复杂事件管理的多元时序数据处理技术研究

李志国[1] 钟 将[2] 钟璐蔓[1]
(重庆工商大学管理学院 重庆 400067)[1] (重庆大学计算机学院 重庆 400044)[2]

摘 要 随着数据量变得不断庞大,将不同业务系统数据融合在一起挖掘潜在价值变得越来越有意义。复杂事件处理技术就是将业务数据抽象为事件序列,通过复杂事件描述方法将有潜在价值的复合数据描述为特定的事件匹配结构。复杂事件检测引擎从大量事件流中检测出满足匹配结构的事件序列,最终输出数据融合结果。但传统复杂事件描述只适用于输入事件流为单一原子事件类型,且谓词约束为简单的属性值比较或聚合操作,事件间为简单的时序约束。这使得传统检测方法无法满足诸如医学、金融等对时间要求比较精确、事件谓词约束要求更加丰富的应用领域。因此,设计了一种能够支持多元事件输入的基于TCN的量化时序约束表示模型和基于时段特征约束的谓词约束表示模型,并且提出了并行化的复杂事件检测算法(PARALLEL-TCSEQ-DETECTION检测算法),使得复杂事件检测方法更加高效。对2045支股票2亿条记录的分析结果表明了提出的复杂事件处理技术的可行性与高效性。

关键词 CEP,TCN,时序特征,事件检测模型,并行
中图法分类号 TP391 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2019.06.007

Study on Processing Technology for Complex Event Management Based on Multivariate Time Series Data

LI Zhi-guo[1] ZHONG Jiang[2] ZHONG Lu-man[1]
(School of Management,Chongqing Technology and Business University,Chongqing 400067,China)[1]
(College of Computer Science,Chongqing University,Chongqing 400044,China)[2]

Abstract As the amount of data becomes bigger and bigger,it is increasingly meaningful to combine different business system data to mine potential values. The complex event processing technology abstracts the business data as an event sequence,and describes the potentially valuable composite data as a specific event matching structure through the event description method. Then the event detection engine detects the event sequence meeting the matching structure from a large number of event flows,and finally outputs the data fusion results. However,in the traditional event description,the input event flow of the event engine is a single atomic event type,the event predicate constraint contains a simple attribute value comparison operation or simple aggregation operation,and the time constraint between events is simple. This makes the traditional detection method cannot be suitable for some application fields in which the time is required to be more accurate and the event predicate constraint is required to be more complex,such as medicine and finance. In light of this,this paper designed a multivariate event input supported quantitative timing constraint representation model based on TCN and predicate constraint representation model based on time-interval feature constraint,and proposed a parallel detection algorithm for complex events(PARALLEL-TCSEQ-DETECTION). The method makes the complex event detection more efficient. The analysis results based on 200 million records of 2045 stocks demonstrate the validity and high efficiency of the proposed processing technology for the complex events.

Keywords CEP,TCN,Timing feature,Event detection model,Parallel

1 引言

近年来,随着计算机、网络、通讯、存储等技术的蓬勃发展,交互式异构类型数据正以指数速度增长。但目前,用户只能在大多数分布式应用系统中直观地感知单一系统数据的行为事件,无法处理大量来自不同系统和不同数据源的数据事件。同时,随着人工智能技术的发展,人们对系统查询的要求越来越高,简单的数据检索功能已经无法满足需求,因而亟需能够快速有效地响应复杂的包含业务逻辑和语义的查询检索系统[1]。为了能够应对新兴的用户需求,并挖掘多系统数据事件之间隐藏的潜在价值,需要对数据进行融合,得到复合的数据结果。因此,基于多元时序事件流的事件检测处理技术

到稿日期:2018-10-08 返修日期:2018-12-07 本文受国家重点研发计划(2017YFB1402400),国家高技术研究发展863计划(2015AA015308),中央高校业务基金项目(106112014CDJZR188801)资助。
李志国(1977-),男,博士,高级工程师,主要研究方向为数据科学、战略管理、精准招商、区块链等,E-mail:lizhiguo@ctbu.edu.cn;**钟 将**(1974-),男,教授,博士生导师,主要研究方向为数据挖掘、可信计算机系统、服务计算;**钟璐蔓**(1974-),女,硕士,助教,主要研究方向为区域经济大数据、复杂事件处理,E-mail:lunazhong@ctbu.edu.cn(通信作者)。

逐渐成为研究的热门课题。

1.1 复杂事件处理机制

由文献[2-4]可知,事件处理技术主要被划分为两种模型:数据流处理模型(Data Stream Processing,DSP)和复杂事件处理模型(Complex Event Processing,CEP)。数据流处理模型主要考虑快速地从大量实时的流式数据中查询数据并且实时更新结果。一般数据源是实时且不间断的,用户的响应时间也是实时的,比如从大型网站流式数据中根据 PV/UV 快速给出用户访问和搜索的内容。数据流处理模型为了做到及时响应,只能做一些简单的数据处理操作。而复杂事件处理模型主要从事件驱动业务[5](Event-driven Business)的思路出发,将系统产生的每一条数据记录看作一个事件,输入的数据流即为事件流,复杂事件处理引擎会对事件流进行过滤、关联等一些操作,然后输出一个更高层次的新事件。复杂事件处理模型对事件流所作的操作一般是基于领域专家的某种规则描述进行的,描述规则一般包含用户感兴趣事件的语义,也就是说,复杂事件的处理更加侧重于识别当下发生的某一个事件,并且能够为用户反馈事件结果。

1.2 复杂事件描述

目前,在复杂事件处理过程中,首先需要进行事件描述,然后由事件检测引擎来解析和识别事件描述模式,最终基于相关检测算法,完成事件检测过程并输出满足约束条件的高层次新事件。而复杂事件描述方法的研究方向目前主要集中在 EDL 事件描述语言的定义的研究上,其直接影响复杂事件描述的精确度和易读性。

2000 年,Kam 等提出了分层的事件描述模型[6],在时序关系表示方面扩展了 ALLEN[7] 的区间代数表示,但该模型存在严重损耗。2003 年,加州大学伯克利分校的 Chandrasekaran 等提出了与传统的关系型数据查询语言 SQL 类似的事件描述语言 TelegraphCQ[8],该描述语言支持窗口操作、时序操作、历史数据的分析等功能。2006 年,斯坦福大学支持的数据流处理语言 CQL[9] 使用了关系→关系操作符、关系→流操作符和流→关系的操作符定义规则;但在事件处理过程中无法记录时间信息,导致其无法面向时序数据进行复杂事件处理。2008 年,Patel 等[10] 和 Wu 等[11] 都针对 Kam 等[6] 提出的有损方法进行了改进,提出了无损的事件描述模型,但其表示方法缺乏事件之间时间关系的约束表示和整个待匹配序列时间窗口的约束表示。2014 年,Brenna 等提出了基于 CQL 引擎且支持查询视图的复杂事件处理方法。包含序列操作和迭代操作的事件描述语言 CEL 被成功应用于不支持约束窗口的 Cayuga 系统[12-13] 中。Gyllstrom 等最早提出了一种拥有能够执行事件序列的描述语言并将其应用在缺乏对聚合操作支持的 SASE 系统[14] 中。SASE+[15] 对 SASE 系统语言进行扩展,增加了对克林闭包、否定和聚合操作的支持,同时包含一个特定的事件约束窗口,但其在时间关系表示方面存在缺陷,无法表示事件之间的时序关系。2015 年,Cugola 等针对事件检测中的不确定性描述进行了相关研究[16]。

以上复杂事件描述语言普遍存在不足,整体对时序关系的处理不够精确,而在面向时序数据的事件处理领域,时间关系是描述事件的主要属性,时序关系的缺失和弱化大大降低了事件语义的描述能力。因此,现阶段迫切需要一个有复杂的谓词属性约束能力且能够描述丰富时序语义的复杂事件处理描述语言。

1.3 复杂事件检测

在复杂事件检测处理技术方面,国内外学者已经进行了广泛的研究。传统研究事件检测系统主要处理 RFID 传感器网络标签获得的数据[17-18],或者监控服务收集的数据,这些数据面向的领域都比较单一,所以事件复杂度也比较小,只停留在简单属性值约束、时间窗口约束等[19]。2006 年,Alves 等成功提出以“事件-条件-动作”(Event-Condition-Action,ECA)模式处理事件[20];两年后,更进一步提出面向时序关系约束的原子事件检测方案[21]。同年,UC Berkeley 大学最早提出了将业务数据作为事件进行处理的 SASE 事件处理系统[14],其将事件语义转换为一系列时序关系、数值约束、时序约束、否定等操作组合的描述,同时,将不同操作分别作为操作符进行逻辑处理,但其一直没有考虑重复匹配的问题。2007 年,Diao 等在 SASE 的基础上加入含有表述待匹配事件序列克林闭包的 SASE+系统[15],由于只采用了滑动时间窗口进行时序约束,SASE+对于事件之间时间关系的处理比较简单。同年,EsperTech 公司开发出开源的 Java 复杂事件处理引擎 Esper[22],该处理引擎由于可用于股票交易系统、网络风险评估、高级监测等包含大量事件流的应用系统而受到了学术界和工程界的广泛关注。Esper 利用有限状态自动机机制进行事件检测,使用类似 SQL 语言的表示方法来描述复杂事件包含的语义,以实时、高效地处理大量事件流数据。与此同时,Cornell 大学开发出与 Esper 处理机制类似的 Cayuga 事件处理系统[12-13],其内部采用一套高效的命名方法实现复杂事件检测处理,同时增加了简单的重复操作和一些时序约束。但 Cayuga 内部采用的命名机制导致用户不能很方便地理解,并且也不支持否定(Negation)操作。

以上系统基本是基于自动机原理实现的,当一个满足复杂事件约束的原子事件出现时,自动机就会自动进行状态转换。但简单的自动机无法匹配已经检测到的事件,且不会考虑原子事件流中事件之间的时间关系,这主要是由于以往的事件检测系统较多地一些 RFID 传感器网络标签获得的数据[17-18]或者监控服务收集的数据,而其面向的数据领域单一,事件复杂度也较小,无法适应大数据时代的复杂异构数据,且难以结合人工智能技术从海量数据中抽取有价值的资源,发挥复杂事件检测技术的最大价值,从而更好地为人类服务。

因此,本文研究的意义在于:针对传统复杂事件检测方法中存在的事件时序约束和谓词约束描述能力的不足,提出支持多元事件输入、量化时序约束和复杂谓词约束的面向多元时序数据的复杂事件检测方法,以有效增强复杂事件的描述和处理能力。

2 复杂事件描述模型

2.1 基于 TCN 的时序关系表示模型

传统复杂事件描述语言主要存在时序关系语义描述不精确的问题,进而导致在面向时序数据的事件检测处理中,不能很好地定义和描述用户感兴趣的事件中包含的时序关系。因此,本文提出一个时间约束网络(Temporal Constraint Net-

work，TCN）来表示事件间的时序关系，从而提升复杂事件描述语言在时序关系上的描述能力。

在本模型中，时间被视为一个事件发生过程持续长度的度量片段。连续的时间片段可以由起始和终止两个固定点及其长度确定。按照时间片段的长短，时间可以被划分为时间点和时间区间。1986 年，Vilain 最早提出时间点描述事件的概念[22]，但由于其过于简单，无法描述时序关系比较复杂的事件，随后各领域的专家开始采用时间区间来描述事件的发生时间[7,23-24]。本文提出的复杂事件描述方法在处理时间关系时借鉴了 Allen 的 7 种关系时间区间表示思路[6]，并结合了基于端点的描述方法[22]。事件的发生时间都基于时间区段表示，包含开始时间和结束时间，瞬时事件被看作起始时间和终止时间相等的事件。因此，假设事件 E 为一个瞬时事件，则 E 的发生事件可以表示为$[E_{t.s}, E_{t.e}]$，其中 $E_{t.e}-E_{t.s}=0$。结合端点描述的时间区间表示如图 1 所示。

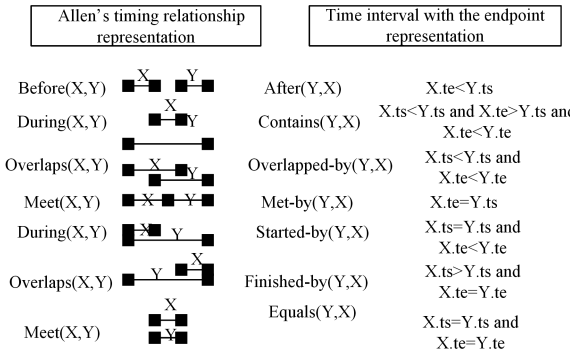

图 1　Allen 的时序关系表示（左）与时间区间端点表示（右）

Fig. 1　Allen's timing relationship representation(left) and time interval with endpoint representation(right)

2.2　基于时段特征的谓词约束表示模型

由于用户感兴趣事件的描述主要包含时序关系和属性值关系的表示，本文在结合 TCN 模型增强量化时序关系语义描述能力的同时，针对事件的属性值约束进一步加以刻画。传统的复杂事件描述方法在定义事件的属性值约束时，往往采用较简单的操作，如等值比较、结合操作、分离操作或者简单的聚合操作，但针对较为复杂的事件类型，传统方法的语义描述能力略显单薄，因此，本文提出基于时段特征的约束表示模型。该模型的数据特征可以表示数据满足的某种特征，如最大、最小、相等、较大、较小等。而时段特征是指数据在某一时间段下的数据特征，比如一段时间内的最大值、平均值等。由于用户感兴趣的事件往往隐含较为复杂的数据特征，并且事件之间存在量化时序关系约束，在复杂事件描述中无法照搬以往较简单的时段特征，因此需要结合数理统计模型来增强时序事件数据特征的表示。

根据数理统计模型，时段特征约束表示模型分别引入了时域特征和频域特征。时域特征包括众数、中位值、四分位差、均值、方差、标准差、极值、过零点、边界点、波段的长短峰值、离散系数等；频域特征包括功率谱、功率密度比、中值频率、平均功率频率等。可以看出，时段特征有着丰富的分类，用户感兴趣的事件都可以找到合适的时段特征来描述约束。比如，对于股票市场一天内出现大幅波动的事件，大幅波动转换到时段特征上一般对应着均值、方差、波段长度峰值的变化。

2.2.1　基于时段特征的谓词约束的定义

首先，利用操作数定义操作类型与相应的计算表达，并借助谓词约束具体的约束条件。具体定义如下：

定义 1（操作数，Operate）　操作数是可被引擎执行的最基本的谓词结构，形式化定义如下：

$Operate::=\langle operatotype, operateDescription\rangle$

将操作数抽象为二元组形式。$operatotype$ 表示操作的类型（数值运算、属性运算、时段特征函数运算），$operateDescription$ 表示谓词操作数对应的计算表达。

定义 2（谓词，Predicate）　谓词表示谓词约束中的一个具体约束条件，由一个完整的布尔表达式构成，包括左右操作及连接运算符，其形式化定义如下：

$Predicate::=\langle predicateid, leftOperate, rightOperate, logicalOperate\rangle$

谓词采用四元组形式表示：$predicateid$ 为谓词的 ID，表示该谓词在谓词约束集合中的顺序；$leftOperate$ 为谓词的左操作数；$rightOperate$ 为谓词的右操作数；$logicalOperate$ 为操作数之间的操作符。

2.2.2　基于时段特征的谓词约束的解析

用户感兴趣事件包含的语义通过事件描述结构进行表述，在处理事件的谓词约束语义之前，需要完成谓词约束解析。如上文所述，谓词约束主要由多个布尔表达式组合而成，其以字符串序列的方式进行存储表示。而在复杂事件处理引擎中，谓词约束以固定的数据结构进行操作，故需要将谓词约束字符串解析为检测模型中定义的谓词结构和操作数结构。同时，在解析谓词约束时也包含识别布尔表达式类型和操作符的过程。在解析谓词约束的过程中，首先需要对约束描述进行分析，实例描述一共包含 5 个布尔表达式，即对应 5 个 Predicate 谓词结构。解析过程如下。

1）约束字符串的切割

在本文扩展的复杂事件描述中，谓词约束由多个布尔表达式以 AND 关键字相连构成，每一个布尔表达式又满足特定的结构，因此需要按照关键字对谓词描述进行切割，识别包含描述结构的所有表达式。

2）布尔表达式到谓词结构的转换

以表达式 $a.price\%500==0$ 和 $avg(b[i].price)<850$ 的转换为例，每一个布尔表达式可以看作“左操作＋运算符＋右操作”的形式，故可以按照运算符进行切割，将布尔表达式分为两部分，即左、右操作数。实例中包含的运算符分别为 ==和<。然后进行操作数的识别，操作数分为 3 种情况：简单的数值、时段特征函数、属性值运算。3 种操作数的识别通过关键字匹配和正则表达式完成。数值操作数序列中只存在数字类型字符；时段特征函数序列包含具体的特征函数名称，常见的特征函数包括{$avg, var, sum, \cdots$}；属性值运算操作数序列包含属性名称、运算符、数值等。

现假设用户感兴趣事件的谓词约束与最终的解析结果的描述分别如下：

Q1：$TCSEQ$（$stockQoute\ a$，$stockQoute+b[]$，$stockQoute\ c$）

WHERE $a.price\ \%\ 500==0$

AND $b[1].price>600$

AND $b[i].price>b[i-1].price$

AND $850>avg(b[i].price)$

AND $c.pirce<1000$

WINDOW(10000 second)

Q2:[

{$Predicate1\langle 1,Left1,Right1,'==\rangle'$;

$Left1=\langle attrbuteType,'a.price\%500\rangle'$;

$Right1=\langle noType,'0'\rangle\}$;

{$Predicate2\langle 2,Left2,Right2,'\rangle'\rangle$;

$Left2=\langle attrbuteType,'b[1].price\rangle$;

$Right2=\langle noType,'600'\rangle\}$;

{$Predicate3\langle 3,Left3,Right3,'>'\rangle$;

$Left3=\langle attrbuteType,'b[i].price\rangle$;

$Right3=\langle attrbuteType,'b[i-1].price'\rangle\}$;

{$Predicate4\langle 4,Left4,Right4,'>'\rangle$;

$Left4=\langle noType,'850'\rangle$;

$Right4=\langle aggreaType,'avg(b[i].price)'\rangle\}$;

{$Predicate5\langle 5,Left5,Right5,'<'\rangle$;

$Left5=\langle attrbuteType,'c.pirce'\rangle$;

$Right5=\langle noType,'1000'\rangle\}$;

]

2.2.3 谓词约束的绑定与模型处理过程

在完成谓词约束解析之后，按照谓词约束结构包含对应状态的标识符，将不同解析结果分别绑定到对应状态的自动机结构上。依次遍历各状态和谓词结构，当满足匹配时直接初始化对应状态的谓词约束属性，并且一个状态可能包含多个谓词结构。基于时段特征的谓词约束表示模型的处理过程如图 2 所示。

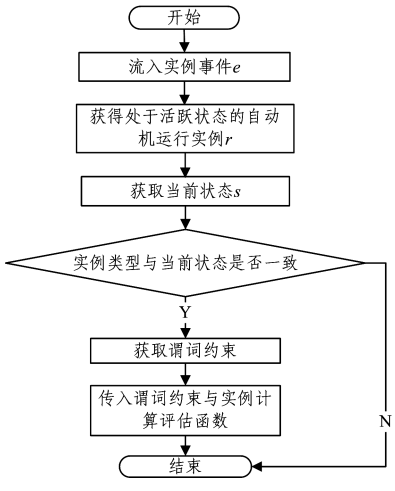

图 2 基于时段特征的谓词约束表示模型

Fig. 2 Predicate constraint representation model based on time-interval feature

2.3 复杂事件描述模型

在进行事件描述时，事件是复杂事件检测系统中最基本的元素。传统的事件分类和定义方式可以根据粒度的不同分为原子事件和复杂事件，也可以根据持续时间的不同分为瞬时事件和区间事件。但传统复杂事件检测系统对输入事件实例类型有严格的限定，输入大多只能支持最基本的原子事件，输出为满足约束条件的复杂事件，而检测引擎不支持多元输入事件流。因此，本文对原有的事件定义方式进行改进，按粒度将事件分为原子事件、聚合事件和复杂事件，进而使得检测引擎的输入可以包含多元事件流。

定义 3(原子事件) 原子事件是业务系统记录的最基本的数据流单元，直接由事件产生最小原子语义描述单元，例如，股票在某一时刻的报价、ICU 重症病房病人在某一时刻的血压监测值、供应链管理系统在某一时刻对于商品的标定等。因此可以将原子事件形式化定义如下：

$$Event=\langle event\_id,event\_type,value,time\_interval\rangle$$

其中，$event\_id$ 为事件 ID，是事件的唯一标识符；$event\_type$ 为事件的类型；$value$ 为事件发生时对应的属性值；$time\_interval$ 为事件的发生时间的二元组，包含开始时间点和结束时间点。

定义 4(聚合事件) 聚合表示数据子集进行的某种运算。聚合事件指在原子事件的基础之上，通过时段特征函数进行聚合运算，得到满足条件的新事件。聚合事件相较原子事件具有更高层次的语义描述。例如，股票 A 在 10:00 到 10:05 期间的平均报价、ICU 重症病房病人在一天内的血压上升比例等。这些都是基于对基本的原子事件应用时段特征函数进行聚合运算得到的，因此可以将聚合事件形式化定义如下：

$$\begin{aligned}&AggregationEvent[E;F][W][H]::=\\&\{\langle e_{id},o_{id},a_{id},value,time\_interval\rangle\mid(Q(\{e_1,e_2,\cdots,e_m\}))\wedge i\in\{1,2,\cdots,m\}(e_i\in E(H))\wedge value=F(\{e_1,e_2,\cdots,e_m\})\wedge m\geqslant 1\wedge i\in\{1,2,\cdots,m\}(o_{id}=e_{i.o\_id})\wedge\lceil\max(e_{i.te})_{i\in\{1,2,\cdots,m\}}-\min(e_{i.ts})_{i\in\{1,2,\cdots,m\}}\rceil=W\wedge time\_Interval_{.ts}=\min(e_{i.ts})_{i\in\{1,2,\cdots,m\}}\wedge time\_Interval_{.te}=\max(e_{i.te})_{i\in\{1,2,\cdots,m\}}\}\end{aligned}$$

其中，E 为原子事件类型，用于限定待匹配事件序列的唯一事件类型；F 为时段特征映射，主要为 avg，var，sum 等的聚合函数；W 为时间窗口，为从开始进入第一个原子事件到检测结束所允许的最大持续时间，W 保证了原子事件之间的关联性；H 为输入事件的轨迹，如 $E[H]=\{e_i\mid i\in(1,\cdots,m)\}$，$e_i$ 的事件类型为 E(在聚合事件检测中，默认事件输入流中的每一个事件实例类型都相同)。

定义 5(复杂事件) 复杂事件是由一系列原子事件、聚合事件或复杂事件通过再次复合得到的新事件。例如，股票 A 在一天内出现大幅上涨后第二天的报价、ICU 重症病房病人在血压上升一个小时以后心率的检测值上升事件等。因此可以将复杂事件形式化定义如下：

$$\begin{aligned}&ComplexEvent[TCN][Q][E_1,E_2,\cdots,E_m][W][H]::=\\&\{\langle e_1,e_2,\cdots,e_m\rangle\mid TCN(e_1,e_2,\cdots,e_m)\wedge Q(\langle e_1,e_2,\cdots,e_m\rangle,O)\wedge\langle e_1,e_2,\cdots,e_m\rangle\in E_1(H)\times E_2(H)\times\cdots\times E_m(H)\wedge\max(e_{i.te})_{i\in\{1,2,\cdots,m\}}-\min(e_{j.ts})_{j\in(1,2,\cdots,m)}<W\}\end{aligned}$$

其中，TCN 为事件之间的量化时序关系约束，在复杂事件检测中，其可以量化描述不同事件之间的时间关系，进而使得事件中包含的时序语义更加丰富精确；Q 为复杂事件约束中的

值量化约束，即谓词约束；$\{E_1, E_2, \cdots, E_m\}$ 表示待匹配序列中事件类型的一个集合，与聚合事件描述不同，复杂事件的输入可以是原子事件、聚合事件、复杂事件等多元事件，输入类型不唯一；W 为复杂事件在进行检测时需要满足的时间窗口；H 为输入事件流轨迹，$E_x[H]=\{e_i \mid i \in (1, \cdots, n)\}$，$e_i$ 的事件类型就是 E_x，即 $E_x[H]$ 表示整个事件流中所有类型为 E_x 的事件实例集合，且 H 包含多元事件类型。

定义 6（事件序列）　事件序列表示事件在指定时间区间上发生的所有事件的集合，比如时间区间 $[T_{i.s}, T_{i.e}]$，$T_{i.s}$ 为区间开始时刻，$T_{i.e}$ 为区间结束时刻。E_i 表示在该区间上出现的所有事件序列。

3　复杂事件检测模型

3.1　基本复杂事件检测模型

复杂事件检测的主要任务是按照事件描述结构进行事件候选集的生成、过滤、聚合，最终输出检测结果，该过程本质上是约束匹配的过程。传统的约束主要为简单时间窗口约束和属性值约束，这导致约束条件语义不够丰富，无法描述语义比较复杂的用户感兴趣事件。本文提出的基本复杂事件检测模型如图 3 所示。

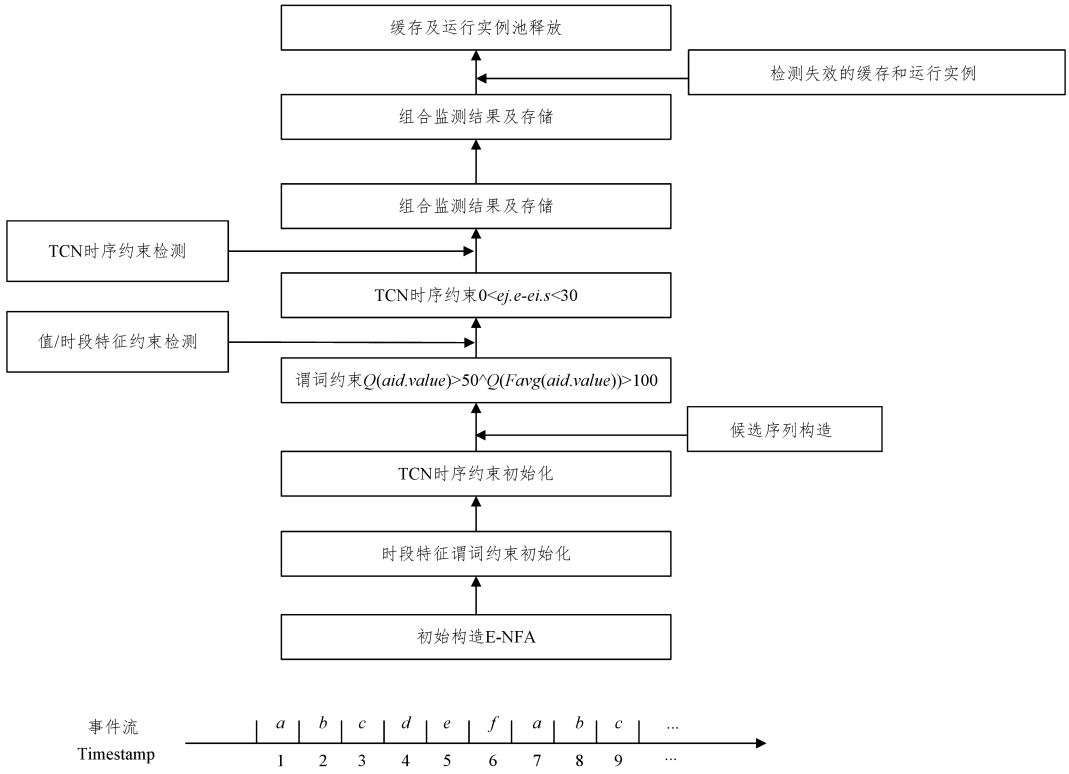

图 3　基本复杂事件检测模型

Fig. 3　Detection model of basic complex event

事件检测过程包含 7 个步骤：E-NFA 构造、谓词约束构造、时序约束 TCN 构造、事件选择策略设定、事件过滤和缓存、检测结果构造、缓存及运行实例池释放。

1）E-NFA 的构造

在复杂事件检测处理中，用户感兴趣的事件采用事件描述语言进行语义描述，即将事件描述结构转换为 E-NFA 模型，转换过程主要通过识别事件描述结构中的子句来确定待检测序列需要包含的事件类型。E-NFA 的构造方法为：

$$E\text{-}NFA::=(S, \Sigma, R, T, \Phi, So, F)$$

其中，S 为有限的状态集合；Σ 为输入事件类型集合；R 为事件之间的谓词约束集合；T 为事件之间的时序约束集合；Φ 为状态转移函数；So 为自动机的非空初始状态集；F 为自动机的终止状态集。假设期望的事件序列子句为 $TCSEQ(E1\ a, E2\ b, E3\ c)$，初始化的 E-NFA 状态集合应该包含 4 个元素，分别为 a，b，c 和 F，并且每个状态需要对应的待匹配事件类型分别为 $E1$，$E2$，$E3$。

2）谓词约束的构造

在事件描述结构中，WHERE 关键字描述了事件包含的谓词约束，包括属性值约束和时段特征约束，由多个布尔表达式字符串组合而成。将多个布尔表达式解析为谓词结构和操作数集合，按照事件类型分别初始化到 E-NFA 不同的待检测状态中。

3）TCN 时序约束的构造

在事件描述结构中，时序约束通过包含多个量化时序关系串的 TCN 子句表示。假定一个期望的 TCN 时序约束为 $TCN\{2 \leqq b.startTime - a.endTime \leqq 5; 1 \leqq d.startTime - b.endTime \leqq 5; 0 \leqq d.startTime - a.endTime \leqq 10\}$，按照分隔符解析得到 3 个时序关系布尔表达式，然后初始化得到一个由时序关系表达式集合构成的 TCN 实例对象，最终将 TCN 实例对象赋值到第一步构造的 E-NFA 实例。

4）事件选择策略的设定

在事件检测过程中，不满足约束的事件是否需要被过滤依赖于具体的事件选择策略。通过关键字 SELECTION_

STRATEGY 子句匹配事件描述规则中设定的具体策略，并对合法性校验，将其赋值到实例事件选择策略的相关属性 E-NFA 上。

5)事件过滤和缓存

随着事件实例进入检测引擎，需要按照过滤条件及事件选择策略判定是否将事件实例添加到事件实例缓存池中。模型 TCSEQ(*E*1 *a*，*E*2 *b*，*E*3 *c*)中，首先根据当前运行实例所处状态判断各事件实例 $a_i \backslash b_i \backslash c_i$ 是否满足谓词约束。若实例池为空且到达了新事件，则判断实例是否满足自动机的初始状态条件 *So*。若满足，则创建一个新的运行实例 *r*1 并添加到运行实例池 runInstancePool 中，同时添加事件实例 $a_i \backslash b_i \backslash c_i$ 到事件实例缓冲池 eventCache 中，否则直接过滤该事件实例。若实例池不为空，需要增加目标事件类型与当前事件实例进行匹配的环节。

6)检测结果的构造与存储

检测结果构造的触发是运行实例池 runInstancePool 中存储到达终止状态的实例 r_i。r_i 记录着满足约束的事件实例集合 events，以事件 ID 的方式表示，而事件实例缓存池采用散列表实现，以键值对〈*eventID*，*eventInstance*〉的形式缓存所有满足约束的事件实例对象。当满足时，将事件序列构造为一个新的高层次事件 e_i，并且 $e_i.startTime = \min\{events[i].startTime \mid i \in \{0,1,\cdots,events.size-1\}\}$，$e_i.endTime = \max\{events[i].endTime \mid i \in \{0,1,\cdots,events.size-1\}\}$。最终将新事件 e_i 存入事件实例数据库中。高层次事件组合并存储的数据库主要应用于聚合事件的检测，其属性值通过时序特征函数计算获得。

7)运行与缓存实例池释放

运行实例池的释放主要有两种情况：1)当运行实例 r_i 到达终止状态时，由于已经完成检测结果的组合与存储等操作，无意义地长期占用实例池会影响引擎的执行效率，应清理运行实例 r_i；2)释放不可能到达终止状态的僵尸实例，用定时任务的方式执行垃圾回收空间释放操作。与运行实例池的清理机制类似，采用定时任务的方式执行缓存实例池的释放清理操作。

3.2 并行检测策略

随着复杂事件数量的增多，基本复杂事件检测模型的处理效率降低，事件匹配过程消耗大部分 CPU 和较长的时间，可见单个引擎的处理效率直接受到输入事件实例数量的限制。因此，本文引入基于事件流划分的并行优化策略，将输入事件流划分为多个片段，分发到不同处理节点，各节点独立承担事件检测任务，从而有效提升事件检测性能。

传统的事件流划分策略需要同时满足谓词约束、时序约束、滑动时间窗口约束，且不同时间窗口内的事件实例不存在关联，即不会被匹配到同一个结果序列集合。通过数据切片选择分割点可以保证检测过程不会出现数据遗漏或者重复检测。基于并行检测策略的常见事件流切割与本文事件流切割过程如图 4 所示。

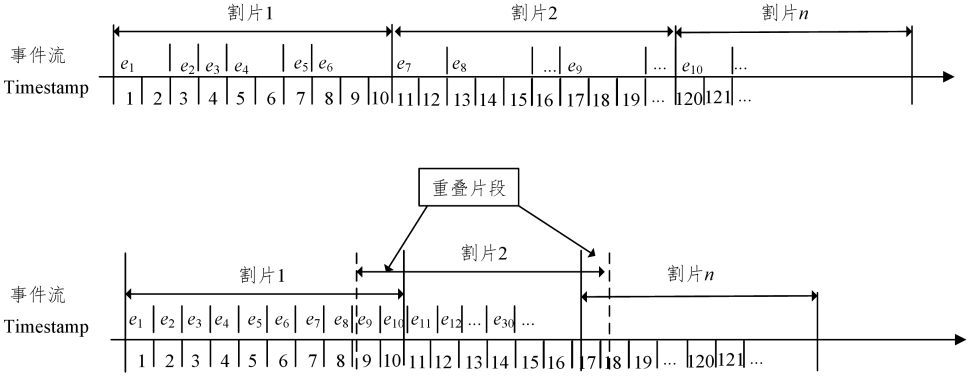

图 4 常见事件流切割与本文事件流切割

Fig. 4 Common event flow split strategy and reasonable event flow split strategy

由图 4 可以看出，常见事件流切割直接采用等距相连切割，e_1 和 e_7 之间的时间跨度已经超过了一个时间窗口，因此不能被匹配到同一个结果组合中。尽管可以得到 *N* 个事件流割片，各割片中包含多个滑动时间窗口，理论上实现了事件流切割策略，但仔细分析会发现其存在数据漏检的问题。比如事件 e_6 与事件 e_7 发生的时间跨度在一个时间窗口之内，并且 e_6 只满足运行实例的一个中间状态，而 e_6 到割片 1 结束的时间间隔又小于一个时间窗口，那么 e_6 本来可以满足某一个匹配序列，但由于时间窗口的约束因素会被错误地过滤掉。

针对如上问题，本文提出采用割片重叠的切分策略，如图 4 所示。该策略可以保证事件流在切分检测过程中不会出现漏检测和重复检测的问题。在事件切割时相连割片之间存在一个长度为 γ 的重叠片段，且长度为一个滑动时间窗口。比如，割片 1 与割片 2 的重叠部分存在事件实例 e_9 和 e_{10}。假设从 e_1-e_8 分别已经有满足匹配的序列被检测到，如 $e_1-e_2-e_5-e_6$，$e_2-e_3-e_6-e_7$，$e_4-e_5-e_7-e_8$。而在整个事件流中，满足匹配的序列还有 $e_5-e_7-e_8-e_9$，$e_{10}-e_{11}-e_{12}$(并且 $e_8-e_9-e_{10}$ 不可能满足序列匹配，因为已经超过了一个滑动时间窗口的约束)。如果采用常见事件流切割方式不考虑重叠，则序列 $e_{10}-e_{11}-e_{12}$ 不会被检测到。而如果采用本文事件流切割方法，虽然在割片 1 中 e_{10} 会被过滤掉，但在割片 2 中可以重新完成匹配，最终得到匹配序列 $e_{10}-e_{11}-e_{12}$；并且割片的长度 γ 可以按照实际情况确定，γ 最小为两个滑动时间窗口的大小，最大没有限制。

3.3 PARALLEL-TCSEQ-DETECTION 检测算法

并行化复杂事件检测模型与基本复杂事件检测模型最大

的不同在于输入事件流的切割与分发。并行化复杂事件检测模型的处理步骤如下：

1）按照集群的计算性能将输入事件流切分为与节点数量一致的 N 块，得到事件流片段 $P_1, P_2, \cdots, P_n$。

2）分发 N 个事件流片段到 N 个节点，各节点分别执行基本的 TCSEQ 检测算法进行事件检测，并且提交计算结果到主节点。

3）统计主节点收到的匹配结果，进行检测结果整合输出或持久化操作。

4　实验结果与分析

4.1　复杂事件描述应用实例

1）聚合事件描述事例

股票市场每日报价数据中，单条记录包含股票编码、时间、开盘价、收盘价、成交量等信息。假设单条记录的收盘价被认为是一个原子事件且类型为 StockQuoto，现有用户感兴趣的聚合事件 P_1 为“检测在上午 10 点开始半小时内股票的平均报价大于 50 元的事件”。则聚合事件 P_1 的形式化描述过程如下：①确定事件类型 $E=$StockQuoto；②构造时段特征约束映射 $F=\{avg\}>50$；③构造时间约束窗口，由 P_1 描述可知，事件流的开始时间为上午 10 点，以每分钟报价且持续半个小时，因此时间窗口 $W=30$；④表示事件流 H，P_1 描述了股票报价事件，事件类型 $E=$StockQuoto，故 $H=\{e_i \mid i\in(1,\cdots,m)\}$。

2）复杂事件描述方法

如上采用股票数据作为事件描述的对象，现假定有复杂事件 P_2 为“检测半个小时内平均股票报价超过 50 元，并且之后一个小时内股票成交量超过 100 000 笔的事件”。P_2 是一个语义比较丰富的复杂事件，其构成包含聚合事件和原子事件。设定股票平均报价事件类型为 Estock_avg，股票成交量事件类型为 Evolume，则复杂事件 P_2 的形式化描述过程如下。

①构造 TCN 量化时序约束

假定股票平均报价事件实例表示为 *estock_avg*，股票成交量事件实例表示为 *evolume*，同样以股票市场开盘时间为基准 0 点，股票报价和成交量一分钟记录一次，因此 TCN 描述关系可以表示为：

$$e_{stock\_avg.e}-e_{stock\_avg.s}=30$$

$$0<e_{volumn_{i.s}}-e_{stock\_avg.e}\leqslant 60$$

$$0<e_{volumn_{j.e}}-e_{volumn_{i.s}}\leqslant 60$$

$$30<e_{volumn_{i.s}}\leqslant 90$$

$$30<e_{volumn_{j.e}}\leqslant 90$$

②构造时间约束窗口为 $W=90$；

③构造事件类型表示（Estock_avg，Evolume）；

④构造事件描述对应的对象类别，$O=$stock；

⑤对象属性值约束，$Q(estock\_avg.value)>50$，$Q(evolume.value)>100\,000$。

4.2　实验环境及数据集

实验中所选用的真实复杂事件数据集综合平衡了规模和种类等因素。大部分复杂事件的规模大于或接近实验单机内存，规模太小的复杂事件由于结果性能的差异不大而不适用于本实验。采用来自金融机构网络的公开实验数据集，表 1 为本文所使用的实验数据集，其为 2007 年 1 月 1 日－2008 年 12 月 31 日期间 2045 支股票每分钟的股票数据，实验数据集共包含 2 亿条记录。所有算法在 32 核 Intel(R) Xeon(R) CPU E5-2670 0 @ 2.60 GHz，32 GB 内存，操作系统为 Centos 6.5 的单机上编译测试。

表 1　实验数据集

Table 1　Experimental datasets

Stock_id	Time_y_m_d	Time_h_m	Start_price	High_price	Low_price	End_price	Volume	Turn_volume	ID
SH1A0003	2007-01-04	10:04	131.987	132.111	131.978	132.052	5348	2317696	351630
SH1A0003	2007-01-04	10:05	132.075	132.075	132.048	132.056	6880	2916064	351631
SH1A0002	2007-01-04	10:05	2914.094	2914.292	2912.305	2912.305	502176	354594816	293551
SH1A0001	2007-01-04	10:05	2769.156	2769.343	2767.464	2767.464	583550	368816128	117395
SH1A0002	2007-01-04	10:06	2766.337	2767.967	2765.544	2766.923	658206	409270272	117396
SH1A0003	2007-01-04	10:06	2911.113	2912.841	2910.273	2911.738	576672	394981376	293552

4.3　实验结果分析

1）检测模型处理能力对比分析

本实验主要对比分析 TCSEQ 检测模型与 SASE 复杂事件检测引擎在不同事件描述下的处理能力。由此，刻画了 5 种不同语义复杂度的事件描述结构，来验证本文检测方法与传统方法的处理能力的差别。采用 2008 年 1 月 1 日到 12 月 31 日期间的股票报价数据进行验证，实验中不同语义事件描述如下：

①Q1：在 100 000 s（即 27.8 h）内满足模式 a（报价为 500 的倍数）、b（b 的报价大于 a）、c（报价小于 2500 元）的 3 个股票报价事件。

②Q2：在 100 000 s（即 27.8 h）单支股票的连续报价出现波动，且满足平均每手报价大于或等于 5000 元同时方差大于 20 的事件序列。

③Q3：在 100 000 s（即 27.8 h）内，满足模式 a（报价为 500 的倍数）、b（b 的报价大于 a）、c（在 a，b 后出现事件 c 的交易量值大于 100 000 支的事件序列）的 3 个事件。

④Q4：在 100 000 s（即 27.8 h）内，满足模式 a（报价为 500 的倍数）、b（a 之后的 1 000 s 内 b 的报价大于 a）、c（b 之后出现 c 的报价小于 2500 元）的 3 个股票报价事件。

⑤Q5：在 100 000 s（即 27.8 h）内，满足模式 a（报价为 500 的倍数）、b（a 之后的 1 000 s～1 500 s 内 b 的交易量小于 100 000 支）、c（满足 b，在 1200 s～1500 s 内出现 c 的报价大于 a）的 3 个股票报价事件。

检测模型处理能力的结果如表 2 所列。

表 2 检测模型处理能力的对比结果

Table 2 Comparison results of processing capacity of detection modelsmodel

事件描述规则	输入事件流类型	谓词约束		滑动时间窗口	TCSEQ 检测模型	SASE 检测模型
		属性值比较操作	量化时序约束操作			
Q1	单一	简单	无	无	支持	支持
Q2	单一	无	简单	无	支持	不支持
Q3	多个	简单	简单	一个	支持	支持
Q4	单个	复杂	简单	多个	支持	不支持
Q5	多元	复杂	复杂	多个	支持	不支持

通过表 2 的 5 组对比实验结果可以看出，Q2 事件描述中包含一个均值操作和一个求方差操作，SASE 检测模型对此无能为力。SASE 只能够支持 Q1，Q3 语义描述的情况，但 TCSEQ 能够支持 Q1－Q5 的各种语义描述的情况。实验结果验证了本文提出的有复杂谓词属性约束能力且能够描述丰富时序语义的复杂事件处理描述语言的有效性。

2)输入三因素对检测性能的影响

随着事件数据量的增多和序列长度的变化，自动机跳跃状态随之增多，导致状态机结构更加复杂，整个事件检测过程中迭代次数增多，从而理论上对检测算法的性能产生了一定的影响。同时，随着滑动时间窗口的变化，事件匹配率也随之增大，处理性能也会发生一定的变化。因此，本实验主要验证输入三因素(事件数据量、待匹配序列长度、滑动时间窗口)对复杂事件检测效率是否会产生影响。本实验主要验证检测系统的吞吐量(单位为 events/second)。实验结果如图 5－图 7 所示。

①因素 1：事件数据量。保持事件的谓词约束和时序约束固定不变，在事件输入流数据量为 30 000～120 000 条实例的情况下进行 7 组对比实验。

②因素 2：待匹配序列长度。在相同的数据集上固定时间窗口(100 000 s)和输入事件流的个数(59 278)，在滑动的待匹配序列长度为 2～16 的情况下进行 8 组对比实验。

③因素 3：滑动时间窗口。保持输入事件流个数(59 278)和事件约束条件不变，在滑动时间窗口大小为 100 000 s～1 000 000 s 的情况下进行 10 组对比实验。

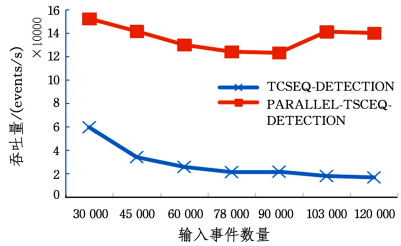

图 5 输入事件数量对事件检测效率的影响

Fig. 5 Effect of number of input events on event detection efficiency

从图 5 中可以看出，随着输入事件数量的增多与滑动时间窗口的增大，满足匹配的事件序列数量增多，TCSEQ-DETECTION 检测算法出现吞吐量下降的趋势，但下降幅度不大，整体保持在低吞吐量范围。而并行化的检测算法 PARALLEL-TCSEQ-DETECTION 有很明显的吞吐量提升，最大处的系统吞吐量可达到基本检测算法的 10 倍之多，且整体呈现较均衡的趋势，中间出现波动点的原因是系统调度出现一些不稳定的情况。

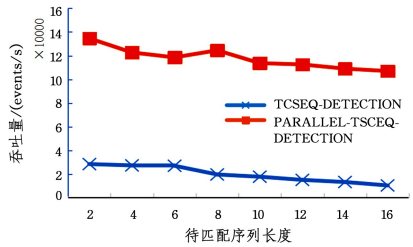

图 6 待匹配序列长度对吞吐量的影响

Fig. 6 Effect of length of matched sequence on throughput

从图 6 中可以看出，在输入事件数量不变的情况下，随着待匹配序列长度的增加，自动机完成匹配需要跳跃的状态增多，缓存数据随之增多，完全满足匹配模式的事件实例相应减少。TCSEQ-DETECTION 检测算法的吞吐量呈现下降的趋势，而并行检测算法 PARALLE-TCSEQ-DETECTION 的吞吐量虽然有一定的下降趋势，但整体处于较稳定的状态，且并行算法的吞吐量明显高于基本检测算法。可见，随着待匹配序列长度的增大，基本检测算法会出现性能瓶颈，而并行算法可以有效地提升检测引擎的效率。

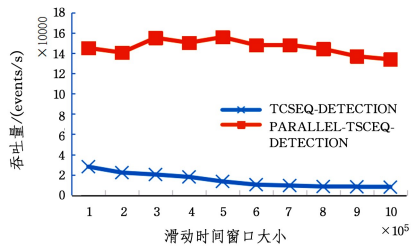

图 7 滑动时间窗口大小对性能的影响

Fig. 7 Effect of sliding time window size on performance

从图 7 中可以看出，随着滑动时间窗口的变化，候选匹配事件序列的范围增大，导致满足匹配的事件序列的数量增多。TCSEQ-DETECTION 的检测效率出现下降的趋势，而并行算法 PARALLEL-TCSEQ-DETECTION 虽然有一定的性能下降，但整体效率更优；并且，随着滑动时间窗口的进一步增大，并行算法的性能损失明显小于基本算法，这主要是由于并行算法对输入事件流进行了分片处理，虽然每个分片节点的滑动时间窗口变大，但输入事件流较小，单节点计算量不会出现猛增，进而使整体检测效率比基本检测算法更优。

结束语 随着信息化的发展，多领域业务系统数据呈现爆炸式增长。从大量分布式业务系统中寻找有价值的信息，对多类别数据进行融合，让数据为人类社会生产发展服务变得尤为紧迫[26]。为解决此问题，本文研究了复杂事件处理技术在股票金融市场数据融合中的应用，通过源数据抽象的多

元事件流之间的量化时序约束和复杂谓词约束关系，引入复杂事件检测引擎，检测出有价值的高层次事件。

通过对实验结果的分析可知，本文复杂事件处理技术可以有效支持多元事件流输入、量化时序约束和复杂的谓词约束处理，并且检测算法具有较高的效率，并行算法的效率一度可以达到基本算法的 10 倍之多。在描述方法中引入 TCN 时间约束网络进行事件间量化时序关系的表示，引入时段特征进行事件复杂谓词约束的表示，能有效解决传统事件描述方法不足的问题。提出的聚合事件描述模型和复杂事件描述模型有效增强了事件描述能力。在检测模型中，基本复杂事件检测算法 TCSEQ-DETECTION 定义了扩展的有限状态自动机 E-NFA，增强了对量化时序约束和复杂谓词约束的处理，同时支持多元事件流输入。针对数据量较大、处理周期较长的情况，本文提出了基于事件流切片的并行复杂事件检测算法 PARALLEL-TCSEQ-DETECTION，有效提升了复杂事件的检测性能。但同时，复杂事件处理技术还有待进一步完善，未来可以考虑在定义事件描述规则库时，引入机器学习、深度学习、人工智能等技术从大量事件数据流中自动学习并构造数据中潜在的复杂事件描述结构，从而减轻领域专家的工作负担。

参 考 文 献

[1] CUGOLA G, MARGARA A. Processing flows of information: From data stream to complex event processing [J]. ACM Computing Surveys(CSUR), 2012, 44(3): 1-62.

[2] ETZION O, NIBLETT P, LUCKHAM D C. Event processing in action[M]. Greenwich: Manning, 2011.

[3] WANG Y H, CAO K, ZHANG X M. Complex event processing over distributed probabilistic event streams[J]. Computers & Mathematics with Applications, 2013, 66(10): 1808-1821.

[4] DAYARATHNA M, PERERA S. Recent advancements in event processing[J]. ACM Computing Surveys(CSUR), 2018, 51(2): 33-69.

[5] XIAO F, ZHAN C, LAI H, et al. New parallel processing strategies in complex event processing systems with data streams[J]. International Journal of Distributed Sensor Networks, 2017, 13(8): 1-15.

[6] Kam P, Fu A W C. Discovering temporal patterns for interval-based events[C]// International Conference on Data Warehousing and Knowledge discovery. Springer Berlin Heidelberg, 2000: 317-326.

[7] ALLEN J F. Maintaining knowledge about temporal intervals [J]. Communications of the ACM, 1983, 26(11): 832-843.

[8] CHANDRASEKARAN S, COOPER O, DESHPANDE A, et al. TelegraphCQ: continuous dataflow processing[C]// Proceedings of the 2003 ACM SIGMOD international conference on Management of data. ACM, 2003: 668-668.

[9] ARASU A, BABU S, WIDOM J. The CQL continuous query language: semantic foundations and query execution [J]. The VLDB Journal—The International Journal on Very Large Data Bases, 2006, 15(2): 121-142.

[10] PATEL D, HSU W, LEE M L. Mining relationships among interval-based events for classification[C]// Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data. ACM, 2008: 393-404.

[11] WU S Y, CHEN Y L. Mining nonambiguous temporal patterns for interval-based events[J]. IEEE Transactions on Knowledge and Data Engineering, 2007, 19(6): 742-758.

[12] BRENNA L, DEMERS A, GEHRKE J, et al. Cayuga: a high-performance event processing engine[C]// Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data. ACM, 2007: 1100-1102.

[13] DEMERS A J, GEHRKE J, PANDA B, et al. Cayuga: A General Purpose Event Monitoring System[C]// CIDR. 2007: 412-422.

[14] GYLLSTROM D, WU E, CHAE H J, et al. SASE: Complex Event Processing over Streams [J/OL]. arXiv preprint cs/0612128, 2006.

[15] DIAO Y, IMMERMAN N, GYLLSTROM D. Sase+: An agile language for kleene closure over event streams[R]. Amherst, UMass Technical Report, 2007.

[16] CUGOLA G, MARGARA A, MATTEUCCI M, et al. Introducing uncertainty in complex event processing: model, implementation, and validation[J]. Computing, 2015, 97(2): 103-144.

[17] WANG F, LIU S, LIU P, et al. Bridging physical and virtual worlds: complex event processing for RFID data streams[C]// International Conference on Extending Database Technology. Springer Berlin Heidelberg, 2006: 588-607.

[18] CHEN Q, LI Z, LIU H. Optimizing complex event processing over RFID data streams[C]// IEEE 24th International Conference on Data Engineering. IEEE, 2008: 1442-1444.

[19] WU E, DIAO Y, RIZVI S. High-performance complex event processing over streams[C]// Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data. ACM, 2006: 407-418.

[20] ALVES D C, ALEXANDRE, et al. Embedded event processing: U. S. Patent 9,712,645[P]. 2017-7-18.

[21] AGRAWAL J, DIAO Y, GYLLSTROM D, et al. Efficient pattern matching over event streams[C]// Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data. ACM, 2008: 147-160.

[22] https://en. wikipedia. org/wiki/Esper.

[23] VILAIN M B, KAUTZ H A. Constraint Propagation Algorithms for Temporal Reasoning[C]// AAAI. 1986: 377-382.

[24] NEBEL B, BÜRCKERT H J. Reasoning about temporal relations: a maximal tractable subclass of Allen's interval algebra [J]. Journal of the ACM(JACM), 1995, 42(1): 43-66.

[25] LEE O J, JUNG J E. Sequence clustering-based automated rule generation for adaptive complex event processing [J]. Future Generation Computer Systems, 2017, 66(9): 100-109.

[26] LI Z G, ZHONG J. Summary of the Application of Data Science in Domestic Management Studies [J]. Computer Science, 2018, 45(9): 38-45. (in Chinese)
李志国，钟将. 数据科学在国内管理学研究中的应用综述[J]. 计算机科学，2018，45(9)：38-45.