基于 OWL 本体和 SWRL 的事件检测的研究与实现①

Research and Implementation of Event Detection Based on OWL and SWRL

吕 律 (广东商学院 数学与计算科学学院 广东 广州 510320)

摘 要: 根据时间和空间信息进行事件检测是危机管理的一个重要问题。通过 OWL 本体和 SWRL 规则构造事件，事件检测对应于知识库中的冲突(conflict)。通过对冲突的推理，找出引发事件的原因。最后通过 JAVA 实现了一个完整的实例来说明如何使用本文方法进行事件检测。

关键词: 本体 SWRL 事件检测(event detection)

随着语义网的发展，本体已经开始被应用于危机管理中。但是研究者更多的是侧重在上层本体(upper ontologies)的构建[1,2]。由于危机管理中包含了大量的时间和空间的信息，文献[3]表明在现有的描述逻辑 SHION(D)的基础上加入时空信息，其推理将变成不可判定(undecidable)。基于此，本题提出了一种将本体(OWL)和 SWRL 规则相结合的方法，进行时空推理。在文献[1,2]中，本体主要是用于对知识的定义，而没有进行推理。而描述逻辑推理机(reasoner)的智能主要来自对不一致性的检测(consistency checking)，目前的推理机主要是找出冲突(conflict)。本文在此基础上，提出一种分析冲突原因的方法。基于这种方法，用户能较容易的发现引发事件的原因(本文中事件检测对应于知识库中的冲突)。

1 本体OWL和SWRL规则

SHION(D)是 OWL－DL 所对应的描述逻辑，它包括角色(roles)：原子(atomic)角色，R-(逆(inverse)角色);概念(concept):原子(atomic)概念，$\neg C$,$(C \cap D)$, $(C \cup D)$, $\forall R.C$, $\exists R.C$, $(\leq n \text{ R.C})$,$(\geq n \text{ R.C})$, $\{i_1,\ldots,i_n\}$。这里角色实际是一种二元关系，例如(“小张”，学生，广东商学院)，这里学生就是一个二元关系，即角色，广东商学院是一个概念。SHION(D)还包括：TBox 公理(axiom)：$C \subseteq D$，$C \equiv D$ (这里 C 和 D 是一般性概念(general concepts[5])，而不是原子概念)；RBox 公理：$R \subseteq S$，Trans(R)(R 和 S 是原子角色(atomic roles))。基于这些定义，我们可以找到 3 种关系：(1)概念之间的关系，记作 T1(例如 $C1 \subseteq C2$); (2)角色和概念之间的领域(domain)/范围(range)的关系，记作 T2(例如, isHardWorking domain Person); (3)角色之间的关系,记作 T3(例如 $R1 \subseteq R2$)。因为角色在 RBox 中是原子角，所以 SHION(D)不能表述:$R_1(?x,?y) \cap R_2(?y, ?z) \rightarrow R_3(?x,?z)$,所以需要引入 SWRL 规则去表述复杂的角色关系。当把 SWRL 应用到 OWL 时，一个问题就是 OWL 是基于开放世界(open world)的假设，而 SWRL 是基于封闭世界(closed world)。下面我们首先讨论开放和封闭世界假设，再讨论如何结合 OWL 和 SWRL 进行时间和空间的推理(reasoning)。

1.1 开放，封闭世界假设

封闭世界假设将基于现有知识无法判定是真的知识，都当作是假的。而开放世界假设则认为这种知识不一定是假的。例如：(1)MemberOf(?m,?ti)∧TrustedInstitute(?ti) ∧ File(?f) → Access(?m,?f).) ；(2)MemberOf(?m,?ti)∧TrustedInstitute(?ti)∧HasCertificate(?m)∧File(?f)→Access(?m,?f)。这两条规则表述的是，(1)当一个人来自可信的机构时，他可以

① 收稿时间:2009-01-19

访问这个文件，(2)当一个人来自不可信的机构，但是他持有有效证件时，也可以访问这个文件。当这 2 条规则，作用于以下事实(fact)：TrustedInstitute(GDCC), MemberOf(lulu,GDCC), MemberOf(Tim, GUDD), HasCertificate(Tim), MemberOf(Marry, GUDD). File(f1). 基于开放世界假设，将得到:Acc- ess(lulu), Access(Tim),因为没有事实表明Marry 具有有效证件，所以 Access(Marry)是未知的。而在封闭世界中，得出的结论: Access(lulu), Access(Tim), Access(Marry)。所以本文解决封闭，开放世界问题的方法就是，将封闭世界的角色进行标注。在这个例子中我们引入一个 SWRL 标签，将“Access”标注为封闭，则在推理是，当角色为封闭世界时，则按封闭世界进行推理，没有进行标注的则按开放世界进行处理。

1.2 时间，空间推理(reasoning)

RCC(Region Connection Calculus)是基于一阶逻辑的关于空间概念和关系的公理[4]。 本文采用 RCC8，即如图 1 所示 8 种关系，EQ (Equal), DC (Disconnected), EC (Externally Connected), PO (Partial Overlap), TPP (Tangential Proper Part), NTPP (Non-Tangential Proper Part)，其中 TPP 和 NTPP 的逆关系为 TPPI 和 NTPPI。为了检测知识库中是否存在冲突(conflict), 文献[4]提出了关于 RCC8 的组合表(composition table)。因为文章篇幅，图 2 给出的是 RCC5，即 5 种关系(PP, PPI, PO, EQ, DR)，RCC8 的组合表请参考文献[4]。

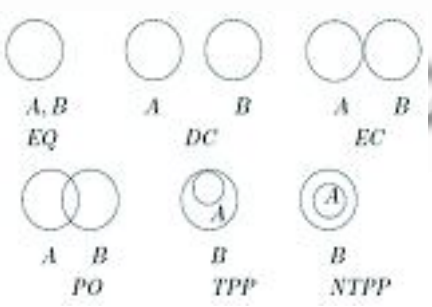

图 1　RCC8 关系

在文献[3]中，作者尝试将 RCC 应用于描述逻辑，但是发现对于复杂的关系 S○T⊆R，这种关系是不可判定(undecidable)的，而 RCC 的组合表正是这种的关系。基于此，本文采用规则，而不是角色关系表示组合表，例如对于图 2 中 RCC5 可以定义如下规则，EQ (?a, ?b) ∧ DR(?b, ?c) → DR (?a, ?c)。其核心思想是在知识库的 TBox 和 ABox 之上，实现一个基于 RCC8 的规则推理，以实现对空间的推理。

○	DR(a,b)	PO(a,b)	EQ(a,b)	PPI(a,b)	PP(a,b)
DR(b,c)	*	DR PO PPI	DR	DR PO PPI	DR
PO(b,c)	DR PO PP	*	PO	PO PPI	DR PO PP
EQ(b,c)	DR	PO	EQ	PPI	PP
PP(b,c)	DR PO PP	PO PP	PP	PO EQ PP PPI	PP
PPI(b,c)	DR	DR PO PPI	PPI	PPI	*

图 2　RCC5 组合表

对时间的推理，本文主要是比较时间发生时间的先后关系。在 SWRL 中定义了 6 种表示比较关系的 build-insswrlb:equal,swrlb:notEqual,swrlb:lessThan,swrlb:lessThanOrEqual,swrlb:greaterThan，swrlb:greaterThanOrEqual. 例如 swrlb:lessThan (?time,xs:time("12:00:00"))表示 12 点之前。

2　基于冲突的推理

上一节讨论了如何结合 OWL 与 SWRL。当用 OWL 和 SWRL 进行时空推理时，一个重要的问题就是如何处理冲突(conflict)，因为描述逻辑推理机的智能是基于对不一致性(即冲突)的检验。[5]定义了 SHOIN(D) 中的两种类型的冲突：(1) $\{C(x),\neg C(x)\}\subseteq ABox$，x 是概念 C 的一个实例(instance)，C 和$\neg C$ 称为 conflict concept[5]。例如 a 是企鹅，企鹅是鸟，鸟会飞，企鹅不会飞，于是可以得出 a 会飞和 a 不会飞的冲突概念。(2) $\{(\leq nR)(x),(>mR)(x)\}\subseteq ABox$，且 n<m。因此我们可以构造以下的规则： hasSmoke(?room) ∧ temperature(?temp) ∧ swrlb:greaterThan (?temp, 70)→Alarm(?room). 这条规则表示，当房间里有烟，并且房间温度在 70 摄氏度以上时，发出警报。假如，房间 315 原来没有警报的(¬Alarm(room-315))，现在传感器发现，该房间有烟(hasSmoke(room-315))，并且温度是 80 摄氏度(temperature(80))。则上面的规则将产生新的事实¬ Alarm(room-315)，这个新的事实(fact)和旧的知识 Alarm (room-315)发生冲突。描述逻辑推理机将自动发现

这一冲突。基于这些事实和规则，可构造出图 3。从图 3，我们可以很容易地发现，事情发生的原因。图 4 是更一般的情况。正如我们前面所提到的，本文通过知识库中的冲突来监测事件。当一个冲突发生了，我们需要自动地找出导致冲突的相关知识，并找出这些知识之间的联系(如图 4 所示)。当我们构造了这样的树型图，用户就能很容易的发现导致事件发生的原因。下面我们将先讨论知识库不包含规则的情况，再讨论包含规则的情况。

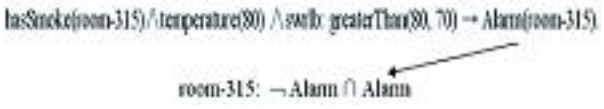

图 3 冲突

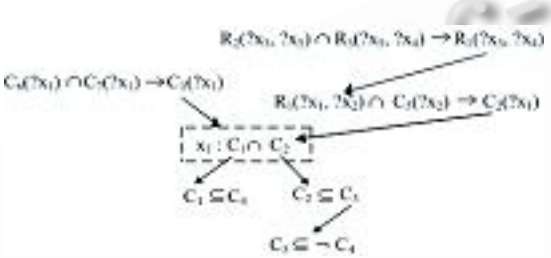

图 4 知识树

2.1 MUPS 树

这一小节先讨论知识库中不包含规则的情况。前面介绍了描述逻辑中的两种冲突，当一个概念(concept)包含冲突时这个概念叫做 unsatisfiable concept。要构造和冲突相关的知识树，首先要找到与冲突相关的知识。文献[6]引入了与冲突相关知识的最小的集合 MUPS。

定义 1. (MUPS)[6],TBox $T' \subseteq T$，C 是 T 中的一个概念，当 C 在 T'中是不可满足的(unsatisfiable)，而在 T'的任意一个子 TBox 中都是可满足的，则称 T'是概念 C 在 TBox 中的 MUPS。MUPS 的实质就是一个集合，当这个集合所包含的任一一个公理被删除时，概念 C 将变成可满足的。

例 1：1) SARS ⊆∃CausedBy. Coronavirus; 2)BirdFlu≡ ∃ CausedBy.AvianInfluenzaVirus; 3)BirdFlu ⊆ ¬ SARS;4)Coronavirus ⊆ ∃ CausedBy.AvianInfluenz aVirus;5)Transitive (CausedBy)

例 1 是 SARS(unsatisfiable concept)的 MUPS，它包含 5 条公理(axioms)。根据定义 1 当例 1 中的任一条公理被移走的之后，SARS 就不再是 unsatisfiable concept。要找出错误，必须要分析 MUPS，为了方便分析，我们提出了一种树型的结构(MUPS 树)。

定义 2. (根节点)，设 UC 是 MUPS(m1)的 unsatisfiable concept，GC1 和 GC2 是一般性的概念(general concept[5])，则根节点(ax(UC))是满足以下条件的公理(axiom[5])：

(1) 若 $UC \subseteq GC1 \in m1$ ， $GC2 \subseteq \neg UC \in m1$ ，则 $UC \subseteq GC1 \cap \neg GC2$。

(2)若 $UC \subseteq GC1 \in m1$，$GC2 \equiv \neg UC \in m1$，则把 ≡做为 ⊆ 处理，$UC \subseteq GC1 \cap \neg GC2$

定义 3. 当满足以下条件时，称公理 ax1 依赖公理 ax2，记作 AD(ax1, ax2)

(1)ax1 和 ax2 都是 T1(见第 1 节中的定义)，GC1 是 ax1 的右边，同时 GC1 是 ax2 的左边；

(2)ax1 是 T1，ax2 是 T2，R 是 ax1 和 ax2 中的角色；

(3)ax1 是 T2，ax2 是 T1，C 是 ax1 和 ax2 中的概念。

定义 4. MUPS 树以 ax(UC)为根节点，子节点(child)和父节点(parent)满足 AD(parent, child)，叶子节点包含 conflict concept 的树。

根据定义 2，3，4 例 1 所对应的 MUPS 树如图 5 所示。

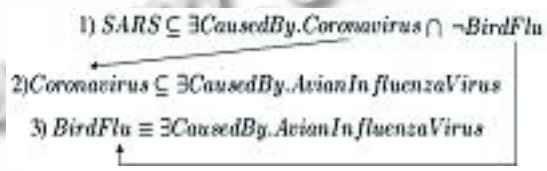

图 5 MUPS 树

2.2 基于规则的 MUPS 树

2.1 节中 MUPS 树不包含规则，当考虑规则时，因为规则中只包含概念和角色的实例(instance)，两个实例之间通过并连接，所以当一个 MUPS 树包含 OWL 和 SWRL 时，有以下 2 种可能的冲突：(1)概念实例冲突，如图 4；(2)角色实例冲突，如图 6。在图 4 中，我们仍然把 x1: C1∩C2 作为根，并且把 MUPS 树分成上下两个部分。当概念冲突出现在规则的前提时，MUPS 树只有下部分。图 4 显示的是概念冲突发生在规则的结论部分。这里要说明一点，根据

SHOIQ(D)的定义，R1 ⊆ ¬R2，R1 ⊆R2∪R3 和 R1 ⊆R2∩R3 是不被允许的。

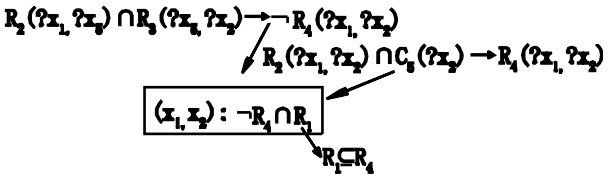

图 6　角色实例冲突

构造包含规则的 MUPS 树的核心思想是，首先构造只有 OWL 的 MUPS 树，然后通过向后搜索(backward search)对规则中的概念和角色实例进行查找，从而构造树的上部分。

2.3 根概念，衍生概念

上面，我们讨论了构造一个 MUPS 树的情况。当存在多个 MUPS 时，有的 MUPS 依赖其他的 MUPS，这里我们采用文献[7]的定义。

定义 5(Root Class)[7]，当一个 unsatisfiable concept 不依赖(不包含)另一个 unsatisfiable concept，则称这个概念为 Root Class。

定义 6(Derived Class)[7]，当一个 unsatisfiable concept 依赖(包含)另一个 unsatisfiable concept，则称这个概念为 Derived Class。，

根据定义 5，6，Derived Class 是从 Root Class 中衍生出来的，所以 Derived Class 包含了 Root Class 中的错误。当有多个 MUPS 存在时，先构造根概念的 MUPS 树，当根概念的问题解决了，派生概念的问题也就解决了。

3　实现

这一节，通过一个完整的例子来说明本文的方法。本文方法用 JAVA 实现，一致性检测(consistency checking)采用的是 Pellet[8]。

例 2：一个汽油站(gs1)在路(road1)上(NTPP(gs1,road1), NTPP 参见图 1)，和 road1 相连的还有路 road2, road3,road4(EC(road1, road2), EC(road1, road3), EC(road1, road4), EC 参见图 1)。开始时，这些路都没有阻塞(blocked(road1), blocked(road2),blocked(road3),blocked(road4))。当以下情况发生时，道路将发生阻塞，(1) broken (?r) → blocked (?r)(当路面被破坏时，道路将发生阻塞); (2) blocked(?r1) ∧ EC(?r1,?r2) → will-be-blocked (?r2)(当一条路发生阻塞时，和它相连的道路也将发生阻塞)，这里我们定义 will-be-blocked blocked。距离 gs1 大约 80m 处有一个酒店(h1), DC (h1, gas1), distance(80)。和 h1 相关的规则如下：(1) explode (?o) → fire(?o). (当物体发生爆炸时，将引发火灾).(2)fire(?o) ∧ building-T1(?b) ∧ DC(?b, ?o) ∧ distance(?d) ∧ swrlb:lessThan(?d,100) → Alarm(?b).(当有火灾发生在距离类型 T1 的建筑(building-T1) 100m 之内时，将产生警报)。T1 建筑定义如下：building-T1 ≡ building ∩ ∃ hasPeople, hotel building ∩ ∃ hasPeople ∩ ∃ hasRestaurant。基于这些事实，描述逻辑推理机(reasoner)将得出：hotel ⊆ building-T1。开始时没有报警：¬Alarm(h1)。假设一天，gs1 发生爆炸，接着破坏了路面，explode(gs1) ∧ NTPP(gs1, road1) → broken(road1)。当有路人报告了汽油站发生爆炸，并破坏了路面这一事实，根据本文方法将生成图 7。图 7 中长方型(■)表示感念和角色，三角形(△)表示并和与关系，圆形(○)表示(→), (⊆), (≡)。图中的正方型(□)表示发生冲突的概念，角色实例。

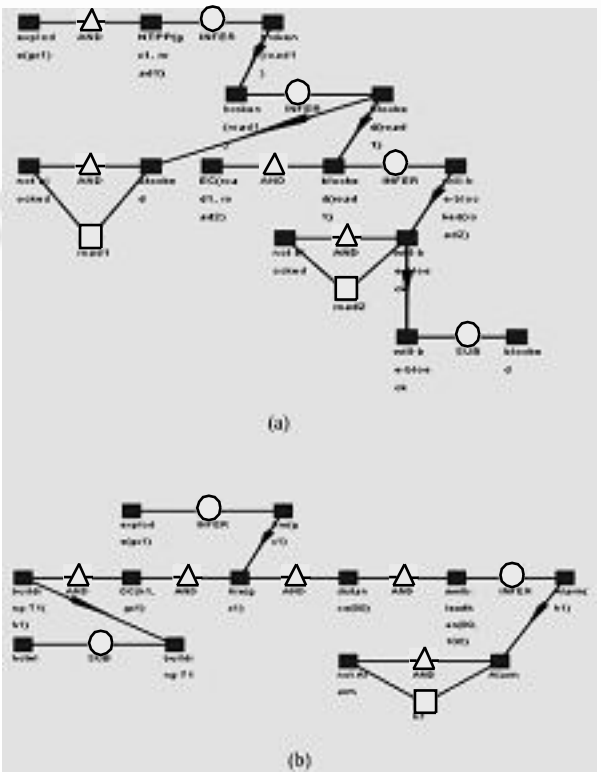

图 7　实例

图 7(a)第一行：explode (gs1) ∧NTPP(gs1, road1) →broken(road1)，第二行：broken (road1)→ blocked (road1)，概念实例 broken (road1)通过箭头连接。第 3 行的左边是 blocked ∩ blocked，所以发生冲突是 road1。第 3 行的右边：EC(road1, road2) ∧ blocked(road1)→ will-be-blocked 满(road2)。第四行：blocked ∩ will-be-blocked，所以发生冲突的是 road2。基于图 7(a)我们将得知，gs1 发生爆炸将导致 road1 堵塞，而 road2 将发生堵塞是 road1 的衍生事件。图 7(b)，第一行: explode (gs1) → fire (gs1),第二行:building-T1(h1)∧DC (h1,gs1)∧fire(gs1)∧distance(80)∧swrlb: lessThan(80,100)→Alarm (h1)。building-T1(h1)之下显示 hotel⊆building- T1。红色正方形 h1 表示 ¬Alarm ∩ Alarm 的冲突。基于图 7(b)，我们将知道导致 h1 发生警报的是 gs1 的爆炸。因为文章篇幅所限，图 7 中没有给出 road3 和 road4 的情况，他们和 road2 的情况是相似的。

4 小结

本文提出了一种结合 OWL 和 SWRL 进行时间和空间推理的方法，并给出一种分析 OWL 和 SWRL 中冲突的树形图。基于这种的树形图，用户能够清晰的发现导致事件产生的原因。最后通过一个实例来说明如何应用本文的方法。

参考文献

1 Matheus CJ, Kokar MM, Baclawski K. (2003) ‘A Core Ontology for Situa tion Awareness.’ Proc. Sixth Intern. Conf. on Information Fusion FUSION2003.

2 Jeong S, Kim HG. ‘Design of Semantically Interoperable Adverse Event Reporting Framework’. Proc. of Asian Semantic Web Conf. 2006.

3 Wessel M. Qualitative Spatial Reasoning with the ALCIRCC Family-First Results and Unanswered Questions Tech. Rep. FBI-HH-M-324/03, FB Informatik, Univ. Hamburg.

4 Randell DA, Cui Z, Cohn AG. A Spatial Logic based on Regions and Connections. Nebel B, Rich C, Swartout W: Principles of Knowledge Representation and Reasoning. Morgan Kaufmann, San Mateo, CA.

5 Baader F, Calvanese D, McGuinness DL, Nardi D, Patel-Schneider PF. eds: The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, 2003.

6 Schlobach S, Huang Z, Cornet R, Van Harmelen F. Debugging Incoherent Terminologies. Journal of Automated Reasoning in Press, 2007.

7 Aditya Kalyanpur,2006 Debugging and repair of OWL ontologies [Ph.D Dissertation]. University of Maryland.

8 PELLET. http://clarkparsia.com/pellet/