第35卷 第13期 计 算 机 工 程 2009年7月
Vol.35 No.13 Computer Engineering July 2009
·软件技术与数据库· 文章编号：1000—3428(2009)13—0023—03 文献标识码：A 中图分类号：TP311

基于概念代数的本体演化

宋奂寰，张志祥
(海军工程大学电子工程学院，武汉 430033)

摘 要：针对本体模型的维护和演化问题，将代数格与概念格相结合，定义概念代数，提出基于概念代数的本体演化3层结构，包括原子概念层、原子概念关系层、应用层。分析概念代数与本体表示语言OWL之间的对应关系。在应用层中，利用概念代数根据应用的需要在原本体模型基础上构建面向应用的概念，并维护原有本体模型。实例证明了该方法的有效性。

关键词：概念代数；本体；Web本体语言

Ontology Evolution Based on Concept Algebra

SONG Huan-huan, ZHANG Zhi-xiang
(Electronic Engineering College, Naval University of Engineering, Wuhan 430033)

【Abstract】Aiming at the maintenance and evolvement of ontology model, this paper combines algebraic lattice with concept lattice, defines concept algebra, and proposes 3-layers architecture of ontology evolution, which consists of atomic concept level, atomic concept relations level, and application level. The corresponding relations between concept algebra and Ontology Web Language(OWL) are analyzed. In the application level, application-oriented concept can be builded according to the applied requirement, and existing ontology model can be maintained. An example is designed to validate the method.

【Key words】concept algebra; ontology; Ontology Web Language(OWL)

形式概念分析(Formal Concept Analysis, FCA)是一种支持数据分析的有效工具。用FCA技术构建本体的典型方法有：Philipp Cimina方法[1-2]，Hele-Mai Haav法[3-4]，GuTao法[5]等。它们利用了概念格构建领域知识结构或概念格管理领域知识，都是对领域知识静态地组织与使用，但是任何技术管理的知识都不是静态的，而是动态、进化的。本文将概念格与本体相结合，提出基于概念代数的本体演化方法。

1 概念代数

概念格由形式背景的概念集合与概念间的父子关系组成。概念格中的每个节点称为形式概念，它由外延和内涵组成。概念代数中形式概念的继承、交、并分别用符号≤，×，＋表示，它们分别由代数格理论中的偏序、交、并约束，引入全概念(T)和空概念(⊥)来定义格的界。

形式概念公理[6]如下：

(1)幂等律：$A+A=A, A\times A=A$；

(2)交换律：$A+B=B+A, A\times B=B\times A$；

(3)结合律：$A+(B+C)=(A+B)+C$，

$A\times(B\times C)=(A\times B)\times C$；

(4)吸收律：$A\times(A+B)=A, A+(A\times B)=A$；

(5)分配律：$A\times(B+C)=(A\times B)+(A\times C)$,

$A+(B\times C)=(A+B)\times(A+C)$；

(6)有界律：$A+\perp=A, A\times\perp=\perp, A\times \mathrm{T}=A, A+\mathrm{T}=\mathrm{T}$。

引入属性函数$\alpha(A)$，意思是属性α的值为A。属性通常不单独以这种形式出现，而是以$A\times\alpha(B)$的一个因子出现，意思是形式概念A的子概念具有属性α，其值是B。属性结构的一般表示形式为$A\times\alpha_1(A_1)\times\alpha_2(A_2)\times\cdots\times\alpha_n(A_n)$。

属性函数公理如下：

(1) $\alpha(A+B)=\alpha(A)+\alpha(B)$；

(2) $\alpha(A\times B)=\alpha(A)\times\alpha(B)$；

(3) $\alpha(\perp)=\perp$；

(4) $\alpha(\mathrm{T})=\mathrm{T}$。

属性函数定理为：属性函数是单调的，即

$B\leqslant A\Rightarrow\alpha(B)\leqslant\alpha(A)$

2 基于概念代数的本体演化

基于概念代数的本体模型如图1所示。

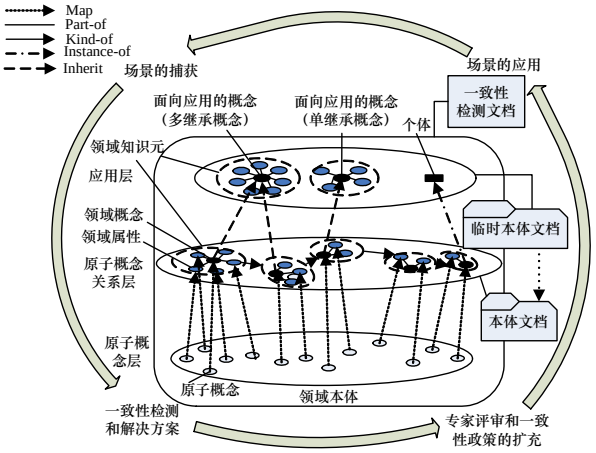

图1 基于概念代数的本体模型

基金项目：湖北省教育厅基金资助项目“汇编遗留系统逆向工程技术研究”(b200714004)

作者简介：宋奂寰(1983－)，女，硕士，主研方向：软件质量保证；张志祥，副教授

收稿日期：2008-11-24 **E-mail：**songhuanhuan_2005@163.com

2.1 原子概念

原子概念层用于存放原子概念及与之相关的知识。原子概念(Atomic Concept, AC)是不能再分解的概念。原子概念由如下四元组描述:

$AC=(e,s,c,o)$

其中，e 为原子概念解释；s 为原子概念的相似词汇；c 为原子概念的性质；o 为原子概念的操作。利用本体进行信息检索时检索到的信息都源于这个四元组，它们存放在数据库中。

2.2 原子概念关系层

原子概念关系层构建原子概念所组成的知识体系。本文将原子概念分成“领域概念”和“领域属性”两大类。领域概念是领域中相对独立的知识单元；领域属性是领域概念具有的性质，是一个概念区别于其他概念的特征。领域概念和描述其特征的领域属性一起描述了一个领域知识元的内部结构。建立领域概念之间的关系，根据概念之间的包含关系、一般和特殊的关系建立领域概念之间的层次关系。领域概念间的层次关系描述了领域知识元的外部结构。

2.3 应用层

按照应用程序的需要，利用原子概念关系层创建应用程序需要的个体或“面向应用的概念”。面向应用的概念是根据应用的需要，由原子概念关系层中的领域知识元作为其组成部分的概念。面向应用的概念有 2 类：单继承概念和多继承概念。单继承概念只继承原子概念关系层的一个领域概念；多继承概念继承原子概念关系层的多个领域概念。面向应用的概念通常在继承得到的领域概念中增加/删除 1 个或多个领域属性。应用层中的概念和个体如图 1 所示。

A, B 是原子概念关系层中 2 个的领域概念，C 是应用层的面向应用的概念，α 是属性，运用概念代数可以创建面向应用的概念 C 和本体维护:

(1)在概念 A 之后插入一个新的概念 C: $C \leqslant A$。

规则: 完成这个任务有 2 种方案:

1)不论 A 是否有子概念，都将 C 作为 A 的子概念，而不影响 A 与其他概念之间的继承关系。

2)如果 A 有子概念，将 A 的所有子概念都转换成 C 的子概念，然后将 C 作为 A 的子概念。

(2)在属性 α 之后插入一个新的属性 β: $\beta \leqslant \alpha$。规则与(1)相似。

(3)C 合并/继承 A 和 B 的交: $C=A\times B$。

(4)C 合并/继承 A 和 B 的并: $C=A+B$。

规则: 如果 A 的某个属性与 B 的某个属性发生冲突，通知应用程序或用户。

(5)C 修改 A，即增加了值为 D 的新属性 α: $C=A\times\alpha(D)$。

规则: 如果 A 的某个属性与新增 α 发生冲突，通知应用程序或用户。

(6)删除概念 A: $A=\perp$。

规则: 如果属性 A 有子概念，完成删除任务有 2 种方案:

1)将 A 连同其子概念一起删除。

2)将 A 的子概念转换成其父概念的子概念，并保证这些子概念原有的继承关系不变，然后删除 A。

(7)从 A 中删除属性 α: $\alpha=\perp$。

规则: 如果属性 α 有子属性，完成删除任务有 2 种方案:

1)将 α 连同其子属性一起删除。

2)将 α 的子属性转换成 A 的子属性，并保证这些子属性原有的继承关系不变，然后从 A 中删除 α。

(8)将概念/属性从一个位置转移到另一个位置。

规则: 首先从其父概念/属性中删除概念/属性，然后将这个概念/属性插入到指定位置。

为了在本体语言中实现概念代数，本文给出概念代数与 Web 本体语言(Ontology Web Language, OWL)之间的对应关系，如表 1 所示，其中，A, B 是用 OWL 声明的概念；α,α_i,α_j 是用 OWL 声明的属性。

表 1 概念代数与 OWL 的对应关系

概念代数	OWL
C	Class
$A\leqslant B$	subClassOf
$\alpha(A)$	α : ObjectProperty, A: range
$C=A\times\alpha(B)$	C: subClassOf A, α : ObjectProperty A: domain, B: range
$C=A+B$	equivalentClass, unionOf
$C=A\times B$	equivalentClass, owl: intersectionOf
$\alpha_i(A)\leqslant\alpha_j(B)$	owl: subPropertyOf

按照表 1 的对应关系，可以实现利用本体模型中已有的概念创建面向应用的概念。例如，已声明了动物(Animal)类和植物(Plant)类，在概念代数的指导下可以构建一个新的概念“素食动物”: $Herbivorel=Animal\times main\_eat(Plant)$ 创建了名为 *Herbivorel* 的概念，它是 *Animal* 的子类，并且具有属性 *main_eat*，其值是 *Plant* 。

2.4 一致性检测循环

本体所代表的知识不是静态的，而是随着时间动态进化、改变的，即本体需要维护。每一次创建或修改称为一个场景，场景用 OWL 描述，被记录在临时本体文档中(见图 1)。临时本体文档与本体文档是分开的，本体文档中记录的是用 OWL 描述的原子概念关系层，而临时本体文档中记录的是用 OWL 描述的场景。

应用层的构建或本体的维护始于场景的捕获，场景是由插入、删除、合并和移动激发的，如果应用程序在临时本体文档中插入一段新的场景，这个场景将立即被捕获。捕获后的场景被送到第 2 个步骤: 一致性检测和解决方案。每一个场景的增加都可能导致本体的不一致性。一致性检测文档中存放的是常见的不一致性案例及解决方案，如果不一致性在一致性检测文档中有记录，则该场景的不一致性将被修正，否则暂时认为该场景不存在不一致性，进入循环的第 3 个步骤: 专家评审和一致性政策的扩充，由专家检测场景与整体本体的一致性，如果存在不一致性，则用相应的方案修正场景，并将不一致性和解决方案增加到一致性检测文档中；否则进入循环的第 4 个步骤: 场景的应用，即将临时本体文档中的场景移动到本体文档中，一次一致性检测循环结束。如果一个新的场景加入到临时本体文档中，新的场景将被捕获，新的一轮一致性检测循环开始。

2.5 逻辑推理

本体的推理子系统用于从显示的 OWL 声明中获得附加的 OWL 声明。本体蕴含的语义是通过本体中概念、属性、个体三者之间存在的关系表达的，因此，在某些情况下为获得本体模型蕴含的丰富的语义，须全面地挖掘本体模型的关系。Jena 提供了推理机(Reasoner)获得各种类型的隐含声明，这些隐含声明存在于继承关系中。表 1 中的 $C=A+B$ 和 $C=A\times B$ 与 OWL 语言中关键字“equivalentClass”之间的对应关系，正是由于 equivalentClass 关键字的存在，Jena 推理机才能够使面向应用的概念继承其父概念的资源。

3 实例及分析

约 50%的软件维护任务是修改遗留系统以满足新的需要。本文用 Protégé 3.2 设计了一个软件维护本体以验证上述方法。软件维护本体分成 2 层：原子概念层和原子概念关系层，存放于本体文档中。原子概念层存放软件的应用领域、商业规则、设计决策和源程序各个模块的功能说明，数据结构的元概念本体如图 2 所示。

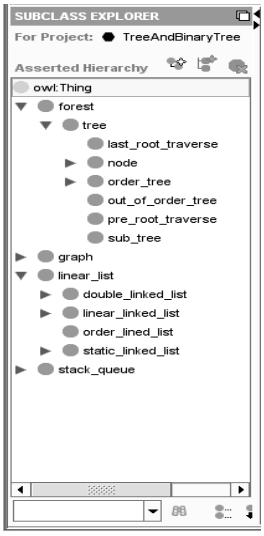

图 2　数据结构的元概念本体

用户使用本软件后，提出新的需求：在已有的 linearlist 和 tree 基础之上，需要一个新的数据结构 tree_linearlist，它具有如下特征：继承 linearlist；继承 tree；继承了 tree 和 linearlist 的所有资源；增加了 1 个属性 traverse，其定义域是 tree，值域是 pre_root_traverse。

利用概念代数，将上述需求用概念代数公式描述：

$$tree\_linearlist = linear\_list + Cause(tree \times traverse(pre\_root\_traverse))$$

根据表 1 将上式转化为如下代码存放于临时本体文档中：

```
<owl:Class rdfID="tree_linerlist">
  <owl:equivalentClass>
   <owl:Class>
    <owl:unionOf rdfparse Type="Collection">
      <owl:Class rdfabout="#linear_list"/>
      <owl:Class rdfabout="#tree"/>
    </ owl:unionOf >
   </owl:Class>
  </owl:equivalentClass>
  </owl:Class>
    <owl:ObjectProperty rdfID="traverse">
     <rdfs:range rdf:resource="#pre_root_traverse"/>
     <rdfs:domain ref:resource="#tree"/>
   </ owl:ObjectProperty >
```

应用程序捕获该场景，进入一致性检测循环，由于场景与整体本体文档是一致的，因此将临时本体文档中的这段场景转移到本体文档中，得到图 3 所示本体，其中线标注的是新增本体，经本体的逻辑推理机推导得到图 4 所示本体，可见新增本体继承了父概念的资源，满足了用户的需求。

在应用程序中查询该新增本体信息的窗口如图 5 所示。

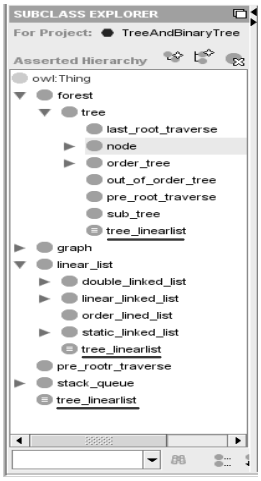

图 3　新数据结构 tree_linearlis

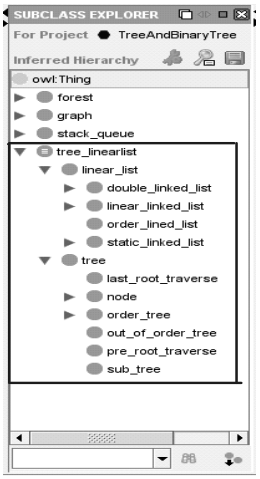

图 4　附加 RDF 的声明

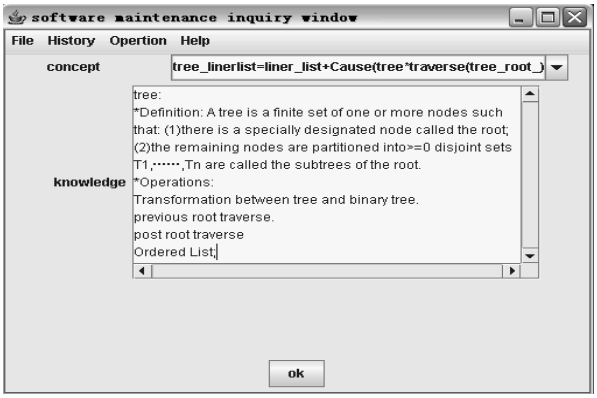

图 5　信息查询窗口

4 结束语

本文提出基于概念代数的本体演化思想，运用该方法可在已有本体模型基础上构建新的本体，建立基于概念代数的本体模型，找出概念代数和 OWL 之间的主要对应关系，但该关系尚不完全，须进一步扩充。

参考文献

[1] Uschold M, Gruninger M. Ontologies: Principles, Methods, and Applications[J]. Knowledge Engineering Review, 1996, 11(2): 93-155.

[2] Cimiano P, Staab S, Tane J. Automatic Acquisition of Taxonomies Form Text: FCA Meets NLP[C]//Proceedings of the International Workshop on Adaptive Text Extraction and Mining. Trento, Italy: [s. n.], 2003.

[3] Cimiano P, Staab S, Tane J. Deriving Concept Hierarchies from Text by Smooth Formal Concept Analysis[C]//Proc. of the GI Workshop on LLWA'03. Karlsruhe, Germany: [s. n.], 2003.

[4] Haav H M. An Application of Inductive Concept Analysis to Construction of Domain-specific Ontologies[C]//Proceedings of the Pre-conference Workshop on VLDB'03. Berlin, Germany: [s. n.], 2003.

[5] Haav H M. A Semi-automatic Method to Ontology Design by Using FCA[C]//Proceedings of CLA'04. Ostrava, Czech Republic:[s. n.], 2004.

[6] 时百胜，余　泓. 概念知识表示和推理[J]. 小型微型计算机系统, 2006, 27(9): 1618-1622.

编辑　顾姣健