IoT 下 CapBAC 规则语义表示及其时间间隔粗糙性分析

梁晓艳[1,2]，杜瑞忠[1,2]

（1. 河北大学网络空间安全与计算机学院，河北 保定 071002；
2. 河北省高可信信息系统重点实验室，河北 保定 071002）

摘 要：为突破 IoT CapBAC 规则时间间隔粗糙性分析瓶颈，提出了规则语义表示模型 IoTACS 以及粗糙性分析算法 IoTRA。在 IoTACS 中，提出 IoT CapBAC 规则领域的概念结构、基本概念的定量数据属性表示以及规则语义树模型；扩展 Time 本体的时间关系，给出其定量逻辑表示。作为 IoTRA 算法的理论基础，证明了粗糙性是造成不一致的根本原因，给出其逻辑表示。案例研究表明，IoTRA 算法有助于提升授权时间的准确性，IoTACS 模型比形式化语义模型具有更高的可读性；仿真实验表明，与 IoTC^2 算法相比，IoTRA 算法响应时间显著减少。

关键词：物联网；基于权能的访问控制；规则语义；粗糙性；语义网

中图分类号：TP393.08

文献标识码：A

DOI: 10.11959/j.issn.1000−436x.2021075

Semantic representation and interval roughness analysis of IoT CapBAC rules

LIANG Xiaoyan[1,2], DU Ruizhong[1,2]

1. School of Cyber Security and Computer, Hebei University, Baoding 071002, China
2. Key Lab on High Trusted Information System of Hebei Province, Baoding 071002, China

Abstract: To break the bottleneck of interval roughness analysis, a rule semantic model IoTACS was proposed, and IoTRA algorithm was given. In IoTACS, a conceptual structure of the IoT CapBAC rule domain was proposed, the quantitative data properties of the basic concepts were given, then the semantic tree model of rule semantics was proposed, the time relations of the time ontology were extended and its quantitative logical representation was given. As the theoretical basis, it was proved that the root cause of interval inconsistency was roughness, and its logical formalization of the roughness was given. The case study shows that IoTRA can help to improve the accuracy of the authorized interval, and the IoTACS model have higher readability than the formal model. The comparison experiments with the IoTC^2 algorithm on the inconsistency analysis show that the response time of the IoTRA algorithm is more significantly reduced.

Keywords: IoT, CapBAC, rule semantics, roughness, semantic Web

1 引言

物联网（IoT, Internet of things）将包括传感器、移动电话、交通工具、电视机、智能家电等的大量设备连接在一起相互通信、交换信息，已深入渗透到人们的日常工作和生活。预计到 2022 年，将有 297 亿台设备连入网络，其中 181 亿台将通过物联网互联[1]。IoT 在智慧城市、智慧运输、智慧网格中获得广泛应用的同时，其隐私和安全性也变得非常重要[2]，访问控制是其中的重要机制[3]。IoT 访

收稿日期：2021−01−14；**修回日期**：2021−03−23

通信作者：杜瑞忠，drzh@hbu.edu.cn

基金项目：国家自然科学基金资助项目（No.61572170）；河北省自然科学基金重点资助项目（No. F2019201290）；河北省自然科学基金资助项目（No. F2018201153, No. F2018201197）

Foundation Items: The National Natural Science Foundation of China (No.61572170), The Key Project of Natural Science Foundation of Hebei Province (No.F2019201290), The Natural Science Foundation of Hebei Province (No.F2018201153, No.F2018201197)

问控制分为基于角色的访问控制（RBAC, role-based access control）[4]、基于属性的访问控制（ABAC, attribute-based access control）[5]和基于权能的访问控制（CapBAC, capability-based access control）[6-7]。其中，CapBAC 由于其轻量级、分布式、动态性和可扩展性、满足最小权限原则的优点，在物联网环境中获得了广泛应用[8-9]。

权能令牌和委托是 CapBAC 的核心。令牌包括主体、客体、操作和时间间隔，其本质上是一种访问控制规则，由访问者自己存放。委托机制使主体可以签发令牌将自己的权限委托给其他主体，这使 CapBAC 机制具有分布式的优点[9]，主体之间的委托图关系中，一个主体的权限可能来自多个委托主体的授权[6]，本文称之为多委托机制。

访问控制规则的准确性关系着 IoT 的安全性和隐私。CapBAC 规则对于主体、客体、操作均采用最小权限原则声明，但是时间间隔却由于依赖委托者的主观判断而导致其相对于准确的时间间隔过大或过小，即具有一定的粗糙性。

时间间隔的粗糙性可能被恶意用户利用。恶意用户可利用多委托机制，向多个委托者提出授权请求，从而获得多个授权者授予的对同一访问授权的令牌，其中，每个授权者都根据自己的理解限制时间间隔，该用户便可获得粗糙性最大的时间间隔授权，从而对 IoT 应用的安全性和隐私造成威胁。

时间间隔的粗糙性是由授权者对权限的时间限制认识不精准导致的，由于思维本身的粗糙性，授权者难以发现自身配置规则的粗糙性。

针对以上问题，本文在实际工作中发现，如果多委托者对同一访问的时间间隔不一致，那么至少一个时间间隔是粗糙的，即规则语义中的时间间隔是粗糙近似表示。因此，本文从时间间隔一致性分析的角度，分析来自多委托的同一访问的时间间隔的差异，进而分析时间间隔的粗糙性，并将分析结果通过高可读性的方式显示给管理员，从而使管理员可以发现规则中的粗糙性。

本文给出一种 IoT 下 CapBAC 规则语义表示模型 IoTACS（IoT access control semantics）及其时间间隔粗糙性分析（IoTRA, IoT CapBAC interval roughness analysis）算法。首先，给出方法的应用场景和系统框架；然后，基于语义网理论建立 IoT CapBAC 规则语义表示模型；最后，界定时间间隔粗糙性，并基于时间间隔一致性给出其形式化表示，基于 Jena API（Jena application programming infterface）和 JavaCC（Java complier compiler） API 给出时间间隔粗糙性分析算法。本文的贡献如下。

1) 本文提出了一种高可读性的 IoT CapBAC 规则语义表示模型，给出 IoT CapBAC 规则的领域概念结构（包括领域的概念和关系），可作为 IoT CapBAC 规则领域的术语集；用概念类的具体实例来描述规则的指称语义，因此与以属性条件为语义组分的形式化方法[10]相比具有更高的可读性。

2) 本文指出了 IoT CapBAC 规则时间间隔的粗糙性问题，并从多委托规则不一致对比角度给出该概念的形式界定。该概念用于刻画规则语义的一种不确定性问题，为后续该问题的研究提供基础。

3) 本文拓展了 OWL-Time（Web ontology language-time）[11]时间间隔关系及其形式表示，可为其他时间本体研究提供参考。

2 相关工作

2.1 IoT CapBAC 规则的安全问题

委托 CapBAC 因为其轻量级、动态性和分布式的特点，被 IoT@Work 广泛应用于 IoT 访问控制场景中[12]。近年来，CapBAC 与区块链结合成为研究热点。CapBAC 方案具有最小权限原则的优点，即每个主机必须使用最小权限来完成访问；此外，该方案允许主体将访问权限委托给其他主体。在最新的研究中，为使委托具有灵活性，委托关系的结构由树变为图，即允许多个主体将自己的权限委托给一个主体，这种情况下，一个主体会拥有多个来自不同委托主体的令牌[6-7]。

由于最小权限原则使主体、操作和客体都是最小元素，不存在包含关系，因此该方案下的访问不存在由于主体、操作和客体的包含关系而导致的不一致问题，但规则中的时间间隔可能存在不一致。例如，在智能家居这类需要人工分配访问权限时间的场景中，由于不同的人对访问控制权限应赋时间的认识不同，对同样访问的多委托授权者签发的令牌中的访问控制规则的时间可能不一致，而现有文献尚未关注该问题。

已有文献研究 IoT CapBAC 方案中的安全问题。Nakamura 等[13-14]指出了 CapBAC 方案中可以导致信息泄露的漏洞，即某用户具有的权能限定在某个时间间隔内，这期间用户将数据存放于另一个在其他时间可访问的数据内，这样就可以使用户在

非法时间访问该数据。但文献[13-14]关注的是 CapBAC 系统流程上的漏洞，而本文关注的是访问控制规则的漏洞。

2.2　语义表示及分析

当前 IoT 的语义表示和分析研究主要关注 IoT 设备配置的表示和分析[10,15]。Farooq 等[10]提出了一种基于 Prolog 的 IoT 配置安全属性声明及配置与安全属性冲突检测的方法。Vannucchi 等[15]提出了一种基于 SMT 的对“事件−条件−动作”IoT 配置规则形式化验证的技术。上述文献采用基于形式化的建模方式，无法表现访问控制意图的高层实体。

本体技术不但可以表现底层的谓词逻辑，也能表现结构化、易理解的领域知识。CapBAC 访问控制规则语义本质上是访问事件，基于本体的 CapBAC 语义表示本质上属于事件本体表示研究范畴。现有事件本体模型主要有 Event-Model-F[16]、LODSE（linking open descriptions of social event）[17]、TEO（time event ontology）[18]等。其中，Event-Model-F 描述时间、空间、物体和人有关的事件本体；LODSE 和 TEO 描述社会事件。现有时间本体模型主要有 OWL 描述的 Time Ontology[11,19]、安全领域的时间本体[20]和 GFO Time Ontology[21]。其中，Time Ontology 提供了时间属性，并且描述了时间点和时间区间的顺序。OWL-Time 时间本体描述了时间相关概念，它用 OWL 的 xsd:dataTime 数据类型定义了 Intervals 和 Time Point，这些类型支持数据的比较，因此也可以表示特定时间点之间顺序关系，但 OWL-Time 没有声明怎样推理 interval 和 point 之间的量化关系。GFO-Time 用一阶逻辑公理表示时间，是时间的代数。此外，TEO 也对 GFO 做了改进，加入了在医学语义中经常出现的“Period Time”的语义表示。

当前时间本体没有给出时间先后关系的形式化表示，本文研究扩展时间的关系并给出形式化表示。当前事件本体中，均认为事件是状态的变化，并且定义的是针对具体领域的事件，本节不关注事件的状态变化，而仅仅关注事件的动作，并且定义 IoT CapBAC 特定领域的事件组分。另外，当前的 IoT 设备配置表示和分析主要基于形式化表示和分析，但该方法一直存在分析效率瓶颈[22]。

综上，事件本体和时间本体可读性较好，为建立新的 IoT CapBAC 语义模型和探索快速的语义分析算法提供了基础，但针对 IoT CapBAC 领域的事件语义及时间关系语义，以及时间间隔粗糙性分析算法尚需要进一步研究。

3　系统框架

系统框架如图 1 所示。令牌签发机制由物联网实体、云服务和部署在区块链上的智能合约构成。其中，物联网实体指用户、智能设备等，智能合约通过云服务接口彼此访问，并通过云服务存放数据。

CapBAC 访问控制令牌签发机制如图 1 中的步骤 1)～步骤 6)、步骤 10)～步骤 12)所示。本文在上面的流程中加入粗糙性检测方法，如图1中的步骤7)～步骤 9)所示，其中关键为步骤 8)和步骤 9)。具体步骤如下。

1) 云服务部署区块链和实现 CapBAC 授权功能的智能合约。

2) 智能合约产生的令牌等数据存放于云服务上。

3) 物联网的实体均向物联网注册，获取 VID 和区块链账号。

4) 实体通过云服务向委托者申请授权。

5) 委托者给出授权的访问控制规则，调用智能合约，提出令牌签发请求。

6) 当前区块链中的智能合约验证请求的有效性。

7) 查询实体的所有令牌。

8) 生成步骤 7)中返回令牌的访问控制规则的语义。

9) 分析语义中的时间间隔的粗糙性。

10) 智能合约根据请求是否有效，签发令牌或拒绝。

11) 返回令牌签发结果给云服务。

12) 返回令牌签发结果给实体。

Nakamura 等[6-7]给出了步骤 7)的实现，粗糙性检测方法的关键在于步骤 8)和步骤 9)，即本文所提的 IoTRA 算法。该算法的理论基础是 CapBAC 规则的语义模型（记为 IoTACS）和时间间隔粗糙性表示方法。其中，前者由 IoT 访问控制术语集（记为 CBACT，描述领域术语集，包括高层意图的概念结构和基本概念的数据属性等）、规则句义（记为 $CBACA_1$，是用该术语集描述的规则的句义）、规则句义隐含的时间关系语义（记为 $CBACA_2$）三部分组成；后者由时间间隔粗糙性（记为 $CBACA_3$）

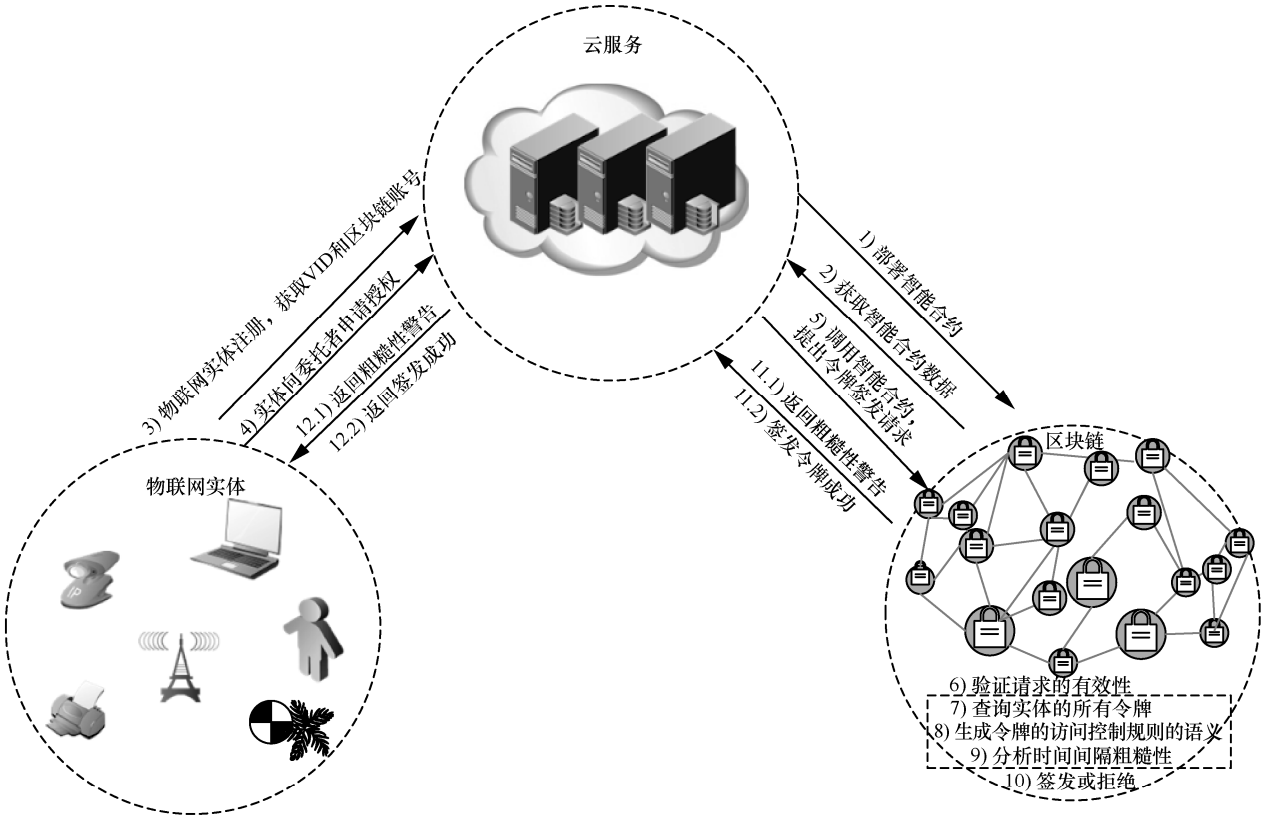

图 1 系统框架

组成。

4 CapBAC 语义的术语集——CBACT

本节用描述逻辑给出CapBAC的术语集的形式化表示。

IoT CapBAC 规则的句法为

$$\text{rule}:=\text{entity}_1,\text{entity}_2,\text{protocol},\text{flag},\text{year}_1,\text{month}_1,\text{day}_1,\text{hour}_1,\text{minute}_1,\text{second}_1,\text{year}_2,\text{month}_2,\text{day}_2,\text{hour}_2,\text{minute}_2,\text{second}_2 \quad (1)$$

其中，rule 为非终结符，其余词法变量为终结符。

按照易于管理员理解的概念及结构，本文定义描述式(1)所示规则语义所需的术语及关系，如图 2 所示。

为描述式(1)的前 4 项语义，本文定义“实体”（Entity）、“操作”（Operate）2 个概念，以及“操作是”（oprIs）、“主体是”（subIs）、“客体是”（objIs）3 个对象关系来界定“访问”概念；用前 4 项的值作为数据属性界定“实体”和“操作”概念。

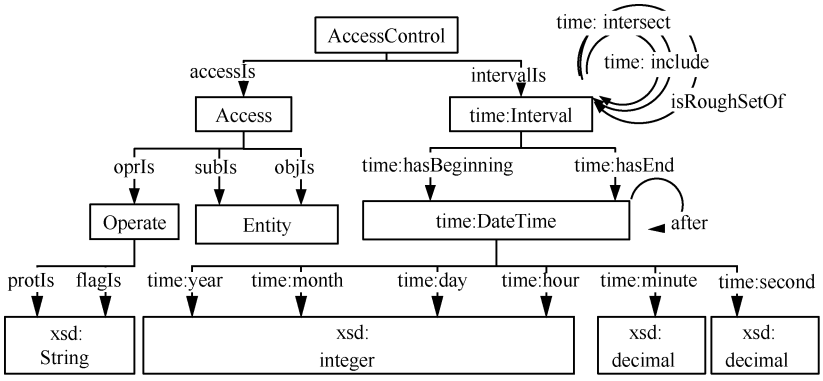

图 2 CapBAC 规则的术语集——CBACT

OWL-Time 本体中的“时间间隔（time:Interval）”概念表示式(1)中后 12 项的语义；该概念由该本体中的“日期时间”（time:DataTime）概念，以及“time:hasBeginning”和“time:hasEnd”这 2 个对象关系来界定；“日期时间”概念由这些属性值作为数据属性的值来界定。

为表示粗糙性，本文定义“粗糙集”（isRoughSetOf）对象关系表示“时间间隔”之间的粗糙性关系；定义“后”（after）关系表示时间之间的顺序关系，进而界定“包含”（include）、“交叉”（intersect）2 种不一致关系。

整个规则语义由上述概念界定的“访问控制”（AccessControl）概念来描述。

综上，术语集中共包含 6 个概念、11 个对象关系以及 8 个数据属性关系。

1) “访问”的概念结构

首先，借助属性值给出基本概念的定义，在此基础上，进一步自底向上给出“访问”的概念结构的形式化表示。

定义 1 操作。操作表示主体通过物联网内的协议对客体完成的动作。在 IoT 环境中，操作由 IoT 中的协议类型和该协议中的标志位唯一标识。如 IoT 中的 MQTT 协议的 PUBLISH、SUBSCRIBE 标志位分别表示写和读操作，或者 HTTP 的 PUT 或 GET 标志位分别表示写和读操作。操作的形式化定义为

$$\begin{aligned}&\mathrm{Operate}\sqsubseteq \mathrm{protocolIs.xsd:String}\\&\mathrm{Operate}\sqsubseteq \mathrm{flagIs.xsd:String}\end{aligned}\tag{2}$$

其中，xsd:String 是 XSD（XML schema definition）中内置的数据类型。下文带“xsd”命名空间的数据类型也是如此，不再赘述。

定义 2 实体。实体是发起操作的主体和执行操作的客体的集合。在 IoT CapBAC 中，主体和客体都是最小粒度，因此不进一步用属性来界定该概念。

定义 3 访问。访问是主体对客体执行的一次操作。该概念由主体、客体、操作唯一标识，因此在定义 1、定义 2 的基础上，引入 3 个对象关系即可形式化表示该概念。

$$\begin{aligned}&\mathrm{Access}\sqsubseteq \mathrm{subIs.Entity}\\&\mathrm{Access}\sqsubseteq \mathrm{objIs.Entity}\\&\mathrm{Access}\sqsubseteq \mathrm{oprIs.Operate}\end{aligned}\tag{3}$$

2) “时间间隔”的概念结构

OWL-Time 本体通过首先定义“日期时间”，在此基础上给出了“时间间隔”的定义，本文直接引入了这种概念结构，并进一步给出其形式化表示。

定义 4 日期时间。日期时间表示日历时钟系统中的一个时刻。对日历时钟系统的不同元素用单独的值构成日期和时间的描述，引入 OWL-Time 的数据属性：年（year）、月（month）、日（day）、小时（hour）、分钟（minute）、秒（second），由 OWL 的 XSD 数据类型（如 Decimal 和 Integer）表示。另外，为后续描述时间间隔之间的关系，还扩展定义了对象属性 after，它表示日期时间之间的“在……后”顺序关系。

$$\begin{aligned}&\mathrm{DateTime}\sqsubseteq \mathrm{time:year.xsd:Integer}\\&\mathrm{DateTime}\sqsubseteq \mathrm{time:month.xsd:Integer}\\&\mathrm{DateTime}\sqsubseteq \mathrm{time:day.xsd:Integer}\\&\mathrm{DateTime}\sqsubseteq \mathrm{time:hour.xsd:Integer}\\&\mathrm{DateTime}\sqsubseteq \mathrm{time:minute.xsd:Decimal}\\&\mathrm{DateTime}\sqsubseteq \mathrm{time:second.xsd:Decimal}\\&\mathrm{DateTime}\sqsubseteq \mathrm{after.DateTime}\end{aligned}\tag{4}$$

定义 5 时间间隔。时间间隔是表示时间日期范围的时间。它由开始的日期时间和结束的日期时间唯一指定；此外，为表示粗糙性，本文引入了时间间隔之间的包含（include）、交叉（intersect）和粗糙集（isRoughSetOf）关系。形式化表示如下

$$\begin{aligned}&\mathrm{Interval}\sqsubseteq \mathrm{hasBeginning.DateTime}\\&\mathrm{Interval}\sqsubseteq \mathrm{hasEnd.DateTime}\\&\mathrm{Interval}\sqsubseteq \mathrm{include.Interval}\\&\mathrm{Interval}\sqsubseteq \mathrm{intersect.Interval}\\&\mathrm{Interval}\sqsubseteq \mathrm{isRoughSetOf.Interval}\end{aligned}\tag{5}$$

3) “访问控制”的概念结构

“访问控制”是允许的访问的声明。它由访问和日期时间间隔唯一确定。形式化表示如下

$$\begin{aligned}&\mathrm{AccessControl}\sqsubseteq \mathrm{accessIs.Access}\\&\mathrm{AccessControl}\sqsubseteq \mathrm{intervalIs.Interval}\end{aligned}\tag{6}$$

5 CapBAC 规则语义——$CBACA_1$ 和 $CBACA_2$

本节首先基于 CapBAC 规则的句法结构给出其句义（$CBACA_1$）表示方法，然后给出句义中隐含的时间关系语义（$CBACA_2$）。

5.1 规则句义——$CBACA_1$

对于满足式(1)句法的 CapBAC 规则，其句义可

看作 CBACT 中概念的实例以及实例间关系的集合。其形式化描述如下

$$\mathrm{CBACA}_1=\{(x:C)\}\cup\{((y,z):r)\} \tag{7}$$

其中，x 和 y 为实例，C 为 x 所属的概念，z 为实例或者值，r 为对象关系或者数据属性关系。

图 3 表示语义树中的 16 个叶子节点中的带“_V”后缀的变量为式(1)中的 16 个终结符词法变量的取值。带“_S”后缀的变量是由带“_V”后缀的变量所生成的新变量，它们用来表示新生成的高层意图对象。

这种句义表示方法与形式化的语义表示方法相比，不但能够表示语义属性的真值表达式，而且能够表示其背后体现的高层访问控制意图的实体组分和实体之间的关系。因而，该句义表示方法可用于表示时间间隔的粗糙性。

5.2 隐含的时间关系语义——CBACA_2

5.1 节给出了句义表示方法。这些语义生成了各个基本概念的实例和依据概念界定关系生成的非基本概念的实例。但这些实例之间隐含的时间关系尚未表示。本节给出生成这些关系的定理。

设 2 个 DateTime 类型的实例 d1 和 d2 的 year、month、day、hour、minute、second 的属性值分别为 y_1、y_2、m_1、m_2、d_1、d_2, h_1、h_2、m_1' 、m_2'、 s_1、s_2。

定理 1 日期时间的“后”关系的逻辑表示为

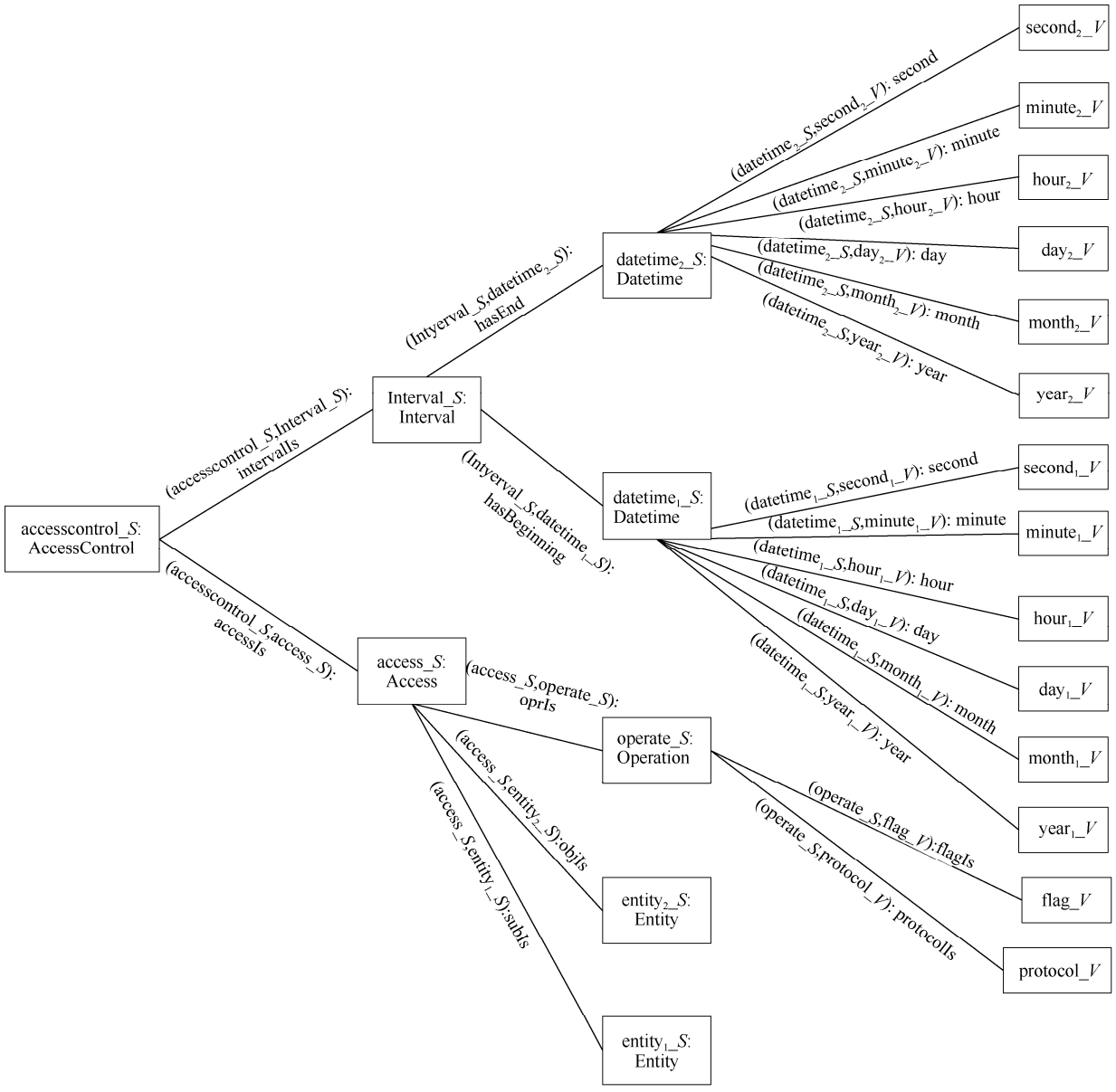

图 3 CapBAC 规则句义的语义树表示

$$\begin{gathered}
\left(\left\langle y_1,y_2\right\rangle:\text{swrl}:\text{lessThan}\right)\vee\\
\left(\left\langle y_1,y_2\right\rangle:\text{swrl}:\text{equal},\left\langle m_1,m_2\right\rangle:\text{swrl}:\text{lessThan}\right)\vee\\
\left(\left\langle y_1,y_2\right\rangle:\text{swrl}:\text{equal},\left\langle m_1,m_2\right\rangle:\text{swrl}:\text{equal},\left\langle d_1,d_2\right\rangle:\text{swrl}:\text{lessThan}\right)\vee\\
\left(\left\langle y_1,y_2\right\rangle:\text{swrl}:\text{equal},\left\langle m_1,m_2\right\rangle:\text{swrl}:\text{equal},\left\langle d_1,d_2\right\rangle:\text{swrl}:\text{equal},\left\langle h_1,h_2\right\rangle:\text{swrl}:\text{lessThan}\right)\vee\\
\left(\left\langle y_1,y_2\right\rangle:\text{swrl}:\text{equal},\left\langle m_1,m_2\right\rangle:\text{swrl}:\text{equal},\left\langle d_1,d_2\right\rangle:\text{swrl}:\text{equal},\left\langle h_1,h_2\right\rangle:\text{swrl}:\text{equal},\left\langle m_1',m_2'\right\rangle:\text{swrl}:\text{lessThan}\right)\vee\\
\frac{\left(\left\langle y_1,y_2\right\rangle:\text{swrl}:\text{equal},\left\langle m_1,m_2\right\rangle:\text{swrl}:\text{equal},\left\langle d_1,d_2\right\rangle:\text{swrl}:\text{equal},\left\langle h_1,h_2\right\rangle:\text{swrl}:\text{equal},\left\langle m_1',m_2'\right\rangle:\text{swrl}:\text{equal},\left\langle s_1,s_2\right\rangle:\text{swrl}:\text{lessThan}\right)}{\left\langle x_2,x_1\right\rangle:\text{after}}
\end{gathered}\tag{8}$$

式(8)中的横线上面是逻辑条件，下面是逻辑结论。条件中以“swrl”为命名空间的谓词为 SWRL（semantic Web rule language）谓词，其中，swrl:lessThan、swrl:equal 分别表示小于、等于数值关系谓词。

通过上述逻辑表达式，可以通过推理日期之间的 after 关系，进而推理出时间间隔之间的包含（include）和交叉（intersect）2 种不一致关系。

设 x_1、x_2 为时间间隔的实例，y_1、y_2、y_3 和 y_4 为日期时间的实例，则有定理 2 和定理 3。

定理 2　Interval 的逻辑表示满足以下条件。

$$\frac{\begin{gathered}\left\langle x_1,y_1\right\rangle:\text{time}:\text{hasBeginning},\left\langle x_1,y_2\right\rangle:\text{time}:\text{hasEnd},\\ \left\langle x_2,y_3\right\rangle:\text{time}:\text{hasBeginning},\left\langle x_2,y_4\right\rangle:\text{time}:\text{hasEnd},\\ \left\langle y_1,y_3\right\rangle:\text{after},\left\langle y_4,y_2\right\rangle:\text{after}\end{gathered}}{\left\langle x_2,x_1\right\rangle:\text{include}}\tag{9}$$

定理 3　Interval 的交叉关系的逻辑表示为

$$\frac{\begin{gathered}\left\langle x_1,y_1\right\rangle:\text{time}:\text{hasBeginning},\left\langle x_1,y_2\right\rangle:\text{time}:\text{hasEnd},\\ \left\langle x_2,y_3\right\rangle:\text{time}:\text{hasBeginning},\left\langle x_2,y_4\right\rangle:\text{time}:\text{hasEnd},\\ \left\langle y_1,y_3\right\rangle:\text{after},\left\langle y_2,y_4\right\rangle:\text{after}\end{gathered}}{\left\langle x_1,x_2\right\rangle:\text{intersect}}\tag{10}$$

由于定理 1、定理 2 和定理 3 较易理解，因而其证明过程不再赘述。

CBACA_2 中的元素为式(8)～式(10)所示推理规则的后件部分。

6　时间间隔粗糙性——CBACA_3

通过对时间间隔不一致现象的分析，本文发现以下规律。

定理 4　如果 2 个规则对同样访问授权的时间间隔不一致，则其意图至少是其中一个时间间隔的粗糙集。

证明　1) 若 2 个规则的访问相同，则该访问的时间间隔意图（即应允许的最准确的时间间隔）应是唯一的。

2) 若 2 个时间间隔不一致，则 2 个时间间隔不等价。

3) 若 2 个时间间隔都是对同一个访问的时间间隔意图的精准表示，则由 1)可推知，2 个时间间隔均与意图等价，即 2 个时间间隔等价。

3)的结论与 2)矛盾，因此 3)的假设不成立。

证毕。

依据定理 4，可以将意图定义为 2 个时间间隔的粗糙集。当其中一个时间间隔为意图的精确表示时，可以看作粗糙集中粗糙度为 0 的极限情况。

当有 2 个访问 A 和 A'，对同样的访问 a，有包含或者交叉的 2 个时间间隔 x 和 y 时，则存在时间间隔意图 z 是 x 的粗糙集，也是 y 的粗糙集。形式化表示为式(11)和式(12)。

$$\begin{gathered}\frac{\begin{gathered}\left\langle A,a\right\rangle:\text{AccessIs},\left\langle A',a\right\rangle:\text{AccessIs},\\ \left\langle A,x\right\rangle:\text{intervalIs},\left\langle A',y\right\rangle:\text{intervalIs},\\ \left\langle x,y\right\rangle:\text{include}\end{gathered}}{z:\text{Interval}}\\ \left\langle z,x\right\rangle:\text{isRoughSetOf},\left\langle z,y\right\rangle:\text{isRoughSetOf}\end{gathered}\tag{11}$$

$$\begin{gathered}\frac{\begin{gathered}\left\langle A,a\right\rangle:\text{AccessIs},\left\langle A',a\right\rangle:\text{AccessIs},\\ \left\langle A,x\right\rangle:\text{intervalIs},\left\langle A',y\right\rangle:\text{intervalIs},\\ \left\langle x,y\right\rangle:\text{intersect}\end{gathered}}{z:\text{Interval}}\\ \left\langle z,x\right\rangle:\text{isRoughSetOf},\left\langle z,y\right\rangle:\text{isRoughSetOf}\end{gathered}\tag{12}$$

式(11)和式(12)所示推理规则的后件部分为新产生的语义，记为 CBACA_3，即时间间隔的粗糙性。该语义的含义为创建新的时间间隔实例 z，它是 x 和 y 的粗糙集。

7 时间间隔粗糙性分析算法

时间间隔粗糙性分析算法伪代码如算法 1 所示。

算法 1 IoTRA 算法

输入 Rules={rule$_i$|0≤i≤t}, IoTACS. //Rules 是待分析的规则集合；IoTACS 是 TDB 形式存储的 OWL 语义，//其内容仅包含 CBACT（如第 4 节所示，由人工创建完成），其他组分由算法依次生成

输出 CBACA$_3$ // CBACA$_3$ 是规则语义中的时间间隔粗糙性（如第 6 节所示）

设 IoTACS、CBACT、CBACA_1^i、$\text{CBACA}_2^{\text{after}}$、$\text{CBACA}_2^{\text{include}}$、$\text{CBACA}_2^{\text{intersect}}$、$\text{CBACA}_2$、$\text{CBACA}_3$ 为各种 OWL 语义的存储变量

for i=0,1,2,⋯,t do

if （rule$_i$ 的语法如式(1)所示）

IoTACS ←IoTACS∪ CBACA_1^i; //利用如图 3 所示的翻译文法对 rule$_i$ 进行语法分析生成规则的句义 CBACA_1^i（如第 5.1 节所示）

end if

end

$\text{CBACA}_2^{\text{after}}$ ← afterTemporalRelationReason();// 推理 DateTime 之间的“after”时间关系（推理规则如式(8)所示）

$\text{CBACA}_2^{\text{include}}$ ← includeTemporalRelationReason(); //推理 Interval 之间的“include”关系（推理规则如式(9)所示）

$\text{CBACA}_2^{\text{intersect}}$ ← intersectTemporalRelationReason();//推理 Interval 之间的“intersect”关系（推理规则如式(10)所示）

CBACA_2← $\text{CBACA}_2^{\text{after}}\cup\text{CBACA}_2^{\text{include}}\cup\text{CBACA}_2^{\text{intersect}}$; //$\text{CBACA}_2$ 为时间关系推理出来的新语义（如第 5.2 节所示）

IoTACS← CBACT ∪CBACA_2; // 将 CBACA_2 存入 IoTACS 中

CBACA_3 ←roughnessReason();// CBACA_3 为依据式(11)、式(12)推理得到的时间间隔的粗糙性

return: CBACA_3

首先借助 Protégé 工具由人工建立术语体系（即第 4 节所示的 CBACT），然后在此基础上，对规则进行语法分析，依照图 3 所示的语义树生成句义（即第 5.1 节所示的 CBACA_1），接着根据式(8)～式(10)推理出时间关系（即第 5.2 节所示的 CBACA_2），根据式(11)～式(12)推理出粗糙关系（即第 6 节所示的 CBACA_3），最后得到语义 IoTACS= CBACT∪$\text{CBACA}_1\cup\text{CBACA}_2\cup\text{CBACA}_3$。

算法 1 中，生成语义部分的时间复杂度与规则的条数 m 相关，IOTACS 中的 CBACA 的语义规模（即断言的数目）与 m 成正比，因而 CBACA_1 语义生成的复杂度为 $O(n)$。语义推理部分，CBACA_2 和 CBACA_3 均使用 SPARQL 语言在 Jena TDB 数据库上完成推理，由于该数据库按照主体、客体、属性三元组分别建立索引，因而，设语义节点个数为 n，推理的复杂度即为查询的复杂度，每一个查询项的复杂度为 $O(\lg n)$，查询的项数为式(8)～式(12)的前件所示，最大为 6 项，因而推理的复杂度为 $O((\lg n)^6)$。

8 案例研究与实验验证

本文给出的 IoT CapBAC 规则语义表示模型，及其基础上的时间间隔一致性分析算法，通过分析来自多委托的同一访问的时间间隔的差异，进而分析出时间间隔的粗糙性，并将分析结果通过高可读性的方式显示给管理员，从而使管理员可以发现规则中的粗糙性。该方法的特点如下。

1) IoTRA 算法的有效性：可以通过一致性对比分析出时间间隔的粗糙性。

2) IoTACS 模型的高可读性：与形式化方法相比，该方法的语义具有更高的可读性，便于管理员理解；可读性采用由于易用而应用最广泛的文本可读性 [23]来度量，计算式为

$$\text{Grade Level} = 0.4\left(\text{Average Sentence Length} + \text{Number of hard words}\right) \tag{13}$$

其中， Average Sentence Length 表示句子包含的语词数；Number of hard words 表示难理解的词，在英语语言中指超过 2 个音节的词，本文将该参数定义为不指代高层对象或关系的词。

3) IoTRA 算法的高效性：与形式化方法相比，该方法的分析速度更快。本节通过仿真较大规模数据，对比该方法和形式化方法的分析时间，说明该方法的高效性。

8.1 案例研究

本节采用智能家居场景中的案例进行研究，但在其他采用 CapBAC 多委托机制的场景中，如家居、

数字医疗、车联网、智能电网、工业与公共基础设施等采用 CapBAC 机制实现访问控制的场景中，也同样适用。

本节案例场景内有 3 个用户：Alice、Bob 和 Eve。Alice 与 Bob 是家人，因此 Alice 将智能家居访问全部委托给 Bob。2020 年 11 月 15 日 Alice 出差一天，她将智能家居中的灌溉服务的访问权在 10:00—12:00 这个时段赋予邻居 Eve。如果 Eve 想多获得灌溉系统的访问权，他可以再向 Bob 申请权限；Bob 平时不灌溉，因此依据 Alice 的出差时间，将灌溉系统的访问时间设置为 8:00—22:00。

首先，用 Protégé 工具建立术语集，如图 4 所示。图 4(a)～图 4(c)分别为用 Protégé 创建的概念集、对象属性关系集和数据属性关系集。这些术语集存储为 OWL 文件。

(a) 概念集　　(b) 对象属性关系集　　(c) 数据属性关系集

图 4　用 Protégé 人工构建的术语

然后，在上述场景中，每次签发 Token 前运行 IoTRA 算法，该算法将本次授权的 Token 与之前的 Token 对比，生成规则的语义、推理隐含的时间关系，并进一步推理出时间间隔粗糙性。运行 IoTRA 算法，IoTACS 的全部语义用 Protégé 显示，如图 5 所示。

1) IoTRA 算法的有效性讨论

图 5 中，“TimeIntention_1”为新生成的 Interval 类型的实例，“IoTACT#2020.1.15_8:00_22:00”和“IoTACT#2020.1.15_10:00_12:00”为其粗糙集。

这样，授权者 Bob 收到该报告后可知授权的时间间隔的粗糙度为 0.86，由于粗糙度太高，因此依据 interval1 调整授权时间为 10:00—12:00，从而有助于提升授权时间的准确性。

该案例研究说明，在家居、数字医疗、车联网、智能电网、工业与公共基础设施等采用 CapBAC 多委托机制实现访问控制的场景中，使用本文算法可以通过不同委托者在授权的时间间隔上的对比发现其中至少一个时间间隔的粗糙性，从而有助于委托者准确调整时间间隔而降低风险。

2) IoTACS 与形式化模型相比较的高可读性

本节对比 IoTACS 与 Farooq 等[10]提出的形式化模型来表示本节案例语义的可读性。

首先，IoTACS 表示的一条语义如图 5 中的 Token_1 所示，其中 8 个实体词、7 个关系词，共计 15 个词，这些语词都是高层对象或关系，均不是难懂词，用式(13)计算 IoTACS 表示的句义的可读性为 0.4×(15+0)=6。

采用文献[10]模型，用 Prolog 表示同样的语义则为

$$\begin{aligned}&\text{user}(\text{UserA})\wedge\text{operate}(\text{HTTP\_GET})\wedge\\&\text{object}(\text{IrrigationEquipment})\wedge\\&\text{year}_1(2020)\wedge\text{month}_1(11)\wedge\\&\text{day}_1(15)\wedge\text{hour}_1(8)\wedge\text{minute1}(0)\wedge\\&\text{second}_1(0)\wedge\text{year}_2(2020)\wedge\text{month}_2(11)\wedge\text{day}_2(15)\wedge\\&\text{hour}_2(22)\wedge\text{minute}_2(0)\wedge\text{second}_2(0)\end{aligned}$$

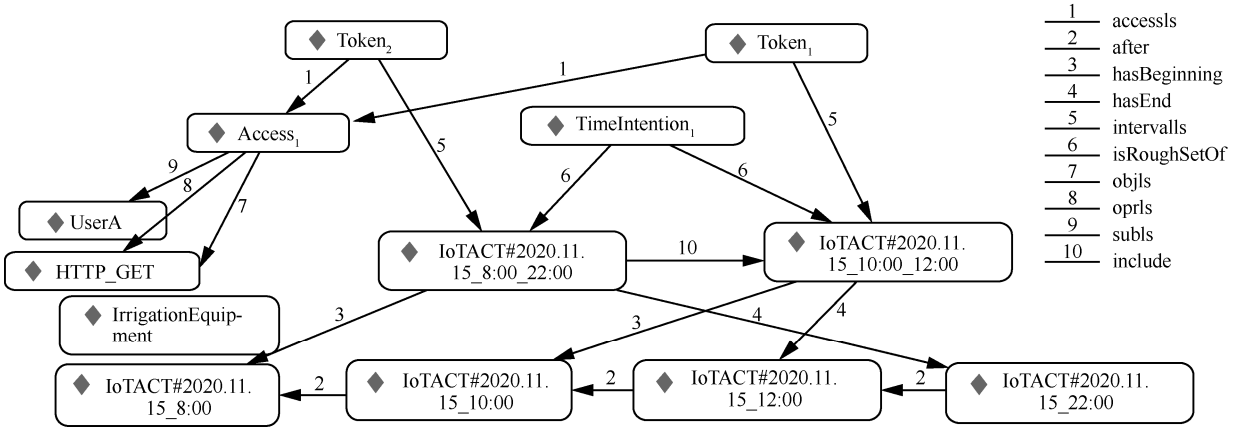

图 5　用 Protégé 显示 IoTACS 的全部语义

根据文献[10]表示的语义用到 30 个语词，包括 15 个对象和 15 个谓词。其中，前 3 个对象和前 3 个谓词是易懂词，其余均为难懂词，可读性为 0.4×(30+24)=21.6。

由此可见，IoTAC 模型的可读性更好。语义的可读性在规则分析中对于辅助管理员理解、诊断问题有重要影响[24]。因此，本文提出的模型更能辅助管理员理解当前的访问控制规则语义及其中的粗糙性。

8.2 分析算法的性能评估

选择与本文最相近的 IoTC^2 算法[10]，来对比在不一致分析上的性能。

本文人工构造了 6 组不同规模的访问控制规则，分别包含 500 条、1 000 条、1 500 条、2 000 条、2 500 条和 3 000 条访问控制规则，并生成了对应的 Prolog 语义和本文的语义。为考虑存在不一致时的分析效率，构造了 3%不一致规则的数据。

本节实验用 IoTRA 算法和 IoTC^2 算法分别析其中的不一致，记录各算法的响应时间。由于 IoTC^2 算法只能分析不一致，不能分析粗糙性，因此本节实验仅对比 2 种算法的不一致分析性能，没有对比粗糙性分析性能。

实验环境如下：操作系统为 Win7，内存为 16 GB, CPU 为 Intel(R) Core(TM)i5-320M 2.50 GHz。响应时间对比如表 1 所示。

表 1 分析不一致性的响应时间对比

规则规模/条	响应时间/s			
	无不一致规则		存在 3%不一致规则	
	IoTRA	IoTC^2	IoTRA	IoTC^2
500	0.5	16.2	0.556	23.6
1 000	0.63	36.71	0.815	80.7
1 500	0.7	59.79	1.037	161.23
2 000	0.87	88.9	1.37	280.19
2 500	0.93	122.62	1.562	444.08
3 000	1.18	158.95	1.987	651.88

从表 1 可以看出，IoTRA 算法的响应时间在不同的不一致规则比例中都显著优于 IoTC^2 算法。这是因为，IoTRA 算法分析不一致的复杂度为 $O((\lg n)^k)$，其中 n 为节点个数，k 为推理规则（如式(8)～式(10)所示）前件条件的最大项数。IoTC^2 算法基于 Prolog API，是一种采用深度有限遍历的方法，依据深度遍历的复杂度，其推理语义的复杂度为 $O((m^2)^l)$，其中 m 为节点个数，l 为推理规则项数。语义节点数 n 和 m 为同量级变量，k 和 l 为同量级变量，因此 IoTRA 算法的时间复杂度小于 IoTC^2 算法。

实验结果表明，本文所提 IoTRA 算法对规则分析的响应性能显著优于 IoTC^2 算法，更适用于大规模的 IoT 访问控制规则分析。

9 结束语

本文为突破 IoT CapBAC 规则委托中的多授权规则时间粗糙性分析的瓶颈，提出规则语义中的时间间隔语义是时间间隔意图的粗糙集近似是导致该问题的根本原因，进而提出时间间隔粗糙性分析方法；给出了 IoT 下 CapBAC 规则的语义模型 IoTAC，形式化界定时间间隔的不一致及粗糙性，并基于语义网开源 Java 架构——Apache Jena API，给出时间间隔粗糙性分析算法 IoTRA。案例研究表明，本文所提算法可在签发访问控制规则前分析出时间间隔粗糙集，有助于管理员及时调整准确的授权时间，并且与形式化模型相比，IoTAC 具有更好的可读性。与 IoTC^2 算法的对比实验说明，IoTRA 算法的响应时间显著减少。本文所提算法为分析 IoT 访问控制规则语义的粗糙性提供了一种思路。将来的研究工作是分析 IoT 访问控制规则语义中的其他不确定性。

参考文献：

[1] Ericsson. Erisson mobility report[R]. 2021.

[2] 周悦芝, 张迪. 近端云计算: 后云计算时代的机遇与挑战[J]. 计算机学报, 2019, 42(4): 677-700.
ZHOU Y Z, ZHANG D. Near-end cloud computing: opportunities and challenges in the post-cloud computing era[J]. Chinese Journal of Computers, 2019, 42(4): 677-700.

[3] 邱宇, 王持, 齐开悦, 等. 智慧健康研究综述: 从云端到边缘的系统[J]. 计算机研究与发展, 2020, 57(1): 53-73.
QIU Y, WANG C, QI K Y, et al. A survey of smart health: system design from the cloud to the edge[J]. Journal of Computer Research and Development, 2020, 57(1): 53-73.

[4] THAKARE A, LEE E, KUMAR A, et al. PARBAC: priority-attribute-based RBAC model for azure IoT cloud[J]. IEEE Internet of Things Journal, 2020, 7(4): 2890-2900.

[5] 杜瑞忠, 刘妍, 田俊峰. 物联网中基于智能合约的访问控制方法[J]. 计算机研究与发展, 2019, 56(10): 2287-2298.
DU R Z, LIU Y, TIAN J F. An access control method using smart contract for Internet of Things[J]. Journal of Computer Research and Development, 2019, 56(10): 2287-2298.

[6] NAKAMURA Y, ZHANG Y, SASABE M, et al. Exploiting smart

contracts for capability-based access control in the Internet of Things[J]. Sensors, 2020, 20(6): 1793.

[7] NAKAMURA Y, ZHANG Y, SASABE M, et al. Capability-based access control for the Internet of things: an Ethereum blockchain-based scheme[C]//2019 IEEE Global Communications Conference. Piscataway: IEEE Press, 2019: 1-6.

[8] TAPAS N, LONGO F, MERLINO G, et al. Experimenting with smart contracts for access control and delegation in IoT[J]. Future Generation Computer Systems, 2020, 111: 324-338.

[9] 张玉清, 周威, 彭安妮. 物联网安全综述[J]. 计算机研究与发展, 2017, 54(10): 2130-2143.
ZHANG Y Q, ZHOU W, PENG A N. Survey of Internet of things security[J]. Journal of Computer Research and Development, 2017, 54(10): 2130-2143.

[10] FAROOQ A, Al-SHARE E, MOYER T. IoTC$^{2:}$ a formal method approach for detecting conflicts in large scale IoT systems[C]//2019 IFIP/IEEE Symposium on Integrated Network and Service Management. Piscataway: IEEE Press, 2019:442-447.

[11] W3C. Time ontology in OWL[R]. 2020.

[12] GUSMEROLI S, PICCIONE S, ROTONDI D. IoT@Work automation middleware system design and architecture[C]//Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation. Piscataway: IEEE Press, 2012: 1-8.

[13] NAKAMURA S, ENOKIDO T, TAKIZAWA M. Time-based legality of information flow in the capability-based access control model for the Internet of Things[J]. Concurrency and Computation: Practice and Experience, 2020: e5944.

[14] NAKAMURA S, ENOKIDO T, TAKIZAWA M. Information flow control based on the CapBAC (capability-based access control) model in the IoT[J]. International Journal of Mobile Computing and Multimedia Communications, 2019, 10(4): 13-25.

[15] VANNUCCHI C, DIAMANTI M, MAZZANTE G, et al. Symbolic verification of event-condition-action rules in intelligent environments[J]. Journal of Reliable Intelligent Environments, 2017, 3(2): 117-130.

[16] SCHERP A, FRANZ T, SAATHOFF C. F——a model of events based on the foundational ontology dolce+DNS ultralight[C]// Proceedings of the Fifth International Conference on Knowledge Capture. Piscataway: IEEE Press, 2009:137-144.

[17] RODRIGUES M, ROCHA SILVA R, BERNARDINO J. Linking open descriptions of social events (LODSE): a new ontology for social event classification[J]. Information, 2018, 9(7): 164.

[18] LI F, DU J C, HE Y Q, et al. Time event ontology (TEO): to support semantic representation and reasoning of complex temporal relations of clinical events[J]. Journal of the American Medical Informatics Association, 2020, 27(7): 1046-1056.

[19] ERMOLAYEV V, BATSAKIS S, KEVERLE N. Ontologies of time: review and trends[J]. International Journal of Computer Science & Applications, 2014, 11(3): 57-115.

[20] SIKOS L F. OWL ontologies in cybersecurity: conceptual modeling of cyber-knowledge[M]. Berlin: Springer, 2019.

[21] BAUMANN R, LOEBE F, HERRE H. Axiomatic theories of the ontology of time in GFO[J]. Applied Ontology, 2014, 9(3): 171-215.

[22] ELMALLAH E S, GOUDA M G. Hardness of firewall analysis[J]. IEEE Transactions on Dependable and Secure Computing, 2017, 14(3): 339-349.

[23] ZAMANIAN M, HEYDARI P. Readability of texts: state of the art[J]. Theory and Practice in Language Studies, 2012, 2(1): 43-53.

[24] VORONKOV A, IWAYA L H, MARTUCCI L A, et al. Systematic literature review on usability of firewall configuration[J]. ACM Computing Surveys, 2018, 50(6): 1-35.

[作者简介]

梁晓艳（1981− ），女，河北衡水人，博士，河北大学讲师，主要研究方向为网络信息安全、网络防御规则语义分析等。

杜瑞忠（1975− ），男，河北献县人，博士，河北大学教授、博士生导师，主要研究方向为可信计算、信息安全等。