面向过程挖掘的日志划分技术综述

林雷蕾[1),2)] 闻立杰[1),2)] 钱 忱[1)] 宗 瓒[1)] 王建民[1),2)]

[1)](清华大学软件学院 北京 100084)

[2)](工业大数据系统与应用北京市重点实验室 北京 100084)

摘 要 过程挖掘的目标是从软件系统产生的日志数据中提取出有价值信息,用于配置或优化已实施的业务过程.与此同时,大数据、物联网等技术的发展不仅使得业务内容愈加复杂,更是加速了业务演化的速度.在此背景下,有必要对原始日志进行划分,使得事件日志通过分解而被更有效地分析,进而提升过程挖掘的质量.日志划分的宗旨是根据不同问题采取不同方法将原始事件日志划分为多个子日志,为后续的过程挖掘研究提供支撑.模型发现是过程挖掘中最重要的应用场景,而该场景面临的两大难题就是模型过于复杂和模型不正确.当前,解决这两个难题的方法分别是轨迹聚类和概念漂移,而这两类方法的本质都是对原始事件日志进行了划分.本文针对轨迹聚类和概念漂移两个分支进行归纳总结,试图厘清日志划分内容中这两个分支的异同点.接着,通过文献规约系统地对现有研究进行统计与分析,揭露了两个研究分支的发展趋势.然后,梳理了现有研究方法的主要思路,将轨迹聚类分为距离驱动、模型驱动和混合聚类三类,并将概念漂移分为单一类型和复合类型两类.最后,利用公开数据集测试不同类型算法的优缺点,并指出未来研究的发展方向.

关键词 过程挖掘;轨迹聚类;概念漂移;业务演化

中图法分类号 TP311 **DOI 号** 10.11897/SP.J.1016.2022.01946

A Survey of Log Division Technique in Process Mining

LIN Lei-Lei[1),2)] WEN Li-Jie[1),2)] QIAN Chen[1)] ZONG Zan[1)] WANG Jian-Min[1),2)]

[1)](*Department of Software, Tsinghua University, Beijing* 100084)

[2)](*Beijing Key Laboratory of Industrial Big Data System and Application, Beijing* 100084)

Abstract Process mining aims to extract the valuable information from event logs generated by software systems, which is often utilized for configuring and optimizing the ongoing business process models. Meanwhile, the development of information technologies (such as big data and the Internet of Things) not only makes process models more structurally- and behaviorally-complex, but also accelerates the speed of business evolution. Under this circumstance, it is necessary to analyze, design and simplify the original event log into sub-logs for effectively-reusable purposes. Therefore, it is more instructive for process mining to mine the process models from the sub-logs instead of the original logs. As we know, data division is to improve model performance through subsets analysis, so the purpose of log division is to divide the original event log into multiple sub-logs by adopting different methods according to different issues. The analysis of these sub-logs can provide support for process mining research, especially in process discovery scenario. It is known that process mining has three application scenarios, namely process discovery, conformance checking and process enhancement. The most crucial learning task in the process

收稿日期:2020-09-23;在线发布日期:2021-09-01.本课题得到国家重点研发计划项目(2019YFB1704003)、国家自然科学基金项目(71690231,62021002)资助.**林雷蕾**,博士,助理研究员,主要研究方向为过程挖掘、软件工程.E-mail:leilei_lin@126.com.**闻立杰**(通信作者),博士,副教授,博士生导师,主要研究领域为过程挖掘、自然语言处理.E-mail:wenlj@tsinghua.edu.cn.**钱 忱**,博士研究生,研究方向为流程文本抽取、自然语言处理.**宗 瓒**,博士研究生,主要研究方向为流程优化.**王建民**,博士,教授,博士生导师,主要研究领域为业务过程管理、人工智能.

mining domain is process discovery that is defined as the construction of a reasonable process model from the original event log. However, the model mined from the original event log in process discovery scenario is always too complex (spaghetti-like model) and inaccurate (neglect evolution). At present, the solutions to these two problems are trace clustering and concept drift, where the trace clustering can be considered a versatile solution for reducing the complexity of the mined models, and concept drift in process mining is to detect changes in event logs for improving the accuracy of process discovery. In our viewpoints, the principles of the two solutions, trace clustering and concept drift, are the same, because they both improve the quality of the mined models by dividing the original log into multiple sub-logs. In this paper, we summarize the two key branches of log division—trace clustering and concept drift, trying to clarity the similarities and differences between them. We find that trace clustering only considers the similarity between different traces in the original log, while ignoring the timestamp attribute on the traces. In contrast, the focus of concept drift research is to find out the time points (or the locations of the traces in the original log), and then divide the original log into sub-logs based on these time points. Moreover, we systematically summarize and analyze the development trend of the related studies through the literature protocol, and find that the growth trend of the concept drift in the past five years is greater than the growth trend of the trace clustering. More concretely, we classify the methods of trace clustering into three categories (i. e., distance-driven, model-driven and hybrid clustering) and classify the methods of concept drift into two types (i. e. single type and composite type). Furthermore, we use the publicly-available datasets to evaluate the advantages and disadvantages of different types of methods, and sketch some potential development directions of future research.

Keywords　process mining; trace clustering; concept drift; business evolution

1　引　言

经济全球化的冲击和多变的市场需求，迫使企业必须思考：(1) 如何利用最少资源在更短时间内生产出更多产品；(2) 如何快速优化已有的服务流程. 与此同时，信息化的发展使得企业在执行实际业务过程中产生了大量的事件日志(event log)，这些数据会以某种形式记录在信息系统中(如 csv 文件). 过程挖掘(process mining)的宗旨是充分挖掘事件日志中有价值的信息(如资源、角色、部门结构等)，这些信息可以用来部署新系统以支持业务过程执行，或作为反馈用于分析和改进已经实施的业务过程[1]. 图 1 展示了过程挖掘在实际场景中的三个经典应用[1]，包括过程发现、合规性检查及过程增强，具体如下：

(1) 过程发现(process discovery). 该场景的输入是业务执行后产生的事件日志，输出是过程模型，解决的是业务过程的自动化建模问题. 过程模型关

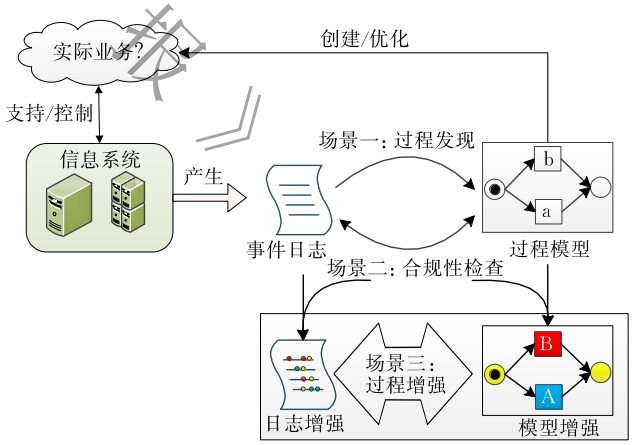


图 1　过程挖掘领域中三个经典场景

注的内容可以是单一维度的，也可以是多个维度的集成模型. 现有研究的维度主要有：关注活动执行先后顺序的控制流维度(control-flow)[2]、关注活动耗时耗费的资源维度[3]和关注业务参与角色的社交维度[4]等. 为了更好地发现日志数据中蕴含的过程模型，许多学者致力于提出高效且准确的挖掘算法：如启发式挖掘算法[5]、基于 Markov 链的挖掘算法[6]、

遗传挖掘算法[7]和 Inductive Miner 挖掘算法[8-9]. 值得一提的是,荷兰 van der Aalst 等人提出的 Alpha 算法[2]由于准确性高且易操作,成为该领域最为经典的挖掘算法. 该算法可以从日志中发现业务活动之间的顺序、选择、并发及长循环等结构,并以 Petri 网作为结果输出. 在此基础上,许多学者对 Alpha 算法进行了扩展,如解决不可见任务算法[10]、非自由选择结构(non-free-choice)算法[11]、短循环结构算法[12]和基于因果完备(causally complete)下的并发结构挖掘算法[13];

(2)合规性检查(conformance checking). 该场景的输入是日志和模型,输出是符合性度量指标或图示化的度量结果,解决的是实际执行行为(日志)与建模行为(模型)之间的共性和差异[14-15]. 例如,管理者想实施一个固定的工作流程(过程模型),而有经验的员工更愿意灵活地服务客户(日志数据). 为了权衡两方的利益,就可以用合规性检查寻找二者间共性和差异. 此外,合规性检查也可以被用来衡量过程挖掘算法的准确率. 常用来度量日志与模型之间的差异性指标有:① 拟合度(fitness)[16]. 评价模型能够重演(replay)日志行为的能力;② 精确度(precision)[17]. 评价模型行为有多少包含在日志;③ 泛化度(generalization)[18]. 评价模型有多少行为不在日志中,即抽象能力. 此外,还有专门针对模型复杂度的度量指标 complexity[19];

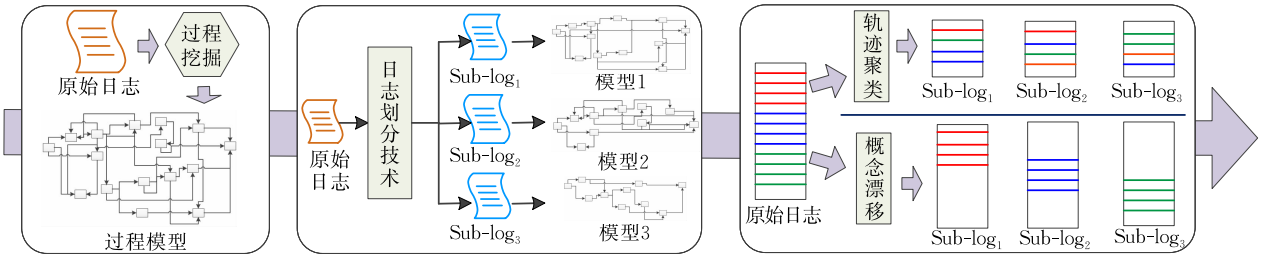
(3)过程增强(process enhancement). 该场景与合规性检查之间没有明显的界限,甚至可以看成是合规性检查场景的延伸. 输入同样是日志数据与过程模型,输出是增强后的日志或模型. 其中,日志增强是根据已有的领域知识(如模型、文档)对日志进行改进,目的是提升数据质量,从而更好发现用户感兴趣的行为模式[20-21]. 如文献[22]对同类型的活动进行映射,将语义相同的细颗粒度(fine-grain)活动映射为同名活动,降低数据的冗余度. 模型增强是利用日志中活动的多维属性信息结合领域知识,对已有模型进行改进,目的是让模型内容更加丰富[23-24]. 其中,包括增强模型的组织信息、时间信息以及模型修复(model repair)[25-26]等.

从三个应用场景来看,过程挖掘主要处理的数据是事件日志,目标是建立高质量的过程模型. 为此,在 IEEE 过程挖掘工作小组①(IEEE Task force on process mining)2011 年发布的过程挖掘宣言中,特地将日志预处理列为该领域的第一个挑战难点[27]. 经过多年的发展,日志预处理得到了众多学者的青睐,研究内容也变得相当丰富. 日志划分技术(log division)是预处理范围中的研究热点,其宗旨是将原始日志划分为多个子日志,分而治之. 本文针对模型的复杂度和正确性两个视角,对日志划分技术中的轨迹聚类和概念漂移检测进行归纳总结,主要章节安排有:第 2 节是例子与问题分析,通过实际案例引出本文综述的主题;第 3 节详细介绍本文如何系统地进行文献收集、筛选的过程;第 4 节是轨迹聚类的研究内容梳理、实验测试及展望;第 5 节是对概念漂移内容的梳理、测试及展望;第 6 节是对日志划分研究内容的一个讨论;第 7 节对本文进行总结与展望.

2 问题背景

过程发现是过程挖掘的核心应用,但在实际应用中常面临的难题是从日志挖掘出来的模型极其复杂且不可读的,不能正确反映出企业的业务情况,这种模型称之为意大利面模型(spaghetti-like model)[28],由图 2(a)所示. 究其原因,是分析人员将企业的所有行为信息展示在一个模型中,这种想法过于理想化. 所以,最好的解决方式就是利用日志划分手段对原始日志进行划分,将原始日志划分为多个子日志,分而治之(如图 2(b)所示). 日志划分的规则与实际解决的问题是相关联,比如:(1)为了解决模型过于复杂问题,可以利用轨迹聚类,将相似的业务行为聚为一个子日志,每个子日志刻画的是原始模型局部的行为;(2)为了解决因多个演化版本混在一起导致的挖掘模型不正确问题,可以对日志进行概念漂移检测,通过检测到的变化点将日志划分为多个子日志,每个子日志刻画的是业务过程的一个版本(如图 2(c)所示). 轨迹聚类和概念漂移都是属于日志划分技术的范畴,在实际解决问题中对于使用哪种技术也容易混乱. 因此,有必要对两种技术的异同点进行阐述. 如表 1 所示:二者最大的相同点是输入一个日志,输出是划分后的多个子日志. 在解决问题方面,轨迹聚类主要解决的是如何降低模型复杂度,而概念漂移解决的是由于业务演化带来的挖掘模型不正确问题. 为此,轨迹聚类研究的对象是行为相似性,利用相似的业务行为进行聚类,从而降低模型的复杂度. 相比之下,概念漂移研究的对象是行为的变化,通过发现行为的变化来找出日志中

① http://www.win.tue.nl/ieeetfpm/doku.php



(a) 实际问题　　(b) 解决方式　　(c) 不同视角

图 2　过程挖掘中实际问题(a)、解决方式(b)和不同视角(c)

的漂移点(也称变化点),进一步利用漂移点将日志划分为不同子日志,从而得到业务过程的不同演化版本.显然,轨迹聚类的视角集中在模型的静态结构上,而概念漂移则是关注模型的动态演化过程.由此可得,二者最大的不同就是概念漂移考虑了日志中的时间维度.在实际应用中,两个技术是可以交叉使用的.如企业可以先利用概念漂移检测出日志中不同版本的业务模型,然后通过轨迹聚类对同一个版本中相似行为进行聚类,使得企业能很好分析出整个业务的演化过程以及每个业务内部有哪些不同的服务行为.

表 1　轨迹聚类与概念漂移之间的异同点

异同点	轨迹聚类	概念漂移
输入输出	输入一个日志,输出多个子日志	
解决问题	降低模型复杂度	提高模型正确性
研究对象	行为相似性	行为变化
不同视角	静态结构视角	动态演化视角
时间维度	不考虑时间属性	考虑时间属性
实际应用	两者可以交叉	

3　文献规约

为了准确定位本文综述的内容,有必要对涉及的文献进行约束.

3.1　研究问题的范围

本文聚焦如何降低模型复杂度和提高模型发现的正确性两个问题,主要回答的内容包括如下:

问题 1. 在过程挖掘领域,如何在预处理阶段降低复杂度和提高正确性?

问题 2. 针对数据完整性,哪些方法是可以保留住原始数据信息?

问题 3. 现有解决模型复杂度及正确性的方法有哪些以及未来趋势是什么?

问题 1 包含两层意思:(1) 如何降低模型复杂度和提高正确性是众多领域都会面临的问题,然而本文将研究的范围限制在过程挖掘领域;(2) 高效的过程挖掘算法在一定程度上也可以复杂度或正确性问题.如模糊挖掘[29].但是,本文综述内容集中在预处理阶段的日志划分技术上.

问题 2 将本文研究的范围限定在不损失日志信息的前提下进行.如日志抽样[30]可以用部分日志代替原始日志,但是,本文认为在大数据处理技术较为成熟的背景下,没有必要通过损失原始数据信息来降低模型复杂度.另外,在提高挖掘模型的正确性上噪音处理[31-32]也是较为常见的处理手段,但不考虑的缘由有两个:一方面,现有噪音处理方法很难精准识别哪些数据是噪音.换句话说,噪音处理容易剔除掉一些有用数据,破坏原有数据的完整性;另一方面,噪音处理技术输入输出都是日志,因此,可作为其他技术的一个前置处理阶段.

问题 3 是本篇论文重点回答的问题.轨迹聚类和概念漂移是日志划分中两个不同视角,表 1 详细展示了二者的异同点.本文将分别讨论这两个主题目前的研究进展,并进行归纳分类,最后探讨一下未来发展的趋势.

3.2　搜索的关键字与数据库

为了尽可能全面地覆盖到所有相关的研究工作本文选取了采用中文和英语两种语言撰写的文献作为综述对象.其中,与本文相关的英文关键字是 process mining、log division、trace clustering、concept drift.此外,根据词根变化及说法差异,扩展搜索了 process discover、log divide、trace cluster、log clustering 及 drift detection (detect).另外,与本文相关的中文关键字是过程挖掘、日志划分、轨迹聚类和概念漂移.由于中英文对照上的差异,还扩展搜索了流程挖掘、日志切割、日志聚类、漂移检测.综上所述,搜索的关键字组合为下:

中文关键字. 轨迹(日志)聚类、概念漂移(漂移检测)+日志划分(切割)或过程挖掘(流程挖掘).

英文关键字. trace (log) clustering (cluster)、concept drift (or drift detection、detect drift)+log

division(divide)或 process mining(discover).

针对以上关键字,本文选取国内外主流的研究型数据库进行搜索:其中,英文检索的数据库是 Web of Science、IEEE Xplore、SpringerLink、Scopus、ACM、ScienceDirect.中文检索的数据库是知网、万方和维普.此外,结合 Google Scholar 和百度学术两大搜索引擎查全补漏.需要说明的是,从检索论文到撰写投稿,这段时间难免会遗漏新的研究成果.但是,基本不影响本文的结论.

3.3 选择标准

显然,即使限定撰写的母语是中文和英文,也限定了文献搜索的数据库,但要想统计及分析完所有相关内容,面临的挑战也是极其巨大的.因此,本文制定了文献筛选的纳入准则和排除准则:

纳入准则:

IN1. 研究对象是过程挖掘领域的日志数据;

IN2. 关注点是过程挖掘领域的聚类和漂移;

IN3. 文献以近十年的研究为主.

日志数据是一个很广的概念,本文着重讨论业务过程中的事件日志(IN1).如图 3 所示:展示了某保险行业在执行理赔业务中产生的日志数据.图 3 的左边是以 XES 格式存储的日志内容,一个业务过程(log)包括了很多过程案例,每个案例是业务的一次完整执行记录,对应本文中的一条轨迹(trace).每个案例又由很多个事件(event)组成,事件里面有其对应的活动名称(concept:name)、时间戳(timestamp)及执行角色(resource)等.若仅考虑日志中控制流,则 XES 日志可抽象为图 3 的右边内容 Event log=[⟨A,B,C,D,E⟩,⟨A,C,B,D,F⟩,…].依照文献[1-2]定义,日志是轨迹的集合,而每条轨迹又是由有限个事件按照执行次序排列而成.单独分析每条轨迹的话,会发现其表达形式与传统的时间序列极为相似.但是,二者的本质区别在于:(1)时间序列以连续数值为主,而事件序列以离散活动为主;(2)时间序列在连续时间点都是数值,数值前后的逻辑关系很弱.相反,事件日志是由具体业务的执行产生,所以活动执行的前后逻辑紧密相关.如图 3 所示,活动 B 的执行依赖于活动 A 的执行.

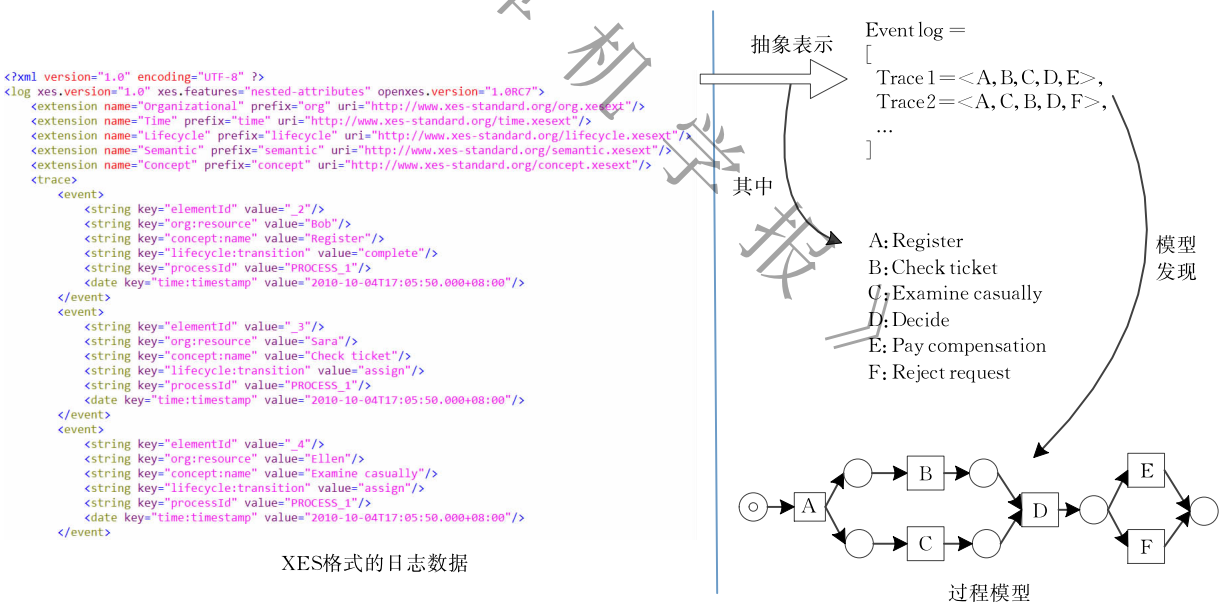


图 3 XES 格式的日志及抽象表

纳入准则 IN2 表明了本文关注的是过程挖掘领域中轨迹聚类(参见第 4 节)和漂移检测(参见第 5 节)两个内容,对于其他领域的相关内容暂不做讨论.另外,IN3 将综述的文献控制在近十年内.同时,为了帮助读者厘清脉络、加深理解,对一些经典的工作(如文献[33]),本文也会做内容分析.

排除准则:

EX1. 文章内容只有题目和摘要,或正文篇幅少于 6 页;

EX2. 撰写语言不是中文或英语;

EX3. 文章只是对已有方法的应用,没有改进.

为了保证文献的真实性和可读性,EX1 和 EX2 分别对搜索到的文献进行排除.EX3 强调研究内容不能是单纯的应用.例如,文献[34]在雾计算(fog computing)中引入了概念漂移,较好地解决了移动端 Apps 的版本问题.然而,该文检测漂移的方法来

自于文献[35]，并未对原有方法进行提升改进.

3.4　文献统计

根据前面的问题分析、搜索关键字以及选择标准，本节对检索文献进行了统计，最终选定的参考文献有轨迹聚类 33 篇，概念漂移 39 篇. 而排除准则排除了轨迹聚类文章 31 篇，概念漂移 28 篇①. 图 4 展示了参考文献在这十年间的趋势以及每两年的具体文献数量：从图中可以看出，近十年来两个方向的论文数量都在增加，从最早的 2 篇增长到后面的 11 篇. 此外，轨迹聚类的关注度在 2016～2017 年有一个很大的增幅，文献数量增长了两倍. 对比之下，概念漂移的最大增幅是在 2018～2019 年，基本上也是增长了两倍. 还有在 2016～2017 年之前，概念漂移也都在保持增长的趋势，只是增幅不大. 本文认为，概念漂移关注度在未来两年应该还是会超过轨迹聚类，但是增幅会放缓，甚至降低.

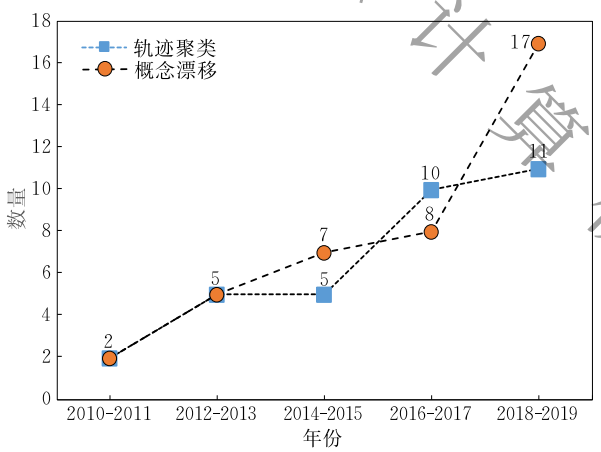


图 4　近十年的研究趋势

4　轨迹聚类

本节针对轨迹聚类的现有研究进行综述，主要包括三个内容：首先，对目前已有的轨迹聚类算法进行综述以及分类；其次，利用真实数据对算法进行测试；最后，总结轨迹聚类的研究趋势.

4.1　聚类算法综述

Greco 等人在 2004 年利用层次聚类挖掘工作流日志中的全局约束(global constraints)[36]算是轨迹聚类的开山之作. 该文指出任务的先后顺序只是一种局部约束(local consraints)，而设计者往往更加感兴趣工作流中常用的执行模式，因此通过轨迹发现的不同簇可以看作是不同执行模式，每种模式代表着一种特殊应用的全局约束. 经过多年发展，轨迹聚类在解决模型行为过于复杂的难题上发挥了较好的效果[33,37]. 轨迹聚类的本质就是找出日志中不同聚类簇，每个簇单独展示了一个特定的过程模型(即完整模型中的部分行为)，通过刻画局部行为来降低模型的复杂度. 例如，医院的诊疗流程是一个非常复杂的过程，而对于肿瘤治疗的专家或部门来说，他们只需知道肿瘤问诊的相关流程就可以进一步优化服务了. 为此，可以利用轨迹聚类从医院系统中聚类出关于肿瘤治疗的日志数据，接着运用已有的模型发现算法[5,8,12]进行自动化建模，为肿瘤部门的诊疗服务提供优化手段.

总体来说，轨迹聚类的算法主要分为三类：(1)基于距离驱动的聚类算法[33,37-43]；(2)基于模型驱动的聚类算法[19,44-48]；(3)集成距离驱动和模型驱动的混合聚类算法[49]. 如图 5 所示，距离驱动的聚类算法包括三个核心内容：首先对日志轨迹进行刻画，将轨迹中的特征表示出来(如轨迹轮廓，traces profile). 接着，通过编辑距离等方法计算出轨迹之间的相似度. 最后，基于传统的聚类算法(如 k-means)将轨迹划分为不同的类簇. 相比之下，模型驱动的聚类算法就另辟蹊径，直接从模型出发，然后通过模型与轨迹之间的评价指标(拟合度、复杂度等)来确定轨迹属于哪个类簇. 其中，训练的模型可能是直接输入的，也可能通过日志中的部分数据集间接得到的. 如果是间接得到的模型，那么如何选择日志中的轨迹也是个难点(如贪心策略). 显而易见，距离驱动的聚类算法考虑到了每条轨迹与其他轨迹的综合信息，即全局考虑了行为的相似度. 而模型驱动算法仅考虑当前轨迹与模型的评价得分，此类算法通常采用随机策略或者贪心策略获取最优解，容易陷入局部最优. 为此，混合聚类算法综合考虑了两者的利弊，集成了两者优势对日志轨迹进行聚类.

(1)距离驱动的聚类算法

根据轨迹刻画的不同，距离聚类可以分为具体型和抽象型两种. 具体型的刻画方法就是不对轨迹做任何转换，直接在轨迹上计算相似度. 如早期的 Bose 等人[33]将日志中的所有活动看成是字母表，则每条轨迹为一个字符串. 然后，使用最小编辑距离(levenshtein distance)计算轨迹之间的相似度. 最后，运用 k-means 进行轨迹聚类. 该方法(Maximal Repeat，简称 MR)的优点就是简单高效，缺点是难以解决循环结构.

① https://pan.baidu.com/s/1B1Q1WdIpsPZT_nArY8umQA 提取码：THUL

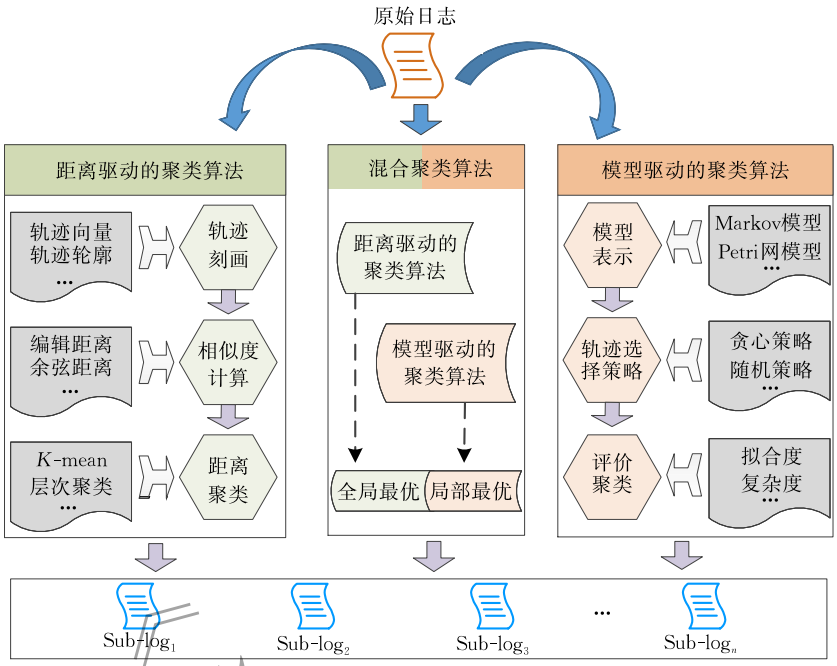


图 5 三种不同类型的聚类算法

相比之下，抽象型的刻画方法在聚类算法中较为常见. 即将轨迹刻画为向量，然后再基于轨迹向量计算对轨迹之间相似度. 其中，轨迹轮廓又是用的最多抽象方法. 轨迹轮廓是轨迹不同视角的向量聚合，它允许用户从不同视角来刻画轨迹，并且可以将所有视角的内容拼接在一起. 如表 2 所示，展示了从活动视角(activities)和角色(originator)视角来刻画图 3 中的日志轨迹. 在活动轮廓中，数值 1 和 0 分别表示的是当前活动在轨迹中出现与不出现. 在发起人轮廓中，数值代表的是当前角色在轨迹中参与了多少个事件. 此外，用户还可以定义其他视角轮廓，如用于刻画活动之间紧邻关系的变迁(transition)轮廓、用于刻画事件执行时间的性能轮廓等等. 基于轨迹轮廓的抽象表示，就可以对轨迹之间的相似度进行计算、聚类. 文献[37]在活动轮廓、活动模式轮廓以及变迁轮廓的基础上对轨迹进行聚合层次聚类(agglomerative hierarchical clustering)，聚合的方式是利用欧式距离取两个簇之间的最远距离作为簇间距离进行聚合，即全链方式(complete linkage). Montani 等人考察实际日志发现很多活动的属性是不能被量化的，如内容相同但名称不同、活动之间存在重叠等. 为此，定义了近邻图距离(neighbors-graph distance)来计算轨迹之间的相似度[38]，最后使用非加权组平均法进行层次聚类. 文献[39]将轨迹相似度映射为活动向量矩阵和变迁向量矩阵两个部分，然后利用余弦距离来计算每个部分的得分，最后利用谱聚类算法(spectral clustering)进行聚类. 与前面工作不同的是：①变迁向量矩阵与变迁轮廓的本质区别在于变迁向量考虑了弱紧邻关系；②考虑了不同视角的权重. 作者通过大量实验得出，模型的复杂主要源自于任务之间的关系复杂，并非任务数量过多. 因此推荐权重比是活动相似度为 0.3，变迁相似度为 0.7. Appice 在文献[40]中指出，如果只是简单将不同视角的向量拼接在一起计算相似度，存在两个问题：第一，忽略了不同视角之间的相互影响；第二，简单拼接所有向量容易引起维度灾难. 为此，引入协同训练策略(co-training strategy)得到不同视角的相似矩阵，接着利用欧氏距离计算轨迹相似度，并且采用 k-medoids 算法代替 k-means 进行聚类.

表 2 针对图 3 中日志数据的轨迹轮廓示例

轨迹 ID	活动轮廓						参与角色			
	A	B	C	D	E	F	Bob	Sara	Ellen	…
Trace 1	1	1	1	1	1	0	1	1	2	…
Trace 2	1	1	1	1	0	1	1	2	1	…
Trace 3	1	1	1	1	1	0	2	1	0	…

Delias 等人指出现有轨迹聚类算法在相似度判定上都是非正即负，过于武断. 因为实际情况是即使考虑了多个视角(如活动、变迁、时间等)，也不能全面且准确地刻画业务模型. 为此，提出了模糊聚类方法[41]. 如图 6 所示，作者根据三个阈值(v_i, q_i, s_i)定义第 i 个视角的四种相似性程度(也称伪序). 然后，通过四种相似性计算出轨迹之间的一致性(concordance)和不一致性(discordance)两个值.

最后，取两个值中的最小值（即 min{concordance，1−discordance}）作为轨迹的距离，并采用聚合层次聚类算法进行轨迹聚类.

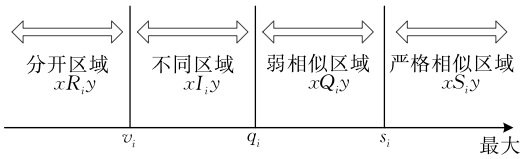


图 6　相似性的伪序

文献[42]为提高聚类效果，在已有的方法上融入专家知识（expert knowledge）. 文章描绘了三种不同的专家知识聚类策略：① 专家驱动初始化（expert-driven initialization）. 即在算法初始化中融入专家知识. 如先利用专家知识对小量数据集进行 k-means 或 k-nn 聚类，这样可以很好控制质心数量，然后在此基础上进行大规模数据集聚类；② 约束聚类（constrained clustering）. 聚类过程中利用专家知识调整轨迹聚类内容，如哪两个元素必须分开，哪些内容放在一起要惩罚等；③ 完整专家方案（complete expert solution）. 在自动获取专家知识的基础上，实时调整聚类内容. 此外，聚类完成后，也可以根据专家知识重新聚类. Lu 等人认为完整专家方案的实施过程较为困难，为此提出了半监督学习的轨迹聚类算法，只需要专家提供少量的样本数据集，就能自动完成轨迹聚类[43].

（2）模型驱动的聚类算法

模型驱动与距离驱动的轨迹聚类算法的主要区别是引入了聚类的参照物（即过程模型）. Ferreira 等人基于一阶马尔科夫模型进行轨迹聚类[44]，其主要内容是先初始化马尔科夫模型的参数，然后通过极大似然估计算法（Expectation-Maximization，EM）不断调整参数模型直至收敛. 在训练过程中，随机选择轨迹进行概率计算. 该方法虽然能解决轨迹信息不完整（如轨迹 ID 缺失）问题，但是严重受限于轨迹的选择次序. De Weerdt 等人认为基于距离聚类仅考虑了轨迹之间的相似性，而聚类好坏的评价指标又是拟合度或复杂度，两者之间存在偏差，即聚类偏好（clustering bias）和评价偏好（evaluation bias）之间的偏差[19]. 为此，文献[19]提出了基于启发式挖掘算法的模型驱动聚类算法，核心步骤是：

① 根据频率（频率高的轨迹变量）或者距离（距离近的轨迹变量）选择轨迹样本，对样本进行启发式挖掘，得到启发式模型；

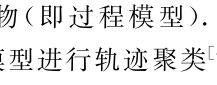
② 计算模型与轨迹样本之间的拟合度. 如果小于给定目标拟合度 tf，则样本为一个簇，跳转到第③步. 如果大于给定的目标拟合度 tf，则重复①、②步；

③ 判定簇的数量是否达到指定数量 nb. 如果等于 nb，则剩下的轨迹全部归为一个簇，算法终止. 如果小于 nb，重复①、②步.

由以上步骤可知，整个算法只负责控制当前簇产生的模型质量（贪心策略），没有整体考虑所有轨迹的影响，容易陷入局部最优. 同时，由于频繁的对样本进行挖掘、评价，算法的时间复杂程度明显高于其他聚类算法.

文献[45]与前面模型驱动的聚类方法不同点在于：① 模型表示不同. 先是基于启发式挖掘算法得到启发模型，然后转为 Petri 网模型；② 指标不同. 评价模型与轨迹之间的指标综合考虑拟合度及复杂度两方面；③ 划分簇的策略不同. 文献[19]是顺序选择日志中轨迹，达到指标就划分为一个簇. 而文献[45]是通过轨迹中是否包含共同模式来划分簇. 首先，找出轨迹中活动的不同模式 $pattern=\{p_1,p_2,\cdots,p_n\}$. 然后，根据轨迹是否含有某个模式 p_i，将日志划分为两部分，计算指标得分 $score_i$. 接着，选择所有模式中得分最高的模式来将日志一分为二. 最后，再分别对两个划分的轨迹集合进行迭代.

表 3　ATC 和 AMSTC 两种方法对图 3 模型和日志的对齐结果比较

方法	质心	日志轨迹	距离	簇
ATC	〈A,B,C,D,E〉	〈A,B,C,D,E〉	0	i
		〈A,C,B,D,E〉	1	$i+1$
AMSTC		〈A,B,C,D,E〉	0	j
		〈A,C,B,D,E〉	0	j

Chatain 等人提出了基于对齐方式的轨迹聚类算法[46-47]. 作者认为业务模型在很多场景下是可以获取的. 因此，可以通过已知模型产生轨迹，把产生的轨迹当做聚类质心，通过不同的质心对已有的日志进行轨迹聚类（Alignmennt-based Trace Clustering，简称 ATC）[46]. 其中，每个质心都满足局部覆盖（partial covering），即每个质心都是从模型开始活动到最终活动的执行结果，并且与给定的日志距离小于给定阈值 d. 该聚类方式虽然新颖，但是无法克服模型中存在并发、循环的情况. 为此，作者在 ATC 基础上扩展了基于模型子集对齐的聚类算法（Alignment and Model Subnet based Trace Clustring，简称 AMSTC）[47]. 如表 3 所示：令〈A,B,C,D,E〉为图 3 模型的一个质心，则利用 ATC 算法对轨迹〈A,B,C,D,

E〉和〈A,C,B,D,E〉聚类,会得到两条轨迹与质心的距离不一样(分别为 0、1),因此两条轨迹会分在不同的簇(即簇 i 和簇 $i+1$).但是,这两条轨迹是明显是由于模型并发结构产生,同属于一个模型,应该被划分在同一个簇.为此,AMSTC 是根据图 3 产生了一个模型子集(即包含选择分支的其中一部分),然后利用模型子集与两条轨迹进行对齐(距离都为 0),所以会把两条轨迹聚在一个簇(即簇 j).基于对齐方式的轨迹聚类本质上都是将过程模型作为参照物,其中 ATC 是利用模型产生的轨迹作为参照,AMSTC 是利用模型的子集作为参照.该方法的出发点是亮点也是挑战点,即如何从已知的模型中找到质心是个难题(如非结构化的模型).

文献[48]将日志中的异常数据称为微过程(micro-process),并认为通过聚类来发现这类数据的共性模式,有助于未来业务的调整.为此,作者的解决步骤包括:第 1 步,先对所有日志挖掘,产生 Petri 网模型;第 2 步,将日志中所有轨迹与 Petri 网模型进行轨迹对齐,拟合度低过设定阈值的轨迹被认为是异常数据,即微过程;第 3 步,随机抽取微过程与 Petri 网模型进行距离计算,计算的内容是轨迹中每个活动与模型中每个活动的算术平均距离;最后一步,利用层次聚类把微过程划分为不同类簇.该方法能充分挖掘日志中包含的所有行为信息(包括异常行为),缺点是面对非结构化模型,无法准确判定哪条轨迹是异常行为.

(3)混合聚类算法

由以上分析可知,距离驱动的轨迹聚类算法全局考虑了所有轨迹之间的相似性,但聚类效果可解释性差[33,37].模型驱动的轨迹聚类算法以模型为参照,聚类结果可解释性强,主要缺点是算法复杂度高,为此常用贪心策略或随机采样来获取近似最优解[19,45].De Koninck 等人在前人基础上提出了基于超实例(super-instances)的混合聚类算法(Trace Clustering using Super-Instrances,简称 TraCluSI)[49].算法核心内容分为三步:第 1 步,距离驱动聚类.先用活动轮廓等特征刻画轨迹,然后利用 k-means 算法进行过聚类(over clustering),将原有日志聚成超多个簇(如 400 个);第 2 步,选择簇的代表轨迹,即超实例.两种策略:① 选择簇中频率最高的轨迹作为超实例;② 选择簇的中心点作为超实例;第 3 步,利用文献[19]中的模型驱动算法对超实例进行聚类得到新的簇,然后再将超实例代表的原有簇中的其他轨迹添加到新簇中.

4.2 评估对比

前面章节概述了当前轨迹聚类算法的研究内容,由于篇幅限制,还有少数研究工作没有被提及(参见本文第 3 页脚注①).幸运的是,已阐述的工作基本上都囊括了该研究方向的主要枝干.本节内容首先是对前面的内容进行一个汇总分析,然后从中选出具有代表性的算法对公开数据进行测试.

表 4 展示了现有轨迹聚类算法的具体研究内容,

表 4 轨迹聚类算法的综述

类型	文献	轨迹刻画	相似度计算	聚类方法	优点	缺点
距离驱动	[33]	字符串	最小编辑距离	k-means	简单、高效	无法处理循环结构
	[37]	轨迹轮廓	欧式距离	聚合层次聚类	多视角	不考虑多视角权重
	[38]	字符串+属性数值	编辑距离+邻图距离	UPGMA-层次聚类	数值及非数值的相似性	忽略上下文信息
	[39]	活动矩阵+变迁矩阵	余弦距离	谱聚类	活动上下文及多视角权重	无法处理非结构化模型
	[40]	轨迹轮廓	欧式距离	k-medoids	协同训练策略	不考虑多视角权重
	[41]	相似矩阵	欧式距离	聚合层次聚类	模糊聚类	准确率低
	[42]	轨迹轮廓	欧式距离	k-means 或 k-nn	融合专家知识	预备知识获取困难
	[43]	频率序列模式	参数计算	样本训练	半监督学习	视角单一,调参难
类型	文献	模型表示	轨迹选择	评价方法	优点	缺点
模型驱动	[19]	启发式模型	贪心策略	拟合度	引入模型挖掘方法	局部最优、耗时
	[44]	一阶马尔科夫链	随机策略	模型收敛	适用多场景(如信息缺失)	模型训练受限于轨迹选择次序
	[45]	Petri 网	贪心策略	拟合度+复杂度	评价更为全面	时间复杂度高
	[46]	Petri 网	随机策略	轨迹对齐	通过模型生成轨迹作为类簇质心	无法处理并发结构
	[47]	Petri 网	随机策略	轨迹与模型对齐	模型子集作为质心	面对非结构化模型,复杂度极高
	[48]	Petri 网	随机策略	轨迹与模型对齐	充分利用异常行为的信息	无法准确判断异常行为
类型	文献	距离聚类	簇刻画	模型聚类	优点	缺点
混合聚类	[49]	轨迹轮廓+k-means	超实例(频率或质心)	启发式模型+F-score	混合聚类	参数敏感

以及各自的优缺点.不仅是对前面轨迹聚类算法的一个归纳总结,也是一个补充.针对表 4 的分析、整理,得出以下发现:

(1)从聚类方向上,距离驱动的算法多为从下至上[37-38,41],即从小的簇到大的簇.而模型驱动的算法多为从上到下[19,45];

(2)在实现方面,距离驱动的算法比较青睐于利用欧式距离计算相似度[37,40-42].同时,模型驱动的算法在选择轨迹的策略上,基本为随机策略或贪心策略的方式[19,44-47].

(3)在关注点上,研究的类型从距离驱动向模型驱动,最后到混合聚类.其中,距离驱动又从无领域知识[37-39]逐渐向引入专家知识的半监督学习发展[42-43].

为了进一步分析三种类型算法的优缺点,本文分别从每种类型算法中选择一个算法进行对比(MR[33]、ActiTrac[19]和 TraCluSI[49]).选择这三个算法的理由是:(1)不同类型.三个算法分别代表了轨迹聚类的一类解决方案;(2)公开源码.这三个算法都在公认的开源平台 ProM[50]上发布源码,保证了实验的可重复性和可靠性.评价聚类效果的指标选用了拟合度和复杂度.其中,拟合度刻画的是轨迹与模型之间的吻合度,越高越好.复杂度度量的是划分后模型的复杂性,显然是越低越好.衡量两个指标的方法很多[37-38,45,49],本文分别选用了连续语义拟合度 *ICS-Fitness*[7]和库所/变迁连通度 *P/T-CD*[19]来计算拟合度和复杂度.

定义 1(Improved Continuous Semantics Fitness, *ICS-Fitness*).　令 L 为非空的日志数据集,令 CM 为过程模型,则二者之间的 *ICS-Fitness*,$f:\mathcal{L}\ \mathcal{CM}\rightarrow(-\infty,1]$ 定义为

$$f(L,CM)=\frac{allParsedActivities(L,CM)-punishment}{numActivitiesLog(L)} \tag{1}$$

其中,惩罚因子 *punishment* 定义为

$$punishment=\frac{allMissingTokens(L,CM)}{|L|-numTracesMissingToken(L,CM)+1}+\frac{allExtraTokensLeftBehind(L,CM)}{|L|-numTracesExtraTokensLeftBehind(L,CM)+1} \tag{2}$$

在上述定义中,*allParsedActivities* 函数计算的是在日志 L 中可以被模型 CM 重现的所有活动数量.*allMissingToken* 函数返回的是每条轨迹逐一与模型对齐过程中缺少的标识(Token)数,而 *allExtraTokenLeftBehind* 则给出未被使用的多余标识数.$|L|$ 为日志中包含的轨迹条数,*numActivitiesLog* 计算的是日志中活动的总个数,*numTracesMissingToken* 返回的是对齐中存在缺失标识的轨迹数量.

直观上看,*allMissingToken* 的数量反应了日志中并发行为与模型中互斥结构的不吻合现象,而 *allExtraTokenLeftBehind* 的数量恰好相反,反应的是日志中的互斥行为与模型中并发结构之间的不吻合之处.为了避免聚类过程中不同大小数据集对评价指标的影响,本文使用加权平均拟合度进行打分,其定义为

$$f_{avg}=\sum_{i=1}^{N}(n_i\cdot f_i)\Big/\sum_{i=1}^{N}n_i \tag{3}$$

其中,N 为聚类个数,n_i 为每个聚类所包含的轨迹数.

定义 2(Place/Transition Connection Degree, *P/T-CD*).　对于含有 $|a|$ 条弧,$|P|$ 个库所,$|T|$ 个变迁的 Petri 网过程模型,其 *P/T-CD* 定义为

$$P/T\text{-}CD=\frac{1}{2}\left(\frac{|a|}{|P|}+\frac{|a|}{|T|}\right) \tag{4}$$

从定义 2 可看出,*P/T-CD* 实际上是算每个库所平均对应的弧数与每个变迁平均对应的弧数.同理,本文仍然使用加权平均数来计算复杂度指标:

$$P/T\text{-}CD_{avg}=\sum_{i=1}^{N}(n_i\cdot P/T\text{-}CD_i)\Big/\sum_{i=1}^{N}n_i \tag{5}$$

其中,$P/T\text{-}CD_i$ 为各个聚类对应模型的连通度值.

此外,本文选用了两个公开数据集对聚类算法进行测试,如表 5 所示.其中,数据集 1 是来自荷兰拉博银行的业务数据,数据集 2 是金融行业的贷款申请业务数据.两个数据集的量级不同,所以能更好地测试不同类型的算法能力.

表 5　实验采用的日志数据概况

编号	来源	轨迹数量	事件数量
数据集 1	BPIC2014①	522	3451
数据集 2	BPIC2017②	31509	1202267

图 7 给出了两个数据集在三种不同类型算法聚类下的对比.整体上讲,随着聚类个数的增加,三种类型的拟合度都会增加,复杂度也会降低.然而聚类个数多,拟合度不一定就高.以图 7(a)中 ActiTrac 算法为例,当聚类个数为 6 时,其拟合度比 5 还低.本

① Http://www.win.tue.nl/bpi/doku.php?id=2014:challenge

② Http://www.win.tue.nl/bpi/doku.php?id=2017:challenge

质原因是模型驱动的聚类算法，只考虑了当前轨迹是否是局部最优，不考虑该轨迹对全局得分的影响. 相比之下，距离驱动的聚类算法 MR 虽然整体拟合度增长缓慢，但是基本保持上升的趋势. 另外，混合聚类算法 TraCluSI 虽然集成了两者优势（先聚类，再从每个类中选择一个质心计算拟合度指标），但如果质心选择不好也会导致拟合度降低. 换句话，随着聚类的增加，只要有一个质心选择不好，那其代表的簇中的所有轨迹都会划分错误. 在复杂度上面的表现，数据集的量级对算法有一定影响. 如图 7(c)中，TraCluSI 的复杂度一直都低于其他两个算法，且模型驱动的聚类方法 ActiTrac 复杂度也优于距离驱动的聚类算法 MR. 但是，在数据集 2 的表现中，MR 的模型复杂度下降比 ActiTrac 快. 根本原因是，ActiTrac 会产生很多小规模的类簇，引发了聚类大小严重失衡问题. 如在图 7(d)中，当聚类个数达到 30 个时，ActiTrac 会产生很多个小规模的划分（最小的簇只有 84 条轨迹，轨迹占比 0.26%），而最后一个簇的轨迹占比 51%且复杂度极高. 相比之下，MR 产生的聚类就相对平衡一些，有效降低了模型的复杂度.

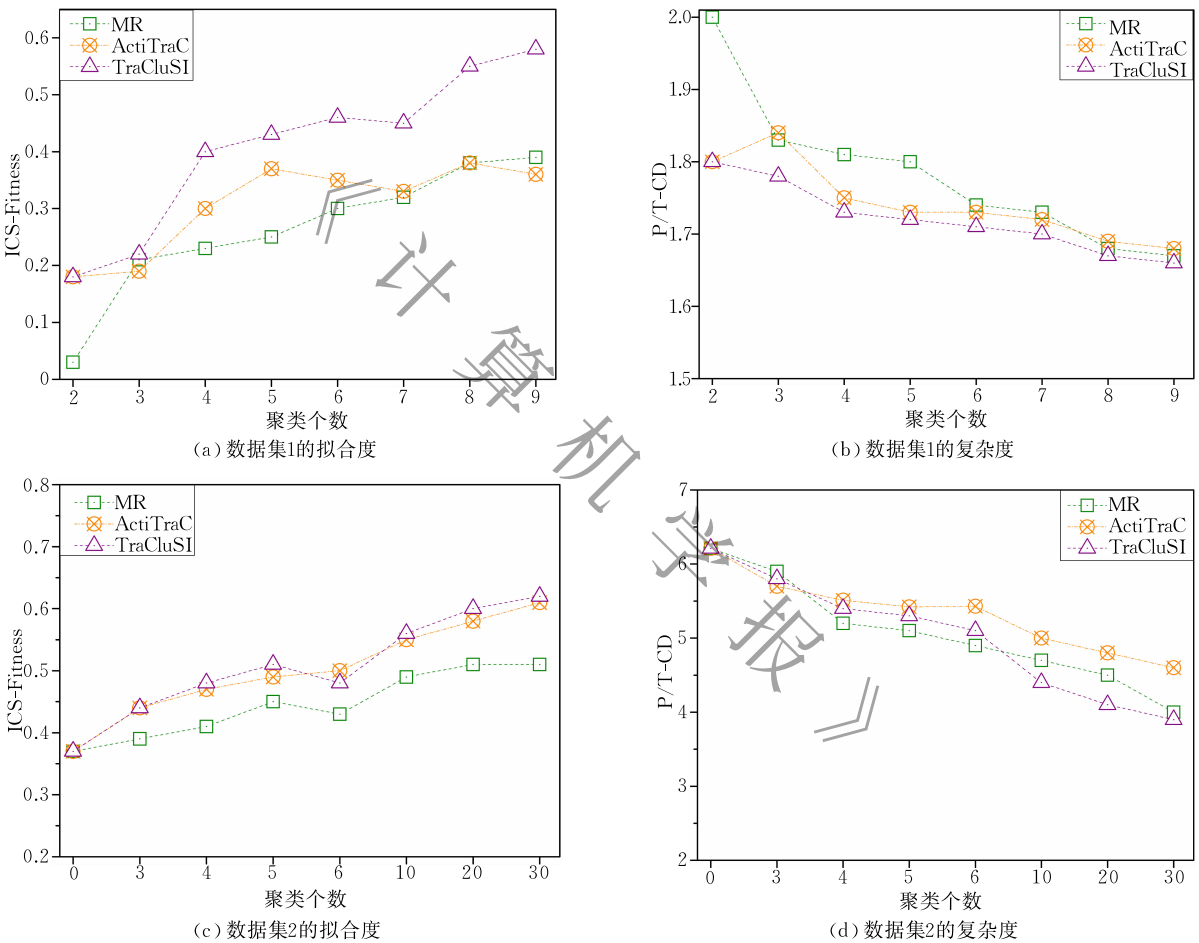


图 7 三种算法在不同数据集上的拟合度及复杂度

在整个实验过程中，耗时情况是 TraCluSI＜MR＜ActiTrac. ActiTrac 整个计算过程需要不断更新模型，重新计算轨迹与模型的拟合度，所以耗时较长. 而相比于 MR，TraCluSI 虽然也用了模型驱动聚类的策略，但是其输入的不是所有轨迹，而是前期通过聚类得到的质心. 以数据集 2 为例子，整个日志包含了 31509 条轨迹，通过 TraCluSI 的第一步聚类后，会得到的是 50 个质心（即 50 条轨迹）. 然后，再用这 50 条轨迹进行模型驱动的聚类. 所以，TraCluSI 在模型驱动聚类阶段，其轨迹使用数量已经降低了 99%.

4.3 小 结

针对 4.1 节的方法梳理以及 4.2 节的实验分析，这部分内容将对轨迹聚类进行一个小结以及展望.

(1) 现有轨迹聚类的研究趋势

① 现有研究从单一的维度（如控制流）逐渐向多维度（资源、时间等）协同训练发展[40]. 控制流是

业务流程的重点分析对象，通过分析其前后依赖关系，可进一步发现流程的瓶颈.然而，日志中还包含了业务的其他信息如执行时间等，充分利用这些信息，可以让分析人员多方位、多层次地了解业务内容.文献[51]以日志中的角色为中心，进行轨迹聚类.首先，利用部分轨迹建立局部过程模型(local process model).然后，基于局部过程模型来建立每条轨迹的角色向量.最后，根据角色向量的距离进行轨迹聚类；

② 聚类不再是单一地非此即彼的二分类问题，而是允许模糊聚类的存在[41].传统轨迹聚类过程中会根据相似性指标将轨迹严格划分在某个类簇，而模糊聚类给出了相似性的不同程度指标，分析人员可以更好把控聚类效果，使得聚类更加人性化；

③ 聚类的可解释性明显提高[42-43].早期的轨迹聚类人为地默认轨迹的相似性指标，如拟合度.但是，用户并不具备领域知识，无法理解最终聚类结果，也很难判定好坏.为此，引入专家知识来指导轨迹聚类，能有效避免指标导向，合理地提升聚类的可解释性.

(2) 未来的研究方向

由 4.2 节可知，现有三种类型的聚类算法，拟合度及复杂度还有很大的提升空间.如图 7(c)、(d)所示，拟合度最高也就在 0.6 左右，相对于原始日志的 0.37，才提升了 23%.在复杂度方面，即使划分了 30 个类簇，P/T-CD 的值还在 4 左右.换而言之，每个节点还存在 4 条边，对于流程管理来说，这个复杂度还是很高.所以，本文认为轨迹聚类还大有作为，具体可以从以下三个方面考虑：

① 在算法层面，更加全面地表示轨迹.以距离驱动的聚类算法为例，无论是 k-means 聚类[42]还是层次聚类[38]，在很多场景下差异性不大，而应该考虑从源头(即轨迹刻画)着手.现有的轨迹刻画无论是字符串还是轮廓，都是人为赋予的表示方法，带有主观性.如果能更加客观且全面地表示出轨迹内容，则聚类效果会更上一层楼.这个方向可以借鉴机器学习方向的相关工作[52]，如文献[53]已成功利用自然语言处理中的词向量的表示方法，将轨迹或者日志片段表示为向量，并在聚类上测试了效果有所提升；

② 在策略层面，尝试提出全新的架构.混合聚类[50]表面上是综合考虑距离驱动聚类及模型驱动聚类的优缺点，但实质上还是先后顺序.即混合聚类是简单地把两者按顺序执行方式拼凑在一起(先距离驱动聚类，再模型驱动聚类)，而没有在聚类每一步综合考虑两者的优缺点.Bagging 是一种集成学习的并行学习架构，其目的是尽可能放大分类器之间的差异[54].对比之下，距离驱动关注相似性而模型驱动关注评价指标，两者满足了“好而不同”的基础.因此，可以借鉴集成学习的思想，提出新的框架保证聚类过程兼顾了二者的优点；

③ 在应用层面，多方位地考虑交叉应用.现有轨迹聚类的方法还是只盯在模型结构层面，不考虑时间演化层面.因此，可以结合概念漂移应用，在聚类的同时考虑时间维度的模型演变.例如，文献[55]在轨迹聚类基础上引入了时间维度的可视化，实时监测每个轨迹类簇的动态变化.如簇中的轨迹数量是否增加或减少等.另外，不频繁行为的过滤[32]，也是可以作为多层次聚类的应用方案.即先利用频繁集将日志划分为多个层次，再在每个层次上进行相似度聚类，应该能提升聚类效果.

5　概念漂移

本节针对日志划分主题下的概念漂移分支进行综述，主要包括三个内容：首先，对目前已有的概念漂移算法进行综述以及分类.其次，利用人工合成数据及实际数据对已有算法进行评测.最后，总结概念漂移方向的研究趋势.

5.1　漂移算法综述

企业的升级或者需求的变化，都会导致业务流程发生变化，而记录业务执行过程的事件日志也会存在演化现象.Bose 等人对 100 多个企业进行跟踪研究，发现日志数据往往包含了多个演化版本的业务行为[56-57].即一个系统日志数据不止对应一个业务过程.因此，通过分析日志中漂移点，可以：(1) 将隶属于不同业务的数据划分开，进而提高后续的模型挖掘及业务过程分析；(2) 监控业务异常变化，及时作出响应.当前，过程挖掘中存在概念漂移的类型主要包括四种[56-59]：突发漂移(sudden drift)、渐变漂移(gradual drift)、增量漂移(incremental drift)和周期性漂移(recurring drift).图 8(a)表示，业务模型 1 在时间点 t_1 由模型 2 替代，这种突发变化称之为突发漂移.如紧急事故的响应机制.图 8(b)展示的是从时间点 t_1 开始，模型 2 逐渐代替模型 1(两者同时运行着)，并在时间点 t_2 完全取代业务模型 1，这种变化称之为渐变漂移.如新老政策的逐步过渡.此外，还有增量漂移(图 8(c))和周期性漂移(图 8

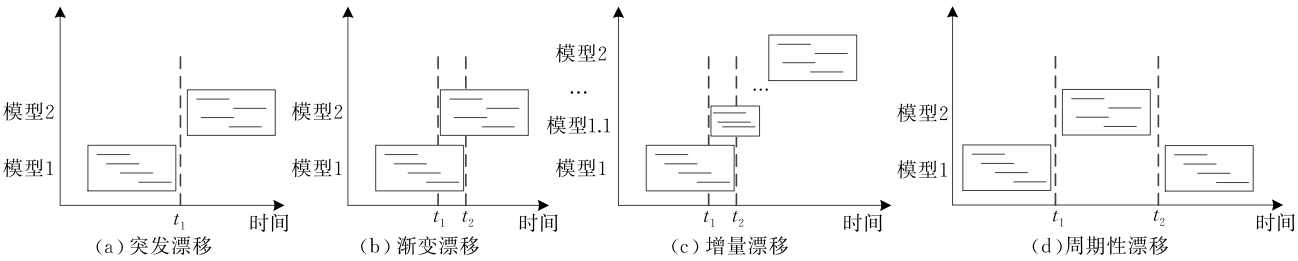


图 8 过程挖掘中四种漂移类型

(d)),其中增量漂移是在原有业务模型的基础上一点点的改变,而周期性漂移表示的是相同业务在不同时间段内的切换,如淡季与旺季的服务流程.根据算法解决漂移的类型不同,将漂移检测算法分为两类进行讨论:(1)单一类型的检测算法.即这类算法只能解决四种漂移类型中的某一种;(2)复合类型的检测算法.即这类算法能从日志中检测出多种漂移类型.

(1)单一类型的检测算法

文献[56]针对日志中突发漂移现象,提出了利用假设检验(hypothesis test)来判定两个种群(population)是否有差异的策略,进而找出日志中的变化点.如图 9(a)所示:首先,由用户指定一个参数 s 用于确定两个种群的大小,种群中每个个体是日志中的一条轨迹.接着,分别对两个种群中的轨迹进行特征抽取.作者定义了两种特征供用户自己选择:全局特征(global features)和局部特征(local features).全局特征有两个,分别是关系类型计数(Relation type Count,RC)和关系熵(Relation Entropy,RE).计算局部特征时,需要指定一个大小为 l 窗口,然后计算窗口中关系类型的计数(Window Count,WC)和两个活动之间的交叉熵 J-Measure.最后,在以上特征的基础上,对两个种群进行假设检验.作者分别用 Hotelling T^2 test 和 K-S test 对全局特征和局部特征进行了测试.注意,两个种群是不重叠的,且每检测完一次,同时往后移动一步.每移动一步,都会检测到一个 P-value,如图 9(b)所示.如果发现 P-value 小于显著性水平 α(significance level),则认为前后两个种群有很大差异.其中,α 是一个经验值,一般取值 0.05.

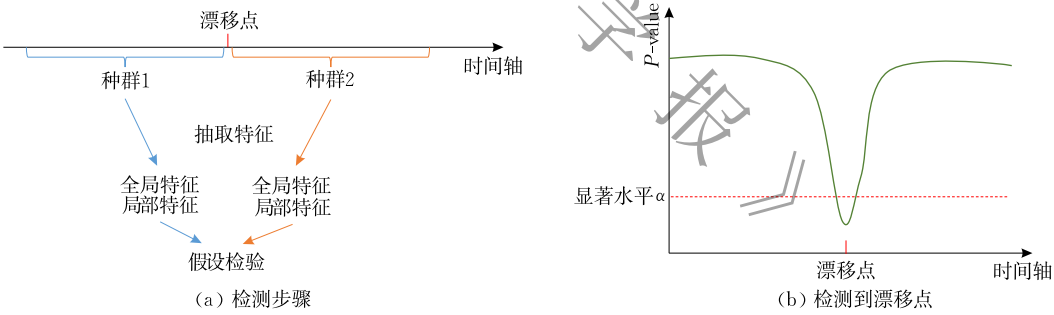


图 9 利用假设检验寻找日志中的漂移点

以上方法虽然能检测到漂移点,但是需要用户指定抽取的特征.换句话,用户必须具备很好的领域知识才能处理好漂移检测问题.文献[60]利用抽象域(abstract domain)来解决概念漂移,其核心步骤包括:①将日志分为训练集和测试集,紧接着求出训练集中所有轨迹的前缀;②根据轨迹前缀构造数值向量,然后将向量转化为多个不等式,进而求出训练集的一个多面体;③最后将测试集的轨迹同样转为数值,看是否落在多面体中,如果没有,即判定日志中出现了漂移.文章巧妙地将漂移检测问题转化为了数值计算问题,不足之处在于检测过程极为耗时.从求解过程中不难发现,主要耗时集中在求解轨迹的前缀部分.此外,算法还有一个缺点就是不能连续检测,一旦检测到漂移点,算法就自动终止了.Maaradji 等人[58]认为文献[56]采用的全局特征和局部特征不能准确反映业务模型的变化.因此,作者借鉴 van der Aalst 等人在文献[2]中提出的并发关系,重新定义了一套 Run 特征,用于描述日志中任意两个活动之间的关系.紧接着,同样是对前后两个窗口中的日志片段进行 Run 特征抽取和假设检验.结果表明,该方法在不同的变化模式中,都能较快且准确地检测到突发漂移.后续又加入了循环关系,提出了 Alpha 特征[35].

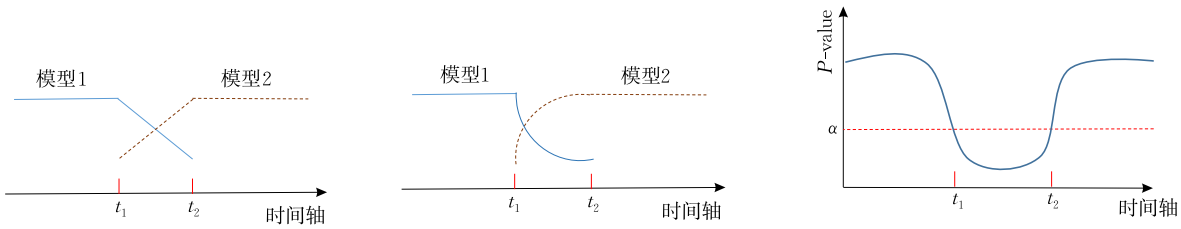
由图 9 可知,假设检验方法每移动一步,都要重新建立特征的分布模型.这种步步为营的方式

虽然稳健，却面临着效率低下的难题. 为此，Zheng 等人另辟蹊径，通过关系聚类方式来寻找漂移点(Tsinghua Process Concept Drift Detection，简称 TPCDD)[61]. TPCDD 的步骤包括：第 1 步，是将整个日志转为二维矩阵. 其中，行代表的是活动之间的直接后继关系或弱因果关系，列代表的是日志中的轨迹(按时间顺序排列)，值为 1 代表关系在轨迹中出现，0 代表没有出现；第 2 步，找出每一行中候选漂移点. 给定一个长度窗口 s，如果某个时间段 t，满足 $t \geq s$ 且其值全部为 0 或者全部为 1，则时间段中的最后一个时间点为一个候选漂移点；第 3 步，对二维矩阵中的所有候选漂移点进行 DBSCAN 聚类，得到类簇的中心点就是实际漂移点. 该方法在保证准确率的基础上大大提升了效率，对于包含上万条轨迹的日志，耗时仅有假设检验的五分之一. 但是，TPCDD 只适用于离线场景. 针对在线场景，文献[62]考虑了轨迹距离、相对时间距离、全局时间三个维度的聚类. 其中，轨迹维度计算的是轨迹之间的活动出现的频率，根据不同频率求出轨迹之间的距离. 相对时间维度计算的是轨迹中事件的耗时分布. 具体是，把轨迹中第一个执行事件的时间当作起始点 0，然后计算后面事件与第一个活动相差的时间(按秒级计算)，依次类推. 接着，根据四分位求出轨迹的四个时间段，并计算每个时间段中包含的事件数量. 最后，根据每条轨迹中不同时间段包含的事件数量来计算相对时间距离. 但是，该方法需要用户根据实时聚类的可视化结果判断哪个时间点发生了漂移，这无疑增加了使用者的负担.

此外，利用已有算法挖掘业务模型并以此为基准来判定概念漂移也是一种新颖的思路. Maggi 等人在文献[63]中基于线性时序逻辑挖掘出声明模型(declarative process model)，然后通过滑动窗口来判定是否需要更新模型. 每更新一次模型，就找到一个突发漂移点. 在滑动窗口过程中，利用 Lossy Counting 算法计算窗口中每个事件的频繁项. 如果频繁项发生了变化，则更新声明模型. 否则，持续滑动窗口. 但是，该方法准确率不高，原因是业务模型本身存在选择分支结构，所以频繁项发生变化不一定能说明模型发生了变化. 文献[64]提出了在日志的不同粒度上进行漂移检测分析，包括模型层面、轨迹层面和活动层面. 在模型层面，首先，对所有日志采用启发式挖掘算法，得到一个启发模型. 接着，根据不同年限，将日志划分为多个片段，依次检测每个片段与启发模型的拟合度、准确度和泛化度. 最后，通过三个指标的变化来断定是否发生了漂移. 在轨迹层面，考虑了轨迹从开始到结束的耗时以及轨迹变体占比两个指标. 在活动层面，监测每个活动不同年份的占比来判定是否发生了变化. 但是，对于三个指标的变化，需要专家知识来判断是否发生漂移.

(2) 复合类型的检测算法

前面的漂移检测算法能较好地解决单一漂移类型的检测，但是现实日志中往往存在多种漂移类型. 为此，Bose 等人在原有工作[56]的基础上进一步解决了日志中的两种漂移检测：突发漂移和渐变漂移[57]. 由图 10 可知，渐变漂移在某个时段是同时存在两个模型的轨迹，而文献[56]解决的是线性渐变漂移，如图 10(a)所示. 但不支持其他类型的渐变漂移检测，如图 10(b)所示的对数型渐变漂移. 该方法在检测两种漂移的过程中，采用的特征和步骤与原有工作基本相同，不同点是根据 P-value 的波谷宽度来判定是否是渐变漂移. 如图 10(c)所示，对比突发漂移，线性渐变漂移的 P-value 值的波谷会比较宽. 类似地，文献[59]将线性渐变漂移转化为混合分布计算，从而确定渐变漂移位置. 作者的出发点是渐变漂移中，位于 t_1 和 t_2 之间的数据分布是模型 1 和模型 2 两个数据分布的混合(如图 10(a)所示). 因此，解决策略是：首先，对日志进行突发漂移检测，利用 Run 特征和假设检验找出日志中所有突发漂移的变化点；接着，根据相邻两个变化点将日志分为前，中，后三个片段，每个片段对应的分布用 H_{k-1}、H_k 和 H_{k+1} 表示. 寻找一个分布 H，且存在两个权重 a 和 b，使得：$H = aH_{k-1} + bH_{k+1}$；最后，利用卡方检验(chi-square test)证明 $H \approx H_k$，即可表明该位置发生了渐变漂移.



(a) 线性渐变　　(b) 对数渐变　　(c) 线性渐变检测结果

图 10　渐变漂移的类型及检测

文献[65]扩展了 TPCDD 的工作用于支持突发漂移和渐变漂移两种类型检测. 该文的思路与文献[59]类似,不同点是,判断中间的日志片段是否为渐变漂移的依据是该日志片段的行为语义与前后两个日志片段的行为语义是否等价. 其中,用于刻画行为语义的是两种活动关系的频繁度. 作者在文中定义了直接后继关系和弱因果关系两种活动关系,但在实现过程中,用户可以增加其他关系进行扩展.

与通过特征抽取来检测复合型漂移不同,Stertz 等人在文献[66]中通过模型挖掘与拟合度计算,同时监测实时日志中的四种漂移类型. 首先,给定一个大小为 k 参考窗口,用于读取轨迹及挖掘历史模型. 如果来了新的轨迹,窗口会把最早的轨迹抛弃掉,并且更新模型. 接着,计算轨迹与模型的拟合度,根据拟合度来判定漂移类型. 判定的标准是:

(1) 增量漂移. 如果存在两个以上的历史模型 M_n 和 M_{n-1},且存在轨迹 t,使得拟合度 $fitness(t,M_{n-1})\geq\sigma$ 且 $fitness(t,M_n)\geq\sigma$,则找到增量漂移. 其中,σ 代表轨迹与模型的拟合度,取值范围是[0,1];

(2) 周期性漂移. 如果存在三个以上历史模型,且不满足(1),同时满足存在两个间隔的历史模型 M_n 和 M_{n-i},二者拟合度得分相差 $|S_n-S_{n-i}|\leq\xi$,则找到周期性漂移. 其中,$n\geq i\geq 2$,S 代表的是窗口中拟合度大于 σ 的轨迹数量占比,而 ξ 的值为人为设定,取值范围是[0,1];

(3) 渐变漂移. 存在一条未执行完的轨迹 t,其拟合度满足 $fitness(t,M)\geq\sigma$;

(4) 突发漂移. 不满足以上三个条件,则判定为突发漂移.

该方法不足之处是:(1) 参数敏感. 任何一个参数的微小变化,都会导致结果不同;(2) 准确率不高. 文章使用 Inductive Miner 算法挖掘出模型后,只考虑了日志的拟合度,不考虑精确度和泛化度. 这样只能检测到新的关系或活动带来的变化,检测不到由消失的关系或者活动引起的漂移现象.

5.2 评估对比

前面具体介绍了过程挖掘中概念漂移检测算法的内容,这个章节对以上算法进行归纳总结(如表 6). 然后,选择一些具有代表性的算法进行实验评估,进一步剖析算法的优缺点.

表 6 概念漂移算法的综述

(表中:①表示突发漂移;②表示渐变漂移;③表示增量漂移;④表示周期性漂移)

类型	文献	变化特征	采用的方法	漂移类型	优点	缺点
单一类型的检测算法	[56]	全局特征 RC 和 RE、局部特征 WC 和 J-measure	假设检验	①	利用不同特征刻画日志分布	准确率不高且耗时
	[60]	数值向量	抽象域	①	将漂移转化为数值计算	算法复杂度高且不能连续检测
	[58]	并发特征 Run	假设检验	①	特征容易获取且准确率高	窗口逐步移动耗时且具有一定延迟
	[35]	循环特征	假设检验	①	描述变化的能力增强	检测结果有延迟
	[61]	直接后继关系和弱因果关系	密度聚类	①	将日志转化为二维数组,高效检测漂移	不支持在线检测
	[62]	活动频率和事件时间	密度聚类	①	多个维度刻画漂移	准确率低且需要专家知识
	[63]	事件频率	模型挖掘+有损计数	①	挖掘算法+频率统计算法	无法解决模型中的选择分支模块
	[64]	拟合度、轨迹变体、活动占比	模型挖掘+数据统计	①	从模型、轨迹、活动三个层次寻找漂移点	严重依赖专家知识
复合类型的检测算法	[57]	全局特征 RC 和 RE、局部特征 WC 和 J-measure	假设检验	①②	利用不同波谷来判定不同漂移类型	需要人为判定变化类型
	[59]	并发特征 Run	假设检验	①②	将渐变漂移转化为寻找相似的特征分布	复杂度高且参数敏感
	[65]	不同关系的频繁度	行为语义是否等价	①②	用行为语义判定渐变漂移	不支持在线检测且渐变类型必须为线性
	[66]	拟合度 Fitness	模型挖掘	①②④③	支持四种漂移类型	Fitness 只能检测某些变化模式,准确率低

表 6 梳理了概念漂移算法的思路及优缺点. 从中可以发现:(1) 现有漂移检测算法,大多数集中在单一的漂移类型[60-64],能同时检测到日志中四种漂移类型的算法较少[66];(2) 假设检验是当前漂移检测的常见手段[56-58],主要原因是其鲁棒性较好,能有效降低由选择分支的执行概率带来的过拟合现象;

(3)用于刻画日志的特征对检测结果有一定影响.如文献[56]与文献[58]都是用假设检验的策略来检测漂移,区别在于前者使用了局部和全局特征,而后者使用的是活动之间的并发特征;(4)多数方法都只能用于离线的场景进行测试,如文献[60-61,65].而在线检测的方法又存在耗时、且检测延迟较多的问题[58-59].

为了进一步评价算法的性能,本文选了Alpha[35]、Run[58]、TPCDD[61]三个算法对日志中突发漂移进行测试,理由有:(1)凸显不同特征和不同方法的作用.Run和Alpha都是采用了假设检验的方法检测漂移,只是抽取的特征不同.因此,可以对比出特征的作用.而TPCDD采用的是密度聚类方法,所以可以测试不同策略的效果;(2)公开源码.三个算法都公开了算法源码,并且Run和Alpha是在同一个开源平台Apromore[67],保证了实验环境的可靠性.评价指标选用了衡量分类问题的F-score[22],其定义如下:

$$F\text{-score}=\frac{2\times precision\times recall}{precision+recall} \tag{6}$$

其中,

$$precision=\frac{TP}{TP+FP} \tag{7}$$

$$recall=\frac{TP}{TP+FN} \tag{8}$$

为了计算真正例(TP)、假正例(FP)和假反例(FN),需要设定一个滞后期(lag period)参数r,用以判定算法检测到的漂移点是属于真正例或其他.判定的标准如下:

(1)真正例.令检测到的漂移点为t',且在区间$[t'-r,t'+r]$内存在一个真实的漂移点t.则t'属于真正例.

(2)假正例.令检测到的漂移点为t',且在区间$[t'-r,t'+r]$内不存在一个真实的漂移点t.则t'属于假正例.

(3)假反例.令真实的漂移点为t,且在区间$[t-r,t+r]$内检测不到漂移点t'.则t属于假反例.

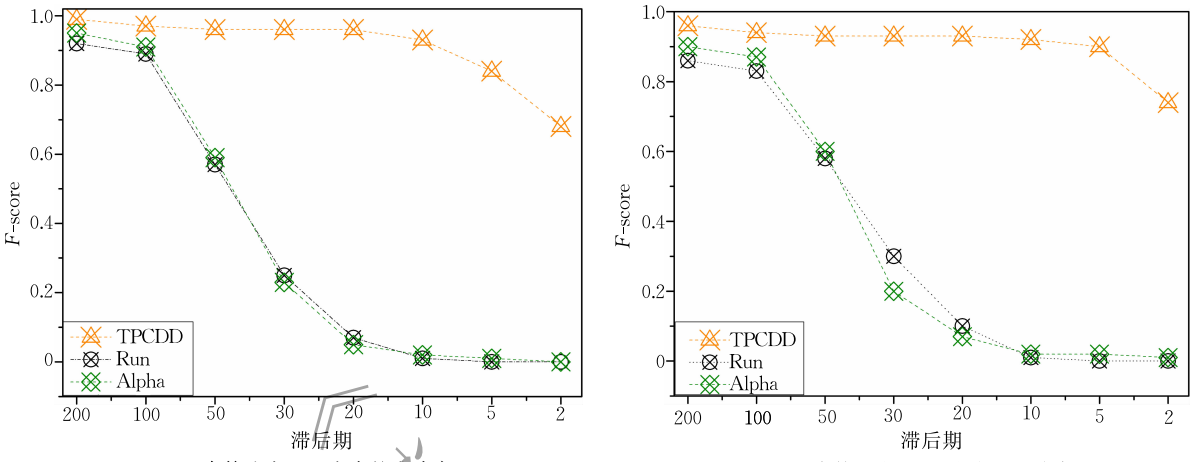
由以上标准可知,参数r的大小,对漂移检测的结果有一定的影响.为此,本文取不同的r值来进行算法评价,滞后期取值越小,说明漂移检测到的变化点越接近真实变化点.测试过程中,选用的数据集来自文献[58]提供的72个有标日志数据.该数据集包含了18种模型演变的模式,表7展示了12种简单变化模式,另外6种变化是在此基础上进行组合(其中,I代表的是增加元素类型,O代表的是调整顺序类型,R代表的是可选择类型).其中,每种模式又生成了四种不同量级的日志,分别是含有2500条轨迹的2.5k日志,含有5000条轨迹的5k日志,含有7500条日志的7.5k日志以及含有10000条轨迹的10k日志.限于篇幅,本文仅展示了5k和10k两种数据集的实验结果,如图11所示.从图11(a)和(b)可以发现,在两种不同的量级数据中,三个算法的准确率都会随着滞后期r的变小而降低.其中,采用Run方法和Alpha方法的拐点在$r=100$.换句话,当滞后期取值小于100后,Run和Alpha的准确率出现了大幅下降现象.主要原因有两方面:(1)两种方法的特征刻画还是欠拟合,无法敏感地检测到微小的模型变化;(2)窗口大小有一定影响.由于统计的是两个窗口之间的特征分布差异,所以需要窗口中包含一定量的不同特征,才能检测到漂移现象.此外,可以发现除了$r=30$或$r=20$外,大部分情况下Alpha的准确率都要优于Run.相比之下,TPCDD算法的准确率明显高于Run和Alpha.TPCDD采用的特征是直接后继和弱紧邻关系,这种特征可以快速地检测到日志中的漂移现象.因此,当r取值小于10时,TPCDD的准确率才出现明显下降.

表7　过程模型里12种不同的变化模式[58]

模式	变化内容	类型
RE	增加/删除片段	I
CF	将两个片段调整选择或顺序	R
LP	将两个片段进行循环或者拆解循环	O
PL	将两个片段进行并发或者顺序	R
CB	跳过某片段	O
CM	将片段移动到选择分支以内或以外	I
CD	使两个片段同步发生	R
CP	重复某个片段	I
PM	将片段移动到并发分支以内或以外	I
RP	替换某个片段	I
SW	交换两个片段	I
FR	修改分支执行的频率	O

此外,测试过程中三个算法的耗时排序是TPCDD<Run<Alpha.主要原因是,TPCDD是将日志中的特征转化为二维矩阵进行计算,所以比较快.但是,这样造成了该算法不支持在线检测.而Run和Alpha的耗时主要集中在每次移动窗口(且每次只移动一步),都要重新抽取日志中的特征进行分布的差异性计算.

为了具体分析每种算法在 18 种不同的变化模式中的表现，本文选取了 10k 量级中不同模式进行测试. 图 12 和图 13 分别展示了 $r=100$ 与 $r=10$ 时，每种模式的检测准确率变化. 由图 12 可知，当 $r=100$ 时，TPCDD 在 IRO、OIR、ORI 三种组合变化模式上，表现的效果欠佳，其他模式基本保持了 100% 的准确率. 此时，Run 方法在 IRO、OIR、LP、CB、CD 五种模式上表现不好，最严重是 CD 模式的准确率为 0. Alpha 方法对 LP、CB、CD 三种模式表现也差强人意. 其中，CB 模式的准确率为 0. 由图 13



(a) 三个算法在5k日志中的准确率　　(b) 三个算法在10k日志中的准确率

图 11　TPCDD、Run 和 Alpha 三个算法在不同量级上的 F-score

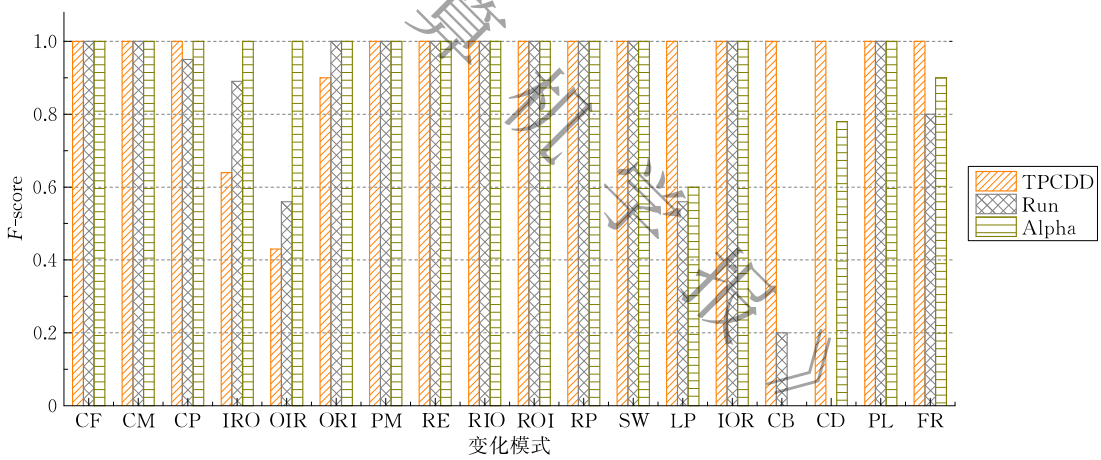


图 12　滞后期 $r=100$ 时 18 种演化模式的准确率

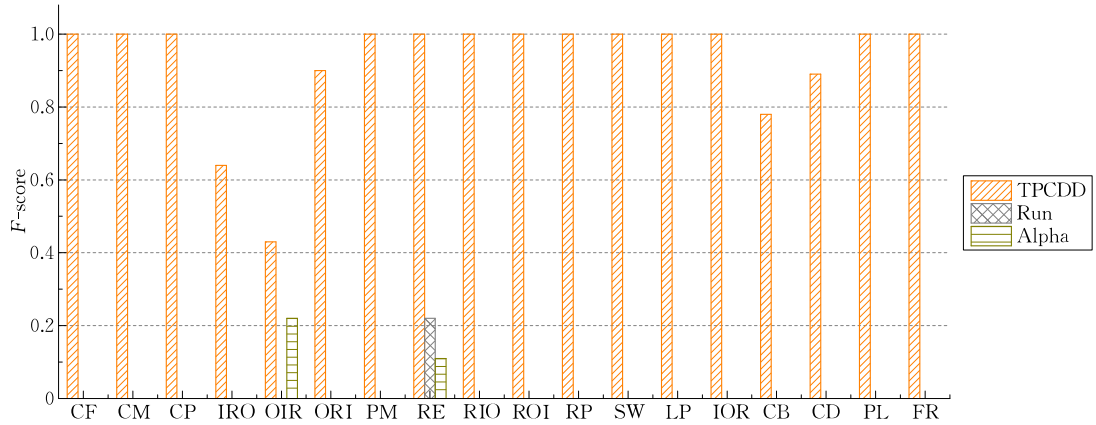


图 13　滞后期 $r=10$ 时 18 种演化模式的准确率

可知，当 r 的值减小到 10 时，Run 和 Alpha 两种方法在 18 种变化模式中，准确率基本降为 0（除了 OIR 和 RE 两种模式外）. 综上所述，无论哪种算法，都很难保证对每种变化模式都有很好的准确率. 还有，Run 和 Alpha 的准确率很受滞后期 r 的取值影响.

5.3　小　结

前面两个章节分别从理论方法和实验测试两方面对当前的概念漂移算法进行了梳理分析，以下针对现有趋势和未来研究方向两方面进行探讨.

（1）现有概念漂移的研究趋势

① 漂移对象和漂移内容多层次化. 研究日志中的漂移对象从突发漂移[56]到渐变漂移[59]，再到同时解决多种漂移[66]，逐渐向多层次化演变. 同时，早期的研究内容重点关注日志轨迹中的漂移现象[56]，现在已经过渡到模型、轨迹和活动三个层面的漂移检测[64].

② 漂移特征从重量级向轻量级演化. 为了找到日志中的漂移，需要定义漂移的特征是什么，进而追踪该特征随着时间推移是否发生了变化. 文献[56-57]定义了全局特征和局部特征用以漂移检测，而 Maaradji 等人却只用 Run 特征就能做到较高的准确率[58]. 相比之下，TPCDD 采用的直接后继关系则是更加轻量级的特征[61]. 在保证准确率的前提下，特征轻量级后，带来的好处就是明显提升了检测效率.

③ 业务变化的维度从控制流维度转移到活动属性维度. 传统的概念漂移指是业务模型中控制流维度发生了变化，而文献[68-69]发现业务流程的模型不变，但执行活动的属性发生了变化，如时间、速度等. 为此，提出基于活动属性的漂移检测方法.

（2）未来的研究方向

概念漂移的研究时间相对于轨迹聚类要稍微晚，目前该方向还处于一个上升期，未来研究方向可以从以下几点着手：

① 从浅层的检测漂移点到更深层的定位及描述漂移点. 漂移检测的未来应该走向更高层次的描述漂移内容变化，而不单单是给出漂移变化的位置. Bose 等人将过程挖掘中的漂移检测划分为三个层次[56-57]：(i) 漂移点检测（change detection）. 这个层次研究的内容是找出日志中具体变化位置；(ii) 变化的定位和描述（change localization and characterization）. 这部分研究是在变化点检测基础上，具体给出业务变化的内容描绘. 如哪个活动或者哪个部分的业务发生了变化；(iii) 揭露整个业务演化过程（unravel process evolution）. 即知道了变化点及变化内容，进一步揭示业务是如何动态地发生演化的. 总结起来，第一个层次是回答漂移的 When，第二个层次回答 What，第三个层次回答了业务变化的 How. 这个方向可以结合流程文本抽取[70-71]工作进行研究，充分利用流程文本的领域知识来描述流程的前后的变化内容.

② 从漂移检测到漂移预测. 目前的漂移检测都是在漂移发生后，才能检测到变化，这种延迟对于实时响应系统来说较难接受. 如果能在变化之前就能发现并预警，就使得企业更好做出响应. 文献[72-73]利用深度学习的 RNN 模型，成功预测了日志中的下一个任务. 为此，可以借鉴其思想，通过预测轨迹片段来预警漂移. 文献[74]讨论了如何选取训练数据能更好地建立预测模型，从而准确地预测未来将要发生的概念漂移. 作者在文中给出了三种选取方案，分别是基于历史数据的 Last 方案、基于当前数据的 Next 方案以及混合了历史数据和当前数据的 Mixed 方案.

③ 聚类与漂移交叉应用. 文献[62]使用聚类的方法来寻找日志中的变化点，其主要思路是在轨迹相似性的基础上，引入了相对时间和全局时间，综合考虑三个维度的距离来判定是否发生漂移. 该方法创新地将轨迹聚类的思路嵌入到漂移检测的问题中，不足之处是需要领域人员参与. 未来的方向可以在这方面多做尝试，比如，利用层次聚类无序地建立多个层次的轨迹簇，根据启发式阈值判断某层所属轨迹是否显著地分布在某个或某些时间戳前后来辅助完成概念漂移任务.

④ 引入专家知识，提升漂移的可解释性和准确率. 无论是轻量级[61]还是重量级[56]的漂移特征，都是分析人员的多年经验，不一定适用于不同业务的漂移检测. 与此同时，不同领域的专家对领域的可变节点有一定的经验积累. 因此，可以先引入少部分的专家知识，进行半监督学习提取出重要的漂移特征，再跟踪特征检测漂移从而提升漂移检测的准确率. 这个理念与前面分析的轨迹聚类中利用专家知识聚类[43]有着异曲同工之处. 此外，文献[75]认为传感器中的属性数据（如温度、湿度、压力等）对于解释工业流程为何发生变化有很大帮助. 所以，作者着重分析传感器中数据的漂移现象，通过这类数据的变化来解释业务流程发生演化的原因或预警业务即将发生演化.

6 讨 论

日志划分的输入和输出非常明确，即输入原始日志，然后通过划分形成多个子日志作为输出. 接着，利用过程挖掘中的模型发现建立准确且复杂度低的过程模型，进而为实际业务提供了配置或优化的手段. 如图 14 所示，在研究内容上，除了轨迹聚类和概念漂移之间的技术路线可以相互借鉴以外，本文认为很多日志划分还需融合其他技术才能更好推进整个领域的发展.

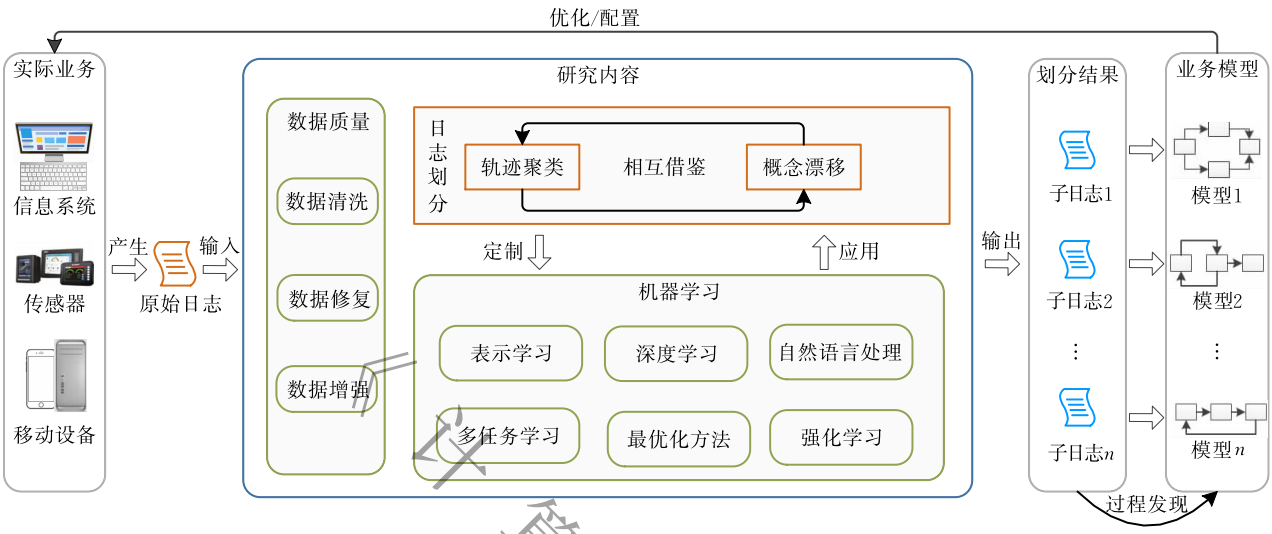


图 14 日志划分的可扩展性研究内容

在数据质量方面，现有日志划分方法少有考虑日志的完整性、正确性等问题. 为此，可以融合数据清洗、数据修复、和数据增强等技术手段来提高日志质量，进而为日志划分提供更好的服务. 其中，数据清洗可过滤日志中的无关数据或重复信息，保证了数据正确性. 数据修复及数据增强可以纠正日志中错误或者补全缺少的属性，确保数据完整性.

在研究方法上，可以引入前沿的机器学习内容，多维度、多角度地完善并提高现有日志划分的准确性. 比如，在 4.2 节实验评估中发现当前聚类方法在降低复杂度问题上，效果还需要提升. 这部分内容可以考虑用表示学习方法，通过层次表示来更好地学习活动-轨迹-日志的潜在语义，进而提升聚类效果. 另外，深度学习中成熟的 CNN 和 RNN 模型，可以分别捕获日志轨迹的局部信息和全局信息，为日志划分提供更好的潜在语义特征表示. 针对聚类或漂移的可解析性问题，可以引入自然语言分析技术，辅助生成易于阅读的自然语言文本片段，增强用户的理解性. 还有，针对现有划分方法中常见的参数多、难配置等问题，可以利用最优化方法中的多个并行网络进行训练调整，从而自动学习最优配置. 此外，现有日志划分技术呈现出从监督学习过渡到半监督学习的趋势，这部分内容可以结合强化学习中激励与惩罚的反馈机制来实现日志划分. 与此同时，现有研究开始尝试在轨迹聚类中引入概念漂移(反之亦然)，多任务学习中的共享表示可以为用户提供一个全新思路.

7 总结与展望

过程挖掘是业务过程管理和数据挖掘的交叉学科，目标是从信息系统的日志数据中抽取出有价值的业务信息，用以优化企业的业务过程管理. 常见的三个应用场景包括过程发现、合规性检查及过程增强. 本文针对过程发现场景中存在的问题(即模型复杂和模型演化)，系统地梳理了现有日志划分中的轨迹聚类和概念漂移两种技术，主要的工作内容包括：

(1) 通过实际问题引出轨迹聚类和概念漂移是解决日志划分问题的两种不同视角，并详细梳理及总结了二者的异同点(如表 1 所示). 其中，最大的区别是轨迹聚类关注的是模型的静态结构而概念漂移侧重于模型的动态演化；

(2) 较为全面地统计了两种技术近十年的研究成果，同时，发现两种技术都处于过程挖掘研究中的上升期，且概念漂移还会持续一段时间；

(3) 详细地讨论了现有轨迹聚类方法的研究思路，并将现有研究分为三种类型：距离驱动的聚类方法、模型驱动的聚类方法以及混合聚类方法. 还有，

归纳并总结了三种类型算法的优缺点.同时,利用真实日志数据进一步评测了三种类型算法的能力,并发现目前算法针对真实日志的表现还有很大的提升空间.最后,总结了当前的研究趋势,并从算法及框架层面讨论了未来研究方向;

(4)梳理了当前概念漂移的主要研究内容,包括单一类型的检测和复合类型的检测.整理并讨论了现有算法的优缺点,另外,通过实验表明现有算法的准确率会随着阈值滞后期的变小而下降.此外,整理并发现概念漂移的内容逐渐走向多层次、多维度.最后,对概念漂移的未来方向进行展望,如从浅层的漂移检测向深层的漂移描述发展等.

此外,当前日志划分的研究内容对输入数据的限制过于苛刻,主要体现在:(1)输入的日志数据必须是完整且均匀分布的时间序列.这个前提假设在工业界往往面临很多挑战(如长尾序列),无法平滑地进行落地应用;(2)输入的数据类型不丰富.比如,现有研究只针对时间序列日志,但是实际业务产生的数据类型往往包含图片、视频等更加丰富的数据形式.而这些数据的研究利用,可以正反馈来促进日志划分,并提高可解释性.因此,未来研究方向可从以上问题出发,进行全新场景的探索研究.

参 考 文 献

[1] Wil van der Aalst W. Process Mining: Discovery, Conformance and Enhancement of Business Processes. Heidelberg, Germany: Springer Publishing Company, 2014

[2] Wil van der Aalst W, Weijters T, Maruster L. Workflow mining: Discovering process models from event logs. IEEE Transactions on Knowledge and Data Engineering, 2004, 16(9): 1128-1142

[3] Wil van der Aalst W, Schonenberg M H, Song M. Time prediction based on process mining. Information Systems, 2011, 36(2): 450-475

[4] Deokar A V, Tao J. OrgMiner: A framework for discovering user-related process intelligence from event logs. Information Systems Frontiers, 2020, 04: 1-20

[5] Weighers A J M M, Aalto W M P, Medeiros A K A. Process mining with the heuristics miner algorithm. Eindhoven University of Technology, 2006, 166: 1-34

[6] Sarno R, Sungkono K R. Hidden Markov model for process mining of parallel business processes. International Review on Computers and Software, 2016, 11(4): 290-306

[7] Medeiros A K A D, Weijters A J M M, Alston W M P. Genetic process mining: An experimental evaluation. Data Mining and Knowledge Discovery, 2007, 14(2): 245-304

[8] Leemans S J J, Fahland D, Aalst W M P. Discovering block-structured process models from event logs—A constructive approach//Proceedings of the International Conference on Application and Theory of Petri Nets and Concurrency. Milan, Italy: Springer-Verlag, 2013: 311-329

[9] Leemans S J J, Fahland D, Aalst W M P. Discovering block-structured process models from event logs containing infrequent Behavior//Proceedings of the International Conference on Business Process Management. Beijing, China, 2013: 66-78

[10] Guo Q, Wen L, Wang J, et al. Mining invisible tasks in non-free-choice constructs//Proceedings of the International Conference on Business Process Management. Innsbruck, Austria: Springer, 2015: 109-125

[11] Wen L, Aalst W M P, Wang J, et al. Mining process models with non-free-choice constructs. Data Mining and Knowledge Discovery, 2007, 15(2): 145-180

[12] Lin Lei-Lei, Zhou Hua, Dai Fei, et al. Approach to mining length-two loops from the log without "aba" pattern. Journal of Software, 2018, 29(11): 3278-3294(in Chinese)
(林雷蕾, 周华, 代飞等. 一种从无"aba"模式的日志中挖掘2度循环的方法. 软件学报, 2018, 29(11): 3278-3294)

[13] Lekić J, Milićev D. Discovering block—Structured parallel process models from causally complete event logs. Journal of Electrical Engineering, 2016, 67(2): 111-123

[14] Sander J J Leemans, Dirk Fahland, Alston W M P. Scalable process discovery and conformance checking. Software and Systems Modeling, 2018, 17(2): 599-631

[15] Sander J J Leemans, Anna F Syring, Alston W M P. Earth movers' stochastic conformance checking//Proceedings of the International Conference on Business Process Management. Vienna, Austria, 2019: 127-143

[16] Adriansyah A, Dongen B F A, Aalst W M P. Conformance checking using cost-based fitness analysis//Proceedings of the IEEE International Enterprise Distributed Object Computing Conference(EDOC). Helsinki, Finland, 2011: 55-64

[17] Adriansyah A, Munozgama J, Carmona J, et al. Alignment based precision checking//Proceedings of the International Conference on Business Process Management. Tallinn, Estonia, 2012: 137-149

[18] Broucke S K L M, Weerdt J D, Vanthienen J, et al. Determining process model precision and generalization with weighted artificial negative events. IEEE Transactions on Knowledge and Data Engineering, 2014, 26(8): 1877-1889

[19] De Weerdt J, vanden Broucke S, Vanthienen J, et al. Active trace clustering for improved process discovery. IEEE Transactions on Knowledge and Data Engineering, 2013, 25(12): 2708-2720

[20] Wang Jianming, Song Shaoxu, Zhu Xiaochen, et al. Efficient recovery of missing events. IEEE Transactions on Knowledge and Data Engineering, 2016, 28(11): 2943-2957

[21] Wang Jianming, Song Shaoxu, Lin Xuemin, et al. Cleaning structured event log: A graph repair approach//Proceedings of the IEEE International Conference on Data Engineering. Seoul, South Korea, 2015: 30-41

[22] Alharbi A, Bulpitt A, Johnson O. Improving pattern detection in healthcare process mining using an interval-based event selection method//Proceedings of the International Conference on Business Process Management. Barcelona, Spain: Springer, 2017: 88-105

[23] Okoye K, Tawil A R H, Naeem U, et al. Semantic process mining towards discovery and enhancement of learning model analysis//Proceedings of the IEEE International Conference on High Performance Computing and Communications. New York, USA, 2015: 363-370

[24] Jareevongpiboon W, Janecek P. Ontological approach to enhance results of business process mining and analysis. Business Process Management Journal, 2013, 19(3): 459-476

[25] Fahland D, Aalst W M P. Model repair — aligning process models to reality. Information Systems, 2015, 47(1): 220-243

[26] Fahland D, Aalst W M P. Repairing process models to reflect reality. Lecture Notes in Computer Science, 2012, 7481: 229-245

[27] Aalst W V D, Adriansyah A, Medeiros A K A D, et al. Process mining manifesto. Lecture Notes in Business Information Processing, 2011, 99: 169-194

[28] Aalst W M P. Process discovery: Capturing the invisible. IEEE Computational Intelligence Magazine, 2010, 5(1): 28-41

[29] Christian W. Günther, Wil van der Aalst. Fuzzy mining — Adaptive process simplification based on multi-perspective metrics//Proceedings of the International Conference on Business Process Management. Brisbane, Australia, 2007: 328-343

[30] Bauer M, Senderovich A, Gal A, et al. How much event data is enough? A statistical framework for process discovery//Proceedings of the International Conference on Advanced Information Systems Engineering (CAiSE). Tallinn, Estonia, 2018: 239-256

[31] Suriadi S, Andrews R, Ter Hofstede AHM, et al. Event log imperfection patterns for process mining: Towards a systematic approach to cleaning event logs. Information Systems, 2017, 64(5): 132-150

[32] Conforti R, La Rosa M, Ter Hofstede AHM. Filtering out infrequent behavior from business process event logs. IEEE Transactions on Knowledge and Data Engineering, 2017, 29(2): 300-314

[33] R P Jagadeesh Chandra Bose, W M P van der Aalst. Context aware trace clustering: Towards improving process mining results//Proceedings of the SIAM International Conference on Data Mining. Nevada, USA, 2009: 401-412

[34] Huang Tao, Xu Boyi, Cai Hongming, et al. A fog computing based concept drift adaptive process mining framework for mobile APPs. Future Generation Computer System, 2018, (89): 670-684

[35] Ostovar A, Maaradji A, La Rosa M, et al. Detecting drift from event streams of unpredictable business processes//Proceedings of the International Conference on Conceptual Modeling. Gifu, Japan, 2016: 330-346

[36] Greco G, Guzzo A, Pontieri L, et al. Mining expressive process models by clustering workflow traces//Proceedings of the Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD). Sydney, Australia, 2004: 52-62

[37] Lee D, Park J, Pulshashi I R, et al. Clustering and operation analysis for assembly blocks using process mining in shipbuilding industry. Lecture Notes in Business Information Processing, 2013, 159: 67-80

[38] Montani S, Leonardi G. Retrieval and clustering for supporting business process adjustment and analysis. Information Systems, 2014, 40: 128-141

[39] Delias P, Doumpos M, Grigoroudis E, et al. Supporting healthcare management decisions via robust clustering of event logs. Knowledge-Based Systems, 2015, 84: 203-213

[40] Appice A, Malerba D. A co-training strategy for multiple view clustering in process mining. IEEE Transactions on Services Computing, 2016, 9(6): 832-845

[41] Delias P, Doumpos M, Grigoroudis E, et al. A non-compensatory approach for trace clustering. International Transactions in Operational Research, 2017, 26(5): 1828-1846

[42] De Koninck P, Nelissen K, Baesens B, et al. An approach for incorporating expert knowledge in trace clustering//Proceedings of the International Conference on Advanced Information Systems Engineering. Essen, Germany: Springer, 2017: 561-576

[43] Lu Xixi, Tabatabaei S A, et al. Trace clustering on very large event data in healthcare using frequent sequence patterns//Proceedings of the International Conference on Business Process Management. Vienna, Austria, 2019: 198-215

[44] Ferreira D, Zacarias M, Malheiros M, et al. Approaching process mining with sequence clustering: experiments and findings//Proceedings of the International Conference on Business Process Management. Brisbane, Australia, 2007, 4714: 360-374

[45] Sun Yaguang, Bauer B. A novel top-down approach for clustering traces//Proceedings of the International Conference on Advanced Information Systems Engineering. Stockholm, Sweden, 2015: 331-345

[46] Chatain T, Carmona J, van Dongen B V. Alignment-based trace clustering//Proceedings of the International Conference on Conceptual Modeling. Valencia, Spain, 2017: 295-308

[47] Boltenhagen M, Chatain T, Carmona J. Generalized alignment-based trace clustering of process behavior//Proceedings of the International Conference on Applications and Theory of Petri Nets and Concurrency. Aachen, Germany, 2019: 237-257

[48] Richter F, Zellner L, Sontheim J, et al. Model-aware clustering of non-conforming traces//Proceedings of the OTM Confederated International Conferences on the Move to Meaningful Internet Systems: CoopIS. Munich, Germany, 2019: 193-200

[49] De Koninck P, De Weerdt J. Scalable mixed-paradigm trace clustering using super-instances//Proceedings of the International Conference on Process Mining. Aachen, Germany, 2019: 17-24

[50] Dongen B F V, Mederios A K A D, Verbeek H M W, et al. The ProM framework: A new era in process mining tool support//Proceedings of the International Conference on Application and Theory of Petri Nets. Berlin, Germany: Springer, 2005: 444-454

[51] Delcoucq L, Lecron F, Fortemps P, et al. Resource-centric process mining: Clustering using local process models//Proceedings of the 35th ACM/SIGAPP Symposium on Applied Computing. Brno, Czech Republic, 2020: 45-52

[52] Sagawa R, Shiba Y, Hirukawa T, et al. Automatic feature extraction using CNN for robust active one-shot scanning//Proceedings of the IEEE 23rd International Conference on Pattern Recognition. Cancun, Mexico, 2016: 234-239

[53] De Koninck Pieter, Seppe vanden Broucke, Jochen De Weerdt. Act2vec, trace2vec, log2vec and model2vec: Representation learning for business processes//Proceedings of the International Conference on Business Process Management. Sydney, Australia, 2018: 9-14

[54] Breiman L. Using iterated bagging to debias regressions. Machine Learning, 2001, 45(3): 261-277

[55] Yeshchenko A, Bayomie D, Gross S, et al. Visualizing business process evolution//Proceedings of the International Conference on Business Process Modeling, Development and Support. Grenoble, France, 2020: 185-192

[56] Bose R P J C, Aalst W M P D, Žliobaite I, et al. Handling concept drift in process mining. Lecture Notes in Computer Science, 2011, 4(1): 391-405

[57] Bose R P J C, van der Aalst W M P, Žliobaite I, Pechenizkiy M. Dealing with concept drifts in process mining. IEEE Transaction on Neural Networks and Learning Systems, 2014, 25(1): 154-171

[58] Maaradji A, Dumas M, La Rosa M, et al. Fast and accurate business process drift detection//Proceedings of the International Conference on Business Process Management. Rio de Janeiro, Brazil: Springer, 2015: 406-422

[59] Maaradji A, Dumas M, La Rosa M, et al. Detecting sudden and gradual drifts in business processes from execution traces. IEEE Transactions on Knowledge and Data Engineering, 2017, 29(10): 2140-2154

[60] Carmona J, Gavalda R. Online techniques for dealing with concept drift in process mining//Proceedings of the International Symposium on Intelligent Data Analysis. Helsinki, Finland, 2012: 90-102

[61] Zheng C, Wen L, Wang J. Detecting process concept drifts from event logs//Proceedings of the International Conferences on the Move to Meaningful Internet Systems. Rhodes, Greece, 2017: 524-542

[62] Barbon Junior S, Tavares G M, da Costa V G T, et al. A framework for human-in-the-loop monitoring of concept-drift detection in event log stream//Companion Proceedings of the Web Conference. Lyon, France, 2018: 319-326

[63] Maggi F M, Burattin A, Cimitile M, et al. Online process discovery to detect concept drifts in LTL-based declarative process models//Proceedings of the OTM Confederated International Conferences on the Move to Meaningful Internet Systems: CoopIS, DOA-Trusted Cloud, and ODBASE 2013. Berlin, Heidelberg: Springer, 2013: 94-111

[64] Kurniati A P, McInerney C, Zucker K, et al. A multi-level approach for identifying process change in cancer pathways//Proceedings of the 17th International Conference on Business Process Management. Vienna, Austria, 2019: 595-607

[65] Zheng Can-Bin, Wu Xuan, Wen Li-Jie, et al. Detecting concept drift of process models from event logs. Computer Integrated Manufacturing Systems, 2019, 25(4): 830-836(in Chinese)
(郑灿彬, 吴翾, 闻立杰等. 从事件日志中发现过程模型的渐变漂移. 计算机集成制造系统, 2019, 25(4): 830-836)

[66] Stertz F, Rinderle-Ma S. Process histories — Detecting and representing concept drifts based on event streams//Proceedings of the OTM Confederated International Conferences on the Move to Meaningful Internet Systems. Valletta, Malta, 2018: 318-335

[67] La Rosa M, Reijers H A, Aalst W M P. APROMORE. Expert Systems with Applications, 2011, 38(6): 7029-7040

[68] Stertz F, Rinderle-Ma S. Detecting and identifying data drifts in process event streams based on process histories//Proceedings of the International Conference on Advanced Information Systems Engineering. Rome, Italy, 2019: 240-252

[69] Richter F, Seidl T. Looking into the TESSERACT: Time-drifts in event streams using series of evolving rolling averages of completion times. Information System, 2019, 84(9): 265-282

[70] Qian Chen, Wen Lijie, Kumar A, Lin Leilei, et al. An approach for process model extraction by multi-grained text classification//Proceedings of the International Conference on Advanced Information Systems Engineering. Grenoble, France, 2020: 268-282

[71] van der Aa H, Leopold H, Reijers H A. Detecting inconsistencies between process models and textual descriptions//Proceedings of the International Conference on Business Process Management. Insbruck, Austria, 2015: 90-105

[72] Evermann J, Rehse J R, Fettke P. A deep learning approach for predicting process behaviour at runtime//Proceedings of the International Conference on Business Process Management. Rio de Janeiro, Brazil, 2016: 327-338

[73] Tax N, Verenich I, La Roa M, et al. Predictive business process monitoring with LSTM neural networks//Proceedings of the International Conference on Advanced Information Systems Engineering. Essen, Germany, 2017: 170-185

[74] Baier L, Reimold J, Kühl N. Handling concept drift for predictions in business process mining//Proceedings of the IEEE 22nd Conference on Business Informatics. Antwerp, Belgium, 2020: 76-83

[75] Stertz F, Rinderle-Ma S, Mangler J. Analyzing process concept drifts based on sensor event streams during runtime//Proceedings of the 18th International Conference on Business Process Management. Seville, Spain, 2020: 202-219

LIN Lei-Lei, Ph. D., assistant researcher. His research interests include process mining and software engineering.

WEN Li-Jie, Ph. D., associate professor, Ph. D. supervisor. His research interests include process mining, and natural language processing.

QIAN Chen, Ph. D. candidate. His research interests include process text extraction and natural language processing.

ZONG Zan, Ph. D. candidate. His research interests focus on process optimization.

WANG Jian-Min, Ph. D., professor, Ph. D. supervisor. His research interests include business process management, and artificial intelligence.

Background

Process mining belongs to software engineer and service computing area, which is aims at extracting valuable knowledge from event logs commonly available in today's information systems. The topics in process mining can be broadly classified into three categories: process discovery (i. e. extracting process models from an event log without using any apriori information), conformance checking (i. e. monitoring deviations by comparing a model and a log) and process model enhancement (i. e. extending or improving an existing model using additional information). Process discovery is the most prominent process mining technique. For many organizations it is surprising to see that existing techniques are indeed able to discover real processes based on example executions in event logs. As we know, a majority of process discovery techniques have been proposed to mine the process models for optimizing the running business processes, but the models mining from the real-life logs are hard to comprehend. This problem seriously hinders the application of process mining in real business. An approach to overcome this is log division that includes two topic: trace clustering and concept drift. Trace clustering aims to solve the problem that the model is too complex, while concept drift is to solve the problem of business evolution. However, the above problems have not been well addressed and remain considerable challenges.

This paper focuses on trace clustering and concept drift in process mining. We summarizes the previous related studies on trace clustering and concept drift, and discuss the similarities and differences between the two branches. In addition, real-life logs and synthetic logs are used to test for analyzing the advantage and disadvantage among these methods. Finally, the problems in each branch are discussed and some future research directions are suggested.

This work was in part supported by the National Key Research and Development Plan (No. 2019YFB1704003), the National Natural Science Foundation of China (Nos. 71690231, 62021002).