Applied Intelligence (2022) 52:16458–16474
https://doi.org/10.1007/s10489-021-03122-7

Mining sequential patterns with flexible constraints from MOOC data

Wei Song[1] · **Wei Ye**[1] · **Philippe Fournier-Viger**[2]

Accepted: 16 December 2021 / Published online: 23 March 2022

Abstract
Online learning is playing an increasingly important role in education. Massive open online course (MOOC) platforms are among the most important tools in online learning, and record historical learning data from an extremely large number of learners. To enhance the learning experience, a promising approach is to apply sequential pattern mining (SPM) to discover useful knowledge in these data. In this paper, mining sequential patterns (SPs) with flexible constraints in MOOC enrollment data is proposed, which follows that research approach. Three constraints are proposed: the length constraint, discreteness constraint, and validity constraint. They are used to describe the effect of the length of enrollment sequences, variance of enrollment dates, and enrollment moments, respectively. To improve the mining efficiency, the three constraints are pushed into the support, which is the most typical parameter in SPM, to form a new parameter called support with flexible constraints (SFC). SFC is proved to satisfy the downward closure property, and two algorithms are proposed to discover SPs with flexible constraints. They traverse the search space in a breadth-first and depth-first manner. The experimental results demonstrate that the proposed algorithms effectively reduce the number of patterns, with comparable performance to classical SPM algorithms.

Keywords Sequential pattern · MOOC · Support with flexible constraints · Downward closure property

1 Introduction

Online education is popular at present because of school closures caused by the breakout of COVID-19 [25]. Since the beginning of 2020, almost all students around the world have experienced online study. Massive open online courses (MOOCs) have become the main online learning method. The main MOOC platforms, for example, EdX and Coursera, are collecting historical learning data from an increasing number of students. Thus, discovering knowledge from MOOC data is a promising approach to improve online learning quality.

Data mining is the non-trivial process of identifying valid, novel, potentially useful, and ultimately, understandable patterns from an extremely large volume of data. As one of the main data mining tasks, pattern mining [35] discovers various interesting, useful, and unexpected patterns efficiently and effectively. Itemsets [32], sequential patterns (SPs) [12], and sub-graphs [7] are typical patterns discovered in pattern mining.

In this paper, data mining is used, or specifically, pattern mining techniques, to discover knowledge hidden in MOOC data. Online learning activities involve temporal factors; hence, SPs play an important role. Thus, mining SPs in learners' historical data is a promising approach to improve online learning quality.

Given a sequence database, the problem of SP mining (SPM) is to discover subsequences whose supports are no lower than a user-specified minimum support [12]. Many algorithms have been proposed, most of which focus on developing efficient strategies for identifying all SPs, which can be categorized into three broad classes: Apriori-based [33], vertical database format [41], and projection-based pattern growth algorithms [27]. Generally, numerous SPs are discovered by typical SPM algorithms, which makes it difficult for people to identify meaningful results. To address this limitation, various constraints, such as gap [26] and discreteness constraints [38], are used to discover effective and actionable SPs.

✉ Wei Song
songwei@ncut.edu.cn

[1] School of Information Science and Technology, North China University of Technology, Beijing 100144, China

[2] College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China

Recently, SPM has been successfully applied in fields such as vehicle trajectory prediction [42] and electronic medicine [24]. Among these application fields, online education is the most promising application domain at present because of school closures caused by the breakout of COVID-19. Considering the characteristics of MOOC data, flexible constraints are incorporated into typical SPM algorithms to discover meaningful SPs to improve learning quality.

The "Course Recommendation" dataset[1] provided by the MoocData platform is used throughout this paper. The dataset was collected from XuetangX,[2] one of the largest MOOC platforms in China. Originally used for course recommendation, the dataset contains the records of 82,535 course enrollment sequences from XuetangX from October 1, 2016 to March 31, 2018. The characteristics of this dataset are shown in Table 1.

Considering the dataset shown in Table 1, the major parts of this study are as follows:

First, the importance of SPs is evaluated from three aspects: the lengths of enrollment sequences containing them, the variance of days within them, and the moments of enrollments in them. These three aspects are modeled using three constraints.

Second, to make the mining process with three constraints efficient, they are integrated into the support, which is the most general parameter for evaluating SPs, to develop a new parameter called support with flexible constraints (SFC). It is proved that SFC also satisfies the downward closure property.

Third, using breadth-first traversal and depth-first traversal, two algorithms for mining SPs with flexible constraints are proposed and explained. In these two algorithms, SFC is used to replace the support directly.

Finally, extensive experiments were conducted on MOOC data. The results demonstrated that the proposed algorithms effectively reduced the number of discovered results with acceptable efficiency and memory consumption.

The remainder of this paper is organized as follows: Related work is described in Section 2, and the SPM problem is defined in Section 3. In Section 4, the three constraints, in addition to their rationality, are discussed. The mining algorithms are described in detail in Section 5. The experimental results are presented and analyzed in Section 6. Finally, conclusions are drawn in Section 7.

[1] http://moocdata.cn/data/course-recommendation
[2] https://next.xuetangx.com/

Table 1 Characteristics of the Course Recommendation dataset

Feature	Number
Time span	547 days
Number of courses	1,302
Number of sequences	82,535
Length of the longest sequence	398
Length of the shortest sequence	3
Average sequence length	5.19

2 Related work

In this section, first, applications of data mining for MOOC data are reviewed. Then, studies on constraint-based SPM are discussed.

2.1 Data mining from MOOC data

Mining knowledge from MOOC data not only helps instructors to improve their teaching materials and methods but also helps learners to access more appropriate courses or learning paths [1]. Data mining from MOOC data is receiving increasing attention, particularly with the rise of online learning during the COVID-19 pandemic. Learning behavior understanding [16], dropout prediction [8], and personalized learning [43] are typical data mining tasks that use MOOC data.

SPM has become an effective tool for analyzing students' online learning behaviors. Fournier-Viger et al. [10] used SPM techniques to mine frequent action sequences and associations between these sequences in a set of recorded usage of the RomanTutor by novices, intermediates, and experts. Using the discovered SPs, learners' actions were tracked, and suggestions were provided to improve the learners' experience. Kinnebrew et al. used SPM and action abstraction to identify important learning behaviors of students in different groups [19]. In their method, both sequence support and instance support were used to evaluate the resulting SPs.

Using SPs to recommend MOOC teaching materials is a promising approach [34]. Taking a student's sequence of past courses, Wang and Zaïane [36] implemented a course recommender system based on three sequence-related approaches, including SPM. Wong et al. used SPM to verify the effect of self-regulated learning (SRL) [37]. Specifically, SPM was used to explore whether differences exist between learners who viewed the SRL-prompt videos and those who did not. The results demonstrated that the SRL-prompt viewers tended to follow the sequential structure of the course provided by the instructor, whereas this was less likely in the group of SRL-prompt non-viewers.

Different from the above-mentioned SPM-based methods, the object of analysis in the present paper is course enrollment MOOC data rather than device usage data, learning behavior data, and video-viewing data.

2.2 Constraint-based SPM

Many SPM algorithms have been proposed to discover frequent SPs (FSPs) [15], high utility SPs [31], negative SPs [5], and SPs from data streams [18].

In many application domains (e.g., music genre classification) [29], SPs confined by predefined constraints are more meaningful than general SPs. A constraint is an additional set of criteria that the user provides to indicate more precisely the types of patterns to be found. This idea has been used from the beginning of the topic of SPM in the GSP algorithm [33]. For constraint-based SPM, the approach used to push the constraints deep into the mining process is important [23].

Time constraints, generally including gap and duration, are the most widely used constraints in SPM. The gap constraint refers to the minimum and maximum amount of time between two consecutive itemsets within an SP, whereas the duration constraint is the maximum time difference for each SP. Li et al. proposed two gap-constrained algorithms [21]: Gap-BIDE and Gap-Connect. The former mines closed gap-constrained subsequences from a set of input sequences and the latter discovers repetitive gap-constrained subsequences from a single input sequence. Wu et al. solved the problem of SPM with periodic wildcard gaps using the data structure of Nettree [39]. Sqn2Vec [26] and NegPSpan [14] are SPM algorithms that use time constraints, and TRuleGrowth [11] is an algorithm for mining sequential rules with a sliding-window constraint.

Length constraints that restrict the minimum/maximum number of items per SP are also commonly used in SPM. cSpade [41] incorporates max-gap, max-span, and length constraints. The length-decreasing support constraint was proposed by Seno and Karypis [30]. Their algorithm SLP-Miner finds all the FSPs whose support decreases as a function of their length. Thus, long SPs that usually have lower supports can also be discovered. WSLPMiner is also an SPM algorithm with a length-decreasing support constraint [40].

Aggregate constraints are imposed on an aggregate of items in an SP, where aggregate functions can be those involving the average, general sum, or minimum/maximum number. Chen et al. proposed the PTAC algorithm to discover SPs with tough aggregate constraints [2]. In their algorithm, two strategies that avoid an unnecessary item check and unnecessary projected database generation are used to improve the efficiency and memory consumption.

Other typical constraints used for SPM also exist, such as the item constraint [6], discreteness constraint [38], and norm constraint [4].

3 Preliminaries

Let Σ be a set of courses. An *item* is represented as a pair (c, t), where $c \in \Sigma$ is a course and t is the enrollment time of c. A *sequence* $S = <(c_1, t_1), (c_2, t_2), \ldots, (c_n, t_n)>$ is a list of time-ordered items, where for any $1 \le i < j \le n$, $t_i < t_j$ holds. The *length* of sequence S, denoted by $|S|$, is the total number of items in S. $S[i]$ $(1 \le i \le n)$ denotes the ith item in S, and $S[i].c$ and $S[i].t$ are the course and enrollment time of $S[i]$, respectively. It should be noted that, at each time, only a single item rather than an itemset is used in this paper. This is because students can only enroll on one course at one time in the MOOC data used in this study.

A sequence $S = <(c_1, t_1), (c_2, t_2), \ldots, (c_n, t_n)>$ is called a *subsequence* of another sequence $S' = <(c'_1, t'_1), (c'_2, t'_2), \ldots, (c'_m, t'_m)>$ $(n \le m)$, and S' a *super-sequence* of S, denoted by $S \sqsubseteq S'$, if there exist integers $1 \le i_1 < \ldots < i_n \le m$ such that $S[1].c = S'[i_1].c$, $S[2].c = S'[i_2].c$, $\ldots$, $S[n].c = S'[i_n].c$. The ordered list of pairs $< S'[i_1], S'[i_2], \ldots, S'[i_n] >$ is called an *occurrence* of S in S', denoted by $Occ(S, S')$. If there exists at least one item $(c_j, t_j) \in S'$, and $(c_j, t_j) \notin S$, S is called a *proper subsequence* of S', or S' a *proper super-sequence* of S, denoted by $S \sqsubset S'$.

A *sequence database* ***SDB*** is a set of 2-tuples (sid, IS), where *sid* is called a *sequence-id* and *IS* an *input sequence*. A tuple (sid, IS) in a sequence database ***SDB*** is said to *contain* a sequence S if S is a subsequence of *IS*. For the MOOC data used in this paper, each sequence S has at most only one occurrence in one input sequence *IS*.

The number of tuples in a sequence database ***SDB*** containing sequence S is called the *support* of S, denoted by $sup(S)$. The set of input sequences in tuples of ***SDB*** containing sequence S is called the *support set* of S, denoted by $sup\_set(S)$.

Consider two input sequences IS_X and IS_Y containing S. It is easy to understand that the enrollment times in $Occ(S, IS_X)$ are not equal to the enrollment times in $Occ(S, IS_Y)$. Thus, the enrollment times are omitted in the mining results in this paper. Formally, $S = <c_1, c_2, \ldots, c_n>$ is called an *SP*, where $c_1, c_2, \ldots, c_n$ are time-ordered courses without specific enrollment times. $S.c_i (1 \le i \le n)$ denotes the ith course of S.

Let *min_sup* be the user-specified *minimum support threshold*. An SP S is an *FSP* in the sequence database ***SDB*** if $sup(S) \ge min\_sup$. The *frequent SPM* problem is to find the complete set of FSPs in ***SDB*** with respect to *min_sup*.

Consider the example sequence database in Table 2. To make the explanation simple and clear, the enrollment time of each item in all the input sequences is omitted. IS_1,

Table 2 Example sequence database

sid	Input sequence
IS_1	Data structure, Introduction to logic, Operating system, Linear algebra, Introduction to big data
IS_2	Linear algebra, Data structure, Operating system, Data mining
IS_3	Database, Principles of economics, Data mining
IS_4	Database, Data structure, Operating system
IS_5	Introduction to big data, Database, Data mining

IS_2, and IS_4 contain the SP $S = <$Data structure, Operating system$>$, and input sequences of these three tuples comprise $sup\_set(S)$. Thus, if the support threshold $min\_sup = 2$, $<$Data structure, Operating system$>$ is an FSP.

Different from traditional classroom teaching, learners on the same MOOC course may be significantly different in age, prerequisite knowledge, and learning objectives. For example, IS_3 is different from the other four input sequences in the example sequence database because IS_3 includes a non-computing course, whereas the other four sequences are all composed of computing courses. Furthermore, some learners may also enroll on many courses without a clear relationship. Thus, mining SPs directly in MOOC data may lead to an extremely large number of uninteresting patterns using substantial computational time and space.

Constraint-based mining may overcome the above-mentioned difficulties because constraints usually confine the patterns to be found to a particular subset that satisfies some strong conditions. Moreover, fewer resulting SPs also reduce the search space, thereby leading to an efficient mining process with small memory consumption. The challenge is how to push the constraints deep into the mining process rather than using constraints to filter the results after all SPs are discovered.

4 Flexible constraints

To determine the interestingness of SPs, three flexible constraints are considered from the perspective of the number of course enrollments within the input sequences, span of enrollment days, and specific enrollment time within a day. To improve efficiency, we push these constraints into the mining process by proving the downward closure property.

4.1 Length constraint

First, the lengths of the enrollment sequences were considered and their distribution seemed to be long tailed. Figure 1 shows the distribution of sequence lengths in the Course Recommendation dataset.

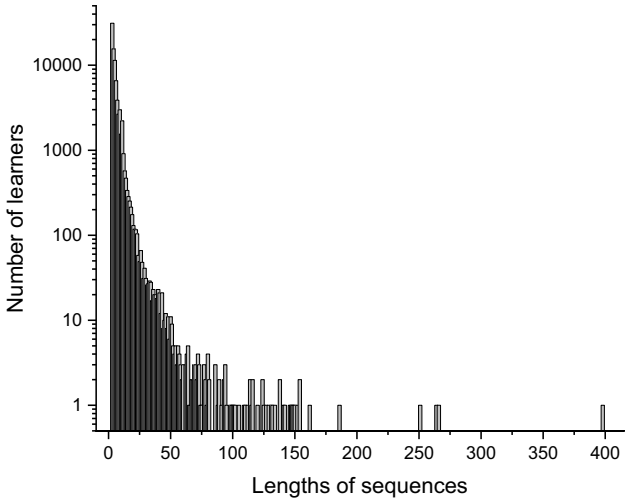

Fig. 1 Distribution of sequence lengths in the Course Recommendation dataset

Figure 1 shows that most sequence lengths are short. Specifically, 37.76% of the sequences have lengths equal to 3, 18.91% of the sequences have lengths equal to 4, 13.81% of the sequences have lengths equal to 5, and only five sequences are longer than 200. This phenomenon illustrates that school education is still the most important channel for people to acquire knowledge, although MOOCs are playing an increasingly important role in learning. Thus, most learners resort to MOOCs as an auxiliary learning method when they encounter problems that they need to solve using knowledge covered by online courses. Learners who have enrolled on multiple courses, or even hundreds of courses, may be platform testers or staff of relevant management departments.

This indicates that enrolling on a few courses is feasible for MOOC learners, whereas enrolling on a large number of courses occurs infrequently. Thus, the argument in this study is that the supports contributed by short sequences and long sequences are not the same, and the support contributed by long sequences is not as important as that contributed by short sequences. To model this fact, the length constraint is defined.

Definition 1 (Length constraint) Let ***SDB*** be the sequence database and S be an SP. The *length constraint* of S with respect to $IS \in sup\_set(S)$ is defined as

$$LC(S, IS) = \exp(-|IS|/max\_L), \tag{1}$$

where max_L is the maximum length of all input sequences in ***SDB***.

In this study, the length of the input sequence is divided by the maximum length of all input sequences to ensure that

the value of $|IS| / max\_L$ is in the range (0, 1], which prevents the decay of the constraint from being too large. To push the length constraint into the mining process, it is incorporated into the support.

Definition 2 (Support with length constraint) The *support with length constraint* (SLC) of an SP S is defined as

$$sup_L(S) = \sum_{IS \in sup\_set(S)} LC(S, IS) = \sum_{IS \in sup\_set(S)} \exp(-|IS|/max\_L) \quad (2)$$

The SLC in Definition 1 reflects that the support contribution decays as the length increases, and it is lower than the general support of the sequence stated in Section 3. The rationality is verified in Lemma 1.

Lemma 1 *Let **SDB** be the sequence database and S be an SP. Then,* $sup_L(S) \leq sup(S)$.

Proof. Suppose that m input sequences in ***SDB*** contain S; that is, there are m input sequences in $sup\_set(S)$, and $sup(S) = m$. For any $IS \in sup\_set(S)$, $|IS| \leq max\_L$. Thus, $0 < (|IS| / max\_L) \leq 1$. Hence, $0 < \exp(-|IS| / max\_L) \leq 1$; that is, $0 < LC(S, IS) \leq 1$. Because there are m input sequences in $sup\_set(S)$, $\sum_{IS \in sup\_set(S)} LC(S, IS) \leq m$; that is, $sup_L(S) \leq sup(S)$.□

The next lemma shows that the support with length constraint satisfies the downward closure property, which is an effective tool for reducing the search space, and is widely used in SPM.

Lemma 2 *For any two SPs S_X and S_Y, if $S_X \sqsubseteq S_Y$, $sup_L(S_Y) \leq sup_L(S_X)$.*

Proof. For $S_X \sqsubseteq S_Y$, $sup\_set(S_Y) \subseteq sup\_set(S_X)$. There are two cases:

(1) If $sup\_set(S_Y) = sup\_set(S_X)$, $sup_L(S_Y) = sup_L(S_X)$.
(2) If $sup\_set(S_Y) \subset sup\_set(S_X)$, input sequences are contained in $sup\_set(S_X)$ but not contained in $sup\_set(S_Y)$. Thus,

$$\begin{aligned} sup_L(S_Y) &= \sum_{IS \in sup\_set(S_Y) \wedge IS \in sup\_set(S_X)} LC(S_Y, IS) \\ &= \sum_{IS \in sup\_set(S_Y) \wedge IS \in sup\_set(S_X)} LC(S_X, IS) \\ &< \sum_{IS \in sup\_set(S_Y) \wedge IS \in sup\_set(S_X)} LC(S_X, IS) + \sum_{IS' \notin sup\_set(S_Y) \wedge IS' \in sup\_set(S_X)} LC(S_X, IS') \\ &= sup_L(S_X) \end{aligned}$$

According to the above discussion, $sup_L(S_Y) \leq sup_L(S_X)$.□

Lemma 2 shows that the length constraint can be pushed into the mining process to speed up the discovery of SPs.

4.2 Discreteness constraint

The discreteness constraint is also proposed, which describes how each enrollment time varies from the mean time in a sequence.

Consider an SP $S = <$Database, Data mining$>$ in the example sequence database shown in Table 2. Both IS_3 and IS_5 contain S. To explain the discreteness constraint, the specific enrollment date of each course of IS_3 and IS_5 is provided. Examples with enrollment dates are shown in Table 3.

Table 3 Two sequences with enrollment dates

sid	Input sequence
IS_3	(Database, 2017/2/24), (Principles of Economics, 2017/2/25), (Data mining, 2017/5/9)
IS_5	(Introduction to big data, 2017/2/14), (Database, 2017/2/18), (Data mining, 2017/2/26)

To engage learners in the MOOC platform, small discreteness among enrollment dates is preferred. From this point of view, for the same SP S, IS_5 contributes more to $sup(S)$ than IS_3. For IS_5, the mean date of two enrollment dates of S is 2017/2/22, and the distance between both enrollment dates and the mean date is 4 days. For IS_3, the mean date of the two enrollment dates of S is 2017/4/2, and the distance between both enrollment dates and the mean date is 37 days. To model this assumption, the discreteness constraint is defined.

Definition 3 (Discreteness constraint) Let $S = <c_1, c_2, \ldots, c_n>$ be an SP. $IS \in sup\_set(S)$ is an input sequence, and there exist integers $1 \leq i_1 < i_2 < \ldots < i_n \leq m$ such that $S.c_1 = IS[i_1].c$, $S.c_2 = IS[i_2].c$, …, $S.c_n = IS[i_n].c$. The discreteness constraint of S with respect to IS is defined as

$$DC(S, IS) = \exp\left(-\sum_{j=1}^{n}\left(IS[i_j].t - \overline{IS[i_n].t}\right)^2\right), \quad (3)$$

where

$$\overline{IS[i_n].t} = \frac{1}{n}\sum_{j=1}^{n} IS[i_j].t. \tag{4}$$

From Definition 3, the discreteness constraint indicates how widely enrollment times in a sequence's occurrence vary. If enrollment times vary greatly from the mean time of a sequence's occurrence, the constraint is small. To simplify the calculation, only the enrollment dates are considered and the specific enrollment moments are omitted when computing the discreteness constraints.

To push the discreteness constraint into the mining process, it is incorporated into the support.

Definition 4 (Support with discreteness constraint) Let $S = <c_1, c_2, \ldots, c_n>$ be an SP. The *support with discreteness constraint* (SDC) of S is defined as

$$sup_D(S) = \sum_{IS\in sup\_set(S)} DC(S, IS) = \sum_{IS\in sup\_set(S)} \exp\left(-\sum_{j=1}^{n}\left(IS[i_j].t - \overline{IS[i_n].t}\right)^2\right). \tag{5}$$

The function exp(-x) is monotone decreasing. To avoid $sup_D(S)$ becoming too small, min–max normalization is used to rescale the enrollment time into the range [0, 1] before the discreteness constraint and SDC are calculated. It can also be proved that the SDC is lower than the general support stated in Section 3.

Lemma 3 *Let **SDB** be the sequence database and S be an SP. Then,* $sup_D(S) \le sup(S)$.

Proof. Suppose that m input sequences in ***SDB*** contain S; that is, there are m input sequences in $sup\_set(S)$ and $sup(S) = m$. For any $IS \in sup\_set(S)$, $\sum_{j=1}^{n}\left(IS[i_j].t - \overline{IS[i_n].t}\right)^2 \ge 0$. Thus, $0 < \exp\left(-\sum_{j=1}^{n}\left(IS[i_j].t - \overline{IS[i_n].t}\right)^2\right) \le 1$; that is, $0 < DC(S, IS) \le 1$. Because there are m input sequences in $sup\_set(S)$, $\sum_{IS\in sup\_set(S)} DC(S, IS) \le m$; that is, $sup_D(S) \le sup(S)$. □

The SDC also satisfies the downward closure property, which is proved in Lemma 4.

Lemma 4 *For any two SPs, S_X and S_Y, if $S_X \sqsubseteq S_Y$, $sup_D(S_Y) \le sup_D(S_X)$.*

Proof. For $S_X \sqsubseteq S_Y$, there are two cases.

(1) If $S_X = S_Y$, $sup_D(S_Y) = sup_D(S_X)$.
(2) If $S_X \sqsubset S_Y$, first consider the case in which $|S_Y| = |S_X| + 1$. Let $S_X = <c_1, c_2, \ldots, c_n>$ and $S_Y = <c_1, c_2, \ldots, c_n, c_{n+1}>$. For an input sequence IS containing both S_X and S_Y, there exist integers $i_1 < \ldots < i_n < i_{n+1}$ such that $IS[i_1].c = c_1, \ldots, IS[i_n].c = c_n, IS[i_{n+1}].c = c_{n+1}$. Then $DC(S_Y, IS) = \exp\left(-\sum_{j=1}^{n+1}\left(IS[i_j].t - \overline{IS[i_{n+1}].t}\right)^2\right)$ and $DC(S_X, IS) = \exp\left(-\sum_{j=1}^{n}\left(IS[i_j].t - \overline{IS[i_n].t}\right)^2\right)$. According to Eq. (4),

$$\begin{aligned}\overline{IS[i_{n+1}].t} &= \frac{1}{n+1}\left(\sum_{j=1}^{n} IS[i_j].t + IS[i_{n+1}].t\right) = \frac{n}{n+1}\frac{1}{n}\sum_{j=1}^{n} IS[i_j].t + \frac{1}{n+1}IS[i_{n+1}].t \\ &= \frac{n}{n+1}\overline{IS[i_n].t} + \frac{1}{n+1}IS[i_{n+1}].t = \frac{n+1-1}{n+1}\overline{IS[i_n].t} + \frac{1}{n+1}IS[i_{n+1}].t \\ &= \overline{IS[i_n].t} + \frac{1}{n+1}\left(IS[i_{n+1}].t - \overline{IS[i_n].t}\right).\end{aligned} \tag{6}$$

By substitution with Eq. (6),

$$\begin{aligned}&\sum_{j=1}^{n+1}(IS[i_j].t - \overline{IS[i_{n+1}].t})^2 = \sum_{j=1}^{n+1}(IS[i_j].t - \overline{IS[i_n].t} - \frac{1}{n+1}(IS[i_{n+1}].t - \overline{IS[i_n].t}))^2 \\ &= \sum_{j=1}^{n+1}((IS[i_j].t - \overline{IS[i_n].t})^2 - \frac{2(IS[i_j].t - \overline{IS[i_n].t})(IS[i_{n+1}].t - \overline{IS[i_n].t})}{n+1} + \frac{(IS[i_{n+1}].t - \overline{IS[i_n].t})^2}{(n+1)^2}) \\ &= \sum_{j=1}^{n+1}(IS[i_j].t - \overline{IS[i_n].t})^2 - \frac{2(IS[i_{n+1}].t - \overline{IS[i_n].t})}{n+1}\sum_{j=1}^{n+1}(IS[i_j].t - \overline{IS[i_n].t}) + \frac{(IS[i_{n+1}].t - \overline{IS[i_n].t})^2}{n+1}.\end{aligned} \tag{7}$$

Table 4 Two input sequences with specific enrollment times

sid	Input sequence
IS_3	(Database, 2017/2/24 9:52:00), (Principles of Economics, 2017/2/25 10:19:00), (Data mining, 2017/5/9 8:22:00)
IS_5	(Introduction to big data, 2017/2/14 8:21:00), (Database, 2017/2/18 3:46:00), (Data mining, 2017/2/26 0:54:00)

For the first term on the right-hand side of the last expression in Eq. (7),

$$\sum_{j=1}^{n+1}\left(IS[i_j].t-\overline{IS[i_n].t}\right)^2=\sum_{j=1}^{n}\left(IS[i_j].t-\overline{IS[i_n].t}\right)^2+\left(IS[i_{n+1}].t-\overline{IS[i_n].t}\right)^2 \tag{8}$$

For the second term on the right-hand side of the last expression in Eq. (7),

$$\begin{aligned}&-\frac{2\left(IS[i_{n+1}].t-\overline{IS[i_n].t}\right)}{n+1}\sum_{j=1}^{n+1}\left(IS[i_j].t-\overline{IS[i_n].t}\right)\\ =&-\frac{2(IS[i_{n+1}].t-\overline{IS[i_n].t})}{n+1}\left(\sum_{j=1}^{n}\left(IS[i_j].t-\overline{IS[i_n].t}\right)+\left(IS[i_{n+1}].t-\overline{IS[i_n].t}\right)\right)\end{aligned} \tag{9}$$

Because $\sum_{j=1}^{n}\left(IS[i_j].t-\overline{IS[i_n].t}\right)=\sum_{j=1}^{n}IS[i_j].t-n\times\overline{IS[i_n].t}=0$,

$$-\frac{2\left(IS[i_{n+1}].t-\overline{IS[i_n].t}\right)}{n+1}\sum_{j=1}^{n+1}\left(IS[i_j].t-\overline{IS[i_n].t}\right)=-\frac{2\left(IS[i_{n+1}].t-\overline{IS[i_n].t}\right)^2}{n+1} \tag{10}$$

Substituting Eqs. (8) and (10) into Eq. (7) yields

$$\begin{aligned}&\sum_{j=1}^{n+1}\left(IS[i_j].t-\overline{IS[i_{n+1}].t}\right)^2\\ =&\sum_{j=1}^{n}\left(IS[i_j].t-\overline{IS[i_n].t}\right)^2+\frac{n}{n+1}\left(IS[i_{n+1}].t-\overline{IS[i_n].t}\right)^2.\end{aligned} \tag{11}$$

Hence, $\sum_{j=1}^{n+1}\left(IS[i_j].t-\overline{IS[i_{n+1}].t}\right)^2\geq\sum_{j=1}^{n}\left(IS[i_j].t-\overline{IS[i_n].t}\right)^2$. Thus, $\exp\left(-\sum_{j=1}^{n+1}\left(IS[i_j].t-\overline{IS[i_{n+1}].t}\right)^2\right)\leq\exp\left(-\sum_{j=1}^{n}\left(IS[i_j].t-\overline{IS[i_n].t}\right)^2\right)$; that is, $DC(S_Y, IS)\leq DC(S_X, IS)$. Thus,

$$\sum_{IS\in sup\_set(S_Y)\wedge IS\in sup\_set(S_X)}DC(S_Y,IS)\leq\sum_{IS\in sup\_set(S_Y)\wedge IS\in sup\_set(S_X)}DC(S_X,IS) \tag{12}$$

Because $S_X\sqsubset S_Y$, $sup\_set(S_Y)\subseteq sup\_set(S_X)$ holds. If $sup\_set(S_Y)=sup\_set(S_X)$, Eq. (12) implies that $sup_D(S_Y)\leq sup_D(S_X)$. If $sup\_set(S_Y)\subset sup\_set(S_X)$, input sequences are contained in $sup\_set(S_X)$ and not contained in $sup\_set(S_Y)$. Thus,

$$\begin{aligned}sup_D(S_Y)&=\sum_{IS\in sup\_set(S_Y)\wedge IS\in sup\_set(S_X)}DC(S_Y,IS)\\&\leq\sum_{IS\in sup\_set(S_Y)\wedge IS\in sup\_set(S_X)}DC(S_X,IS)\\&<\sum_{IS\in sup\_set(S_Y)\wedge IS\in sup\_set(S_X)}DC(S_X,IS)+\sum_{IS'\notin sup\_set(S_Y)\wedge IS'\in sup\_set(S_X)}DC(S_X,IS')\\&=sup_D(S_X)\end{aligned}$$

According to the above discussion, $sup_D(S_Y)\leq sup_D(S_X)$ when $|S_Y|=|S_X|+1$.

When $|S_Y|=|S_X|+m$ $(m>1)$, $(m-1)$ SPs $S_1, S_2,\ldots, S_{m-1}$ can be identified such that $S_X\sqsubset S_1\sqsubset S_2\sqsubset\ldots\sqsubset S_{m-2}\sqsubset S_{m-1}\sqsubset S_Y$ and $|S_Y|=|S_{m-1}|+1=|S_{m-2}|+2=\ldots=|S_1|+m-1=|S_X|+m$. Similar to the case in which $|S_Y|=|S_X|+1$, $sup_D(S_Y)\leq sup_D(S_{m-1})\leq sup_D(S_{m-2})\leq\ldots\leq sup_D(S_1)\leq sup_D(S_X)$.

According to the above discussion, $sup_D(S_Y)\leq sup_D(S_X)$ when $S_X\sqsubseteq S_Y$.□

Lemma 4 shows that the discreteness constraint can also be pushed into the mining process to speed up the discovery of SPs.

4.3 Validity constraint

The validity constraint is also proposed, which distinguishes serious learning from casual learning enrollments. The object of this constraint is still the enrollment time, that is, the specific moment within a day.

Consider IS_3 and IS_5 in the example sequence database in Table 2. The specific enrollment moment is shown in Table 4. It should be noted that the format of Table 4 is the same as the original format of the Course Recommendation dataset. To simplify the explanation, some information was omitted in the previous examples.

The motivation for defining the validity constraint is that enrollments during normal working hours are often generated by learners who have a strong desire to learn, whereas enrollments during non-working hours are often generated by learners who simply want to gain some basic knowledge. For IS_3 and IS_5 in Table 4, although both contain the SP $S=<$Database, Data mining$>$, the enrollment moments for IS_3 are during working hours, whereas the enrollment moments for IS_5 are during non-working hours

(early morning and midnight). It is assumed that IS_3 contributes more to $sup(S)$ than IS_5. To model this assumption, the validity constraint is defined. In this paper, enrollment during normal working hours is called *valid enrollment* and enrollment during non-working hours is called *casual enrollment*. For example, if normal working hours are set to the period 8:00–22:59 and non-working hours to the period 23:00–7:59, S has two valid enrollments in IS_3 and two casual enrollments in IS_5.

Definition 5 (Validity constraint) Let $S = <c_1, c_2, \ldots, c_n>$ be an SP. Suppose that $IS \in sup\_set(S)$ is an input sequence. The validity constraint of S with respect to IS is defined as

$$VC(S, IS) = \exp(-num\_l/max\_L), \tag{13}$$

where num_l is the number of casual enrollments of S in IS and max_L is the maximum length of all input sequences in ***SDB***.

From Definition 5, the validity constraint distinguishes between standard learning behavior and casual learning behavior. For $S = <$Database, Data mining$>$, $VC(S, IS_3) = 1$, which indicates that $sup(S)$ does not decay in IS_3 with respect to enrollment moments because both enrollments are valid enrollments.

To push the validity constraint into the mining process, it is incorporated into the support.

Definition 6 (Support with validity constraint) Let S be an SP. The *support with validity constraint* (SVC) of S is defined as

$$sup_V(S) = \sum_{IS \in sup\_set(S)} VC(S, IS) = \sum_{IS \in sup\_set(S)} \exp(-num\_l/max\_L). \tag{14}$$

It can also be proved that the SVC is lower than the general support stated in Section 3.

Lemma 5 *Let* ***SDB*** *be the sequence database and S be an SP. Then*, $sup_V(S) \le sup(S)$.

Proof. Suppose that m input sequences in ***SDB*** contain S; that is, there are m input sequences in $sup\_set(S)$ and $sup(S) = m$. For any $IS \in sup\_set(S)$, $num\_l(S, IS) / max\_L \ge 0$. Thus, $0 < \exp(-num\_l(S, IS) / max\_L) \le 1$; that is, $0 < VC(S, IS) \le 1$. Because there are m input sequences in $sup\_set(S)$, $\sum_{IS \in sup\_set(S)} VC(S, IS) \le m$; that is, $sup_V(S) \le sup(S)$. □

The SVC also satisfies the downward closure property, which is proved in Lemma 6.

Lemma 6 *For any two SPs* S_X *and* S_Y, *if* $S_X \sqsubseteq S_Y$, $sup_V(S_Y) \le sup_V(S_X)$.

Proof. For $S_X \sqsubseteq S_Y$, there are two cases.

(1) If $S_X = S_Y$, $sup_V(S_Y) = sup_V(S_X)$.
(2) If $S_X \sqsubset S_Y$, for an input sequence IS containing both S_X and S_Y, $num\_l(S_X, IS)/max\_L \le num\_l(S_Y, IS)/max\_L$. Thus, $\exp(-num\_l(S_X, IS)/max\_L) \ge \exp(-num\_l(S_Y, IS)/max\_L)$; that is,

$$VC(S_X, IS) \ge VC(S_Y, IS). \tag{15}$$

Because $S_X \sqsubset S_Y$, $sup\_set(S_Y) \subseteq sup\_set(S_X)$ holds. If $sup\_set(S_Y) = sup\_set(S_X)$,

$$\begin{aligned} sup_V(S_Y) &= \sum_{IS \in sup\_set(S_Y)} VC(S_Y, IS) = \sum_{IS \in sup\_set(S_X)} VC(S_Y, IS) \\ &\le \sum_{IS \in sup\_set(S_X)} VC(S_X, IS) \\ &= sup_V(S_X). \end{aligned}$$

If $sup\_set(S_Y) \subset sup\_set(S_X)$, input sequences are contained in $sup\_set(S_X)$ and not contained in $sup\_set(S_Y)$. Thus,

$$\begin{aligned} sup_V(S_Y) &= \sum_{IS \in sup\_set(S_Y) \wedge IS \in sup\_set(S_X)} VC(S_Y, IS) \\ &\le \sum_{IS \in sup\_set(S_Y) \wedge IS \in sup\_set(S_X)} VC(S_X, IS) \\ &< \sum_{IS \in sup\_set(S_Y) \wedge IS \in sup\_set(S_X)} VC(S_X, IS) + \sum_{IS' \notin sup\_set(S_Y) \wedge IS' \in sup\_set(S_X)} VC(S_X, IS') \\ &= sup_V(S_X) \end{aligned}$$

According to the above discussion, $sup_V(S_Y) \le sup_V(S_X)$ when $S_X \sqsubseteq S_Y$.□

Lemma 6 shows that the validity constraint can be pushed into the mining process to speed up the discovery of SPs.

4.4 Constraint integration

To speed up the SPM process, the length constraint, discreteness constraint, and validity constraint are integrated flexibly into one constraint, and the general support is replaced.

Definition 7 (SFC) Let S be an SP. The SFC of S is defined as

$$sup_{FC}(S) = \alpha \times sup_L(S) + \beta \times sup_D(S) + \gamma \times sup_V(S) \quad (16)$$

where α ($0 \le \alpha \le 1$) is the *length factor*, β ($0 \le \beta \le 1$) is the *discreteness factor*, and γ ($0 \le \gamma \le 1$) is the *validity factor* such that

$$\alpha + \beta + \gamma = 1 \quad (17)$$

For an SP S, $sup_{FC}(S)$ reflects the decay of $sup(S)$ affected by the input sequences in $sup\_set(S)$, including the lengths of these input sequences, variances of the enrollment dates in these input sequences, and enrollment moments within a day in these input sequences. If the lengths of input sequences in $sup\_set(S)$ are short, the variances of the enrollment dates are small, and there are few casual enrollments, then there will be more opportunities to discover S when using the proposed algorithms.

It also can be proved that the SFC is lower than the general support.

Theorem 1 *Let **SDB** be the sequence database and* $S = <c_1, c_2, \ldots, c_n>$ *be an SP. Then,* $sup_{FC}(S) \le sup(S)$.

Proof. Let IS be an input sequence and $IS \in sup\_set(S)$. There exist integers $1 \le i_1 < \ldots < i_n$ such that $S.c_1 = IS[i_1].c$, $S.c_2 = IS[i_2].c$, …, $S.c_n = IS[i_n].c$. According to Lemma 1,

$$0 < LC(S, IS) = \exp(-|IS|/max\_L) \le 1. \quad (18)$$

Similarly, according to Lemmas 3 and 5,

$$0 < DC(S, IS) = \exp\left(-\sum_{j=1}^{n}\left(IS[i_j].t - \overline{IS[i_n].t}\right)^2\right) \le 1, \quad (19)$$

$$0 < VC(S, IS) = \exp(-num\_l(S, IS)/max\_L) \le 1. \quad (20)$$

Assume $LC(S, IS) \ge DC(S, IS)$ and $LC(S, IS) \ge VC(S, IS)$. Then,

$$\begin{aligned} &\alpha \times LC(S, IS) + \beta \times DC(S, IS) + \gamma \times VC(S, IS) \\ &\le \alpha \times LC(S, IS) + \beta \times LC(S, IS) + \gamma \times LC(S, IS) \\ &= (\alpha + \beta + \gamma) \times LC(S, IS). \end{aligned}$$

According to Eq. (17),

$$\alpha \times LC(S, IS) + \beta \times DC(S, IS) + \gamma \times VC(S, IS) \le 1. \quad (21)$$

For the other two cases, (1) $DC(S, IS) \ge LC(S, IS)$ and $DC(S, IS) \ge VC(S, IS)$ and (2) $VC(S, IS) \ge LC(S, IS)$ and $VC(S, IS) \ge DC(S, IS)$, it can be concluded that Eq. (21) holds similarly.

Suppose that m input sequences in ***SDB*** contain S; that is, there are m input sequences in $sup\_set(S)$, and $sup(S) = m$. According to Eq. (21),

$$\begin{aligned} sup_{FC}(S) &= \alpha \times sup_L(S) + \beta \times sup_D(S) + \gamma \times sup_V(S) \\ &= \textstyle\sum_{IS \in sup\_set(S)} (\alpha \times LC(S, IS) + \beta \times DC(S, IS) + \gamma \times VC(S, IS)) \\ &\le m = sup(S). \end{aligned}$$

According to the above discussion, $sup_{FC}(S) \le sup(S)$. □

Using SFC to replace the support can guarantee mining efficiency because it also satisfies the downward closure property.

Theorem 2 *For any two SPs* S_X *and* S_Y, *if* $S_X \sqsubseteq S_Y$, $sup_{FC}(S_Y) \le sup_{FC}(S_X)$.

Proof. According to Lemma 2, $sup_L(S_Y) \le sup_L(S_X)$. Because $0 \le \alpha \le 1$,

$$\alpha \times sup_L(S_Y) \le \alpha \times sup_L(S_X) \quad (22)$$

Similarly, according to Lemmas 4 and 6,

$$\beta \times sup_D(S_Y) \le \beta \times sup_D(S_X), \quad (23)$$

$$\gamma \times sup_V(S_Y) \le \gamma \times sup_V(S_X). \quad (24)$$

Summing Eqs. (22), (23), and (24) yields

$$\begin{aligned} sup_{FC}(S_Y) &= \alpha \times sup_L(S_Y) + \beta \times sup_D(S_Y) + \gamma \times sup_V(S_Y) \\ &\le \alpha \times sup_L(S_X) + \beta \times sup_D(S_X) + \gamma \times sup_V(S_X) \\ &= sup_{FC}(S_X) \end{aligned}$$

According to the above discussion, $sup_{FC}(S_Y) \le sup_{FC}(S_X)$ if $S_X \sqsubseteq S_Y$.□

Using Theorem 2, when an SP's SFC is found to be lower than the minimum support threshold, all its super patterns can be safely pruned when using the proposed algorithms.

Given the above discussion, the problem to be solved is redefined as follows: Given a positive integer *min_sup* as the *minimum support threshold*, an SP S is a *flexible-constraint-based SP* (FCSP) in the sequence database ***SDB*** if $sup_{FC}(S) \ge min\_sup$. An FCSP with length l is called an l-FCSP. The *flexible-constraint-based SPM* (FCSPM) problem is to find the complete set of FCSPs with respect to ***SDB*** and *min_sup*.

Theorem 3 *Let S_FCSP and S_FSP be the sets of FCSPs and FSPs with respect to the same min_sup, respectively. Then,* $S\_FCSP \subseteq S\_FSP$.

Proof. For $\forall\, S \in S\_FCSP$, $sup_{FC}(S) \ge min\_sup$. According to Theorem 1, $sup_{FC}(S) \le sup(S)$. Hence, $sup(S) \ge min\_sup$, and $S \in S\_FSP$. Thus, $S\_FCSP \subseteq S\_FSP$.□

From Theorem 3, the set of FCSPs is a subset of the set of FSPs when the same threshold is set.

5 Algorithm description

To discover FCSPs, two algorithms are proposed. One traverses the search space level-by-level and is called SPM using flexible constraints level-wisely (SPM-FC-L), and the other traverses the search space using recursive projections and is called SPM using flexible constraints by projection (SPM-FC-P). The SPM-FC-L algorithm is convenient to implement, whereas SPM-FC-P is more efficient. Because it was proved in Section 4.4 that SFC satisfies the downward closure property, as does the support, the support is replaced by SFC in both algorithms directly.

5.1 SPM-FC-L algorithm

To replace the support with SFC, it is natural to discover the FCSPs based on the GSP algorithm [33]. Algorithm 1 describes the proposed SPM-FC-L for mining FCSPs.

Algorithm 1	SPM-FC-L
Input:	Sequence database ***SDB***, minimum support threshold *min_sup*
Output:	All FCSPs
1	$FS_1 \leftarrow$ all 1-FCSPs;
2	$k = 1$;
3	**while** ($FS_k \neq \varnothing$) **do**
4	CS_{k+1} = candidate_gen(FS_k);
5	$FS_{k+1} = \{S \mid S \in CS_{k+1}, sup_{FC}(S) \geq min\_sup\}$;
6	k++;
7	**end while**
8	return $\cup_k FS_k$.

In Algorithm 1, FCSPs with single items are first discovered on Line 1. FS_k is used to denote the set of FCSPs with length k. The initial value of k is set to one on Line 2. The main loop discovers all FCSPs using a candidate generation-and-test methodology (Lines 3–7). On Line 4, the function candidate_gen (described in Algorithm 2) is called to generate candidates with length $(k+1)$. CS_k is used to denote the set of candidate FCSPs with length k. On Line 5, only candidates with SFC no lower than *min_sup* are kept. The number of iterations is incremented by one on Line 6. Finally, on Line 8, all the discovered FCSPs are returned.

Algorithm 2	Function candidate_gen(FS_k)
1	**for** each sequence $S_m \in FS_k$ **do**
2	**for** each sequence $S_n \in FS_k$ **do**
3	**if** ($S_m.c_1 == S_n.c_1 \wedge S_m.c_2 == S_n.c_2 \wedge \ldots \wedge S_m.c_{k-1} == S_n.c_{k-1} \wedge S_m.c_k \neq S_n.c_k$) **then**
4	$C_m = < S_m.c_1, S_m.c_2, \ldots, S_m.c_{k-1}, S_m.c_k, S_n.c_k >$;
5	**if** all subsets of C_m with length k are in FS_k **then**
6	$CS_{k+1} \leftarrow C_m$;
7	**end if**
8	$C_n = < S_m.c_1, S_m.c_2, \ldots, S_m.c_{k-1}, S_n.c_k, S_m.c_k >$;
9	**if** all subsets of C_n with length k are in FS_k **then**
10	$CS_{k+1} \leftarrow C_n$;
11	**end if**
12	**end if**
13	**end for**
14	**end for**
15	**return** CS_{k+1}.

The function candidate_gen generates the candidate FCSPs with length $(k+1)$ by joining two k-FCSPs that share the first $(k\text{-}1)$ common courses. For each such pair of FCSPs, two candidates can be generated. Each candidate is not retained until all its subsequences are FCSPs because of the downward closure property of SFC. Different from typical SPM algorithms that use both itemset-extension and sequence-extension to generate new candidates, only sequence-extension is considered. This is because there is only one course enrollment at one time in the Course Recommendation dataset used in this paper.

5.2 SPM-FC-P algorithm

In this section, another FCSP mining algorithm, SPM-FC-P, is proposed that uses the recursive sequence database projection approach. To explain the algorithm, the following concepts of sequence database projection are introduced.

Let $S_X = < c_1, c_2, \ldots, c_n >$ and $S_Y = < c_1, c_2, \ldots, c_m >$ be two SPs. S_Y is called a *prefix* of S_X if (1) $m < n$ and (2) there exist integers $1 \leq i_1 < i_2 < \ldots < i_m < n$ such that $S_{Y.C_1} = S_{X.C_{i_1}}$, $S_{Y.C_2} = S_{X.C_{i_2}}, \ldots, S_{Y.C_m} = S_{X.C_{i_m}}$. $S_Z = < c_{i_m+1}, c_{i_m+2}, \ldots, c_n >$ is called the *suffix* of S_X with respect to prefix S_Y, and denoted by $S_Z = S_X / S_Y$.

Table 5 S-projected database in the example sequence database

sid	Input sequence
IS'_1	Linear algebra, Introduction to big data
IS'_2	Data mining

Although $S = <$Data structure, Operating system$>$ is contained by IS_1, IS_2, and IS_4, the S-projected database is only composed of two suffixes because $IS_4 / S = \varnothing$

For example, SP $S_Y = <$Data structure, Operating system$>$ is a prefix of $S_X = <$Data structure, Introduction to logic, Operating system, Linear algebra, Introduction to big data$>$, and $S_Z = <$Linear algebra, Introduction to big data$>$ is a suffix of S_X with respect to S_Y.

Let S be an SP in a sequence database $\boldsymbol{SDB}$. The *S-projected database*, denoted by $\boldsymbol{SDB}|_S$, is the collection of suffixes of input sequences in $\boldsymbol{SDB}$ with respect to prefix S.

The sequence database in Table 2 is considered as an example. Consider $S = <$Data structure, Operating system$>$. The S-projected database is shown in Table 5.

According to the above concepts, Algorithm 3 describes the proposed SPM-FC-P for mining FCSPs.

Algorithm 3	SPM-FC-P
Input:	Sequential pattern S, S-projected database $\boldsymbol{SDB} \mid_S$
Output:	All FCSPs
1	Find all 1-FCSPs in $\boldsymbol{SDB} \mid_S$ and denote the set of them by FS_1;
2	**for** each $it \in FS_1$ **do**
3	Append it after the last item of S to form a new FCSP S';
4	Output S';
5	Construct S'-projected database $\boldsymbol{SDB} \mid_{S'}$;
6	SPM-FC-P(S', $\boldsymbol{SDB} \mid_{S'}$);
7	**end for**

In the S-projected database, all 1-FCSPs are enumerated on Line 1. Then the main loop (Lines 2–7) generates new FCSPs by appending each 1-FCSP to the current FCSP. On Line 3, a 1-FCSP is appended after the last item of the current FCSP to form a new FCSP. According to previous SPM algorithms based on pattern growth [27], it is easy to understand that the SFC of the new SP is the same as the SFC of the appended item. Thus, it is also an FCSP. Then, the newly formed FCSP is output on Line 4 and its projected database is constructed on Line 5. On Line 6, the SPM-FC-P procedure is called to generate FCSPs recursively. It should be noted that, when SPM-FC-P is called the first time, S is an empty set and $\boldsymbol{SDB} \mid_S$ is $\boldsymbol{SDB}$ itself.

5.3 Summary of the proposed algorithms

Discovering SPs from MOOC learning data is important for improving the online learning experience. To the best of the authors' knowledge, this is the first work on extracting constraint-based SPs from MOOC data. The novelty of the two proposed algorithms can be summarized as follows.

First, the interestingness of the resulting SPs is measured from three perspectives: the number of courses in which students were enrolled, date span of course enrollment, and specific enrollment moment in a day. Thus, the problem of the extremely large number of resulting SPs of a typical FSP mining problem can be solved, to great extent. Additionally, the FCSPs are more meaningful than FSPs that use frequency only.

Second, the downward closure property was also proved to be satisfied for FCSPs. Thus, the two algorithms for mining FCSPs are not only easy to implement but also effective in reducing the extremely large search space. Therefore, the efficiency of both SPM-FC-L and SPM-FC-P is comparable with that of counter level-wise and projection-based SPM algorithms.

Finally, the three constraints, that is, length constraint, discreteness constraint, and validity constraint, were also all proved to satisfy the downward closure property. Hence, these three constraints can be used separately according to the specific application scenario, which makes the two proposed algorithms suitable for general usage.

6 Experimental results

In this section, the performance of the proposed algorithms is evaluated and they are compared with two general SPM algorithms: GSP [33] and PrefixSpan [27], and one constraint-based sequential rule mining algorithm, TRuleGrowth [11]. The source code of each algorithm was downloaded from the SPMF data mining library [9]. To run GSP, PrefixSpan, and TRuleGrowth on the Course Recommendation dataset, the dataset was transformed by deleting the specific enrollment time and retaining the order of course enrollment within each sequence. It should be noted that TRuleGrowth is an algorithm with a sliding-window constraint for mining partially ordered sequential rules. For a fair comparison, when TRuleGrowth was run, the part that calculated confidence was blocked. Thus, in Sects. 6.1 and 6.2, only the time and memory required for TRuleGrowth to mine the SPs is recorded, and the time and memory required for TRuleGrowth to generate rules from the SPs is ignored. Similarly, the number of results for TRuleGrowth is also the number of discovered SPs rather than the number of sequential rules.

The experiments were conducted on a computer with a 2-Core 1.80 GHz CPU and 8 GB memory running 64-bit macOS Mojave (macOS 10.14). The programs were written in Java. It should be noted that the support used for evaluation was the ratio of the number of input sequences containing the target pattern to the total number of input sequences

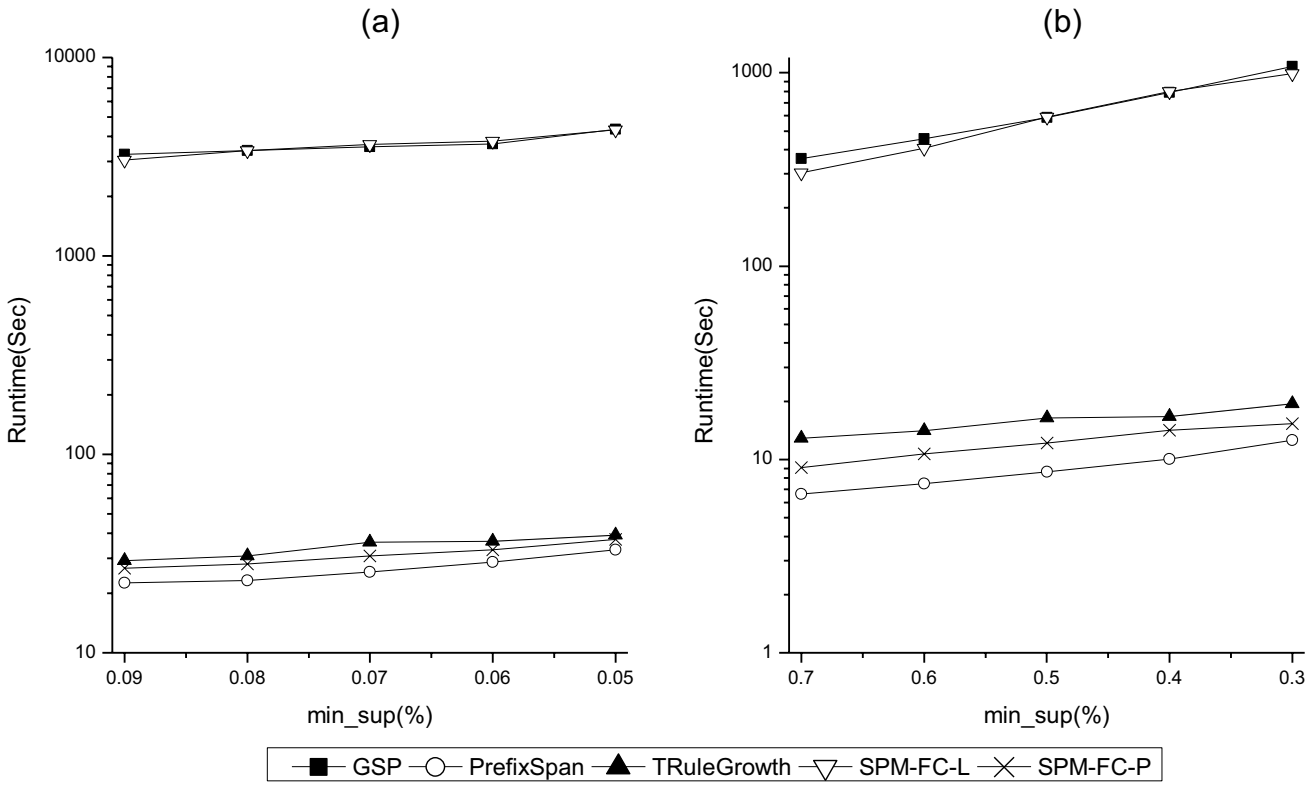

Fig. 2 Comparison of execution times

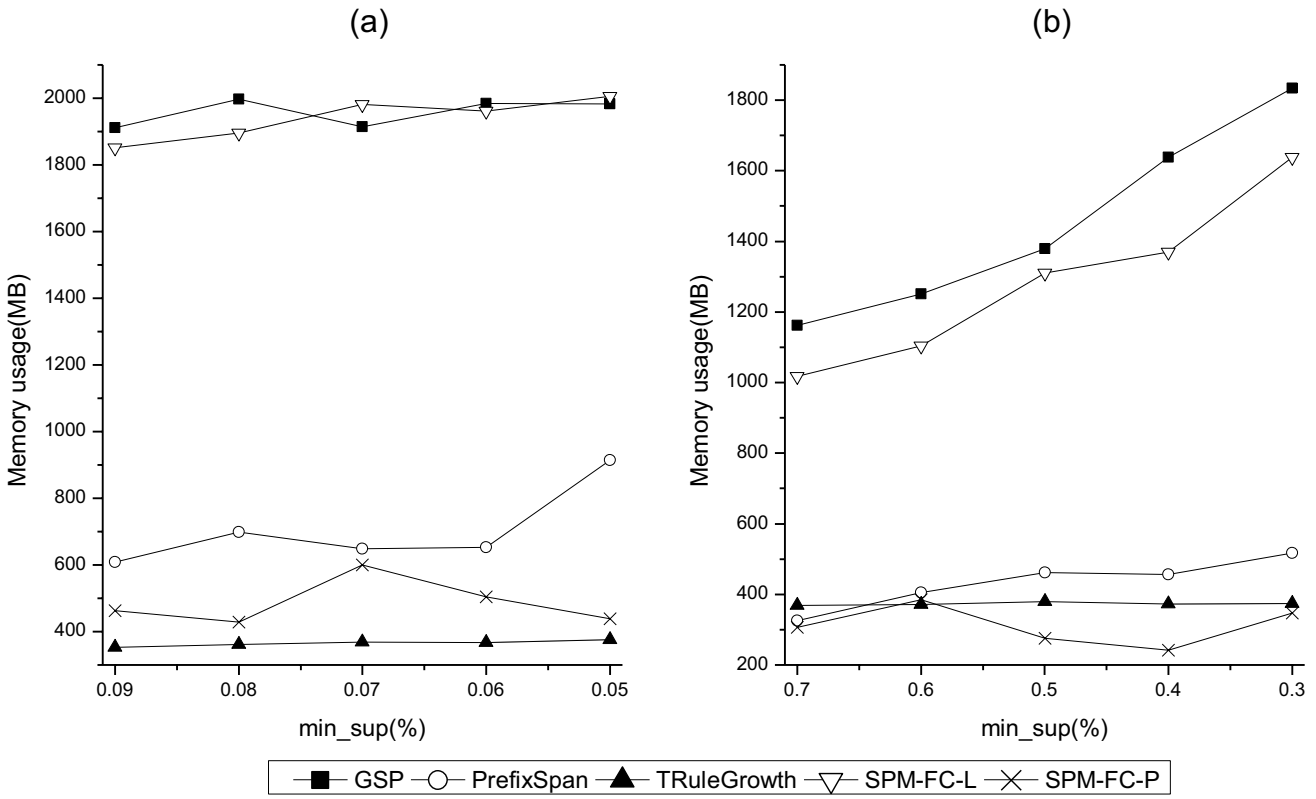

Fig. 3 Comparison of memory usage

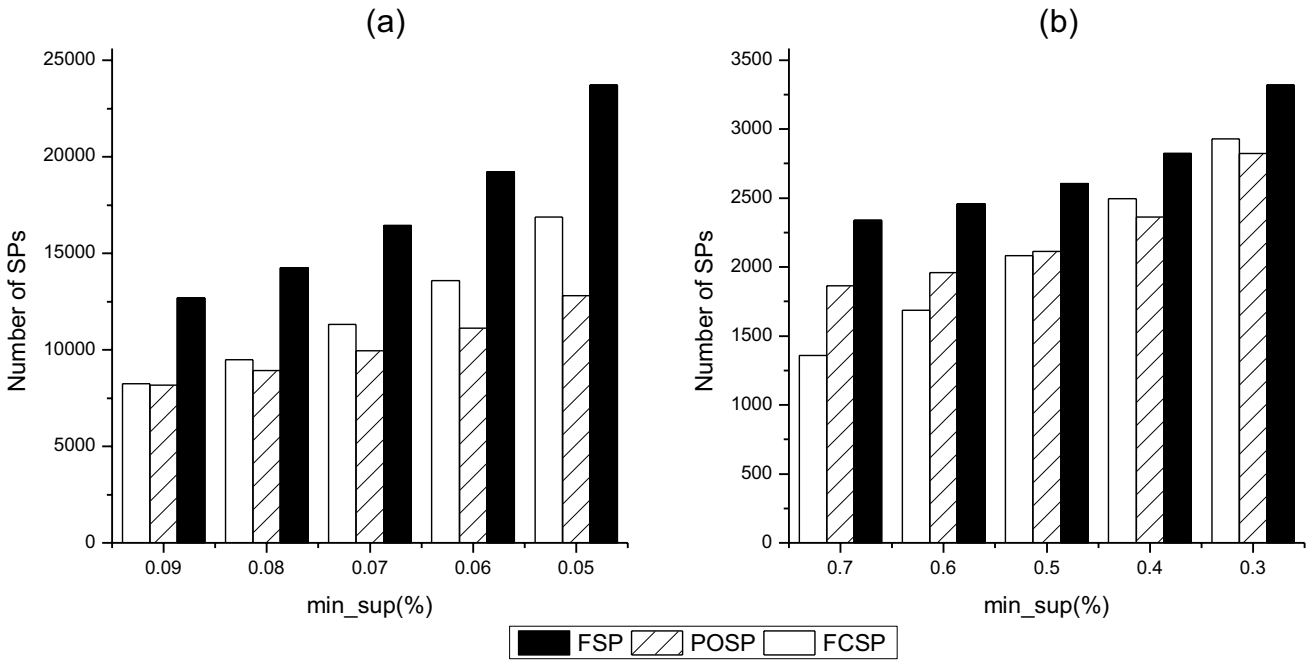

Fig. 4 Number of discovered patterns

in the sequence database; that is, the support values used in experiments were in the range [0, 1].

In the proposed model, the length factor α ($0 \leq \alpha \leq 1$), discreteness factor β ($0 \leq \beta \leq 1$), and validity factor γ ($0 \leq \gamma \leq 1$) had to be set to appropriate values. First, the approximate ranges of these parameters were outlined, and then their optimal values were determined using progressive refinement. For all the experiments, $\alpha = 1/6$, $\beta = 3/6$, and $\gamma = 2/6$.

6.1 Runtime

First, the efficiency performance of these algorithms was demonstrated. When measuring the runtime, the minimum support threshold was varied. Because there was only one dataset, the same dataset was tested using two groups of minimum support thresholds in the experiments in Sects. 6.1 to 6.3.

In Fig. 2, the efficiency of the five algorithms can be categorized into two groups. Generally, the three projection-based algorithms (PrefixSpan, TRuleGrowth, and SPM-FC-P) were faster than the two level-wise algorithms (GSP and SPM-FC-L). This is consistent with the existing consensus in the field of pattern mining; that is, pattern-growth-based algorithms are more efficient because numerous candidates and multiple database scans can be avoided effectively. The two proposed algorithms demonstrated efficiency comparable with their counterpart algorithms. Specifically, SPM-FC-L was slightly faster than GSP, and SPM-FC-P was slightly slower than PrefixSpan and slightly faster than TRuleGrowth.

In this set of experiments, the two proposed algorithms were not faster than PrefixSpan. This can be explained by the following two aspects. The low efficiency of SPM-FC-L was caused by its level-wise search space traversal, whereas the main reason that algorithm SPM-FC-P was slightly slower than PrefixSpan is that SPM-FC-P had to calculate three types of constraints in addition to the corresponding supports, and then integrate them into an SFC.

6.2 Memory consumption

The memory usage of the five algorithms was also compared. The results are shown in Fig. 3.

The plots of the results for this set of comparisons can also be divided into two categories that are similar to the results in Fig. 2. For the two level-wise algorithms, the proposed SPM-FC-L algorithm consumed less memory than the GSP algorithm, on average, whereas for the three projection-based algorithms, the memory consumption of the proposed SPM-FC-P algorithm was less than that of PrefixSpan, and comparable with that of TRuleGrowth. For example, when the minimum support threshold was 0.4%, SPM-FC-P saved nearly half the memory compared with PrefixSpan. This was mainly because a considerable number of SPs were not FCSPs when using SFC. Thus, fewer results for the proposed algorithms could avoid unnecessary

Table 6 Performance comparison for the level-wise algorithms

Algorithm	Runtime (Sec)	Memory usage (MB)	Number of SPs
SPM-LC-L	378.39	1212.01	2347
SPM-DC-L	601.68	1474.88	1911
SPM-VC-L	561.53	1318.12	2143
SPM-FC-L	589.31	1310.34	2081

join operations and database projections, which led to less memory consumption.

For SPM-FC-P and TRuleGrowth, the memory consumption of SPM-FC-P in the first set of experiments was worse than that of TRuleGrowth, whereas the memory consumption of SPM-FC-P in the second set of experiments was better than that of TRuleGrowth. The results were closely related to the number of discovered SPs; that is, SPM-FC-P consumed more memory than TRuleGrowth when the number of discovered FCSPs was more than the number of SPs discovered by TRuleGrowth, whereas SPM-FC-P consumed less memory than TRuleGrowth when the number of discovered FCSPs was fewer than the number of SPs discovered by TRuleGrowth, on average. This is also verified in the comparison of the number of discovered patterns in Section 6.3.

6.3 Number of discovered patterns

The number of SPs discovered by our algorithms was also compared with the number of SPs discovered by the other three algorithms. The results are shown in Fig. 4. Because SPM-FC-L and SPM-FC-P returned the same results, and GSP and PrefixSpan returned the same results, the results for the comparison were discovered using SPM-FC-P (FCSPs) and PrefixSpan (FSPs), respectively. Because the SPs discovered by TRuleGrowth, used for extracting partially ordered sequential rules (POSRs), are different from both FCSPs and FSPs, this type of SP is denoted by partially ordered SPs (POSPs) in this set of experiments.

Figure 4 shows that the number of FCSPs was always smaller than the number of FSPs. This reflects that flexible constraints could present fewer results to users according to the characteristics of MOOC data. Generally, the greater the number of results found, the greater the number of results the proposed algorithms could reduce. For example, when *min_sup* was 0.05%, the maximum number of FSPs and FCSPs could be determined, and the number of FCSPs was 6,837 smaller than the number of FSPs.

For the results discovered by TRuleGrowth, the number of FCSPs was sometimes less than the number of POSPs, but more often, the number of FCSPs was more than that of POSPs. The reason behind these results is that POSPs are used for generating POSRs pair by pair. Within each pair of POSPs, one POSP is tested for the antecedent, and the other is verified for the consequent. Items in each POSP are unordered. Thus, a large number of permutation results of SPs caused by different orders are avoided. Thus, the number of final resulting POSPs is reduced accordingly.

Table 7 Performance comparison for the projection-based algorithms

Algorithm	Runtime (Sec)	Memory usage (MB)	Number of SPs
SPM-LC-P	21.45	559.19	10,472
SPM-DC-P	31.53	694.98	11,847
SPM-VC-P	29.99	628.09	11,678
SPM-FC-P	30.71	599.87	11,325

Table 8 Two input sequences containing S_1

sid	Input sequence
IS_α	(Ideological and moral cultivation, 2016/10/18 3:47), (Introduction to Zizhi Tongjian, 2016/12/6 8:03), (Hybrid learning, 2016/12/6 1:42), (News photography, 2016/12/6 6:35), (The practice of MOOC teaching, 2016/12/6 12:19), (Literature management and information analysis, 2017/9/8 6:05), (Chinese culture, 2017/9/8 7:29), (Traditional Chinese medicine health preservation, 2017/9/8 7:50)
IS_β	(The practice of MOOC teaching, 2017/2/20 11:56), (Literature management and information analysis, 2017/4/26 0:20), (History of Chinese Architecture, 2017/9/8 5:35), (Engineering geology, 2017/9/8 6:50), (Traditional Chinese medicine health preservation, 2017/9/8 7:42)

6.4 Impact of a single constraint

The two proposed algorithms measure the importance of FCSPs with SFC, which is the integration of SLC, SDC, and SVC. To show the effect of each constraint, the performance of each proposed algorithm was compared with that of its counterpart that uses only one constraint.

As discussed in Sects. 6.1 and 6.2, the performance of SPM-FC-L was lower than the performance of SPM-FC-P. Therefore, the comparison between the four level-wise algorithms was conducted using a group of high thresholds. The three level-wise algorithms used only the length constraint, discreteness constraint, and validity constraint denoted by SPM-LC-L, SPM-DC-L, and SPM-VC-L, respectively. The comparison between the four projection-based algorithms was conducted on the group of low thresholds. The three projection-based algorithms used only the length constraint, discreteness constraint, and validity constraint denoted by SPM-LC-P, SPM-DC-P, and SPM-VC-P, respectively.

Table 9 Two input sequences containing S_2

sid	Input sequence
IS_γ	(Ideological and moral cultivation and legal basis, 2016/10/27 14:43), (Scenario and policy, 2017/3/17 2:54), (Introduction to financial engineering, 2017/10/26 10:13), (Fiscal policy and tax reform, 2017/11/3 12:40), (Career planning, 2017/11/3 14:17), (Traditional Chinese rites, 2017/12/8 14:57)
IS_δ	(Ideological and moral cultivation and legal basis, 2016/11/2 2:42), (Scenario and policy, 2017/5/12 7:58), (Fiscal policy and tax reform, 2017/10/14 7:16), (Traditional Chinese rites, 2017/12/24 13:24)

The runtime, memory consumption, and the number of discovered SPs were compared, and the middle threshold of each threshold group was used, that is, 0.5% for level-wise algorithms and 0.07% for projection-based algorithms. The comparison results are shown in Tables 6 and 7.

From Tables 6 and 7, the algorithms that only considered the length constraint (SPM-LC-L and SPM-LC-P) performed best, the two algorithms that only considered the discreteness constraint (SPM-DC-L and SPM-DC-P) performed worst, and the performance of the two proposed algorithms (SPM-FC-L and SPM-FC-P) using three constraints was between the performance of the three algorithms using a single constraint. Compared with the other two constraints, the length constraint was the easiest to calculate. Furthermore, the value of SLC decreased as the length of the input sequence increased. Without considering the actual meaning of the discovered SPs, these two features of SLC made the two length-constraint-based algorithms perform best, on average.

6.5 Pattern analysis

From Theorem 3, any FCSP is also an FSP. To show the effect of the constraints, two typical FSPs that were not FCSPs were analyzed.

When *min_sup* was set to 0.75%, $S_1 = <$Literature management and information analysis, Traditional Chinese medicine health preservation$>$ was discovered as an FSP, but not an FCSP. To analyze the reason for this, two random input sequences containing S_1 are shown in Table 8.

For the two selected input sequences, IS_α was long, and contained two casual enrollments, whereas IS_β was a typical input sequence that satisfied all three constraints (length, discreteness, and casual enrollments). Similarly, other input sequences containing S_1 reduced the SFC because of the length, discreteness, and validity, hence S_1 was filtered out by the proposed algorithms.

As another example, when *min_sup* was set to 0.3%, $S_2 = <$Ideological and moral cultivation and legal basis, Fiscal policy and tax reform, Traditional Chinese rites$>$ was discovered as an FSP, but not an FCSP. Similarly, two input sequences containing S_2 were randomly selected, and are shown in Table 9.

From Table 9, $sup_{FC}(S_2)$ reduced because of IS_γ for two reasons. One is that IS_γ was long, which led to a small contribution to $sup_{FC}(S_2)$. The other is high discreteness; hence, the contribution to $sup_{FC}(S_2)$ was small. In addition to high discreteness, two out of the three items in S_2 with respect to IS_δ were casual enrollments. Thus, the contribution of IS_δ to $sup_{FC}(S_2)$ was small.

To further analyze the interestingness of the resulting SPs, the differences between the FCSP results and the results discovered by only using one constraint were compared. This was achieved by checking the results discussed in Section 6.4. When the minimum threshold was set to 0.06%, two interesting FCSPs that were not discovered by any single constraint were selected. They were $S_3 = <$Surgical nursing, Discipline studies in nursing, Community nursing, Geriatric nursing$>$ and $S_4 = <$Surgical nursing, Community nursing, Gynecology nursing$>$. Both S_3 and S_4 are courses in nursing. They are certainly interesting and useful for people who want to study nursing, medicine, or related courses.

The above pattern analysis has illustrated that the proposed constraints can effectively filter patterns that are deemed to be less interesting.

7 Conclusions and future work

MOOCs are changing education at the present time. SPM is an effective tool for analyzing the historical behavior of numerous online learners. By analyzing the characteristics of MOOC data, flexible constraints were considered from the perspectives of the length of enrollment sequences, span of enrollment dates, and enrollment moments. To push these constraints deep into the mining process, the SFC was designed step by step, and it was proved that this new parameter also satisfies the downward closure property, which reduced the search space greatly and effectively. Two algorithms called SPM-FC-L and SPM-FC-P were proposed for the breadth-first and depth-first traversal of the search space, respectively. The experimental results demonstrated that the proposed algorithms discovered fewer results than FSPs. Furthermore, their efficiency and memory consumption were comparable with classical SPM algorithms.

To the best of the authors' knowledge, there has been very little research on SPM from MOOC data, let alone incorporating constraints. The proofs of downward closure allow the three constraints to be used together or individually according to the real-world problem. Therefore, the two proposed algorithms are meaningful in terms of whether they improve the design of MOOCs or improve the learning quality of learners.

Designing more efficient algorithms to discover FCSPs by proposing novel search space traversal and pruning strategies will be attempted in future work. Furthermore, FCSPs will be used instead of FSPs to recommend more suitable learning resources to learners. Other potential interesting future work includes feature selection in actionable SPs [22], visualization of FCSPs [3, 13], and mining FCSP with a deep neural network [17, 20, 28, 44].

Funding This work was supported by the National Natural Science Foundation of China (61977001) and Great Wall Scholar Program (CIT&TCD20190305).

Declarations

Conflict of Interest The authors declare that they have no conflict of interest.

References

1. Baker RS (2014) Educational data mining: an advance for intelligent systems in education. IEEE Intell Syst 29(3):78–82
2. Chen E, Cao H, Li Q, Qian T (2008) Efficient strategies for tough aggregate constraint-based sequential pattern mining. Inf Sci 178(6):1498–1518
3. Chen Y, He F, Li H, Zhang D, Wu Y (2020) A full migration BBO algorithm with enhanced population quality bounds for multimodal biomedical image registration. Appl. Soft Comput. 93
4. Diop L, Diop C T, Giacometti A, Li D, Soulet A (2018) Sequential pattern sampling with norm constraints. In: Proceedings of the 2018 IEEE International Conference on Data Mining (ICDM'18), pp 89–98
5. Dong X, Gong Y, Cao L (2020) e-RNSP: an efficient method for mining repetition negative sequential patterns. IEEE Trans Cybern 50(5):2084–2096
6. Duong HV, Truong TC, Tran AN, Le B (2020) Fast generation of sequential patterns with item constraints from concise representations. Knowl Inf Syst 62(6):2191–2223
7. Fan W, Hu C (2017) Big graph analyses: from queries to dependencies and association rules. Data Sci Eng 2(1):36–55
8. Feng W, Tang J, Liu T X (2019) Understanding dropouts in MOOCs. In: Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI'19), pp 517–524
9. Fournier-Viger P, Lin J C-W, Gomariz A, Gueniche T, Soltani A, Deng Z, Lam H T (2016) The SPMF open-source data mining library version 2. In: Proceedings of the 19th European Conference on Machine Learning and Knowledge Discovery in Databases (PKDD'16), pp 36–40
10. Fournier-Viger P, Nkambou R, Mayers A (2008) Evaluating spatial representations and skills in a simulator-based tutoring system. IEEE Trans Learn Technol 1(1):63–74
11. Fournier-Viger P, Wu C-W, Tseng VS, Cao L, Nkambou R (2015) Mining partially-ordered sequential rules common to multiple sequences. IEEE Trans Knowl Data Eng 27(8):2203–2216
12. Gan W, Lin J C-W, Fournier-Viger P, Chao H-C, Yu P S (2019) A survey of parallel sequential pattern mining. ACM Trans Knowl Discov Data 13 (3)
13. Guo Y, Guo S, Jin Z, Kaul S, Gotz D, Cao N (2021) A survey on visual analysis of event sequence data. IEEE Trans Vis Comput Graph
14. Guyet T, Quiniou R (2020) NegPSpan: efficient extraction of negative sequential patterns with embedding constraints. Data Min Knowl Discov 34(2):563–609
15. Huynh B, Vo B, Snásel V (2017) An efficient method for mining frequent sequential patterns using multi-Core processors. Appl Intell 46(3):703–716
16. Jalal A, Mahmood M (2019) Students' behavior mining in e-learning environment using cognitive processes with information technologies. Educ Inf Technol 24(5):2797–2821
17. Jamshed A, Mallick B, Kumar P (2020) Deep learning-based sequential pattern mining for progressive database. Soft Comput 24:17233–17246
18. Jaysawal B P, Huang J-W (2018) PSP-AMS: progressive mining of sequential patterns across multiple streams. ACM Trans Knowl Discov Data 13(1)
19. Kinnebrew JS, Loretz KM, Biswas G (2013) A contextualized, differential sequence mining method to derive students' learning behavior patterns. J Educ Data Min 5(1):190–219
20. Kumar N, Sukavanam N (2020) An improved CNN framework for detecting and tracking human body in unconstraint environment. Knowl Based Syst 193
21. Li C, Yang Q, Wang J, Li M (2012) Efficient mining of gap-constrained subsequences and its various applications. ACM Trans Knowl Discov Data 6(1)
22. Li H, He F, Chen Y, Pan Y (2021) MLFS-CCDE: multi-objective large-scale feature selection by cooperative coevolutionary differential evolution. Memetic Comput 13(1):1–18
23. Li Y, Wang G, Yuan Y, Cao X, Yuan L, Lin X (2018) PrivTS: differentially private frequent time-constrained sequential pattern mining. In: Proceedings of the 23rd International Conference on Database Systems for Advanced Applications (DASFAA'18), pp 92–111
24. Le H H, Yamada T, Honda Y, Kayahara M, Kushima M, Araki K, Yokota H (2019) Analyzing sequence pattern variants in sequential pattern mining and its application to electronic medical record systems. In: Proceedings of the 30th International Conference on Database and Expert Systems Applications (DEXA'19), pp 393–408
25. Nawaz MS, Fournier-Viger P, Shojaee A, Fujita H (2021) Using artificial intelligence techniques for COVID-19 genome analysis. Appl Intell 51(5):3086–3103
26. Nguyen D, Luo W, Nguyen T D, Venkatesh S, Phung D Q (2018) Sqn2Vec: learning sequence representation via sequential patterns with a gap constraint. In: Proceedings of European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD'18), pp 569–584
27. Pei J, Han J, Mortazavi-Asl B, Wang J, Pinto H, Chen Q, Dayal U, Hsu M (2004) Mining sequential patterns by pattern-growth: the PrefixSpan approach. IEEE Trans Knowl Data Eng 16(11):1424–1440
28. Quan Q, He F, Li H (2021) A multi-phase blending method with incremental intensity for training detection networks. Vis Comput 37(2):245–259
29. Ren J-M, Jang J-SR (2012) Discovering time-constrained sequential patterns for music genre classification. IEEE Trans Speech Audio Process 20(4):1134–1144
30. Seno M, Karypis G (2002) SLPMiner: an algorithm for finding frequent sequential patterns using length-decreasing support constraint. In: Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02), pp 418–425
31. Song W, Rong K (2018) Mining high utility sequential patterns using maximal remaining Utility. In: Proceedings of the Third International Conference on Data Mining and Big Data (DMBD'18), pp 466–477
32. Song W, Zhang ZH, Li JH (2016) A high utility itemset mining algorithm based on subsume index. Knowl Inf Syst 49(1):315–340

33. Srikant R, Agrawal R (1996) Mining sequential patterns: generalizations and performance improvements. In: Proceedings of the 5th International Conference on Extending Database Technology (EDBT'96), pp 3–17
34. Uddin I, Imran AS, Muhammad K, Fayyaz N, Sajjad M (2021) A systematic mapping review on MOOC recommender systems. IEEE Access 9:118379–118405
35. Ventura S, Luna JM (2016) Pattern mining with evolutionary algorithms. Springer, Cham, Switzerland
36. Wang R, Zaïane O R (2018) Sequence-based approaches to course recommender systems. In: Proceedings of the 29th International Conference on Database and Expert Systems Applications (DEXA'18), pp 35–50
37. Wong J, Khalil M, Baars M, de Koning B B, Paas F (2019) Exploring sequences of learner activities in relation to self-regulated learning in a massive open online course. Comput Educ 140
38. Wu R, Li Q, Chen X (2019) Mining contrast sequential pattern based on subsequence time distribution variation with discreteness constraints. Appl Intell 49(12):4348–4360
39. Wu Y, Wang L, Ren J, Ding W, Wu X (2014) Mining sequential patterns with periodic wildcard gaps. Appl Intell 41(1):99–116
40. Yun U, Ryu KH (2010) Discovering important sequential patterns with length-decreasing weighted support constraints. Int J Inf Technol Decis Mak 9(4):575–599
41. Zaki M J (2000) Sequence mining in categorical domains: incorporating constraints. In: Proceedings of the 2000 ACM CIKM International Conference on Information and Knowledge Management (CIKM'20), pp 422–429
42. Zhang H, He L (2021) Data mining method of sequential patterns for vehicle trajectory prediction in VANET. Wirel Pers Commun 117(2):417–429
43. Zhang M, Zhu J, Wang Z, Chen Y (2019) Providing personalized learning guidance in MOOCs by multi-source data analysis. World Wide Web 22(3):1189–1219
44. Zhang S, He F (2020) DRCDN: learning deep residual convolutional dehazing networks. Vis Comput 36(9):1797–1808