循环结构的形式化推导*

李贤贞[1,2],吴茂念[1],杨 静[1]

(1.贵州大学 计算机科学与信息学院,贵州 贵阳 550025;
2.中国科学院国家天文台,北京 100012)

摘 要:介绍了Dijkstra的形式化推导方法的主要思想、步骤及要点。该方法主张程序开发和程序证明同时进行,先确定好描述程序功能的断言,再通过形式化方法推导出正确的程序。选择具有代表性的循环结构的实例进行推导证明,并对循环结构的形式化推导进行阐述说明。

关键词:形式化方法;程序正确性;循环不变式;界函数

中图分类号:TP301 **文献标识码**:A **文章编号**:1674-7720(2014)05-0082-02

Formal derivation method of repetitive construct

Li Xianzhen[1,2], Wu Maonian[1], Yang Jing[1]

(1.College of Computer Science & Information, Guizhou University, Guiyang 550025, China;
2.NAOC, Beijing 100020, China)

Abstract: This document introduces the basic theory of Dijkstra's formal derivation method, which be of the view that the development and demonstration of programs processed simultaneously, confirming the assert of describing the function of the program, and then deduced a correct and proper algorithm formally. And this document is added a representative example of struct to illustrate the theory.

Key words: formal methods; program correctness; loop invariant; boundary function

算法是计算机科学的核心,而算法的正确性是近几年讨论的热点问题,但是效果并不明显。一般情况下,程序的正确性都是针对已经编好的程序,通过测试用例,尽可能地找出程序的漏洞,但这种方法并不能从根本上保证程序的正确性。采用形式化的方法[1]来进行设计程序,是先将需要解决的问题精确描述出来,再根据某种形式化规则进行推理,最终得到正确且结构化的程序。目前存在很多种形式化方法,Dijkstra的最弱前置条件程序推导;英国爱丁堡大学的Burstall和Darlington所研制的ZAP系统;基于公理语义的Z;基于指称语义的VDM;基于抽象机的B方法;江西师范大学提出的PAR(Partition And Recur)方法[2-5]等。

如果能找出一套形式化方法,实现程序的自动化开发和证明,将使得开发周期大大缩短,降低程序开发的成本,也将不再有后期维护的后顾之忧。Dijkstra主张程序开发和程序证明同时进行,属于半自动化的形式化方法[6]。需要人为地找出确定描述程序功能的断言、循环不变式以及t函数。若能提出某种方法实现此过程的自动化,将有望找出自动化的形式化推导。

*基金项目:国家自然科学基金项目(61262029)

1 形式化推导的基本思想

1.1 {Q}S{R}系统

设S是一个程序语句,S的前断言为Q,后断言为R,记法{Q}S{R}表示如果在S执行之前谓词Q为真,那么在S执行之后谓词R也真[7]。

1.2 最弱前置条件wp(S,R)

对于给定的程序S,wp(S,R)是一个状态集合,以该集合中任一状态作为初始状态执行程序S都能保证程序终止且满足后置条件R;反之,能使程序终止,且终止状态满足后置条件R的初始状态必属于wp(S,R)所定义的状态集合。即对程序S来说,wp(S,R)是属于后置条件R的最弱前置条件。

1.3 空语句

“skip”表示空语句,即什么都不执行。

$$wp(skip,R)=R \tag{1}$$

即对于任意的后置条件 R，其空语句下的最弱前置条件也为 R。

1.4 赋值语句

赋值语句的语句形式 $x:=E$，指变量 x 被表达式 E 所替换。

$$wp(\text{“}x:=E\text{”},R)=R_{E\to x} \tag{2}$$

即对于任意的后置条件 R，其赋值语句下的最弱前置条件是将 R 中所有出现的 x 都用 E 来代替。

1.5 分号语句

分号语句的语句形式 $S_1;S_2$，指先激活 S_1，执行结束后再激活 S_2，如式(3)：

$$wp(\text{“}S_1;S_2\text{”},R)=wp(S_1,wp(S_2,R)) \tag{3}$$

即对于任意的后置条件 R，其分号语句下的最弱前置条件为 R 在 S_2 下的最弱前置条件作为 S_1 的后置条件，再在 S_1 下的最弱前置条件。

1.6 选择语句

“IF”表示选择语句。语句形式如下：

if $B_1\to SL_1|B_2\to SL_2|\cdots|B_n\to SL_n|f_i$

其中 $B_1,B_2,\cdots,B_n$ 都是警卫，选择所有警卫为真的其中一个 B_i，执行 SL_i 语句体，然后 IF 终止。

$$\begin{aligned}&wp(IF,R)=(B_1\ \text{or}\ B_2\ \text{or}\cdots B_n)\ \text{and}\\&(B_1\Rightarrow wp(SL_1,R))\ \text{and}\\&(B_2\Rightarrow wp(SL_2,R))\ \text{and}\\&\cdots\ \text{and}\\&(B_n\Rightarrow wp(SL_n,R))\end{aligned} \tag{4}$$

1.7 循环语句

“DO”表示循环语句。语句形式如下：

do $B_1\to SL_1|B_2\to SL_2|\cdots|B_n\to SL_n|$ od

其中 $B_1,B_2,\cdots,B_n$ 都是警卫，如果 B_i 为真，则执行 SL_i 语句体，循环执行，直至所有的警卫为假，则循环终止。

$$\begin{aligned}&H_0(R)=R\ \text{and}\ \text{non}(E\ j:1\leqslant j\leqslant n:B_j)\\&\text{for}\ k>0:H_k(R)=wp(IF,H_{k-1}(R))\ \text{or}\ H_0(R)\\&wp(DO,R)=(E\ k:k\geqslant 0:H_k(R))\end{aligned} \tag{5}$$

1.8 循环结构的基本原理

若

$$(P\ \text{and}\ BB)\Rightarrow wp(IF,P) \tag{6}$$

则

$$(P\ \text{and}\ wp(DO,T))\Rightarrow wp(DO,P\ \text{and}\ \text{non}\ BB) \tag{7}$$

$$BB=(E\ j:1\leqslant j\leqslant n:B_j) \tag{8}$$

其中 P 为循环不变式[8]，即循环执行之前 P 为真，且每次循环重复执行之后还为真。

1.9 t 函数

t 函数是一个整型函数，且需满足以下条件：

$$(P\ \text{and}\ BB)\Rightarrow(t>0) \tag{9}$$

$$(P\ \text{and}\ BB\ \text{and}\ t\leqslant t_0+1)\Rightarrow wp(IF,t\leqslant t_0) \tag{10}$$

即如果 BB 满足 $t>0$，且卫式命令的每次执行都会使得 t 至少减 1，则程序是可终止的。

2 循环结构形式化推导的一般步骤

(1)对于给定的实际问题，经过分析用形式化的方法写出后置条件 R，找出循环不变式 P，以及保证程序终止的函数 t。

(2)由于终止条件时必须满足后置条件 R，即

$$P\ \text{and}\ \text{non}\ BB\Rightarrow R \tag{11}$$

从而找出警卫 BB，即循环结构的条件。

(3)根据循环不变式 P 和后置条件 R 寻找可行的初始化条件。

(4)根据循环结构的基本原理，由

$$(P\ \text{and}\ B_j)\Rightarrow(wp(SL_J,P)\ \text{and}\ wdec(SL_J,t)) \tag{12}$$

得出循环体和 B_j。

3 用形式化推导过程求任意正整数的阶乘

(1)用变量 W 来存最后求得的值。则后置条件

$$R:n>0\ \text{and}\ W=\prod(A\ i:0<i\leqslant n:i) \tag{13}$$

因为程序必须满足所有的正整数，如果不采用循环语句就很难看出 R 是如何得到的。所以需寻求一个循环不变式，最好能比较容易建立，而最终又要有(P and non BB)=>R。选择一个稍弱于 R 的式子，也就是得到终态的一个泛化。而泛化一个式子的典型做法就是用一个变量来代替一个常量，所以用变量 j 来代替常量 n，并加入变量范围，则循环不变式

$$P:0<j\leqslant n\ \text{and}\ W=\prod(A\ i:0<i\leqslant j:i) \tag{14}$$

而 t 函数每次都需单调递减，可设 t 函数：

$$t=n-j \tag{15}$$

(2)由循环不变式 P 和后置条件 R 可得出：

$$(P\ \text{and}\ j=n)\Rightarrow R$$

则

$$BB=(j\neq n) \tag{16}$$

(3)为了验证这个 P 是否有效，首先必须有一个比较易行的方式来开始。由

$$(j=1\ \text{and}\ W=1)\Rightarrow P$$

则初始化为

$j,w=1,1$

(4)由于式(14)、式(15)，则

$$P\ \text{and}\ (j\neq n)\Rightarrow(t\geqslant 0)$$

结合式(16)，可知 t 函数满足了式(9)。

为了保证 t 至少减 1，可以让 j 加 1，那么 W 就要乘以 $(j+1)$，则

$$\begin{aligned}&wp(\text{“}j,W:=j+1,W\times(j+1)\text{”},P)\\&=(0<j+1\leqslant n\ \text{and}\ W\times(j+1))\end{aligned} \tag{17}$$

$$=\prod(A\ i:0<i\leqslant j+1:i)$$

所以

$$P \text{ and } (j \neq n) \Rightarrow wp(\text{“}j, W:=j+1, W\times(j+1)\text{”}, P) \quad (18)$$

则满足循环结构基本原理的前提条件式(6),再由

$$wp(\text{“}j, W:=j+1, W\times(j+1)\text{”}, n-j \leqslant t_0)$$
$$=(n-j-1 \leqslant t_0) \quad (19)$$

则

$$t_{min}=n-j-1$$
$$t_{min} \leqslant t-1 \quad (20)$$

所以

$$wdec(\text{“}j, W:=j+1, W\times(j+1)\text{”}, n-j)$$
$$=(n-j-1 \leqslant n-j-1) \quad (21)$$
$$=T$$

即

$$(P \text{ and } j \neq n) \Rightarrow wp(\text{“}j, W:=j+1, W\times(j+1)\text{”}, P) \quad (22)$$
$$\text{and } wdec(\text{“}j, W:=j+1, W\times(j+1)\text{”}, n-j)$$

从而得出了 BB

$$BB=B_1=(j \neq n) \quad (23)$$

(5)程序段为:

$j, W:=1, 1;$

do

$j \neq n \rightarrow j, W:=j+1, W\times(j+1)$

od

严格按照形式化推导的方式开发得出循环结构,保证了此程序的完全正确性。

本文简要介绍了 Dijkstra 的最弱前置条件程序推导方法,并通过开发并证明任意正整数的阶乘来说明此方法的步骤及其要点。此例子中,需要人为地寻找出后置条件 R、循环不变式 P、以及 t 函数。自动化的方式推导出 R,P 或 t 函数可以作为下一步的研究课题。而自动化生成正确的程序是一个长期性的国际难题,是一项富有创造性和挑战性的活动,值得进一步研究更多的算法,寻找形式化推导的一般规律,尽可能将创造性劳动变为非创造性劳动,使形式化方法走出实验室,给工程程序的开发带来帮助。

参考文献

[1] 唐稚松,林惠民.功能描述导引的程序综合[M].北京:中国学术期刊电子出版社,1983.

[2] 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.

[3] 王昕,袁超伟.一种安全协议的形式化分析方法[J].计算机工程,2010,36(7):82-84.

[4] 杨晨,薛锦云,苏昭.三个经典数学问题的形式化开发[J].计算机与现代化,2010,180(8):1-4.

[5] 王昌晶,薛锦云.算法及其时间复杂度可同步形式化推导的方法[J].计算机应用研究,2008,25(3):681-683.

[6] WYBE D E. A Discipline of programming [M]. America, 1976.

[7] 杨帆,翟岩慧,曲开社,等.基于形式概念分析的词义解释研究[J].计算机科学,2011,38(10):189-191.

[8] 雷富兴,张来顺,石荣刚,等.循环条件的形式化推导在程序验证中的应用 [J]. 计算机工程与设计,2010,31(14):3193-1397.

(收稿日期:2013-12-19)

作者简介:

李贤贞,女,1989 年生,硕士研究生,主要研究方向:计算机软件理论。

吴茂念,男,1975 年生,副教授,主要研究方向:形式化方法。

杨静,女,1965 年生,副教授,主要研究方向:形式化方法。