计算机研究与发展
Journal of Computer Research and Development
ISSN 1000-1239/CN 11-1777/TP
50(6): 1163-1176, 2013

面向物联网的分布式上下文敏感复杂事件处理方法

曹科宁[1,2] 王永恒[1] 李仁发[1] 王凤娟[1]
[1](湖南大学信息科学与工程学院 长沙 410082)
[2](湖南省质量技术监督局 长沙 410111)
(9462478@qq.com)

A Distributed Context-Aware Complex Event Processing Method for Internet of Things

Cao Kening[1,2], Wang Yongheng[1], Li Renfa[1], and Wang Fengjuan[1]
[1](*College of Information Science and Engineering, Hunan University, Changsha* 410082)
[2](*Administration of Quality and Technology Supervision of Hunan Province, Changsha* 410111)

Abstract The data produced by Internet of things has the characteristic of big-data and those data can hardly be processed by present data processing technologies. As the key part of Internet of things, complex event process (CEP) has two characteristics of big data: quantity, complexity and has the need of on-time processing. Context-awareness is an important feature of CEP engine. In this paper, a high-performance distributed context-aware CEP architecture and method is proposed for Internet of things. The method uses fuzzy ontology to create query model to support event query of uncertainty and fuzzy. Based on fuzzy ontology query and similarity based distributed reasoning, complex event query plans are generated and context-aware queries are rewritten into context independent sub-queries. The sub-queries are optimized and executed parallel based on data partition. The experimental results show that this method can support fuzzy context in CEP and has better performance and scalability than other methods.

Key words Internet of things; big data; complex event processing; distribute; fuzzy ontology; query rewriting

摘 要 物联网产生的数据具有大数据特征,而这些数据难以用现有数据处理技术进行有效处理.作为物联网中间件的核心技术,复杂事件处理技术具备大数据的海量、复杂性等特征和实时处理的需求.上下文敏感是复杂事件处理引擎的重要特征.提出一种高效的面向物联网的分布式上下文敏感复杂事件处理架构和方法.该方法使用模糊本体进行上下文建模,以支持事件的不确定性及模糊事件查询问题.以基于模糊本体的查询和基于相似性的分布式推理为基础,生成复杂事件查询规划,并通过查询重写,把上下文相关查询转换为上下文无关子查询.根据不同的事件模型和上下文划分数据,并通过优化和多级并行来提高性能.实验结果表明该方法能够处理模糊事件上下文,对于面向物联网的分布式上下文敏感复杂事件处理具有比一般方法更好的性能和可伸缩性.

关键词 物联网;大数据;复杂事件处理;分布式;模糊本体;查询重写

中图法分类号 TP391

随着信息技术的发展,数据的累积飞速增长,人类正在进入"大数据"的时代.随着2012年3月美国政府发起《大数据研究和发展倡议》并以此形成新的国家战略,其他国家也纷纷提出自己的大数据战略.

收稿日期:2012-12-25;**修回日期**:2013-04-11
基金项目:国家自然科学基金项目(61173036);国家"八六三"高技术研究发展计划基金项目(2012AA01A301-01);湖南省自然科学基金项目(13JJ3046);长沙市科技计划基金项目(K1203026-11)

大数据具有以下特点:1)数据量巨大;2)数据类型繁多,如文字、视频、图片、音频、时间信息、地理位置信息等;3)价值密度低.以视频为例,在连续不断的监控过程中,可能有用的数据仅仅有一两秒;4)实时性要求高,要求处理速度快.

物联网(Internet of things, IoT)是大数据的重要来源之一.在交通、工业、零售等领域,已经建立起了超过3千万传感器构成的传感器网络,并且其数量以每年超过30%的速度增长[1].这些传感器不停地获取数据,其数据积累的速率远远超过以前的常规应用.

物联网中间件在处理物联网事件中具有关键作用.物联网中设备的一次读取产生的数据大多含有时间戳、空间戳和设备戳,我们称之为原始事件.对于大型的物联网系统,其事件是多种多样的,包括RFID或传感器产生的数据流、GPS信号、天气预报消息等.我们把这些种类繁多的事件合在一起称为事件云.这些由设备直接产生的原始事件中所包含的语义信息是非常有限的,仅能从中获取一些简单的信息.而在实际的物联网应用中,我们更关心的是具有业务逻辑和规则的复杂信息,只有像"车辆离开车库"、"室内温度上升了1℃"这样的信号,对用户才是有意义的.为了获得这些有意义的事件,需要基于某种规则将原始事件处理成复杂事件,我们称这种对原始事件进行处理后得到的对用户有意义的上层事件为复杂事件(complex event),这个处理过程为复杂事件处理(complex event processing, CEP)[2].

复杂事件处理在主动数据库中已经做了大量的研究,常用的方法都是基于固定的数据结构,如树、图、自动机或Petri网.这些方法很难灵活地对事件查询语言的实现进行优化,也很难根据应用需求的变化来支持查询语言的扩展.针对这些问题,Diao等人提出了一种基于查询规划的复合事件检测方法SASE[3].

和普适计算一样,上下文对复杂事件处理具有重要的意义.事件发生的上下文不同,可能采取的处理策略也不同.在上下文敏感计算和普适计算中,已经对上下文的表示和推理进行了大量研究.但是目前关于上下文敏感复杂事件处理的工作很少.有些文章提出了一些想法和体系结构,但缺乏实现的细节.

RFID阅读器和传感器数据流产生原始事件.由于传感器噪声等原因,物联网事件具有不确定性,同样,上下文也具有不确定性.因此,复杂事件处理引擎在处理事件和上下文的不确定具有重要作用.

当前对于大数据的研究主要集中在历史数据的分析和处理,特别是数据仓库和数据挖掘领域.主要表现为数据规模、数据类型、模式和数据的关系、处理对象、处理工具5个方面[4].常见的方法是采用大量服务器构成的集群或专业的云计算平台(如亚马逊的AWS),基于Hadoop这样的分布式计算框架,把传统的数据分析算法改造成MapReduce的算法.但这样的方法并不适合于物联网的数据处理.因为Hadoop/MapReduce的框架适用于处理静态数据和历史数据,而物联网产生的是实时的数据流,并且往往要求实时的响应.在带有控制的物联网系统中数据还会经常变化.这些特点导致物联网中大数据的处理无法直接采用Hadoop/MapReduce的框架.

针对上述问题,本文提出一种面向物联网的分布式上下文敏感复杂事件处理方法(distributed context-aware complex event processing, DCCEP).该方法扩展了查询规划,采用模糊本体表示事件上下文,并提出分布式上下文推理.通过查询重写的方法,把上下文敏感的复杂事件查询重写为上下文无关子查询.针对大数据的处理,DCCEP采取了分布式的体系结构,并通过多级的并行来提高处理性能.

1 相关研究

1.1 复杂事件处理

复杂事件处理即检测不断产生的数据流,根据每个事件发生的集合/序列识别复杂事件,然后对检测到的状态作出回应.

复杂事件处理主要有4步:1)从大量数据中获取原始事件;2)根据具体规则检测关联性事件或聚合事件,用事件操作符创建有意义事件;3)处理原始事件或复杂事件,获取这些事件的时间、因果关系、层次关系及其他语义关系;4)为了确保事件信息发送到用户,向应用层作出回应.

在主动数据库及RFID的研究中,复杂事件处理大都采用固定的数据结构.Li等人采用了基于树结构的复杂事件处理方法,并通过事件分组的方法对算法进行优化[5].Wang等人使用有向图在RFID事件流中进行复杂事件处理[6].Jin等人采用定时Petri网(timed Petri-net, TPN)在RFID数据流中检测复杂事件[7].

SASE是一种基于查询规划的高性能复杂事件处理方法,SEQ事件检测作为SASE的主要部分,

采用有限状态机(NFA)和活动实例栈(AIS)[3]. 近年来有一些针对 SASE 的改进算法,Agrawal 等人提出了改进的有限状态机机(NFA^b)模型来提供更丰富的查询能力,如使用 skip_till_next_match 和 skip_till_any_match 等查询指令[8],Zhang 等人对 SASE 的模型进行了改进来支持非确定时标[9].

传统的基于集中式的复杂事件处理结构当客户增多时,这种体系结构需要更多的带宽和更强的计算能力,同时由于网络拥塞和单点失效等问题,这种结构也很难达到良好的稳定性和可伸缩性,而当前很多物联网应用本身就是分布式的,必须采取分布式的事件处理方法. Akdere 等人提出了面向分布式资源的 CEP 方法[10],Ku 等人提出了一种面向 RFID 应用的分布式 CEP 方法[11].

1.2 上下文敏感事件处理

上下文模型在上下文系统的应用和开发中具有重要作用,在上下文的相关研究中文献[12]提出了各种上下文表示模型,包括键-值模型、面向对象的模型和基于本体的模型. 一般认为本体是事件上下文表示的最佳模型,但传统的本体无法处理不确定的知识这限制了其在不确定事件处理中的应用,因此近年出现了一些模糊本体模型和推理方面的研究:Almeida 等人提出了一种面向智能环境的基于模糊逻辑的模糊模型[13];Singh 等人提出了一种模糊集成本体模型(FOIM),试图把模糊逻辑集成到本体设计结构中[14];Zhang 等人提出了面向分布式模糊推理分解的分布式模糊推理 Petri 网模型[15];Cai 等人研究了模糊本体中的隶属性和典型性问题,并提出了一种面向推荐系统的新型模糊本体模型[16].

上下文感知在异构环境中具有重要作用. 上下文敏感系统面临的挑战主要是实时地对于用户上下文作出合适的决策. 用智能的方式处理上下文数据称为上下文推理. Lee 等人提出了基于相似性的上下文推理,定义了上下文模型之间的相似性[17]. 该方法不需要初始的上下文信息,从而减少了推理过程的复杂性. 除此之外,该方法使用基于相似性分布式推理的 Mahalanobis 间隔加入了上下文模型的关联,寻找一种最佳的上下文模型,并且减少两个上下文模型之间的模糊来增加推理的可信度.

近年来也出现了一些上下文敏感事件处理的研究工作:Helmer 等人描述了基于上下文的事件处理框架,并总结了目前支持常用的事件处理系统的上下文[18];Ashish 等人提出了一种基于本体的上下文敏感复杂事件处理方法[19];Zhou 等人提出了一种非精确语义复杂事件处理框架,在复杂事件处理引擎中加入语义知识,实现非确定复杂事件模式检测[20],在 Teymourian 的研究中,加入了本体和声明性规则,优化了智能事件处理机[21]. 但目前的文章大多讨论上下文敏感复杂事件处理的思路和框架,而缺少把上下文引入复杂事件处理算法的细节.

1.3 查询重写

查询重写在数据库中已经进行了广泛的研究和应用. 近年来人们开始研究通过查询重写来进行本体推理和事件处理:Rabinovich 等人提出了一种模式重写的形式模型,并讨论了如何在复杂模式检测中使用该模型[22];Schultz-Møller 等人提出了一种基于查询重写的分布式复杂事件处理方法[23]. 在上述研究中,在事件模型被转换成事件自动机之前,使用高层查询语言表示,并用更有效的方式进行重写. 为实现检测的可扩展性,自动机通过集群器进行分发. Reukert 等人提出基于过滤的重写规则,将谓词下推在模式匹配过程的查询优化中[24]. 并提出了重写匹配过程的模型工具和推荐系统;Lembo 等人介绍了矛盾 DL-Lite 本体的查询重写方法[25]. 但基于查询重写的上下文敏感复杂事件处理方面目前研究工作很少.

2 事件及上下文模型

2.1 事件模型

具有相同特征的一类事件为一个事件类型. 在本文的事件模型中,事件处理系统的输入是由原始事件组成的数据流. 事件类型用大写字母表示,为了简化,原始事件用小写字母和数字表示,如 a_1,其中数字表示时间戳.

复杂事件用二元组来表示($Element$,$Rule$),其中 $Element$ 是复杂事件的组成元素,$Element$={原子事件,复杂事件};$Rule$ 为运算规则,本文的复杂事件模型主要包括 ALL, ANY, SEQ, COUNT 等[26]. 这些模型组合后形成层次复杂模型.

例 1. 复杂事件模型. 事件 E_1:时间 T_1 经过路口 I_1 的车辆超过 30. 事件 E_2:时间窗口为 T_2 时,汽车 A 按序经过路况 I_2,I_3,I_4. 那么 E_1 是 COUNT 模型,E_2 是 SEQ 模型.

定义 1. 概率原始事件. 物联网事件云中的原始事件即是在某个时间的一次发生. 概率原始事件表示为(A,T,Pr),其中 A 为事件属性的集合,T 为事件

发生的时标，Pr 为事件发生的概率值，它代表由原始信号转换为确切的物联网事件的概率.

定义 2. 概率复杂事件. 概率复杂事件是由原始事件或者复杂事件按照一定的运算规则形成的事件. 一个概率复杂事件表示为 (E, R, Ts, Pr)，其中 E 是复杂事件的组成元素，R 为事件合成规则，Ts 为事件的时间跨度，Pr 为事件概率值.

2.2 上下文模型

在事件处理领域，上下文可以定义为一些特殊的条件，根据这些条件可以对事件实例进行划分，从而使被划分的事件能够关联在一起来处理. 上下文可能有一个或几个上下文信息，可以划分一个或多个上下文分区[23]. 表 1 列出了本文处理的具体上下文类型，这些上下文经过组合后形成复合上下文.

Table 1 Context Types

表 1 上下文类型

Pattern Name	Explanation
Event interval	A window is opened or closed when an event processing agent receives a particular event as its input.
Slidding fixed interval	Each window is an interval with a fixed temporal size or a fixed number of events.
Slidding event interval	The opening of each new window, and its duration, is determined by counting the number of events received by the event processing agent.
Fixed location	A fixed location context has predefined context partitions based on specific spatial entities.
Entity distance	Assign events to context partitions based on their distance from a given entity.
Event distance	Assign events to context partitions if they occurred within a specific distance from the location of the event that triggered the creation of the partition.
State-oriented	A single partition based on the state of the external entity.
Segmentation oriented	Assigns events to context partitions based on the values of one or more event attributes.

例 2. 事件上下文类型. 上下文 C_1：汽车旅馆 M_1 周围 2 km 范围. 上下文 C_2：交通事故 A_1 周围 10 km 范围. 上下文 C_3：公路交通状态是缓慢(某段公路的交通状态有交通流畅、交通缓慢、交通停止). C_1 是实体距离上下文，C_2 是事件距离上下文，C_3 是面向状态的上下文.

3 分布式上下文复杂事件处理

3.1 系统体系结构

上下文复杂事件处理系统框架如图 1 所示. 复杂事件处理引擎主要有查询规划器、上下文管理和

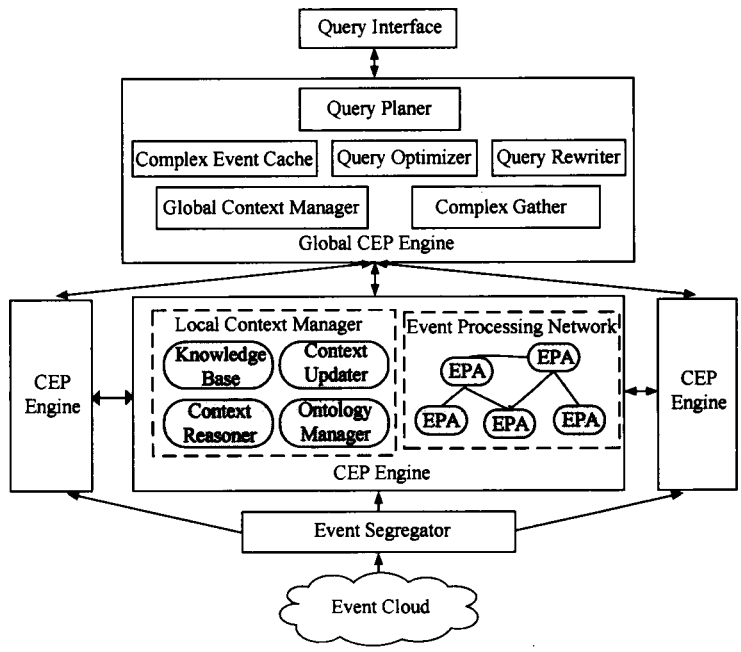


Fig. 1 System architecture.

图 1 系统结构

事件处理网络构成.

1) 上下文管理器

上下文管理器用于管理事件上下文.上下文管理器中的本体管理器有3种类型的接口:①图形界面接口,由本体工程师在编辑本体时使用;②API接口,当上下文变化时调用这个接口进行本体的更新;③查询接口,由事件处理引擎调用,用于获取上下文信息.上下文更新器收集事件属性并进行分析,然后调用本体管理器的API接口来更新上下文.上下文推理器用于发掘新的知识或者解决现有知识中的冲突.另外有一些背景知识放在特定的知识基而不是本体当中.

2) 查询管理器

查询规划器是查询管理器的中枢,因为本文复杂事件处理引擎是基于查询规划.查询规划器对用户的查询请求进行分析并生成相应的查询规划.上下文敏感查询由查询重写器根据上下文信息重写为上下文无关子查询,然后再由事件处理网络执行这些上下文无关子查询.查询优化器对基于查询规划的事件类型、上下文类型和事件划分进行优化.

3) 事件处理网络

实时CEP引擎负责对物联网事件云中的基础事件进行实时处理.实时CEP引擎的核心是由事件处理Agent组成的网络(EPA网络),其中EPA网络是由各种不同类型的EPA组成的,如基于过滤器代理、模型检测代理和变换代理等[16].本文主要研究模型检测代理.不同类型的EPA可能使用不同类型的数据结构.在本文研究工作中,对于序列复杂事件的处理采用基于有限状态机的EPA;对于层次复杂事件的处理采用基于匹配树的EPA.平行和分布式处理方法支持分布式应用并提高了性能.本文也对面向分布式概率事件流的复杂事件处理作了相关研究,关于EPA的相关详细信息可参看本文同一作者的其他研究成果.

3.2 基于上下文表示的模糊本体

本文的上下文表示是基于模糊本体框架[16],并针对事件处理进行了优化.

定义3. 模糊本体.在特定领域Δ上的模糊本体O表示为$O\Delta=(C,R,P,I,A)$,其中C为模糊概念的集合,P为概念的模糊属性的集合,I是概念实例的集合,R是模糊角色(对象及概念间的关联)的集合,A为公理的集合.

定义4. 模糊概念.模糊概念C定义为$C=\{a_1^{v_1},a_2^{v_2},\cdots,a_n^{v_n}\}$,$a_i$表示对象,$v_i$是对象$i$在$C$中的成员度,模糊概念中对象成员度的隶属函数为$\mu_C:A\to[0,1]$,$A$为对象的集合.

定义5. 概念包含.对于模糊概念$X=\{a_1^{w_1},a_2^{w_2},\cdots,a_n^{w_n}\}$和$Y=\{a_1^{v_1},a_2^{v_2},\cdots,a_n^{v_n}\}$,$a_i$为对象,$w_i$和$v_i$为对象的成员度.如果存在$a_i^{w_i}\in X$,$a_i^{v_i}\in Y$,$v_i\geqslant w_i$,那么$X$被包含于$Y$,定义为$X\subseteq Y$.

定义6. 模糊角色.模糊角色R是域内两个对象之间的二元模糊集,它是成对对象的一个集合,定义为$R=\{(a_1,b_1)^{w_1},(a_2,b_2)^{w_2},\cdots,(a_n,b_n)^{w_n}\}$,其中$a_i$和$b_i$是两个对象,$w_i$是关系的度,且$w_i$计算函数为$\mu_R:A\times B\to[0,1]$,$A$和$B$为对象集合,其中$A$为角色域,$B$为角色范围.

定义7. 模糊角色包含.对于模糊角色$S=\{(a_1,b_1)^{w_1},(a_2,b_2)^{w_2},\cdots,(a_n,b_n)^{w_n}\}$和$T=\{(a_1,d_1)^{v_1},(c_2,d_2)^{v_2},\cdots,(c_n,d_n)^{v_n}\}$,如果$\forall(a_i,b_i)^{w_i}\in S,(a_i,b_i)^{v_i}\in T,v_i\geqslant w_i$,那么$S$被包含于$T$,表示为$S\subseteq T$.

定义8. 模糊属性.模糊属性P定义为$P=R.C$,P是模糊角色,C是模糊角色范围内的模糊概念.

定义9. 模糊属性包含.两个模糊属性$P_1=\{((a,c),c)^{v_{1i}}\mid((a,c),c)^{w_{1i}}\in S,c^{y_{1i}}\in C\}$和$P_2=\{((a,c),c)^{v_{2i}}\mid((a,c),c)^{w_{2i}}\in S,c^{y_{2i}}\in D\}$,如果$\forall((a,c),c),((a,c),c)^{v_{1i}}\in P_1,((a,c),c)^{v_{2i}}\in P_2$,$v_{2i}\geqslant v_{1i}$,那么$p_1$被包含于$P_2$,即$P_1\subseteq P_2$.

定义10. 有变量的模糊属性.模糊属性表示为$\langle V,M\rangle$,V是语义参数集合,M是函数成员.

例3. 模糊本体.例如"红色轿车"是一个模糊概念,它是另外一个概念"轿车"的子集."张三非常喜欢跑车"是一个模糊角色的实例,关联程度是"非常高".模糊属性"驾驶速度"的值可以表示为{"S(慢)","M(中)","F(快)"},S,M,F定义如式(1)所示:

$$S=\int_0^{100}\left(\frac{100-x}{100}\right)\Big/x;$$

$$M=\int_0^{100}\left(\frac{x}{100}\right)\Big/x+\int_{100}^{200}\left(\frac{200-x}{100}\right)\Big/x;$$

$$F=\int_{100}^{200}\left(\frac{x-100}{100}\right)\Big/x. \tag{1}$$

根据基于上下文表示的模糊本体,本文提出模糊上下文的概念如下:

定义11. 模糊上下文.模糊上下文FC定义为$FC=\langle N_c,N_o,N_p\rangle$,$N_c$是模糊概念的集合,$N_o$是概念集合,$N_p$是模糊属性集合.

本文建立了一个交通领域模糊本体,其中的一

部分如图 2 所示. 本体的表示基于 Fuzzy OWL 2[①], 本体推理部件基于 FuzzyDL[②] 构建.

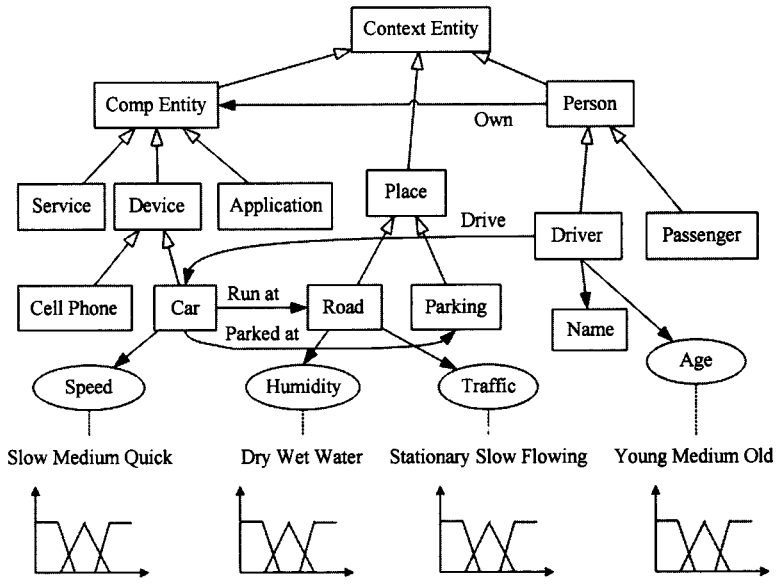


Fig. 2 A simplified traffic domain fuzzy ontology.

图 2 交通领域本体的一部分

3.3 分布式多层上下文推理方法

在复杂的面向物联网的应用中,基于逻辑的推理方法变得较复杂,为了解决大数据问题,本文提出一个基于多层次上下文模型下的基于相似性分布式推理方法,如图 3 所示.

此模型中,第 1 层根据模糊本体实例生成上下文状态,其中模糊本体实例由复杂事件处理引擎生成,由于事件处理代理可能是多个,所以模糊本体实例为分布式的. 第 $i+1$ 层的上下文状态可由下一层的上下文推断出.

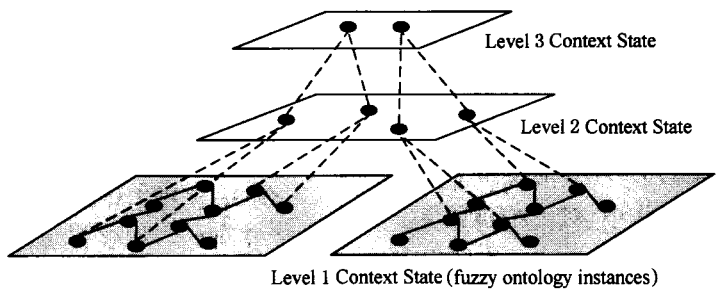


Fig. 3 Multi-level context model.

图 3 多层次上下文模型

1) 高层上下文推理

不确定推理过程从低层上下文推导出高层上下文状态,其中低层上下文基于模糊 D-S 证据理论[27].

定义 12. 预期确定性和概率性. 预期确定性 $EC(B)$ 和概率性 $EII(B)$ 计算公式如下:

$$EC(B)=\sum_i m(A_i)\cdot \inf(A_i\Rightarrow B), \quad (2)$$

$$EII(B)=\sum_i m(A_i)\cdot \sup(A_i\cap B), \quad (3)$$

其中 m 为集合函数,当 A_i 和 B 不为模糊集时,$EC(B)$ 和 $EII(B)$ 为标准的 D-S 理论似然度.

定义 13. 包容性测量. $I(A\subset B)$ 为集合 A 包含于集合 B 中的度. 计算如下:

$$I(A\subset B)=\frac{\min_x\{1,1+\mu_B(x)-\mu_A(x)\}}{\max_x(\mu_A(x))}. \quad (4)$$

信任函数计算如下:

$$Bel(B)=\sum_A I(A\subset B)m(A). \quad (5)$$

定义 14. 扩展 Dempster 组合规则. Dempster

① http://gaia.isti.cnr.it/~straccia/software/FuzzyOWL/

② http://www.straccia.info/software/fuzzyDL/fuzzyDL.html

组合规则扩展如下:

$$m_1 \oplus m_2(C) = \frac{\sum_{A\cap B=C} \max_{x_i} \mu_{A\cap B}(X_i) m_1(A) m_x(B)}{1-\sum_{A,B}(1-\max_{x_i}\mu_{A\cap B}(X_i)) m_1(A) m_x(B)}. \quad (6)$$

基于扩展的 Dempster 组合规则,组合低层的上下文(证据)可以推导出高层的上下文(结论). 冲突因子用于解决 Zadeh 悖论问题,并使用证据选择策略进行优化. 在基本概率分布中,质量分布函数是根据设备的优先级设置的,如传感器和 RFID 读写器. 当证据数据很大时通过质量函数对这些证据排序,为提高推理性能将忽略证据数据小的质量函数.

2) 基于相似性的分布式上下文推理

在基于相似性的分布式上下文推理中,在不同复杂事件处理引擎中的高层上下文是通过 D-S 证据理论融合的. 在上下文更新处理过程中,基于模糊集相似性将新的上下文融合到存在的上下文中.

定义 15. 模糊集相似性. 识别框架为 $\Theta=\{\theta_1,\theta_2,\cdots,\theta_n\}$,模糊集 $\tilde{A}$ 和 $\tilde{C}$ 是两个随机模糊子集,则它们之间的相似度为

$$\omega(\tilde{C},\tilde{A}) = 1-\frac{1}{|\Theta|}\sum_i |\mu\tilde{C}(\theta_i)-\mu\tilde{A}(\theta_i)|. \quad (7)$$

基于相似性的分布式上下文推理处理算法如下:

算法 1. 分布式上下文推理.

输入:*level*1*Context* 为第 1 层上下文,*isUpdate* 表示是否仅更新;

输出:*Context* 为层次的上下文.

```
levels[1]←level1Context;
currentLevel←1;
loop
  evidenceSet ← SelectEvidence (levels[i],
    isUpdate);
  levels[currentLevel+1]←ClusterEvidence
    (evidenceSet,isUpdate);
  if (levels[currentLevel+1] is not changed)
    break
  end if
  controlNode←NegotiateControlNode();
  if (controlNode is ThisNode)
    ReceiveContext(newLevels[i+1]);
    if (isUpdate)
      MergeContext(levels[i+1],
      newLevels[i+1]);
    end if
  else
    SendContext(levels[i+1]);
  end if
end loop
return levels.
```

算法中的 *SelectEvidence* 函数执行证据选择策略. 逐级处理底层上下文,并根据定义 15 中的相似度进行聚类,生成高层上下文. *ClusterEvidence* 函数采用动态规划的方法,确保局部上下文更新时算法的效率.

汇聚高层上下文时,使用一个协商协议来选择控制节点(*NegotiateControlNode*()),从而确定分布式复杂事件处理引擎之间的上下文传输. 协议的主要内容是每个节点广播自己的负载和待传输的上下文大小,每个节点根据收到的消息计算自己作为控制节点的总代价,然后再进行一轮广播,最终代价最小的节点被选为控制节点. 此协议会将网络的数据流控制到最小,并平衡分布式节点间的负载.

算法的第 1 次执行会建立起上下文层次结构. 之后当上下文发生少量改变时生成新的高层上下文,并根据定义 14 计算新的上下文和本层的其他上下文之间的相似度. 新的上下文将会汇聚到相似度最高的已有上下文中.

使用上述推理过程可以得到高性能的分布式推理. 在算法 1 中并没有全局控制节点,即该算法为完全分布式. 传统分布式推理过程会随着上下文更新进行划分或合并,因此,当处理代理数量较大时推理过程将变得复杂. 而本文研究方法中,新上下文汇聚到相似度最高的已有上下文中,而不会影响其他分支的上下文.

3.4 分布式查询重写方法

算法 2 为本文的查询重写算法. 算法可以分解为 4 步:查询分析、上下文解决和事件流过滤,数据划分、查询规划器生成和执行.

复杂事件查询语句通过语义解析器分析,并创建语义树. 事件模型和上下文包含在语义树中,并且两者可以进行组合.

分布式查询重写方法的处理算法如下:

算法 2. *DCCEP*(*EC*,*Q*).

输入:*EC* 为原始事件流、*Q* 为查询语句;

输出:*EI* 为满足查询结果的事件实例.

```
subQueries←GlobalResolve(Q,globalContext);
isControlNode←GetControlNode(Q);
```

```
partitions←Partition(EC,nodes,
  globalContext);
for i=1 to nodesNumber do
  results[i]←subCEP(partitions[i],
    subQueries[i],isControlNode);
end
EI←MergeResult(results,globalContext);
Return EI.
SubCEP(SC,SQ,isControlNode):
```

输入:*SC* *SC* 为局部事件流,*SQ* 为复杂事件查询字句, *i* _*control*_*node* 表示是否需要 *sub*_*control* 节点;

输出:*SEI* 为子查询结果的复杂事件模型.

```
SEI←∅,event_filter←∅,temp_result←∅;
/*查询分析*/
eventPattern←AnalysisQuery(SQ);
eventContext←AnalysisContext(SQ);
/* resolve context and filter event stream */
for each sec eventContext do
  ResolveContextParameters (sec);
  sc←DecomposeContext(sec);
  eventFilter←eventFilterResolveEventFilter
    (sec);
end
dataWindow←Filter(eventFilter,SC);
/*数据划分和生成规划器*/
eventPartitions←PartitionEvent(dataWindow,
  eventContext,eventPattern);
subQueries←GenerateSubQuery(eventPartitions,
  eventContext,eventPattern);
/*并行执行子查询*/
i←1;
for each q sub_queries do
result[i]←executeParallel(q);
i←i+1;
end
SEI←MergeResult(result);
if (isControlNode)
  controlNode←getControlNode();
  if (controlNode==localNode)
      tempResult←ReceiveTempResult();
      SEI←MergeTempResult(TempResult,
        SEI);
  else
      SendTempResult(SEI);
  end if
end if
return SEI.
```

1) 上下文解决和事件流过滤

在局部 CEP 算法中,首先对事件模式和上下文进行详细解析. 对于带参数的上下文,首先要处理参数. 例如"酒店 *A* 前面 500 m 位置"这个上下文,首先要确定酒店 *A* 的位置. 本文采用 3 种方法:①事件数据库. 所有用户的查询会保存在事件数据库并索引,另外应用系统的一些事件信息也保存在事件数据库中. ②本体. 有些信息如酒店的位置,保存在本体中. ③子查询. 如果参数值不能从数据库或本体获取,就需要生成一个子查询来获取. 对于复合上下文中的每个基本上下文,有时需要进行上下文的分解.

2) 数据划分

为了提高效率,在局部 CEP 中进一步采取了数据划分和并行方法. 数据划分采用 3 种方式. ①基于空间上下文的划分. 首先把整个空间划分为 N 个局域(N 是一个可以根据计算资源设置的参数),位于某个区域的事件序列划分到该区域. 所有跨越了区域的事件序列,划分到一个新的 $N+1$ 区域,如图 4 所示. 这样的划分可以避免区域之间的事件重合,便于实现并行算法. ②基于分段的上下文划分. 如上下文"$30 \leqslant speed \leqslant 50$"可以划分为"$30 \leqslant speed < 40$"和"$40 \leqslant speed < 50$". ③基于时间的划分. 即根据事件的时标划分为多个时间段.

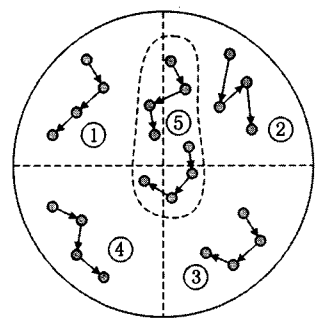


Fig. 4 Event partition by spatial context.

图 4 局部 CEP 引擎基于空间上下文的数据划分

数据划分依据以下原则:①如果数据窗口的长度小于某个阈值 α,则无需进行数据划分;②总体划分数不超过 N(N 根据代价分析和实验结果确定);③划分方式的优先级为:空间划分→分段划分→时间划分.

3) 查询规划器生成和执行

在这一步中,对于每个数据划分生成一个子查

询.分解后每个子查询在局部节点上并行执行,最终合并结果.算法的总体代价估计为:

$$Cost = Tp + \max_i(Tpi + \max_{ij}(Tsij) + Tmi) + Tm, \quad (8)$$

其中,Tp 和 Tm 分别为分布式查询分解和全局结果合并的代价,Tpi 和 Tmi 分别为局部查询分解和结果合并的代价.$Tsij$ 为局部节点 i 上子查询 j 的执行代价.对于全局查询分解和结果合并,采取文献[18]中的方法.对于局部数据划分和结果合并的方法,分析如下.

① 空间上下文划分:划分是基于"主对象"(例如车联网中的车辆)的位置,不同划分中的事件实例没有重叠,这样易于实现并行处理,Tmi 基本可忽略不计.

② 分段划分方式:可以根据 MAX,TOP-K 等事件的主属性进行划分,从而查询规划比较容易实现优化.例如在处理 MAX 事件时,只需要处理主属性的数值最高的那个分区,其他分区可以忽略.但对于有些事件模式如 ALL,SEQ 等,处理代价较高,应该尽量避免使用这种分区方式.

③ 时间划分方式:除了 ALL 和 SEQ 外,其他基本复杂事件模式的并行查询规划比较容易实现.对于 SEQ 模式,我们采用的方法是在每个分区建立有限状态机和活动实例栈,最终链接各个分区的实例栈来建立全局实例栈.对于共享主存的系统,无需数据移动.

例如在在交通物联网中,查找一个交通事故的目击证人.基本的模糊查询为"查询在时间段 T 内,事故地点 A 附近驶过的车辆".假设事先进行了车辆位置的相似度层次聚类,则可以在上下文中推理得到和 A 距离比较接近的所有传感器位置.根据这些传感器位置,确定目标车辆的行驶路线,如经过 A 的车辆;连续经过 B,C,D 的车辆等.上述查询被重写为多个上下文无关查询.如果事故地点位于不同区域的交叉点附近,则相关事件及上下文可能位于不同的服务器,从而需要进行分布式处理.在单个服务器上,事件的检测则可以通过本文的并行处理方法来提高效率.通过这个例子也可以看出总体模型设计是能够解决物联网中的实际问题并简化用户工作的.

在分布式复杂事件处理系统中,局部复杂事件处理引擎可能需要其他节点的数据.本文系统可以传输除原始数据外的临时结果.为了解决网络带宽和避免全局节点的负载,算法中选择一个合适的局部节点作为控制节点(*getControlNode*()方法),将临时结果发送到此控制节点,而不是全部发送到全局节点.从分布式上下文推理和分布式查询重写算法描述中可以看出,分布和并行算法会增加通信开销但不会明显增加内存的开销.

3.5 上下文无关并行复杂事件处理

对于物联网事件云来说,CEP 引擎的性能是至关重要的.其中,顺序模式(SEQ)处理的性能是最关键的.基本的序列模式检测方法采用有限状态机(NFA)的方法,如图 5 所示.在 NFA 中,由初始状态开始,当扫描到 SEQ 中的一个事件时进入下一个状态.当扫描到 SEQ 的最后一个事件时进入 NFA 的接受状态,也就意味着有 SEQ 事件被检测到.为了提高性能,把触发每个状态的事件记录到活动实例栈 AIS 中,并建立每个事件和前一个状态中事件的链表.这样在检测 SEQ 事件时只需搜索 AIS 中的链表.

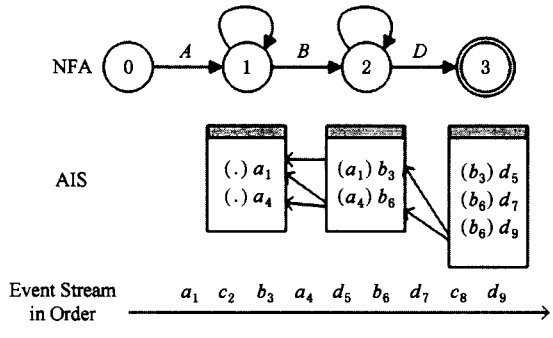


Fig. 5 Basic SEQ event query method.

图 5 基本 SEQ 事件检测方法

在基本的 SEQ 事件检测中,如果滑动窗口非常大,则算法的运行效率难以满足快速反应应用的需求.为了优化性能,本文提出并行的 SEQ 事件检测方法.并行的方法首先基于数据分区,根据时间段把滑动窗口中的数据划分为 N 个分区(N 是可以设置的值,通常和处理器数量或节点数量有关).算法执行过程如下:

1) 主控进程对任务进行划分,启动 N 个任务进程来处理滑动窗口中的数据;

2) 任务进程 P_i 运行基本 SEQ 检测算法,其输入为数据分区 Di,运行后直接输出其局部结果;

3) 主控进程对各个任务进程产生的 AIS 进行链接,然后根据链接的 AIS 输出剩余的结果.

其中第 2 步直接输出局部结果,是因为其局部结果必然是整体结果的一部分.剩下的关键问题就是 AIS 的合并.AIS 的合并基于本文提出的定理 1

实现.

定理 1. 并行 SEQ 操作的 AIS 合并. 设 P_1, P_2,…,P_n 为 n 个任务进程,NFA 中有 k 个状态 0, 1,…,$k-1$,则 $P_i(1<i\leqslant n)$ 的 AIS 中和 NFA 的状态 $j(1<j\leqslant k-1)$ 对应的所有事件,可以链接到 P_{i-1} 的 AIS 中和 NFA 的状态 $j-1$ 对应的所有事件.

证明. 由于我们假设输入的数据是有序的,分区是时间段进行划分的,因此 P_i 中的所有事件其事件戳大于 P_{i-1} 中的所有事件. 因此进程 P_{i-1} 的 AIS 中和 NFA 的状态 $j-1$ 对应的所有事件,发生在 P_i 的 AIS 中和 NFA 的状态 j 对应的所有事件之前,且位于 NFA 的前一个状态,根据 AIS 的构建规则可以建立相应的链接.

根据定理 1 对多个局部 AIS 建立链接以后,通过对全局 AIS 的搜索容易得到全局的输出结果.

例 4. 并行 SEQ 检测示例. 设输入的事件流为 (a_1 c_2 b_3 a_4 d_5 b_6 d_7 a_8 b_9 c_{10} a_{11} d_{12} b_{13} d_{14}),划分为两个分区,$a_1\sim d_7$ 为分区 1,$a_8\sim d_{14}$ 为分区 2. 分区 1 和分区 2 的事件分别由任务进程 1 和任务进程 2 处理,如图 6 所示. 任务进程 1 建立局部 AIS 并输出结果 (a_1 b_3 d_5),(a_1 b_3 d_7)等. 任务进程 2 建立局部 AIS 并输出结果 (a_8 b_9 d_{12}),(a_8 b_9 d_{14})等. 最后主控进程根据定理 1 链接局部 AIS 并输出全局结果 (a_1 b_9 d_{14}),(a_4 b_9 d_{14})等.

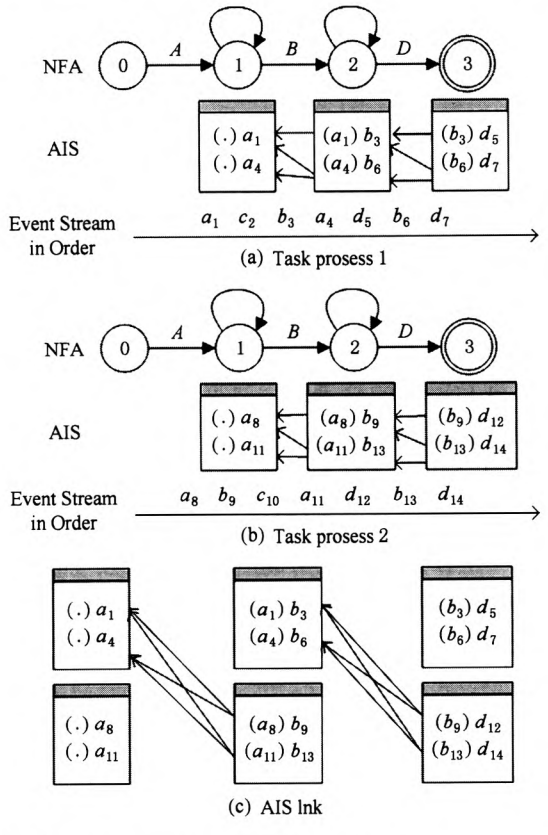


Fig. 6 The example of parallel complex event processing.

图 6 并行复杂事件检测示例

本文主要是对概率事件的处理,下面提出对于单个 SEQ 事件流的概率的计算.

定义 16. 单个 SEQ 事件流的概率计算. SEQ 事件流 $E=SEQ(e_1,e_2,\cdots e_n)$,原始事件划分为 S 和 T,集合 T 包含无关的原始事件,集合 S 包含多个马尔可夫链,则概率计算如下:

$$Pr(E)=\prod_{e_j\in T}Pr(e_j)\cdot \prod_{s_i\in S}(Pr(e_{i1})\prod_{m=1}^{|s_i|-1}Pr(e_{m+1}\mid e_m)), \tag{9}$$

s_i 为集合 S 的马尔可夫链,e_{i1} 是 s_i 的第 1 个事件,$Pr(e_{m+1}\mid e_m)$为条件概率.

扩展的 AIS 为概率活动实例栈(probabilistic active instance stack, PAIS),如例 5 所述.

例 5. 并行概率 SEQ 事件处理. 输入的概率事件流为 $a_1(0.6)$,$c_2(0.7)$,$b_3(0.5)$,$a_4(0.9)$,$d_5(0.8)$,$b_6(0.6)$,$d_7(0.8)$,$a_8(0.7)$,$b_9(0.5)$,$c_{10}(0.9)$,$a_{11}(0.6)$,$d_{12}(0.8)$,$b_{13}(0.6)$,$d_{14}(0.7)$,部分条件概率表如表 2 所示. 图 7 是创建 SEQ(A,B,D)的 NFA 和 AIS,为了并行处理,事件流划分为两个子事件流,根据定义 16 得到 $Pr(a_1,b_3,d_7)=0.6\times0.6\times0.9=0.324$,$Pr(a_1,b_{13},d_{14})=0.6\times0.7\times0.9=0.378$,通过查询局部 AIS 和全局 AIS,可以得到查询 SEQ(A,B,D)的所有实例概率.

Table 2 Condition Probability

表 2 条件概率表

Condition Event	Probability
$b_3\mid a_1$	0.6
$b_6\mid a_1$	0.7
$d_7\mid b_3$	0.9
$b_9\mid a_8$	0.6
$d_{14}\mid b_9$	0.8
$b_{13}\mid a_1$	0.7
$d_{14}\mid b_6$	0.9
⋮	⋮

在实际应用中,产生的复杂事件概率存在临界值,通过启发式策略来提高性能. 1)如果原始事件概率小于临界值,将不被加到 PAIS 中;2)搜索 AIS 事件链时计算当前路径的概率,如果此概率小于临界值,终止此搜索路径.

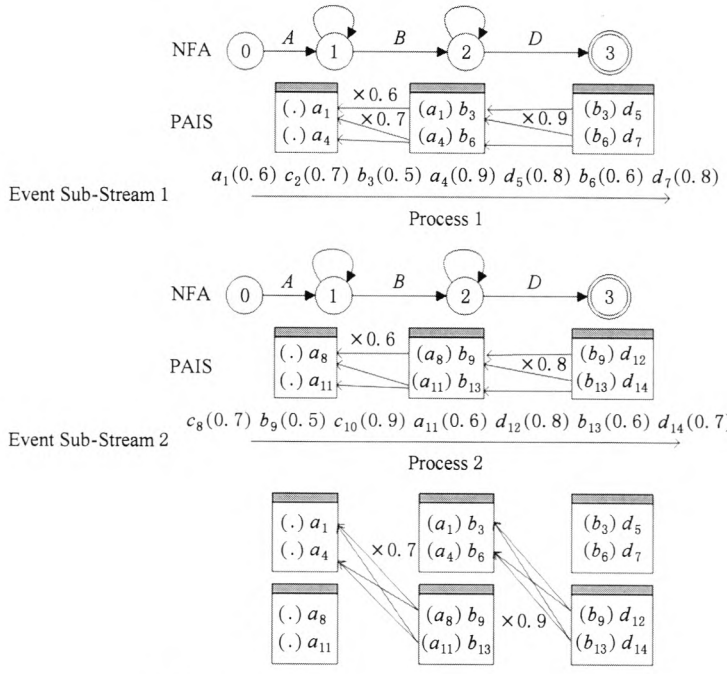


Fig. 7 Parallel probability complex event processing.

图 7 并行概率复杂事件处理

3.6 上下文无关分布式复杂事件处理

本文的研究方法中,通过数据移动和移动中间结果集支持分布式处理.其中,数据移动是根据起始时间和结束时间来划分数据窗口,例如图 8 为 4 个不同节点上的有序事件流,根据上述规则划分为 7 个时间段,为了满足并行处理方法,数据移动后需保证数据不存在时间重叠.如果某个时间段包含了多个事件流,把具有最大数据的节点作为主节点,并把此段内的其他节点的所有数据移动到主节点.假设经过划分后有 r 个段,需要移动的数据大小是 d_1, $d_2,\cdots,d_r$,则总的数据移动大小为 $D=\sum_{i=1}^{r}d_i$.在分布式处理中,需要将 PAIS 最大的作为主节点,并把其他节点的 PAIS 发送到主节点.

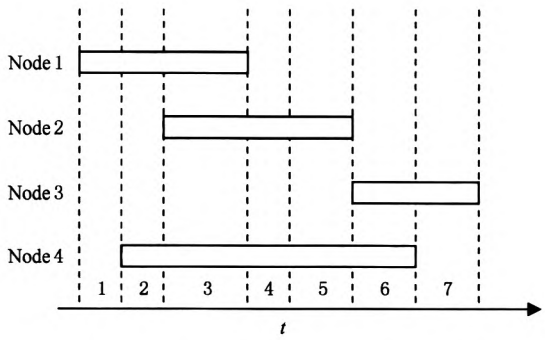


Fig. 8 The data distribution of different node.

图 8 不同节点的数据分布

数据移动在时间重叠率比较大时效率较低,使用移动中间结果方法较为有效.移动中间结果不必移动原始事件数据,根据局部事件流创建每个节点的 PAIS,并输出局部结果.把 PAIS 最大的节点作为主节点,其他节点的 PAIS 发送到主节点.最后主节点汇聚所有其他 PAIS 到自己的 PAIS,并创建新的 PAIS.

4 实验研究

本文采用的实验环境为 4 台 4 GB 内存的惠普服务器,通过高速网络连接,每台有 4 个英特尔 Xeon 处理器.其中 1 台作为主节点,其他 3 台作为从节点.操作系统为 Redhat Advanced Linux 5.基于 NetLogo 开发了一个车联网仿真器,可以根据设备配置自动生成各种 RFID 和传感器事件.根据配置不同,系统能够从简单到复杂准确模拟多种车联网情境.系统产生的事件根据所处的位置划分到多个 CEP 服务器进行处理.根据用户的查询,CEP 引擎可以从仿真器读取原始事件并进行复杂事件处理.为了支持模糊上下文,我们建立了一个基本的小城市车联网模糊本体.

DCCEP 的性能和事件选择率 S_f 相关,S_f 定义为 $N_p/N_w\times100\%$,其中 N_p 为复杂事件的事件实例

数量，N_w 为数据窗口的长度. 本文实验采用稠密和稀疏两种类型的数据，稠密数据的 $S_f>10\%$，稀疏数据的 $S_f<0.1\%$. 为了测试性能我们把 DCCEP 和基本方法作比较. 基本方法采用扩展的 SASE 和树结构在局部节点处理复杂事件[18]，其上下文支持和分布式支持通过人工重写的方式实现.

第 1 个实验测试了分布式多层次上下文推理方法(DMCR). 我们实现了 Amir 的“划分与合并(P&M)”方法[28]，并将 P&M 方法和本文研究的推理方法(DMCR)进行对比. 结果如图 9 所示. 从图 9 可以看出当上下文层次增加时 DMCR 的性能优于 P&M 方法，DMCR 的运行时间减少比 P&M 慢，原因在于 DMCR 使用基于相似性方法，避免了许多高层上下文节点的计算.

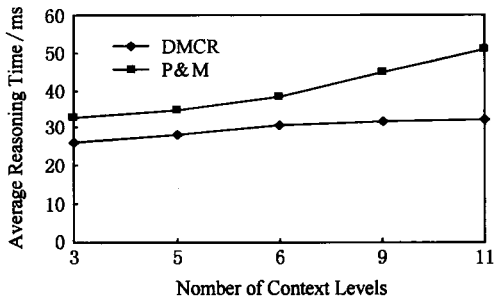


Fig. 9 Event partition by spatial context.

图 9 空间上下文的事件划分

第 2 个实验在稠密数据上对比 DCCEP 和基本方法，基本方法是基于有限状态机和树来处理复杂事件[29]. 由于基本方法不支持上下文，本文研究重写了上下文以支持简单的上下文感知. 实验中选择 SEQ 和 AND 模式来测试性能和模式中事件数量的关系，数据窗口设置为 3×106. 结果如图 10 所示. 从图 10 可以看出，当事件数量增加时两种方法的性能都下降，原因是事件数量增加导致状态机和树结构变的复杂. 但 DCCEP 方法的性能优于基本方法，并且其性能下降较慢，原因是采用了查询优化和子查询并行的方法.

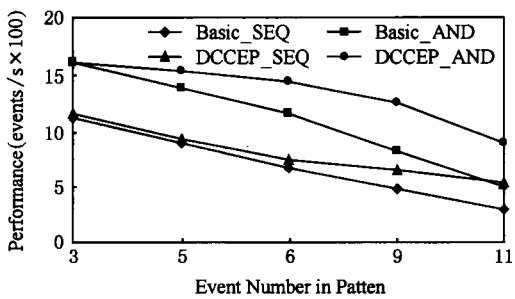


Fig. 10 Performance comparision of basic method and DCCEP on dense data set.

图 10 稠密数据下 DCCEP 和两种基本方法的性能对比

在第 3 个实验中，对比了稠密数据和稀疏数据上的性能，结果如图 11 所示. 图中“_D”代表稠密数据，“_S”代表稀疏数据. 从图 11 可以看出算法在稀疏数据上的性能高于在稠密数据上的性能，原因是在稠密数据上有限状态机和匹配树会变得比较复杂. 当事件数量增加时两种方法的性能都下降，但由于采用了查询优化 DCCEP 算法的性能下降小于基本方法.

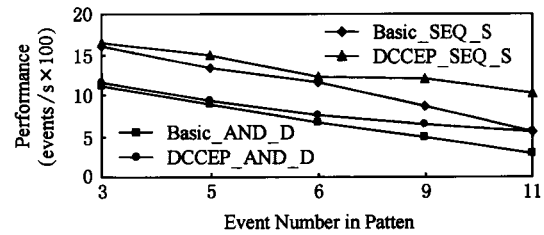


Fig. 11 Performance comparision of basic method and DCCEP on dense data set and sparse data set.

图 11 DCCEP 和两种基本方法在稠密和稀疏数据下的性能对比

最后一个实验对 DCCEP 在稀疏数据上的可伸缩性进行了实验，结果如图 12 所示. 实验中选择了 SEQ，AND 和 TOP-K 模式. 结果显示当数据量增大到某一个值时算法性能没有明显下降. DCCEP 具有很好的可扩展性，原因在于它使用优化的基于相似性的分布式上下文推理方法，并优化了查询重写方法，充分利用多个服务器和处理机.

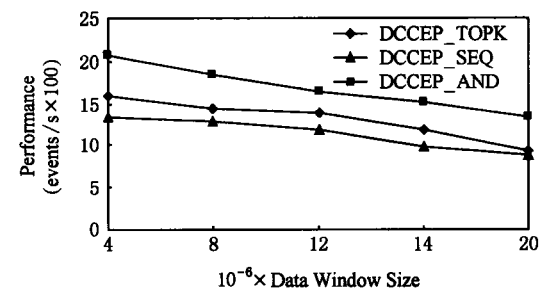


Fig. 12 Scalability of DCCEP on sparse data set.

图 12 稀疏数据下 DCCEP 的可扩展性

从上述实验可以看出，本文研究方法在处理物联网的分布式上下文敏感复杂事件时是有效的，在大规模物联网应用中具有比一般方法更好的性能和可伸缩性，并且在开销方面，CPU 的利用率和算法的并行度相关. 对单个节点而言，当任务数为 $k\times N$ 时效率最好(其中 N 为 CPU 数量，k 为一个系数).

5 总结和展望

由于传统的 Hadoop/MapReduce 的框架不能完全适用于处理物联网实时数据,本文提出了一个面向物联网的高性能分布式上下文敏感复杂事件处理方法,有效提高了数据处理效率.该方法使用模糊本体来进行上下文建模,从而解决事件的不确定问题及模糊事件查询问题.采用基于相似性分布式上下文推理,并通过查询重写,把上下文相关查询转换为上下文无关子查询.根据不同的事件模型和上下文划分数据,并通过优化和并行执行子查询来提高性能.同时,本文还包括了针对上下文无关事件处理的进一步分布和并行算法.实验表明该方法支持复杂事件的模糊查询,并比一般方法具有较好的性能和可扩展性,并且在数据量比较大时,该方法比一般的方法更具备性能优势.

但本文还有一些不足之处,首先,当模糊本体的规模比较大时上下文推理的性能仍需提高,尤其是D-S证据理论的计算.其次,分布式上下文敏感复杂事件处理方法还需要对层次复杂事件进行进一步的优化,下一步研究将重点解决这些问题.

参 考 文 献

[1] JManyika J, Chui M, Brown B, et al. Big data: The next frontier for innovation, competition, and productivity [R]. McKinsey, 2011 [2012-06-03]. http://www.mckinsey.com/insights/business_technology/big_data_the_next_frontier_for_innovation

[2] Luckham D. The power of events: An introduction to complex event processing in distributed enterprise systems [M]. Reading, MA: Addison-Wesley, 2002

[3] Wu E, Diao Y, Rizvi S. High-performance complex event processing over streams [C] //Proc of the 2006 ACM SIGMOD Int Conf on Management of data, New York: ACM, 2006: 407-418

[4] Meng Xiaofeng, Ci Xiang. Big data management: Concepts, techniques and challenges [J]. Journal of Computer Research and Development, 2013, 50(1): 146-169 (in Chinese)
(孟小峰, 慈祥. 大数据管理:概念、技术与挑战[J]. 计算机研究与发展, 2013, 50(1): 146-169)

[5] Li Y, Wang J, Feng L, et al. Accelerating sequence event detection through condensed composition [C] //Proc of the 5th Int Conf on Ubiquitous Information Technologies & Applications. Piscataway, NJ: IEEE, 2010: 1-6

[6] Wang F, Liu S, Liu P, et al. Bridging physical and virtual worlds: Complex event processing for RFID data streams [C] //Proc of the 10th Int Conf on Extending Database Technology. Berlin: Springer, 2006: 588-607

[7] Jin X, Lee X, Kong N, et al. Efficient complex event processing over RFID data stream [C] //Proc of the 7th IEEE/ACIS Int Conf on Computer and Information Science. Piscataway, NJ: IEEE, 2008: 75-81

[8] Agrawal J, Diao Y, Gyllstrom D, et al. Efficient pattern matching over event streams [C] //Proc of the 8th ACM SIGMOD Int Conf on Management of Data. New York: ACM, 2008: 147-160

[9] Zhang H, Diao Y, Immerman N. Recognizing patterns in streams with imprecise timestamps [J]. VLDB Endowment, 2010, 3(1): 244-255

[10] Akdere M, Çetintemel U, Tatbul N. Plan-based complex event detection across distributed sources [J]. VLDB Endowment, 2008, 1(1): 66-77

[11] Ku T, Zhu Y, Hu K, et al. A novel distributed complex event processing for RFID application [C] //Proc of the 3rd Int Conf on Convergence and Hybrid Information Technology. Piscataway, NJ: IEEE, 2008: 1113-1117

[12] Miraoui M, Tadj C, Ben Amar C. Context modeling and context-aware service adaptation for pervasive computing systems [J]. International Journal of Computer and Information Science and Engineering (IJCISE), 2008, 2(3): 148-157

[13] Almeida A, López-de-Ipiña D. Modeling and managing ambiguous context in intelligent environments [C]//Proc of the 5th Int Symp of Ubiquitous Computing and Ambient Intelligence (UCAMI 2011). Basel, Switzerland: MDPI, 2011

[14] Singh A, Juneja D, Sharma A K. A fuzzy integrated ontology model to manage uncertainty in semantic Web [J]. The FIOM Int Journal on Computer Science and Engineering (IJCSE), 2011, 3(3): 1057-1062

[15] Zhang H W, Chen K. Building social relationship ontology model based on fuzzy set [J]. JDCTA: Int Journal of Digital Content Technology and its Applications, 2012, 6(15): 459-466

[16] Cai Y, Yeung C A, Leung H. Fuzzy Computational Ontologies in Contexts [M]. Beijing: Higher Education Press, 2012

[17] Lee C H, Pham T H, Hee Seong Kim, et al. Similarity based distributed context reasoning with layer context modeling [C] //Proc of the 35th IEEE Annual Computer Software and Applications Conf. Piscataway, NJ: IEEE, 2011: 320-325

[18] Helmer S, Poul A, Xhafa F. Reasoning in Event-Based Distributed Systems [M]. Berlin: Springer, 2011

[19] Ashish K A. Ontology-driven complex event processing in heterogeneous sensor networks [C] //Proc of the 5th ACM Int Conf on Distributed Event-Based Systems. New York: ACM, 2011: 23-28

[20] Zhou Q, Simmhan Y, Prasanna V K. Towards an inexact semantic complex event processing framework [C] //Proc of the 5th ACM Int Conf on Distributed Event-Based Systems. New York: ACM, 2011: 401-402

[21] Teymourian K, Paschke A. Enabling knowledge-based complex event processing [C] //Proc of the 2010 EDBT/ICDT Workshops. New York: ACM, 2010: 37:1-37:7

[22] Rabinovich E, Etzion O, Gal A. Pattern rewriting framework for event processing optimization [C] //Proc of the 5th ACM Int Conf on Distributed Event-Based Systems. New York: ACM, 2011: 101-112

[23] Schultz-Møller N P, Migliavacca M, Pietzuch P. Distributed complex event processing with query rewriting [C] //Proc of the 3rd ACM Int Conf on Distributed Event-Based Systems. New York: ACM, 2009: 1-12

[24] Peukert E, Berthold H, Rahm E. Rewrite techniques for performance optimization of schema matching processes [C] //Proc of the 13th Int Conf on Extending Database Technology (EDBT). New York: ACM, 2010: 453-464

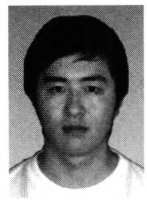
[25] Lembo D, Lenzerini M, Rosati R, et al. Query rewriting for inconsistent DL-lite ontologies [C] //Proc of the 5th Int Conf on Web Reasoning and Rule Systems. Berlin: Springer, 2011: 155-169

[26] Etzion O, Niblett P. Event Processing in Action [M]. Stamford: Manning Publications, 2010

[27] Li G Q, Zou H, Yang F. Fuzzy ontology and fuzzy D-S evidence theory based context modeling and uncertainty reasoning [J]. Journal of Convergence Information Technology, 2011, 6(12): 185-189

[28] Amir Padovitz, Seng Wai Loke, Arkady B. Zaslavsky: Multiple-agent perspectives in reasoning about situations for context-aware pervasive computing systems [J]. IEEE Trans on Systems, Man, and Cybernetics, Part A, 2008, 38(4): 729-742

[29] Wang Y H, Yang S H. Plan-based distributed complex event processing for RFID application [C] //Proc of the 2009 Int Conf on Computational Intelligence and Software Engineering. Piscataway, NJ: IEEE, 2009: 1-4



Cao Kening, born in 1982. PhD candidate of Hunan University. Student member of China Computer Federation. His current research interests include Internet of things, complex event processing. etc.

Wang Yongheng, born in 1973. Assistant Professor of Hunan University. Senior member of China Computer Federation. His main research interests include Internet of things, data mining.

Li Renfa, born in 1957. Professor and PhD supervisor of Hunan University. Senior member of China Computer Federation. His main research interests include CPS, embedded systems and wireless sensor networks.

Wang Fengjuan, born in 1987. Master of Hunan University. Her current research interests include Internet of things, complex event processing. etc (wang_fengjuan31@163.com).

面向物联网的分布式上下文敏感复杂事件处理方法

作者： 曹科宁， 王永恒， 李仁发， 王凤娟， Cao Kening， Wang Yongheng， Li Renfa， Wang Fengjuan
作者单位： 曹科宁,Cao Kening(湖南大学信息科学与工程学院 长沙 410082;湖南省质量技术监督局 长沙410111)，王永恒,李仁发,王凤娟,Wang Yongheng,Li Renfa,Wang Fengjuan(湖南大学信息科学与工程学院 长沙 410082)

刊名： 计算机研究与发展 ISTIC EI PKU
英文刊名： Journal of Computer Research and Development
年，卷(期)： 2013,50(6)
被引用次数： 4次

参考文献(29条)

1. JManyika J;Chui M;Brown B Big data:The next frontier for innovation,competition,and productivity 2012
2. Luckham D The power of events:An introduction to complex event processing in distributed enterprise systems 2002
3. Wu E;Diao Y;Rizvi S High-performance complex event processing over streams 2006
4. 孟小峰,慈祥 大数据管理：概念、技术与挑战[期刊论文]-计算机研究与发展 2013(1)
5. Li Y;Wang J;Feng L Accelerating sequence event detection through condensed composition 2010
6. Wang F;Liu S;Liu P Bridging physical and virtual worlds:Complex event processing for RFID data streams 2006
7. Jin X;Lee X;Kong N Efficient complex event processing over RFID data stream 2008
8. Agrawal J;Diao Y;Gyllstrom D Efficient pattern matching over event streams 2008
9. Zhang H;Diao Y;Immerman N Recognizing patterns in streams with imprecise timestamps 2010(01)
10. Akdere M;Cetintemel U;Tatbul N Plan based complex event detection across distributed sources 2008(01)
11. Ku T;Zhu Y;Hu K A novel distributed complex event processing for RFID application 2008
12. Miraoui M;Tadj C;Ben Amar C Context modeling and context-aware service adaptation for pervasive computing systems 2008(03)
13. Almeida A;L6pez-de-Ipi(n)a D Modeling and managing ambiguous context in intelligent environments 2011
14. Singh A;Juneja D;Sharma A K A fuzzy integrated ontology model to manage uncertainty in semantic Web 2011(03)
15. Zhang H W;Chen K Building social relationship ontology model based on fuzzy set 2012(15)
16. Cai Y;Yeung C A;Leung H Fuzzy Computational Ontologies in Contexts 2012
17. Lee C H;Pham T H;Hee Seong Kim Similarity based distributed context reasoning with layer context modeling 2011
18. Helmer S;Poul A;Xhafa F Reasoning in Event-Based Distributed Systems 2011
19. Ashish K A Ontology-driven complex event processing in heterogeneous sensor networks 2011
20. Zhou Q;Simmhan Y;Prasanna V K Towards an inexact semantic complex event processing framework 2011
21. Teymourian K;Paschke A Enabling knowledge-based complex event processing 2010
22. Rabinovich E;Etzion O;Gal A Pattern rewriting framework for event processing optimization 2011
23. Schultz-Moller N P;Migliavacca M;Pietzuch P Distributed complex event processing with query rewriting 2009
24. Peukert E;Berthold H;Rahm E Rewrite techniques for performance optimization of schema matching processes 2010
25. Lembo D;Lenzerini M;Rosati R Query rewriting for inconsistent DL-lite ontologies 2011
26. Etzion O;Niblett P Event Processing in Action 2010
27. Li G Q;Zou H;Yang F Fuzzy ontology and fuzzy D-S evidence theory based context modeling and uncertainty reasoning 2011(12)
28. Amir Padovitz;Seng Wai Loke;Arkady B Zaslavsky:Multiple-agent perspectives in reasoning about situations for context-aware pervasive computing systems 2008(04)
29. Wang Y H;Yang S H Plan-based distributed complex event processing for RFID application 2009

引证文献(2条)

1. 雷一原 物联网复杂事件的实时处理路径探析[期刊论文]-计算机光盘软件与应用 2013(22)
2. 陈淡泊,仓一倩 基于Hadoop的改进apriori算法应用[期刊论文]-微型电脑应用 2015(10)