含噪音处理的工作流挖掘算法在资源规划方面的应用①

黄晓瑜[1], 黄晓雯[2]

[1](福州大学 经济与管理学院, 福州 350116)
[2](厦门大学 统计系, 厦门 361005)

摘 要: 为了提升业务流程执行效率, 提高流程建模的客观性, 在考虑日志文件存在噪音的前提下, 对结构化工作流模型的四种基本结构进行分析, 研究从日志文件中挖掘出流程模型的结构化算法. 从获取日志文件信息、提炼简单活动序列、挖掘结构化分支模型和合并最终模型四个部分进行描述, 然后以 Petri 网的形式展现最终模型, 最后进行资源负载分析, 为决策者进行资源规划提供决策依据, 有利于提高资源利用率和流程执行效率, 进而提高企业效益.

关键词: 工作流挖掘; Petri 网; 噪音; 事件日志; 结构化工作流网

Workflow Mining Algorithm with Noise Processing Applied in Resources Planning

HUANG Xiao-Yu[1], HUANG Xiao-Wen[2]

[1](School of Economics and Management, Fuzhou University, Fuzhou 350116, China)
[2](Department of Statistics, Xiamen University, Xiamen 361005, China)

Abstract: To improve the efficiency of business process and the objectivity of process modeling, we analyse four basic structures of structural workflow net under the premise of the existence of noise in the log file, furthermore, mining algorithm of a structured process model from the log file was researched. This paper expatiates in four parts, includeinggetting information from log file, extracting all simple paths, mining structured branch model and merging final model, finally it shows the model in the form of Petri nets. On the basis of that, the analysis of resource is proposed, it offers the supporting decision tools for decision-makers in resource planning and improves the efficiency of resource utilizing, eventually it increases the efficiency of enterprises.

Key words: workflow mining; Petri Net; noise; event Logs; structural workflow net

在这个信息高度膨胀的时代, 基于信息流通需要, 大部分企业选择使用 ERP、CRM、SCM、WFM 等先进的业务管理软件系统来管理业务信息, 在这些系统中, 管理者可以根据特定的业务项目来配置流程运行的顺序, 然后通过图形化的显示方式来指导操作者执行, 如何建立完整的流程模型来正确描述业务过程, 是当前亟需解决的一个重要问题[1].

在传统流程建模方法中, 设计者将重点放在工作流设计和配置阶段, 而对于工作流诊断这一环节, 很少组织分析系统的运行数据, 此外, 大部分流程设计者来源于组织外部, 他们熟练掌握工作流语言, 但是却不熟悉整个业务流程的运行过程, 他们对业务的认知一般来自管理者的描述, 导致他们对业务流程的了解具有很强的主观性, 从而影响对业务流程的正确建模, 此外, 传统建模方法过于注重流程自动化的实现, 往往忽略了工作流技术的柔性和诊断正误, 导致建模结果与真正的流程产生偏差. 作为业务流程建模的工具之一, 流程挖掘能够有效解决传统建模的主观臆断问题, 从而为企业业务流程再造提供支持.

工作流挖掘(Workflow Minning)又名流程挖掘(Process Mining), 指从实际执行集合中提取出结构化过程描述, 通过分析日志信息建立流程模型, 以实现工作流的模型优化、智能管理和柔性管理[3]. 传统的挖掘算法都假定日志文件是完整的, 不存在噪音, 本文

① 收稿时间:2015-04-12;收到修改稿时间:2015-05-28

在考虑日志文件存在噪音的前提下, 以卫浴行业的水龙头生产加工流程为例, 通过在结构化挖掘算法中设置阈值来过滤噪音, 对结构化工作流模型的四种基本结构进行分析, 从而挖掘出日志文件中的业务流程, 然后以 Petri 网的形式展现出来, 并在此基础上对流程中涉及的设备进行资源负载分析, 以提高设备资源利用率和生产效率, 由于本文是在考虑噪音的基础上挖掘流程, 即不要求流程是正确的且完整的, 更符合现实情况, 故适用范围相对更广.

1　相关研究

流程挖掘的定义最早是在 1998 年由 R.Agrawl, D.Gunopolos 及 F.Leymann 提出的. 目前国际上比较认可的流程挖掘定义为: 流程挖掘是指那些从实际执行集合中提取出结构化流程描述的方法[2]. 从流程挖掘的定义中可以看出流程挖掘的目的是从日志数据中获取信息, 并且建立清晰的流程模型, 同时要保证构建的流程模型和实际的业务流程的一致性.

事实上, 从日志文件中挖掘出业务流程模型的方法并不是新兴的方法, Alast 和 Weijter 等人最早提出用 α 算法来从工作流日志文件中挖掘出流程模型, 并利用 Petri 网来表示最终模型[5], 但是, α 算法可以挖掘的流程模型有限, 因此文章[6]在 α 算法的基础上又提出一种 α*算法, 该算法可以挖掘出业务流程中的重复任务, 但是在挖掘流程模型时这些算法并没有进一步分析和处理每个业务节点所包含的资源等信息, 文献[7]介绍了一种流程挖掘工具, 该工具同样基于日志文件, 它可以读取可扩展标记语言(Extensible Markup Language), 即.XML 格式的日志文件, 并利用 Petri 网作为模型输出工具, 但是该文章只是初步分析了基于日志文件的流程挖掘方法, 且假设算法对噪音具有鲁棒性, 排除了噪音的影响.

2　基础知识

对于工作流的定义尚未完全统一, 工作流管理联盟将其定义为: 工作流是一类能够完全或者部分自动执行的经营过程, 根据一系列过程规则, 使文档、信息或任务能够在不同的执行者之间传递和执行[4]. 接下来给出 Petri 网和 SWF 的定义, 以便做进一步应用分析.

定义 1. Petri 网. 一个 Petri 网是一个三元组(P,T,F):

P 是有限个库所(用长方形表示)的集合; T 是有限个变迁(用圆形表示)的集合($P\cap T=\varnothing$); 并且 $F(P\times T)\cup(T\times P)$是弧的集合(流关系).

库所 p 称作变迁 t 的输入库所, 当且仅当存在一个从 p 到 t 的有向弧. 库所 p 称作变迁 t 的输出库所, 当且仅当存在一个从 t 到 p 的有向弧. 使用·t 表示变迁 t 的输入库所集合, 符号 t·、·p、p·有类似的含义.

定义 2. SWF 网[8]. WF 网 N=(P, T, F)是结构化工作流网(SWF 网)当且仅当:

(1)对所有的 $p\in p$ 且 $t\in T$, $(p,t)\in F$: $|P\cdot|>1$ 则$|.t|=1$;

(2)对所有的 $p\in P$ 且 $t\in T$. $(p,t)\in F$: $|\cdot t|>1$ 则$|.p|=1$;

(3)不存在隐式库所.

定义 3. 基于日志顺序关系. 设 W 是任务集 T 上的事件日志, 即 $W\in P(T^*)$, 事件日志是一个四元组 $W=(T_W,O_w,t_{wi},t_{wo})$, $T_w=\{t_{wi},t_{wo},t_1,\cdots,t_n\}$, 其中 t_{wi} 是事件日志的起始任务, t_{wo} 是事件日志的结束任务, $O_w=\{>_w,\to_w,\#_w,\|_w\}$ 是元素之间的关系.

设 $a,b\in T$:

① $a>_w b$ 当且仅当存在一个路径 $\sigma=\langle t_1,t_2,t_3,\ldots,t_n\rangle$, $i\in\{1,2,\cdots,n\}$, 满足 $\sigma\in W$ 且 $t_i=a$, $t_{i+1}=b$;

② $a\to_w b$ 当且仅当 $a>_w b$ 且 $b\not>_w a$;

③ $a\#_w b$ 当且仅当 $a\not>_w b$, 且 $b\not>_w a$;

④ $a\|_w b$ 当且仅当 $a>_w b$ 且 $b>_w a$.

3　含噪音处理的结构化工作流挖掘算法的基本思想

本文基于日志文件提出的含噪音处理的结构化工作流挖掘算法的基本思想是从工作流网的运行日志中挖掘出所记录的业务模型, 然后使用 Petri 网展现最终模型, 为进一步进行资源规划提供依据. 其基本框架如图 1 所示.

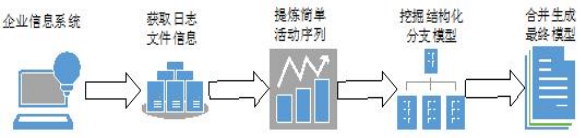

图 1　含噪音处理的结构化流程挖掘的基本框架

首先从业务管理信息系统的数据库中导出企业实际执行的工作流日志文件, 分析日志文件, 合并重复出现的序列, 提炼日志文件中所有简单活动序列, 过滤日志噪音对流程执行的影响, 用结构化挖掘算法挖

掘出顺序、选择、循环和并行四种基本结构, 最后合并四种基本结构生成最终模型, 并用 Petri 网表示.

4 含噪音处理的结构化工作流挖掘算法应用

本文以卫浴行业为背景, 结合水龙头生产流程模型实例, 根据图 1 中所示的基本框架来阐述在考虑噪音的前提下, 利用结构化挖掘算法从事件日志中挖掘出工作流模型的方法.

4.1 水龙头生产流程模型简介

水龙头生产加工流程涉及合金溶解、造模、铸造、机加、抛光和外涉电镀等多道工序, 本文只考虑铸造、机加、抛光和外涉电镀四个过程. 流程简单描述如下: 首先是铸造, 将熔融后的铜料浇注于模具内, 低档水龙头用翻砂浇铸, 高品质龙头用重力铸造, 待冷却凝固后开模卸料清理浇冒口, 将冷却的铸造坯件进行自检并送入落砂机滚筒陶砂清理, 合格品继续加工, 不合格品回炉熔融成铜合金水, 下一步将坯件放入抛丸机研磨整光; 然后进行机加工序, 首先选择夹具刀具和工件进行调模试加工, 经过首件检验合格后正式批量生产, 过程中操作者进行自检, 检验员巡检, 完工后进行全数检验, 合格品才可流到下道工序进行试压检验; 接下来是抛光, 用研磨料和砂带把表面磨细和修整外形轮廓; 最后进行电镀, 电镀前要检查龙头的杂质含量, 才能进行镀铜、镀镍、镀铬, 增强龙头的耐腐耐磨性, 再用 24 小时乙酸盐雾试验进行检验, 完成后即可进行最后的组装包装工序.

4.2 工作流挖掘前提假设

为了能利用工作流挖掘进行流程建模, 需要作出以下假设:

(1)企业使用的业务管理系统详细记录了业务流程日志(活动节点名、业务人员出入系统的时间、活动的起止时间、设备使用情况等)

(2)流程中所有的节点名称都存储在排好顺序的事件表中

(3)结构化日志对于每一个流程只存储一个开始点和一个结束点[9]

(4)挖掘算法能够准确地识别活动的循环、顺序、选择、并行等 4 种基本结构. 而且可以把这四种结构的算法转变成结构化的数据信息

(5)企业中与业务流程相关的信息都储存在系统中, 并记录在日志文件中

(6)形成顺序关系的两个活动不应有时间间隔, 即后面活动的开始时间与前面活动的结束时间相同

4.3 工作流挖掘算法应用

4.3.1 获取日志文件信息

TinyXML 是一个开源 XML 解析库, 简单易用且小巧玲珑, 根据可扩展标记语言的特点, 本文使用 TinyXML 文件解析库编写一段 c++算法来读取.XML 格式的日志文件. 图 2 为水龙头生产加工流程模型的部分日志文件信息, 算法 1 能够读取并分析日志文件(TapProcess.xml)中的数据信息.

```
<ProcessInstance id="1" description="">
<AuditTrailEntry>
    <WorkflowModelEvent>Foundry casting</ WorkflowModelEvent>
    <EventType>start</ EventType>
    <Timestamp>2013-04-08T08:30:00.000+08:00</ Timestamp>
    <Originator>Foundry Mold</ Originator>
</ AuditTrailEntry>
<AuditTrailEntry>
    <WorkflowModelEvent>Foundry casting</ WorkflowModelEvent>
    <EventType>complete</ EventType>
    <Timestamp>2013-04-12T15:30:00.000+08:00</ Timestamp>
```

图 2 水龙头生产加工流程模型的部分日志文件信息

算法 1

```
输入: .XML 格式的日志文件
#include <iostream>
#include "tinystr.h"
using namespace std;
bool ReadXmlFile(string& TapProcess.xml)
{ try
    {
    CString appPath = GetAppPath();
    string seperator = "\\";
    string fullPath = appPath.GetBuffer(0) +seperato
    r+ TapProcess.xml;
    TiXmlDocument *myDocument = new TiXmlDo
    cument(fullPath.c_str());
    myDocument->LoadFile();
TiXmlElement *RootElement = myDocument->RootE
lement();
cout << RootElement->Value() << endl;
get the attribute of first Element . 获得第一个节点的
属性信息
output the attribute of first Element . 输出第一个节
```

点的属性信息
}输出: 日志文件数据信息

ID	WorkflowModelEvent	EventType	Timestamp	Originator
1	Foundry casting	start	2013 04 08 08 22 00	Foundry Mold
1	Foundry casting	complete	2013 04 12 15 30 00	Foundry Mold
1	Cutting	start	2013 04 13 08 17 00	Shakeout machine
1	Cutting	complete	2013 04 13 11 29 00	Shakeout machine
1	Polishing	start	2013 04 13 13 32 00	Shot-blasting machine
1	Polishing	complete	2013 04 13 15 30 00	Shot-blasting machine
1	Machining	start	2013 04 13 16 38 00	Metal Cutting machine
1	Machining	complete	2013 04 13 15 30 00	Metal Cutting machine
1	Inspection	start	2013 04 15 07 25 00	Compression-testing machine
1	Inspection	complete	2013 04 15 09 30 00	Compression-testing machine
1	Grind	start	2013 04 15 07 25 00	Grinding Belt
1	Grind	complete	2013 04 15 09 30 00	Grinding Belt
1	Grind	start	2013 04 15 07 25 00	Grinding Belt
1	Grind	complete	2013 04 15 09 30 00	Grinding Belt
1	Polishing	start	2013 04 15 10 05 00	Polishing machine

图 3 从日志文件中挖掘出来的部分信息

通过算法可以从日志文件中挖掘出部分节点的信息如图 3 所示. 为方便描述, 后文将采用以下简称来表示各个活动: 翻砂浇铸(FCT)、重力铸造(GCT)、切割(Cut)、抛光(Pls)、铸品检查(CI)、回炉熔融(Rlm)、机械加工(Mac)、检验(Insp)、研磨(Gri)、抛光(Pls)、镀前检查(CP)、电镀(Elp)、组装(Ass)、包装(Pac).

4.3.2 提炼简单活动序列

为了正确挖掘日志文件中所包含的业务流程, 使用算法 2 来提取日志中的一般序列, 但是考虑到重复序列存在的普遍性, 算法 2 对日志文件中出现的重复序列进行合并处理, 进而得到水龙头生产流程中所有的简单活动序列. 其中 L 为活动序列总条数, Num[i]存放第 i 条活动序列所含活动的数目, point[i]存放第 i 条活动序列中的活动, C[i]存放 point[i]中的活动名称, Trace[i]存放活动序列 i 本身.

算法 2

输入: 挖掘出的流程日志文件中按顺序出现的活动序列中的每个活动的信息, 参数 i 为实数
输出: 所有简单活动序列
Step1 把图 5 中的记录结果按照实例 ID 分类, ID 相同的属于同一个实例
Step2 把同一个实例中的活动按照时间戳进行排序, 每个实例对应一条基本活动序列
Step3 令 i=1
Step3.1 令 num[i]=第 i 条活动序列中所含的活动数目, 第 i 条活动序列中的活动存放进 point[i]
Step3.2 将 point[i]中的活动有序存放到 Trace[i]中
Step3.3 若 i>L, 继续 step4, 若 i≤L, i=i+1, 返回 step3.1
Step4 令 i=1 到 L, C[i]=Trace[i]中每个活动节点名称
Step5 令 i=1
Step5.1 当 C[i]不为空时, 将 C[i]与 C[i]到 C[L]比较, 若发现重复序列, 则将重复序列设为 null, 保证序列的唯一性
Step5.2 输出简单序列 C[i]
Step5.3 若 i>L, 循环结束, 否则, i=i+1, 返回 step5.1

得到简单活动序列如图 4 所示.

s1:start→FCT→Cut→Pls→CI→Mac→Insp→Gri→Pls→CP→Elp→Ass→Pac→complete
s2:start→FCT→Pls→Cut→CI→Mac→Insp→Gri→Pls→CP→Elp→Ass→Pac→complete
s3: start→FCT→Cut→Pls→CI→Rlm→complete
s4: start→FCT→Cut→Pls→CI→Mac→Insp→Pls→Gri→CP→Elp→Ass→Pac→complete
s5: start→FCT→Cut→Pls→CI→Mac→Insp→Gri→Pls→CP→Insp→Gri→Pls→CP→Elp→
s6: start→GCT→Cut→Pls→CI→Mac→Insp→Gri→Pls→CP→Elp→Ass→Pac→complete

图 4 日志文件中包含的部分简单活动序列

从挖掘结果可知, 日志文件中一共包含 10 条简单活动序列, 接下来要分析这些序列的结构特征, 由于这些序列中不包含隐式库所, 因此可以用结构化挖掘算法挖掘最终模型.

4.3.3 挖掘结构化分支模型

通过对水龙头生产加工流程的日志文件中得到的活动轨迹进行分析, 可以发现序列中存在结构化工作流网的四种基本结构: 顺序、并行、选择、循环, 能否正确地识别并挖掘这些结构, 是建模成功的关键. 根据这四种结构的特点, 考虑到流程中记录的数据不一定是正确完整的, 本文提出一种含噪音处理的结构化挖掘算法. 在算法提出之前, 先做出以下定义:

定义4. 任务集合[10]

a>>>b: 任务a后面直接或间接跟着b, 中间没有a.

a>>>b: 任务 a 后面不会直接/间接地跟着 b 或者 a.

P a>>>b: 任务a和任务b之间的任务集合.

a>>b: 在w的所有任务中, b总是间接或直接的跟在任务a的后面.

ot: 任务t前面的任务集合.

$ot=\{A|\exists a\in T \wedge a>t\rightarrow a\in A)\}$

to: 任务t后面的任务集合

to={B|∃b∈T∧t>b→b∈B}}

算法 3

输入: i 流程执行日志 W={a1,a2,…,ap},其中 ap 为流程案例 ii 活动集合 T={t1,t2,…,tk},其中 tk 为任务 iii ∮={(tm, tn)|tm,tn∈T,m,n=1, …,k,m≠n}为从 T 中任取两个任务 tm,tn 组成的有序任务对集合, 共 k(k-1)种取法 iv 参数初始化 i=1, c=0, ß=0.8, A=0, H=0, N=0.8,

输出: 因果关系任务序列

Step1: 任取(tm,tn)∈ ∮

Step2: 判断(tm,tn)是否为因果关系对, 到 ai 中去检索

Step2.1: 若 |tmo|=|otn|=1 且 tm>tn, 则 Ai=1, C=C+1

若|tmo|=|otn|=1 且|P tm>>>tn|=h, 则 Ai= ßh, C=C+1

若在 ai 中未检索到(tm,tn)任务对, 则 Ai=0

Step2.2: 若 i<p, 则 i=i+1, 返回 step2, 否则转 step2.3

Step2.3: A=∑Ai, 若 A/C≥N, 则(tm,tn)是因果关系对(tm→w tn), 若 A/C<N, 则(tm,tn)不是因果关系对, ∮=∮-{(tm,tn)}

Step2.4: 若 ∮≠Φ, 则返回 step1

顺序结构在流程日志中是最普遍存在的, 有些是显性顺序结构, 有些则是由于受到其他结构干扰难以直接判断出来, 本实例所涉及的水龙头生产流程为显性顺序结构. 算法 3 可以有效挖掘出显性顺序结构, 通过设置阈值 N 和参数 ß, 可以有效摒除噪音的影响, 假设 tm>>>tn, 介于 tm 和 tn 之间的事件数是 h, 则因果值 A 增大 ßh, 计数器 C=C+1. 随着 h 的增加, 因果值增量逐渐变小. 遍历整个日志文件后, A 除以 C 即可用于判断 tm 和 tn 之间的关系. 利用算法 3 可以判断出日志文件中活动间的基本关系, 其中, 因果关系对的任务形成顺序结构, 直接将活动转化成变迁, 活动之间状态的转移关系转化成库所, 便可得到图 5s1 所示的顺序序列, 序列 s2,s3,s4,s6,s7,s8,s9 同理可得.

算法 4

输入: i 流程执行日志 W={a1,a2,…,ap},其中 ap 为流程案例 ii 活动集合 T={t1,t2,…,tk},其中 tk 为任务 iii ∮={(tm, tn)|tm,tn∈T,m,n=1, …,k,m≠n}为从 T 中任取两个任务 tm,tn 组成的有序任务对集合, 共 k(k-1)种取法 iv 参数初始化 i=1, c=0, A=0, N=0.8,P=Φ

输出: 选择关系任务序列

Step1: 任取 tm,tn∈ ∮

Step2: 判断 tm,tn 是否为选择关系, 到 ai 中去检索

Step2.1: 若在 ai 的流程路径 Ơ=s1s2s3…中找到任务 sj=tm, 则 A=A+1, 若 Sj+2=tn, 则 P=P ∪ {Sj+1}, C=C+1, 否则 C=C-1

Step2.2: 若 i<p, i=i+1, 返回 step2, 否则转 step2.3

Step2.3: 若 $\frac{A}{C}$>N 且|P|>>1, 则 tm+P+tn 为选择关系, tm 和 tn 是选择任务的起点和终点任务

每个选择结构都始于一个开始任务, 终于一个结束任务, 并且在开始和结束任务之间只能选择一个任务, 在一般的选择关系中, sj=tm, 则有 Sj+2=tn, 但由于可能存在噪音干扰, 使得 Sj+2≠tn, 因此本文通过设置阈值 N 来过滤噪音, 实现对选择关系的挖掘.

算法 5

输入: i 流程执行日志 W={a1,a2,…,ap},其中 ap 为流程案例 ii 活动集合 T={t1,t2,…,tk},其中 tk 为任务 iii ∮={(tm, tn)|tm,tn∈T,m,n=1, …,k,m≠n}为从 T 中任取两个任务 tm,tn 组成的有序任务对集合, 共 k(k-1)种取法 iv 参数初始化 i=1, flag=false, q=1, P=Φ

输出: 循环关系任务序列

Step1: 任务 tq∈T(q=1, …, k)

Step2: 判断 tq 是否为循环关系, 到 ai 中去检索

Step2.1: 若在 ai 的流程路径 Ơ=s1s2s3…中找到任务 sj+1=tq(循环长度为 1)或 sj+1≠tq, sj+2=tq(循环长度为 2)或 sj+1≠tq, sj+2≠tq, sj+3=tq(循环长度为 3), 则 flag=true, 否则 flag=false

Step2.2: 若循环长度为 1, 则 P=Φ, 若循环长度为 2, 则 P=P∪{ Sj+1}, 若循环长度为 2, 则 P=P∪{ Sj+1, sj+2},

Step2.3: 若 i<p, 则 i=i+1, 返回 step2, 否则转 step2.4

Step2.4: 若 flag=true, 则 tq 是循环任务, tq+P+tq 为循环序列

Step3: 若 q<k, 则 q=q+1, 返回 step1

算法 5 可以挖掘出循环长度为 1 至 3 的循环结构, 长度大于 3 的循环结构可以使用 α 算法进行挖掘[11]. 根据所得结果, 将活动转化成变迁, 活动之间状态的转移关系转化成库所, 便可得到图 5s5、5s10 所示的循环序列. 由于篇幅有限, 并行任务挖掘算法类似上述算法, 不再赘述. 以上算法中, N 的取值是根据经验确定的, 若流程中存在明显噪音, 可以适当降低 N 的取值, 提高结果的准确性.

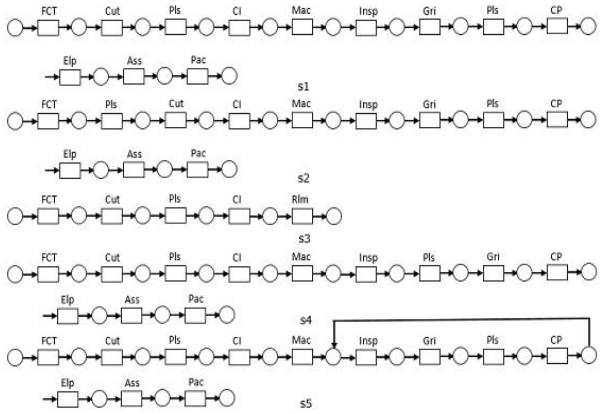

图 5　由简单活动序列得到的结构化分支模型

4.3.4 合并最终模型

在水龙头生产流程的结构化分支模型后, 接下来要根据算法挖掘的结果合并各个分支结构, 构建最终的业务模型. 合并 s1 和 s2(或 s6 和 s7)可以挖掘出活动 Pls 和 Cut 之间的并行关系, 对 s4 和 s5(或 s9 和 s10)进行分析可以判定 Insp、Gri、Pls 和 CP 构成一个循环, 且 Pls 和 Gri 存在并行关系, 合并 s1 和 s3 得出 Rlm 和 Mac 为选择关系, 最后合并 s1 和 s6 知 FCT 和 GCT 也为选择关系, 得出最终的生产流程模型, 如图 6 所示.

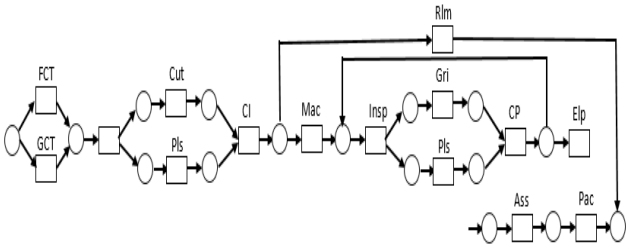

图 6　完整的水龙头生产流程模型

5　资源负载分析

在挖掘出的工作流模型的基础上, 可以进一步对流程中所涉及的资源进行分析. 制造业的设备成本昂贵, 企业当然希望设备的利用率越高越好, 这样才可以提高企业的效益, 增强竞争能力. 一般我们总是假定每种资源的数量是确定的, 但是事实上并非如此. 雇员可能生病、休假甚至辞职, 有时还受到季节性的因素影响, 设备可能故障或者进行采购, 所以能力规划总是要基于特定的能力需求. 本文主要分析设备的占有率和能力之间的关系, 对于那些不需要设备来执行的任务自然不占用设备可用时间(比如铸品检查工序), 通过在泉州智能制造的课题研究中对泉州某卫浴企业的实地调研得知水龙头生产过程中每道工序的平均执行时间等数据, 其中, 铸品检查工序后, 平均有 3%的铸件由于未通过密封性检验会回炉熔融成铜合金水回收, 其余铸件进入机加工序; 经过镀前检查, 12%的不合格工件需重新进行研磨抛光, 88%的合格工件进入下一道工序, 因此研磨、抛光和镀前检查的平均执行次数为 1.14 次, 最后所有的工件都会进入电镀工序. 企业的平均产能为每季度生产 10000 个水龙头, 其中低挡品和高档品的比例为 4:6, 则每个任务的能力需求如表 1.

表 1　每个任务的能力需求

任务名(设备名)	每季度生产个数	单位时间(h)	平均小时数
Foundry casting(Foundry Mold)	4000	3	12000
Gravity casting(Gravity Casting M)	6000	2	12000
Cutting(Shakeout M)	10000	0.7	7000
Polishing(Shot-blasting M)	10000	1.2	12000
Recycled melt(Resistance furnace)	300	2	600
Machining(Metal cutting M)	9700	2	19400
Inspection(Compress-testing)	9700	0.5	4850
Grind(Grinding Belt)	11058	0.8	8846
Polishing(Polishing M)	11058	1	11058
Check before Plating(Ultrasonoscope)	11058	1.5	16587
Electroplate(Salt spray test M)	9700	1	9700
Assemble(Assemble M)	9700	0.2	1940
Packing(Packing M)	9700	0.1	970

[注]表中 M 表示 machine

确定每个任务被执行的平均次数可以通过构造和可达图同构的马尔可夫链, 并添加适当的成本函数[12], 但是构造马尔可夫链需要计算机的支持而且非常耗时, 所以本文采用的方法是基于图 7 描述的设计模式, 其中 N 为发生次数, a 为发生概率, 这些模式不但能够用来构造安全且合理的工作流网, 也能用来确定每个任务被执行的平均次数, 此外, 在更多的过程中, 由于大量循环结构的存在, 计算会更加复杂, 可以自动生成马尔可夫链来计算每个任务的能力需求. 根据每个任务的能力需求, 可以得出设备的利用率如表 2 所示.

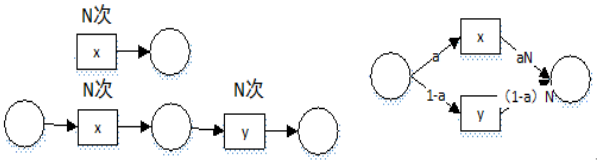

图 7　每个人物相对于原始任务 X 的执行次数

表 2 设备利用率

设备名	设备可用时间(h)	设备数	设备利用率(%)
Foundry Mold	2000	10	60
Gravity Casting machine	2200	8	68
Shakeout M	2000	4	88
Shot-blasting M	1900	10	63
Resistance furnace	1500	2	20
Metal cutting M	2500	9	86
Compression-testing	1500	5	65
Grinding Belt	2000	5	88
Polishing M	2000	5	111
Ultrasonoscope	2100	9	88
Salt spray test M	2300	4	105
Assemble M	1900	3	34
Packing M	400	7	35

一般认为设备利用率在 85%及其以上为设备得到合理利用, , 大于 100 为设备过度运行, 小于 100 表示设备仍有可用时数, 可以进一步规划资源, 以提高产能. 从表 2 可以看出, 设备 Polishing machine 和 Salt spray test machine 的资源负载率过高, 很可能成为生产过程中的瓶颈, 企业可以考虑引进新设备或改善工艺流程来降低每台设备的利用率, 相反地, Assemble machine 和 Packing machine 的利用率相对过低, 为了提高效益, 可考虑将闲置的机器外包或者提高产量, 同时要注意机器设备的保养, 这样可以延长设备的寿命, 有助于提高了设备的利用率. 此外, Foundry Mold 、Gravity Casting machine 、Shot-blasting machine 等也存在设备利用率偏低的问题, Resistance furnace 利用率较低是因为本文不考虑铜合金熔融的准备过程, 该过程大量使用 Resistance furnace.

通过对算法得到的工作流网进行分析, 可以得知水龙头生产过程的资源分配和利用情况, 便于识别瓶颈资源, 了解资源整体利用水平, 也有助于辅助决策者进一步进行资源规划和能力需求决策, 提高资源弹性, 提高资源利用率. 比如可以利用帕累托定律对资源进行进一步分析, 尽可能并行执行任务来缩短完成时间, 确保资源能胜任尽可能多的任务, 或者按照处理时间来处理案例, 赋予处理时间短的案例比处理时间长的案例更高的优先级.

6 结语

本文在考虑日志文件存在噪音的前提下, 对结构化工作流模型的四种基本结构进行分析, 研究从日志文件中挖掘出流程模型的结构化算法, 然后以 Petri 网的形式展现最终模型, 并在此基础上进行资源负载分析, 有助于辅助决策者进行资源规划, 提高设备利用率, 进而提高企业效率和效益. 但是本文对噪音的处理还不够深入, 阈值的设定带有一定的主观性, 如何提高阈值的准确度有待进一步研究, 此外, 本文偏重研究工作流模型的挖掘过程, 对于资源规划仅仅在所述案例范围内做简单分析, 如何进一步进行资源规划将是未来研究的重点.

参考文献

1 潘海兰.一种建模的新技术:流程挖掘.上海第二工业大学学报,2006,23(6):127–132.

2 van der Alast WMP, van Dongen BF, Herbst J, Maruster L. Workflow mining: A survey of issues and approaches. Data & Knowledge Engineering, 2003, 47: 237–267.

3 Agrawal R, Gunopulos D, Leymamn F. Mining process models from workflow logs. Proc.of the 6th Int. Conf. on Extending Database Technology (EDBT), Valencia Spain, Expanded version available as IBM Research Report, 1998.

4 Liu Y, Zhang H, Li C, et al. Workflow simulation for operational decision support using event graph through process mining. Decison Support Systems, 2012, 52(3): 685–697.

5 van der Alast WMP, Weijters T, Maruster L. Workflow mining: discovering process models from event logs. IEEE Trans. on Knowledge and Data Engineering, 2004, 16(9): 1128–1142.

6 Li J, Liu D,Yang B. Process mining: Extending a-algorithm to mine duplicate tasks in process logs. Lecture Notes in Computer Science, 2007, 4537: 396–407.

7 van Dongen BF, van der Alast WMP. Emit: A process mining tool. Applications and Theory of Petri Nets 2nd ed, Berlin, Germany: Springer Verlag, 2004: 454–463.

8 van der Alast WMP, Hee Kees V.工作流管理——模型、方法和系统.北京:清华大学出版社,2004.

9 孔继利,贾国柱.基于流程挖掘的流程建模方法研究.中国管理信息化,2008,21(11):59.

10 张立群.支持业务流程建模的块结构流程挖掘技术的研究[博士学位论文].济南:山东大学,2010.

11 钟越,马光思,柯贤波.基于扩展 Petri 网的工作流建模及应用.现代电子技术,2007,30(9).

12 范玉顺,罗海滨,林慧苹,赵虹.工作流管理技术基础.北京:清华大学出版社,200l.