# 一种基于案例簇和同步核的过程发现算法

鲁法明[1,2,3], 曾庆田[1,2], 段 华[2], 刘 聪[2]
(1. 山东科技大学矿山灾害预防控制省部共建国家重点实验室培育基地, 山东青岛 266590;
2. 山东科技大学, 山东青岛 266590; 3. 同济大学嵌入式系统与服务计算教育部重点实验室, 上海 200092)

**摘 要:** 任务依赖关系随案例属性值变化而不同以及不可见任务的挖掘是过程发现中的两个难点. 为解决上述问题, 本文提出一种基于案例簇和同步核的过程发现方法. 首先, 分案例簇挖掘业务过程子模型中的任务依赖关系, 借助同步核对这些子过程模型进行建模; 之后, 提出一种基于返回核的循环结构建模方法, 并给出从组合案例挖掘返回核的算法; 最后, 将各案例簇对应的同步核与返回核进行集成, 并将集成后的同步核转换为 WF-net 模型, 由此实现了业务过程 WF-net 模型的重构. 本文方法可有效解决不可见任务的挖掘以及任务依赖关系随案例属性值变化而不同的问题.

**关键词:** 过程挖掘; 工作流网; 案例簇
**中图分类号:** TP301 **文献标识码:** A **文章编号:** 0372-2112 (2015)06-1127-10
**电子学报 URL:** http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2015.06.014

# A Process Discovery Algorithm Based on Case Clusters and Synchronization Cores

LU Fa-ming[1,2,3], ZENG Qing-tian[1,2], DUAN Hua[2], LIU Cong[2]
(1. *State Key Laboratory of Mining Disaster Prevention and Control Co-founded by Shandong Province and the Ministry of Science and Technology, Shandong University of Science and Technology, Qingdao, Shandong* 266590, *China*;
2. *Shandong University of Science and Technology, Qingdao, Shandong* 266590, *China*; 3. *The Key Laboratory of Embedded System and Service Computing, Ministry of Education, Tongji University, Shanghai* 200092, *China*)

**Abstract:** The fact that task dependencies may change with case attribute values and the existence of invisible tasks makes it difficult to discover a correct process model from event logs. To address these obstacles, a novel process discovery technique based on case clusters and synchronization cores is presented. The task dependencies among case clusters are first mined, and the sub-process model corresponding to case clusters is obtained by synchronization cores. Then, return cores is proposed to model loop structures and the discovery algorithm of return cores from composite cases is provided. Finally, the synchronization cores and the return cores are integrated, and the integrated synchronization cores are transformed to WF-nets to achieve the business process reconstruction. The proposed technique can solve both the mining problem of invisible tasks and the problem that task dependencies change with attribute values.

**Key words:** process mining; workflow net; case cluster

## 1 引言

过程挖掘技术从事件日志抽取知识以发现、检测和改进业务过程[1]. 随着企业对业务过程改进诉求的不断增强, 过程挖掘技术受到越来越多的关注[2~4]. 作为过程挖掘的典型应用场景, 过程发现指在没有先验知识的前提下, 从事件日志出发挖掘、重构业务过程的 BPMN[5]模型、WF-net[6]等过程模型.

目前已有很多过程发现算法被提出[6~17], 任务间依赖关系的挖掘是多数过程发现算法的关键. 但是, 现有过程发现算法通常假设任务关系在同一业务过程中固定不变. 这种假定在某些场合下是错误的, 以网上

收稿日期: 2013-11-01; 修回日期: 2014-07-04; 责任编辑: 梅志强
基金项目: 国家自然科学基金(No.61170079, No.61202152, No.61472229); 青岛市科技计划基础研究项目(No.13-1-4-153-jch); 山东优秀中青年科学家科研奖励基金(No.BS2014DX013); 同济大学嵌入式系统与服务计算教育部重点实验室开放课题基金(No.ESSCKF201403)

购物过程为例,当付款方式为“货到付款”时,先发货后付款;当付款方式是“预付款”时,先付款后发货.“付款”和“发货”两个任务的依赖关系随案例属性“付款方式”的不同恰好相反.这种任务依赖关系随案例属性取值不同而变化的情况经常会导致已有过程发现算法挖掘结果的错误[18].针对这一问题,文献[18]通过引入案例簇的概念给出了挖掘任务间分条件依赖关系的具体算法.但是,过程发现的最终目标应重构出业务过程的具体模型.因此,本文在文献[18]已有成果基础上研究如何由挖掘到的任务关系导出具体模型.

在挖掘的过程模型表示方面,WF-net 具有直观的图形表示、严格的语义以及丰富的模型分析方法,许多过程发现算法以 WF-net 模型为挖掘目标[9~13].但是,WF-net 中的部分变迁用以表达任务之间的同步关系,这些变迁不对应实际业务过程中的任务,在事件日志中也就不存在相应的事件,这导致了不可见任务的挖掘难题[10,13].为解决上述问题,本文提出一种基于同步核的任务同步关系建模机制,首先将基于案例簇得到的任务关系和循环结构转换为同步核,然后给出由同步核转换为 WF-net 模型的方法.由同步核到 WF-net 模型转换过程中会自动得到各个不可见任务对应的变迁,由此解决了不可见任务的挖掘问题.而且,案例分簇的思想使本文方法能处理任务依赖关系随案例属性取值不同而变化的问题.

## 2 业务过程及事件日志引例

图 1 为美国高速公路冰雪事件应急响应过程[19]的 BPMN 模型.模型中涉及的任务、案例相关的属性及任务执行条件等信息见表 1.

**表 1 符号含义**

| 符号 | 含义 | 符号 | 含义 |
|---|---|---|---|
| $t_0$ | 事件信息登记 | $t_{10}$ | 总结存档 |
| $t_1$ | 路面结冰检测 | $v_1$ | 积雪状态(是否结冰) |
| $t_2$ | 积雪深度测量 | $v_2$ | 积雪深度 |
| $t_3$ | 路面温度测量 | $v_3$ | 路面温度 |
| $t_4$ | 发布预警信息 | $v_4$ | 是否开始下一轮除雪 |
| $t_5$ | 融雪剂融雪 | $E_1$ | $v_1=\text{false}\wedge v_2\leqslant 5\wedge(v_3>12\wedge v_3\leqslant 32)$ |
| $t_6$ | 机械除雪 | $E_2$ | $v_1=\text{false}\wedge v_2\leqslant 5\wedge(v_3\leqslant 12\vee v_3>32)$ |
| $t_7$ | 牵引增强 | $E_3$ | $v_1=\text{true}\vee v_2>5$ |
| $t_8$ | 非常规措施 | $E_4$ | $v_4=\text{true}$ |
| $t_9$ | 处置情况评估 | $E_5$ | $v_4=\text{false}$ |

上述流程的一个运行日志可由链接 http://pan.baidu.com/share/link?shareid=1425955987&uk=1175501481 中的文件 Appendix2.xes 得到.该日志中共包含 16 个案例,表 2 中以表格形式给出了当中两个案例的执行过程与案例属性信息,后文将结合该日志以及表 2 中的两个案例说明相关概念和方法.

**表 2 部分运行日志**

| 案例 ID | 事件 ID | 属性信息 | | | | |
|---|---|---|---|---|---|---|
| | | activity | $v_1$ | $v_2$ | $v_3$ | $v_4$ |
| $c_1$ | $e_1$ | $t_0$ | | | | |
| | $e_2$ | $t_1$ | false | | | |
| | $e_3$ | $t_2$ | | 4 | | |
| | $e_4$ | $t_3$ | | | 13 | |
| | $e_5$ | $t_4$ | | | | |
| | $e_6$ | $t_5$ | | | | |
| | $e_7$ | $t_9$ | | | | false |
| | $e_8$ | $t_{10}$ | | | | |
| …… | | | | | | |
| $c_{14}$ | $e_{109}$ | $t_0$ | | | | |
| | $e_{110}$ | $t_3$ | | | 26 | |
| | $e_{111}$ | $t_1$ | true | | | |
| | $e_{112}$ | $t_2$ | | 6 | | |
| | $e_{113}$ | $t_4$ | | | | |
| | $e_{114}$ | $t_8$ | | | | |
| | $e_{115}$ | $t_9$ | | | | true |
| | $e_{116}$ | $t_1$ | false | | | |
| | $e_{117}$ | $t_2$ | | 5 | | |
| | $e_{118}$ | $t_3$ | | | 25 | |
| | $e_{119}$ | $t_5$ | | | | |
| | $e_{120}$ | $t_4$ | | | | |
| | $e_{121}$ | $t_9$ | | | | false |
| | $e_{122}$ | $t_{10}$ | | | | |
| …… | | | | | | |

所谓事件日志指案例组成的多重集,上述日志实例记为 $L=\{c_1,c_2,\cdots,c_{16}\}$,其对应的事件集为 $\{e_1,e_2,\cdots,e_{159}\}$,日志中出现的任务集 $T$ 为 $\{t_0,t_1,t_2,t_3,t_4,t_5,t_6,t_7,t_8,t_9,t_{10}\}$,案例属型及事件属性组成的属性集 $V$ 为 $\{\text{activity},v_1,v_2,v_3,v_4\}$.对任意事件 $e$ 与属性 $v$,记 $\#_v(e)$为属性 $v$ 对应于事件 $e$ 的取值,则 $\#_{\text{activity}}(e_1)$为 $t_0$,$\#_{v1}(e_2)$为 false.如果案例中每个事件仅对应一个任务,则 $\#_v(e)$可写作 $\#_v(c,a)$,当中 $a$ 为事件 $e$ 对应的任务,$c$ 为案例 ID.

此外,每个案例对应多个事件,这些事件按发生先后构成的序列称为案例轨迹,用任务代替案例轨迹中的事件所得序列称为案例的任务轨迹,记作 $\#_{\text{activityTrace}}(c)$,如 $L$ 中 $\#_{\text{activityTrace}}(c_1)=\langle t_0,t_1,t_2,t_3,t_4,t_5,t_9,t_{10}\rangle$.

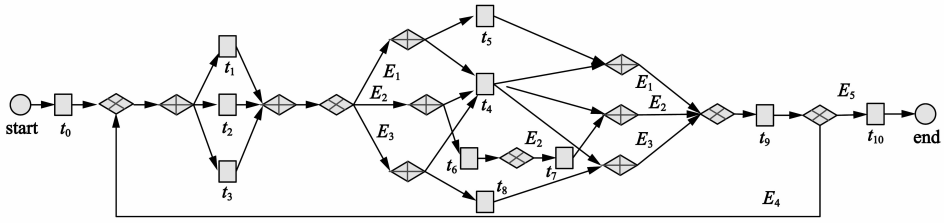



**图1　高速公路冰雪事件应急响应过程的BPMN模型**

## 3　案例簇及其对应同步核的挖掘

### 3.1　案例簇

**定义 1[18](基本案例、组合案例)**　设 $c$ 为一个案例,$a$ 为一任务,$a$ 在 $\#_{\text{activityTrace}}(c)$ 中的出现次数记作 $\#_{\text{occur}}(a,c)$,如果:$\forall a:\#_{\text{occur}}(a,c)\leqslant 1$,则 $c$ 称作一个基本案例.否则,称 $c$ 为组合案例.

日志实例 $L$ 中,$c_1$ 是基本案例,因 $\forall a\in T$ 均有 $\#_{\text{occur}}(a,c_1)\leqslant 1$,$c_{14}$ 为组合案例,因 $\#_{\text{occur}}(t_1,c_{14})=2$.

**定义 2[18](案例的特征向量)**　设 $c$ 为一个案例,$T$ 为任务集,$B(T):T\to N$ 是定义在 $T$ 上的袋集,$\boldsymbol{X}\in B$ 为一个袋,如果 $\forall a\in T:\boldsymbol{X}(a)=\#_{\text{occur}}(a,c)$ 成立,则称 $\boldsymbol{X}$ 为 $c$ 的特征向量,记作 $\#_{\text{feature}}(c)$.

日志 $L$ 中,案例 $c_1,c_4,c_{10},c_{12}$ 的特征向量为 $\boldsymbol{X}_1=[t_0^1,t_1^1,t_2^1,t_3^1,t_4^1,t_5^1,t_9^1,t_{10}^1]$,$c_2,c_5,c_6,c_9$ 的为 $\boldsymbol{X}_2=[t_0^1,t_1^1,t_2^1,t_3^1,t_4^1,t_6^1,t_7^1,t_9^1,t_{10}^1]$,$c_3,c_7,c_8,c_{11},c_{13}$ 的为 $\boldsymbol{X}_3=[t_0^1,t_1^1,t_2^1,t_3^1,t_4^1,t_8^1,t_9^1,t_{10}^1]$,$c_{14}$ 的特征向量为 $\boldsymbol{X}_4=[t_0^1,t_1^2,t_2^2,t_3^2,t_4^2,t_5^1,t_8^1,t_9^2,t_{10}^1]$.

**定义 3[18](案例簇)**　设 $L$ 为一个事件日志,$\boldsymbol{X}$ 为 $L$ 中某个案例的特征向量,所谓特征向量 $\boldsymbol{X}$ 决定的案例簇指日志 $L$ 中所有符合 $\#_{\text{feature}}(c)=\boldsymbol{X}$ 的基本案例 $c$ 组成的集合,记作 $\pounds(\boldsymbol{X})$.

日志实例 $L$ 中,$\pounds(\boldsymbol{X}_1)=\{c_1,c_4,c_{10},c_{12}\}$,$\pounds(\boldsymbol{X}_2)=\{c_2,c_5,c_6,c_9\}$,$\pounds(\boldsymbol{X}_3)=\{c_3,c_7,c_8,c_{11},c_{13}\}$.

仅当案例满足特定条件时,案例执行的任务集以及任务执行次数会一致,称该条件为案例簇的特征条件,其严格定义如下:

**定义 4[18](案例簇的特征条件)**　设 $\pounds(\boldsymbol{X})$ 为一个案例簇,$E$ 为定义在属性集 $V$ 上的一个布尔表达式,如果案例 $c\in\pounds(\boldsymbol{X})$ 当且仅当 $c$ 满足 $E$,则称 $E$ 为 $\pounds(\boldsymbol{X})$ 的特征条件,记作 $\#_{\text{cond}}(\pounds(\boldsymbol{X}))$.

在实例 $L$ 中,$\#_{\text{cond}}(\pounds(\boldsymbol{X}_1))$ 为 $\#v_1(c,t_1)=\text{false}\wedge\#v_2(c,t_2)\leqslant 5\wedge\#v_3(c,t_3)>12\wedge\#v_3(c,t_3)\leqslant 32$,因有且仅有案例 $c_1,c_4,c_{10},c_{12}$ 满足这一条件.此外,可将属性 $\#v_1(c,t_1)$ 简写为 $v_1$,$\#v_2(c,t_2)$ 简写为 $v_2$,$\#v_3(c,t_3)$ 简写为 $v_3$,$\#v_4(c,t_9)$ 简写为 $v_4$,因分别只有一个任务对这些属性进行赋值.从而,$\#_{\text{cond}}(\pounds(\boldsymbol{X}_1))$ 可简化为 $v_1=\text{false}\wedge v_2\leqslant 5\wedge(v_3>12\wedge v_3\leqslant 32)$,即为表 1 中的条件 $E_1$.类似地,$\#_{\text{cond}}(\pounds(\boldsymbol{X}_2))$ 化为 $E_2:(v_1=\text{false}\wedge v_2\leqslant 5)\wedge(v_3\leqslant 12\vee v_1>32)$,$\#_{\text{cond}}(\pounds(\boldsymbol{X}_3))$ 化为 $E_3:v_1=\text{true}\vee v_2>5$.

### 3.2　案例簇内部的任务关系

案例簇对应业务过程中不包含循环和冲突的子模型,当中的任务仅存在因果和并发两类关系,相关定义如下.

**定义 5[18](基于案例簇的任务关系)**　设 $\pounds(\boldsymbol{X})$ 是特征向量 $\boldsymbol{X}$ 决定的案例簇,$a$ 与 $b$ 是两个不同的任务,

(1)若 $\exists c\in\pounds(\boldsymbol{X})\,\exists\sigma=\langle t_1,t_2,t_3,\cdots,t_n\rangle\,\exists i,j\in\{1,\cdots,n-1\}$ 使 $i<j\wedge t_i=a\wedge t_j=b\wedge\sigma=\#_{\text{activityTrace}}(c)$,则称 $a$ 与 $b$ 在案例簇 $\pounds(\boldsymbol{X})$ 中存在可达关系,记为 $a\gg_{\boldsymbol{X}}b$;

(2)若 $a\gg_{\boldsymbol{X}}b$ 成立而 $b\gg_{\boldsymbol{X}}a$ 不成立,则称 $a$ 与 $b$ 在案例簇 $\pounds(\boldsymbol{X})$ 中存在因果关系,记为 $a\to_{\boldsymbol{X}}b$;

(3)若 $a\gg_{\boldsymbol{X}}b$ 与 $b\gg_{\boldsymbol{X}}a$ 同时成立,则称 $a$ 与 $b$ 在案例簇 $\pounds(\boldsymbol{X})$ 中存在并发关系,记为 $a\parallel_{\boldsymbol{X}}b$.

以 $\pounds(\boldsymbol{X}_2)=\{c_2,c_5,c_6,c_9\}$ 为例.由 $c_2$ 的任务轨迹 $\langle t_0,t_3,t_2,t_1,t_4,t_6,t_7,t_9,t_{10}\rangle$ 易见 $t_2\gg_{\boldsymbol{X}_2}t_1$,由 $c_5$ 的任务轨迹 $\langle t_0,t_1,t_2,t_3,t_6,t_7,t_4,t_9,t_{10}\rangle$ 易见 $t_1\gg_{\boldsymbol{X}_2}t_2$.根据定义 5,$t_1$ 与 $t_2$ 具有并发关系 $t_1\parallel_{\boldsymbol{X}_2}t_2$.$t_0$ 与 $t_1$ 具有因果关系 $t_0\to_{\boldsymbol{X}_2}t_1$,因为在 $\pounds(\boldsymbol{X}_2)$ 的所有任务轨迹中,$t_0\gg_{\boldsymbol{X}_2}t_1$ 成立而 $t_1\gg_{\boldsymbol{X}_2}t_0$ 不成立.

文献[18]给出了案例簇、案例簇特征条件及案例簇内任务关系的挖掘算法.以日志实例 $L$ 为输入,所得各案例簇内的任务关系见表 3.

### 3.3　同步核的挖掘

本节首先引入同步核的概念,之后给出由案例簇内部任务关系挖掘同步核的方法.

**定义 6(同步核)**　记 $A_1$、$A_2$ 为两个任务集合,$E$ 是定义在案例相关属性集上的布尔表达式,$\text{syn}=\langle A_1,A_2,E\rangle$ 称为一个同步核,如果它满足如下条件:

**条件 1**　当 $E$ 成立时,$A_1$ 中任意两个不同的任务 $t$ 和 $t'$ 满足 $t\parallel_{\boldsymbol{X}}t'$;

**条件 2**　当 $E$ 成立时,$A_2$ 中任意两个不同的任务 $t$ 和 $t'$ 满足 $t\parallel_{\boldsymbol{X}}t'$;

**条件 3**　当 $E$ 成立时,$A_2$ 中的任务使能当且仅当 $A_1$ 中的任务全部执行完毕;

**条件 4** $A_1$与 $A_2$是满足上述条件的极大任务子集.

**表 3 案例簇 $\pounds(X_1)$、$\pounds(X_2)$与 $\pounds(X_3)$内部的任务关系**

| 案例簇 | 任务间的关系 |
|---|---|
| $\pounds(X_1)=\{c_1, c_4, c_{10}, c_{12}\}$ | $\{t_0\rightarrow t_1, t_0\rightarrow t_2, t_0\rightarrow t_3, t_0\rightarrow t_4, t_0\rightarrow t_5, t_0\rightarrow t_9, t_0\rightarrow t_{10}, t_1 \parallel t_2, t_1 \parallel t_3, t_1\rightarrow t_4, t_1\rightarrow t_5, t_1\rightarrow t_9, t_1\rightarrow t_{10}, t_2 \parallel t_1, t_2 \parallel t_3, t_2\rightarrow t_4, t_2\rightarrow t_5, t_2\rightarrow t_9, t_2\rightarrow t_{10}, t_3 \parallel t_1, t_3 \parallel t_2, t_3\rightarrow t_4, t_3\rightarrow t_5, t_3\rightarrow t_9, t_3\rightarrow t_{10}, t_4 \parallel t_5, t_4\rightarrow t_9, t_4\rightarrow t_{10}, t_5 \parallel t_4, t_5\rightarrow t_9, t_5\rightarrow t_{10}\}$ |
| $\pounds(X_2)=\{c_2, c_5, c_6, c_9\}$ | $\{t_0\rightarrow t_1, t_0\rightarrow t_2, t_0\rightarrow t_3, t_0\rightarrow t_4, t_0\rightarrow t_6, t_0\rightarrow t_7, t_0\rightarrow t_9, t_0\rightarrow t_{10}, t_1 \parallel t_2, t_1 \parallel t_3, t_1\rightarrow t_4, t_1\rightarrow t_6, t_1\rightarrow t_7, t_1\rightarrow t_9, t_1\rightarrow t_{10}, t_2 \parallel t_1, t_2 \parallel t_3, t_2\rightarrow t_4, t_2\rightarrow t_6, t_2\rightarrow t_7, t_2\rightarrow t_9, t_2\rightarrow t_{10}, t_3 \parallel t_1, t_3 \parallel t_2, t_3\rightarrow t_4, t_3\rightarrow t_6, t_3\rightarrow t_7, t_3\rightarrow t_9, t_3\rightarrow t_{10}, t_4 \parallel t_6, t_4 \parallel t_7, t_4\rightarrow t_9, t_4\rightarrow t_{10}, t_6 \parallel t_4, t_6\rightarrow t_7, t_6\rightarrow t_9, t_6\rightarrow t_{10}, t_7 \parallel t_4, t_7\rightarrow t_9, t_7\rightarrow t_{10}, t_9\rightarrow t_{10}\}$ |
| $\pounds(X_3)=\{c_3, c_7, c_8, c_{11}, c_{13}\}$ | $\{t_0\rightarrow t_1, t_0\rightarrow t_2, t_0\rightarrow t_3, t_0\rightarrow t_4, t_0\rightarrow t_8, t_0\rightarrow t_9, t_0\rightarrow t_{10}, t_1 \parallel t_2, t_1 \parallel t_3, t_1\rightarrow t_4, t_1\rightarrow t_8, t_1\rightarrow t_9, t_1\rightarrow t_{10}, t_2 \parallel t_1, t_2 \parallel t_3, t_2\rightarrow t_4, t_2\rightarrow t_8, t_2\rightarrow t_9, t_2\rightarrow t_{10}, t_3 \parallel t_1, t_3 \parallel t_2, t_3\rightarrow t_4, t_3\rightarrow t_8, t_3\rightarrow t_9, t_3\rightarrow t_{10}, t_4 \parallel t_8, t_4\rightarrow t_9, t_4\rightarrow t_{10}, t_8 \parallel t_4, t_8\rightarrow t_9, t_8\rightarrow t_{10}\}$ |

其中,称 $A_1$为同步核 syn 的源任务集,$A_2$为 syn 的目标任务集,$E$ 为 syn 的同步条件,源任务集为 $\varnothing$ 的同步核为起始同步核,目标任务集为 $\varnothing$ 的同步核为结束同步核,其他同步核为标准同步核.

根据定义 6,$\text{syn}_1=\langle\{t_1, t_2, t_3\}, \{t_4, t_5\}, E_1\rangle$是图 1 业务过程的两个标准同步核,起始同步核为$\langle\varnothing, \{t_0\}, \text{true}\rangle$,结束同步核为$\langle\{t_{10}\}, \varnothing, \text{true}\rangle$.

在 3.2 节所得案例簇内部任务关系的基础上,可由定理 1 得到案例簇对应的同步核.

**定理 1** 设 $\pounds(X)$是一个案例簇,$E$ 为其特征条件,$T$ 为其任务集,$\rightarrow_X$为其内部任务因果关系的集合,$\parallel_X$为任务并发关系的集合,$T_1$与 $T_2$为 $T$ 的两个子集,如果下述条件满足,则$\langle T_1, T_2, E\rangle$是该过程的一个同步核:

**条件 1** 对于 $T_1$(或者 $T_2$)中任意两个不同的任务 $t$ 和 $t'$,$t \parallel_X t'$始终成立;

**条件 2** $\forall t_1\in T_1 \forall t_2\in T_2: t_1\rightarrow_X t_2$恒成立;

**条件 3** 对 $\forall T_1'\subseteq T$,若 $T_1'\neq T_1 \wedge \forall t_1\in T_1' \ \forall t_2\in T_2: (t_1, t_2)\in\rightarrow_{\pounds}$成立,则存在 $t_1'\in T_1'$与 $t_1\in T_1$使得 $(t_1', t_1)\in\rightarrow_{\pounds}$成立;

**条件 4** 对 $\forall T_2'\subseteq T$,若 $T_2'\neq T_2 \wedge \forall t_1\in T_1 \forall t_2\in T_2': (t_1, t_2)\in\rightarrow_X$成立,则存在任务 $t_2'\in T_2'$与 $t_2\in T_2$使得$(t_2, t_2')\in\rightarrow_X$成立;

**条件 5** $T_1$、$T_2$是满足上述条件的 $T$ 的极大子集.

证明:条件 1 说明 $T_1$和 $T_2$内部的任务在条件 $E$ 成立时相互间具有并发关系,满足定义 6 中同步核的前两个要求;条件 2 ~ 条件 4 说明条件 $E$ 成立时 $T_1$中的任务完成后 $T_2$中的任务立即使能,满足定义 6 中同步核的第三个要求,条件 5 满足定义 6 中同步核的最后一个要求,由此可知,$\langle T_1, T_2, E\rangle$是一个同步核.

由定理 1 可得由案例簇内部的任务关系挖掘同步核的步骤如下:(1)将不依赖其他任务便可直接执行的任务放入集合 $T_0$,构造起始同步核$\langle\varnothing, T_0, E\rangle$;(2)对于 $T_0$以外的任务 $t$,先计算其在各个任务轨迹中的直接前驱构成的任务集 $T_{\text{pre}}(t)$,如果 $T_{\text{pre}}(t)$为 $T_0$的子集则分两种情况处理:当已经得到的同步核中存在源任务集为 $T_{\text{pre}}(t)$的同步核 $\text{syn}=\langle T_1, T_2, E\rangle$时,将该同步核的目标任务集 $T_2$更新为 $T_2\cup\{t\}$;否则,添加新的同步核$\langle T_{\text{pre}}(t), \{t\}, E\rangle$;(3)无论上述哪一种情况,处理完毕后都将 $t$ 并入集合 $T_0$;(4)对 $T\text{-}T_0$中剩余的任务重复执行上述步骤(2)和步骤(3),直至 $T\text{-}T_0$为空;(5)统计未在同步核源任务集中出现的任务,将它们作为源任务集构造结束同步器.详细步骤见算法 1,其时间复杂度为 $O(n^4 * |\text{SynSet}|)$,其中$|\text{SynSet}|$是同步核个数.

**算法 1 由案例簇内部的任务关系挖掘同步核**

**输入**:案例簇 $\pounds(X)$及其内部任务因果关系的集合$\rightarrow_X$
**输出**:案例簇 $\pounds(X)$对应的同步核集合 Syn

```
1:  记案例簇£(X)的特征条件为 E,£(X)中任务的集合为 T
2:  T0: = {t | t∈T ∧ ∀t'∈T:(t',t)∉→X};
3:  Syn: = {⟨∅, T0, E⟩};
4:  T: = T-T0;
5:  WHILE (T≠∅){
6:    FOR EACH(t∈T){
7:      Tpre(t): = {t' | t'∈T ∧ (t',t)∈→X};
8:      IF(Tpre(t)⊆T0){
9:        FOR EACH t'∈Tpre(t){
10:         FOR EACH(syn = ⟨T1, T2, E⟩∈Syn){
11:         IF(t'∈T2) Tpre(t): = Tpre(t)-T1;
12:         }//for each syn in Syn
13:       }//for eacht'∈Tpre(t)
14:       isPreSetRepeated: = false;
15:       FOR EACH(syn = ⟨T1, T2, E⟩∈Syn){
16:       IF(T1 = Tpre(t)){
17:         isPreSetRepeated: = true;
18:         syn: = ⟨T1, T2∪{t}, E⟩; BREAK;
19:       }
20:       }//for each syn in Syn
21:     IF(isPreSetRepeated = false)
              Syn: = Syn∪{⟨Tpre(t), {t}, E⟩};
22:       T0: = T0∪{t};
23:       T: = T-{t};
24:     }//if(Tpre(t)⊆T0)
25:   }//for each (t in T)
```

26:}//while ($T \neq \varnothing$)
27:$T_{end}$:={$t \mid t \in T_0 \wedge \forall t' \in T_0:(t,t') \notin \rightarrow_X$};
28:Syn:=Syn$\cup$ {$\langle T_{end}, \varnothing, E\rangle$};
29:RETURN Syn

以表 3 中的案例簇、任务关系为输入,执行算法 1 可得表 4 中各案例簇对应的同步核.

**表 4　各案例簇对应的同步核的集合**

| 案例簇 | 案例簇对应的同步核 |
|---|---|
| $\pounds(X_1)$ | {$\langle\varnothing,\{t_0\},E_1\rangle$,$\langle\{t_0\},\{t_1,t_2,t_3\},E_1\rangle$,$\langle\{t_1,t_2,t_3\},\{t_4,t_5\},E_1\rangle$,$\langle\{t_4,t_5\},\{t_9\},E_1\rangle$,$\langle\{t_9\},\{t_{10}\},E_1\rangle$,$\langle\{t_{10}\},\varnothing,E_1\rangle$} |
| $\pounds(X_2)$ | {$\langle\varnothing,\{t_0\},E_2\rangle$,$\langle\{t_0\},\{t_1,t_2,t_3\},E_2\rangle$,$\langle\{t_1,t_2,t_3\},\{t_4,t_6\},E_2\rangle$,$\langle\{t_6\},\{t_7\},E_2\rangle$,$\langle\{t_4,t_7\},\{t_9\},E_2\rangle$,$\langle\{t_9\},\{t_{10}\},E_2\rangle$,$\langle\{t_{10}\},\varnothing,E_2\rangle$} |
| $\pounds(X_3)$ | {$\langle\varnothing,\{t_0\},E_3\rangle$,$\langle\{t_0\},\{t_1,t_2,t_3\},E_3\rangle$,$\langle\{t_1,t_2,t_3\},\{t_4,t_8\},E_3\rangle$,$\langle\{t_4,t_8\},\{t_9\},E_3\rangle$,$\langle\{t_9\},\{t_{10}\},E_3\rangle$,$\langle\{t_{10}\},\varnothing,E_3\rangle$} |

## 4　循环结构挖掘与同步核集成

前述同步核未对循环结构进行刻画,为表达循环结构,本节引入返回核的概念,并给出由组合案例挖掘返回核的方法,最后将同步核和返回核进行集成化简.

### 4.1　返回核

循环结构通常由源任务集、目标任务集和循环条件三要素构成.所谓源任务指循环执行前(即重复任务出现前)最后执行的一或多个相互并发的任务;所谓目标任务指循环执行前已经执行过的一或多个相互并发的任务,目标任务在循环执行之初会被再次使能;当执行完源任务集中的任务后,循环条件用来决定是否回退过程状态、再次使能目标任务集中曾经执行过的任务.另外,循环执行之前,案例一定是按照某案例簇对应的子过程模型被加工处理.基于上述分析,循环结构可以通过如下定义的一类特殊同步核来建模.

**定义 7(返回核)**　设 $P$ 是一个业务过程,return = $\langle A_1, A_2, \text{Cond}\rangle$为该过程的一个同步核,$\pounds(\boldsymbol{X})$是一个案例簇,$\rightarrow_X$ 是案例簇 $\pounds(\boldsymbol{X})$对应的任务间因果关系的集合,$A$ 是 $\boldsymbol{X}$ 对应的任务集,如果以下条件成立,则称 return 为与案例簇 $\pounds(\boldsymbol{X})$相容的一个返回核:

**条件 1**　$A_1 \subseteq A \ \wedge \ A_2 \subseteq A$;

**条件 2**　$\forall a \in A_1 \forall b \in A_2:(b,a) \in \rightarrow_X$ 恒成立;

**条件 3**　Cond$\wedge$ $\#_{\text{cond}}(\pounds(\boldsymbol{X})) \neq$ false.

根据定义 7,返回核首先是一个同步核,能表达过程中任务集合 $A_1$与 $A_2$在条件 Cond 成立时的直接依赖关系;其次,返回核在条件 Cond 成立时将过程从一种状态后退到之前的某一个状态;最后,Cond 对应循环结构的循环条件,条件 3 保证属于案例簇 $\pounds(\boldsymbol{X})$的案例存在使该条件成立并执行循环的可能.

图 1 中,一旦{$t_9$}执行完,当 $v_4$ = true 时集合{$t_1, t_2, t_3$}中的任务使能,故 $r = \langle\{t_9\},\{t_1,t_2,t_3\},v_4 = \text{true}\rangle$是一个同步核;其次,该同步核是与案例簇 $\pounds(\boldsymbol{X}_2) = \{c_2, c_5, c_6, c_9\}$相容的一个返回核,因为:(1) $\pounds(\boldsymbol{X}_2)$中关系 $t_1 \rightarrow t_9 \wedge t_2 \rightarrow t_9 \wedge t_3 \rightarrow t_9$成立,这意味着{$t_9$}执行完毕后转而执行{$t_1, t_2, t_3$}是一种状态后退;(2) $\#_{\text{Cond}}(\pounds(\boldsymbol{X}_2)) \wedge v_4 = \text{true}$ 的值"$v_1 = \text{false} \wedge v_2 \leqslant 5 \wedge (v_3 \leqslant 12 \vee v_3 \rangle 32) \wedge v_4 = \text{true}$"不为 false,这说明 $\pounds(\boldsymbol{X}_2)$中的案例在执行过程中属性取值可能满足循环执行的条件.类似可证 $r = \langle\{t_9\},\{t_1,t_2,t_3\},v_4 = \text{true}\rangle$与另外两个案例簇 $\pounds(\boldsymbol{X}_1)$和 $\pounds(\boldsymbol{X}_3)$也相容.

显然,循环结构的执行会导致组合案例的产生.设 return = $\langle A_1, A_2, \text{Cond}\rangle$是与 $\pounds(\boldsymbol{X})$相容的一个返回核,并设 $c$ 是对应于 return 和 $\pounds(\boldsymbol{X})$的一个组合案例,则在 $\#_{\text{activityTrace}}(c)$中:(1) $A_2$中的任务必然是首批重复出现的若干连续的任务;(2) $A_1$中的任务必然是 $A_2$中任务重复出现前最后出现的一批连续的任务;(3)在任务重复前出现的任务都来自于 $\boldsymbol{X}$;(4)在任务重复前,各案例相关属性的取值与案例簇 $\pounds(\boldsymbol{X})$的特征条件不冲突.

基于上述特点,返回核可通过分析组合案例的任务轨迹以及案例相关属性的取值信息挖掘得到.设待挖掘返回核为 return = $\langle A_1, A_2, \text{Cond}\rangle$,下面结合组合案例 $c_{14}$说明具体挖掘步骤:

(1)根据组合案例任务轨迹确定可能与 return 相容的案例簇.

$c_{14}$的任务轨迹为 $\#_{\text{activityTrace}}(c_{14}) = \langle t_0, t_3, t_1, t_2, t_4, t_8, t_9, t_1, t_2, t_3, t_5, t_4, t_9, t_{10}\rangle$.该轨迹中,出现重复任务前的任务集合为{$t_0, t_3, t_1, t_2, t_4, t_8, t_9$}.显然,只有 $\pounds(\boldsymbol{X}_3)$涵盖任务集{$t_0, t_3, t_1, t_2, t_4, t_8, t_9$},因此,$\pounds(\boldsymbol{X}_3)$是唯一可能与 return 相容的案例簇.

(2)分析重复任务出现前组合案例各属性值,据此筛选上步所得与 return 相容的案例簇.

$c_{14}$中任务重复前各属性值为 $v_1$ = true、$v_2 = 6$、$v_4$ = true.易见$(v_1 = \text{true} \wedge v_2 = 6 \wedge v_4 = \text{true}) \wedge \#_{\text{cond}}(\pounds(\boldsymbol{X}_3)) \neq$ false.从而,$\pounds(\boldsymbol{X}_3)$与 return 相容.反之,则 $\pounds(\boldsymbol{X}_3)$不可能与 return 相容.

(3)根据组合案例任务轨迹、与 return 相容的案例簇确定 return 的目标任务集 $A_2$.

$c_{14}$的任务轨迹中首次重复出现的连续的任务子序列为 $t_1t_2t_3$,故返回核的目标任务集 $A_2$一定是{$t_1$}、{$t_1, t_2$}或者{$t_1, t_2, t_3$}之一.再据定义 7,$A_2$中任意两个任务在案例簇 $\pounds(\boldsymbol{X}_3)$中应具有并发关系.而根据表 3 中

$\pounds(\boldsymbol{X}_3)$对应的任务关系，$t_1$、$t_2$与 $t_3$之间均是并发关系.故，$\{t_1\}$、$\{t_1,t_2\}$或者$\{t_1,t_2,t_3\}$均可能是 $A_2$的值.此时，令 $A_2$是当中任务最多的一个集合，即令 $A_2=\{t_1,t_2,t_3\}$.此外，若有多个与 return 相容的案例簇，则应针对每个相容案例簇分别确定目标任务集.

(4)根据组合案例任务轨迹以及与 return 相容的案例簇确定 return 的源任务集 $A_1$.

$c_{14}$的任务轨迹中任务重复前的子序列是 $t_0t_3t_1t_2t_4t_8t_9$，故返回核源任务集 $A_1$一定是$\{t_9\}$、$\{t_8,t_9\}$，…，或者$\{t_0,t_3,t_1,t_2,t_4,t_8,t_9\}$之一.再根据定义 7，$A_1$中的任意两个任务在案例簇 $\pounds(\boldsymbol{X}_3)$中应具有并发关系.根据表 3，$t_9$之前的所有任务均与 $t_9$具有因果关系而非并发关系，故 $A_1$只能是$\{t_9\}$.若此时所得 $A_1$仍有多种可能，则令 $A_1$是当中任务最多的一个集合.此外，若第 2 步所得与 return 相容的案例簇不止一个，则应针对每个相容的案例簇分别确定源任务集.

(5)在组合案例属性值、以及与 return 相容的案例簇所包含基本案例属性值的基础上，构造用于学习循环条件的数据集.

$c_{14}$在执行完毕 $A_1$中的任务后发生了循环，而案例簇 $\pounds(\boldsymbol{X}_3)$中的基本案例在完成 $A_1$中的任务后不执行循环.由此以来，可根据组合案例中循环执行前的属性值构造一条数据记录，并为该记录添加标记循环发生的属性值 $A_1@A_2$(此处为“$\{t_9\}@\{t_1,t_2,t_3\}$”).此外，根据案例簇 $\pounds(\boldsymbol{X}_3)$中各基本案例在完成 $A_2$中任务后各属性值也构造相应记录，并为这些记录添加标记循环未发生的属性值“no return”.基于上述记录构造数据集 DataSet($A_1$, $A_2$, $\pounds(\boldsymbol{X}_3)$)，构造结果见表 5.如果与 return 相容的案例簇有多个，则要针对所有相容案例簇构造相应数据集.

**表 5 对应于 $c_{14}$的数据集** DataSet($\{t_9\}@\{t_1,t_2,t_3\}$, $\pounds(\boldsymbol{X}_3)$)

| $v_1$ | $v_2$ | $v_3$ | $v_4$ | … | 循环发生标记 |
|---|---|---|---|---|---|
| true | 6 | 26 | true | … | $\{t_9\}@\{t_1,t_2,t_3\}$ |
| true | 6 | 10 | false | … | no return |
| false | 7 | 2 | false | … | no return |
| true | 4 | 19 | false | … | no return |
| true | 3 | 19 | false | … | no return |
| false | 6 | 3 | false | … | no return |

(6)按前述步骤处理剩余的组合案例，对于每个新得到的数据集 DataSet($A'_1$, $A'_2$, $\pounds(\boldsymbol{X}')$)，均检测是否已有数据集 DataSet($A_1$, $A_2$, $\pounds(\boldsymbol{X})$)满足 $A_1\bigcap A'_1\neq\Phi\wedge A_2=A'_2\wedge\boldsymbol{X}=\boldsymbol{X}'$.若存在则将这两个数据集合并，并将合并后的数据集对应的源任务集合设置为 $A_1'\bigcap A_1$.最后，根据合并后的数据集挖掘返回核对应的循环条件.

以组合案例 $c_{15}$为例，经前 5 步可得表 6 所示数据集，因该数据集所对应案例簇的特征向量不同于表 5 数据集，故两者无需合并.

**表 6 对应于 $c_{15}$的数据集** DataSet($\{t_9\}@\{t_1,t_2,t_3\}$, $\pounds(\boldsymbol{X}_2)$)

| $v_1$ | $v_2$ | $v_3$ | $v_4$ | … | 循环发生标记 |
|---|---|---|---|---|---|
| false | 1 | 9 | true | … | $\{t_9\}@\{t_1,t_2,t_3\}$ |
| false | 5 | 12 | false | … | no return |
| false | 4 | 7 | false | … | no return |
| false | 2 | 33 | false | … | no return |
| false | 2 | 34 | false | … | no return |

最后，对每一个最终得到的数据集 DataSet($A_1@A_2$, $\pounds(\boldsymbol{X})$)，将“循环发生标记”作为分类变量、其他属性作预测变量，执行 C4.5 算法[20]生成决策树.记由决策树所得对应于类别 $A_1@A_2$的分类条件为 Cond，则$\langle A_1,A_2,\text{Cond}\wedge\#_{\text{cond}}(\pounds(\boldsymbol{X}))\rangle$为原过程的一个返回核.以表 5 数据集 DataSet($\{t_9\}@\{t_1,t_2,t_3\}$, $\pounds(\boldsymbol{X}_3)$)为例，执行 C4.5 算法后，易见条件“$v_4$ = true”成立时，案例会执行从$\{t_9\}$到$\{t_1,t_2,t_3\}$的循环；条件“$v_4$ = false”成立时循环不会发生.由此可得返回核 $\text{syn}_{-1}=\langle\{t_9\},\{t_1,t_2,t_3\},v_4=\text{true}\wedge E_3\rangle$.

## 4.2 同步核的更新

前文所得同步核$\langle\{t_9\},\{t_{10}\},E_3\rangle$意味着$\{t_9\}$执行完毕后，只要条件 $E_3$成立，则任务$\{t_{10}\}$使能.然而，根据图 1 中的过程定义，$\{t_9\}$执行完毕后，$\{t_{10}\}$的使能还有一个前提条件：未发生$\{t_9\}$到$\{t_1,t_2,t_3\}$的循环.根据上节所得返回核$\langle\{t_9\},\{t_1,t_2,t_3\},v_4=\text{true}\wedge E_3\rangle$可知，$\{t_9\}$执行完毕后，若条件 $v_4=\text{true}\wedge E_3$满足则过程会发生循环.综上，同步核($\langle\{t_9\},\{t_{10}\},E_3\rangle$的同步条件应该被更新为 $E_3\wedge!(v_4=\text{true}\wedge E_3)$.至此，我们得到了一个更新后的同步核($\langle\{t_9\},\{t_{10}\},E_3\wedge!(v_4=\text{true}\wedge E_3)\rangle$)，显然，该同步核符合考虑循环结构后的冰雪事件应急响应过程的行为定义.一般而言，考虑循环结构后，可根据定理 2 更新各个案例簇对应的同步核.

**定理 2** 设 retrun = $\langle A_1,A_2,\text{Cond}\rangle$为过程 $P$ 的一个返回核，$\pounds(\boldsymbol{X})$是与 return 相容一个案例簇，Syn 为该案例簇对应的同步核的集合，如果 $\text{syn\_original}=\langle A_1',A_2',\#_{\text{cond}}(\pounds(\boldsymbol{X}))\rangle\in\text{Syn}$ 满足 $A_1'=A_1$，则 syn_original 应该被更新为 $\text{syn\_updated}=\langle A_1',A_2',\#_{\text{cond}}(\pounds(\boldsymbol{X}))\wedge!\ \text{Cond}\rangle$.

证明：案例簇对应的同步核 $\text{syn_original}=\langle A_1',A_2',\#_{\text{cond}}(\pounds(\boldsymbol{X}))\rangle$意味着当集合 $A_1'$中的任务执行完

成后继续向前执行集合 $A_2'$ 中的任务,该同步核是在不考虑循环结构的前提下得到的.如果得到一个返回核 return = $\langle A_1, A_2, \text{Cond}\rangle$,则表示集合 $A_1$(即 $A_1'$)中的任务执行完成后,条件 Cond 满足则业务过程应该发生回退而重复执行 $A_2$中的任务.因此,考虑循环结构后,$A_1'$到 $A_2'$的跳转仅在满足 $\#_{\text{cond}}(\pounds(\boldsymbol{X}) \wedge !\ \text{Cond}$ 时才执行,故 syn_original = $\langle A_1', A_2', \#_{\text{cond}}(\pounds(\boldsymbol{X}))\rangle$应更新为 syn_updated = $\langle A_1', A_2', \#_{\text{cond}}(\pounds(\boldsymbol{X})) \wedge !\ \text{Cond}\rangle$.

根据定理 2 处理组合案例 $c_{14}$、$c_{15}$、$c_{16}$,可得返回核与更新后的同步核如表 7 所示.

**表 7　返回核以及更新后的同步核集合**

| 案例簇 | 相容返回核 | 更新后的同步核信息 |
|---|---|---|
| $\pounds(\boldsymbol{X}_1)$ | $\langle\{t_9\}, \{t_1, t_2, t_3\}, v_4 = \text{true} \wedge E_1\rangle$ | $\{\langle\varnothing, \{t_0\}, E_1\rangle, \langle\{t_0\}, \{t_1, t_2, t_3\}, E_1\rangle, \langle\{t_1, t_2, t_3\}, \{t_4, t_5\}, E_1\rangle, \langle\{t_4, t_5\}, \{t_9\}, E_1\rangle, \langle\{t_9\}, \{t_{10}\}, E_1 \wedge v_4 = \text{false}\rangle, \langle\{t_{10}\}, \varnothing, E_1\rangle\}$ |
| $\pounds(\boldsymbol{X}_2)$ | $\langle\{t_9\}, \{t_1, t_2, t_3\}, v_4 = \text{true} \wedge E_2\rangle$ | $\{\langle\varnothing, \{t_0\}, E_2\rangle, \langle\{t_0\}, \{t_1, t_2, t_3\}, E_2\rangle, \langle\{t_1, t_2, t_3\}, \{t_4, t_6\}, E_2\rangle, \langle\{t_6\}, \{t_7\}, E_2\rangle, \langle\{t_4, t_7\}, \{t_9\}, E_2\rangle, \langle\{t_9\}, \{t_{10}\}, E_2 \wedge v_4 = \text{false}\rangle, \langle\{t_{10}\}, \varnothing, E_2\rangle\}$ |
| $\pounds(\boldsymbol{X}_3)$ | $\langle\{t_9\}, \{t_1, t_2, t_3\}, v_4 = \text{true} \wedge E_3\rangle$ | $\{\langle\varnothing, \{t_0\}, E_3\rangle, \langle\{t_0\}, \{t_1, t_2, t_3\}, E_3\rangle, \langle\{t_1, t_2, t_3\}, \{t_4, t_8\}, E_3\rangle, \langle\{t_4, t_8\}, \{t_9\}, E_3\rangle, \langle\{t_9\}, \{t_{10}\}, E_3 \wedge v_4 = \text{false}\rangle, \langle\{t_{10}\}, \varnothing, E_3\rangle\}$ |

### 4.3　同步核与返回核的集成

同步核与返回核描述的同是任务之间的依赖关系,可借助定理 3 对两者进行集成.

**定理 3**　设 $P$ 为一个业务过程,$\text{syn}_1 = \langle A_1, A_2, \text{Cond}_1\rangle$与 $\text{syn}_2 = \langle A_1', A_2', \text{Cond}_2\rangle$是该过程的两个同步核(或返回核).若 $A_1 = A_1' \wedge A_2 = A_2'$成立,则 $\text{syn} = \langle A_1, A_2, \text{Cond}_1 \vee \text{Cond}_2\rangle$是属于该过程的一个同步核.

证明:根据同步核与返回核的定义,$\text{syn}_1 = \langle A_1, A_2, \text{Cond}_1\rangle$与 $\text{syn}_2 = \langle A_1', A_2', \text{Cond}_2\rangle$意味着:(1)只要条件 $\text{Cond}_1$或者 $\text{Cond}_2$成立,$A_1$(亦即 $A_1'$)或 $A_2$(亦即 $A_2'$)内部的任务具有并发关系;(2)只要条件 $\text{Cond}_1$或者 $\text{Cond}_2$成立,$A_1$(亦即 $A_1'$)中的任务一旦全部执行完毕,则 $A_2$(亦即 $A_2'$)中的任务使能.由此可得,当条件 $\text{Cond}_1 \vee \text{Cond}_2$成立时,一方面,$A_1$或 $A_2$内部的任务具有并发关系;另一方面,一旦 $A_1$中的任务全部执行完毕则 $A_2$中的任务使能.由此可知 $\text{syn} = \langle A_1, A_2, \text{Cond}_1 \vee \text{Cond}_2\rangle$ 是 $P$ 的一个同步核(或返回核).

以表 7 中的同步核与返回核为例,由定理 3 可得集成后的同步核集合 IntegratedSYN = $\{\langle\varnothing, \{t_0\}, \text{true}\rangle, \langle\{t_0\}, \{t_1, t_2, t_3\}, \text{true}\rangle, \langle\{t_1, t_2, t_3\}, \{t_4, t_5\}, \#_{\text{cond}}(\pounds(\boldsymbol{X}_1))\rangle, \langle\{t_4, t_5\}, \{t_9\}, \#_{\text{cond}}(\pounds(\boldsymbol{X}_1))\rangle, \langle\{t_9\}, \{t_{10}\}, v_4 = \text{false}\rangle, \langle\{t_{10}\}, \varnothing, \text{true}\rangle, \langle\{t_1, t_2, t_3\}, \{t_4, t_6\}, \#_{\text{cond}}(\pounds(\boldsymbol{X}_2))\rangle, \langle\{t_6\}, \{t_7\}, \#_{\text{cond}}(\pounds(\boldsymbol{X}_2))\rangle, \langle\{t_4, t_7\}, \{t_9\}, \#_{\text{cond}}(\pounds(\boldsymbol{X}_2))\rangle, \langle\{t_1, t_2, t_3\}, \{t_4, t_8\}, \#_{\text{cond}}(\pounds(\boldsymbol{X}_3))\rangle, \langle\{t_4, t_8\}, \{t_9\}, \#_{\text{cond}}(\pounds(\boldsymbol{X}_3))\rangle, \langle\{t_9\}, \{t_1, t_2, t_3\}, v_4 = \text{true}\rangle\}$.对照图 1 中的业务过程定义可见,这些同步核完整、准确地描述了任务间依赖关系和循环结构.

综上所述,可根据算法 2 从事件日志挖掘同步核.其中,第 1 ~ 9 步识别基本案例与组合案例并将各基本案例归入相应案例簇,该步操作可通过多线程等并行技术将原始日志分为多个案例子序列进行并行处理;第 12 ~ 15 步计算案例簇对应的同步核,该步骤也可借助多线程等技术使每个线程独立处理若干案例簇.

记 $|L|$为日志中包含的案例数量,$n$ 为日志中出现的任务数,$K$ 为参与并行处理的处理机数目,$M$ 为案例簇数目,$|\pounds(\boldsymbol{X})|$为各个案例簇包含的基本案例的最大个数,|Syn|为各个案例簇对应的同步核的最大个数,|CC|为组合案例数量,length 为任务轨迹的最大长度,|Loop|为循环结构数量,|SynAndReturn|为同步核和返回核的总个数.在不考虑处理机间通信代价的前提下,整个算法的时间复杂度如下:

$$O(|L|^3 + (|L| * n^2 + M * |\pounds(\boldsymbol{X})| * n^2 + M * |\text{Syn}| * n^4 + |\text{CC}| * \text{length} * M * |\text{Syn}| * n + |\text{Loop}| * M * |\pounds(\boldsymbol{X})|^3 + |\text{Loop}| * M * |\text{Syn}|)/K)$$

**算法 2　由事件日志挖掘同步核的算法**

**输入**:事件日志 $L$

**输出**:业务过程中的同步核集合 IntegratedSYN

1:　FOR EACH $c \in L$ IN PARALLEL {
2:　　IF ( $c$ 是基本案例) {
3:　　$\boldsymbol{X}$: = $\#_{\text{feature}}(c)$
4:　　$\pounds(\boldsymbol{X})$: = $\pounds(\boldsymbol{X}) \cup \{c\}$
5:　　}//if
6:　　ELSE {　//$c$ 是一个组合案例
7:　　CC: = CC $\cup \{c\}$
8:　}//else
9:　}//for eachc$\in L$ in parallel
10:执行 C4.5 算法求各个案例簇的特征条件;
11:SYN: = $\varnothing$;
12:FOR EACH(案例簇 $\pounds(\boldsymbol{X})$)IN PARALLEL{
13:　　根据文献[18]中的算法挖掘 $\pounds(\boldsymbol{X})$内部任务关系;
14:　　执行算法 1 挖掘案例簇 $\pounds(\boldsymbol{X})$对应的同步核,记第 $i$ 个案例簇对应所得同步核集合为 $\text{Syn}_i$;
15:} //for each
16:SYN: = $\bigcup_i \text{Syn}_i$
17:以 CC 为输入,按 4.1 节方法挖掘返回核,并根据定理 2 对 SYN 中同步核进行更新.记得到的返回核与更新后同步核的集合为

SynAndReturn;

18:按定理 3 集成 SynAndReturn 中的同步核与返回核,记集成后的同步核集合为 IntegratedSYN;

19:RETURN IntegratedSYN

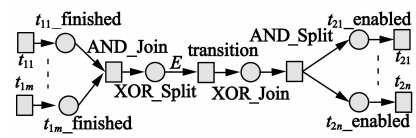
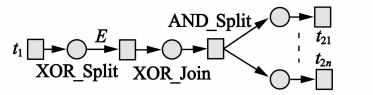
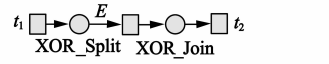
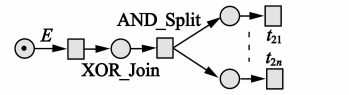
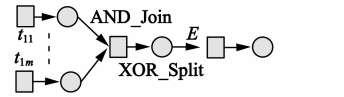
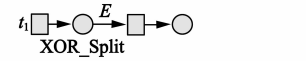
## 5 基于同步核的 WF-net 模型发现

本节将同步核转换为 WF-net,从而实现业务过程 WF-net 模型的重构.

以标准同步核 syn = 〈$T_1, T_2, E$〉为例,设 $T_1$为{$t_{11}$, $t_{12}$,…, $t_{1m}$}, $T_2$为{$t_{21}, t_{22}, \cdots, t_{2n}$}.由同步核含义,可设置库所 $t_{11}$_finished,…, $t_{1m}$_finished 分别表达任务 $t_{11}$,…, $t_{1m}$完成的状态,库所 XOR_Split 表达集合 $T_1$中的任务全部完成的状态,变迁 AND_Join 表达由单个任务的完成状态向 $T_1$中所有任务都完成的状态之间的迁移;同时设置库所 $t_{21}$_enabled,…, $t_{2n}$_enabled 分别表达任务 $t_{21}$,…, $t_{2n}$使能的状态,库所 XOR_Join 表达集合 $T_2$中的所有任务均被使能状态,变迁 AND_Split 表达上述两种状态之间的迁移;由 syn = 〈$T_1, T_2, E$〉的定义可知,当 $T_1$中的任务全部完成(库所 XOR_Split 有一个托肯)且条件 $E$ 成立时集合 $T_2$中所有任务被并发使能(向库所 XOR_Join 输出一个标记),设置变迁 transition 表达这一动作.由此可得表 8 中标准同步核到 WF-net 的转换规则.其它转换规则类似可得.

以前文所挖掘同步核集合 IntegratedSYN 为例,由表 8 转换规则可得图 2 所示 WF-net.其中未加标注的变迁对应不可见任务,部分流关系上标注了执行条件.不难发现,该 WF-net 与图 1 中的过程行为等价.

表 8 同步核到 WF-net 的转换规则

| | 同步核 | WF-net 模型 |
|---|---|---|
| 标准同步核或返回核 | $T_1=\{t_{11},\cdots,t_{1m}\}$<br>$T_2=\{t_{21},\cdots,t_{2n}\}$<br>syn = 〈$T_1, T_2, E$〉 |  |
| | $T_2=\{t_{21},\cdots,t_{2n}\}$<br>syn = 〈{$t_1$}, $T_2, E$〉 |  |
| | $T_1=\{t_{11},\cdots,t_{1m}\}$<br>syn = 〈$T_1$, {$t_2$}, $E$〉 |  |
| | syn = 〈{$t_1$}, {$t_2$}, $E$〉 |  |
| 起始同步核 | $T_2=\{t_{21},\cdots,t_{2n}\}$<br>syn = 〈∅, $T_2, E$〉 |  |
| | syn = 〈∅, {$t_2$}, $E$〉 |  |
| 结束同步核 | $T_1=\{t_{11},\cdots,t_{1m}\}$<br>syn = 〈$T_1$, ∅, $E$〉 |  |
| | syn = 〈{$t_1$}, ∅, $E$〉 |  |

## 6 相关工作比较与日志完备性分析

### 6.1 相关工作比较

下面结合日志实例 $L$ 的挖掘结果对嵌入到 Prom6.1[1]平台中的部分过程发现算法与本文方法进行对比分析.

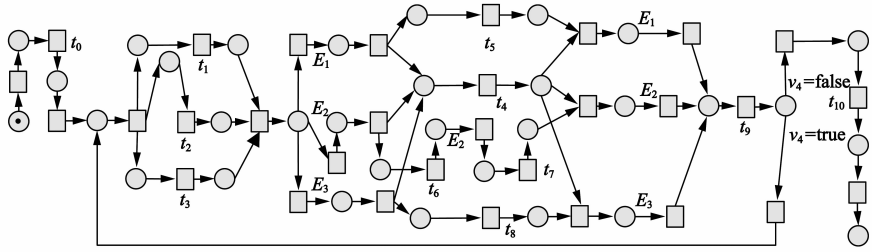



图2 由同步核集合IntegratedSYN转换得到的WF-net

AlphaMiner[9]挖掘得到的 WF-net 模型中只有变迁 $t_0$可以引发,之后,网系统由于 $t_1$、$t_2$、$t_3$与 $t_9$之间存在的循环依赖关系而进入死锁状态.之所以如此,一方面是 Alpha 算法无法处理不可见任务的挖掘,另一方面在于 Alpha 算法要求“只要两个活动能邻接地发生,则日志中必须存在一个两者邻接发生的活动轨迹”,而日志 $L$ 不满足这一条件.为解决不可见任务的挖掘问题,文献[12]将过程模型中包含的不可见任务进行功能分类,给

出了 SIDE、SKIP、REDO 以及 SWITCH 等四类不可见任务的形式化定义,同时给出了从日志中检测这四类不可见任务以及挖掘包含这四类不可见任务的 WF-net 的 alpha # 算法.相比而言,本文中,不可见任务是在将同步核转换为 WF-net 模型时得到的,前述四类不可见任务不必区分对待.但是,表达同一过程结构时,本文方法得到的 WF-net 模型中部分不可见任务实际可以进行合并化简,而 alpha # 算法不存在这一问题,这也是本文工作后续可以改进的一个方向.

HeuristicsMiner[16] 和 GeneticMiner[14] 挖掘到的模型中,任务 $t_4$不必等待 $t_1$、$t_2$和 $t_3$全部完成就可使能,该行为也不符合图 1 中业务过程的行为定义,原因在于这两个算法是以活动图为挖掘对象,而活动图无法有效表示任务之间的并发和冲突关系.

最后,如果忽略原始过程定义中的任务执行条件信息,则 ILPMiner[15] 挖掘到的 WF-net 模型与图 1 中的过程模型行为相同,但这一点并非总是成立.文献[18]中给出的一个日志实例便是如此,由于该日志对应的过程中有两个任务在不同的案例相关属性取值时具有相反的因果依赖关系,此时,ILPMiner 算法认为这些任务之间具有并发关系,从而挖掘到的过程模型与原始过程定义不符.本文方法能正确解决这一问题,因为本文进行模型发现前对案例进行了分簇,各案例簇对应的子模型中不存在上述结构,而且在基于同步核对各案例簇对应的子过程模型进行融合时能正确发现上述结构.

综述所述,基于案例簇与同步核的过程发现方法能处理任务依赖关系随案例相关属性取值而变化的现象,也能有效挖掘 WF-net 模型中存在的不可见任务.挖掘出的模型中包含流关系执行条件等语义信息,对日志完备性的要求较 Alpha 等经典算法更低,且部分步骤可并行化以提高算法效率,这是本文方法的优点.不过,本文方法也存在一些问题,例如,算法挖掘对象不支持重名任务,不显式支持非自由选择结构,虽然本文算法挖掘到的流关系执行条件可以隐式表达出各种长距离依赖关系.

此外,前文假设过程模型的循环回路上不存在新任务.而实际中,循环回路上可能存在结构复杂的子过程.此时,该循环结构对应的组合案例不存在与其相容的案例簇,前文方法无法正确挖掘该循环结构.但是,循环回路中出现的任务不会出现在任何基本案例中,据此可以识别出这些循环结构对应的任务轨迹子序列.接下来,只需将这些子序列从事件日志中单独提取出来组成一个新的子日志.如此以来,处理后的日志所对应的过程模型中上述循环结构会逐步减少.如此重复,最后可使得各个日志对应的过程模型中不存在上述循环结构.此时,可以用前文方法对这些子日志分别进行挖掘,最后将挖掘到的子模型进行合并即可.例如,在图 1 所示流程中,如果标注 $E_4$的循环回路上存在一个由任务 $A$ 和 $B$ 构成的并发结构,则事件日志中各基本案例对应的任务轨迹不会变化,各案例簇所对应同步核的挖掘方法和结果不会受影响.但是所有组合案例的任务轨迹中都会在任务 $t_9$与 $t_1$、$t_2$、$t_3$之间多出一个子序列 $AB$ 或者 $BA$.不难发现,任务 $A$ 与 $B$ 不会出现在任意一个案例簇的特征向量中,将所有组合案例任务轨迹中出现的由 $A$、$B$ 构成的子序列从任务轨迹中进行删除并单独保存,会发现这些子序列由 $AB$ 和 $BA$ 构成.删除 $A$、$B$ 之后各组合案例运行轨迹与正文中保持一致,由此仍然可以得到返回核$\langle\{t_9\},\{t_1,t_2,t_3\},v_4=\text{true}\rangle$.但是,由子序列 $AB$ 与 $BA$ 结合本文方法可以发现同步核$\langle\varnothing,\{A,B\},\text{true}\rangle$,再分析组合案例运行轨迹会发现每次执行完毕任务$\{t_9\}$后会执行$\{A,B\}$,而且每次执行完毕$\{A,B\}$后会执行任务$\{t_1,t_2,t_3\}$,由此可知,返回核$\langle\{t_9\},\{t_1,t_2,t_3\},v_4=\text{true}\rangle$需更新为$\langle\{t_9\},\{A,B\},v_4=\text{true}\rangle$和$\langle\{t_1,t_2,t_3\},\{A,B\},v_4=\text{true}\rangle$.至此,循环回路中子结构的挖掘完毕.

### 6.2　日志完备性分析

为保证本文过程发现方法所得结果的正确性,事件日志需满足如下完备性要求:(1)记录的案例必须足够丰富以保证所有的案例簇都能得到;(2)对于案例簇内具有并发关系的两个任务 $a$ 与 $b$,日志需包含至少两个隶属该案例簇的基本案例,在其中一个案例轨迹中 $a$ 先于 $b$ 执行,另一个案例轨迹 $b$ 先于 $a$ 执行;(3)对于每个循环结构以及与其相容的案例簇,日志中至少存在一个与之对应的组合案例;(4)案例相关的属性取值信息最好在日志中进行了记录,文中需要据此挖掘案例簇的特征条件.当日志中仅记录了任务轨迹信息时,可以人为引入一些布尔变量来表达案例簇特征条件和循环条件.例如,假设共有 3 个案例簇,则可以将一个案例簇的特征条件用布尔型变量 $\text{ClusterCond}_1$表示,第二个案例簇的特征条件设置为 $\text{ClusterCond}_2 \wedge (!\ \text{ClusterCond}_1)$,第三个案例簇特征条件设置为$(!\ \text{ClusterCond}_2) \wedge (!\ \text{ClusterCond}_1)$.在此基础上,各同步核与返回核的源任务集和目标任务集仍可得到,但是准确的条件信息无法获取.

## 7　总结

本文提出了基于案例簇和同步核的业务过程发现方法,具体工作包括三方面:(1)研究了基于同步核的任务同步机制建模方法,给出了由案例簇内的任务关系挖掘同步核的算法;(2)提出了基于返回核的循环结构建模方法以及由组合案例挖掘返回核的具体算法;

(3)研究了同步核与返回核的集成方法,设计了将集成后的同步核转换为 WF-net 的具体算法,实现了整体业务 WF-net 模型的重构.在后续工作中,我们将对不完备日志的挖掘以及算法的并行化等问题进一步研究.

## 参考文献


[1] W M P van der Aalst. Process Mining: Discovery, Conformance and Enhancement of Business Processes [M]. Heidelberg: Springer Verlag, 2011. 124 – 187.

[2] Chris J Turner, Ashutosh Tiwari, Richard Olaiya, Yuchun Xu. Process mining: from theory to practice [J]. Business Process Management Journal, 2012, 18(3): 493 – 512.

[3] Sherry X Sun, Qingtian Zeng, Huaiqing Wang. Process-mining-based workflow model fragmentation for distributed execution [J]. IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, 2011, 41(2): 294 – 310.

[4] Hua Duan, Qingtian Zeng, Huaiqing Wang, et al. Classification and evaluation of timed running logs of workflows based on process mining[J]. Journal of Systems and Software, 2009, 400 – 410.

[5] OMG. Business Process Modeling Notation (BPMN), Version 2.0[DB/OL]. http://www.bpmn.org/, 2014 – 7 – 24

[6] W M P van der Aalst, K M van Hee. Workflow Management: Models, Methods, and Systems[M]. London: MIT Press, 2004. 31 – 74.

[7] Qingtian Zeng, Sherry X Sun, Hua Duan, Cong Liu, Huaiqing Wang. Cross-organizationalcollaborative workflow mining from a multi-source log[J]. Decision Support Systems, 2013, 54(3): 1280 – 1301.

[8] 宋炜,刘强.基于模拟退火算法的过程挖掘研究[J].电子学报,2008,36(4A):135 – 139.
Song Wei, Liu Qiang. Business process mining based on simulated annealing[J]. Acta Electronica Sinica, 2008, 36(4A): 135 – 139. (in Chinese)

[9] W M P van der Aalst, A J M M Weijters, L Maruster. Workflow mining: Discovering process models from event logs[J]. IEEE Transactions on Knowledge and Data Engineering, 2004, 16(9): 1128 – 1142.

[10] Lijie Wen, Jianmin Wang, Jiaguang Sun. Mining invisible tasks from event logs[A]. APWeb/WAIM 2007[C]. Berlin, Germany: Springer Verlag, 2007. 358 – 365.

[11] Lijie Wen, Jianmin Wang, W M P van der Aalst, Biqing Huang, Jiaguang Sun. A novel approach for process mining based on event types[J]. J Intell Inf Syst, 2009, 32(2): 163 – 190.

[12] Lijie Wen, Jianmin Wang, W M P van der Aalst, Biqing Huang, Jiaguang Sun. Mining process models with prime invisible tasks[J]. Data Knowl Eng 2010, 69(10): 999 – 1021.

[13] 闻立杰.基于工作流网的过程挖掘算法研究[D].北京:清华大学,2007.

[14] A K A de Medeiros, A J M M Weijters, W M P van der Aalst. Genetic process mining: Anexperimental evaluation[J]. Data Mining and Knowledge Discovery, 2007, 14(2): 245 – 304.

[15] R Bergenthum, J Desel, R Lorenz, S Mauser. Process mining based on regions of languages[A]. International Conference on Business Process Management [C]. Berlin, Germany: Springer Verlag, 2007. 375 – 383.

[16] A J M M Weijters, J T S Ribeiro. Flexible heuristics miner (FHM)[A]. 2011 IEEE Symposium on Computational Intelligence and Data Mining[C]. Paris: IEEE, 2010. 310 – 317

[17] 顿海强,赵文,邓鹏鹏,等.一种基于 RFID 数据集的物品工作流挖掘方法[J].电子学报,2009,36(B12):86 – 93.
Dun Hai-qiang, Zhao Wen, Deng Peng-peng, et al. A commodity workflow mining approach based on RFID data sets [J]. Acta Electronica Sinica, 2009, 36(B12): 86 – 93. (in Chinese)

[18] 鲁法明,曾庆田,包云霞,段华,张昊.基于流程案例簇的任务关系挖掘算法[J].计算机集成制造系统,2013,19(8):1771 – 1783.

[19] Blackburn R R, Bauer K M, Amsler D E, Boselly S E, McElroy A D. Snow and Ice Control: Guidelines for Materials and Methods[R]. Washington, DC: NCHRP, 2004.

[20] Quinlan J R. C4.5: Programs for Machine Learning[M]. New South Wales: Morgan Kaufmann Publishers, 1993. 32 – 103


## 作者简介


**鲁法明** 男,1981 年 5 月生,山东新泰人,博士,山东科技大学讲师,研究方向包括过程挖掘,Petri 网等.
E-amail: fm_lu@163.com

**曾庆田** 男,1976 年 1 月生,山东高密人,博士,山东科技大学教授、博导,研究方向包括过程挖掘、智能信息服务,Petri 网等.

**段 华** 女,1976 年 10 月生,山东临清人,博士,山东科技大学副教授,研究方向包括机器学习、优化方法等.

**刘 聪** 男,1990 年 8 生,山东淄博人,硕士研究生,研究兴趣包括 Petri 网、流程建摸与分析等.