

基于动词属性的模板化自动代码生成

汪 畅,王 铮,张胜歧
(重庆大学 计算机学院,重庆 400044)

摘 要:介绍了一种自动代码生成的方法。提出了以动词为中心,基于属性的语义处理方法理论。在此思想理论的指导下,建立了相应的知识库和语义处理规则库,并详细研究了受限自然语言语句中词语的语义处理过程。最后将受限自然语言理解应用自动代码生成中去,通过对已经规范化的受限汉语语句中的各个动词进行分类并赋予其属性概念,依据知识库和规则库,对受限语句进行语义分析,将之转换为中间语言,并结合可定制的模板方法,在程序生成引擎中自动生成代码。

关键词:自动生成;动词属性;语义分析;中间语言;模板

中图分类号:TP311

文献标识码:A

文章编号:1673-629X(2010)05-0104-04

Template Automatic Code Generation Based on Properties of the Verb

WANG Chang, WANG Zheng, ZHANG Sheng-qi

(Department of Computer Science, Chongqing University, Chongqing 400044, China)

Abstract: Describe an automatic code generation approach. Proposed to the verb as the center, property-based theory of the semantic approach. In this ideological and theoretical guidance, establish the knowledge base and the corresponding processing rules and semantics of the library, and a detailed study of the limited terms of natural language statement, the semantics of the process. Finally, the limited natural language understanding application of automatic code generation. Through the restricted Chinese statements have been standardized in the various verbs and give their properties to classify the concept, based on knowledge base and rule base, the statement is limited to semantic analysis, will be converted into the intermediate language, combined with customizable templates method, and then automatically generated code in the code generation engine.

Key words: automatic generation; verbs property; semantic analysis; intermediate language; template

0 引言

传统软件开发方式通常效率较低而且开发周期较长,并且存在大量的重复劳动。随着软件工程方法的发展,许多的软件开发者在思索,如何弥补传统开发方式的不足,能够提高开发效率、缩短开发周期、降低开发成本以及减小需求变更对系统的影响。自动代码生成技术在一定程度上解决了这个问题。通用的代码生成方法可以带来软件开发效率的提高,并进而改进软件开发的过程,具有普遍的指导意义。

近年来,在自动代码生成技术研究和实现方面,国外相继提出了包括基于元数据驱动、XML、MDA、Struts等方案^[1]。这些方案对自动代码生成技术的发展具有积极的影响。然而在目前已经实现了的各种自

动代码生成技术都还存在各方面的不足,还在不断发展中。

文中设计了一种自动代码生成系统,利用语义分析器生成中间语言文件并结合模板文件和模板引擎技术,实现了自动程序设计的功能,并验证了这种解决方案的可行性。并进一步展望这个系统的发展方向及自动代码生成技术的发展方向。

1 系统模型分析

自动代码生成的过程就是理解自然语言并将其转换为计算机语言的过程^[2]。在这个过程中,为了实现理解自然语言的功能,将对自然语言进行语法、句法和语义分析,并将之转换为中间语言。然后结合可定制的预置模板来生成计算机可识别的程序设计语言。

在文中的代码生成实践中,建立了一个基于受限汉语语句的代码生成系统,简称 LCGS。其工作流程

收稿日期:2009-09-03;修回日期:2009-12-14

作者简介:汪 畅(1983-),男,硕士生,研究方向为软件自动化等;
王 铮,副教授,主要研究方向为软件自动化、嵌入式等。

如图1所示。

LCGS将对一系列的输入的文件,依据规则库中的规则进行语义分析处理并将之转换为中间语言文件。再由代码生成引擎根据中间语言调用模板库中的模板,生成一个或者是多个的包含所需生成的代码的输出文件。在这之中:

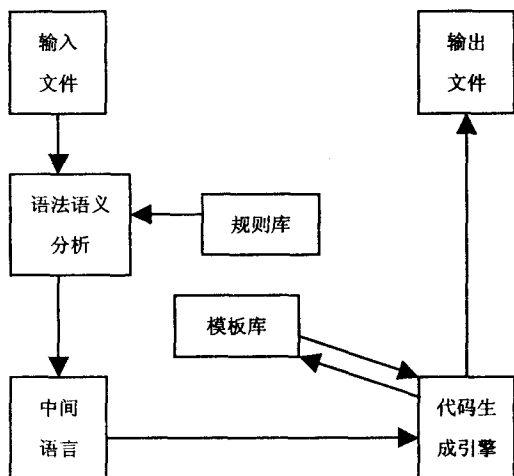


图1 LCGS 工作流程图

输入文件,是已经规范化并且经过分词处理的受限汉语语句。

输出文件,是产生的计算机可识别的源代码文件。

规则库文件,是预置的语法、语义规则。

模板库文件,是用户自己定制的各种预置语义的模板,是可重用的部分。

2 系统实现的关键部分

2.1 语义分析及中间语言生成的实现

2.1.1 动词通用模型

文中是以输入文件中的动词作为中心来分析各项之间的关系,因此动词的属性解释是系统实现的最重要的部分之一。

在汉语中,同一动词在不同的语义环境中有不同的意义,在这种情况下,对其建立通用语义模型是很困难的^[3]。文中的研究是针对特定领域的受限汉语的,因此,可以为短语动词建立一个在特定领域中的通用语义模型,并在实现时根据具体语义来实例化,以满足需求。其通用模型为:

$GenericMode(verb) = \{ semantic \mid semantic \text{ 为 } verb \text{ 在这个领域的通用语义模型} \}$

2.1.2 受限汉语动词分类

动词分类是指动词按组合特点类聚的分析。对于动词分类的问题研究,在现代汉语语法研究中占有重要地位。回顾现代汉语语法研究的历程,发现有关动词分类的研究,确定其标准,主要从这几个方面进行

的^[4]:

(1)根据动词所表达的意义来进行分类。

(2)从动词语法功能上进行划分。

(3)根据动词所必须联系的名词性成分的数目进行分类。

在该系统中,输入文件中的短语动词使用词义分类的方法可以分为以下几类:

(1)判断动词,表示对事物的判断或肯定;

(2)有无动词,表示事物的有无;

(3)比拟动词,表示事物之间的比拟关系;

(4)行止动词,表示动作开始、进行或停止;

(5)行为他动词,表示以某种事物为对象的动作行为。

2.1.3 属性规则设计

动词属性是指动词发生时伴随其产生的客观事实^[5]。对动词属性的研究有助于我们更深刻地挖掘动词本身具有的特性,对于自然语言理解有着积极的意义。

动词属性表现为名词。动词进行语义分析时,必须依据规则判断名词是否是作为动词的属性,并确定名词与动词的属性关系,从而可以通过这种方式确定短语描述的事件之间的关系^[6]。

在LCGS中,把对短语进行分词之后得到的最小语义单位称为单位元,并为单位元设计了属性规则,主要规则如下:

(1)词类规则: < 词类, 属性 >

(2)单位元分类规则: < 单位元, 词类 >

(3)单位元模板规则: < 单位元, 模板名 >

2.1.4 语义分析

在LCGS中,语义分析过程是在单位元的基础上进行的。一个单位元就代表一个分类,这个分类名字就是单位元具体存在的表示。

语义分析算法的思想为^[7]:将输入文件中的分词进一步转化为单位元,使原来用户输入的语句转化为一些单位元的组合,然后利用单位元的属性规则将拆开的单位元重新组合起来,同时生成语法树,并根据语法树继续进行分析,生成中间代码。

单位元重构算法的具体流程如下:

(1)将输入文件中的分词转化为单位元,将其存放在预先定义的单位元数组中。

(2)将输入的单位元遍历整个规则库中的规则,看其是否有规则,并进行匹配,如有则转(3),否则转(5)。

(3)将匹配规则的前提替换使用该规则的单位元,再遍历所有规则直到无规则匹配,将最后一次匹配规则前提压入分析栈中。

(4)如果栈为空,则将分析栈中弹出的单位元重新压入分析栈中,转(5);否则增加将分析栈中的弹出的新单位元与原单位元组合为一个新单位元,遍历整个规则典中的规则,看是否有规则匹配,如有则转(3),否则转(4)。

(5)如单位元数组指针指向最后一个元素,则转(6);否则输入单位元数组的下一个单位元,转(2)。

(6)如分析栈中只有一个单位元,则表示单位元重构成成功;否则表示不符合语法规则。

2.1.5 中间语言设计

中间语言的存在是为了使计算机更好地将汉语转化为计算机语言,因此其应该遵循几个原则:

(1)正确性:中间语言能够准确地反应原来的意思,同时也要能方便地转化为计算机语言。

(2)完整性:中间语言能够实现在特定的领域中描述相关的事件。

(3)可扩展性:为了满足未来可能变动的需要,要求中间语言应该可以方便地扩展其所需功能。

在 PCGS 中采用的中间语言是一个集合,记录了代码生成所需的各种信息,其结构如下所示:

IL 信息: < 组序号, 动词单位元, 动词熟悉单位元, 模板名, 参数个数, 返回值类型, 参数类型, 附加信息 >

2.2 模板设计

模板是自动代码生成中的一个重要组成部分,是包含了各种预定义的文件,可以由用户定制^[8]。文中使用的模板库是依据动词模型而建立的包含动词和名词的预置语义模板。

模板文件分为两部分:

固定模板:固定模板描述了内容不变的具体模板定义,在模板参与到代码生成引擎的整合时,只是简单的将其固定的内容整合到目标文件中。该模板的定义是经过严格测试的源代码,是可以反复使用的公共源代码。固定模板主要实现了动词的语义预置,如各种预置的排序算法。

可变模板:可变模板描述模板的抽象定义,是对通用动词模型中的各个动词的抽象的定义,在代码生成引擎整合代码时需要依据中间文件的数据来完成抽象定义到源代码的动态裁剪。

文中使用“< \$ >”标记来标记模板:

格式: < \$ String >. < \$... >之中直接跟字符串,实现模板类型、参数、名字的定义和替换。

模板文件的定义:

/*

* 方法注释

```
* /
< $ Type > < $ Compare > (< $ Type para0,
$ Type para1, ... >)
{
//要做的事的实现
}
```

在模板文件的定义中,“< \$ >”中的都是可变量定义,是可由用户定制的内容。模板的定义在经过代码生成引擎后,将被处理成目标源代码,然后将输出结果代码。

2.3 代码生成引擎的实现

文中 LCGS 将实现 C 语言的代码生成。LCGS 的代码生成引擎根据对中间代码的分析结果进行模板匹配,生成的可调用模板来整合为完整的 C 程序代码,然后把生成的 C 程序代码写入到输出文件。

在模板匹配中生成的模板信息包含在 Template 结构数组中,Template 结构的描述如下:

```
Struct Template {
char * templateName: //模板名;
int paraNumber: //模板参数的个数;
char * paraType: //模板参数类型;
char * returnType: //模板的返回类型;
}
```

通过 Template 数组,代码生成引擎知道需要调用模板库中的那些模板,以及如何实现模板的具体化,以此来完成不同功能。

代码生成引擎生成算法的流程如下:

(1)调用模板库中的基础模板(即包含了 main 函数和标准头文件的模板),生成 C 程序的标准头文件和空的 main 函数。将模板代码直接写入到输出的代码文本文件中。

(2)从 Template 数组取一个元素,据此在模板库中查找出所需的模板,将包含模板的头文件插入到主程序的头文件之前;将其定义代码根据参数的个数和类型进行修剪;然后在 main 函数内的开始处根据参数的个数和类型及模板返回值类型定义变量,以便作为调用模块的参数传入和返回。

(3)重复步骤(2),直至 Template 结构数组中所有元素均已访问。

(4)根据对中间代码的分析结果提取出控制语句及对应的模板,在 main 函数中插入相应的控制语句和对应的模板,并且给模板传入相应的实参。

例如:输入“比较/两个/整数/的/大小/”经过语义分析,代码生成引擎的处理后,得到的 Template 结构数组中包含有一个这样的元素:

```
{“Compare”,2,“int,int”,“int”}
```

这个元素说明 main 函数需要调用一个 Compare 模板,它需要传递两个整型参数。这个程序的生成过程如下:

(1)在输出文件中插入基础模板。

(2)在 main 函数前插入经过修剪的 Compare 模板的定义;在 main 函数开始处定义两个整型参数变量及一个返回整型变量。

(3)在 main 函数中依据控制语句顺序插入这两个变量的输入模板、Compare 模板调用及输出模板,并将定义的两个整型量作为参数传递给 Compare 模板。

3 系统测试结果

输入

有其他的行^[4]

输出

Main. c:

```
#include“GetLine. h”
```

```
#include <stdio. h>
```

```
void main()
```

```
{
```

```
int len;
```

```
char line[MAXLINE];
```

```
len = GetLine(line, MAXLINE)
```

```
}
```

GetLine. h:

```
/*
```

```
* GetLine: read a line into s, return length */
```

```
#define MAXLINE 1000
```

```
int GetLine(char para0[],int para1)
```

```
{
```

```
int c, i;
```

```
for (i=0; i < para1-1 && (c=getchar())!= EOF && c!= '\n'; ++i)
```

```
para0[i] = c;
```

```
if (c == '\n')
```

```
{
```

```
para0[i] = c;
```

```
++i;
```

```
}
```

```
para0[i] = '\0';
```

```
return i;
```

```
}
```

目前测试的模块模板库规模比较小,在有效的测试用例中,绝大部分比较规范的输入语句都能得到和语句意思相符合的程序。

4 结束语

文中实现了自动代码的生成系统,但仍然存在很大的可改进之处:

一是目前只能使用模板库中已有模板来生成程序,需要添加大量的模板来完善;同时当前的模板代码是固定的,如果语义解释的内容有变动,模板代码也需要进行大量的改变。如果能实现通过代码生成模板,这将大大减少模板的开发时间。

二是 LCGS 应用还只是针对一些简单的模型,不够完善,可以扩大代码生成范围,比如可以生成整个应用程序框架,实现真正意义上模型自动向代码转换,达到自动部署和运行的目的。

参考文献:

- [1] Burginsky F, Finnie M, Yu P. Automatic Code Generation from Design Patterns[J]. IBM Systems Journal, 1996, 35 (2):151-171.
- [2] 李家治,陈永明. 机器理解汉语——实验 I[J]. 心理学报, 1982(1):32-43.
- [3] 马庆株. 汉语语义语法范畴问题[M]. 北京:北京语言文化大学出版社,1998.
- [4] 陈祖荣. 浅谈动词的分类问题[J]. 四川师范学院学报:哲学社会科学版,1995(1):56-58.
- [5] 马庆株. 汉语动词和动词性结构(一编)[M]. 北京:北京大学出版社,2005.
- [6] 钱 进,王希杰. 词义性别原型与构词模式[J]. 江西社会科学,2004(9):149-153.
- [7] Sakurai T, Shibagaki T, Shinbori T. An Automatic Programming System Based on Modular Integrated - concept Architecture[C]//Proc. of IECON'90. California, USA: Industrial Electronics Society, 1990:1303-1308.
- [8] Kernighan B W, Ritchie D M. The C Programming Language [M]. [s.l.]:Prentice - Hall, 1988.

(上接第 103 页)

京:机械工业出版社,2005.

- [8] Shepherd, G. Visual C++ .Net 技术内幕[M]. 第 6 版. 北京:清华大学出版社,2004.

- [9] 张信一,李代平,罗伟刚. 并行程序开发平台的可视化实现[J]. 计算机应用研究,2004(11):266-269.