计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.200700125

基于 WFT-net 验证合理性的动态数据精炼策略

陶小燕　闫春钢　刘关俊
同济大学计算机科学与技术系　上海 201804
同济大学嵌入式系统与服务计算教育部重点实验室　上海 201804
(txyccl@163.com)

摘　要　带有数据表的工作流网(WFT-net)用于验证业务流程的合理性,包括正确的行为逻辑和满足的数据需求。在某些情况下,静态数据精炼策略存在无法反映流程中所有可能执行路径的情况,这会导致检测正确率不理想等问题。为此,文中提出了一种新的动态数据精炼策略。首先,提出了在流程运行当前状态下评估与被写数据元素相关联的数据表和谓词状态的方法,捕捉数据流状态的实时变化,全面反映流程执行所有可达的状态,避免执行路径的丢失。此外,当流程执行陷入会导致数据流状态无限更新的循环时,通过适当调整赋值精炼规则的方式,来避免状态的无限延伸。然后,基于所有可能执行路径来检测流程的合理性。最后,在不同业务流程实例上的实验结果表明,该动态数据精炼策略能够有效提高合理性检测的正确率。

关键词:Petri 网;数据表;合理性;数据精炼;循环结构
中图法分类号　TP306

Dynamic Data Refining Strategy for Soundness Verification Based on WFT-net

TAO Xiao-yan, YAN Chun-gang and LIU Guan-jun
College of Computer Science and Technology, Tongji University, Shanghai 201804, China
The Key Laboratory of the Ministry of Education for Embedded System and Service Computing, Tongji University, Shanghai 201804, China

Abstract　The workflow net with data tables (WFT-net) has been proposed to verify the soundness of business processes, to ensure the correctness of business logics and the satisfiability of data requirements. In some cases, the static data refining strategy may not reflect all possible execution situations of the business process, which can cause problems such as poor detection accuracy. To this end, a new dynamic data refining strategy is proposed in this paper. First, a method for evaluating the status of tables and predicates associated with the written data element in the current state of the WFT-net is given, to capture real-time changes in data-flow status, and to fully reflect all reachable states in process execution, so as to avoid the loss of the execution path. In addition, when the process execution is caught in a loop that will cause the data-flow status to be updated infinitely, the data assignment rules are appropriately adjusted to avoid the consequent infinite state. Then, the soundness of the business process is verified based on its all possible execution situations. At last, experimental results based on different business process instances show that the dynamic data refining strategy is able to improve the accuracy of soundness verification.

Keywords　Petri net, Data tables, Soundness, Data refinement, Loop structure

1　引言

在工作流/业务流程管理系统中,不正确的业务流程设计可能会导致数据或行为异常[1]。为检测此类错误,业务流程建模模型的正确性规范[2]变得尤为重要,其中合理性是不可或缺的部分。一般而言,合理性[3]要求流程能够无死锁、无活锁、无死变迁地正常终止,可基于工作流网(WF-net)进行检测[4-6]。数据流作为业务流程不可或缺的组成部分[7-8],不正确的数据部署[9-11]也会引起流程无法正常到达终止状态,即使该流程的控制流是合理的。为此,含数据流的多种工作流网模型相继出现并用于验证合理性,如 WFD-net[12],WDG[13],DPN[14],WTC-net[15]。但是,这些工作并未考虑业务活动中的流程数据与底层数据库间的交互,无法描述系统中数据自身约束的设计需求(简称数据需求)。

为解决上述问题,文献[16]中提出了带有数据表的工作流网(WFT-net),并从数据需求的角度丰富了合理性的概念,且基于下列数据精炼[17]策略验证了业务流程的合理性:每个数据元素仅在第一次被写时根据当前数据库和关联谓词的状态进行赋值精炼,之后其精炼值始终保持不变。这种数据精炼策略是静态的。

到稿日期:2020-07-20　返修日期:2020-08-20　本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家重点研发项目(2018YFB2100801)
This work was supported by the National Key Research and Development Program of China(2018YFB2100801).
通信作者:闫春钢(cgyan2@163.com)

实际上，由于活动的执行促使流程数据和底层数据库不断交互，数据库的状态随着流程运行不断发生变化。此时，数据元素始终如一的精炼值可能会导致死变迁或流程执行陷入死循环等情况，具体实例将在下文进行阐述。概括地说，即使WFT-net的控制流本身是合理的，但是在静态数据精炼策略下却被判定为不合理。

直观地，可以从根源入手来解决上述问题：数据元素每次被写时都将根据当前数据库和关联谓词的状态进行赋值精炼。但这也带来了新的挑战：如何处理非静态数据精炼策略下无界数据引起的流程流状态无限延伸？为此，本文首先根据流程执行当前状态对被写数据元素进行赋值精炼，全面反映流程所有可能的执行路径；然后针对会导致数据流状态无限更新的循环结构，适当调整赋值精炼规则，限制循环结构的执行次数，阻止状态无限延伸，避免造成活锁，进而提高检测的正确率。

2 WFT-net 模型

为增加可读性，本节在阐述动机之前，先回顾WFT-net模型及其相关概念，包括发生规则、可达性以及合理性。

定义 1 8元组 $DN=(P,T,F,\mathcal{D},\mathcal{R},O_{\mathcal{D}},O_{\mathcal{R}},G)$ 是一个WFT-net，其中：

(1)(P,T,F)是一个 *WF-net*；

(2)$\mathcal{D}$是一个有限的数据元素集；

(3)$\mathcal{R}=\{R_1,\cdots,R_n\}$是一个关系数据库，其中 $R_1,\cdots,R_n$ 是数据表；

(4)$O_{\mathcal{D}}=\{rd,wt,dt\}$是$\mathcal{T}$上的一组数据元素操作，其中 $rd:T\to 2^{\mathcal{D}}$，$wt:T\to 2^{\mathcal{D}}$，$dt:T\to 2^{\mathcal{D}}$ 分别是数据读、写、删除操作的标签函数；

(5)$O_{\mathcal{R}}=\{sel,ins,upd,del\}$是$\mathcal{R}$上的一组数据表操作，其中 $sel:T\to 2^{\mathcal{R}}$ 是查询数据表中元组和属性的标签函数，$ins:T\to 2^{\mathcal{R}}$ 是将元组插入到数据中的标签函数，$upd:T\to 2^{\mathcal{R}}$ 是更新数据表元组中非标识属性的值的标签函数，而 $del:T\to 2^{\mathcal{R}}$ 是删除数据表元组的标签函数；

(6)$G:T\to\mathcal{G}_{\Pi}$将守卫函数分配给变迁，其中$\mathcal{G}_{\Pi}$表示守卫函数的集合，每个守卫函数是谓词集 $\Pi=\{\pi_1,\pi_2,\cdots,\pi_n\}$ 上的一个布尔表达式。$\ell_{\Pi}:\Pi\to 2^{\mathcal{D}}\cup 2^{\mathcal{R}}$表示与谓词相关联的数据元素与数据表构成的集合。

WFT-net在某个时刻的运行状态称为一个配置，定义如下。

定义 2 $DN=(P,T,F,\mathcal{D},\mathcal{R},O_{\mathcal{D}},O_{\mathcal{R}},G)$ 是一个WFT-net。$c=(m,\theta,\omega,\sigma)$是 DN 的一个配置，其中：

(1)$m:P\to N$ 为工作流网(P,T,F)的标识函数，其中 $N=\{0,1,2,\cdots\}$为非负整数集；

(2)$\theta:\mathcal{D}\to\{T,\perp\}$表示每个数据元素 d 的赋值状态，$\theta(d)=T$ 表示 d 已定义，若 d 被写，则有 $\theta(d)=T$，而 $\theta(d)=\perp$ 表示 d 未定义，若 d 被删除则有 $\theta(d)=\perp$；

(3)$\omega:\mathcal{R}\to\{T,\perp\}$表示每个数据表 R 的赋值状态，$\omega(R)=T$ 表示 R 已定义，若给定属性对 R 进行初始化，则认为 R 是已定义的，而 $\omega(R)=\perp$ 表示 R 未定义；

(4)$\sigma:\Pi\to\{\text{true},\text{false},\perp\}$表示每个谓词 π 的赋值状态，$\sigma(\pi)=\text{true}$ 表示 π 为真，$\sigma(\pi)=\text{false}$ 表示 π 为假，而 $\sigma(\pi)=\perp$ 表示 d 未定义，若 $\exists d,R\in\ell_{\Pi}(\pi):\theta(d)=\perp\vee\omega(R)=\perp$，则有 $\sigma(\pi)=\perp$。

在一个配置 $c=(m,\theta,\omega,\sigma)$中，$(\theta,\omega,\sigma)$代表数据流状态。另外，$\eta:\mathcal{G}\to\{\text{true},\text{false},\perp\}$表示每个守卫函数的赋值状态，包括真(true)、假(false)与未定义($\perp$)。

定义 3 $DN=(P,T,F,\mathcal{D},\mathcal{R},O_{\mathcal{D}},O_{\mathcal{R}},G)$是一个WFT-net。变迁 $t\in T$ 在配置 $c=(m,\theta,\omega,\sigma)$ 使能，即 $c[t\rangle$，当且仅当：

(1)$m[t\rangle$；

(2)$\forall d\in rd(t):\theta(d)=T$；

(3)$\forall R\in sel(t)\cup ins(t)\cup upd(t)\cup del(t):\omega(R)=T$；

(4)$\eta(G(t))=\text{true}$。

变迁 t 的发生可能会生成多个新的配置。令 $ins(R)$表示添加到表 R 中的元组构成的集合，$del(R)$表示从 R 中删除的元组构成的集合，$upd(R)$表示 R 中要被更新的元组构成的集合，且更改后得到的新元组构成的集合记为 $upd'(R)$。

定义 4 $DN=(P,T,F,\mathcal{D},\mathcal{R},O_{\mathcal{D}},O_{\mathcal{R}},G)$是一个WFT-net。在配置 $c=(m,\theta,\omega,\sigma)$使能的变迁 $t\in T$ 被触发之后产生一个新的配置集 C，记为 $c[t\rangle C$，其中：

$$
\begin{aligned}
C=\{&(m',\theta',\omega',\sigma')\mid m[t\rangle m'\\
&\wedge(\forall d\in wt(t):\theta'(d)=T)\\
&\wedge(\forall d\in dt(t):\theta'(d)=\perp)\\
&\wedge(\forall R\in ins(t):\omega'(R)=\omega(R)\cup ins(R))\\
&\wedge(\forall R\in upd(t):\omega'(R)=\omega(R)\cup upd'(R)\backslash upd(R))\\
&\wedge(\forall R\in del(t):\omega'(R)=\omega(R)\backslash del(R))\\
&\wedge(\forall\pi\in\Pi:\ell_{\Pi}(\pi)\cap(wt(t)\backslash dt(t))\neq\emptyset\Rightarrow\sigma'(\pi)=\perp)\\
&\wedge(\forall\pi\in\Pi:\forall d,R\in\ell_{\Pi}(\pi):\theta'(d)=T\wedge\omega'(R)=T)\Rightarrow\sigma'(\pi)\in\{\text{true},\text{false}\})\\
&\wedge(\forall\pi\in\Pi:\ell_{\Pi}(\pi)\cap(wt(t)\cup dt(t)\cup ins(t)\cup upd(t)\cup del(t))=\emptyset\Rightarrow\sigma'(\pi)=\sigma(\pi))\}。
\end{aligned}
$$

定义 5 $DN=(P,T,F,\mathcal{D},\mathcal{R},O_{\mathcal{D}},O_{\mathcal{R}},G)$是一个WFT-net。$c$ 和 c' 为 DN 的两个配置，C 和 C' 为 DN 的两个配置集。

(1)如果存在 $t\in T$ 满足 $c[t\rangle C$，则称 c 到 C 有一个必然发生步(must-step)，记为 $c\to_{must}C$；

(2)如果 $c\to_{must}C$ 且 $c'\in C$ 成立，即 $c[t\rangle c'$，则称从 c 到 c' 有一个可能发生步(may-step)，记为 $c\to_{may}c'$；

(3)如果存在一个配置序列 $c_0,c_1,\cdots,c_n$ 满足 $c_i\to_{may}c_{i+1}$ $(0\leqslant i\leqslant n)$，令 $c_0=c,c_n=c'$，则称 c' 从 c 可能可达(may-reachable)，也称 c 到 c' 存在一条长为 n 的可能发生路径(may-path)，记为 $c\to^{*}_{may}c'$；

(4)如果 $C_0=\{c\}$，$C_n=C$，且 $C_{i+1}=\{x\in C'\mid\exists y\in C_i,\exists C':y\to_{must}C'\}\cup\{x\in C'\mid\forall t\in T:\neg x[t\rangle\}$，则称 C 从 c 必然可达(must-reachable)，也称 c 到 C 存在一条长为 n 的必然混合发生路径(must-hyper-path)，记为 $c\to^{*}_{must}C$。

令 C_c 为从 c 出发的所有可能可达配置构成的集合。

定义 6 $DN=(P,T,F,\mathcal{D},\mathcal{R},O_{\mathcal{D}},O_{\mathcal{R}},G)$是一个WFT-net。$c_0=(m_0,\theta_0,\omega_0,\sigma_0)$为初始配置，$C_f$ 为终止配置集。一个LTL-FO公式集 $\Lambda=\{\lambda_1,\cdots,\lambda_n\}$表示 DN 需要满足的数据需求。如果满足以下两个条件，则 DN 是合理的。

(1)DN 是控制流合理的，即：

1) $\forall c\in C_{c_0}\backslash C_f$，$\exists c_f\in C_f: c\rightarrow_{may} c_f$；

2) $\forall c\in C_{c_0}: c[m]\geqslant[end]\Rightarrow c\in C_f$；

3) $\forall t\in T, \forall c\in C_{c_0}: c[t\rangle$。

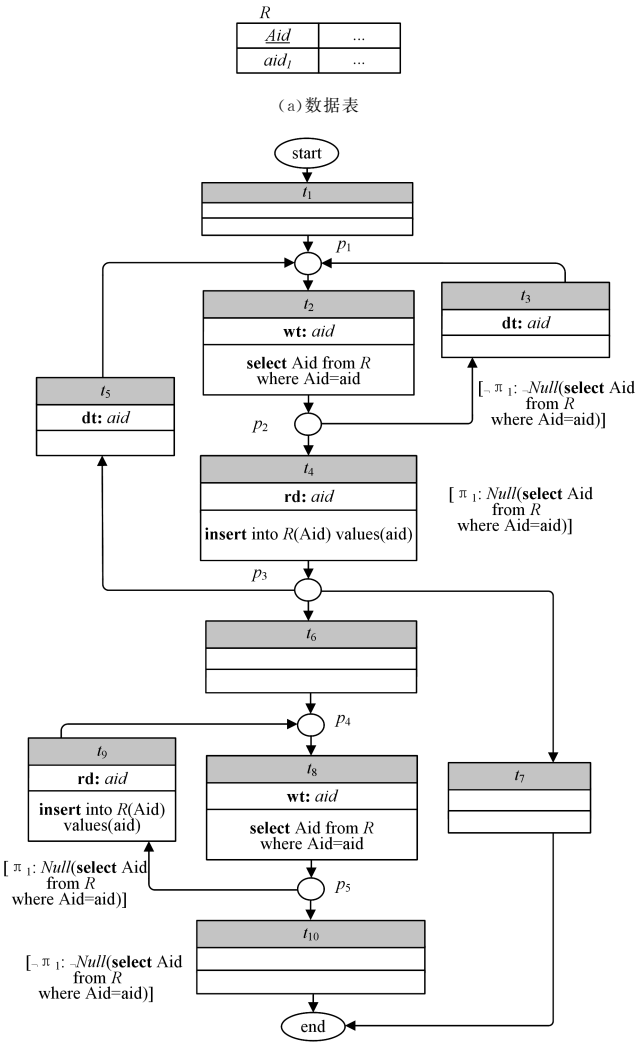
(2)DN 是数据流合理的，当且仅当 DN 满足所有数据要求，即 $\forall \lambda_i\in\wedge: DN\vdash\lambda_i$。

WFT-net 的其他相关概念请参见文献[17]，此处不再赘述。

3 动机例子

在图 1 的 WFT-net 中，数据元素集 $\mathcal{D}=\{aid\}$，aid 的值域为有理数域 $\mathbb{Q}$。

给定初始配置 $c_0=([start],\{aid\rightarrow\perp\},\{R=\{(aid_1)\}\},\{\pi_1\rightarrow\perp\})$，简记为 $c_0=([start],\{\perp\},\{(aid_1)\},\{\perp\})$，该 WFT-net 按照前文所述静态数据精炼策略生成的可达图如图 2 所示，其中数据元素 aid 的精炼值始终为 aid_1 和 aid_2。需要说明的是，在此过程中，循环默认只执行一次。例如，对于循环 $t_5-t_2-(t_3)-t_4$，触发变迁 t_4 会使能变迁 t_5，但 t_5 只被执行一次。



(a)数据表

(b)基本业务逻辑、数据元素/表操作与守卫函数

图 1 WFT-net 实例

Fig. 1 Example of WFT-net

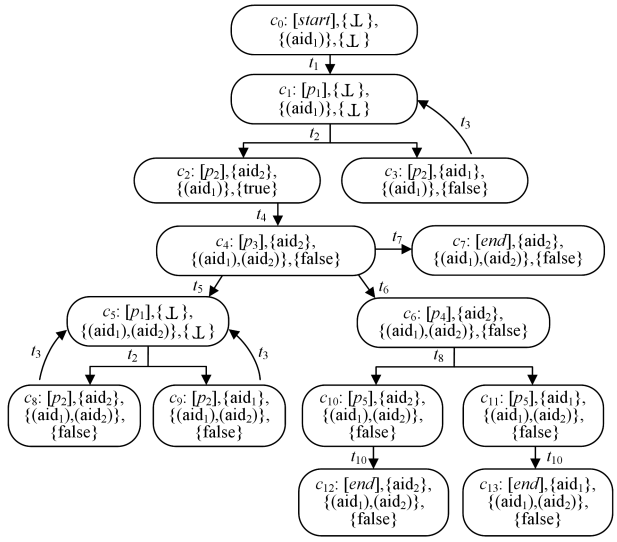


图 2 图 1 在静态数据精炼策略下生成的可达图

Fig. 2 Rechability graph of Fig. 1 based on static data refinement

在该 WFT-net 中，只有变迁 t_2 和 t_8 上有 aid 的写操作。从图 2 可以看出，当 t_2 在配置 c_5 被触发后，aid 的精炼值 aid_1 和 aid_2 皆评估谓词 π_1 为 false，此时仅变迁 t_3 有发生权，这将导致流程执行一直陷在循环 t_2-t_3，无法正常终止。同理，当 t_8 在配置 c_6 被触发后，aid 的精炼值 aid_1 和 aid_2 也皆评估谓词 π_1 为 false，此时仅变迁 t_{10} 有发生权，且触发 t_{10} 后流程执行终止，这就意味着变迁 t_9 从未被执行，属于死变迁。由此判断该 WFT-net 在控制层是不合理的。

事实上，若 t_2 在配置 c_5 被触发后 aid 能被赋值 $aid_2\in\mathbb{Q}$，那么谓词 π_1 能够评估为 true，此时变迁 t_4 有发生权，触发 t_4 即可退出循环 t_2-t_3。同理，变迁 t_9 也不会成为死变迁。此时，该 WFT-net 被认为在控制层是合理的。

综上所述，合理性的验证依赖于具体的数据精炼策略。但是前文所给的静态数据精炼策略并未考虑数据流的实时变化，无法正确反映业务流程的合理性。因此，本文将对该静态数据精炼策略进行优化。

4 动态数据精炼策略

为确保精炼策略满足行为性质保持，本文沿用文献[17]中的数据精炼原则，即数据元素的赋值考虑当前数据库所有可能的状态(已出现的和未出现的)以及相关谓词所有可能的赋值情况(真与假)。最重要的是，数据元素在每次被写时都按照上述规则进行赋值精炼，而不是采取第一次被写时得到的精炼值。这种数据精炼策略是非静态的。

然而，上述这种非静态数据精炼策略应用于循环结构可能会导致系统状态无限延伸，形成活锁，致使流程无法正常终止。例如，当图 1 执行完变迁发生序列 $\tau: t_1t_2t_4t_6$ 到达配置 $c_{i1}=([p_4,\{aid_2\},\{(aid_1),(aid_2)\},\{false\})$ 后，变迁 t_8 有发生权。按照非静态数据精炼策略，每次触发 t_8 都要根据当前数据流状态对 aid 进行赋值精炼，且结果一定包含一个使得谓词 π_1 被评估为 true 的精炼值，该值对应一个新的配置(如图 3 中的 c_{i2})。在新配置下，变迁 t_9 有发生权，而触发 t_9 又会使能 t_8，如此反复，流程会一直执行循环 t_8-t_9，形成活锁，如

图 3 所示。因此该 WFT-net 的控制流是不合理的。

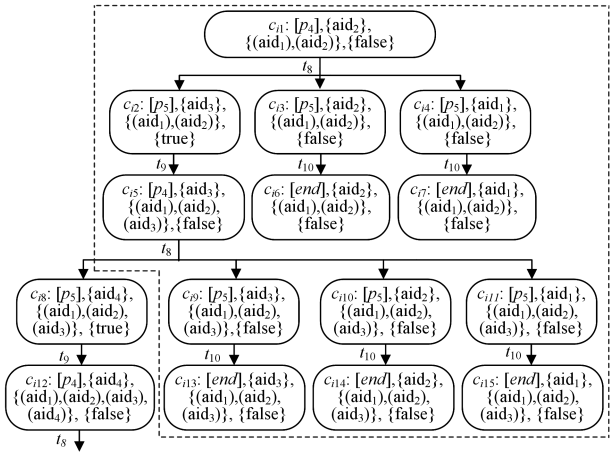


图 3 图 1 的循环 t_8-t_9 在非静态数据精炼策略下的执行

Fig. 3 Execution of loop t_8-t_9 in Fig. 1 based on non-static data refinement

针对上述问题，若仅通过强行限定循环执行次数来终止循环是不完善的，只能在本质上通过调整数据精炼策略来阻止状态无限延伸。例如，假设规定循环 t_8-t_9 只被执行一次，对应到图 3，流程执行到达配置 c_{i5} 之后强行终止，不再触发 t_8。此时虽然流程运行结束，但是 c_{i5} 并不是 WFT-net 的终止配置，该 WFT-net 在控制流依然被判定是不合理的。

在不丢失流程执行路径的基调上，本文对上述非静态数据精炼策略进行如下调整：WFT-net 从 c_0 执行变迁序列 τ：$t_1\cdots t_m$ 到达配置 c，当数据元素 d 在配置 c 被变迁 t 写时，先按照上述非静态数据精炼策略对 d 进行赋值，得到精炼值集合 $V_d^c{}'$，若从 c_0 到 c 的所有可能可达路径已包含 t，且 DN 存在一个其上守卫函数与 d 相关的变迁 t'，触发 t' 之后不再使能 t，则将 $V_d^c{}'$ 中所有能够使能 t' 的元素构成的集合 V_d^c 作为 d 在 c 被 t 写时对应的精炼值集合；否则，$V_d^c{}'$ 即为 d 在 c 被 t 写时对应的精炼值集合 V_d^c。

原来的非静态数据精炼策略经上述略微调整之后，本质上依然是根据当前数据库和相关谓词的状态对数据元素进行赋值精炼的，故较之未调整过的策略，将调整过后的非静态数据精炼策略正式称为动态数据精炼策略，形式化表达如下：DN 是一个 WFT-net，其初始配置为 c_0。设 V_d^c 为数据元素 d 在配置 c 被变迁 t 写入时对应的精炼赋值集。设从 c_0 到 c 有 $n(n\geqslant 1)$ 条可能可达路径，每条这样的路径对应的变迁发生序列分别为 $\tau_i: t_{i1}t_{i2}\cdots t_{ii_m}(1\leqslant i\leqslant n)$，$\Xi_c=\{\tau_i\mid 1\leqslant i\leqslant n\}$。

为便于形式化表达，令 B_d 表示 d 的值域，$\mathscr{R}_d$ 表示与 d 关联的所有数据表的集合，Π_d 表示与 d 相关的所有谓词的集合，则 V_d^c 的生成方法如算法 1 所示。

算法 1 生成 d 的精炼赋值集 V_d^c

输入：一个 WFT-net DN，c_0，d，c，t，B_d

输出：一个精炼赋值集 V_d^c

1. 初始化 $V_d^c{}'=\emptyset$，并计算 Ξ_c；
2. if $\mathscr{R}_d=\emptyset\wedge\ \Pi_d=\emptyset$ then
3. if $(\exists\tau_i\in\Xi_c:\ t\in\tau_i)\wedge(\exists c'\in C_{c_0}:\ c_0\rightarrow^*_{may}c'\rightarrow^*_{may}c\wedge c'[t\rangle)$ then
4. 令 $V_d^c{}'=V_d^c{}'$；
5. else
6. 从 B_d 中任选一个元素 d′添加到 $V_d^c{}'$；
7. end if
8. else
9. if $\mathscr{R}_d\neq\emptyset$ then
10. for $\forall R_i\in\mathscr{R}_d$ do
11. 在配置 c 下将 d 存储在 R_i 中的所有值添加到 $V_d^c{}'$ 中；
12. end for
13. 从 $B_d\backslash V_d^c{}'$ 中任选一个元素 d′添加到 $V_d^c{}'$；
14. end if
15. if $\Pi_d\neq\emptyset$ then
16. for $\forall\pi_i\in\Pi_d:\theta(d)=T\Rightarrow\sigma(\pi_i)=T$ do
17. if $V_d^c{}'$ 中的任何元素都不能使 π_i 为真(resp. 假) then
18. 从 B_d 中选择一个使 π_i 为真(resp. 假)的元素 d_i(resp. d_i')添加到 $V_d^c{}'$；
19. end if
20. end for
21. end if
22. if $\forall\tau_i\in\Xi_c:(\tau_i:\ t_{i1}\cdots t\cdots t_{ii_m}\wedge\exists t'\in(t^{\cdot\cdot}\cup\cdots\cup t_{ii_m}^{\cdot\cdot}\backslash\{t\cdots t_{ii_m}\}:d\in\ell(G(t')))$ then
23. 移除 $V_d^c{}'$ 中所有使得 G(t′)为假的元素；
24. end if
25. end if
26. 输出 $V_d^c{}'=V_d^c$。

例如，当图 1 执行完变迁发生序列 τ：$t_1t_2t_4t_6$ 到达配置 $c_{i1}=([p_4],\{aid_2\},\{(aid_1),(aid_2)\},\{\text{false}\})$ 后，循环 t_8-t_9 在算法 1 的动态数据精炼策略下的执行情况如图 3 中虚线框所示。当 aid 在 c_{i1} 被 t_8 写时，可得 $V_{aid}^{c_{i1}}{}'=\{aid_1,aid_2,aid_3\}$，此时 $t_8\notin\tau$，因此 $V_{aid}^{c_{i5}}=V_{aid}^{c_{i5}}{}'=\{aid_1,aid_2,aid_3\}$。同理，$V_{aid}^{c_{i9}}{}'=\{aid_1,aid_2,aid_3,aid_4\}$，此时有 $\tau':t_1t_2t_4t_6t_8t_9$ 使得 $t_8\in\tau'$，且存在 $t_{10}\in t_8^{\cdot\cdot}\cup t_9^{\cdot\cdot}\backslash\{t_8,t_9\}$ 使得 $aid\in\ell(G(t_{10}))$，而 $V_{aid}^{c_{i1}}{}'$ 中只有 aid_4 能在 c_{i5} 使得 $G(t_{10})$ 为假，因此可得 $V_{aid}^{c_{i5}}=V_{aid}^{c_{i1}}{}'\backslash\{aid_4\}=\{aid_1,aid_2,aid_3\}$。

注意，若在到达 c_{i5} 时就终止循环 t_8-t_9 的再执行，则意味着流程只要执行过 t_9 就不能再执行 t_8，显然这是不正确的。到此，算法 1 所示的动态数据精炼策略能在不丢失流程可能执行路径的前提下，正确反映 WFT-net 控制流所在的 WF-net 的合理性。

5 实验分析

5.1 实验工具与设计

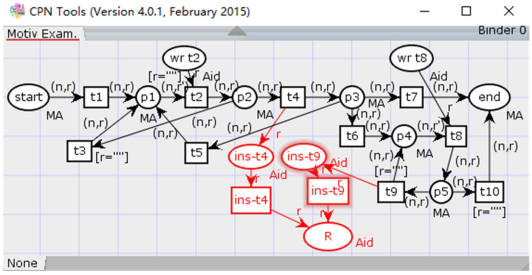
给定初始配置，本文基于 CPN Tools 模拟 WFT-net 在动态数据精炼策略下所有可能的执行情况，验证 WFT-net 的合理性。以图 1 为例，假设该 WFT-net 的数据需求为每个 aid 只能被注册一次，其合理性验证步骤具体如下：

(1)在 CPN Tools 中输入 WFT-net 模型，如图 4(a)所示，图中黑色图框对应 WFT-net 的控制流以及变迁上的数据元素操作和数据约束，红色图框对应变迁上的数据库操作。

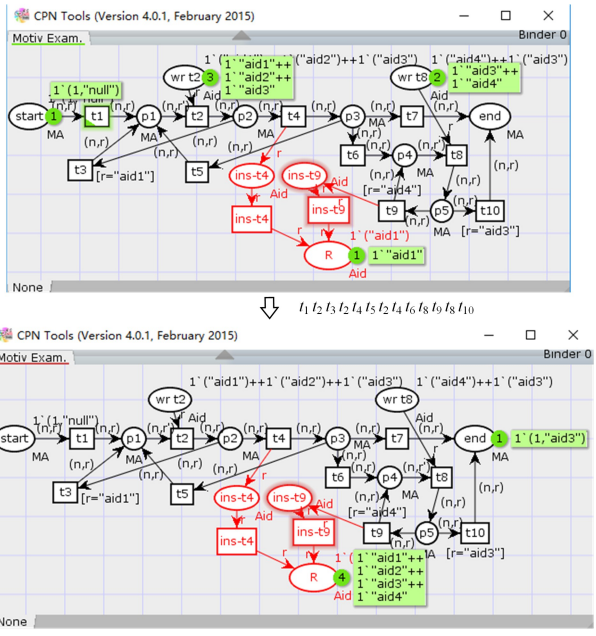
(2)输入给定的初始配置，执行每条可发生变迁的序列，在此过程中，按照算法 1 计算被写数据元素的精炼赋值集，如

给定初始配置 $c_0=([start],\{\perp\},\{(aid_1)\},\{\perp\})$，在执行变迁发生序列 $t_1t_2t_3t_2t_4t_5t_2t_4t_6t_8t_9t_8t_{10}$ 的过程中，aid 依次被 t_2，t_2，t_2，t_8，t_8 写入，即 aid 需精炼 5 次。

(3)检测每条变迁执行序列在每组数据精炼赋值下能否到达终点(即 end 库所)：若不能，则 WFT-net 不合理；若能，则验证终止状态的数据库是否满足所规定的数据需求：若不满足，则 WFT-net 不合理；若满足，记录所有变迁执行序列中的变迁，查看是否有死变迁：若有，则 WFT-net 不合理；否则，判定 WFT-net 是合理的。例如，$t_1t_2t_3t_2t_4t_5t_2t_4t_6t_8t_9t_8t_{10}$ 在“aid_1，aid_2，aid_3，aid_4，aid_3”这组数据精炼赋值下能到达终点，且数据库中不存在重复的元组，即满足数据需求，如图 4(b)所示。其他执行情况与此类似，此处不详尽列出。经验证，图 1 的 WFT-net 在动态数据精炼策略下是合理的。



(a)CPN Tools 中的 WFT-net



(b)变迁发生序列在一组数据精炼赋值下的执行结果

图 4 基于 CPN Tools 验证图 1 的合理性

Fig. 4 Soundness verification of Fig. 1 based on CPN Tools

5.2 实验数据与结果分析

本节首先基于文献[17]将工作流验证模型 WFT-net 与 WF-net 和 WFD-net 在表达和分析能力方面进行对比，结果如表 1 所列。WFT-net 模型考虑了数据的具体值和流程每次运行产生的数据日志，不仅克服了 WFD-net 模型以已定义或未定义来描述数据状态的局限性，还可以检测更多与数据需求相关的业务流程设计缺陷。

表 1 3 种工作流验证模型的比较

Table 1 Comparison of three workflow verification models

模型	WF-net	WFD-net	WFT-net
控制流	库所、变迁、弧	库所、变迁、弧	库所、变迁、弧
数据流	无	数据操作、数据约束	数据操作、数据约束、数据值、数据日志
逻辑错误	死锁、活锁、死变迁	死锁、活锁、死变迁	死锁、活锁、死变迁
数据错误	无	数据不一致、数据缺失、数据冗余、数据丢失、	数据不一致、数据缺失、数据冗余、数据丢失、与数据需求相关的业务流程缺陷
合理性	控制流合理(无死锁、无活锁、无死变迁)	控制流合理(无死锁、无活锁、无死变迁)	控制流合理(无死锁、无活锁、无死变迁)、数据流合理(满足数据需求)

本文实验数据来源于文献[17]中的 8 个测试实例，且根据前文检测结果对不合理的实例初步进行了故障修复，接着分别建立它们的 WF-net，WFD-net 和 WFT-net 模型，所得模型的基本信息如表 2 所列。每个实例的 WFD-net 和 WFT-net 的控制流都与其 WF-net 的控制流一致。因此，表 2 并未列出 WFD-net 和 WFT-net 的控制流规模。

表 2 实验用例

Table 2 Benchmarks

BM	WF-net			WFD-net		WFT-net		
	$\|T\|$	$\|P\|$	$\|F\|$	$\|\mathcal{D}\|$	$\|\mathcal{G}\|$	$\|\mathcal{D}\|$	$\|\mathcal{R}\|$	$\|\mathcal{G}\|$
BM1	16	11	32	5	10	5	2	10
BM2	17	13	36	4	6	4	1	8
BM3	19	13	38	4	8	4	2	12
BM4	17	13	34	5	6	5	2	6
BM5	10	9	20	3	4	3	1	4
BM6	16	12	32	5	8	5	1	8
BM7	13	10	26	3	4	3	1	6
BM8	19	12	37	3	8	3	1	8

为说明本文方法的有效性，我们在 CPN Tools 中进行了下列两组对比实验。首先，我们分别基于 WF-net，WFD-net 和 WFT-net(在动态数据精炼策略下)验证这些实例的合理性，结果如表 3 所列。

表 3 基于 WF-net，WFD-net 和 WFT-net 的验证结果对比

Table 3 Comparison of experimental results based on WF-net, WFD-net and WFT-net

BM	Sound			
	WF-net control flow	WFD-net control flow	WFT-net control flow	WFT-net data flow
BM1	yes	yes	yes	yes
BM2	no	yes	yes	yes
BM3	yes	yes	yes	yes
BM4	yes	yes	yes	yes
BM5	yes	yes	yes	yes
BM6	yes	yes	yes	yes
BM7	yes	yes	yes	yes
BM8	yes	yes	yes	yes

从表 3 可以看出，虽然这些实例都是合理的，但是，WF-net 和 WFD-net 都只从控制流进行验证，无法描述与验证系

统规定的数据需求。特别地，这些数据需求对于检查系统是否存在一些业务流程设计缺陷非常重要，尽管该系统流程始终能到达终止状态。因此，WFT-net 模型与其合理性检测方法是有意义的。

然后，在相同初始配置下，我们基于 WFT-net 模型又分别根据文献[17]所述静态数据精炼策略和本文所述动态数据精炼策略验证了这些实例的合理性，结果如表 4 所列。

表 4 静态和动态数据精炼策略下的验证结果对比

Table 4 Comparison of experimental results based on static and dynamic data refinement

BM	Sound on static data refinement			Sound on dynamic data refinement		
	control flow	data flow	WFT-net	control flow	data flow	WFT-net
BM1	yes	yes	yes	yes	yes	yes
BM2	no	—	no	yes	yes	yes
BM3	yes	yes	yes	yes	yes	yes
BM4	no	—	no	yes	yes	yes
BM5	yes	yes	yes	yes	yes	yes
BM6	yes	yes	yes	yes	yes	yes
BM7	yes	yes	yes	yes	yes	yes
BM8	no	—	no	yes	yes	yes

所有这些实例都是合理的。但是，在静态数据精炼策略下，BM2，BM4 和 BM8 都出现了死锁，即控制流不合理，此时即使数据流的合理性无法判断，但可以确定整个流程是不合理的。因此，本文提出的动态数据精炼策略是有意义的，能提高合理性验证的正确性。

结束语 对 WFT-net 模型而言，良好的数据精炼策略能正确反映流程的合理性。数据精炼只有适应流程状态的动态变化，才能完全捕捉流程所有可能的执行情况。在此基础上进行的合理性验证是可靠的，结果也是准确的。

由于数据精炼是依赖于数据库状态的，且精炼时数据库表中的元组越多，可达状态的比例就越大，这会导致状态爆炸；下一步将进一步优化数据精炼规则，精简数据元素的赋值个数，减少状态空间，以提升检测效率。

参 考 文 献

[1] DUMAS M, LA ROSA M, MENDLING J, et al. Business process management [M]. Berlin: Springer-Verlag, 2013.

[2] DEHNERT J, ZIMMERMANN A. On the suitability of correctness criteria for business process models [C] // International Conference on Business Process Management. Springer, Berlin, Heidelberg, 2005: 386-391.

[3] VAN DER AALST W M P. Structural characterizations of sound workflow nets [J]. Computing science reports, 1996, 96(23): 18-22.

[4] CLEMPNER J B. Classical workflow nets and workflow nets with reset arcs: using Lyapunov stability for soundness verification [J]. Journal of Experimental & Theoretical Artificial Intelligence, 2017, 29(1): 43-57.

[5] BI H H, ZHAO J L. Applying propositional logic to workflow verification [J]. Information Technology and Management, 2004, 5(3/4): 293-318.

[6] BARKAOUI K, BEN AYED R, SBAI Z. Workflow soundness verification based on structure theory of Petri nets [J]. International Journal of Computing and Information Sciences, 2007, 5(1): 51-61.

[7] COMBI C, OLIBONI B, WESKE M, et al. Conceptual modeling of inter-dependencies between processes and data [C] // Proceedings of the 33rd Annual ACM Symposium on Applied Computing. 2018: 110-119.

[8] TSOURY A, SOFFER P, REINHARTZ B I. Towards impact analysis of data in business processes [M] // Enterprise, Business-Process and Information Systems Modeling. Springer, Cham, 2016: 125-140.

[9] SUN S X, ZHAO J L. Formal workflow design analytics using data flow modeling [J]. Decision Support Systems, 2013, 55(1): 270-283.

[10] AWAD A, DECKER G, LOHMANN N. Diagnosing and repairing data anomalies in process models [C] // International Conference on Business Process Management. Springer, Berlin, Heidelberg, 2009: 5-16.

[11] MEDA H S, SEN A K, BAGCHI A. Detecting data flow errors in workflows: A systematic graph traversal approach [C] // 17th Annual Workshop on Information Technologies & Systems (WITS) Paper. 2007.

[12] SIDOROVA N, STAHL C, TRČKA N. Soundness verification for conceptual workflow nets with data: Early detection of errors with the most precision possible [J]. Information Systems, 2011, 36(7): 1026-1043.

[13] BORREGO D, ESHUIS R, GÓMEZ-LÓPEZ M T, et al. Diagnosing correctness of semantic workflow models [J]. Data & Knowledge Engineering, 2013, 87: 167-184.

[14] DE LEONI M, FELLI P, MONTALI M. A holistic approach for soundness verification of decision-aware process models [C] // International Conference on Conceptual Modeling. Springer, Cham, 2018: 219-235.

[15] WANG Z X, WANG J M, ZHU X C, et al. Verification of workflow nets with transition conditions [J]. Journal of Zhejiang University Science C, 2012, 13(7): 483-509.

[16] TAO X Y, LIU G J, YANG B, et al. Workflow nets with tables and their soundness [J]. IEEE Transactions on Industrial Informatics, 2019, 16(3): 1503-1515.

[17] SMITH G, DERRICK J. Verifying data refinements using a model checker [J]. Formal Aspects of Computing, 2006, 18(3): 264-287.

TAO Xiao-yan, born in 1987, Ph.D. Her main research interests include model checking, service computing and business process management.

YAN Chun-gang, born in 1963, postgraduate, Ph.D, professor, Ph.D supervisor. Her main research interests include computer collaboration and service computing, trusted computing and Petri net modeling and analysis.