一种面向对象程序自动化方法的研究*

*全炳哲 *金淳兆 **玄顺姬

*(吉林大学计算机系 长春 130023)

**(吉林工业大学计算机系 长春 130025)

摘要 构造性类型理论可作为研究程序自动化的理论基础. 本文根据一种支持面向对象计算的类型理论,探讨了面向对象程序自动化的途径. 面向对象程序的程序单位是类,它是数据和一组相关操作的统一体,程序开发的关键在于开发这些操作,本文重点讨论了开发这些操作的方法.

关键词 构造性类型理论,面向对象程序设计,程序自动化,软件自动化.

中图分类号 TP311

"软件开发自动化是提高软件生产率,保证软件产品可靠性的途径之一. 算法设计是软件开发中最困难,也是最富创造性的活动,因而算法设计自动化的研究构成了软件开发自动化研究的核心内容". [1]近年来,软件自动化方面的研究工作很活跃,例如,基于演绎推理的途径、程序变换途径、基于归纳推理的程序综合、基于可复用构件的软件构造方法、过程实现途径和基于知识的程序综合途径等方面的研究. 但是,这些工作仍处于研究和探索阶段.

基于演绎推理的程序自动化方法出现较早. 80 年代初开始,Manna 深入研究了基于 Tablueax 方法的一种程序自动化方法[2,3],它把非子句归结、数学归纳及转换规则融于一个演绎系统. 基于 Marin-Löf 构造性类型理论的程序构造技术是近几年受到人们重视的程序自动化方法. 这种类型理论既是一种逻辑,又是一种程序设计语言. 在该理论中程序规约(Program specification)可作为类型. 另一方面,类型又是公式,公式的证明过程对应于程序的开发过程,由证明可抽取程序. 近年来,许多研究人员探讨了基于类型理论的程序自动化技术. [4~6]Nuprl 系统[7]是基于 Martin-Löf 类型理论的一种程序开发系统. 在该系统中以自顶向下方式构造证明,证明结束后便可抽取相应的程序.

为研究面向对象软件自动化问题,基于 Martin-Löf 的直觉主义逻辑,我们曾提出了支持面向对象程序设计的构造性类型理论 TTOOP(type theory for object-oriented programming). [8]本文将讨论基于 TTOOP 的程序设计方法. 第 1 节介绍 TTOOP 类规约和类的描述方法,第 2 和 3 节讨论面向对象软件的开发方法,第 4 节是结束语. 文中未给出的记法参

* 本文研究得到国家自然科学基金、国家"九五"攻关项目和国家 863 高科技项目基金资助. 作者**全炳哲**,1961 年生,副教授,主要研究领域为软件自动化,面向对象技术,软件重用. **金淳兆**,1937 年生,教授,主要研究领域为软件自动化,面向对象技术,软件重用,软件工程. **玄顺姬**,女,1961 年生,讲师,主要研究领域为软件自动化,面向对象技术.

本文通讯联系人:全炳哲,长春 130023,吉林大学计算机系

本文 1996-06-10 收到修改稿

见文献[8,9].

1　类规约及类

类是面向对象程序的基本组成部分,因此用面向对象方法开发程序的核心是类的构造方法.在构造性类型理论中,公式当作类型,是程序规约,公式的证明过程对应于程序的开发过程.若给定的公式得到证明,则可从证明抽取满足程序规约的正确程序.在TTOOP中,我们引入了类规约的概念.它是一种类型,类规约的证明过程就是类的开发过程.由于在面向对象计算中,类是一种模板,由它生成性质相类似的对象,因此TTOOP中类也作为一种类型,其元素是对象.类规约类型的形式如下:

$(\exists\ C{:}U_0)$

$(\exists\ (v_1{::}B_1){:}U_0)$

...

$(\exists\ (v_m{:}B_m{:}U_0)$

$(\exists\ f_1{:}\ C{\Rightarrow}D_1{\Rightarrow}A_1\wedge C))$

...

$(\exists\ f_n{:}\ C{\Rightarrow}D_n{\Rightarrow}(A_n\wedge C))$

$(S(v_1{::}B_1,\ldots v_m{::}B_m,f_1,\ldots,f_n)\wedge P(v,\ldots,v_m,B_1,\ldots,B_m,f_1,\ldots,f_n))$

其中 $v_i{::}B_i$ 表示标记类型,用于说明类的实例变量 v_i 及其类型 B_i;D_i 表示操作的定义域类型,A_i 为基本类型或已定义的类.S 是一个等式 $C=_{U0}(v_1{::}B_1\wedge\ldots v_m{::}B_m\wedge\ldots\{f_1\}\wedge\ldots\wedge\{f_n\})$,用于描述 C 类型的结构,称为结构公式,其中 $\{f_i\}$ 表示由 f_i 元素构成的类型.P 用于描述 B_i 类型及 f_j 性质,称为特性公式.

操作 f_i 是满足 $C{\Rightarrow}D_i{\Rightarrow}(A_i\wedge C)$ 类型及特性公式的某元素.其中第1个 C 表示消息的接收者类型,D_i 表示参数类型,$A_i\wedge C$ 表示操作的结果类型.这个结果类型表示操作均有2个返回值,其中第1个返回值类似于一般程序设计语言中操作的返回值,第2个返回值表示新生成的对象,称为操作的结果对象.

在TTOOP中,特征公式 P 采用如下形式:

$(B_1=_{u0}T_1\wedge\ldots B_m=_{u0}T_m\wedge\forall\ obj{:}C\ \forall\ param{:}D_1.Q_1\wedge\ldots\wedge\forall\ obj{:}C\ \forall\ param{:}D_n.Q_n)$

其中 $B_i=_{u0}T_i$ 用于描述实例变量的类型,$\forall\ obj{:}C\ \forall\ param{:}D_j.Q_j$ 用于描述操作 f_j 的功能特性.

根据类规约的结构,类规约的元素结构如下:

$$(C,(v_1{::}B_1,(\ldots,(v_m{::}B_m,(f_1,(\ldots,(f_n,p)\ldots)$$

其中 $C,v_1{::}B_1,\ldots,v_m{::}B_m$ 是类型,$f_1,\ldots,f_n$ 是操作,p 是 P 的一个证明.如上的类规约元素简记为 $(C,v_1{::}B_1,\ldots,v_m{::}B_m,f_1,\ldots,f_n,p)$.

定理 1.1. 设 *ClassSpec* 是一个类规约.若 $(C,v_1{::}B_1,\ldots,v_m{::}B_m,f_1,\ldots,f_n,p){:}$ *ClassSpec*,则 $v_1{::}B_1\wedge\ldots v_m{::}B_m\wedge\{f_1\}\wedge\ldots\wedge\{f_n\}$ 是满足类规约的类型.

定义 1.1. 设 *ClassSpec* 是一个类规约,且 $(C,v_1{::}B_1,\ldots,v_m{::}B_m,f_1,\ldots,f_n,p){:}$ *ClassSpec*,则类型 $v_1{::}B_1\wedge\ldots v_m{::}B_m\wedge\{f_1\}\wedge\ldots\wedge\{f_n\}$ 称为满足 *ClassSpec* 的类.

2 程序构造的基本原理

设 *ClassSpec* 是一个类规约. 如果我们能够构造 $v_i::B_i$ 和 f_j,则很容易构造满足该类规约的一个类. 由于 *ClassSpec* 中有形式为 $B_i=_{w0}T_i$ 的公式,所以易于构造 $v_i::B_i$. 因此,如何构造 f_j 是开发类的核心问题.

定理 2.1. 若 $r:\forall\ f_1:A_1\Rightarrow B_1.\exists\ f_2:A_2\Rightarrow B_2.\forall\ y:A_2.Q(y,f_2\ y,f_1)$,且$\exists\ f_1:A_1\Rightarrow B_1.(\forall\ x:A_1.P\ (y,f_2\ y,f_1)[Fst\ (r\ f_1)/f_2])$可证,则$\exists\ f_1:A_1\Rightarrow B_1.\exists\ f_2:A_2\Rightarrow B_2(\forall\ x:A_1.P(x,f_1\ x,f_2)\wedge\forall\ y:A_2.Q(y,f_2\ y,f_1))$可证.

证明: ①若 $F1:A_1\Rightarrow B_1$,则根据全称量词的消去规则有

$r\ F1:\exists\ f_2:A_2\Rightarrow B_2.\forall\ y:A_2.Q(y,f_2\ y,f_1)[F1/f_1]$

因此,根据存在量词的消去规则有 $Fst\ (r\ F1):A_2\Rightarrow B_2$

所以,$\exists\ f_1:A_1\Rightarrow B_1.(\forall\ x:A_1.P\ (y,f_2\ y,f_1)[Fst\ (r\ f_1)/f_2])$是类型.

②设 $q:\exists\ f1:A_1\Rightarrow B_1.\forall\ x:A_1.P\ (y,f_2\ y,f_1)[Fst\ (r\ f_1)/f_2])$,则根据存在量词的消去规则有

$$Fst\ q:A_1\Rightarrow B_1 \tag{1}$$

$$Snd\ q:\forall\ x:A_1.P\ (y,f_2\ y,f_1)[Fst\ (r\ f_1)/f_2][Fst\ q/f_1]$$

$$即\ Snd\ q:\forall\ x:A_1.P\ (y,f_2\ y,f_1)[Fst\ (r\ (Fst\ q))/f_2][Fst\ q/f_1] \tag{2}$$

因为 $Fst\ q:A_1\Rightarrow B_1$

所以,根据全称量词的消去规则有

$$r\ (Fst\ q):\exists\ f_2:A_2\Rightarrow B_2.\forall\ y:A_2.Q(y,f_2\ y,f_1)[Fst\ q/f_1]$$

$$所以\ Fst(r(Fst\ q)):A_2\Rightarrow B_2 \tag{3}$$

$$且\ Snd(r(Fst\ q)):\forall\ y:A_2.Q(y,f_2\ y,f_1)[Fst\ q/f_1][Fst(r(Fst\ q))/f_2] \tag{4}$$

根据(2)和(4),利用合取引入规则有 $(Snd\ q,Snd(r(Fst\ q))):(\forall\ x:A_1.P(x,f_1\ x,f_2)\wedge\forall\ y:A_2.Q(y,f_2\ y,f_1))\ [Fst\ q/f_1][Fst(r(Fst\ q))/f_2]$ (5)

根据(1),(3)和(5),利用存在量词的引入规则有$(Fst\ q,Fst\ (r\ (Fst\ q)),(Snd\ q,Snd\ (r\ (Fst\ q))))$:$\exists\ f_1:A_1\Rightarrow B_1.\exists\ f_2:A_2\Rightarrow B_2(\forall\ x:A_1.P(x,f_1\ x,f_2)\wedge\forall\ y:A_2.Q(y,f_2\ y,f_1))$

证毕.

推论 2.1. 若 $r:\forall\ f_1:A_1\Rightarrow B_1.\exists\ f_2:A_2\Rightarrow B_2.\forall\ y:A_2.Q(y,f_2\ y,f_1)$,且 $q:\exists\ f_1:A_1\Rightarrow B_1.(\forall\ x:A_1.P(y,f_2\ y,f_1)[Fst(r\ f_1)/f_2])$,则$(Fst\ q,Fst(r(Fst\ q)),(Snd\ q,Snd(r(Fst\ q))))$是$\exists\ f_1:A_1\Rightarrow B_1.\exists\ f_2:A_2\Rightarrow B_2(\forall\ x:A_1.P(x,f_1\ x,f_2)\wedge\forall\ y:A_2.Q(y,f_2\ y,f_1))$的一个证明.

引理 2.1. 设 B 是类型,其中不含变元 x,且 $r:\forall\ x:A.\exists\ y:B.P(x,y)$,则有 ①$\lambda x:A.Fst(r\ x):A\Rightarrow B$; ②$\lambda x:A.Snd(r\ x):\forall\ x:A.P(x,Fst(r\ x))$.

证明:①

$$\frac{r:\forall\ x:A.\exists\ y:B.P(x,y)\quad [a:A]^1}{r\ a:\exists\ y:B.P(a,y)}\ (\forall\ E)$$

$$\frac{r\ a:\exists\ y:B.P(a,y)}{Fst(r\ a):B}\ (\exists\ E)$$

$$(\Rightarrow I)_1$$

$\lambda x:A.\ Fst(r\ x):A\Rightarrow B$

注:由于 B 是独立于 A 之元素的类型,所以可使用蕴含的消去规则.

②

$$\dfrac{\dfrac{\dfrac{r:\forall\ x:A.\exists\ y:B.P(x,y)\quad [a:A]^1}{r\ a:\exists\ y:B.P(a,y)}\ (\exists\ E)}{Snd(r\ a):P(a,Fst(r\ a))}\ (\forall\ I)_1}{\lambda x:A.Snd(r\ x):\forall\ x:A.P(x,Fst(r\ x))}$$

证毕.

定理 2.2. 若$\forall\ x:A.\exists\ y:B.P(x,y)$可证,则$\exists\ f:A\Rightarrow B.\forall\ x:A.P(x,f\ x)$亦可证.

证明:设$r:\forall\ x:A.\exists\ y:B.P(x,y)$,则根据引理 2.1 有 $\lambda x:A.Fst(r\ x):A\Rightarrow B$ (1)

$\lambda x:A.Snd(r\ x):\forall\ x:A.P(x,Fst(r\ x))$ (2)

令 $g=_{df}\lambda x:A.Fst(r\ x)$,则根据(2)有 $\lambda x:A.Snd(r\ x):\forall\ x:A.P(x,g\ x)$ (3)

令 $P(x,g\ x)=_{df}Q(x,g)$,则根据(3)有 $\lambda x:A.Snd(r\ x):\forall\ x:A.Q(x,g)$

另外,根据(1)有 $g:A\Rightarrow B$

因为 $\dfrac{g:A\Rightarrow B\quad \lambda x:A.Snd(r\ x):\forall\ x:A.Q(x,g)}{(g,\lambda x:A.Snd(r\ x)):\exists\ f:A\Rightarrow B.\forall\ x:A.Q(x,f)}(\exists\ I)$

所以$(g,\lambda x:A.Snd(r\ x)):\exists\ f:A\Rightarrow B.\forall\ x:A.P(x,f\ x)$　　证毕.

3　程序构造过程

本节讨论根据给定的类规约,开发类的过程.为了便于讨论问题,且不失一般性,在下面的讨论中假定类规约中只有 2 个关于操作的描述.即类规约 *cspec* 如下:

$(\exists\ C:U_0)$

$(\exists\ (v_1::B_1):U_0)$

...

$(\exists\ (v_m::B_m:U_0)$

$(\exists\ f_1:\alpha_1\Rightarrow\beta_1)$

$(\exists\ f_2:\alpha_2\Rightarrow\beta_2)$

$(C=_{u0}v_1::B_1\wedge\ldots v_m::B_m\wedge\{f_1\}\wedge\{f_2\})\wedge B_1=_{u0}T_1\wedge\ldots B_m=_{u0}T_m\wedge$

$\forall\ x:\alpha_1.P(x,f_1\ x,f_2)\wedge\forall\ y:\alpha_2.Q(y,f_2\ y,f_1))$

(1) 用 *f*1 和 *f*2 分别替换 *cspec* 中关于 C 的元素的第 $m+1$ 个分量和第 $m+2$ 个分量的描述,所得到的类规约记为 *cspec*1.

由于 *cspec* 中出现形式为 $C=_{u0}v_1::B_1\wedge\ldots v_m::B_m\wedge\{f_1\}\wedge\{f_2\}$ 的等式,因而类型 C 应该是一个合取形式的公式.设 $obj:C$,则 *obj* 的形式是$(\sigma_1,\ldots\sigma_1,f_1,f_2)$. *obj* 的第 $m+1$ 分量(即 *snd snd*...*snd fst obj*,其中有 m 个 *snd*)表示 f_1,第 $m+2$ 个分量表示 f_2.

性质 3.1. *cspec* $=_{u1}$ *cspec*1. 换句话说,若 $a:cspec$,则 $a:cspec1$,反之,若 $b:cspec1$,则 $b:cspec$.

(2) 确定实例变量类型,并初步确定 C 类型.

根据 *cspec*1 中的 $B_i=_{u0}T_i$ 等式,确定 $v_i::B_i$ 为 $v_i::T_i$,B_i 为 T_i.

初步确定 C 为类型 $v_1::T_1\wedge\ldots\wedge v_m::T_m\wedge TRIV\wedge TRIV$,并记它为 *Class*.

利用以上结果,导出如下类型.

$\exists\ f_1:\alpha_1\Rightarrow\beta_1.\exists\ f_2:\alpha_2\Rightarrow\beta_2.((\forall\ x:\alpha_1.P1(x,f_1\ x,f_2)\wedge\forall\ y:\alpha_2.Q1(y,f_2\ y,f_1))[Class/C][v_1::T_1/v_1::B_1]\ldots[v_m::T_m/v_m::B_m][T_1/B_1]\ldots[T_m/B_m])$

其中$P1=_{df}P[f_1/C$ 的元素的第 $m+1$ 个分量$][f_2/C$ 的元素的第 $m+2$ 个分量$]$

$Q1=_{df}Q[f_1/C$ 的元素的第 $m+1$ 个分量$][f_2/C$ 的元素的第 $m+2$ 个分量$]$

所导出的类型称为 $cspec2$，并简记为

$\exists\ f_1:\alpha_1\Rightarrow\beta_1.\exists\ f_2:\alpha_2\Rightarrow\beta_2.(\forall\ x:\alpha_1.P2(x,f_1\ x,f_2)\wedge\forall\ y:\alpha_2.Q2(y,f_2\ y,f_1))$

(3) 构造 f_1 和 f_2

首先，证明$\forall\ f_1:\alpha_1\Rightarrow\beta_1.\exists\ f_2:\alpha_2\Rightarrow\beta_2.\forall\ y:\alpha_2.Q2(y,f_2\ y,f_1)$

假设 $F_1:\alpha_1\Rightarrow\beta_1$，利用定理 2.2 证明

$\exists\ f_2:\alpha_2\Rightarrow\beta_2.\forall\ y:\alpha_2.Q2(y,f_2\ y,F_1)$，且记其元素为$(G_1,q)$

则根据全称量词的引入规则有

$\lambda f_1:\alpha_1\Rightarrow\beta_1.(G_1,q):\forall\ f_1:\alpha_1\Rightarrow\beta_1.\exists\ f_2:\alpha_2\Rightarrow\beta_2.\forall\ y:\alpha_2.Q2(y,f_2\ y,f_1)$

其次，利用定理 2.2 证明

$\exists\ f_1:\alpha_1\Rightarrow\beta_1.(\forall\ x:\alpha_1.P2(y,f_2\ y,f_1)[G_1\ f_1/f_2])$，且记其元素为$(F_1,p)$.

根据推论 2.1 有

性质 3.2. $(q\ F_1,p):(\forall\ x:\alpha_1.P2(x,f_1\ x,f_2)\wedge\forall\ y:\alpha_2.Q2(y,f_2\ y,f_1))[F_1/f_1][G_1F_1/f_2]$

性质 3.3. 将 $cspec2$ 中的 $Class$ 替换为 $v_1::T_1\wedge\ldots\wedge v_m::T_m\wedge\{f_1\}\wedge\{f_2\}$，所得到的类型记为 $cspec3$. 若$(F_1,G_1\ F_1,r):cspec2$，则$(F_1,G_1\ F_1,r):cspec3$.

证明：当不涉及元素的性质时，$\{f_1\}$就是一种 $TRIV$ 类型，$\{f_2\}$亦是.

根据 $cspec2$ 的定义，$cspec2$ 中不出现关于 $Class$ 的元素的第 $m+1$ 和 $m+2$ 个分量的描述. 因此，若$(F_1,G_1\ F_1,r):cspec2$，则$(F_1,G_1\ F_1,r):cspec3$. 证毕.

定理 3.1. 设 $\lambda f_1:\alpha_1\Rightarrow\beta_1.(G_1,q):\forall\ f_1:\alpha_1\Rightarrow\beta_1.\exists\ f_2:\alpha_2\Rightarrow\beta_2.\forall\ y:\alpha_2.Q2(y,f_2\ y,f_1)$，$(F_1,p):\exists\ f_1:\alpha_1\Rightarrow\beta_1.(\forall\ x:\alpha_1.P2(y,f_2\ y,f_1)[G_1\ f_1/f_2])$，$C=_{df}v_1::T_1\wedge\ldots\wedge v_m::T_m\wedge\{F_1\}\wedge\{G_1\ F_1\}$，$v_1::B_1=_{df}v_1::T_1,\ldots,v_m::B_m=_{df}v_m::T_m$，$B_1=_{df}T_1,\ldots,B_m=_{df}T_m$，$f_1=_{df}F_1$，$f_2=_{df}G_1\ F_1$，则 $cspec$ 可证.

证明：根据存在量词的引入规则，要证明本定理，只要证明如下公式即可.

$(C=_{u0}v_1::B_1\wedge\ldots v_m::B_m\wedge\{f_1\}\wedge\{f_2\})\wedge B_1=_{u0}T_1\wedge\ldots B_m=_{u0}T_m\wedge$

$\forall\ x:\alpha_1.P(x,f_1\ x,f_2)\wedge\forall\ y:\alpha_2.Q(y,f_2\ y,f_1))[T_1\wedge\ldots\wedge T_m\wedge\{F_1\}\wedge\{G_1\ F_1\}/C][v_1::T_1/v_1::B_1]\ldots[v_m::T_m/v_m::B_m][T_1/B_1]\ldots[T_m/B_m][F_1/f_1][G_1\ F_1/f_2]$ (1)

(1)式的证明可分解为如下合取子式的证明：

$v_1::T_1\wedge\ldots v_m::T_m\wedge\{F_1\}\wedge\{G_1\ F_1\}=_{u0}v_1::T_1\wedge\ldots v_m::T_m\wedge\{F_1\}\wedge\{G_1\ F_1\}$ (2)

$T_1=_{u0}T_1$ (3)

……

$T_m=_{u0}T_m$ $(m+1)$

$(\forall\ x:\alpha_1.P2(x,f_1\ x,f_2)\wedge\forall\ y:\alpha_2.Q2(y,f_2\ y,f_1))[F_1/f_1][G_1\ F_1/f_2]$ $(m+2)$

根据等式公式的引入规则 $\dfrac{a:A}{r(a):a=_A a}$

可得到合取子式(1)～$(m+1)$的证明.

根据性质3.2,$(q\ F_1, p)$是合取子式$(m+2)$的证明.　证毕.

定理3.1 构造了满足 *cspec* 的一个类,即 $C =_{df} v_1::T_1 \wedge \ldots \wedge v_m::T_m \wedge \{F_1\} \wedge \{G_1\ F_1\}$,由此完成类的构造过程.

4 结束语

由于直觉主义逻辑的构造性证明特性,可把构造性类型理论作为程序逻辑,同时可作为研究程序自动化的一种较理想的理论基础.本文探讨了构造性类型理论在面向对象程序自动化中的应用问题,并提出了根据类规约推导类的方法.在程序设计中,根据所采用的程序设计方法,程序单位的划分是有区别的.过程式程序的程序单位是子程序(即函数和过程子程序),函数式程序的程序单位是函数,而面向对象程序的程序单位是数据和一组相关操作的统一体,称为类.类中一组操作的相关性体现在它们之间存在相互引用关系.本文重点讨论了逐个开发类中一组操作的可行性和方法.利用这一方法可把推导面向对象程序的问题简化为推导类中操作的问题,其中对每个操作的推导过程是相对独立的.对于操作的推导,我们可直接采用Martin-Löf类型理论,详细论述见文献[4,8～10],具体实现方法可参考Nuprl系统的设计与实现方法[7],该系统根据操作的规约,通过证明方法构造满足规约的程序,其中程序是以λ表达式表示的.

文献[4,7,9,10]着重讨论了基于Martin-Löf类型理论的函数式程序设计问题,而本文重点讨论了基于Martin-Löf类型理论的面向对象程序设计问题.根据本文的讨论,首先,Martin-Löf类型理论可作为面向对象程序设计的基础,其次,基于TTOOP的面向对象程序推导方法并不增加程序推导的复杂度.

参考文献

1 国家自然科学基金会.计算机科学学术.北京:科学出版社,1994.

2 Manna Z, Waldinger R. A deductive approach to program synthesis. ACM Trans. on Programming Language and Systems, 1980, **2(1)**:90～121.

3 Manna Z, Waldinger R. Fundamentals of deductive program synthesis. IEEE Trans. on Software Engineering, 1992, **18(8)**:674～704.

4 Galmiche D. Program development in constructive type theory. Theoretical Computer Science, 1992, **94**:237～259.

5 Lu Jianguo, Xu Jiafu. Analogical program derivation based on type theory. Theoretical Computer Science, 1993, **113**:259～272.

6 Zhu Mingyuan. Program transformation in constructive type theory. ACM SIGPLAN Notices, 1995, **30(1)**:11～19.

7 Constable R L. Implementing mathematics with the nuprl proof development system. Prentice-Hall, Inc., 1986.

8 全炳哲.基于类型理论的面向对象程序设计的研究[博士论文].吉林大学,1996.

9 Thompson S. Type theory and functional programming. Addison-Wesley Publishing Company Inc., 1991.

10 Nordstrom B, Petersson K, Smith J M. Programming in Martin-Löf's type theory. Oxford: Clarendon Press, 1990.

AN OBJECT-ORIENTED AUTOMATIC PROGRAMMING IN TYPE THEORY

*QUAN Bingzhe *JIN Chunzhao **XUAN Shunji

*(*Department of Computer Science Jilin University Changchun* 130023)

**(*Department of Computer Science Jilin University of Technology Changchun* 130025)

Abstract A constructive type theory may be used as a theoretical foundation to study automatic programming. This paper, using a type theory which supports object-oriented computation, discusses an approach of object-oriented automatic programming. The program unit of an object-oriented program is class, encapsulating data and a group of related operations. The key problem of the program development is to implement these operations, and it is the main concern of this paper.

Key words Constructive type theory, object-oriented programming, automatic programming, software automation.

Class number TP311