Pattern Mining: Current Challenges and Opportunities

Philippe Fournier-Viger[1], Wensheng Gan[2], Youxi Wu[3], Mourad Nouioua[4], Wei Song[5], Tin Truong[6], and Hai Duong[6]

[1] Shenzhen University, Shenzhen, China
[2] Jinan University, Guangzhou, China
[3] Hebei University of Technology, Tianjin, China
[4] University of Bordj Bou Arreridj, Bordj Bou Arreridj, Algeria
[5] North China University of Technology, Beijing, China
[6] Dalat University, Dalat, Vietnam
philfv@szu.edu.cn,wsgan001@gmail.com,wuc567@163.com,mouradnouioua@gmail.com, songwei@ncut.edu.cn,tintc@dlu.edu.vn, haidv@dlu.edu.vn

Abstract. Pattern mining is a key subfield of data mining that aims at developing algorithms to discover interesting patterns in databases. The discovered patterns can be used to help understanding the data and also to perform other tasks such as classification and prediction. After more than two decades of research in this field, great advances have been achieved in terms of theory, algorithms, and applications. However, there still remains many important challenges to be solved and also many unexplored areas. Based on this observations, this paper provides an overview of six key challenges that are promising topics for research and describe some interesting opportunities. Those challenges were identified by researchers from the field, and are: (1) mining patterns in complex graph data, (2) targeted pattern mining, (3) repetitive sequential pattern mining, (4) incremental, stream, and interactive pattern mining, (5) heuristic pattern mining, and (6) mining interesting patterns.

Keywords: Data Mining · Pattern Mining · Challenges · Opportunities.

1 Introduction

Nowadays, large amounts of data of various types are stored in databases of various organizations. Hence, it has become important for many organizations to develop automatic or semi-automatic tools to analyze data. Pattern mining is a subfield of data mining that aims at identifying interesting and useful patterns in data. The aim is to find patterns that are easily interpretable by users, and thus can help in understanding the data. Patterns can be used to support decision-making but also to perform other tasks such as classification, clustering and prediction. Pattern mining research started more than two decades ago. While initial studies have focused on discovering frequent patterns on data such as shopping data, the field has rapidly changed to consider other data types and

pattern types. Also, major improvements have been made to algorithms and data structures to improve efficiency, scalability, and provide more features.

This paper provides an overview of key challenges and opportunities in pattern mining, that deserve more attention. To write this paper, seven researchers from the field of pattern mining were invited to write about a key challenge of their choice. **Six challenges have been identified:**

1. C1: Mining patterns in complex graph data (*by P. Fournier-Viger*)
2. C2: Targeted pattern mining (*by W. Gan*)
3. C3: Repetitive sequential pattern mining (*by Y. Wu*)
4. C4: Interactive pattern mining (*by M. Nouioua*)
5. C5: Heuristic pattern mining (*by W. Song*)
6. C6: Mining interesting patterns (*by T. Truong and H. Duong*)

The rest of this paper is organized as follows. The Sections 2 to 6 describe the six challenges. Then, Section 7 draws a conclusion.

2 C1: Mining Patterns in Complex Graph Data

The first studies on frequent pattern mining have focused on analyzing transaction databases, which are tables of records described using binary attributes. Although this data representation has many applications, it remains very simple and thus it is unsuitable for many applications where complex data must be analyzed. Hence, a current trend in pattern mining is to develop algorithms to analyze complex data such as temporal data, spatial data, time series and graphs. Graph data has attracted the attention of many researchers in recent years [13, 18] as it can encode various types of information such as social links in friendship-based social networks, chemical molecules, roads between cities, flights between airports, and co-authorship of papers in academia.

Initial studies on graph pattern mining have focused on *frequent subgraph mining*, which aims at finding connected subgraphs that are common to many graphs of a graph database, or that appear frequently in a single graph [18]. In the original problem, the input graphs have a rather simple form. They are static, have vertices and edges which can each have at most one label, the graphs must be connected, self-loops are forbidden (an edge from a vertex to itself), and there can be at most a single edge between two vertices. This simple representation restricts the applications of frequent subgraph mining. To broaden the applications of pattern mining in graphs, the following challenges must be solved.

Handling more complex types of graphs. A key challenge is to consider richer types of graphs such as those shown in Fig. 1 [13]. Those are *directed graphs* (where edges may have directions), *weighted graphs* (where numbers are assigned to edges to indicate the strength of relationships), *attributed graphs* (where each node can be described using many categorical or nominal attributes), *multi-labeled graphs* (where edges and vertices may have multiple labels), and multi-relational graphs (graphs where multiple edges of different types may connect nodes) [13]. For some domains such as social network analysis, handling

rich graph data is crucial. For example, a user profile (vertex) on a social network may be described using many attributes and persons have various types of relationships (edges).

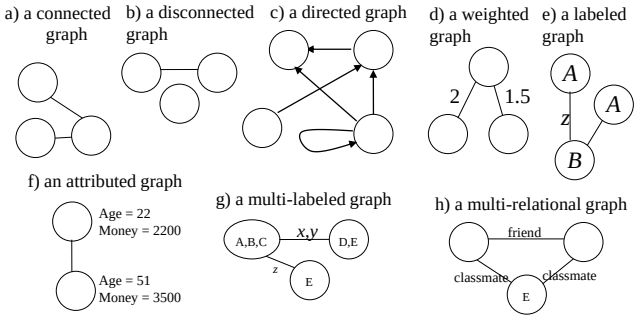

Fig. 1. Different types of graphs

Handling dynamic graphs. Another key challenge is to mine patterns in graphs that change over time. There are three main types of changes: *topological changes* (edges may be added or removed), *label evolution* (node labels may change over time), and a mix of both two cases [13]. Many algorithms for analyzing dynamic graphs adopt a snapshot model where graphs are observed at different timestamps. Other models of time should also be studied such as events having a duration (each event has a start and end time and may overlap with other events). A related research direction is to design algorithms to update patterns incrementally when new data arrives, or to process evolving graphs in real-time as a stream. It is also possible to search for different types of temporal patterns in a dynamic graph such as sequences of changes, periodic patterns (a pattern that is repeating itself over time) and trending patterns (a pattern that has an increasing trend over time) and attribute rules [12].

Discovering specialized types of graphs. Another important challenge in graph pattern mining is to design algorithms that are specialized for mining specific patterns rather than more general patterns. For example, algorithms have been designed to mine sub-trees [29] or paths instead of more complex graphs. The benefit of solving some more specialized graph pattern mining problem is that more efficient algorithm can be developed due to specialized optimizations.

Discovering novel pattern types. Many recent studies have focused on finding new pattern types or to use new criteria to select patterns. For instance, a trend is to design algorithms to find statistically significant patterns based on statistical tests, or to use correlation measures to filter out spurious patterns.

Handling multi-modal data. Another interesting research direction in graph pattern mining is to combine graph data with other data types to perform a joint analysis of this data. Multi-modal data refers to data of different modes such as graph data, combined with video data and audio data.

Developing solutions to applied graph pattern mining problems. Another important research topic in graph mining is to design specialized solutions to address the needs of some given applications. For instance, it was shown to be advantageous to develop custom algorithms to analyze alarm data from a computer network rather than using generic graph mining algorithms [12]. By designing a tailored solution, better performance may be obtained and more interesting patterns (e.g. by using custom measures to select patterns).

3 C2: Targeted Pattern Mining

The current pattern mining literature provides various methods to find all interesting patterns using several parameters. In other words, most of the pattern mining algorithms aims at discovering the complete set of patterns (i.e., itemsets, rules, sequences, graphs, etc.) that satisfy predefined thresholds. However, in general, a huge number of discovered patterns may not be interesting, which are usually based on variations special interest. To filter out redundant information and obtain concise results, *targeted pattern mining* (TPM, or called *targeted pattern search*) provides a different solution to the classic pattern mining problem. To be specific, instead of discovering a large number of patterns that may not be the target ones, users in TPM could input a single or several targets at a time and then discover/query the desired patterns containing the input target [26, 44]. Therefore, the interactive TPM method can return the concise queries with the user-defined targets.

In summary, the goal of TPM is to discover a particular subset or group that contain one or several special patterns. TPM is often more interesting and reliable for finite samples resulting in potential finite subset. Different from traditional pattern mining algorithms, TPM is computationally much more difficult to compute the subset from the potential search space. How to estimate special subset but not the all patterns satisfy the given parameters is quite challenging.

Several definitions about targeted pattern have been provided before, including targeted frequent itemset mining [20], targeted sequence mining [5], targeted high-utility itemset mining [26], and targeted high-utility sequence mining [44]. Up to now, several frequency-based or utility-driven TPM models have been developed, as briefly reviewed below.

Targeted FIM and ARM algorithms. Kubat et al. [20] studied specialized queries of frequent itemsets in a transaction database. All the rules (i.e., targeted queries) can be extracted from the designed Itemset-Tree. Here the user-specified itemset is antecedent in rules. An improved Itemset-Tree [14] was designed for quickly querying frequent itemsets during the operation process. Both algorithms adopt the minimum confidence and support measurement, and the improved Itemset-Tree can be updated incrementally with new transactions. For multitude-targeted mining, the guided FP-growth [30] was designed to determine the frequency of each given itemset based on target Itemset-Tree. After that, a constraint-based ARM query model [1] was also introduced for exploratory analysis of diverse clinical databases.

Targeted SPM algorithms. The sequential ordering of items is commonly seen in real-life applications. To handle the sequence data that is more complex than transaction data, Chueh et al. [8] reversed the original sequences to discover targeted sequential patterns with time intervals. Based on the definition of targeted SPM, Chand et al. [5] proposed a novel SPM algorithm to discover patterns with checking whether they satisfied the recency and monetary constraint and also were target-oriented. However, the target pattern in this approach is defined in the end of each sequence. A goal-oriented algorithm [7] can extract the transaction activities before losing the customer. By utilizing TPM, this algorithm can handle the problem of determining whether a customer is leaving and toward a specific goal.

Utility-driven TPM algorithms. Previous TPM algorithms mainly adopt the measurement of frequency and confidence, but them do not involve the concept of utility [15], which is helpful for discovering more informative patterns and knowledge. Recently, Miao et al. [26] are the first to introduce a targeted high-utility itemset querying model (abbreviated as TargetUM). TargetUM introduced several key definitions and formulated the problem of mining the desired set of high-utility itemsets containing given target items. A utility-based trie tree was designed to index and query target itemsets on-the-fly. Consider the sequence data, Zhang et al. [44] introduced targeted high-utility sequence querying problem and proposed the TUSQ algorithm. Targeted utility-chain and two novel upper bounds on utility measurement (namely suffix remain utility and terminated descendants utility) are proposed in the TUSQ model.

Several open problems of targeted pattern mining/search and interesting directions (including but not limited to) in the future are highlighted in detail below. It is important to note that these open problems are also widespread in other pattern mining tasks.

- **What type of data to be mined**. As we know, there are many types of data in real world, such as transaction data, sequence, streaming data, spatiotemporal data, complex event, time-series, text and web, multi-media, graphs, social network, and uncertain data. How to design effective TPM algorithms to deal with these data is very urgent and more challenging.
- **What kind of pattern or knowledge to be mined**. For example, there are two categories, descriptive vs. predictive data mining, which is based on different kind of knowledge. As reviewed before, itemset, sequence, rule, graph, and event are the different kinds of patterns that are extracted from various types of data. However, few TPM algorithms can discover these kinds of patterns.
- **More effective data structure**. According to the current studies, the indexing and searching in TPM are more challenging than that of traditional pattern mining. In particular, when dealing with big data, we need more effective data structures to store rich information from data.
- **More powerful strategies**. Due to the difficulty, the search space of TPM has an explosion. Thus, how to reduce the search space using powerful prun-

ing strategies (w.r.t. upper bounds) plays a key role in improving the performance of the TPM algorithm.
- **Different applications**. In general, there are many applications of data mining methods, including discrimination, association analysis, classification, clustering, trend/deviation, outlier detection, etc. It is clear that different application requires a special solution of TPM.
- **Visualization.** It is interesting that the data and mining results will be displayed automatically in search process. In the future, there are many opportunities to increase the interpretability of the results, the ease of use of the model, and the interactivity of the mining process.

To summary, targeted pattern mining/search is difficult and quite different from previous mining methods. In the future, there are many opportunities and interesting work in this research field.

4 C3: Repetitive sequential pattern mining

Sequential pattern mining (SPM) has been used in keyphrase extraction [42] and feature selection [41]. The goal of SPM is to discover interesting subsequences (also called patterns). The most common problem is to mine frequent patterns whose supports are no less than a user-defined parameter called *minsup*. The definitions are as follows.

Definition 1. (sequence and sequence database) Suppose we have a set of items σ. A sequence S is an ordered list of itemsets $S = \{s_1, s_2, \cdots, s_n\}$, where s_i is an itemset, which is a subset of σ. A sequence database is composed of k sequences, i.e. $SDB = \{S_1, S_2, \cdots, S_k\}$.

Example 1. For a sale dataset, suppose there are five products: a, b, c, d, and e, i.e. $\sigma = \{a, b, c, d, e\}$. Suppose customer 1 first purchased items a, b, and c, then bought a, b, and e, then purchased c, then bought (a, b, d), and e, then purchased a and c, and finally bought (a, c) and e. The shopping sequence of customer 1 is $S_1 = \{s_1, s_2, s_3, s_4, s_5, s_6\} = \{(a, b, c), (a, b, e), (c), (a, b, d, e), (a, c), (a, c, e)\}$. Similarly, we assume that for customer 2, $S_2 = \{s_1, s_2\} = \{(a, b, d), (c)\}$. Thus, the sequence database is $SDB = \{S_1, S_2\}$.

This kind of sequence format is quite general since the sequence is an ordered list of itemsets, which means that each itemset contains one or more items. Thus, such sequence is called a sequence with itemsets. But for many applications, the data is represented as an ordered list of items called a sequence with items, which means that each itemset contains only one item, e.g. DNA sequences, protein sequences, virus sequences, and time series. For example, "attaaagg" is a segment of the SARS-CoV-2 virus.

Definition 2. (pattern and occurrence) A pattern $P = p_1, p_2, \cdots, p_m$ is also a sequence. A pattern P is a subsequence of a sequence $S = \{s_1, s_2, \cdots, s_n\}$ if and only if $p_1 \subseteq s_{i_1}, p_2 \subseteq s_{i_2}, \cdots, p_m \subseteq s_{i_m}$, and $1 \leq i_1 < i_2 < \cdots < i_m \leq n$. $I = <i_1, i_2, \cdots, i_m>$ is an occurrence of pattern P in sequence S.

Example 2. Pattern $P = \{(a, b), (c)\}$ occurs in sequences S_1 and S_2. For example, <1,3> is an occurrence of pattern P in sequence S_1, since $p_1 = (a, b) \subseteq s_1 = (a, b, c)$ and $p_2 = (c) \subseteq s_3 = (c)$.

Definition 3. (support and frequent pattern) The support is the number of occurrences of a pattern P in a sequence database SDB, represented as $sup(P, SDB)$. If the support is no less than the predefined threshold *minsup*, then the pattern is called a frequent pattern.

Classical SPM cares if a pattern occurs in a sequence or not, but it ignores a pattern's repetitions in a sequence. For example, P occurs in both S_1 and S_2. Thus, the support of P in SDB is 2. However, pattern P occurs many times in sequence S_1. If we neglect the repetition, many important interesting patterns will be lost. However, researchers mainly focused on mining the repetitive patterns in a sequence database with items, rather than in a sequence database with itemsets. Various methods have been investigated to mine various kinds of patterns such as patterns without gap [6], patterns with self-adaptive gap [41], and patterns with gap constraint [21, 27, 28, 40]. An illustrative example is given.

Example 3. Suppose we have a sequence $S = s_1s_2s_3s_4s_5s_6s_7s_8 = aabababa$.

(1) Pattern without gap: Pattern without gap is also called consecutive subsequences [6], i.e. for occurrence $I = <i_1, i_2, \cdots, i_m>$, it requires that $i_2 = i_1 + 1$, $i_3 = i_2 + 1$, $\cdots$, $i_m = i_{m-1} + 1$. For example, there are two occurrences of pattern $P = p_1p_2p_3 =$ aba in sequence S: <2,3,4> and <6,7,8>. The advantage of this method is that it is easy to calculate the support. However, the restriction is too strict, which will lead to the loss of a lot of important information.

(2) Pattern with self-adaptive gap [41]: It means that there is no constraint on the occurrence. For example, <1,7,8> is an occurrence of $P = aba$ in S. The advantage of this method is that users do not need any prior knowledge and it is easy to find the characteristics of the sequence database. However, there are too many occurrences, which will lead to difficulties in analyzing the results.

(3) Pattern with gap constraint: In this case, users should predefine a gap = [M, N], and for each occurrence, it needs to satisfy that $M \leq i_k - i_{k-1} - 1 \leq N$ $(1 < k \leq m)$, where M and N are the minimum and maximum wildcards. This method can prune some meaningless occurrences. For example, if gap = [0,2], <1,3,4> is an occurrence of pattern $P = aba$ in sequence S, since $p_1 = s_1 = a$, $p_2 = s_3 = b$, $p_3 = s_4 = a$, and both 1,3 and 3,4 satisfy the gap constraint [0,2], while <1,5,6> is not an occurrence, since 5 - 1 - 1 = 3 > 2. This approach not only is more challenging, but also has many types. As far as we know there are four types: no condition [27], the one-off condition [21], the nonoverlapping condition [40], and the disjoint condition [28].

• No condition means that each item can be reused [27]. Therefore, all ten occurrences of $P = aba$ with gap [0,2] in sequence S are acceptable under no condition: <1,3,4>, <1,3,6>, <2,3,4>, <2,3,6>, <2,5,6>, <2,5,8>, <4,5,6>, <4,5,8>, <4,7,8>, and <6,7,8>.

• The one-off condition means that each item can be used at most once [21]. Therefore, <1,3,4> and <2,5,6> are two occurrences of P in S which satisfy the one-off condition, while <1,3,4> and <4,5,6> do not.

• The nonoverlapping condition means that each item cannot be reused by the same p_j, but can be reused by different p_j [40]. <1,3,4> and <4,5,6> satisfy the nonoverlapping condition, since in <1,3,4>, p_3 matches $s4$ and in <4,5,6> p_1 matches s_4. However, <1,3,6> and <2,3,4> do not satisfy the nonoverlapping condition, since in both occurrences, p_2 matches s_3. Hence, there are three occurrences of P in S under the nonoverlapping condition: <1,3,4>, <4,5,6>, and <6,7,8>.

• The disjoint condition means that the maximum position of an occurrence should be less than the minimum position of the next occurrence [28]. For example, there are two occurrences of P in S: <1,3,4> and <6,7,8>.

Although the four conditions are very similar, their characteristics are different. The advantages of no condition are that the support can be calculated in polynomial time and it is a complete mining approach. However, this mining approach does not satisfy the Apriori property and has to apply the Apriori-like strategy to generate candidate patterns. For the one-off condition, although it satisfies the Apriori property, it cannot exactly calculate its support, since it is an NP-Hard problem. Therefore, the mining approach is an approximate mining approach. Although both the nonoverlapping condition and the disjoint condition satisfy the Apriori property, and the support can be calculated in polynomial time [39], the disjoint condition is easier to calculate than the nonoverlapping condition and may lose some feasible occurrences.

If we apply the four conditions in a sequence database with itemsets, it is a more challenging task, since the support calculation and candidate generation are significantly different from those for a sequence database with items. For the support calculation, in a sequence with items, we require that $p_j = s_{i_j}$, while in a sequence with itemsets, we require that $p_j \subseteq s_{i_j}$. For candidate generation, in a sequence database with items, we only apply S-Concatenation to generate candidates, while in a sequence database with itemsets, we adopt S-Concatenation and I-Concatenation to generate candidates.

Hence, the following tasks should be further investigated in sequence databases with itemsets. 1). What are the computational complexities of calculating the supports under different conditions? 2). Given a database with itemsets, how to design effective mining algorithms for these conditions? 3). If the dataset is dynamic or a stream database, how to design effective mining algorithms? 4). A variety of SPM methods were proposed to meet different requirements, such as closed SPM, maximal SPM, top-k SPM, compressing SPM, co-occurrence SPM, rare SPM, negative SPM, tri-partition SPM, and high utility SPM. However, most of them neglect the repetitions and consider sequence databases with itemsets. If the repetitions cannot be neglected, how to design effective mining algorithms? 5). For a specific problem, there are many approaches to solve it. However, what is the best approach? For example, for a sequence classification problem, there are many methods to extract the features, such as frequent pat-

terns and contrast patterns under the four conditions. However, which one is the best approach?

5 C4: Incremental, Stream and Interactive Pattern Mining

A key limitation of traditional pattern mining algorithms such as Apriori and FP-Growth is that they are batch algorithms. This means that if the input database is updated, the user needs to run again the algorithm to get new results even if the database is slightly changed. Consequently, classical algorithms are inefficient for various real applications where databases are dynamics. To address this challenge, various approaches have been adopted which can be roughly classified into three categories: (1) Incremental pattern mining algorithms, (2) Stream pattern mining algorithms and (3) Interactive pattern mining algorithms.

Incremental pattern mining algorithms are designed to update the set of discovered patterns once the database is updated by inserting or deleting some transactions. To avoid repetitively scanning the database, a strategy is to use a buffer that contains the set of almost frequent itemsets in memory [19, 23]. **Stream pattern mining algorithms** are designed to deal with databases that change in real-time and where new data may arrive at a very high speed. These algorithms aim to process transactions quickly to return an approximate set of patterns rather that the complete set. Two representative algorithms for incremental pattern mining are estDec and estDec+ [32]. estDec employs a lexicographic tree structure called a prefix tree to identify and maintain significant itemsets from an online data stream. Significant itemsets are itemsets that may be frequent itemsets in the near future. It has been observed that the size of the prefix tree, which is located in the main memory, becomes very large as the number of significant itemsets increases. Thus, if the size of the prefix tree becomes larger than the available memory space, estDec fails to identify new significant itemsets. As a result, the accuracy of estDec results is degraded [32]. estDec+ and other algorithms have been designed to solve this problem.

Interactive pattern mining tries to handle dynamic databases differently by injecting users preferences, users feedback or user targeted queries, into the mining process [3, 4, 14, 20, 22]. In contrast with incremental and stream pattern mining where algorithms aim to maintain and update a large set of patterns that may be uninteresting to users, interactive pattern mining algorithms focus only on some specific sets of patterns that are needed by the user. Besides, several approaches have been designed which can generally be classified in three categories: (1) Targeted querying based approaches, (2) Users feed-backs based approaches and (3) Visualization based approaches.

Targeted querying based approaches. These approaches let the user search for patterns containing specific items by sending some targeted queries to the system to search for interesting patterns. Then, the system interacts and tries to give quick answers to the user queries [14, 20, 22]. See Section 3 for more details.

Users feedback based approaches. Users feedback based approaches are more interactive comparing with targeted querying based approaches. The key idea is to progressively address feedback sent by users during the mining process. Bhuiyan et al. [4] proposed an interactive pattern mining system that is based on the sampling of frequent patterns from hidden datasets. Hidden datasets exist in various real applications where the data owner and the data analyst is no necessarily the same entity. Thus, the data analyst may not have the full access to the data and the data owner has to maintain the confidentiality of the data by providing to analysts only some samples from data that would be beneficial to him but without giving him the possibility to reconstruct the entire dataset from the given samples [4]. The proposed interactive systems aims to continuously update effective sampling distributions by binary feedback from the users. The proposed system works as follows: Using a Markov Chain Monte Carlo (MCMC) sampling method, the system return a small set of frequent patterns (samples) to each analysts (user). Then, each analyst sends a feedback about its associated samples. The feedback used in this method is a simple feedback where the response of a user on a pattern is to indicate if this pattern is interesting or uninteresting. The system defines a scoring function based on users' feedback and updates each sampling distribution taking into consideration its corresponding user's interests. Following these steps, the proposed system can progressively address the user preferences so that the data remains confidential. Experiments on itemset and graph mining datasets demonstrate the usefulness of the proposed system. Based on the same approach, an improved version of this system was proposed [3]. Besides, authors have adopted a better scoring function for graph data by using graph topology and new improved feedback mechanisms, namely, periodic feedback and conditional periodic feedback.

Another common problem in pattern mining that motivates researchers to design interactive pattern discovery tools is the problem of pattern explosion [11]. More precisely, traditional pattern mining algorithms discover a large number of patterns, of which many are redundant or similar. As a result, the analyst or the data expert should invest substantial efforts to look for the desired patterns which is not an easy task. To overcome this limitation, an interactive pattern discovery framework was proposed [11] for two mining tasks, frequent itemset mining and subgroup discovery. The proposed framework consists of three steps: (1) Mining patterns, (2) Interacting with the user and (3) learning user-specific pattern interestingness. Besides, The user is only asked to rank small sets of patterns, while a ranking function is inferred from users feedback using preference learning techniques. In the experimental results, it has been demonstrated that the system was able to learn accurate pattern rankings for both mining tasks.

Visualization based approaches. Another important aspect to design a good interactive pattern mining system is the visualisation aspect. More precisely, data visualisation techniques play an important role in making the discovered knowledge understandable and interpretable by humans [17]. In fact, the output of the implemented algorithms is presented to the user only in a textual form, which may impose many limitations such as the difficulty to identify similar

patterns and the difficulty to understand the relation between patterns. There are various visualization techniques for different forms of patterns. For instance, researchers [2], have used a lattice based representation based on the Hasse diagram to visualise the output of frequent itemset mining. All possible itemsets can be represented in the diagram and the frequent itemsets are highlighted in bold. Other visualisation techniques have been used to efficiently present itemsets to the user such as pixel based visualization and tree based visualisation [17]. As for itemsets mining, various visualization tools were proposed for the other pattern mining problems such as mining association rules, mining sequential patterns and mining episodes. The reader can refer to [17] where a detailed survey that present the visualisation techniques designed for each mining task.

6 C5: Heuristic Pattern Mining

Since the emergence of the subfield, the excessive number of results caused by combinatorial explosion has been the most fundamental problem encountered in pattern mining. Although many efficient pattern mining algorithms have been proposed, the excessive results still lead to a high computational cost, and the high computational cost required to mine all exact patterns is not proportional to the actionability of the mining results. Occasionally, the result of a large number of discovered patterns can even lead to decision makers not knowing how to use them. Furthermore, for application fields such as recommender systems, it is not necessary to use all exact patterns.

To solve this problem, heuristic pattern mining (HPM) algorithms have been developed to identify an approximate subset of all patterns within a reasonable time. Inspired by biological [9] and physical [35] phenomena, heuristic methods are effective for solving combinatorial problems such as pattern mining. Compared with exact pattern mining algorithms, HPM algorithms are efficient and do not need any domain knowledge in advance. Several heuristic algorithms are used in the subfield of pattern mining, such as the genetic algorithm (GA) [9], particle swarm optimization (PSO) [24], artificial bee colony (ABC) [33] , cross-entropy (CE) [35], and bat algorithm (BA) [34]. In addition to achieving high efficiency, these HPM algorithms can also discover a sufficient number of exact patterns. From the perspective of the key components of the entire mining process, the following challenges are summarized.

Identifying the appropriate objective. For heuristic methods, a fitness function is the objective function used to measure the performance of each individual to determine which will survive and reproduce into the next generation. In existing HPM algorithms, typical measures, for example, support [9] and utility [34], are used as fitness functions directly. Standard fitness functions are easy to implement; however, for more specific patterns (closed pattern or top-k pattern), flexible fitness functions must be applied, which are more difficult to implement. Furthermore, in complex scenarios, for example, choosing sets of products that are both economical and fast to deliver, using multi-objective fitness functions [45] is also a challenging issue.

Speed up the mining process. Determining how to narrow the search space is a general key issue in algorithm design. HPM algorithms also have this problem. The downward closure property is the most widely used principle in HPM algorithms. Specifically, support is used for frequent itemsets [9], and transaction-weighted utilization is used for high utility itemsets [34]. Other strategies include using the lengths of the discovered patterns to generate promising patterns [33] and developing a tree structure to avoid invalid combinations [24]. Considering the characteristics of heuristic methods and the resulting patterns, it would be interesting to develop a new data structure or pruning strategies to improve mining efficiency.

Diversifying the discovered results. Heuristic methods are likely to fall into local optima. This is also true for HPM algorithms. When HPM algorithms fall into local optimal values, it is difficult to produce new patterns in subsequent iterations. Inspired by the mutation operator of the GA, randomly generating some individuals in the next generation is the most important approach to avoid falling into local optima [9]. Although determining how to increase the diversity of results is difficult, a first step is to measure the diversity of results, such as the bit edit distance [35].

Designing a general framework . Pattern mining is different from problems in which there are relatively few best values, all patterns have support/utilities or other measures no lower than the minimum threshold are the mining targets. In addition to the GA, PSO, ABC, CE, and BA, other heuristic methods such as the ant colony system and dolphin echolocation optimization have been used for pattern mining recently. However, attempting to use each heuristic method individually to mine patterns is not only expensive but also infeasible. Therefore, integrating all the objectives, processes, and results into a general HPM framework is a promising approach [10, 34].

At the present time, HPM mainly focuses on itemsets. Thus, discovering a complex pattern, for example, sequential pattern or graph pattern [31], using heuristic methods is challenging.

7 C6: Mining Interesting Patterns

The problem of Interesting Pattern Mining (IPM) plays an important role in Data Mining. The goal is to discover patterns of interest to users in databases (DBs) where interest is measured using functions. One of the first interestingness functions used in IPM is the support function to mine frequent patterns (FPM) in binary DBs. A pattern is said to be frequent in a binary DB if the number of its appearances in transactions of the database (or its support) is no less than a user-predefined minimum support threshold. For the special measure, in order to efficiently solve the combinatorial explosion in FPM, a nice property of the support, the Downward Closure or Anti-Monotonicity - AM, has been applied. This property states that if a pattern is not frequent (infrequent), all its super-patterns are also infrequent, or the whole branch rooted at the infrequent pattern (on the prefix search tree) can be pruned immediately.

However, the support measure is not suitable for all applications. Thus, other interestingness functions have been designed to find important patterns that may be rare but useful or interesting for many real-life applications. Some of the most popular kinds are utility functions of patterns in quantitative DBs (QDBs). Utility functions can be used for example to find the most profitable purchase patterns in customer transactions. Note that the support can be seen as a special utility function. A simple QDB is called a quantitative transaction DB (QTDB), where each (input) transaction is a quantitative itemset (a set of quantitative items). A more general QDB is quantitative sequence DB (QSDB) of which each input quantitative sequence consists of a sequence of quantitative itemsets.

Moreover, a key challenge in the problem of high utility pattern mining (HUPM) is that such utility functions usually do not satisfy the AM property. To overcome this challenge, we need to devise upper bounds or weak upper bounds (on the utilities) that satisfy the AM property or weaker (such as Anti-Monotone like - AML) ones. In this context, given a utility function u of patterns, a function ub is said to be an upper bound (UB) on u if $ub(x) \geq u(x)$ for any pattern x. And a function wub is said to be a weak upper bound (WUB) on u if $wub(x) \geq u(y)$ for any extension pattern y of x. Usually, given a (W)UB, the tighter (W)UB is, the stronger its pruning ability is. The effort and time for devising good and tight (W)UBs is often very long.

For example, in the first problem of high utility itemset mining (HUIM) on a QTDB D, the utility u of an itemset A is defined as the summation of its utilities $u(A,T)$ in all transactions T of D containing A, where the utility $u(A,T)$ of A in T is the summation of utilities of items of A appearing in T. Similarly, for the second problem of high average utility itemset mining (HAUIM) in a QTDB D, the average utility au of an itemset A is defined as the utility $u(A)$ divided by its length $length(A)$. From the first time 2004 [43] (2009) where HUIM (HAUIM, respectively) was proposed, it took more than 8 (10) years to obtain good tighter UBs based on the remaining utility [25] (WUBs based on vertical representation of QTDB [37], respectively). It is worthy to note that for the average utility au, besides UBs (on it), there are many WUBs, which are much tighter than the UBs. The number of WUBs found so far is about five times more than that of UBs, and devising such good WUBs requires much effort and time.

For the more general problems of high utility sequence mining (HUSM) on a QSDB D, because each sequence α may appear multiple times in an input quantitative sequence (IQS) Ψ of D, there are many ways to define the utility of α in Ψ. There are two popular kinds of such utilities, denoted as $u_{max}(\alpha, \Psi)$ and $u_{min}(\alpha, \Psi)$, that are respectively defined as the maximum and minimum values among utilities of occurrences of α in Ψ. Then, $u_{max}(\alpha)$ and $u_{min}(\alpha)$ are respectively the summation of $u_{max}(\alpha, \Psi)$ and $u_{min}(\alpha, \Psi)$ of α in all IQSs Ψ containing α. Similarly, there are two other kinds of utilities named $au_{max}(\alpha)$ and $au_{min}(\alpha)$ that are respectively defined as $u_{max}(\alpha)$ and $u_{min}(\alpha)$ divided by length of α. For the first (third) utility u_{max} (au_{max}), to find good UBs [16] (WUBs [36], respectively), it took about 10 years (8 years, respectively).

Furthermore, devising such UBs (for example on u_{max}) without mathematically proving it strictly may lead to inexactness in corresponding algorithms [16].

For the new second (fourth) utility u_{min} (au_{min}), the time for devising good UBs (WUBs) on it has decreased significantly only in one paper (e.g. [38] for au_{min}). Thus, from the theoretical results presented in the paper, a natural and useful question that has been raised is how to propose a generic framework for the IPM problem according to any new interestingness function, and a general and simple method to quickly design (W)UBs on functions using weeks instead of years? In more details, given a QSDB D and a new interestingness function itr that may not satisfy AM and a user-specified minimum interestingness threshold mi, the corresponding IPM problem is to mine the set $\{\alpha|itr(\alpha) \geq mi\}$ of all highly interesting patterns. The *first* question is how to quickly devise (W)UBs on itr so that they are as tight as possible and have anti-monotone-like properties? The goal of these requirements is to allow significantly reducing the search space. The *second* question is how to transform checking the anti-monotone-like properties of itr in the whole D into simpler one in each input quantitative sequence? Moreover, these theoretical results must be proven strictly in mathematical language. Then, the main challenge that aims at significantly reducing time for devising good (W)UBs on itr will be solved.

8 Conclusion

The field of pattern mining has been rapidly changing. This paper has provided an overview of six key challenges, each identified by a researcher from the field.

References

1. Abeysinghe, R., Cui, L.: Query-constraint-based mining of association rules for exploratory analysis of clinical datasets in the national sleep research resource. BMC Medical Informatics and Decision Making **18**(2), 58 (2018)
2. Alsallakh, B., Micallef, L., Aigner, W., Hauser, H., Miksch, S., Rodgers, P.: The state-of-the-art of set visualization. In: Computer Graphics Forum. vol. 35, pp. 234–260. Wiley Online Library (2016)
3. Bhuiyan, M., Hasan, M.A.: Interactive knowledge discovery from hidden data through sampling of frequent patterns. Statistical Analysis and Data Mining: The ASA Data Science Journal **9**(4), 205–229 (2016)
4. Bhuiyan, M., Mukhopadhyay, S., Hasan, M.A.: Interactive pattern mining on hidden data: a sampling-based solution. In: Proceedings of the 21st ACM Intern. conference on Information and knowledge management. pp. 95–104 (2012)
5. Chand, C., Thakkar, A., Ganatra, A.: Target oriented sequential pattern mining using recency and monetary constraints. Intern. Journal of Computer Applications **45**(10) (2012)
6. Chen, M.S., Park, J.S., Yu, P.S.: Efficient data mining for path traversal patterns. IEEE Transactions on Knowledge and Data Engineering **10**(2), 209–221 (1998)
7. Chiang, D.A., Wang, Y.F., Lee, S.L., Lin, C.J.: Goal-oriented sequential pattern for network banking churn analysis. Expert Systems with Applications **25**(3), 293–302 (2003)

8. Chueh, H.E., et al.: Mining target-oriented sequential patterns with time-intervals. Intern. Journal of Computer Science & Information Technology **2**(4), 113–123 (2010)
9. Djenouri, Y., Comuzzi, M.: Combining apriori heuristic and bio-inspired algorithms for solving the frequent itemsets mining problem. Inf. Sci **420**, 1–15 (2017)
10. Djenouri, Y., D., D., Belhadi, A., Fournier-Viger, P., Lin, J.C.W.: A new framework for metaheuristic-based frequent itemset mining. Appl. Intell **48**(12), 4775–4791 (2018)
11. Dzyuba, V., Leeuwen, M.v., Nijssen, S., De Raedt, L.: Interactive learning of pattern rankings. Intern. Journal on Artificial Intelligence Tools **23**(06), 1460026 (2014)
12. Fournier-Viger, P., Cheng, C., Cheng, Z., Lin, J.C., Selmaoui-Folcher, N.: Mining significant trend sequences in dynamic attributed graphs. Knowl. Based Syst. **182** (2019)
13. Fournier-Viger, P., He, G., Cheng, C., Li, J., Zhou, M., Lin, J.C., Yun, U.: A survey of pattern mining in dynamic graphs. Wiley Interdiscip. Rev. Data Min. Knowl. Discov. **10**(6) (2020)
14. Fournier-Viger, P., Mwamikazi, E., Gueniche, T., Faghihi, U.: Meit: Memory efficient itemset tree for targeted association rule mining. In: Intern. Conference on Advanced Data Mining and Applications. pp. 95–106. Springer (2013)
15. Gan, W., Lin, J.C.W., Fournier-Viger, P., Chao, H.C., Tseng, V.S., Yu, P.S.: A survey of utility-oriented pattern mining. IEEE Transactions on Knowledge and Data Engineering **33**(4), 1306–1327 (2021)
16. Gan, W., Lin, J.C.W., Zhang, J., Chao, H.C., Fujita, H., Philip, S.Y.: ProUM: Projection-based utility mining on sequence data. Information Sciences **513**, 222–240 (2020)
17. Jentner, W., Keim, D.A.: Visualization and visual analytic techniques for patterns. High-Utility Pattern Mining pp. 303–337 (2019)
18. Jiang, C., Coenen, F., Zito, M.: A survey of frequent subgraph mining algorithms. Knowledge Engineering Review **28**, 75–105 (2013)
19. Koh, J.L., Shieh, S.F.: An efficient approach for maintaining association rules based on adjusting fp-tree structures. In: Intern. Conference on Database Systems for Advanced Applications. pp. 417–424. Springer (2004)
20. Kubat, M., Hafez, A., Raghavan, V.V., Lekkala, J.R., Chen, W.K.: Itemset trees for targeted association querying. IEEE Transactions on Knowledge and Data Engineering **15**(6), 1522–1534 (2003)
21. Lam, H.T., Morchen, F., Fradkin, D., Calders, T.: Mining compressing sequential patterns. Statistical Analysis and Data Mining: The ASA Data Science Journal **7**(1), 34–52 (2014)
22. Li, X., Li, J., Fournier-Viger, P., Nawaz, M.S., Yao, J., Lin, J.C.W.: Mining productive itemsets in dynamic databases. IEEE Access **8**, 140122–140144 (2020)
23. Lin, C.W., Hong, T.P., Lu, W.H.: The pre-fufp algorithm for incremental mining. Expert Systems with Applications **36**(5), 9498–9505 (2009)
24. Lin, J.C.W., Yang, L., Fournier-Viger, P., Hong, T.P., Voznak, M.: A binary pso approach to mine high-utility itemsets. Soft Comput **21**(17), 5103–5121 (2017)
25. Liu, M., Qu, J.: Mining high utility itemsets without candidate generation. In: Proceedings of the 21st ACM Intern. conference on Information and knowledge management. pp. 55–64 (2012)
26. Miao, J., Wan, S., Gan, W., Sun, J., Chen, J.: TargetUM: Targeted high-utility itemset querying. arXiv preprint arXiv:2111.00309 (2021)

27. Min, F., Zhang, Z.H., Zhai, W.J., Shen, R.P.: Frequent pattern discovery with tri-partition alphabets. Information Sciences **507**, 715–732 (2020)
28. Ouarem, O., Nouioua, F., Fournier-Viger, P.: Mining episode rules from event sequences under non-overlapping frequency. In: Intern. Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems. Springer, Cham. pp. 73–85 (2021)
29. Qu, W., Yan, D., Guo, G., Wang, X., Zou, L., Zhou, Y.: Parallel mining of frequent subtree patterns. In: Software Foundations for Data Interoperability and Large Scale Graph Data Analytics, pp. 18–32. Springer (2020)
30. Shabtay, L., Yaari, R., Dattner, I.: A guided fp-growth algorithm for multitude-targeted mining of big data. arXiv preprint arXiv:1803.06632 (2018)
31. Shelokar, P., Quirin, A., Cordón, O.: Three-objective subgraph mining using multiobjective evolutionary programming. Comput. Syst. Sci **80**(1), 16–26 (2014)
32. Shin, S.J., Lee, D.S., Lee, W.S.: Cp-tree: An adaptive synopsis structure for compressing frequent itemsets over online data streams. Information Sciences **278**, 559–576 (2014)
33. Song, W., Huang, C.: Discovering high utility itemsets based on the artificial bee colony algorithm. In: The 22nd Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining. Springer, Cham. pp. 3–14 (2018)
34. Song, W., Huang, C.: Mining high utility itemsets using bio-inspired algorithms: a diverse optimal value framework. IEEE Access **6**, 19568–19582 (2018)
35. Song, W., Zheng, C., Huang, C., Liu, L.: Heuristically mining the top-k high-utility itemsets with cross-entropy optimization. Appl. Intell
36. Truong, T., Duong, H., Le, B., Fournier-Viger, P.: Ehausm: An efficient algorithm for high average utility sequence mining. Information Sciences **515**, 302–323 (2020)
37. Truong, T., Duong, H., Le, B., Fournier-Viger, P., Yun, U.: Efficient high average-utility itemset mining using novel vertical weak upper-bounds. Knowledge-Based Systems **183**, 104847 (2019)
38. Truong, T., Duong, H., Le, B., Fournier-Viger, P., Yun, U.: Frequent high minimum average utility sequence mining with constraints in dynamic databases using efficient pruning strategies. Applied Intelligence pp. 1–23 (2021)
39. Wu, Y., Shen, C., Jiang, H., Wu, X.: Strict pattern matching under non-overlapping condition. Science China Information Sciences **50**(1), 012101 (2017)
40. Wu, Y., Tong, Y., Zhu, X., Wu, X.: Nosep: Nonoverlapping sequence pattern mining with gap constraints. IEEE Transactions on Cybernetics **48**(10), 2809–2822 (2018)
41. Wu, Y., Wang, Y., Li, Y., Zhu, X., Wu, X.: Self-adaptive nonoverlapping contrast sequential pattern mining. IEEE Transactions on Cybernetics (2021)
42. Xie, F., Wu, X., Zhu, X.: Efficient sequential pattern mining with wildcards for keyphrase extraction. Knowledge-Based Systems **115**, 27–39 (2017)
43. Yao, H., Hamilton, H.J., Butz, C.J.: A foundational approach to mining itemset utilities from databases. In: Proceedings of the 2004 SIAM Intern. Conference on Data Mining. pp. 482–486. SIAM (2004)
44. Zhang, C., Du, Z., Dai, Q., Gan, W., Weng, J., Yu, P.S.: TUSQ: Targeted high-utility sequence querying. arXiv preprint arXiv:2103.16615 (2021)
45. Zhang, L., Fu, G., Cheng, F., Qiu, J., Su, Y.: A multi-objective evolutionary approach for mining frequent and high utility itemsets. Appl. Soft Comput **62**, 974–986 (2018)