基于领域本体的半结构化文本知识自动获取方法的设计和实现

王海涛[1),2)] 曹存根[1)] 高 颖[3)]

[1)](中国科学院计算技术研究所智能信息处理重点实验室 北京 100080)

[2)](中国科学院研究生院 北京 100039)

[3)](中国科学院软件研究所 北京 100080)

摘 要 文章介绍了一种新的基于领域本体的文本知识自动获取方法的设计和实现.通过引入领域本体,实现了半结构化文本知识的完全自动获取.该方法具有较好的通用性,把人们从繁重的手工劳动中解放出来,并能极大地提高知识获取的效率.已经设计并实现了:基于领域本体的知识获取方法 OMKast;一种参数化的知识编程语言 ePKPL,用以实现 OMKast、ePKPL 虚拟机 VM(即 ePKPL 运行环境).目前,这项工作已经在中草药、音乐、西医等领域进行了应用,取得了满意的结果.

关键词 文本知识获取;知识编程语言;虚拟机;多主体系统;领域本体

中图法分类号 TP18

Design and Implementation of a System for Ontology-Mediated Knowledge Acquisition from Semi-Structured Text

WANG Hai-Tao[1),2)] CAO Cun-Gen[1)] GAO Ying[3)]

[1)](*Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing* 100080)

[2)](*Graduate School of the Chinese Academy of Sciences, Beijing* 100039)

[3)](*Institute of Software, Chinese Academy of Sciences, Beijing* 100080)

Abstract In the paper, a new ontology-mediated method for automatic knowledge acquisition from text is presented. And based on the method, a system has been implemented and tested in several applications. With domain-specific ontologies, knowledge acquisition from semi-structured text is automated completely. Not only it makes human beings releasing from tedious manual work, but also the efficiency of acquisition is improved remarkably. Additionally, it is a foundation for knowledge acquisition from free text. This paper presents the main framework and implementation details of the system. It includes a) a method of ontology-mediated knowledge acquisition from semi-structured text, b) an executable parametric knowledge programming language (ePKPL), and c) an ePKPL virtual machine. This system has been tested in three different domains, including Chinese herbs, music, and west medicine, and the results are satisfactory.

Keywords knowledge acquisition from text; knowledge programming language; virtual machine; multi-agent system; domain-specific ontologies

---

收稿日期:2004-03-17;**修改稿收到日期**:2005-10-24.**本课题得到自然科学基金**(60273019,60573063,60573064,60496326)**和国家"九七三"重点基础研究发展规划项目基金**(2003CB317008,2002DEA30036,G1999032701)**资助.****王海涛,男,1979 年生,博士研究生,主要研究方向包括自动故事生成、文本知识获取.** E-mail: htwang@ict.ac.cn.**曹存根,男,1964 年生,博士,研究员,主要研究领域为人工智能.高 颖,女,1977 年生,博士研究生,主要研究方向包括领域文本知识获取、领域本体形式分析、相关逻辑.**

1 引　言

知识是智能的基础.人类现有知识绝大部分是以文本为载体的.如何让计算机更好地从文本中自动获取知识,一直是知识工程领域需要解决的难题之一[1~4].

随着信息技术的广泛普及和应用,人们对知识服务的需求越来越强烈.知识服务离不开大型知识库的支持.仅仅依靠繁重的手工劳动来建设大型知识库,必将严重影响知识服务的能力和质量.因此,目前当务之急是寻求一种通用的自动的文本知识获取方法.

传统的文本知识获取方法主要有两种:一种是采用通用的算法处理自然语言文本,从文本中抽取概念以及概念之间的关系.这种方法知识获取量大,但所获取的知识类型较单一,知识的表示形式也相对简单.另一种方法是通过与知识工程师进行交互,使用一些知识获取平台或管理环境,实现知识的获取.这种方法不仅手工工作量大,而且通用性也较差[1~9].

鉴于现有文本知识获取方法的缺陷,本文提出一种"基于领域本体的半结构化文本知识自动获取方法"(Ontology-Mediated Knowledge acquisition from semi-structured text,OMKast).所谓半结构化文本(Semi-Structured Text,SST)是指一类特殊的自然语言文本.这类文本的结构和表达形式较为规范,同时文本表达内容又是自由的.专业手册、辞典等都是比较常见的半结构化文本.图 1 是一段有关中草药的半结构化文本例子.

```
*生姜
……
[性味归经]辛,微温.归肺、脾、胃经.
[配伍应用]
1.治劳嗽,常与萝卜汁、梨汁、蜂蜜、人乳等配伍.
2.治中恶昏倒、中风痰迷,多与陈皮、半夏、木香、甘草等药同用.
3.治妊娠呕吐,每与冰糖、红糖配伍.
4.治肺有寒邪、痰多咳嗽,当与杏仁、紫苏、陈皮等药合用.
……
*桑叶
……
[性味归经]甘、苦,寒.归肺、肝经.
[配伍应用]
1.治肺热咳嗽、恶心痰多、口渴咽干、大便干结,常与甘菊、生地、枇杷叶、陈皮、黄芩、枳壳、芦根等药配伍.
2.治外感风热、发热咳嗽,多与菊花、桔梗、薄荷等药同用.
3.治头目晕眩,每与枸杞子、决明子、菊花等药合用.
4.治肝阳眩晕、目赤头痛,当与白芍、石决明、菊花等药配伍.
……
```

图 1　中草药半结构化文本示例

OMKast 方法可大致描述如下:输入半结构化文本,根据预先定义的文法对输入文本进行语法分析,然后以领域本体为指导,采用适当的操作抽取需要的知识.其中,文法和获取行为的定义主要取决于文本的结构和表达形式以及知识获取需求.OMKast 主要由两部分组成:

(1)可执行的参数化知识编程语言(executable Parametric Knowledge Programming Language,ePKPL),这是一种为实现文本知识自动获取而设计开发的通用知识编程语言,用以定义文本文法结构和具体的知识获取行为;

(2)文本知识自动获取平台,即 ePKPL 虚拟机(Virtual Machine,VM),为 ePKPL 程序提供运行环境.

本文第 2 节介绍领域本体及其表示;第 3 节和第 4 节分别详述 OMKast 的两大组成部分:知识编程语言 ePKPL 及 ePKPL 虚拟机 VM;第 5 节是应用结果和性能评价;第 6 节总结全文.

2 领域本体及表示

领域本体(Domain-Specific Ontologies,DSO)通过定义类(categories)、实例(instances)、属性(attributes)、关系(relations)、公理(axioms)等元素[3],刻画领域中的类和实例及其之间的层次关系,对领域知识进行归纳和抽象.类指领域中具有相同属性的一类事物,组成某类的那些个别事物称为类的实例.领域本体(本文以下亦简称本体)在一定程度上是对现实世界的反映.有关领域本体建设的方法和问题在本文中不再赘述.

在 OMKast 知识获取过程之前，知识工程师根据文本内容及相关领域知识手工定义本体. 一个具体的本体中，类不仅能够规定实例可能具有的属性，而且可以对属性取值进行限制. 领域本体可以看作是知识获取的预备知识，是进一步获取其它知识的基础，其主要作用是规定获取内容，检查获取结果的有效性.

为便于操作，我们把预先定义的本体以及从文本中获取到的目标知识都用同一种知识描述语言——NKI 本体语言(NKIL)来表示. NKIL 是一种框架描述语言[3].

3 ePKPL 设计

3.1 ePKPL 设计的目的和特点

OMKast 方法虽然基于领域本体，但由于本体的多样性和复杂性，它又必须独立于本体. 所以，需要在 OMKast 和领域本体以及用户之间建立一种通用的机制，使 OMKast 能和不同领域本体结合，完成获取任务. 依据这种想法，我们设计了 ePKPL. 知识工程师只需根据知识获取需求编写一段 ePKPL 程序，就可以有针对性地实现对文本知识的快速自动获取. 下面我们给出 ePKPL 的形式定义.

定义 1. 可执行的参数知识编程语言 ePKPL=〈O,C,S,A〉，其中，

(1)O 是领域本体引用语言，在 ePKPL 程序中引入本体知识，指导主体中的操作；

(2)C 是常数定义语言，把 ePKPL 程序中可能经常用到的一些文本内容以常数形式定义出来；

(3)S 是文法及产生式集合定义语言，定义输入文本的结构和表达形式，用以解析文本；

(4)A 是主体定义语言，定义用来进行知识获取的主体，包括激活环境、指派用来解析文本的文法，并描述针对解析结果的知识获取行为.

ePKPL 和通常所说的编程语言有很大的不同，具有下列特点：

(1)ePKPL 程序的处理对象是半结构化文本，获取的知识用 NKIL 表示.

(2)ePKPL 程序是基于领域本体的. 在 NKI 知识库中，我们已经手工建立了很多领域本体. 这些本体负责指导 ePKPL 程序的执行过程[3].

(3)ePKPL 程序采用多主体的形式书写. ePKPL 程序中的每个主体定义了针对不同输入文本的不同操作. 采用多主体的思想，使每个主体功能专一，使用灵活，便于维护、重用和调度[10].

(4)ePKPL 程序对文本的处理结果正确性、粒度、执行效率等，取决于预先定义的文法和知识获取行为，与 ePKPL 语言本身无关.

3.2 ePKPL 程序的组成及语法

一个完整的 ePKPL 程序通常由以下几部分组成：本体文件引用、常数定义、文法定义、产生式集合定义、主体定义. 其中，文法定义和主体定义是最重要的部分.

3.2.1 领域本体引用

领域本体是 OMKast 知识获取的指导. 本体的引用必须在 ePKPL 程序的开头进行声明. 格式为：#include〈本体文件路径〉.

3.2.2 常数定义

由于自然语言的随意性，常常会出现很多词语或符号(比如标点符号等)，或者使用频率较高，或者意义相近，或者出现位置相似，或者难于表达. 为使 ePKPL 程序设计起来更加简单灵活，我们把上述类型的词语和符号定义为常数，用“defconstant”标识，格式如下：

〈常数定义〉::=defconstant〈常数集合名〉
‘{’
{〈常数定义〉}$^{+}$
‘}’

〈常数定义〉::=〈常数名〉:〈常数表达式序列〉

〈常数表达式序列〉::=〈出现形式〉{‘|’〈出现形式〉}.

3.2.3 文法定义和产生式集合定义

文法定义和产生式集合定义是 ePKPL 程序的重要组成部分之一，描述输入文本的结构和表达形式，是对文本进行语法分析的依据. 由于汉语自然语言文法纷繁复杂，规则较多，目前还没有一个形式化系统能对汉语文法进行描述. 但是半结构化文本的文法描述则相对容易. 对文本文法进行描述的目的在于帮助计算机“理解”自然语言书写的文本，定义的粒度完全由知识获取需求决定，没有必要对自然语言进行完全详尽的刻画. 这里，我们借用计算机科学中的形式文法[11]对自然语言进行描述.

ePKPL 程序中的文法描述的语言对象可以是词(组)、短语、句子、甚至段落、篇章等任意长度的文本. 一个 ePKPL 程序可以根据需要定义多个文法，分别描述不同结构形式的文本. 输入文本经语法分析后，会生成一棵或多棵语法树，所有后续知识获取行为都将围绕生成的语法树展开. 一旦输入文本匹配某个主体规定的文法，就表明文本中含有可获取

的知识.

文法定义和产生式集合定义格式如下：

〈文法定义〉::=defsyntax〈文法名〉[〈继承部分〉]
　　‘{’
　　〈产生式规则集合〉
　　‘}’
〈产生式集合定义〉::=defpattern〈产生式集合名〉
　　[〈继承部分〉]
　　‘{’
　　〈产生式规则集合〉
　　‘}’
〈继承部分〉::=:继承〈继承对象名〉{,和〈继承对象名〉}
〈继承对象名〉::=〈文法名〉|〈产生式集合名〉.

ePKPL 程序中每个 defsyntax 所定义的文法都必须是满足如下定义的上下文无关文法.

定义 2.　上下文无关文法 $G=\langle S,V_T,V_N,\mathcal{P}\rangle$：

(1)V_T是一个非空有限集. 它的元素是 ePKPL 程序中自定义的文法终结符(包括文本变量、关键字、常数),在语法树中作为叶节点出现(如图 2).

```
defsyntax〈性味归经〉
{
 〈性味归经〉::=〈#无空格字符串序列 1/〈 !顿逗号〉$〈 !空格〉〉.归〈#无空格字符串序列 2/、$〈 !空格〉〉经
}
defsyntax〈配伍应用〉
{
 〈配伍应用〉::=〈病情配伍药伍〉〈 !回车键〉|〈病情配伍药伍〉〈 !回车键〉〈配伍应用〉
 〈病情配伍药伍〉::=〈病情条件〉〈 !逗号〉〈配伍药伍〉
 〈病情条件〉::=〈#正整数$. $-).〈 !防治〉〈#无空格字符串序列 1/、/. $〈 !空格〉$〈 !等症〉$〈 !修饰语〉〉[〈 !等症〉]
 〈配伍药伍〉::=〈配伍药伍 1〉|〈配伍药伍 2〉
 〈配伍药伍 1〉::=〈 !修饰语〉〈#无空格字符串序列 2/〈 !顿逗号〉$〈 !空格〉$〈 !等药〉〉[〈 !等药〉]〈 !配伍词〉[〈注释文本〉]
 〈配伍药伍 2〉::=〈 !单用〉[〈注释文本〉]
 〈注释文本〉::=〈#无空格字符串$〈 !空格〉〉
}
```

文法开始符　常数　文法关键字　文法变量　文本变量

图 2　文法定义示例

(2)V_N是一个非空有限集,它的元素是文法变量,在语法树中是内部节点. $V_N \cap V_T=\varnothing$.

(3)$\mathcal{P}$是一个有限个产生式构成的集合,也即〈产生式规则集合〉. 每个产生式的形式为 $P::=\alpha$,其中,$P\in V_N$,$\alpha\in(V_N\cup V_T)^*$. 产生式以 BNF (Backus-Naur Form)方法表示,必须以右递归形式定义.

(4)$S\in V_N$是文法开始符,是在标识符“defsyntax”后指定的文法名称,是语法树的根,必须且只能在某个产生式的左边出现一次.

一个文法或产生式集合可以从其它文法或产生式集合中继承一些产生式规则. “继承”使得产生式可以重用,减少了很多重复定义的工作,也增加了 ePKPL 程序的灵活性. 当然,不允许循环继承出现.

```
defpattern〈通用模式〉
{
  〈括号模式〉::=〈 !左括号〉〈#字符串〉〈 !右括号〉
  〈序列〉::=〈#无空格字符串〉[〈_xl〉]
  〈疾病〉::=〈#无空格字符串〉
}

defsyntax〈泛称模式〉:继承〈通用模式〉
{
  〈泛称模式〉::=[〈疾病〉][〈括号模式〉][的]泛称〈_0001〉〈序列〉|〈序列〉〈_0001〉〈疾病〉[〈括号模式〉][的]泛称
  〈_0001〉::=是|为
}
```

图 3　产生式集合定义和继承示例

借助继承方式定义文法或产生式集合时,服从“局部优先”原则. 即:当从一个文法变量进一步扩展时,首先在当前的文法中找其对应的产生式. 如果找不到,再去继承的文法或产生式集合中查找. 此外,我们允许“多重多层混合继承”,即一个文法(产生式集合),可以从多个文法(产生式集合)继承——多重继承;每个被继承的文法(产生式集合)还可以继承其它的文法(产生式集合)——多层继承;同一文法(产生式集合)可同时继承其它的文法和产生式集合——混合继承.

文法和产生式集合虽然在定义格式方面极其相似,但含义却截然不同. 文法是一组产生式的有机结合,可直接用于对输入文本进行语法分析. 然而,产生式集合只是一些相互独立的产生式的集合,并不

能直接用来进行语法分析.因此导致二者在被继承时,也有本质的区别.如果是从文法继承,不仅要"借用"需要的产生式,而且这个被借用的产生式中新出现的那些非终结符所对应的产生式也要相应的借用下来,可以认为是借用语法的某段子树.但是,如果从产生式集合中继承,只"借用"需要的那一个产生式规则而已.也就是说,产生式集合中定义的产生式是松散结合在一起的,它们之间没有任何语义或逻辑上的关系,不能表达一个完整的语法含义;而文法中的产生式是相辅相成有机结合的,必须结合使用,并且可以表达完整的语法含义.

文法的定义和分析与应用领域是独立的,借助BNF的严谨表示形式,辅以灵活的继承机制,使得文法定义更加清晰灵活,具有更好的通用性和扩充性.

3.2.4 主体定义

ePKPL程序中,主体定义也是不可缺少的部分,所有文本分析和处理工作都是在主体内指定的.每个ePKPL程序中可以定义多个主体,每个主体分析处理符合某一文法的文本.多个主体协同工作,完成更为复杂的知识获取任务.

定义3. 主体 A=〈Env,Syn,Act〉,其中,

(1)Env是主体激活环境,它包括上标号和下标号;

(2)Syn是主体对文本进行分析所需要的文法知识;

(3)Act是主体行为,也称语义动作,描述主体所执行的知识获取行为.

主体的定义格式如下:

〈主体定义〉::=defagent〈主体名〉:主体本体
'{'
上标号:〈标号序列〉
下标号:〈标号序列〉
模式:〈模式名〉
{语义动作:[〈语义动作序列〉]}$^+$
'}'

〈标号序列〉::=〈标号常数〉{,或〈标号常数〉}

〈标号常数〉::=〈已定义常数〉|〈文本字符串〉

〈模式名〉::=〈文法名〉|〈常数名〉|〈文本变量名〉

〈语义动作序列〉::=〈条件语义动作〉|〈无条件语义动作〉

〈无条件语义动作〉::=〈系统语义动作和函数〉
{∧〈系统语义动作和函数〉}

〈条件语义动作〉::=〈条件表达式〉→〈无条件语义动作〉

〈条件表达式〉::=for(〈操作节点〉)(〈条件〉)|
forall(〈操作节点〉)(〈条件〉)

〈操作节点〉::=[〈祖先节点〉.]〈指定节点〉

〈条件〉::=〈和条件〉{∨〈条件〉}

〈和条件〉::=〈布尔值函数〉{∧〈和条件〉}.

主体执行机制:当输入文本满足某个主体的Env时,则该主体被激活;然后用Syn对文本进行语法分析.如果分析正确,对分析得到的语法树进行先根遍历,执行Act中的行为,获得需要的知识,并按需要的表示形式重新组织.

语义动作分为有条件语义动作和无条件语义动作两种.每个有条件语义动作的定义都是和生成语法树中的某个节点相对应的,而无条件语义动作是主体语法分析成功后必需执行的动作,与具体的分析结果无关.通常,语义动作都是根据知识获取需求以及文本的结构形式而给出的.语义动作执行方式和结果,依赖于语法分析的结果.因此一般来说,当文法改变时,语义动作也要相应地改动.语义动作或函数的参数可以是字符串,是文法中定义的终结符、非终结符,也可以嵌套调用其它的函数.附表1中,对函数及语义动作的形式及意义进行了简要说明.附图1是根据上文图1中的文本和图2中的文法给出的几个简单主体的例子.图4是ePKPL程序执行后获取的知识,并以NKIL表示.

```
defframe生姜:中药本体
{
        味性:辛
        气性:微温
        归经:肺经,和脾经,和胃经
        治疗:劳嗽
              :配伍萝卜汁,和梨汁,和蜂蜜,和人乳
        治疗:中恶昏倒,和中风痰迷
              :配伍陈皮,和半夏,和木香,和甘草
        治疗:妊娠呕吐
              :配伍冰糖,和红糖
        治疗:肺有寒邪,和痰多咳嗽
              :配伍杏仁,和紫苏,和陈皮
}
defframe桑叶:中药本体
{
        味性:甘,和苦
        气性:寒
        归经:肺经,和肝经
        治疗:肺热咳嗽,和恶心痰多,和口渴咽干,和大便干结
              :配伍甘菊,和生地,和枇杷叶,和陈皮,和黄芩,
               和枳壳,和芦根
        治疗:外感风热,和发热咳嗽
              :配伍菊花,和桔梗,和薄荷
        治疗:头目晕眩
              :配伍枸杞子,和决明子,和菊花
        治疗:肝阳眩晕,和目赤头痛
              :配伍白芍,和石决明,和菊花
}
```

图4 用NKIL描述获取的知识

4 ePKPL运行平台设计

4.1 虚拟机设计

ePKPL程序执行平台,我们称为"虚拟机"(Vir-

tual Machine，VM）. 虚拟机不仅是 ePKPL 程序的编辑、编译、调试和运行的平台，而且还集成了输入文本编辑，输出结果编辑，错误自动检查及提示等功能，并为知识工程师提供了一个友好的操作界面. 由于自然语言表达自由随意，同一文本可能会得到多个文法分析结果，出现二义性，因此 VM 还可以根据需要生成所有可能的结果.

在 VM 运行 ePKPL 程序前，需要对其进行编译和初始化，包括：

(1)ePKPL 程序的词法语法检查；

(2)引用本体初始化；

(3)构造常数表；

(4)初始化文法及产生式集合；

(5)初始化主体.

初始化成功后，VM 就可以读取输入文本，执行 ePKPL 程序. 基本过程(见图 5)如下：

1. 文本切分. 从文本中顺序截取一段文字，得到一个或多个被激活的主体. 选择一个主体转步 2.

2. 语法分析. 根据主体中指定的文法模式对截取的文本进行语法分析，如果成功，转步 3；否则选择另外一个可被激活的主体，转步 1；如果其它被激活主体都不可用，转步 4.

3. 执行语义动作. 先根据顺序遍历语法树，执行每个节点上定义的语义动作，如果成功，转步 4；如果失败，选择另外一个可被激活的主体，转步 2；如果其它被激活主体都不可用，转步 4.

4. 当前切分结果处理完毕. 如果文本全部处理完毕，程序结束退出；否则转步 1.

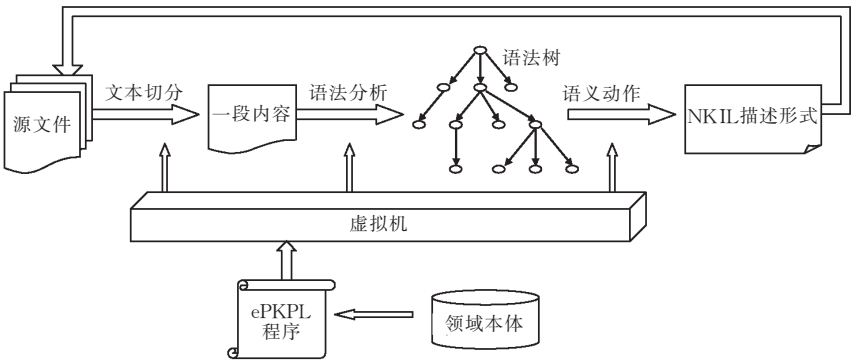

图 5　ePKPL 程序的执行

在 ePKPL 执行过程中，语法分析是最复杂的一个部分，我们采用自顶向下、自左向右、文法驱动的文法分析策略，也称为“递归下降分析法”. 生成语法树的过程是根据输入串不断进行最左推导的过程，该部分的详细算法如下：

1. 将文法开始符，记为 s_0，加入到待匹配文法链表 L 中，文法符号栈 Stack 置为空；

2. 对输入串 T 进行语法分析；如果 T 为空，并且 L 为空，转到步 4；否则，如果 T 不为空，并且 L 为空，栈 Stack 为空，语法树生成失败，错误退出；其它情况，转到步 2.1；

2.1. 从 L 中取出第一个文法符号 s，并将其从 L 中删除；

2.2. 如果 s 是引用的常数，如果 T 的开头和 s 所引用的某些常数项对应，选择其中最长的一项作为 s 和 T 匹配的常数，并将该常数项从 T 的开头删除，如果同时 s 是可选项，将 s 压栈，转到步 2；如果匹配成功，但 s 是必选项，转到步 2；如果匹配失败，转到步 3；

2.3. 如果 s 是文本变量，即形如〈#x〉的文法符号，从 L 中取出开头的文法符号 s'，并将其从 L 中删除；

2.3.1. 如果 s' 是引用的常数，从 T 中查找 s' 中常数第一次出现的位置，将该位置之前的一段字符串作为 s 的值，并将找到的常数作为 s' 的值，同时将该常数及其前面的字符串从 T 中删除，如果同时 s' 是可选项，将 s' 压栈，转到步 2；如果匹配成功，但 s' 是必选项，转到步 2；如果常数匹配失败，转到步 3；

2.3.2. 如果 s 是文法变量，即 s 曾在某个文法产生式的左端出现，从 s 所定义的产生式中选择一种进行扩展，并将新的产生式中的文法符号 $s_1, s_2, \cdots, s_n$ 加入 L 链头；转到步 2.3；

2.4. 如果 s 是文法变量，从 s 所定义的产生式中选择一种进行扩展，并将新的产生式中的文法符号 $s_1, s_2, \cdots, s_n$ 加入 L 链头；如果 s 有多种扩展形式，将 s 压栈，转到步 2；如果 s 只有一种扩展形式，转到步 2；

3. 如果 s 是可选项，置空，转到步 2；否则，如果栈为空，语法分析失败，错误退出；否则，取出栈顶结点，加到 L 链头，转到步 2；

4. 根据生成的语法树执行主体中定义的语义动作，生成 NKIL 知识表示形式.

4.2　VM 的辅助功能

由于自然语言表达灵活自由，自动处理很可能出现错误，因此，我们在 VM 中还增加了基于统计和规则的自动纠错机制. 利用词频信息以及一些知识检查规则，VM 可以自动对结果进行校验，找出可能的错误，并自动给出纠错提示，减少了知识获取和结果校对过程中的人工操作.

5 OMKast 应用及结果评价

目前,我们已经在 Windows 2000 操作系统下用 Visual C++实现了 VM(运行界面见图 6). 在很多领域,专业知识常常以手册、辞典等为载体,这些都可以认为是典型的半结构化的文本. 在应用过程中,我们遇到很多结构复杂的文本(如有关西医领域的文本),这种情况下,只是先对文本进行一些简单的手工修改和标记,使之结构形式清晰明确,然后即可应用 OMKast 方法处理. OMKast 方法在多个领域的应用结果如表 1 所示. 从表中不难看出,OMKast 方法完全适合建设大型知识库的要求. 此外,值得一提的是,OMKast 方法避免了手工处理所引入的错误,减少了结果校对的工作量及难度. 结果表明,OMKast 方法是令人满意的.

图 6 VM 界面示意图

表 1 结果统计

应用领域	文本大小(KByte)	主体个数	运行时间(min)	准确率(%)	运行环境
中草药	约 700	20～30	<1	>97	CPU: PIII 866
音乐	约 1200	30～40	<2	>97	Memory:128M
西医	约 2000	40～50	<5	>85	OS: Windows 2000 Professional

6 结 论

为了克服传统的文本知识获取方法在效率、获取内容、知识表示和通用性等方面存在的不足,我们设计了基于领域本体的半结构化文本知识自动获取方法 OMKast,从而有效地提高了知识获取效率,获取的知识类型更加丰富. 通过在中草药、音乐、西医三个领域的应用表明,OMKast 方法具有良好的通用性和可扩充性,在知识表示方面也更加灵活,生成结果不仅可读性强,而且也便于机器进一步处理. 此外,经过改进,我们成功地将 OMKast 方法应用于考古领域自然语言自由文本的语法语义标注以及一些信息服务业务中,达到了预想的效果. 由于在很多领域,专业知识常常以专业手册、辞书、辞典等为载体,因此,OMKast 方法对于快速获取领域知识、建设大型知识库有着巨大的应用潜力.

参 考 文 献

1 Reddy R.. Three open problems in AI. Journal of the ACM, 2003, 50(1): 83～86

2 Feigenbaum E.. Some challenges and grand challenges for computational intelligence. Journal of the ACM, 2003, 50(1): 32～40

3 Cao C. *et al*.. Progress in the development of national knowledge infrastructure. Journal of Computer Science & Technology, 2002, 17(5): 523～534

4 Lu Ru-Qian. Knowledge Engineering and Knowledge Science at

the Turn of the Century. Beijing: Tsinghua University Press, 2001(in Chinese)
(陆汝钤. 世纪之交的知识工程与知识科学. 北京:清华大学出版社,2001)

5 Barnett J., Knight K., Mani I., Rich E.. Knowledge and natural language processing. Communications of the ACM, 1990, 33(8): 50～71

6 Tan A.. Text Mining: The state of the art and the challenges. In: Proceeding of the Pacific Asia Conference on Knowledge Discovery and Data Mining PAKDD'99 Workshop on Knowledge Discovery from Advanced Databases, Beijing, China, 1999, 65～70

7 Cao C., Wang H., Sui Y.. Knowledge modeling and acquisition of traditional Chinese herbal drugs and formulas. International Journal of Artificial Intelligence in Medicine, 2004, 32(1): 3～13

8 Richardson S., Dolan W., Vanderwende L.. MindNet: Acquiring and structuring semantic information from text. In: Proceedings of the 36th Annum Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics. ACL, Montreal, Quebec, Canada, 1998, 2: 1098～1102

9 Hahn U., Klenner M., Schnattinger K.. Learning from texts: A terminological metareasoning perspective. In: Wermter S., Riloff E., Scheler G. eds.. Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing. Berlin: Springer, 1996, 453～468

10 Shi Zhong-Zhi. Intelligent Agents and Applications. Beijing: Science Press, 2001(in Chinese)
(史忠植. 智能主体及其应用. 北京:科学出版社, 2001)

11 Chen Huo-Wang, Qian Jia-Hua, Sun Yong-Qiang. Principle of Compiling of Programming Language. 2nd Edition. Beijing: National Defence Industry Press, 1999(in Chinese)
(陈火旺,钱家骅,孙永强. 程序设计语言编译原理. 第二版. 北京:国防工业出版社, 1999)

附表 1　系统定义函数及语义动作.

类型	说明	函数及语义动作定义
布尔值函数	判断两个参数的相等或包含关系， 如果为真返回 TRUE， 否则返回 FALSE	streq(p_1, p_2):判断参数是否相等 strneq(p_1, p_2):判断参数是否不等 contain(p_1, p_2):p_1是否包含 p_2 not-contain(p_1, p_2):p_1是否不包含 p_2 begin-with(p_1, p_2):p_1是否以 p_2开头 not-begin-with(p_1, p_2):p_1是否不以 p_2开头 end-with(p_1, p_2):p_1是否以 p_2结尾 not-end-with(p_1, p_2):p_1是否不以 p_2结尾
字符串值函数	返回值为字符串	fetch(p):取变量 p 的值 strcat(p_1{, p_2}$^+$):字符串连接 suffix(〈字符串序列〉,〈后缀〉):在序列每一项的末尾都加上后缀 prefix(〈字符串序列〉,〈前缀〉):在序列每一项的开头都加上前缀
无值函数	不返回任何值	save(〈标示符〉,〈值〉):把〈值〉以〈标识符〉命名,并保存起来
语义动作	创建、关闭动作 框架是类的实例,关闭后的框架不允许任何语义动作对其进行任何操作	create-frame(〈框架名〉,〈类名〉):创建指定类的一个实例框架 close-frame(〈框架名〉):关闭该实例框架,使其不再可以操作 close-category(〈类名〉):关闭指定类的所有实例框架 close-all():关闭所有类的所有实例框架 closelastframe(〈类名〉):关闭指定类最后生成的那个实例框架 closefirstframe(〈类名〉):关闭指定类最早生成的那个实例框架
	槽插入动作 〈位置〉::=first|last|all first:插入到指定类的第一个打开的实例框架中. last:插入到指定类的最后一个打开的实例框架中.是位置参数缺省时的默认值. all:表示插入到指定类的所有打开的实例框架中.	insert-category([〈位置〉,]〈类名〉,〈槽定义〉):在指定类指定位置的实例框架中,插入一个槽定义 insert-frame(〈实例名〉,〈类名〉,〈槽定义〉):在指定类的指定实例框架中,插入一个槽定义 insert-def-category([〈位置〉,]〈槽定义〉):在默认类的指定位置的实例框架中插入一个槽定义 insert-def-frame(〈槽定义〉):在默认类的默认实例框架中插入一个槽定义 sel-insert-category([〈位置〉,]〈类名〉,〈槽定义〉):从槽定义中选择符合要求的值,插入指定的实例框架中(下面三个动作的意义可由以上推得) sel-insert-def-frame(〈槽定义〉) sel-insert-frame(〈实例名〉,〈类名〉,〈槽定义〉) sel-insert-def-category([〈位置〉,]〈槽定义〉) 其中, 〈槽定义〉::=〈槽名〉,〈槽值〉[,〈分隔符〉]{;〈侧面名〉,〈侧面值〉[,〈分隔符〉]}

```
defagent 中药框架主体:主体本体
{
  上标号:"*"
  下标号:〈 !标号〉
  模式:〈#无空格字符串 $〈 !空格〉〉
  语义动作:close-category("中药本体")
  语义动作:for(〈#无空格字符串 $〈 !空格〉〉)→create-frame(〈#无空格字符串 $〈 !空格〉〉,"中药本体")
}
defagent 性味归经属性主体:主体本体
{
  上标号:"[性味归经]"
  下标号:〈 !标号〉
  模式:〈性味归经〉
  语义动作:for(〈#无空格字符串序列 1/〈 !顿逗号〉$〈 !空格〉〉)→
            sel-insert-category("中药本体","味性",〈#无空格字符串序列 1/〈 !顿逗号〉$〈 !空格〉〉,",和")∧
            sel-insert-category("中药本体","气性",〈#无空格字符串序列 1/〈 !顿逗号〉$〈 !空格〉〉,",和")∧
            sel-insert-category("中药本体","毒性",〈#无空格字符串序列 1/〈 !顿逗号〉$〈 !空格〉〉,",和")
  语义动作:for(〈#无空格字符串序列 2/、$〈 !空格〉〉)→
            insert-category("中药本体","归经",suffix(〈#无空格字符串序列 2/、$〈 !空格〉〉,"经"),",和")
}
defagent 配伍应用主体:主体本体
{
  上标号:"[配伍应用]"
  下标号:〈 !标号〉
  模式:〈配伍应用〉
  语义动作:forall(〈病情配伍药伍〉)→insert-category("中药本体","治疗",〈#无空格字符串序列 1/、/. $〈 !空格〉
            $〈 !等症〉$〈 !修饰语〉〉,",和";"配伍",〈#无空格字符串序列 2/〈 !顿逗号〉$〈 !空格〉$〈 !等药〉〉,
            ",和";"注释",〈注释文本〉)
}
```

附图 1 主体定义示例

WANG Hai-Tao, born in 1979, Ph. D. candidate. His research interests include automatic story generation and knowledge acquisition from text.

CAO Cun-Gen, born in 1964, Ph. D., professor. His main research interests focus on artificial intelligence.

GAO Ying, born in 1977, Ph. D. candidate. Her research interests include domain knowledge acquisition from text, formal analysis of domain-specific ontologies, and relevant logics.

Background

One of the central problems in the Natural Science Foundation (grant Nos. 60273019, 60496326, 60573063, 60573064), and the National Basic Research Program of China (973 Program) (grants No. 2003CB317008 and G1999032701) is to automatically acquire domain knowledge from massive knowledge sources. OMKast introduced in the paper is an efficient tool for such acquiring task. The experimental results have shown that the system can not only be used for extracting knowledge from textbooks, but also used in civil information services.