DOI: 10.16452/j.cnki.sdkjzk.2023.06.009 文章编号:1672-3767(2023)06-0075-10

基于逻辑 Petri 网的循环选择驱动循环结构过程模型修复方法

刘 伟,薄玉娟,杜玉越,孙红伟
(山东科技大学 计算机科学与工程学院,山东 青岛 266590)

摘 要:模型修复是一种过程增强技术。现有模型修复方法较少考虑含间接依赖关系的过程模型,在表示结构间的间接依赖关系时存在不足,修复含循环选择驱动循环结构的模型时难以描述结构间的间接依赖关系。本研究基于逻辑 Petri 网,针对循环选择驱动循环结构,提出在修复循环选择驱动循环结构的同时可以表达结构间的间接依赖关系的方法。在从日志中获取循环序列和选择序列算法的基础上,通过定理确定模型与日志是否存在偏差并找到偏差位置;根据不同结构提出修复算法修复模型,并使用关联规则描述结构间的间接依赖关系;最后通过实验证明方法的可行性。

关键词:间接依赖关系;逻辑 Petri 网;循环结构;过程模型;修复方法
中图分类号:TP311 **文献标志码**:A

Repair method for process model of loop-choice-driven loop structure based on logic Petri nets

LIU Wei, BO Yujuan, DU Yuyue, SUN Hongwei
(College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China)

Abstract: Model repair is a technique applied to process enhancement. But due to the rare consideration of process models involving indirect dependencies, the existing model repair methods are insufficient in representing the indirect dependencies among different structures, especially in describing the indirect dependencies between structures when they are used to repair a model with a loop-choice-driven loop structure. This paper proposed a model repair method for loop-choice-driven loop structures based on logic Petri nets, which could express the indirect dependencies between structures while repairing the loop-choice-driven loop structure. Based on the proposed algorithms for extracting loop sequence and choice sequence from the logs, deviations between the model and logs, as well as the deviation locations, were determined by using theorems. Then a repair algorithm was presented to repair the model according to different structures and the indirect dependencies among different structures were described by using the association rules. Finally, the feasibility of the proposed method was validated by experiments.

Key words: indirect dependency; logic Petri nets; loop structure; process model; repair method

如今多数企业使用信息系统管理业务流程,而过程挖掘主要从信息系统中抽取信息[1]。过程挖掘可分为过程发现、一致性检查和过程增强等阶段[2]。过程发现通过算法从事件日志中挖掘模型;一致性检查是将日志与模型进行比对[3-4],若出现偏差说明过程模型不能反映现实,需要对过程模型进行修复;过程增强使用现有事件日志对原始模型进行改进或扩展,以适应更多事件日志。

收稿日期:2022-06-17
基金项目:教育部人文社科规划基金项目(23YJAZH084);青岛市社会科学规划项目(QDSKL2201131);山东省教育教学研究重点课题(2023JXZ001)
作者简介:刘 伟(1977—),男,山东泰安人,教授,博士生导师,主要从事过程挖掘、Petri 网理论与应用等研究.
E-mail:liuwei_doctor@yeah.net
薄玉娟(1997—),女,山东临沂人,硕士研究生,主要从事过程挖掘、Petri 网理论等研究.

现有模型修复方法主要使用校准获得偏差[5]。比如,最具代表性的 Fahland 方法[6]根据过程发现算法挖掘子过程,修复后的模型无限次重复子过程,然而这种子过程通常不允许重复;Knapsack 方法[7]通过设置模型移动和日志移动成本,比较多种不同方案得到最小总成本,获得最佳模型修复方法,但该方法修复的模型包含不可见变迁。这两种方法修复的过程模型虽然提高拟合度,但不能保证模型的简洁度,不能获得很好的修复效果。文献[8]基于 Petri 网的过程模型修复方法,针对循环结构和选择结构,提出模型偏差域识别方法,该方法虽然保证了简洁度,但没有考虑各结构间的逻辑关系。为了解决该问题,基于逻辑 Petri 网,文献[9]提出构造自由循环结构过程模型的修复方法,可根据事件日志识别有问题的变迁构造循环结构。

现有模型修复方法很少考虑包含间接依赖关系的过程模型。如含循环选择驱动循环结构的过程模型,由两个循环块和一个选择块组成,第二个循环体的循环次数取决于第一个循环体的循环次数和执行的选择分支[10]。随着业务流程愈加复杂,流程出现了各种特殊事件日志,如第一个循环块没有执行完就执行选择块,选择分支混合执行,间接依赖关系发生改变等。现有修复方法通过添加不可见变迁或自循环修复模型,但简洁度不高,修复结果不能满足要求。因此,本研究针对循环选择驱动循环结构,提出一种基于逻辑 Petri 网的过程模型修复方法。首先,针对循环选择驱动循环结构,提出从事件日志中提取循环子日志和选择子日志的算法,判断模型中的循环结构和选择结构;其次,提出两个定理用于发现事件日志和原始模型中关于循环和选择结构的偏差,找到偏差位置并提出修复算法进行模型修复;然后,引入关联规则表达不同结构间的间接依赖关系;最后,对建筑公司工程项目过程进行模拟实验,验证提出方法的合理性和可行性。

1 基本概念

首先介绍一些基本概念,包括迹和事件日志、Petri 网、过程模型、逻辑 Petri 网等。

定义 1(迹和事件日志)[11] A 是一个活动集合,A^* 表示集合 A 上所有有限序列,$B(A^*)$表示集合 A 上的多集集合,$\sigma\in A^*$ 为一个迹,$L\in B(A^*)$为一个事件日志。

定义 2(前驱与后继)[11] $\sigma\in A^*$ 为一个迹,活动 $a\in\sigma$,且在 σ 中 a 的位置为 i,${}^{\#}a$ 是 a 的前驱,在 σ 中的位置为 $i-1$;$a^{\#}$ 是 a 的后继,在 σ 中的位置为 $i+1$。

定义 3(Petri 网)[12-14] $PN=(P, T; F, M)$为一个 Petri 网,当且仅当:

1) $N=(P, T; F)$为一个网,P 为库所集,T 为变迁集;

2) $F\subseteq(P\times T)\cup(T\times P)$表示有向连接弧集合;

3) $M:P\to N$ 为一个标识。

定义 4(前集和后集)[14] $PN=(P, T; F, M)$为一个 Petri 网,$\bullet x=\{y\in P\cup T\wedge(y, x)\in F\}$为 x 的前集,$x\bullet=\{y\in P\cup T\wedge(x, y)\in F\}$为 x 的后集。

定义 5(过程模型)[15] $N_s=(PN, a, M_i, M_f)$为一个过程模型,其中:

1) $PN=(P, T; F, M)$为一个 Petri 网;

2) $a:T\to A\cup\{\tau\}$是变迁到活动的映射函数,τ 为不可见变迁;

3) $m_i=[p_i]\in M_i$ 是初始标识,$m_f=[p_o]\in M_f$ 是终止标识。

图 1 为一个基于 Petri 网的过程模型实例。

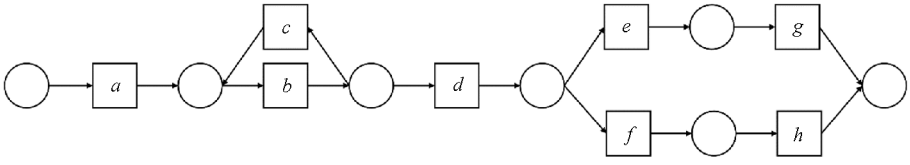

图 1 基于 Petri 网的过程模型实例

Fig. 1 Process model example based on Petri nets

定义 6(过程树)[16] A 为活动集合,$\oplus=\{\to, \times, \wedge, \circlearrowleft\}$ 为操作符集合,τ 为不可见变迁,其中:

1) $a\in A\cup\{\tau\}$为一棵过程树;

2) 若 PT_1, PT_2,…,PT_n 为 n 棵过程树,则$\oplus(PT_1, PT_2,\cdots,PT_n)$也是一棵过程树,→表示顺序关系,×表示选择关系,∧表示并发关系,∪表示循环关系。

图 1 过程模型可表示为过程树 PT_1:→(a, ∪(b,c),d,×(→(e,g),→(f,h)))。

定义 7(逻辑 Petri 网)[17]　$LPN=(P, T; F, I, O, M)$为一个逻辑 Petri 网,其中:

1) P 是有限库所集;

2) $T=T_I\cup T_O\cup T_D$ 是有限变迁集,$P\cup T\neq\emptyset$,$P\cap T=\emptyset$;若 $t\in T_I\cup T_O$, $\bullet t\cap t\bullet=\emptyset$;其中,$T_I$ 为逻辑输入变迁集,T_O 为逻辑输出变迁集,T_D 为经典 Petri 网变迁集;

3) $F\subseteq(P\times T)\cup(T\times P)$为有向连接弧集合;

4) I 是从逻辑输入变迁到逻辑输入函数的映射,对 $\forall t\in T_I$,$I(t)=f_I(t)$;

5) O 是从逻辑输出变迁到逻辑输出函数的映射,对 $\forall t\in T_O$,$O(t)=f_O(t)$;

6) $M:P\to N$ 为一个标识函数。

2　循环选择及其驱动循环序列确定算法

本部分介绍循环选择驱动循环结构,并划分为循环结构和选择结构分别进行分析。图 2 为一个循环选择驱动循环结构过程模型,使用关联规则表示不同结构之间的间接依赖关系,比如关联规则$\langle b, c\rangle^2\wedge\langle e, g\rangle^1\Rightarrow\langle j, k\rangle^1$ 表示循环活动 b、c 发生两次后执行 e、g 选择分支,则第二个循环块执行一次循环。

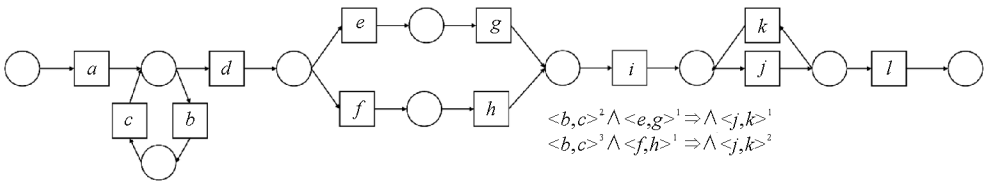

图 2　循环选择驱动循环结构过程模型

Fig. 2　Process model of loop-choice-dirven loop structure

2.1　循环序列确定算法

对循环选择驱动循环结构进行修复,首先要找到模型中所有循环结构。假设每个过程模型都有一个唯一的过程树,过程树中每个操作符代表模型的一个结构。例如在模型中有一个循环结构,则相应过程树节点 $n=$“∪”。为方便起见,$F_i(n)$表示 n 节点处第 i 个子树的所有叶节点。

定义 8(活动发生次数)　$\sigma\in L$ 为日志中的一个迹,$s\in\sigma$ 为一个序列,对于任意活动 $a\in\sigma$,num(a,σ)表示活动 a 在迹中出现的次数,num(s,σ)表示序列 s 在迹中出现的次数。

定义 9(循环活动)　$\sigma\in L$ 为日志中的迹,$a\in\sigma$ 为一个活动,若 num(a,σ)>1,则 a 为循环活动。

循环活动集合 $S_{LA}=\{a\in\sigma\mid\exists\sigma\in L\wedge\text{num}(a,\sigma)>1\}$。

定义 10(活动集)　A 为活动集合,对任意序列 $s\in A^*$,$\kappa(s)$为序列 s 中的活动集。

定义 11(循环序列)　S_{LA} 为循环活动集合,$s=\langle s[1],s[2],\cdots,s[i],\cdots,s[n]\rangle\in S_{LA}{}^*$ 为一个循环序列,其中 $s[i]\in S_{LA}$,$i\in\{1,2,\cdots,n-1\}$。以$|s|$表示序列 s 的长度。若:

1) $\kappa(s)=1$,s 为自循环序列;

2) $\kappa(s)=2$,且 $s[1]>_L s[2]$,s 为短循环序列,$|s|=2$;

3) $\kappa(s)>2$,且 $s[i]>_L s[i+1]$,$|s|-2>i>0$,s 为长循环序列。

L_s 为所有循环序列集合。

定义 12(循环开始活动和循环结束活动)　S_{LA} 为循环活动集合,$s\in S_{LA}{}^*$ 为一个循环序列,循环开始活动为 S_{LA_s},循环结束活动为 S_{LA_e},其中:

1) $S_{LA\_s}=(a_i\in S_{LA}\mid a_i=s_i[1])$;

2) $S_{LA\_e}=(a_i\in S_{LA}\mid a_i=s_i[n])$,$i\in\{1,2,\cdots,m\}$。

算法 1 为循环序列确定算法。步骤 1)初始化循环序列;步骤 2)、3)遍历日志和迹;步骤 4)~5)表明若迹 σ_i 中的活动 a_j 出现次数>1,则活动 a_j 为循环活动;步骤 6)~8)遍历循环活动集合,添加循环活动到循环序列 s 中;步骤 9)~12)表示若循环活动 a,b 为顺序关系且活动 b 为 s 的第一个活动,把活动 a 添加到 s 中且在活动 b 之前;步骤 13)~15)表示若循环活动 a,b 为顺序关系且活动 a 为 s 的最后一个活动,则把活动 b 添加到 s 中且在活动 a 之后;步骤 16)、17)把得到的循环序列 s 添加到循环序列集合 L_s 中并返回 L_s。

例 1 $L=\{\langle a,b,c,g\rangle,\langle a,b,c,d,e,g\rangle,\langle a,b,c,d,e,b,c,d,e,g\rangle\}$。$\mathrm{num}(b,\sigma_3)=2$,$\mathrm{num}(c,\sigma_3)=2$,$\mathrm{num}(d,\sigma_3)=2$,$\mathrm{num}(e,\sigma_3)=2$,所以 $S_{LA}=\{b,c,d,e\}$。根据算法 1,$S_{LA}=S_{LA}-b=\{c,d,e\}$,$s=\langle b\rangle$,因为 $b>_L c$,有 $s=s+c=\langle b,c\rangle$,$S_{LA}=S_{LA}-c=\{d,e\}$,同理得到 $s=\langle b,c,d,e\rangle$。

算法 1 循环序列确定算法

输入:事件日志 L

输出:循环序列 L_s

1) $S_{LA}\leftarrow\varnothing$;$s\leftarrow\varnothing$;
2) For ($i=1,\sigma_i\in L,i++;|L|$) do
3) For ($j=1,a_j\in\sigma;j++;|\sigma|$) do
4) if ($\mathrm{num}(a_j,\sigma_i)>1$) then
5) $S_{LA}=S_{LA}\cup a_j$
6) For ($k=1,a_k\in S_{LA};k++;|S_{LA}|$) do
7) $S_{LA}\leftarrow S_{LA}-a_k$
8) $s\leftarrow s+a_k$
9) For ($\forall a,b\in S_{LA}$)do
10) if ($a>_L b$ and $b=s[1]$) then
11) $s\leftarrow a+s$
12) $S_{LA}\leftarrow S_{LA}-a$
13) if ($a>_L b$ and $a=s[|s|]$) then
14) $s\leftarrow s+b$
15) $S_{LA}\leftarrow S_{LA}-b$
16) $L_s\leftarrow L_s\cup s$
17) return L_s

2.2 选择序列确定算法

提取出事件日志中的循环序列后,下一步确定事件日志中的选择序列。

定义 13(选择活动) $\sigma_1,\sigma_2\in L$ 为日志中的两个迹,$a\in\sigma$ 为迹中的一个活动,若 $a\in\sigma_1$,$a\notin\sigma_2$ 且 $a\notin S_{LA}$,则 a 为选择活动。选择活动集合 $S_{CA}=\{a\in A\mid\exists\sigma_1,\sigma_2\in L\wedge a\in\sigma_1\wedge a\notin{}_2\wedge a\notin S_{LA}\}$。

定义 14(选择序列) S_{CA} 为选择活动集合,$f\in S_{CA}{}^*$ 为一个选择序列,其中 $\forall f[i]\in S_{CA}$,若:

1) $\kappa(f)=1$,f 为只包含一个活动的选择序列;

2) $\kappa(f)\geqslant 2$,且 $f[i]>_L f[i+1]$,$i\in\{1,2,\cdots,n-1\}$,f 为长度为 n 的选择序列;

L_f 为所有选择序列集合。

定义 15(选择开始活动和选择结束活动) S_{CA} 为选择活动集合,$f\in S_{CA}{}^*$ 为一个选择序列,选择开始活动为 S_{CA_s},选择结束活动为 S_{CA_e},其中:

1) $S_{CA\_s}=(a_i\in S_{CA}\mid a_i=f_i[1])$;

2) $S_{CA\_e}=(a_i\in S_{CA}\mid a_i=f_i[n])$,$i\in\{1,2,\cdots,m\}$.

定义 16(选择分支) $N_s=(PN,a,M_i,M_f)$为一个过程模型,PT 为对应过程树,$CB=(t_1,t_2,\cdots,t_n)$是一个选择分支元组,其中:

1) $\exists n=\text{“}\times\text{”}\in PT$;

2) $t_k\in F_i(n)$,$1\leqslant k\leqslant m$,$1\leqslant i\leqslant n$。

选择分支集合 $S_{CB}=\{(t_1,t_2,\cdots,t_n)\mid t_k\in F_i(n),1\leqslant k\leqslant m,1\leqslant i\leqslant n,\forall n=\text{“}\times\text{”}\in PT\}$。

算法 2 为选择序列确定算法。步骤 1)初始化选择序列 f,选择活动集合 S_{CA};步骤 2)遍历日志;步骤 3)~4)表明若活动 a 属于 L,但不属于迹 σ_i,则 a 为选择活动;步骤 5)~9)遍历 S_{CA},若活动 a,b 为顺序关系且 a 不属于 S_{CA},b 是选择活动,则添加 b 到 f;步骤 10)~12)表示若活动 a,b 为顺序关系且 a 属于 S_{CA},b 不是选择活动,则添加 a 到 f;步骤 13)~14)把 f 添加到选择序列集合 L_f 中并返回 L_f。

例 2 事件日志 $L=\{\sigma_1,\sigma_2\}=\{\langle a,b,d,f\rangle,\langle a,c,e,f\rangle\}$,选择活动集合 $S_{CA}=\{b,c,d,e\}$。对于 σ_1,有 $a>_L b$ 且 $b\in S_{CA}$,$a\notin S_{CA}$。根据算法 2,$f=\langle b\rangle$,$S_{CA}=\{c,d,e\}$,$d>_L f$ 且 $d\in S_{CA}$,$f\notin S_{CA}$,选择序列 $f=\langle b,d\rangle$,$S_{CA}=\{c,e\}$。同理可得选择序列集合 $L_f=\{\langle b,d\rangle,\langle c,e\rangle\}$。

2.3　循环选择驱动循环序列确定算法

定义 17(切分子迹)　$\sigma=\langle r_1,s,r_2,\cdots,r_n\rangle\in L$ 为日志上的一个迹,$\mathrm{del}(s,\sigma)=\{\langle r_1\rangle,\langle r_2,\cdots,r_n\rangle\}$。

定义 18(关联对)　$s_1,s_2\in L_s$ 为循环序列,$f\in L_f$ 为选择序列,$a\in s_1$,$b\in s_2$ 且 $b\in S_{LA\_e}$,$c\in f$,且 $c\in S_{CA\_e}$。一个关联对为 $\bar{\omega}=(X,Y)$,其中:$X=s_1^{\mathrm{num}(a,\sigma)}\wedge f^{\mathrm{num}(b,\sigma)}$,$Y=s_2^{\mathrm{num}(c,\sigma)}$,关联对集合为 $A_{\bar{\omega}s}$。

算法 3 为循环选择驱动循环序列确定算法。步骤 1)初始化关联对集合 $A_{\bar{\omega}s}$,循环选择驱动循环序列 $L_{lc\to l}$;步骤 2)、3)遍历事件日志和循环序列集合;步骤 4)～7)表明若有循环序列 s_j 属于 σ_i,活动 a 属于 s_j,且 a 是循环结束活动,记录 a 出现次数,同时切分子迹 σ_i 为 λ_1;步骤 8)遍历选择序列集合;步骤 9)～12)表明若有选择序列 f_k 属于 $\lambda_1[2]$,活动 b 属于 f_k,且 b 是选择开始活动,记录 b 出现次数,同时切分子迹 $\lambda_1[2]$为 λ_2;步骤 13)遍历循环序列集合;步骤 14)～16)表明若有循环序列 s_m 属于 $\lambda_2{}^{[2]}$,活动 c 属于 s_m,且 c 是循环结束活动,记录 c 出现次数;步骤 17)、18)得到 $A_{\bar{\omega}s}$;步骤 19)、20)得到并返回 $L_{lc\to l}$。

关联规则用于发现活动或结构间的间接依赖关系,表示为 $X\Rightarrow Y$,即如果 X 发生,则 Y 发生。结合算法 3 得到的关联对,组合成关联规则表示结构间的间接依赖关系。关联规则用 $R=(X\Rightarrow Y)$表示,$X=s_1^{\mathrm{num}(a,\sigma)}\wedge f^{\mathrm{num}(b,\sigma)}$,$Y=s_2^{\mathrm{num}(c,\sigma)}$。关联规则集合为 S_R,前集 X 集合为 S_{pre_R},后集 Y 集合为 S_{por_R}。

算法 2　选择序列确定算法

输入:事件日志

输出:选择序列 L_f

1) $S_{CA}\leftarrow\emptyset$,$f\leftarrow\emptyset$;
2) For ($i=1,\sigma_i\in L;i++;|L|$) do
3)　if $a\in L\wedge a\notin\sigma_i$ then
4)　$S_{CA}\leftarrow S_{CA}\cup a$
5) For ($i=0;i<|S_{CA}|;i++$) do
6) For $\forall a,b\in A$ do
7)　if ($a>_L b\wedge a\notin S_{CA}\wedge b\in S_{CA}$) then
8)　$f\leftarrow f+b$
9)　$S_{CA}\leftarrow S_{CA}-b$
10)　if ($a>_L b\wedge a\in S_{CA}\wedge b\notin S_{CA}$) then
11)　$f\leftarrow f+a$
12)　$S_{CA}\leftarrow S_{CA}-a$
13) $L_f\leftarrow L_f\cup f$
14) return L_f

算法 3　循环选择驱动循环序列确定算法

输入:事件日志 L,S_{LA_e},S_{CA_s},L_s,L_f

输出:循环序列 $L_{lc\to l}$

1) $\bar{\omega}\leftarrow\emptyset$,$\lambda\leftarrow\emptyset$,$A_{\bar{\omega}s}\leftarrow\emptyset$,$L_{lc\to l}\leftarrow\emptyset$
2) For ($i=1,\sigma_i;i++;|L|$) do
3)　For ($j=1,s_j\in L_s;j++;|L_s|$) do
4)　if ($s_j\subseteq\sigma_i$) then
5)　if ($a\in s_j$,and $a\in S_{LA\_e}$) then
6)　num(a,σ_i)
7)　$\lambda_1=\mathrm{del}(s_j,\sigma_i)$
8)　For ($k=1,f_k\in L_f;k++;|L_f|$) do
9)　if ($f_k\subseteq\lambda_1[2]$) then
10)　if ($b\in f_k$,and $b\in S_{CA\_s}$) then
11)　num(b,σ_i)
12)　$\lambda_2=\mathrm{del}(f_k,\lambda_1[2])$
13)　For ($m=1,s_m\in L_s;m++;|L_s|$) do
14)　if ($s_m\subseteq\lambda_2[2]$) then
15)　if ($c\in s_m$,and $c\in S_{LA\_e}$) then
16)　num(c,σ_i)
17)　$\bar{\omega}\leftarrow\bar{\omega}+\langle s_1^{\mathrm{num}(a,\sigma)}\wedge f^{\mathrm{num}(b,\sigma)},s_2^{\mathrm{num}(c,\sigma)}\rangle$
18)　$A_{\bar{\omega}s}\leftarrow A_{\bar{\omega}s}+\bar{\omega}$
19)　$L_{lc\to l}\leftarrow L_{lc\to l}+\langle s_1,f,s_2\rangle$
20) return $L_{lc\to l}$

3　循环选择驱动循环结构过程模型修复方法

对于循环选择驱动循环结构,不同循环次数和选择分支执行会影响第二个循环块的执行。在实际过程中,可能会发生循环结构没有执行完就执行选择结构、选择块的选择分支交叉执行等情况。此时迹中没有包含完整循环活动,或迹中选择活动来自不同的选择分支。本节给出含循环选择驱动循环结构过程模型修复算法,在确定过程模型中的偏差位置后,对循环选择驱动循环结构进行修复。

3.1　偏差位置确定算法

本研究将循环选择驱动循环结构分为不同结构分别确定偏差位置。步骤为:①判断循环结构是否存在偏差。若某循环结构不存在循环前进路径[1]、循环活动出现次数不同且出现次数大于 1,则确定循环结构存在偏差;②判断选择结构是否存在偏差。若某迹中选择活动来自不同的选择分支,说明选择结构存在偏差;③通过关联规则判定原过程模型的间接依赖关系是否正确,若存在错误,则利用关联规则进行改进。

定理 1　若 $a\in S_{LA}$ 是一个没有循环前进路径的循环结构中的循环活动,且存在某个迹有 $\mathrm{num}(a,\sigma)>$

$\mathrm{num}(a^{\#},\sigma)>1$,则偏差位置在循环结构且 a 为偏差变迁。

证明:循环活动 a 有 $\mathrm{num}(a,\sigma)>1$,即在某迹中出现次数大于 1,说明循环体执行了多次,同时 $\mathrm{num}(a,\sigma)>\mathrm{num}(a^{\#},\sigma)$,$a$ 的出现次数大于 a 的后继变迁出现次数,说明 a 执行完后没有执行 $a^{\#}$,循环体没有执行完,确定循环结构存在偏差。

定理 2 a、$b\in S_{CA}$ 为两个选择活动,且 a、b 属于不同的选择分支,若存在某个迹同时含有 a、b 两个选择活动,则选择结构含有偏差,偏差变迁为 a、b。

证明:a、$b\in S_{CA}$ 为两个选择活动,且 a、b 选择分支不同,说明 a 和 b 不能同时出现在一个迹中,因为选择结构的每次执行只会执行一个选择分支,若存在某个迹同时含有 a、b 两个选择活动,说明 a、b 所属的选择分支同时混合执行,确定选择结构含有偏差。

算法 4 为偏差变迁确定算法。步骤 1)初始化偏差变迁 Dt_l、Dt_c;步骤 2)遍历日志;步骤 3)~4)表明若活动 a,$a^{\#}$ 属于循环活动 S_{LA},并且 a 在迹中出现次数大于 $a^{\#}$ 在迹中出现次数大于 1,则 a 为循环偏差变迁;步骤 5)~6)表明若活动 b、c 属于选择活动 S_{CA},b 和 c 属于不同选择分支,则 b、c 为选择偏差变迁;步骤 7)返回偏差变迁。

算法 4 偏差变迁确定算法

输入:事件日志,循环活动 S_{LA},选择活动 S_{CA};

输出:偏差变迁 Dt_l,Dt_c.

1) Dt_l,$Dt_c\leftarrow\varnothing$
2) For ($i=1,\sigma_i\in L;i{+}{+};i\leqslant|L|$) do
3) if ($a,a^{\#}\in S_{LA}\wedge\mathrm{num}(a,\sigma)>\mathrm{num}(a^{\#},\sigma)>1$) then
4) $Dt_l\leftarrow a$
5) if ($b,c\in S_{CA}\wedge b\in S_{CBi}\wedge c\in S_{CBj}$) then
6) $Dt_c\leftarrow b,c$
7) return Dt_l,Dt_c

算法 5 循环选择驱动循环结构过程模型修复算法

输入:事件日志,偏差变迁 Dt_l,Dt_c,选择开始活动 S_{CA_s},选择结束活动 S_{CA_e};

输出:修复后的模型 LPN'.

1) $LPN'\leftarrow Ns$
2) 根据算法 4,找到 Dt_l,Dt_c,
3) if ($\exists Dt\in Dt_l$) then
4) For ($Dt_k,k=1;k{+}{+};k\leqslant|Dt_l|$) do
5) $DP=(t_i,p_o)$
6) $t_i=Dt_k,p_o={}^{\bullet}Dt_k$
7) $OP1={}^{\bullet}Dt_k,NP1=Dt_k{}^{\bullet}$
8) $F'=F'\cup(t_i\rightarrow p_o)$
9) $O(t_i)=OP1\otimes NP1$
10) if ($\exists Dt_i,Dt_j\in Dt_c$) then
11) if ($Dt_i\notin S_{CA\_e}\wedge Dt_j\notin S_{CA\_s}$) then
12) $DP=(t_i,p_o)$
13) $t_i=Dt_i,p_o={}^{\bullet}Dt_j$
14) $OP2=Dt_i{}^{\bullet},NP2={}^{\bullet}Dt_j$
15) $F'=F'\cup(t_i\rightarrow p_o)$
16) $O(t_i)=OP2\otimes NP2$
17) return LPN'

3.2 循环选择驱动循环结构过程模型修复方法

算法 5 为循环选择驱动循环结构过程模型修复算法。步骤 1)初始化 LPN';步骤 2)根据算法 4 找到偏差变迁 Dt_l,Dt_c;步骤 3)~9)表明若存在循环偏差变迁 Dt_k,遍历 Dt_l,确定偏差位置 $DP=(t_i,\ p_o)$,其中 $t_i=Dt_k$,$p_o={}^{\bullet}Dt_k$;步骤 10)~16)表明若存在选择偏差变迁 Dt_i 和 Dt_j,同时 Dt_i 不是选择结束活动,Dt_j 不是选择开始活动,确定偏差位置 $DP=(t_i,\ p_o)$,其中 $t_i=Dt_i$,$p_o={}^{\bullet}Dt_j$;步骤 17)返回 LPN'。

例 3 $L_1=\{\sigma_1,\sigma_2,\sigma_3,\sigma_4\}=\{\langle a,b,c,d,e,g,j,l\rangle,\langle a,b,c,b,c,b,d,e,h,i,j,k,j,k,j,k,j,l\rangle,\langle a,b,c,b,c,d,e,g,i,j,k,j,l\rangle\langle a,b,c,b,c,b,c,d,f,h,i,j,k,j,k,j,l\rangle\}$。在过程模型 N_{S2} 中,循环活动为$\{b,c,j,k\}$,选择活动为$\{e,f,g,h\}$。对于迹 σ_2,包含活动 b 和 c 的循环块中 b 的发生次数比 c 的发生次数多并且两个活动发生了多次,说明循环结构发生偏差。根据算法 4 确定偏差变迁为 b,b 的前集库所为 p_2,根据算法 5 确定偏差位置 $DP=(b,\ p_2)$。添加一条从偏差变迁 b 到前集库所 p_2 的弧,修改 b 为逻辑输出变迁,$O(b)=p_2\otimes p_3$。迹 σ_2 中的选择活动来自两个不同的选择分支,根据算法 4 确定偏差变迁为 e 和 h,e 的后集库所为 p_5,h 的前集库所为 p_6。因为 h 为选择结束活动,根据算法 5 确定偏差位置 $DP=(e,\ p_6)$,添加一条从偏差变迁 e 到前集库所 p_6 的弧,修改 e 为逻辑输出变迁,$O(e)=p_5\otimes p_6$,得到图 3。Fahland 方法添加了一个重复变迁,两个不可见变迁和六条弧重放 L_1 中的迹,如图 4。相对于只添加了两条弧的图 3,Fahl-

and 方法比较复杂,并且不能表示过程模型不同结构间的间接依赖关系。

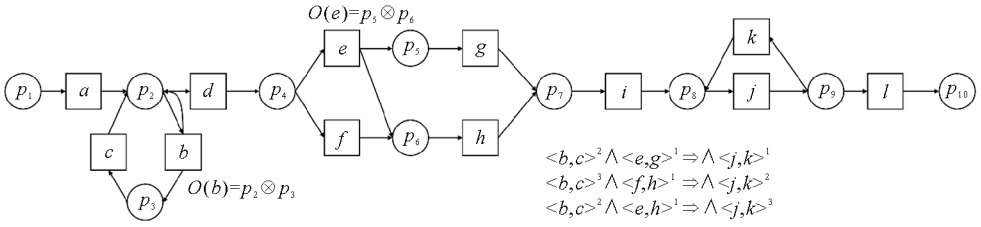

图 3　基于逻辑 Petri 网的过程模型修复方法修复的模型

Fig. 3　Model repaired by process model repair method based on logic Petri nets

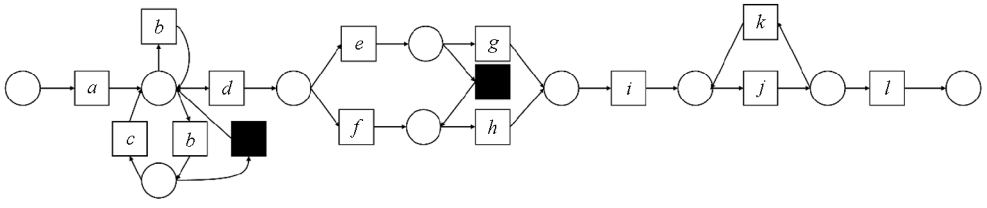

图 4　基于 Fahland 方法修复的模型

Fig. 4　Model repaired by Fahland's method

4　模拟实验

本研究选用 Fahland 方法进行对比实验,评估提出方法的正确性和有效性。基于逻辑 Petri 网的模型修复采用手工模拟方式,Fahland 方法已在过程挖掘工具 Prom6.6(http://www.promtools.org/prom6/)中实现。

4.1　实验数据与模型

以建筑公司工程项目实施过程为例,如图 5 所示。建筑公司首先投标,中标后监理单位对施工方案进行专业评审,如果评审不通过需要对施工方案进行修改,重新确定施工方案。确定方案后需要准备施工材料,

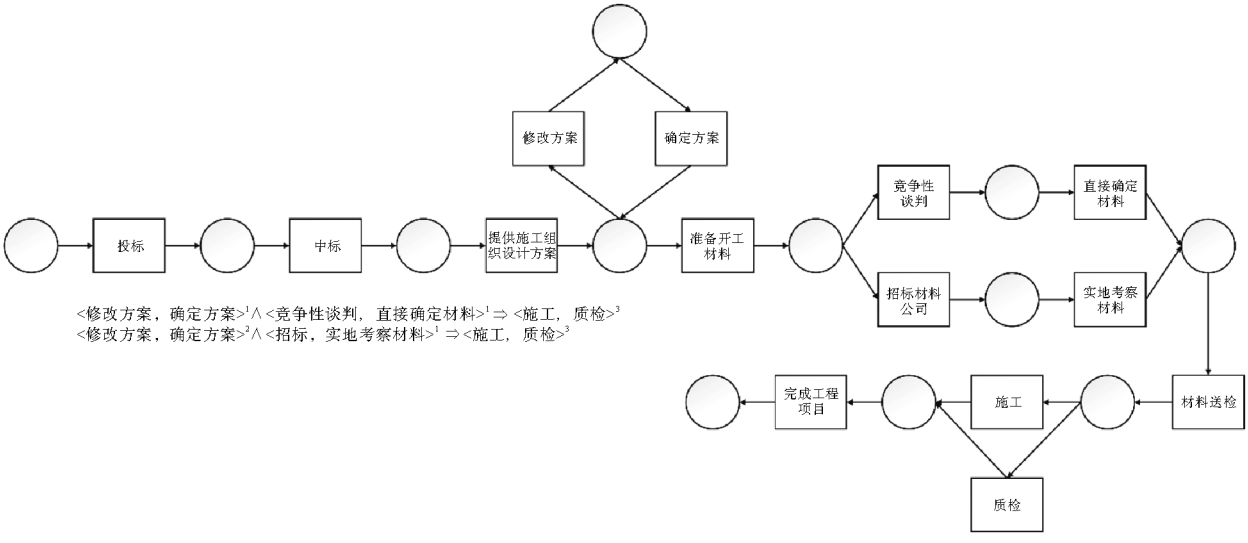

图 5　建筑公司工程项目实施过程模型

Fig. 5　Process model for a construction company to implement an engineering project

建筑公司有两种方案选择材料公司:竞争性谈判或者招标,谈判可直接确定材料公司并确定材料,招标则要

实地考察材料后才能确定材料。确定施工材料后要对材料送检后施工,施工时要定期质检,最后完成工程项目。通过算法 3 可以发现日志中存在间接依赖关系,比如在修改方案、确定方案发生一次后,选择竞争性谈判确定材料公司可以使施工质检阶段执行 3 次等,故可以确定此模型为含循环选择驱动循环结构过程模型。

实际过程可能会出现一些情况:对施工方案进行修改,修改完成后不需要再进行评审、经过竞争性谈判选择的材料公司依旧需要实地考察等。此时原始模型不能满足这些情况,需要修复模型。从某建筑公司系统中获取 10 组事件日志 L_1～L_{10},如表 1 所示(可以通过链接 https://www.aliyundrive.com/s/BWaeSsq1zQv 获取)。本研究基于这 10 组事件日志进行模型修复。

表 1 事件日志信息

Table 1 Information of event logs

事件日志	日志长度	事件数量	变迁数量	迹的长度
L_1	214	2 432	14	9～17
L_2	336	3 643	14	9～17
L_3	427	4 218	14	10～19
L_4	518	5 321	14	10～19
L_5	634	6 437	14	11～21
L_6	753	7 463	14	11～21
L_7	825	8 326	14	12～23
L_8	973	9 756	14	12～23
L_9	1 021	1 036	14	14～25
L_{10}	1 254	1 147	14	14～25

4.2 模型修复实验

通过与 Fahland 方法对比,验证本研究方法的合理性和正确性。图 6 为利用 Fahland 方法修复后的模型,比图 5 增加了 1 个重复变迁、2 个不可见变迁和 6 条流关系。基于逻辑 Petri 网的修复方法如图 7 所示,比图 5 增加了 2 条弧和 2 个逻辑函数,同时根据关联规则添加了新的间接依赖关系。

4.3 模型分析

本节从拟合度、精确度、简洁度三方面对两种修复方法得到的模型进行分析。拟合度用于确保事件日志中行为的发生,模型的拟合度越高,说明模型重复事件日志的能力越强,模型的质量越好;精确度是指过程模型不会发生除事件日志外其他行为的能力,模型的精确度越高,模型生成外部事件日志的迹就越少,模型的质量也越好;简洁度是用简单的模型重演日志中所有迹的能力,简洁度太低会影响模型的可读性。逻辑 Petri 网模型拟合度、精确度的计算方法见文献[18]。

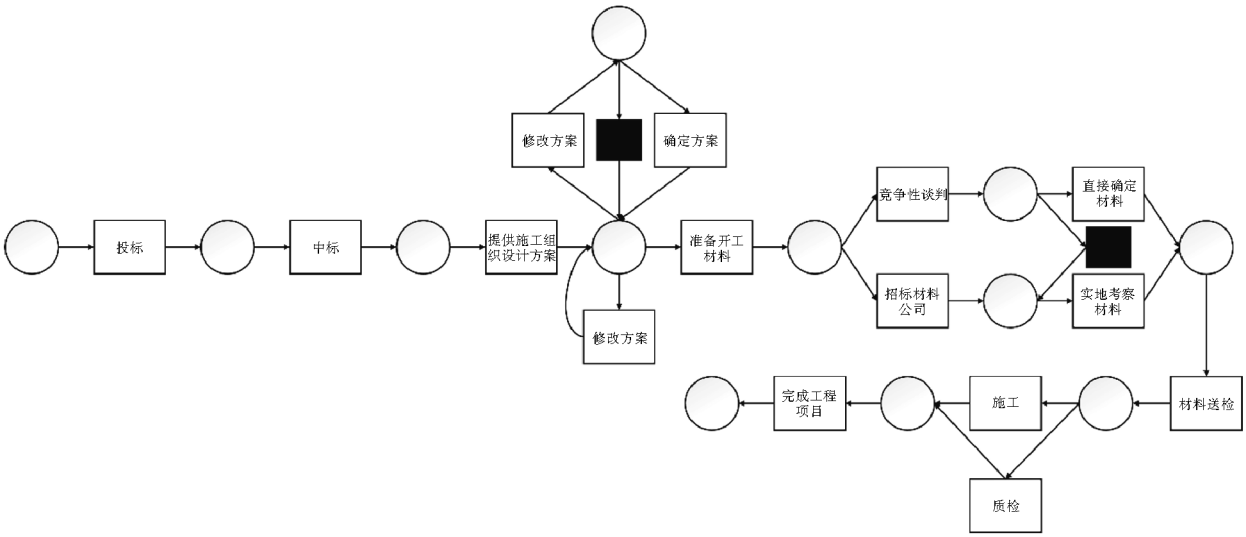

图 6 基于 Fahland 方法修复的模型

Fig. 6 Model repaired by Fahland's method

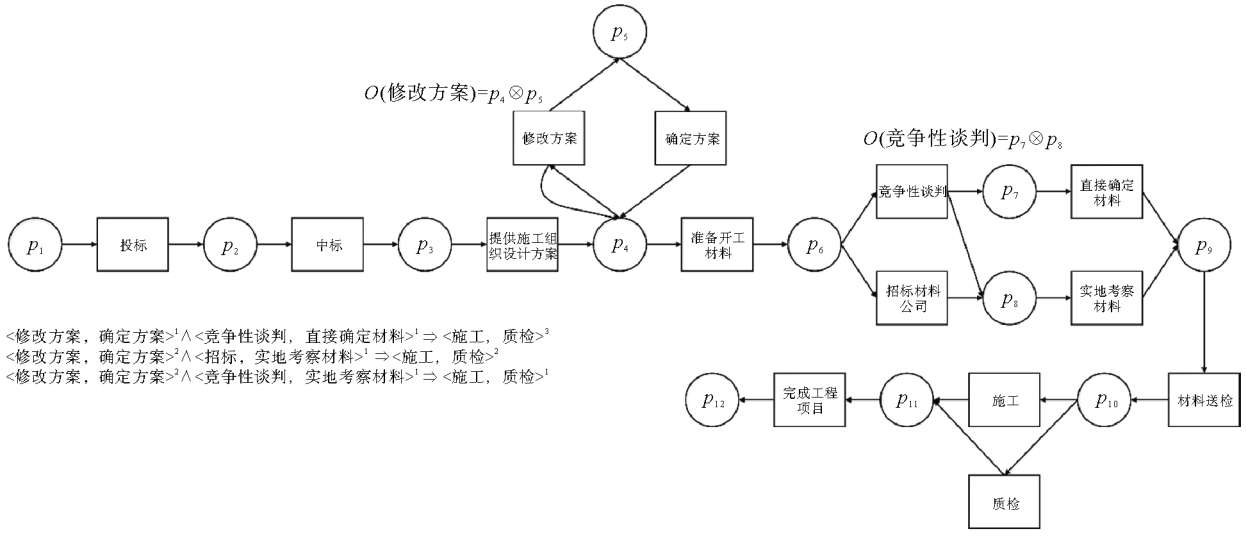

图 7　基于逻辑 Petri 网的过程模型修复方法修复的模型

Fig. 7　Model repaired by process model repair method based on logic Petri nets method

不同日志情况下,两种方法修复的模型拟合度、精确度分别如图 8、图 9 所示。相较于 Fahland 方法,本研究方法修复后模型的拟合度、精确度更高,修复效果更好。Fahland 方法在修复过程模型时增加了自循环,导致过程模型产生许多事件日志之外的迹,使得修复后的模型拟合度、精确度降低。本研究基于逻辑 Petri 网,针对循环选择驱动循环结构,提出修复方法,在修复循环选择驱动循环结构的同时可以表达结构间的间接依赖关系,提高了修复后模型的拟合度和精确度。

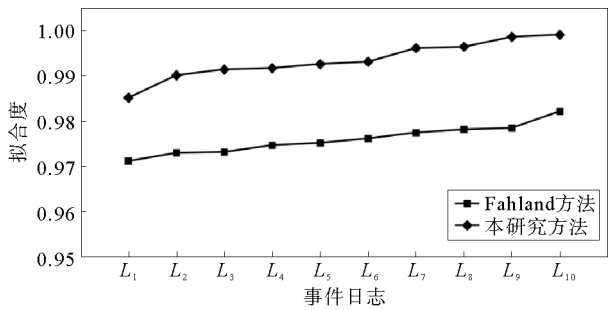

图 8　拟合度变化曲线

Fig. 8　Variation curve of fitness

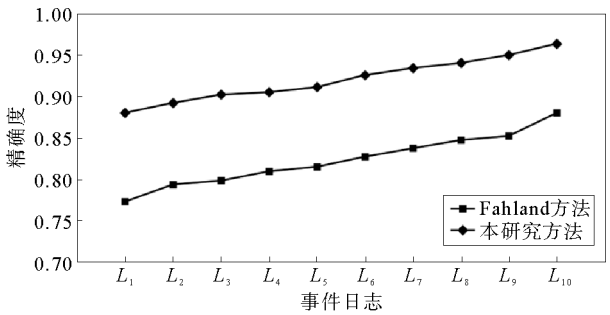

图 9　精确度变化曲线

Fig. 9　Variation curve of precision

从修复模型的库所、变迁、流关系上对两种方法进行简洁度对比,结果如表 2 所示,可以看出,本研究方法得到的模型简洁度更高。与 Fahland 方法相比,本研究方法提高了模型的简洁度,同时也表示了不同结构之间的间接依赖关系。

表 2　模型简洁度比较

Table 2　Comparison of model simplicity

修复方法	库所增加数	变迁增加数	总变迁数	流关系增加数
Fahland	0	3	17	6
本研究方法	0	0	14	2

5　结论

本研究针对循环选择驱动循环结构过程模型提出一种新的模型修复方法,将循环选择驱动循环结构分成三部分分别进行修复。由于不同结构确定偏差的方式不同,提出两个定理判断各个结构是否存在偏差。在使用确定偏差变迁算法得到循环偏差变迁和选择偏差变迁后,在不添加不可见变迁和重复变迁的情况下,使用提出的修复算法进行模型修复。实验表明,本研究方法不仅能够保持原始结构,而且能够正确描述各结

构之间的间接依赖关系。下一步将研究更加复杂的组合结构,进行模型修复并研究之间的间接依赖关系。

参考文献:

[1] AALST W. Process mining:Data science in action[M]. Berlin:Springer,2016:2-53.

[2] LEEMANS S,FAHLAND D,AALST W. Discovering block-structured process models from event logs:A constructive approach[C]//Application and Theory of Petri Nets and Concurrency:Proceedings of 34th International Conference,PETRI NETS 2013. Berlin:Springer-Verlag,2013:311-329.

[3] AALST W. Process mining:Discovery,conformance and enhancement of business process[M]. Berlin:Springer,2011:191-211.

[4] 陈金栋,刘伟,冯新,等. 基于逻辑工作流网的有限无死锁组合[J]. 山东科技大学学报(自然科学版),2020,39(5):89-97.
CHEN Jindong,LIU Wei,FENG Xin,et al. Finite deadlock-free combination based on logical workflow network[J]. Journal of Shandong University of Science and Technology(Natural Science),2020,39(5):89-97.

[5] ADRIANSYAH A,MUNOZ-GAME J,CARMONA J,et al. Aligning based precision checking[C]//Proceedings of the Business Process Management Workshops. Berlin:Springer-Verlag,2013:137-149.

[6] FAHLAND D,AALST W. Model repair-aligning process models to reality[J]. Information Systems,2015,47(1):220-243.

[7] POLYVYANYY A,AALST W,HOFSTEDE A,et al. Impact-driven process model repair[J/OL]. ACM Transactions on Software Engineering and Methodology,2016,25(4):28. DOI:10.1145/2980764.

[8] 杜玉越,孙亚男,刘伟. 基于 Petri 网的模型偏差域识别与模型修正[J]. 计算机研究与发展,2016,53(8):1766-1780.
DU Yuyue,SUN Yanan,LIU Wei. Petri nets based recognition of model deviation domains and model repair[J]. Journal of Computer Research and Development,2016,53(8):1766-1780.

[9] HE Z Y,DU Y Y,QI L,et al. A model repair approach based on Petri nets by constructing free-loop structures[J]. IEEE Access,2019,7(2):24214-24230.

[10] SUN H W,LIU W,QI L,et al. An algorithm for mining indirect dependencies from loop-choice-driven loop structure via Petri nets[J]. IEEE Transactions on Systems,Man,and Cybernetics:Systems,2022,52(9):5411-5423.

[11] WU N Q,ZHOU M C,LI Z C. Short-term scheduling of crude-oil operations:Enhancement of crude-oil operations scheduling using a Petri net-based control-theoretic approach[J]. IEEE Robotics & Automation Magazine,2015,22(2):64-76.

[12] 刘伟,史晓浩,孙红伟. 基于逻辑混合 Petri 网的混合系统建模与分析[J]. 山东科技大学学报(自然科学版),2021,40(4):65-75.
LIU Wei,SHI Xiaohao,SUN Hongwei. Modeling and analysis of hybrid systems based on logic hybrid Petri nets[J]. Journal of Shandong University of Science and Technology(Natural Science),2021,40(4):65-75.

[13] RAN N,SU H Y,WANG S G. An improved approach to test diagnosability of bounded Petri nets[J]. IEEE/CAA Journal of Automatica Sinica,2017,4(2):297-303.

[14] YANG F,WU N,QIAO Y,et al. Polynomial approach to optimal one-wafer cyclic scheduling of treelike hybrid multi-cluster tools via Petri nets[J]. IEEE/CAA Journal of Automatica Sinica,2018,5(1):270-280.

[15] YANG F J,WU N Q,QIAO Y,et al. Scheduling of single-arm cluster tools for an atomic layer deposition process with residency time constraints[J]. IEEE Transactions on Systems Man & Cybernetics Systems,2017,47(3):502-516.

[16] BUIJS M,DONGEN B,AALST W. A genetic algorithm for discovering process trees[C]//IEEE Congress on Evolutionary Computation(CEC 2012). Brisbane:IEEE,2012:1-8.

[17] HU Q,DU Y Y,YU S. Service net algebra based on logic Petri nets[J]. Information Sciences,2014,268:271-289.

[18] ADRIANSYAH A. Aligning observed and modeled behavior[D]. Eindhoven:Eindhoven University of Technology,2014:129-134.

(责任编辑:齐敏华)