(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 105659263 A
(43)申请公布日 2016.06.08

(21)申请号 201480056774.4
(22)申请日 2014.09.24
(30)优先权数据
13250102.4 2013.09.26 EP
(85)PCT国际申请进入国家阶段日
2016.04.15
(86)PCT国际申请的申请数据
PCT/GB2014/000378 2014.09.24
(87)PCT国际申请的公布数据
WO2015/044629 EN 2015.04.02
(71)申请人 英国电讯有限公司
地址 英国伦敦
(72)发明人 贝南·阿斯文
特雷弗·菲利浦·马丁
(74)专利代理机构 北京三友知识产权代理有限公司 11127
代理人 吕俊刚 杨薇
(51)Int.Cl.
G06Q 10/04(2006.01)

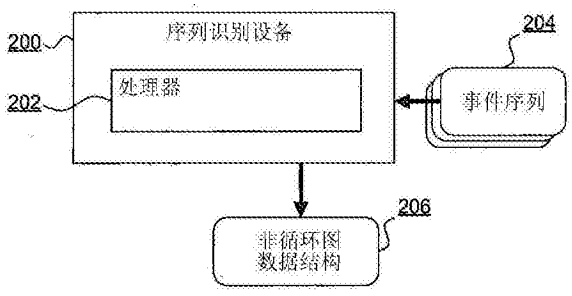
权利要求书2页 说明书18页 附图11页

(54)发明名称

序列识别

(57)摘要

一种包括处理器的序列识别设备，其中，所述设备适于生成在多个按时间排序的事件中识别的事件序列中的事件的等价类的有向非循环图数据结构，其中，所述设备还适于向所述图添加一个或更多个其它事件序列的表示，使得序列的具有公共等价类的最初子序列和最终子序列中的一个或更多个被组合在所述图中。



CN 105659263 A

1.一种包括处理器的序列识别设备，其中，所述设备适于生成在多个按时间排序的事件中识别的事件序列中的事件的等价类的有向非循环图数据结构，

其中，所述设备还适于向所述图添加其它事件序列的表示，使得事件序列的具有公共等价类的最初子序列和最终子序列被组合在所述图中，所述设备还包括：

序列识别器，其适于基于定义事件之间的至少一个关系的至少一个序列扩展关系来识别所述事件序列和所述其它事件序列；

事件归类器，其适于基于至少一个事件归类定义来确定事件的等价类；以及

事件过滤器组件，其适于基于所述图来过滤到来的按时间排序的事件，

其中，所述事件过滤器组件还适于基于所述至少一个序列扩展关系以及每个到来事件到等价类的归类来遍历所述图，以识别用所述图表示的到来事件的序列，以及其中，所述事件过滤器组件还适于识别与用所述图表示的等价类的序列不一致的到来事件。

2.根据权利要求1所述的序列识别设备，所述序列识别设备还包括通知器，所述通知器适于响应于所述事件过滤器组件进行的识别来生成通知。

3.根据权利要求1所述的序列识别设备，所述序列识别设备还包括预测器，所述预测器适于通过所述事件过滤器组件的遍历，将预测的未来到来事件的至少一个预测的等价类识别为在有向非循环图中下一个指示的等价类。

4.根据权利要求1所述的序列识别设备，其中，定义所述至少一个序列扩展关系，使得基于至少一个关系标准的满意度的度量并且响应于所述度量满足预定阈值来确定事件之间的关系。

5.根据权利要求1所述的序列识别设备，其中，各事件包括多个公共属性，各公共属性具有对所有事件公共的域，以及

其中，基于多个公共属性通过至少一个标准来定义各事件归类。

6.根据权利要求5所述的序列识别设备，其中，所述事件归类器用至少一个事件归类的所述至少一个标准基于事件的满意度的度量来确定事件的等价类。

7.根据之前任一权利要求所述的序列识别设备，其中，所述图具有至少两个边，各边对应于至少一个事件的等价类，并且其中，所述设备还适于生成各事件与对应图边之间的关联，使得能基于边识别事件。

8.一种用于识别多个按时间排序的事件中的事件序列的序列识别设备，各事件是计算机系统能访问的数据项，所述设备包括：

存储组件，其用于存储：

i)至少一个序列扩展关系，其定义事件之间的至少一个关系用于识别事件序列；以及

ii)至少一个事件归类定义，其用于将事件序列中的事件归类；

序列识别器，其适于基于所述至少一个序列扩展关系来识别事件的第一序列和第二序列，使得所述多个事件中的各事件属于所述第一序列和所述第二序列中的至多一个；

事件归类器，其适于基于所述至少一个事件归类定义来确定事件的所述第一序列和所述第二序列中的各事件的事件归类；

数据结构处理器，其适于生成有向非循环图数据结构；

其中，所述数据结构处理器还适于针对所述第一序列，生成事件归类的有向非循环图，使得所述图的各边对应于所述第一序列中的事件的事件归类，

其中，所述数据结构处理器还适于利用所述图数据结构处理所述第二序列，以将所述第二序列中的事件的事件归类添加到所述图，使得所述第一序列和所述第二序列的具有公共事件归类的最初子序列和最终子序列被组合在所述图数据结构中。

9.一种序列识别的计算机实现的方法，所述方法包括：

生成在多个按时间排序的事件中识别的事件序列中的事件的等价类的有向非循环图数据结构；

向所述图添加其它事件序列的表示，使得事件序列的具有公共等价类的最初子序列和最终子序列被组合在所述图中；

基于各到来事件到至少一个等价类的归类来遍历所述图，以识别用所述图表示的到来事件的序列；以及

识别与用所述图表示的等价类的序列不一致的到来事件。

10.根据权利要求15所述的计算机实现的方法，所述方法还包括通过事件过滤器组件的遍历，将预测的未来到来事件的至少一个预测的等价类识别为在所述有向非循环图中下一个指示的等价类。

11.一种用于多个按时间排序的事件的序列识别的计算机实现的方法，各事件是计算机系统能访问的数据项，所述方法包括以下步骤：

接收至少一个序列扩展关系，所述至少一个序列扩展关系定义事件之间的至少一个关系用于识别事件序列；

接收事件归类的至少一个定义，所述至少一个定义用于将事件序列中的事件归类；

确定事件的第一序列中的各事件的事件归类，所述第一序列是基于所述序列扩展关系来识别的；

生成针对所述第一序列的事件归类的有向非循环图数据结构，其中，所述图的各边对应于所述第一序列中的事件的事件归类；

确定事件的第二序列中的各事件的事件归类，所述第二序列是基于所述至少一个序列扩展关系来识别的，使得所述多个事件中的各事件属于所述第一序列和所述第二序列中的至多一个；

利用所述图数据结构处理所述第二序列，以将所述第二序列的事件的事件归类添加到所述图，

其中，在处理步骤中，所述第一序列和所述第二序列的具有公共事件归类的最初子序列和最终子序列被组合在所述图数据结构中。

12.一种计算机程序元件，所述计算机程序元件包括计算机程序代码，当所述计算机程序代码被加载在计算机系统中并且在所述计算机系统上执行时，致使计算机执行根据权利要求9至10中的任一项要求保护的计算机实现的方法。

序列识别

技术领域

[0001] 本发明涉及事件的序列识别。特别地，本发明涉及表示事件序列以有效过滤到来事件并且预测未来事件。

背景技术

[0002] 随着信息的生成激增，由系统、软件、装置、传感器和所有方式的其它实体创建巨大量的数据。一些数据的意图是供人查看、问题识别或诊断、扫描、解析或挖掘。当生成数据集并且以更大量、更大速率和有可能更大复杂度和细节进行存储时，引起存储、操纵、处理或使用数据的“大数据”问题。

[0003] 具体地，可能存在问题的是识别数据内的含义，或者识别大或复杂数据集中的数据项之间的关系。另外，可实时生成数据并且由数据存储组件或数据处理组件以规则或可变间隔并且以预定或可变数量进行接收。一些数据项是随时间推移生成的，用于指示、监测、记载或记录实体、发生的事、状态、事件、意外发生的事、改变、问题或其它。这些数据项可被统称为“事件”。事件包括作为属性的事件信息并且关联有诸如时间和/或日期戳的时间标记。因此，事件是以时间序列生成的。事件的数据集示例包括(还有其它)：网络访问日志；软件监测日志；处理单元状态信息事件；诸如构建访问事件的物理安全性信息；数据发送记录；安全资源的访问控制记录；硬件或软件组件、资源或个体的活动性的指示符；用于配置硬件组件或软件组件、资源或个体的配置信息。

[0004] 事件是可或不可与其它事件直接或间接关联的离散数据项。确定事件之间的关系需要详细分析和比较各个事件并且经常涉及导致得到不合适结论的关系的假正确定。诸如用于将事件信息建模的时间序列分析和机器学习方法的统计方法并不是非常适用的，因为在一些情况下它们需要数值特征，并且因为它们通常力求将数据拟合成已知分布。有证据表明，人的行为序列可与这些分布大有不同—例如，按照诸如发送电子邮件、交换消息、由人控制的车辆交通、交易等的异步事件的序列。在论文“The origin of bursts and heavy tails in human dynamic”(A.L.Barabasi,Nature,pp.207-211,2005)中，Barabasi表明许多活动没有遵守泊松统计，而是替代地由可能跟随有没有活动的长时间段的剧烈活动的短时间段组成。

[0005] 与统计方法和机器学习相关的问题是，这些方法通常需要大量用于形成有意义模型的示例。在出现新行为模式的情况下(例如，在网络入侵事件中)，重要的是可快速检测到该模式(即，在已经看到统计学上大量的事变之前)。恶意代理甚至在可检测到该模式之前可改变该模式。

[0006] 事件序列的识别是普遍未解决的问题。例如，互联网日志、物理访问日志、交易记录、电子邮件和电话记录全都包含与不同系统用户相关的多个重叠事件序列。可从这些事件序列中挖掘出的信息是对于理解当前行为、预测未来行为并且识别非标准模式和可能安全漏洞而言重要的资源。

发明内容

[0007] 本发明因此在第一方面提供了一种包括处理器的序列识别设备，其中，所述设备适于生成在多个按时间排序的事件中识别的事件序列中的事件的等价类的有向非循环图数据结构，其中，所述设备还适于向所述图添加其它事件序列的表示，使得事件序列的具有公共等价类的最初子序列和最终子序列被组合在所述图中。

[0008] 优选地，所述设备还包括序列识别器，所述序列识别器适于基于定义事件之间的至少一个关系的至少一个序列扩展关系来识别所述事件序列和所述其它事件序列。

[0009] 优选地，所述设备还包括事件归类器，所述事件归类器适于基于至少一个事件归类定义来确定事件的等价类。

[0010] 优选地，所述设备还包括事件过滤器组件，所述事件过滤器组件适于基于所述图来过滤到来的按时间排序的事件。

[0011] 优选地，所述事件过滤器组件还适于基于所述至少一个序列扩展关系以及到来事件中的每个到等价类的归类来遍历所述图，以便识别用所述图表示的到来事件的序列。

[0012] 优选地，所述事件过滤器组件还适于识别与用所述图表示的等价类的序列不一致的到来事件。

[0013] 优选地，所述设备还包括通知器，所述通知器适于响应于所述事件过滤器组件进行的识别来生成通知。

[0014] 优选地，所述设备还包括预测器，所述预测器适于通过所述事件过滤器组件的遍历，将预测的未来到来事件的至少一个预测的等价类识别为在所述有向非循环图中下一个指示的等价类。

[0015] 优选地，定义所述至少一个序列扩展关系，使得基于至少一个关系标准的满意度的度量并且响应于所述度量满足预定阈值来确定事件之间的关系。

[0016] 优选地，各事件包括多个公共属性，各公共属性具有所有事件公共的域，并且其中，基于多个公共属性通过至少一个标准来定义各事件归类。

[0017] 优选地，所述事件归类器用至少一个事件归类的至少一个标准基于事件的满意度的度量来确定事件的等价类。

[0018] 优选地，所述图具有至少两个边，各边对应于至少一个事件的等价类，并且其中，所述设备还适于生成各事件和对应图边之间的关联，使得可基于边识别事件。

[0019] 按照第二方面，本发明相应提供了一种用于识别多个按时间排序的事件中的事件序列的序列识别设备，各事件是计算机系统能访问的数据项，所述设备包括：存储组件，其用于存储：限定事件之间的至少一个关系的至少一个序列扩展关系用于识别事件序列，以及用于将事件序列中的事件归类的至少一个事件归类定义；序列识别器，其适于基于所述至少一个序列扩展关系来识别事件的第一序列和第二序列，使得多个事件中的各事件属于所述第一序列和所述第二序列中的至多一个；事件归类器，其适于基于所述至少一个事件归类定义来确定事件的所述第一序列和所述第二序列中的各事件的事件归类；数据结构处理器，其适于生成有向非循环图数据结构；其中，所述数据结构处理器还适于针对所述第一序列，生成事件归类的有向非循环图，使得所述图的各边对应于所述第一序列中的事件的事件归类，其中，所述数据结构处理器还适于利用所述图数据结构处理所述第二序列，以将

所述第二序列中的事件的事件归类添加到所述图中，使得所述第一序列和所述第二序列中的具有公共事件归类的最初子序列和最终子序列被组合在所述图数据结构中。

[0020] 按照第三方面，本发明相应提供了一种计算机实现的序列识别方法，所述序列识别方法包括：生成在多个按时间排序的事件中识别的事件序列中的事件的等价类的有向非循环图数据结构；向所述图添加其它事件序列的表示，使得事件序列的具有公共等价类的最初子序列和最终子序列被组合在所述图中。

[0021] 优选地，所述方法还包括基于各到来事件到至少一个等价类的归类来遍历所述图，以便识别用所述图表示的到来事件的序列。

[0022] 优选地，所述方法还包括识别与用所述图表示的等价类的序列不一致的到来事件。

[0023] 优选地，所述方法还包括通过所述事件过滤器组件的遍历，将预测的未来到来事件的至少一个预测的等价类识别为在所述有向非循环图中下一个指示的等价类。

[0024] 按照第四方面，本发明相应提供了一种用于多个按时间排序的事件的序列识别的计算机实现的方法，各事件是计算机系统能访问的数据项，所述方法包括以下步骤：接收至少一个序列扩展关系，所述至少一个序列扩展关系定义事件之间的至少一个关系用于识别事件序列；接收事件归类的至少一个定义，所述至少一个定义用于将事件序列中的事件归类；确定事件的第一序列中的各事件的事件归类，所述第一序列是基于所述序列扩展关系来识别的；生成针对所述第一序列的事件归类的有向非循环图数据结构，其中，所述图的各边对应于所述第一序列中的事件的事件归类；确定事件的第二序列中的各事件的事件归类，所述第二序列是基于所述至少一个序列扩展关系来识别的，使得多个事件中的各事件属于所述第一序列和所述第二序列中的至多一个；利用所述图数据结构处理所述第二序列，以将所述第二序列中的事件的事件归类添加到所述图，其中，在处理步骤中，所述第一序列和所述第二序列中的具有公共事件归类的最初子序列和最终子序列被组合在所述有向非循环图中。

[0025] 按照第五方面，本发明相应提供了一种计算机程序元件，所述计算机程序元件包括计算机程序代码，当所述计算机程序代码被加载在计算机系统中并且在所述计算机系统上执行时，致使所述计算机执行如上所述的计算机实现的方法。

附图说明

[0026] 现在，将参照附图仅仅以举例方式描述本发明的优选实施方式，其中：

[0027] 图1是适于本发明的实施方式的操作的计算机系统的框图；

[0028] 图2是按照本发明的优选实施方式的用于识别多个事件中的序列的序列识别设备的组件图；

[0029] 图3是按照本发明的一个实施方式的图2的序列识别设备的方法的流程图；

[0030] 图4是按照本发明的一个实施方式的使用中的序列识别设备的组件图；

[0031] 图5是按照本发明的一个实施方式的图4的序列识别设备的方法的流程图；

[0032] 图6a至图6e是示出图2至图5的实施方式采用和生成的示例性数据结构的组件图；

[0033] 图7是按照本发明的替代实施方式的使用中的序列识别设备的组件图；

[0034] 图8是按照本发明的替代实施方式的图7的过滤器的方法的流程图；

[0035] 图9是按照本发明的示例性实施方式的AllowedActions表；

[0036] 图10是按照本发明的示例性实施方式的第一序列的有向非循环图表示；

[0037] 图11是按照本发明的示例性实施方式的第一序列、第二序列和第三序列的有向非循环图表示；

[0038] 图12是按照本发明的实施方式中的示例性算法生成的第一序列和第二序列的有向非循环图表示；

[0039] 图13是按照本发明的实施方式中的示例性算法生成的第一序列、第二序列和第三序列的有向非循环图表示；以及

[0040] 图14是按照本发明的实施方式中的示例性算法生成的第一序列、第二序列、第三序列和第四序列的有向非循环图表示。

具体实施方式

[0041] 图1是适于本发明的实施方式的操作的计算机系统的框图。中央处理器单元(CPU)102借助数据总线108与存储器104和输入/输出(I/O)接口106通信连接。存储器104可以是诸如随机存取存储器(RAM)或非易失性存储装置的任何读/写存储装置。非易失性存储装置的示例包括盘或带型存储装置。I/O接口106是用于输入或输出数据或既输入数据又输出数据的接口。能与I/O接口106连接的I/O装置的示例包括键盘、鼠标、显示器(诸如，监视器)和网络连接。

[0042] 图2是按照本发明的优选实施方式的用于识别多个事件中的序列的序列识别设备的组件图。序列识别设备200包括用于承担设备功能的全部或部分的处理器202。以下，将相对于本发明的多个实施方式描述序列识别设备200的各种功能和组件，本领域的技术人员应该理解，处理器202可适于以各种构造执行、履行、构成或封装一个或更多个这些功能和组件。例如，处理器202可以是诸如通用计算装置(诸如，图1中描绘的计算装置)的CPU 102的一个或更多个CPU。因此，本文中描绘的特定实施方式单纯是示例性的，可替代地采用任何合适构造的组件。

[0043] 序列识别设备200适于接收事件序列204作为来自多个按时间排序的事件中的事件的序列。这多个按时间排序的事件可被存储在数据结构、表、数据库或类似物中，或者另选地，这些事件可被作为事件流接收。使用这多个按时间排序的事件，基于如下所述定义的序列扩展关系来识别事件序列204。可通过序列识别设备200外部的组件(诸如，事件序列识别器)确定事件序列204，或者另选地，可通过序列识别设备200本身确定事件序列204。

[0044] 序列识别设备200还适于确定各事件序列204中的各事件的等价类。等价类是通过一个或更多个事件归类定义来定义的事件的类或类型并且用于将事件分类或归类。在一个实施方式中，序列识别设备200适于基于如下所述的一个或更多个事件归类定义来确定各事件的等价类本身。在替代实施方式中，序列识别设备200通过从序列识别设备200外部的组件接收事件的等价类来确定事件的等价类。

[0045] 序列识别设备200还适于生成有向非循环图(DAG)数据结构206作为事件序列204中的第一个事件序列的等价类的数据结构表示。例如，DAG数据结构206可以是存储在计算机系统的存储器104(诸如，与序列识别设备200关联或者序列识别设备200包括的存储器)中的数据结构。在一个实施方式中，使用数据结构元素作为具有存储器指针的节点来存储

DAG数据结构206，存储器指针用于提供作为DAG边的节点之间的链路。以下，描述DAG数据结构206的示例性实施方式。

[0046] 序列识别设备200还适于将一个或更多个其它事件序列204的表示添加到DAG数据结构中。因此，序列识别设备200接收一个或更多个其它事件序列204并且修改DAG数据结构206，以将这些其它事件序列的表示包括在DAG内。这些其它事件序列中的事件的等价类可以是公共的。例如，第一事件序列开始处的事件的等价类可与第二事件序列开始处的事件的等价类公共。序列识别设备200组合DAG数据结构206中表示的这些公共的子序列，使得在DAG数据结构206中表示基于事件的子序列的具有公共等价类的第一事件序列和第二事件序列之间的关系。序列识别设备200适于组合事件序列的具有公共等价类的最初子序列和最终子序列的DAG数据结构206中的等价类表示(“最初”是在事件序列开始处，“最终”是在事件序列结尾处)。

[0047] 图3是按照本发明的优选实施方式的图2的序列识别设备200的方法的流程图。初始地，在步骤302中，序列识别设备200生成事件序列204中的事件的等价类的DAG数据结构206。随后，在步骤304中，序列识别设备200将其它事件序列204的表示添加到DAG数据结构206中。步骤304中进行的添加包括如上所述组合DAG数据结构206中的等价类表示。

[0048] 序列识别设备200生成的DAG数据结构206包括各事件序列204的等价类的有向表示。对于处理后续接收的按时间排序的事件的流而言，这种表示是特别有利的。通过使用这种DAG数据结构206，可以有效过滤到来的按时间排序的事件的流，以通过对于新事件遍历DAG，识别已知事件序列。DAG数据结构206是特别有益的，因为它表示事件的等价类，所以在用于生成DAG的多个事件中或者在到来事件的流中，基于DAG的过滤过程不会因个体事件的特定特征的解释而被妨碍。另外，可使用对于到来事件遍历DAG的这种方法，以有效地识别与DAG表示的事件序列不相关的新事件序列。在需要识别新序列的情况下，这些识别是有用的。另外，DAG数据结构206允许有效识别与现有序列具有公共子序列的新序列(诸如，包括具有公共等价类的事件的最初或最终子序列的新事件序列)。

[0049] DAG数据结构206还适于预测事件的未来类或类型，并且通过外插法，可使用DAG基于用于生成DAG的事件序列来预测一个或更多个未来事件。当响应于到来的按时间排序的事件的序列部分地遍历通过DAG数据结构206的路径时，可基于DAG中的下一个元素来预测一个或更多个潜在后续事件分类。另外，可使用导致通过DAG的路径的这种部分遍历的序列中的现有事件的属性来生成一个或更多个预测事件。另外，这些预测可以附加地基于序列扩展关系，以通知关于一个或更多个预测未来事件的属性值的确定。例如，在DAG数据结构206代表计算机网络入侵检测系统中的已知攻击的事件序列的情况下，若各事件对应于诸如网络请求、响应、发送的分组或其它网络发生的网络动作，可使用DAG用到来的事件流预测一个或更多个未来事件，以在发生潜在新攻击之前识别它。即使使用到来事件序列来仅仅部分遍历通过DAG的路径，这种提早识别也可是有效的。可确定到来事件序列的等价类与DAG中的等价类的路径的近似程度，并且反应于阈值程度，可识别预测攻击。

[0050] DAG数据结构206还适于识别与可基于通过DAG数据结构206的路径的相似性而相关的事件关联的实体。例如，与完全不同的实体相关但使用事件分类的公共图(诸如，组合图或子图)在DAG中表示的事件可识别实体之间的关系。因此，在实体构成物理对象、装置或人并且事件指示与实体相关的行为、动作、改变或其它发生的事的情况下，由于事件分类公

共性，导致可使用DAG将实体分组。例如，带时间戳的事件可涉及雇员使用安全设施访问资源，诸如，经由带徽章锁(badge-locked)的门进入安全建筑物，或者借助认证系统访问安全网络。这些事件可包括发生的事(诸如，“发生进入的事”和“发生离开的事”)的类型的指示，该指示表明访问资源的开始和停止。另外，事件可包括正被访问资源的识别(诸如，建筑物或网络标识符)。可使用事件之间的序列扩展关系(诸如，雇员标识符的身份和时间限制)来识别这些事件的序列。序列识别设备200生成的DAG数据结构206将这些序列中的事件的等价类建模。这些类可包括例如通过发生的事(“进入”或“离开”)的类型、一天内的时间(例如，“早上”或“下午”)和资源的标识符(建筑物或网络标识符)表征的类。作为在DAG数据结构206中表示的事件序列，可发现与不同雇员相关的事件序列在DAG中有重叠并且因此被组合。基于这种组合，可将这些雇员识别为是相似的。例如，可将早上进入特定建筑物并且下午离开同一建筑物的雇员识别为只在单个地点工作的雇员群体。还可基于DAG辨认其它不同这些群体。在按已知威胁分组的实体可经受严格审查的安全应用中，识别实体群组可以是有价值的。

[0051] 图4是按照本发明的一个实施方式的使用中的序列识别设备200的组件图。图4的某些元件与如之前描述的图2一样，这里将不再对这些元件进行重复。图4的实施方式示出图3中用于生成DAG数据结构206的布置的一个示例性实现方式。图4的序列识别设备200适于接收多个按时间排序的事件422。这多个事件422中的各事件是用于记录之前所描述类型的发生的事(还有其它的)的数据项、数据结构、消息、记录或其它合适手段。事件422构成到序列识别设备200的数据输入并且可被存储在与设备200关联的或者能与设备200通信的数据存储体中。例如，事件422可被存储成表数据结构、数据库、文件、消息列表或其它合适格式。另选地，事件422可通过通信机构(诸如，软件或硬件接口或网络)由序列识别设备200单独地或者成批地接收。各事件422包括用于指示多个按时间排序的事件中的事件位置的时间信息(诸如，时间和/或日期戳)。此时间信息可以是绝对的或相对的。各事件422具有应当被统称为属性的多个字段、列、要素、值、参数或数据项。最优选地，属性是用属性名称来识别的，但用于一致引用事件特定属性的偏移、地址、指示符、标识符、查找或其它合适手段也是可能的。在优选实施方式中，所有事件422的属性是公共的，使得各事件具有所有属性，并且对于所有事件，各属性的域是相同的。在替代实施方式中，一些事件还具有除了公共属性之外的属性，并且对于所有事件，用于生成序列和分类事件的属性子集是公共的。

[0052] 序列识别设备200还包括存储组件410，存储组件410存储一个或更多个序列扩展关系412和一个或更多个事件归类定义414。序列扩展关系412是基于公共事件属性的事件422之间的关系。在事件序列204中，通过一个或更多个序列扩展关系412将各事件与时间在先事件相关。事件序列中的第一事件与在先事件不相关。因此，序列扩展关系412用于定义事件和时间在后事件之间的关系，以便构成事件序列的全部或部分。序列扩展关系412中的一个或更多个可被实现为标准，一对事件满足标准确定事件之间的关系。在一个实施方式中，标准可以对关系起决定性作用。在替代实施方式中，序列扩展关系412中的一个或更多个可被实现为用于确定一对事件之间关系的事件的特征度量。以此方式，可定义模糊关系，使得事件之间的关系基于以事件属性值为基础的特征的一个或更多个度量和与这些度量相关的一个或更多个条件或标准。因此，在一些实施方式中，定义一个或更多个序列扩展关系412，使得基于关系标准的满意度的度量并且响应于度量满足预定阈值来确定事件之间

的关系。

[0053] 事件归类定义414定义被称为等价类或事件类别的事件的类或类型。等价类提供了根据事件归类定义414将多个事件归类为“等价”事件的机制。事件归类定义414基于所有事件公共的事件属性。优选地，事件归类定义414中的每个是基于多个公共属性用至少一个标准来定义的。事件归类定义414中的一个或更多个可被实现为一个或更多个标准，事件满足标准可用于确定事件属于等价类。在一个实施方式中，标准可对事件归类起决定性作用。在替代实施方式中，事件归类定义414中的一个或更多个可被实现为用于确定事件的一个或更多个等价类的基于事件属性的事件的特征度量。以此方式，可定义与等价类的模糊关联，使得事件和等价类之间的关联基于以事件属性值为基础的特征的一个或更多个度量和与这些度量相关的一个或更多个条件或标准。因此，在一些实施方式中，定义一个或更多个事件归类定义414，使得基于事件相对于一个或更多个标准的满意度的度量来确定事件的等价类。

[0054] 在使用中，通过序列识别器416接收序列扩展关系412。序列识别器是硬件、软件或固件组件，适于基于序列扩展关系412来识别多个按时间排序的事件422中的事件序列204。在一个实施方式中，序列识别器416处理多个事件422中的各事件并且应用与序列扩展关系412中的每个关联的标准，以便确定事件是否与之前事件相关。将相关事件存储为事件序列204，事件序列204可随着多个事件422中有更多事件被处理而增长。能料想到，一些事件与之前事件不相关，这些事件可构成新序列的开始。另外，有一些事件不会出现在序列204中的任一个中。可识别这些事件或者将其带上标志，以作进一步考虑。本领域的技术人员应该理解，序列识别器416能进行操作，以识别、监测和跟踪同时发生的多个潜在或实际的序列，以基于序列扩展关系412来识别这多个事件422中存在的所有事件序列204。

[0055] 另外，在使用中，通过事件归类器418接收事件归类定义414。事件归类器是硬件、软件或固件组件，适于确定各事件序列204中的各事件的等价类。在一个实施方式中，事件归类器418接收处理各事件序列204中的各事件并且应用与事件归类定义414中的每个关联的标准，以便确定合适的等价类。

[0056] 序列识别设备200还包括数据结构处理器410作为适于生成各事件序列204中的各事件的DAG数据结构206的硬件、软件或固件组件。在优选实施方式中，DAG数据结构206包括节点和边，使得各边对应于序列中的事件的等价类。因此，在使用中，数据结构处理器420生成第一事件序列204′的最初DAG数据结构206，该DAG数据结构206包括均与序列中的事件的等价类对应的多个图边。这些边连接表示事件序列204′的序列扩展关系410(但不具体与之关联)的节点。因此，在处理第一事件序列204′之后，将DAG数据结构206生成为图，该图具有从起始节点到结束节点的单条直线路径，边对应于沿着该路径链接节点的序列中的各事件的等价类。随后，数据结构处理器420处理其它事件序列204″、204″′，从而将各其它事件序列204″、204″′的表示添加到DAG数据结构206中。特别地，在数据结构处理器420确定第一序列204′和其它事件序列204″、204″′中的一个或更多个最初和最终子序列具有公共的事件归类的情况下，子序列组合在DAG数据结构206中。因此，DAG是事件序列204的等价类的最小表示，其中，具有包括一系列公共等价类的事件子序列的事件序列在DAG数据结构206中合并并且仅被表示一次。因此，DAG数据结构206可分支并且链接于起始节点和结束节点之间的点，以限定起始节点和结束节点之间的路径。

[0057] 本领域的技术人员应该理解，虽然处理器202、序列标识符416、事件归类器418和数据结构处理器420在图4中被图示为单独组件，但至少这些组件中的任一个或全部可在本发明的实施方式中被组合、合并或进一步细分。例如，序列标识符416和事件归类器420可以是单个组件。另外，数据结构处理器420可被省略，由处理器202或序列识别设备200的任何其它合适组件执行其功能。还应该理解，虽然存储组件410被图示为与设备200形成一体，但存储器可另选地设置在设备200外部或者设置为设备200的子组件的一体部分。例如，存储组件410可诸如通过软件和/或硬件接口或网络设置在外部装置或与序列识别设备200通信连接的设备并且被保持在此。

[0058] 图5是按照本发明的一个实施方式的图4的序列识别设备200的方法的流程图。初始地，在步骤500中，序列识别器416通过访问包含事件记录的数据存储体、数据库或表来访问按时间排序的多个事件422。在步骤502中，序列识别器416从存储组件410接收序列扩展关系412。在步骤504中，事件归类器418从存储组件410接收事件归类定义414。在步骤506中，序列识别器416基于序列扩展关系412识别第一事件序列204′。在步骤508中，事件归类器418确定第一事件序列204′中的各事件的等价类。在步骤510中，数据结构处理器420生成等价类的DAG数据结构206，以表示第一事件序列204′。随后，在步骤512中，序列识别器416将至少一个其它事件序列204″识别为第二事件序列204″。在步骤514中，事件归类器418确定第二事件序列204″中的各事件的等价类。在步骤516中，数据结构处理器420利用DAG数据结构206处理第二事件序列204″，以将第二事件序列204″中的事件的等价类添加到DAG数据结构206中。

[0059] 应该理解，图5中图示的并且以上描述的流程图步骤的特定排序不受限制，可另选地采用任何其它合适的步骤和/或步骤的次序。

[0060] 图6a至图6e是示出图2至图5的实施方式采用和生成的示例性数据结构的组件图。图6a示出示例性的事件数据结构740。事件740包括时间戳742作为时间指示符的示例。时间戳742可指示多个事件422中的所有事件一致应用的生成时间、派送时间、接收时间或其它时间点。时间戳742提供了可用于确定和确认多个事件422的按时间排序性质的手段。例如，如果多个事件422不是按时间排序的，则可使用时间戳742来分选事件，以得到按时间排序的多个事件422。事件740还包括多个公共属性744。属性744对于多个事件422中的所有事件而言是公共的。使用属性744中的全部或子集来定义序列扩展关系412。另外，使用属性744中的全部或子集来定义事件归类定义414。属性744中的每个具有对于所有事件而言公共的域。

[0061] 图6a还示出示例性的序列扩展关系数据结构412′。序列扩展关系数据结构412′包括基于事件属性744通过一个或更多个标准750定义的关系748。图6a还示出示例性的事件归类定义数据结构414′。事件归类定义数据结构414′包括多个等价类定义754a、754b，等价类定义754a、754b均是基于事件属性744通过一个或更多个标准756a、756b定义的。

[0062] 图6b示出均包括时间戳742和属性744的多个按时间排序的事件422。这多个事件422被图示为事件流，这是可通过序列识别设备200接收事件的一种方式。这多个事件422可同等地存储在如上所述的表或其它合适数据结构中。

[0063] 图6c示出第一示例性DAG数据结构。图6c的DAG表示两个事件的至少一个事件序列的等价类，第二事件通过序列扩展关系与第一事件相关。事件序列中的第一事件表示为具

有等价类“类1”。事件序列中的第二事件表示为具有等价类“类2”。该图由分别用预定的标记为“S”和“F”的起始节点和结束节点限定界限。用节点“1”指示事件之间的关系，事件序列中的事件之间的时间关系提供了该图的边(等价类)的方向。因此，图6c提供了事件序列的DAG表示。根据图6c的DAG具有不同事件但具有包括等价类的事件的其它事件序列可被称为类似于用于生成图6c的事件序列。

[0064] 图6d示出第二示例性DAG数据结构。图6d的DAG与图6a共享一些特征(诸如，起始节点和结束节点)。图6d的DAG表示至少两个事件序列的等价类，各事件序列具有三个事件的长度。第一事件序列包括按时间次序分别具有等价类“类1”、“类4”和“类1”的事件。第二事件序列包括按时间次序分别具有等价类“类2”、“类3”和“类1”的事件。这两个事件序列在各序列末尾的子序列有重叠，因为这两个事件序列中的最后事件具有等价类“类1”。因此，图6d的DAG组合标记为“3”的节点和结束节点“F”之间的各序列中的最后事件的边。

[0065] 图6e示出第三示例性DAG数据结构。图6e的DAG表示至少两个事件序列的等价类，其中，各事件序列在各序列开始的子序列有重叠。这两个事件序列开始处的事件属于等价类“类1”。因此，图6e的DAG组合起始节点“S”和标记为“1”的节点之间的各序列中的第一事件的边。

[0066] 优选地，DAG数据结构206的边与用于生成DAG数据结构206的事件关联，使得可以将DAG中的等价类表示与对应事件序列中被归类为等价类的事件相关。例如，DAG数据结构206可向用户呈现可视化形式，供其分析、查看或其它原因。用户可使用此关联，基于DAG中的边来导航至事件序列中的特定事件。对本领域的技术人员将明显的是，关联可以是单向(例如，DAG边参考事件或事件参考DAG边)或双向的。

[0067] 图7按照本发明的替代实施方式的使用中的序列识别设备200的组件图。图7的特征中的一些与以上相对于图2和图4描述的特征相同，这里将不再重复这些特征。图7的序列识别设备200还包括过滤器732作为适于基于DAG数据结构206接收和过滤到来的按时间排序的事件730的硬件、软件或固件组件。DAG数据结构206是根据以上相对于图2至图6描述的组件、方法和技术预定的。到来的事件730是被过滤器732过滤的新事件。过滤器732构成采用定义的DAG数据结构206过滤到来的新事件730的组件。例如，过滤器732适于有效过滤按时间排序的事件730的到来流，以识别与从DAG数据结构206中得知的序列对应的事件730的到来流中的事件序列。这是通过以下来实现的：过滤器732对于到来流730中的事件遍历DAG数据结构732，其中，到来的事件730满足序列扩展关系412。

[0068] 因此，在从到来事件730的流接收到新事件时，过滤器732分两个方面进行操作：第一，过滤器732确定新事件是否按照序列扩展关系412与之前接收的事件相关；第二，过滤器732确定新事件是否对应于DAG数据结构206中表示的等价类，该等价类是通过DAG遍历的路径的部分。在第一方面，过滤器732可适于存储所有事件的记录，因为它们被依次接收，以寻找并且识别可与新事件相关的之前接收的事件。在第二方面，过滤器732可适于同时承担并且记录对DAG数据结构206进行的潜在众多遍历，每次遍历对应于源自到来事件730的流的所有部分接收的事件序列。因此，过滤器732优选地设置有用于存储有关接收到事件的信息并且用于存储所有部分接收到的事件序列的DAG遍历信息的存储器、存储体、数据区或类似物。

[0069] 以此方式，过滤器732提供识别事件730的到来流中的已知事件序列的有效方式，

即使是在事件序列到达时散布了其它事件或事件序列的情况下。另外，过滤器732可用于有效识别与DAG表示的事件序列不相关的新事件序列。在需要识别新序列的情况下(诸如，为了添加到DAG数据结构206中)，这些识别可以是有用的。另选地，可使用这些新序列的识别来识别非典型的、可疑的、有疑问的或者说感兴趣的事件序列。例如，在定义用于表示可接受事件序列的DAG数据结构206的情况下，过滤器732可识别不符合DAG表示的任何序列的新序列。本领域的技术人员应该理解，过滤器732可适于在并非在DAG开始(或起始)的节点或边处开始遍历DAG数据结构206，使得可识别与DAG数据结构206中表示的子序列部分地对应的新事件序列。

[0070] 在优选实施方式中，过滤器732设置有通知器736a，通知器736a是响应于处理事件730的到来流而生成通知的硬件、软件或固件组件。例如，在过滤器732识别与DAG数据结构206表示的任何序列不对应的新事件序列的情况下，通知器736a可生成合适的通知。另外地或另选地，在过滤器732识别与DAG数据结构206表示的序列对应或部分对应的事件序列的情况下，通知器736a可生成合适的通知。

[0071] 图7的序列识别设备200还包括预测器734，预测器734是适于接收到来的按时间排序的事件730并且基于预定的DAG数据结构206预测未来事件的一个或更多个等价类或未来事件本身的硬件、软件或固件组件。

[0072] 在从事件730的到来流中接收到新事件时，预测器734分三个方面进行操作：第一，预测器734确定新事件是否按照序列扩展关系412与之前接收的事件相关；第二，预测器734确定新事件是否对应于DAG数据结构206中表示的等价类，该等价类是通过DAG遍历的路径的部分；第三，预测器734基于通过DAG遍历的路径，识别DAG中的一个或更多个潜在的下一个等价类。在第一方面和第二方面，预测器734可适于存储所有事件的记录，因为它们被接收并且同时承担和记录对DAG数据结构206进行的潜在众多遍历，如同过滤器732的情况一样。因此，预测器732优选地设置有用于存储有关接收到事件的信息并且用于存储所有部分地接收到的事件序列的DAG遍历信息的存储器、存储体、数据区或类似物。在第三方面，预测器734适于确定来自DAG的一个或更多个预测等价类，作为事件730的到来流中接收的事件序列的DAG数据结构206的遍历中从当前节点的出边。在最简单的情况下，针对预测的未来事件来识别出边表示的等价类。在一些实施方式中，如下所述，预测可以更复杂。

[0073] 在一个实施方式中，当预测器732识别到未来事件的不止一个预测的等价类时，预测器732还适于基于导致DAG数据结构206的定义中使用的预测和事件的事件序列中接收的事件的统计、语义或内容分析来评价最有可能的预测的等价类。因此，在统计上、语义上或字义上与用于定义通过DAG的特定路径的事件更类似的事件730的到来流中的事件序列可造成特定路径被赋予比替代路径更高的权重(因此更有可能)。然后，预测的下一个等价类可以确定为最有可能的等价路径。

[0074] 另外，在一些实施方式中，预测器732可采用来自导致预测的到来事件流中识别到的事件序列中的事件的包括属性值的事件信息。可使用该事件信息通过基于该事件信息填充预测的事件属性值来生成新预测事件。例如，可基于当前事件序列中的事件之间的间隔来预测时间戳信息。另外，序列扩展关系412充当对预测事件中的潜在属性值的限制，使得所有预测属性值必须至少满足与序列扩展关系412关联的标准。还可使用类似技术来预测其它属性值或这些值的范围或列举。

[0075] 在优选实施方式中，过滤器732和预测器734中的任一个或两个设置有通知器736a、736b，通知器736a、736b是响应于处理事件730的到来流而生成通知的硬件、软件或固件组件。例如，在过滤器732识别与DAG数据结构206表示的任何序列不对应的新事件序列的情况下，通知器736a可生成合适的通知。另外地或另选地，在过滤器732识别与DAG数据结构206表示的序列对应或部分对应的事件序列的情况下，通知器736a可生成合适的通知。类似地，预测器734使用通知器736b生成预测等价类或事件的通知。

[0076] 为了避免有疑问，经过滤器732和/或预测器734处理的按时间排序的到来事件730的流对于用于生成DAG数据结构206的多个事件422而言是有区别的。因此，序列识别设备200针对两个事件集进行操作：用于生成DAG数据结构的第一事件集422；供过滤器732和/或预测器734处理的第二事件集(到来事件730)。本领域的技术人员应该清楚，可另外地使用到来事件730，以通过将到来事件730的流中识别到的事件序列的表示添加到DAG数据结构206中来适应、演变、修改或补充DAG数据结构206，如本发明的实施方式会需要的。

[0077] 本领域的技术人员应该清楚，在过滤器732和预测器734被图示为被包括在序列识别设备200中时，可省略过滤器732或预测器734中的任一个。另选地，可通过单个联合组件或以不同方式细分的组件来提供过滤器732和预测器734提供的功能和设施。另外，可通过序列识别设备200外部的一个或更多个组件(诸如，通过硬件或软件接口或者通过网络与序列识别设备200通信的组件)来提供过滤器732和/或预测器734提供的功能和设施。

[0078] 图8是按照本发明的替代实施方式的图7的过滤器732的方法的流程图。初始地，在步骤850中，过滤器732从多个到来事件730接收新到来事件。在步骤852中，过滤器732确定接收到的到来事件是否扩展了滤波器732当前正在处理的事件序列。该确定是基于之前接收事件的记录、之前识别的部分事件序列和序列扩展关系412。如果接收到的事件没有扩展之前接收到的事件序列，则该方法在步骤856中将接收到的事件作为潜在新事件序列的开始。针对接收到的事件，将DAG数据结构206的遍历初始化成起始节点“S”。

[0079] 另选地，在步骤854中，如果接收到的事件没有扩展之前接收到的部分事件序列，则该方法针对部分事件序列中接收的最近事件，识别之前接收到的部分事件序列和DAG数据结构206中的当前节点。

[0080] 在步骤858中，该方法确定接收到的事件的等价类。在步骤860中，该方法确定所确定的等价类是否匹配DAG遍历中从当前节点的出边。如果等价类不匹配出边，则步骤864得到的结论是接收到的事件没有对应于DAG中的任一条路径并且不符合DAG表示的任一个事件序列，所述方法终止。

[0081] 如果等价类匹配出边，则步骤862针对部分事件序列在DAG中沿着识别的出边遍历DAG数据结构206到达新当前节点。如果步骤866确定新当前节点是结束节点“F”，则所述方法终止，否则所述方法在步骤868中接收下一个到来事件并且重复至步骤852。

[0082] 现在，将仅仅以举例的方式来描述本发明的详细示例性实施方式。在示例性实施方式中，事件数据是带时间戳的表格格式(例如，作为带有存储日期和时间信息的一个或更多个指定字段的用逗号分开的值)并且以序列的方式(逐行地或者以可逐行进行处理的较大群组)到达。表格中的各列具有域D_i和对应的属性名称A_i。存在起到标识符(例如，行数或事件id)作用的特殊域O。正式地，用以下函数表达数据：

[0083] $f:O\rightarrow D_1\times D_2\times\cdots\times D_n$

[0084] 该函数可被书写为以下关系：

[0085] $R \subseteq O \times D_1 \times D_2 \times \ldots \times D_n$

[0086] 其中，任何给定标识符o_i最多出现一次。使用记号$Ak(o_i)$来指代对象o_i的第k个属性的值。

[0087] 本发明的实施方式力求找到排序后的事件序列(随后，类似序列的群组)。为了实现这个，定义序列扩展关系。

[0088] 在示例性实施方式中，事件序列遵守以下规则：

[0089] • 各事件处于至多一个序列中

[0090] • 按日期和时间将序列中的事件排序

[0091] • 通过诸如等价、容差和其它关系的其属性间关系来链接事件及其后继者

[0092] 这些被称为序列扩展关系。注意的是，对于不同的序列，可以具有不同的序列扩展关系。另外，可以动态地改变序列扩展关系。在下述的图结构中，序列扩展关系与图中的节点关联。在示例性实施方式中，不是现有序列的部分的任何事件被视为新序列的开始。对于任何属性A_i，可定义容差关系R_i，其中

[0093] $R_i: D_i \times D_i \rightarrow [0,1]$

[0094] 是反身对称模糊关系并且

[0095] $\forall j: R_i(A_i(O_i), A_i(O_i)) = 1$

[0096] 然后，通过属性A链接的对象的容差类是

[0097] $T(A_i, o_m) = \{o_j / x_{mj} \mid R_i(A_i(o_m), A_i(o_j)) = x_{mj}\}$

[0098] 注意的是，这个集包括(带有成员1)具有属性值$A_i(o_m)$的所有对象。容差种类可被等价表达为成对的集。

[0099] 最后，包括全序关系P_T的情况，全序关系P_T是针对表示时间戳的区别属性(或小属性集)来定义的。然后可定义序列和投影序列：

[0100] $\forall i: \; P_T\left(A_T(o_i), A_T(o_i)\right) = 1$

[0101] $\forall i \neq j: \; P_T\left(A_T(o_i), A_T(o_j)\right) > 0 \rightarrow P_T\left(A_T(o_j), A_T(o_i)\right) = 0$

[0102] $Q(o_t) = (o_i / x_{ti} \mid P_T(o_t, o_i) = x_{ti})$

[0103] 其中，A_T是时间戳属性(或多个属性)并且事件排序将时间排序建模。对于所有i而言，时间属性t_i遵守：$t_i \leq t_{i+1}$。这被当作单个属性，尽管可被存储为不止一个(诸如，日期、一天内的时间)。在示例性实施方式中，针对合适的域，定义多个序列扩展关系$R_1 \cdots R_n$。如果

[0104] $\min\left(Q_T(o_i, o_j), \min_m\left(R_m(o_i, o_j)\right)\right) \geq \mu$

[0105] 则两个事件o_i和o_j有可能链接于同一序列中，即，所有需要的属性满足指定的序列扩展关系达到大于某个阈值μ的程度。因此

[0106] $potential-link(o_i, o_j, \mu) \leftrightarrow \min\left(Q_T(o_i, o_j), \min_m\left(R_m(o_i, o_j)\right)\right) \geq \mu$

[0107] 以及

[0108] $linked(o_i, o_j, \mu) \leftrightarrow potential-link(o_i, o_j, \mu)$

[0109] AND

[0110] $\nexists o_k : \left(potential-link(o_i, o_k, \mu) \quad AND \quad potential-link(o_k, o_j, \mu) \right)$

[0111] 即，如果两个事件满足指定容差和等价关系达到大于某个阈值μ的程度并且没有居间事件，则这两个事件是链接的。

[0112] 在示例性实施方式中，还针对用于比较不同序列中的事件并且将其归类的一些域，定义等价类。针对一个或更多个域的等价类是用各域中的值来表示的—例如，针对自然数定义的关系“hasTheSameParity”可包括诸如(0,2)、(0,4)、(2,4)、(1,5)等对。两个等价类(代表偶数和奇数的集合)可被写[0]和[1]，因为在关系“hasTheSameParity”下所有元素链接到0或1。类似地，对于按天和小时值指代的时间，可针对工作日高峰期(例如，天＝“周一至周五”，小时＝“8、9、17、18”)、其它工作日(例如，天＝“周一至周五”，小时≠“8、9、17、18”)和周末(例如，天＝“周六、周日”)定义等价类。这些可容易地扩展成模糊等价类。等价类分割对象，使得各对象属于考虑的各域的恰好一个等价类。在模糊的情况下，假设重叠类中成员的总和是1，，至少一个成员假设是0.5或更大。在创建图时，仅仅考虑最大的成员。在两个相等成员(例如，0.5)的情况下，使用判断过程来选择一个等价类。正式地，对于指定的属性A_i

[0113] $S(A_i, o_m)=\{o_j | A_i(o_j)=A_i(o_m)\}$

[0114] 相关联的等价类的集合(也被称为基本构思)是

[0115] $C_i=\{S(A_i, o_m) | o_m \in O\}$

[0116] (例如，如下所述的时间和过去的时间)。

[0117] 在命题情况下，C_i只包含一个集合，这个集合中的元素是属性i为真的对象。在模糊情况下，元素等价于某个程度。通过指定成员阈值，提供了等价关系的嵌套集合，使得一旦已知成员阈值，就可如清晰情况下一样进行该技术。操作可扩展到多个属性。使用所选择的属性来找到“EventCategorisation”。这是源自一个或更多个属性(或属性的n元组)的等价类的有序集合。

[0118] $B_k \in \{A_1, \cdots, A_n\}$

[0119] $EventCategorisation(o_i)=([B_k(o_i) | k=1, \cdots m])$

[0120] 即，各B_k是属性中的一个或更多个并且某个对象o_i的事件归类是通过对应于其属性值的等价类来给出的。注意的是，结果并不取决于处理属性的次序。可优化这个次序，以在决策从给定节点跟随哪个边时给出最快性能。对于任一个序列集，可使用如图10和图11中所示的DAG来创建序列的最小表示。该图是没有循环的确定有限自动机。用带标签的边表示各事件。边标签表明可应用于事件的等价类，在以下被称为事件归类。源节点“S”是用于所有序列的单个起始点。为了确保特有的结束节点“F”，向所有序列附加虚拟的“序列结束”(“#END”)事件。

[0121] 现在，将基于2009年的IEEE“Visual Analytics Science and Technology”(VAST)挑战使用的样本数据来描述使用中的示例性实施方式的示例。样本数据模拟雇员经由众多入口进入带徽章锁的房间。总之，数据集中的事件包括六个属性：作为唯一事件标识符的“eventID”；作为是“10”、“11”或“12”的唯一雇员标识符的“Date”、“Time”、“Emp”或“Employee”；作为安全入口的唯一标识符的“Entrance”，或者是“b”对应于访问建筑物，或

者“c”对应于访问建筑物的分类部门;作为是“in”或“out”的访问方向的“Direction”。

[0122] 下表提供了样本数据集。注意的是,为了便于读取以识别事件序列,雇员已经将数据排序,尽管在使用中,事件将是按时间排序的。

[0123]

eventID	Date	Time	Employee	Entrance	Direction
1	jan-2	7:30	10	b	in
2	jan-2	13:30	10	b	in
3	jan-2	14:10	10	c	in
4	jan-2	14:40	10	c	out
5	jan-2	9:30	11	b	in
6	jan-2	10:20	11	c	in
7	jan-2	13:20	11	c	out
8	jan-2	14:10	11	c	in
9	jan-2	15:00	11	c	out
10	jan-3	9:20	10	b	in
11	jan-3	10:40	10	c	in
12	jan-3	14:00	10	c	out
13	jan-3	14:40	10	c	in
14	jan-3	16:50	10	c	out
15	jan-3	9:00	12	b	in
16	jan-3	10:20	12	c	in
17	jan-3	13:00	12	c	out
18	jan-3	14:30	12	c	in
19	jan-3	15:10	12	c	out

[0124] 首先,将序列扩展关系的集合限定为等式和允许的转变关系的集合,以检测候选序列。对于n个事件的候选序列:

[0125] $S_1=(o_{11},o_{12},o_{13},\cdots,o_{1n})$

[0126] 定义以下的计算量

[0127] ElapsedTime $\Delta T_{ij}=Time(o_{ij})-Time(o_{ij-1})$

[0128] with $\Delta T_{i1}=Time(o_{i1})$

[0129] 和限制(对于j>1)

[0130] $Date(o_{ij})=Date(o_{ij-1})$

[0131] $0<Time(o_{ij})-Time(o_{ij-1})\leq T_{thresh}$

[0132] $Emp(o_{ij})=Emp(o_{ij-1})$

[0133] $(Action(o_{ij-1}),Action(o_{ij}))\in AllowedActions$

[0134] where $Action(o_{ij})=(Entrance(o_{ij}),Direction(o_{ij}))$

[0135] 其中,通过图9中的表给出关系“AllowedActions”。在图9的表中,用行指示第一动作,用列指示后面的动作。

[0136] 这些限制可被总结为

[0137] • 单个序列中的事件是指同一雇员;以及

[0138] • 单个序列中的连续事件符合允许在位置之间转变并且在同一天,在彼此的指定时间内。

[0139] 选择合适的时间阈值,诸如,$T_{thresh}=8$。这样确保在最后事件是新序列之后超过8小时的任何事。通过应用序列扩展关系来识别候选序列。任何序列要么在之前已经看到,要么是新序列。根据样本数据,候选序列由以下事件组成:

[0140] 1-2-3-4

[0141] 5-6-7-8-9

[0142] 10-11-12-13-14

[0143] 15-16-17-18-19

[0144] 还定义等价类"EventCategorisation"以比较不同序列中的事件:

[0145] EquivalentAction=I_{Action}

[0146] 对于Direction In,EquivalentEventTime={[7],[8],…}

[0147] 对于Direction Out,EquivalentElapsedTime={[0],[1],[2],…}

[0148] 其中,I是身份关系并且标记[7]代表从7:00-7:59等起的所有开始时间的集合。用这个定义,事件5和10被视为是等价的,因为它们均具有Entrance="b"、Direction="in"且"Time"处于"7:00-7:59"。正式地,

[0149] EventCategorisation(o_5)=([b,in],[7])

[0150] EventCategorisation(o_{10})=([b,in],[7])

[0151] 类似地,事件7和12是等价的,因为它们均具有Entrance="c"、Direction="Out"且"ElapsedTime"处于"3:00-3:59"。识别的各序列被实现为图,该图上标记有序列的事件归类,并且将多个序列组合成代表截至此时看到的所有序列的归类形式的最小DAG,如图10和图11中所示。

[0152] 假设通过唯一编号来指代这些节点,由于图是确定性的,因此各出边是唯一的。因此,边可通过其起始节点及其部分事件归类来指定。还可接受的是,如果关于其起始节点不存在含糊不清,用其部分事件归类标签来表示边。针对节点的"InDegree"、"OutDegree"、"IncomingEdges"和"OutgoingEdges"使用标准定义,从而分别给出入边的数量、出边的数量,入边的集合和出边的集合。函数"Start"和"End"还可应用于边,以便分别找到或设置起始节点和结束节点。另外,可使用函数"EdgeCategorisation"找到边的归类类。另外,可定义函数"ExistsSimilarEdge(edge,endnode)"以当如下时返回"真":

[0153] • "edge"具有结束节点"endnode"、事件归类"L"和起始节点"S1";

[0154] • 第二不同边具有相同的结束节点和事件归类"L",但具有不同的起始节点"S2";以及

[0155] • "S1"和"S2"具有相同的入边:

[0156] IncomingEdges(S1)=IncomingEdges(S2)。

[0157] 如果存在这种边,则通过函数"StartOfSimilarEdge(edge,endnode)"返回其起始节点。函数"CreateNewNode(Incoming,Outgoing)"用入边和出边的指定集合创建新节点。

[0158] 可使用DAG来识别已经看到的事件的序列。如果观察到新序列(即,通过至少一个事件归类在图中与各序列不同的序列),则可使用诸如以下提供的算法将新序列添加到图

中。注意的是，该算法假设图G＝(V,E)，使得新节点被添加到集合V中并且边被添加到集合E中/从集合E中删除。该算法分三个不同阶段来进行。在第一部分和第二部分中，该算法通过新事件序列和DAG从起始节点“S”开始逐步移动。如果事件归类匹配出边，则该算法跟随该边到达下一个节点并且移动到事件序列中的下一个事件。如果新节点具有多于一个入边，则该算法复制它；副本采用刚刚跟随的入边，并且原始节点保持所有其它入边。两个副本都具有相同的输出边集合。该算法的这个部分找到具有一个或更多个公共开始事件的其它序列。

[0159] 如果在某个点，到达不具有匹配下一个事件归类的出边的节点。创建针对序列剩余部分的新边和节点，最终连接到结束节点“F”。注意的是，当序列是新的时，该算法必须到达没有出边匹配下一个事件的归类的点；如果这发生在起始节点“S”，则实际上错过第一阶段。

[0160] 最终，在第三阶段中，该算法搜索具有一个或更多个公共结束事件的序列。只要有可能，就将路径合并。图12、图13和图14示出前两个序列之后、随后在添加第三序列之后并且最终在添加第四序列之后的DAG的发展。

[0161]

```
Algorithm ExtendGraph
Input :Graph G with start node S, end node F, representing the current DAWG
    (minimal)
        CandidateSequence Q[0 - NQ]  representing the candidate sequence;
        each element is an event identifier. The sequence is terminated by
        #END
        NB the sequence is not already present in the graph.
```

[0162]

```
Output : updated minimal graph, incorporating the new sequence

Local variables :  Node startNode, newNode, endNode, matchingNode
       Edge currentEdge, matchingEdge
       Categorisation currentCategorisation
       integer seqCounter;

 startNode = S
 seqCounter = 0

WHILE  EventCategorisation(S[seqCounter]) ∈ OutgoingEdges(StartNode)

    currentEdge = (startNode, EventCategorisation(Q[seqCounter] )
    endNode = End (currentEdge)
    IF InDegree (endNode) > 1
    THEN
        newNode = CreateNewNode({currentEdge}, OutgoingEdges(endNode))
        IncomingEdges(endNode) = IncomingEdges (endNode) - currentEdge
        startNode = newNode
    ELSE
        startNode = endNode
    seqCounter++
ENDWHILE

WHILE seqCounter < NQ                              // create new path
    currentEdge = (startNode, EventCategorisation (S[seqCounter]) )
    startNode = CreateNewNode({currentEdge}, { })
    seqCounter++
ENDWHILE

currentCategorisation = #END
currentEdge = (startNode, #END )          // last edge, labelled by #END
IncomingEdges(F) = IncomingEdges (F) + currentEdge
endNode = F

WHILE nextEdgeSet contains exactly one element (i.e currentEdge)
                AND ExistsSimilarEdge(currentEdge, endnode)
    matchingNode = StartOfSimilarEdge(currentEdge, endnode)
    startNode = Start (currentEdge)
    IncomingEdges(endNode) = IncomingEdges (endNode) - {currentEdge}
    nextEdgeSet = IncomingEdges (startNode)
    IncomingEdges (matchingNode) = nextEdgeSet U IncomingEdges
    (matchingNode)
    endNode = matchingNode
    currentEdge ∈ edgeSet //choose any element,"while" loop terminates if>1
END WHILE

Algorithm ReduceGraph
Input :Graph G, start node S, end node F, the current DAWG (minimal)
       Sequence C[0 - NQ]  representing the sequence of event categories to
       be removed. Each element is an event categorisation. The sequence is
```

[0163]

```
        terminated by #END   NB the sequence must be present in the graph
        and there must be at least one sequence in the graph after removal.

Output : updated minimal graph, excluding the removed sequence

Local variables :  Node startNode, endNode
       Edge currentEdge, matchingEdge
       Categorisation currentCategorisation
       integer seqCounter;

startNode = S
seqCounter = 0
currentEdge = (startNode, C[0])
endNode =  End(currentEdge)

WHILE  endNode   F
    WHILE OutDegree(startNode) > 1
              AND InDegree(endNode) == 1
              AND OutDegree(endNode) == 1
        currentEdge = (endNode, C[seqCounter])
        seqCounter++
    END WHILE
    IF (InDegree(endNode) > 1)
        delete path from startNode to endNode
    startNode = endNode
    currentEdge = (startNode, C[seqCounter])
    endNode =  End(currentEdge)
    seqCounter++
END WHILE
```

[0164] 在所描述的本发明的实施方式至少部分能使用受软件控制的可编程处理装置(诸如,微处理器、数字信号处理器或其它处理装置、数据处理设备或系统)实现的范围内,应该理解,用于将实现上述方法的可编程装置、设备后系统的计算机程序被设想为本发明的一个方面。计算机程序可被实施为源代码或者经历汇编以便在处理装置、设备或系统上实现或者可例如被实施为结果代码。

[0165] 适当地,将计算机程序以机器或装置可读形式存储在载体介质上,例如,存储在固态存储器、诸如盘或带的磁性存储器、诸如压缩盘或数字通用盘等光学或磁-光学可读存储器中,并且处理装置利用程序或其部分,以构造它以便进行操作。可以从在诸如电子信号、射频载波或光学载波中实施的远程源供应计算机程序。这种载体介质还可被设想为本发明的一些方面。

[0166] 本领域的技术人员应该理解,尽管已经针对上述示例实施方式描述了本发明,但本发明不限于此并且存在许多落入本发明范围内的可能变形形式和修改形式。

[0167] 本发明的范围包括本文中公开的任何新颖特征或特征的组合。由此,在执行这个应用或者源自其的任何这些其它应用期间,申请人给出通知,通知可将新权利要求书调节成这些特征或特征的组合。特别地,参照随附权利要求书,可将从属权利要求中的特征与独立权利要求中的特征组合并且可以任何合适方式并且不仅仅以权利要求书中列举的特定组合来组合各个独立权利要求中的特征。

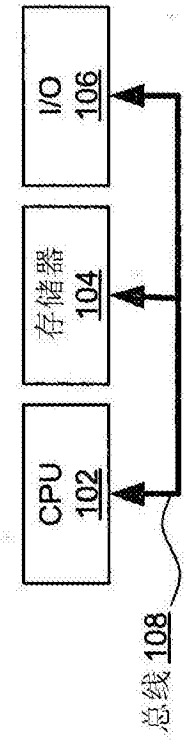

I/O
106
存储器
104
CPU
102
总线 108

图1

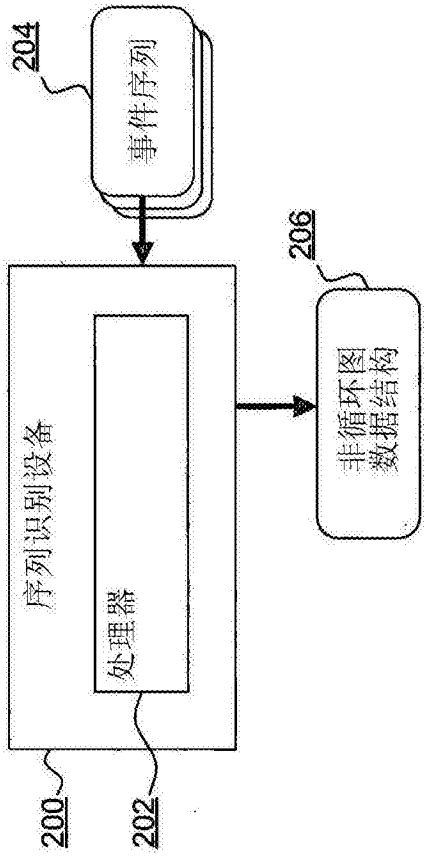

204
事件序列
206
序列识别设备
处理器
非循环图
数据结构
200
202

图2

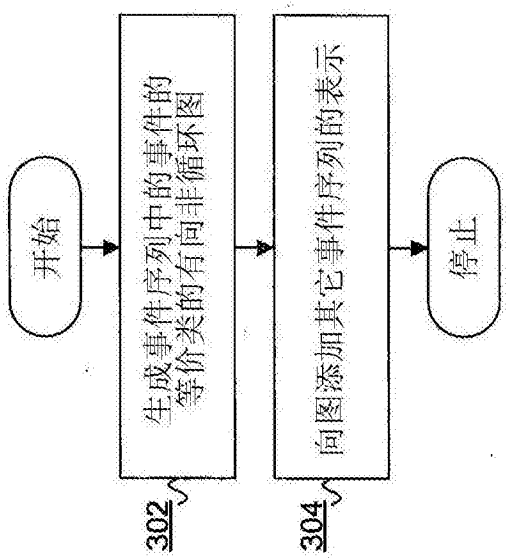

开始
生成事件序列中的事件的
等价类的有向非循环图
向图添加其它事件序列的表示
停止
302
304

图3

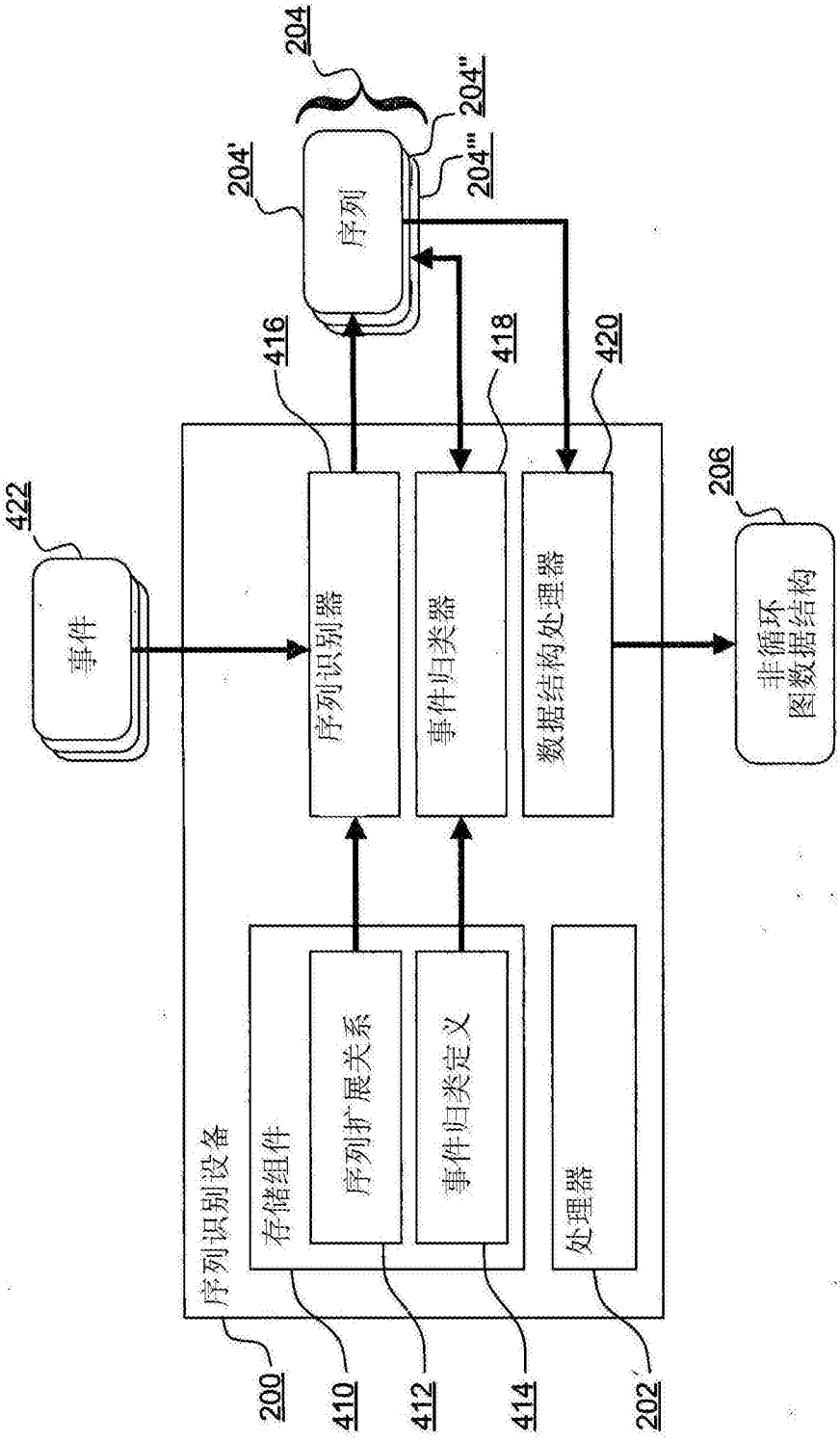

204
204"
204'
204'''
序列
416
418
420
422
206
事件
序列识别器
事件归类器
数据结构处理器
非循环
图数据结构
序列识别设备
存储组件
序列扩展关系
事件归类定义
处理器
200
410
412
414
202

图4

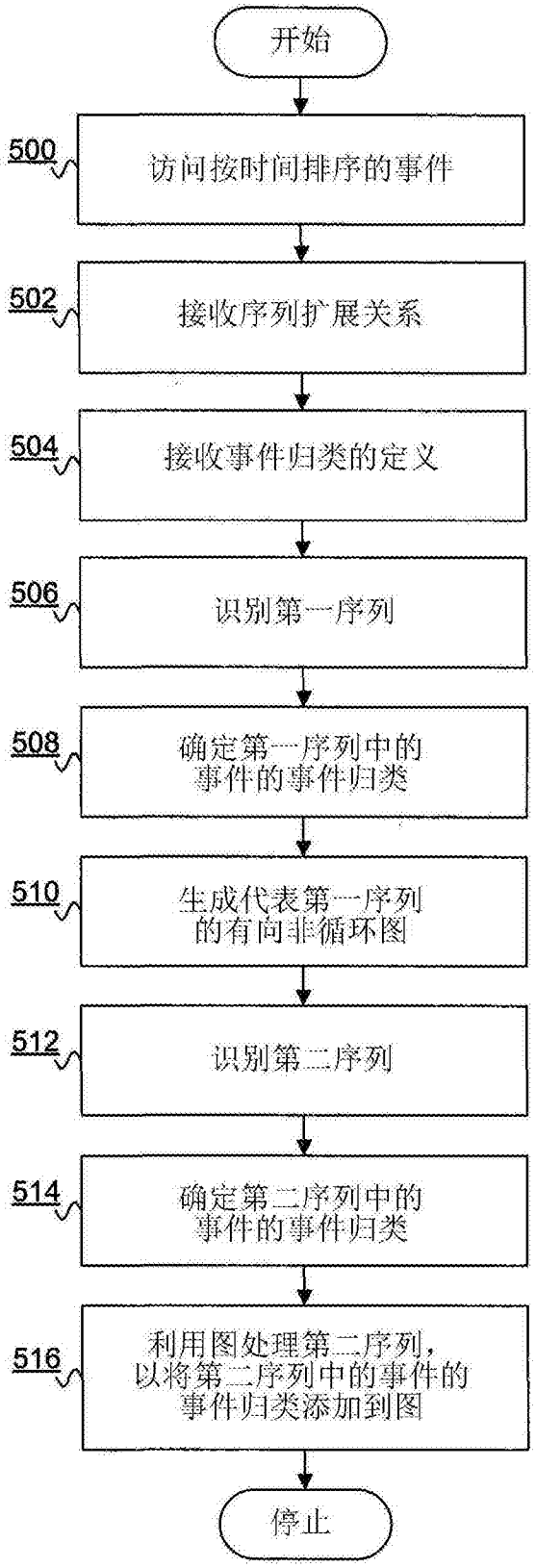

开始
500
访问按时间排序的事件
502
接收序列扩展关系
504
接收事件归类的定义
506
识别第一序列
508
确定第一序列中的事件的事件归类
510
生成代表第一序列的有向非循环图
512
识别第二序列
514
确定第二序列中的事件的事件归类
516
利用图处理第二序列，以将第二序列中的事件的事件归类添加到图
停止

图5

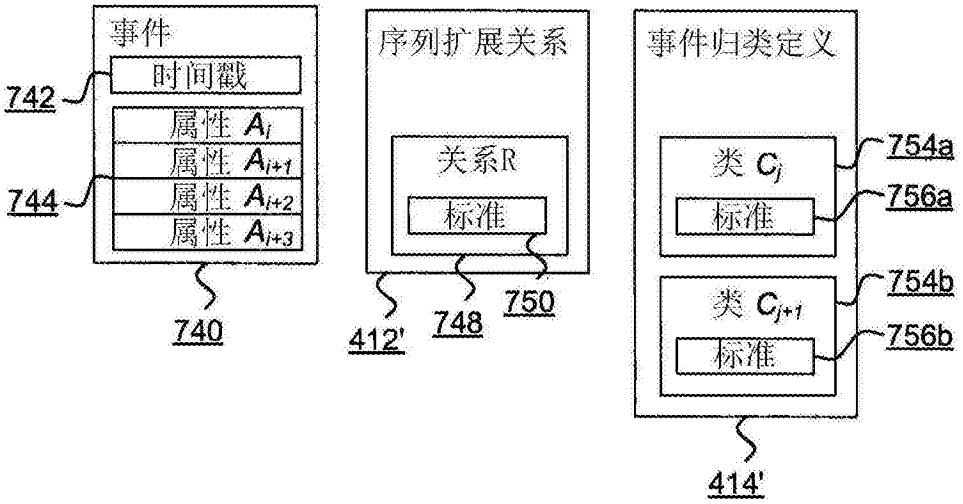

事件
时间戳
属性 A_i
属性 A_{i+1}
属性 A_{i+2}
属性 A_{i+3}
742
744
740
序列扩展关系
关系R
标准
412'
748
750
事件归类定义
类 C_j
标准
754a
756a
类 C_{j+1}
标准
754b
756b
414'

图6a

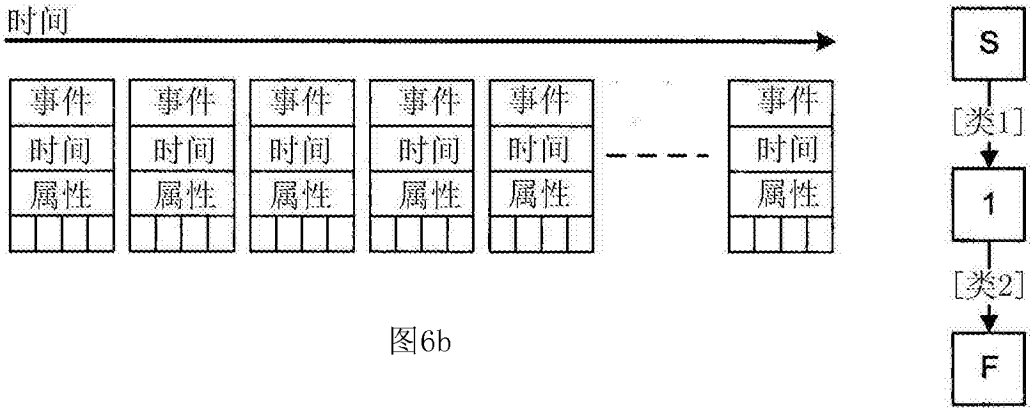

时间
事件
时间
属性
S
[类1]
1
[类2]
F

图6b

图6c

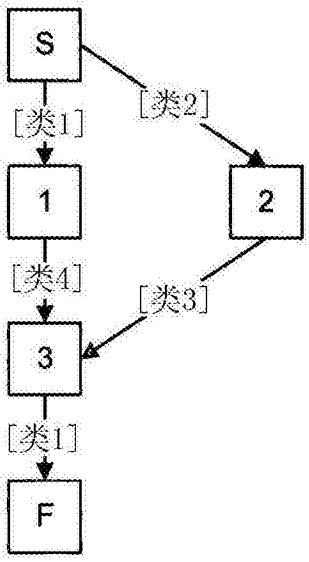


图6d

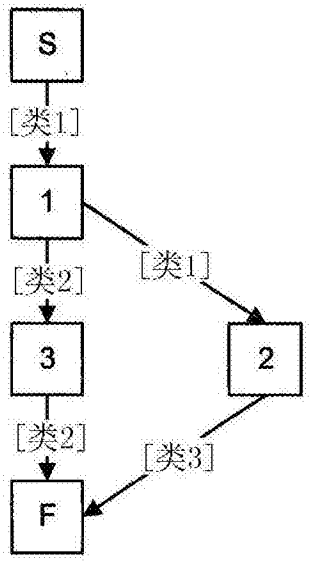


图6e

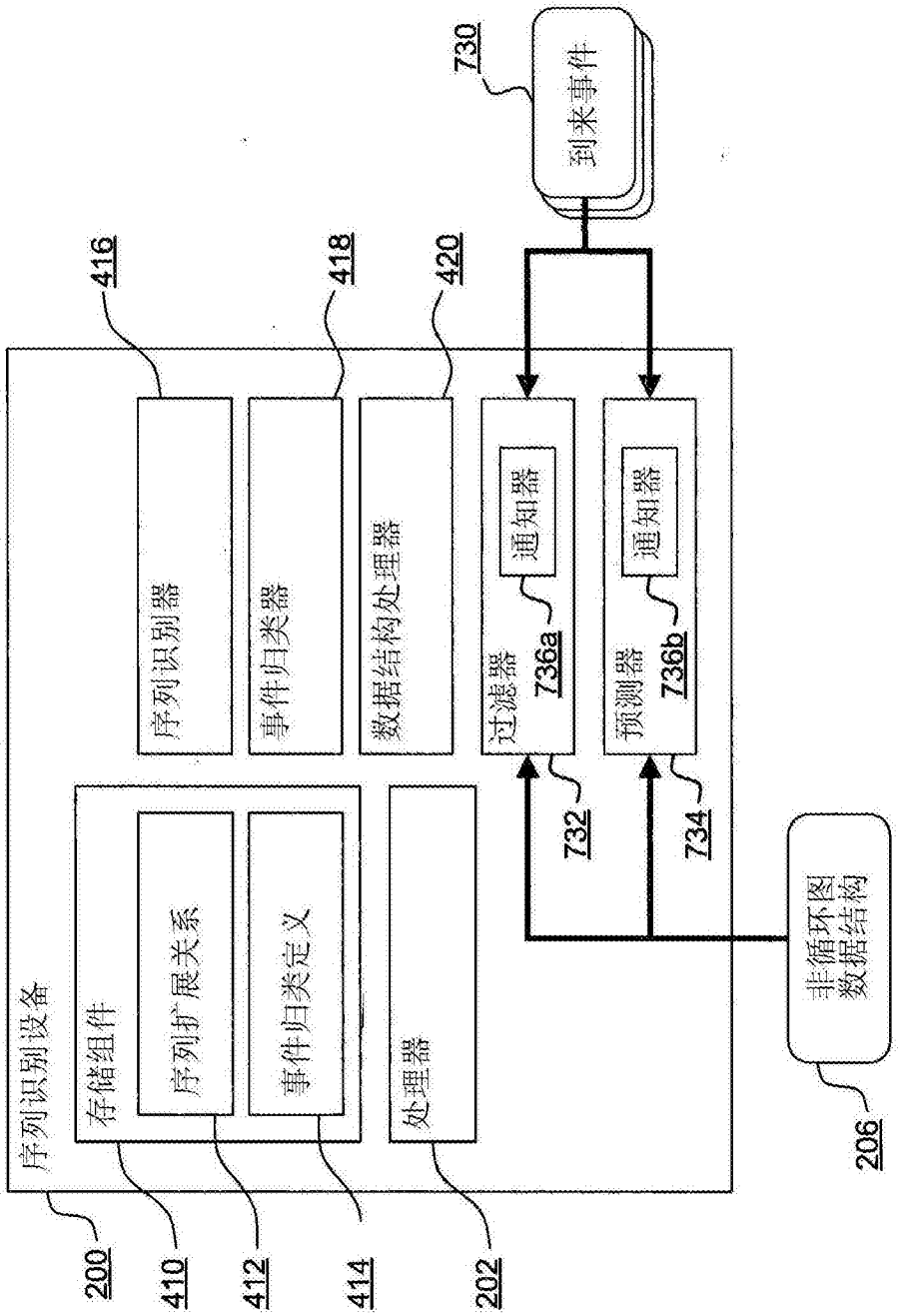

730
到来事件
416
418
420
序列识别器
事件归类器
数据结构处理器
过滤器
通知器
736a
预测器
通知器
736b
732
734
非循环图
数据结构
206
序列识别设备
存储组件
序列扩展关系
事件归类定义
处理器
200
410
412
414
202

图7

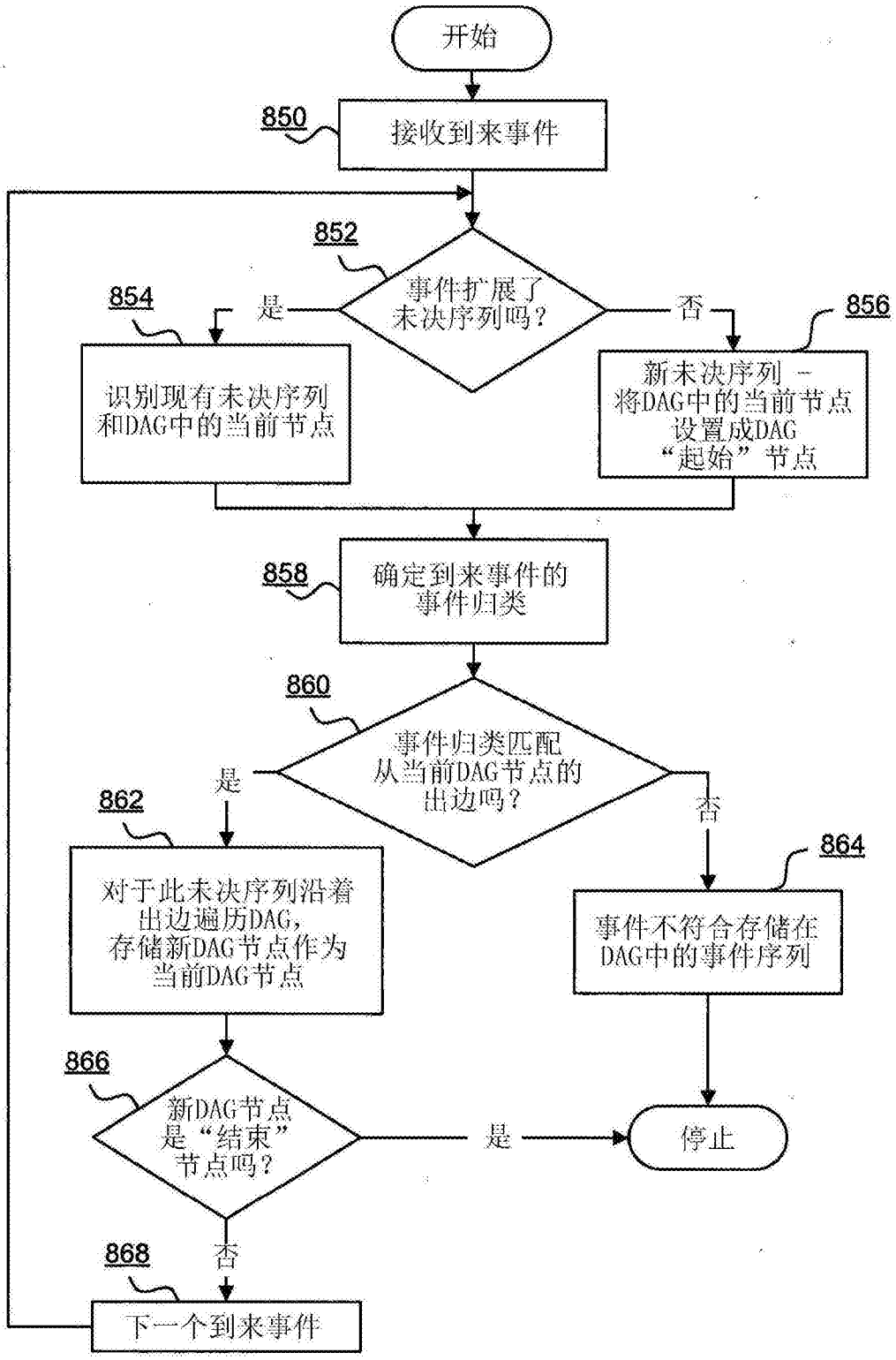

开始
850
接收到来事件
852
事件扩展了未决序列吗？
是
否
854
识别现有未决序列和DAG中的当前节点
856
新未决序列 – 将DAG中的当前节点设置成DAG“起始”节点
858
确定到来事件的事件归类
860
事件归类匹配从当前DAG节点的出边吗？
是
否
862
对于此未决序列沿着出边遍历DAG，存储新DAG节点作为当前DAG节点
864
事件不符合存储在DAG中的事件序列
866
新DAG节点是“结束”节点吗？
是
停止
否
868
下一个到来事件

图8

	b,in	c, in	c, out
b,in	✓		✓
c, in			
c, out	✓	✓	

图9

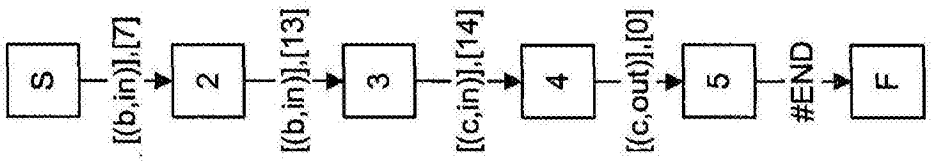


图10

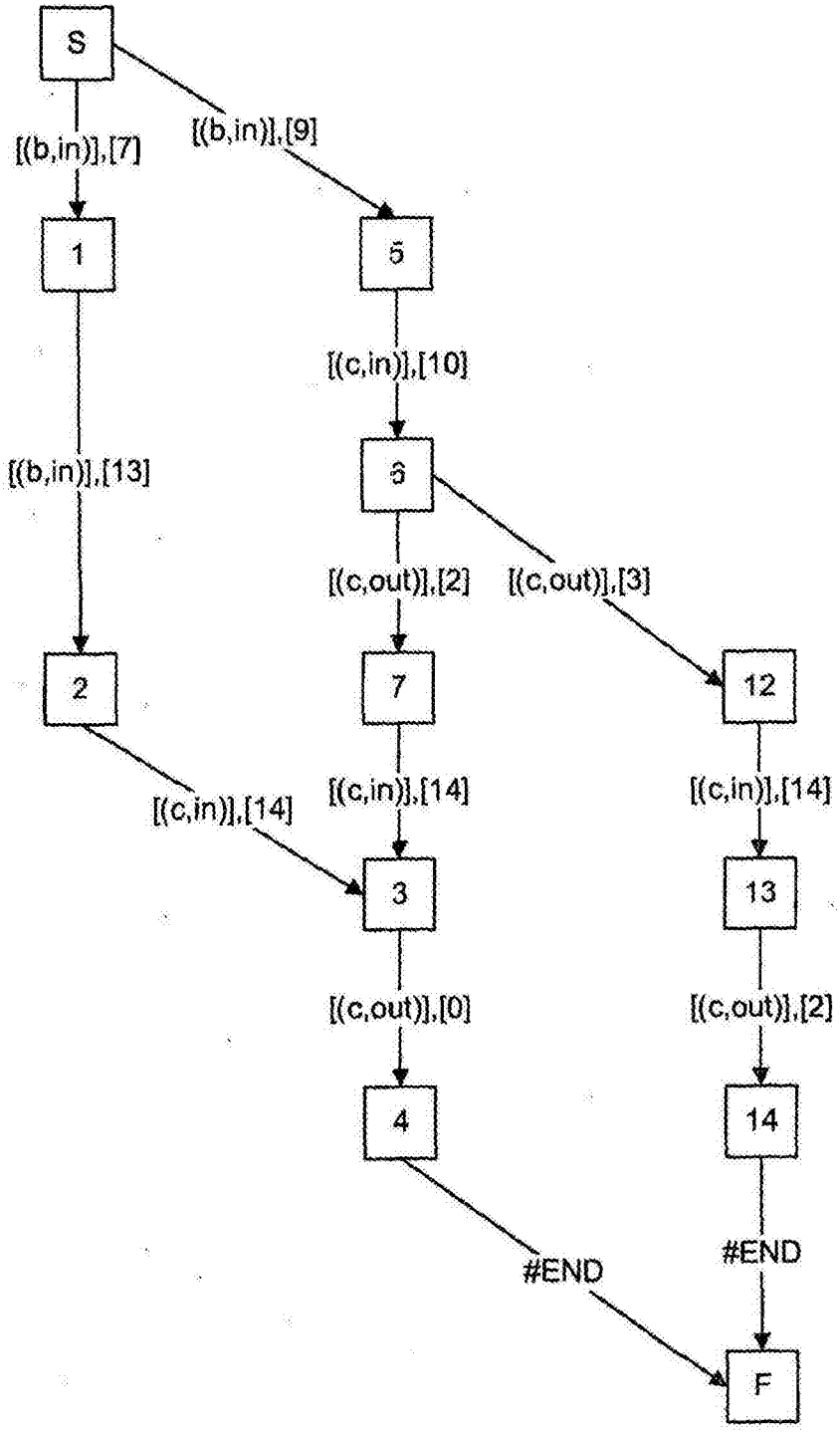

S
[(b,in)],[7]
[(b,in)],[9]
1
5
[(c,in)],[10]
[(b,in)],[13]
6
[(c,out)],[2]
[(c,out)],[3]
2
7
12
[(c,in)],[14]
[(c,in)],[14]
[(c,in)],[14]
3
13
[(c,out)],[0]
[(c,out)],[2]
4
14
#END
#END
F

图11

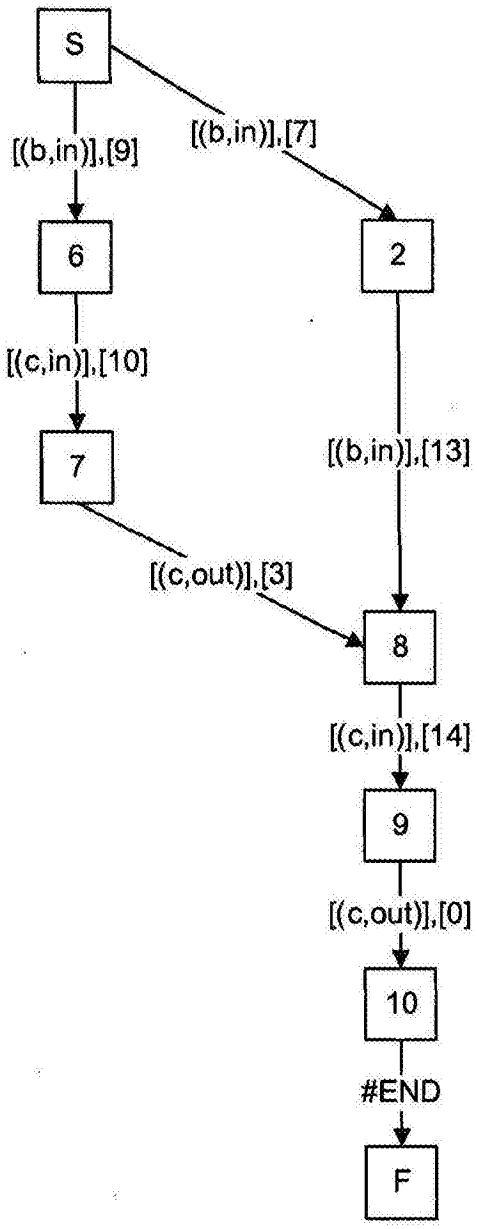

S
[(b,in)],[9]
[(b,in)],[7]
6
2
[(c,in)],[10]
7
[(b,in)],[13]
[(c,out)],[3]
8
[(c,in)],[14]
9
[(c,out)],[0]
10
#END
F

图12

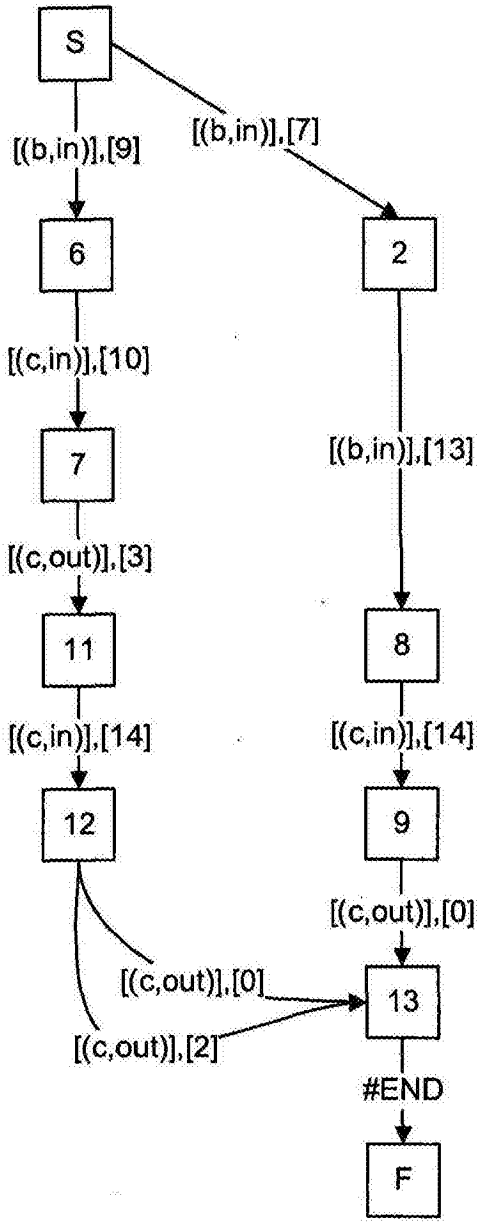

S
[(b,in)],[9]
[(b,in)],[7]
6
2
[(c,in)],[10]
7
[(b,in)],[13]
[(c,out)],[3]
11
8
[(c,in)],[14]
[(c,in)],[14]
12
9
[(c,out)],[0]
[(c,out)],[0]
13
[(c,out)],[2]
#END
F

图13

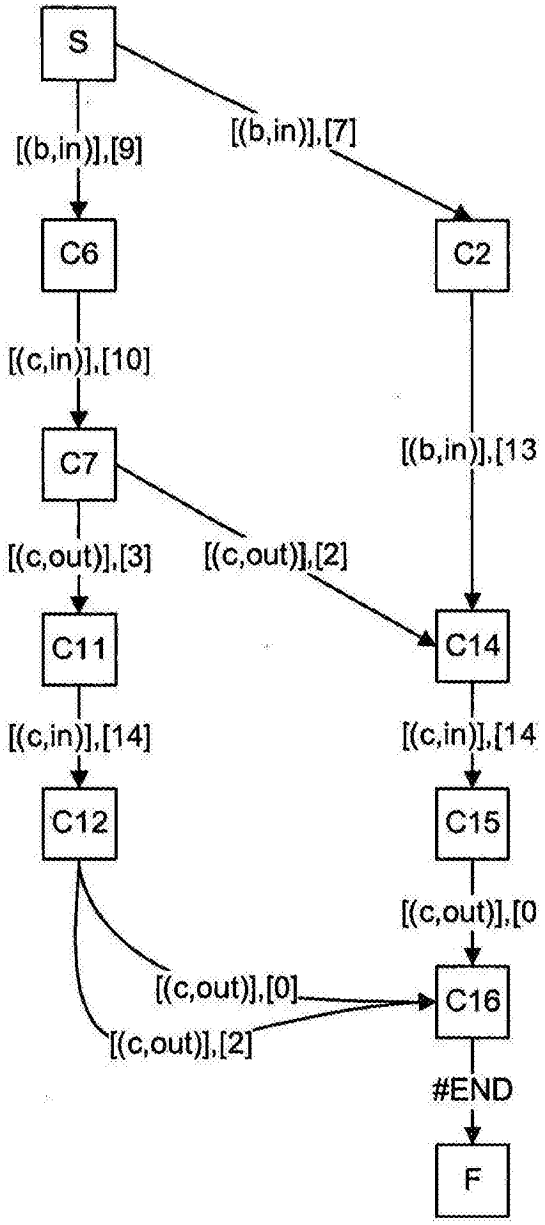

S
[(b,in)],[9]
[(b,in)],[7]
C6
C2
[(c,in)],[10]
C7
[(b,in)],[13
[(c,out)],[3]
[(c,out)],[2]
C11
C14
[(c,in)],[14]
[(c,in)],[14
C12
C15
[(c,out)],[0
[(c,out)],[0]
C16
[(c,out)],[2]
#END
F

图14