SRLtoRadl 生成系统及其范畴论语义

王昌晶[1,2,3,4],薛锦云[1,2],左正康[1,2,3,4]
(1.江西师范大学省高性能计算技术重点实验室,江西南昌 330022;2.中国科学院软件研究所计算机科学国家重点实验室,北京 100190;
3.中国科学院研究生院,北京 100190;4.江西师范大学计算机信息工程学院,江西南昌 330022)

摘 要: 形式化软件规约技术是保证软件质量和提高软件生产率非常有用和重要的手段,但是形式化软件规约的获取是需求工程中一项相当困难的任务.本文针对问题需求自动化转换为形式化规约这个重要问题,研究从结构化需求语言 SRL 到形式化规约语言 Radl 自动生成系统及其高可靠性理论.为此,设计了一种受控自然语言-结构化需求语言 SRL 来描述问题需求;使用基于规则的方法,将结构化需求语言 SRL 通过分析-转换-综合三阶段生成为形式化软件规约 Radl;在该方法的指导下,设计并实现了从结构化需求语言 SRL 到形式化软件规约 Radl 的生成系统 SRLtoRadl;进一步,使用范畴论框架建立了 SRLtoRadl 生成系统生成过程的语义模型.实际效果表明该系统能有效的生成高质量形式化软件规约 Radl.

关键词: 结构化需求语言; 形式化软件规约; 自动生成系统; 高可靠; 范畴论语义
中图分类号: TP311 **文献标识码:** A **文章编号:** 0372-2112 (2014)01-0137-07
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2014.01.022

SRLtoRadl Generation System and Its Category Semantics

WANG Chang-jing[1,2,3,4], XUE Jin-yun[1,2], ZUO Zheng-kang[1,2,3,4]
(1. *Key Laboratory for High-Performance Computing Technology*, *Jiangxi Normal University*, *Nanchang*, *Jiangxi* 330022, *China*;
2. *National Key Laboratory for Computer Science*, *Institute of Software*, *Chinese Academy of Sciences*, *Beijing* 100190, *China*;
3. *Graduate University of Chinese Academy of Sciences*, *Beijing* 100190, *China*;
4. *College of Computer Information and Engineering*, *Jiangxi Normal University*, *Nanchang*, *Jiangxi* 330022, *China*)

Abstract: Formal specification techniques are very useful and important for improving quality and productivity of software. But acquisition of formal software specification is a quite difficult task in requirement engineering. The research objective aims to the important problem about automating conversion from problem requirement into a formal specification. This paper studies the automatic generation system from structural requirement language (SRL) into formal specification-Radl and its high reliability theory. We design a controlled natural language-SRL to describe the problem requirement. Using rule-based method, convert SRL into Radl by analysis-transformation-synthesis three stages. Under the guidance of the rule-based generation method, we design and implement the generation system SRLtoRadl from SRL to Radl. Furthermore, establish the generation process semantic model of SRLtoRadl using category theory framework. Practical effects manifest it can effectively generate formal software specification of high quality.

Key words: structural requirement language; formal software specification; automatic generation system; high reliability; category semantics

1 引言

形式化软件规约技术便于软件系统原型、分析、验证与最终的实现,是保证软件质量和提高软件生产率非常有用和重要的手段,尤其是在开发高可信软件系统场合.形式化软件规约使用数学表示法来精确地描述软件系统,使用形式化规约带来潜在的利益包括更高的生产率、早期错误移除、自动代码生成等.

尽管研究者在软件综合、验证与测试等方面取得了较大的进展,但是由于要求具备良好的数学基础和抽象

收稿日期:2011-05-16;修回日期:2012-10-25;责任编辑:孙瑶
基金项目:国家自然科学基金重大国际(地区)合作与交流项目(No.61020106009);国家自然科学基金(No.61272075,No.61363012);江西省自然科学青年科学基金(No.20122BAB211030);江西省教育厅一般科学基金(No.GJJ13225)

思维能力,形式化软件规约的获取仍然是一项相当困难的任务,在形式化软件规约获取方面的研究进展缓慢[1].当前现状是风险投资者(Stakeholders)普遍使用自然语言作为指定软件系统需求的主要手段,虽然自然语言本身是面向对象的和描述性的,有着强大的表示能力,然而它具有不清晰性、歧义性与不完整性,因此使用自然语言书写的需求文档必须由软件工程师手工转换为清晰的、无歧义的与完整的形式化规约语言.当要构建的软件系统十分复杂时,转换工作不但重要而且容易出错,因此通过自动化转换获取形式化软件规约就显得尤为必要,这已经成为需求工程的重要问题之一.自动化转换最主要的瓶颈有两个:一是自然语言固有的歧义性和复杂性;二是自然语言与形式化规约两个领域不同级别的形式化.正因如此,国内外很少有工作试图将需求文档自动化转换为形式化规约语言.

本文旨在针对问题需求自动化转换为形式化规约这个重要问题,研究从结构化需求语言 SRL 到形式化规约语言 Radl 自动生成系统及其高可靠性理论.为了使研究工作与 PAR 方法及其支撑平台无缝衔接,本文使用 Radl 语言作为形式化软件规约语言[2~4].

2 结构化需求语言 SRL

为了减少或消除自然语言固有的歧义性与复杂性,设计了一种受控自然语言——结构化需求语言 SRL 来描述问题需求[5].使用 SRL 用户即使不精通形式化规约语言,也可以方便地学习、使用、阅读与书写问题需求.本章先对软件需求自然语言结构进行分析,然后阐述设计的结构化需求语言 SRL 语法文本及其特点.

2.1 软件需求自然语言结构分析

受控自然语言是自然语言的子集,它通过对自然语言限制语法与词汇来减少或消除自然语言固有的歧义性和复杂性.迄今为止,世界上已经定义了超过 40 多种受控自然语言,涵盖了汉语、英语、西班牙语、法语、德语、希腊语、日语和瑞典语等.受控自然语言便于人人通讯和人机通讯.目前,受控自然语言正在逐步成为成熟和有用的语言,广泛应用于学术界与工业界,包括航空航天、制造、原油开采、商业规则、公共管理、医药和生物领域.受控自然语言正在成为研究的热点,从 2009 年起每年举办一次受控自然语言国际研讨会(CNL).

区别于其他的受控自然语言,软件需求自然语言有它自身的特色,它在词汇、句型、语义、语用等方面都有其规律.

2.2 SRL 语言文本

SRL语言词语使用上下文无关文法(Context Free Grammar,CFG)来描述其语法,下面给出使用 CFG 描述的 SRL 语言文本,它一共只含 19 条规则.这 19 条规则简单、易学、易用,用户很容易掌握,据此可以方便地描述问题需求.约定如下:

::=被定义为;< >非终结符;[ ]可选项;|任选项;{}重复项.

(1) <算法规约>::=<算法名称>
[<辅助变量说明>;]
[<算法输入描述>;]
<算法输出描述>;

(2) <算法名称>::=名称:标识符;

(3) <辅助变量说明>::=辅助变量:<数据说明表>|ε

(4) <算法输入描述>::=输入:<数据说明表>[满足<谓词式>]|ε

(5) <算法输出描述>::=输出:<数据说明表>满足<谓词式>

(6) <数据说明表>::={<类型名><标识符>;}+

(7) <类型名>::=<基本类型>|<构造类型>

(8) <基本类型>::=整型|实型|字符型|逻辑型|<通用类型>

(9) <构造类型>::=<基本类型> 数组|集合|包|序列|树|图

(10) <谓词式>::=<表达式>|<量词式>|(<谓词式>)

(11) <表达式>::=<简单表达式>[<逻辑运算符> <简单表达式>]|如果<谓词式> 那么<谓词式>[否则<谓词式>]

(12) <量词式>::=<量词>:<标识符表> 满足<谓词式> 求<谓词式>

(13) <简单表达式>::=<项>[<关系运算符><项>]

(14) <项>::= <因子>[<算术运算符>]<因子>

(15) <因子>::= 常量|变量|函数调用|<简单表达式>

(16) <量词>::=所有|存在|累加|累乘|最大|最小|累交|累并|抽象累加|抽象累乘

(17) <逻辑运算符>::=与|或|非|蕴含|等价

(18) <关系运算符>::==|<>|<|<=|>|>=

(19) <算术运算符>::=+|-|*|/|div|mod|^|抽象加|抽象乘

2.3 SRL 特点

① 功能强大;
② 高度数据抽象和可扩充性;
③ 支持泛型及其约束机制;
④ 丰富的语料库;

⑤ 量词结构化.

3 SRLtoRadl 生成系统的设计与实现

3.1 基于规则的形式化软件规约 Radl 生成方法

为了描述结构化需求语言 SRL 与形式化软件规约 Radl 两个领域不同级别的形式化,并且进一步刻画 SRL 语言与 Radl 语言之间的转换关系,可以通过设计这两种不同级别语言的语法规则、语言之间的转换规则来处理.通过深入分析,提炼出源语言 SRL 分析规则、目标语言 Radl 生成规则以及 SRL 到 Radl 转换规则三组规则,它们都使用 CFG 来描述.下面举例说明如下:

① SRL 分析规则:

<量词式>::=<量词><标识符表>满足<谓词式>求<谓词式>

② Radl 生成规则:

<quantifier_exp>::=<quantifier><id_list>:<predication_exp>:<predication_exp>

由于源语言与目标语言之间存在着调整次序、修改、增加、删除操作,为了建立它们之间的关系,通过将结构化需求语言 SRL 与形式化规约语言 Radl 两种语言之间的 CFG 文法建立关联,得到两者之间的转换规则.

③ SRL 到 Radl 转换规则(索引代表链接约束,用来对齐):

<量词式>::=<量词:1><标识符表:2> 满足 <谓词式:3> 求<谓词式:4>

<quantifier_exp>::=<quantifier:1><id_list:2><predication_exp:3>:<predication_exp:4>

SRL 语言与 Radl 语言之间还可能存在多对一、一对多的情形.多对一是由于源语言 SRL 可能存在多种不同表述,包括使用不同词汇,词序颠倒等.我们允许出现这种情况,这是通过设计常用双语语料库与增加转换规则来处理的.一对多是由于源语言 SRL 可能存在歧义性,用户需要根据上下文知识或设定优先级来消歧,这将在 3.2.1 节词法分析、3.2.2 节语法分析与转换中介绍.

在上述三类规则库的支撑下,基于规则的形式化软件规约 Radl 生成方法包含下面三阶段(简称为分析-转换-综合三阶段):

步骤① 源语言分析部分,使用源语言 SRL 分析规则分析输入源语言句子 s;

步骤② 转换部分,使用源语言 SRL 到目标语言 Radl 转换规则以建立目标语言推导序列;

步骤③ 目标语言生成部分,对目标语言 Radl 推导序列使用目标语言 Radl 生成规则得到输出目标语言句子 t.

3.2 SRLtoRadl 生成系统的设计与实现

在基于规则的生成方法指导下,我们设计并实现了从结构化需求语言 SRL 到形式化软件规约 Radl 的生成系统 SRLtoRadl.使用自然语言处理的技术对生成系统 SRLtoRadl 词法分析、语法分析与转换、语义分析中的诸多难点进行了处理.SRLtoRadl 生成系统是利用 Jbuilder 9.0 在 Windows XP 操作系统环境下开发的,由生成引擎和知识库两部分组成,其中生成引擎包括词法分析、语法分析与转换、语义分析三部分,知识库(Knowledge Base,简称 KB)包括表格管理、规则库、出错处理三部分,系统体系结构如图 1 所示.本节简要阐述 SRLtoRadl 生成系统各主要部分的设计与实现.

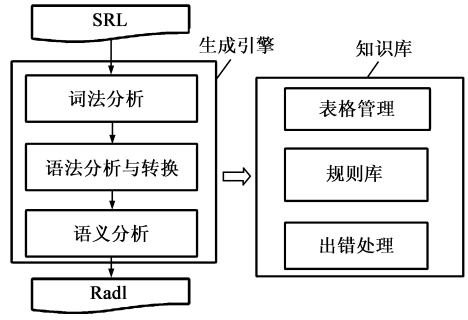

图1 SRLtoRadl生成系统体系结构

3.2.1 词法分析

词法分析主要任务是对 SRL 中文自动分词和词性标注.根据 SRL 描述软件需求的特点,并考虑到后续语法分析的要求,把单词分成常用词汇、保留字、标识符、各种数据类型、算术运算符、关系运算符、逻辑运算符、量词、界符等类别.分词过程中可能遇到歧义(包括交集型歧义与组合型歧义),为了尽量较少歧义,系统使用“最少分词法”(最短路径法)[6]来对 SRL 语言进行自动分词与词性标注.

3.2.2 语法分析与转换

语法分析与转换的任务是识别 SRL 句子的语法结构,并转换生成为形式化规约 Radl 句子.本文使用“线图分析法”[6]进行语法分析与转换.

对于分析过程中产生的结构歧义性(即同一个句子对应两棵以上的语法分析树),引入两个启发性策略来处理:

策略① 非终结符少的转换规则优先于非终结符多的转换规则;

策略② 对于一个输入串而言,分析步骤越短(推导序列越短)越优先.

3.2.3 语义分析

语义分析的任务是解释结构化需求语言 SRL 句子各个部分(单词、语法规则及断言)的意义.SRLtoRadl 生

成系统使用 Chomsky 提出的上下文无关文法 CFG 描述源语言 SRL 的语法,但是仍然存在两个问题:一是 Chomsky 语法生成能力太强,产生许多不符合语法或有歧义的句子;二是标记十分简单,分析能力有限,难以反映自然语言的复杂特性,如量词的嵌套.

本文使用功能合一文法[7](Functional Unification Grammar,FUG)来描述结构化需求语言 SRL 的语义.FUG 适合规则系统,效率也很高,这是我们采用它的主要原因.FUG 采用复杂特征集来描述词、语法规则、句子的结构功能以及语义信息.它试图以单一形式的结构模式来描述特征组合、功能分配、词条和组成成分的顺序,以达到对句子的完全功能描述.由于结构化需求语言 SRL 是结构化的,而复杂特征集也是结构化的,因此使用复杂特征集可以有效地表示 SRL 的语法规则(词组),尤其是描述后置断言中带扩展的量词及其嵌套结构.复杂特征集便于运算,两个复杂特征集通过合一运算可以产生另一个复杂特征集,这与 SRL 句法分析中句型或断言(句子)的产生是一致的.合一运算的作用是合并原有子结构的语义特征信息,构造新的语义特征结构;它还可以检查特征的相容性和规则执行的前提条件是否满足,如果参与合一的特征相冲突,就立即宣布合一失败.

下面先给出复杂特征集与合一运算的定义.

定义 1 复杂特征集功能描述 设 α 为一个复杂特征集,当且仅当 α 可以表示为:

$$\begin{cases} f_1 = v_1 \\ f_2 = v_2 \\ \cdots \\ f_n = v_n \end{cases} \qquad n \geq 1$$

其中,f_i 表示特征名,v_i 表示特征值,且满足以下条件:

(1)特征名 f_i 为原子,特征值 v_i 为原子或另一个复杂特征集;

(2)$\alpha(f_i) = v_i (i = 1, \cdots, n)$,读作:复杂特征集 α 中,特征 f_i 的值等于 v_i.

定义 2 复杂特征集相容 若 α、β 均为复杂特征集,则 α、β 是相容的,当且仅当:

(1)如果 $\alpha(f) = a$、$\beta(f) = b$,且 a、b 都是原子,那么 α、β 是相容的,当且仅当 $a = b$;

(2)如果 $\alpha(f)$、$\beta(f)$均为复杂特征集,α、β 是相容的,当且仅当 $\alpha(f)$、$\beta(f)$相容.

定义 3 合一运算的递归定义

(1)在 a、b 都是原子的情况下,如果 $a = b$,那么 $a \underline{\cup} b = a$,否则 $a \underline{\cup} b = \varnothing$;

(2)如果 α、β 均为复杂特征集,则

i 若 $\alpha(f) = v$,但 $\beta(f)$的值未经定义,则 $f = v$ 属于 $\alpha \underline{\cup} \beta$;

ii 若 $\beta(f) = v$,但 $\alpha(f)$的值未经定义,则 $f = v$ 属于 $\alpha \underline{\cup} \beta$;

iii 若 $\alpha(f) = v_1$,但 $\beta(f) = v_2$,且 v_1 与 v_2 相容(不相抵触),则 $f = (v_1 \underline{\cup} v_2)$属于,否则 $\alpha \underline{\cup} \beta = \varnothing$.

合一运算具有以下的性质:

定理 1 交换律:$\alpha \underline{\cup} \beta = \beta \underline{\cup} \alpha$

定理 2 结合律:$\alpha \underline{\cup} (\beta \underline{\cup} \gamma) = (\alpha \underline{\cup} \beta) \underline{\cup} \gamma$

推论 1 合一运算执行次序与结果无关.

该推论说明了可以并行的执行合一运算,这样显然效率更高.

3.2.4 可扩充性

结构化需求语言 SRL 提供的自定义类型允许用户定义像数据库表、映射、结构类型和其它抽象数据类型,这给 SRL 语言带来了极大的可扩充性,据此可以描述特定领域的问题需求.我们正是根据这种思想,定义了抽象数据类型 Table 及其上的系统连接、数据定义、数据操作等运算,使得 SRL 可以描述数据库应用软件的需求.并进而对规则库进行扩充,可以方便的将该生成方法扩充到数据库应用程序领域.

3.2.5 系统运行界面

SRLtoRadl 生成系统运行界面如图 2 所示:

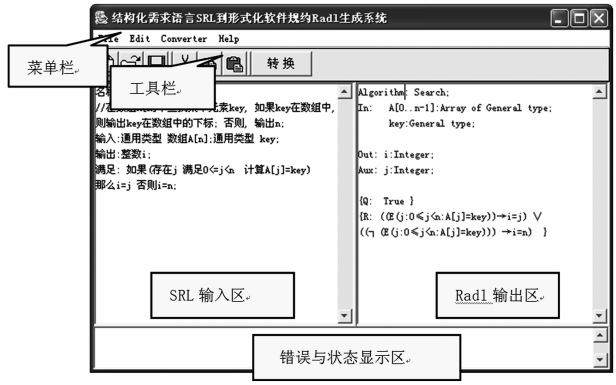

图2 系统运行界面

4 SRLtoRadl 生成系统范畴论语义

虽然使用 SRLtoRadl 生成系统能够生成许多问题需求的形式化规格说明,对生成得到的形式化规格说明通过 PAR 平台运行并仔细测试发现它们也都是正确的,但是如何从理论上保证生成系统的高可靠性? 本章试图使用范畴论框架建立 SRLtoRadl 生成系统生成过程的语义模型,从而从理论上保证生成系统高可靠性.

4.1 生成过程语义模型

基于规则的形式化软件规约 Radl 生成方法分为分

析-转换-综合三个阶段,下面进一步使用范畴论框架来建立了生成系统生成过程的语义模型.

定义 4　一个语言可以定义为 4 元组 $G=(N,\Sigma,P,S)$,其中 N 是非终结符的有限集合;Σ 是终结符的有限集合,$N\cap\Sigma=\varnothing$;$V=N\cup\Sigma$ 称总词汇表;P 是一组重写规则的有限集合:$P=\{\alpha\rightarrow\beta\}$,其中,$\alpha,\beta$ 是 V 中元素构成的串,但 α 中至少应含有一个非终结符号;$S\in N$,称为句子符或初始符.

定义 5　如果 P 中的规则满足如下形式:$A\rightarrow\alpha$,其中 $A\in N,\alpha\in(N\cup\Sigma)^*$,则称该文法为上下文无关文法(CFG).

生成系统中源语言 SRL 与目标语言 Radl 均使用 CFG 来定义其语法.

本节主要使用到两类推导态射:基于源语言 SRL 分析规则的推导 A、基于源语言 SRL 到目标语言 Radl 转换规则的推导 T. 为了后文叙述方便,先对其进行说明如下:

基于源语言 SRL 分析规则的推导 A:在生成过程分析阶段,使用源语言 SRL 分析规则对源语言句子 s 重写而完成的推导,它是一种纵向关系;

基于源语言 SRL 到目标语言 Radl 转换规则的推导 T:在生成过程转换阶段,使用源语言 SRL 到目标语言 Radl 转换规则对源语言句子 s 重写而完成的推导,它是一种横向关系.

定理 3　生成过程分析阶段,基于源语言分析规则的推导系统构成一个源语言范畴 SL_CAT.

同理可证明下面的定理 4:

定理 4　生成过程综合阶段,基于目标语言生成规则的推导系统构成一个目标语言范畴 DL_CAT.

源语言范畴 SL_CAT 与目标语言范畴 DL_CAT 之间关系如下:

定理 5　从源语言范畴 SL_CAT 到目标语言范畴 DL_CAT 构成一个函子 C_FUN.

定理 5 说明了从源语言范畴 SL_CAT 转换到目标语言范畴 DL_CAT 是保持了适当结构的映射. 通过语法分析过程,可以发现这种适当的结构是一种树状结构,设计生成系统必须不能破坏该树状结构.

同理,如果将态射的方向反转,可以证明:

定理 6　从目标语言范畴 DL_CAT 到源语言范畴 SL_CAT 构成一个反变函子 OPP_C_FUN.

定理 6 说明了转换过程是可逆的,这为逆向工程的可行性提供了理论依据.

定义 6　**一轮分析-转换-综合**　指在生成过程中,对源语言句型 s 使用一次基于源语言分析规则的推导,再使用一次基于转换规则的推导以建立目标语言 CFG 推导序列的一步推导,以得到一个输出目标语言的句型.

由定义可知,一轮分析-转换-综合中包含了一个使用基于分析规则的推导 A,一个使用基于转换规则的推导 T,以及最后的生成结果. 对于每一个 SRL-Radl 转换规则对,显然有:

推论 2　在一轮分析-转换-综合中使用基于分析规则的推导与使用基于转换规则的推导与次序无关.

该推论说明了在生成过程中可以并行的应用基于分析规则的推导与基于转换规则的推导,这样设计的生成算法显然效率更高.

定理 7　生成过程中,每一轮分析-转换-综合构成源语言范畴 SL_CAT 一个语言外推 L_PUL.

在范畴中,外推的作用是信息融合. 由于外推是余极限的特例,上述定理也可以写成:生成过程中,每一轮分析-转换-综合构成源语言范畴 SL_CAT 一个语言余极限 L_COL.

定理 8　整个分析-转换-综合过程可以通过不断对源语言范畴 SL_CAT 求外推得到,直到输入源语言句子 s 分析完成.

推论 3　整个分析-转换-综合过程梯状图是可交换的.

该推论说明了设计基于规则的生成系统,采用独立分析-独立转换的生成方法与采用分析与转换同步进行的生成方法语义上是等价的. 当然,前者适合于一种语言转换为多种语言,这样独立分析过程可以重用;后者可以使用并行算法,生成效率显然更高.

定理 9　基于规则的形式化软件规约 Radl 生成过程得到的最终生成结果是可构造的.

可以通过在源语言范畴 SL_CAT 的推导集合中增加语法分析中的源语言分析规则信息使得源语言范畴 SL_CA 携带语法分析证据.

定理 10　携带语法分析证据的源语言范畴 SL_CAT 构成一个范畴 SYN_SL_CAT.

推论 4　基于携带语法分析证据生成过程得到的最终生成结果是可构造的,而且可以保证其语法正确.

4.2　语义分析过程语义模型

由于语义分析在生成过程中可以独立的进行,为了清晰起见,我们把语义分析过程的语义模型专门放在这一节中给出. 本节假定采用功能合一文法来描述结构化需求语言 SRL 的语义,详见 3.2.3 节.

定理 11　基于复杂特征集的推导系统构成一个特征范畴 C_CAT.

定理 12　复杂特征集的合一运算构成一个特征和 C_SUM.

由定理 12 可知,两个复杂特征集 α,β "合一"运算的含义是包含且仅包含 α,β 的属性及其值,除此之外没有任何多余的东西.

可以通过在分析规则中增加复杂特征集使得源语言范畴 SL_CA 同时携带语义分析证据.

定理 13 携带语义分析证据的范畴 SYN_SL_CAT 进一步构成一个范畴 SYN_SEM_SL-CAT.

推论 5 语法分析与语义分析可以通过合一运算同步的进行分析.

推论 6 基于携带语法与语义分析证据生成过程得到的最终生成结果是可构造的,构造过程不仅可以保证其语法是正确的,还可以保证其语义也是正确的.

推论 7 携带语法与语义分析证据基于规则的形式化软件规约生成过程得到的最终生成结果是可构造的,构造过程不仅可以保证其语法是正确的,还可以同步的保证其语义也是正确的.

该推论说明了在生成过程中,语法和语义分析都可以以合一作为基本运算.这样不仅句子的合法性可以通过语义的手段来判断,而且还可以把句子的语法结构和语义表示用合一运算这种方式更加自然地衔接起来.这样设计与实现生成系统可以同步的进行语法分析与语义分析,效率显然更高.

定理 14 范畴 SYN_SL_CAT 是到范畴 SYN_SEM_SL_CAT 的遗忘函子.

定理 15 范畴 SL_CAT 是到范畴 SYN_SL_CAT 的遗忘函子.

定理 14 与 15 说明了忽略掉语义信息与语法信息生成过程仍然是可以继续的.这可以指导编译器或是生成器的设计,如目前的编译器通常仅保留语法信息,而忽略掉部分语义信息进行编译,因此只能检查语法错误和部分语义错误.

5 相关工作比较

国内外很多学者使用不同的方法在形式化软件规约或算法规约生成方面开展了许多研究工作.归纳起来,使用的方法主要可以分为两类:基于领域知识的方法与基于转换生成的方法,这两类方法与本文工作比较如下:

基于知识的方法使用领域知识作为分析需求的基础,重要工作如需求学徒[8],SPECIFIER[9],生成式编程[10],问题框架方法[11],MLIRF 方法[12]等.基于知识的方法一般都具有一个领域知识库,采用人工智能推导技术来对问题求解.该方法一般适用于特定应用领域形式化规约的开发,如信息系统、算法领域、调度领域等;基于这些方法的软件系统实现难度较大,也难以保证实现过程的正确性.本文采用的生成方法面向非特定应用领域,通过用户自定义 ADT 类型,也可以方便的扩充其他特定应用领域;该生成方法可以看做一个产生式系统,易于实现;进一步,可以使用范畴论框架建立生成过程的语义模型.

基于转换生成的方法提供了自动(或人机交互)的 CASE 工具由半形式化的问题表示(如图表、受限自然语言)或非形式化的问题表示(如自由的自然语言)来生成形式化规约.前者主要使用逐步求精技术,如何求精以及如何保证求精前后语义正确性都是很困难的任务[13,14];后者主要使用自然语言处理技术[15~19].它们都侧重于语法级的分析与转换,一般是将半形式化或非形式化的问题表示转换到命令式、面向对象式或申述式形式化规约语言,如 Z、Object Z、VDM++、Prolog 等,缺乏深层次语义级的分析与转换;它们一般能较好地生成形式化规约框架(如类框架)或逻辑简单问题的形式化软件规约(如数据库增加、删除、修改操作),而难以生成复杂算法问题形式化规约.本文生成方法可以归类为一种基于转换生成的方法,不仅可以对语法进行分析与转换,通过对语法规则引入复杂特征集与合一运算,还可以进行深层次的语义分析与转换;它不仅可以生成数据库应用软件形式化规约,还可以生成复杂算法的形式化规约;进一步,可以使用范畴论框架建立生成过程的语义模型.

6 总结与展望

本文的研究成果可以从理论和实现两个方面来总结.本文在理论方面的主要成果是使用范畴论框架建立了 SRLtoRadl 生成系统生成过程的语义模型.本文在实现方面的主要成果一是设计了一种受控自然语言-结构化需求语言 SRL 来描述问题需求;二是设计并实现了从结构化需求语言 SRL 到形式化软件规约 Radl 的生成系统 SRLtoRadl.

我们进一步的工作一是继续完善 SRLtoRadl 生成系统并集成到整个 PAR 平台,二是将本文的研究结果进行扩展,进一步探索使用范畴论理论框架建立 PAR 平台由形式化 Radl 规约生成可执行程序的生成过程语义模型.

参考文献

[1] 金芝等编著.软件需求工程:原理与方法[M].北京:科学出版社,2008.

[2] Xue Jinyun. A practicable approach for formal development of algorithmic programs [A]. Proceedings of the International Symposium on Future Software Technology[C]. Tokyo: Software Engineers Association, 1999. 158 - 160.

[3] Xue Jinyun. PAR method and its supporting platform[A]. Pro-

ceedings of the First International Workshop on Asian Working Conference on Verified Software [C]. Macao: UNU-IIST, 2006.10 – 20.

[4] 石海鹤,薛锦云.基于 PAR 的算法形式化开发[J].计算机学报,2009,32(5):982 – 991.
Shi Haihe, Xue Jinyun. PAR-based formal development of algorithms[J]. Chinese Journal of Computer, 2009, 32(5): 982 – 991. (in Chinese)

[5] 王昌晶,薛锦云.PAR 平台中结构化需求语言研究[J].计算机科学,2006,33(8):99 – 101.
Wang Changjing, Xue Jinyun. Research in structural requirement language of PAR platform[J]. Computer Science, 2006, 33(8): 99 – 101. (in Chinese)

[6] James Allen. Natural Language Understanding[M]. 2nd ed. Massachusetts: Addison-Wesley, 1994.

[7] Martin Kay. Parsing in functional grammar[A]. D Dowty, L Karttunen, A Zwicky Eds. Natural Language Parsing[C]. Cambridge: Cambridge University Press, 1985.125 – 138.

[8] Reubenstein Howard B, et al. The requirements apprentice: automated assistance for requirements acquisition[J]. IEEE Transaction on Software Engineering, 1991, 17(3): 226 – 240.

[9] Miriyala Kanth, et al. Automatic derivation of formal software specifications from informal descriptions[J]. IEEE Transaction on Software Engineering, 1991, 17(10): 1126 – 1142.

[10] Krzysztof Czarnecki, Ulrich W Eisenecker. Generative Programming-Methods, Tools and Applications [M]. Massachusetts: Addison-Wesley, 2000.

[11] Michael Jackson. The problem frames approach to software engineering[A]. Proceeding of the 14th Asia-Pacific Software Engineering Conference[C]. California: IEEE Computer Society, 2007.14 – 14.

[12] Dong Yunmei, Li Kaide, et al. Design and implementation of the formal specification acquisition system SAQ[A]. Proceedings of Conference on Software: Theory and Practice, IFIP 16th World Computer Congress[C]. Beijing: Publishing House of Electronics Industry, 2000.201 – 211.

[13] Lijun Dong, Jiafu Xu. Formal semantics of some functional constructs of the software requirements definition language NDRDL[A]. Proceeding of the 21st International Computer Software and Applications Conference[C]. California: IEEE Computer Society, 1997.642 – 645.

[14] 张家重,等.对象式软件需求模型及其机器支撑[J].软件学报,1998,9 (6):414 – 418.
Zhang jiazhong, et al. A hierarchical object-oriented software requirements model and its mechanical support[J]. Journal of Software, 1998, 9(6): 414 – 418. (in Chinese)

[15] Sullabi Mohammed, Shukur Zarina. SNL2Z: Tool for translating an informal structured software specification into formal specification[J]. American Journal of Applied Sciences, 2008, 5(4): 378 – 384.

[16] M G Ilieva, Olga Ormandjieva. Automatic transition of natural language software requirements specification into formal presentation[A]. Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems [C]. Berlin: Springer, 2005.392 – 397.

[17] Shukur Zarina, et al. M2Z: A tool for translating a natural language software specification into Z[A]. Proceedings of the 4th International Conference on Formal Engineering Methods[C]. Berlin: Springer, 2002.406 – 410.

[18] Lee Beum-Seuk, et al. Automated conversion from requirements documentation to an object-oriented formal specification language[A]. Proceedings of the ACM Symposium on Applied Computing[C]. New York: ACM, 2002.932 – 936.

[19] Fuchs Norbert E, Schwertel Uta, Schwitter Rolf. Attempto controlled English-not just another logic specification language[A]. Proceedings of the 8th International Workshop of Logic Programming Synthesis and Transformation [C]. Berlin: Springer, 1998.1 – 20.

作者简介

王昌晶　男,1977 年 10 月出生,江西丰城人.博士,副教授,硕导.研究方向:软件形式化与自动化,软件规约方法.
E-mail: wcj771006@163.com

薛锦云　男,1947 年 7 月出生,江苏海门人.教授,博导.研究方向:软件形式化与自动化,可信软件的构造与验证.

左正康　男,1980 年 3 月出生,江西抚州人.博士、讲师,研究方向:软件形式化与自动化,泛型程序设计.