◎研发、设计、测试◎

基于 OCL 的配置工具研究与实现

闫会强[1],肖国玺[2],檀润华[1],王秀娟[2]
YAN Hui-qiang[1],XIAO Guo-xi[2],TAN Run-hua[1],WANG Xiu-juan[2]

1.河北工业大学 创新设计研究所,天津 300130
2.河北工业大学 计算机科学与软件工程学院,天津 300130
1.Institute of Design for Innovation,Hebei University of Technology,Tianjin 300130,China
2.School of Computer Science and Engineering,Hebei University of Technology,Tianjin 300130,China
E-mail:yanhuiqiang@scse.hebut.edu.cn

YAN Hui-qiang,XIAO Guo-xi,TAN Run-hua,et al.Research and realization of OCL-based configuration tool.Computer Engineering and Applications,2009,45(6):73-77.

Abstract: Mass customization is the trend of modern industry,which not only has the advantage of low cost,but also can meet individual customer requirements.Product configuration is an important method to realize mass customization,and configuration tool is the core of product configuration.GPCT(Generic Product Configuration Tool) is developed by Institue of Design for Innovation, Hebei University of Technology and it is a domain-independent configuration tool.GPCT utilizes UML to represent configuration model and OCL(Object Constraint Language) to represent constraints among product parts.The constraint maintenance and evaluation of GPCT are presented in this paper,taking CNC vertical grinder being as an example,the application of GPCT is introduced.

Key words: mass customization;object constraint language;product configuration;configuration model

摘 要:大规模定制是现代工业发展的趋势。它不但具有大规模生产的低成本优势,而且能够满足用户的个性化需求。产品配置是实现大规模定制的重要方法,实现产品配置的核心是配置工具。GPCT(Generic Product Configuration Tool)是河北工业大学创新设计研究所开发的领域无关的配置工具。它以 UML 表示产品配置模型,以 OCL 文法表达产品部件间的约束。给出了 GPCT 的约束维护和验证方法,并以数控立式磨床为例,介绍了 GPCT 的实际应用。

关键词:大规模定制;对象约束语言;产品配置;配置模型
DOI:10.3778/j.issn.1002-8331.2009.06.022 **文章编号:**1002-8331(2009)06-0073-05 **文献标识码:**A **中图分类号:**TP31

1 概述

配置是人工智能领域研究的问题之一,而产品配置设计的思想最早是由 Freeman 和 Newell 在 1971 年提出的。但是普遍接受的关于配置的定义由 Mittal 等给出,他们将配置定义为[1]:

条件:(1)一组预先定义的组件,组件通过属性和其它组件连接的接口以及施加在接口上的约束来描述。(2)预期配置的功能描述。(3)可能存在的最优化选择标准。

构建:满足用户需求的配置是选择的一组组件和描述组件间连接关系的描述,或者检测与需求的不一致性。

因此配置是从预定义的组件中选择满足用户需求和约束关系的一组组件集合。当把产品模型的描述范围界定在配置领域的时候,称为配置模型[2]。文章采用配置模型描述组件间的构成关系和约束规则。

统一建模语言 UML(Unified Modeling Language)是一种用于描述、构造软件系统以及商业建模的语言,它综合了在大型、复杂系统的建模领域得到认可的优秀的软件工程方法。OCL 是 UML 的非常重要的组成部分, 能够以一种无歧义的方式定义业务规则。目前 UML 在工业界产品模型的设计中同样得到了广泛应用。GPCT 采用 UML 和 OCL 表示配置模型,图 1 给出了数控立式磨床配置模型的一部分。配置模型中产品的组成部分称为部件,实例化后称为部件实例。

GPCT 的构成如图 2 所示。词法分析和语法分析检查 OCL 约束书写是否正确;配置模型由产品树和 OCL 约束构成;配置约束模板简化配置约束的编写;配置控制在配置过程中为用户

基金项目:国家自然科学基金(the National Natural Science Foundation of China under Grant No.50675059);天津市自然科学基金(the Natural Science Foundation of Tianjin of China under Grant No.07JCZDJC08900);河北省自然科学基金(the Natural Science Foundation of Hebei Province of China under Grant No.E2008000101)。

作者简介:闫会强(1972-),博士生,主要研究方向:大规模定制、产品配置。

收稿日期:2008-10-15 **修回日期:**2008-11-28

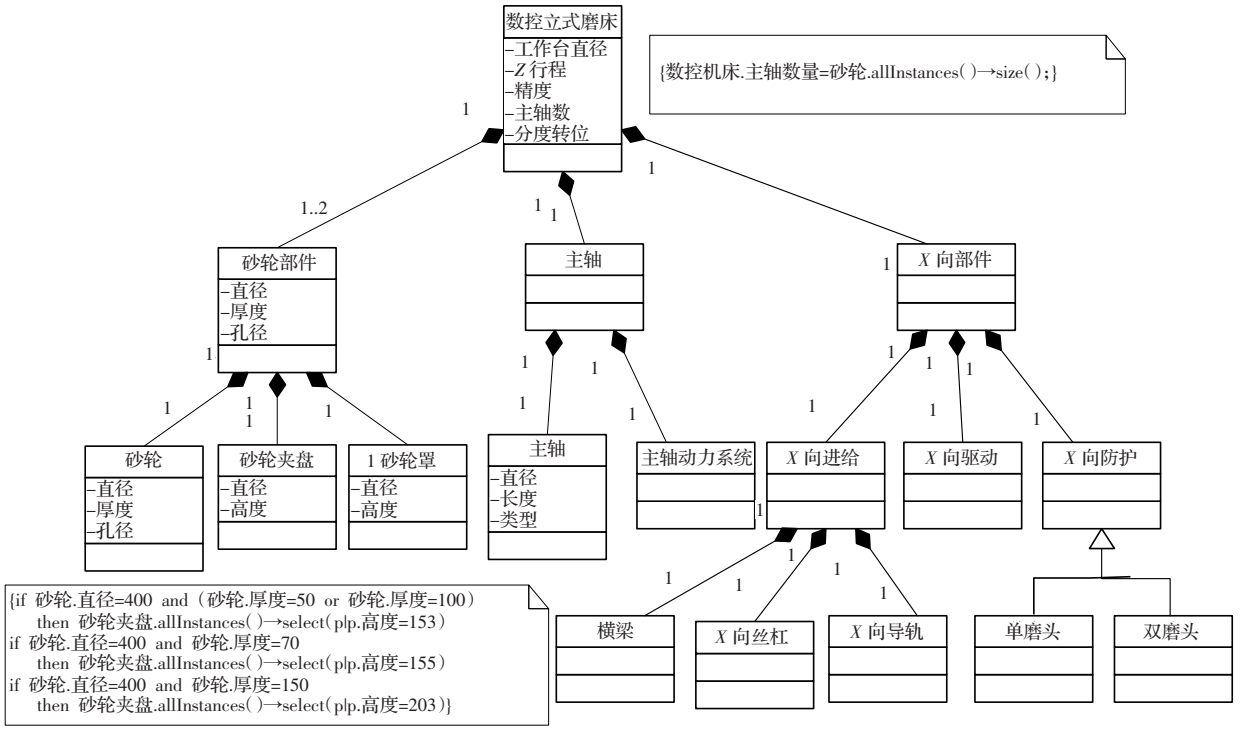

图1 数控立式磨床配置模型的一部分

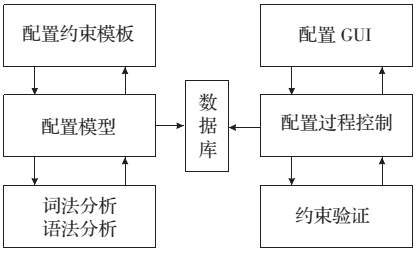

图2 GPCT组成

配置提供帮助;约束验证在配置过程中对配置选项是否满足配置约束进行验证。配置GUI是用户配置界面。重点介绍了应用OCL文法实现GPCT约束处理的关键技术,并以数控立式磨床为例说明了GPCT的应用。

2 OCL文法和应用

OCL起源于1997年BIM公司为响应OMG的"面向对象分析和设计标准"征求稿所提交的"对象时间限制提议",OCL是该提议的部分内容。

用OCL可以描述4类约束,分别是不变量、前置条件、后置条件和监护条件[3]。

(1)不变量是在属性的生命期内一直保持为真的规则。

(2)前置条件是在一个操作被调用时必须为真的约束。它是一个断言,不是可执行语句。

(3)后置条件就是在操作完成时必须为真的约束。它不是可执行语句而是断言,必须为真。

(4)监护规则是在对象能够从一种状态转变为另一种状态前其值必须为真的约束。

GPCT目前只使用了不变量约束条件,下面是两个约束的例子。

约束1

Context 数控立式磨床 inv:

Self.主轴数量=砂轮.allInstance()->size()。

约束2

Context 砂轮 inv:

if self.直径=400 and (self.厚度=50 or self.厚度=100)

then 砂轮夹盘.allInstances()->

select(p |p.高度=153)

这里Context用于指明约束的上下文,inv用于指明约束的类型是invariant。Self指代约束上下文指定的部件。对于约束1而言指代数控立式磨床,对于约束2而言指代砂轮。约束1要求"数控立式机床的主轴数量和选择的砂轮数量相等"。约束要求"砂轮的直径等于100、高度等于50或者100时,所选择所有砂轮夹盘的高度是150"。

OCL是一种强类型的语言,任何一个元素都是有类型的,并且任何操作的返回值都有一个确定的类型。OCL基本数据类型包括Boolean、Integer、Real和String。除此之外还包括UML模型中定义的各种部件类型以及集合类型(set、Sequence、Bag和OrderedSet)。

OCL对基本数据类型的操作如表1所示。OCL是基于集合论的,但并不是集合论中所有的集合操作在OCL中都具有相应的符号表达。例如投影操作"project"在OCL中没有相应的符号。OCL对集合类型给出的主要操作包括Select、Reject、

表1 OCL基本数据类型操作

数据类型	操作
Integer	*,+,-,/,abs()
Real	*,+,-,/, floor()
Boolean	and,or,xor,not,implies,if-then-else
String	concat(),size(),substring()

ForAll、Exists、Iterate 等操作。

OCL 是一种查询语言，任何 OCL 的动作都不会对模型本身造成任何影响或者改变，这也意味着系统状态不会因为 OCL 约束表达式而改变[3]。因此 OCL 适合作为配置的约束描述语言。

3　OCL 约束

GPCT 中约束处理是核心，主要包含两部分，约束维护和约束验证，如图 3 所示。

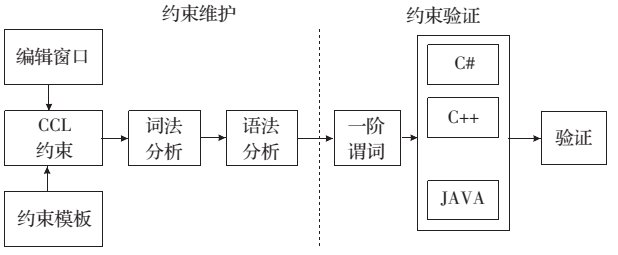

图 3　GPCT 约束处理

3.1　词法分析

词法分析的任务是对输入的字符串形式的 OCL 约束按顺序进行扫描，在扫描的同时，根据 OCL 的词法规则识别出具有独立意义的单词，并产生与其等价的属性字作为输出。词法分析确保输入的所有单词是正确的。GPCT 只是采用了 OCL 的部分文法做为配置约束语法。GPCT 配置约束语法中不支持：

(1)定义变量语句，即不使用 let 语句；

(2)前置条件和后置条件语句；

(3)消息处理语句；

(4)记录类型；

(5)路径定义。

词法分析程序将 OCL 文法单词分为以下几类：

(1)关键字：Integer、Real、Boolean、String 等。

(2)常数：整型常数、布尔型常数、字符型常数等。

(3)运算符：包括算术运算符(*，+，-，/等)、字符串运算符(concat()，size()，substring())、逻辑运算符(and，or，xor，not，implies，if-then-else)和集合运算符(select()、reject()、includes()和 excludes()等)。

(4)分界符：双引号、(和)等。

(5)标识符号：配置模型中部件名称、部件实例名称和属性。

词法分析程序根据输入的 OCL 约束字符串，按照如上的分类方法进行识别，凡是不属于上述 5 类单词的均给出错误信息，并指明位置。

3.2　语法分析

语法分析程序以词法分析输出 OCL 规则字符串作为输入，根据 OCL 的文法规则作语法检查，确保 OCL 规则全部书写正确。GPCT 文法分析功能完成约束表达式的分析，GPCT 所使用的表达式文法是基于 OCL 表达式文法，但是消除了左递归和回溯问题。这里只给出了表达式文法，如下所示，其它文法可以衍生出来。

(1)<logical-implies-expression>::=
<logical-xor-expression><logical-implies-expression'>
(2)<logical-implies-expression'>::=implies
<logical-xor-expression><logical-implies-expression'>
(3)<logical-implies-expression'>::=null
(4)<logical-xor-expression>::=<logical-or-expression>
<logical-xor-expression'>
(5)<logical-xor-expression'>::=xor
<logical-or-expression><logical-xor-expression'>
(6)<logical-xor-expression'>::=null
(7)<logical-or-expression>::=<logical-and-expression>
<logical-or-expression'>
(8)<logical-or-expression'>::=or<logical-or-expression>
<logical-or-expression'>
(9)<logical-or-expression'>::=null
(10)<logical-and-expression>::=<relational-expression>
<logical-and-expression'>
(11)<logical-and-expression'>::=and
<relational-expression><logical-and-expression'>
(12)<logical-and-expression'>::=null
(13)<relational-expression>::=<add-expression>
<relational-expression'>
(14)<relational-expression'>::=<relational-operator>
<add-expression>
(15)<relational-expression'>::=null
(16)~(21)<relational-operator>::=<|<=|>|>=|<>|=
(22)<add-expression>::=<mul-expression>
<add-expression'>
(23)~(25)<add-expression'>::=+<mul-expression>
<add-expression'>|
-<mul-expression><add-expression'>| null|
(26)<mul-expression>::=<unary-expression>
<mul-expression'>
(27)~(29)<mul-expression'>::=*<unary-expression>
<mul-expression'>|null|
/<unary-expression><mul-expression'>
(30)~(32)<unary-expression>::=<primary-expression>|
-<unary-expression>|
not<unary-expression>
(33)~(34)<primary-expression>::=<literal>
<primary-expression'>|
(<expression>)<primary-expression'>
(35)~(37)<primary-expression'>::=.<property-call>
<primary-expression'>|
-><method-call><primary-expression'>|null
(38)<property-call>::=< literal >
(39)~(42)<method-call>::=select()|
reject()|allInstance()|size()

因为 GPCT 表达式文法不含直接左递归、间接左递归和回溯问题，因此可以采用 LL(1)语法分析方法[4]。LL(1)分析方法由三部分构成，如图 4 所示。输入串是使用 OCL 语言编写的约束规则，分析栈用于存放分析过程中的文法符号，总控程序利

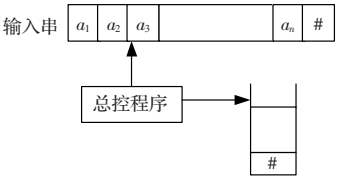

图 4　LL(1)分析方法组成

表 2 LL(1)分析表

非终结符	终结符													
	implies	xor	or	And	<|<=|>|>=|=|<>	+|-	*|/	部件|部件实例|属性	-|not	(	literal	.|->	Select|reject|等
logical-implies-expression								(1)	(1)	(1)	(1)		
logical-implies-expression'	(2)				(3)	(3)	(3)	(3)	(3)	(3)	(3)	(3)	
logical-xor-expression								(4)	(4)	(4)	(4)		
logical-xor-expression'	(6)	(5)			(6)	(6)	(6)	(6)	(6)	(6)	(6)		
logical-or-expression								(7)	(7)	(7)	(7)		
logical-or-expression'	(9)	(9)	(8)		(9)	(9)	(9)	(9)	(9)	(9)	(9)		
logical-and-expression								(10)	(10)	(10)	(10)		
logical-and-expression'	(12)	(12)	(12)	(11)	(12)	(12)	(12)	(12)	(12)	(12)	(12)		
relational-expression								(13)	(13)	(13)	(13)		
relational-expression'	(15)				(14)								
relational-operator					(16)~(21)								
add-expression						(22)	(22)	(22)	(22)	(22)	(22)		
add-expression'	(25)	(25)	(25)	(25)	(25)	(23)	(25)	(25)	(25)	(25)	(25)		
						(24)							
mul-expression						(26)	(26)	(26)	(26)	(26)	(26)		
mul-expression'	(29)	(29)	(29)	(29)	(29)	(29)	(27)(28)	(29)	(29)	(29)	(29)		
unary-expression						(30)	(30)	(30)	(31)(32)	(30)	(30)		
primary-expression								(33)		(34)	(33)		
primary-expression'	(37)	(37)	(37)	(37)	(37)	(37)	(37)	(37)	(37)	(37)	(37)	(35)(36)	
property-call								(38)					
method-call													(39)~(42)

用 LL(1)分析表和分析栈对输入串识别和分析。根据 GPCT 表达式文法构造的 LL(1)分析表如表 2 所示。

3.3 OCL 规则验证

OCL 是基于一阶谓词逻辑的，因此 OCL 约束验证可以使用一阶谓词逻辑[3]。对于 GPCT 而言，验证前先将约束改写为一阶谓词，然后转化为 C# 语言进行验证。

GPCT 采用文献[5]的方法将 OCL 约束转化为一阶谓词形式，但是进行了简化。

(1)OCL 基本数据类型

OCL 基本数据类型可以出现在子句中无需任何转换操作。

(2)Context

Context 指明了约束存在的上下文。约束中的 Self 和上下文指定的部件一致。因此对于约束中的 Self 使用 Type(实例名,类名)转换。例如,配置过程中选择了数控立式磨床 A,则 Self 转换为 type(数控立式磨床 A,数控立式磨床)。约束中其它的组件名使用相同的转换方法。

(3)属性

对于规则中指明的属性使用 Val(部件实例,属性 P,属性值)。该子句为真的条件是指定部件中已经定义了属性 P,然后根据配置对象确定该属性值。例如

数控立式磨床.主轴数量=1

当配置选项是数控立式磨床 A 时,转换后为

Val(数控立式磨床 A,主轴数量,1)

(4)基本操作符和常量

基本操作符和常量转换时,直接将操作符左侧的符号改变为对应值,而常量不做变化。例如

```
Context 数控立式磨床 inv:
        Self.主轴数量>1
```

对此约束转换后得到如下子句

type(数控立式磨床 A,数控立式磨床)∧val(数控立式磨床 A,主轴数量,2)∧2>1

(5)If 条件句

If 条件句的形式为 if P then Q,当 p 为真时,q 也为真时,子句值为真;当 p 为假时,子句为真。因此此种形式的 OCL 语句等价于 $\sim P \vee Q$。

对于形式为 if P then Q else R 的条件句而言，等价于 $(P \wedge Q) \vee R$。

(6)关联部件访问

从一个部件访问另一个部件在约束中经常遇到,这类操作形如部件名.部件名,如砂轮模块.砂轮。此类操作转换为如下的子句。

Type(部件实例,部件)∧Type(部件实例,部件)∧Relation(部件,部件)

例如,砂轮模块.砂轮验证时转换为:

type(砂轮模块 A,砂轮模块)∧type(砂轮 A,砂轮) ∧Relation(砂轮模块,砂轮)

(7)Select 和 Reject 操作

Select 的表达形式为,collection->Select(布尔表达式),是集合元素中选择使布尔表达式为真的集合元素。下面是一个选择操作的例子。

```
Context 砂轮 inv:
if self.直径=400 and self.厚度=50 or self.厚度=100)
        then 砂轮夹盘.allInstances()->
                Select(p |p.高度=153)
```

在砂轮直径和厚度满足一定的条件后,要求所选择所有砂轮夹盘实例的高度等于 153。假设所选择夹盘分别为砂轮夹盘 1、砂轮夹盘 2 和砂轮夹盘 3,高度分别为 150、153 和 153。Select 部分转换后的子句为:

Val(砂轮夹盘 1，高度，150)∧150=153∧
Val(砂轮夹盘 2，高度，153)∧153=153　∧
Val(砂轮夹盘 3，高度，153)∧153=153

同理可以给出 Reject 操作的转换子句。

OCL 是一种声明式语言，描述了做什么，没有描述如何做[3]。GPCT 采用 C# 语言，为了实现约束验证，需要将转换后的一阶谓词用 C# 语言实现。转换时最重要的是确定属性名、操作和部件间的联系[6]，具体转换细节这里不做讨论。

4　配置约束模板

掌握 OCL 语言对于用户来说不是一件容易的事情。为了使得书写约束方便快捷，一方面，设计了易学易用的约束规则编写画面如图 5 所示；另一方面，GPCT 将约束模板定义为插件，新的约束模板可以方便的插入到 GPCT 中。目前支持的约束模板包括：属性值约束模板；Exist 模板；IfThenElse 模板；ForAll 模板；排斥矩阵模板。

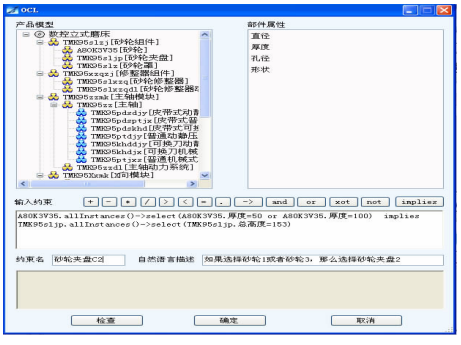

图 5　OCL 编辑画面

当 GPCT 应用于不同领域时，可以开发出不同的约束模板作为插件，以方便用户使用。

5　配置过程控制

配置过程控制是指配置过程中用户可以采用的策略。目前 GPCT 支持启发式和非启发式两种配置策略。启发式配置策略分为 Propose-and-Backtrack 和 Propose-and-Revise[7]。Propose-and-Backtrack 策略指在配置过程中配置工具根据配置规则给出配置选项，用户确定配置选项后进行约束验证，如果验证失败，取消此选项并选择其它配置选项；如果没有合适的配置选项，就取消上一级的配置选项。Propose-and-Revise 策略是指在配置过程中一旦发现没有配置选项的情形，不是取消前面的选项，而是修改上一级的选项。非启发式策略是指配置过程中，配置选项完全由用户指定，系统不给出任何配置选项提示；系统记录配置顺序，用户可以任意取消已经做的配置；配置约束的验证时机由用户决定，可以做一个配置选项后立即做约束验证，也可以配置一部分后进行约束验证。

6　结论

GPCT 的特点是领域无关。目前 GPCT 已经应用于数控立式机床的配置设计，如图 6 所示。应用 GPCT，产品设计周期由原来的 40 天以上缩短为 10 天左右。

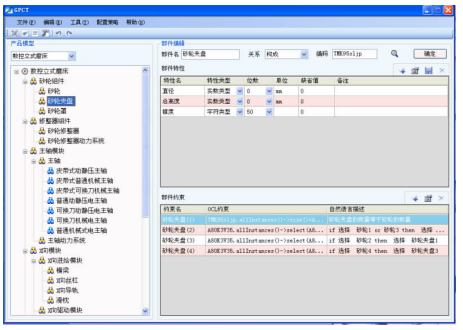

图 6　GPCT 系统

对于复杂机械产品而言，约束规则成百上千，如何保证约束的一致性非常重要。例如，如果一个规则要求 $I>5$，而另一个规则要求 $I<4$，两个约束是互相矛盾的。如何检查出互相矛盾的约束规则是下一步工作。

目前，GPCT 还不能支持 3-D，实现“所见即所得”是系统需要完善的功能。

参考文献：

[1] Mittal S，Frayman F.Towards a generic model of configuration tasks [C]//Proceedings of the 11th IJCAI，San Mateo，CA，Morgan Kaufman，1989：1395-1401.

[2] Tiihonen J，Lehtonen T，Soininen T，et al.Modeling configurable product families[C]//4th WDK Workshop on Product Structuring，Delft University of Technology，October 22-23 1998.

[3] Object Constraint Language Specification(OCL)[EB/OL].http://www.omg.org.

[4] 陈英.编译原理[M].2 版.北京：北京理工大学出版社，2006.

[5] Felfernig A，Friedrich G，Jannach D.Generating product configuration knowledg internationa bases from precise domain extended UML models[C]//Proceedings of the Conference on Software Engineeringand Knowledge Engineering (SEKE'2000)，Chicago，USA，2000：284-293.

[6] Lengyel L，Levendovszky T，Charaf H.Implementing an OCL compiler for NET[C]//Proceedings of the 3rd International Conference on NET Technologies，Pilsen，Czech Republic，May-June 2005：121-130.

[7] Wielinga B J，Schreiber A T.Configuration design problem solving[J].IEEE Expert：Intelligent Systems and Their Applications，1997，12(2)：49-56.