一种可降低结构性变化的本体演化算法

陆晨阳[1,2],袁　嵩[1,2],高　峰[1,2,3],顾进广[1,2]
(1. 武汉科技大学 计算机科学与技术学院,湖北 武汉 430065;
2. 湖北省智能信息处理与实时工业系统重点实验室,湖北 武汉 430065;
3. 中冶南方连铸技术工程有限责任公司,湖北 武汉 430223)

摘　要:在 Web 语义环境中,如何降低本体在不断演化的过程中的结构性变化是一项具有挑战性的任务。针对这一问题,提出了一种本体演化算法,旨在降低本体演化过程中的结构性变化,从而提高本体的稳定性和可维护性。与传统的本体演化方法主要关注实体间关系不同,该文在计算本体演化代价时在实体频率和实体间关系频率的基础上综合考虑了实体之间的概念关系,从而实现了对实体影响的更细粒度分析。此外,还提出新的附加演化策略,以便在演化过程中选取最优的演化策略,保留更多复杂的语义关系以实现最小结构性变化。实验结果表明,提出的演化方法在处理涉及复杂语义关系的本体演化时表现出色,有效地降低了演化代价并提高了本体演化前后的结构相似度。这些研究成果在 Web 语义环境中对本体进行演化和更新方面具有重要意义,为本体工程领域提供了有益的改进方向和实践经验。

关键词:本体;本体演化;语义关系;演化策略;结构性变化
中图分类号:TP311. 1　　**文献标识码**:A　　**文章编号**:1673-629X(2024)06-0045-08
doi:10. 20165/j. cnki. ISSN1673-629X. 2024. 0090

An Ontology Evolution Algorithm for Reducing Structural Changes

LU Chen-yang[1,2], YUAN Song[1,2], GAO Feng[1,2,3], GU Jin-guang[1,2]
(1. School of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan 430065, China;
2. Hubei Province Key Laboratory of Intelligent Information Processing and Real-time Industrial, Wuhan 430065, China;
3. WISDRI CCTEC Engineering Co., Ltd., Wuhan 430223, China)

Abstract: In the Web semantic environment, how to reduce the structural changes of ontologies during their continuous evolution is challenging. Aiming at the problem, we propose an ontology arithmetic algorithm to reduce the structural changes during ontology evolution and improve the stability and maintainability of ontologies. Unlike traditional ontology evolution methods that mainly focus on inter-entity relationships, we integrate the conceptual relationships between entities based on entity frequencies and inter-entity relationship frequencies when calculating the ontology evolution cost, thus realizing a more fine-grained analysis of entity impact. In addition, we propose additional evolution strategies to select the optimal evolution strategy during the evolution process and retain more complex semantic relations to achieve minimal structural changes. Experimental results show that the evolutionary approach proposed performs well in dealing with ontology evolution involving complex semantic relations, effectively reducing the evolutionary cost and improving the structural similarity before and after ontology evolution. These research results are of great significance in evolving and updating ontologies in Web semantic environments and provide useful improvement directions and practical experience in the field of ontology engineering.

Key words: ontology; ontology evolution; semantic relationships; evolutionary approach; structural changes

收稿日期:2023-08-27　　**修回日期**:2023-12-27
基金项目:国家自然科学基金(U1836118);国家重点研发计划(2022YFC3300800);武汉市重点研发计划(2022012202015070);武汉东湖新技术开发区“揭榜挂帅”项目(2022KJB126);湖北省教育厅科学研究计划项目(B2022016);武汉市知识创新专项-曙光计划项目(2023010201020409)。
作者简介:陆晨阳(2000-),男,硕士研究生,研究方向为语义网及知识图谱;通信作者:顾进广(1974-),男,教授,博士,CCF 杰出会员(05460D),研究方向为语义网与知识图谱、分布式计算等。

0 引 言

本体作为语义 Web 体系架构的第 4 层,是语义 Web 实现的关键。本体是某一领域共享概念模型的明确表示和描述,是一组概念及其概念间关系的集合,为语义网实现提供了基础和关键[1]。在动态发展的语义 Web 环境中,领域知识不断演变,因此相关本体需要进行及时的迭代更新以适应变化。本体演化的原因包括:领域知识的不断变化、数据来源的变化、用户反馈和需求的变化以及本体设计缺陷的发现。这些因素驱动本体进行演化以保持与现实世界的同步,满足用户需求并提高本体的质量和可用性[2]。本体在演化过程中,由于其中的概念、约束关系等某部分的改变,会导致本体进入到不一致的状态,本体内部的不一致性对基于本体的应用和任务会产生不良影响,因此降低本体在演化过程中的结构性变化具有重要的意义。

目前,许多学者对本体演化进行了大量研究[3-5]。在本体演化的过程中对本体中的实例和关系进行精确的度量,有助于更细致地分析本体在演化前后的内部变化。针对此问题,金龙飞等人[6]从定量分析角度给出了本体演化波及效应的分析和具体计算方法,通过建立本体邻接矩阵和可达矩阵,以矩阵变化计算本体元素对本体结构综合影响力。该方法并未考虑到本体元素之间的各种关系,鉴于此刘晨等人[7]分析了服务之间的依赖关系,提出了量化变更影响范围的数学公式;王孝满等人[8]将同时发生的演化需求定义为上下文窗口,并分析窗口内演化需求之间的关系,进而对多窗口进行演化操作;陈晶等[9-10]基于本体中语义关系出现的频率对本体演化过程实例节点的影响进行了量化。上述方法虽然量化了本体演化中节点和关系的权重[11],但均未考虑演化策略对本体结构的影响。

演化策略的选择在本体演化过程中也同样重要。目前,模拟流行的本体编辑工具 Protege[12]、KAON[13]、OilED[14]等都提供了执行本体变更的能力,但它们在变更执行方面较为薄弱。通常情况下,在执行一个变更需求时,这些工具只会进行简单的逻辑处理,以保证本体变更后的一致性,例如删除一个概念总是导致其所有子概念被删除。虽然这种简单的方式易于实现,但并没有减轻本体演化的负担。然而,实际上变更请求之后,存在许多实现一致性的方法。例如,当层次结构中的一个概念被删除时,其所有子概念也可能会被删除或重新连接到其他概念。如果保留了子概念,则删除的概念的属性可能会被传播,其实例可能会被分布等。因此,对于本体中的每个变更,都可能生成不同的附加变更集,从而导致不同的最终一致状态。国内外学者也对此进行了研究,Stojanovic 等人[15]提出了一种用户驱动的本体演化管理方法,将本体演化过程划分为 6 个阶段,分别是变化的捕获、变化的表示、变化的语义、变化的实现、变化的传播以及变化的确认。为了简化演化过程,他们还提出了一些演化策略。这一演化过程需要领域专家从始至终参与本体演化工程,在执行每个变化之前,生成一个变化的影响列表,并呈现给领域专家。领域专家需要理解这个列表的含义,并确定是否执行或取消该变化。Konstantinos I. Kotis 等人[4]也在文献中指出,本体工程师和用户等外在需求是本体演化的重要先决条件。针对过多的人工参与问题,周栩等人[16]提出了附加演化操作和演化策略。此方法虽然提出了具体的策略选择取代了人工参与本体的演化但并没有考虑不同等级关系的演化代价不同以及本体中语义关系的影响。

针对本体演化过程中节点和关系的演化代价的量化:该文引入了本体演化的波及效应来更精确地获得本体中实体的影响力,通过实体之间的语义关系帮助本体使用者对本体进行细致地分析。

针对本体演化过程中演化策略的问题:该文引入了本体一致性集合[17],对概念之间的继承、等价和不相交关系,以及属性之间的继承、等价关系进行分析。在此基础上,提出了新的附加演化操作和演化策略,以确保本体在演化过程中仍然保持一致性。

实验结果表明,该算法能有效地降低本体演化中的结构性变化,保持本体演化前后整体结构的稳定。

1 本体的语义表示

本体是某一领域的一组概念及其概念间关系的集合,该文将通过本体的语义关系图模型来表示本体中各概念属性以及属性概念之间的关系。本体的语义关系图是一个带标签的有向图 G[18](如图 1),$G = \{E \mid R\}$,其中 $E = \{e_i \mid i \in 1,2,\cdots,n, e_i \in O\}$,表示本体中的概念集合,在语义图中的表现形式为节点。$R = \{r_{ij} \mid r_{ij} \in O\}$ 表示本体中概念和概念之前的关系,在语义图中的表现形式为节点与节点之间带标签的有向边,如关系 r_{ij} 表示节点 e_i 和节点 e_j 之间的语义关系,节点 e_i 是关系起点,节点 e_j 是关系终点。

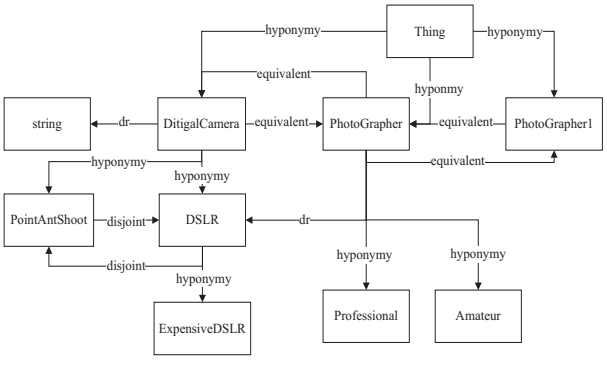

图 1 摄影器材本体图

本体演化在语义关系模型图中的表现形式主要是概念节点和关系有向边的增加、删除和编辑。当本体发生演化时,首先判断节点是否发生变化,如果节点发生改变,则删除需要改变的节点和以该节点为起点终点的有向边,再增加新节点并完善新节点和原有节点的关系形成有向边。完成更新本体中的一个概念以及与之相连的关系,称为一次本体的演化。在演化过程中,当某一节点发生变化时,如果该节点涉及的关系比较简单,按照上述基础的演化过程直接删除再新增节点带来的结构性变化代价不大。但是当节点涉及到的关系太复杂时,上述基础演化过程的删除重建节点的方法会造成与该变化节点相邻的所有节点这个范围内的结构发生较大的变化,进而对上下游基于本体的任务、应用产生不良影响。

2　可降低结构性变化的本体演化算法

该文提出的可降低结构性变化的本体演化算法主要包含 3 个阶段:本体预处理、权重计算、演化策略选择。在本体预处理阶段,对本体中的关系进行了初始分组,以便后续关系权重和节点权重的计算;在权重计算阶段,使用基于 TFOF 的波及效应分析方法[9],从而能够准确地计算实例节点和关系的权重;在演化策略选择阶段,首先确定哪些实体需要进行演化。然后选择效果最好的演化策略来保持本体的结构稳定性,从而降低结构性变化。

2.1　本体语义关系的权重

在本体语义关系模型图中,两个节点之间的关系体现了头尾节点之间关系的密切程度。本体语义关系模型图中包含的节点越少,节点之间的有向边越少,本体关系的复杂度就越低,两个概念节点之前的密切程度就越强。不同的密切程度在本体整体结构中的占比不同,密切程度越高对本体结构的影响越大,反之则越小。

在领域本体中,主要关系集合可以定义为 5 种关系:Hyponymy(上下文关系)表示上下文关系;UnionOf(整体-部分关系)表示整体与部分之间的关系;Equivalent(概念等价关系)表示概念之间的等价关系;Disjoint(不交关系)表示概念之间不相交的关系;Domain(定义域值域关系)表示属性或概念与其定义域或值域之间的关系。其他的关系都可以作为主要关系的细分,表 2 列出了对主要关系的等级划分及初始权重。参考文献[19]的方法,该文将本体中的语义关系划分为 3 个等级,并给出了初始的关系权重,具体如表 1 所示。

表 1 中的关系权重是根据关系等级给出的初始权重值。然而,考虑到该文的主要关注点是本体演化过程中的结构性变化,需要更加精确地计算每个关系的权重值。为此,采用了基于 TFOF 的波及效应分析方法,该方法能够更准确地衡量本体中的语义关系。下面以摄影器材本体(如图 1 所示)为例,具体介绍计算过程:

表 1　本体中关系等级及初始权重

关系名称	初始权重	关系级别
Equivalent	1	1
Hyponymy Disjoint UnionOf	1/2	2
Domain	1/3	3

(1)计算关系 r_{ij} 在本体的各个节点中所占比重(见表 2)。

表 2　摄影本体中关系在各个节点中所占权重

节点	以此节点为起始的关系	所占权重
Thing	hyponymy	1.5
DitigalCamera	hyponymy	1.5
	equivalent	1.25
	dr	1.25
PhotoGrapher1	equivalent	1.5
PhotoGrapher	hyponymy	1.5
	equivalent	1.5
PointAntShoot	disjoint	1.5
DSLR	hyponymy	1.5
	dr	1.5
	disjoint	1.5

$$\mathrm{tf}(r_{ij},e_i)=1+\frac{0.5\times f(r_{ij},e_i)}{\max\{f(r_x,e_i)\mid r_x\in R(e_i)\}}$$
$$i=1,2,\cdots,n,r_{ij}\in R(e_i) \tag{1}$$

在上述公式中:

r_{ij}:代表本体中出现的关系;

e_i:代表本体中的某个实体;

$f(r_{ij},e_i)$:代表与实体 e_i 相关并以 e_i 为关系起点的关系 r_{ij} 出现的次数;

$R(e_i)$:代表与 e_i 相连并且以 e_i 为关系起点的关系种类集合;

$\max\{f(r_x,e_i)\mid r_x\in R(e_i)\}$:代表与实体 e_i 相关并以 e_i 为关系起点的关系出现的最多的次数。

(2)计算关系 r_{ij} 在本体中出现次数的平均权重(见表 3)。

表 3　摄影本体中关系出现次数的平均权重

关系	平均权重
hyponymy	1.5
equivalent	1.42
dr	1.38
disjoint	1.5

$$\mathrm{tf}(r_{ij},O)=1+\frac{\sum_{i=1}^{n}\mathrm{tf}(r_{ij},e_i)}{\sum_{i=1}^{n}f(r_{ij},e_i)}$$
$$e_i\in O,r_{ij}\in R(e_i) \quad (2)$$

(3)计算关系 r_{ij} 在本体中出现的广泛程度(见表 4)。

$$\mathrm{df}(r_{ij},O)=1+\log\left(1+\frac{\sum_{i=1}^{n}\mathrm{rcount}(e_i,r_{ij},O)}{\sum_{i=1}^{n}\mathrm{count}(e_i,O)}\right)$$
$$r_{ij}\in R(e_i) \quad (3)$$

在上述公式中:

$\sum_{i=1}^{n}\mathrm{rcount}(e_i,r_{ij},O)$:代表以关系起点实体的数量;

$\sum_{i=1}^{n}\mathrm{count}(e_i,O)$:代表本体中实体的数量。

表 4 摄影本体中关系出现的广泛程度

关系	广泛程度(权重)
hyponymy	1.26
equivalent	1.15
dr	1.08
disjoint	1.08

(4)计算关系 r_{ij} 在本体中使用的普遍程度(见表 5)。

$$\mathrm{TFOF}(r_{ij},O)=\mathrm{tf}(r_{ij},O)\times\mathrm{df}(r_{ij},O) \quad (4)$$

表 5 摄影本体中关系使用的普遍程度

关系	普遍程度(权重)
hyponymy	1.89
equivalent	1.63
dr	1.49
disjoint	1.62

(5)计算关系 r_{ij} 在本体中的权重(见表 6)。

$$w(r_{ij})=\mathrm{initial}(r)\times\frac{\mathrm{TFOF}(r,O)}{\max\{\mathrm{TFOF}(r_i,O)\}}$$
$$r_i\in R(e_i),e_i\in O,i=1,2,\cdots,n \quad (5)$$

在上述公式中:

$\mathrm{initial}(r)$:代表关系 r_i 的初始权重。

表 6 摄影本体中关系的权重

关系	权重
hyponymy	0.5
equivalent	0.86
dr	0.26
disjoint	0.43

2.2 本体演化的附加演化策略

在本体的演化过程中,节点的结构性代价并不仅与本身有关,还受到其周围与之有关系相连接的节点的演化影响。例如,节点 e 在演化过程中,其周围删除了一些节点,那么节点 e 的本体综合比重会增加;如果节点 e 在本体演化过程中,其周围没有增加或删除对其有影响的节点,即没有增加或删除以节点 e 为头尾节点的关系,那么节点 e 在本体演化前后的综合比重不会发生变化,本体演化前后不会发生结构性变化。

因此,在本体演化过程中,当节点发生变化时,需要按照一定的演化策略来选择对本体结构影响最小的演化路径。为此,文献[15]提出了一种基于演化代价约束的本体演化方法。该方法指出,在本体演化过程中所消耗的资源被称为演化代价,而演化代价由两个部分组成:实体自身结构带来的结构代价和外在环境因素带来的潜在代价。结构代价仅与实体能够直接或间接连接的实体个数相关。实体的权重与其关联的实体个数直接相关,并且与其关联实体之间的语义关系有关。因此,该文的结构代价定义为该节点对所有可达节点的影响力之和。另一方面,潜在代价由领域专家定义,并且领域专家可以通过领域知识对本体中较为重要的实体设置较大的潜在代价值。

为保持本体的一致性,该文定义了本体一致性为本体在演化过程中需要遵守的不变性约束。该约束被定义为从本体的定义中无法推导出矛盾的结论。描述了可能出现的矛盾情况,称之为一个约束集合,如表 7 所示。

表 7 本体一致性定义

约束名称	约束内容
唯一 ID 约束	所有实体都只有一个唯一的 ID
概念继承体系根不变性约束	概念的继承体系是一个有向无存在一个根概念 Thing 且它是其他所有概念的父概念
概念闭包不变性约束	本体中除根概念外的任何一个概念至少都有一个父概念
属性闭包不变性	概念和属性之间可以建立 domain 和 range 关系
数据属性不变性	一个数据属性的值域不可以是概念
属性继承闭包不变性	父属性及子属性必须是属性集合的实体
实例不变性	每一个实例必须关联于一个概念

经过对本体一致性约束集合的分析,结合本体中关系等级及初始权重(详见表 1),得出了以下表格所示的附加演化策略(详见表 8)。这些策略旨在引导本体的演化过程,使其更加稳健和高效。为了确定最终的演化策略,该文结合了节点和关系的权重,计算本体演化前后整体的结构性变化。具体而言,使用了基于

TFOF 的波及效应分析来计量每个语义关系的权重,考虑了节点的关联情况和周围关系的变化对本体结构的影响。接着,对每个附加演化策略计算了其在本体演化过程中的结构性代价。

在这个过程中,目标是选取结构性变化最小的路径作为最终的演化策略。希望通过最小化结构性代价,降低本体演化对现有结构的影响,并尽可能保持本体的稳定性和一致性。表 8 展示了附加演化策略,其中每一行对应一个演化路径,包含了需要进行的节点增加、删除以及关系调整的操作,以便进行比较,选取最优的演化策略,以实现结构性变化的最小化。

表 8　本体演化的附加演化策略

策略名称	策略描述
concepts shallow delete	将要删除概念的子概念连接到其父概念上或连接到根节点上
concepts deep delete	将要删除的概念下的子概念一并删除
property ascend	将要删除概念的属性变成其父概念的属性或变成根节点的属性
property descend	将要删除概念的属性变成其子概念的属性
property delete	将要删除概念的属性删除
equivalent inherit	将向上部分整体概念与向下部分整体概念添加部分整体关系
unionOf inherit	将向上等价概念与向下等价概念添加等价关系
disjoint descend	将子概念与其他概念添加不相交关系
disjoint inherit	将与此概念有等价关系的概念与其他概念添加不相交关系

2.3　算法分析

在本体演化过程中,主要任务是确定本体中存在不一致的节点,该文将这些节点称为预演化节点。接着,计算与预演化节点相关联的节点和节点之间关系的权重。通过比较不同演化策略的路径的演化代价,可以推算出最小演化代价的路径,并计算出演化后本体所有实体的权重,从而最终判断本体结构性变化的大小。总体的计算方法可以由算法 1 来实现。

算法 1:获取本体演化最小演化代价路径。

输入:不一致实体节点数组。

输出:最小演化代价路径及演化后本体所有实体权重。

```
(1) function getShortestWeightPath(nodeOperationLists):
(2) for each no_consistentlists:
(3)     resultPath = getAllNodesShortestPath() #获取总体最短演化路径
(4) for each path in resultPath:
(5)     endPathList = getEndPathList(path) #获取演化后最终路径数组
(6) refreshGraph() #获取变动节点以及构建新的存储邻接表
(7) nodesWeight = getNodesWeight() #计算模型所有节点权重
(8) return resultPath, nodesWeight
```

在本体演化的存储结构更新步骤中,主要任务是确定需要进行更改的节点,并构建新的邻接表存储结构。在这一步骤中,本体可能会增加或删除节点,因此需要有效地处理这些变化。当增加节点时,需要将新节点标记为需要增加的节点,并将其添加到邻接表中相应的位置。这确保了新节点在本体中正确地被添加,并与其他相关节点建立关联;而在删除节点时,不需要将其标记为需要更改的节点,需要考虑其依赖节点所带来的影响。因此,需要将依赖于删除节点的节点标记为需要更改的节点,以便在后续更新中处理它们的关联关系。

算法 2 是算法 1 中更新邻接表的具体实现。在这个算法中,通过遍历本体中的节点和关系,识别需要增加和删除的节点,并相应地更新邻接表的结构。同时,也处理了依赖节点的标记,以确保所有相关节点都能正确地反映本体的演化变化。

算法 2:获取变动节点以及构建新的存储邻接表。

输入:本体演化后的图信息路径以及演化前的节点信息和边信息。

输出:需变更的节点和演化后的语义关系图对应的邻接表。

```
(1) function refreshGraph(path,nodeList,oldGraph):
(2) for each i in nodeList1:
(3) if nodeList1[i] is null:
(4) Node.add(in)
(5) for each i in nodeList:
(6) if new nodeList[i] is null:
(7) for j=0 to oldGraph.length - 1:
(8) if oldGraph[j].tail = in and new nodeList.get(oldGraph[j].head) is not null:
(9) Node.add(oldGraph[j].head)
```

更新邻接表后,需要确定具体需要更新的节点,以计算结构的变化率。由于一个节点的波及效应值受到依赖它的节点的影响,而该节点依赖的节点的波及效应值会受到该节点的影响。在这个步骤中,主要工作是计算更新后节点的权重,该步骤由算法 3 实现,即算法 1 中的 getNodesWeight 函数。

算法 3:重新计算节点权重。

输入:存储图链表和路径数组。

输出:节点的权重。

```
(1) function getNodesWeight(graph,pathList):
(2)  relationsWeight = getWeight(graph)
(3)  nodesWeight = {}
```

```
(4) for each v in graph.vertList:
(5)     weightLists = getShortestPaths(v, graph, relationsWeight, pathList)
(6)     weight = 0
(7) for each weightList in weightLists:
(8)     weight += weightLists[weightList]
(9)     nodesWeight[v] = round(weight, 2)
```

3 实验与评估

3.1 结构相似率

为了验证算法在降低本体演化中结构性变化的效果,引入了本体演化前后本体结构相似率的测量指标,该指标旨在衡量本体演化后整体结构与演化前的相似程度。定义结构相似率为演化后所有实体权重与演化前所有实体权重之比。具体计算方法如下:

$$\text{结构相似率} = \frac{\text{演化后所有实体权重}}{\text{演化前所有实体权重}} \tag{6}$$

通过结构相似率,能够了解本体演化后整体结构性变化的程度。如果结构相似率接近于 1,说明本体演化后的结构性变化较小,保持了较高的相似性;而如果结构相似率远离 1,则意味着本体经历了较大的结构性变化。

3.2 数据集

该文使用两个本体进行实验,分别为:Drosophila Phenotype Ontology(dpo),dpo 本体中是关于果蝇表型的各项数据;FlyBase Controlled Vocabulary(fbcv),fbcv 是一个由 FlyBase 发布的结构化受控词汇表本体,包括果蝇表型本体。并利用摄影器材本体(图 1)进行个例分析。数据集如表 9 所示。

表 9 数据集

本体名称	下载网址	节点数	关系数
dpo	https://www.ebi.ac.uk/ols/ontologies/dpo	316	1 449
fbcv	https://www.ebi.ac.uk/ols/ontologies/fbcv	1 256	3 213

3.3 实验分析

3.3.1 本体演化实验

考虑到本体所有实体的权重较大可能会导致演化前后的结构相似率变化不够明显的情况。为了更好地验证文中算法在降低本体演化中结构性变化方面的效果,选择使用局部子集作为实验对象,dpo 子集包含 10 个节点和 15 条关系;fbcv 子集包含 16 个节点和 35 条关系,使用这个子集来比较所提出的方法与文献[8-9]中的方法之间的差异。实验环境使用 64 位 Windows,32 GB 内存的机器进行实验,编译环境为 IDEA 2023.1.3 版本。在实际情况中,增加操作对本体的影响相对较小,不会对本体原有实体和关系的变化造成较大影响。因此,该文选择删除实体和删除关系操作模拟本体演化过程,表 10 展示了经过删除实体和删除关系操作后,本体演化前后的结构相似率的结果。

表 10 本体演化前后的结构相似率

数据集	删除节点	结构相似率		
		文献[8]	文献[9]	文中
dpo	RO_0002082	0.922 4	0.938 9	0.939 7
	BFO_0000035	0.587 3	0.725 7	0.815 3
	BFO_0000060	0.696 5	0.725 5	0.771 9
	BFO_0000062	0.718 3	0.744 9	0.745 2
fbcv	RO_0003021	0.937 7	0.949 2	0.957 6
	RO_0003026	0.912 8	0.927 8	0.932 2
	BFO_0000120	0.655 8	0.692 3	0.731 9
	BFO_0000125	0.693 1	0.738 3	0.758 4

通过实验数据的分析,不难观察到,那些影响节点数量较多的节点往往伴随着更为复杂的关系网络。在这种情况下,文中算法比文献[8-9]表现出更高的结构相似率,能更为出色地维护了本体的结构一致性。与此相反,对于那些关系较为简单的节点而言,在演化过程中其对整个本体所带来的影响则相对有限。这也就意味着相应的演化策略选择变得较为受限,从而导致了实验结果与文中算法的表现趋于相近。此外,对于本体中的单个关系而言,它对整体结构的影响很小。其主要影响是在计算关系类型权重环节,单独删除一条关系会影响该关系的头节点的权重,但对于整体结构的影响则较小,从而删除关系 RO 的实验结果相近。

3.3.2 基于摄影器材本体的演化分析

为了更加直观地说明文中算法以及演化策略在本体演化中有效性,本节将结合本体语义关系图通过一个具体的本体实例进行更加详细直观的个例分析。以图 1 的相机摄影等本体为例,删除实例 DSLR,以及实例 DigitalCamera 和 Photographer 的主要执行步骤如下:以 Remove(DSLR)操作为例,该操作将删除 DSLR 实体后,会导致以下实体的不一致性:DitigalCamera、ExpensiveDSLR、PointAndShoot 和 Photographer。

解决这些节点的不一致性,可以采取以下演化策略:concepts shallow delete(概念浅层删除)、concepts deep delete(概念深层删除)、disjoint descend(不交降低)、property ascend(属性上升)、property descend(属性降低)和 property delete(属性删除)。根据该文提出的演化算法和演化策略,需要从这些演化策略中选择

出能够实现最小演化代价的演化路径。具体来说,可以采取以下演化路径来实现最小演化代价:

(1)删除 DSLR;

(2)删除 DSLR 和 ExpensiveDSLR 之间的关系;

(3)将 ExpensiveDSLR 作为 DitigalCamera 的子概念;

(4)删除 DSLR 和 PointAndShoot 之间的关系;

(5)将 ExpensiveDSLR 与 PointAntShoot 添加 disjoint 关系;

(6)删除 DSLR 和 Photographer 之间的关系;

(7)将 ExpensiveDSLR 与 Photographer 添加 dr 关系。

演化后的本体结构如图 2 所示。

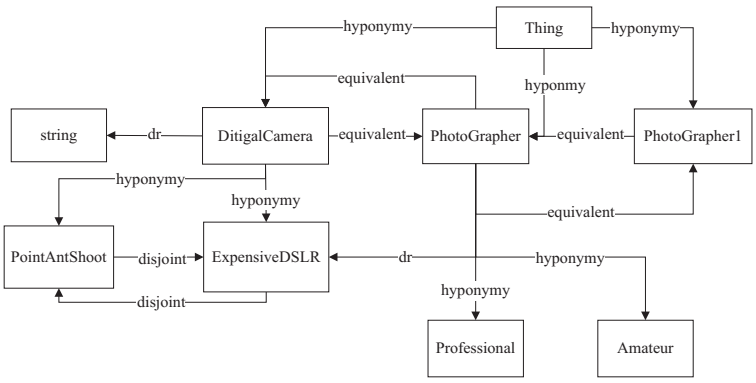

图 2　摄影器材演化后本体图(文中)

而在文献[8-9]中,最小演化代价的路径为:

(1)删除 DSLR;

(2)删除 DSLR 和 ExpensiveDSLR 之间的关系;

(3)添加 ExpensiveDSLR 和 DitigalCamera 之间的关系;

(4)删除 DSLR 和 Photographer 之间的关系;

(5)将 ExpensiveDSLR 与 Photographer 添加 dr 关系。

演化后的本体结构如图 3 所示。

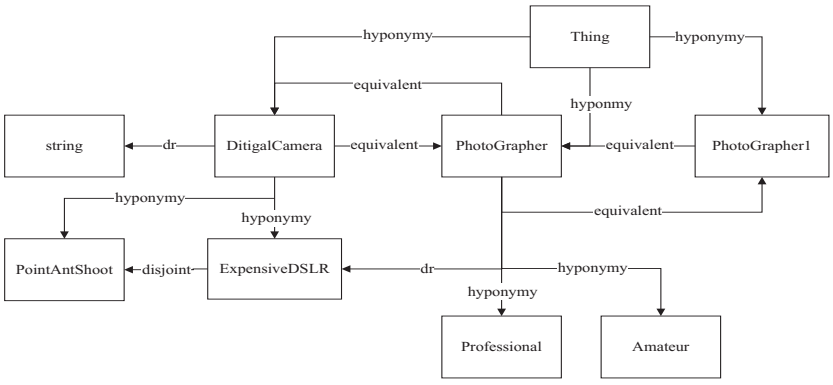

图 3　摄影器材演化后本体图(文献[8-9])

表 11 是在摄影本体上进行删除操作,文献[8-9]和文中的结果对比。

表 11　摄影本体演化前后的结构相似率

操作	结构相似率		
	文献[8]	文献[9]	文中
Remove(DSLR)	0.842 6	0.917 1	0.928 3
Remove(DitigalCamera)	0.731 1	0.732 6	0.732 6
Remove(PhotoGrapher)	0.596 8	0.691 5	0.743 2

通过分析本体语义关系图,可以得出与之前实验结果一致的结论:尽管在关系复杂的节点发生演化后,对本体结构的影响较大;而关系较为单一的节点发生演化后,对本体结构的影响相对较小。然而,经过对实验结果的综合对比分析,所提出的本体演化算法能够在更大程度上保持结构相似性,即在本体演化过程中对结构的保持和优化方面具有显著优势。

在文中实验中,特别关注与变更实体相关的复杂语义关系的处理效果。实验数据表明,提出的可降低结构性变化的本体演化方法在处理与变更实体相关的复杂语义关系时表现较好。这说明该算法在保持本体结构中复杂语义关系的一致性和优化方面具有显著优势。然而,在处理简单语义关系时,与另一方法的结果没有显著差异。这是由于简单语义关系的变化相对较少,文中算法与对比算法在这方面的处理效果较为接近。尽管如此,文中算法仍然能够保持较高的结构相似性,使得本体的整体结构性变化较小。

4 结束语

该文提出了一种可降低本体结构性变化的演化方法,采用邻接表作为存储结构,有效地存储本体结构中的节点,并便于计算本体演化的代价。与传统方法仅考虑实体影响节点个数相比,该方法引入了实体之间的关系信息,从而更细粒度地考虑实体的影响,有效降低了本体演化的代价。此外,通过引入本体一致性集合,在本体演化过程中保留了更多复杂的语义关系,从而提高了算法的表现力和适用性。

然而,该演化算法还存在一些问题需要改进:通常情况下,本体的演化往往只在局部进行,变化的范围相对较小。尽管这符合大多数实际本体的情况,但是对于一些本体来说,在演化过程中变化幅度较大,需要改变许多层次关系。在这种情况下,该方法的准确性可能会降低。因此,未来需要探索新的补救措施,以适应那些涉及到大量层次关系改变的本体演化场景。

参考文献:

[1] 董 尹,刘千里,胡雅萍. 弱信号介入的供应链风险识别本体构建:从顶层本体到领域本体[J]. 情报工程,2020,6(3):78-91.

[2] 黄承曦. 基于语义的微服务应用构建及治理研究[D]. 上海:上海交通大学,2020.

[3] SAFYAN M,QAYYUM Z U,SARWAR S,et al. Ontology evolution for personalised and adaptive activity recognition[J]. IET Wireless Sensor Systems,2019,9(4):193-200.

[4] KOTIS K I,VOUROS G A,SPILIOTOPOULOS D. Ontology engineering methodologies for the evolution of living and reused ontologies: status, trends, findings and recommendations[J]. The Knowledge Engineering Review,2020,35:4-38.

[5] PERNISCH R,DELL'AGLIO D,BERNSTEIN A. Beware of the hierarchy—an analysis of ontology evolution and the materialisation impact for biomedical ontologies[J]. Journal of Web Semantics,2021,70:100685.

[6] 金龙飞,刘 磊. 一种本体演化波及效应分析方法[J]. 电子学报,2006,34(8):1469-1474.

[7] 刘 晨,韩燕波,陈旺虎,等. MINI——一种可减小变更影响范围的本体演化算法[J]. 计算机学报,2008,31(5):711-720.

[8] 王孝满,闫晶晶,李晓阳. 一种基于上下文窗口的本体演化方法[J]. 计算机工程,2011,37(19):53-55.

[9] 陈 晶,刘 钊,顾进广,等. 本体演化中基于 TFOF 的波及效应分析[J]. 武汉大学学报:理学版, 2020,66(2):197-204.

[10] 陈 晶,刘 钊,顾进广,等. 本体演化的波及效应计算优化研究[J]. 计算机应用研究,2020,37(8):2366-2370.

[11] ELFAKI A O,ALFAIFI Y H. Systematic approach for measuring semantic relatedness between ontologies[J]. Electronics,2023,12(6):1394-1341.

[12] NOY N F,FERGERSON R W,MUSEN M A. The knowledge model of Protege-2000:combining interoperability and flexibility[C]//International conference on knowledge engineering and knowledge management. Berlin:Springer,2000:17.

[13] GABEL T,SURE Y,VOELKER J. D3. 1. 1. a:KAON - ontology management infrastructure[J]. SEKT Informal Deliverable,2004,3:1-58.

[14] BECHHOFER S. OilEd:a reason-able ontology editor for the semantic web[C]//KI 2001:Advances in Artificial Intelligence:Joint German//Austrian Conference on AI. Vienna:Springer,2001:396.

[15] STOJANOVIC L,MAEDCHE A,MOTIK B,et al. User-driven ontology evolution management[C]//Knowledge engineering and knowledge management:ontologies and the semantic web:13th international conference. Berlin:Springer,2002:285.

[16] 周 栩,罗景文,周 桐,等. 一种基于演化代价约束的本体演化方法[J]. 吉林大学学报:理学版,2010,48(4):646-652.

[17] 许 勇,王智学,李宗勇. 领域本体的一致性检查[J]. 计算机工程,2009,35(1):55-57.

[18] GONG J,FU H. Ripple-effect analysis of ontology evolution based on CK-modes algorithm[C]//2020 IEEE 7th international conference on industrial engineering and applications (ICIEA). Bangkok:IEEE,2020:638.

[19] 张 祥,李 星,温韵清,等. 语义网虚拟本体构建[J]. 东南大学学报:自然科学版,2015,45(4):652-656.