第2章 相关工作

2. 1. 相关挖掘算法介绍

本节介绍一些主流的流程挖掘算法，其中重点介绍与本文工作相关的系列流程挖掘算法（算法、算法和算法）。其中算法是系列算法的鼻祖，算法和算法分别对算法做出了改进，算法可以挖掘出含有非自由选择结构的流程模型，算法可以挖掘出含有不可见任务的流程模型。

2. 1. 1. 算法

算法[16]关注的重点在于事件日志中不同活动之间的因果关系。算法根据事件日志中出现的先后顺序定义了四种活动间的次序关系。不妨以a、b表示事件日志中的两个不同活动，那么四种关系分别是：

1.

2.

3.

4.。

根据上述定义的四种关系，算法找出所有满足如下关系的集合:

|  |  |  |
| --- | --- | --- |
|  |  | （2-1） |

即集合中每个元素是二元组(A,B)，其中A和B都是日志中活动的集合。A和B中的每个元素都满足“”关系，而A中每个元素和B中每个元素都满足“”关系。在获取集合后，其中中每一个最大的元素都对应挖掘后模型的库所，而事件日志中的每个活动对应挖掘后过程模型的变迁。根据中的关系，可以确定挖掘后模型的每一条边，进而获取挖掘以后的过程模型。

算法具有算法流程简单的特点，同时因为算法基于流程的推理，具有算法正确性可以证明的优点。然而正式因为算法简单，算法无法挖掘出流程模型中的一些复杂结构。如算法无法挖掘含有不可见任务和非自由选择结构的流程模型。在算法的基础上，系列的算法和分别具备了能够挖掘非自由选择结构和不可见任务的能力。

* + 1. 算法

算法中将工作流模型中任务的依赖关系分为了两类：显式依赖关系(Explicit Dependency)和隐式依赖关系(Implicit Dependency)，并且通过找寻事件间的隐式依赖关系来挖掘工作流模型中的非自由选择结构。

显式依赖的两个变迁是指在一个工作流网中，两个变迁之间是满足连接性与连续性。而隐式依赖是指工作流网中两个变迁需要满足连接性、不连续性和可达性。定义2.1和2.2分别给出了显式依赖和隐式依赖的定义。

**定义2.1(显式依赖)**： 是一个工作流网，那么对于任意两个变迁，a，b之间存在一个显式依赖关系当且仅当以下两个条件满足：

连接性：；

连续性：存在一个标识s，,满足且。

**定义2.2(隐式依赖)：** 是一个合理的工作流网，那么对于任意两个变迁，a，b之间存在一个隐式依赖关系当且仅当以下三个条件满足：

连接性：；

不连续性：不存在一个标识s,,满足且；

可达性：存在一个标号s，满足, 并且另外一个标号>,满足。

如图2.1所示，T1与T3，T2与T3均为显式依赖。而T1与T4之间，T2与T5之间则是隐式依赖。如果一个模型中的仅含有显式依赖关系，那么这个模型中就不含有非自有选择结构，大多数过程挖掘算法都可以将模型正确的挖掘出来。而当工作流模型中含有隐式依赖关系时，则这个模型中含有非自有选择结构，绝大部分过程挖掘算法不可以将正确的模型挖掘出来，这正是算法所解决的问题。

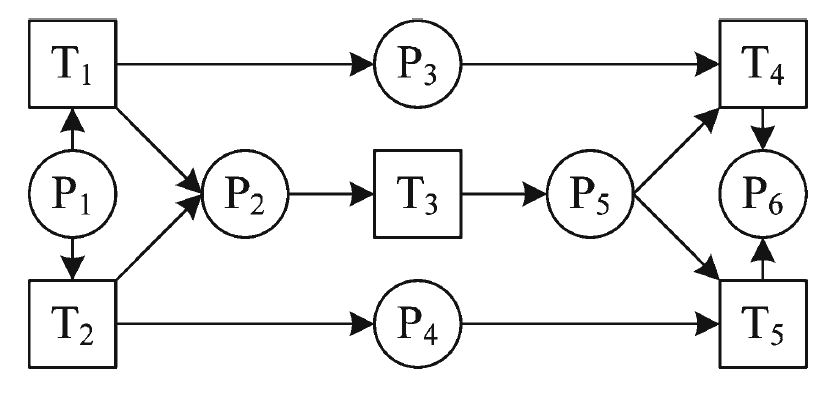


图2.1一个同时具有显式与隐式依赖的模型

算法指出了三种计算隐式依赖的方法。

表2.1 中三种关系

|  |  |
| --- | --- |
| **关系** | **意义及说明** |
|  | 当且仅当,且存在一个任务,和两个不同的库所,其中,且a,a, b,b, 并且不存在任务,满足或者 |
|  | 当且仅当,|a|>1且存在一个任务满足且存在一个库所满足不存在任务,或，存在一个任务满足或 |
|  | 当且仅当,||>1且存在一个任务满足且存在一个库所满足不存在任务,或，存在一个任务满足或 |
|  | 当且仅当或 |
|  | 当且仅当存在两个任务,满足，,,,,且 |

算法在找到三种间接依赖关系后，将间接依赖关系认为是普通的依赖关系“”并如算法中的公式来找寻变迁、库所、边集合，并将找到的集合转换为对应的工作流模型。

* + 1. 算法

不可见任务(invisible task)是工作流模型中一类常见的结构元素，由一种不含有具体任务信息的变迁来表示。不可见任务与其相连的边表达了工作流网中的拓扑结构信息。算法无法处理含有不可见任务的模型，算法中根据不可见任务的功能，将其分为了如下五个种类：

1. 跳过(SKIP):如图2.2所示，跳过类型的不可见任务表达了工作流模型中跳过执行一部分任务的语义。图中的不可见任务使得流程模型可以跳过执行任务B，即流程的事件日志为{ABC,AC}。

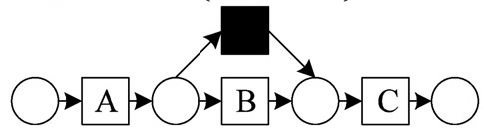


图2.2 的过程模型

1. 重复(REDO): 如图2.3所示，重复类型的不可见任务表达了工作流模型中重复执行一部分任务的语义。图中的不可见任务使得流程模型可以重复执行任务B，即流程的事件日志可以为{ABC, ABBC, ABBBC，…}。

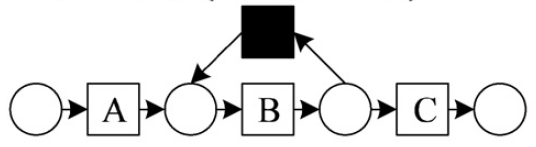


图2.3 的过程模型

1. 跳转(SWITCH): 如图2.4所示，跳转的不可见任务表达了多个互斥执行的分支中可以进行跳转的语义。图中的不可见任务使得流程模型可以从分支AC切换到BD分支上，即流程的事件日志可以为{AC, BD, AD}。

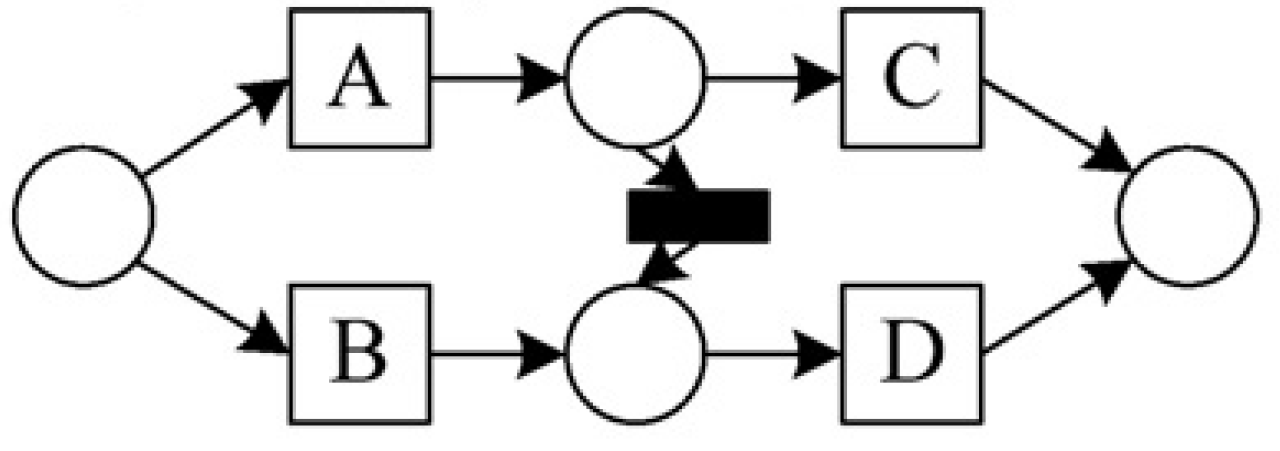


图2.4 的过程模型

1. 初始(INITIALIZE):如图2.5所示，初始的不可见任务表达了在起始库所后有多个并发任务分支或者不存在源变迁的语义。图中的不可见任务使得流程模型中可以含有两个并发任务分支e一起开始,即流程的事件日志可以为{BCE,CBE,BDE,DBE}。

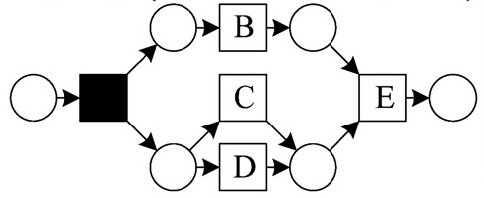


图2.5 的过程模型

1. 终结(FINALIZE)：如图2.6所示，终结的不可见任务表达了在汇结库所前有多个并发任务分支或不存在汇结变迁的语义。图中的不可见任务使得流程模型中可以有两个任务分支B、(CD)一起结束，即流程的事件日志可以为{EBC, ECB, EBD, EDB}。

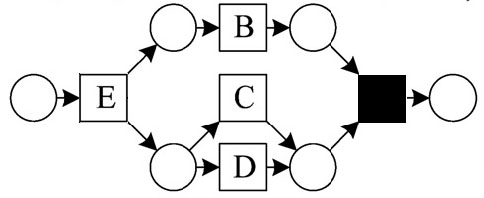


图2.6 的过程模型

针对不可见任务，算法引入了任务间的虚假依赖关系(mendacious dependency)，并将这种依赖关系与不可见任务进行了对应。

在算法中,通过如下的规则来发现关系：令是一个含有不可见任务的工作流模型，是N的事件日志。

**a**当且仅当:

*.*

在图2.7所示的工作流模型片段中，存在一个不可见任务t。如果变迁y与变迁x相同，那么变迁t是一个SHORT-SKIP类型的不可见任务。如果变迁x可达变迁y，那么变迁t是一个LONG-SKIP类型的不可见任务。如果变迁a与b相同，那么变迁t是一个SHORT-REDO类型的不可见任务。如果变迁b可达变迁a，那么变迁t是一个LONG-REDO类型的不可见任务。如果上述的情况都不存在，那么变迁t是一个SWITCH类型的不可见任务。

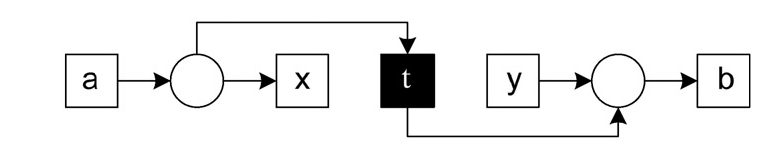


图2.7 **关系**的模型片段

算法在找到虚假依赖关系后，将虚假依赖关系认为是普通的依赖关系“”并如算法中的公式来找寻变迁、库所、边集合，并将找到的集合转换为对应的工作流模型。

* + 1. 其他流程挖掘算法

除去基于推理的、、算法，本节介绍一些其他的主流流程挖掘算法，包括Genetic、Heuristic、Region和ILP算法。

**Genetic 算法**

Genetic[22]算法整合了遗传算法的流程，可以进行全局的搜索，并且可以在一定程度上避免事件日志噪声的影响。而且，通过设置遗传过程中的选择函数，Genetic算法可以对于挖掘以后的结果进行偏好导向。与遗传算法流程类似，Genetic过程挖掘算法主要有如下几步：

首先根据事件日志构造初始种群。初始种群的构造有两种方法：第一种是根据事件日志中的活动，随机的生成若干过程模型作为初始种群。第二种方法是根据事件日志，利用一些诸如Heuristics之类高效率的挖掘算法，产生若干过程模型，作为初始种群。第二种方法相对于第一种方法可以在一定程度上减少后续的迭代次数。

在获取初始种群后，算法的第二步是个体评价，即计算种群中每个过程模型的拟合度。过程模型的拟合度是通过计算事件日志与模型之间的拟合程度，即拟合度计算的是事件日志中有多少事件轨迹可以由模型所表达。

算法的第三步是通过选择运算、交叉运算和变异运算生成下一代种群。选择运算指在个体评价中，选择拟合度较高的部分过程模型直接保留到下一代。交叉运算是指将两个或两个以上的过程模型进行重新组合，组成一个新的模型。变异运算是指在过程模型内部修改，生成新的模型。

在获得了下一代的种群后算法重复第二步和第三步，计算每一代种群的拟合度，直到找到拟合度最大的过程模型后停止算法，并将拟合度最大的模型作为结果输出。

算法通过进行全局搜索可以获得最优的过程模型。但是考虑到效率问题，有时算法并不会以找到拟合度最大的模型的方式结束。一种结束算法的方法是在生成给定个数的过程模型后结束，并在生成的所有模型中挑选最适合的模型。另一种结束算法的方法是在满足连续若干代种群的最大拟合度呈下降趋势后结束算法。

**Heuristic 算法**

Heuristics算法[23]挖掘得到的过程模型以Casual Net[27]的形式保存，Casual Net可以通过一定的规则转换成为Petri网。Heuristics 算法计算过程中考虑了事件日志中每一个事件轨迹的频率，这样在一定程度上可以排除日志中噪声对挖掘结果的影响。Heuristics算法流程主要分为两个步骤。

第一步，Heuristics算法将事件日志转换为中间结构Dependency Graph（即依赖图）。依赖图是一种图的数据结构，直接反映了事件日志中活动间的拓扑结构。即图中的点是事件日志中的活动，图中的边代表日志中活动间的顺联关系。通过如下的两个标准，Heuristics算法在依赖图中剔除部分的边，对于事件日志进行降噪：

|  |  |  |
| --- | --- | --- |
|  |  | 2-2 |
|  | | | 2-3 |

获得事件日志对应的依赖图后，Heuristics算法第二步将依赖图转换为Casual Net. Casual Net 是一个六元组(。其中A代表活动的集合，在事件日志中已经有所定义。代表模型的初始活动和终结活动，可以通过人为进行设置。D指活动间的依赖关系，这一部分可以通过算法上一步生成的依赖图直接获得。I和O分别指每个活动的前驱活动集合和后继活动集合。根据Casual Net的定义，我们发现组成Casual Net的六元组中，只有I和O是需要进一步确定的。在这一步骤中，Heuristics算法再一次考虑事件日志中的每个事件轨迹的出现频率，对于每个活动，算法仅将发生了具有一定频率的事件轨迹中的前后顺序的活动添加到I和O集合。当Casual Net中每个元素都确定了以后，算法结束，生成的Casual Net即为事件日志的挖掘结果。

**Region 算法**

Region[24]算法是一种基于区域(region)的业务流程挖掘算法，通过使用变迁系统[30]作为算法中间结构的挖掘算法。因此算法包含了变迁系统的优点，如可以充分反映事件日志中活动间的关系，又如Region算法可以避免“状态爆炸”，即控制挖掘后的过程模型结构不会很庞大。Region算法主要分为如下两个步骤。

首先，Region算法从事件日志中提取变迁系统。变迁系统是一个三元组(S, A, T),其中S指状态的集合，A指事件中活动的集合。T是集合的子集，即，代表一个状态经过活动A后变成另一个状态。其中事件日志中执行到的每一步对应一个变迁系统的状态，而变迁系统的状态根据可以对应若干事件日志执行的步骤。根据变迁系统的定义，算法这一步骤是将读取事件日志，构造所有的状态，并且根据状态间的变换关系，确定变迁集合T。

在获取变迁系统以后，算法的第2步是从变迁系统中获得过程模型。在本算法中，过程模型是用Petri网存储的。因为在第一步确定变迁系统时，变迁已经与事件日志中的活动相对应了。所以这一步主要确定Petri网中的库所和边。在算法的这一步中，提出了区域的概念。区域是指变迁系统中若干库所构成的集合，其中，变迁系统中所有相同的变迁要么进入该集合，要么从该集合出去。每一个区域对应挖掘以后的过程模型的库所，变迁系统中变迁与区域的连接关系对应过程模型中的每一条边。因此算法得到了挖掘以后的过程模型，该部分可以直接利用Petrify工具直接完成。

**ILP 算法**

ILP算法（插入引用）同Region算法类似，也是一种基于区域的业务流程挖掘算法，然而与Region不同的是，ILP算法是基于语言相关的区域(language-based region)。在ILP算法中，语言是指事件日志中的所有事件轨迹。

显然对于一个在任务集合T上事件日志W（即语言）而言，如果构造一个只包含所有的任务作为变迁，不含有边或者库所的工作流模型N。显然N是一个满足事件日志W的工作流模型，但是N所能支持的语言远大于W。因此，ILP算法中通过构造整数规划的方法，在N中构建库所和边，限制N所能表达的事件轨迹，进而达到流程挖掘的目的。

可以看出，算法分为两步，第一步是根据输入的事件日志构造出一个整数线性规划问题。第二步是求解这个整数线性规划问题，进而构造出流程模型所必须的边和库所 。

* 1. 流程挖掘算法评估框架介绍

本节介绍两种流程挖掘算法评估框架，分别是基于专用模型全集和专用模型子集与推荐技术的。本文的工作重点之一典型模型库的构建就是为流程模型的评估框架而设计的。

* 2. 1. 基于专用模型全集的流程挖掘算法评估框架

Rozinat等人对流程挖掘算法评估框架做出了开创性的工作【谭世杰28】。Rozinat等人指出，虽然现在已经有很多流程挖掘算法被提出，但是尚未有一个可以评估这些算法所挖掘模型质量的普遍方法。同时，Rozinat等人设计了一个流程挖掘算法评估框架，并给出了评估框架所应该具有的基本元素。

图2.8展示了Rozinat等人提出的流程挖掘评估框架，其中包括：

库模块：库模块中包含两类文件，分别是事件日志和过程模型。事件日志由正例和负例组成，而过程模型则包括参考模型和发现模型，发现模型是指利用流程挖掘算法挖掘而得到的流程模型。

日志生成模块：日志生成模块是指根据流程模型生成对应的事件日志。其中输入的流程模型来自于库模块中的参考模型，而产生的事件日志则可以存储在库模块中。

流程发现模块：流程发现模块从事件日志中利用流程挖掘算法来挖掘出流程模型。输入的事件日志来自于库模型，而输出的流程模型则存储于库模块的发现模型中。

修改和验证工具：主要针对库模块中的数据（即流程模型和事件日志）进行验证与修改。如在流程挖掘前，需要验证事件日志是否是完备的。而对于事件日志修改，如在事件日志中添加噪音，可以验证流程挖掘算法是否可以处理含有噪音的事件日志。

评估和比较模块：在框架中分别列举了在过程挖掘中已经存在的两种不同的评估方法，分别是使用统计的方法和机器学习的方法。

王建民等人在Rozinat提出的流程挖掘算法评估框架做出扩展【引用sac2012】,将对一组流程挖掘算法从相似度的维度上进行了经验评估。

评估中选取了一组手工流程模型和一组来自实际工厂中的流程模型来作为参考模型，评估了主流的流程挖掘算法、、、Genetic、 DTGenetic、Heuristic和Region。

评估首先调用日志生成模块，为两组参考模型集合的每个模型生成对应的事件日志。然后调用流程发现模块，分别用每个流程挖掘算法挖掘事件日志。接着调用评估和比较模块，分别利用行为和结构的相似度计算方法计算挖掘获得流程模型和参考模型之间的相似度，通过比较这些相似度之间的差异来对流程挖掘算法做经验的评估。

评估中获得了一些有价值的结论，如在人工流程模型数据集上，对于含有不可见任务的流程模型挖掘效果最好，对含有非自由选择结构的流程模型挖掘效果最好，DTGenetic对于含有重名任务的挖掘效果最好。在真实的数据集中，算法的表现最优。

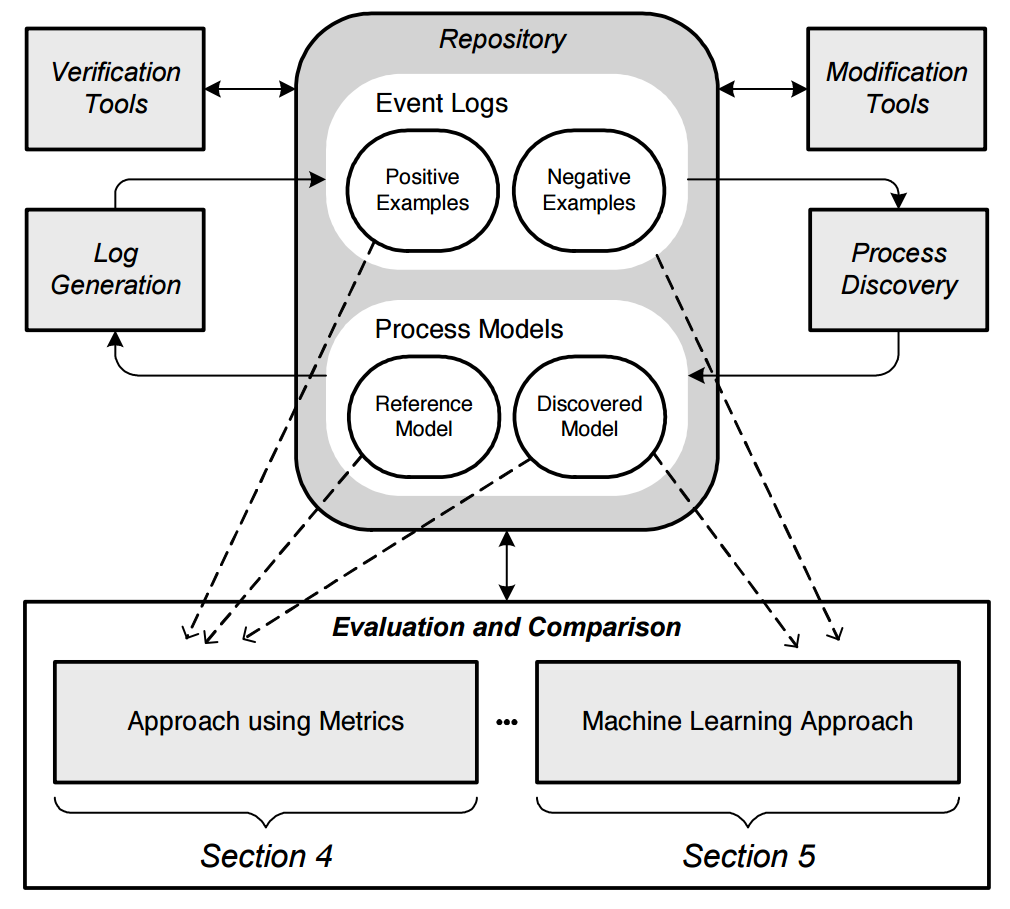


图2.8 Rozinat等提出的流程挖掘评估框架

* + 1. 基于专用模型子集与推荐技术的流程挖掘算法评估框架

虽然上一节基于专用模型全集的流程挖掘算法评估框架可以评估不同的业务流程算法，但是这个评估框架存在一个显著的缺点，即当参考模型集合较大、流程挖掘算法较多时，流程挖掘算法评估则计算量超大且十分耗时。

为此，王建民教授等人提出了基于专用模型子集与推荐技术的流程挖掘算法评估框架。顾名思义，只需要对于参考模型集合中一部分模型分析，而对于其他流程模型只需要利用推荐技术估算即可。图2.9为该评估框架的结构图，其中虚线部分即为Rozinat等人提出的基于专用模型全集的流程挖掘算法评估框架。除此之外，还有两个部分，分别是*学习阶段*和*推荐阶段*。

对于一组流程模型，选择一部分模型作为参考模型(Reference Model)，利用Rozinat提出的流程评估框架，生成参考模型对应的事件日志，然后分别利用流程挖掘算法计算出挖掘模型，最后通过评估挖掘模型与参考模型之间的相似度。

在*学习阶段*,首先在步骤4，提取参考模型集合中模型的结构特征，然后在步骤5中，根据流程模型对提取的特征进行降维操作（如主成分分析），最后在步骤6，依据挖掘算法在流程模型中的表现和降维以后的模型特征，构建回归模型。

在*推荐阶段*，对于模型集合中非参考模型中的部分，在步骤7中首先提取模型的特征，然后在步骤8中，利用学习阶段中获得的回归模型和模型的特征可以估算出流程挖掘算法在这个流程模型上的表现。

对于大部分模型，省去了繁杂的实际计算，基于专用模型子集与推荐技术的流程模型评估框架相比基于模型全集的流程模型评估框架在评估在时间开销上小很多。

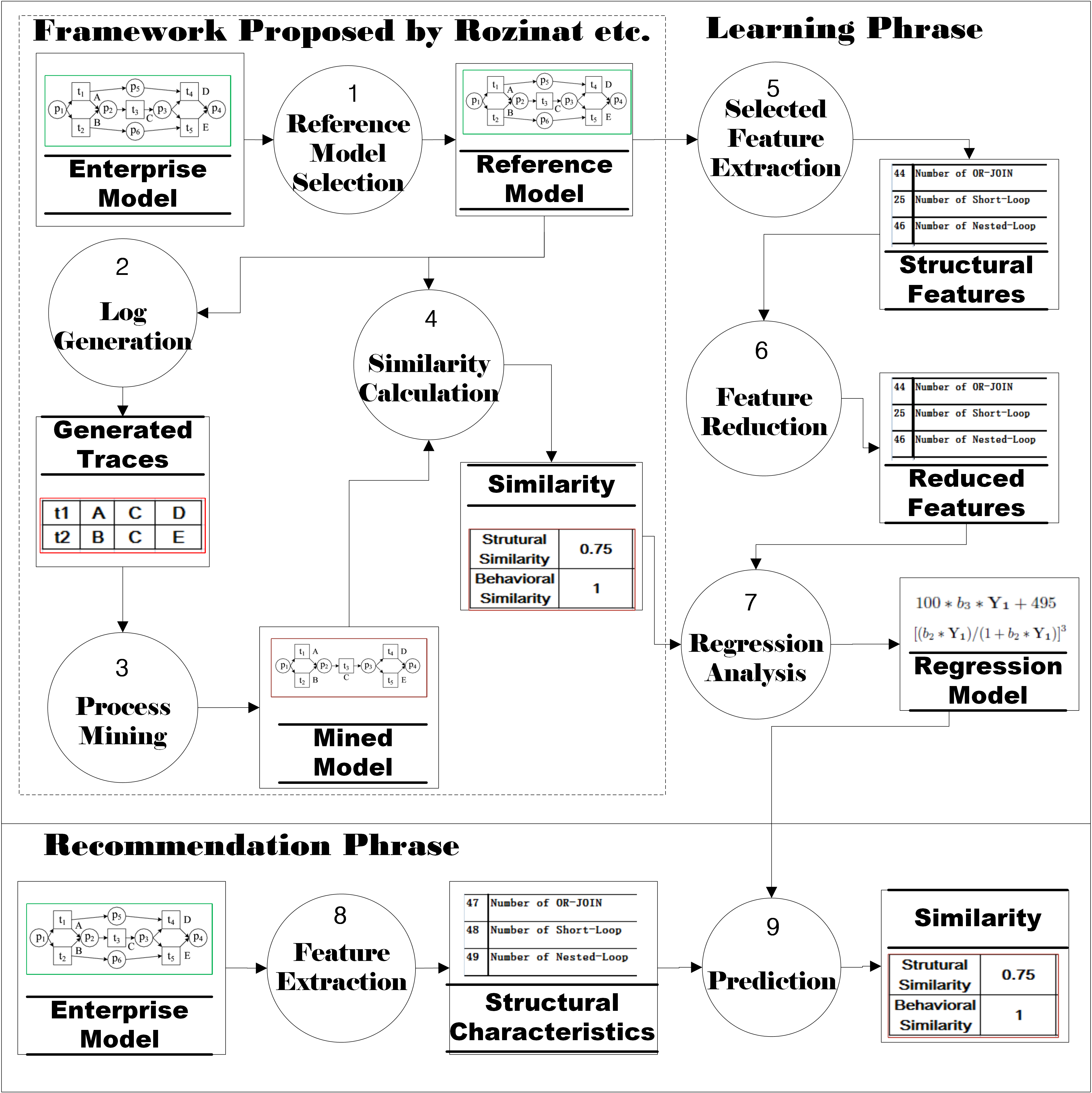
****

图2.9 基于专用参考模型子集和推荐技术的流程挖掘评估框架

* 1. 本章小结

本章从流程挖掘算法和流程挖掘算法评估框架两个角度去总结了研究领域当前的研究成果。首先回顾了几种主流的业务流程挖掘算法，并重点总结了基于推理的系列的流程挖掘算法(算法、算法和算法)，然后分析并总结了基于专用模型全集和基于专用模型子集与推荐技术的流程挖掘算法评估框架。