DOI:10.13398/j.cnki.issn1673-260x.2019.10.022

基于流程挖掘的并行优化算法

邵叱风

(安徽理工大学, 安徽 淮南 232001)

摘 要:过程挖掘技术将观察到的行为(即事件日志)与建模的行为(如bpmn模型或petri网)联系起来.可以通过对日志进行挖掘,获得实际业务流程模型,再对其中关联严格结构检测,通过并行优化的方法优化实际业务流程模型.目前的业务流程优化主要是针对管理者或开发者给出的业务流程模型,其与实际运行中的业务流程可能与之有些许偏差,从而影响了优化结果.在此提出一种新方法对业务流程进行优化.首先使用Prom框架中的Alpha Miner及Heuristic miner获得流程模型的Petri Net及C-net结构; 然后使用最大分解的方法修复流程模型;针对修复后的模型提出其中高频简单网部分,并发现关联严格结构,使用Colored Petri nets对关联严格结构重新构造为关联并行结构,确定优化结果.该方法通过实际业务流程的研究案例及比较实验进行评估,其结果表明关联并行优化可明显缩短实际业务流程耗时.

关键词:Colored Petri nets;C-net;最大分解;关联严格;关联并行

中图分类号:TP391.9 **文献标识码**:A **文章编号**:1673-260X(2019)10-0066-05

1 引言

1.1 研究背景

随着大数据时代的到来,业务流程的管理拥有举足轻重的地位.业务流程管理的核心内容便是业务流程挖掘及优化,由此对流程挖掘及优化方面的深入研究日益重要.本文着重于修复和优化的方面.修复流程的目标是细化现有流程模型,使它们符合给定的事件日志.优化流程的目标是修改现有的流程模型,使其对应的实际业务流程更加合理、便捷.本文所考虑的修复及优化方法分别称为分解模型修复方法与并行优化算法.假设事件日志描述了正确的和最新的业务流程行为,而流程模型可能是错误的.模型修复后是最新的且正确的,可它表示的流程不一定是最合适的流程.在此,先以实际流程日志为基础,通过执行Prom平台中的Alpha Miner获得实际流程运行初始Petri Net模型, 并使用Heuristic miner获得实际流程运行C-net模型,通过分析绘制出最新流程模型,然后使用最大分解算法[1]对流程模型与初始日志进行三步操作:(1)将流程模型和事件日志分解为模型片段和子日志,(2)选择需要修复的片段,(3)使用流程发现算法修复选中的片段.获得修复模型后,进行三步操作:(1)将模型中高频部分提出,(2) 检测提取模型中关联严格结构,(3)使用并行优化算法优化提取的模型.最后得到符合实际业务流程的模型优化后的模型.

近年来,过程挖掘[2]是被广泛研究与应用的新兴技术,主要分为以下三个阶段:过程发现、服从性校验、模型完善[3].其中,流程发现是流程挖掘领域中最关键的问题之一,其目的是从事件日志中发现流程模型来描述日志[4]中观察到的行为.在文献中提出了许多流程挖掘技术,例如,在基于Alpha算法的抽取技术[5]和它的变体[6,8,9],启发式挖掘算法[10],基于状态空间搜索的遗传算法[11],基于语言的区域算法[12],基于状态的区域算法[13],复杂感知算法[14],基于正确块结构的算法[15],和无锁保证算法[16].

1.2 动机案例

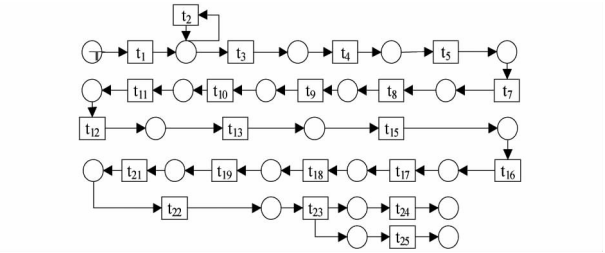

图1 政府在线采购系统流程图

收稿日期:2019-08-09

基金项目:国家自然科学基金项目(61402011,61572035);安徽省自然科学基金(1508085MF111,1608085QF149);安徽省高校自然科学基金重点项目(KJ2016A208)

通信作者:邵叱风(1995-),男,安徽省合肥市,研究生在读,主要研究方向为Petri网、过程挖掘及模型修复(fc_shao@126.com)

通过政府在线采购系统进行采购是当今政府常见的采购方式,本文使用政府采购的在线采购流程作为研究案例来分析所提出的方法,图 1 给出政府在线采购系统案例的给定模型 M.

由于政府采购系统是在不断完善中,所以实际的流程模型是不断变化的, 给定的模型 M 很难保证其是最新的且正确的系统流程模型,如果在此模型上对系统流程模型进行优化,难以达到预期的采购系统模型优化效果.

表 1 政府采购系统的最新事件日志

案列	可观测的迹	案例数
1	$t_1\ t_2\ t_3\ t_4\ t_6\ t_5\ t_7\ t_8\ t_9\ t_{10}\ t_{11}\ t_{12}\ t_{14}\ t_{13}\ t_{15}\ t_{16}\ t_{17}\ t_{18}\ t_{19}\ t_{20}$	23
2	$t_1\ t_3\ t_4\ t_6\ t_5\ t_7\ t_8\ t_9\ t_{10}\ t_{11}\ t_{12}\ t_{14}\ t_{13}\ t_{15}\ t_{16}\ t_{17}\ t_{18}\ t_{19}\ t_{20}$	32
3	$t_1\ t_2\ t_3\ t_4\ t_6\ t_5\ t_7\ t_8\ t_9\ t_{10}\ t_{11}\ t_{12}\ t_{14}\ t_{13}\ t_{15}\ t_{16}\ t_{17}\ t_{18}\ t_{19}\ t_{21}\ t_{22}\ t_{23}\ t_{24}\ t_{25}$	17
4	$t_1\ t_2\ t_3\ t_4\ t_6\ t_5\ t_7\ t_8\ t_9\ t_{10}\ t_{11}\ t_{12}\ t_{14}\ t_{13}\ t_{15}\ t_{16}\ t_{17}\ t_{18}\ t_{19}\ t_{21}\ t_{22}\ t_{23}\ t_{25}\ t_{24}$	11
5	$t_1\ t_2\ t_3\ t_4\ t_6\ t_5\ t_7\ t_8\ t_9\ t_{10}\ t_{11}\ t_{12}\ t_{14}\ t_{13}\ t_{15}\ t_{16}\ t_{17}\ t_{18}\ t_{19}\ t_{21}\ t_{22}\ t_{23}\ t_{24}\ t_{25}$	28
6	$t_1\ t_3\ t_4\ t_6\ t_5\ t_7\ t_8\ t_9\ t_{10}\ t_{11}\ t_{12}\ t_{14}\ t_{13}\ t_{15}\ t_{16}\ t_{17}\ t_{18}\ t_{19}\ t_{21}\ t_{22}\ t_{23}\ t_{25}\ t_{24}$	22
7	$t_1\ t_2\ t_2\ t_3\ t_4\ t_6\ t_5\ t_7\ t_8\ t_9\ t_{10}\ t_{11}\ t_{12}\ t_{14}\ t_{13}\ t_{15}\ t_{16}\ t_{17}\ t_{18}\ t_{19}\ t_{20}$	9
8	$t_1\ t_2\ t_2\ t_3\ t_4\ t_6\ t_5\ t_7\ t_8\ t_9\ t_{10}\ t_{11}\ t_{12}\ t_{14}\ t_{13}\ t_{15}\ t_{16}\ t_{17}\ t_{18}\ t_{19}\ t_{21}\ t_{22}\ t_{23}\ t_{24}\ t_{25}$	11
9	$t_1\ t_2\ t_2\ t_3\ t_4\ t_6\ t_5\ t_7\ t_8\ t_9\ t_{10}\ t_{11}\ t_{12}\ t_{14}\ t_{13}\ t_{15}\ t_{16}\ t_{17}\ t_{18}\ t_{19}\ t_{21}\ t_{22}\ t_{23}\ t_{25}\ t_{24}$	4

2 基本概念

定义 1 (Colored Petri nets)一个简单的(带有 k 种颜色的) 颜色 Petri 网是一个五元组 $\sum=(S,T;F;W,M)$其中(S,T;F)是一个网,$W:F\to\{0,1,2,\cdots\}^k$, $W:S\to\{0,1,2,\cdots\}^k$ 对 $t\in T$,如果 $s\in {}^gt\to M(s)\geqslant W(s,t)$,那么变迁 t 在标识 M 有发生权(M[t>),在标识 M 下发生变迁 t,产生一个新的标识 M′(M[t>M′),

$$M'(s)=\begin{cases}M(s)-W(s,t),若\ s\in {}^gt-t^g\\ M(s)+W(t,s),s\in t^g-{}^gt\\ M(s)-W(s,t)+W(t,s),若\ s\in {}^gtIt^g\\ M(s),其它\end{cases}$$

定义 2 (C-net)一个因过网是一个一元组 C=(A,a_i,a_0,D,I,O),其中:

·A⊆A 是有限的活动集合;

·$a_i\in A$ 是开始活动;

·$a_0\in A$ 是结束活动;

·D⊆A×A 是依赖关系;

·AS={X⊆P(A)|X={ø}∨ø∉X};

·I∈A→AS 定义了每个活动的可能输入绑定集;

·O∈A→AS 定义了每个活动的可能输出绑定集.

·使得

·$D=\{(a_1,a_2)\in A\times A|a_1\in U_{as\in I(a_2)}as\}$;

·$D=\{(a_1,a_2)\in A\times A|a_2\in U_{as\in I(a_1)}as\}$;

·$\{a_i\}=\{a\in A|I(a)=\{ø\}\}$;

·$\{a_o\}=\{a\in A|O(a)=\{ø\}\}$;;并且图(A,D)中的所有活动都在一条从 a_i 到 a_o 的路径上.

定义 3 (关联严格)Petri Net 模型对应实际业务流程,任意一个变迁均对应有实际操作,如果操作 B 必须有操作 A 的结果才可进行, 则称操作 A、B 对应变迁具有关联严格,记 CB(A,B).

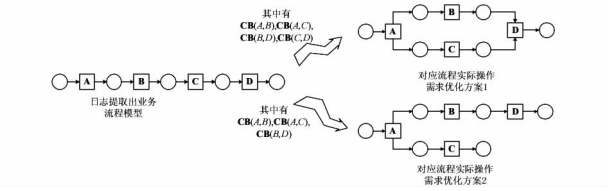

图 2 使用关联严格关系优化过程模型

如图 2 所示,其中 A,B,C,D 如图所示为严格序关系,但在实际流程操作中,操作 B 的前提为 A,操作 C 的前提为 A,操作 D 的前提为 B、C.分别记为 CB(A,B),CB(A,C),CB(B,D),CB(C,D).则可对流程模型进行方案 1 优化,并不会发生任何阻塞.情况 2 则可使用方案 2 进行优化.

3 流程挖掘及修复

本节基于表 1 政府采购系统的事件日志可以对现有的政府采购系统流程模型进行挖掘及修复.此处采用 Prom 平台中的 Alpha Miner 及 Mine for a Heuristics Net using Heuristics Miner 两个插件对日志进行挖掘处理,并使用最大分解算法对模型进行修复.

3.1 政府采购流程日志挖掘 (Alpha-algorithm、Heuristic miner)

对表 1 中的日志进行挖掘操作,此处(默认插件设置)可获得如图 3、4 所示 Petri net 流程模型及 C-net 流程模型.

由图 4 中 C-net 流程模型的 Fitness=0.999 可

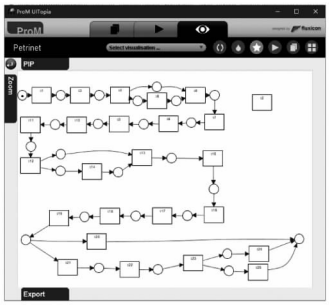

图 3　Alpha Miner 所获 Petri net 流程模型

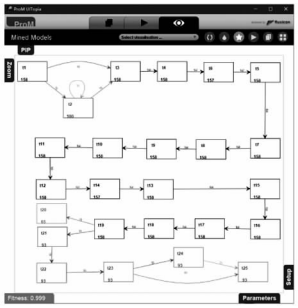

图 4　Heuristics Miner 所获 C-net 流程模型

知,此刻的因果网能极大程度上表达日志 1 中的流程.结合图 3 中 Petri net 流程模型,可得最新日志 1 中的政府采购流程模型如图 5 所示.此刻的初步流程模型相较于给定的模型 M 已经多出了 t_6、t_{14}、t_{20} 三个操作流程,可见保持最新且正确的流程模型是极其困难的.故在对系统流程进行优化时最好采用最新的日志进行挖掘,以获得相对更适应当前系统的流程模型.

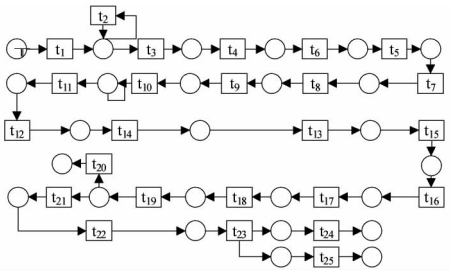

图 5　使用 prom 获得初步流程模型

3.2　政府采购流程日志修复(最大分解法)

对图 5 中的初步流程模型及表 1 中的日志使用最大分解法优化,如图 6 所示,此刻获得的实时流程模型在初始状态时,增加了 2 个 Token,根据实际业务流程分析可知,t_6,t_{14} 两个操作与另外两个系统有交互操作,相较于初步流程模型更加细化了.

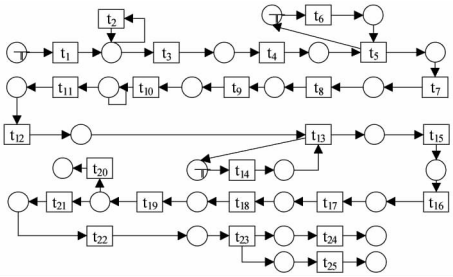

图 6　使用最大分解获得实时流程模型

4　业务流程中关联严格并行优化算法

过程模型的 Petri 网模型的性能分析通常从以下评估指标中观察,例如有效性,适应性,过程周期时间,效率和过程成本.过程周期时间和效率是相关的,并且难以测量有效性和适应性.故在此主要性能评价指标选用流程周期长度.在 Petri 网中主要依靠实体 token 的数据传递来实现时间参数的计算,其中业务流程平均周期时间计算公式为:

$$\overline{\text{Total.Time}}=\frac{1}{n}\sum_{j=1}^{n}\sum_{i=1}^{m}(\text{InToken.Time}+\text{Waiting.Time}_i+\text{Act.Time}_i)$$

其中 InToken.Time 初始为 0,在这里我们假设所有变迁在 token 输入库所后立刻发生,即 Waiting.Time=0.n 是流程个数,m 是每个流程的变迁个数.在模型优化过程中,分析变迁之间的关联严格关系,并设计如下所示的 ProcessOptimization 算法.

该算法包括流程模型中的变迁的重新组合,以及有色 Petri 网形式的输出,其中不同的颜色对应于不同的分支,通过并发的方式减少流程耗时.还有对颜色 petri 网 colorSet 以及对应颜色路径耗时的输出,最后使用冒泡排序对耗时进行排序输出.鉴于重新构造系统模型会改变原先流程,可能会使某些系统的开发维护产生大量工作,所以在保证优化效率及资源合理利用前提下,建议对流程中部分高频片段进行优化.

5　实验评估

并行优化算法主要思想即结合关联严格结构,重新构造为并发结构,以减少流程耗时.首先对任意模型(例如图 2 模型),设 InToken.Time=0 及 Waiting.Time=0,且任意 $\text{Act.Time}_i>0$,采用并行操作前的流程耗时为:

$$\overline{\text{Total.Time}}=\text{Act.Time}_A+\text{Act.Time}_B+\text{Act.Time}_C+\text{Act.Time}_D$$

优化方案 1 中流程耗时为:

$$\overline{\text{Total.Time}}=\text{Act.Time}_A+\max\{\text{Act.Time}_B,\text{Act.Time}_C\}\ \text{Act.Time}_D$$

显然

$$\max\{\text{Act.Time}_B,\text{Act.Time}_C\}<\text{Act.Time}_B+\text{Act.Time}_C$$

优化方案 2 中流程耗时为:

$$\overline{\text{Total.Time}}=\text{Act.Time}_A+\max\{(\text{Act.Time}_B+\text{Act.Time}_D),\text{Act.Time}_C\}$$

显然

$$\max\{(\text{Act.Time}_B+\text{Act.Time}_D),\text{Act.Time}_C\}<\text{Act.Time}_B+\text{Act.Time}_C+\text{Act.Time}_D$$

无论任意并发执行方案,均缩短了流程执行耗时.说明了并行优化的可行性.

```
Algorithm 1 Process Optimization
  Data: TimePetrinet N(S,T,F,Time);
  Result: CPN cpn ,cpn.TotalTime ;
1   Color New_Color ;
2   CPN New_CPN ;
3   cpn.TotalTime ;
4   colorSet ;
5   MaxTime ;
6   For each N.Transition in N:                    //Output optimization network structure
7       If (isCPNImpty(New_CPN)):
8           New_Color = createNewCPNColor();
9           New_Transition =creatTransitionWithColor(N.Transition, New_Color);
10          Add_Transition_To_CPN(New_Transition, New_CPN );
11      else
12          For each New_CPN.Transition in New_CPN:
13              If(isCombine(N.Transition, New_CPN.Transition))
14                  New_Transition =creatTransitionWithColor(Transition, New_CPN.Transition.Color);
15                  Add_Transition_To_CPN(New_Transition, New_CPN );
16                  breakout;
17              endIf
18          endEatch
19          If(isNotAdd(New_CPN)):
20              New_Color = createNewCPNColor();
21              New_Transition =creatTransitionWithColor(N.Transition, New_Color);
22              Add_Transition_To_CPN(New_Transition, New_CPN);
23          endIf
24      endIf
25  endEatch
26  For each cpn.Transition in cpn:                //Output color set
27      If(isEmpty(colorSet)):
28          Add_cpn.Transitionin.color_To_colorSet(cpn.Transition.Color);
29      else if(isNotIn(colorSet, cpn.Transition.Color))
30          Add_cpn.Transitionin.color_To_colorSet(cpn.Transition.Color);
31      endIf
32  endEatch
33  For each color in colorSet:                    //The output time corresponds to the color
34      color.Time ;
35      For each cpn.Transition in cpn:
36          if(isSame(color, cpn.Transition.color)):
37              color.Time+ = cpn.Transition.Time
38          endif
39      endEatch
40  endEatch
41  return cpn,cpn.totalTime                       //Use bubble sort output  MaxTime (cpn.TotalTime)
```

5.1 实验设置

对于图 6 使用最大分解获得的实时流程模型，其中的高频事件如图 7. 不妨取出如图 8 所示模型进行优化实验.其关联严格关系 CB(t_7,t_8),CB(t_7,t_9),CB(t_9,t_{10}),CB(t_8,t_{11}),CB(t_{10},t_{11}).

图 7　日志中事件频率

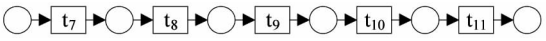

图 8　抽取高频部分待优化

5.2 实验结果分析

5.2.1 仿真实验

将高频部分代入算法进行优化，对于模型重构部分有如图 9 所示过程.

表 2　待优化场地安排流程模型的相关信息

变迁名称	代表含义	参数(周期/天)
t_7	签订委托协议	2–5 天
t_8	单一来源公示	15 天
t_9	场地预约	3 天
t_{10}	预约安排表	4–8 天
t_{11}	包段划分	2 天

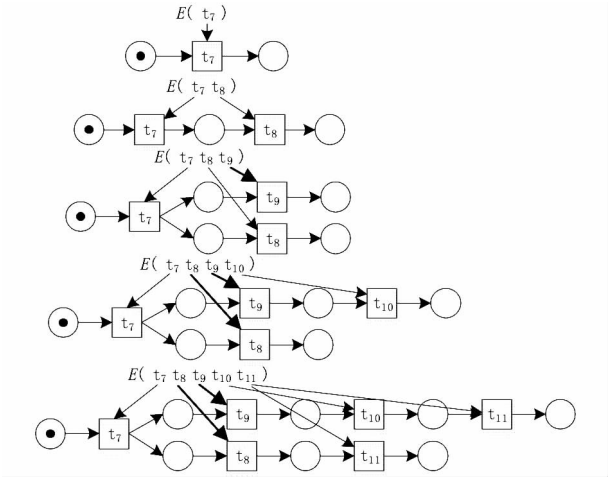

图 9　算法执行过程

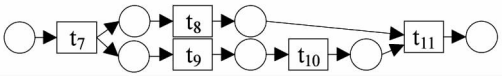

图 10　抽取高频部分优化结果

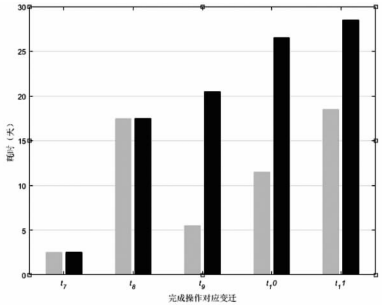

图 11　优化前后耗时对比

5.2.2 结果评估

使用表 2 中周期参数，使用计算公式

$$\overline{\text{Total.Time}}=\frac{1}{n}\sum_{j=1}^{n}\sum_{i=1}^{m}(\text{InToken.Time}+\text{Waiting.Time}_i+\text{Act.Time}_i)$$

计算，设 InToken.Time=0 及 Waiting.Time=0，可得如图 11 所示完成流程各阶段耗时.

选择处理时间减少率作为评估处理效率的指标，并且处理时间减少率用于反映处理优化之后整个处理时间减少的影响. 从上述模拟计算得出，优化前后图所示业务流程的平均周期分别为 28.5 天和 19.5 天，整体流程时间缩短率为 31.6%.因此，所采取的优化措施是正确的，有效地减少了图 3 的业务流程周期，并提高了流程执行的效率.因此，该过程

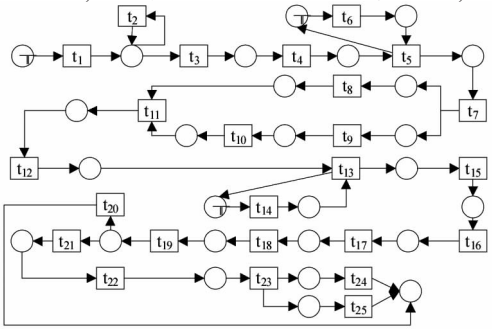

图 12　最终采购流程模型

的成本降低到一定程度,此优化算法是实际可行的.

最终优化模型如图 12 所示, 相较于给定的模型 M 增加了 3 个变迁及三个初始 Token,反映了当前系统与其他系统的交互,且在场地安排流程实行了优化,大大缩短了流程耗时.基于模型挖掘的并行优化算法在实际流程优化中是极符合优化要求的.

6 结束语

本文在已有研究的基础上,给出了基于过程挖掘的并行优化方法.首先使用 Prom 框架中的 Alpha Miner 及 Heuristic miner 获得流程模型的 Petri Net 及 C-net 结构;然后使用最大分解的方法修复流程模型;针对修复后的模型提出其中高频部分(模型优化可能会修改流程逻辑, 针对高频大量事件,减少系统二次开发工作量, 相对提高优化的实际意义),并发现关联严格结构,使用 Colored Petri nets 对关联严格结构重新构造为关联并行结构(通过不等式证明并发操作减少耗时的可行性), 确定优化后流程模型结构.

未来关于过程挖掘进行优化的研究还有很多工作去做.例如,在基于整数线性规划的约束体下,从包含大量事件的日志中挖掘满足需求的实时业务流程模型.

参考文献:

〔1〕Mitsyuk, A.A., Lomazova, I.A., Shugurov, I.S., and van der Aalst, W.M.P., Process model repair by detecting unfitting fragments, Supplementary Proceedings of AIST 2017, CEUR-WS.org, 2017.

〔2〕宋健,方贤文,王丽丽.基于流程切的过程模型挖掘方法[J].计算机科学,2019,46(7):1-7.

〔3〕VANDERASLST W, WEIJTERS T, MARUSTER L. Process Mining: Discovery, Conformance and Enhancement of Business Processes[M]. Springer-Verlag, Berlin, 2011.

〔4〕Wil van de. Aalst, Arya Adriansyah, Ana Karla Alves de Medeiros, Franco Arcieri, Thomas Baier, Tobias Blickle, Jagadeesh Chandra Bose, Peter van den Brand, Ronald Brandtjen, Joos Buijs, et al., Process mining manifesto, in: Business Process Management Workshops, Springer, 2012, pp.169 - 194.

〔5〕Wil van der Aalst, Ton Weijters, Laura Maruster, Workflow mining:Discovering process models from event logs, IEEE Trans. Knowl. Data Eng.16 (9) (2004) 1128 - 1142.

〔6〕L. Wen, J. Wang, W.M. Aalst, et al., A novel approach for process mining based on event types, J. Intell. Inf. Syst. 32 (2) (2009) 163 - 190.

〔7〕L. Wen, W.M. Aalst, J. Wang, et al., Mining process models with non-free-choice constructs, Data Min. Knowl. Discov. 15 (2) (2007)145 - 180.

〔8〕L. Wen, J. Wang, J. Sun, Mining invisible tasks from event logs, in: Advances in Data and Web Management, Springer, 2007, pp. 358 - 365.

〔9〕L. Wen, J. Wang, W.M.P. van der Aalst, B. Huang, J. Sun, Mining process models with prime invisible tasks, Data Knowl. Eng. 69 (10) (2010) 999 - 1021.

〔10〕A.J.M.M. Weijters, Wil M.P. van der Aalst, A.K. Alves De Medeiros, Process mining with the heuristics miner algorithm, in: Tech. Rep, Vol. 166, Technische Universiteit Eindhoven, WP, 2006, pp. 1 - 34.

〔11〕A.K. Alves De Medeiros, A.J.M.M. Weijters, Wil M.P. van der Aalst, Genetic process mining: a basic approach and its challenges, in: Business Process Management Workshops, Springer, 2006, pp. 203 - 215.

〔12〕Robin Bergenthum, J?rg Desel, Robert Lorenz, Sebastian Mauser, Process mining based on regions of languages, in: Business Process Management, Springer, 2007, pp. 375 - 383.

〔13〕Josep Carmona, Jordi Cortadella, Michael Kishinevsky, A region-based algorithm for discovering petri nets from event logs, in: Business Process Management, Springer, 2008, pp. 358 - 373.

〔14〕Chathura C. Ekanayake, Marlon Dumas, Luciano Garc?a-Bañuelos, Marcello La Rosa, Slice, mine and dice: Complexity-aware automated discovery of business process models, in: International Conference on Business Process Management, Springer, 2013.

〔15〕Sander J.J. Leemans, Dirk Fahland, Wil M.P. van der Aalst, Discovering block-structured process models from incomplete event logs, Lecture Notes in Comput. Sci. 8489 (2014) 311 - 329.

〔16〕Adriano Augusto, Raffaele Conforti, Marlon Dumas, Marcello La Rosa, Split miner: discovering accurate and simple business process models from event logs, in: IEEE International Conference on Data Mining, 2017.