摘　要

Web 作为一个全球化信息空间，蕴含着巨大的潜在价值，如何在庞杂的数据中准确地抽取出用户想要的信息成为一个非常重要的课题。尽管目前已对 Web 数据抽取技术进行了大量的研究工作，但是现有的技术缺乏对数据本身的描述，不含清晰的语义信息，模式也不太明确，难以适应各个网站的结构各异，形态多样的特点。这使得应用程序无法直接解析并利用 Web 上海量的信息，造成资源极大的浪费。

针对上述问题，本文在结合知识密集型 Web 站点的数据特征的基础上，引入 Suffix Tree 技术，根据不同网站的格式特征，提取出有效的数据模式，并利用基于语义的本体建立方法，借助 Protégé 工具建立领域本体，完成信息抽取过程中语义信息的扩展，消除了同类信息源 Web 页面的异构性。

本文在研究了本体技术和半结构化 Web 信息抽取技术总体解决方案的基础上，着重研究了知识密集型 Web 站点的信息数据抽取的实现技术。通过对传统信息抽取方法的基本原理、技术及发展现状等方面的分析研究，提出了由本体驱动，并根据文档结构和特征匹配来进行信息定位和信息抽取的模型，详细描述了该模型的设计思想和抽取流程。

该系统首先获取指定的 HTML 格式的 Web 文档，根据基于栈结构与链式结构的 HTML 到 XML 文档转换算法，将 Web 页面转换为 XML 格式，从而解决了 Web 文档之间的异构问题；然后利用 Suffix Tree 技术从该 XML 文档中提取数据模式；同时利用基于语义的本体建立方法，为这些信息增加语义信息；并用本体描述语言 OWL 形式化地描述该领域本体，从中归纳抽取规则；最后将抽取出来的数据转换成具有语义的 RDF 数据模型。

论文通过本体技术的应用实现了语义信息附加，利用 Suffix Tree 技术完成了 Web 页面结构的数据模式提取。其工作实现了知识密集型 Web 站点上的信息数据源的模式提取以及信息抽取模型，方便用户使用有价值的 Web 信息资源，同时也为充分利用 Web 上的海量数据提供了一个有效的工具。

关键词： Suffix tree；语义；Web 信息抽取；本体；XML

Abstract

As a global information space, Web contains tremendous intrinsic value, how to extract the information that user need exactly from complex data becomes a very important issue. Although a great deal of research have been carried out for web data extraction, existing technology is lack of description to data itself and never contains clear semantic information, pattern is not specific neither, which is difficult to fit the web's characteristic of diversity in structure and pattern, which makes application program cannot analysis and make use of the mass information on web directly which causes huge waste.

This article introduces Suffix Tree technology coupling with data characteristic of a knowledge intensive web site, extracts available data pattern, creates domain Ontology with Protégé tools realizes semantic information expansion in the process of information extraction and eliminates the isomerism of homogeneous message source of web site by use of method for establishing based on semantic.

This article emphasizes the implementation technique of information data extraction of knowledge intensive web site based on the overall solution of Ontology technology and Semi-Structured web information extraction technology. This article puts forward an Ontology-driven information extraction pattern information-positioning by file structure and Feature Matching via analysis and research on fundamental principle of conventional method of information extraction, technology and development status, details design thinking of the pattern and the flow of extraction.

This system resolves the isomerism problem among the web files by gaining the specified HTML web page in the first place, transferring the web page to well-formed XML file based on the file Converting arithmetic based on Stack Structure and link Structure, then extracting data pattern from the XML file with Suffix Tree technology, increasing semantic information for these information

by use of Ontology establishing method, makes a formal description for the domain Ontology with OWL which is Web Ontology Language, generating extraction rule-base, realizes the transition from data extracted to rdf data model which containes semantie information.

This thesis realizes semantic information affixation via application of Ontology, finishes data pattern extraction of web site structure by use of Suffix Tree technology. The job realizes the pattern extraction of information data source on knowledge-intensive web site, which can help user discover valuable information resource on web and provides a effectual tool to make use of the mass data on web at the same time.

Key words: Suffix tree; Semantie; Web information extraction;Ontology; XML

哈尔滨工程大学
学位论文原创性声明

本人郑重声明：本论文的所有工作，是在导师的指导下，由作者本人独立完成的。有关观点、方法、数据和文献的引用已在文中指出，并与参考文献相对应。除文中已注明引用的内容外，本论文不包含任何其他个人或集体已经公开发表的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

作者（签字）：黄文利

日期：2009年3月13日

哈尔滨工程大学
学位论文授权使用声明

本人完全了解学校保护知识产权的有关规定，即研究生在校攻读学位期间论文工作的知识产权属于哈尔滨工程大学。哈尔滨工程大学有权保留并向国家有关部门或机构送交论文的复印件。本人允许哈尔滨工程大学将论文的部分或全部内容编入有关数据库进行检索，可采用影印、缩印或扫描等复制手段保存和汇编本学位论文，可以公布论文的全部内容。同时本人保证毕业后结合学位论文研究课题再撰写的论文一律注明作者第一署名单位为哈尔滨工程大学。涉密学位论文待解密后适用本声明。

本论文（□在授予学位后即可　□在授予学位12个月后　□解密后）由哈尔滨工程大学送交有关部门进行保存、汇编等。

作者（签字）：黄文利　　导师（签字）：

日期：2009年3月13日　　2009年3月13日

第1章 绪论

1.1 选题背景及意义

随着计算机技术、通讯技术的飞速发展和个人计算机的普及，Internet作为一个全球网络，越来越融入到人们的生活、工作、学习、商务活动中去。企业及个人通过建立网站或网页及时发布自己的信息、资源、需求，同时又通过网络来寻求帮助，获取信息。Web给我们的工作学习带来了很多的便利：首先它是快速便捷的。一台电脑，一根网线，用户就能够自由遨游网络，浏览新闻、发布信息，真正做到“不出户而知天下事”；其次它是信息共享的。Internet在提出时，就讲究资源共享，这样大大加快了信息交流，知识的传播速度；同时它也是内容丰富的。Web上的内容五花八门，包罗万象，上至天文，下至地理，即使是个人的随心所想，人生感悟也是应有尽有，这是其他信息载体所无法比拟的；最后它还是互动互惠的。所有其他的信息载体，如报纸杂志、电视，它们都是单向的，用户只能接受或不接受，而广域网却给我们带来了互动性，用户可以通过它发布问题、回答问题、甚至是在线交流。

Web在给我们带来便捷、快速、廉价、丰富的信息的同时，也给我们带来了一个问题。由于越来越多的企业和个人通过Web发布信息，使得Web上的信息量以指数级的速度在增长，Web上浩大的信息量和用户的需求之间产生了严重的不平衡和矛盾，用户为了获取自己需要的一点点信息，可能需要花费几十分钟、几个小时、甚至更长时间来搜索网页，查找信息，一不小心就会淹没在信息的海洋中。虽然现在出现了形形色色的搜索引擎，但是这种基于关键字的搜索，先要了解相关网页内容的概要，得到的却还是一个庞大的结果集，这个结果集只是给出具体的信息还是要用户进入到具体网页中查找，这和具体的搜索引擎的性能有关，同时这样就浪费了大量的人力、物力和时间。即使获得了相关内容的网页，如何将这些网页中有用的信息抽取出来加以保存，作为自己信息库中的信息，也不是一件简单事情。

上述问题的关键在于Web信息的发布与浏览都是通过基于HTML或语法的页面实现的，而HTML或XML是非结构化、半结构化的语言，它们无

法被计算机所理解，也无法像传统的数据库那样，提供结构化的、功能强大的、高效的查询语句。如何让计算机从 Web 数据源中获取用户所需的信息，这正是信息抽取的任务[15]。

信息抽取技术的核心是能够从 Web 页面所包含的无结构或半结构的信息中识别用户感兴趣的数据[55]，并将其转化为更为结构化、语义更为清晰的格式。但是目前 Web 上的数据大部分都是以 HTML 形式出现的，主要目的是为了显示，让人通过浏览器浏览，缺乏对数据本身的描述，不含清晰的语义信息，模式也不太明确。这使得应用程序无法直接解析并利用 Web 上海量的信息，造成资源极大的浪费，而信息抽取技术不但可以直接定位到用户所需的信息，而且采用一定的方式为这些信息增加语义和模式信息，为 Web 查询提供一种更为精确的方法，为 Web 中信息的再利用提供了可能。为了更好地解决 Web 信息抽取面临的诸多问题和不足，有必要对 Web 信息抽取问题作进一步研究。

1.2 信息抽取技术的研究现状

随着互联网的不断发展，从 Web 上抽取和集成信息逐渐成为研究热点。目前的信息抽取方法主要是通过手工方式或者自动半自动方式生成抽取规则，然后由软件智能体依据这些抽取规则从页面中抽取数据。

文献[1]是 TSIMMIS 数据集成项目的一部分，它设计了一种声明型的高级语言书写模板来定义抽取规则，描述包装器将接收的查询语言和将要返回的对象。当查询与某模板匹配时，从属于该模板的行为被触发，将集成系统的查询转换为面向数据源的查询，并将返回结果翻译为集成系统能够识别的形式。因为该方法必须适应整个集成系统设计需要，所以不具有普遍适用性。

文献[2]采用用户输入页面描述文件对层次结构进行抽取。该描述文件需要用户描述抽取过程的具体变量并编写抽取方法，只适用于某种特殊的页面。

文献[3]采用正则表达式来定义抽取规则，对论文按照结构顺序依次抽取。该方法要求论文必须书写规范，且只适合于论文文献，对其他类型的文献则需设计另外的抽取规则。

文献[4]是最早把归纳学习方法引入包装器的。通过分析 Web 数据源，识

别出一类可以通过高效学习得到并对实际数据源具有足够表达能力的包装器HLRT(Hypertext Learning Rules Transform)。但HLRT主要针对那些以表格布局显示内容的数据源，有其局限性。

文献[5]采用启发式方法，按照各个部分字体大小和缩进距离推导出页面的层次结构。该方法对于没有标出字体大小和缩进距离的部分无法抽取。

文献[6]提出了一种从产品页面中自动归纳抽取数据模式的方法。该方法能有效地去除无效数据区域，得到结构良好的XML文档，然后从XML文档中归纳出页面的数据模式。但该方法在数据模式发现过程中需要很大的计算量，要多次计算页面的内容相似度和DOM子树的创新度，抽取过程较复杂。

文献[7]给出了一种基于本体论的抽取方法，能够抽取用户感兴趣的信息。但该方法首先需要一个完备的领域本体的参与，并且需要人工构造抽取规则，自动化程度较低。

文献[8]使用基于XML的WMA(Web mining agent)从页面中抽取数据，WMA需要一种专门的描述语言来表示和识别隐藏在HTML页面中的数据，最后把抽取的数据存放在数据库中。该方法虽然能够有效地从页面中抽取数据，但是需要用户首先学习WMA描述语言，使用上带来了很大的不便。

目前的抽取方法存在的主要问题如下：

1. 系统的可移植能力有待进一步提高，目前的信息抽取系统主要针对特定的某一领域，如果应用到其它领域则要重新设计抽取模式。如文献[4]主要用于处理表格数据，而文献[5]不能处理文档中的表格和列表。

2.不同的页面需要编写不同的包装器。由于数据抽取过程与具体的页面结构缠杂在一起，因此当页面的结构发生变化时，就需要重新编写新的包装器，使得包装器不具有重用性。

3.开发和使用包装器程序需要较高的技巧，如文献[2]输入的页面描述文件需要用户对其输入方法有清晰的理解，因而不利于数据集成系统的推广。

4.不能一次提取多个有效数据。如文献[7]只能一次提取一篇论文的信息，如果页面中含有多篇论文，不能一次提取出来。

5.在一个新的领域上建立抽取系统时自动化程度太低，需要众多领域专家和计算机语言学家的共同努力，既费时又费力。

针对当前各种包装器存在的不足，为了将用户需要的所有数据都抽取出

来，并且构造出适合于各种不同页面的抽取方法，本文以本体论为依托，采用用户自定义的本体作为数据抽取格式，并结合启发式规则进行信息抽取。本文方法的目标是：提供一种对于使用者来说简单、易操作的页面描述方法，对于各种页面中的结构化和半结构化数据都统一处理，并能把用户感兴趣的信息尽量多地抽取出来。

1.3 本文研究内容

根据对 www 的统计，设计者在设计网站的组织结构时，为了便于组织管理，他们往往会根据人对知识的理解而将网站内容进行分类，从而将主题相同的页面放在 Web 服务器的同一个目录下，或根据主题层次组织成树形目录，这些目录和页面的层次结构将映射到具体页面以及目录的 URL 上。即使在同一个页面中，信息的组织也有其规律性一簇聚性，就是内容含义相同或相似的信息组织在一起，在页面上体现为占据某一块页面。另外，对于大型的网站，例如购物网等，发布的网页大部分是通过相应的后台数据库生成、以 HTML 文档的形式来显示的，使得内容网页的组织结构具有很大的相似性，这类网站通常称为数据密集型网站。这些结构类似的网页所共有的组织结构称为模板。本文研究的切入点就是这种网页结构类似的数据密集型网站。

本文的研究目标是：对于网页结构类似的信息密集型 Web 站点，系统可以自动解析出这种 Web 站点上页面的数据特征，构造出一种新的基于本体的信息抽取模型，从而利用本体库对同类网页进行信息抽取。该模型能够实现较高的自动化程度，只需少量的人工干预即可达到抽取规则的自动生成。

具体工作包括：第一，介绍了当前信息抽取的主要方法，分析和比较了各种方法中的代表性抽取工具，并总结比较当前信息抽取技术的特点，并指出其中的不足。第二，分析当前的 HTML 到 XML 转换的主要方法，提出了一种新的转换算法，该算法是基于栈结构与链式结构的，它完成了将大量的 HTML 文档转换为 XML 的形式方便地形成 XML 数据源，消除了 HTML 格式中不严格语法，为下一步的信息抽取作有效的铺垫。第三，介绍了 Suffix Tree [7]技术，完成了样本页面的 Suffix Tree 构造，给出了从 Suffix Tree 中归纳学习页面的数据模式的方法。第四，根据图书领域中知识密集型 Web 站点

的数据概念，利用本体学习的方法和 Protégé 工具，构造一个简单的图书领域本体，最后转化成 OWL 文件，从中提取抽取规则。最后，从知识密集型 Web 站点“当当图书网”中下载部分页面作为实验对象，抽取相应的信息保存到数据库中，对模型进行测试和分析。

总之，本文把本体技术，XML 技术与信息抽取结合在一起，提出了一个基于本体的 Web 页面结构化信息抽取模型。模型首先将 HTML 文档转换成 XML 格式，并从该文档中归纳学习出 Web 页面的数据模式，然后构造了一个图书领域本体，进而从领域本体中归纳学习抽取规则，指导具体的抽取动作。本体技术的引入提高了系统的效率、可扩展性和可移植性，克服了当前信息抽取方法的缺点。

1.4 论文结构

全文共分五章，各章的内容概括如下：

第 1 章是绪论，介绍课题提出的背景及意义，分析了信息抽取的目前的研究状况，给出了主要内容及结构组织。

第 2 章介绍信息抽取的有关工作基础，包括对当前抽取技术的分析和概括，以及各种信息抽取技术的特点比较；同时，介绍了 XML 的基础知识介绍，给出了 XML 在信息抽取系统中的优势；最后，介绍了本体的基础知识以及本文所采用的信息抽取方法，为以后章节铺开思路。

第 3 章给出了信息抽取系统的框架结构，详细介绍基于本体的信息抽取模型各阶段的主要任务。首先，说明了 HTML 到 XML 文档解析的必要性；接着，介绍了模式提取中涉及到 Suffix Tree 技术；然后，介绍了基于语义的本体构造过程；最后，给出了信息抽取的规则和 RDF 数据模型的生成方法。

第 4 章对 HTML 格式到 XML 格式的转换方法进行了分析和总结，提出了基于栈结构与链式结构的 HTML 到 XML 的转换算法。

第 5 章介绍了模式提取的方法，提出了基于后缀树的数据模式提取算法，完成了备选数据模式集的生成和有效数据模式的提取。

第 6 章基于前四章的理论研究，开发了一个快速、通用的 Web 信息抽取原型系统，并给出了运行结果。

第 2 章 相关工作基础

从 Web 上的信息抽取技术是目前热点的研究项目。这项技术处于不断地更新和发展中，目前 Web 信息抽取研究的重点之一，是探索怎样能够能够从 Web 页面所包含的无结构或半结构的信息中识别用户感兴趣的数据，并将其转化为更为结构化、语义更为清晰的格式。本章 2.1 节从方法的原理出发，介绍几种 Web 信息的抽取技术，并分析这几类技术。

本文研究的信息抽取方法涉及到 XML，本体及其相关标准技术，这些标准技术是本文工作依靠的技术基础，本章 2.2 与 2.3 节介绍这些相关的技术。

2.1 Web 信息抽取技术概述

随着计算机的普及以及互联网的迅猛发展，大量的信息以电子文档的形式出现在人们面前。为了应对信息爆炸带来的严重挑战，迫切需要一些自动化的工具帮助人们在海量信息源中迅速找到真正需要的信息。信息抽取技术的研究正是在这种背景下产生的。

2.1.1 Web 信息抽取技术

1. 信息抽取技术的产生与发展

从自然语言文本中获取结构化信息的研究最早开始于 20 世纪 60 年代中期，这被看作是信息抽取技术的初始研究，它以两个长期的、研究性的自然语言处理项目为代表[19]。

美国纽约大学开展的 Linguistic String 项目[20]开始于上世纪 60 年代中期并一直延续到 80 年代。该项目的主要研究内容是建立一个大规模的英语计算语法，与之相关的应用是从医疗领域的 X 光报告和医院出院记录中抽取信息格式，这种信息格式实际上就是现在所说的模板[57]。

自动信息抽取技术则是近二三十年才发展起来的。有两个因素对其发展有重要的影响[56]：一是在线和离线文本数量的几何级增加，另一个是“消息理解研讨会”(MUC)近十几年来对该领域的关注和推动。这些系统通常只在

很窄的知识领域范围内运行良好，向其它新领域移植的性能却很差。

中文信息抽取方面的研究起步较晚[56]，主要的研究工作集中在对中文命名实体的识别方面，在设计实现完整的中文信息抽取系统方面还处在探索阶段。

2. 信息抽取系统的功能

信息抽取系统的主要功能是从文本中抽取出特定的事实信息并且将其形成结构化的数据填入一个数据库中供用户查询使用。通常，被抽取出来的信息以结构化的形式描述，可以直接存入数据库中，供用户查询使用。

信息抽取系统的实现一般有两种方法[59]：一是知识工程方法：由专家对语料库进行分析、调整，从而人工制定规则、模板。比如对命名实体的识别，可以采用基于规则的方法，采用有限状态自动机来实现。另一个就是自动训练方法：给出标注的例子文档集，通过机器学习来推导模板和模板的自动填充规则，也可以应用统计学的方法来抽取，比如中文人名的抽取。

本文主要采用自动训练集的方法。

3. 信息抽取系统的模型结构

信息抽取通常要从对样本文档的分析开始，通过解析样本文档，提取出页面的数据模式，然后得到抽取规则，满足质量标准的抽取规则组成一个知识模式库，用来抽取有效数据变成结构化信息。最后要对整个抽取系统进行质量评价，分析信息抽取系统的性能。常见的信息抽取模型如图 2.1 所示。

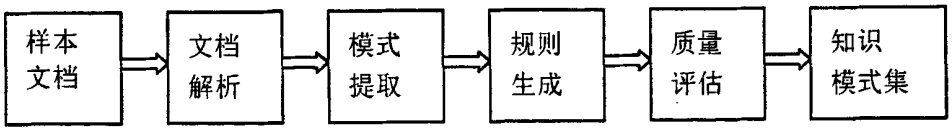


图 2.1 常见的信息抽取模型

其中，样本文档是信息抽取系统的输入，一般为随机抽取的几个样本文档的集合；文档解析就是对输入的样本文档的源码进行解析，分析出各个数据项的源码构成，并用一定形式化的形式表达出来；模式提取阶段是信息抽取的核心部分，完成对文档内容或结构的分析与理解，从样本文档中分析数据的分布架构，发现文档中隐含的数据模式，进而构建对应的数据模型，从这些数据分布架构中抽取出有用的数据概念，得到相关的数据模型；规则生成可看作一个规则生成器，根据文档结构的形式化表达进行归纳学习，生成

抽取规则；质量评价是对生成的规则依据评价标准进行质量评估，评价标准主要有两个，即查全率和准确率；最后，满足质量标准的抽取规则要进入模式库存储，生成的抽取规则按一定的格式存储到知识模式库中，这些规则可以在以后的抽取过程中用来指导抽取动作。

2.1.2 现有 Web 信息抽取方法的技术路线分析与比较

传统的构造 Wrapper 的方式是手工编码，既费时费力、容易出错，还需要由专家完成，于是许多半自动化或自动化的方法被提出。对 Web 信息抽取方法的分类角度很多[60]，如根据自动化程度分为手工、半自动和全自动。根据方法的原理可分为基于包装器归纳、基于自然语言理解、基于 Ontology 方法和基于 HTML 方法等。本节从方法的原理出发，介绍几种 Web 信息的抽取技术，并结合目前较为典型的系统来分析这几类 Web 数据抽取方法。

1. 基于自然语言处理方式的信息抽取

自然语言处理技术通常用于自由文本的数据抽取，需要经过的处理步骤包括：句法分析、语义标注、专有对象的识别和抽取规则。

基于自然语言的信息抽取技术是将 Web 文档视为文本进行处理的，其缺点也较为明显：抽取的实现没有利用 Web 文档独特于普通文本的层次特性，抽取规则表达能力有限；获得有效的抽取规则需要大量的样本学习，达到全自动的程序较难，而且速度较慢，对于操作海量数据来说这是一个大问题。

2. 基于包装器归纳方式的信息抽取

包装器由一系列的抽取规则以及应用这些规则的程序代码组成。通常，一个包装器只能处理一种特定的信息源。从几个不同信息源中抽取信息，需要一系列的包装器程序库。

包装器归纳方式的信息抽取根据事先由用户标记的样本实例应用机器学习方式的归纳算法，生成基于定界符的抽取规则。采用这种原理的典型的系统有 STALKER[28,29]，WIEN[31]，SOFTMEALY[30]。下面根据具体的系统来详细分析这类信息抽取技术。

STALKER：该系统中语义的附加和模式的定义是在用户定义嵌入式分类树阶段。该系统是按结构抽取和按文本抽取的结合，所以可以抽取复杂的对

象。但是规则中的定界符不仅是由 HTML 标记组成，还有某类网页经常出现的关键字组成。所以该类信息抽取不但对页面的结构有所依赖，而且对网页的内容也有所依赖，要想获得精确的抽取规则，必须进行大量的样本训练。

3. 基于ONTOLOGY方式的信息抽取

该类信息抽取介绍主要利用对数据本身的描述信息实现抽取，对网页结构的依赖较少。由Brigham Yong University(BYU)开发的信息抽取工具[36]中采用了这种方式，另外QUIXOTE[37,38]也采用了这种方式。

BYU：该系统中语义的附加和模式的定义是在书写某一领域的ontology的时候完成的，即人工方式。系统最大的优点是对网页的结构依赖较少，只要创建的应用领域ontology足够强大丰富，系统可以对某一应用领域中的各种网页实现信息抽取。由于是根据数据本身实现信息抽取的，因此在减少了对网页的结构依赖的同时，增加了对网页中包含的数据结构的要求。

4. 基于HTML结构的信息抽取

该类信息抽取技术的特点是，根据 Web 页面的结构定位信息。在信息抽取之前，通过解析器将 Web 文档解析成语法树，通过自动或者半自动的方式产生抽取规则，将信息抽取转化为对语法树的操作实现信息抽取。采用该类介绍的典型系统有 LIXTO[39,40]，XWRAP[41,42]，Road Runner[43]和 W4F[45]等。下面对具有代表性的系统进行分析，详细的研究请参考对应的参考文献。

LIXTO：在该系统中，语义信息是在样本学习阶段，由用户加入的，采用了先模式的方式，事先由用户在可视化的界面中定义模式，抽取出的数据最终以 XML 格式存放。LIXTO 在一定程度上简化了信息抽取的步骤，增强了信息抽取技术实用性。不足之处在于：它的抽取规则使用基于 Datalog 的 Elog 语言描述的，实现和优化校困难，另外抽取规则中抽取信息的描述不够丰富，而且对网页中的超链接不作处理，不支持图像信息和文献信息的处理。

5. 基于Web查询的信息抽取

使用 Web 的相关技术解决 Web 的问题称为 Web 技术规范。上述的信息抽取工具，采用了不同的原理，抽取规则的形式和感兴趣信息的定位方式也各不相同，因此均不具有通用性。具有 Web 技术规范的信息抽取，将 Web 信息抽取转化为使用标准的 Web 查询语言对 Web 文档的查询，具有通用性。采用该类技术的典型的系统有：Web-OQL[32,33]以及自主开发的原型系统

PQAgent[34,35]。

PQAgent：该系统采用先模式的方式，由用户附加语义并确定模式。抽取规则以 XQuery 的形式表示。应用抽取规则可直接定位到对象。相对于前面的系统，该系统的抽取规则相当健壮，有很强的表达能力，并统一了 HTML 和 XML 查询，不仅便于最终用户使用，也便于作为包装器，由应用查询调用，这是其它方法无法比拟的优点。但是系统对于网页结构的依赖性仍较强。

6. 几种方法的比较

可见，不同的信息抽取系统针对不同领域使用了不同的技术路线，具有各自的特点。现在对它们的几个性能总结如表2.1所示。

表 2.1 不同信息抽取系统的性能对照表

抽取工具		自动化程度	复杂对象支持	GUI 界面	XML 输出	非 HTML 文档支持
基于 HTML 结构的工具	W4F	半自动化	No	Yes	Yes	None
	XWRAP	自动化	Yes	Yes	Yes	None
	RoadRunner	自动化	Yes	Yes	No	None
基于自然语言处理的工具	PAPIER	半自动化	No	Yes	No	Full
	WHISK	半自动化	No	Yes	No	Full
基于机器学习的工具	WIEN	半自动化	No	Yes	No	Partial
	SoftMealy	半自动化	Partial	Yes	No	Partial
	STALKER	半自动化	Yes	Yes	No	Partial
基于本体的工具	BYU-Ontos	人工干预本体构造	Yes	Yes	No	Full

根据以上的分析可知，目前各类信息抽取技术中生成规则的依据主要有以下3类：页面结构信息、页面的内容和自然语言语义、语法信息。

各类信息抽取技术中语义的附加方式有多种。基于自然语言理解方式的信息抽取中在一定程度上可以通过自然语言语法、语义获得抽取出的信息的语义，但要取得较好的效果不易；全自动的信息抽取根据页面中的HTML标记间的关系抽出数据，但是抽取出的数据依旧没有语义信息；大量的系统采

用人工或者半人工的方式附加语义，这种方法简单、易用。语义项的定位主要有两种方式：根据内容实现信息抽取的系统中，通过感兴趣信息的在文本中的上下文实现语义项的定位；根据结构实现信息抽取的系统中，通过感兴趣的信息在页面中所处的位置信息实现语义项的定位，或者二者兼用。

2.1.3 本文的 Web 信息抽取方法

信息抽取技术的核心是能够从无结构或半结构的信息中识别用户感兴趣的数据，并将其转化为更为结构化、语义更为清晰的格式[44]。在 2.1.2 中，对常见的信息抽取系统进行了归纳总结，比较了各种抽取工具采用的技术路线和性能指标。比较发现，虽然目前国内外对于 Web 信息抽取技术的研究很多，但是一般的抽取系统只是运用众多抽取技术中的一种，所以在实际的运用中都存在不足之处。而且抽取规则的生成方式一般都采用手工的方式，这样对于用户的要求就很高，一旦目标网站有所改动，必须随时修改程序，修改调试起来非常繁琐。本文提出的 Web 抽取系统将多种抽取技术（基于 HTML 网页结构的方法、基于本体的方法等）较好的融合在一起，借助本体归纳学习抽取规则，这使得该系统具有较强的适应性和健壮性。

2.2 XML 相关技术

XML是一种结构化的数据表现方式，有严格的语法结构，因此，将HTML格式的WEB页面转化成格式良好的XML文档，为后面的模式提取准备好了很好的文档结构。首先，介绍下XML的基础知识。

2.2.1 XML 基础知识介绍

1. XML语法

XML 的语法规则很简单而且非常严格。正因如此，开发读取和操纵 XML 的软件很简单。

规则如下：1）不存在任何“<”或“>”以非标记的形式出现，在需要以非标记的形式使用“<”或“>”时，要将其进行转义编码；2）所有的标记都必须成对出现；3）所有出现在标记中的属性值都必须被引号引起来；4）标记在嵌套

时不能出现交叉。

2.XML文档

下面是一个简单的XML文档。文档的第一行是一个声明，它定义了XML的版本和文档所使用的字符编码。第一行描述了文档的根元素，接下来四行是根元素的子元素，最后一行是根元素的结束标记。下面给出一个简单的XML文档

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
<to>Tove</to>
<from> Jani</from>
<heading>Remider</heading>
<body>Don't forget me this weeked</body>
</note>
```

从这个例子可以看出，XML文档是自描述的。很容易理解每个元素的意义。这是因为每个元素都有标记信息来描述元素的内容。

XML是一个元标记语言。也就是说，允许开发人员定义自己的标记。如上面的文档所示，所有的标记都是自定义的。XML的语法很严格。形式良好限制是XML文档的最低要求。简单的说，XML文档应有而且只有一个根元素，所有开始标记要有相应的结束标记，元素要正确的嵌套等等。

2.2.2 XML在信息抽取系统中的优势及其应用

1.样本文档标记语言的特点及比较

系统首先要从Web站点上下载一定数量的Web页面作为样本文档，分析样本文档的HTML结构特点，从中学习生成整个站点的抽取规则。下面对各种标记语言的特点进行分析比较。

（1）.HTML超文本标记语言

HTML（即超文本标记语言）是用于创建Web页面和Web信息发布的第一个通用语言，它提供跨平台的文档共享。

HTML文档由标记和元素组成。HTML标记只描述文档的外观，而不描

述档的内容本身里面有什么。HTML 是不明白网页内容的，这样就造成了内容搜索差异和不确定性。另一个问题是，HTML 不是可扩展的，这意味着没有一种方便的途径来扩展标记。每一个新标记的引入都会造成系统的不一致性和对标准的修订。

（2）XML 可扩展标记语言

XML(Extensible Markup Language，可扩展标记语言)是用来定义其它语言的一种元语言，其前身是 SGML（标准通用标记语言）。它没有标记集，也没有语法规则，但是它有句法规则。

任何 XML 文档对任何类型的应用以及正确的解析都必须是良构的，即每一个打开的标记都必须有匹配的结束标记，不得含有次序颠倒的标记，并且在语句构成上应符合技术规范的要求[51]。

2. 使用XML的优势

目前 Web 中仍然是 HTML 文档占主体，并且仍在爆炸式增长。由于 HTML 具有结构简单性和灵活性，极大地促进了信息产业的发展，所以相当长一段时间里，HTML 仍然是 Web 的重要信息源。因而论文中讨论的 Web 数据就是 HTML 文档数据，也就是说第二章中获取的 Web 数据源是 HTML 文档数据。

正是由于 HTML 结构太灵活和自由，造成了一个致命的缺陷：难以检索或者是难于提取隐藏其中的数据。针对 HTML 的这种缺陷，XML 语言一方面继承了 HTML 的灵活性和简单性，另一方面又对其存在的问题做了很大的改进，最重要的就是强制结构的完整性和标签的自定义性[50]。此外，针对 XML 的研究以及支持 XML 的工具也不断涌现。但是，HTML 的存在已有多年的历史，XML 取代 HTML 尚需时日，因此研究如何将 HTML 格式转换为 XML 格式具有重要的现实应用价值。

Web 包装器是将 HTML 信息转换为 XML 的有力工具。这是因为 HTML 只描述数据的展示即数据看起来是什么样子，是机器不能理解的仅供浏览的文本，而 XML 则描述数据的内容，是机器能理解的智能文本。因此，要转换 HTML 文档为 XML 文档，必须提供 Web 包装器，抽取 HTML 文档的内容，再将这些内容映射为 XML 文档。

2.3 本体技术

本体作为一种能在语义和知识层次上描述信息系统的概念模型建模工具，在计算机领域得到了广泛的应用，尤其在信息检索和信息抽取中。在信息抽取过程中，如所要抽取的数据来自不同的网站，而由于不同的网站对于同一个数据有不同的表示，所以需要一个强大的、组织良好的知识库的支持。通过建立领域本体库，包含有该领域尽可能完备的知识，来解决所需数据信息的定位以及数据抽取中涉及到的语义问题，如多义词和同义词等。

下面，介绍下本体相关知识。

2.3.1 本体基础知识介绍

本体知识是基于本体的信息抽取系统抽取信息的基础，所以在此简要介绍一下本体的基础知识。

1. 本体的提出

ontology 最早是一个哲学上的概念，可以追溯到公元前古希腊哲学家亚里士德。它在哲学中的定义为“对世界上客观存在物的系统地描述，即存在论”，是客观存在的一个系统的解释或说明，它不依赖于任何一门特定的描述语言，关心的是客观现实的抽象本质。

在人工智能界，最早给出 ontology 定义的是 Neches 等人[47]，1991 年，他们将 ontology 定义为“给出构成相关领域词汇的基本术语和关系，以及利用这些术语和关系构成的规定这些词汇外延的规则的定义”。Gruber 于 1993 年提出了最著名并被引用得最为广泛的定义：“本体是概念模型的明确的规范说明”。后来，Studer 对已有的本体定义进行了深入的研究，1998 年，将 Ontolgoy 定义为“共享概念模型明确的形式化规范说明”。

尽管定义有很多不同的方式但是从内涵上来看不同研究者对于本体的认识是统一的，都把本体当作是领域内部不同主体之间进行交流的一种语义基础，即由本体提供明确定义的词汇表，从不同层次的形式化模式上给出这些词汇和词汇间相互关系的明确定义。因此，本体的用途包括交流、共享、互操作、重用等等。

2. 本体的作用和应用

目前，本体已经是知识工程和人工智能研究的一个重要问题[3]，而且在知识管理、自然语言处理、电子商务、信息检索、数据库设计与集成等领域应用极为广泛。其中，本体主要起到了如下作用：

（1）本体提供了一种结构化表示领域知识的手段。在本体中，不仅明确表示领域概念，还明确说明概念之间的关系，如函数和公理等，并且支持对领域规则进行明确的描述。

（2）本体支持对知识的重用，当构造基于知识的系统时，用已有的本体作为起点和基础来指导知识的获取，可以避免重复的领域知识分析，提高速度和可靠性。

（3）本体是领域知识的形式化表示。现代本体表示语言一般具有严格的逻辑基础，这样可支持对隐含知识进行推理。

（4）本体为人和主体之间的沟通和交流提供了共享之基础。也方便了不同领域的系统开发人员和研究人员之间的沟通和交流。

总的来说，构造本体的目的是为了实现某种程度的知识共享和重用。

3. 本体的分类

目前本体的分类还没有统一的标准。如根据本体的形式化程度不同，可把本体分为高度非形式化的、结构非形式化的、半形式化的和严格形式化的。

根据本体的描述对象不同，可以把本体分为特殊领域本体（如医药等）、一般世界知识本体、问题求解本体和知识表示语言本体等。

目前虽没有能够被广泛接受的分类标准。但以下几个概念的定义意义明确，从某种程度上提供了本体的分类方法[54]：

（1）领域本体：包含特定领域的相关知识，它提供特定领域的概念定义和概念之间的关系，提供该领域中发生的活动及主要理论和基本原理等。如企业本体、医学概念本体等。

（2）通用本体：通常覆盖多个领域。如 CYC：中科院“常识知识的实用研究”中结合的 Agent 和本体的知识库等。

（3）表示本体：以知识表示语言为描述对象的本体。在表示本体中，类、对象、关系、属性、槽等术语要经过严谨的分析和定义。

（4）任务本体：主要研究可共享的问题求解方法，其实质是从推理和问题求解的角度刻画领域知识。任务本体有助于解决领域知识不能以与其使用

方式无关的形式表示问题，对知识库系统的重用和组件化的开发十分重要。

4. 本体的构造规则

目前已有的本体很多，出于对各自问题域和具体工程的考虑，构造本体的过程也是各不相同的。如：Mike Uschold 等[25]提出的骨架法(Skeletal Methodology)、Gruninger 等[26]的企业建模法(TOVE)等。这些方法并没有成为一种成熟的工程方法论。直到 1995 年，Gruber 提出了最有影响的有益于构造 Ontology 的 5 条构造准则[27]：

明确性：Ontology 应该明确有效地表达概念，尽可能使用标准术语。定义应该尽可能的完整。所有定义应该用自然语言加以说明。

一致性：由术语得出的推论与术语本身的含义是相容的，不会产生矛盾。

可扩展性：本体应该为可预料到的任务提供概念基础，支持在已有的概念基础上定义新的术语，以满足特殊的需求，而无须修改已有的概念定义。

最小编码偏见：创建本体的时候，应该在知识层面上进行概念化，不依赖于特定的符号层面的编码，因为实际的系统可能采用不同的知识表示方法。

最小承诺：对待建模对象给出尽可能少的约束。

5. 本体描述语言OWL

为了 Web 上的应用程序能够方便地使用本体，需要通用的本体语言来对本体进行表示，就像 XML 作为标准的数据交换语言一样。用户可以用本体语言为领域模型编写清晰的、形式化的概念描述。大量的研究工作者活跃在该领域，因此诞生了许多种本体描述语言。目前已有多种本体描述语言：RDF 和 RDF-S、DAML+OIL、OWL、SHOE、KIF、XOL、OCML、Ontolingua、CycL、Loom 等。其中 OWL（Web ontology Language[53]）是 W3C 推荐的语义 Web 中本体描述语言的标准。

OWL 全称 Web ontology Language，它主要通过扩展建模原语进一步扩充了语言表达能力。可以认为 OWL 是用 XML 语法、RDF 模型定义的一种描述逻辑语言。OWL 语言可以形式化的、明确的描述本体语义，不仅可以进行简单的检索，而且可以根据语义进行逻辑推理。OWL 有良好的扩展性，而且 OWL 具有形式化的语义表达能力，具有互操作性等优势，可以解决关键词匹配方法难以解决的一词多义、同义词等问题。

OWL 能够清晰地表达本体中概念的定义以及概念之间的关系。OWL 相

对 XML、RDF 和 RDF Schema 拥有更多的机制来表达语义，从而 OWL 超越了 XML、RDF 和 RDF Schema 仅仅能够表达网上机器可读的文档内容的能力。

2.3.2 使用本体的优势

语义异构是数据集成过程中需要解决的一个重要问题，它是指本地数据意义的不同点或相似点[52]。有意义的数据集成依赖于发现模式元素之间的不同点和相似点，在集成过程中需要考虑数据的语义。语义是人们根据他们对世界的理解而赋予数据的解释，对数据的不同解释就引起了语义异构。

解决语义异构问题的方法需要对异构和自治的软件系统进行装备，使之具有共享和交换信息的能力。这一点可以通过很多方式实现，但每一种只适合于一定的环境。其中一个解决方法是开发者自己编写转换系统间术语集的代码。当只有少数几个系统需要交互时，这种方法还比较有效。然而，当有更多的系统加入进来使语义异构的程度增加时，开发费用就会大幅增长，这种方法的实用性就会大打折扣。本体通过对概念的严格定义和概念之间的关系来确定概念的精确含义，表示共同认可的、可共享的知识，从而能够解决上面的问题。一个本体为特定领域的实体给出名字和描述，使用谓词来表示这些实体之间的关系[48]。它为表示和交流领域的知识提供了一个词汇库，并给出了一系列包含在词汇库中的术语的关系，因此，它具有描述数据源语义和解决异构的潜力，可将本体用于数据抽取任务。

总之，在信息抽取集成中使用本体有许多优点。首先，本体提供了一个丰富的、预定义的词汇库，可作为与数据源的稳定的概念接口，并且独立于数据模式。第二，本体表示的知识足够支持所有相关信息源的转换。第三，本体支持一致的管理和非一致数据的识别等。

2.4 本章小结

本章介绍了信息抽取的产生及其发展历史，阐述了信息抽取的定义、功能和模型结构，并对常见的信息抽取系统进行了归纳总结，比较了各种抽取工具采用的技术路线和性能指标；然后分别介绍了 XML 和本体技术，并详细阐述了这两种技术在信息抽取中的优势。

第 3 章 基于语义的 Web 信息抽取系统的设计

前面对常见的信息抽取系统进行了归纳总结，比较了各种抽取工具采用的技术路线和性能指标。比较发现，虽然目前国内外对于 Web 信息抽取技术的研究很多，但是一般的抽取系统只是运用众多抽取技术中的一种，所以在实际的运用中都存在不足之处。而且抽取规则的生成方式一般都是采用手工的方式，这样对于用户的要求就特别高，一旦目标网站有所改动，必须随时修改程序，修改调试起来非常繁琐。本文提出的 Web 抽取系统将多种抽取技术（基于 HTML 网页结构的方法、基于本体的方法等）融合在一起，借助本体归纳学习抽取规则，这使得该系统具适应性和健壮性。

3.1 系统的框架模型结构

信息抽取的目的就是要让有用的信息以统一的形式集成在一起，这样可以方便对其检查和比较。另外，信息以统一的形式集成还有利于对数据做自动化处理，例如用数据挖掘的方法发现和解释数据模型，为信息检索提供数据支持等[49]。信息抽取技术并不试图全面理解整个 Web 文档，只是对文档中包含相关信息的部分进行分析。

知识密集型 Web 站点向人们提供了丰富的产品信息，页面中包含产品特征的详细描述。具体而言，知识密集型 Web 站点主要具有以下几个特征：

1.每个页面中都包含多个产品信息，描述产品的性能参数，并且页面中包含的产品数量相对稳定。

2.一个站点中，不同的产品页面具有相似的结构。目前的大部分知识密集型 Web 站点，由于页面数量巨大，大多是由机器自动生成的。于是，虽然不同的 Web 页面显示了不同的产品数据，却具有类似的结构和布局。

3.页面中描述产品信息的模块也具有相似的结构。页面中显示的产品属于同一产品类别，因此在各个产品信息中包含了相似甚至相同的信息项。

4.页面中产品数据多用 HTML 语言描述。HTML 标记给页面设计带来了很大的方便，但是正是由于 HTML 太过灵活，也随之带来了一些问题，比如

页面的数据与表现交错在一起，显示的数据没有一定的结构和模式。虽然用户可以很好地理解页面数据格式，但是对于计算机来说却很难解析其语义。因此，从 Web 页面抽取数据，需要抽取页面的结构和内容。

本文结合知识密集型 Web 站点的特征，提出了一种基于语义的 Web 页面结构化信息抽取模型，一次从页面中抽取多个数据项。其框架结构如下图：

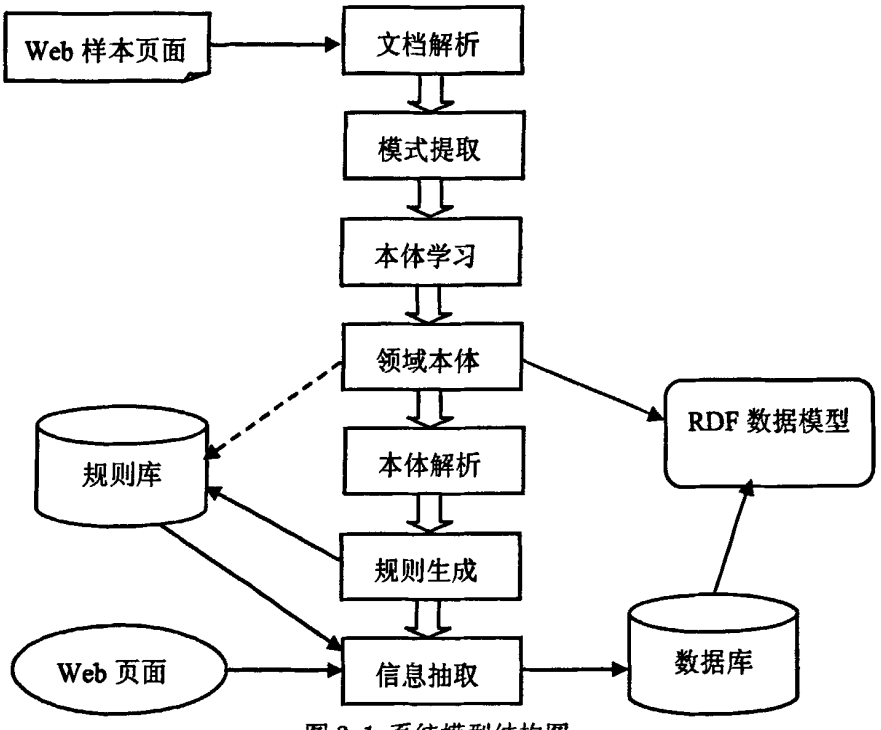


图 3.1 系统模型结构图

该模型主要有文档解析，模式提取，本体学习与领域本体构建，规则库生成与信息抽取算法生成五个阶段，每一阶段的具体描述如下：

文档解析阶段，样本文档多数使用HTML语言来描述的，而大多数HTML页面所蕴含的数据是非结构化的；同时，它又不是完全非结构化的，文本、图片、声音、图像等数据会按照一定的方式组织起来，具有一定的描述层次和结构，所以从整体上看，HTML 页面还是有一定结构的，但没有结构化数据那样严谨的结构模式，所以通过文档解析过程，完成将 HTML 文档转换为格式良好的 XML 文档，为后面的模式提取做好样本文档的格式准备。

模式提取阶段，由于样本文档页面中描述产品信息的模块具有相似的结构，系统按照一定的准则，将这些相似的结构模式提取出来，从而为本体的

建立奠定了基础。

本体技术阶段，该模型把本体技术引入到了信息抽取过程中来。本体作为一种能在语义和知识层次上描述信息系统的概念模型建模工具，在计算机领域得到了广泛的应用，尤其在信息检索和信息抽取中。在信息抽取过程中，比如所要抽取的图书信息来自不同的网站，而由于不同的网站对于同一个数据有不同的表示，所以需要一个强大的、组织良好的知识库的支持。通过建立领域本体库，包含有该领域尽可能完备的知识（如概念以及概念间的关系，概念的各种属性等），来解决所需数据信息的定位以及数据抽取中涉及到的语义问题，如多义词和同义词等。

抽取规则生成阶段，对 OWL 进行操作，归纳学习出抽取规则，作为抽取具体信息项的重要依据，指导抽取动作。根据定义好的抽取规则从已经获得的网页数据中获得数据块，根据本体库的内容，将数据块所包含的数据准确抽取出来，同时解决相应的同义词，多义词以及数据项不完整和排序不固定的问题。

下面分别从文档解析、模式提取、领域本体的建立以及抽取规则及其执行算法四个方面展开对整个 Web 信息抽取系统作详细介绍。

3.2 文档解析

首先介绍框架模型中文档解析阶段。文档解析的任务是纠正原始 Web 文档中的结构不完整性等错误，转换成结构良好的等价文档。

目前 Web 中仍然是 HTML 文档占主体，并且仍在爆炸式增长。由于 HTML 具有结构简单性和灵活性，极大地促进了信息产业的发展，所以相当长一段时间里，HTML 仍然是 Web 的重要信息源。因而论文中讨论的 Web 数据就是 HTML 文档数据。

正是由于 HTML 结构太灵活和自由，造成了一个致命的缺陷：难以检索或者是提取隐藏其中的数据。针对 HTML 的这种缺陷，XML 语言一方面继承了 HTML 的灵活性和简单性，另一方面又对其存在的问题做了很大的改进，最重要的就是强制结构的完整性和标签的自定义性。正是因为 XML 比 HTML 具有更多的优点，人们普遍认为：XML 最终会取代 HTML 而成为

Web 的通用语言。此外，针对 XML 的研究以及支持 XML 的工具也不断涌现。但是，HTML 的存在已有多年的历史，XML 取代 HTML 尚需时日，因此研究如何把 HTML 格式的文档转化成结构良好的等价文档具有重要的现实应用价值。论文将在第四章中，具体讲述转换算法的实现过程。

3.3 模式提取

一个站点中，不同的产品页面具有相似的结构。目前的大部分知识密集型 Web 站点，由于页面数量巨大，不可能由人工进行定制，大多是由机器自动生成的。于是，虽然不同的 Web 页面显示了不同的产品数据，却具有类似的结构和布局。

从样本文档中归纳学习页面模式是信息抽取系统中的重要环节，模式提取模块就是针对这一任务而设计的。本文参考文献[9]的方法，借助Suffix Tree结构给出了自动提取Web页面数据模式的过程。论文将在第五章中，具体讲述转换算法的实现过程。

3.4 基于语义的本体建立方法

本体学习以及领域本体的构造是系统模型中重要的一部分，同时也是本节的研究重点。下面介绍下基于语义的本体的构建方法。

分析了获取概念及其关系是本体建立的瓶颈的原因，本文提出一种对于在不同的网站上获取的信息的基于语义的本体建立方法，将其中语义相同或相近的术语通过聚类生成概念是获取概念的可行方法。本文以模式提取中得到的术语语义为基础，经过基于语义的聚类，生成概念集；再根据术语语义关系与概念关系之间的映射规则建立概念的关联，从而构造概念树并建立本体。这种半自动的本体建立方法避免了因对概念理解不统一而产生的不一致性，得到的本体易于扩展。

3.4.1 本体定义方法

定义本体为 Ont = (C,R)，R = {r | r = f:$c_i \rightarrow c_j, c_i \in C,\ c_j \in C$}。式中，C 为模式提取中得到的领域 D 内概念的有限集，C = {c | c∈D}；R 为定义在 C

×C 上的关系 r 的有限集；r 为描述两个不同概念之间的某种映射的关系。

1. 概念的定义

概念是具有相同属性的对象类，每个概念可用多元组表示 c = (tc,dc,A,Coms, Cons)。tc 是本体中命名此概念的唯一术语，c 与 tc 一一对应。dc 是概念的描述。由于命名概念的术语 tc 与概念 c 是一一对应的，因此可以用术语的语义作为概念的描述。用 tc 在当前上下文中语义对应的序数 k 来表示 tc 的语义，并作为概念 c 的描述，即 c→dc = S(c→tc) = k。基于语义的概念描述方法具有以下优点：①能够有效地避免一词多义和一义多词现象；②将概念的描述用自然数表示，便于计算处理。

A 是表示概念内涵的属性集合。属性分为简单属性和复合属性，简单属性 a = (ta,da,Type,U)，其中，ta 为命名此属性的术语；da 为属性的描述；Type 为属性的类型，Type∈{int,real,string,bool}，分别表示整数型、实数型、字符串型、布尔型；U 是属性的单位。复合属性 ca = {ta,{$a_1,a_2,...$}}；Coms 是本体承诺的集合。本体承诺是知识库中的术语与本体论中相同或相近概念之间的形式化映射[65]，用(t,i)表示概念与第 i 类样本的术语 t 是相同或相近的；Cons 表示属性的取值范围、不同概念的属性之间的关系规则等约束。

2. 关系的定义

将本体中的关系分为基本关系和自定义关系两类，本文重点讨论基本关系。根据对象之间的基本联系[66]，定义如下三种概念之间的基本关系：

（1）层次关系：概念之间具有包含关系，如机床包括车床、铣床、钻床等，铣床包括立铣、龙门铣等。更一般、更抽象、更广义的概念具有较高的层次，层次较高的称为父概念，层次较低的称为子概念，分别用 Super_Concept(cSuper,cSub)和 Sub_Concept(cSub,cSuper)表示，即 cSuper 是 cSub 的父概念，cSub 是 cSuper 的子概念。根据概念层次的高低形成概念树，概念树是本体的基本架构。

如果存在 Super_Concept (c_0,c_1)Super_Concept (c_1,c_2),...,Super_Concept (c_{n-1},c_n)，称$c_0,c_1,c_2,\cdots,c_n$为一个概念世系(lineage)。定义$c_0,c_1,c_2,\cdots,c_{n-1}$为$c_n$的祖先概念，用 Ancestor_Concept(c_i,c_n)表示(i=0,1,$\cdots,n-1$)。在概念世系$c_0,c_1,c_2,\cdots,c_n$，定义两个概念之间的距离 d(c_i,c_j) = j-i。

（2）整体/部分关系：如果c_1是c_2的一个部分(部件)，称c_1是c_2的部分概

念，用 Part_Concept(c_1, c_2)表示，其逆关系为 Whole_Concept(c_2, c_1)。

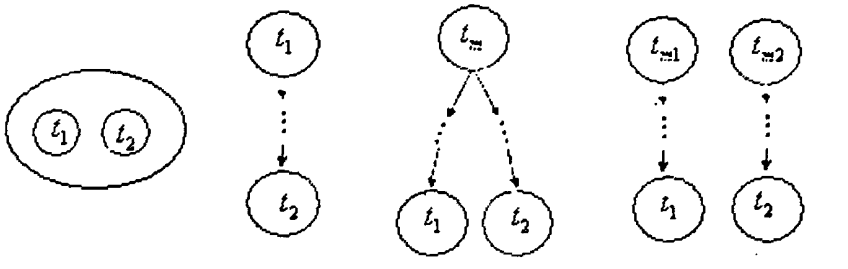
（3）同类概念：一个概念的所有子概念可以按照不同的分类标准分成不同的类，如机床可按加工方法不同分为车床、铣床等，也可按加工精度分为普通机床和数控机床。车床和数控机床都是机床的子概念，但它们的内涵和外延显然不同。将属于同一类别的子概念定义为同类概念，用 Same_Class_Concept(c_1, c_2)表示。

基于语义的本体建立方法从各模式中提取出来的术语集出发，将不同模式中语义相同或相近的术语通过聚类生成概念，根据概念名称之间的语义关系建立概念之间的关系，初步建立本体；由本体定义新的概念及其关系，不断地扩展本体。

3.4.2 基于术语语义聚类生成概念的方法

由于大部分术语具有一词多义现象，用 t(k)特指语义为第 k 项的术语 t。t(k 的同义词集合用 SSYN(t(k))表示，同样，用 SHYP(t(k))、SMER(t(k))分别表示 t(k)的上位词集合和部分词集合。

分析可得，两个术语 t_1 与 t_2 之间的语义关系可能有四种情况(图 3.2)。



(a) t_1 与 t_2 是同义词　(b) t_1 是 t_2 的上位词　(c) t_1 与 t_2 具有相同的 unique beginner tm　(d) t_1 与 t_2 的 Unique beginner 不同

图 3.2 术语之间语义关系的四种情况

规定这四种情况下它们的相似度 A(t_1, t_2)分别如下：

（a）如果 t_1 与 t_2 是同义词，即 $t_1 \in$SSYN(t_2)，则 A =1；

（b）如果 t_1 是 t_2 的上位词(t_1 比 t_2 的语义更加一般)，即存在 $h_1, h_2, \cdots, h_n$

($n \geqslant 0$)，使$t_1 \in \mathrm{SHYP}(h_1), h_1 \in \mathrm{SHYP}(h_2), \cdots, h_n \in \mathrm{SHYP}(t_2)$，则 $\mathrm{A} = \frac{1}{n+2}$；

（c）t_1与t_2具有相同的 unique beginner tm 及$h_{11}, h_{12}, \cdots, h_{1p}$，$h_{21}, h_{22}, \cdots, h_{2q}$ ($\mathrm{p} \geqslant 0, \mathrm{q} \geqslant 0$)，使$t_m \in \mathrm{SHYP}(h_{11}), h_{11} \in \mathrm{SHYP}(h_{12}), \cdots, h_{1p} \in \mathrm{SHYP}(t_1)$且$t_m \in \mathrm{SHYP}(h_{21}), h_{21} \in \mathrm{SHYP}(h_{22}), \cdots, h_{2q} \in \mathrm{SHYP}(t_2)$，则 $\mathrm{A} = \frac{1}{(p+2)(q+2)}$；

（d）t_1与t_2的 Unique beginner 不同，则 A =0。

对于任何术语集合：$TS_1 = \{t_{11}, t_{12}, \cdots, t_{1m}\}$，$TS_2 = \{t_{21}, t_{22}, \cdots, t_{2n}\}$，定义它们的相似度为 $\mathrm{A}(TS_1, TS_2) = \min\mathrm{A}(t_{1i}, t_{2j})(\mathrm{i} = 1,2,\cdots,\mathrm{m}; \mathrm{j} = 1,2,\cdots,\mathrm{n})$。

对术语集采用分层聚类技术[67]，算法如下：

（1）令 TS = $TS_1 \cup TS_2$，每个术语自成一类，组成聚类集$c_1 = \{t_{11}\}, \ldots, c_m = \{t_{1m}\}, c_{m+1} = \{t_{21}\}, \ldots, c_{m+n} = \{t_{2n}\}$ C=$\{c_1, \ldots, c_m, c_{m+1}, \ldots, c_{m+n}\}$

（2）计算 $\mathrm{A}(c_i, c_j)(\mathrm{i} \neq \mathrm{j})$。

（3）如果 $\max\mathrm{A}(c_i, c_j) \geqslant 0.5$，令 $c_i = c_i \cup c_j$，C = C-c_j；否则结束。

（4）重复(2)，直至 C 中只有一个元素。

结合一个例子说明生成概念的步骤。TS_1 = {automobile(1),motor(1), model(2)}，TS_2 = {car(1),engine(1),gearbox(1)}

令 C= $TS_1 \cup TS_2$ ={automobile(1),motor(1),model(2),car(1),engine(1),gear –box (1)}，用一个对称阵表示 C 中术语两两之间的相似度，规定 $\mathrm{A}(c_i, c_i) = 1$。

$$A = \left[A\left(c_i, c_j\right) \right] = \begin{bmatrix} 1 & & & & & \\ 0.036 & 1 & & & & \\ 0 & 0 & 1 & & & \\ 1 & 0.036 & 0 & 1 & & \\ 0.029 & 0.5 & 0 & 0.029 & 1 & \\ 0.021 & 0.033 & 0 & 0.021 & 0.028 & 1 \end{bmatrix}$$

最终得到的聚类集为 C={{automobile(1),car(1)},{motor(1),engine(1)}, gearbox(1),model(2)}。

3.4.3 概念关系的建立

用来命名概念的术语具有同义、反义、广义、狭义等语义关系，术语的语义关系与概念的关系之间存在某种对应联系。因此，可以借助术语的语义关系来建立概念之间的关系。给定概念$c_1 = (tc_1, dc_1, A_1, Coms_1, Cons_1)$，$c_2$=

$(tc_2, dc_2, A_2, Coms_2, Cons_2)$，根据它们之间的语义关系，可以建立规则集，用来建立概念之间的关系。

以概念作为结点，用层次关系（父概念或子概念）为边将概念连接起来可构造概念树。为了使概念树完整，除了聚类生成的概念外，有时需要添加连接型概念到概念树中。连接型概念与通过聚类生成的概念的区别在于，其本体承诺为空。概念树可以清晰地表达本体中概念之间的层次关系。以此为骨架，添加其他类型的边，如表示整体/部分关系的边，构造完备的本体。图3.3是某车方案设计知识库的概念树（局部）。

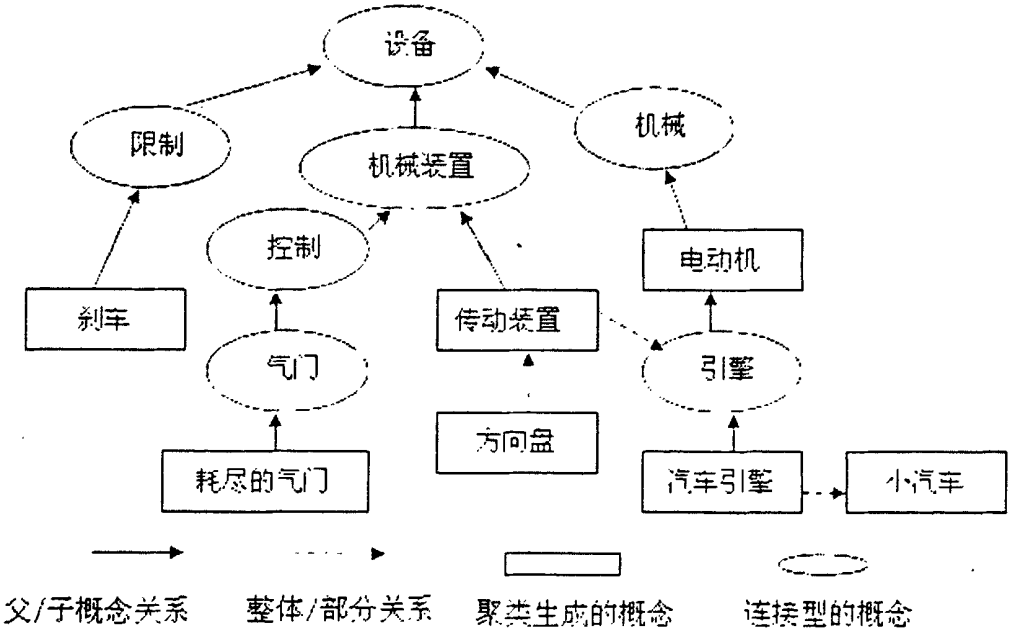


图3.3 某车方案设计知识库的概念树(局部)

3.4.4 本体学习与领域本体的构建

结合本体学习的技术和方法，把有效的概念结合在一起，构造了一个小型的图书领域本体。本体学习是半自动构建本体的一系列方法和技术，它通过利用各种数据源，比如Web文档，以半自动方式新建或扩充改编已有本体，由此构建一个新本体[34]。一个图书领域本体的结构如图3.4所示。

本文在本体的构造过程中，借助了常用的本体构造工具Protégé，构造了一个小型的图书领域本体，最后用OWL描述了该本体，详细过程见第六章。

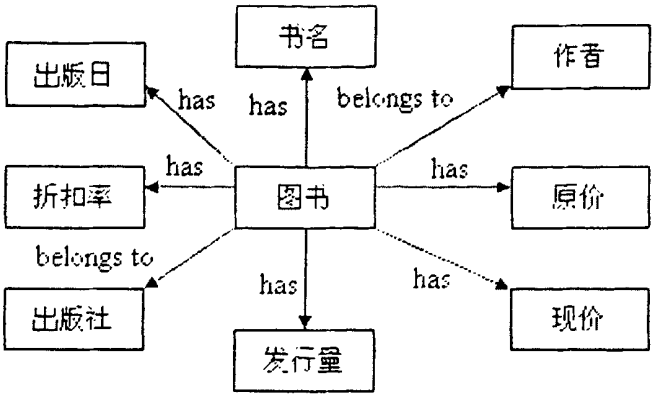


图 3.4 图书领域本体结构图

本体方法的引入消除了同类信息源 Web 页面的异构性。比如有两个网上图书销售系统 A 和 B，可能两者在图书的信息布局上不尽相同，但却可以根据相同的概念描述（比如书名，作者，价格，出版时间等等），生成各自的抽取规则，提取信息存储到结构相同的数据库中。总之，本文提出的基于本体的信息抽取模型把待抽取的概念包含在本体中，然后再从本体的 OWL 文件中提取概念及概念属性。该方法提高了抽取效率，并且本体的引入使该方法具有了较好的可移植性。

3.5 基于 OWL 描述本体的语义信息抽取

这是本文信息抽取的核心部分。文中描述了一种基于 OWL 本体抽取出可以被语义 Web Agent 理解的语义数据的方法，通过上面的步骤，可以获得抽取数据项的本体描述，在抽取过程中先从该 OWL 本体模型中归纳生成抽取规则，然后将抽取出的数据加上语义标记，转换成语义 Web Agent 可以接收的 RDF 格式，从而可以促进语义 Web 的发展。

3.5.1 基于本体的抽取规则的生成

因为进行数据抽取的数据源一般都是某一领域的一组数据，所以首先将包含待抽取信息的页面过滤，转化成结构基本一致的各项记录，各记录项中的同一数据项的相对路径基本一致。如果各记录项中的数据项的相对位置有所变动，抽取路径就不止一个。

规则生成阶段的任务是归纳学习抽取规则，作为信息抽取的重要依据。抽取规则是通过对本体的解析获得的，而本体解析实际上。就是解析描述本体的 OWL 文件。下面先给出抽取规则的定义。

定义 1.抽取规则定义为一个五元组：

Concept：{Pre, Path, Type, Other {;}}*·

其中各项的含义为：①Concept 表示需要抽取的信息项的物理名称。②Pre 表示该信息项的前导符。这个前导符的作用在于：如果一个信息有多条路径表达式，则会在路径定位时产生二义性，这时前导符可以作为辅助定位手段，若符合路径，并且具有该前导符，则定位成功。③path 表示信息项的路径表达式，它包含了信息项的完整路径，各条路径之间用"|"分隔。④Type 表示：规则限制的抽取类型，即信息项的类型，如字符、数字、日期等；⑤Other 表示信息项的补充说明·⑤*表示可重复多次。此处表示信息项可能有多种出现方式，各种出现方式之间用";"分隔。例如下面一段代码：

```
<body><h1><b>Durability of materials in molten aluninum alloys</b></h1>
<p><b> Author: </b>
<a href= "http:zju.edu.cn/name?Yan M" </a>
<br/><b> Corporate Source: </b> Zhejiang Univ,Dept M at Sci
<br/><b> Journal: </b> JOURAL OF MATERTALS SCIENCE,2001</p>
<table border="1">
<tr><td><b> ISSN: </b> 1359 </td></tr>
<tr><td><b> Publication Year: </b> 2000 </td></tr>
<tr><td><b> Document Type</b> Review</td></tr>
<tr><td><b> Language: </b> English</td></tr></table>
```

生成的部分抽取规则如下：

{Title}:{Null},{body:h1:b},String,Null

{Vice title}:{Null},{body:h2:b},String,Null;Empty

{Author}:{Author:},{body,p:a},String,URL

{Corporate Source}:{Null},{body:p:br/},String,Null

…

规则第一行说明需要抽取的信息项 Title 是 Web 文档的第一项内容，它无前导符，源码路径为{body:h1:b}，其类型为 String，无补充信息·规则第二条说明 Vice Title 有两种可能：（1）它是一个无前导符、路径为{body:h2:b}的 String，并且无补充信息；（2）在某些 Web 文档中可能不存在这个数据

项（表示为 Empty）。规则第三条说明 Author 为有前导符“Author:”、路径为{body: p: a}的一个 String，另外它还有一个 URL 属性值需要抽取。

5.1 规则生成算法

本文依据构造的图书领域本体，从领域本体中归纳学习 Web 抽取规则，规则抽取的算法描述如下：

输入：（1）产品信息的描述项<item1><item2>...<item m>（2）本体的 OWL 描述文件，本体中包含了样本页面的概念，一个概念对应一个数据项描述：每个页面中有 k 个产品的描述信息，每个产品信息出 m 个概念进行描述，样本页面中总共有 $k \times m$ 条数据项。

输出:抽取规则 ExtractRules。

算法描述：

{ ExtractRules=Φ

for i=1 to m do

{

解析本体的 OWL 描述文件，从本体中提取出样本页面中所有的有效概 pre_concept(i)，对概念 Pre_concept(i)，，寻找相匹配的概念属性描述（即 path 属性），生成规则{ERi1，，ERi2，……，ERik}添加到 ER(i)中；其中每个抽取规则 ER 中的起始路径 path 满足

$$path_{it} = path_{i(t-1)}\langle item_t\rangle\langle item_{t+1}\rangle...\langle item_m\rangle\langle item_1\rangle...\langle item_{t-1}\rangle, t = 2,...,k$$

}

for i=1 to m do

for j=1 to n do

{

把概念在 Web 页面中出现的路径作为一个抽取规则添加到 Extract Rules 中，即{ERi1，ERi2，…，ERik}添加到 Extract Rules 中，令 i=1，2，…，m

}

Return ExtractRules=ExtractRules[1]+ExtractRules[2]+…+ExtractRules[m]

}

若一个数据项的抽取路径有多个，提取时使用标识符的方法来定位数据

项的位置，标识符就是能唯一标识该数据项的字符串或字符串组合，如可以使用引导符(“:”前面的字符)作为标识符；若当前标识符标识的数据项的路径与抽取规则中的路径比较是否有相同的，以决定使用哪个规则。引导符后面的数据就是该抽取规则所要抽取的内容，提取后生成相应的 SQL 语句插入到关系表中，当一个记录项的各个数据项完成抽取后从而转到下一个记录项。

3.5.2 关系数据库的创建

首先创建一个主表，其每一行都代表一个资源实例，然后将与该资源相关的属性添加进来，如果是一个对象属性，那么就创建一个对象标识符作为该属性的属性值，然后为每个对象属性再类似上面创建一个表，类似上面将与该对象属性相关的属性添加进来，如果是一个数据类型属性就直接将其值插入到关系表中。如下面的表所示。

表 3.1 书信息对照表

Book	Name	Author	Price	.....
0001	单片机轻松入门	1001	2001	.....
0002	形式语言与自动机理论	1002	2002	.....

表 3.2 作者信息对照表

Author	Name	Sex	.....
1001	周坚	男	.....
1002	蒋宗礼	男	.....

表 3.3 价格信息对应表对照表

Price	PriceUnit	PriceVal	.....
2001	dallor	5.95	.....
2002	dallor	6.01	.....

3.5.3 具有语义信息的 RDF 数据模型的生成

互联网中所有的信息定位都可以通过 URI 来描述，它是一类元数据。通过对元数据的描述和处理来理解和处理互联网中大量的有一定语义关联的信息资源是自动处理这些信息的一种方法和手段。资源描述框架 RDF 是一种描述和交换元数据的框架[68]，它提供了在互联网中交换机器可理解信息的应用系统之间的互操作性，强调了对互联网资源的自动处理。在表达语义方面，RDF 提供了一种统一的数据模型，这种模型为 Web 上的知识表达、知识推理和知识校验提供了底层的标准，因此本文利用数据库与本体库相结合生成 RDF 数据模型图，来表示抽取数据的语义信息。

基本思想：首先扫描主表中的每一行，其每一行都代表一个资源实例，对于与该资源相关的每一列属性，首先找到其对应的本体定义，如果是一个数据类型属性就直接将其写入到 RDF 代码中；如果是一个对象属性，那么就找到与该对象属性对应的表并找到与属性值相等的行，扫描该行，类似上面将与该对象属性相关的属性依次按照 RDF 格式规范写入到 RDF 文件中。

1RDF 数据的生成算法：

```
generateRDFInstances( )
{ ontoURI="http://somewhere/someontology#";
  someURI ="http://somewhere/books.html";
for 主表中的每一行
{ table=主表;
   instance=someURI+ "#" +table 中第一列的值(一个 OID);
   Resource R=createResource(instance);
for table 中的每一列 A
{ //B 为外部本体定义的且与 A 相等的属性;
Property p=createProperty(ontoURI,B);
if (p 为数据类型属性) then
{ range=A 的值;
R.addProperty(p,range);
}
```

else //即为一对象类型属性;

{ range=someURI+“#”+A 的值(一个 OID);

Resoure Rsu=createResource(range);

R.addProperty(p,Rsu);

R=Rsu;

table=与 table 通过 OID 相关联的表;

}}}}

2 生成 RDF 数据

<rdf:RDF xmlns:rdf=“http://www.w3.org/1999/02/22-rdf-syntax-ns#”

xmlns:ontoURI=“http://somewhere/someontology#”>……

<rdf:Description rdf:about=“http://somewhere/book.html#Price2001”>

<ontoURI:PriceUnit>dollar</ontoURI:PriceUnit>

<ontoURI:PriceVal>5.95</ontoURI:PriceVal>

</rdf:Description>

<rdf:Description rdf:about=“http://somewhere/book.html#Book0001”>

<ontoURI:Name>单片机轻松入门</ontoURI:Name>

<ontoURI:hasPrice rdf:resource=“http://somewhere/book.html#Price2001”/>

</rdf:Description>……

</rdf:RDF>

3 生成 RDF 数据图模型

在图 3.5 中，椭圆表示的是一个资源，有向边表示的是该资源的一个属性，方框表示该属性的属性值，可以看出任何一个能够理解引入本体的 Agent 都能够可以在这些 RDF 数据中查找到具有某个特定属性值的实例，如可以查找 Stevens W.写的所有书。如果一个出版社的 Agent 和一个书店的 Agent 都能够理解这个本体，那么书店和出版社之间就可以进行通信，当书店的某种书少于一定的数量时就可以从出版社订购，出版社也可以根据各个书店的销售情况掌握某种书的受欢迎程度。

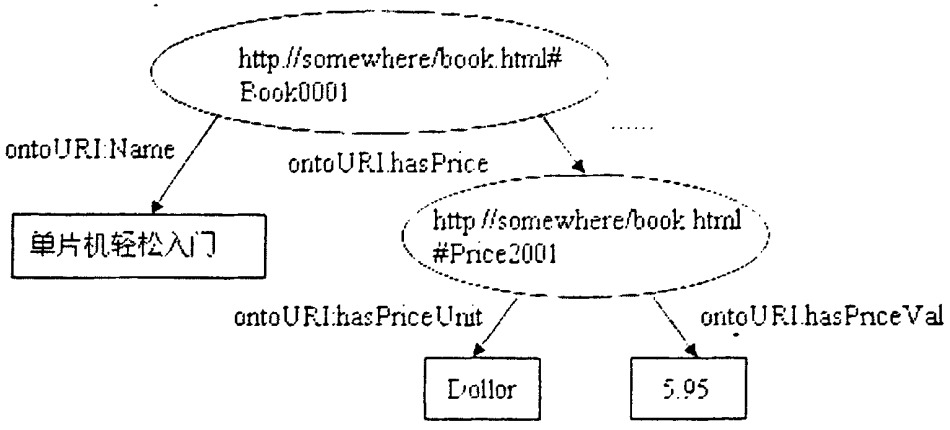


图 3.5 生成 RDF 数据图模型

以上，就是使用本体抽取出语义数据的方法。基于本体的数据抽取可以跨不同网站，即使是领域发生变化也只需对本体进行修改。在数据抽取出后为其添加语义标记，即转换成具有语义信息的 RDF 格式。

3.6 本章小结

本章给出了基于本体的信息抽取系统的原型结构框架图，然后对信息抽取的抽取流程和规则生成的整个过程进行了详细阐述，最后具体讲述了基于语义的本体建立方法，并给出了从领域本体中归纳学习抽取规则的具体算法和生成具有语义信息的 RDF 数据模型的方法。

第 4 章 基于栈与链式结构的 HTML 到 XML 转换算法

Web 网页上大多数的信息都是用 HTML 写的，因为它只是一种用于浏览信息的语言，不能表达数据本身，Web 网页还没有形成一个良好的结构化文档的存贮，而只是一个结构不规范的 HTML 页的聚集，因此迫切希望来自 Web 网页资源的信息以一种结构化的方式来存贮。XML 和它的各种扩展功能如数据模型、查询语言等是实现结构化方式的一种，是一种元语言，可以弥补很多 HTML 的不足，可以为下面的信息抽取工作提供良好的文档格式。因此，本章提出了一种实现 HTML 到 XML 转换的方法。

4.1 问题描述

现在已经有一些成熟的和发展较快的支持 XML 到 HTML 转换的机制[12]，但在 HTML 到 XML 方向的转换方面却没有一定策略和机制的支持。从目前的研究看，实现 HTML 到 XML 的转换方法主要基于内容和形式。如文献[13]提出了一种基于内容的 HTML 到 XML 转换策略，定义了 HTML 中数据与格式信息的分离规则和这种规则到 XML 模式的一种映射。W3C 发布的 XHTML 1.0 规范采用模块化方案，将现有的 HTML 4.0DTD 模块化，进行形式上的转换。笔者所探讨的方法也是从形式上实现 HTML 到 XML 格式的转换，为信息抽取提供可用的格式。

目前实现 HTML 到 XML 转换的工具有不少，如 HTMLTidy[6]，但这些工具有很多不足之处，如 Tidy 要求 HTML 文档中的标签至少是循环嵌套的。对这些缺点，很多人做了相关的研究，如文献[16]提到基于 HTML 的有序性，使用栈这种数据结构来进行转换，但是栈在进行查找比较时不如链表高效。笔者给出了一种基于栈结构与链式结构的从 HTML 格式到 XML 格式的转换方法，用栈这种数据结构完成 HTML 标签的匹配，以链表为存储结构，详细阐述了消除 HTML 格式中不严格语法的通用算法，最后给出了输出 XML 格式的算法，为信息抽取提供了可用的格式。

4.2 HTML 到 XML 的转换原理与步骤

HTML 到 XML 的转换为信息抽取提供了样本文档的格式基础，也是本章的重点研究内容。下面从现有的 HTML 到 XML 转换系统和算法、HTML 和 XML 格式的比较、转换原理与步骤等方面来介绍本章转换算法的基础。

4.2.1 现有的转换系统和算法

包装器就是一个能够将数据从网页（例如 HTML 页面）中转换出来并且将它们还原为结构化的数据（例如 XML 数据）的软件程序。

编写包装器的方法经历了手工编写、半自动化生成和现在正在研究的全自动化生成三个阶段。

在手工编写阶段，由专门的具有一定知识的专业人员首先对网页进行分析，然后写出包装器，在这个阶段中，对于这样的专业人员的要求非常高，并且是一个非常困难的工作。

随着人工智能技术的使用，采取了机器学习、数据挖掘和概念建模等方式，在一定程度上使得包装器的产生工作能够自动地进行。但是，在这些技术中都要求不仅要由用户提供标识样本集，还要有一定的先验知识，因此，还需要进行改进。

近年来，提出了两个能够基本全部自动化的方法，这两个方法只要输入相应的网页就能够自动进行分析和生成包装器，并且将数据从网页中提取出来，在整个过程中需要用户干预的地方只是最后的数据结构的语义分析部分。采取这样的全自动方式，大大减轻了用户的工作量，使得网页的转换工作有了较大的进步。但是，这两种方法也都存在着一定的缺点。其中，RoadRunner 系统中的 ALIAN 算法根据比较样本页的 HTML 代码之间的匹配与不匹配部分来确定一个公用的包装器，但是在这样的匹配过程中，该算法只是简单地把 HTML 代码作为字符串流来进行比较，并没有利用 HTML 代码在结构上的一些特性，使得其对可重复项的处理较为复杂。EXAGL 算法的核心是寻找并确定网页之间的最大频繁发生等价类，再根据该类产生模板并且抽取数据，通过对该算法的实验数据的分析发现，该算法所抽取的数据在结构上存在一定的问题，对于可重复项的确定有一定的错误。

要完成从 HTML 到 XML 的转换，首先必须考虑 HTML 文档的结构。HTML 文档的内容是有顺序关系的，在转换过程中要求保持这种有序性，同时还要保证转换前后信息的完整性。其次虽然二者语法上都源于 SGML 的思想，但 HTML 不允许用户定义自己的扩展标签，而 XML 可以让用户自行定义标记及属性名，从而结构化地描述信息内容。国内迄今为止的研究基本上是处于包装器的半自动生成阶段，尚未见到自动识别网页并产生包装器抽取数据的方法的有关研究资料。国内较为典型的系统和算法有：

1.中国人民大学数据与知识研究所提出的基于预定义模式的包装器[46]，由用户定义模式并给出模式与 HTML 网页的映射关系，接着系统推导出规则同时生成包装器。

2.中科院软件所提出的基于 DOM 的信息提取[13]，该算法以文档对象模型(DOM)为基础，把所要提取的信息在 DOM 层次结构中的路径作为信息抽取的“坐标”，并以这个基本原理为基础设计了一种归纳学习算法来半自动化地生成提取规则，然后根据提取规则生成 Java 类，将该类作为 Web 数据源包装器组成的重要构件。

3.中国科技大学提出的基于多叉树的 HTML 到 XML 的转换方法[45]，基本思想是构造一棵完整的 HTML 多叉树，并且在构造 HTML 树的同时，采用一些策略，将 HTML 树中不严格的语法消除，最后再由 HTML 树来产生相应的 XML 文件，这样就得到了格式良好的 XML 文件。

4.西安电子科技大学所提出的一种基于内容的 HTML 到 XML 转换策略[13]，通过对 HTML 结构和语法特点的分析，定义了一种分离 HTML 格式信息与表达有效语义的内容数据的标记规则，建立了该标记规则到 XML 模式的一种影射。

5.中南大学所提出的一种基于栈结构的 HTML 到 XML 的转换方法[16]，通过对 HTML 与 XML 结构和语法特点的分析与比较，阐述了消除 HTML 格式中不严格语法的通用算法，最后给出了输出 XML 格式的算法。

对 HTML 向 XML 转换的研究主要是集中在形式转换的研究上，实现真正意义上的 HTML 向 XML 的转换，所需工作量大且方法非常复杂，而现在最迫切的任务就是要研究合适的转换方法来实现将大量的 HTML 文档转换为 XML 的形式。

论文借鉴上述研究的一些思想，提出了一种新的转换算法，出发点是为了简化 HTML 到 XML 的转换工作，可以方便地形成 XML 数据源，为下一步的信息抽取作有效的铺垫。

4.2.2 HTML 和 XML 格式比较

从语法上看，HTML 和 XML 没有本质的区别，两者都是源于 SGML (Standard General MarkupLanguage)的思想。只是 HTML 规定了固定数据的标签集合，不允许用户定义自己的扩展标签，而且 HTML 在语法结构上的规格限制是比较松散的，就像控制标记中的英文名称是大小不拘的那样，另外也没有严格要求每个控制标记都要有相应的结束控制标记。这是因为 HTML 对语法结构的要求并不十分严谨，外加浏览器本身对 HTML 文件的容错性很高，即使 HTML 文件中有部分的语法结构错误，浏览器也能将 HTML 文件显示。而 XML 可以让信息提供者根据需要，自行定义标记及属性名，结构化地描述信息内容。另外，HTML 语言是一种不甚严格的语言，很多时候读者直接看到的效果并非是原代码表现出来的效果，而是浏览器很善意的向读者屈服，将一些不完全的代码也正常显示。然而，XML 主要用来存储和发送数据信息，所以其语法规则必然要求得非常严格，具体表现就是所有的元素必须有闭合标记。如果不考虑这些差别，XML 只会比 HTML 多出第一行的 <? xml version="1.0"encoding="GB2312"?>而其他的内容都是一样的。

在把 HTML 文件转换为 XML 文件时，所要消除的 HTML 格式中不严格的语法主要包含以下 2 种情形:一是 HTML 中存在唯一的根元素，但是起始标记和结束标记定界的元素并不是逐层嵌套的，即一些标记不需闭合，而一些标记可以是开放的（即此标记可用闭合标记也可不用），但是在浏览器中不会引起任何问题。二是标记嵌套不恰当，如示例仍然可以正常工作。以上情形在 XML 中是不允许出现的。

对于一个格式良好的 HTML 文件来说，下列一段简单 HTML 文本及相应的 XML 文本，只需要在首行加上 XML 的第一行代码，同时将相应的文件扩展名.html 改为.xml，就转换为 XML 文件了。那么也就是说，对于 HTML 格式到 XML 格式转换的关键就是将 HTML 文本消除不严格的语法，形成格

式良好的 HTML 文件，使得起始标记和结束标记定界的元素逐层嵌套。

(a)一段简单的格式完整的 HTML 文本

```
<html>
<head><title>学校</title></head>
<body>
<H1>哈尔滨工程大学</H1>
</body>
</html>
```

(b)相应的 XML 格式

```
<?xml version="1.0" encoding="GB2312" ?>
<html>
<head><title>学校</title></head>
<body>
<H1>哈尔滨工程大学</H1>
</body>
</html>
```

图 4.1 HTML 与 XML 结构比较图

由于 HTML 的有序性，那么可以运用数据结构中的栈结构来检验 HTML 文件的良好性，控制栈的入栈和出栈来达到检验的目的，运用二叉链表的插入和删除来完成 HTML 文档的起始标记和结束标记的逐层嵌套。

4.2.3 转换原理与步骤

对于一个格式良好的 HTML 文件来说，一段简单的 HTML 文本及相应的 XML 文本，只需要在首行加上 XML 的第一行代码<? xmlversion '1.0'encoding='GB2312'?>，同时将相应的文件扩展名.html 改为.xml，就转换为 XML 文件了。也就是说，对于 HTML 格式到 XML 格式转换的关键就是将 HTML 文本消除不严格的语法，形成格式良好的 HTML 文件，使得起始标记和结束标记界定的元素逐层嵌套。

由于XML严格的语法，所有的XML标记都必须有一个匹配的结束标记，有一个包含所有其他元素的父（或根）元素，那么可以运用数据结构中的栈结构与二叉链表数据结构来检验 HTML 文件的良好性，通过控制栈的入栈和出栈来达到检验的目的，通过控制二叉链表的插入和删除来完成 HTML 文档的规范化，这也是整个方法的核心。最后输出转换后的 XML 文档。整个转换过程分为以下几个子过程。

1.解析 HTML 文档为二叉链表结构。

2.对二叉链表中的结点依次入栈来检验 HTML 文档中标签的的嵌套性。对于不嵌套的标签，进行出栈处理，并在二叉链表中完成嵌套的完善处理，

进而消除 HTML 文本中不严格的语法，然后输出格式良好的 HTML 文件。

3.按照生成的格式良好的 HTML 文件，输出相应的 XML 文件。

4.3 HTML 到 XML 转换算法

由于 HTML 只描述数据的展示即数据看起来是什么样子，是机器不能理解的仅供浏览的文本，而 XML 则描述数据的内容，是机器能理解的智能文本。因此，将 HTML 文档转换为 XML 文档是十分必要的。本文提出了一种新的转换算法，它是以栈结构和链式的数据结构为基础的。

4.3.1 解析 HTML 文档为二叉链表结构

采用的“引用”功能来实现链表的操作。引用是用来“引用”某一个数据对象的，它除了表示某块内存地址外，还能表示其他信息，如该数据对象的类型等。

二叉链表结点的 2 个链域分别为指向该结点的第 1 个孩子结点和下一个兄弟结点，命名为 firstchild 和 nextsibling，链表结点如图 4.2 所示。

图 4.2 二叉链表结点

二叉链表结点的程序实现如下：

```
Class CSNode
{public String data;
Public CSNode firstchild,nextsibling;
Public CSNode(String value)
{data=value;
Firstchild=null;
Nextsibling=null;}}
```

依据对 HTML 遍历过程中访问各结点的顺序特点，可以很容易地抽取文档中的信息并以一个二叉链表的形式进行存储表示。HTML 文档可以看成是

由文本与 HTML 标签组成的文档，每个标签作为一个结点，每相邻两个标签中间的文字作为一个结点。按照这样的规律，HTML 文档就可以被看成是由标签与文本信息组成信息块，然后将每一个信息块作为一个元素，连接成为一个二叉链表。

4.3.2 消除 HTML 的不严格语法

使用栈结构与二叉链表结构消除 HTML 文本中不严格的语法，输出格式良好的文件。为了实现 HTML 的栈结构表示，定义了一个字符串数组 S，作为数据元素。

其过程是依次从二叉链表中读入字符，判断读入的字符是为“<”还是为“</”，并且记住每一个“<”的位置。若为“<”，则表示标签的开始，将整个标签，即以最近的“>”结尾的字符串读入栈中。若为“</”，则将标签内容与栈顶元素比较，若相同，则栈顶出栈，表示该层起始标记和结束标记定界的元素是逐层嵌套的。若不相同，那么栈顶元素依次出栈，直至将与现有标签内容相同的栈元素出栈，也就是说，从现有标签内容开始和结束的标记中所包含的定界元素不是逐层嵌套的。即每个栈元素出栈时，都将在原 HTML 文件中相应的二叉链表中的位置补上结束标记，其相应的位置为栈顶元素所对应的下一个“<”出现的位置。即若栈元素为“B”，则在下一个“<”出现的位置补上“</B>”标记。增加的方法为，如果找不到与读入的结束标记相等的节点元素，可能是原 HTML 文件中不存在相等的节点元素，也可能是原 HTML 文件中存在标记嵌套错误，但此嵌套错误已被前一个读入结束标记修正了，故找不到与此读入结束标记相匹配的起始标记。如示例，当读入时，根据上面的方法会在前自动加入；当再读入后面的时，在 L 中就找不到匹配的，故应删除 HTML 中的。

```
While((ch=fgetc(HTMLfilestream))! =EOF)
{If (ch= ='<')
  {ch=fgetc (HTMLfilestream);
 If (ch= ='/')
{将“</”和“>”的内容读入并赋值给字符串变量 s 作为读入结束标记。如果
```

s 的值等于栈顶的元素的值，则将栈顶元素出栈。否则，栈顶元素依次出栈，直至与现有标签内容 s 相同的栈元素出栈，每个栈元素出栈时，在二叉链表中对应的下一个“<”出现的位置补上相应的结束标记。然后继续读二叉链表结点。}

Else

读入字符串将'<'与'>'之间的内容作为一个元素加入栈中；

}}

如果读完二叉链表结点后，栈中还有除头节点外的元素存在，则把这些元素的内容分别加上“</”和“>”后插入到二叉链表尾部。

现在来分析一些特殊的情况，对算法进行一些修正：

1.对于 HTML 中含有 script 脚本，由于脚本中的表达式可能包含“<”、“</”字符，在读入 HTML 文本时若不对其进行处理，则会影响算法的正确性。所以，记录是否进入脚本区很重要。

2.对于 HTML 中常含有的注释行，某些 HTML 文件其头部常含有的一些特殊段，甚至标签中的某些属性参数的设定，都可以在转换的格式良好的 HTML 文本中省略。只要这些内容仅仅是页面的显示，而与信息抽取的内容无关。

3.不包括在以上两种情况的其他特殊情况，需要根据实际情况进行特殊处理。

4.3.3 XML 文档的输出

若把二叉链表在结构形式上看作一个二叉树的话，对二叉链表的遍历有先根遍历、中根遍历、后根遍历等方式。根据 HTML 文档映射到二叉链表的过程，对于一个结点，它的 first child 域指向前一个结点，而 next sibling 域则指向其后一个结点。改进的遍历算法复杂度没有发生变化，这有利于 XML 文档的生成。

输出 XML 文件这个子过程很简单，只要将前一步生成的 HTML 文本的开始加上<?xmlversion=，，1.0，，eneoding='GB2312'?>即可。输出过程如下，其中 stream 为 HTML 文件。

```
voidoutputMXL(stream)
{
fprintf("<?xmlversion='1.0' encoding='GB2312'?>\n");
fprintf(stream):
}
```

使用基于栈结构的 HTML 到 XML 的转换方法，有效地将 HTML 文件转换为 XML 格式。简化了信息抽取工作，可以方便地形成 XML 数据源。为处理 XML 数据，并提取出适当的数据模式做了有效的铺垫。

4.4 本章小结

本章首先分析了现有的 HTML 文档到 XML 格式转换算法，指出了他们的不足指出，最后给出了一种基于栈结构与链式结构的从 HTML 格式到 XML 格式的转换方法，详细阐述了消除 HTML 格式中不严格语法以及输出 XML 格式的通用算法。

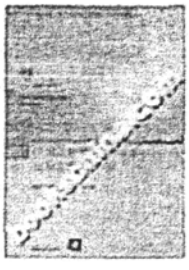
第 5 章 基于 Suffix Tree 的数据模式提取算法研究

从样本文档中归纳学习页面模式是信息抽取系统中的重要环节，模式提取模块就是针对这一任务而设计的。本章参考文献[9]的方法，借助Suffix Tree结构给出了自动提取Web页面数据模式的过程。

5.1 模式提取的方法介绍

数据模式是数据的概念、组成、结构以及相互关系的总称，涉及到数据的描述范围、描述的方式和描述的结果。Web 页面上的数据具有无结构或半结构化的特征，上面隐含的数据模式计算机无法直接识别，需要借助某种方式提取出页面隐含的数据模式。

比如，在一个知识密集型 Web 站点的页面中具有如图 5.1 所示的图书信息，它由 Web 语言直接显示在浏览器中。



书名：《中国计算机学会学术年鉴报告2005》

作者：中国计算机学会学术工作委员会 主编　　出版社：清华大学

出版日期：2006-8-1　　ISBN：7302135037

原价：40元　现价：34元　您节省了6元

图 5.1 Web 页面中显示的图书信息

在浏览器上，可以直接看出图书的基本信息，如图书名称、作者、出版社、出版日期、图书号等等。然而，由于 Web 页面只是关注信息的显示方式，不去关注页面的结构和数据分布，使得计算机无法自动识别上述图书信息，无法理解各种标识的语义。计算机只能识别如图 5.2 所示页面 XML 源代码中的各种标记，根据 Web 页面的 XML 源代码查找有限的信息。

```
table id=table2 cellspacing=0 cellpadding=0 width=480 boı de1=0
tbody
tr - td <<中国计算机科学技术发展报告 2005>> td - tr
tr - td 中国计算机学会学术工作委员会 主编 td
td 出版社：清华大学 td - tr
tr - td 出版日期：2006-8-1 td
td - td ISBN: 7302135037 tr
tr - td 原价：40元 现价：34元 您节省了6元 td - tr
tbody
table
```

图 5.2 Web 页面的 XML 源代码

于是，需要从样本文档中归纳学习数据模式，变成形式化的描述。上述图书信息的文档结构描述可抽象成如下形式：<table><tbody><tr><td>Text()</td></tr><tr><td>Text()</td><td>Text()</td></tr><tr><td>Text()</td><td>Text()</td></tr><tr><td><Text></td></tr></tbody></table>

其中 Text()代表图书的所有文字信息。为方便起见，把上述标记简记为：<book><title></title><author></author><others></other></book>。

分析发现，在此类知识密集型站点的 Web 页面上包含多个图书信息（一般一个页面中会列出十个左右的图书信息），也就是标记串<book><title><title><author></author><others></others><book>会在页面中重复出现多次。当然，除此之外还有很多无效的标记串混杂在 XML 页面中，比如站点的广告信息，为方便使用而设置的工具条等等。模式提取的任务就是在复杂的 XML 页面中，消除冗余标记的干扰，提取出包含有用信息的数据模式，也就是描述图书信息的子串<book><title></title><author></author><others> </others> </book>。

针对这种知识密集型 Web 页面的数据分布特点，本文参考文献[32]的方法，借助 Suffix Tree 结构给出了自动提取 Web 页面数据模式的过程，模式提取的过程如图 5.3 所示。

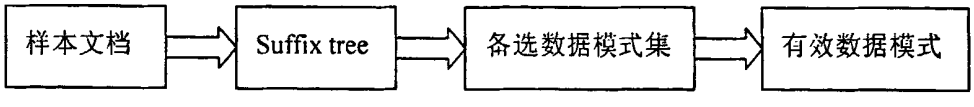


图 5.3 模式提取流程图

5.2 样本文档的 Suffix Tree 解析

样本文档的解析是数据模式提取的基础，然而 Suffix Tree 以文本所有后缀为关键词的 Patricia 树。后缀树的引入主要是针对字符串的高效查找子串查找、最长重复子串、最长公共子串、回文子串。本文正是利用 Suffix Tree 的最长公共子串提取方法，来实现备选数据模式集的生成，同时利用最长重复子串提取算法完成了有效数据模式提取的。

后缀树（suffix tree）是存储了给定字符串的所有后缀的压缩 trie 树[53] (compacted trie)。它将关键词作为二进制位串以树的形式索引起来，从根结点到叶子结点的每一条路径都代表一个关键词位串，从而使原字符串的任何一个子串都出现在树中由根节点开始的某一路径上。在 Suffix Tree 中，关键词的具体信息都保存在叶子结点上，Suffix Tree 的内部结点则用来记录关键词的路径，它有三个基本的数据项：比较位、左指针、右指针，其中，左指针和右指针分别指向该结点的左、右子树，比较位记录的是从根结点到达该结点的所有位串中第一个不相同位的位置。由于比较位的存在，途经该结点的位串将选择不同的后继路径。由于只比较不同的位，所以 Suffix Tree 的查询速度极快。

基于这种数据结构，可以简单而高效的解决很多复杂的字符串处理方面的问题，如在线性时间内解决字符串查找和匹配问题，快速发现和定位字符串中的重复子串问题等。因此，后缀树被广泛应用在与字符串处理相关的各种领域里，如生物信息学、分子生物学以及数据处理、信息检索、数据压缩和信号处理等领域[54]。

正是由于引入了比较位，所以避免了对关键词的逐位比较，保证了 Suffix Tree 中每一个内部结点都有子树上。

结合图 5.1 中提取出来的图书信息的描述可抽象成如下形式：<table><tbody><tr><td>Text()</td></tr><tr><td>Text()</td><td>Text()</td></tr><tr><td>Text()</td><td>Text()</td></tr><tr><td><Text></td></tr></tbody></table>，分析发现，该字符串是由有限的文档标签加上<Text>所组成的。如果能够将这些有限的标签进行二进制编码，那么该文档的后缀树上最多只能有两个分支，既方便了比较查询。

本文中后面提到的后缀树将都是经过编码的字符串组成的即全部是由 0 与 1 组成。这也就表明，当 Suffix Tree 中存有 n 个关键词即 n 个叶子结点时，其内部结点为 n-1 个，总结点数为 2n-1，所以 Suffix Tree 的空间复杂度为 O(n)，其中 n 为所有字符串的长度总和。

另外，本文引用后缀树的优势还有以下几点原因：

（1）Suffix Tree 构建后查找和遍历复杂度为 O(m)，m 为查找串长度。正是由于后缀树有较小的查找和遍历复杂度，在最大字符串匹配算法中，节省了时间，即提高了备选模式集生成的时间效率；

（2）后缀树可以处理参数匹配，可以将程序变量作为字符串参数作条件匹配。因此，方便了在提取有效模式时，计算两个字符串相似度；

（3）构建后缀树的时间复杂度是线性的，为 O(nlog min (|sigma|,n))，其中|sigma|为字母表大小，n 为串长度；

（4）有成熟的后缀树构建并行算法，即提供了为字符串构建后缀树的良好基础。

5.2.1 Suffix Tree 的定义

定义 5.1 字符串的后缀：

设有字符集合 Σ，及字符串 S∈Σ+；对所有字符串 P，称 P 为 S 的后缀当且仅当 P=S[i..|s|]，其中 i=1，2，…，|s|，|s|为串 S 的长度。空串 ε 也是 S 的后缀。实际上，ε 是任何字符串的后缀。设有字符串 S_1[1..m]，S_2[1..n]∈Σ+，且 m≤n，若有 S_1[1]=S2[i]，S1[2]=S2[i+1]，…，S_1[m]= S_2[i+m-1]，1≤i≤n-m+1，则有 $S^1 \subseteq S^2$。

定义 5.2 后缀树：

字符串 S 的后缀树 T={V，root，E，T，Σ}，是以 root 为根节点的一棵树，其中：V 是树中所有的节点的集合，且 root∈V；E 是树中标记边的集合；T 是树中叶节点集合，且 TV。n∈V，有 $l_1l_2...l_i$ 与之对应，$l_1l_2...l_i$ 是起始于 root 终止于节点 n 的路径上所有边标记的连接，称为节点 n 的路径，且 T 中所有节点的路径都是唯一的；$|l_1l_2...l_i|$ 为节点 n 的深度。除了根节点 root 外，V 中每个内部节点有至少两个子节点，且每条标记边上的边标记均为 S 中的非空

子串。节点 n 发出的任意两条标记边的边标记以 Σ 中不同的字符开头。由于 T 中每个叶节点的路径都与 S 的一个后缀对应，故称 T 为 S 的后缀树。

输入字符集：字符串中可能出现的所有字符的集合。

后缀：假设字符串 $S=s_1s_2...s_i...s_n$，其中，s_i 属于输入字符集，那么 $S_i=s_is_{i+1}...s_n$ 是 S 从位置 i 开始的后缀。例如：假设有 D＝abab；那么：D_1=abab；D_2=bab；D_3=ab；D_4 =b；

后缀树：一个由 n 个字符构成的字符串 S，它的后缀树是一棵有根的有向树，共有 n 个叶子，分别标记为 1 到 n。每一条边都用 S 的非空子串来表示。从任意一个节点出来的两条边，它们必须以不同的字符开始。从根节点到叶子节点 i 顺序经过的树边的串联，恰好为 S 从 i 位置开始的后缀，即 Si。

外在状态：如果从根节点出发，一个子串在后缀树的某一节点，那么这个子串表示的状态被称为外在状态；反之，则称为内在状态。

后缀链：假设有子串 $D_i=\alpha D_i'$，其中 D_i 表示外在状态，且结束在节点 υ，α 表示一个字符，如果存在 Di'也表示外在状态，且结束在节点 υ'，那么从 υ 指向 υ'的指针被称为后缀链，记作 $s_i(\upsilon)=\upsilon'$。

根据定义 5.2 可知，一个具有 m 个字符的字符串 S 的后缀树 T，就是一个包含一个根节点的有向树，该树恰好带有 m 个叶节点，这些叶节点被赋予从 1 到 m 的标号。后缀树的关键特征是：对于任何叶节点 i，从根节点到该叶节点所经历的边的所有标识串联起来后恰好拼出 S 的从 i 位置开始的后缀，即 S[i..m]。对于 S 的每一个子串 v，树中存在一个路径为 u 的节点，v 是节点 u 的路径所表示的串的前缀。为了使 S 的后缀中不存在任何一个后缀是其他后缀的前缀，通常在字符串的末尾加入一个特殊字符$，它不出现在 S 中。这就保证后缀树有|s|+1 个叶节点，代表着 S$的所有不同的非空后缀。后缀树中的边和节点可以用它们在 S 中的起始位置和长度表示，因此，后缀树可以被存储在 O(|s|)的空间中。

5.2.2 Suffix Tree 的构造方法

利用后缀树来解决模式提取相关的问题时，首先需要建立后缀树，因此，如何高效地构造后缀树便成了问题的关键。

后缀树是一种数据结构，它支持有效的字符串匹配和查询，人们在对其研究方面作了大量的工作。目前，对于串行的后缀树构造算法的研究已经非常成熟，人们提出了很多线性时间和空间的算法。1973 年，Weiner[22] 第一次提出了后缀树这种数据结构，并给出了线性时间构造算法，该算法使用了较多的时间和存储空间，在目前它已经没有使用的必要了，但是在字符串处理领域仍然是一个历史丰碑。几年后，McCreight[23] 提出了更节约空间和降低时间复杂度的算法，他使用后缀链接(Suffix Link)等技术来使该算法的时间复杂度减少到线性。但是他们的算法描述都很晦涩，难以理解，从而限制了后缀树的广泛使用。直到 1995 年，Ukkonen 提出了改进的线性时间建树算法，该算法使用了后缀链接技术，其最大特点是它是一种从左到右构建字符串的后缀树的在线(On-line)算法，也就是说算法的任何阶段都生成当前子串的后缀树，同时它易于理解，而且具有在线特性，所以，Ukkonen 提出的算法在应用中被广泛接受。文献[62] 对后缀树及其应用作了较为全面的阐述。鉴于上述比较，本文在实现中将采用 UkKonen 算法。下面，给出后缀树构造算法的基本思想。

依次为字符串的每个前缀构造后缀树，当前构造的后缀树是在上一棵已建成的后缀树的基础上得到的。若字符串 S 的长度为 n，则将算法分成 n 个周期，每个 i+1 周期又被分为 i+1 个扩展，在第 i+1 个周期的第 j 个扩展，算法首先找到从根节点开始标志子串 S[j,..., i]的路径的结束位置，然后将 S(i+1)加入子串进行扩展，除非 S(i+1)已经存在。因此，在 i+1 个周期，S[1,..., i+1]先被插入树中，以此类推，插入 S[2,..., i+1]，S[3,..., i+1]...，在 i+1 个周期的第 j 个扩展，确保插入了字符 S(i+1)（除非 S(i+1)已经存在）。为了能在 O(n)时间内建立起后缀树，Ukkonen 算法引入了后缀链。

根据算法的构造算法的基本思想，就可以完成样本文档的 Suffix Tree 的构造了，下面给出具体构造方法。

假设有字符串 D，按序将 D 中的字符添加到后缀树中。例如：D=abab$，那么添加字符的次序为：a，b，a，b，$。在添加第 i 个字符的过程中，理论上来说，需要处理所有新增的后缀。例如，添加$时，需要处理的后缀有：abab$，bab$，ab$，b$，$。

在添加过程中，可能出现的三种情况：树中的当前树边需要扩展，例如：

当添加第 2 个字符 b 时，需要将表示字符串 a 的边扩展成表示字符串 ab 的边，见图 5.4。

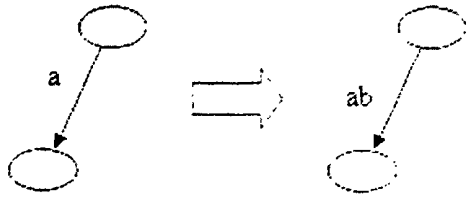


图 5.4 添加 D="abab$"的字符"b"时的扩展操作

解决方法：边用(left，right)来表示，指示其在字符串中的起止位置，例如：(2，4)表示字符串 bab。对于新添加的边，用(left，∞)来表示。其中 left 是其在字符串中的起始位置，∞表示当前处理的位置。那么处理第 1 个字符 a 时，∞表示 1；处理第 2 个字符 b 时，∞表示 2，依次类推。

在现有结点上添加新边，例如：当添加第一个 b 时，还需要在根结点下添加表示字符串 b 的边，见图 5.5。

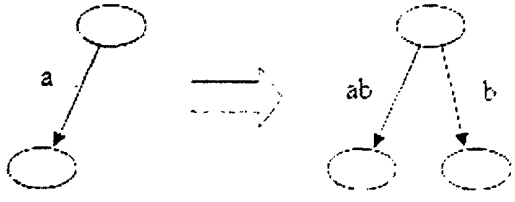


图 5.5 添加 D="abab$"的字符"b"时的添加操作

将边断开，添加中间结点和叶子结点。例如：在添加$的时候，需要将表示 abab 的边断开，添加中间结点和表示$的叶子边。见图 5.6。

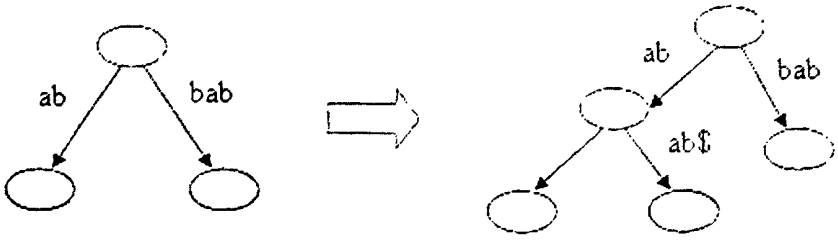


图 5.6 添加 D="abab$"的字符" $"时的分裂操作

5.2.3 样本文档解析举例

在一些 Web 页面中，待抽取的信息经常被组织在一定的结构中，在页面上有一定的重复模式。比如，电子商务型站点中，产品信息以一种固定的格式在页面中连续显示多次，产品信息被组织在一个模板中。如果忽略显示的具体产品内容，则可以识别这种隐含在页面中的模板。如在图 5.7 所示的例子中，所有的文字描述，如英语、80 等等，都用 Text 取代，则字符序列“<LI>Text()<I>Text()</I>”总共重复出现了两次。

<h1>显示各科成绩</h1>

英语：80
数学：97

图 5.7 Suffix Tree 举例

由于标记是组成数据显示的主要组成部分，而标记之间的文字字符串才是真正从浏览器中看到的内容。因此，可以把两个标记之间的文字字符串看作一个描述单元，就像是一个单独的标记符号一样。在下面的描述中，把所有的文字描述记作 Text()，每一个<tag>标记用各自的名字表示。如是一个标签标记；Text()是一个文字描述标记，用来表示包含在两个标签标记之间的文字信息。

标签标记有多种分类，可以根据自己的需要确定使用哪种分类。比如，标记 BODY 中所包含的标记可以被分成两组：块级标记和文本级。块级标记用来定义文档的结构，文本级标记用来定义某些特征。

块级标记包括头部标记(<head>)，列表标记(<UL>)，表格标记(<table>)等。文本级记包括在文字描述块中用来修饰文字的标记，比如<I>，<font>等。这样一来，如果只考虑块级标记，则上面例子中可以记成标记串<h1></h1>
。下面，对这个标记串构造一棵后缀树。

为了保证后缀树中存储的是二进制位串，便于查询，所以首先要对上面的标记进行编码个标记都用唯一的固定长度的 0、1 二进制位串来表示。编码标准主要有三种:

（1）使用字符 *ASCII* 码唯一地表示每个标记串中的英文字符，从而每个标

签将是唯一的二进制串形式。

（2）得到标记串中所有不同标记的集合$\sum_{token}$，取二进制串的长度为：

$l=\log_2\left(|\sum_{token}|+1\right)$。

（3）得到标记串中所有字符的集合$\sum_{character}$，取二进制串的长度为：

$l=\log_2\left(|\sum_{character}|+1\right)$。

这里选择第二种方式作为编码依据。在上面的标记串中取二进制长度为3，对应的编码如表5.1所示。

表5.1 Suffix Tree举例编码表

<h1>	000	</h1>	001
	010		011
	100		101
Text()	110		

所以整个标记串用对应的二进制编码为：000 001 010 100 110 100 110 011$。总共3×8个二进制位，其中$表示字符串的结尾。

下面再求出该标记串的所有可能的后缀子串，如下所示:

Suffix1 000 001 010 100 110 100 110 011$

Suffix2 001 010 100 110 100 110 011$

Suffix3 010 100 110 100 110 011$

Suffix4 100 110 100 110 011$

Suffix5 110 100 110 011$

Suffix6 100 110 011$

Suffix7 110 011$

Suffix8 011$

图5.8示意了该后缀树的结构。内部节点用圆来表示，叶节点用矩形来表示。矩形中的数字表示对应后缀的开始位置。

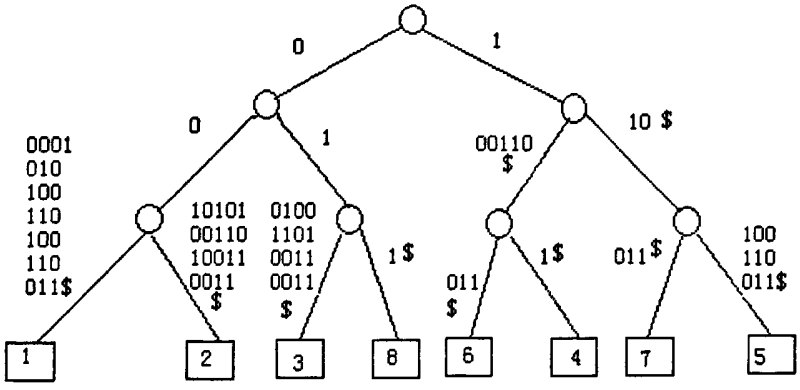


图 5.8 字符串的后缀树示例

5.3 备选数据模式集的生成

每一个样本文档的 HTML 源码都可以看作一个输入串，用来构造一棵 Suffix Tree。Web 文档是由 HTML 标记和元素（即产品信息等）组成的，而文档中的所有产品信息的文字描述可以看作一个元素 Text()。

实际上，一般的 Web 页面结构都比较复杂，不可能直接找出页面的有效数据模式。但是，可以借助文档的 Suffix Tree 结构首先得到页面的备选数据模式集合，也就是在文档的 HTML 源码中重复出现的子串集合。理论上，Suffix Tree 中每一个内部结点对应的字符串都可以看作输入串中的重复子串，但是，一个长度为 n 的输入子串，会有 n-1 个内部结点，而这 n-1 个内部结点中仅有少量结点对于提取数据模式有用。

5.3.1 相关定义

在备选数据模式集生成算法中，涉及到的基本定义如下：

P：模式字符串，长度为 n；

T：待匹配字符串，长度为 m；

S(P$)：字符串 P 的后缀树，$作为终止符号，可以使每个后缀都在叶子节点结束。

(x, l , r)：对应着子串 P[,l..., r]，且 P[l]=x；如果节点 u，v 之间有一条边 (x, ,l r)，则：son(u, x)=v， first(u, x)=，llen(u, x)=r-l+1；

M(T, P)i：表示字符串序列 T 起始 i 位置的后缀与字符串序列 P 的最长公共子串的长度，且该最长公共子串必须以 T[i]为首字符，这些 M(T, P)i 的值称作匹配数据。例如，T=华硕拒录乙肝求职者被告上法庭，P=学生求职因乙肝被拒状告华硕，则 M(T, P)1=2（华硕），M(T, P)5=2（乙肝），M(T, P)7=2（求职）。

5.3.2 备选数据模式集的生成算法

Matching Statistics 算法的基本思想是：首先为字符串 P 构造一棵后缀树 S(P$)，然后搜索该后缀树，找到匹配 T[i...m]的最长前缀的位置，其经过的树边上字符串的总长度即为 M(T, P)i 的值。从而，可以从 M(T,P)i 知道字符串 T 和 P 的对应公共子串的长度及该子串在 T 中的位置，即 M(T,P)i 的值和 i。于是，得到一个备选数据模式。对所有样本文档后缀树进行两两比较，便可以得到备选数据模式集。

同时，为了进一步去掉无效的数据模式，本文还增加了下面三个约束条件：

①规定一个频率阈值 φ_{fre}，只有在 S 中出现频率大于这个频率阈值的最大重复子串才是一个备选数据模式子串。即 $if\ \ length(\alpha_i) < \varphi_{length}\ then\ \alpha_i \notin \sum_{candidate}$。

②规定一个长度阈值 φ_{length}，只有长度大于这个长度闭值的最大重复子串才是一个备选数据模式子串。即 $if\ length(\alpha_i) < \varphi_{length}\ then\ \alpha_i \notin \sum_{candidate}$。

③以 HTML 结束标记（如</html）为起始标记的标记子串不是一个备选数据模式子串。

根据最大重复子串的定义和以上几个约束条件，从文档的 Suffix Tree 结构中筛选得到页面的备选数据模式集合 $\sum_{candidate} = \{\alpha_1, \alpha_2, ..., \alpha_n\}$，其中频率阈值和长度阈值的取值一般为一个经验值。

5.4 有效数据模式提取算法

有效数据模式提取的目标是从备选数据模式集$\sum_{candidate}$中筛选得到有效数据模式集$\sum_{effective}$，有效模式集中子串的个数在满足筛选条件的基础上越少越好，便于提取数据模式。最终希望得到一个有效数据模式即可。

有效数据模式反映了 Web 文档的数据结构和待抽取的信息点特征，得到了文档的有效模式也就掌握了文档的数据分布规律，从而为抽取规则的生成奠定了基础。

5.4.1 有效模式提取的依据

本文给出了相似度计算方法作为有效模式提取的依据：

根据文献[63]中测度相似性系统相似度的基本方法，计算字符串的相似度，一般考虑两方面的因素：相似字符的数量和相似字符的相似元数值大小。由于本文只是针对备选模式集进行相似度计算，因此只考虑字符串的字面特征，因此，采用最长公共子序列法来计算相似度便可以满足要求。

定义 5.3：一个给定序列的子序列的条件是在该序列中删去若干元素后得到的序列。确切地说，若给定序列 X=<$x_1, x_2, \ldots, x_m$>，则另一序列 Z=<$z_1, z_2, \ldots, z_k$>是 X 的子序列的条件是指存在一个严格递增的下标序列<$i_1, i_2, \ldots, i_k$>，使得对于所有 j=1, 2,..., k，有 $X_{ij}=Z_j$。给定两个序列 X 和 Y，当另一序列 Z 既是 X 的子序列又是 Y 的子序列时，称 Z 是序列 X 和 Y 的公共子序列，最长公共子序列即所有公共子序列中长度最长的。D.S.Hirschberg 提出的算法[64]可以在 O(nm)时间内求解出两个字符串的最长公共子序列。

假设字符串 S_1 的长度为 m，字符串 S_2 的长度为 n，S_1 和 S_2 的最长公共子序列的长度为 k，则 S_1 和 S_2 的相似度为：Sim(S_1, S_2)=max($k/m, k/n$)。

此时，S_1 的长度 m=20，S_2 的长度 n=20，最长公共子序列（多以吵闹讨公道）的长度 L=7，则 S_1 和 S_2 的相似度 Smi (S_1, S_2)=0.35。

定义：评估函数score为一个字符串可以有多个模板，采用评估函数score

来选择最好的一个。假设字符串为X，模板为Y，X针对模板Y的k个分割点依次为 $p_1, p_2, ..., p_k$， $str(p_i)$ $(0 \le i \le k)$为X中从 p_i 开始的符合模板Y的子串。length(str(pi)) 为str(pi) 的长度。

$$score = \frac{\sum_{i=1}^{k} length(str(p_i))}{length(x)} \qquad (5\text{-}1)$$

score 的数值越大，所有的str(p_i) $(0 \le i \le k)$对X的覆盖率越高，模板也就越好。

定义5.4：通过遍历后缀树bs，得到bs 的任何一个子串α，计算子串α对bs 的覆盖度。覆盖度最大、子串长度较小的子串α为最优分割子串。

5.4.2 有效数据模式提取算法

正是利用了Suffix Tree的最大重复字串算法，完成了有效数据模式的提取。算法描述如下：

（1）对所有备选模式构造后缀树，通过遍历后缀树 b_i，得到 b_i 的任何一个子串 α_i，计算子串 α_i 对 b_i 的覆盖度。覆盖度最大、子串长度较小的子串 c_i 为 b_i 的最优模式，并根据上面介绍 c_i 的最长公共子序列法计算出相似度，执行（2）。

（2）规定一个相似度阈值 φ_{fre}，如果相似性系数平均值大于该阈值，执行下一步；否则执行（1）。

（3）对每个备选模式计算 $M(a_i)=(a_{i0}+a_{i1}+...+a_{im})/n$ 值，取出最大的模式 X，转下一步。

（4）计算 X 的最佳重复标记序列 γ，然后使用 M 替换单个或者连续的 γ，这样产生 β

（5）计算 β 的最佳重复标记序列 δ。If δ 为空，γ 最优数据模式。If δ 不为空，根据评估函数选择一个最优数据模式。例如：

例1 :如果α="A ,X ,Y,X ,Y,X ,Y,B ,A ,X ,Y,X ,Y,B ,A ,X ,Y,X ,Y,X ,Y,X , Y, X ,Y,B"得到最优分割子串γ="X ,Y,"；使用M替换γ和连续的γ，得到δ="A ,M ,B ,A ,M ,B ,A ,M ,B"；得到β="A ,M ,B ,"score (γ) = 20/ 26 score (β,γ)

= 26/ 26 = 1 。因为score (β,γ) > score (γ)，模板(β,γ) 即“A , (X ,Y,) ×B”为最优数据模式。

例2 :如果α=“tr ,tr ,tr ,tr,tr”，得到最优分割子串γ=“tr”；使用M 代替γ和连续的γ得到δ= M；得到β= null。因为β为空，“tr”为最优数据模式。

根据以上算法思想，对备选数据模式集进行筛选，得到样本页面的有效数据模式作为整个站点中Web页面的数据模式，也就是在Web页面中连续重复出现的某个HTML标记子串。在知识密集型站点中希望得到描述产品信息的模式子串，比如在图书销售的Web页面中，希望得到详细描述图书信息的模式子串<book><title></title><author></author><others> </others> </book>。

5.5 本章小结

本章首先介绍了Suffix Tree的基本知识，给出了Suffix Tree的基本定义和基本方法。利用Suffix Tree结构提取出备选数据模式集，最后给出有效模式提取算法。

第 6 章 实例验证与分析

本章以前几章的理论为基础，在实践中设计了一个小型的图书领域本体，并构建了一个小型的 Web 信息抽取系统。

在准备阶段，首先从当当图书网上下载了 150 个页面，作为实验对象。然后从里面选取几个页面作为样本页面，归纳出站点的整体数据模式，构造领域本体，生成抽取规则，对其余页面进行信息抽取。

6.1 样本页面的获取及处理

首先要从 Web 站点中选择下载一定数量的页面作为实验对象，抽取里面的几个作为样本页面。下载页面前首先要指定信息源的网页的 URL，系统将以该 URL 为基础获得目标网页。

实验中首先得到了如图 6.1 所示的样本页面。其对应的 HTML 源代码如图 6.2 所示。

计算机图形学基础教程

顾客评分：☆☆☆☆

孙家广　胡事民

清华大学出版社

本书是介绍计算机图形学基本原理和最新应用的基础教材。它是作者在清华大学多年教学经验的基础上，同时参考了国内外最新相关资料和部分最新研究成果编写而成的。全书按内容分为[illegible]章，分别介绍计算机图形学的

~~¥20.00~~ ¥15.00 折扣：75折 节省：¥5.00

神经网络设计

顾客评分：☆☆☆☆☆

[illegible] 哈根 [illegible] 戴莫

出版社：机械工业出版社

本书主要讲述神经网络的基本概念，介绍实用的网络模型、学习规则和训练方法。全书分13章，内容涵盖神经元模型和网络结构、感知机学习规则、有监督的Hebb学习、Widrow—Hoff学习算法、反向传播算法及其变形、..

~~¥49.00~~ ¥36.70 折扣：75折 节省：¥12.30

图 6.1 计算机类图书样本页面

```
<div class="list_r_list">
<h2>
<a name="link_prd_name" target="_blank">计算机图形学基础教程</a>
</h2>
<h4 class="list_r_list_h4">作　者：
<a target="_blank" name="作者" 孙家广 </a>，
<a target="_blank" name="作者" 胡事民 </a>编著
</h4>
<h4>出版社：
<a target="_blank" name="出版社" 清华大学出版社 </a>
</h4>
<h4>出版时间：2005年02月</h4>
<h5>本书是介绍计算机图形学基本原理和最新进展的基础教材。它是作者在清华
大学多年教学经验的基础上，同时参考了国内外最新相关资料和部分最新研究成
果编写而成的。全书按内容分为5章，分别对介绍计算机图形学的...
</h5>
<div class="clear"></div>
<h6><span class="del">￥20.00</span>
<span class="red">￥15.00</span>　折扣：75折　节省：￥5.00</h6>
</div>
<div class="clear"></div>

<div class="list_r_list">
<h2><a name="link_prd_name" target="_blank">神经网络设计</a></h2>
<h4 class="list_r_list_h4">作　者：[美]
<a target="_blank" name="作者" 哈根 </a>等 著，
<a target="_blank" name="作者" 戴葵 </a>等 译</h4>
<h4>出版社：<a target="_blank" name="出版社" 机械工业出版社 </a></h4>
<h4>出版时间：2002年09月</h4>
<h5>本书主要讲述神经网络的基本概念、介绍实用的网络模型、学习规则和训练方法。
全书分19章，内容涵盖神经元模型和网络结构、感知机学习规则、有监督的Hebb学习、
Widrow—Hoff学习算法、反向传播算法及其变形、...</h5>
<div class="clear"></div>
<h6><span class="del">￥49.00</span>
<span class="red">￥36.70</span>　折扣：75折　节省：￥12.30</h6>
</div>
<div class="clear"></div>
```

图 6.2 计算机图书页面源代码

在提取数据模式时，希望文档具有良好的结构，以便于能够更快更准确的获得页面的数据模式，所以首先要把图 6.2 所示的 HTML 源码转换成结构良好的 XML 文档。在第四章中，已经详细介绍了本文采用的基于栈结构与链式结构的 HTML 文档转换为 XML 文档的转换算法。接下来的任务就是在该 XML 文档的基础上提取数据模式。

6.2 Suffix Tree 编码实现

通过对样本文档的 HTML 代码的分析，得到下面的统计数据：样本文档

中出现的所有 HTML 标记。对文档中的 40 个标记进行编码。编码长度为 6。编码结果如表 6.1 所示。

表 6.1 标签编码对照表

编号	标记名称	编码	编号	标记名称	编码
1	<body>	000000	21	<td>	010100
2	</body>	000001	22	</td>	010101
3	<head>	000010	23	<form>	010110
4	</head>	000011	24	</form>	010111
5	<html>	000100	25	<select>	011000
6	</html>	000101	26	</select>	011001
7	<a>	000110	27	<table>	011010
8		000111	28	</table>	011011
9		001000	29		011100
10		001001	30		011101
11		001010	31	<h3>	011110
12		001011	32	</h3>	011111
13	<tr>	001100	33	<title>	100000
14	</tr>	001101	34	</title>	100001
15	<iframe>	001110	35		100010
16	</iframe>	001111	36		100011
17		010000	37	<link>	100100
18		010001	38	<IMG>	100101
19		010010	39	<script>	100110
20		010011	40	</script>	100111

除了对 HTML 标记进行编码之外，还要对文本标记 Text()进行编码，编码为 101000。最后根据样本文档的 HTML 标记串，找出所有的 HTML 标记后缀子串，就可以构造样本文档的 Suffix Tree 了。本文所选取的样本文档的总长度为 33638，处理后含有 HTML 标记和文本标记 Text()的总个数为 1467 个。可见样本文档的后缀集合比较复杂，不再一一列举，这里仅给出一个构

造 Suffix Tree 的简单过程。

例如：对<td>text()</td>构造 Suffix Tree。

1. 对其编码为：001100 010100 101000 010101 001101
2. 所有可能后缀：Suffix1: 001100 010100 101000 010101 001101$
   Suffix2: 010100 101000 010101 001101$
   Suffix3: 101000 010101 001101$
   Suffix4: 010101 001101$
   Suffix5: 001101$
3. 根据该标记串的后缀集合，构造对应的 Suffix Tree，构造方法如 5.2 节所述。

6.3 构造领域本体

为了从 Suffix Tree 中得到有效数据模式，需要设置相似度参数，本文的参数的设置为 0.7。应用该参数，最终从 Suffix Tree 中得到的描述图书信息的数据模式描述子串为： <h2><a>text()</h2><h4>text()<a>text()text()<a>text()text()</h4><h4>text()<a>text()</h4><h4>text()</h4><h5>text()</h5> <h6>text()text()text()</h6>

从中筛选得到图书的描述概念包括：书名、作者、出版社、出版日期、原价和现价，折扣与节省价格。以这些从 Web 页面中提取出的有效概念为基础，借助 Protégé 工具制作了一个图书领域本体。

最后把本体转化成对应的 OWL 文件，在 OWL 文件中详细描述了各个概念、概念之间的关系，以及概念具有的属性。如：

（1）每一本图书都有一个市场价格，对应的 OWL 描述为：

<owl:ObjectProperty rdf:ID="hasPrice">

<rdfs:label rdf:resource=" #RMB"/>

<rdfs:range rdf:resource=" #Price"/>

<rdfs:range rdf:resource=" #Book"/>

<owl:objectProperty rdf:ID="hasPrice">

<rdfs:range rdf:resource="#Price"/>

<rdfs:domain rdf:resource="#Book"/>

</owl:ObjectProperty>

（2）每一本图书都有一个作者，对应的 OWL 描述为：

<owl:ObjectProperty rdf:ID="hasAuthor">

<rdfs:domain rdf:resource="#Book"/>

<rdfs:range rdf:resource="#Author"/>

</owl:ObjectProperty>

（3）每一本图书都有一个题目，对应的 OWL 描述为：

<owl:ObjectProperty rdf:ID="hasTitle">

<rdfs:range rdf:resource="#Title"/>

<rdfs:domain rdf:resource="#Book"/>

</owl:ObjectProperty>

（4）每一本图书都有一个出版日期，对应的 OWL 描述为：

<owl:ObjectProperty rdf:ID="hasDate">

<rdfs:range rdf:resource="#Date"/>

<rdfs:domain rdf:resource="#Book"/>

</owl:ObjectProperty>

（5）每一本图书都隶属于某个出版社，对应的 OWL 描述为：

<owl:ObjectProperty rdf:ID="belongtoPublication">

<rdfs:domain rdf:resource="#Book"/>

<rdfs:range rdf:resource="#Publication"/>

</owl:ObjectProperty>

书名、作者、出版社、出版日期、价格在 Web 页面中都有一个起始路径，对应的 OWL 描述为：

<owl:DatetypeProperty rdf:ID="hasPath">

<rdfs:domain>

<owl:Class>

<owl:unionOf rdf:parseType="Collection">

<owl:Class rdf:about="#Author"/>

<owl:Class rdf:about="#Date"/>

<owl:Class rdf:about="#Publication"/>

<owl:Class rdf:about="#Title"/>

<owl:Class rdf:about="#Price"/>

</owl:unionOf>

</owl:Class>

</rdfs:domain>

</owl:DatetypeProperty>

其他各属性（起始标记、结束标记等）的描述与此类似。限于篇幅，只罗列图书本体库定义的一部分。在采用 OWL 语言（见 2.3 介绍）描述图书本体中，可以通过标签<rdfs:label>来描述对于同一概念或同一概念间关系的不同表达，包括不同语言以及不同研究组织、机构等的表达。通过标签<owl:Class>和<rdfs:subClassOf>来定义图书领域中的概念以及概念间的关系，另外标签<owl:DatatypeProperty>来定义概念的属性，<owl:ObjectProperty>来表示概念与概念间的关系。另外 OWL 语言的另一特征就是提供了推理的能力。通过标签<owl:TransitiveProperty>来表示传递属性（即 P（x,y）and P（y,z）=>P（x,z）），标签<owl:SymmetricProperty>表示对称属性（即 P（x,y）iff P（y,x））， 标签<owl:FunctionalProperty>表示功能属性（即 P（x,y）and P（x,z）=>y=z），标签<owl:inverseOf>表示逆属性（即 P1(x,y)iff P2(y,x)）和<owl:InverseFunctionalProperty>表示逆功能属性（即 P（y,x）and P（z,x）=>y=z）。当然还有很多定义标签。通过这些标签定义概念和概念间的关系，并进行推理，从而可以获取更多的知识。

可见，OWL 文件中包含了待抽取信息的所有属性描述，以及它们之间的关系，完全可以从 OWL 中提取出抽取规则。而 OWL 又是一种符合 XML 语法的语言，所以借助操作 XML 文件的类库就可以实现从 OWL 文件中归纳抽取规则了。

6.4 抽取 Web 信息

最后，根据第四章中介绍的抽取规则生成算法，从本体中归纳学习出了

抽取规则，然后应用这些抽取规则对样本页面之外的实验对象进行了信息抽取，提取出了结构化的信息，显示结果如图 6.3 所示。

信息抽取的性能是由查全率和准确率来衡量的。系统中得到的查全率和准确率都在 89%以上，实现了较高的效率。根据衡量信息抽取系统的性能指标，以及对“当当图书网”中 Web 页面的测试，得到表 6.2 所示的结果。

```
auther = Graham
auther = Donald
auther = Knuth
auther = Oren
publishing_company = 机械工业出版社
publishing_time = 出版时间：2002年08月
price = ￥49.00
price =
price = ￥36.75
price =   折扣：75折   节省：￥12.25
-------------------------------------
title = 新编微计算机原理解题指南
auther = 马争
publishing_company = 电子工业出版社
publishing_time = 出版时间：2005年08月
price = ￥28.00
price =
price = ￥21.00
price =   折扣：75折   节省：￥7.00
-------------------------------------
title = 单片机轻松入门（第2版）
auther = 周坚
publishing_company = 北京航天航空大学出版社
publishing_time = 出版时间：2007年02月
price = ￥28.00
price =
price = ￥21.00
price =   折扣：75折   节省：￥7.00
-------------------------------------
title = 形式语言与自动机理论（第2版）
auther = 蒋宗礼
auther = 姜守旭
publishing_company = 清华大学出版社
publishing_time = 出版时间：2007年07月
price = ￥29.00
price =
price = ￥21.80
price =   折扣：75折   节省：￥7.20
```

图 6.3 抽取结果显示

表 6.2 实验结果

抽取的数据	查全率（%）	查准率（%）
书名	91.2	90.8
作者	92.1	91.2
出版社	89.8	89.3
出版日期	94.2	93.7
原价	93.2	92.8
现价	90.4	90.1
打折率/省钱	90.3	90.1

6.5 实验结果及分析

在实际应用当中，用户很少有耐心选择多个页面（比如>5）进行信息抽取实验。信息抽取中的度量表转召回率和准确率作为对实验效果的评测。作者利用本文所给出的方法在小样本（≤5）的情况下对 HTML 文档提取的准确度进行了实验。实验结果如下：：

表 6.3 图书信息抽取实验结果

网站	样本长度	所有需要抽取的图书信息数	抽取出来的图书信息数	正确抽出结果的图书信息数	召回率（%）	正确率（%）
当当网	3	212	195	191	92.1	95
当当网	4	212	198	194	93.3	96
当当网	5	212	199	199	93.8	94
卓越网	3	232	209	205	90.2	96
卓越网	4	232	213	211	91.7	95
卓越网	5	232	216	214	93.2	93

召回率是指实际抽取出来的图书信息的数目占所有需要抽取的图书信息数目的比率。准确率是指抽取出来的正确的信息占抽取出来的所有图书信息的比率。

由表6.3可知，本文所提出的抽取系统能够很好地制定抽取规则并能够准确的抽取所需要的内容。在实验过程中，首先从当当和卓越图书网站中下载

几个图书销售页面作为样本页面，从样本页面中得到了大量的备选数据模式，然后根据数据模式抽取参数一步步筛选得到了有效数据模式，然后抽象出具体的概念描述，作为本体生成的基本要素。进而借助Protégé工具生成图书领域本体，其中在概念属性中设置了待抽取项的HTML路径信息。然后把该图书领域本体转化成对应的OWL文件，用OWL语言描述该领域本体所有信息。最后对OWL进行操作，归纳学习出抽取规则，作为抽取具体信息项的重要依据，指导抽取动作。从实验结果表中，可以看出虽然样本数量很小，但是两个网站信息的提取都得到了较高的查准率和召回率。

6.6 本章小结

本章以当当与卓越图书网上的 Web 页面为信息源，通过对其 Web 页面数据特征的分析，构造了一个小型的图书领域本体，实现了一个基于本体的 Web 页面结构化信息抽取系统，最后从本体中归纳学习出了抽取规则，对该站点中的 Web 页面进行信息抽取，实验表明，该系统实现了一定的召回率和准确率，论证了本课题研究的成果和价值。

结　论

随着人类进入信息社会，互联网获得了飞速的发展，Web 信息也随之急剧膨胀，然而在这种状况下，却一直缺乏有效的信息抽取机制。目前的信息抽取系统主要依靠手工方式产生抽取规则，抽取效率不高。并且大部分的 Web 信息抽取工具都是针对某一个特定的网站的网页进行包装器的编写，将 Web 访问信息、抽取模式、结果数据保存模式硬编码到程序中去。这样一旦目标网页的结构发生变化，就需要修改源程序代码，重新编译。由于 Web 网页的结构和内容是经常变动的，导致程序的维护和应用性很差。另外，在信息抽取系统中，最难解决的是所需数据信息的定位以及数据抽取中涉及到的语义问题，如多义词和同义词等。本文主要围绕以下几个方面展开工作。

（1）分析当前 Web 页面的结构特点，并对 HTML 与 XML 格式进行比较，总结转换原理与步骤，最后提出基于栈与链式结构的 HTML 到 XML 的转换算法。

（2）根据 Suffix Tree 的构造方法，构造样本页面的 Suffix Tree，并研究了最大公共子串提取方法，完成备选数据模式集的生成，最后根据最大重复子串提取算法，实现有效数据模式的提取。

（3）利用 Suffix Tree 解析出来的页面数据模式，得到具体的概念描述集合，并根据这些 Web 页面中的概念，然后再根据概念间的联系以及概念约束条件，借助术语语义聚类生成概念的方法，结合本体学习的技术和方法，进而构造了一个领域本体。

（4）借助已建立的领域本体，归纳学习得到抽取规则，以此作为抽取 Web 页面数据的依据。

（5）将抽取出来的数据存入数据库中，并结合本体库生成具有语义信息的 RDF 数据模型。

总之。本文在研究了 Web 信息抽取技术以及本体理论与 Suffix Tree 技术的基础上完成了根据知识密集型 Web 站点的结构和特征匹配来进行信息定位和信息抽取的方法。利用该系统可以从知识密集型 Web 站点中提取数据模式，抽取多个数据项。本体的引入，最大程度上解决了多义词等语义问题，

消除了语义异构现象，提高了抽取系统的性能和可移植性。

当然，由于时间和资源的局限，本课题还有很多方面可以进一步研究：首先，对本体进行进一步的扩充和完善，以适应网页数据复杂多变的特点，提高信息抽取的查全率和查准率。在本文的信息抽取方法中起着非常重要的作用，本体的完备性直接影响了信息抽取的效果。其次，本系统的算法实现中，样本信息文件数量可能不能达到标准的测试数量，所实现的算法虽然有较高的准确率，但是并未尝试大量的有价值的实验来验证其抽取的准确性和效率。第三，本文训练和测试的网页都是人工获得，如何自动周期性的自动获得网页也是需要考虑的问题。

总之，信息抽取经过二十多年尤其是最近十多年的发展，已经成为计算机领域中一个重要的分支。未来的信息抽取系统将是灵活和可升级的抽取规则归纳生成系统，以适应不断增长的动态网络的需要，同时，未来的信息抽取系统将会是动态的和开放的，具有较好的移植性和健壮性，能够胜任多领域的信息抽取工作。

参考文献

[1] Hammer J，Garcia-Molina H，Nestorov S，et al. Template-based wrapper in the TSIMMIS system (system demonstration)[A]. In：Proceedings of ACM SIGMOD Conference on Man-agent of Data，Tucson，Arizona，1997，532~535P.

[2] Hammer J，Garcia-Molina H，Cho J，et al. Extracting semistructured information from the Web [A]. In：Proceedings of Workshop on Management of Semi-Structured Data，Tucson，Arizona，1997，18~25 P.

[3] 李朝光，张铭，邓志鸿等. 论文元数据信息的自动抽取[J]. 计算机工程与应用，2002，38(21)：189~191 页.

[4] Kushmerick N， Weld D，et al. Induction for information extraction [A]. In：Proceedings of the 15th International Joint Conference on Artificial Intelligent， Nagoya，1997，2：729~737 P.

[5] Ashish N， Knoblock C. Wrapper generation for semi-structured internet sources[A]. In ： Proceedings of Workshop on Management of Semi-Structured Data，Tucson，Arizona，1997，10~17 P.

[6] Shiren Ye. Learning object model from product Web pages. IEEE Trans on Knowledge and Data Engineering，2006，18(3)：334-349 P.

[7] 周明建，高济，李飞. 基于本体论的 Web 信息抽取. 计算机辅助设计与图形学学报，2004，16(4)：535-541 页.

[8] H Ouahid，A Karmouch. An XML-based Web mining agent. MATA'99，Ottawa，1999，12-13 P.

[9] Chia-Hui Chang，Shao-Chen Lui，Yen-Chin Wu. Applying pattern mining to Web information extraction，The 5th Pacific-Asia Conference on Knowledge Discovery and Data Mining，Hong Kong，2001,13-15 P.

[10] 陈玉芳，葛隧和. 一个基于 XML 的 Web 数据收集模型的研究. 计算机工程与应用，2004(10),23-24 页.

[11] D. Buttler, L. Liu, and C. Pu "A Fully Automated Extraction System for the World Wide Web" Proc. IEEE Int'l Conf. Distributed Computing Systems (ICDCS), p. 361, 2001

[12] K. R. Muller et al, "An Introduction to Kernel-Based Learning Algorithms" IEEE Neural Networks, vol. 12, no. 2, pp. 181-201, 2001

[13] 李清山，陈平. 一种基于内容的 HTML 到 XML 的转化策略. 计算机工程与应用，2001(9)

[14] 李孝东，顾毓清. 基于 DOM 的 Web 信息提取. 计算机学报，2002(5)

[15] 李建波，李晓华，董树明，杨科华. 一种基于 XML 的 Web 信息抽取方法. 情报杂志，2006(8)

[16] 吴相智，刘卫国，费洪晓. 一种基于栈结构的 HTML 到 XML 的转换方法. 长沙交通学院学报，2004(6)

[17] Hirschberg D S. A Linear Space Algorithm for Computing Maximal Common Subsequences[J]. Communications of the ACM, 1975, 18(6): 341-343P

[18] Zhou Meil. i Some Concepts and Mathematical Consideration of Similarity System Theory[J]. Journal of System Science and System Engineering, 1992, 1(1): 84-92P

[19] Gaizauskas R, Wilks Y, Information Extraction: Beyond Document Retrieval. Journal of Documentation, 1997

[20] Sager N, Natural Language Information Processing, Reading, Massachusetts: Addison Wesley, 1981

[21] Dejong G, An Overview of the FRUMP System. In: LEHNERT, W., & RINGLE, M. h. (eds), Strategies for Natural Language Processing. Lawrence Erlbaum, 1982, 149-176P

[22] Weiner P. Linear Pattern Matching Algorithms[C]. In: Proceedings of the14th IEEE Annual Symposium on Switching and Automata Theory, 1973: 1-11P

[23] Mc Creight E.A Space-economical Suffix Tree Construction Algorithm[J]. Journal of the ACM, 1976, 23(2): 262-272P

[24] Gusfield D.Algorithms on Strings, Trees, and Sequence: Computer Science and Computational Biology[M]. New York : Cambridge University Press, 1997: 87-207P

[25] M. Uschold, Michael Gruninger. Ontology: Principles, Methods and Applications. Knowledge Engineering Review, Vol. 11 No. 2, June 1996

[26] Gruninger, M. and MS Fox. Methodology for the Design and Evaluation of Ontologies, Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, IJCAI-95, Montreal. 1995

[27] TR Gruber. Towards Principles for the Design of Ontologies Used for Knowledge Sharing[J]. International Journal of Human -Computer Studies, 1995(43). 907-928P

[28] Musleal , MintonS , KllolockC. Hierarehieal Wrapper induction for semi-structured information sources[J] . Autonomous Agents and Multi-agent Systems, 2001, 4(1/2): 93-114P

[29] MUSLEA I. MINION S, CRAIG A, et al. A hierarchical approach to wrapper induction[C]·The Third International Conference on Autonomous Agents, Washington, USA, 1999

[30] HSU C N , DUNG M. Generating finite-state transducers for semi-structured data extraction from the Web[J]. Information System, 1998, 23(8), 521-538P

[31] Kuslllneriek N. Wrapper induction : efficiency and expressiveness [J]. Artificial Intelligence Journal, 2000, 118(2): 15-68P

[32] AROCENA G, MENDELZON A. Web OQL: Restructuring documents, databases and webs[Z]. In Proceedings of the 14th ICDE Conference, Orlandom, Florida, USA, 1998

[33] GUTAVO AROCENA. Web OQL: Exploiting document structure in the Web queries[D] . Toronto: Masters's thesis. University of Toronto, 1997

[34] 徐林昊，杨文柱，陈少飞. 基于 XPath 的 Web 信息抽取[Z] . 19 届全国数据库会议，郑州，2002

[35] 杨文柱，徐林昊，郝亚南. 个性化的 Web 查询助手的设计与实现[Z] . 19

届全国数据库会议，郑州，2002

[36] EMBLEY D，CAMPBELL D，JIANG S，et al. Conceptual0 model-based data extraction from multiple record web pages[J] . Data and Knowledge Engineering，1999，31(3)：227-251P

[37] CHRISTINA YIP CHUNG，MICHAEL GERTZ，NEEL SUN DARESAN. Reverse engineering for Web data：From visual to semantic structures[Z] . In Proceedings of 18th international Conference on Data Engineering，San Jose，California，2002

[38] CHRISTINA YIP CHUNG，NEEL SUNDARESAN. Quixote：Building XML repositories from topic specific web documents[Z]. In Fourth Int. Workshop on the Web and Databases，2001

[39] ROBERT BAUMGARTNER，SERGIO FLESCA，GEORG GOTTLOB. Supervised wrapper generation with lixto[Z]. Proceedings of 27th International Conference on Very Large Databases，2001

[40] ROBERT BAUMGARTNER，SERGIO FLESCA，GEORG GOTTLOB. Visual Web information extraction wih lixto[Z]，Proceedings of the 27th VLDB Conference，Roma，Italy，2001

[41] LIU L，HAN W. XWRAP：An XML-enabled wrapper construction system for Web information sources[Z]. In Proceedings of the International Conference on Data Engineering，San Diego，2000

[42] LIU L，HAN W，BUTTLER D，et al. An XML-Based wrapper generator for Web information extraction[Z]. In Proceedings of ACM SIGMOD International Conference on Management of data，Philadaclphia，Pennsylvania USA，1999

[43] WALTER CRESCENZI，GIANSALVATORE MECCA. Road Runner：towards automatic data extraction from large Web sites[Z]. In Proceedings of th 27th International Conference on Very Large Database. Roma，Italy，2001

[44] 李保利，陈玉忠，俞士汉. 信息抽取研究综述. 计算机工程与应用，2003年，第10期

[45] 张文斌，陈恩红，王进. 一种基于多叉树的 HTML 到 XML 的转换方法. 小型微型计算机系统，2003，24(9)：1617-1620 页.

[46] 李效东，殷敏情. 基于 DOM 的 Web 信息提取. 计算机学报，2002，25(5)：526~533 页.

[47] 邓志鸿，唐世渭等. ontology 研究综述[J]. 北京大学学报(自然科学版). 2002. 38(5)：730-738 页.

[48] 廖乐健，曹大元，李新颖. 基于 ontology 的信息抽取. 计算机工程与应用，2002 年，第 23 期.

[49] 李芳，盛焕烨，姚天防. 信息检索与信息抽取技术的研究. 计算机应用研究，2002 年，第 1 期.

[50] Jennifer Widom，Data Management for XML，Research Directions，IEEE Data Engineering Bulletin，Special Issue on XML，1999，PZ-4.

[51] 徐惠英，刘月诗. 精彩 HTML 与 XML[M]. 台北：网奕资讯出版社.

[52] 放媛. web 信息抽取系统 SEU 设计与实现[0] . 江苏南京：东南大学，2006.

[53] M. Dean，D. Connolly，Evan Harmelen，J. Hendle. I. Horroeks，D. L. Mc Guinness，RF. Patel-Sehneider，and L. Andrea Stein. OWL Web ontology Language1. 0 Reference. Technical report World Wide Web Consortium. July2002.

[54] 王昕. 综述：本体的概念，方法和应用[Z] http：//www.prdm.net/papers/knowledge/Ontology_20overview.html.

[55] 陈少飞，郝亚南，李天柱. Web 信息抽取技术研究进展[J] . 河北大学学报(自然科学版)，2003，23(l)：106-112 页.

[56] 朱靖波，姚天顺. 中文信息自动抽取[J]. 东北大学学报(自然科学版)，1998，19(l)：52-54 页.

[57] 姜吉发. 自由文本的信息抽取模式获取的研究[D] . 北京：中国科学院，2004.

[58] 崔继馨，孔维平. web 信息抽取技术的研究[J]. 信息技术教育，2004，NO. 10：109-110 页.

[59] 张成洪，古晓洪，白延红. web 数据抽取技术研究进展[J]. 计算机科学，

2004，31(2)：129-131，151 页

[60] 王庆一，王继成，周源远，袁春风. 多信息块 Web 页面的信息抽取. 计算机应用研究，2002 年，第 10 期

[61] 阿捷. 将 HTML 转为 XML 的开源工具 HTML Tidy. [DB/OL]. [2006 -11 -28]. http：//www. w3cn. org/resource/down/2004/48.html

[62] Gusfield D. Algorithms on Strings，Trees，and Sequence：Computer Science and Computational Biology[M]. New York ： Cambridge University Press，1997：87-207P

[63] Zhou Meil. i Some Concepts and Mathematical Consideration of Similarity System Theory[J]. Journal of System Science and System Engineering，1992，1(1)：84-92P

[64] Hirschberg D S. A Linear Space Algorithm for Computing Maximal Common Subsequences[J]. Communications of the ACM，1975，18(6)：341-343P

[65] Waterson A， Preece A. Verifying Ontological Com-mitment in Knowledge-based Systems. Knowledge-Based Systems， 1999，12：45~54P

[66] 何新贵. 模糊信息处理技术. 北京：国防工业出版社，2002

[67] Dillon W R，Goldstein M. Multivariate Analysis：Methods and Applications. New York：John Wiley& Sons，1984

[68] S Weibel.Metadata:The Foundations of Resource Description[J].D-Lib Magazine,2005

攻读硕士学位期间发表的论文和取得的科研成果

科研项目

[1] 2007.09-2008.12 研究生信息综合管理系统

[2] 2007.09-2008.12 省基金：基于 HLA 的分布式虚拟场景仿真平台的研究与设计

致　谢

值此论文完成之际，我首先要感谢我已故的导师刘群教授，衷心地祝福他能够在天堂安息。刘群老师自身不断进取、仁爱和蔼、学士渊博，对我的关怀和悉心教导将永远留在学生的记忆里，他永远活在学生的心中。他从不停止对新领域探索工作的精神为学生树立了学习的标尺，学生将会以老师的希望与企盼为动力，认真求实，勇于上进，走好人生的每一步。

我要衷心地感谢我现在的导师王卓副教授，感谢王老师一年多来给予我的悉心指导和帮助。在论文研究过程中，得到了指导教师王卓副教授的细心指导。王老师广阔的视野、深刻的学术思想以及开明的科研作风对我产生了深远的影响。在王卓副教授的具体指导下，从论文题目的确定到各阶段的任务都顺利完成。在整个研究生学习阶段，王卓副教授严谨踏实的作风、精益求精的治学态度、孜孜以求的进取精神为我培养良好的学风和提高工作能力树立了学习的榜样。同时，王卓副教授勤奋坚韧的性格，开拓创新的思想，谦恭礼让的作风，良好的修养，无时无刻不影响着我，激励着我。从王卓副教授身上学到的，是我一生宝贵的财富。在此，谨向尊敬的导师王卓老师表示我最衷心的感谢和崇高的敬意！

在哈尔滨工程大学的学习和生活对我一生的影响都是深远的，也留下了许多美好的记忆。在这里我认识了许多学识渊博的老师，从他们那里获得了很多的教益和帮助。

另外，感谢实验室各位同学。他们端正的学习态度，积极的进取精神一直激励着我不断学习。他们给我提出了许多的宝贵意见和参考资料，和他们的交流总是使我的思路豁然开朗。

最后，感谢我的父亲母亲，我的家人，感谢他们一直以来给我在生活和学业上无私的关心、资助和支持，希望他们永远健康。