计算机研究与发展
Journal of Computer Research and Development
ISSN 1000-1239,CN 11-1777/TP

《计算机研究与发展》网络首发论文

题目：基于可预测适合度的选择性模型修复
作者：张力雯，方贤文，邵叱风，王丽丽
收稿日期：2021-06-03
网络首发日期：2021-12-16
引用格式：张力雯，方贤文，邵叱风，王丽丽．基于可预测适合度的选择性模型修复[J/OL]．计算机研究与发展.
https://kns.cnki.net/kcms/detail/11.1777.TP.20211215.1042.004.html

计 算 机 研 究 与 发 展 DOI:10.75 44/issn1000-1239.20210538
Journal of Computer Research and Development

基于可预测适合度的选择性模型修复

张力雯[1] 方贤文[1] 邵叱风[1] 王丽丽[1,2]

[1]（安徽理工大学数学与大数据学院 安徽淮南 232001）

[2]（嵌入式系统与服务计算教育部重点实验室（同济大学）上海 201804）

（1758678508@qq.com）

Alternative Model Repair Based on the Predictable Fitness

Zhang Liwen[1], Fang Xianwen[1], Shao Chifeng[1], and Wang Lili[1,2]

[1]（*School of Mathematics and Big Data, Anhui University of Science and Technology, Huainan, Anhui 232001*）

[2]（*Key Laboratory of Embedded System and Service Computing (Tongji University), Ministry of Education, Shanghai 201804*）

Abstract The recorded behavior in information system is constantly changing, so the deviation is occurred between the event log and the given process model. Two kinds of deviations are produced by the event log, and the percentage of each deviation is uncertain in the total number of deviations. The iterative deviation generated by self-cycling and the non-iterative deviation in the event log is repaired by same form in existing method. Furthermore, different repair form is alternatively executed under the ideal fitness of 1. Therefore, it is difficult to always ensure the fitness and precision in a reasonable range. To solve this problem, a new repair method was proposed to predict the configured fitness based on the total cost of the iterative observable deviations. Moreover, all the deviations are wholly configured when the predictable fitness meet a given threshold. The variant between the event log and the process model is found by optimal alignment when the predictable fitness does not meet the given threshold. Configuration optimization or form of self-loop insert is used to repair iterative observable deviation in the each variant based on actual condition. Simulation experiment is used to verify different data sets. The results show that the proposed method is beneficial to maximally improve the precision under ensuring reasonable fitness.

Keywords non-iterative deviation; iterative deviation; predictable fitness; variant; configuration optimization; self-loop insert

收稿日期：2021-06-03；**修回日期**：2021-11-11

基金项目：国家自然科学基金项目（61572035，61402011）；安徽省自然科学基金项目（2008085QD178）；安徽省学术和技术带头人项目(2019H239)；安徽省高校优秀人才支持计划项目（gxyqZD2020020）；嵌入式系统与服务计算教育部重点实验开放课题（ESSCKF2018-04）

This work was supported by the National Natural Science Foundation of China (61572035, 61402011), the Natural Science Foundation of Anhui Province (2008085QD178), the Academic and Technical Leader Foundation of Anhui Province (2019H239), the College Excellent Young Talents Fund Project of Anhui Province (gxyqZD2020020), and the Open Project of the Key Laboratory of Embedded System and Service Computing of Ministry of Education (ESSCKF2018-04).

通信作者：方贤文（280060673@qq.com）.

摘要 由于信息系统记录的行为不断变化，因此事件日志与给定模型之间往往存在偏差．事件日志可能产生２种不同类型的偏差，且每种偏差在偏差总数中的占比值是不确定的．已有方法采用固定方式修复日志中非迭代偏差和自循环产生的迭代偏差,或在理想适合度被设定为１的前提下选择执行不同的修复方式，因而很难保证适合度与精度始终在合理范围．针对这一问题，提出１种修复方法根据迭代可观测偏差总成本预测配置优化后的适合度，并在其满足给定阈值的情况下对所有偏差进行整体配置．当预测适合度不满足给定阈值时，进一步通过最优对齐发现事件日志与过程模型之间的变体,并根据每个变体的实际情况使用配置优化或者自循环插入的方式修复可观测偏差.仿真实验中对不同数据集进行了验证,结果表明:在始终保证适合度合理的前提下提出方法能够最大程度地改善精度.

关键词 非迭代偏差；迭代偏差；可预测适合度；变体；配置优化；自循环插入

中图分类号 TP18；TP311.13

过程挖掘技术是从信息系统所记录的事件日志中提取知识发现过程模型,并通过提供技术和工具对其实现性能上的改善．过程挖掘技术主要从３个方面对业务流程进行分析：过程发现、服从性校验、过程完善，其中服从性校验与过程完善之间有着密切的联系[1-3]．模型修复是过程完善的一种，其旨在使事件日志能够回放于过程模型，且修复后的模型与初始模型尽可能相似．通常使用一致性校验的４个度量标准来对模型修复方法的性能进行评估，即适合度、精度、简化度、泛化度[4]．适合度是指事件日志在过程模型上的回放程度，而精度则描述了回放形式的准确程度[5]．能够回放的活动不一定被准确回放，而准确回放的活动一定可以被回放.简化度决定了修复后模型的结构复杂性[6]．泛化度表示修复后模型与初始模型之间的相似性，也就是说过程模型被限制为事件日志中可观测行为的程度．如果仅根据事件日志的可观测行为修复过程模型，则泛化度将受到影响．否则，过度泛化会降低精度造成欠拟合的情况[7]．模型修复主要处理使得事件日志无法在过程模型上正常回放的偏差活动，因此适合度和精度是评估其性能最为重要的２个指标[8-9]．

模型修复通过一致性校验的方式检测事件日志回放于过程模型时产生的偏差,并基于这些偏差信息对过程模型进行修复[10-11]．一致性校验主要包括迹对齐与行为关系匹配２种方法,且最优对齐是迹对齐中最优的一种检测方式[12-13]．行为关系匹配用于发现对应活动之间不同的行为关系，也就是产生偏差的非拟合行为[14]．最优对齐能够检测出事件日志与过程模型之间发生次数最少的偏差，并确定其具体发生位置．通过层层排除的方法构建最优对齐，并根据偏差的不同单位成本进一步分析其最优性，从而获得任意业务流程的最小偏差成本[15]．模型修复通常基于偏差信息执行相应的操作，其中主要包括以下２种类型的偏差[16]：１）事件日志产生的可观测偏差,该偏差的修复形式同时影响适合度与精度;2)过程模型产生的偏差,其会阻碍事件日志正常的回放．这种偏差的修复形式仅对适合度有影响．就平衡适合度与精度而言需要考虑上述第１类偏差的修复问题,现存技术主要对这类偏差采用以下２种修复形

式：1）以自循环的方式将可观测偏差插入过程模型；2）构建偏差与隐变迁的冲突子结构，并将其添加于过程模型的合适位置．前者无法准确回放非迭代的可观测偏差，而后者不能回放迭代的可观测偏差，因此适合度与精度始终无法得到很好的平衡[17]．由于循环路径的重复性，因而使得一些相同的偏差不断的发生．现存方法通常没有单独对循环部分的修复问题进行分析，而是采用与非循环部分一致的修复方法．因此，当事件日志与过程模型之间存在循环路径产生的偏差时，很容易在改善适合度的同时导致精度成倍的下降．理想的修复性能是在合理程度上保证适合度的同时尽可能提高精度，且不增加修复活动的数量（拥有相同标签和位置的所有偏差被视为一个修复活动．值得注意的是，为了准确计算适和度与精度，每一个需要被修复的偏差活动都被看作一个独立的单位成本进行核算(其中包括偏差的迭代发生)[18]．

图 1 使用 BPMN(business process model and notation)描述了一个投保流程的模型 M，其在信息系统中被记录为 $L=\{(ABEIIIJK),(ABFJIIIK),(ACDGGGHIJK),(ADCGHIJK)\}$．方便起见，统一使用 A 到 K 之间的单词表示 L 与 M 中所有任务的对应标签．通过最优对齐可搜索到 L 与 M 之间非循环部分存在的 3 种偏差：1）L 中的非迭代可观测偏差 {D,G,I,J}；2）M 上能够捕获且使得 L 无法正常回放的偏差 {D}；3）L 中的迭代可观测偏差 $\{I^2,G^2\}$（偏差右上角的数字表示事件日志中偏差自身迭代发生的次数）．将偏差活动的单位成本设置为“1”，可计算出 L 与 M 之间的适合度和精度分别为 $fitness=1-Dev_{cost}/(|L|+|M_L|)\approx 0.67$，$precision=1$（$Dev_{cost}$ 为偏差成本，$|L|+|M_L|$ 为日志与模型在对齐过程中所有元素的个数）．初始模型上没有自循环路径产生的迭代活动时，核算精度需要将日志预处理为与模型完全拟合的子日志，因而在这种情况下精度始终为“1”[19]．上述第 2 类偏差来自于 M，因此其修复形式不影响精度．通过在此类偏差元素的原有位置添加隐变迁进行修复，可以使这部分的适合度得到完整提升．上述第 1 类和第 3 类偏差均来自于 L，以自循环方式将其插入 M 进行修复，则第 1 类偏差无法被准确回放．修复后 L 与 M 之间的适合度为 1，但精度却下降了 0.32．将 L 中的偏差元素与隐变迁组建为一个冲突子结构，并添加到过程模型上的合适位置，该方法无法回放第 3 类偏差．修复后 L 与 M 之间的精度为 1，但适合度却只有 0.88．这种情况下，要么以大幅度牺牲精度为代价来保证适合度最终为 1，要么精度为 1 但适合度则不够合理．

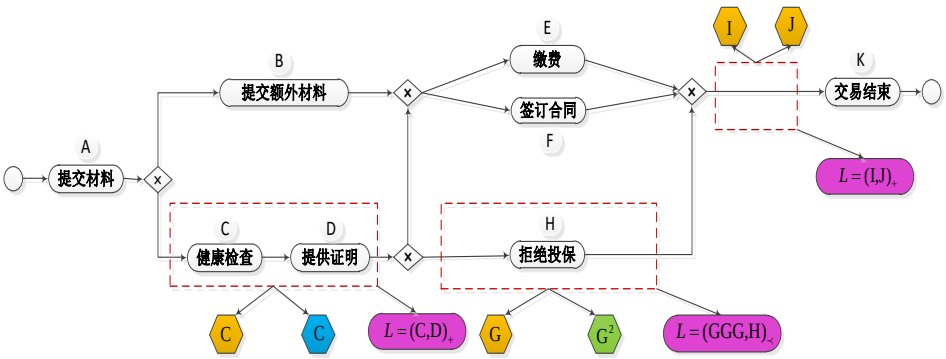


Fig. 1 Life insurance application process and deviations bsaed on information system

图 1 人生保险申请过程和基于信息系统产生的偏差

本文针对上述问题主要进行了以下 2 方面的改进：1）根据迭代可观测偏差的总成本预测使用配置方法进行整体优化后的适合度，并当其满足给定阈值范围时确定对所有可观测偏差进行配置优化．2）当预测适合度不满足阈值时，利用行为关系的匹配性发现事件日志与过程模型之间存在的变体，并通过检测每个变体中是否包含迭代可观测偏差来执行合适的操作．由于低频过滤不在本文的研究范围之内，因此文中默认事件日志是已经过滤后的有效事件序列集．

1 相关工作

本文所提出的方法主要涉及模型修复与配置优化 2 种技术[20-21]．模型修复的目标是使事件日志能够在修复后的过程模型上回放，并尽可能地维护原始模型的特有行为．采用伪布尔约束来处理多目标问题，该方法以最小代价实现了模型的最大回放[22]．与自动操作相比，手动修复可以尽可能地避免过程模型中不必要的冗余行为[23]．上述 2 种方法在合理提高适合度的同时尽可能降低修复成本，但其忽视了一致性度量中的其他重要标准，即精度．将定义的扩展对齐添加到过程模型的可达标记中，计算模型中的偏差并对其进行修复[24]．通过 Petri 网和各选择结构之间的转换关系确定模型的修复位置[25]．上述 2 种方法综合考虑了各种度量标准的改进，但仅适用于选择结构的网系统．将多个经典问题的解决方案用来搜索事件日志与过程模型之间所需的修复活动，旨在以最小的代价处理最多的偏差[26]．这种方法需要牺牲大量精度来换取适合度的提升．基于 4 个度量标准分析过程模型的修复方法，并将满足并发关系的可观测偏差子结构以自循环方式插入到过程模型上，其在一定程度上改善了修复后的精度，但仍然无法对非迭代插入偏差进行准确修复．在保证适合度为 1 的前提下根据当前变体的实际情况选择执行不同的操作，其往往由于过分追求适合度而导致精度仍然可能存在不合理的现象[27]．

配置优化在给定约束条件下发现一个业务流程中具有共性的参考模型，并将其与变化模型之间产生的变量进行兼容[28-29]．这种技术通过在控制流中添加/移除可配置活动的选择子结构来实现优化[30-31]．基于事件日志所记录的真实行为构建可配置片段，并通过对其进行优化获得一个具有共性和个性行为的过程模型[32]．配置优化技术构建日志中可观测偏差与隐变迁的冲突子结构，并通过将其添加到过程模型的合适位置实现修复[33]．使用这种方法不能回放日志中自循环路径产生的迭代可观测偏差，因而无法保证修复后的适合度在合理范围内．本文首先对整体配置后的适合度进行预测，并根据阈值条件选择是否对初始模型执行整体配置．当不能进行整体配置时，需要发现事件日志与过程模型之间的变体，并根据每个变体中是否包含迭代可观测偏差选择执行不同的操作．循环部分的修复要根据循环中产生偏差的不同情况来执行，从而最大程度的兼顾适合度与精度．

2 准备工作

定义 1. 事件日志. 将元组 $L=(\kappa,q,e,\varpi,\ell,\angle)$ 记为事件日志. κ 为日志中的一条迹，q 为多重迹的统一案例号. 由于事件日志中可能存在不同的多重发生迹，因此其中所有案例号都属于一个案例集 q^*，即 $\forall q=q^*$. e 为日志中的事件元素，记作 $\forall e_i\in L$. ϖ 为事件日志中所有事件的标签集，且 ℓ 将每个事件指定为其对应的标签，$\sum_{i=1}^{|L|}\ell(e_i)=\varpi$. $\angle$ 为相邻事件之间的流关系，记作 $\angle\subseteq E\times E$.

定义 2. 标签工作流网系统. 将元组 $N^*=(P,T,F,Z,\lambda,\delta,D,p_{\text{ini}},p_{\text{final}})$ 定义为标签工作流网系统. P，T，F 分别表示网系统 N^* 上的库所、变迁以它们之间的流关系，$F\subseteq(P\prec F)\cup(F\prec P)$. Z 为网系统 N^* 中所有变迁的标签集，且 λ 将每个变迁指定到对应的标签，$\sum_{j=1}^{|T|}\lambda(t_j)=Z$. δ 为网系统上从初始结点到终止结点之间的一条完整序列，d 为每条序列 δ 的案例号，记作 δ_d. 由于网系统 N^* 中可能存在不同的序列，因此其中所有发生序列都属于一个序列集 δ_d^*，即 $\delta_d^*=\sum_{d=1}^{D}\delta_d$. $p_{\text{ini}}\in P$ 与 $p_{\text{final}}\in P$ 分别为工作流网的初始库所与终止库所，当且仅当 $|p_{\text{ini}}|=|p_{\text{final}}|=1$ 时网系统 N^* 为工作流网.

定义 3. 最优对齐. 对齐是指事件日志的一条迹与过程模型上任意发生序列的比较，而事件日志与过程模型之间的对齐集则为 $\xi^*=\sum_{c=1}^{|C|}\sum_{d=1}^{|D|}(\kappa_c,\delta_d)$. 最优对齐则是寻找与每条迹之间满足最小偏差成本的发生序列，即 $\xi_{\text{op}}^*=\sum_{c=1}^{|C|}\sum_{d=1}^{|D|}(\kappa_c,\delta_d)_{\text{op}}$，$(Dev_{\text{cost}}(\kappa_c,\delta_d)\le Dev_{\text{cost}}(\kappa_c,\delta_d'))\wedge\exists(\kappa_c,\delta_d')\in\xi_{\kappa_c}^*$（$Dev_{\text{cost}}$ 是对齐中偏差成本的函数）. 最优对齐所检测到的偏差可分为以下 2 种类型：1）插入偏差，仅在事件日志上可观测的活动，记作 $Esert(\ell(e))$，$\forall e\in L$；2）跳过偏差，仅在过程模型中存在的且可阻碍事件日志正常回放的活动，记作 $Skip(\lambda(t))$，$\forall t\in T$. 将所有偏差的单位成本设定为“1”.

定义 4. 拟合模式与非拟合变体. 非拟合变体是指引起事件日志与过程模型之间部分偏差的对应子模式[34]. 在最优对齐中划分由不同行为变化所产生的偏差，并根据每个行为变化中的活动在所有对齐中的行为关系确定变体类型.

例如，$L=\{(\text{ABCDEFGHK}),(\text{ACBDEFGHK})\}$ 被回放于过程模型上，该过程模型的发生序列集为 $\delta^*=\{(\text{ABCDEIJK}),(\text{ABCEDJIK}),(\text{ABCDEIJK}),(\text{ABEDJIK})\}$. 图 2 中找出 3 组偏差 (B,C)，(F,G,H)，(I,J) 分别属于并发/因果、插入因果行为包含及跳过因果行为包含的变体，记作 $v(\text{C,D})_{L+\wedge N^*\prec}$，$v(\text{F,G,H})_{L\subseteq\prec}$，$v(\text{I,J})_{N^*\subseteq\prec}$. 值得注意的是，事件日志或网系统 N^* 中单个出现的偏差可被看作一个独立的变体，记作 $v_{L||N^*}(\ell(e_i)||\lambda(t_i))$. 图 2 中 $B_{\text{fit}(\prec)}=(\text{D,E})$ 和 $B_{\text{fit}(a)}=\{\text{A,K}\}$ 表示事件日志与过程模型之间拟合的行为模式，其中 $B_{\text{fit}(\prec)}=(\text{D,E})$ 为拟合的因果行为模式，而 $B_{\text{fit}(a)}=(\text{A,K})$ 则是单个活动的拟合行为模式.

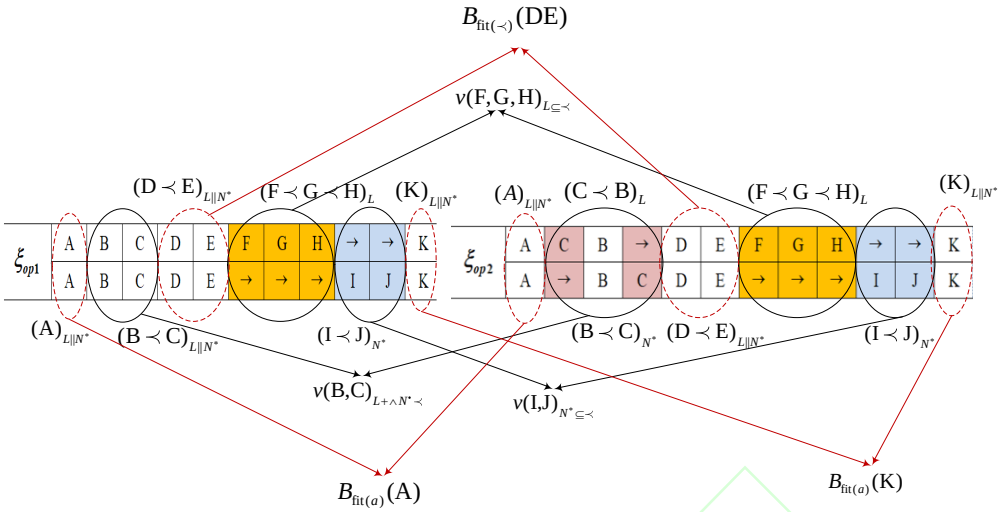
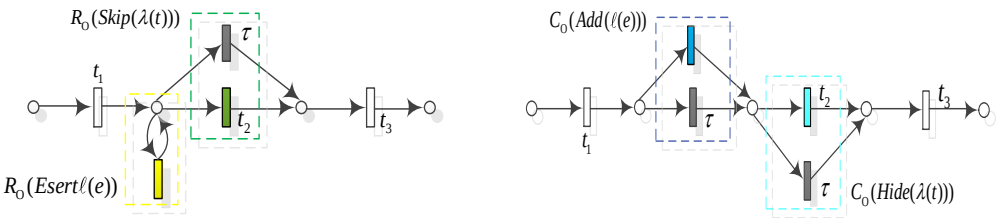


Fig.2 Search for variants based on optimal alignment

图 2 基于最优对齐搜索变体

定义 5. 迭代插入偏差. 迭代插入偏差是由事件日志中自循环所产生的特殊偏差，记作 $\lambda=(\chi, IEsert\ell(e), IEEsert\ell(e), n)$，其中 χ 为所有迭代插入偏差的活动标签集. 迭代插入偏差主要包括 2 种类型：1）仅由自循环所产生的插入偏差，即 $IEsert\ell(e)$；2）自循环产生的插入偏差与单次发生的插入偏差相结合，即 $IEEsert\ell(e)$. n 是迭代插入偏差的重复发生次数，分别标注为 $IEsert\ell(e)^{n-1}$，$IEEsert\ell(e)^n$. 由于 2 种迭代插入偏差的含义不同，因此 $IEsert$ 的重复次数从第 2 次发生开始计数，而 $IEEsert$ 则需要记录它的完整发生次数.

定义 6. 修复操作与配置操作. 修复操作利用最优对齐检测偏差的完整信息，并采用自循环插入日志中的偏差 / 隐变迁跳过模型中的偏差对网 N^* 进行修复. 根据偏差的不同结构类型，可将修复操作分为以下 2 种：1）单个活动的修复操作 $R_{\mathrm{O}}=sl\_Esert\ell(e_i)$ 或者 $R_{\mathrm{O}}=Skip\lambda(t_j)$；2）子结构的修复操作 $R_{\mathrm{O}^*}=sl\_E(\ell(e_1),...,\ell(e_i))_{+,\prec,\times}$ 或者 $R_{\mathrm{O}^*}=Skip(\lambda(t_1),...,\lambda(t_j))_{+,\prec,\times}$. 图 3（a）中描述了修复操作的实例. 配置优化通常是对业务流程之间所有不匹配的行为进行优化，而本文则只考虑回放过程中非拟合行为变体所产生的差异. 配置操作是在模型上添加日志中不可回放的行为或者通过隐变迁隐藏模型中阻碍日志回放的行为，并将配置操作执行的所有行为称作可配置行为[35]. 根据可配置行为的不同结构类型将配置操作分为以下 2 种：1）单个活动的配置操作是 $C_{\mathrm{O}}=Add(\ell(e_i))=(\ell(e_i),\tau)_\times$ 或者 $C_{\mathrm{O}}=Hide(\lambda(t_j))=(\tau,\lambda(t_j))_\times$；2）子结构的配置操作 $C_{\mathrm{O}^*}=N_{Add(\ell(e_1),...,\ell(e_i))\prec,+,\times}=((\ell(e_1),...,\ell(e_i))_{\prec,+,\times},\tau)_\times$ 或者 $C_{\mathrm{O}^*}=N_{Hide(\lambda(t_1),...,\lambda(t_j))\prec,+,\times}=(\tau,(\lambda(t_1),...,\lambda(t_j))_{\prec,+,\times})_\times$. 图 3（b）中描述了配置操作的实例.



(a) 修复操作 (b) 配置操作

Fig.3 Examples of repair and configuration operations

图 3 修复操作与配置操作的实例

3 操作确定及故障诊断

事件日志与过程模型在实际运行中可能会存在一些不被期望的非一致行为，其中模型修复主要对日志回放于模型所产生的非拟合行为进行处理. 本节根据可预测适合度是否在给定合理范围内来确定不同的修复方案，并分别从 2 种角度对一个变体中的故障进行检测与分析.

3.1 确定不同情况下的修复方式

为了尽可能兼顾事件日志与修复后模型之间的适合度与精度，可设定适合度的合理范围，并在此范围内最大化提升精度. 使用配置操作对初始模型进行整体修复后的新模型不包含迭代活动，且任意事件日志与该模型之间的精度值始终为 1. 然而，这种情况下所有迭代插入偏差不能被回放，以至于适合度可能会受到不同程度的影响. 因此，当使用配置操作整体修复后的可预测适合度值低于合理范围时，需要发现事件日志与过程模型之间的变体，并根据每个变体的具体情况选择执行配置操作和修复操作[36].

可预测适合度是指使用配置操作对初始模型进行整体修复后的适合度，其中所有迭代插入偏差无法被修复. 根据迭代插入偏差总成本核算可预测适合度的公式为 $per\_fitness = 1 - IDev_{cost} / (|L| + |M_L|)$（$IDev_{cost}$ 为迭代插入偏差的成本函数）. 当 $pre\_fitness \geq 0.9$ 时，使用配置操作对初始模型进行整体修复（给定阈值被设置为 0.9，且 $0.9 < pre\_fitness \leq 1$ 被看作是可预测适合度的合理范围）. 当 $pre\_fitness < 0.9$ 时需要识别事件日志与过程模型之间存在的变体，并根据每个变体中是否包含迭代插入偏差来选择修复操作或配置操作. 图 4 中描述了 2 种情况下的修复方案，即使用配置操作进行整体修复或在不同变体中选择执行配置操作与修复操作.

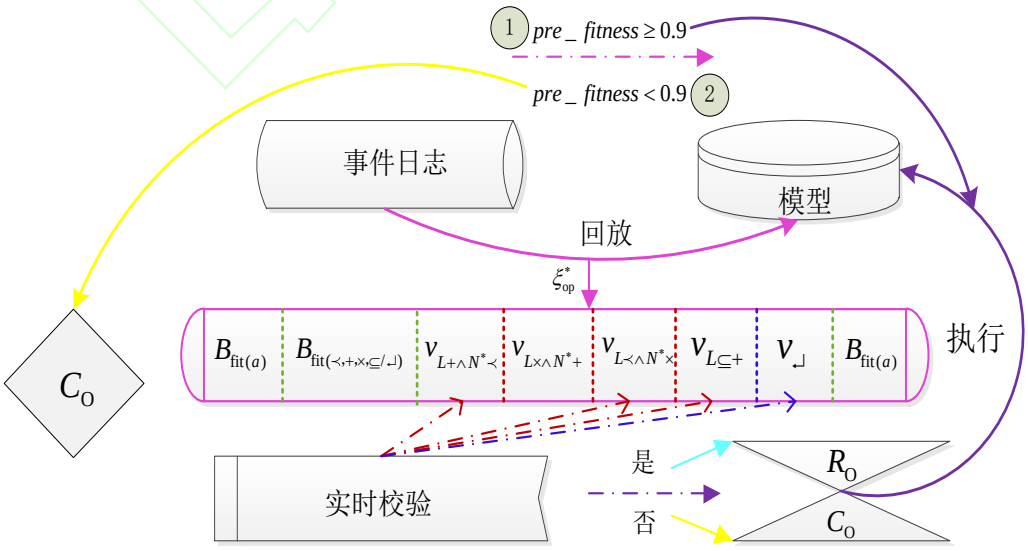
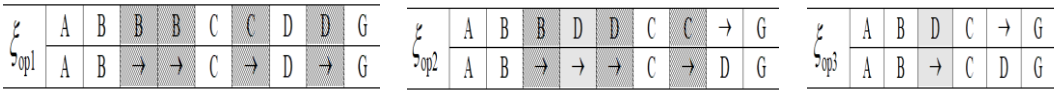


Fig.4 The alternative principle for different operations

图 4 不同操作的选择原理

3.2 检测偏差

修复操作依赖于最优对齐检测到的偏差元素来优化过程模型，其中迭代插入偏差是一种特殊的偏差类型，即只有当日志记录的单个偏差元素经过自循环路径时才会发生[37]. 设定事件日志 L 回放于初始网系统 N^*={(ABCDG),(AEFG)}时产生了图 5 中的 3 组最优对齐.



(a) 最优对齐 1　　(b) 最优对齐 2　　(c) 最优对齐 3

Fig.5 Three optimal alignments

图 5　　3 组最优对齐

根据图 5 中的 3 组对齐可获得一个变体 $v(\mathrm{C,D})_{L+\wedge N^*\prec}$，并且还有 6 个完全由自循环路径产生的迭代插入偏差，即 ξ_{op1} 中的 $IEsert(\mathrm{B})^2$， $IEsert(\mathrm{C})^1$， $IEsert(\mathrm{D})^1$ 和 ξ_{op2} 中的 $IEsert(\mathrm{B})^1$， $IEsert(\mathrm{D})^1$，$IEsert(\mathrm{C})^1$（在 3 组对齐中使用斜线阴影填充表示这类偏差）. 由于它们在正常路径下不能发生，因此即使只发生 1 次也会被看作迭代插入偏差处理. ξ_{op2} 中的插入偏差 $IEEsert(\mathrm{D})^2$ 既产生于并发/因果的变体也产生于自循环路径，且 ξ_{op3} 中的非迭代插入偏差 $Esert(\mathrm{D})$ 包含于 ξ_{op2} 中的迭代偏差 $IEEsert(\mathrm{D})^2$. 值得注意的是， ξ_{op2} 中用纯色阴影表示来自于变体 $v(\mathrm{C,D})_{L+\wedge N^*\prec}$ 中的插入偏差，其可与 $IEsert(\mathrm{D})^1$ 合并在一起，并记作 $IEEsert(\mathrm{D})^2$（根据定义 5 可知， $IEEsert(\mathrm{D})^2$ 中 D 的迭代次数为 1）.

3.3 识别可配置行为

配置是一种将参考模型与定制模型之间不匹配行为相互兼容的优化技术. 在发现可配置行为的过程中，将事件日志看作是不含有迭代插入偏差的一个参考过程模型，而给定过程模型则是一个变化模型[38]. 为了能够更好的兼顾适合度与精度，在一定条件下需要基于可配置行为在初始模型上执行配置操作，从而完成模型修复的功能.

设定事件日志与过程模型之间的变体分别为 $\{v_L(a_d), v(a_g,a_h)_{L\times\wedge N^*+}, v(a_i,a_j)_{L\subseteq+}\}$， 其中 $v_L(a_d)$ 为仅日志中可观测到的单个活动变体，因而被看作一个可配置活动 $Add(a_d)$. $v(a_g,a_h)_{L\times\wedge N^*+}$ 可展开为 $(a_g \prec Hide(a_h)) \| (Hide(a_g) \prec a_h)$，其中选择发生的可配置活动之间满足冲突行为关系，将它们构建为可配置的冲突子结构 $N_{Hide(a_g,a_h)\times}$. $v(a_i,a_j)_{L\subseteq+}$ 展开为 $(Add(a_i) \prec Add(a_j)) \| (Add(a_j) \prec Add(a_i))$，其中直接跟随的可配置活动之间满足并发行为关系，将它们构建为可配置的并发子结构 $N_{Add(a_i,a_j)+}$. 配置操作根据定义 6 描述的方式在初始模型上对可配置活动/子结构进行修复. 值得注意的是，当可预测适合度在合理范围时，无需发现事件日志与过程模型之间的变体，而是直接使用配置操作对初始模型进行整体修复. 在这种情况下，将最优对齐检测出的迭代插入偏差移除，并使其余偏差转换为相对应的可配置行为.

4 基于可预测适合度的选择性修复

适合度与精度是衡量业务流程一致性的 2 个重要依据，因此关于事件日志对过程模型进行修复时需要同时兼顾适合度与精度. 修复操作与配置操作在修复中分别对适合度和精度进行优先考虑，使用其中一种操作对初始模型进行整体修复，往往会造成回放性能的改善过于局限. 综上所述，为了能够合理的权衡适合度与精度，需要在修复初始模型时根据具体情况选择不同的操作.

4.1 非循环部分的选择性模型修复

事件日志与过程模型之间的可预测适合度在阈值范围之内时，需采用配置操作对过程模型进行整体修复. 当可预测适合度不在阈值范围内时，则根据每个变体中是否包含迭代插入偏差选择修复操作或配置操作. 本节介绍各种非循环部分产生变体的修复方式，也就是修复操作与配置操作的选择性执行. 一个包含迭代插入偏差的变体中也可能存在非迭代插入偏差，因此需要执行修复操作. 修复操作能够使这种变体中的所有插入偏差回放于过程模型上，但其中非迭代插入偏差不能被准确回放. 另一方面，在不包含迭代插入偏差的变体中执行配置操作，可以使其中所有插入偏差被准确回放. 图 6 首先判定可预测适合度不在合理范围内，并给定事件日志与过程模型之间存在的 3 个变体 $v(a_i,a_{i+1})_{L\prec\wedge N^*\times}$，$v(a_{i+2},a_{i+3})_{L\times\wedge N^*+}$，$v(a_{i+4},a_{i+5})_{L\subseteq\prec}$. $v(a_i,a_{i+1})_{L\prec\wedge N^*\times}$ 中包含插入偏差但没有迭代插入偏差，使用配置操作可完整提升该变体的适合度和精度. $v(a_{i+2},a_{i+3})_{L\times\wedge N^*+}$ 中不包含插入偏差，因此使用修复操作或配置操作都能够完整提升它的适合度，且精度不受影响. $v(a_{i+4},a_{i+5})_{L\subseteq\prec}$ 中包含迭代插入偏差，使用修复操作可完整提升其适合度，但精度会由于非迭代插入偏差无法被准确回放而受到影响[39].

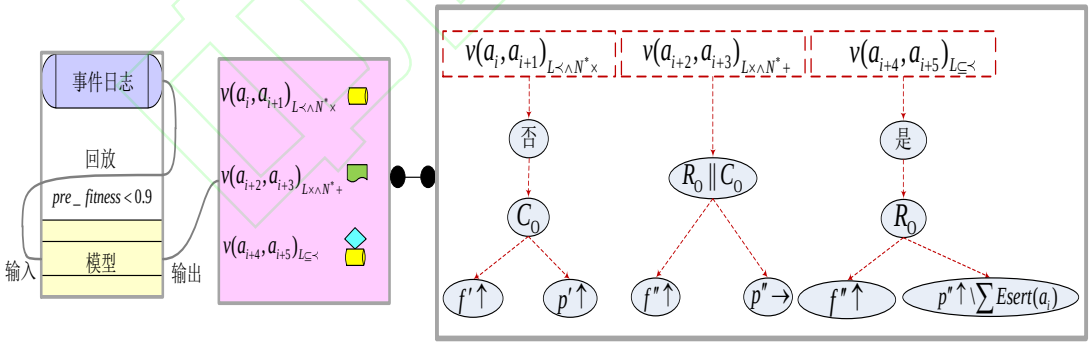


Fig.6 Execution program and effect of real-time repair

图 6 实时修复的执行方案及效果

设定事件日志为 $L=\{(\mathrm{ABCDHIJ})^{7},(\mathrm{ABDDDCIHJ})^{21},(\mathrm{AEFFFGIHJ})^{9},(\mathrm{AFEGHIJ})^{2}\}$，将其回放于图 7 描述的初始网系统 N^*. 若每个偏差的单位成本为 1，则可预测适合度为 $1-\frac{60}{517}=0.88<0.9$. 因此，需要根据每个变体中是否包含迭代插入偏差来选择合适的操作进行修复.

如图 7 所示，L 与 N^* 之间的变体分别为 $v(\mathrm{C,D})_{L+\wedge N^*\prec}$，$v(\mathrm{E,F})_{L+\wedge N^*\times}$，$v(\mathrm{H,I})_{L\subseteq+}$，它们可选择的

修复方式为以下 2 种：1）弯虚线部分表示当前变体中含有迭代插入偏差时所做的修复操作；2）直虚线部分描述了当前变体中不包含迭代插入偏差时所做的配置操作（空心箭头连接的部分表示修复操作与配置操作的共有部分）．值得注意的是，当 $pre\_fitness \geq 0.9$ 时使用直虚线描述的配置操作对过程模型进行整体修复．

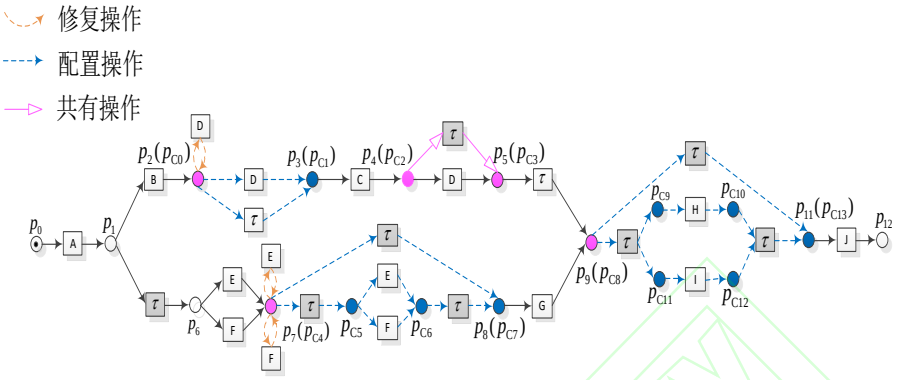


Fig.7 Repair instance of the acyclic part based on an optional operation

图 7 非循环部分基于可选操作的修复实例

根据 3.1 节的方法检测出变体 $v(\mathrm{C,D})_{L+\wedge N^{*}\prec}$ 和 $v(\mathrm{E,F})_{L+\wedge N^{*}\times}$ 中含有迭代插入偏差，因此选择执行修复操作．由于变体 $v(\mathrm{H,I})_{L\subseteq+}$ 中没有迭代插入偏差，则选择执行配置操作．$v(\mathrm{C,D})_{L+\wedge N^{*}\prec}$ 和 $v(\mathrm{E,F})_{L+\wedge N^{*}\times}$ 中包含的偏差分别为 $(IEEsert(\mathrm{D})^{3(21)}, Skip(\mathrm{D})^{1(21)})$ 和 $(IEEsert(\mathrm{F})^{3(9)}, Esert(\mathrm{E})^{1(2)})$（其中 $n-1$ 表示迭代次数，(m) 表示迹发生的多重次数，一个偏差元素的成本记作 $n \times m$）．$v(\mathrm{H,I})_{L\subseteq+}$ 中包含的偏差为 $(Esert(\mathrm{H})^{1(7+21+9+2)}, Esert(\mathrm{I})^{1(7+21+9+2)})$．在 $v(\mathrm{C,D})_{L+\wedge N^{*}\prec}$ 和 $v(\mathrm{E,F})_{L+\wedge N^{*}\times}$ 上执行修复操作后能够修复的偏差成本为 84 和 29，而准确修复的插入偏差成本则为 42 和 18（也就是说非迭代插入偏差 $Esert(\mathrm{D})^{1(21)}$，$Esert(\mathrm{F})^{1(9)}$，$Esert(\mathrm{E})^{1(2)}$ 无法被准确修复，且跳过偏差 $Skip(\mathrm{D})^{1(21)}$ 的修复不影响精度，因而不被计入需要准确修复的偏差中）．在 $v(\mathrm{H,I})_{L\subseteq+}$ 上执行配置操作后能够修复的偏差成本为 78，而能够准确修复的插入偏差成本也为 78．综上所述，根据不同情况选择执行 2 种操作后的适合度与精度分别为 $fitness = 1-\frac{191-191}{517}=1$ 和 $precision = \frac{341-170+14+84+36+4}{341} \approx 0.91$．由于并发行为中活动发生顺序的随机性会使配置回放图表的结构较为复杂，因此当 $v_{L+\wedge N^{*}\prec}$ 和 $v_{L+\wedge N^{*}\times}$ 中的活动个数大于 2 时，可分别根据图 8 和图 9 中描述的方式寻找配置操作 C_{O}，并确定其具体位置．图 8 和图 9 中使用虚线矩形方框来标注 $v_{L+\wedge N^{*}\prec}$ 和 $v_{L+\wedge N^{*}\times}$ 所对应的配置操作 C_{O}．

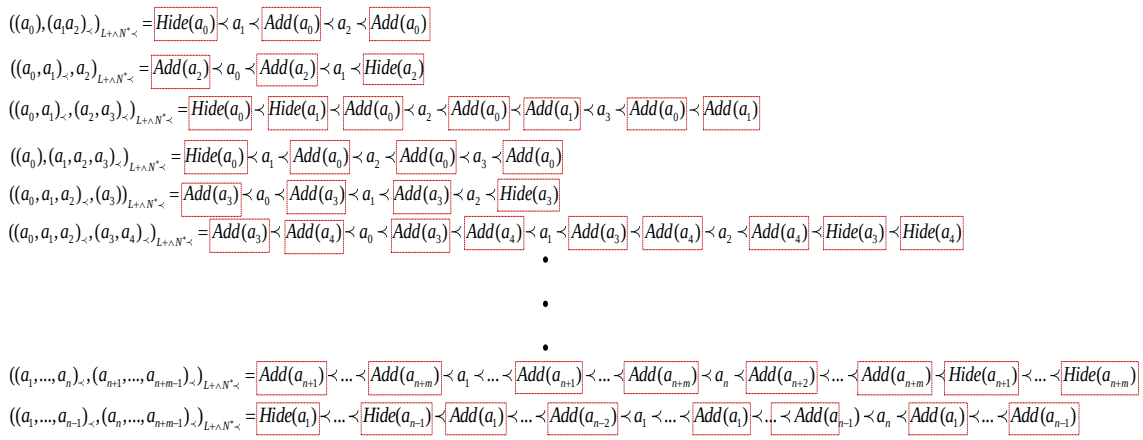


Fig.8 Search for configuration operation in concurrency/causality variant

图 8 搜索并发/因果变体中的配置操作

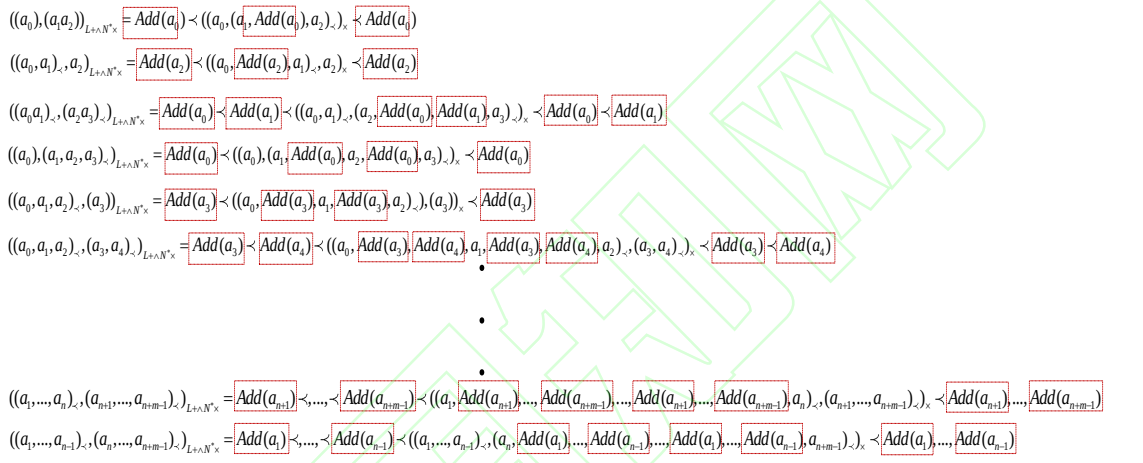


Fig.9 Search for configuration operation in concurrency/conflict variant

图 9 搜索并发/冲突变体中的配置操作

非循环部分的变体主要由非一致的行为轮廓关系和行为包含产生，算法 1 描述了非循环部分产生变体的配置操作与修复操作. 将空集赋予配置操作与修复操作的集合 C_{O}，R_{O}（第②行）. 首先判定可预测适合度是否在合理范围内，如果在合理范围内则统一使用配置操作修复所有偏差（⑤～⑦行）. 当可预测适合度不在合理范围内时，根据以下不同变体中是否包含迭代插入偏差选择合适的操作：1）构建并发和因果行为关系产生变体的配置操作（⑧～⑪行）或修复操作（第⑫行）；2）构建并发和冲突行为关系产生变体的配置操作（⑮～⑱行）或修复操作（第⑲行）；3）构建因果和冲突行为关系产生变体的配置操作（㉒～㉕行）或修复操作（第㉖行）；4）构建 3 种行为包含产生变体的配置操作（㉙～㉜行）或修复操作（第㉝行）. 由于迭代插入偏差的发生是随机的，因此所有自循环插入偏差的情况都被关联于修复操作集中，即 $R_{\mathrm{O}}(V_{(L+\wedge N^{*}\prec),\ (L+\wedge N^{*}\times),\ (L\prec\wedge N^{*}\times),\ (L\subseteq\prec,+,\times)}) \leftarrow \chi$. 不包含插入偏差变体的修复操作与配置操作可互换，即 $R_{\mathrm{O}}(V_{(L\times\wedge N^{*}+),\ (L\times\wedge N^{*}\prec),\ (N^{*}\subseteq\prec,+,\times)}) \leftarrow C_{\mathrm{O}}(V_{(L\times\wedge N^{*}+),\ (L\times\wedge N^{*}\prec),\ (N^{*}\subseteq\prec,+,\times)})$. 最后，返回可能在因果 / 并发 / 冲突 / 行为包含引起变体中执行的所有配置操作与修复操作集 C_{O}， R_{O}.

算法 1. 非循环部分产生变体的选择性修复.

输入：网系统 N^*，事件日志 L，所有偏差之和 $\sum D$，迭代插入偏差的成本判定函数 ψ，选择任意一个或几个元素的函数 σ，配置操作 c_{O}，修复操作 r_{O}；

输出：配置操作集 C_{O}，修复操作集 R_{O}.

① $C_{\mathrm{O}} \leftarrow \varnothing$， $R_{\mathrm{O}} \leftarrow \varnothing$；

② for each $e_i \in L$ do

③ for each $t_j \in T$ do

④ for each $a_k \in Z$ do

⑤ if $pre\_fitness = 1 - IDev_{\mathrm{cost}} / |L| + |M_L| \geq 0.9$ then

⑥ $C_{\mathrm{O}}(\sum D)$；

⑦ else

⑧ if $\ell(e_i) = \lambda(t_j) = a_k \wedge v_{L+\wedge N^*\prec}((a_1, a_2) \| ((a_1, ..., a_n)_{\prec}, (a_{n+1}, ..., a_{n+m})_{\prec}))$ then

⑨ if $\psi(IDev_{\mathrm{cost}}) = 0$ then

⑩ $c_{\mathrm{O1}} \leftarrow (Add(a_2) \cup Hide(a_2)) \| \lambda(v_{L+\wedge N^*\prec})$；

⑪ else

⑫ $r_{\mathrm{O1}} \leftarrow (skip(a_2) \cup sl\_Esert(a_2)) \| (N_{Skip \cup sl-Esert(a_{n+1}, ..., a_{n+m})_{\times}} \cup sl\_Esert(\sigma(a_1, ..., a_{n+m})))$；

⑬ end if

⑭ end if

⑮ if $\ell(e_i) = \lambda(t_j) = a_k \wedge v_{L+\wedge N^*\times}((a_1, a_2) \| ((a_1, ..., a_n)_{\prec}, (a_{n+1}, ..., a_{n+m})_{\prec})) \| v_{L\times\wedge N^*+}(a_1, a_2)$ then

⑯ if $\psi(IDev_{\mathrm{cost}}) = 0$ then

⑰ $c_{\mathrm{O2}} \leftarrow (N_{Add(a_1, a_2)_{\times}} \| \hbar(v_{L+\wedge N^*\times})) \| (Hide(a_1) \cup Hide(a_2))$；

⑱ else

⑲ $r_{\mathrm{O2}} \leftarrow (N_{sl\_Esert(a_1, a_2)_{\times}} \| N_{sl\_Esert((a_1, ..., a_n), (a_{n+1}, ..., a_{n+m}))_{\times}}) \| r_{\mathrm{O2}} \rightleftarrows c_{\mathrm{O2}}$；

⑳ end if

㉑ end if

㉒ if $\ell(e_i) = \lambda(t_j) = a_k \wedge v_{L\prec\wedge N^*\times} \| v_{L\times\wedge N^*\prec}(a_1, a_2)$ then

㉓ if $\psi(IDev_{\mathrm{cost}}) = 0$ then

㉔ $c_{\mathrm{O3}} \leftarrow (Add(a_1 \| a_2) \| (Hide(a_1) \cup Hide(a_2)))$；

㉕ else

㉖ $r_{\mathrm{O3}} \leftarrow sl - Esert(a_1 \| a_2) \| r_{\mathrm{O3}} \rightleftarrows c_{\mathrm{O3}}$；

㉗ end if

㉘ end if

㉙ if $v_{L\|N^*\subseteq+\|\prec\|\times}(a_1,a_2)\in R(\wp_L)\|R(\wp_{N^*})\wedge v_{N^*\|L\subseteq+\|\prec\|\times}(a_1,a_2)\notin R(\wp_{N^*})\|R(\wp_L)$ then

㉚ if $\psi(IDev_{\text{cost}})=0$ then

㉛ $c_{O4}\leftarrow(\tau\prec(N_{Add(a_1,a_{n2})\prec\|+\|\times},\tau)_\times\prec\tau)/(\tau\prec(\tau,(N_{Hide(a_1,a_2)\prec\|+\|\times},\tau)_\times\prec\tau)$;

㉜ else

㉝ $r_{O4}\leftarrow(sl\_Esert(a_1,a_2)_{+\|\prec\|\times})\|r_{O4}\rightleftarrows c_{O4}$;

㉝ end if

㉞ end if

㉟ end if

㊱ end for

㊲ end for

㊳ end for

㊴ $C_O=C_O\cup c_{O1}\cup c_{O2}\cup c_{O3}\cup c_{O4}$， $R_O=R_O\cup r_{O1}\cup r_{O2}\cup r_{O3}\cup r_{O4}$;

㊵ return C_O， R_O.

4.2 循环部分的模型修复

循环部分的修复首先需要识别其所发生的位置，由于长度为 1 或者 2 的短循环可通过 ∂_+ 算法来识别[40]，因此这里仅对包含 3 个及以上活动的循环进行分析．设定 $p_{\text{start}}\cdots l_{\text{return}}\cdots p_{\text{end}}$ 为一条循环路径，其中 p_{start} 和 p_{end} 分别表示循环路径上的起始库所与结束库所，l_{return} 表示返回路径上的行为模式．$loop_{\text{body}}=(p_{\text{f(entry)}}\cdots p_{\text{start}},p_{\text{start}}\cdots l_{\text{return}}\cdots p_{\text{end}})$ 表示一个完整的循环体，也就是循环中所包含的所有活动以及它们之间的行为关系，其中 $p_{\text{f(entry)}}\cdots p_{\text{start}}$ 为非循环部分．判定一个循环体结构的存在需要满足以下 2 点：1）循环体输入库所 p_{entry} 的因果后置库所 $p_{\text{f(entry)}}$ 与结束库所 l_{end} 是一致的，即 $p_{\text{f(entry)}}=p_{\text{end}}$；2）$p_{\text{f(entry)}}$ 的执行配置尺寸小于 p_{end}，记作 $p_{\text{f(entry)}}=p_{\text{end}}\wedge C(p_{\text{f(entry)}})<C(p_{\text{end}})$[41]．图 10 中描述了一个完整的循环体，并分别将 p_{entry}，p_{end}，p_{start} 3 个库所进行标注．

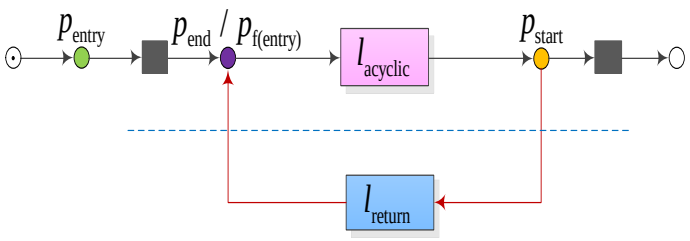


Fig.10 Path partitioning of the cyclic part

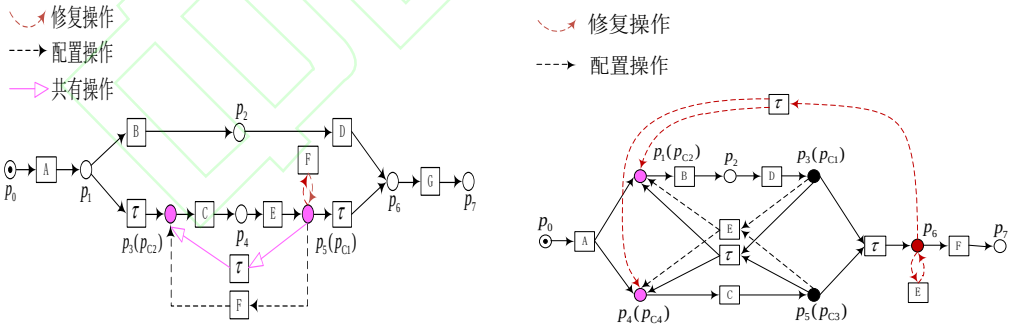
图 10 循环部分的路径划分

通过一个不等式来描述返回路径上的活动：

$$V(a|_{\ell(e),\lambda(t)})=\begin{cases}a, & \text{if } a\notin l_{\text{return}}.\\ a(l_{\text{return}}), & \text{otherwise.}\end{cases}$$

循环部分可能出现的变体有以下 3 种：1）只有事件日志中含有循环路径；2）非循环部分存在变体（这个问题在非循环部分进行讨论）；3）返回路径上存在变体.

修复循环部分需要根据以上 3 种情况产生的变体执行操作，本节在不考虑非循环部分的情况下对其中 2 种进行分析：1）仅事件日志中含有循环路径. 例如，图 11（a）所描述的初始网系统 N^* 中没有循环路径，而事件日志中则有一条循环路径 $l_{\text{aend}}...l_{\text{return}}...l_{\text{f(entry)}}=\text{E...F(A)...C}$，由此所产生的变体为 $v_{L_1}(\text{E...F(A)...C})$. 在这种情况下最优对齐检测到的偏差为 $Esert(\text{F})$，而配置回放图表发现的可配置活动为 $Add(\text{F})$. 图 11（a）中弯虚线部分表示 $v_{L_1}(\text{E...F(A)...C})$ 中含有迭代插入偏差时的修复操作，其位置为 $loc(Esert(\text{F}))\cup loc(\tau)=p_5\cup(p_5\sim p_3)$. 反之，则使用直虚线部分描述它的配置操作，其在网系统 N^* 上的位置为 $loc(Add(\text{F}))=(p_{C1}\sim p_{C2})$. 值得注意的是，空心箭头连接的部分是 2 种操作的共有部分. 2）事件日志与过程模型中都含有循环路径，但在返回路径 l_{return} 上的行为不一致. 例如，图 11（b）所描述的网系统 N^* 中的循环路径是 $l_{\text{aend}}...l_{\text{f(entry)}}=\text{D/C...B/C}$，而事件日志 L 中的循环路径则为 $l_{\text{aend}}...l_{\text{return}}...l_{\text{f(entry)}}=\text{D/C...E(A)...B/C}$，由此产生变体 $v_L(\text{E(A)})$. 图 11（b）中使用弯虚线描述了 $v_L(\text{E(A)})$ 中含有迭代插入偏差时对网系统 N^* 所做的修复操作，其位置为 $loc(Esert(\text{E}))\cup loc(\tau)=p_6\cup(p_6\sim(p_1\cup p_4))$. 反之，则采用直虚线所表示的配置操作，其位置为 $loc(Add(\text{E}))=((p_{C1}\cup p_{C3})\sim(p_{C2}\cup p_{C4}))$. 当上述第 1 种情况下非循环部分一致，且返回路径上没有活动存在时，2 种操作都直接使用隐变迁连接循环路径，从而准确回放当前变体中的所有偏差. 值得注意的是，如果可预测适合度在合理阈值范围内，则对循环部分统一实行直虚线描述的配置操作.



(a) 仅事件日志有循环的修复　　(b) 事件日志与网系统 N^* 均有循环的修复

Fig.11 The repairs of the cyclic part in two cases

图 11　2 种情况下的非拟合循环路径的修复

算法 2 是在默认非循环部分可回放的前提下，构建循环路径产生变体的 2 种操作. 首先，将空集赋予循环路径的修复操作和配置操作集 C_{01oop}，R_{01oop}（第①行）. 根据网 N^* 是否包含循环路径分 2 部分进行讨论：1）仅事件日志包含循环路径，此时若返回路径上没有活动则修复操作与配置操作相同(⑤～⑥行). 当返回路径上存在活动时，判定可预测适合度是否在合

理范围内，如果在合理范围内则统一使用配置操作修复所有偏差（⑦～⑩行），如果不在合理范围内就根据循环变体中是否包含迭代插入偏差选择合适的操作（⑪～⑮）．2）事件日志与网 N_{loop}^{*} 都包含循环路径时，仍需要判定可预测适合度是否在合理范围内，如果在合理范围内则统一使用配置操作修复所有偏差（⑱～㉑），如果不在合理范围内就根据具体情况在返回路径上执行配置操作和修复操作（㉒～㉖行）．最终，返回对循环路径上所有情况下可能执行的配置操作与修复操作集 C_{Oloop}， R_{Oloop} （第㉜行）．

算法 2．循环部分产生变体的选择性修复．

输入：有循环的网系统 N_{loop}^{*}，有循环的事件日志 L_{loop}，配置操作 c_{Oloop}，修复操作 r_{Oloop}；

输出：配置操作集 C_{Oloop}，修复操作集 R_{Oloop}．

① $C_{\text{Oloop}} \leftarrow \varnothing$， $R_{\text{Oloop}} \leftarrow \varnothing$；

② for each $e \in L_{\text{loop}}$ do

③ for each $t \in T$ do

④ for each $a \in Z$ do

⑤ if $l_{\text{aend}} \ldots l_{\text{f(entry)}} \in L_{\text{loop}} \wedge l_{\text{return}} = \varnothing$ then

⑥ $c_{\text{Oloop1}} \leftarrow (a, \tau, a')_{\prec}$， $r_{\text{Oloop1}} \leftarrow (a, \tau, a')_{\prec}$；

⑦ if $l_{\text{aend}} \ldots l_{\text{f(entry)}} \in L_{\text{loop}} \wedge l_{\text{aend}} \ldots l_{\text{f(entry)}} \notin N^{*}_{\text{loop}} \wedge a'' \in l_{\text{return}}$ then

⑧ if $pre\_fitness \geq 0.9$ then

⑨ $C_{\text{O}}(\sum D_{\text{loop}})$；

⑩ else

⑪ if $\psi(IDev_{\text{cost}}) = 0$ then

⑫ $c_{\text{Oloop2}} \leftarrow (a, (Add(a''), \tau)_{\times}, a')_{\prec}$；

⑬ else

⑭ $r_{\text{Oloop2}} \leftarrow (a, sl\_Eserta'', a')_{\prec}$；

⑮ end if

⑯ end if

⑰ end if

⑱ if $l_{L\text{aend}} \ldots l_{L\text{f(entry)}} \in L_{\text{loop}} \wedge l_{N^{*}\text{aend}} \ldots l_{N^{*}\text{f(entry)}} \in N^{*}_{\text{loop}} \wedge l_{L\text{return}} \neq l_{N^{*}\text{return}}$ then

⑲ if $pre\_fitness \geq 0.9$ then

⑳ $C_{\text{O}}(\sum D_{\text{loop}})$；

㉑ else

㉒ if $\psi(IDev_{\text{cost}}) = 0$ then

㉓ $c_{\text{Oloop3}} \leftarrow Add(a) \,||\, Hide(a) = (a,\tau)_{\times} \,||\, (\tau,a)_{\times}$;

㉔ else

㉕ $r_{\text{Oloop3}} \leftarrow sl\_Esert(a) \,||\, Skip(a) = sloop(a) \,||\, (\tau,a)_{\times}$;

㉖ end if

㉗ end if

㉘ end if

㉙ end for

㉚ end for

㉛ end for

㉜ $C_{\text{Oloop}} = C_{\text{Oloop}} \cup c_{\text{Oloop1}} \cup c_{\text{Oloop2}} \cup c_{\text{Oloop3}}$, $R_{\text{Oloop}} = R_{\text{Oloop}} \cup r_{\text{Oloop1}} \cup r_{\text{Oloop2}} \cup r_{\text{Oloop3}}$;

㉝ return C_{Oloop} , R_{Oloop} .

4.3 基于偏差子结构改善修复操作

对包含迭代插入偏差的变体执行修复操作，无法准确修复其中所有的非迭代插入偏差. 针对这一问题，将此类变体中满足一定行为关系的非迭代插入偏差构造成子结构[42]. 多个偏差被看作为一个整体会减少事件日志中需要被修复的插入偏差个数. 这种方法对于精度改善的逻辑表达式为 $Esert(\ell(e_1),\ell(e_2))_{+/\prec} \to \frac{|L|-E\_Dev_{\cos t}}{|L|-\sum_v|\ell(e_1)+\ell(e_2)|\bullet|q|}$（$E\_Dev_{\cos t}$ 为插入偏差的总成本）. $Esert(\ell(e_1),\ell(e_2),...)_{+/\prec}$ 表示在一个变体中满足并发/序列关系的偏差元素. 插入偏差的个数不能通过冲突子结构减少，因而修复操作的偏差子结构仅由并发和因果关系组成.

5 评估

实验将本文方法与现存 3 种方法进行比较，并通过分析不同数据集上 4 种方法修复后的适合度与精度评估它们的性能. 现存 3 种方法分别为：1）在过程模型上使用自循环的方式插入事件日志中可观测的单个偏差，记作 R_1；2）构建隐变迁与可配置行为的冲突子结构，从而将事件日志中可配置行为添加到过程模型，记作 R_2；3）构建满足并发/序列关系的插入偏差子结构，并使用自循环的方式将其插入过程模型，记作 R_3. 提出方法记作 R_{m}. 值得注意的是，由于模型中跳过偏差的修复不影响精度，因而 4 种方法都使用隐变迁跳过的方式对其进行修复. 部分现存模型修复方法的性能可以使用 Prom 框架进行验证，但它的原始代码并没有提供其他方法的改变部分. 为保证实验结果的公平性，我们在此基础上使用 Java 语言编写程序验证 4 种修复方法的性能.

5.1 数据集

实验分别使用由人造和真实业务流程产生的 2 种数据集进行评估，记作 D_{a}，D_{r}. 通过对真实和人造业务流程中的所有行为进行模拟，获得与它们一致的发生序列集 δ_{r}^* 和 δ_{a}^*，并将 δ_{r}^* 和 δ_{a}^* 作为不同数据集的给定过程模型. 根据事件日志与过程模型之间可能存在的变体类型对 δ_{r}^* 和 δ_{a}^* 进行更改，从而产生 2 个执行序列集 σ_{a}^*，σ_{r}^*. 与此同时，根据循环路径可能出现的不同变体，对 δ_{r}^* 进行更改，产生执行序列集 σ_{loop}^*. 在 σ_{a}^*，σ_{r}^* 和 σ_{loop}^* 中随机生成不同个数的自循环，并由此将每个执行序列集进一步划分为 3 个执行序列集，即 $(\sigma_{\mathrm{a1}}^*,\sigma_{\mathrm{a2}}^*,\sigma_{\mathrm{a3}}^*)$，$(\sigma_{\mathrm{r1}}^*,\sigma_{\mathrm{r2}}^*,\sigma_{\mathrm{r3}}^*)$，$(\sigma_{\mathrm{loop1}}^*,\sigma_{\mathrm{loop2}}^*,\sigma_{\mathrm{loop3}}^*)$. 使用 PSLG 插件将 9 个执行序列集分别自动生成为不同的事件日志，并从每个生成日志中挑选 10 条能够描述所有行为关系的执行序列. 基于每条执行序列的发生次数之上重复模拟 20 次，从而建立一个新的日志用于实验，记作 $(L_{\mathrm{a1}}^*,L_{\mathrm{a2}}^*,L_{\mathrm{a3}}^*)$，$(L_{\mathrm{r1}}^*,L_{\mathrm{r2}}^*,L_{\mathrm{r3}}^*)$，$(L_{\mathrm{loop1}}^*,L_{\mathrm{loop2}}^*,L_{\mathrm{loop3}}^*)$. 实验使用的 9 个数据集信息被记录于表 1 中，其中 $|M_L|$ 表示过程模型在对齐过程中包含的活动个数，因而在核算时需要考虑与之比较的每条迹的发生次数.

Table 1 Information of Data Sets

表 1 数据集信息

数据集	活动个数	$\lvert L\rvert(10^2)$	$\lvert M_L\rvert(10^2)$

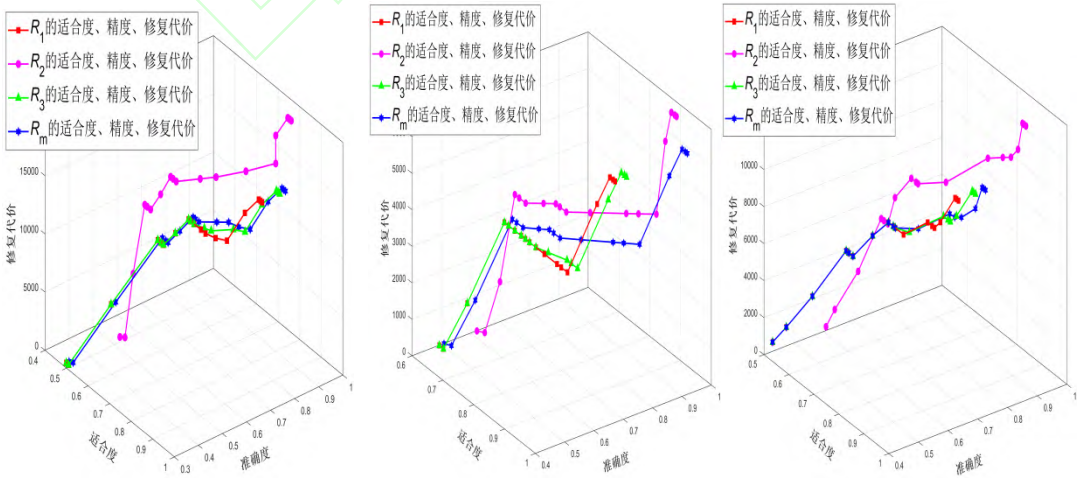
D_{a1}	21	520	240
D_{a2}	21	366	220
D_{a3}	22	482	482
D_{r1}	25	464	338
D_{r2}	25	340	464
D_{r3}	25	436	246
$D_{..1}$	23	596	224
$D_{..2}$	25	500	320
$D_{..3}$	25	558	344

5.2 实验结果

为保证适合度与精度的改善能够被更加清楚的观测到，实验数据默认 4 种方法的精度初始值均为所有插入偏差都无法被准确修复的结果，并通过变体的逐个修复不断产生变化.

5.2.1 人造业务流程数据集的实验结果

图 12 中 3 个数据集的可预测适合度分别为 0.76，0.89，0.84，因此 R_m 需要通过校验各种变体中是否包含迭代插入偏差，并根据校验结果选择合适的操作对每个变体进行修复. R_2 使用配置操作对所有偏差进行修复，因此能够使精度最终为 1. 然而，R_2 不能修复迭代插入偏差，以至于适合度无法得到保证. 图 12（a）～（c）中 R_2 修复后的适合度分别为 0.76，0.89，0.84，且均不在适合度的合理范围内. R_m 能够修复所有偏差，但不能准确修复包含迭代插入偏差变体中的非迭代插入偏差. 因此，包含迭代插入偏差变体中的非迭代插入偏差成本占比值越小，R_m 的精度值就越高. 图 12（b）中包含迭代插入偏差变体中的非迭代插入偏差成本在总成本中的占比值最小，因此其修复后的精度值比图 12（a）（c）分别高出 16% 和 19%. 相较于 R_1，R_3 来说，R_m 能够准确修复不包含迭代插入偏差变体中的所有非迭代插入偏差. 因此，在适合度一样的前提下 R_m 修复后的精度值始终高于或等于 R_3，且 R_3 始终高于或等于 R_1. 图 12 中的结果表明，提出方法 R_m 在确保适合度合理的情况下，精度值为 4 种方法中最高的.



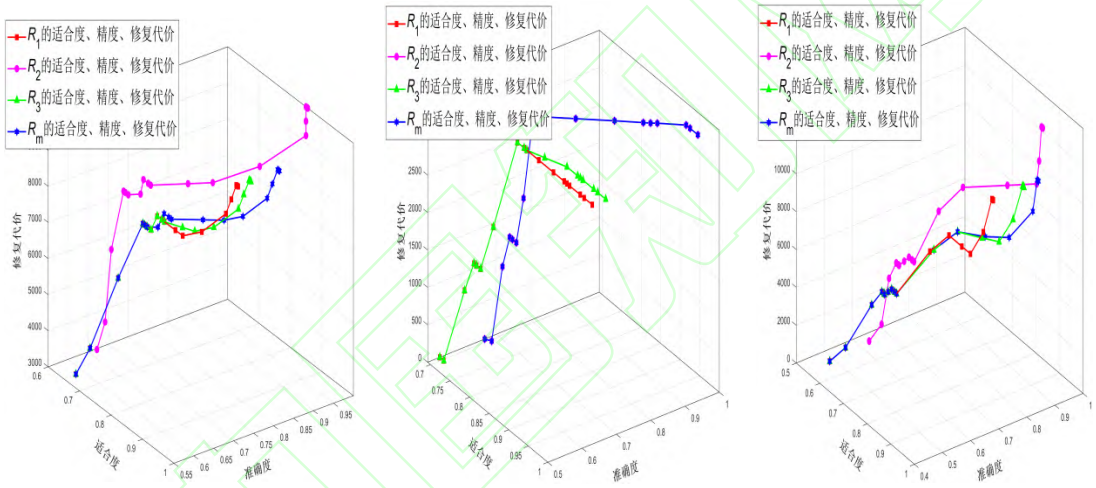
(a) 人造数据集 1　　(b) 人造数据集 2　　(c) 人造数据集 3

Fig.12 The running results of an artificial business process

图 12 人造业务流程的运行结果

5.2.2 真实业务流程数据集的实验结果

图 13（a）（c）的可预测适合度分别为 0.86 和 0.83，因此需要切换执行 2 种操作修复不同的变体．图 13（a）中使用 R_m 修复后的精度值远高于 R_1 和 R_3，这是由于这个数据集中迭代插入偏差在总偏差成本的占比值最大．如图 13（c）所示，虽然 R_1 与 R_m 之间精度值的差异很大，但 R_3 与 R_m 之间精度值的差异却仅为 5%．这是由于图 13（c）的数据集中能够组建为子结构的偏差最多，其可以有效减少日志中需要修复的插入偏差个数．图 13（a）（c）中，使用 R_2 修复后的适合度为 0.86 和 0.83，因而不在合理范围内．图 13（b）中的可预测适合度为 0.95，因此使用配置操作对所有偏差进行修复．这种情况下 R_2 和 R_m 修复后的适合度与精度相同，也就是适合度为 0.95 而精度则能够达到 1．图 13 中 3 个数据集的结果均显示，提出方法 R_m 能够在保证适合度合理的前提下使得精度值达到最高．



(a) 真实数据集 1　　(b) 真实数据集 2　　(c) 真实数据集 3

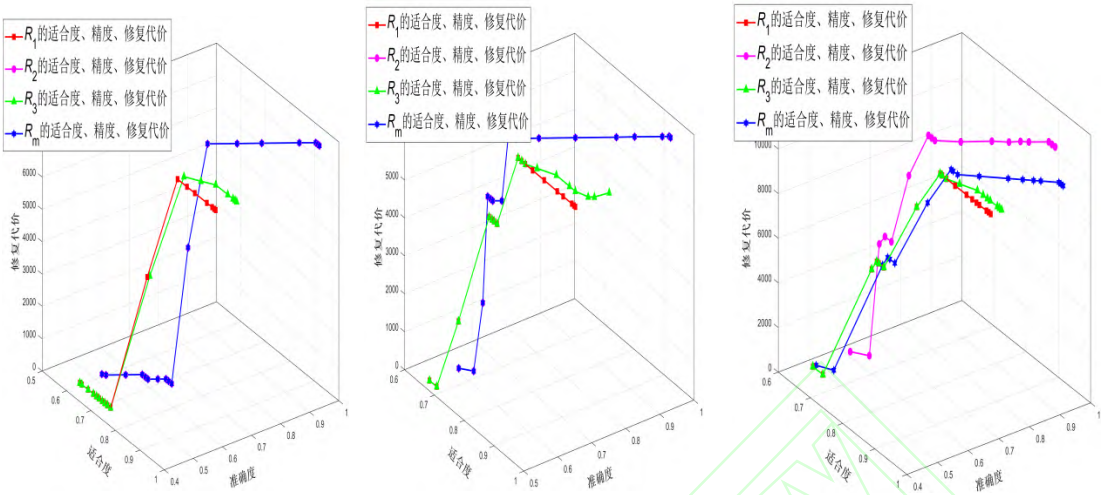
Fig.13 The running results of an real-life business process

图 13. 真实业务流程的运行结果

5.2.3 有循环数据集的实验结果

图 14 中描述了有循环数据集的修复性能，结果表明使用提出方法 R_m 修复后可在保证适合度合理的情况下使精度最优．图 14（a）（b）中的预测适合度均为 0.92，并使用配置操作修复所有偏差．图 14（c）的预测适合度为 0.89，则需要根据当前变体是否包含迭代插入偏差选择合适的操作．由于循环可能产生很多重复的偏差，因此当非循环部分产生变体中不包含迭代插入偏差时，会造成使用 R_m 与 R_1 修复后的精度差异很大．迭代插入偏差成本在总偏差成本中占比越小，R_m 与 R_1 之间精度的差异值越大．图 14（b）中迭代插入偏差的占比值比图 14（c）低 9%，然而图 14（c）中 R_m 相较于 R_1 在精度上的改善反而比图 14（b）高了 4%．造成这种情况的原因为图 14（c）中仅事件日志中有循环，且其返回路径上无活动．因此，R_m

与 R_1， R_3 都使用隐变迁连接循环路径，从而使 3 种方法对这种循环产生的偏差具有相同的修复效果.



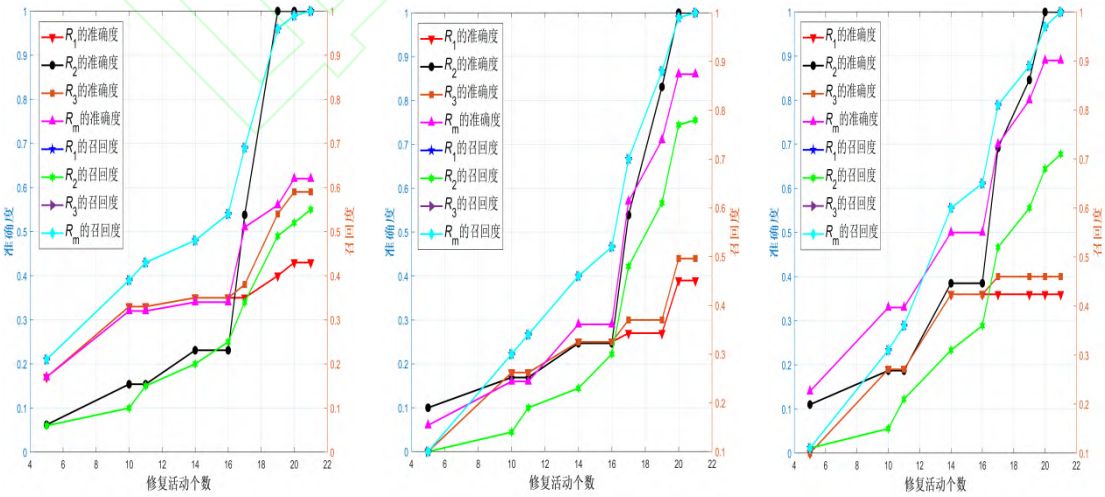
(a) 循环数据集 1　　(b) 循环数据集 2　　(c) 循环数据集 3

Fig.14 The running results of an business process with loop

图 14 有循环路径的业务流程运行结果

5.3 召回度、准确度以及修复活动的数量

召回度是指修复后模型能够回放的偏差成本在总偏差成本中的占比值，准确度是指能够准确回放的偏差成本在总插入偏差成本中的占比值，修复活动是指修复过程中不同标签或不同位置的偏差个数. 选择图 13（a）、图 14（b）（c）中的 3 个数据集进行验证，图 15（a）（c）中数据集的可预测适合度分别为 0.86 和 0.89，R_1， R_3， R_m 的召回度最终都能够达到 1，且 R_m 的准确度高于 R_1， R_3. 然而，图 15（a）和（c）中 R_2 受迭代插入偏差无法被修复的影响，它们的召回度分别为 0.55 和 0.71. 图 15（b）中数据集的可预测适合度为 0.92， R_2 与 R_m 的召回度均为 0.78，且精度为 1.



(a) 真实数据集 1　　(b) 循环数据集 2　　(c) 循环数据集 3

Fig.15 Recall, accuracy, and number of repair activities

图 15 召回度、准确度以及修复活动个数

6 总结

本文提出一种新的方法来修复事件日志无法回放于过程模型的行为，并根据不同情况选择合适的修复方式．该方法能够在确保适合度合理的前提下，尽可能改善精度，其主要从以下几步进行分析：

1） 根据迭代插入偏差预测适合度，并设定合理的阈值范围．

2）预测适合度合理时，使用配置操作对所有偏差进行修复，从而能够很好地兼顾适合度与精度．反之，则需要发现事件日志与过程模型之间的变体，并检测每个变体中是否包含迭代插入偏差．

3）一个变体包含迭代插入偏差时，将其中所有直接跟随的非迭代插入偏差根据行为关系构建为子结构，并执行修复操作．反之，则对其执行配置操作．由于本文的修复方法将配置优化引入到模型修复中，因此使部分修复后的变体在精度上得到完整提升．

文中将循环路径的修复问题单独提出，根据循环路径产生变体的不同类型从 3 方面进行分析．这样可以针对不同情况对修复方案进行调整，从而进一步改善循环部分的修复性能．

参考文献

[1] van der Aalst WMP, 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] van der Alast WMP. Distributed process discovery and conformance checking [G] // LNCS 7212: Proc of the 15th Int Conf on Fundamental Approaches to Software Engineering. Berlin: Springer, 2012: 1-25

[3] Fahland D, van der Aalst WMP. Repairing process models to reflect reality [G] // LNCS 7481: Proc of the 10th Int Conf on Business Process Management. Berlin: Springer, 2012: 229-245

[4] Leoni D M, van der Aalst WMP, Dongen B. Data- and resource-aware conformance checking of business processes [G] // LNBIP 117: Proc of the 15th Int Conf on Business Information Systems. Berlin: Springer, 2012: 48-59

[5] van der Aalst WMP, Adriansyah A, Dongen B. Replaying history on process models for conformance checking and performance analysis [J].Wiley Interdisciplinary Reviews-Data Mining and Knowledge Discovery, 2012, 2(2):182-192

[6] Fahland D, van der Aalst WMP. Simplifying discovered process models in a controlled manner [J]. Information Systems, 2013, 38(4): 585-605

[7] vanden Broucke SKLM, Weerdt JD, Vanthienen J, et al. Determining process model precision and generalization with weighted artificial negative events [J]. IEEE Transactions on Knowledge & Data Engineering, 2014, 26(8):1877-1889

[8] Sun Jinyong, Gu Tianlong, Wen Lijie, et al. Adaptation algorithm of semantic workflows based on behavioral characteristics [J]. Journal of Software, 2018, 29(11): 3260-3277 (in Chinese)

（孙晋永, 古天龙, 闻立杰，等. 基于行为特征的语义工作流修正算法[J]. 软件学报, 2018, 29(11):3260-3277）

[9] Wang Lu, Du Yuyue, Qi Hongda. Process model repair based on firing sequence [J]. Journal of Computer Research and Development, 2018, 55(3): 585-601（in Chinese）

（王路, 杜玉越, 祁宏达. 基于引发序列的流程模型修正[J]. 计算机研究与发展, 2018, 55(3):585-601）

[10] Hao Zhong, Hong Mei. Mining repair model for exception-related bug [J]. Journal of Systems and Software, 2018, 141:16-31

[11] Han VDA, Leopold H, Reijers H. Efficient Process conformance checking on the basis of uncertain event-to-activity mappings [J]. IEEE Transactions on Knowledge and Data Engineering, 2019, 32(5): 927-940

[12] Rozinat A, van der Aalst WMP. Conformance checking of processes based on monitoring real behavior [J]. Information Systems, 2007, 33(1): 64-95

[13] Adriansyah A, Dongen B, van der Aalst WMP. Conformance checking using cost-based fitness analysis [C]// Proc of the 15th IEEE Int Enterprise Distributed Object Computing Conf. Piscataway, NJ: IEEE, 2011: 55-64

[14] Garcia-Banuelos L, Beest NV, Dumas M, et al. Complete and interpretable conformance checking of business processes [J]. IEEE

Transactions on Software Engineering, 2018, 44(3): 262-290
[15] Zhang Liwen, Fang Xianwen. Business process fitness analysis based on alignment processing and deviation detection [J]. Computer Integrated Manufacturing System, 2020, 23(5): 1573-1581 (in Chinese)
（张力雯，方贤文. 基于对齐处理与偏差检测的业务流程适合度分析[J]. 计算机集成制造系统，2020，23(5):1573-1581）
[16] He Zhaoyang, Du Yuye, Qi Liang, et al. A model repair approach based on Petri nets by constructing free-loop structures [J]. IEEE Access, 2019, 7: 24214-24230
[17] Adriansyah A, Munoz-Gama J, Carmona J, et al. Measuring precision of modeled behavior [J]. Information Systems and E-business Management, 2015, 13(1): 37-67
[18] Toon C, Christian W G, Mykola P, et al. Using minimum description length for process mining [C]// Proc of the 2009 ACM Symp on Applied Computing. New York: ACM, 2009: 1451-1455
[19] Tax N , Lu Xixi , Sidorova N, et al. The imprecisions of precision measures in process mining [J]. Information Processing Letters, 2018, 135: 1-8
[20] Fahland D, van der Aalst WMP. Model repair-aligning process models to reality [J]. Information Systems, 2015, 47: 220-243
[21] Asadi M, Mohabbati B, Gröner G, et al. Development and validation of customized process models [J]. Journal of Systems and Software, 2014, 96: 73-92
[22] Francescomarino CD, Tiella R, Ghidini C, et al. A multi-objective approach to business process repair [G]// LNCS 8831: Proc of the 12th Int Conf on Service-Oriented Computing. Berlin: Springer, 2014: 32-46
[23] Cervantes AA, Beest N, Rosa ML, et al. Interactive and incremental business process model repair [G]// LNCS 10573: OTM Confederated Int Conf on the Move to Meaningful Internet Systems, Berlin: Springer, 2017: 53-74
[24] Qi Hongda, Du Yuyue, Qi Liang, et al. An approach to repair Petri net-based process models with choice structures [J]. Enterprise Information Systems, 2018, 12 (6/7/8/9/10): 1149-1179
[25] Zhang Xize, Du Yuyue , Qi Liang, et al. Repairing process models containing choice structures via logic etri nets [J]. IEEE Access, 2018, 6: 53796-53810
[26] Polyvyanyy A, van der Aalst WMP, Hofstede A, et al. Impact-driven process model repair [J]. ACM Transactions on Software Engineering and Methodology, 2016, 25(4): 1-60
[27] Zhang Liwen, Fang Xianwen, Shao Chifeng, et al. Real-time repair of business processes based on alternative operations in case of uncertainty [J]. IEEE Access, 2020, 9: 23672-23690
[28] Wang Huaqing, Xia Xin, Wen Lijie, et al. Business process recommendation based on structural relations [J]. Computer Integrated Manufacturing System, 2020, 26(6): 1445-1455 (in Chinese)
（王华清, 夏鑫, 闻立杰，等. 基于结构关系的过程模型推荐方法[J]. 计算机集成制造系统, 2020, 26(6):1445-1455）
[29] Bolt A, De Leoni M, van der Aalst WMP, Process variant comparison: Using event logs to detect differences in behavior and business rules [J]. Information Systems, 2018, 74(1): 53–66
[30] Rosa ML, van der Aalst WMP, Dumas M, et al. Business process variability modeling: A survey [J]. ACM Computing Surveys, 2017, 50(1): 1-45
[31] Buijs JCAM, Dongen BFV, van der Aalst WMP. Mining configurable process models from collections of event logs [G] // LNCS 8094: Proc of the 11th Int Conf on Business Process Management. Berlin: Springer, 2013: 33-48
[32] van der Aalst WMP, Dreiling A, Gottschalk F, et al. Configurable process models as a basis for reference modeling [G] // LNCS 3812: Proc of Int Conf on Business Process Management. Berlin: Springer, 2006: 512-518
[33] Li Chen, Reichert M, Wombacher A. Mining business process variants: Challenges, scenarios, algorithms [J]. Data and knowledge Engineering, 2011, 70(5): 409-434
[34] Fang Huan, Jin Pengpeng, Fang Xianwen, et al. Process variants cluster mining method based on causal behavioral profiles [J]. Computer Integrated Manufacturing System, 2020, 23(5): 1538-1547 (in Chinese)
（方欢，金朋朋，方贤文，等. 基于因果行为轮廓的流程变体聚类挖掘方法[J]. 计算机集成制造系统，2020，23(5):1538-1547）
[35] Armas-Cervantes A, Baldan P, Dumas M, et al. Diagnosing behavioral differences between business process models [J]. Information Systems, 2016, 56: 304-325
[36] Hu Qiang, Ren Zhikao, Zhao Zhen, et al. Study on structure evolution for service processes based on logic petri net [J]. Journal of Software, 2018, 29(9): 2697-2715 (in Chinese)
（胡强, 任志考, 赵振等. 基于逻辑 Petri 网的服务流程结构演化研究[J]. 软件学报, 2018, 29(9):2697-2715）
[37] Song Wei, Xia Xiaoxu, Jacobsen HA, et al. Efficient alignment between event logs and process models [J]. IEEE Transactions on Services Computing, 2017, 10(1): 136-149
[38] Assy N, Gaaloul W, Defude B. Mining configurable process fragments for business process design [G] // LNCS 8463: Proc of the 9th Int Conf on Design Science Research in Information Systems. Berlin: Springer, 2014: 209-224
[39] Fang Xianwen, Zhao Fang, Fang Huan, et al. The fusion analysis method about the change region of the business process model

based on behavior inclusion in petri net [J]. Journal of Computer Science, 2018, 23(5): 695-708 (in Chinese)
(方贤文，赵芳，方欢,等.基于 Petri 网的 Behavior Inclusion 业务流程变化域融合分析[J].计算机学报，2018，23(5)：695-708)
[40] Medeiros AKA, Dongen BF, van der Aalst WMP, et al. Process mining: Extending the alpha-algorithm to mine short loops [R]. Netherlands: Eindhoven University of Technology, 2004
[41] Medeiros AKA, van der Aalst WMP, Weijters AJMM. Workflow mining: Current status and future direction [G] // LNCS 2888: OTM Confederated Int Conf "On the Move to Meaningful Internet Systems". Berlin: Springer, 2003: 389-406
[42] Fang Huan, Li Dongyue, Sun Shuya. Process conformance checking method based on alignment of direct succession relations [J]. Computer Integrated Manufacturing System, 2020, 23(5): 1887-1895 (in Chinese)
(方欢，李东月，孙书亚.基于直接后继关系对齐的过程符合性检测[J].计算机集成制造系统，2020，23(5):1887-1895)

作者简介：

Zhang Liwen, born in 1988. PhD candidate. Her main research interests include Petri net, process mining, model repair, and configuration optimization.
张力雯, 1988 年生. 博士研究生. 主要研究方向为 Petri 网、偏差检测以及模型修复.

Fang Xianwen, born in 1989. PhD, professor, PhD supervisor. Member of CCF. His main research interests include Petri net, trustworthy software, and Web services.
方贤文，1975 年生. 博士，教授，博士生导师. CCF 会员. 主要研究方向为 Petri 网和可信软件.

Shao Chifeng, born in 1995. Master candidate. His main research interests include Petri net, process mining, machine learning.
邵叱风，1995 年生. 硕士研究生. 主要研究方向为 Petri 网、过程挖掘、机器学习。

Wang Lili, born in 1983. PhD candidate. Associate professor, master supervisor. Her main research interests include Petri net and formal verification of software.
王丽丽，1983 年生. 博士研究生，副教授，硕士生导师. 主要研究方向为 Petri 网和软件的正式验证等.

作者贡献声明

张力雯，主要负责阅读相关文献并结合自身研究确定文章的创新点，撰写文章并绘制图表，设计实验方案以及实验所需的数据集。

方贤文，把控文章的创新点，对文章内容进行修改与凝练，提供项目支持。

邵叱风，完成文章的实验操作。

王丽丽，检查文章的疏漏，配合实验部分的完成，提供项目支持。