一种利用日志划分从复杂日志中挖掘块结构过程的方法

段 瑞 方 欢 詹 悦
(安徽理工大学数学与大数据学院 安徽 淮南 232001)

摘 要 随着企业的发展,系统产生并记录的日志越来越多,从繁琐复杂的日志中挖掘块结构的过程变得更加具有挑战性。文中提出了纵向划分日志的方法,该方法极大地减少了每个日志划分的实例数,并缩短了每条迹的长度。该方法被用来处理复杂日志,并从中挖掘出精确的模型。日志划分的基础是活动划分。首先,基于行为关联的思想,提出共同变迁的概念,实现相互关联活动的聚集划分。然后,从日志所含共同变迁的数量的角度出发,用相互区别但又相互交错的方法划分活动集,从而实现模块和日志的划分。所提出的模块和日志划分方法可以迭代进行,直到日志划分得足够简单为止。最后,从每个划分后的简单日志中挖掘出一个块结构,通过组合块结构形成合理的整体系统模型,并通过 Prom 实验验证了所提方法的可行性。

关键词 复杂日志,块结构,共同变迁,日志划分
中图法分类号 O175 **文献标识码** A **DOI** 10.11896/jsjkx.180901710

Approach for Mining Block Structure Process from Complex Log Using Log Partitioning

DUAN Rui FANG Huan ZHAN Yue
(School of Mathematics and Big Data, Anhui University of Science and Technology, Huainan, Anhui 232001, China)

Abstract With the development of enterprises, more and more logs are generated and recorded by the system, and the process of mining block structures from cumbersome and complicated logs becomes more challenging. This paper proposed an approach of vertically dividing logs, greatly reducing the number of instances of each log partition, and shortening the length of each trace to process complex logs and mining accurate models from them. The basis of log division is activity division. Firstly, based on the idea of behavioral association, the concept of common transition is proposed to realize the aggregation division of interrelated activities. Then, from the perspective of the number of common transitions in the log, the activity set is divided by mutually different and interleaved methods, thereby realizing the division of modules and logs. The proposed module and log partitioning method can be iterated until that the log partitioning is simple enough. Finally, a block structure is mined from each divided simple log, and a reasonable overall system model is formed by combining block structures. The feasibility of the proposed method is verified by Prom experiment.

Keywords Complex log, Block structure, Common transition, Log partitioning

高效的企业管理可以提高企业竞争力,而业务流程又是企业管理的重要组成部分。因此,如何从业务系统所记录的日志中获取准确的系统参考模型成为关键。过程挖掘技术(Process Mining,PM)应运而生,它旨在从事件日志中挖掘出合理且满足人们需求的过程模型。

过去人们研究并提出了多种过程挖掘算法,从活动间的顺序关系中发现过程模型。例如,文献[1]提出 α-算法,它是早期被广泛应用的过程挖掘算法;文献[2]基于 α-算法,从事件日志中挖掘不可见任务。一种由 α-算法产生的启发式挖掘算法在文献[3]中被提出,它仅考虑事件的顺序;文献[4]阐述了一种基于文献[3]的改进算法——灵活启发式挖掘算法,它是一种灵活的控制流挖掘算法。这两种算法都能处理噪音和低频行为。

为了提高挖掘的效率,文献[5]提出了遗传挖掘算法,该算法通过一种有效的因果矩阵结构来提高空间搜索的效率。文献[6]使用一种分治策略来递归地构建一个过程模型,直到所有迹都被处理,该方法被称为归纳式算法。有时候可获得的日志是不完备的,为此文献[7]给出一种从不完备日志中发现块结构过程模型的方法,该方法利用对完备性不敏感的概率行为关系,给出一种适用性更强的过程挖掘算法。

发现块结构技术是过程挖掘方法的一大进步。文献[8]通过构建一个可扩展的框架和算法 B 及其改进算法 B',从任何给定的日志中发现合理的、符合可观察到行为的块结构过程模型集。文献[9]提出一种从事件日志中萃取任务块的方法,该方法通过建立日志的间接后继矩阵并分析后继矩阵中值的关系来发现后继模式,进而发现完整的过程模型。流程

到稿日期:2018-09-12 返修日期:2019-01-06 本文受国家自然科学基金项目(61472003,61402011,61572035),安徽省自然科学基金项目(1608085QF149),安徽省高校优秀青年人才支持项目(gxyqZD2018038),安徽省博士后基金项目(2018B288)资助。

段 瑞(1993—),男,硕士,主要研究方向为 Petri 网理论与应用、过程挖掘方法,E-mail:85768312@qq.com;方 欢(1982—),女,博士,副教授,主要研究方向为业务流程管理系统,E-mail:fanghuan0307@163.com;詹 悦(1994—),女,硕士,主要研究方向为 Petri 网理论与应用。

树是一种典型的块结构,文献[10]提出一种通过生成流程树挖掘局部过程模型的方法,该方法依据 5 种标准评估并选择局部过程模型,扩展生成新的流程树,以此迭代,直到任务完成。

随着过程发现技术的不断完善和更新,挖掘方法开始更加重视模型的精确度和质量。文献[11]提出一种提高发现过程模型质量的聚类算法。文献[12]使用基于时间的标签细化来发现更加精确的过程模型,提出一种基于事件的时间属性自动细化标签的方法,该方法可以依据事件的时间属性区分相同事件类型的不同实例的行为。文献[13]通过过滤混沌活动,从事件日志中发现更精确的过程模型,其中提到现实生活中的事件往往存在混沌活动,这些活动不依赖于过程的状态,它们在任意的时间点都可能发生,过滤掉这些活动可以提高挖掘到的过程模型的质量。

随着社会和企业的发展,可得到的日志越来越多,从繁琐复杂的日志中挖掘过程模型变得更加具有挑战性。以往的过程挖掘方法往往存在以下几种局限的一种或多种:1)不能挖掘循环结构,2)不能处理复杂日志,3)从复杂日志中挖掘到的模型不够精确。为了从完备的复杂日志中挖掘出精确的模型,本文提出一种新的基于日志划分的块结构过程挖掘方法。该方法利用动态规划原理,通过日志划分有效地简化了日志,把原日志划分成更简单且易分析处理的日志,从日志划分中挖掘出的块结构为对应日志的精确模型时,组合所有块结构得到的模型即是原日志的精确模型。本文方法通过纵向划分日志,既能减少日志中迹的数量,又能缩短迹,极大程度地简化了日志,且不存在上述局限。

本文第 2 节给出共同变迁的定义及基于共同变迁的日志划分,进而提出基于日志划分的块结构挖掘方法;第 3 节给出无共同变迁的挖掘方法;第 4 节设计了基于日志划分的块结构挖掘算法;第 5 节通过实验验证了该算法的可行性,并对比分析结果;最后总结全文并展望未来工作。

1　基本概念

定义 1(标签 Petri 网)　满足下列条件的一个五元组 $LN=(P,T;F,\Sigma,\ell)$ 被称作标签 Petri 网,其中,P 是库所,T 是变迁,F 是流关系,Σ 是表示变迁的标签集合。

(1) $P\cup T\neq\emptyset$;

(2) $P\cap T=\emptyset$;

(3) $F\subseteq(P\times T)\cup(T\times P)$;

(4) $\ell:T\to\Sigma\cup\{\varepsilon\}$ 是标签函数。

记 $X=P\cup T$,对于 $x\in X$,x 的前集记为 $\bullet x=\{y\in X\mid(y,x)\in F\}$,$x$ 的后集记为 $x\bullet=\{y\in X\mid(x,y)\in F\}$,$\bullet(x\bullet)=\{z\in X\mid y\in X\wedge(x,y)\in F\wedge(z,y)\in F\}$。一个 Petri 网是自由选择的,当且仅当 $\forall p\in P\wedge|p\bullet|>1\Rightarrow\bullet(p\bullet)=\{p\}$。

用 M 表示一个标签 Petri 网的标识,$M(p)$ 表示库所 $p\in P$ 的 token 数,$M(p)>0$ 表示 p 被 M 标记。变迁 $t\in T$ 能引发,如果 $\forall p'\in\bullet t$,有 $M(p')>0$,记作 $M[t>$。一个 Petri 网是安全的,当且仅当 $\forall M'$ 是一个可达标识,$\forall p\in P\Rightarrow M'(p)\leqslant 1$。

定义 2(事件日志)　$e\in\Sigma$ 是一个事件,$\sigma\in L$ 是一条迹,$\sigma\in\Sigma^*$ 是由一系列事件组成的非空序列,且 $\exists M''$,有 $M''[\sigma>$,事件日志 $L\in P(\Sigma^*)$ 是迹的多重集合,$P(\Sigma^*)$ 是 Σ^* 的幂集,$L\subseteq\Sigma^*$。

定义 3(工作流网)　满足下列条件的三元组 $WFN=(N,i,o)$ 被称作工作流网,其中 $N=LN=(P,T;F,\Sigma,\ell)$ 是标签 Petri 网,$i\in P$ 是输入库所,$o\in P$ 是输出库所,Σ 是表示变迁的标签集合。

(1) $\bullet i=\emptyset$,$o\bullet=\emptyset$,$i\neq o$;

(2) M_i 和 M_o 分别为初始和最终标识,$M_i(i)=M_o(o)=1$;

(3) σ 是 WFN 的一条迹,满足 $M_i\xrightarrow{\sigma}M_o$,即 $M_i[\sigma>M_o$。

活动划分是简化日志的关键,简化日志的方法是映射,迹在活动集上的映射记为 $\sigma\uparrow\Sigma$,如 $\sigma=\langle abcde\rangle$,$\Sigma=\{a,c,e\}$,则 $\sigma\uparrow\Sigma=\langle ace\rangle$。日志在活动集上的映射记为 $L\uparrow\Sigma$,每个 $\sigma\in L$,$\sigma\uparrow\Sigma=\sigma'$,所有 σ' 组成的新日志 $L'=L\uparrow\Sigma$。

2　基于共同变迁的块挖掘方法

为了处理复杂日志,把活动和日志划分成有利于精确挖掘的划分。本文模仿计算机算法中常用的动态规划原理,动态规划常常被用来求解最优化问题。适合应用动态规划方法求解的最优化问题应具备两个要素:最优子结构和子问题重叠。最优子结构是问题的最优解包含其子问题的最优解。假设已知一个给定复杂日志的最优模型,则该模型的每个块结构是其对应日志的最优模型,且每个块结构可以独立挖掘。利用"剪切-粘贴"技术证明:假定某些块结构不是其对应日志的最优模型,则从原复杂日志的最优模型中"剪切"掉这些块结构,并"粘贴"上它们的最优模型,得到一个更优模型,这与前提假设相矛盾。子问题重叠是反复求解相同的子问题,本文提出的过程挖掘方法不会出现完全相同的日志划分,但会出现具有相同划分条件的日志。精确模型可以看作最优模型(最优解)的近似解。

本文的主要贡献有:1)在观察分析日志时,发现有些活动出现在所有迹中,且是有序的,称为共同变迁,利用共同变迁把所有活动划分成有限个活动集,并依据活动划分得出模块划分和日志划分;2)从处理复杂日志的能力及挖掘块结构两个方面考虑,提出一种划分日志的方法,使得每个日志划分对应一个块结构(精确模型),且通过组合所有块结构最终能够得到一个合理的模型,即能够从复杂日志中挖掘出满足人们需求、合理且精确的模型;3)给出一种具体的递归挖掘算法,并通过实验验证了算法的可行性。

2.1　基于共同变迁的活动划分

定义 4(共同变迁,开始变迁,结束变迁)　L_N 是工作流网 $WFN=(N,i,o)$ 的日志,称满足下列条件的活动集 Σ_t 为共同变迁集:1) $\Sigma_t=\{x/\forall\sigma\in L_N[x\in\sigma]\}$;2) Σ_t 是一个有序的集合,即 $\forall j<j'$,$\forall\sigma\in L_N$,$\sigma\uparrow\{\Sigma_t(j),\Sigma_t(j')\}=\langle\Sigma_t(j),\Sigma_t(j')\rangle$,$\Sigma_t(j)$ 表示 Σ_t 中的第 j 个元素。$\forall\sigma\in L_N$,$\sigma(1)$ 是开始变迁,$\sigma(|\sigma|)$ 是结束变迁,L_N 可能含有多个开始和结束变迁,它们的集合分别记为 Σ_i 和 Σ_o。

定义 5(基于共同变迁的活动划分)　L_N 是工作流网 $WFN=(N,i,o)$ 的日志,Σ_L 是所有活动集,Σ_t 是共同变迁集,Σ_i 是输入变迁集,Σ_o 是输出变迁集,当 $\Sigma_t\backslash(\Sigma_i\cup\Sigma_o)\neq\emptyset$ 时,划分活动集 Σ_L,$j=0,1,2,\cdots,|\Sigma_t|$,$\forall\sigma\in L_N$,必有 $\Sigma_t(j)=\sigma(j')$,

$\Sigma_t(j+1)=\sigma(j'')$，则 Σ_L 的 j 划分为 $\Sigma_j=\{\sigma(j'+1),\cdots,\sigma(j''-1)\}$。若 $\sigma(j'+1)=\sigma(j'')$，$\Sigma_j=\emptyset$。其中，$\forall\sigma\in L_N$，$\Sigma_t(0)=\Sigma_t(|\Sigma_t|+1)=\sigma(0)=\sigma(|\sigma|+1)=\varepsilon$，$\varepsilon$ 是不存在变迁，即$\{\varepsilon\}=\emptyset$。

以一个实例日志 L 进行说明：$L=\{\langle ABCBCBDFH\rangle,\langle ABCBDEGH\rangle,\langle ABCBDGEH\rangle,\langle ABCBDFH\rangle,\langle ABDEGH\rangle,\langle ABDGEH\rangle,\langle ABDFH\rangle\}$。分析日志 L，$\Sigma_t=\{A,D,H\}$，$\Sigma_i=\{A\}$，$\Sigma_o=\{H\}$，$\Sigma_t\backslash(\Sigma_i\cup\Sigma_o)=\{D\}$，划分 Σ_t，有 $\Sigma_1=\{B,C\}$，$\Sigma_2=\{E,F,G\}$，$\Sigma_0=\Sigma_3=\emptyset$，共同变迁和划分块之间有一定的行为关系，记为 $A\Rightarrow\Sigma_1\Rightarrow D\Rightarrow\Sigma_2\Rightarrow H$，符号$\Rightarrow$表示模糊的顺序关系，需进行具体判定。

2.2 基于活动划分的模块划分

利用共同变迁划分活动之后，为了简化日志，需要进一步划分模块，考虑每个活动划分及其两端的变迁对应一个模块（左右两端可能有一端无共同变迁）。

定义 6（基于共同变迁的模块划分） L_N 是工作流网 $WFN=(N,i,o)$ 的日志，Σ_t 是共同变迁集，Σ_i 是输入变迁集，Σ_o 是输出变迁集，$\Sigma_0,\Sigma_1,\cdots,\Sigma_n$ 是活动划分，$1\leqslant n\leqslant|\Sigma_t|+1$。若 $\Sigma_j=\emptyset$，$\Sigma_t(j)$ 和 $\Sigma_t(j+1)$ 组成一个顺序模块。类似地，若有连续的 $\Sigma_j=\emptyset,\Sigma_{j+1}=\emptyset,\cdots$，则 $\Sigma_t(j),\Sigma_t(j+1),\cdots$，组成一个顺序模块。对 $0\leqslant j\leqslant|\Sigma_t|$，若 $\Sigma_j\neq\emptyset$，划分对应的模块 $M_j=\Sigma_t(j)\Rightarrow\Sigma_j\Rightarrow\Sigma_t(j+1)$，$M_j$ 的活动集合记为 Σ_{M_j}。

以日志 L 为例，$\Sigma_1=\{B,C\}$，因此模块 $M_1=A\Rightarrow\{B,C\}\Rightarrow D$；同理，模块 $M_2=D\Rightarrow\{E,F,G\}\Rightarrow H$，$M_0=M_3=\emptyset$。由活动划分产生模块划分，进而产生日志划分，日志划分是一种基于活动划分和模块划分的日志简化方法。

2.3 基于活动划分和模块划分的日志划分

定义 7（基于共同变迁的日志划分） L_N 是工作流网 $WFN=(N,i,o)$ 的日志，Σ_t 是共同变迁集，Σ_i 是输入变迁集，Σ_o 是输出变迁集，$\Sigma_0,\Sigma_1,\cdots,\Sigma_n$ 是活动划分，$M_0,M_1,\cdots,M_n$ 是模块划分，则日志 L_N 的 j 划分 $L_j=L_N\uparrow\Sigma_{M_j}$。

以 L 为例，$\Sigma_{M_1}=\{A,B,C,D\}$，$\Sigma_{M_2}=\{D,E,F,G,H\}$，日志划分为 $L_1=\{\langle ABD\rangle^3,\langle ABCBD\rangle^3,\langle ABCBCBD\rangle\}$，$L_2=\{\langle DFH\rangle^3,\langle DEGH\rangle^2,\langle DGEH\rangle^2\}$。

直观上看，完备日志越简单，挖掘“失误”越少，模型越精确。当日志足够简单时，使用文献[1,3,6]中所提方法能挖掘出精确的模型。利用共同变迁把日志 L 划分成两个日志，每个日志相比于原日志都更简洁且更易处理。从每个日志划分中能发现一个块结构，利用块结构之间的重复共同变迁组合所有块结构，得到最终的工作流网。

2.4 基于日志划分的块结构过程挖掘

划分日志是简化日志的过程，划分日志得到若干简单的日志划分，基于每个日志划分可以轻松挖掘到一个工作流网。以日志 L 为例，基于 L 的两个划分 L_1、和 L_2 能够挖掘到两个模型，分别为 M_1 和 M_2，组合它们得到的模型如图 1 所示。组合的方法是：必有某两个块结构带有相同的共同变迁，重合相同的共同变迁即可。

从图 1 中看出，该方法挖掘到的块结构相交，相交处为共同变迁。结合日志划分 L_2，从图 1 中的模块 M_2 发现：M_2 仍然可以继续划分。L_2 中只含有开始变迁 D 和结束变迁 H 两个共同变迁，不满足定义 5 中的 $\Sigma_t\backslash(\Sigma_i\cup\Sigma_o)\neq\emptyset$，因此不能利用共同变迁进行活动划分。

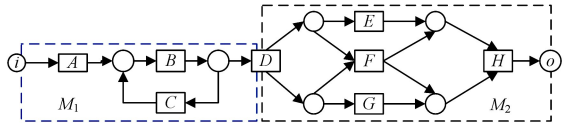


图 1 日志 L 的工作流网

Fig. 1 Workflow net of log L

3 无共同变迁的块结构挖掘方法

不是所有日志都含有共同变迁，本文称这类日志为无共同变迁类日志，把只含有共同的开始变迁或共同的结束变迁的日志也归为无共同变迁类，即满足 $\Sigma_t\backslash(\Sigma_i\cup\Sigma_o)=\emptyset$ 的日志。这类日志不能进行基于共同变迁的活动划分和日志划分。下面给出此类日志的划分方法。

3.1 无共同变迁的活动划分

按一定条件把日志 L 中所有的迹分成不同集合，每个集合组成的日志 $\vec{L}$ 被称为日志 L 的条件子日志，$\vec{L}\subseteq L$。以一个新的实例日志 L' 进行说明：$L'=\{\langle abcefglglgh\rangle,\langle abcej\rangle,\langle acbefgh\rangle,\langle abcefgh\rangle,\langle dj\rangle,\langle dfgh\rangle,\langle abcefglgh\rangle,\langle acbefglglgh\rangle,\langle dfglgh\rangle,\langle acbefglglglgh\rangle,\langle dfglglgh\rangle,\langle acbej\rangle\}$。

定义 8（无共同变迁的活动划分） L_N 是工作流网 $WFN=(N,i,o)$ 的日志，Σ_t 是共同变迁集，Σ_i 是输入变迁集，Σ_o 是输出变迁集，若 $\Sigma_t\backslash(\Sigma_i\cup\Sigma_o)=\emptyset$，对于 $1\leqslant\forall j\leqslant|L_N|$，按条件 $\forall\sigma,\sigma'\in\vec{L}_j[\sigma(1)=\sigma'(1)\wedge\sigma(|\sigma|)=\sigma'(|\sigma'|)]$挑选 L_N 的所有条件子日志 $\vec{L}_1,\vec{L}_2,\cdots,\vec{L}_n$，满足 $j'\neq j''[\vec{L}_{j'}\cap\vec{L}_{j''}=\emptyset]$。它们的活动集合为 $\vec{\Sigma}_1,\vec{\Sigma}_2,\cdots,\vec{\Sigma}_n$。称活动集 $\Sigma_j=\{x/x\in\vec{\Sigma}_{j'}\wedge x\notin\vec{\Sigma}_{j''}\wedge j'\neq j''\}$为 Σ_L 的一个活动划分。

值得注意的是：1）若 $|\Sigma_i|=|\Sigma_o|=1$，则划分 Σ_L 为 Σ_i，$\Sigma_L\backslash(\Sigma_i\cup\Sigma_o)$，$\Sigma_o$，利用定义 8 划分活动集 $\Sigma_L\backslash(\Sigma_i\cup\Sigma_o)$；2）若 $|\Sigma_i|=1\wedge|\Sigma_o|>1$，则划分 Σ_L 为 Σ_i，$\Sigma_L\backslash\Sigma_i$，利用定义 8 划分活动集 $\Sigma_L\backslash\Sigma_i$；3）若 $|\Sigma_i|>1\wedge|\Sigma_o|=1$，则划分 Σ_L 为 $\Sigma_L\backslash\Sigma_o$，$\Sigma_o$，利用定义 8 划分活动集 $\Sigma_L\backslash\Sigma_o$；4）由 $j'\neq j''$可知，依据某两个条件子日志的活动集产生的两个活动划分是排它选择的。

以日志 L'为例，挑选 L'的条件子日志，$\vec{L}_1=\{\langle dj\rangle\}$，$\vec{L}_2=\{\langle abcej\rangle,\langle acbej\rangle\}$，$\vec{L}_3=\{\langle dfgh\rangle,\langle dfglgh\rangle,\langle dfglglgh\rangle\}$，$\vec{L}_4=\{\langle abcefgh\rangle,\cdots\}$。对应的活动集分别为：$\vec{\Sigma}_1=\{d,j\}$，$\vec{\Sigma}_2=\{a,b,c,e,j\}$，$\vec{\Sigma}_3=\{d,f,g,l,h\}$，$\vec{\Sigma}_4=\{a,b,c,e,f,g,l,h\}$。因为 $d\in\vec{\Sigma}_1\wedge d\notin\vec{\Sigma}_2$，所以 $\Sigma_1=\{d\}$ 是日志 L'的一个划分；同理，日志 L'的活动划分有：$\Sigma_2=\{j\}$，$\Sigma_3=\{f,g,l,h\}$，$\Sigma_4=\{a,b,c,e\}$，$\Sigma_5=\{d,f,g,l,h\}$，$\Sigma_6=\{a,b,c,e,j\}$。

3.2 基于活动划分的日志划分

活动划分完成之后，与基于共同变迁的模块划分不一样的是每个活动划分即对应一个模块。参考定义 7，日志 L_N 在每个活动划分上的映射即为一个日志划分。

以日志 L'为例，L'的 6 个活动划分对应了 6 个日志划分，分别为：$L_1'=\{\langle d\rangle^4\}$，$L_2'=\{\langle j\rangle^3\}$，$L_3'=\{\langle fgh\rangle^3,\langle fglgh\rangle^3,\langle fglglgh\rangle^3\}$，$L_4'=\{\langle abce\rangle^4,\langle acbe\rangle^4\}$，$L_5'=\{\langle dfgh\rangle,\langle dfglgh\rangle,\langle dfglglgh\rangle\}$，$L_6'=\{\langle abcej\rangle,\langle acbej\rangle\}$。

日志 L'有 6 个模块划分，分别为：$M_1=d$，$M_2=j$，$M_3=$

$f \Rightarrow g \Rightarrow l \Rightarrow h$,$M_4 = a \Rightarrow b \Rightarrow c \Rightarrow e$,$M_5 = d \Rightarrow M_3$,$M_6 = M_4 \Rightarrow j$。由前文可知:$\times(M_1, M_4)$,$\times(M_2, M_3)$,$\times$表示排他。这样共产生 6 个模块和 2 个约束行为关系。由 M_5 和 M_6 可知:$\rightarrow(M_1, M_3)$,$\rightarrow(M_4, M_2)$,$\rightarrow$表示顺序,即 $M_5 = M_1 \Rightarrow M_3$,$M_6 = M_4 \Rightarrow M_2$。

图 2 是日志 L'的工作流网,组合的方法是:模块间的行为关系决定了它们的相对位置和连接方式,在模块间添加一些库所和流关系,使得模块间的行为关系得到满足。从图 2 中可以看出:模型中还存在$\rightarrow(M_4, M_3)$和$\rightarrow(M_1, M_2)$两种行为关系,由排它选择关系$\times(M_1, M_4)$和$\times(M_2, M_3)$、顺序关系$\rightarrow(M_1, M_3)$和$\rightarrow(M_4, M_2)$即可推出。

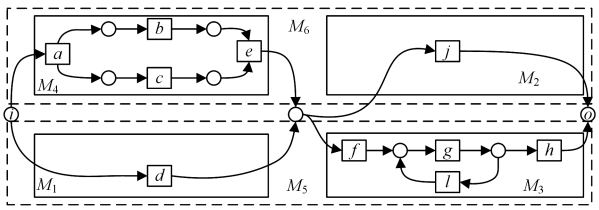


图 2　日志 L'的工作流网

Fig. 2　Workflow net of log L'

3.3　日志划分的再划分

分析图 2 可知,模块 M_4 还能继续分块,把日志 L'的日志划分 L_4'当作原日志,使用本文的日志划分方法划分 L_4',L_4'的共同变迁集为$\{a, e\}$,则把 L_4'划分成 3 个日志划分,对应模块 $M_4 = a \Rightarrow \{b, c\} \Rightarrow e$。模块 M_3 是由循环结构构成的模型,活动划分为$\{f\}$,$\{g, l\}$,$\{h\}$,$M_3 = f \Rightarrow \{g, l\} \Rightarrow h$。其实,$L_3'$和 L_4'已经足够简单,不需要再进行日志划分就能挖掘出精确的过程模型。基于日志划分的块结构挖掘方法中的块结构可能由一个或多个变迁组成,若每一个块结构对应的日志划分仍比较复杂,则可以继续划分该日志划分,直到每个日志划分都不再复杂,这是一个反复调用自身的递归过程。

4　块结构挖掘算法

基于日志划分的块结构挖掘方法是针对复杂日志的。为了提高该方法的适用性,本文给出了适用所有日志的挖掘算法。在挖掘之前,需要辨别日志的类型,再进行划分。下面将给出日志划分算法,以划分不同类型的日志,并从中挖掘出精确的块结构模型,该算法简记为 LP 算法。

4.1　算法设计

算法 1　LP

Block-Structure-Mining-Algorithm-based-on-Log-Partition(L)

1. 当日志 L 足够简单,或对应一个简单的块结构时
2.　利用已有的成熟方法从 L 中挖掘出精确的块结构
3. 计算 L 的活动集、共同变迁集、开始和结束变迁集,分别记作 Σ_L,Σ_t,Σ_i,Σ_o
4. if $\Sigma_t \neq \Sigma_L \wedge \Sigma_t \backslash (\Sigma_i \cup \Sigma_o) \neq \emptyset$
5.　for j=0 to $|\Sigma_t|$
6.　　$\exists j', j'', \forall \sigma \in L, \Sigma_t(j) = \sigma(j') \wedge \Sigma_t(j+1) = \sigma(j'')$
7.　　$\Sigma_j = \{\sigma(j'), \cdots, \sigma(j''-1)\}$
8.　　$M_j = \Sigma_t(j) \Rightarrow \Sigma_j \Rightarrow \Sigma_t(j+1)$
9.　　calculate activity set of M_j, recorded as Σ_{M_j}
10.　　$L_j = L \uparrow \Sigma_{M_j}$
11.　　Block-Structure-Mining-Algorithm-based-on-Log-Partition(L_j)
12. if $\Sigma_t \backslash (\Sigma_i \cup \Sigma_o) = \emptyset \wedge |\Sigma_i| > 1 \wedge |\Sigma_o| > 1$
13.　按下述条件选择所有条件子日志:$\forall \sigma, \sigma' \in \vec{L}_j [\sigma(1) = \sigma'(1) \wedge \sigma(|\sigma|) = \sigma'(|\sigma'|)]$
14.　计算$\vec{L}_j$的活动集,记作$\vec{\Sigma}_j$
15.　按下述条件计算活动划分:$\Sigma_j = \{x / x \in \vec{\Sigma}_{j'} \wedge x \notin \vec{\Sigma}_{j''} \wedge j' \neq j''\}$,记作 Σ_j
16.　for each activity partition Σ_j
17.　　$L_j = L \uparrow \Sigma_j$
18.　　Block-Structure-Mining-Algorithm-based-on-Log-Partition(L_j)
19. if $\Sigma_t \backslash (\Sigma_i \cup \Sigma_o) = \emptyset \wedge |\Sigma_i| = 1 \wedge |\Sigma_o| = 1$
20.　$L_m = L \uparrow (\Sigma_t \backslash (\Sigma_i \cup \Sigma_o))$
21.　Block-Structure-Mining-Algorithm-based-on-Log-Partition(L_m)
22. if $\Sigma_t \backslash (\Sigma_i \cup \Sigma_o) = \emptyset \wedge |\Sigma_i| = 1 \wedge |\Sigma_o| > 1$
23.　$L_m = L \uparrow (\Sigma_t \backslash \Sigma_i)$
24.　Block-Structure-Mining-Algorithm-based-on-Log-Partition(L_m)
25. if $\Sigma_t \backslash (\Sigma_i \cup \Sigma_o) = \emptyset \wedge |\Sigma_i| > 1 \wedge |\Sigma_o| = 1$
26.　$L_m = L \uparrow (\Sigma_t \backslash \Sigma_o)$
27.　Block-Structure-Mining-Algorithm-based-on-Log-Partition(L_m)

分析 LP 算法,第 1－2 行为递归出口,当日志足够简单,用已有方法能挖掘出精确的模型时,从日志中挖掘精确的块结构模型。第 3 行计算日志的所有活动集、共同变迁集、输入变迁集及输出变迁集。第 4－11 行基于共同变迁集划分日志,并从中挖掘出精确的块结构,其中第 4 行为判断条件,第 6－7 行为活动划分,第 8 行为模块划分,第 9－10 行为日志划分,第 11 行递归调用 LP 算法处理每个日志划分。第 12－18 行处理一种特殊的日志,即包含选择结构的日志,选择结构内部可能包含其他结构,其中第 12 行为判断条件,第 13 行挑选所有条件子日志,第 14－17 行计算日志划分,第 18 行递归调用 LP 算法处理每个日志划分。剩余 LP 算法的所有语句用于处理一些其他特殊的日志,其中第 19－21 行处理除开始和结束变迁外共同变迁集为空的情况,第 21 行即第 12－18 行处理的情况,第 22－24 行与第 25－27 行类似。

定理 1(精确性)　假定 LP 算法是正确的,若 LP 算法输出的块结构是精确的,那么由所有块结构组成的模型也是精确的。

证明:(反证法)假设由所有块结构组成的模型是不精确的,因为块结构的组合是基于日志的行为关系进行的,所以该不精确的模型中必有某一处或多处不精确,这与定理假设条件相矛盾。当日志完备且足够简单时,利用已有的成熟过程挖掘方法能够挖掘出精确的模型,因此,定理的假设条件显然也是成立的。第 5 节将通过实验验证算法的正确性和可行性。

LP 算法的输入为完备日志或完备复杂日志,输出为一系列精确的块结构,这些块结构之间总有某两个具有共同变迁或具有顺序之外的排他关系。基于此,组合所有块结构,即从日志或复杂日志中得到精确的带有块结构的模型(定理 1)。

4.2　算法复杂度

本节分析 LP 算法性能的另一个表现:算法复杂度。给定一个日志 L,L 包含的实例数为 m,包含的活动数为 n,现以 L 为输入分析算法的复杂度。LP 算法的核心是划分日志,对于一个包含 n 个活动的日志 L,经过一系列日志划分操作后,最多能够得到 $n + \lfloor n/2 \rfloor \cdot \lceil n/2 \rceil$ 个日志划分,其中$\lfloor x \rfloor$表示小于 x 的最大整数,$\lceil x \rceil$表示大于 x 的最小整数。划分日志的主要时间花费为映射,L 共含有 m 条实例,则划分日志的

时间复杂度为 $O(m(n+\lfloor n/2 \rfloor \cdot \lceil n/2 \rceil))$。在实际运行中，算法复杂度为 $O(mn')$，其中 n' 是远小于 $n+\lfloor n/2 \rfloor \cdot \lceil n/2 \rceil$ 的数，在多数情况下，一个包含 n 个活动的日志 L 的日志划分数小于或远小于 n。因此，在实际运行中的大多数情况下，算法复杂度为 $O(mn')$，其中 n' 小于或远小于 n，可以看出，算法在划分日志上花费的时间并不多，时间主要花费在从日志划分中挖掘出精确的模型过程中。本文使用已有的成果处理问题——从日志划分中挖掘出精确的模型。

5 实验设计与分析

5.1 实验设计

本文以 ProM 框架[14]为工具，把一个银行转账系统产生的复杂日志 L_B 作为输入，来验证本文所提方法。该日志包含 111 个活动、2 000 条迹、128 460 个事件，每条迹平均包含 64 个事件和 59 个活动类型。限于篇幅，本文不具体给出日志及变迁的含义。

输入日志 L_B，模拟 LP 算法在 ProM 框架中应用产生工作流网的步骤。

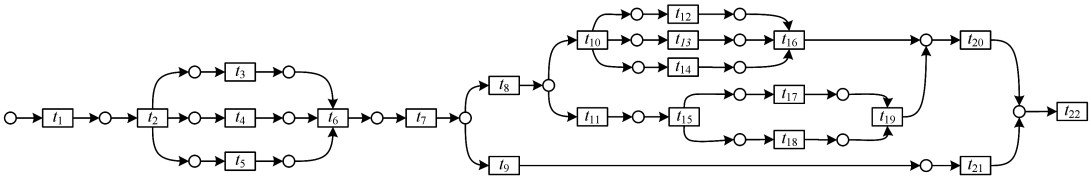
执行算法第 3－4 行。$\Sigma_t=\{t_1,t_2,t_6,t_7,t_{22},t_{61},t_{62},t_{69},t_{84},t_{85},t_{99},t_{100},t_{110},t_{111}\}$ 是 L_B 的共同变迁集，开始变迁集 $\Sigma_i=\{t_1\}$，结束变迁集 $\Sigma_o=\{t_{111}\}$。

执行算法第 7－8 行。划分 L_B 的活动：$\Sigma_2,\Sigma_4,\Sigma_5,\Sigma_7,\Sigma_8,\Sigma_9,\Sigma_{10},\Sigma_{12},\Sigma_0=\Sigma_1=\Sigma_3=\Sigma_6=\Sigma_{11}=\Sigma_{13}=\Sigma_{14}=\emptyset$。

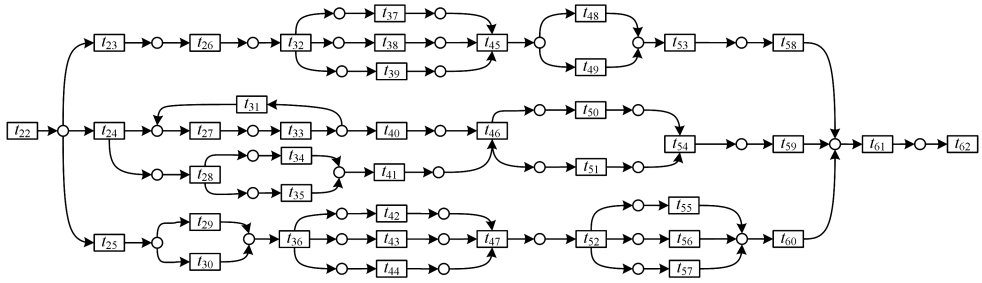
执行算法第 9 行。对应的模块：$M_0=M_{14}=\emptyset$，$M_1=t_1\Rightarrow t_2$，$M_2=t_2\Rightarrow\Sigma_2\Rightarrow t_6$，$M_3=t_6\Rightarrow t_7$，$M_4=t_7\Rightarrow\Sigma_7\Rightarrow t_{22}$，$M_5=t_{22}\Rightarrow\Sigma_5\Rightarrow t_{61}$，$M_6=t_{61}\Rightarrow t_{62}$，$M_7=t_{62}\Rightarrow\Sigma_7\Rightarrow t_{69}$，$M_8=t_{69}\Rightarrow\Sigma_8\Rightarrow t_{84}$，$M_9=t_{84}\Rightarrow t_{85}$，$M_{10}=t_{85}\Rightarrow\Sigma_{10}\Rightarrow t_{99}$，$M_{11}=t_{99}\Rightarrow t_{100}$，$M_{12}=t_{100}\Rightarrow\Sigma_{12}\Rightarrow t_{110}$，$M_3=t_{110}\Rightarrow t_{111}$。

执行算法第 10－11 行。划分日志 L_B 为 14 个日志划分：$L_1,\cdots,L_{13}$。

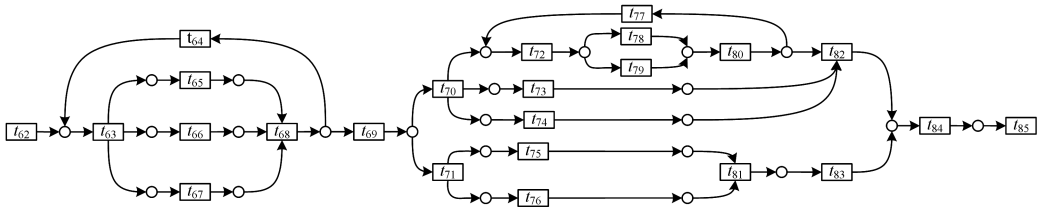
执行算法第 12 行。调用该算法处理 14 个日志划分，其中处理 L_4,L_5,L_8,L_{10} 时，先执行算法第 20－21 行，再执行算法第 22 行，然后执行算法第 13－19 行，分别产生日志划分：$L_{4.1},L_{4.2};L_{5.1},L_{5.2},L_{5.3};L_{8.1},L_{8.2};L_{10.1},L_{10.2}$。其中 $L_{4.1}$ 仍能继续划分为 $L_{4.1.1},L_{4.1.2}$。每个日志划分都对应着一个精确的块结构，正如前面所言，总有某两个块结构之间存在着共同变迁或具有顺序之外的排他关系。组合所有块结构，得到精确的模型，如图 3 所示。



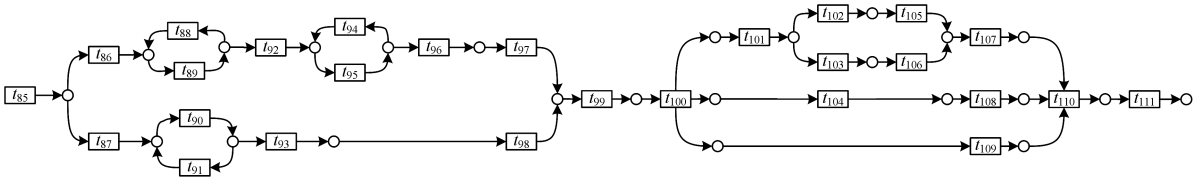
(a)银行转账模型的模块 1－4



(b)银行转账模型的模块 5－6



(c)银行转账模型的模块 7－9



(d)银行转账模型的模块 10－13

图 3 银行转账的块结构模型

Fig. 3 Block structure model of bank transfer

5.2　实验结果分析

图 3 是 LP 算法从银行转账的复杂日志中挖掘到的块结构组合而成的精确模型,该模型具有明显的块结构。为了便于直观、清晰地观察模型,把它分成 4 个子图,每个子图由若干块结构组成。该模型包含了日志中的所有可观测行为,验证了本文所设计算法的正确性和可行性。LP 算法把日志纵向划分成若干个日志划分,日志划分之间存在一定的行为关系,日志划分内及日志划分间的行为关系构成日志的所有行为,这保证了用 LP 算法划分日志不会遗漏行为关系。

当日志划分足够简单时,从日志划分中能够挖掘出精确的模型(块结构),依据动态规划原理,子问题的最优解组合成原问题的最优解,通过组合所有块结构得到一个原日志的精确模型。每个日志划分对应的块结构可以独立挖掘,选择一种已有的成熟挖掘方法,如文献[1,3,6]所提方法,可以从足够简单的日志划分中挖掘出精确的模型。这与定理 1 相互印证。

当日志简单或比较简单时,使用本文的块结构过程挖掘方法其实就是直接执行 LP 算法的第 1－2 行,即使用已有的成熟挖掘方法处理日志;但当日志复杂或者非常复杂时,本文方法的优势就突显出来了,这是因为本文方法能有效处理复杂日志(5.1 节实验验证了 LP 算法的正确性)。因此,存在一个日志复杂度交叉点,把日志分为交叉点之前和交叉点之后。

结束语　本文用两个实例日志 L 和 L' 说明了基于日志划分的块结构挖掘方法,设计了 LP 算法,并用实验验证了算法的可行性。把一个复杂繁琐的日志纵向划分成有限个日志划分,从日志划分中可以挖掘出精确的块结构,依据定理 1,组合块结构得到的最终模型也是精确的,即从复杂日志中挖掘出精确的模型。

未来,希望通过进一步研究基于日志划分的块结构过程挖掘方法和其他过程挖掘方法,分析与比较各类挖掘算法的复杂度量化指标,给出一种具体的界定范围,指标化定义 LP 算法第 1－2 行。

参 考 文 献

[1] AALST W V D, WEIJTERS T, MARUSTER L. Workflow Mining: Discovering Process Models from Event Logs[J]. IEEE Transactions on Knowledge & Data Engineering, 2004, 16(9): 1128-1142.

[2] WEN L, WANG J, SUN J. Mining invisible tasks from event logs[C]// Joint, Asia-Pacific Web and, International Conference on Web-Age Information Management Conference on Advances in Data and Web Management. Springer-Verlag, 2007: 358-365.

[3] WEIJTERS A, AALST W V D. Process mining with the heuristics miner-algorithm[J]. Eindhoven University of Technology, 2006, 166: 1-34.

[4] WEIJTERS A J M M, RIBEIRO J T S. Flexible Heuristics Miner (FHM)[C]// Computational Intelligence and Data Mining. IEEE, 2011: 310-317.

[5] VANDEN BROUCKE S K L M, VANTHIENEN J, BAESENS B. Declarative process discovery with evolutionary computing [C]// Evolutionary Computation. IEEE, 2014: 2412-2419.

[6] LEEMANS S J J, FAHLAND D, AALST W M P V D. Scalable process discovery and conformance checking[J]. Software & Systems Modeling, 2018, 17(2): 599-631.

[7] LEEMANS S J J, FAHLAND D, AALST W M P V D. Discovering Block-Structured Process Models from Incomplete Event Logs[C]// International Conference on Applications and Theory of Petri Nets and Concurrency. Springer International Publishing, 2014: 91-110.

[8] LEEMANS S J J, FAHLAND D, AALST W M P V D. Discovering block-structured process models from event logs—a constructive approach[C]// International Conference on Application and Theory of Petri Nets and Concurrency. Springer-Verlag, 2013: 311-329.

[9] BOUSHABA S, KABBAJ M I, BAKKOURY Z. Process discovery: Automated approach for block discovery[C]// International Conference on Evaluation of Novel Approaches To Software Engineering. IEEE, 2015: 1-8.

[10] TAX N, SIDOROVA N, HAAKMA R, et al. Mining local process models[J]. Journal of Innovation in Digital Ecosystems, 2016, 3(2): 183-196.

[11] WEERDT J D, BROUCKE S V, VANTHIENEN J, et al. Active Trace Clustering for Improved Process Discovery[J]. IEEE Transactions on Knowledge & Data Engineering, 2013, 25(12): 2708-2720.

[12] TAX N, ALASGAROV E, SIDOROVA N, et al. Time-Based Label Refinements to Discover More Precise Process Models [J]. Journal of Ambient Intelligence and Smart Environments, 2017, 11(2): 1001-1015.

[13] TAX N, SIDOROVA N, AALST W M P V D. Discovering more precise process models from event logs by filtering out chaotic activities[J]. Journal of Intelligent Information Systems, 2019, 52(1): 107-139.

[14] VAN DONGEN B F, DE MEDEIROS A K A, VERBEEK H M W, et al. The ProM Framework: A New Era in Process Mining Tool Support[C]// International Conference on Applications and Theory of Petri Nets. Springer-Verlag, 2005: 444-454.