物联网事件处理与融合感知

王骁

章节

事件描述与数据处理

事件流检测与事件共享

时空语义的数据融合

环境上下文感知技术

# 引言

随着物联网IoT（Internet of Things）与人工智能AI（Artificial Intelligence）领域的发展，人工智能逐渐赋能物联网，增强物联网的智能能力，面向未来万物互联时代的到来。AI带来的高数据处理能力，与物联网带来的海量数据，构成一种极强的AIoT技术应用场景，在工业生产、智慧家庭、智慧城市等领域得到了广泛应用。物联网体系中的感知层带来的大数据问题，需要物联网中间件与应用层具备数据与事件处理的能力，才能对数据与事件进行有效的管理与利用，体现万物互联的信息与价值。在物联网技术中，复杂事件处理与环境上下文融合感知是两项关键技术。复杂事件处理用于从分散的系统中提取信息并构建和管理信息，事件是该处理技术的核心。环境上下文融合感知是利用物联场景中的时间、空间与事件上下文信息，提供性能与效果更优的感知能力，增强对物联网所在环境、场景行为的语义理解。这两项技术十分契合物联网领域诸多场景的基本任务，属于当前物联网领域的研究热点。

本章内容面向物联网事件处理与环境上下文融合感知技术。其中，物联网事件处理包括两个部分，分别是事件描述与数据处理、事件流检测与事件共享；环境上下文融合感知技术包括两个部分，分别是时空语义的数据融合、环境上下文感知技术。事件描述与事件处理部分，根据物联网的特征与需求，对物联网事件与事件流进行定义，并对物联网数据抽象的描述方法进行介绍；事件及事件信息的背后支撑者是数据，因此介绍和讲解其涉及到的关键数据预处理技术与事件处理框架。事件流检测与事件共享部分，核心在于复杂事件处理技术，以及事件共享机制。以上两部分提供了物联网主要的事件描述与处理方法基础，物联网事件包含丰富的时空语义信息与环境上下文信息，物联网技术要求对这两类信息进行有效利用，环境上下文融合感知提供了这类能力。时空语义的数据融合部分，解答何谓时空语义数据，对时空语义进行描述与约束，揭示关键的时空语义融合算法。环境上下文感知技术部分部分，介绍环境上下文感知技术的关键问题、生命周期、建模与决策推理模型关键技术。

# 事件描述与事件处理

「内容规划」

1. 事件描述

事件描述及事件分类，分为以下几个部分：

* 物联网数据抽象描述方法（数据模型）
* 物联网事件与事件流定义及其特点

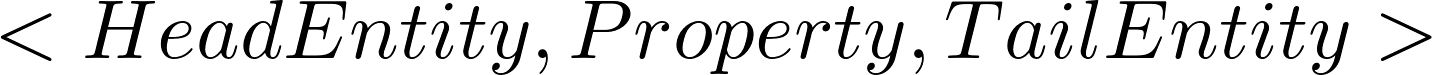
1. 事件处理

* 数据预处理
* 物联网事件处理框架
* 基于事件描述的事件处理系统

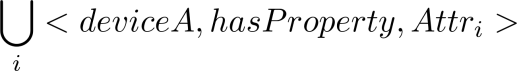
## 物联网数据抽象描述方法

物联网正在推动基础架构中具有唯一性、可识别性的嵌入式计算设备的互联。面对海量的可连接设备，一个实际的物联网系统可能从成千上万个互联设备中收集大量数据。这种数据爆炸需要高度灵活的数据模型，才能收集数据的过程与结果进行有效利用。数据模型可以让一个物联网系统具备监视、处理、优化和分析数据以获得对数据的洞悉，进而做出更好的决策或任何其他可行的结果。

在之前的章节中，我们对物联网的语义信息表达技术进行了介绍。对感知数据进行语义建模，是物联网数据抽象描述方法的核心。以Ontology技术为核心数据模型，将物联网系统与场景中的各个“物”与“物”的属性归纳为语义实体Entity。每个语义实体包含诸多属性，可以采用三元组的形式描述实体之间的关系：



三元组中分别为头实体、属性关系、尾实体。采用三元组的形式化描述，当头实体HeadEntit代表物联网系统中的“物”，属性关系代表语义“hasProperty”即拥有某种属性，尾实体TailEntity代表某种具体的属性概念时，可以用一些列三元组的组合，抽象出“物”的属性的形式化描述，这里假设“物”为deviceA，则可以用以下描述表征deviceA的属性：



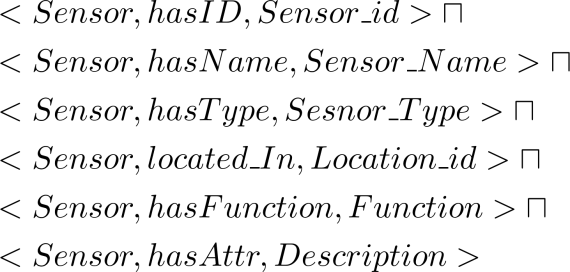
其中Attri为可选的属性，通常包括功能、类型、数据、位置等。以下是对常用的实体以及属性表示和描述进行总结。

|  |  |  |
| --- | --- | --- |
| 概念 | 描述 | 示例 |
| Entity | 头实体与尾实体，记录抽象的概念。一般对物联网场景的描述离不开传感设备、位置、时间、数据值、属性描述等。 | * Sensor * Device * Description * Location * Value |
| Property | 描述头实体与尾实体之间的关系，通常包含概念之间的转移逻辑。 | * hasClass * hasProperty * hasValue * locatedIn * hasAttr |

举一个具体的例子，对于一个传感器:

/private/var/folders/y7/_dxbgjfd16g8grhzs9syx0980000gn/T/com.kingsoft.wpsoffice.mac/wpsoffice.xPlmAYwpsoffice

括号内代表其拥有的相关概念和实体，则可表示为：



每个实体的数据可以分为静态数据（名称、固定位置、固定属性等）与动态数据（时间、测量值等）。在本书前文的感知数据语义部分有更多介绍，不再赘述。

## 物联网事件与事件流

在拥有了对物联网数据抽象的能力后，可以将物联网事件定义为传感器或设备触发而产生的物联网数据。

**1. 物联网事件**

**物联网事件event**: 物联网系统中，由传感器或设备等感知实体被触发，进而获取到的包含状态属性（时间属性、空间属性）与设备属性等信息的数据。

根据对物联网事件的定义，物联网事件具有两个重要特点，即触发性与包含属性的数据特性：

1. 触发性：来源于物联网传感层的特性，传感设备与传感网络具备监测能力，受外部与内部环境的作用触发产生数据。
2. 数据的属性特性：只有包含一定的状态、设备属性信息的数据，才是有效的事件，这些包含的信息我们希望物联网系统能够理解与运用的语义信息。

物联网的数据处理常面向数据流。在一个时间无限延续的环境中，触发的事件产生了排布在时间序列上的一个个数据，通常被成为事件流。海量的物联网事件数据所形成的事件流，作为数据流的一种，具有大数据所常有的6V（Volume, Velocity, Variety, Veracity, Valence, Value）特性，除此之外事件流相较单个事件，包含更多的语义与上下文信息。事件流同时也带了了更多挑战，这些挑战体现在相较处理单个数据，事件流需要考虑物联网的非均匀分布特点，解决数据传输过程中由硬件、网络、存储、计算等带来的数据偏差与事件乱序现象，这些数据包含一定的误差，数据不能按真实产生的真实时间戳形成有序输入输出。

**2. 物联网事件的分类**

由于物联网数据的语义隐藏在数据中，因此需要物联网系统具备对象的语义表达能力。具备语义能力的物联网系统处理对象为事件流，通过多个事件流筛选、聚合等实时计算产生语义更丰富的事件，称为复杂事件。根据事件本身的语义复杂性，将事件分为原始事件、基本事件、复杂事件。下面分别对三种类型事件进行介绍。

图 物联网事件分类

原始事件（primitive event, PE）。原始事件来源为物联网感知层中的感知设备。感知设备对目标对象的感知，产生一些列具有时间戳、空间位置、设备属性的结果。这些结果一部分能被直接利用，但考虑到感知设备获得的原始数据需要被进一步处理，以增强其数据质量、提取更多有效语义信息，所以往往不被直接利用。原始事件的属性通常由感知设备和感知目标共同决定。一个包含感知设备、感知目标、感知数据、感知时间的原始事件可以进行如下的形式化表示：

wpsoffice

基本事件（basic event, BE）。基本事件是由原始事件构成的事件流，一般作为上层复杂事件的输入。其一般可以包含简单的语义信息，表示一定语义目标对象的时空关联行为。相较于原始事件，基本事件更多被用户定义，是一些列用户感兴趣的监测结果，这样的监测结果往往并不复杂，能够被简单的规则所表达，因此也为用户对基本事件和事件参数的定义提供了可行性。

复杂事件（complex event, CE）。事件是系统中的重要情况，复杂事件是一种聚合事件，通过使用事件算子，将基本事件或复杂事件进行复合产生而产生的。复杂事件抽象上层语义对象之间的关系，以基本事件的事件流为核心。

## 物联网事件处理

物联网系统中包含三类事件，原始事件、基本事件与复杂事件。不同层次事件的事件处理目标存在差异，一般情况下，原始事件处理主要需求为对原始数据的清洗、对数据偏差的修正；基本事件处理考虑将原始数据依据一定规则形成中间数据，这部分中间数据是上层复杂事件处理的输入，而这些规则往往面向众多简单事件中的一些具有特定语义的事件的提取过程与抽象过程；复杂事件处理则是对多个原始、基本事件构成的事件流进行筛选、聚合等实时计算。

本节主要介绍原始事件处理和复杂事件处理概述，介绍事件语言和事件处理过程中的表达式与操作符。根据上节定义，基本事件的处理主要由用户兴趣和用户规则决定，因其自定义性较强，为了不失一般性不在本节介绍。

### 原始事件处理

物联网系统中感知设备对数据的采集、传输、收集集中工作，由于感知设备的感知精度、存储性能、网络传输性能之间的差异性，为原始数据的可靠性带来挑战。为了提供可靠语义的原始事件，原始事件处理需要解决各属性数据上的偏差。

考虑到物联网的开放特性，其面向的环境之间的差异可能是巨大的，针对具体场景的传感问题往往需要考虑特殊的原始数据的处理方式。为了不失一般性，本节提出和介绍在任何场景都可能用到的原始数据与原始事件处理方法，可以为特殊场景的原始事件处理设计提供参考。首先在下面的表格中给出概念性介绍，其次对其中涉及到的具体数学方法与处理方法进一步介绍。

表 原始事件处理

|  |  |
| --- | --- |
| 原始事件的属性与特征的明确 | 明确原始事件应包含哪些属性、哪些特征，这些属性与特征是否连续，是否为类别信息。 |
| 缺失值处理 | 缺失值检查，采用适合的方式对缺失值进行弥补，保证数据完整性。 |
| 噪声处理 | 原始数据噪声极大的影响其他任务的结果，需要通过滤波等方式去除噪声。 |
| 类别型信息的编码 | 离散的类别信息需要通过编码的方式，使传输和后续处理成为可能。 |

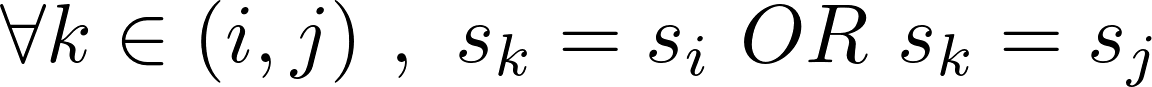
#### 缺失值处理

缺失值产生的原因，主要是因为物联网场景中，感知设备精度与工作状态无法得到保障，存在可能性不能在合适的时间做出正确响应，感知设备可能面临失灵、失效等复杂情况。物联网系统中常用低功耗、无线通信设备，这些设备终端为了功耗和性能的节约，可能采用一些不包含认证过程、不具有可靠性的协议，因此从网络上带来数据丢失的可能。在现实中，无线低功耗设备发出的数据分组丢失是常见状况。因此，即便物联网系统尽可能保证设备正常工作，对缺失值的处理仍然是必不可少的，原始事件处理后续的复杂事件处理往往需要数据具备一定的完整性与可靠性。

缺失值常与时间相关。无论是感知设备工作机制，还是通信协议的机制，造成的原始数据流在某个时间应该存在感知结果，但实际缺失感知结果的情况，是本节面对的缺失值。对此问题进行抽象表达，对原始数据流对应的时间戳wpsoffice，两个相邻时间戳之间的事件上的原始数据是缺失的数据。对于中间的缺失值，以下是三种缺失值补充方法。

1. **最后观测原则**

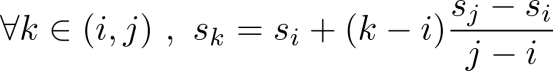
最后观测原则认为两个事件之间的缺失值，应该与最近一次观测的结果保持一致。为观测到的情况代表没有产生新的事件，因而事件的结果没有发生变化。假设wpsoffice，其中i<j，是两个时间戳，两个时间戳之间的值为缺失值wpsoffice，那么采用最后观测原则补充缺失值应为：



其中前者采用向前最后观测原则，后者采用向后最后观测原则。

**2. 线性插值**

对于时间i与时间j之间的缺失值，采用线插值表示为



**3. 非线性插值**

非线性插值不是一种插值，是许多种不是线性插值方法的统称。其中三阶样条插值是常用的插值方法。

**缺失值处理方法比较**

缺失值处理的效果因物联网场景而异。线性插值与非线性插值对于数值型属性的处理效果较好，最后观测原则对于类别型属性则更合适。下图是3种缺失值处理方式在一定数据上的效果，红点表示原始事件的真实值;3种不同颜色的曲线中，蓝色曲线代表最后观测原则，黄色曲线代表线性插值，绿色曲线代表非线性插值（三阶样条插值）。这些原始事件在时间上是不等时间间隔的，前6个事件之间时间间隔较大，其余事件事件间隔较小。将3种处理方法的比较结果总结在下表种。

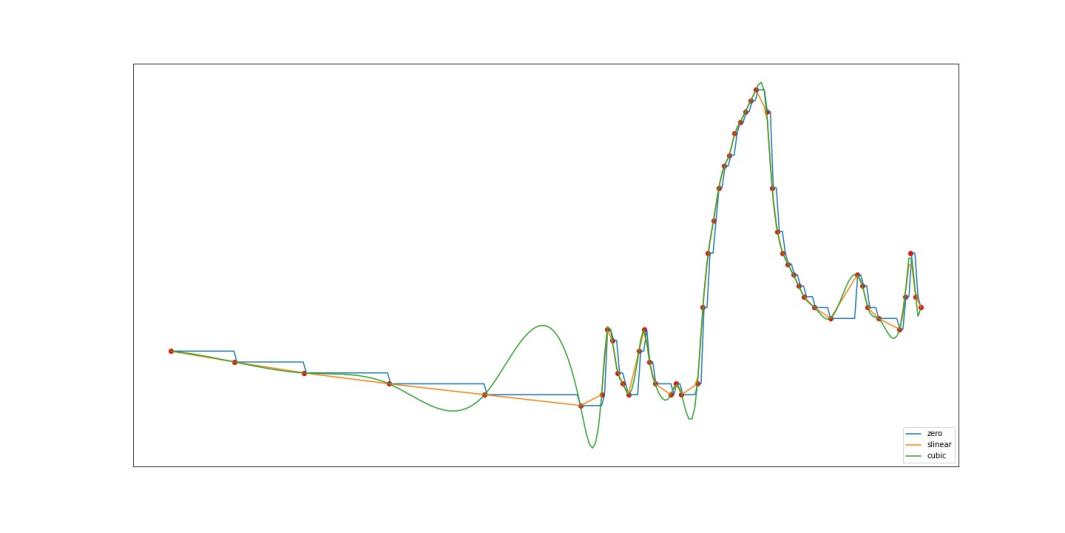


图 3种缺失值处理方法的效果

表 3种缺失值处理方法比较结果

|  |  |  |
| --- | --- | --- |
| 缺失值处理方法 | 优势场景 | 劣势场景 |
| 最后观测原则 | 类别型属性；  长时间间隔事件； | 短时间间隔场景；  精度要求较高的场景； |
| 线性插值 | 数值型属性；  长时间间隔与短时间间隔事件； | 产生复杂数值事件中缺失值较多的情况； |
| 非线性插值 | 数值型属性；  短时间间隔事件；  产生复杂数值事件中缺失值较多的情况； | 长时间间隔事件，容易产生过拟合现象； |

#### 噪声处理

原始事件伴随的原始数据常充满噪声，物联网诸多应用场景——比如广泛应用的可穿戴设备，在设备上部署的加速度计与回旋计常能采集到与动作无关的原始数据。这些无关原始数据可能影响后续任务，比如当使用一定的机器学习模型对动作数据进行处理时，无关数据影响整个识别过程的学习、识别与泛化能力。噪声为人工智能赋能物联网带来了障碍，源于人工智能往往以数据驱动，对数据的可靠性存在一定的要求。物联网事件处理，需要尽可能的在贴近数据来源的层次，对噪声进行处理，提高原始数据的可靠性。噪声处理的方法在信号处理等学科中被广泛介绍，其因信号的类型和层次而异，常涉及频谱变换、频率信号分析、滤波等。

本节的噪声处理面向感知设备采集的结果。对于原始事件与相应的原始数据的噪声处理，通常对物联系统中的噪声进行一定的假设。物联系统中的噪声分为系统噪声和随机噪声。系统误差来源感知设备与感知网络所处环境，以及环境特殊因素。随机误差来源于感知场景中发生的随机事件、感知设备精度、感知传输过程等。

**1. 系统噪声**

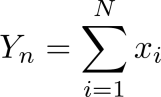
系统噪声是比较棘手的噪声，其往往由于对物联场景考虑不周，而导致获得的原始数据与真实情况之间存在一定程度的偏差。举一个简单且有些极端的例子，使用某个感知范围为1米的传感器监测是否有人通过，但其安装位置与人常经过的位置距离在1米以上。这样的失误反应在原始事件上，监测结果为一直无人经过，无论数据层面进行何种处理，都无法解决其错误结果。所以消除系统误差，往往需要优化感知系统，这是一项复杂的工作，往往伴随大量的实验与测试。

**2. 随机噪声**

相较系统噪声，随机噪声对感知结果的影响是一种时间和空间独立的随机过程，不受时间、空间对幅度、波形、相位进行随机影响。物联系统中的感知设备，感知获得原始事件结果存在许多子过程。一个原始事件从事件源处产生，经过事件信道最终达到事件宿，在事件宿处产生为原始事件。

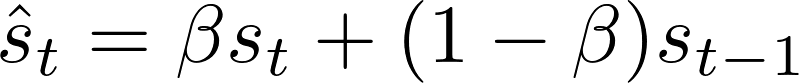
原始事件传递模型

事件源、事件信道、事件宿中又包含许多过程，而这些过程又因具体物联系统的不同而存在很大的差异性。考虑其中的随机噪声问题，假设在事件传递的过程中，子过程i对原始事件的某一项原始数据产生的影响为xi，于是原始事件传递过程中的N个子过程对原始事件中该项原始数据产生的影响为：



由于这些因素很多，子工程N被细分时，N的数目相对较大。子过程对原始数据的影响往往是随机的、人们无法控制的、时有时无、时大时小、时正时负。Yn记录了这些噪声带来的误差之和。由于单个子过程的影响十分微小，这些子过程的噪声往往服从正太分布和均匀分布。根据林德伯格-莱维中心极限定理，当N趋向无穷，服从均匀分布的子过程噪声之和弱收敛于正态分布，而正太分布之和也为正太分布。于是，原始事件中的原始数据适合用正态分布去逼近该变量的分布。因此，对于单传感器的随机噪声，常采用滤波进行处理；对于多传感器，则常用以卡尔曼滤波为基础的方法进行处理。

一种简单的滤波方法，采用相邻两次原始事件的原始数据进行滤波，通过控制参数wpsoffice进行加权求和，得到



当β为0时所有时刻全部等于0时刻。该滤波器就可以简单的削弱随机噪声对结果的影响，使原始事件结果中的突变更少、结果更加连续和平滑。

### 复杂事件处理

复杂事件处理是物联网事件处理的一个重要研究领域，是物联网事件驱动框架的重要组成部分，作为一种实时数据的处理方法受到大数据领域的大量关注。在物联网应用系统中，依据用户需求对事件进行语义提取、语义转换和语义抽象的过程，往往集中在复杂事件处理技术。如果说原始事件处理是以数据为中心的处理技术，那么复杂事件处理技术是以事件为中心的事件处理方法，基于在线事件流对数据进行建模，提高事件作为处理对象的处理效率，也是多个基本事件符合形成语义丰富的复杂事件过程。

复杂事件处理技术可以追溯至RAPIDE[[[1]](#endnote-0)]，该工作提出了复杂事件处理的事件层次结构。在物联网场景中，现实世界的活动可以表示为子活动，这些活动产生、映射到物联网系统，形成事件流和事件云。David Luckham等人在1998年的工作[[[2]](#endnote-1)]中提出了事件分层及分层视图。对复杂事件系统进行层次结构抽象，复杂事件对业务分析建立是啊金层次结构模型，使用复杂处理规则还原活动中分解丢失的信息，利用事件过滤、聚合、模式匹配等方法将抽象程度较低的事件流融合构造出抽象程度较高的、信息密度较大的事件流。

复杂事件处理分为两个层次：

1）事件层：主要功能为事件语言与查询、事件表达与操作；

2）事件流层：主要功能为事件过滤、聚合、模式识别；

本节针对事件处理，所以本小节介绍复杂事件处理中的事件层，下一大节的事件流处理部分介绍复杂事件处理的事件流层技术。

#### 事件语言与查询

复杂事件处理要求对事件进行过滤、聚合等操作，发觉事件的约束，对事件进行关联，提供事件的表达与查询能力。物联网语义技术常采用资源描述框架（RDF）与网络本体语言（OWL）对事件及资源进行描述。

面向传统的传感器网络的非本体语言中，SASE[[[3]](#endnote-2)]语言是具有代表性的事件语言。该语言最初用以对RFID无线传感网络的事件进行复杂事件过滤、事件关联和模式匹配，将相关事件转换到匹配的特定模式上，为外部的监测应用提供服务。通过基于时间的约束和基于事件值的约束，SASE对特定事件进行过滤。SASE主要的语言结构为下：

EVENT <event pattern>

[ WHERE <qualification> ]

[ WITHIN <window> ]

该语言中，event pattern表示事件表达式，由基本事件、复杂事件、和事件操作符组成；WHERE子句通过逻辑判断，对qualification构成的简单的参数化的谓词进行判断，该谓词表达通过关系运算符连接构成；WITHIN子句限定了一个时间窗window，与事件的生命周期相关。

事件语言的一个特点就是支持扩展。任何事件语言都不应该是完全封闭的，可扩展性提供了优化能力。举例而言，SASE事件语言基础上可以进行一定的扩展，比如引入如下语言结构：

[ FROM <stream> ]

[ EVERY <time interval> ]

其中FROM子句提供输入流名称，stream表述输入流数据；EVERY子句用来确定执行周期，time interval表示执行查询的周期。这样的可扩展性使得事件语言在面向复杂任务时，具有优化自身表达能力的任务适应性。

以SASE为代表的传统复杂事件处理，可以看出其特点为基于实体-属性模型，复杂事件处理系统根据用户给出的schema对数据进行理解和分析。比如SASE语言中，被<>包含的部分往往是一系列逻辑运算符与实体的组合，就是典型的schema。

随着语义技术的发展，领域内越来越多的研究采用资源描述框架（RDF）和网络本体语言（OWL）对物联场景进行描述，包括物联场景中的事件。RDF使用SPO三元组表述任何事实，其形式正如本章第一部分物联网数据抽象方法中介绍，这里SPO分别代表（subject，predicate，object）。视subject/object为节点，predicate为边，则RDF构成一张图。RDF图中的节点可以是国际资源标识符IRL、字面值以及空节点。相比实体-属性模型，RDF文档可以更灵活的描述半结构、非结构化的数据，支持高效的融合操作。

RDFS（RDF schema）在RDF进行了扩展，允许用户使用自定义的类和属性，其关于数据模型的集合TBox和包含的事实集合ABox构成了知识库。OWL是一种基于RDF描述的逻辑语言，支持了数据建模和推理。新兴的复杂事件处理，利用RDF文档进行事件表示，使用事件处理本体语言构造本体库对事件语义进行描述，复杂事件处理引擎利用本体描述推断事件所对应的复杂事件处理规则，对事件进行进一步处理。

处理复杂事件的语义技术方面，针对事件描述采用事件处理本体语言和事件处理规则语言。事件处理本体语言使用语义本体结构对事件处理逻辑和整体框架进行描述，事件处理规则语言使用过程话的方法描述事件处理的算法细节。

**事件处理本体语言**，可以基于OWL生命逻辑进行表达。采用语义化事件表示模型，通过本体库提供事件数据的语义信息和领域知识。下面介绍集中描述逻辑：



其中A表示原子概念，P表示原子角色，P-表示简单角色的逆；Ｂ表示基本概念，基本概念可以是一个简单概念或者由遗存关系表示的概念；Ｒ表示基本角色，基本角色可以是一个简单角色或者简单角色的逆。Ｃ表示概念，概念可以是一个基本概念或者基本概念的补集，Ｅ表示角色，角色定义为一个基本角色或者基本角色的补集。其中对描述逻辑进行建模时常使用到的RDF词汇有一下几个：

* rdfs:Class 类在本体模型中表示一类包含相同属性/约束的对象的集合；
* rdf:Property 属性在本体模型中表示类之间、实例之间的二元关系；
* rdf:subClassOf 表示类之间的继承关系，是一种具体的rdf:Property
* rdfs:domain 描述属性的定义域；
* rdfs:range 表示属性的值域；
* rdf:Thing 本体模型中包含所有实例的类；
* rdf:Nothing 定义空集；

通过不限于上述的词汇，可以设计满足描述逻辑所需要的本体模型。

**事件处理规则语言**，可以融合SPARQL的部分特定，提供基于结构化查询语言的事件处理语言。事件处理规则语言常包括以下几部分：

1. CONSTRUCT triple\_template\_def

定义规则输出，三元组模版表达式。

1. FROM stream\_def

FROM语句定义规则依赖的数据流，事件流可以指定特定事件窗口、长度以及包含简单事件过滤规则。可以引入JOIN语句对多个事件流进行关联查询。

1. WHERE condition

定义查询对象的过滤条件。

1. GROUP BY

对查询结果进行分组聚合。可以配合SELECT语句以及聚合函数进行结果聚合。

1. ORDER BY

指定结果的顺序。

1. LIMIT

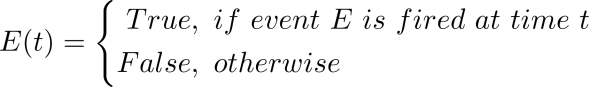
对结果进行限制，一般是数量限制、顺序限制、表达式限制。

其中较为重要的是FROM语句，语句中包含事件流表达式，提供事件类型、规则、模式的匹配过滤。

#### 事件表达与操作

事件表达通常由事件表达式构成。事件表达式是使用事件代数来表示复杂事件与子事件的结合关系的表达方法。物联网场景复杂多样，数据具有时空关联性，导致物联网事件间的关系多态化。事件操作符表达事件之间的关系，体现在事件的时间和逻辑上，因此大致分为两类：一类表达事件的持续事件，一类表达事件的逻辑执行方式。从事件的执行方式，事件相关的操作可分为描述自身状态、描述多个事件之间的关系。事件之间的关系包括时序关系、逻辑关系。

事件表达式的核心在于关系与属性的描述。关系和属性被成为特征时，事件表达式是离散映射，从时域映射到布尔域，采用特征函数的形式化表达如下：



基本事件操作符，功能涵盖描述事件的发生顺序、发生逻辑关系。基本事件操作符的形式是由用户定义的，但其功能往往涵盖以下几种，这里采用一些形式化描述。

1. SEQ(;)

事件E1和E2的顺序发生可以表示为E1;E2，(E1;E2)(t)表示该命题为True。

1. NEG()：用以表示没有检测到事件的发生。
2. AND()：表示事件的共同发生。
3. OR()：表示事件中只有一个事件发生。
4. WITHIN()：事件在某个限定范围发生。

事件聚合函数是对事件的操作，下面介绍一些常用的聚合函数。

1. SUM： 对事件进行求和操作。
2. AVG：对事件在给定维度或属性取平均。
3. COUNT：事件在给定维度或属性进行计数。
4. MAX：事件在给定维度或属性的最大值。
5. MIN：事件在给定维度或属性的最小值。

# 事件流检测与事件共享

「内容规划」

1. 事件流检测
   1. 事件流检测基本问题与形式化描述
      1. 时间乱序问题
      2. 分散来源问题与分布式集群流处理
   2. 事件流检测相关基本技术
   3. 复杂事件流检测模型
   4. 分布式流处理平台
2. 事件共享机制
   1. 查询机制语义模型
   2. 查询规则
   3. 共享机制与层次模型

上节中指出复杂事件处理分为事件层和事件流层两个层次，并介绍了事件层相关技术。结合事件层技术，解决事件形成事件流所带来的挑战，是事件流层技术的主要目的。事件流层的主要任务是事件流检测和事件共享，其中事件流检测常用以应对物联网场景中常出现的复杂事件乱序问题，事件共享解决事件查询中常的重复存储和重复查询问题。

## 事件流检测基本问题

物联网场景中的硬件故障、网络连接障碍、网络拥塞、传输与存储延时等问题，为物联网事件流的感知、存储、转发、执行带来挑战。随着流数据的应用不断扩大，各种应用根据各自的需求可能会处理来自不同数据源的数据。应用所依赖的数据源之间存在复杂的依赖关系，复杂事件处理系统的事件流层也需要针对此进行处理。

由物联场景的特点和物联应用的多元数据依赖特点，物联网复杂事件处理、事件流检测面临的诸多挑战中，事件时间乱序问题和分散来源事件分布式处理组成物联网事件流检测中的双流问题，即时间流问题与空间流问题，只不过这里的空间流可能是实际物理空间也可能是虚拟的数据空间。双流问题由于其不同特点，采用不同的解决思路和处理方法。

1. 时间流问题：时间乱序

时间乱序问题在事件流中的体现，通常是时间戳乱序问题。时间戳乱序可能会引起以下问题：

A. 可能引起异常匹配；

B. 可能因乱序事件的延迟到达而产生错误的事件删除；

C. 可能引起无法判断复杂事件是否发生；

频繁发生的乱序事件会大大降低复杂事件处理和事件流处理的效果，导致无法检测出复杂事件。错误的事件检测往往导致进行错误的执行或无法执行有效操作，从而导致物联网服务的失效。

1. 空间流问题：分散来源

一般复杂事件处理系统接收数量巨大且来源分散的事件流，需要采用分布式集群流处理平台来处理数据。影响运行效率的两个方面主要是事件源的匹配方法和查询的执行策略[[[4]](#endnote-3)][[[5]](#endnote-4)]。事件源匹配问题，需要能够处理数据量大、数据价值低等特点的数据流的匹配技术；查询执行，则需要对查询策略进行优化，以特定的策略与倾向指定查询计划，提高系统运行效率。

## 事件流检测相关基本技术

在在实际的事件流检测系统中，由于通过感知设备到复杂事件的过程中带来乱序性、海量性、强不确定性等特征，海量多源、乱序、不确定事件流是事件流检测中的重点，以适应物理系统的海量数据处理需求。复杂事件检测方法在事件流上的应用，主要在3类事件流上进行，分别为有序事件流、乱序事件流和多概率事件流。有序事件流存在一些共性的检测方法与常用技术，并存在许多针对共性技术的改进工作，以提高反应速度、解决检测时间长、检测效率低的问题。针对物联网这个特定环境下的乱序事件流通用处理方法尚无，目前的研究主要是面向特定任务的优化处理。对多概率、不确定性事件流的方法，主要在有序事件流的共性方法上进行改进。因此，本小节将存在的共性方法作为事件流检测相关的基本技术，加以介绍。

### Hash冲突与链地址法

### NFA

### Petri网

### 匹配树

### 有向图

### 区间时序逻辑

## 复杂事件流检测模型

### 面向乱序修正任务的事件流检测模型

#### 空间回收机制及其驱动机制

在复杂事件的事件流层检测过程中，满足NFA状态的事件，通过链表长期存储在内存中等待与之匹配的事件的到来。但随着检测的进行，大量被存储的中间结果可能导致系统的不稳定。时间窗约束要求只有在某一时间范围内成功复合的事件序列才有可能被输出，所以没有必送对那些过期事件进行长期保存。为了使这些无用数据不占用磁盘空间，使这部分未使用的存储空间真正成为系统的待分配空间，就必须采用空间回收技术将这些不满足条件的事件从存储区删除，使空间得以释放，供系统或其他用户使用。

内存空间需要清理通常在以下4种情况发生时：

1. 获取的原始事件不是触发事件，需要将该事件从缓存区中删除；
2. 子结点的定时机制触发，删除该子结点所在的子链表；
3. 不满足事件约束的新触发子事件所在的子链表删除；
4. 事件序列成功匹配时，子结点所在子链表的删除；

**脉动事件**：常将上面4种情况成为脉动事件，每一种情况的发生成为一个“脉动”。脉动事件的产生与很多因素有关，脉动事件的产生往往伴随对存储事件的哈希链表的修正。

空间回收机制需要一定的规则作为驱动，以驱动对脉动事件的检测。一般的驱动方式分为两种，即时间驱动机制、事件驱动机制。下面对驱动机制进行介绍。

1. **时间驱动机制**

系统根据确定的时间步长进行推进，当达到下一个步长时刻，进行一次更新。在一个时间步长内，不考虑是否有脉动事件发生。一个事件步长内可能没有脉动事件发生，也可能有多个脉动事件发生，其结果都在下一步长时刻到来时获得检测结果。

频率的变化对空间回收机制检测结果具有影响。随着步长的增加，脉动事件检测结果的准确度呈现下降趋势，但检测效率呈现上升趋势。反之，随着步长的缩短，脉动事件检测结果的准确度呈现上升趋势，但检测效率呈现下降趋势。系统的效率和准确度，呈现相互对立的矛盾。

1. **事件驱动机制**

事件驱动机制指由脉动事件产生更新。每个脉动时间的发生，都会让系统进行一次更新。与时间驱动机制相比，不会再出现多个脉动事件重叠的情况，准确度最高。系统中发生脉动事件的频率不确定，利用事件驱动机制可以避免一些不必要的更新，但如果脉动事件发生的频率较低，会降低空间回收机制的检查频率、进而降低其效率。

1. **混合驱动机制**

时间与事件驱动机制有各自的优点，在脉动事件的时间步长较大时，时间驱动可以有效提高系统的效率，而事件驱动则表现出更好的延迟性能。混合驱动机制就是以时间驱动机制为基础，对固定时间步长内发生的事件进行评估。当检测到脉动事件发生时，以脉动事件触发更新；若到达下一步长时刻未检测到脉动事件，则在此步长时刻对事件进行清楚回收内存空间。采用混合驱动，可以在脉动事件发生时进行实时更新，也可以在时间步长取较大时依旧保持准确度。

#### 事件流检测过程

事件之间乱序问题对顺序操作符存在影响，在检测过程中修正乱序问题，常采用SASE方法。该方法主要使用NFA技术和AIS技术，其中NFA为非确定有限自动机，AIS（Active Instance Stack）为主态实例堆栈，二者为基础完成海量数据流上所需事件序列的快速检测，是目前存在较优的检测方法。Hash表结构常用来代替SASE方法中的AIS，存储检测到的中间结果，利用Hash表映射、插入和查找等技术，解决AIS结构在事件流检测中的大量重复查找操作和回溯问题。本小节介绍一种基于Hash技术与NFA技术的、在检测过程中修正乱序问题的过程。具体过程如下。

（1）

问题：只能抄原文

#### 事件流检测算法

利用Hash表结构能够高效存储事件，其O(1)的查找性能以及链地址法的冲突解决能力，可以根据时间戳顺序在任何地方进行插入操作，Hash的唯一性避免了对同一数据的重复存储。

问题：只能抄原文

### 面向多源数据的事件流检测模型

许多物联场景中包含容量大、来源多样化、需要快速相应的数据特点，使得要求系统直接从海量信息流中直接获取符合自身要求的信息。

多源海量数据流的处理，要求更强的复杂事件构造与提取能力，应对在海量数据流中的查询。Petri网常被用来代替在事件流检测中常用的NFA，以从多源数据流中进行原子事件检测。结合Petri网和Hash表的事件流检测方法，主要包括以下四个步骤：

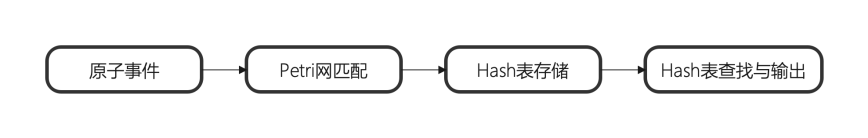


图 Petri网+Hash表的事件流检测步骤

1. 原子事件：这里特指从多源数据流中提取基本事件；
2. Petri网匹配：使用Petri网对原子事件进行匹配；
3. Hash表存储：将匹配原子事件作为中间结果，使用Hash 表结构存储；
4. Hash表查找与输出：通过使用Hash表查询来查询有关事件序列并输出检测结果；

#### 检测原理及过程

## 分布式流处理平台介绍

## 事件共享机制

### 用户级查询共享

### 基于有向无环图的事件共享机制

### 基于参数上下文的事件共享机制

# 时空语义的数据融合

「内容规划」

1. 什么是时空语义数据？
   1. 时空语义概念
   2. 物联网的时空语义特性
   3. 物联网时空信息处理框架
2. 时空语义描述
   1. 时空关系类型
   2. 时空对象与属性描述
3. 时空语义处理技术
   1. 时空语义的融合算法

# 环境上下文感知技术

「内容规划」

1. 什么是环境上下文感知技术？
   1. 上下文感知概念
   2. 上下文感知特征
   3. 物联网上下文信息分类
2. 上下文感知生命周期
   1. 生命周期
   2. CXaaS上下文即服务
3. 上下文推理决策模型
   1. 上下文模型构建框架
   2. 推理与模式识别技术

参考文献

1. [] Luckham D C, Kenney J J, Augustin L M, et al. Specification and analysis of system architecture using Rapide[J]. IEEE Transactions on Software Engineering, 1995, 21(4): 336-354. [↑](#endnote-ref-0)
2. [] Luckham D C, Frasca B. Complex event processing in distributed systems[J]. Computer Systems Laboratory Technical Report CSL-TR-98-754. Stanford University, Stanford, 1998, 28: 16. [↑](#endnote-ref-1)
3. [] 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. 2006: 407-418. [↑](#endnote-ref-2)
4. [] Prasan Roy,S. Seshadri,S. Sudarshan,Siddhesh Bhobe. Efficient and extensible algorithms for multi query optimization[J]. ACM SIGMOD Record,2000,29(2). [↑](#endnote-ref-3)
5. [] Rimma V. Nehme,Karen Works,Chuan Lei,Elke A. Rundensteiner,Elisa Bertino. Multi-route query processing and optimization[J]. Journal of Computer and System Sciences,2013,79(3). [↑](#endnote-ref-4)