在MOOCCube中发现具有效率约束的序列模式

彭亚威,李艳红*,杨洋,张法
(中南民族大学 计算机科学学院,武汉 430074)

摘 要 大规模开放在线课程(MOOC)平台作为在线学习的重要工具,能够捕捉丰富的学习行为数据.MOOCube作为关键的数据存储库,汇聚了来自学堂在线的48640个学习行为序列.然而,鉴于MOOC数据的复杂性,传统的序列模式挖掘(SPM)算法在处理这类数据时可能会遇到挑战,导致挖掘出大量不相关的模式.为了提升挖掘结果的有效性,提出了一种效率约束序列模式挖掘(ECSPM)方法.该方法引入了出勤性、离散性和辍学性三大关键约束,这些约束能够精准捕捉学习行为中的不同特征对序列模式挖掘的影响.值得注意的是,这些约束具有向下封闭性质,确保了它们在模式挖掘过程中的有效性.为了发现序列模式(SP),精心设计了3种算法.这些算法结合了逐级搜索空间遍历或递归投影技术,并将成本概念融入模式挖掘过程中.实验评估结果表明:本文提出的算法效果显著.ECSPM不仅显著减少了发现的模式的数量,而且其性能与经典SPM算法相当,甚至在某些方面更优.

关键词 学习行为;序列模式挖掘;效率约束序列模式挖掘;成本
中图分类号 TP399 **文献标志码** **文章编号** 1672-4321(2025)03-0373-11
doi:10.20056/j.cnki.ZNMDZK.20250311

Discovering sequential patterns with efficiency constraints in MOOCCube

PENG Yawei, LI Yanhong*, YANG Yang, ZHANG Fa
(College of Computer Science, South-Central Minzu University, Wuhan 430074, China)

Abstract Massive Open Online Course (MOOC) platforms have emerged as crucial tools in online learning, capturing vast learning behavior data. MOOCube, a significant data repository, encompasses 48640 learning behavior sequences extracted from XuetangX. However, traditional Sequential Pattern Mining (SPM) algorithms may struggle to handle the complexity of MOOC data, leading to irrelevant patterns. This paper proposes Efficiency-Constrained Sequential Pattern Mining (ECSPM) to facilitate the discovery of useful patterns. Three constraints are introduced: attendance, discreteness, and dropout. These constraints capture the influence of different features of learning behavior on sequential patternsmining. Importantly, these constraints are proven to satisfy the downward closure property, ensuring their effectiveness in shaping the mining process. To discover Sequential Patterns, three algorithms were proposed. These algorithms employ level-by-level search space traversal or recursive projection techniques while integrating the concept of cost into pattern mining. Experimental evaluation confirms the effectiveness of the proposed algorithm. ECSPM has successfully reduced the number of discovered patterns while maintaining comparable performance to the classical SPM algorithm.

Keywords learning behavior; sequential pattern mining; efficiency-constrained sequential pattern mining; cost

近年来随着新冠肺炎(COVID-19)疫情的爆发,大规模开放在线课程(MOOC)的招生规模急剧扩大,为数据挖掘研究提供了丰富而互动的多媒体平台[1].探索MOOC数据中的学习行为特征对于优化学习体验至关重要[2].数据挖掘技术被广泛应用于MOOC数据集中,以提取有价值的见解,尤其关注学生行为

收稿日期 2024-06-09
* **通信作者** 李艳红(1973-),女,教授,博士,研究方向:时空数据处理、网络大数据、DB与AI融合,E-mail:liyanhong@mail.scuec.edu.cn
基金项目 湖北省自然科学基金资助项目(2017CFB135);中央高校基本科研业务费专项资金资助项目(CZY23019);网络创新及应用型人才课程实践教学研究项目(2019年第一批)

分析[3].学生行为数据的分析对学业成绩预测[4]、课程推荐[5]及课程质量满意度研究[6]等领域均产生了重要影响.借助数据挖掘技术,研究人员能更深入地理解学生行为及其潜在影响[7].

序列模式挖掘(SPM)是分析学生行为记录的关键手段,旨在发现支持度达到或超过预设最小阈值的子序列.目前,已有多种算法应用于此领域,包括广度优先[8]、深度优先[9]和模式增长算法[10].这些算法有效地从数据集中识别和提取模式,生成大量序列模式(SP).尽管这些结果证明了SPM算法的有效性,但过多的模式可能导致识别有意义且可操作的见解变得困难.为了克服这一挑战,研究人员引入了各种约束来完善结果并关注最重要的SP.通过纳入间隙约束[11]和离散性约束[12]等约束,研究人员可以将SP列表缩小到具有实际意义的SP.

近年来,SPM在车辆轨迹预测[13]、在线学习[14]、课程推荐[15]等领域取得了显著成果.特别是在大规模开放在线课程(MOOC)领域,其应用前景广阔.然而,MOOC数据因其庞大的规模和复杂性,包含用户交互、课程材料和带时间戳的操作等多种信息,使得传统的SPM算法难以处理,并可能产生不相关的模式.因此,将约束融入传统的SPM算法,以有效地发现有意义的序列模式至关重要[16].

本文采用出勤性、离散性和辍学性三个约束,并将其集成到支撑测量中.本研究证明这些修改后的支撑保留了基本的向下封闭属性,确保了方法的可靠性.为发现具有效率约束的序列模式(SP),本文提出了3种算法:ECSPM-L、ECSPM-P和ECSPM-C.这些算法结合逐级搜索空间遍历或递归投影技术,同时考虑成本因素,优化模式挖掘过程.实验中使用的数据集是MOOCCube,可从MoocData平台(http://moocdata.cn/data/MOOCCube)获取.该数据集来源于中国知名MOOC网站学堂在线(https://next.xuetangx.com),包含2015年7月7日—2020年4月17日期间的48640个行为序列和685个课程[17].

1 相关工作

1.1 SPM的模式增长算法

传统的序列模式挖掘算法可以分为深度优先、广度优先和模式增长算法,其中模式增长算法是序列模式挖掘领域的主流策略,例如FreeSpan[18],PrefixSpan[10]和MaxSP[19]算法,其高效性和准确性备受认可[20].其核心思想是专注于数据库中的现有模式,进而仅对相关模式进行深度挖掘,以生成高效、有意义的结果.在众多模式增长算法中,PrefixSpan尤为突出.它运用前缀策略,通过不断向现有模式添加新元素来逐步扩展,确保算法能够充分考虑事件或元素的顺序,从而精准发现频繁子序列.模式增长算法的优点是只考虑数据库中存在的模式.因此,PrefixSpan在生物信息学、市场篮分析、网络挖掘以及流程挖掘等多个领域得到了广泛应用,成为序列模式挖掘研究人员的得力工具,近年来已经提出了许多其他的算法和扩展[21].

1.2 SPM中约束的集成

为提升序列模式的实用性,减少模式数量,研究者们提出了在SPM中集成约束的策略.这些约束通常由用户根据特定需求设定,用以筛选符合特定属性或要求的模式[22].这些约束可涵盖模式长度[23]、项目组合[24]、时间间隔[25]等多个方面.特别是在时间约束序列模式挖掘中,最小间隔和最大间隔限制是常见的约束条件[8].前者确保连续事件在合理时间范围内发生,捕捉时间依赖性;后者则排除时间间隔过长的事件,保证所发现模式的实用性.此外,滑动时间窗口约束允许模式跨越多个事务,只要这些事务在特定时间窗口内发生,增强了模式挖掘的灵活性[26].本研究将针对MOOCCube数据集中的时间信息提出相应约束,以发现更具实际意义的模式.

1.3 SPM中的成本整合

虽然成本因素在模式挖掘中的考虑相对较少,但仍有相关工作对此进行了探索.例如,Dalmas[27]在分析医院事件日志时考虑了每个事件的成本值,以评估成本对治疗方法的影响.CEPN[28]是另一种方法,它使用带有成本值的事件日志来评估模式的成本效益.本研究旨在将成本信息纳入序列模式挖掘中,特别是针对MOOCCube数据集.通过考虑成本因素,优化挖掘过程,并发现符合指定效率约束的重要模式.这将有助于在实际应用中实现更经济、更高效的模式挖掘.

2 问题定义

考虑一组课程,用集合Δ来表示.每门课程都通过一个项目来呈现,该项目是一个包含两个元素的元组(c, T),其中,c是课程集Δ的一个子集,它标识特定的课程;T是另一个包含4个元素的集合,即

$T=\{s,e,g,d\}$，分别代表课程的开始时间s、结束时间e、进度时间g和总持续时间d. 序列S是一个包含n个按时间顺序排列的项目的列表，具体形式为$S=\langle(c_1,T_1),(c_2,T_2),\cdots,(c_n,T_n)\rangle$. 序列$S$中的每个项目都代表一门课程及其相关信息. 在序列$S$中，对于任意$1\leqslant i<j\leqslant n$，都满足条件$T_i.s<T_j.s$，这确保了项目是按照它们的开始时间进行排序的. 序列S的长度$|S|$表示序列中项目的总数. 本研究使用$S[i]$引用序列中的特定项目，其中$1\leqslant i\leqslant n$. $S[i].c$表示与第i个项目相关的课程，而$S[i].s$、$S[i].e$、$S[i].g$和$S[i].d$则分别表示该课程的开始时间、结束时间、进度时间和总持续时间. 定义模式$p=\langle c'_{a1},c'_{a2},\cdots,c'_{ao}\rangle$，对于$1\leqslant a_1\leqslant a_2\leqslant\cdots\leqslant a_o\leqslant n$，存在$c'_{a1}=c_1,c'_{a2}=c_2,\cdots,c'_{ao}=c_n$，则表示序列$S$包含模式$p$，模式$p$在序列$S$中的项目集记作$S^p$.

序列行为记录(SBR)是由m个序列组成的集合，记作$\mathrm{SBR}=\{S_1,S_2,\cdots,S_m\}$. 特别需要注意的是，在输入序列$S$中，每个模式$p$在MOOCCube数据集中至多出现一次. 在SBR中，模式p的支持度用$\mathrm{sup}(p)$表示，指模式p在SBR中出现的次数. 同时，模式p的支持集记作$\mathrm{supset}(p)$，则是包含模式p的SBR序列集合. 若模式p的支持度$\mathrm{sup}(p)$达到或超过用户设定的最小支持度阈值minsup，则该模式p被视为频繁的.

例如在表1中呈现的序列行为记录，每个项目的时间维度信息(T)的样本如表2所示. 在SBR中，序列S_1、S_2和S_3均包含模式p =< 操作系统, 数据库 >，这3个序列共同构成了该模式的支持集$\mathrm{supset}(p)$. 为了确定频繁序列模式(FSP)，需要设定一个名为minsup的支持度阈值. 假设minsup设置为2，那么模式p =< 操作系统, 数据库 >即满足频繁序列模式的条件，因为它至少出现在两个序列中. 然而在MOOCCube数据集中挖掘序列模式时，需要采取优化措施，直接挖掘可能产生大量琐碎或无意义的模式，分析这些模式不仅计算成本高昂且耗时，对于决策目的而言也缺乏实用性.

表1　序列行为记录

Tab. 1　Sequential behavior recording

Id	Sequence
S_1	<操作系统，计算机网络，数据库，数据挖掘>
S_2	<操作系统，数据库，计算机体系结构>
S_3	<操作系统，数据库>
S_4	<算法设计与分析，数据挖掘>

表2　两条序列的时间维度信息

Tab. 2　Time dimension information of two sequences

Id	Sequence
S_2	（操作系统，2017/2/20，2017/4/20，265，435），（数据库，2017/3/12，2017/6/1，315，671），（计算机体系结构，2017/10/20，2018/1/4，306，498）
S_3	（操作系统，2017/3/21，2017/4/6，73，435），（数据库，2017/9/20，2017/12/6，483，671）

在序列模式挖掘中引入约束是应对大型结果集和计算效率挑战的一种有效技术. 这些约束定义了模式必须满足的严格条件[29]. 结合MOOCCube数据集复杂的时间维度特性，本文专注于在序列行为记录数据中发现符合效率约束的顺序模式(ECSP). 效率约束序列模式挖掘(ECSPM)问题的目标是找到符合SBR中指定效率约束的完整ECSP集合.

3　本文方法

本研究聚焦于分析在线教育领域的MOOCCube数据集，旨在从学习序列长度、学习注册时间、学习周期有效时长及总时长四个维度深入探索MOOC的学习序列. 为有效建模，引入了出勤性、离散性和辍学性三个关键约束. 本文的方法将这些特定约束融入支持度之中，并经过验证，这些修改后的支持度确实遵循向下闭合属性.

本文提出了3种算法，均旨在发现效率约束的序列模式(ECSP)，这些算法采用了不同的策略来整合效率约束，概述如下：

ECSPM-L(逐级效率约束序列模式挖掘)：该算法通过逐级遍历搜索空间，深入探索每个层级，并在整个过程中充分考虑效率约束；

ECSPM-P(基于投影的效率约束序列模式挖掘)：此算法运用递归投影技术导航搜索空间，通过将序列投影至更小的子序列上，并在合并效率约束的同时，递归地探索这些子序列；

ECSPM-C(集成成本的效率约束序列模式挖掘)：该算法在ECSPM-P的基础上进一步扩展，通过集成成本信息，为搜索空间的探索过程提供指导，确保在探索过程中充分考虑效率约束.

3.1　出勤性约束

定义1　(出勤性约束的支持度)　对于模式p，在特定的序列集合S的上下文中，模式p在序列S中

的项目集记作S^p,其出勤性约束是一个量化指标,用于衡量p的出勤情况,计算公式为:

$$\mathrm{AttC}(p,S)=\exp\left(-\sum_{i=1}^{|S^p|}\left(\frac{S^p[i].e-S^p[i].s}{S^p[i].d}\cdot S^p[i].g\right)\right). \tag{1}$$

进一步地,本研究将出勤性约束与支持度概念相结合,定义了模式p的出勤性约束的支持度,以评估模式p在数据集中的重要性,计算公式为:

$$\sup_{\mathrm{AttC}}(p)=\sum_{S\in \mathrm{supset}(p)}\mathrm{AttC}(p,S)=\sum_{S\in \mathrm{supset}(p)}\exp\left(-\sum_{i=1}^{|S^p|}\left(\frac{S^p[i].e-S^p[i].s}{S^p[i].d}\cdot S^p[i].g\right)\right). \tag{2}$$

定理1 对于任意两个模式p_X和p_Y,如果p_X是p_Y的子集($p_X\subseteq p_Y$),那么$\sup_{\mathrm{AttC}}(p_Y)\leqslant\sup_{\mathrm{AttC}}(p_X)$.

证明 对于$p_X\subseteq p_X$,有两种情况:

1) 如果$p_X=p_Y$,$\mathrm{supset}(p_Y)=\mathrm{supset}(p_X)$,$\sup_{\mathrm{AttC}}(p_Y)=\sup_{\mathrm{AttC}}(p_X)$;

2)如果$p_X\subset p_Y$,$\mathrm{supset}(p_Y)\subseteq\mathrm{supset}(p_X)$,对于同样包含$p_X$和$p_Y$的序列$S$,

$$\mathrm{AttC}(p_X,S)\geqslant\mathrm{AttC}(p_Y,S). \tag{3}$$

如果$\mathrm{supset}(p_Y)=\mathrm{supset}(p_X)$,那么:

$$\sup_{\mathrm{AttC}}(p_Y)=\sum_{S\in\mathrm{supset}(p_Y)}\mathrm{AttC}(p_Y,S)=\sum_{S\in\mathrm{supset}(p_X)}\mathrm{AttC}(p_Y,S)\leqslant\sum_{S\in\mathrm{supset}(p_X)}\mathrm{AttC}(p_X,S)=sup_{AttC}(p_X); \tag{4}$$

如果$\mathrm{supset}(p_Y)\subset\mathrm{supset}(p_X)$,那么:

$$\sup_{\mathrm{AttC}}(p_Y)=\sum_{S\in\mathrm{supset}(p_Y)\wedge S\in\mathrm{supset}(p_X)}\mathrm{AttC}(p_Y,S)\leqslant\sum_{S\in\mathrm{supset}(p_Y)\wedge S\in\mathrm{supset}(p_X)}\mathrm{AttC}(p_X,S)<\sum_{S\in\mathrm{supset}(p_Y)\wedge S\in\mathrm{supset}(p_X)}\mathrm{AttC}(p_X,S)+\sum_{S'\notin\mathrm{supset}(p_Y)\wedge S'\in\mathrm{supset}(p_X)}\mathrm{AttC}(p_X,S')=\sup_{\mathrm{AttC}}(p_X). \tag{5}$$

经过验证,出勤性约束的支持度的一个显著优势在于其满足向下封闭属性,这一特性对于修剪搜索空间、降低挖掘算法的计算复杂性具有重要意义.

3.2 离散性约束

定义2 (离散性约束的支持度) 为了捕捉序列内学习行为初始时刻离散性的变化,本研究提出离散性约束.以表1中的序列为例,考虑模式p =< 操作系统, 数据库 >,序列S_2和S_3均包含该模式.表2还展示了学习行为的开始时间、结束时间、进度时间和课程总持续时间的具体数据.在MOOC平台上,学习行为时间离散性小的序列更受青睐.这意味着,当多个序列包含相同的模式时,那些在学习行为开始时表现出较小离散性的序列,对模式的支持度贡献会更大.以序列S_2为例,模式p的学习行为初始日期跨度为2017年2月20日—2017年3月12日,两个初始日期的平均日期为2017年3月2日,它们与平均日期之间的距离为10 d.而序列S_3中,模式p的学习行为初始日期从2017年3月21日延续至2017年9月20日,两个初始日期的平均日期为2017年月日,它们与平均日期之间的距离为91 d.因此从优化角度看,S_2对模式p的支持度贡献更大.

为了量化序列S上下文中模式p的离散性,模式p在序列S中的项目集记作S^p,将模式p的离散型约束定义如下:

$$\mathrm{DisC}(p,S)=\exp\left(-\sum_{i=1}^{|S^p|}\left(S^p[i].s-\overline{S^p[|S^p|].s}\right)^2\right), \tag{6}$$

其中,$\overline{S^p[|S^p|].s}=\frac{1}{|S^p|}\sum_{i=1}^{|S^p|}S^p[i].s$.

根据这一定义,离散性约束反映了学习行为在序列中时间变化的程度.如果学习行为的初始时间与序列中平均学习行为时间差异显著,则约束值较小.

为了评估数据集中模式p的重要性,本研究引入了离散性约束的支持度,它将支持度与离散性约束相结合:

$$\sup_{\mathrm{DisC}}(p)=\sum_{S\in\mathrm{supset}(p)}\mathrm{DisC}(p,S)=\sum_{S\in\mathrm{supset}(p)}\exp\left(-\sum_{i=1}^{|S^p|}\left(S^p[i].s-\overline{S^p[|S^p|].s}\right)^2\right). \tag{7}$$

定理2 对于任意两个模式p_X和p_Y,如果p_X是p_Y的子集($p_X\subseteq p_Y$),那么$\sup_{\mathrm{DisC}}(p_Y)\leqslant\sup_{\mathrm{DisC}}(p_X)$.

证明 对于$p_X\subseteq p_Y$,有两种情况:

1) 如果$p_X=p_Y$,$\mathrm{supset}(p_Y)=\mathrm{supset}(p_X)$,$\sup_{\mathrm{DisC}}(p_Y)=\sup_{\mathrm{DisC}}(p_X)$;

2)如果$p_X\subset p_Y$,$\mathrm{supset}(p_Y)\subseteq\mathrm{supset}(p_X)$,设$|p_Y|=|p_X|+1$,$p_X=\langle c_1,c_2,\cdots,c_n\rangle$,$p_Y=\langle c_1,c_2,\cdots,c_n,c_{n+1}\rangle$,并且考虑同样包含$p_X$和$p_Y$的序列$S$.于是存在$S[1].c=c_1,\cdots,S[n].c=c_n,S[n+1].c=c_{n+1}$,

满足 $\mathrm{DisC}(p_X,S)=\exp\left(-\sum_{i=1}^{n}\left(S[i].s-\overline{S[n].s}\right)^2\right)$，

$\mathrm{DisC}(p_Y,S)=\exp\left(-\sum_{i=1}^{n+1}\left(S[i].s-\overline{S[n].s}\right)^2\right)$.

$$\overline{S[n+1].s}=\frac{1}{n+1}\left(\sum_{i=1}^{n}S[i].s+S[n+1].s\right)=\frac{n}{n+1}\overline{S[n].s}+\frac{1}{n+1}S[n+1].s=\overline{S[n].s}+\frac{1}{n+1}\left(S[n+1].s-\overline{S[n].s}\right). \tag{8}$$

于是：

$$\sum_{i=1}^{n+1}\left(S[i].s-\overline{S[n].s}\right)^2=\sum_{i=1}^{n+1}\left(S[i].s-\overline{S[n].s}-\frac{1}{n+1}(S[n+1]s.-S[n].s)\right)^2=\sum_{i=1}^{n+1}\left(S[i].s-\overline{S[n].s}\right)^2+\frac{\left(S[n+1].s-\overline{S[n].s}\right)^2}{n+1}-\frac{2\left(S[n+1].s-\overline{S[n].s}\right)}{n+1}\sum_{i=1}^{n+1}\left(S[i].s-\overline{S[n].s}\right), \tag{9}$$

其中(9)式第1项：

$$\sum_{i=1}^{n+1}\left(S[i].s-\overline{S[n].s}\right)^2=\sum_{i=1}^{n}\left(S[i].s-\overline{S[n].s}\right)^2+\left(S[n+1].s-\overline{S[n].s}\right)^2, \tag{10}$$

其中(9)式第3项：

$$\frac{2\left(S[n+1].s-\overline{S[n].s}\right)}{n+1}\sum_{i=1}^{n}\left(S[i].s-\overline{S[n].s}\right)=\frac{2\left(S[n+1]s-\overline{S[n].s}\right)}{n+1}\left(\sum_{i=1}^{n}\left(S[i].s-\overline{S[n].s}\right)\right)+ \tag{11}$$

$$\left(S[i].s-\overline{S[n].s}\right)=\frac{2\left(S[n+1].s-\overline{S[n].s}\right)^2}{n+1}.$$

整理(9)、(10)、(11)式得到：

$$\sum_{i=1}^{n+1}\left(S[i].s-\overline{S[n+1].s}\right)^2=\sum_{i=1}^{n+1}\left(S[i].s-\overline{S[n].s}\right)^2+\frac{n}{n+1}\left(S[n+1].s-\overline{S[n].s}\right)^2, \tag{12}$$

因此 $\sum_{i=1}^{n+1}\left(S[i].s-\overline{S[n].s}\right)^2\geqslant\sum_{i=1}^{n}\left(S[i].s-\overline{S[n].s}\right)^2$，$\mathrm{DisC}(p_Y,S)\leqslant DisC(p_X,S)$，因为 $p_X\subset p_Y$，$\mathrm{supset}(p_Y)\subseteq\mathrm{supset}(p_X)$；

如果 $\mathrm{supset}(p_Y)=\mathrm{supset}(p_X)$，那么：

$$\sup_{\mathrm{DisC}}(p_Y)=\sum_{S\in\mathrm{supset}(p_Y)}\mathrm{DisC}(p_Y,S)=\sum_{S\in\mathrm{supset}(p_X)}\mathrm{DisC}(p_Y,S)\leqslant\sum_{S\in\mathrm{supset}(p_X)}\mathrm{DisC}(p_X,S)=\sup_{\mathrm{DisC}}(p_X). \tag{13}$$

如果 $\mathrm{supset}(p_Y)\subset\mathrm{supset}(p_X)$，那么：

$$\sup_{\mathrm{DisC}}(p_Y)=\sum_{S\in\mathrm{supset}(p_Y)\wedge S\in\mathrm{supset}(p_X)}\mathrm{DisC}(p_Y,S)\leqslant\sum_{S\in\mathrm{supset}(p_Y)\wedge S\in\mathrm{supset}(p_X)}\mathrm{DisC}(p_X,S)<\sum_{S\in\mathrm{supset}(p_Y)\wedge S\in\mathrm{supset}(p_X)}DisC(p_X,S)+\sum_{S'\in\mathrm{supset}(p_Y)\wedge S'\in\mathrm{supset}(p_X)}\mathrm{DisC}(p_X,S')=\sup_{\mathrm{DisC}}(p_X). \tag{14}$$

根据以上分析，当 $|p_Y|=|p_X|+1$，$\sup_{\mathrm{DisC}}(p_Y)\leqslant\sup_{\mathrm{DisC}}(p_Y)$，同理当 $|p_Y|=|p_X|+m(m>1)$ 时，$\sup_{\mathrm{DisC}}(p_Y)\leqslant\sup_{\mathrm{DisC}}(p_Y)$.

这一性质表明离散性约束的支持度满足向下封闭性质(即如果一个项集被认为是频繁的，那么该频繁项集的任何子集也必定是频繁的；反之，如果一个项集被认为是不频繁的，那么包含它的项集必定也是不频繁的)，这对于修剪搜索空间和降低挖掘算法计算复杂性具有重要意义.

3.3　辍学性约束

定义 3　(辍学性约束的支持度)　引入辍学性约束的初衷在于，非辍学学习行为通常反映出学习者对学习的强烈承诺，而辍学主导的学习行为则多发生在容易放弃教育计划的学习者中[30]. 通过设定一个阈值 θ，当学习行为的进度时间占总课程持续时间的比例低于此阈值时，即可将该学习行为归类为辍学主导. 以表2中的序列 S_2 和 S_3 为例，它们都包含模式 p =< 操作系统, 数据库 >. 若将阈值 θ 设为0.4，则 p 在 S_2 中有2个非辍学学习行为，而 S_3 中仅有一个非辍学学习行为和一个导致辍学的学习行为. 因此，在这种情况下，S_2 对 $\sup(p)$ 的贡献大于 S_3.

为了量化序列中模式 p 在特定序列 S 的辍学情况，本研究引入了辍学性约束这一概念. 辍学性约束的计算公式为：

$$\mathrm{DropC}(p,S)=\exp(-\mathrm{NumL}/\mathrm{MaxL}), \tag{15}$$

其中，NumL代表序列 S 中模式 p 的辍学主导学习行为次数，而MaxL则是整个序列集SBR中所有序列的最大长度.

辍学性约束的引入可以明确区分辍学主导行为

与非辍学行为.以表2中的模式p =< 操作系统,数据库 >为例,通过应用辍学约束,计算出$\mathrm{DropC}(p, S_2) = 1$,这表明在学习行为方面,模式$p$在序列$S_2$中的支持度并未因辍学而衰减,因为$S_2$中的两种学习行为均属于非辍学行为.

为了评估数据集中模式p的重要性,本研究进一步提出了辍学性约束的支持度的概念.它结合了支持度概念与辍学性约束,具体计算公式如下:

$$\sup_{\mathrm{DropC}}(p) = \sum_{S \in \mathrm{supset}(p)} \mathrm{DropC}(p, S) = \sum_{S \in \mathrm{supset}(p)} \exp(-\mathrm{NumL}/\mathrm{MaxL}). \tag{16}$$

定理3 对于任意两个模式p_X和p_Y,如果p_X是p_Y的子集($p_X \subseteq p_Y$),那么$\sup_{\mathrm{DropC}}(p_Y) \leqslant \sup_{\mathrm{DropC}}(p_X)$.

对于任意两个模式p_X和p_Y,如果$p_X \subseteq p_Y$,考虑同样包含p_X和p_Y的序列S,因为$\mathrm{DropC}(p_Y, S) \leqslant \mathrm{DropC}(p_X, S)$,那么$\sup_{\mathrm{DropC}}(p_Y) \leqslant \sup_{\mathrm{DropC}}(p_X)$.这一性质表明辍学性约束的支持度满足向下封闭性,这有助于简化搜索空间并降低挖掘算法的计算复杂度.

3.4 约束集成

通过将出勤性、离散性和辍学性三个约束相结合,挖掘算法能够专注于那些同时满足这些约束标准的模式.这种集成约束的方法有效替代了传统的支持度,从而加速了序列模式挖掘的处理过程.

定义4 (效率性约束的支持度) 为了全面评估数据集中模式p的重要性,本研究提出了效率性约束的支持度的概念.它将支持度与效率约束相结合,具体计算公式为:

$$\sup_{\mathrm{EffC}}(p) = \alpha \times \sup_{\mathrm{AttC}}(p) + \beta \times \sup_{\mathrm{DisC}}(p) + \gamma \times \sup_{\mathrm{DropC}}(p), \tag{17}$$

其中,α、β和γ分别代表出勤因子、离散因子和辍学因子,其取值范围均为[0,1],且满足$\alpha + \beta + \gamma = 1$的条件.对于任意两个模式$p_X$和$p_Y$,如果$p_X$是$p_Y$的子集,则必有效率性约束的支持度$\sup_{\mathrm{EffC}}(p_Y) \leqslant \sup_{\mathrm{EffC}}(p_X)$,这同样体现了向下封闭性.

本研究的核心任务是识别序列行为记录数据集中的效率约束序列模式(ECSP),这通过设定最小支持度阈值并确保模式满足效率性约束来实现.如果某序列模式p的效率性约束的支持度$\sup_{\mathrm{EffC}}(p)$不小于设定的最小支持度阈值minsup,则该模式p即被视为ECSP.特别地,长度为l的ECSP被特别标记为l-ECSP.

3.5 ECSPM-L算法

针对ECSPM问题的求解,本研究采用了一种逐级遍历的策略来探索搜索空间,提出一种名为ECSPM-L的算法,该算法在每个遍历层级都融入了效率约束的考量.算法1详细阐述了ECSPM-L算法在挖掘ECSP时的运作流程.

算法1 ECSPM-L

输入: Sequen Behavior Recording SBR, minimum support threshold minsup

输出: All ECSPs

1 $ES_1 \Leftarrow$ 1-ECSP;
2 k=1;
3 while $ES_k \neq \varnothing$ do
4 根据ES_k生成CS_{k+1};
5 $ES_{K+1} = \{S \mid S \in CS_{K+1}, \sup_{\mathrm{EffC}}(S) \geqslant \mathrm{minsup}\}$;
6 k++;
7 end while
8 return $\bigcup_k ES_k$

ECSPM-L算法以序列行为记录SBR和最小支持度阈值minsup作为输入,输出则是挖掘到的ECSP集合.在算法1中,使用ES_k来表示长度为k的ECSP,整个ECSP集合则由所有不同长度的ES_k组成.算法首先初始化ES_1以存储所有1-*ECSP*(第1行).随后,算法开始一个循环,迭代次数从1开始递增,直到所有ECSP被找到为止(第3-7行).在每次迭代中,算法根据当前的ES_k生成长度为k+1的候选ECSP,这些候选序列被表示为CS_{k+1},如果候选序列的效率约束性支持度不小于minsup,则将其加入到ES_{k+1}中.

3.6 ECSPM-P算法

为了进一步优化搜索过程,本研究还提出了一种名为ECSPM-P的算法,该算法采用递归投影的方式来遍历搜索空间并解决ECSP问题.与ECSPM-L相比,ECSPM-P在处理大规模搜索空间时更为高效.在介绍ECSPM-P算法之前,先解释一下序列数据库投影的基本概念.

给定序列行为记录SBR中的模式p和序列S,如果p出现在S中,那么p对S的投影是指序列S中从p第一次出现之后的部分.对于SBR中的模式p,其投影数据库$\mathrm{SBR}|_p$是由SBR中所有以p为前缀的投影序列组成.以表1为例,如果考虑模式p = < 操作系统,数据库 >,则p的投影数据库将如表3所示.基于上述概念,进一步设计了ECSPM-P算法,该算法专门用于挖掘ECSP.算法2详细描述了

ECSPM-P的执行过程.

表3　序列行为记录中的p-projected数据库

Tab. 3　p-projected database in sequential behavior recording

Id	Pattern
S_1'	<数据挖掘>
S_2'	<计算机体系结构>

算法2 ECSPM-P

输入：Sequential Pattern p, p-projected database $\mathrm{SBR}|_p$, min sup

输出：All ECSPs

```
1  ES_1 ⇐ 1-ECSP;
2  for each q ∈ ES_1 do
3    if sup_Effc(p ∪ q) ≥ minsup then
4      将q添加到p的末尾，构成新的ECSP p′;
5      输出p′;
6      构建SBR|_q′;
7      ECSPM-P(p′, SBR|_q′, minsup);
8    end if
9  end for
```

ECSPM-P算法以序列模式p、其投影数据库$SBR|_p$和最小支持度阈值minsup作为输入，输出则是挖掘到的ECSP集合. 在算法开始时，首先在p的投影数据库中识别并保存所有长度为1的ECSP，即ES_1（第1行）. 随后，算法进入一个主循环，该循环遍历ES_1中的每个1-ECSP，记作q. 在每次迭代中（第2-9行），通过将q附加到现有模式p的末尾来生成新的ECSP. 新生成的ECSP在第4行被创建，并在第5行输出. 同时，构建与新生成的ECSP相关的投影数据库（第6行）. 此外，算法还递归调用ECSPM-P过程以进一步扩展其他ECSP（第7行）. 值得注意的是，当首次调用ECSPM-P过程时，参数p是一个空集，因此投影数据库$\mathrm{SBR}|_p$实际上就是原始的SBR.

3.7　ECSPM-C算法

ECSPM-C是一种将递归投影和成本纳入模式挖掘的ECSP挖掘算法. ECSPM-C基于CEPN算法，利用递归投影来遍历搜索空间. 值得注意的是，在ECSPM-C中，MOOCCube数据集中课程的进度时间被用作执行序列模式挖掘的成本.

定义5　（成本衡量）　模式p的成本是基于模式在每个序列中第一次出现的度量.

$$c(p,S)=\sum_{i=1}^{|S^p|}S^p[i].g. \tag{18}$$

模式p的平均成本定义如下：

$$ac(p)=\frac{\sum_{p\subseteq S\wedge S\in \mathrm{SBR}} c(p,S)}{\left|\sup_{\mathrm{Effc}}(p)\right|}. \tag{19}$$

定义6　（平均支持度成本）　对于模式p，设$c_j(p,S)$表示第j个最小的$c(p,S)$，模式p的平均支持度成本（ASC）定义为：

$$\mathrm{asc}(p)=\frac{\sum_{j=1,2,\ldots,\mathrm{minsup}} c_j(p,S)}{\left|\sup_{\mathrm{Effc}}(p)\right|}. \tag{20}$$

定理4　模式p的ASC小于或等于其平均值，ASC是平均成本的下限. 因为$\sup_{\mathrm{Effc}}(p_Y)\geqslant \mathrm{minsup}$，于是$\mathrm{asc}(p)=\dfrac{\sum_{j=1,2,\cdots,\mathrm{minsup}} c_j(p,S)}{\left|\sup_{\mathrm{Effc}}(p)\right|}\leqslant \mathrm{ac}(p)$.

定理5　对于任意两个模式p_X和p_Y，如果p_X是p_Y的子集，则ASC表现出单调性，即$\mathrm{asc}(p_X)\leqslant \mathrm{asc}(p_Y)$.

定理6　对于模式p，如果$\mathrm{asc}(p)>\mathrm{maxcost}$，则$\mathrm{ac}(p)>\mathrm{maxcost}$，所以模式$p$不被视为ECSP，此外模式$p$的任何超集都不被视为ECSP.

通过利用ASC度量和这些定理中建立的原理，研究人员可以有效地估计与序列模式相关的平均成本，并更有效地应用剪枝技术来发现ECSP.

基于前面的讨论，算法3提出了ECSPM-C算法. ECSPM-C将序列模式p、p的投影数据库$\mathrm{SBR}|_p$、minsup和maxcost作为输入，并将ECSP集合作为输出. 首先，p是一个空集，$\mathrm{SBR}|_p$是SBR本身，具有单个项目的ECSP由ES_1发现并保存（第1行）；然后对于ES_1中的每个1-ECSP，记作q，主循环通过将q附加到p的末尾来生成新的ECSP（第2-13行）. 如果$p\cup q$的支持度大于或等于最小支持度阈值（minsup），并且$p\cup q$的平均成本不超过最大成本阈值（maxcost），算法将q添加到p的末尾以创建新的ECSP p'；然后，这个新的ECSP p'将作为有效的ECSP输出（第4-7行）. 此后使用定理3来检查是否满足搜索空间缩减条件. 如果$p\cup q$的支持度小于或等于最大成本阈值，则构建p'的投影数据库. 随后调用ECSPM-C过程以递归地生成ECSP（第8-10行）.

4　实验结果

本实验采用了一台配备Intel（R）Core（TM）i5-8265U CPU和16 GB内存的计算机，并在64位Windows 10操作系统上运行. 实验的核心目标是评估ECSPM算法的有效性，为此，将其与3种常用的

算法3 ECSPM-C

输入：Sequential pattern p, p-projected database $SBR|_p$, minsup, maxcost

输出：All ECSPs

```
1   ES_1 ⇐ 1-ECSP;
2   for each q ∈ ES_1 do
3     if sup_EfIC(p ∪ q) ≥ minsup then
4       if ac(p ∪ q) ≤ maxcost then
5         将q添加到p的末尾,构成新的ECSP p';
6         输出p';
7       end if
8       if asc(p ∪ q) ≤ maxcost then
9         构建SBR|_q';
10        ECSPM-C(p', SBR|_q', minsup, maxcost);
11      end if
12    end if
13  end for
```

SPM算法——GSP、PrefixSpan和CEPN进行了对比.这些比较算法的源代码均来源于备受推崇的SPMF数据挖掘库,该库在序列模式挖掘领域具有广泛的影响力.

在实验中重点探讨了出勤性因子α、离散性因子β和辍学性因子γ的最优值,这些因素在算法效果上有着举足轻重的作用.为此,通过在[0, 1]范围内调整这些参数的值,详细记录了执行时间、内存使用情况及发现的序列模式数量.这些关键数据汇总于表4中.为了更直观地评估算法性能,引入了TMN这一指标,它综合了执行时间、内存使用量和发现模式数量,TMN值越小,表明算法性能越优越.经过仔细权衡,实验选择了α=3/6,β=2/6,γ=1/6这一参数组合,因其对应的TMN值最小.随后,进一步调整最小支持度阈值,从0.02~0.08以及0.2~0.7,对算法的性能和有效性进行全面分析.

表4 参数调整

Tab. 4 Parameters adjustment

α	β	γ	运行时间	内存开销	发现的序列模式数量	TMN
1/6	1/6	4/6	13.85	323.80	2967	12994.04
1/6	2/6	3/6	12.21	428.59	2060	10527.49
1/6	3/6	2/6	10.49	437.64	1271	5698.20
1/6	4/6	1/6	10.62	495.08	699	3489.03
2/6	1/6	3/6	14.20	675.65	2057	19272.78
2/6	2/6	2/6	11.55	438.66	1258	6224.30
2/6	3/6	1/6	10.56	375.95	657	2547.17
3/6	1/6	2/6	8.87	317.54	1244	3421.70
3/6	2/6	1/6	8.61	341.72	640	1838.88
4/6	1/6	1/6	8.55	389.24	588	1911.00

首先对比6种算法的执行时间,如图1所示,与两种广度优先算法(GSP和ECSPM-L)相比,4种基于投影的算法(PrefixSpan、ECSPM-P、CEPN和ECSPM-C)展现出了更出色的性能.这符合模式增长算法在模式挖掘领域广受认可的现状,它们以高效发现数据集中模式的能力著称.值得注意的是,由于提出的算法中引入了相关约束,这增加了额外的计算负担,导致与基准算法相比执行时间略有增加.尽管如此,ECSPM-L的性能仍略优于GSP,ECSPM-P略优于PrefixSpan,而ECSPM-C略优于CEPN.重要的是,提出的算法的主要目标并非单纯追求速度,而是通过整合相关约束来增强模式挖掘,从而提高模式发现的有效性和准确性.

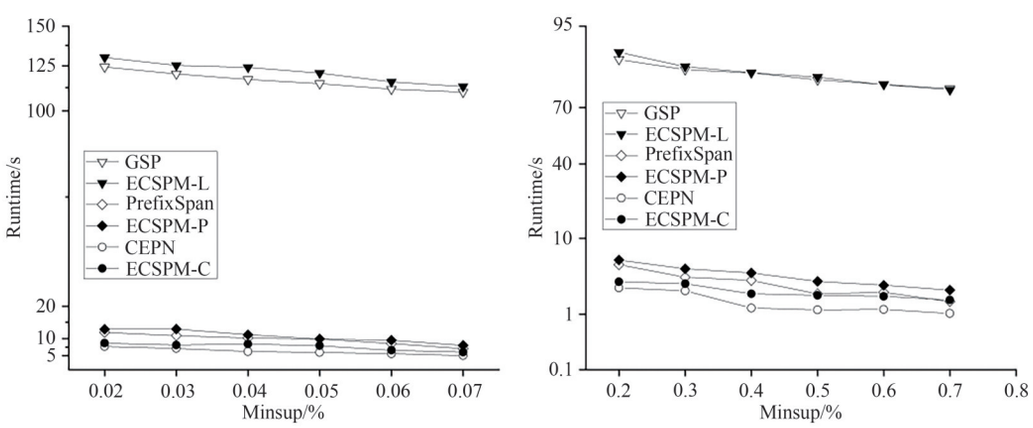

图1 运行时间比较

Fig. 1 Comparison of runtime

然后比较6种算法的内存使用情况,如图2所示.观察结果显示,在两种逐级遍历算法中,提出的ECSPM-L算法相较于GSP算法在内存消耗上更为节省.这表明本文提出的算法在内存效率方面具有优势,能够在减少内存需求的同时实现高效的模式挖掘.在基于投影的4种算法中,ECSPM-P相较于

PrefixSpan内存消耗更低，而ECSPM-C相较于CEPN也展现出更低的内存使用量. 这主要归功于本文算法的设计，它有效地减少了连接操作和不必要的数据库投影，从而显著降低了内存使用量.

接着探讨效率约束对发现序列模式数量的影响，如图3所示. 为了更清晰地展示比较结果，省略了基于ECSPM-L和ECSPM-P的某些结果，因为它们产生了相同的输出，而GSP和PrefixSpan的结果也相同. 此外，由于ECSPM-C和CEPN具有额外的输入参数maxcost，因此它们发现的序列模式与ECSPM-L和ECSPM-P不同. 从图中可以看出，ECSPM-P发现的模式数量始终少于PrefixSpan，而ECSPM-C发现的模式数量也始终少于CEPN. 这一结果突显了本研究的一个重要发现，即效率约束的引入导致了识别模式数量的减少. 这一结果源于本研究对MOOCCube数据集中固有学习特征的深入分析.

最后通过模式分析展现效率约束对模式挖掘的影响，当minsup=0.3%，p =< 大学物理2, 线性代数理论, 计算机科学和Python编程导论 >被发现是FSP，但不是ECSP. 如表5所示，随机选择两个包含p的序列S_1和S_2，可以计算出S_1对于离散性约束支持度的贡献较小，S_2对于出勤性约束支持度的贡献较小，且S_2存在辍学性行为，综合S_1和S_2对于效率约束支持度的贡献较小. 进一步从MOOCCube数据集中选择两个包含ECSP的序列，如S_3 =< 催化剂设计与制备，药理学，有机化学，药物化学，分析化学 > 和 $_4$ =< 中药学，有机化学，

生物化学，食品化学，药物化学，它们都是药学类课程，对于想学习化学、药学类课程的人来说，它

表5　包含模式p的两条序列

Tab. 5　Two sequences containing the pattern p

Id	Sequence
S_1	（组合数学，2017/8/5，2017/10/9，534，628），（大学物理2，2017/8/27，2018/1/25，471，679），（理论力学，2017/8/27，2017/11/26，513，627），（线性代数理论，2017/9/15，2018/6/3，490，589），（计算机科学和Python编程导论，2017/12/4，2018/6/19，301，436）
S_2	（大学物理2，2018/3/7，2018/5/6，120，679），（线性代数理论，2018/3/10，2018/6/20，412，627），（Linux系统管理，2018/4/11，2018/11/2，137，392），（计算机科学和Python编程导论，2018/6/30，2018/9/20，218，436）

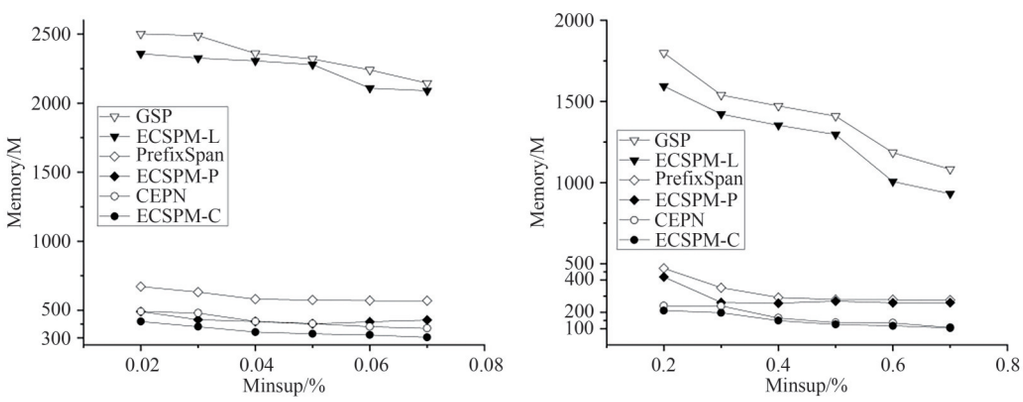

图2　内存消耗比较

Fig. 2　Comparison of memory usage

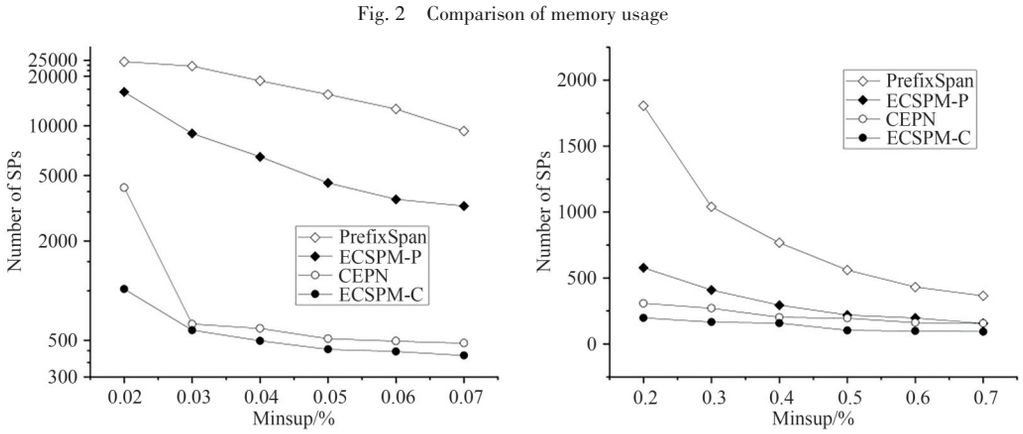

图3　发现序列模式的数量

Fig. 3　Number of discovered sequential patterns

们都是有意义的模式.上述模式分析表明,本研究所提出的效率约束可以有效地过滤不那么相关的模式,保留较有意义的模式.通过考虑与效率相关的特定约束,本研究的目标在于更深入地探究这些约束如何影响挖掘过程,并最终影响所发现模式的数量和性质.

5 结论

本研究针对MOOCCube数据集的具体学习行为进行了深入分析,并据此引入了多个维度的效率约束,包括学习序列的长度、学习注册时间、学习周期的有效时长和总时长.通过创新性地提出效率性约束的支持度(EffCS),本文成功地将这些效率约束整合至模式挖掘过程中,同时严格证明了EffCS保留了基本的向下闭合特性,确保了挖掘结果的可靠性.

基于上述理论支撑,开发了3种算法——ECSPM-L、ECSPM-P和ECSPM-C,用以发现具有效率约束的序列模式.这些算法巧妙地结合了逐级搜索空间遍历或递归投影技术,并将成本概念融入模式挖掘中,实现了对效率约束的有效处理.实验评估结果表明:相较于FSP,ECSPM能够识别出数量更少的模式,这充分验证了其在考虑和满足特定效率约束方面的有效性;同时,ECSPM在效率和内存消耗方面与经典SPM算法相当,展现了良好的性能.

展望未来,本研究将致力于开发更为高效的策略,以进一步优化ECSP的发现过程.此外,本研究计划将效率约束应用于其他模式发现任务,如高效用序列模式的挖掘,通过组合不同的约束条件,进一步提升高效用序列模式的整体效用和相关性[31-32].未来,将深入研究ECSP在在线学习平台及其他相关应用中的有效应用策略,以期为提升学习效率和优化学习资源分配提供有力支持.

参 考 文 献

[1] GLANTZ E J, GAMRAT C. The new post-pandemic normal of college traditions[A]. ACM. Proceedings of the 21st Annual Conference on Information Technology Education. New York: Association for Computing Machinery, 2020: 279-284.

[2] WEI S, WEI Y. Mining unexpected sequential patterns from MOOC data[C]//2021 IEEE International Conference on Big Knowledge. Auckland: IEEE, 2021: 434-439.

[3] JACQUELINE W, MOHAMMAD K, MARTINE B, et al. Exploring sequences of learner activities in relation to self-regulated learning in a massive open online course[J]. Computers & Education, 2019, 140:103595.

[4] ZHU X, YE Y, ZHAO L, et al.MOOC behavior analysis and academic performance prediction based on entropy[J]. Sensors, 2021, 21(19): 6629.

[5] ZHANG J, HAO B, CHEN B, et al. Hierarchical reinforcement learning for course recommendation in MOOCs[J]. Proceedings of the AAAI Conference on Artificial Intelligence, 2019, 33: 435-442.

[6] MOZHAEVA G, MASLOVA D, YAKOVLEVA K. Correlation of MOOC students' behavior patterns and their satisfaction with the quality of the course[A]. BCPED. New Silk Road: Business Cooperation and Prospective of Economic Development. St. Petersburg: Atlantis Press, 2019: 623-629.

[7] QU S, LI K, FAN Z, et al. Behavior pattern based performance evaluation in MOOCs[C]//Advances in Information and Communication: Proceedings of the 2021 Future of Information and Communication Conference. Cham: Springer, 2021: 444-452.

[8] SRIKANT R, AGRAWAL R.Mining sequential patterns: Generalizations and performance improvements[C]// International Conference on Extending Database Technology. Berlin, Heidelberg: Springer, 1996: 1-17.

[9] FOURNIER-VIGER P, GOMARIZ A, CAMPOS M, et al. Fast vertical mining of sequential patterns using co-occurrence information[C]//Advances in Knowledge Discovery and Data Mining. Tainan: Springer, 2014: 40-52.

[10] PEI J, HAN J W, MORTAZAVI-ASL B, et al. Mining sequential patterns by pattern-growth: The prefixspan approach[J]. IEEE Transactions on Knowledge and Data Engineering, 2004, 16(11): 1424-1440.

[11] NGUYEN D, LUO W, NGUYEN T D, et al. Sqn2vec: Learning sequence representation via sequential patterns with a gap constraint[C]//. In Machine Learning and Knowledge Discovery in Databases: European Conference. Dublin: Springer, 2019: 569-584.

[12] WU R, LI Q, CHEN X. Mining contrast sequential pattern based on subsequence time distribution variation with discreteness constraints[J]. Applied Intelligence, 2019, 49(12): 4348-4360.

[13] ZHANG H, LEL. Data mining method of sequential patterns for vehicle trajectory prediction in VANET[J]. Wireless Personal Communications, 2012, 117(2): 417-429.

[14] DOKO E, BEXHETI LA, HAMITI M, et al. Sequential

pattern mining model to identify the most important or difficult learning topics via mobile technologies [J]. International Journal of Interactive Mobile Technologies, 2018, 12(4): 109-122.

[15] AL-TWIJRI M I, LUNA J M, HERRERA F, et al. Course recommendation based on sequences: An evolutionary search of emerging sequential patterns [J]. Cognitive Computation, 2022, 14(4): 1474-1495.

[16] VAN T, VO B, LE B. Mining sequential patterns with itemset constraints [J]. Knowledge and Information Systems, 2018, 57: 311-330.

[17] YU J, LUO G, XIAO T. MOOCCube: A large-scale data repository for NLP applications in MOOCs [A]. ACL. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Online: Association for Computational Linguistics, 2020: 135-3142.

[18] HAN J, PEI J, MORTAZAVI-ASL B, et al. FreeSpan: Frequent pattern-projected sequential pattern mining [C]// Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Boston: Scopus, 2000: 355-359.

[19] FOURNIER-VIGER P, WU C W, TSENG V S. Mining maximal sequential patterns without candidate maintenance [C]//ADMA. In Advanced Data Mining and Applications: 9th International Conference. Hangzhou: Springer, 2013: 169-180.

[20] FOURNIER-VIGER P, GAN W, WU Y, et al. Pattern mining: Current challenges and opportunities [C]// DASFAA. International Conference on Database Systems for Advanced Applications. Cham: Springer, 2022: 34-49.

[21] WU Y, ZHU C, LI Y, et al. NetNCSP: Nonoverlapping closed sequential pattern mining [J]. Knowledge-based Systems, 2020, 196: 105812.

[22] FOURNIER-VIGER P, LIN J C, KIRAN R U, et al. A survey of sequential pattern mining [J]. Data Science and Pattern Recognition, 2017, 1(1): 54-77.

[23] FOURNIER-VIGER P, WU C W, TSENG V S, et al. Mining partially-ordered sequential rules common to multiple sequences [J]. IEEE Transactions on Knowledge and Data Engineering, 2015, 27(8): 2203-2216.

[24] CHEN E, CAO H, LI Q, et al. Efficient strategies for tough aggregate constraint-based sequential pattern mining [J]. Information Sciences, 2008, 178(6): 1498-1518.

[25] AOGA J O, GUNS T, SCHAUS P. Mining time-constrained sequential patterns with constraint programming [J]. Constraints, 2017, 22: 548-570.

[26] LIN M Y, LEE S Y. Efficient mining of sequential patterns with time constraints by delimited pattern growth [J]. Knowledge and Information Systems, 2015, 7: 499-514.

[27] DALMAS B, FOURNIER-VIGER P, NORRE S. TWINCLE: A constrained sequential rule mining algorithm for event logs [J]. Procedia Computer Science, 2017, 112: 205-214.

[28] FOURNIER-VIGER P, LI J, LIN J C, et al. Mining cost-effective patterns in event logs [J]. Knowledge-Based Systems, 2020, 191: 105241.

[29] SONG W, YE W, FOURNIER-VIGER P. Mining sequential patterns with flexible constraints from MOOC data [J]. Applied Intelligence, 2022, 52(14): 16458-16474.

[30] PRENKAJ B, STILO G, MADEDDU L. Challenges and solutions to the student dropout prediction problem in online courses [A]. ACM. Proceedings of the 29th ACMInternational Conference on Information and Knowledge Management. New York: Association for Computing Machinery, 2020: 3513-3514.

[31] RITIKA, GUPTA S K. HUFTI-SPM: High-utility and frequent time-interval sequential pattern mining from transactional databases [J]. International Journal of Data Science and Analytics, 2020, 13(3): 1-12.

[32] 冯雨,李艳红,任佳宇.多重背景下的top-k路径序列查询[J].中南民族大学学报(自然科学版),2024,43(163):835-843.

(责编　曹东,校对　姚春娜)