018). Effects of population initialization on evolutionary techniques for subgroup discovery in high dimensional datasets, *Proceedings of the 2018 7th Brazilian Conference on Intelligent Systems (BRACIS),* 25-30.

Deng, J., Kang, B., & Lijffijt, J. (2020). Explainable subgraphs with surprising densities: A subgroup discovery approach. *Proceedings of the 2020 SIAM International Conference on Data Mining*. Society for Industrial and Applied Mathematics. doi:10.1137/1.9781611976236.66

Du, Y., Meng, X., & Zhang, Y. (2019). CVTM: A content-venue-aware topic model for group event recommendation. *IEEE Transactions on Knowledge and Data Engineering*, *32*(7), 1290–1303. doi:10.1109/TKDE.2019.2904066

Du, Y., Meng, X., Zhang, Y., & Lv, P. (2019). GERF: A group event recommendation framework based on learning-to-rank. *IEEE Transactions on Knowledge and Data Engineering*, *32*(4), 674–687. doi:10.1109/TKDE.2019.2893361

Fatemi, S. M., Hosseini, S. M., Kamandi, A., & Shabankhah, M. (2021). CL-MAX: A clustering-based approximation algorithm for mining maximal frequent itemsets. *International Journal of Machine Learning and Cybernetics*, *12*(2), 365–383. doi:10.1007/s13042-020-01177-5

Gouda, K., & Zaki, M. J. (2001) Efficiently mining maximal frequent itemsets, *Proceedings of the 2001 IEEE International Conference on Data Mining.* IEEE. doi:10.1109/ICDM.2001.989514

Grahne, G. (2003). High performance mining of maximal frequent itemsets, *Proceeding of the 6th SIAM International Workshop on High Performance Data Mining*. SIAM.

Hamid, Z., & Khafaji, H. K. (2021). Classification of Arabic documents depending on maximal frequent itemsets, *Proceedings of the Journal of Physics: Conference Series*. IOP Science. doi:10.1088/1742-6596/1804/1/012009

Han, J., Pei, J., & Yin, Y. (2000). Mining frequent patterns without candidate generation. *SIGMOD Record*, *29*(2), 1–12. doi:10.1145/335191.335372

Haseli, G., Sheikh, R., Wang, J., Tomaskova, H., & Tirkolaee, E. B. (1881). A novel approach for group decision making based on the best-worst method (G-BWM): Application to supply chain management. *Mathematics*, *2021*, 9.

Helal, S. (2016). Subgroup discovery algorithms: A survey and empirical evaluation. *Journal of Computer Science and Technology*, *31*(3), 561–576. doi:10.1007/s11390-016-1647-1

Herrera, F., Carmona, C. J., González, P., & del Jesus, M. J. (2011). An overview on subgroup discovery: Foundations and applications. *Knowledge and Information Systems*, *29*(3), 495–525. doi:10.1007/s10115-010-0356-2

Huan, J., Wang, W., & Prins, J. (2004) SPIN: Mining maximal frequent subgraphs from graph databases. *Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining.* ACM.

Jeong, H. J., & Kim, M. H. (2019). HGGC: A hybrid group recommendation model considering group cohesion. *Expert Systems with Applications*, *136*, 73–82. doi:10.1016/j.eswa.2019.05.054

Ji, G. L., Yang M, Song Y Q, et al. (2005). Fast updating maximum frequent itemsets. *Jisuanji Xuebao (Chin. J. Comput.), 28*(1), 128-135.

Ji, K., Chen, Z., Sun, R., Ma, K., Yuan, Z., & Xu, G. (2018). T: A generative model with individual and subgroup-based topics for group recommendation. *Expert Systems with Applications*, *94*, 81–93. doi:10.1016/j.eswa.2017.10.037

Khafaji, H. K. (2021). A new algorithm for extracting textual maximal frequent itemsets from Arabic Documents. *Proceedings of the Journal of Physics: Conference Series,* 012012.

Khrouf, H., & Troncy, R. (2018). Topical community detection in event-based social network[J]. *arXiv preprint arXiv*:1803.04354.

Kim, J., Guo, T., & Feng, K. (2020). Densely connected user community and location cluster search in location-based social networks, *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, (pp. 2199-2209). ACM. doi:10.1145/3318464.3380603

Kuramochi, M., & Karypis, G. (2001) Frequent subgraph discovery, *Proceedings of the 2001 IEEE international conference on data mining.* IEEE. doi:10.1109/ICDM.2001.989534

Lan, T., Guo, L., Li, X., & Chen, G. (2022). Research on the prediction system of event attendance in an event-based social network. *Wireless Communications and Mobile Computing*, *2022*, 1701345. doi:10.1155/2022/1701345

Lemmerich, F., Atzmueller, M., & Puppe, F. (2016). Fast exhaustive subgroup discovery with numerical target concepts [J]. *Data Mining and Knowledge Discovery*, *30*(3), 711–762. doi:10.1007/s10618-015-0436-8

Li, F. (2020). An efficient mining algorithm for maximal frequent patterns in uncertain graph database. *Journal of Intelligent & Fuzzy Systems*, *39*(5), 7021–7033. doi:10.3233/JIFS-200237

Li, X., Sun, C., & Zia, M. A. (2020). Social influence based community detection in event-based social networks. *Information Processing & Management*, *57*(6), 102353. doi:10.1016/j.ipm.2020.102353

Liao, G., & Deng, X. (2020). Leveraging social relationship-based graph attention model for group event recommendation. *Wireless Communications and Mobile Computing*, *2020*, 1–14. doi:10.1155/2020/8834450

Liao, G., Deng, X., & Huang, X. (2020). FHAN: Feature-level hierarchical attention network for group event recommendation, *Proceeding of the Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data*, (pp. 478-492). Springer. doi:10.1007/978-3-030-60259-8_35

Liao, G., Huang, X., Mao, M., Wan, C., Liu, X., & Liu, D. (2019). Group event recommendation in event-based social networks considering unexperienced events. *IEEE Access : Practical Innovations, Open Solutions*, *7*, 96650–96671. doi:10.1109/ACCESS.2019.2929247

Liao, G., Huang, X., & Xiong, N. N. (2006). An intelligent group event recommendation system in social networks. *arXiv preprint arXiv*:2006.08893.

Liao, G. Q., Lan, T. M., Huang, X. M., Chen, H., Wan, C. X., Liu, D. X., & Liu, X. P. (2021). Survey on recommendation systems in event-based social networks. [in Chinese]. *Journal of Software*, *32*(2), 424–444.

Lin, D. I., & Kedem, Z. M. (2002). Pincer-search: An efficient algorithm for discovering the maximum frequent set. *IEEE Transactions on Knowledge and Data Engineering*, *14*(3), 553–566. doi:10.1109/TKDE.2002.1000342

Liu, L., Yu, S., Wei, X., & Ning, Z. (2018). An improved Apriori–based algorithm for friends recommendation in microblog. *International Journal of Communication Systems*, *31*(2), e3453. doi:10.1002/dac.3453

Liu, X., He, Q., & Tian, Y. (2012).Event-based social networks: linking the online and offline social worlds, *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, (pp. 1032-1040). ACM. doi:10.1145/2339530.2339693

Lucas, T., Vimieiro, R., & Ludermir, T. (2018). SSDP+: A diverse and more informative subgroup discovery approach for high dimensional data, *Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC).* IEEE. doi:10.1109/CEC.2018.8477855

Meeng, M., & Knobbe, A. (2021). For real: A thorough look at numeric attributes in subgroup discovery. *Data Mining and Knowledge Discovery*, *35*(1), 158–212. doi:10.1007/s10618-020-00703-x

Nguyen, L. B. Q., Zelinka, I., Snasel, V., Nguyen, L. T. T., & Vo, B. (2022). Subgraph mining in a large graph: A review. *Wiley Interdisciplinary Reviews. Data Mining and Knowledge Discovery*, *12*(4), e1454. doi:10.1002/widm.1454

Padillo, F., Luna, J. M., & Ventura, S. (2017). Exhaustive search algorithms to mine subgroups on big data using apache spark. *Progress in Artificial Intelligence*, *6*(2), 145–158. doi:10.1007/s13748-017-0112-x

Purushotham, S., & Kuo, C. C. J. (2016). Personalized group recommender systems for location-and event-based social networks. [TSAS]. *ACM Transactions on Spatial Algorithms and Systems*, *2*(4), 1–29. doi:10.1145/2987381

Salem, S., Alokshiya, M., & Hasan, M. A. (2021). RASMA: A reverse search algorithm for mining maximal frequent subgraphs. *BioData Mining*, *14*(1), 1–23. doi:10.1186/s13040-021-00250-1 PMID:33726790

Trinh, T., Wu, D., Huang, J. Z., & Azhar, M. (2020). Activeness and loyalty analysis in event-based social networks. *Entropy (Basel, Switzerland)*, *22*(1), 119. doi:10.3390/e22010119 PMID:33285894

Trinh, T., Wu, D., Wang, R., & Huang, J. Z. (2021). An effective content-based event recommendation model. *Multimedia Tools and Applications*, *80*(11), 16599–16618. doi:10.1007/s11042-020-08884-9

Ventura, S., & Luna, J. M. (2018). *Supervised descriptive pattern mining*. Springer International Publishing. doi:10.1007/978-3-319-98140-6

Vinh, T., L., Nguyen, Pham, T, A., Tay, Y., et al. (2019). Interact and decide: Medley of sub-attention networks for effective group recommendation, *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, (pp. 255-264). ACM.

Wang, X., Nie, L., Song, X., Zhang, D., & Chua, T.-S. (2017). Unifying virtual and physical worlds: Learning toward local and global consistency. [TOIS]. *ACM Transactions on Information Systems*, *36*(1), 1–26. doi:10.1145/3052774

Wu, D., Ren, J., & Sheng, L. (2018). Uncertain maximal frequent subgraph mining algorithm based on adjacency matrix and weight. *International Journal of Machine Learning and Cybernetics*, *9*(9), 1445–1455. doi:10.1007/s13042-017-0655-y

Yan, X., & Han, J. (2002). Gspan: Graph-based substructure pattern mining, *Proceedings of the 2002 IEEE International Conference on Data Mining*. *Proceedings of the IEEE*, *2002*, 721–724.

Yin, H., Hu, Z., & Zhou, X. (2016). Discovering interpretable geo-social communities for user behavior prediction, *Proceeding of the IEEE International Conference on Data Engineering*, (pp. 942-953). IEEE. doi:10.1109/ICDE.2016.7498303

Yuan, Q., Cong, G., & Lin, C. Y. (2014). COM: A generative model for group recommendation. *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,* (pp. 163-172). IEEE. doi:10.1145/2623330.2623616

Zhang, Y., Yu, W., Ma, X., Ogura, H., & Ye, D. (2021). Multi-objective optimization for high-dimensional maximal frequent itemset mining. *Applied Sciences (Basel, Switzerland)*, *11*(19), 8971. doi:10.3390/app11198971

Zhang, Y., Yu, W., Zhu, Q., Ma, X., & Ogura, H. (2021). Right-hand side expanding algorithm for maximal frequent itemset mining. *Applied Sciences (Basel, Switzerland)*, *11*(21), 10399. doi:10.3390/app112110399

Tianming Lan received a M.S. degree in computer application technology from Nanchang University, Nanchang, China, in 2006. He is currently pursuing a Ph.D. degree at the School of Information Management, Jiangxi University of Finance and Economics, Nanchang. He is also a Lecturer with the College of Mathematics and Computer, Wuyi University, Nanping, Fujian Province. His research interests include data mining, social network analysis, and recommender systems.

Lei Guo is an Associate Professor at the School of Mathematics and Computer Science at Wuyi University, Nanping, Fujian Province. His research fields include Complex network, deep learning, recommendation systems, etc.

Tianming Lan received the M.S. degree in computer application technology from Nanchang University, Nanchang, China, in 2006. he is currently pursuing the Ph.D. degree with the School of Information Management, Jiangxi University of Finance and Economics, Nanchang. he is also a Lecturer with the College of Mathematics and Computer,Wuyi University, Nanping, Fujian Province. His research interests include data mining, social network analysis, and recommender systems.

Lei Guo is an associate professor at the School of Mathematics and Computer Science at Wuyi University, Nanping, Fujian Province. His research fields include Complex network, deep learning, recommendation systems, etc,