

过程挖掘中一种能发现重复任务的扩展 α 算法

李嘉菲 刘大有 杨 博

(吉林大学计算机科学与技术学院 长春 130012)

(吉林大学符号计算与知识工程教育部重点实验室 长春 130012)

摘 要 基于 α -算法,提出了能发现工作流日志中重复任务的过程挖掘算法 α^{**} ,并给出了正确性证明.该算法先通过机器学习的方法分析重复任务的性质,给出了判定重复任务的定理并证明了其正确性;然后使用这些定理判断并标识出日志中的所有重复任务;最后,采用 α -算法从标识后的日志中提取出工作流网,并对其进行调整得到包含重复任务的工作流网模型.通过模拟实验验证了算法的有效性,与现有的重复任务挖掘方法的实验结果相比证实了文中提出的方法具有更高的效率.

关键词 过程挖掘;工作流挖掘;重复任务;Petri 网;工作流网
中图法分类号 TP311

Process Mining: An Extended α -Algorithm to Discovery Duplicate Tasks

LI Jia-Fei LIU Da-You YANG Bo

(College of Computer Science & Technology, Jilin University, Changchun 130012)

(Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun 130012)

Abstract Based on the α -algorithm, an improved process mining algorithm called α^{**} is presented and a proof of its correctness is also provided. First, the properties of duplicate tasks are analyzed through the techniques of machine learning, several theorems to judge the duplicate tasks and their proofs are given. Then, all the duplicate tasks in workflow logs are discovered and identified by them. Finally, the workflow net is extracted from the identified log using the α -algorithm and fine-tuned to get the result workflow model containing duplicate tasks. Experiments illustrate the validity of α^{**} -algorithm, the experiment results prove the higher efficiency of the α^{**} comparing with the existing duplicate tasks mining algorithm.

Keywords process mining; workflow mining; duplicate tasks; Petri nets; workflow nets

1 引 言

目前,在很多信息系统的日志文件中存在着大量与过程有关的数据,可人们对如何使用这些系统执行时记录的数据知之甚少,大多数的软件只能测

量某些关键的性能指标,无法从日志中获取有用的知识.过程挖掘是指从一组真正的过程执行信息中提取出结构化的过程描述的方法^[1],又称为工作流挖掘.它通常分为3个阶段:预处理、处理中和后处理阶段^[2].由于过程挖掘能从日志数据中发现知识,因此寻找准确有效的过程挖掘算法受到了研究者们

收稿日期:2007-01-23;修改稿收到日期:2007-06-05. 本课题得到国家自然科学基金重大项目(60496321)、国家自然科学基金(60373098, 60573073, 60503016)、国家“八六三”高技术研究发展计划项目基金(2006AA10Z245)、吉林省科技发展计划重大项目基金(20020303)、吉林省科技发展计划项目基金(20030523)和欧盟项目 TH/Asia Link/010(111084)项目基金资助. 李嘉菲,女,1976年生,讲师,博士研究生,研究方向为工作流技术、工作流挖掘及工作流管理等. E-mail: jiafei@jlu.edu.cn. 刘大有,男,1942年生,教授,博士生导师,主要研究领域为知识工程与专家系统、多 Agent、移动 Agent 系统与智能 Agent、数据挖掘、工作流技术及工作流挖掘等. 杨 博,男,1974年生,副教授,研究领域为数据挖掘、移动 Agent 系统与智能 Agent 等.

的广泛关注。

有关过程挖掘的研究主要集中在: 根据日志中记录的事件的二元顺序关系来挖掘一些启发式规则, 然后再使用这些规则从日志中提取过程模型。很多在实际 workflow 模型中普遍存在的复杂结构, 如重复任务(duplicate tasks)、隐藏任务(hidden tasks)、隐式库所(implicit places)和非自由选择结构(non-free-choices)等增加了过程挖掘的难度^[3]。

针对过程挖掘问题的特点, 研究者们采用多种方法进行了尝试。Agrawal^[4] 最早将过程挖掘引入到 workflow 管理的环境中。Cook 和 Wolf 在软件工程中考虑了类似的问题^[5], 他们给出有限自动机模型、马尔可夫和神经网络三种方法, 并且认为前两种方法比后一种方法更有前途, 但这些方法都是针对顺序过程的。在此工作的基础上, 他们还提出了基于概率的方法来发现日志轨迹中并行行为模式的方法^[6]。Herbst 和 Karagiannis 也在 workflow 的环境中使用归纳的方法完成了过程挖掘。他们基于隐马尔可夫模型提出了两种不同的归纳算法^[7]。一种是自底向上的, 从特殊到一般的方法, 另一种则采用自顶向下, 从一般到特殊的策略。这两种方法能挖掘出包含重复任务和并行行为的模型。Aalst 和他的研究小组的工作主要集中在挖掘并行过程上, 他们提出了能构建出 workflow 网形式的过程模型的 α -算法^[2]。

对比其它的算法, Aalst 的 α -算法证明了能准确挖掘出的模型种类, 这使得该算法的理论基础更加坚实, 在解决实际问题时更为有效。目前该算法及其扩展在医院、政府机关和移动通信系统等领域的实际应用取得了令人满意的结果。

α -算法是通过分析事件间可能存在的后继($>w$)、因果($\rightarrow w$)、并行($\parallel w$)和不相关($\#w$)四种依赖关系工作的, 算法要求使用的日志对于后继关系是完整且没有噪音的^[2]。这些依赖关系都是根据日志中的局部信息定义的, 这使得 α -算法无法处理非局部信息和非自由选择结构, 基于集合的工作方式也决定了它无法处理重复任务。针对 α -算法的局限性, de Medeiros 和 Wen 分别给出了改进算法。de Medeiros 分析了 α -算法的局限并提出了能处理短循环的算法^[2], Wen 在此基础上继续扩展, 给出了能处理非自由选择结构也能发现任务间的隐式依赖的方法^[8]。

在实际的模型中经常出现重复任务, 例如在某旅行社的预订流程中, 顾客可以只预订酒店, 也可以

只预订航班或者两种都预订。若用 A 代表“酒店预订”, B 代表“航班预订”, 则任务 A 和 B 都会在流程中重复出现。由于 α -算法采用集合来确定任务间的因果关系, 而在集合中不允许存在相同的元素, 因此在处理重复任务时, 算法将它们视为同一个任务, 也就挖掘不出正确的模型。机器学习的目的就是将数据库和信息系统中的信息自动提炼并转换成知识, 然后自动地加入到知识库中。机器学习的一般过程是建立理论、形成假设和进行归纳推理^[9]。利用机器学习的方法从日志数据中学习出能判定重复任务的知识后, 用其标识出日志中的重复任务, 再使用 α -算法就可以使标识后的重复任务都包含在集合中。这种新的结合方式使得现有的 α -算法能很容易地区分出重复任务, 从而可以完成对实际应用中大量包含重复任务的工作流模型的挖掘, 具有较大的理论和应用价值。Li^[10] 虽然提出了 α^* -算法来挖掘包含重复任务的工作流模型, 但没能处理选择/选择这种位置关系。

为此, 本文进一步扩展了 α -算法, 使其能挖掘出日志中的重复任务。首先, 通过机器学习的方法分析了重复任务的性质, 进而给出判定重复任务的定理并证明了其正确性。在此基础上, 提出一种可以从 workflow 日志中提取出包含重复任务的工作流网(Workflow nets)的方法—— α^{**} -算法, 之后给出算法的正确性证明, 并通过模拟实验验证了算法的有效性, 分析了算法的效率和性能。

本文第 2 节给出问题定义; 第 3 节介绍 α^{**} -算法并论证其正确性; 第 4 节以实验结果说明新算法的有效性; 最后总结全文的工作。

2 问题定义

一个任务有时可能在同一 workflow 模型中多次出现。此时, 多个任务具有相同的名称, 这样的同名任务就是重复任务^[3]。重复任务在实际的工作流模型中普遍存在, 例如, 图 1 中的模型 N_4 描述了前面提到的旅行社的预订流程。

重复任务可能出现在 workflow 网中的顺序、并行或选择结构中^[3]。因此, 考虑重复任务间的位置关系时, 若去掉重复的位置组合情况, 则共有 6 种关系, 分别是顺序/顺序、顺序/并行、顺序/选择、并行/并行、并行/选择和选择/选择。图 1 给出了这些位置关系的模型实例。实际上, 其他包含重复任务的模型都可以视为这 6 种位置关系的组合。

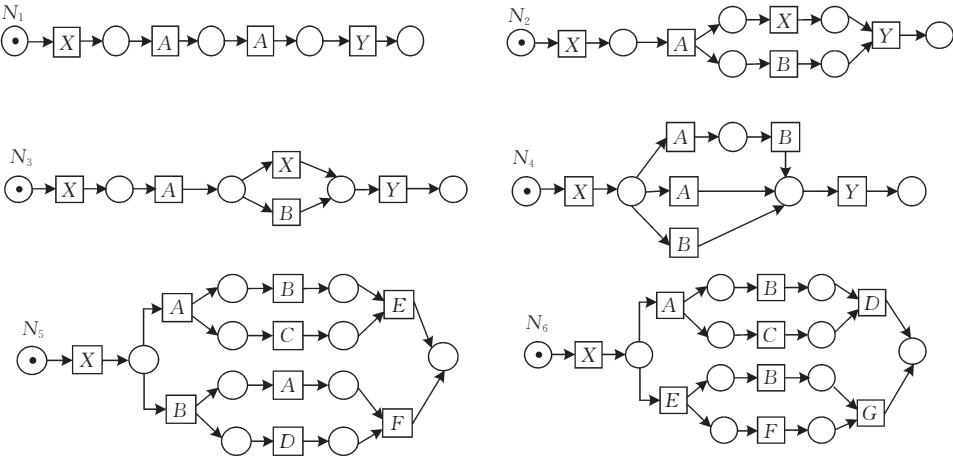


图 1 重复任务间可能存在的位置关系示例模型

3 挖掘重复任务的 α^{**} -算法

3.1 构造任务的前驱/后继表

α^{**} -算法首先要构造每个任务的前驱/后继表,该表可以用来帮助判定该任务是否属于重复任务.对于在每个工作流轨迹中出现的任务 A ,将从工作流日志中提取下列信息:①直接在任务 A 前面出现的任务的名称(用 T_p 标识);②直接在任务 A 后面出现的任务的名称(用 T_s 标识).这些信息都被保存在任务 A 的前驱/后继表中.

根据图 1 所示的过程模型,本文为每个模型分别模拟产生了一个随机的包含 1000 个工作流轨迹的工作流日志.图 2 给出的各个模型的日志中列举

了包含每个任务的各种可能出现情况的工作流轨迹,图 3 的前驱/后继表则给出了在 6 个日志中的有代表性的轨迹中出现的重复任务的前驱任务和后继任务.其中任务标识 $\delta_i(t,n)$ 表示任务 t 在工作流轨迹 δ_i 的第 n 次出现.例如, $\delta_1(X,1)$ 是指任务 X 在工作流轨迹 δ_1 的第一次出现.需要注意的是,图 1 中的模型 N_1, N_2 和 N_3 中出现的重复任务中的一个任务节点(node)属于顺序的事件流,而另一个则分别属于顺序、并行和选择事件流.而模型 N_5 和 N_6 中的重复任务 B 的一个节点则包含在一个并行的事件流中(任务 A 的 AND-split 结构),另一个则分别出现在选择和并行事件流中.模型 N_4 中的重复任务 A 的两次出现则都是在选择事件流中.

实例标识	事件轨迹
δ_1	X A A Y

(a) N_1 的日志 L_1

实例标识	事件轨迹
δ_1	X A B X Y
δ_2	X A X B Y

(b) N_2 的日志 L_2

实例标识	事件轨迹
δ_1	X A X Y
δ_2	X A B Y

(c) N_3 的日志 L_3

实例标识	事件轨迹
δ_1	X A B Y
δ_2	X A Y
δ_3	X B Y

(d) N_4 的日志 L_4

实例标识	事件轨迹
δ_1	X A B C E
δ_2	X A C B E
δ_3	X B A D F
δ_4	X B D A F

(e) N_5 的日志 L_5

实例标识	事件轨迹
δ_1	X A B C D
δ_2	X A C B D
δ_3	X E B F G
δ_4	X E F B G

(f) N_6 的日志 L_6

图 2 图 1 中模型的日志示例

从图 3(a)中可以看出:① $\delta_1(A,1)$ 和 $\delta_1(A,2)$ 的前驱任务不同;② $\delta_1(A,1)$ 和 $\delta_1(A,2)$ 的后继也不一样;③ $\delta_1(A,1)$ 和 $\delta_1(A,2)$ 出现在同一个轨迹 δ_1 中.对于图 3(b)~图 3(c)中的 $\delta_1(X,1)$ 和 $\delta_1(X,2)$ 以及图 3(b)的 $\delta_2(X,1)$ 和 $\delta_2(X,2)$ 可以得到类似的结论.图 3(b)中的信息还说明:① $\delta_1(X,2)$ 和 $\delta_2(X,2)$ 的前驱不同,后继也相异;② $\delta_1(X,2)$ 的前驱和后继与 $\delta_2(X,2)$ 的交叉相等;③ $\delta_1(X,2)$ 和 $\delta_2(X,2)$

在不同的工作流轨迹中出现.

图 3(e)描述了 N_5 中任务 B 的前驱和后继任务信息.从图 3(e)中可以发现:① $\delta_1(B,1)$ 的前驱任务和后继任务与 $\delta_4(B,1)$ 的完全不同;② $\delta_3(B,1)$ 和 $\delta_4(B,1)$ 的前驱相同而后继不同;③ 在事件轨迹 δ_3 中,任务 A 的后继任务是任务 D ;而在轨迹 δ_4 中 A 的前驱任务是 D ;④ $\delta_1(B,1)$ 的前驱和后继与 $\delta_2(B,1)$ 和 $\delta_3(B,1)$ 的交叉相等;⑤ $\delta_1(B,1), \delta_2(B,1),$

$\delta_3(B,1)$ 和 $\delta_4(B,1)$ 出现在不同的事件轨迹中.图3(f)中任务 B 在 workflow 轨迹中的性质与上面的分析结果类似.

图3(d)中的信息表明:① $\delta_1(A,1)$ 和 $\delta_2(A,1)$ 的前驱相同而后继不同;②在事件轨迹 δ_1 和 δ_2 中,任务 B 后紧跟着任务 Y ,但任务 Y 未直接出现在 B 前面;③ $\delta_1(A,1)$ 和 $\delta_2(A,1)$ 出现在不同的事件轨迹中.下一节本文将根据这些观察结果归纳出几条判定定理来识别日志中的重复任务.

任务标识	T_p	T_s
$\delta_1(A,1)$	X	A
$\delta_1(A,2)$	A	Y

(a) N_1 中任务A的前驱/后继表

任务标识	T_p	T_s
$\delta_1(X,1)$	\sim	A
$\delta_1(X,2)$	A	Y
$\delta_2(X,1)$	\sim	A

(c) N_3 中任务X的前驱/后继表

任务标识	T_p	T_s
$\delta_1(B,1)$	A	C
$\delta_2(B,1)$	C	E
$\delta_3(B,1)$	X	A
$\delta_4(B,1)$	X	D

(e) N_5 中任务B的前驱/后继表

任务标识	T_p	T_s
$\delta_1(X,1)$	\sim	A
$\delta_1(X,2)$	B	Y
$\delta_2(X,1)$	\sim	A
$\delta_2(X,2)$	A	B

(b) N_2 中任务X的前驱/后继表

任务标识	T_p	T_s
$\delta_1(A,1)$	X	B
$\delta_2(A,2)$	X	Y

(d) N_4 中任务A的前驱/后继表

任务标识	T_p	T_s
$\delta_1(B,1)$	A	C
$\delta_2(B,1)$	C	D
$\delta_3(B,1)$	E	F
$\delta_4(B,1)$	F	G

(f) N_6 中任务B的前驱/后继表

图3 图1模型中重复任务的前驱/后继表

3.2 判定重复任务

对出现在顺序结构中重复任务的判定相对容易,这是因为:在包含重复任务的工作流模型中,如果同名任务的前驱任务和后继任务总是不同的话,那么毫无疑问这是两个有相同名称的重复任务.反之,如果名称相同的任务拥有同样的前驱和后继任务,那么它们无论如何也不可能是两个不同的任务.不过,在并行或选择结构中,情况就变得复杂了.有时,即使同名任务有相同的前驱或者后继任务,也很难判定这种情形是由单一任务还是出现在选择结构中的重复任务产生的,例如模型 N_4 中的任务 A .在很多时候,虽然两个 workflow 轨迹中的同名任务也有不同的前驱任务和后继任务,但是我们却不能判定它们是否为重复任务.因为此时它们的前驱任务和后继任务可能存在交叉相等的情况,即该任务在一个轨迹中的前驱与它在另一个轨迹中的后继相同.这种情形既可以在单一任务属于并行分支时出现,也可以在有重复任务时发生.另一种可能的情形是

虽然不存在交叉相等,但是该任务前后都与一个选择结构相连,如图4所示 workflow 模型中的任务 E .这种情况可以定义为模型中的某些任务具有选择关系.上述这些情况都增加了判定重复任务的复杂性.下面我们给出选择关系的定义.

定义3(选择关系 \square_w). 设 W 是任务集 T 上的完整的工作流日志,即 $W \in \rho(T^*)$. a, b 是 T 中的任务,任务 a 和 b 具有选择关系,当且仅当 W 中存在轨迹 δ_1 和 δ_2 ,其中 $\delta_1 = t_1 t_2 t_3 \cdots t_n, \delta_2 = t'_1 t'_2 t'_3 \cdots t'_n$,且存在 $i \in \{1, 2, \cdots, n-2\}$ 满足 $t_i = a \wedge t'_i = b \wedge a \#_w b \wedge t_{i+1} = t'_{i+1} \wedge t_{i+2} = t'_{i+2}$.记以 $a \square_w b$.

图4中的任务 B 和 C 具有选择关系,因为 workflow 日志中存在轨迹 $\delta_1 = XABEFY$ 和 $\delta_2 = XACEFY$,且有 $i=3$ 满足 $t_3 = B \wedge t'_3 = C \wedge B \#_w C \wedge t_4 = t'_4 = E \wedge t_5 = t'_5 = F$.

在前一节中,图3(a)中任务 A 的前驱/后继表里的信息说明 $\delta_1(A,1)$ 和 $\delta_1(A,2)$ 是由两个不同的同名任务分别产生的,因为它们的前驱和后继任务完全不同,而且它们都属于同一个事件轨迹.在同一轨迹中,没有必要考虑交叉相等的情况,这是因为即使存在交叉相等,也一定是由重复任务而不是并行事件流中的单一任务引起的,例如图3(a)中的 $\delta_1(A,1)$ 和 $\delta_1(A,2)$.根据图3(b)和(c)中任务 X 的前驱/后继表里的信息,也可以对 $\delta_1(X,1)$ 和 $\delta_1(X,2)$ 得出类似的结论.另外,当两个同名任务出现在两个不同的轨迹中时,如果它们的前驱后继既不相同也不交叉相等,且它们的前驱任务间也不存在选择关系,则很明显它们是由重复任务产生的.图3(e)和(f)中 $\delta_1(B,1)$ 和 $\delta_4(B,1)$ 的信息就说明了这一点.从这些观察结果中可归纳出判定重复任务的定理1如下.

定理1. 设 $N=(P,T,F)$ 为一个合理的工作流网, W 是 N 的一完整的工作流日志, δ_i, δ_j 是 W 中的工作流轨迹,任务 $a \in T, a \in \delta_i$ 且 $a \in \delta_j, T_p$ 和 T_s 分别是任务 a 在轨迹 δ_i 中的前驱任务和后继任务, T'_p 和 T'_s 分别是任务 a 在轨迹 δ_j 中的前驱任务和后继任务, U 是重复关系集合.那么

- (1)若 $\delta_i = \delta_j, T_p \neq T'_p$ 且 $T_s \neq T'_s$,则 $\langle \delta_i(a, n_1), \delta_j(a, n_2) \rangle \in U$;
- (2)若 $\delta_i \neq \delta_j, T_p \neq T'_p, T_s \neq T'_s, T_p \neq T'_s, T_s \neq$

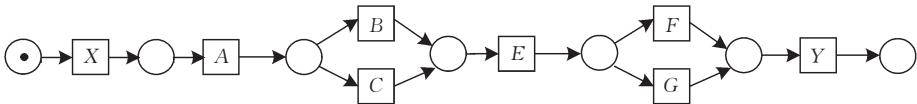


图4 选择关系示例图

T_P 且 $T_P \sqsubseteq_w T'_P$, 则 $\langle \delta_i(a, n_1), \delta_j(a, n_2) \rangle \in U$.

证明. 反证法. 假定 $\delta_i(a, n_1)$ 和 $\delta_j(a, n_2)$ 是相同的任务产生的. 下面分别证明, 两种情况下都将出现矛盾. 这里, 我们使用文献[11]中定义 3.2 给出的任务间的顺序关系.

(1) 设 $\delta_i = a_1 a_2 a_3 \cdots a_n, a_i = a$, 如果 $i \in \{2, \dots, n-1\}$, 则 $T_P = a_{i-1}, T_S = a_{i+1}$. 因为 $\delta_i = \delta_j$, 故 $T'_P = a_{i-1}, T'_S = a_{i+1}$. 这就意味着 $(T_P = T'_P)$ 且 $(T_S = T'_S)$. 与条件矛盾. $i \in \{1, n\}$ 时也有类似地结论.

(2) 由已知条件, 可知 $T_P > a, T'_P > a, a > T_S$ 且 $a > T'_S$. 往证 T_P 和 T'_P 的关系是 $\#$ 或 \parallel . 若 $T_P \rightarrow T'_P$, 由于 $T'_P > a$, 则存在一个实施序列(firing sequence), 它使得 T_P 在 T'_P 前实施(fire), 而实施 T'_P 后将直接实施任务 a . 此时在挖掘对应的 Petri 网模型时, 用来表示任务 T_P 的变迁后面可以紧跟着一个库所或两个以上库所, 这样可能产生两种结果, 或者 T_P 后不会直接出现 a , 或者任务 a 成为一个死任务. 这两种情况都不可能产生已知条件中的日志, 出现矛盾. 假定 $T'_P \rightarrow T_P$ 时的证明类似. 由文献[11]中的性质 3.3

可知 T_P 和 T'_P 的关系只能是 $\#$ 或 \parallel . 如果 $\delta_i(a, N_1)$ 和 $\delta_j(a, N_2)$ 是同一个任务产生的, 因为 $a > T_S$ 且 $a > T'_S$, 故日志中一定存在轨迹 $\delta_1 = t_1 t_2 t_3 \cdots T_P a T_S \cdots t_n$ 和 $\delta_2 = t'_1 t'_2 t'_3 \cdots T'_P a T'_S \cdots t'_n$. 此时 $T_P \sqsubseteq_w T'_P$, 与条件矛盾. 证毕.

在从图 2 中的日志提取的重复任务的前驱/后继表上应用定理 1, 得到图 5 的标识出重复任务的工作流日志. 图中的 A_1, X_1, B_1 就是标识后的重复任务. 比较图 5 的日志和图 1 的模型可以发现: 定理 1 已经准确地标识出部分重复任务, 如 N_1, N_2, N_3 和 N_6 中的重复任务, 但它却没能区分 N_4 和 N_5 中的重复任务. 实际上, 在出现交叉相等时, 如果能确定该任务属于某个并行的事件流, 那么就可以断定这两个工作流轨迹中的同名事件是由同一个任务产生的, 否则它们就与重复的两个任务相对应. 在讨论并行结构中任务的性质之前, 先定义两个函数 $pred(\delta, t)$ 和 $succ(\delta, t)$, 它们分别得到任务 t 在轨迹 δ 中的前驱任务和后继任务.

实例标识	事件轨迹
δ_1	$X A A_1 Y$

(a) N_1 的日志 LL_1

实例标识	事件轨迹
δ_1	$X A B X_1 Y$
δ_2	$X A X_1 B Y$

(b) N_2 的日志 LL_2

实例标识	事件轨迹
δ_1	$X A X_1 Y$
δ_2	$X A B Y$

(c) N_3 的日志 LL_3

实例标识	事件轨迹
δ_1	$X A B Y$
δ_2	$X A Y$
δ_3	$X B Y$

(d) N_4 的日志 LL_4

实例标识	事件轨迹
δ_1	$X A B C E$
δ_2	$X A C B E$
δ_3	$X B A D F$
δ_4	$X B D A F$

(e) N_5 的日志 LL_5

实例标识	事件轨迹
δ_1	$X A B C D$
δ_2	$X A C B D$
δ_3	$X E B_1 F G$
δ_4	$X E F B_1 G$

(f) N_6 的日志 LL_6

图 5 对图 2 中的日志应用定理 1 后的日志

在图 5(e) 的事件轨迹 δ_1 中, $\delta_1(B, 1)$ 的后继是任务 C , 而 $succ(\delta_1, C)$ 的结果是 E , 它正是 $\delta_2(B, 1)$ 的后继. 在 δ_2 中, $\delta_2(B, 1)$ 的前驱任务是 C , 而 $pred(\delta_2, C)$ 是任务 A 正是 $\delta_1(B, 1)$ 的前驱. 图 5(f) 中的 $\delta_1(B, 1)$ 和 $\delta_2(B, 1), \delta_3(B, 1)$ 和 $\delta_4(B, 1)$ 也有类似的性质. 可是, 对于图 5(e) 中的 $\delta_1(B, 1)$ 和 $\delta_3(B, 1)$, 虽然它们的前驱和后继也交叉相等, 但是由于它们不满足上面的性质, 所以它们是两个重复的任务 B 产生的. 根据这些观察结果可以得到判定重复任务的定理 2.

定理 2. 设 $N = (P, T, F)$ 为一个合理的工作流网, W 是 N 的一完整的工作流日志, δ_i, δ_j 是 W 中的工作流轨迹, 任务 $a \in T, a \in \delta_i$ 且 $a \in \delta_j, T_P$ 和 T_S 分别是任务 a 在轨迹 δ_i 中的前驱任务和后继任务, T'_P 和 T'_S 分别是任务 a 在轨迹 δ_j 中的前驱任务和后继任务, U 是重复关系集合. 那么

- (1) 若 $\delta_i \neq \delta_j, T_P = T'_S$ 且 $T'_P \neq pred(\delta_i, T_P)$, 则 $\langle \delta_i(a, n_1), \delta_j(a, n_2) \rangle \in U$;
- (2) 若 $\delta_i \neq \delta_j, T_P = T'_S$ 且 $T_S \neq succ(\delta_j, T'_S)$, 则 $\langle \delta_i(a, n_1), \delta_j(a, n_2) \rangle \in U$;
- (3) 若 $\delta_i \neq \delta_j, T_S = T'_P$ 且 $T_P \neq pred(\delta_j, T'_P)$, 则 $\langle \delta_i(a, n_1), \delta_j(a, n_2) \rangle \in U$;
- (4) 若 $\delta_i \neq \delta_j, T_S = T'_P$ 且 $T'_S \neq succ(\delta_i, T_S)$, 则 $\langle \delta_i(a, n_1), \delta_j(a, n_2) \rangle \in U$.

证明. 反证法. 假定 $\delta_i(a, n_1)$ 和 $\delta_j(a, n_2)$ 是相同的任务产生的, 即只有一个任务 a . 下面分别证明在以下 4 种情况都将出现矛盾. 这里, 我们仍使用文献[11]中定义 3.2 给出的任务间的顺序关系.

(1) 因为由 $T_P = pred(\delta_i, a), T'_S = succ(\delta_j, a)$, 可知 $T_P > a, a > T'_S$, 又 $T_P = T'_S$ 且只有一个任务 a 存在, 故可得到 $T_P \parallel a$. 此时 T_P 和 a 有相同的前驱任务. 即在轨迹 δ_j 中, a 的前驱是 T'_P , 它也是轨迹 δ_i

中 T_P 的前驱任务. 由此立即可得到 $T'_P = pred(\delta_i, T_P)$, 这与条件相矛盾. 得证.

(2) 因为由 $T_P = pred(\delta_i, a), T'_S = succ(\delta_j, a)$, 可知 $T_P > a, a > T'_S$, 又 $T_P = T'_S$ 且只有一个任务 a 存在, 故可得到 $a \parallel T'_S$. 此时 a 和 T'_S 有相同的后继任务, 即在轨迹 δ_i 中, a 的后继是 T_S , 它也是轨迹 δ_j 中 T'_S 的后继任务. 由此立即可得到 $T_S = succ(\delta_j, T'_S)$, 这与条件相矛盾. 得证.

(3) 证明与(1)的证明类似.

(4) 证明与(2)的证明类似. 证毕.

若 workflow 轨迹中的两个同名事件有相同的前驱或者后继任务, 定理 1 和定理 2 会将其判定为由同一任务产生, 如图 5(e) 中的 $\delta_3(B, 1)$ 和 $\delta_4(B, 1)$. 但是, 存在重复任务时也可能产生这种情形. 例如, 图 5(d) 中的 $\delta_1(A, 1)$ 和 $\delta_2(A, 1)$. 虽然前驱都为 X , 后继分别为 B 和 Y , 但是在图 1 对应的模型 N_4 中, 它们显然是两个不同的任务. 实际上, 在前驱相等时, 如果该任务是单一任务, 则在 Petri 网中用来表示该任务的变迁后面可以紧跟着一个库所或两个以上库所. N_3 和 N_5 中的任务 A 就说明了这两种情况. 此时, 它们的后继任务间的关系是 \parallel 或 $\#$. 根据文献[11]中的性质 3.3, 可知, 如果两个有相同前驱的同名事件的后继任务间是另外两种关系 \rightarrow 或 \rightarrow^{-1} , 则这两个同名事件与重复的两个任务相对应. 例如, 图 5(d) 中的信息显示任务 A 在轨迹 δ_1 和 δ_2 中的后继任务 B 和 Y 的关系是 $B \rightarrow Y$. 值得注意的是, 若去掉 N_4 中的任务 Y , A 的一个后继变为空时, 也能得到相同结论. 而在后继相同而前驱不同时, 亦可得到类似结论. 由此我们得到判定重复任务的定理 3 和定理 4.

定理 3. 设 $N = (P, T, F)$ 为一个合理的工作流网, W 是 N 的一完整的工作流日志, δ_i, δ_j 是 W 中的工作流轨迹, 任务 $a \in T, a \in \delta_i$ 且 $a \in \delta_j, T_P$ 和 T_S 分别是任务 a 在轨迹 δ_i 中的前驱任务和后继任务, T'_P 和 T'_S 分别是任务 a 在轨迹 δ_j 中的前驱任务

和后继任务, U 是重复关系集合, 那么

(1) 若 $T_P = T'_P, T_S \neq T'_S$ 且 $T_S = '\sim'$, 则 $\langle \delta_i(a, n_1), \delta_j(a, n_2) \rangle \in U$;

(2) 若 $T_P = T'_P, T_S \neq T'_S$ 且 $T'_S = '\sim'$, 则 $\langle \delta_i(a, n_1), \delta_j(a, n_2) \rangle \in U$;

(3) 若 $T_P = T'_P, T_S \neq T'_S$ 且 $T_S \rightarrow T'_S$, 则 $\langle \delta_i(a, n_1), \delta_j(a, n_2) \rangle \in U$;

(4) 若 $T_P = T'_P, T_S \neq T'_S$ 且 $T_S \rightarrow^{-1} T'_S$, 则 $\langle \delta_i(a, n_1), \delta_j(a, n_2) \rangle \in U$.

证明. 反证法. 假定 $\delta_i(a, n_1)$ 和 $\delta_j(a, n_2)$ 是相同的任务产生的, 即只有一个任务 a . 下面分别证明在以下 4 种情况都将出现矛盾. 这里, 我们仍使用文献[11]中定义 3.2 给出的任务间的顺序关系.

(1) 因为 $T_S = '\sim'$ 且 $T_S \neq T'_S$, 存在两个库所 $p_1, p_2 \in P, p_1$ 是汇库所, 而 $p_2 \in a \cdot \cap T'_S$. 在实施 a 后, a 将为 p_1 和 p_2 分别产生一个标记(token). 易知, 存在一个实施序列使得 T'_S 在 a 后实施而且 p_2 中的标识被送到汇库所 o 中. 这就是说, 汇库所中有两个标识. 故 N 不是安全、合理的. 与条件矛盾. 得证.

(2) 证明与(1)中证明类似.

(3) 由 T_S 和 T'_S 是 a 在不同轨迹中的后继任务知, $a > T_S$ 且 $a > T'_S$. 因为 $T_S \rightarrow T'_S$, 故存在一个实施序列使得 a 在 T_S 前实施, 且实施 T_S 后紧接着实施 T'_S . 此时可能产生两种情形, 或者 a 后不会直接出现 T'_S , 或者任务 T'_S 成为一个死任务. 这两种情况都不可能产生已知条件中的日志, 出现矛盾. 得证.

(4) 证明与(3)中证明类似. 证毕.

定理 4. 设 $N = (P, T, F)$ 为一个合理的工作流网, W 是 N 的一完整的工作流日志, δ_i, δ_j 是 W 中的工作流轨迹, 任务 $a \in T, a \in \delta_i$ 且 $a \in \delta_j, T_P$ 和 T_S 分别是任务 a 在轨迹 δ_i 中的前驱任务和后继任务, T'_P 和 T'_S 分别是任务 a 在轨迹 δ_j 中的前驱任务和后继任务, U 是重复关系集合. 那么

(5) 若 $T_S = T'_S, T_P \neq T'_P$ 且 $T_P = '\sim'$, 则 $\langle \delta_i(a, n_1), \delta_j(a, n_2) \rangle \in U$;

实例标识	事件轨迹
δ_1	$X A A_1 Y$

(a) N_1 的日志 DL_1

实例标识	事件轨迹
δ_1	$X A B X_1 Y$
δ_2	$X A X_1 B Y$

(b) N_2 的日志 DL_2

实例标识	事件轨迹
δ_1	$X A X_1 Y$
δ_2	$X A B Y$

(c) N_3 的日志 DL_3

实例标识	事件轨迹
δ_1	$X A B Y$
δ_2	$X A_1 Y$
δ_3	$X B_1 Y$

(d) N_4 的日志 DL_4

实例标识	事件轨迹
δ_1	$X A B C E$
δ_2	$X A C B E$
δ_3	$X B_1 A_1 D F$
δ_4	$X B_1 D A_1 F$

(e) N_5 的日志 DL_5

实例标识	事件轨迹
δ_1	$X A B C D$
δ_2	$X A C B D$
δ_3	$X E B_1 F G$
δ_4	$X E F B_1 G$

(f) N_6 的日志 DL_6

(6) 若 $T_S = T'_S, T_P \neq T'_P$ 且 $T'_P = \sim$, 则 $\langle \delta_i(a, n_1), \delta_j(a, n_2) \rangle \in U$;

(7) 若 $T_S = T'_S, T_P \neq T'_P$ 且 $T_P \rightarrow T'_P$, 则 $\langle \delta_i(a, n_1), \delta_j(a, n_2) \rangle \in U$;

(8) 若 $T_S = T'_S, T_P \neq T'_P$ 且 $T_P \rightarrow^{-1} T'_P$, 则 $\langle \delta_i(a, n_1), \delta_j(a, n_2) \rangle \in U$.

定理 4 的证明过程与定理 3 的类似. 略.

在图 2 的日志中提取的前驱/后继表上应用上面四条判定定理后, 可得到图 6 的工作流日志. 将该日志与图 1 中的过程模型作比较后, 可以看出图 1 中各工作流网模型中的重复任务已经都被准确地标识出来.

3.3 从标识的日志中生成工作流网

本文提出的处理工作流网中重复任务方案的主要工作集中在过程挖掘的预处理和后处理阶段, 且假定日志完整, 没有噪音. 在预处理阶段判定重复任务并用不同的标记对其进行标识. 判定某个任务为重复任务时, 使用上面的判定定理在该任务的前驱/后继表中搜索和判定, 搜索时按照该任务在每个事件轨迹中的出现顺序进行, 判定出该任务是重复任务的同时在日志中对其重新命名. 重命名的方法是在原来的任务名后添加一个序号, 例如 $B1, B2, 1A, 1B$. 为了使用方便, 序号和原来的任务名要从不同的字符集中选取. 在判定过程中, 那些已经被检查过但没有被重新命名的任务无需和它后面的同名任务作比较, 因为它前面的与其相同的任务已经同其后的同名任务比较过了. 类似地, 如果要检查的任务已经被标识过, 也没必要将其与原来的任务作比较. 上述做法都有助于提高算法的效率.

基于前面的判定定理, 本文扩展 α 算法, 提出能够挖掘出重复任务的 α^{**} -算法. 其中 T 为任务集合, W 是任务集 T 上的工作流日志, α 算法参见文献[11]中的定义 4.9, 顺序关系参见文献[11]中的定义 3.2. 具体算法如下.

算法 1. 挖掘重复任务的 α^{**} -算法.

输入: 任务集合 T , 工作流日志 W

输出: 工作流网 N

1. 根据工作流日志 W 计算出在 W 中出现的所有任务构成的集合 T_{\log} .
2. 将保存标识出重复任务的日志 W^{-DT} 初始化为 W .
3. 将描述是否存在重复任务的变量 $isDup$ 置为假.
4. 将描述是否要对日志 W^{-DT} 进行标识的变量 $isIdentify$ 置为假.
5. 对于 T_{\log} 中的每个任务 t , 进行如下操作:
 - 5.1. 根据任务 t 和日志 W^{-DT} 生成 t 的前驱/后继表 λ .

5.2. 对于表 λ 中每一行对应的元组 θ , 进行如下操作:

5.2.1. 将 λ 中那些与 θ 不同的元组抽取出来构成表 λ' .

5.2.2. 对于表 λ' 中每一行对应的元组 θ' , 进行如下操作:

5.2.2.1. 根据前面的定理 1 到定理 4 来判断元组 θ 和 θ' 中的任务 t 是否是重复任务, 结果在变量 $isDup$ 中保存.

5.2.2.2. 如果 $isDup$ 为真, 则将 t 的前驱/后继表 λ 中与元组 θ' 对应的元组中的任务 t 重命名为 t' , 然后将 t' 添加到 T_{\log} , 以便对 t' 进一步判定, 同时将变量 $isIdentify$ 置为真; 否则返回步 5.2.2.

5.3. 如果 $isIdentify$ 为真, 则根据已经标识的 t 的前驱/后继表对日志 W^{-DT} 进行标识, 结果保存在 W^{-DT} 中; 否则返回步 5.

6. 调用文献[11]中定义 4.9 的 α 算法对标识出重复任务的日志 W^{-DT} 进行挖掘, 提取的工作流网保存在 $(P_{W^{-DT}}, T_{W^{-DT}}, F_{W^{-DT}})$ 中.

7. 将最终结果的库所集合 P_W 置为 $P_{W^{-DT}}$.

8. 将 $T_{W^{-DT}}$ 中重新命名的重复任务的名称恢复到它们原来的名称后保存在 T_W .

9. 将 $F_{W^{-DT}}$ 中那些与重新命名的重复任务有关的那些弧连到具有原来的名称的任务上, 然后保存在 F_W .

10. 将 N 置为 (P_W, T_W, F_W) , 算法终止.

下面我们将通过两个定理证明算法 1 的正确性.

定理 5. 设 $N = (P, T, F)$ 为一个合理的工作流网, S 是 N 中重复任务的集合, D 为将 N 中的重复任务重新命名后的任务集合, $F_{-DL D}$ 为 N 中与重复任务有关的流关系集合, $F_{+DL D}$ 是将在 $F_{-DL D}$ 中的流关系中的任务修改为 D 中的名称后得到的流关系集合. 设 $N' = (P', T', F')$ 是一个 Petri 网, 其中 $P' = P, T' = T \cup D, F' = (F \setminus F_{-DL D}) \cup F_{+DL D}$. W 是 N 的一完整的工作流日志, W^{-DT} 是将 W 中的重复任务标识出后形成的工作流日志, 则有下列的结论成立:

(1) N' 是一个没有重复任务的合理的结构化工作流网.

(2) $\alpha(W^{-DT}) = N'$.

证明. 我们通过检查 N' 中没有重复任务而且是合理的、结构化的工作流网来证明结论(1), 通过说明 W^{-DT} 是 N' 的完整的工作流日志来证明结论(2).

(1) 由 N' 的定义知, N' 中不包含重复任务. 接着, 我们来证明 N' 的合理性. 根据定理 1 到定理 4, 易知 N 中的每个重复任务都被标识出来, N' 是通过将 N 中的重复任务重新命名得到的. 因此, 任何一个重新命名的重复任务的实施并不改变 N 的标识 (marking), 而且 N 的所有可达标识的集合与 N' 的

一致. 故, N 的合理性说明 N' 是合理的. 由 N' 和 N 可达标识集合的一致性, 易知 N' 是一个工作流网. 接下来只需证明 N' 是结构化的工作流网即可. 由于产生 N' 时, 只是重新命名了 N 中的重复任务而并没有改变 N 中库所和变迁间的结构关系, 且已知 N 是一个结构化工作流网, 故 N' 也是一结构化工作流网. 得证.

(2) 令 W' 是一完整的工作流日志. 由于 N' 和 N 的可达标识集合是相同的, 所以, 对于任何两个变迁 $a, b \in T' - D$, 若 $a >_w b$, 则 $a >_{w'} b$. 因为变迁 a 和 b 表示的都不是重复任务, 而且产生 W^{-DT} 时, 在 W 中并没有修改它们, 故 $a >_{w^{-DT}} b$ 成立. 若 $a \in D$ 或 $b \in D$, 则易找到其在 S 中的相应的原始变迁, 设它们为 a', b' . 因 $S \subset T$, 故 $a' >_w b'$. 由于 W^{-DT} 是通过将 W 中轨迹中的重复任务修改成 D 中的任务得到的, 所以重新标识 a' 和 b' 后, $a >_{w^{-DT}} b$ 仍然成立. 因此, W^{-DT} 是 N' 的一个完整的工作流日志. 前面已经证明 N' 是不包含重复任务的, 合理的结构化工作流网, 由文献[11]中的定理 4.10 知, $\alpha(W^{-DT}) = N'$. 证毕.

定理 6. 设 $N = (P, T, F)$ 为一个合理的工作流网, W 是 N 的一完整的工作流日志, 使用文献[11]中的定义 3.2 中的顺序关系, 有 $\alpha^{**}(W) = N$.

证明. 令 $\alpha^{**}(W) = (P_w, T_w, F_w)$. 已知日志 W 是完整的, 根据定理 1 到定理 4, 若 W 中存在重复任务, 则它们能被准确地标识出来. 因此, 易知 W^{-DT} 是与 W 相对应的、完整的且不包含重复任务的日志. 文献[11]中的定理 4.10 已经证明了 α 算法能从完整的工作流日志中挖掘出不包含重复任务的合理的工作流网. 设 $\alpha(W^{-DT}) = (P_{w^{-DT}},$

$T_{w^{-DT}}, F_{w^{-DT}}) = N'$, 根据定理 5, 可得 N' 是一个不包含重复任务的合理的结构化工作流网, 而且有 $P_{w^{-DT}} = P, T_{w^{-DT}} = T \cup D$ 和 $F_{w^{-DT}} = (F \setminus F_{-DL,D}) \cup F_{+DL,D}$ 成立. 显然, 由算法 1 的步 7, 有 $P_w = P_{w^{-DT}}$, 故 $P = P_w$. 接下来, 只需证明步 8 和步 9 能将任务集 $T_{w^{-DT}}$ 和流关系集 $F_{w^{-DT}}$ 中所有标识过的重复任务恢复成它们原来的名称即可. 可以通过使 $T_w = T_{w^{-DT}} \setminus D$ 及 $F_w = (F_{w^{-DT}} \setminus F_{+DL,D}) \cup F_{-DL,D}$ 来分别实现这两个步骤. 此时, 根据定理 5 可知 $T = T_w, F = F_w$. 因此有 $\alpha^{**}(W) = N$ 成立. 证毕.

4 实验及结果分析

本文算法的源程序采用 Microsoft Visual C++ 编写, 软件平台是 Windows 2000, 机器配置为 P4 (2.0GHz), 512Ram, 80GB 硬盘. 作者实现了包含 α^{**} -算法的实验原型. 该原型系统能读取工作流管理系统 Staffware 产生的文本形式的工作流日志文件. 实验使用的有些模型选自文献, 如模型 M1、M2 和 M3 就是 Herbst 使用的例子^[7], 有些模型是我们定义的, 如前面提到的 6 种位置关系的不同组合实例, 其他的模型来自于与启明信息技术股份有限公司的 ERP 合作项目. 实验时, 首先为每个模型产生了一个随机的包含 1000 个事件轨迹的工作流日志, 接着对这些日志数据进行学习. 图 7 给出了原工作流网和经 α^{**} -算法挖掘后的模型实例对比图, 由于篇幅限制, 这里只给出了模型 M1 和 M9 的挖掘结果. 通过比较可以发现, α^{**} -算法能识别出重复任务, 挖掘出与原来模型一致的工作流模型. 表 1 给出了具体的实验数据.

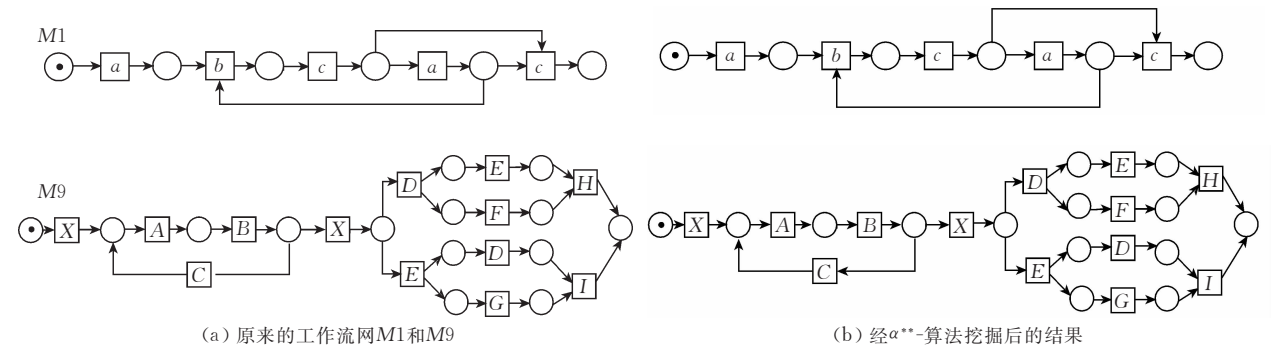


图 7 实验结果实例图

上述的实验结果说明, 对于 Herbst 使用的实验模型, 本文提出的 α^{**} -算法对比 Herbst 的合并方法具有更高的效率和更好的性能. 随着日志中事件标

识(event token)数目的增加, 挖掘模型所用的时间也逐渐变长. 工作流网中包含的重复任务数越多, 所需时间也越长. 模型 M4 未能被准确挖掘出来的原

因在于其先驱/后继表中的信息太少,使得算法无法从这些有限的数据中正确地学习. 模型 M8 未能被准确挖掘出来的原因在于其包含非自由选择结构,使得算法虽然能发现重复任务但无法处理模型中的

复杂结构. 表 1 和图 7 的实验结果表明,本文提出的 α^{**} -算法可以将大多数的工作流网还原成原来的模型,用时更少,这种先标识出日志中的重复任务,再对标识后的日志进行挖掘的算法是有效的.

表 1 实验结果

模型名称	事件标识数量	重复任务数量	结果模型数据 变迁数/库所数/弧数 (原始模型数据)	α^{**} -算法		Herbst 算法	
				时间/s	正确?	时间/s	正确?
M1	5975	2	5/6/12 (5/6/12)	0.600	是	0.8	是
M2	8750	1	5/6/11 (5/6/11)	0.670	是	0.8	是
M3	7250	1	9/11/22 (9/11/22)	0.830	是	2.0	是
M4	1350	2	2/3/5 (4/3/8)	0.161	否	—	—
M5	6000	1	10/12/24 (10/12/24)	0.603	是	—	—
M6	4000	1	6/5/12 (6/5/12)	0.562	是	—	—
M7	8240	2	12/13/30 (12/13/30)	0.816	是	—	—
M8	10050	3	12/14/33 (12/14/29)	1.182	否	—	—
M9	11300	3	13/14/30 (13/14/33)	1.293	是	—	—

5 结 论

本文针对工作流网中的任务重复性问题进行了研究,将机器学习技术与 α -算法相结合,提出了能挖掘出工作流网中重复任务的方法. 在归纳出 6 种重复任务间可能出现的位置关系的基础上,通过对几个包含这 6 种关系的结构复杂性不同且重复任务数目相异的工作流日志文件进行学习,得出了判定重复任务的定理并证明了其正确性,进而给出了能发现重复任务的过程挖掘算法及其正确性证明. 实验结果表明该方法是有用的,判定定理对重复任务的判断是正确的. 关于将来的工作,我们将对该算法进行扩展,使其能够处理不完整的且包含噪音的工作流日志,并设法获得更大量的实际数据对算法做进一步的验证和改进.

致 谢 作者诚挚地感谢审稿老师所提出的宝贵意见!

参 考 文 献

[1] van der Aalst W M P, van Dongen B F, Herbst J, Maruster L, Schimm G, Weijters A J M M. Workflow mining: A survey of issues and approaches. *Data and Knowledge Engineering*, 2003, 47(2): 237-267

[2] de Medeiros A K A, van Dongen B F, van der Aalst W M P, Weijters A J M M. Process mining: Extending the α -algorithm to mine short loops. *Eindhoven University of Technology*, Eindhoven: BETA Working Paper Series WP 113, 2004

[3] de Medeiros A K A, van der Aalst W M P, Weijters A J M M. Workflow mining: Current status and future Directions// Meersman R, Tari Z, Schmidt DC. *On the Move to Mean-*

ingful Internet Systems 2003: CoopIS, DOA, and ODBASE. Berlin: Springer-Verlag, 2003: 389-406

[4] Agrawal R, Gunopulos D, Leymann F. Mining process models from workflow logs//*Proceedings of the 6th International Conference on Extending Database Technology*, Valencia, Spain, 1998: 469-483

[5] Cook J E, Wolf A L. Discovering models of software processes from event-based data. *ACM Transactions on Software Engineering and Methodology*, 1998, 7(3): 215-249

[6] Cook J E, Du Zhi-Dian, Liu Chong-Bing, Wolf A L. Discovering models of behavior for concurrent workflows. *Computer in Industry*, 2004, 53(3): 297-319

[7] Herbst J, Karagiannis D. Integrating machine learning and workflow management to support acquisition and adaptation of workflow models. *International Journal of Intelligent Systems in Accounting, Finance and Management*, 2000, 9(2): 67-92

[8] Wen Li-Jie, Wang Jian-Min, Sun Jia-Guang. Detecting implicit dependencies between tasks from event logs//Zhou X, Lin X, Lu H et al. *Proceedings of the 8th Asia-Pacific Web Conference*. Lecture Notes in Computer Science 3841, Berlin: Springer-Verlag, 2006: 591-603

[9] Jiao Li-Cheng, Liu Fang, Gou Shui-Ping, Liu Jing, Chen Li. *Intelligent Data Mining and Knowledge Discovery*. Xi'an: Xidian University Press, 2006(in Chinese)

(焦李成,刘芳,龚水平,刘静,陈莉. 智能数据挖掘与知识发现. 西安:西安电子科技大学出版社,2006)

[10] Li Jia-Fei, Liu Da-You, Yu Wan-Jun. A process mining algorithm to discovery duplicate tasks. *Journal of Jilin University (Engineering and Technology Edition)*, 2007, 37(1): 106-110(in Chinese)

(李嘉菲,刘大有,于万钧. 一种能发现重复任务的过程挖掘算法. 吉林大学学报(工学版),2007, 37(1): 106-110)

[11] van der Aalst W M P, Weijters Ton, Maruster Laura. Workflow mining: Discovering process models from event logs. *IEEE Transactions on Knowledge Data Engineering*, 2004, 16(9): 1128-1142



LIU Da-You, born in 1942, professor, Ph. D. supervi-

LI Jia-Fei, born in 1976, lecture, Ph. D. candidate. Her research interests include workflow technology, process mining and workflow management.

sor. His research interests include knowledge engineering and expert system, multi agent, mobile agents system and intelligent agent, data mining, workflow technology and process mining, etc.

YANG Bo, born in 1974, associate professor. His research interests include data mining, mobile agents system and intelligent agent, etc.

Background

Process mining aims at a more fine grained analysis of processes based on event logs. The goal of process mining is to extract information about processes from these logs. Most research in process mining focuses on mining heuristics primarily based on binary ordering relations of the events in a workflow log. A lot of work has been done on utilizing heuristics to distill a process model from event logs and many valuable progresses are made in the domain. However, all the existing heuristic-based mining algorithms have their limitations. There are still many challenging problems that the existing mining algorithms cannot handle. Duplicate tasks are one of them.

The α algorithm theoretically constructs the final process model in WF-nets, which is a subset of Petri nets. This algorithm is proven to be correct for a large class of processes, but like most other techniques it has problems in dealing with duplicate tasks. In this paper, combining techniques of machine leaning and the α -algorithm, a new algorithm called α^{**} that can deal with duplicate tasks is proposed. First, the properties of duplicate tasks are analyzed through the techniques of machine learning, several theorems to judge the duplicate tasks and their proofs are given. Then, all the dupli-

cate tasks in workflow log are discovered and identified by them. Finally, the workflow net is extracted from the identified log using the α -algorithm and fine-tuned to get the result workflow model containing duplicate tasks. Experiments illustrate the validity of α^{**} -algorithm, the experiment results prove the higher efficiency of the α^{**} compared to existing duplicate tasks mining algorithm.

This algorithm can be used in various domains, e. g. , governmental agencies, municipalities, hospitals, ERP systems, etc. It will improve the efficiency and performance of the discovery of process with duplicate tasks.

This work is supported by NSFC Major Research Program, Basic Theory and Core Techniques of Non Canonical Knowledge (No. 60496321); National Natural Science Foundation of China (No. 60373098, 60573073, 60503016), the National High-Tech Research and Development Plan of China (No. 2006AA10Z245), the Major Program of Science and Technology Development Plan of Jilin Province (No. 20020303), the Science and Technology Development Plan of Jilin Province (No. 20030523), European Commission under grant No. TH/Asia Link/010 (111084).