摘要

关键词：数据质量；本体；Jena；Jess；熵权法

Abstract

Keywords:Informatica Data Quality；ontology；Jena；Jess；Entropy weight method

目录

[摘要 1](#_Toc9982)

[Abstract 1](#_Toc15031)

[目录 1](#_Toc15422)

[第1章 概述 2](#_Toc14847)

[1.1 选题背景及意义 2](#_Toc20586)

[1.2 国内外研究的现状 4](#_Toc23056)

[第2章 本体及相关技术 5](#_Toc8863)

[2.1 本体简介 5](#_Toc29666)

[2.2 OWL语言 5](#_Toc11513)

[2.3 Protege本体建模工具 6](#_Toc12802)

[2.4 Apache Jena技术 6](#_Toc11568)

[2.5 Jess介绍 6](#_Toc2875)

[2.6 数据质量 7](#_Toc30474)

[2.6熵权法 7](#_Toc957)

[2.7 Java 7](#_Toc1904)

[2.8 IntelliJ IDEA介绍 8](#_Toc28766)

[第3章 数据质量本体构建 9](#_Toc17711)

[3.1 本体的概念、关系以及约束公理 9](#_Toc5341)

[3.2 本体的构建准则 11](#_Toc1187)

[3.3 数据质量本体构建流程 12](#_Toc32084)

[3.4 数据质量本体类以及属性 14](#_Toc19938)

[3.5 数据质量规则对在本体中的描述 15](#_Toc13382)

[第4章 基于Jess的数据质量评价方法 17](#_Toc8483)

[4.1 基于SWRL的Jess推理 17](#_Toc16861)

[4.2 熵权法在本体中的设计实现 18](#_Toc4979)

[4.3 数据质量的jess推理实现 20](#_Toc1619)

[第5章 系统实现 20](#_Toc11767)

[5.1 数据质量本体 21](#_Toc20657)

[5.2 熵权法的实现 24](#_Toc3141)

[5.3 数据质量评价实现 24](#_Toc23601)

[第6章 系统测试 25](#_Toc23206)

[6.1 测试目标 25](#_Toc29195)

[6.2 测试方法 25](#_Toc6970)

[6.3 测试用例 26](#_Toc6789)

[结论 27](#_Toc4429)

[参考文献 28](#_Toc24424)

[致谢 29](#_Toc27756)

第1章 概述

## 1.1 选题背景及意义

近些年来，计算机技术以及信息处理技术获得了跨越式的发展，也因此产生了大量的数据，由此标志着大数据[1]时代的到来。

要做出正确的决策，离不开数据架构中更加统一、精确的大数据，也就是说，所拥有的数据越准确、正确，越能够与真实的世界线连接。数据是信息的载体[2]，高质量的数据是数据分体、数据挖掘等数据信息提取技术能够得到无误信息与结果的基础。因此，各企业所拥有数据的质量也成为各企业关注的焦点。

在大数据时代，经常会听到人们谈论关于拥有大量数据，却缺少有用信息的问题。其中，数据分析技术不完善是最重要的问题。除此，保有数据质量较低也是重要因素之一，这就致使空有大量数据，而不能产出应有的数据效益。

目前，有效、完善管理数据采集、录入等各个环节，仍然没有一个统一、标准化的解决方案。虽然人类从很久之前就在研究数据质量，涉及范围也非常广泛，比如人工智能、知识工程[3]、数据仓库、统计学等，而大家也越来越依赖于数据，越来越关心数据分析、挖掘的结果，但是诸如数据的错误录入、错误拼写、数据空缺、不同来源数据冲突等一直无法有效解决问题的存在，致使我们获取到的数据通常是脏数据[4]。脏数据的存在对于大数据分析与研究的影响，就可能因为错误的结果导致错误的决策，导致错失良机还是小事，但是若因此蒙受巨大损失，就显得满盘皆输了。由此可见，提高数据质量、去除脏数据是多么重要的需求。大数据发展到现在，对于这两个问题的解决方案很多，但终究没有比较完美成体系的解决方案。

随着近几年人工智能和知识工程的飞速发展，本体在众多领域也受到了广泛的关注。由于大数据具有大数据量、数据应用范围广泛的性质，而本体却有着共享与复用的性质[5]。，因此，我们顺理成章的考虑用本体的技术方法来解决大数据所存在的问题。

数据质量本体是一种领域本体[6]，通过本体进行的数据质量评价，能够独立于具体的应用和所在的环境。这样，在数据评价领域，数据质量本体[7]就具有超乎想象的公用性，能够使得前有方法中关系模型进行的数据评估所存在的不具适用性的通病。关系模型进行的数据评估都是应用于具体的某一种应用和具体所处的环境，并且在处理数据质量评价时，必须分别对其中的复杂约束问题和简单约束问题应用不同的解决方案，使得无端增加了评估算法的空间复杂度和时间复杂度。最麻烦的问题还在于无法共享，更无法适用于不同源的数据和不同应用的数据。

数据质量本体是对数据评估领域知识的建模[8]，是对数据评估领域知识的规范化模式。主要是抽取了数据质量的评估规则，并且对这些规则进行抽象，提取来构建数据质量本体。通过对数据评估领域知识的规范化，形式数据评估领域的评估标准，使得在数据评估领域有一套可以通用，共享的数据评估标准。

数据质量本体构建及应用研究是针对数据质量出现的问题，利用本体的思想和方法来构建数据质量本体[9]。通过数据的评估规则约束，例如非空约束，值域约束，逻辑依赖约束，存在性依赖约束，等值约束，等值一致性约束，逻辑一致性约束等，抽取出数据质量本体中的属性。利用数据质量本体，对数据库中的数据进行质量评估，选出其中的不合格数据。所以数据质量的评估对于数据的分析以及研究具有很大的现实意义，能够选出不合格的数据，分析数据质量的高低。

## 1.2 国内外研究的现状

关于本体研究已经持续很多年了，在前期，都是由人手工的创建所需的本体架构。自然了，这种方法[10]存在在着很多的缺点。在手工创建之前，需要对本体所要解决的问题和所在的领域有一定的了解，然后通过人工的发掘其中的类（即概念），并需要理清楚这些类之间的附属关系和作用层次，之后，才可以根据整理好的思路，运用诸如Protege等可以进行本体编辑的工具创建所需的本体，最后再对创建的本体进行评价，以更好的修正和完善该本体。

本体编辑工具归根结底也只是一个工具，只不过提供了一些便捷和友好的UI界面。而本体类的构思，理清明晰的类关系和层次，仍然是由对应领域专家来完成。伴随着对本体的研究逐渐趋于成熟，创建本体产生了半自动化的方案。它能利用已存在的资源去降低人在本体创建过程中的工作量。目前，创建本体的半自动化研究在国内外都取得了一定的成果。应用二维的方式来记录类和关系，这是对本体创建的一次重大突破。眼前，分析关系模式的语义信息来抽取本体构建时赖以基础的类和关系。综上所述，在没有一个比较完美的系统开发出来之前，应用领域本体的技术是数据质量评估的最好选择。

关于数据质量的研究，相对国内，国外要开始的早得多，并取得了不菲的成果。而国内借助于后起优势，奋赶直前，与国际的距离已越来越小。在信息爆炸的今天，对数据精确处理的需求一天比一天紧迫。在大量社会资源投入的情况下，数据质量的研究必将成指数型爆发，相信不许很久，数据质量评估与分析将成为各个企业发现企业问题，发现商机的有利工具。本文也将基于Jess[11]对数据质量进行评价研究。

第2章 本体及相关技术

## 2.1 本体简介

本体（ontology），该词诞生于哲学，最初是对事物本身的探究。在人工智能界，Neches等人给出了本体最早的概念。他们认为：“本体定义了组成主题领域的词汇表的基本术语及其关系，以及结合这些术语和关系来定义词汇表外延的规则。”（“An ontology defines the basic terms and relations comprising the vocabulary of a topic area,as well as the rules for combining terms and relations to define extensions to the vocabulary.[12]”）。随着越来越多得人对本体进行研究，其他人也根据自己的理解给出了他们自己的理解，其中得到普遍认同的是Gruber提出的。他的意思就是，本体是各个概念在逻辑理论上的关系形成的一种分类方法。在具体的一个领域中，本体就是领域术语和它们之间的关系和层次。本体也可以成为软件开发的对象模式和类的定义。

## 2.2 OWL语言

OWL：Web Ontology Language，网络本体语言，用于对本体语义进行描述。OWL是以DAML+0IL为基础开发的，因此也保留了DAML+0IL的部分特性。W3C给出了OWL Lite、OWL DL和OWL Full三种不同的OWL语言，用以应对不同的需求选择不同的语言类型。

OWL Lite具有最弱的表达能力。它牺牲了部分公理约束，从而提高了推理的效率。OWL Lite是仅需要一个分类层次和简单约束的用户的最佳选择。

OWL DL（Description Logic，描述逻辑）具有较强的表达能力。它注重于可判定的推理能力，而不关注对于RDF的相性。如果需要逻辑的推理能力，那么就可以选择OWL DL。

OWL Full没有任何限制，持有OWL语言的全部成分，但不能保证是可进行判定推理的。对于不要求计算，需要强大的表达和自由的需求，可以考虑OWL Full。

## 2.3 Protege本体建模工具

Protege是一个开源的，并被广泛应用的本体创建和编辑的软件。它也能够实现一些其他的功能比如推理，排列等。它具有以下特征[13]：

（1）可扩展的知识模型能够使用户重新定义原始知识集合；

（2）强大的本体导入导出功能，可以从RDFS、带DTD的XML文件、XMLSchema等文件中导入本体，也可以将本体转化为多种形式化语言描述，如RDF(S)、OWL等；

（3）具有友好的开发界面；

（4）具有强大的功能插件体系和开放的模块化风格，基于开放式组件的体系结构使系统开发者可以通过生成恰当的插件以增加新的功能；

（5）提供一个半自动化工具PROMPT用于自动地执行本体的合并和排列；

（6）Protege软件支持基于框架和应用于语义网两种类型的本体建模方式。

由于该软件易于上手、性能优越、开源免费、可扩展、持续更新等诸多优点而受到广泛的使用。

## 2.4 Apache Jena技术

Jena是Java应用程序对本体文件进行解析的一个接口，最开始是由HP公司开发维护，后来转由Apache维护。其实，Protege也是在Jena基础上开发的。Jena除了处理owl文件以外，还可以处理RDF，这是Jena最基本的功能，除此之外，就是实现了基于规则的推理。有了Jena，我们就可以实现应用程序对本体文件的解析，而不用我们每个人都去写一套类库或者方法库去解析本体文件。就好比我们需要用电脑去完成我们的各种工作，但我们不必每个人都去造自己的电脑，甚至我们也不必知道如何去造一台电脑，只需别人造好，我们拿来用就可以了。在Jena中，处理owl文件是与处理一般RDFS文件有着很大的区别的。OntModel是专门处理本体的，它具有父接口Model许多没有的方法，比如listClasses()方法是列出所有的类，listObjectProperties()方法是列出所有的Object属性，这两个概念是本体中才有的。另外，Jena还提供了OntDocumentManager（本体文档管理器）用来协助管理Ontology中的class。

## 2.5 Jess介绍

Jess是[Java](http://baike.baidu.com/item/Java/85979" \t "http://baike.baidu.com/_blank)平台上的[规则引擎](http://baike.baidu.com/item/%E8%A7%84%E5%88%99%E5%BC%95%E6%93%8E" \t "http://baike.baidu.com/_blank)，它是CLIPS[程序设计语言](http://baike.baidu.com/item/%E7%A8%8B%E5%BA%8F%E8%AE%BE%E8%AE%A1%E8%AF%AD%E8%A8%80" \t "http://baike.baidu.com/_blank)的超集，由Sandia国家实验室的Ernest Friedman-Hill开发。

Jess是一个体积最小、运行速度最快的推理引擎。它通过模式匹配，读取一条或几条规则，通过规则中的事实和关系，推理出一个新的事实。它是使用一种解决复杂的多对多模式匹配很高效的Rete算法来处理规则，这是它快的重要原因。

Jess有许多特性：后向[链表](http://baike.baidu.com/item/%E9%93%BE%E8%A1%A8" \t "http://baike.baidu.com/_blank)、[运行内存](http://baike.baidu.com/item/%E8%BF%90%E8%A1%8C%E5%86%85%E5%AD%98" \t "http://baike.baidu.com/_blank)查询以及操作和直接推理Java对象的能力。Jess也可以在不编译Java代码的情况下，直接创建对象并调用Java中的方法。

## 2.6 数据质量

在一个企业中，用户数据是最具价值的无形资产。高质量的数据通过分析就能够让公司及时做出决策，把即将发生的危机扼杀在摇篮之中，即将呈现的商机提前研发布局。

具有较高一致性、完整性、准确性的数据，才能称得上是高质量的数据。一个企业中数据质量的高低是根据数据分析后结果作出正确决策的决定因素。本文就是对数据质量评估方法的实际介绍与实现。

## 2.6熵权法

熵是一个系统无序程度的度量[13]。熵值越大，表示一个系统越混乱。在信息系统中，熵值越小，则说明该组数据指标所提供的有效信息量越大，也因此该指标应该占有更大的权值，这就是熵权法的基本思路。层次分析法收到人为因素影响太大，结果太过于主观，结合熵权法，可以在一定程度上降低主观因素造成大影响，是数据质量评价的结果更加贴近于事实。

## 2.7 Java

Java是一种拥有跨平台、面向对象、泛型编程的特性，被广泛应用的一种编程语言。Java是以C++为基础开发的一种开源语言，既吸收了C++面向对象的优秀思想，摒弃了C++中难以理解的多重继承和地址操作，又实现了扩平台，泛型约束，分布式，嵌入式和对web的支持，使得Java成为众多编程语言中的佼佼者。本文选择Java的一个重要原因是因为Java语言有很多的第三方类库，免去了在前人已解决问题上浪费时间，提高了开发的效率。

## 2.8 IntelliJ IDEA介绍

IDEA全称IntelliJ IDEA，是Java语言开发的IDE（集成开发工具）。在Eclipse一家独大，傲视群雄的时候，这个新的IDE悄然诞生。由于其强大的功能完爆Eclipse，使得IDEA迅速在编程社区刷屏。但目前其取代Eclipse的最大障碍就是完整版是收费的，且比较贵。不过对于一般情况而言，社区免费版已经可以满足用户使用了。

IDEA也是一款可扩展的软件。其在智能代码提示，重构，调试，主流框架的支持等方面远超Eclipse等其他IDE，对于版本控制工具也有很好的支持。

IDEA所提倡的是智能编码，是减少程序员的工作，IDEA有如下特色功能：

完美的自动代码完成

丰富的导航模式

JUnit的完美支持

对重构的优越支持

智能提示

XML的完美支持

代码检查

历史记录功能

对JSP的完全支持

可自定义的排版功能

智能编辑

EJB支持

多点编辑模式

预置模板

JavaDoc预览支持

动态语法检测

版本控制完美支持

未使用代码的检查

智能选取

智能生成

正则表达式的查找和替换功能

程序员意图支持

第3章 数据质量本体构建

## 3.1 本体的概念、关系以及约束公理

3.1.1本体的概念

本体在经过如此长久地发展后，有了很多的定义。被普遍接受的定义是本体具有以下四层含义[14]：

① 概念模型，从具体情况事物抽离出独立于该事物的概念得到的模型；

② 明确性，对于概念的类型、关系、约束等进行明确的定义；

③ 形式化，能够被计算机理解；

④ 共享，在领域内，概念应该被广泛认同。

本体相当于一个领域术语收录集，记录了在领域中被广泛认同的概念，以及他们之间的关系、层次，使之清晰地呈现在使用本体的人们眼前。

3.1.2本体之间的关系

在前辈的研究积累下，已经确定本体中存在四种最基本的关系。它们分别是：部分与整体的包含关系，父类与子类的继承关系，对象与实例的映射关系，类与类之间的逻辑关系。详细如下：

（1）part-of：部分和整体的关系，表示一个类是另一个的子类，但是该类也是一个独立的类，例如树叶是树的一部分，但是树叶也是单独的一个类；

（2）kind-of：继承关系，表示一个类是另一个类的子类，对应于OWL语言的owl：SubClassOf,拥有这个关系的子类将继承父类的所有属性及约束；

（3）instance-of：实例关系，表示一个实例是属于某个类的，例如学生类中有一个实例是张三这个学生，那么张三和学生类之间的关系就是实例关系；

（4）attribute-of：属性关系，表示的是两个类之间的关系，对应于OWL语言中的owl：Property，属性关系表现了概念之间的关系，对应的OWL语言表示有：owl：ObjectProperty和owl：DatatypeProperty；

3.1.3本体的约束公理

约束(Restriction)公理是OWL类公理的核心部分，用来详细描述概念所存在的各种约束条件。每个约束公理都是一个匿名类，是被约束类的一个父类。每个约束公理只表示一个约束条件，是对某个特定属性在特定类之上的约束，也就是说，这个约束只在包含这个属性的类定义中有效，如果没有明确声明，其他包含这个属性的类定义则不受此约束。这样定义的本体满足了最小的本体承诺(Minimal Ontological Commitment)规则，即本体的承诺应该最小，只需要定义约束最弱的公理，满足特定的需求即可。约束公理可以分为值约束(value constraint)和基数约束(cardinality constraint)两种，值约束限制属性的值域(range)，基数约束限制属性取值的个数。

1.值约束公理

（1）allValuesFrom：相当于全称量词。被约束属性的所有取值都必须是由allValuesFrom所指定的类的实例，或者是指定值域的数值，取值可以为空，应用用实例属性断言的取值中；

（2）someValuesFrom：相当于存在量词。被约束属性的所有取值中至少有一个是someValuesFrom所指定的类的实例，或者是指定值域的数值。取值可以不能为空，应用用实例属性断言的取值中；

（3）allValuesFrom和someValuesFrom称为量词约束，可以用一个三元组表示：<quantifier,property,filler>，即一个量词、一个属性以及和拥有这个属性的一个类；

（4）hasValue：被约束属性的所有取值中至少有一个是hasValue所指定的值或者与指定的值语义上相当。取值不能为空，用该公理来表示非空的约束。

2.基数约束公理

（1）maxCardinality：被约束属性的值(个体或数值)最多能取maxCardinality所指定的数目的不同值，该约束用来控制实例的属性取值个数，不能超过maxCardinality所指定的数目；

（2）minCardinality：被约束属性的值(个体或数值)最少应取minCardinality所指定的数目的不同值，该属性用来控制实例属性取值个数，当minCardinality取值为1时，表示该属性取值至少有一个，可用该约束来控制非空约束；

（3）cardinality：指定被约束属性的取值基数，可以用一组取值一致的maxCardinality和minCardinality来替代。

OWL 约束公理的应用主要在以下3个方面：

(1) 定义类的必要条件。仅有必要条件约束的类称为Primitive类，也叫Partial类。只要一个个体是某个Primitive类的实例，就可以判断这个个体一定满足这些条件；但如果一个个体都满足这些条件，也不能就此判断这个个体是这个Primitive类的实例。

(2) 定义类的充要条件。有充要条件约束的类称为Defined类，也叫Complete类。只要一个个体是某个Defined类的实例，就可以判断这个个体一定满足这些条件，反过来，如果一个个体都满足这些条件，也可以判断这个个体是这个Defined类的实例。

(3) 定义类的封闭公理(Closure Axioms)。封闭公理用约束的形式说明某个类上某个属性的取值范围，不同于论域和值域，前者是局部约束，后者是全局约束。封闭公理有两种:

(1) 用allValuesFrom封闭someValuesFrom；

(2) 用cardinality =n封闭allValuesFrom。

## 3.2 本体的构建准则

具体工程和领域的不同，所要考虑的重点也不同，本体构建的过程方法也各不相同。目前还没有一个统一的本体构建方法标准，1995 年 Gruber 提出的五条准则[22]在本体构建领域内得到了比较广泛的认可。

(1) 清晰性(Clarity)、客观性(Objectivity)与明确性(Clarity)：本体应该能够有效的说明所属的内涵，即能够通过自然的语言对需要定义的术语进行客观的、明确的定义，即能够有效地表达出所需定义的术语意思。并且定义还应该可用形式化的逻辑公理表达出。

(2) 完全性(Completenese)：尽可能完整的定义本体的概念，可以完备的清晰表达出所需描述的术语内含。

(3) 一致性(Coherence)：本体应该具有前后的一致性，即通过术语得到的推论和术语原本的含义间是相容的，即在与其定义一致的推理时不会产生矛盾；使用自然语言进行说明的文档与本体公理也要保持一致。

(4) 最大单调可扩展性(Extendibility)：本体应该可为后期可预见的任务提供概念基础，能够在现有的概念基础之上增添新的术语，即无需修改现有的概念定义就可向本体中再添加新的专用或通用术语。

(5) 最 小 承 诺 (Minimal ontology commitment) 和 最 小 编 码 偏 好(Minimal encoding bias)：在本体构建初期时本体约定应该最小，仅仅需满足特定的共享需求，之后的使用者可以按照自己的需要对本体进行专业化和实例化的约束。本体的承诺是指如何通过相容的、一致的方式来使用领域中的术语所达成的共识。

后来的一些本体研究人员又归纳了许多面向具体操作的补充性规则，Arpirez[22]提出了以下三条比较著名的规则：

(1) 概念命名标准化

(2) 概念层次多样化

(3) 语义距离最小化

概念命名标准化可以以增强本体的适应性，便于本体的共享。概念层次多样化可以有效支持本体的多继承机制能力；语义距离最小化是指将内含相近的概念提取出来，使用同一元语进行表示。

以上的几个规则较为抽象，仅对本体的构建起到大致指导性作用。而实际的本体构建过程，可在大致参考以上几个原则的基础之上再依据本项目的实际情况进一步深化。虽然当前还没有统一的本体构建方法与方法性能评估的标准，但是构建领域本体时需要领域专家的参与是本体领域公认的。

## 3.3 数据质量本体构建流程

数据质量与本体发展到现在，已经有很多种的构建方法了。不过，目前仍然不能实现本体构建的完全自动化，各个本体的研发机构也都采用自己的本体构建机制与方法。下面介绍几个在学术界中比较被普遍接受的本体构建方法。

IDEF-5方法

本文中采用的就是IDEF-5方法，它是在结构化分析方法的基础上发展起来的。它的主要思想是通过使用图表语言和细化说明语言，获取关于客观存在的概念、属性和关系，最终形式化为本体。它的创建步骤如下图所示：

图3-1 IDEF-5方法

定义课题、组织队伍

收集数据

本体优化与验证

本体初步开发

分析数据

(1) Skeletal Methodolody骨架法

骨架法是专门用来创建企业本体的方法。它常用来充当商业企业间术语和定义的集合。它的创建步骤如下图所示：

图3-2 Skeletal Methodolody骨架法

不符合

确定本体的应用目的和范围

本体分析

本体表示

评价

本体评价

建立本体

符合

(3) TOVE企业建模法

TOVE法是加拿大Toronto大学企业集成实验室开发的应用于商业过程和活动建模领域的本体开发模式。

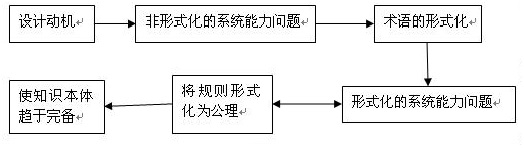


图3-3 TOVE企业建模法

(4) Methontology方法

该方法是由GOMEZ-PEREZ法和骨架法结合而产生的一种本体构建方法，这是一种更加具有普遍性的构建方法。它将本体开发进程和本体生命周期分开为两个方面，用不同的两个技术予以支持，形成类软件工程的开发方法。它的创建步骤如下：

管理阶段：用于规划任务进度，确立所需资源，以及如何保证质量等问题

开发阶段：分为五个步骤，分别为规范说明、概念化、形式化、执行以及维护

维护阶段：也分为五个步骤，包括知识获取、系统集成、评价、文档说明和配置管理

(5) 七步法

七步法是由斯坦福大学医学院开发的本体构建方法。该方法由七个步骤组成：确定本体的专业领域和范畴；是否可以复用现有本体；列出本体中的重要术语；定义类和类的等级关系；定义类的属性；定义属性的分面；创建实例[3]。

当然了，数据质量本体的构建方法并不局限于此。不管选用哪种本体构建方法，可扩展性、术语清晰、推理一致、最小编码、本体约定最小都是领域专家在构建本体时应遵循的基本规则。具体情况下，也要根据自己所在的领域，追求的目的等各方面因素，选择或者开发最有效的本体构建方法。不过，考虑到成本的问题，在有时虽不是最有效但不存在较大问题的情况下，可以复用现有本体。只是一种极大节约成本且行之有效的方法。

## 3.4 数据质量本体类以及属性

通过protege创建数据质量本体类，部分属性如下：

表3-1 本体类

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名称 | 对象属性 | 数据类型 | 定义域 | 值域 |
| HasMorethan | 是 | String | DATACOLUMN | DATACOLUMN |
| HasLessthan | 是 | String | DATACOLUMN | DATACOLUMN |
| HasExist | 是 | String | DATACOLUMN | DATACOLUMN |
| Equals | 是 | String | AssessedColumn | AssessedColumn |
| GreaterThan | 是 | String | AssessedColumn | AssessedColumn |
| NotEquals | 是 | String | AssessedColumn | AssessedColumn |
| LessThan | 是 | String | AssessedColumn | AssessedColumn |
| NOTNULL | 否 | Boolean | DATACOLUMN | Boolean |
| TABLE\_ID | 否 | String | DATATABLE | String |
| COLUMN\_ID | 否 | String | DATACOLUMN | String |
| COLUMN\_NAME | 否 | String | DATACOLUMN | String |
| DATA\_TYPE | 否 | String | DATACOLUMN | String |

## 3.5 数据质量规则对在本体中的描述

数据质量本体是对数据质量的评估规则构建一个本体，数据质量的具体评估方法还没有系统的评估框架。所以本文在进行数据质量本体构建时，将抽取数据质量评估规则中具体代表性的数据质量评估规则，具体的数据质量评估规则如下所示：

1. 非空约束：非空约束是完整性约束的一个部分，非空约束要求数据表中某个字段的取值不能为空。

表3-2 非空约束

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 数据属性 | 定义域 | 值域 |
| 1 | NOTNULL | DataColumn | Boolean |

1. 值域约束：值域约束要求数据表中某个字段的取值必须在一定的范围内。

表3-3 值域约束

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 数据属性 | 定义域 | 值域 |
| 1 | MIX\_VALUE | DataColumn | Float |
| 2 | MAX\_VALUE | DataColumn | Float |

1. 等值函数依赖约束：等值函数依赖规则表示数据表中的数据，在同一记录上，该数据项上的数据必须（或在满足某一条件时必须）由另一个或多个数据项上的数据通过函数计算得出。

表3-4 等值函数依赖约束

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 对象属性 | 定义域 | 值域 |
| 1 | Equals | AssessedColumn | AssessedColumn |

1. 逻辑依赖约束：逻辑依赖约束表示数据表中的数据，在同一记录上，该数据项上的数据必须（或在满足某一条件时必须）与另一个或多个数据项上的数据满足某种逻辑关系（如大于、小于等）。

表3-5 逻辑依赖约束

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 对象属性 | 定义域 | 值域 |
| 1 | LessThan | AssessedColumn | AssessedColumn |
| 2 | GreaterThan | AssessedColumn | AssessedColumn |
| 3 | NotEquals | AssessedColumn | AssessedColumn |

1. 等值一致性依赖约束：等值一致性依赖约束表示数据表中的数据，数据项上的数据必须（或在满足某一条件时必须）由另一个或多个数据表中的一个或多个数据项上的数据计算得出。

表3-6 等值一致性依赖约束

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 对象属性 | 定义域 | 值域 |
| 1 | ReferenceEquals | AssessedColumn | ReferenceColumn |

1. 逻辑一致性依赖约束：逻辑一致性依赖约束表示数据表中的数据，该数据项上的数据必须（或在满足某一条件时必须）与另一个或多个数据表中的一个数据项上的数据满足某种逻辑关系（如大于、小于等）。

表3-7 逻辑一致性依赖约束

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 对象属性 | 定义域 | 值域 |
| 1 | ReferenceGreaterThan | AssessedColumn | ReferenceColumn |
| 2 | ReferenceLessThan | AssessedColumn | ReferenceColumn |

1. 存在一致性依赖约束：存在一致性依赖约束表示数据表中的数据项必须（或在满足某一条件时必须）在另一个或多个数据表中的一个数据项上的数据中出现。

表3-8 存在一致性依赖约束

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 对象属性 | 定义域 | 值域 |
| 1 | HasExist | DataColumn | DataColumn |

第4章 基于Jess的数据质量评价方法

## 4.1 基于SWRL的Jess推理

4.1.1 SWRL概述

由于本体本身并不支持没有用于推理的规则，而数据质量评价需要加入支持更广泛的推理的规则，所以我们引入SWRL。它是以语义的方式呈现规则的一种语言。SWRL 规则呈现为前提和结论之间的蕴涵式，主要作用于OWL的实例及其联系，一般形式为：[7]。

4.2.2 Jess与Protege的集成

在Protege上安装JessTab插件后就完成了集成。它可以完成下面四个任务:①将SWRL的规则转换为Jess的规则或直接在Jess中基于本体概念创建规则；②将有关OWL个体的知识表示为Jess的事实；③基于规则和事实运行Jess推理引擎进行推理；④用推理后得到的新事实更新OWL知识库。工作过程如下图所示[7]。

类、属性

属

性

值

SWRL规则

个体

(实例)

本

体

知识库

规则

事实

Jess

推

理

引

擎

新事实

(1)

(2)

(3)

(4)

## 4.2 熵权法在本体中的设计实现

4.2.1熵权法用于数据质量评价的基本原理

熵权法是在已知对象集合各评定指标值的基础上，计算出在竞争意义上的激烈程度[6]。熵是对混乱程度的度量。在数据质量评价中，随着信息量的增加，不确定的因素就会越少，即熵就会越小；反之熵就会越大。 熵值为1即不存在有价值的信息，其熵权则对应为0，这就表示取消该指标。熵权的取值范围是[0,1]，熵权越大表示该指标越为重要，对最终结果的影响也越大。我们用表示第个指标的熵权，表示指标总数，则。

4.2.2熵权法的具体设计步骤

1. 首先要确立的，就是熵权法的评价对象，即为指标水平矩阵。矩阵

 （4-1）

其中表示第个评定对象的第个指标的数值即对应位置的重要性，表示横坐标的长度即评价规则的个数，表示纵坐标的长度即指标字段（本体待评价实例）的个数。

(2) 异质指标同质化处理。进行同质化是因为存在不同指标之间计量单位不同的情况。在我们进行综合指标计算之前，要先进行解决标准化处理的问题。由于正向指标与负向指标代表的含义不同，所以将正负指标分开处理。公式如下：

正向指标：

 （4-2）

负向指标：

 （4-3）

(3) 计算第项指标的熵值。

为了使层次分析法和熵权法（在下面解释）同时适用，对作如下处理：

 （4-4）

处理之后的公式如下：

 （4-5）

式中：，，

(4) 计算评价指标的熵权。公式如下：

 （4-5）

式中：

4.2.3计算综合指标权重

熵权法极大地利用了原始数据中所存在的信息，客观的反映了结果。但是它的缺陷别是不能结合专家的经验与知识，以及决策者的侧重点，造成结构不够切合实际需求。而层次分析法恰好与之相反，是以专家的知识和经验、决策者的侧重点为核心，导致主观随意性过高的问题。综合来看，将两者结合起来使用就能达到理想的效果。

 （4-5）

式中：

4.2.4确定综合结果

通过加权由综合权重计算出综合结果

 （4-5）

## 4.3 数据质量的jess推理实现

1. 完整性  
   完整性（Integrity）用来描述数值信息的完整程度。完整性包括完整性约束规则、非空约束规则、连续性约束规则三个维度的规则。
2. 一致性

一致性（Consistency）用来描述同一记录在多个数据集上存储的同一变量的一致程度 。一致性包括三层含义：一种是数据集之间的一致性关系，是指数据集中的某些属性与其他数据集的属性集合的数据存在等值关系或逻辑关系；一种是数据集内部的一致性关系，也就是数据集的属性之间应该满足等值或逻辑关系；最后一种是数据集之间的存在关系，具体指表之间的父子引用关系。一致性包括等值一致性依赖约束规则、存在一致性依赖约束规则、逻辑一致性依赖约束规则三个维度的规则。一致性的三种含义对应三个维度的约束规则。  
 3. 准确性

准确性（Accuracy）是描述待测数据与真实数据的符合程度，数据的准确性体现在数据内容和形式上的准确，即数据的类型、格式、精度及值域范围的准确[22]。例如在企业的实际生产中，准确性往往是指在对数据的操作过程中采集的数据与真实值之间的差异，值的误差越小准确性越高。  
影响准确性的原因很多，在数据的录入、修改、审核、传输每个环节都有可能出现。我们需要从产生误差的原因来分析影响准确性的因素，使误差尽可能的控制在理想的范围内，提高数据的准确性。准确性包含值域约束规则、代码约束规则、逻辑依赖约束规则、词法约束规则、非空约束规则、等值函数依赖约束规则六个维度。

第5章 系统实现

## 5.1 数据质量本体

在做数据质量评价之前，先要创建基本的数据质量本体。

（1）打开protege，可以看到如下小窗口，我们这里选择new project；

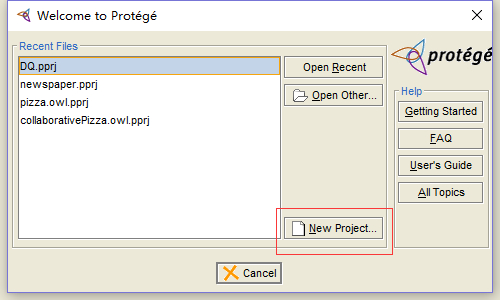


图5-1 创建新工程

（2）选择OWL/RDF Files；

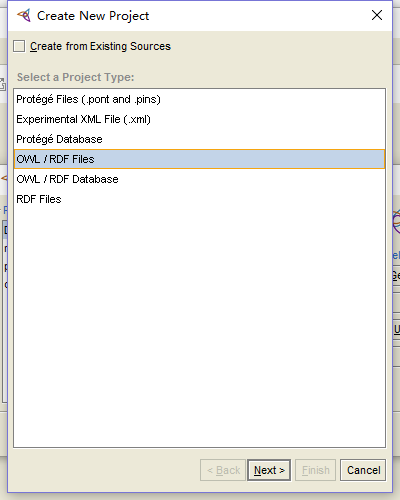


图5-2 选择文件类型

（3）填写本体URI，默认即可；

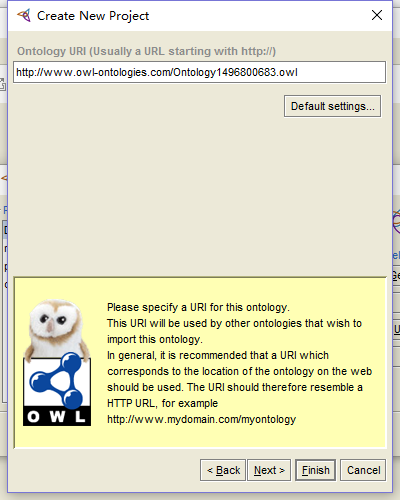


图5-3 填写本体URI

（4）选择文件类型，此处我们选择OWL Full类型；

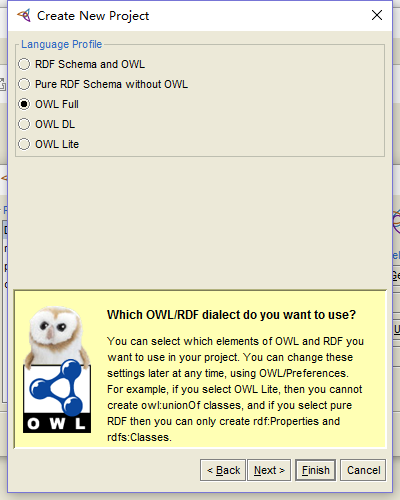


图5-4 选择文件类型

（5）选择视图，可不必在意，创建后可在主界面右下角随意切换。

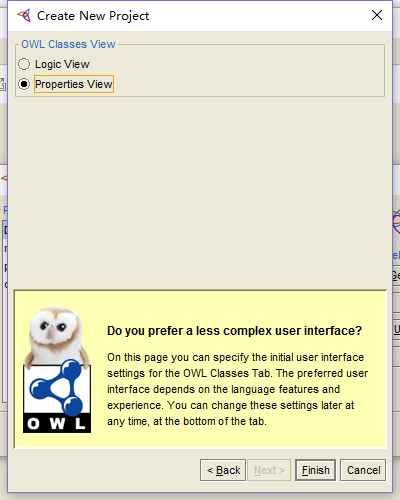


图5-5 选择视图

（6）通过Protege创建class。新建本体后切换到OWL Classes标签下，右键owl:Thing，选择Create subclass。

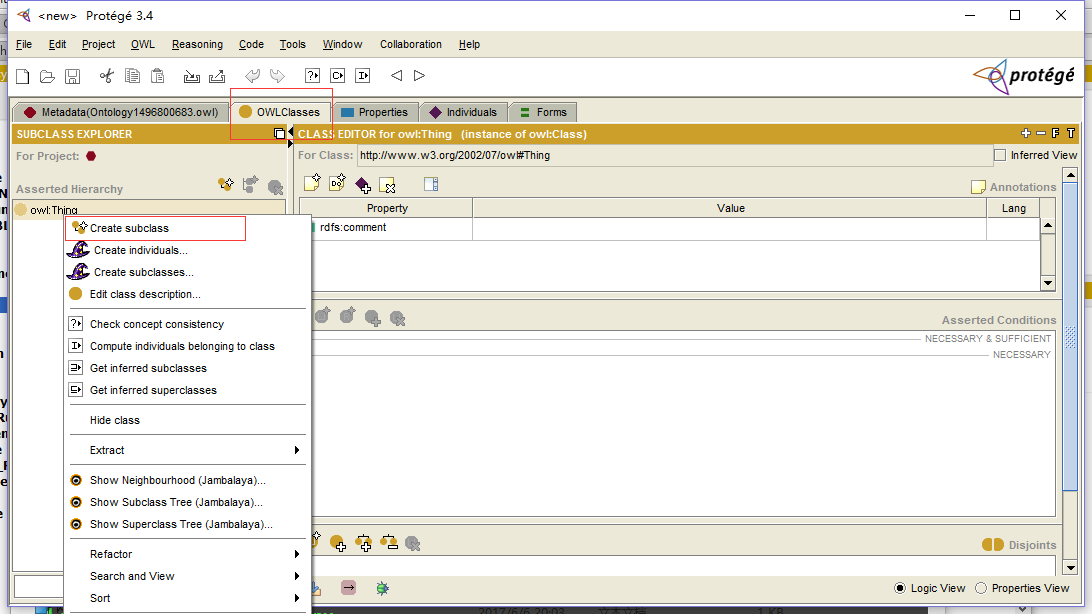


图5-6 通过Protege创建class

（7）通过Protege编辑class。创建class后双击创建的class即可进入编辑界面，如图5-7。

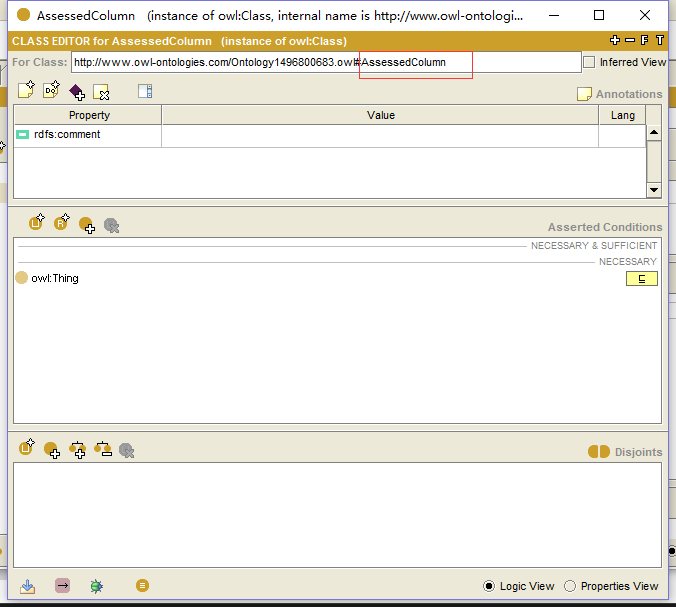


图5-7 通过Protege编辑class

1. 通过Protege创建Properties。选择Properties标签
2. 通过Protege创建Object Properties
3. 通过Protege创建Data Properties
4. 通过Protege创建Individuals

## 5.2 熵权法的实现

## 5.3 数据质量评价实现

第6章 系统测试

## 6.1 测试目标

软件工程是一个范围广泛的概念和知识[11]，其中重要的一项正是软件测试。测试的目的决定了测试方法。如果测试的目的是为了尽可能多地找出错误，那么测试就应该直接针对软件比较复杂的部分或是以前出错比较多的位置进行测试。例如在过去的几年中，许多安全研究人员提出要为Web平台提供更加严格的标准，从而可以准确地推理Web安全问题[12]，那么就应该针对安全问题专门进行测试。

如果测试目的是为了给最终用户提供具有一定可信度的质量评价，那么测试就应该直接针对在实际应用中会经常用到的假设进行测试。软件测试是为了发现错误而执行程序的过程。就算没有发现错误的测试也是有价值的，完整的测试是评定测试质量的一种方法。详细而严谨的可靠性增长模型可以证明这一点。

## 6.2 测试方法

软件测试的方法分为：白盒测试和黑盒测试。其中白盒测试是对软件过程性细节做出细致的检查。这样的方法是把测试对象看作是一个打开的盒子，它允许测试人员利用程序内部的逻辑结构以及有关的信息，设计或者选择测试用例，对程序所有的逻辑路径进行测试。通过在不同的节点来检查程序的状态，确定实际状态是否与预期达成一致。因此，白盒测试又称为结构测试或者是逻辑测试。

而软件的黑盒测试意味着测试要在软件的接口处进行测试。这样的方法是把测试对象看成一个黑盒子，测试人员完全不考虑程序内部的逻辑结构和内部属性，只是根据程序的需求规格说明书，检查程序的功能是否符合它的功能说明，因此黑盒测试又叫功能测试或者数据驱动测试。黑盒测试主要是为了发现以下几类错误[13]：

1．是否有不正确或遗漏的功能

2．在接口上，输入是否正确的接受，能否输出正确的结果

3．是否有数据结构错误或外部信息（例如数据文件）访问错误。

4．性能上是否能够满足要求。

5．是否有初始化或者终止性错误。

## 6.3 测试用例

结论

参考文献

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

[6] 李汶静．基于熵权ＡＨＰ法的企业价值组合评估应用研究［D］．成都：西南石油大学，2010．

[7] 纪兆辉、李存华.基于SWRL和Jess构造语义Web规则及其对策分析.淮海工学院学报 第18卷第4期

[11] Nuno H. Flores, Ana C.R. Paiva, Pedro Letra. Software Engineering Management Education through Game Design Patterns[J]. Procedia - Social and Behavioral Sciences Volume 228, 2016, 7: 436-442.

[12] Michele Bugliesi , Stefano Calzavara, , Riccardo Focardi. Formal methods for web security[J].Journal of Logical and Algebraic Methods in Programming Volume 87, 2017,2:110–126

[13] 万年红,李翔.软件黑盒测试的方法与实践[J]. 计算机工程,2000,(12):91-93+164.

致谢