文章编号:1000-5862(2018)04-0379-05

算法的形式化推导与基于 Isabelle 的自动化验证

齐蕾蕾,杨庆红*,游 颖
(江西师范大学计算机信息工程学院,江西 南昌 330022)

摘要:可信软件的不断发展进一步推动了形式化方法的深入研究. 结合实际应用中的 2 个问题,采用基于递推关系的算法形式化方法,演示了算法的形式化推导过程,并运用 Isabelle 定理证明器结合 Dijkstra 最弱前置谓词方法,对得到的算法程序进行了自动化验证,避免了手工验证过程繁琐和易出错等问题. 研究表明:基于递推关系的算法形式化方法不仅可以提高开发算法的效率,而且通过数学变换保证推导过程的正确性,从而有效保证了算法和程序的正确性.

关键词:形式化方法;Isabelle 定理证明器;自动化验证;形式化推导
中图分类号:TP 311.5 文献标志码:A DOI:10.16357/j.cnki.issn1000-5862.2018.04.10

0 引言

作为解决软件危机的方法之一,形式化方法[1]具有严格的数学语言和精确的逻辑语义,从而保证软件在开发过程中的正确性.

形式化方法主要包括形式化推导和形式化验证 2 个部分[2]. 形式化推导是通过对问题程序规约进行精确变换,最终得到算法程序. 形式化验证是在对软件进行规约的基础上,分析软件是否具有所期望的性质. 形式化方法对保证软件正确性和可靠性具有十分重要的意义. 大量形式化方法的涌现,对程序设计方法学的发展起了积极的推动作用. 根据不同的分类方式,可将形式化方法分为不同的类型:根据描述方式的不同,可将形式化方法分为模型描述型和性质描述型 2 类;根据表达能力的不同,可以将形式化方法分为模型方法、代数方法、进程代数方法、逻辑方法和网络模型方法共 5 类[3].

算法形式化的方法主要包括 2 种类型,一种是用于程序正确性的验证方法,如 Floyd 不变式断言证明方法[4]、Hoare 公理化方法[5]、Dijkstra 最弱前置谓词方法[6]和 Manna 子目标断言方法等;另一种是用于程序的开发方法,如 Z 方法[7]、B 方法[8]和 PAR 方法[9]等. 其中,Z 方法由集合理论作为支撑,运用相关模式结构,但是由于缺乏明确的公理描述,因此不是一种成熟的开发方法. B 方法在 Z 方法的基础上进行改进,加入谓词转换和最弱前置条件等技术,运用统一的符号表示法进行书写. PAR 方法与 PAR 平台是一种通用软件设计方法和程序设计环境,能够支持软件开发全过程,如基于递推关系算法设计、抽象程序设计、数据库应用软件开发、基于构件的软件开发等. 自 1989 年以来,薛锦云等对 PAR 方法和 PAR 平台进行了不断的扩充与发展,最终形成了成熟的规约和算法描述语言 Radl、抽象程序设计语言 Apla,同时结合程序设计方法学和规约变换规则,实现了 Radl 语言算法转换为 Apla 语言抽象程序的生成系统,Apla 语言程序转换成 Java、C++、Delphi 和 VB 等语言可执行程序的生成系统等. 国家总装备部、北京军区和装甲兵学院等多个军事部门已率先将这些成果应用于我国重要国防军事项目的建设中,取得了较大的军事效益和经济效益.

本文采用基于递推技术的算法形式化方法开发算法程序,重点研究了基于 PAR 方法的算法形式化推导方法. 以股票最大收益和基因序列问题为实例,从问题的程序规约出发,通过程序规约变换技术,得到问题的递推关系和循环不变式,并以此为基础形式化推导出问题的算法程序.

为了进一步验证上述推导得到的算法程序的正确性,本文运用定理证明器 Isabelle,采用后推证明结合 Dijkstra 最弱前置谓词方法,对算法程序进行自动化验证.

1 程序规约变换技术

程序规约的精确化,变换的合理性直接影响程

收稿日期:2017-11-03
基金项目:国家自然科学基金(61363013)资助项目.
通信作者:杨庆红(1968-),女,江西南昌人,教授,主要从事软件形式化、智能教育软件的研究. E-mail:yangqh120@163.com

序的正确性验证、形式化推导和程序功能的实现. 程序规约采用量词的形式表示为$(Q_i:r(i):f(i))$,表示"在范围$r(i)$内,对函数$f(i)$应用q运算所得到的量",其中q可以是$\forall$(全称量词),$\exists$(存在量词),$\sum$(求和量词),$\prod$(求积量词),MIN(求最小值量词),MAX(求最大值量词),N(计数量词)等[10]. 本文在程序规约的变换中用到以下量词性质:

1)单一范围:$(Q_i:i=k;f(i))=f(k)$;

2)范围分裂:$(Q_i:r(i):f(i))=(Q_i:r(i)\wedge b(i):f(i))q(Q_i:r(i)\wedge b(i):f(i))$;

3)交叉积性质:$(Q_i,J:r(i)\wedge s(i,J):f(i,J))=(Q_i:r(i):(Q_J:s(i,J):f(i,J)))$;

4)函数结合律:$(Q_i:r(i):f(i)qg(i))=(Q_i:r(i):f(i))q(Q_i:r(i):g(i))$.

2 形式化推导实例

2.1 股票最大收益问题

问题描述:连续n天观察一只给定股票的收盘价格,其中天数计为$i=1,2,\cdots,n$. 对第i天有这个股票当天的收盘价格$p(i)$,假设股票不能当天买卖. 求这只股票在n天内的最大收益.

1)给出问题规约. 用$\max income(n)$表示该只股票的最大收益,即在n天的股票波动行情中,总是在波段最低点买入,在波段最高点抛出. 因计算收益从第2天开始考虑,同时为了将最后一天的情况纳入一般情况考虑,假设$p[n]=-\infty$.

Q:给定股票收盘价格数组$p[0:n-1]$,$n>1$;

R:$\max income(n)=(\sum i,j:1\leqslant i<j\leqslant n-1\wedge p[i-1]>p[i]\wedge p[j]>p[i+1]\wedge(\forall k:i\leqslant k<j:p[k]\leqslant p[k+1]):p[j]-p[i])$.

在R中,根据对i、j的约束条件知,该问题实际是在p数组中寻找若干升序段$p[i..j]$,$p[i]$为升序段的最小值,$p[j]$为升序段的最大值,最大收益即为求各升序段的收益之和. 由于升序段$p[i..j]$的收益$p[j]-p[i]$等价于$p[i+1]-p[i]+p[i+2]-p[i+1]+\cdots+p[j]-p[j-1]$.

因此,考虑各升序段的收益之和,可将后置谓词R变换成R:$\max income(n)=(\sum i:1\leqslant i\leqslant n-1\wedge p[i-1]<p[i]:p[j]-p[i-1])$.

2)分划原问题并寻找递推关系. 考虑一般情况下$\max income(m)=(\sum i:1\leqslant i\leqslant m-1\wedge p[i-1]<p[i]:p[i]-p[i-1])=(\sum i:1\leqslant i\leqslant m-2\wedge p[i-1]<p[i]:p[j]-p[i-1])+(\sum i:i=m-1\wedge p[i-1]<p[i]:p[i]-p[i-1])$ [范围分裂] $=\max income(m-1)+(\sum i:i=m-1\wedge p[m-2]<p[m-1]:p[j]-p[i-1])$.

(i)当$p[m-2]<p[m-1]$时,$\max income(m)=\max income(m-1)+(\sum i:i=m-1:p[i]-p[i-1])$ [单一范围] $=\max income(m-1)+p[m-1]-p[m-2]$.

(ii)当$p[m-2]\geqslant p[m-1]$时,$\max income(m)=\max income(m-1)+(\sum i:\text{false}:p[i]-p[i-1])$ [恒等元性质] $=\max income(m-1)+0=\max income(m-1)$.

于是得到$\max income(m)$的递推关系如下:

$$\max income(m)=\begin{cases}\max income(m-1)+(p[m-1]-p[m-2]), & p[m-2]<p[m-1],\\ \max income(m-1), & p[m-2]<p[m-1].\end{cases}$$

3)开发循环不变式. 用变量s始终表示$\max income(m-1)$的值,可以得到循环不变式为

ρ:$s=\max income(m-1)\wedge(1<m\leqslant n+1)$.

4)设计算法程序.

```
begin
m:=2;s:=0;
do m≤n→
  if p[m-2]<p[m-1]→s:=s+
    p[m-1]-p[m-2];
  ▯ p[m-2]≥p[m-1]→skip;
  fi;
  m:=m+1;
od;
end.
```

2.2 基因序列问题

问题描述:有一串已知的基因序列,如"ACGTASFJKACGTACGTACGTACGTAKDAC …"其中ACGT是一组标志基因,计算基因序列中最长连续标志基因组的个数,并记录起始和终止位置.

用特殊字符'O'替换标志基因组ACGT,因此基因序列变换为"OASFJKOOOOAKDAC…",即求基因序列中连续为'O'的子序列的最大长度.

1)给出问题的规约:Q:$n\geqslant 1$;R:$\max genome(n)=(\text{MAX } i,j:1\leqslant i\leqslant j\leqslant n\wedge(\forall k:i\leqslant k\leqslant j:a[k]='O'):j-i+1))$

2)分划原问题并寻找递推关系. 令$marker(i,j)=(\forall k:i\leqslant k\leqslant j:a[k]='O')$表示在$a[i:j]$中是连续为'O'的基因序列.

$marker(i,j)=(\forall k:i\leqslant k\leqslant j:a[k]='O')=(\forall k:i\leqslant k\leqslant j-1:a[k]='O')\wedge(\forall k:k=j:a[k]='O')$ [范围分裂] $=(\forall k:i\leqslant k\leqslant j-1:a[k]='O')\wedge(a[j]='O')$ [单一范围] $=marker(i,j-1)\wedge(a[j]='O')$.

考虑一般情况下,$\max genome(m)=(\text{MAX } i,j:$

$1\leqslant i\leqslant j\leqslant m\wedge marker(i,j):j-i+1)=((\mathrm{MAX}\ j:1\leqslant j\leqslant m:(\mathrm{MAX}\ i:1\leqslant i\leqslant j\wedge marker(i,j):j-i+1))$ [交叉积性质]. 令 $\max(j)=(\mathrm{MAX}\ i:1\leqslant i\leqslant j\wedge marker(i,j):j-i+1)$ 表示以 $a[j]$ 为最右端的连续为 'O' 的最长基因序列的长度. $\max(j)=(\mathrm{MAX}\ i:1\leqslant i\leqslant j\wedge marker(i,j):j-i+1)=(\mathrm{MAX}\ i:1\leqslant i\leqslant j\wedge marker(i,j-1)\wedge(a[j]='\mathrm{O}'):j-i+1)$.

(i) 当 $a[j]\neq$ 'O' 时,$marker(i,j)=marker(i,j-1)\wedge(a[j]='\mathrm{O}')=marker(i,j-1)\wedge\mathrm{false}=\mathrm{false}$;$\max(j)=(\mathrm{MAX}\ i:\mathrm{false}:j-i+1)$ [恒等元性质,因为长度不能为负数,所以最小值为 0] $=0$.

(ii) 当 $a[j]=$ 'O' 时,$marker(i,j)=marker(i,j-1)\wedge(a[j]='\mathrm{O}')=marker(i,j-1)\wedge\mathrm{true}=marker(i,j-1)$,$\max(j)=(\mathrm{MAX}\ i:1\leqslant i\leqslant j\wedge marker(i,j):j-i+1)=\max((\mathrm{MAX}\ i:1\leqslant i\leqslant j-1\wedge marker(i,j):j-i+1),(\mathrm{MAX}\ i:i=j\wedge marker(i,j):j-i+1))$ [范围分裂] $=\max((\mathrm{MAX}\ i:1\leqslant i\leqslant j-1\wedge marker(i,j):j-i+1),(\mathrm{MAX}\ i:i=j\wedge marker(j,j):j-i+1))=\max((\mathrm{MAX}\ i:1\leqslant i\leqslant j-1\wedge marker(i,j):j-i+1),(\mathrm{MAX}\ i:i=j\wedge true:j-i+1)=\max((\mathrm{MAX}\ i:1\leqslant i\leqslant j-1\wedge marker(i,j):j-i+1),1)$ [单一范围] $=\max((\mathrm{MAX}\ i:1\leqslant i\leqslant j-1\wedge marker(i,j-1):j-i+1),1)=\max((\mathrm{MAX}\ i:1\leqslant i\leqslant j-1\wedge marker(i,j-1):j-1-i+1+1),1)=\max((\mathrm{MAX}\ i:1\leqslant i\leqslant j-1\wedge marker(i,j-1):j-1-i+1),0)+1$ [分配率] $=\max(\max(j-1),0)+1=\max(j-1)+1$.

于是得到 $\max(j)$ 的递推关系如下:

$$\max(j)=\begin{cases}\max(j-1)+1, & a[j]='\mathrm{O}',\\ 0, & a[j]\neq'\mathrm{O}'.\end{cases}$$

$\max genome(m)=(\mathrm{MAX}\ j:1\leqslant j\leqslant m:\max(j))=\max((\mathrm{MAX}\ j:1\leqslant j\leqslant m-1:\max(j)),\max(m))$ [范围分裂,单点范围] $=\max(\max genome(m-1),\max(m))$.

根据上述推导变换得到以下递推关系式:

$$\max(j)=\begin{cases}\max(j-1)+1, & a[j]='\mathrm{O}',\\ 0, & a[j]\neq'\mathrm{O}'.\end{cases}$$

$\max genome(m)=\max(\max genome(m-1),\max(m))$.

3) 开发循环不变式. 在算法程序中用变量 mg 始终记录 $\max genome(m-1)$,用变量 m_j 始终记录 $\max(m-1)$,则循环不变式为 $\rho:mg=\max genome(m-1)\wedge m_j=\max(m-1)\wedge 1\leqslant m\leqslant n+1$.

4) 设计算法程序:

```
begin
m:=1;mg:=0;mj:=0;left:=0;right:=0;
do m≤n→if(a[m]='O')→mj:=mj+1;
        ▯(a[m]≠'O')→mj:=0;
     fi;
     mg:=max(mg,mj);
     if(mg=mj)→right:=m;
               left:=right-mg+1;
      ▯(mg≠mj)→skip;
     fi;
     m:=m+1;
od;
end.
```

left 表示连续为特殊字符 'O' 的最长子序列的起始位置,right 表示连续为特殊字符 'O' 的最长子序列的终止位置.

3　基于 Isabelle 的自动化验证

3.1　Isabelle 概述

Isabelle 是基于元逻辑开发的定理证明辅助工具,由英国剑桥大学 L. C. Paulson 教授和德国慕尼黑技术大学 T. Nipkow 教授于 1986 年联合开发的基于高阶逻辑的交互式定理证明器.

Isabelle 系统作为一个通用的定理证明辅助工具,一方面,它可以作为快速原型推理系统的通用框架;另一方面,它支持 1 阶逻辑、公理集合理论、高阶逻辑和构造逻辑等多种对象逻辑,而绝大多数定理证明器只支持一种逻辑,如 PVS 和 COQ 仅支持高阶逻辑[11].

Isabelle 传统应用领域为数学定理证明. 目前,Isabelle在数学定理[12]、协议[13]、一阶相对论[14] 以及 SAT 求解器[15] 等方面的自动化验证具有广泛的应用,但用于算法程序正确性自动化验证方面的研究较少. 本文将结合 Dijsktra 的程序验证思想,运用 Isabelle 的证明语言并合理调用 Isabelle 中的策略和方法,对上述推导的 2 个算法程序进行自动化验证,探索 Isabelle 在算法程序正确性自动化验证方面的有效途径.

3.2　Dijkstra 最弱前置谓词

Dijkstra 最弱前置谓词方法以一个小型程序语言为基础,主要包含空语句、赋值语句、复合语句、选择语句、循环语句和过程调用语句. Dijkstra 设计了一种谓词变换函数 $WP(Q,R)$,使 Q 的执行能够终止并且终止后条件 R 成立的最弱前置条件. 同时给出了计算各种语言成分 WP 函数的方法,从而形成了一个较完整的证明系统,用于对程序的完全正确性进行证明.

Dijkstra 最弱前置谓词方法中循环语句的一般形式为

```
do C1→S1;
 ▯ C2→S2;
```

…

$\mathbf{I}\ C_n \rightarrow S_n$;

od,

其中 $C_1, C_2, \cdots, C_n$ 均为布尔表达式,$S_1, S_2, \cdots, S_n$ 为任意语句,$C_i \rightarrow S_i$ 称为条件子句.

设 $Guard = C_1 \vee C_2 \vee \cdots \vee C_n$. 要证明 $\{Q\}$ do $\{R\}$ 的正确性,关键在于找到 do 语句的循环不变式 ρ,使以下条件成立:(i) $Q ==> \rho$(保证开始迭代之前 ρ 为真),(ii) $\rho \wedge Ci ==> WP("S", \rho)\ 1 \leqslant i \leqslant n$,(iii) $\rho \wedge \neg\ Guard ==> R$(迭代完成后 R 为真).

4 自动化验证实例

4.1 股票最大收益问题的自动化验证

1)创建理论文件 maxincome. thy 使用 imports 命令选择 Main 理论作为父理论.

```
theory maxincome
importsMain
```

2) Isabelle 含有存放数组的 list 类型,因此无需另外定义.

3)描述求解问题的形式规约.

Q:给定股票收盘价格数组 $p\ [0:n-1]$,$n>1$;

R:max*income*$(n) = (\sum i{:}1 \leqslant i \leqslant n-1 \wedge p\ [i-1] < p\ [i]{:}p\ [i] - p\ [i-1])$.

4)开发循环不变式 ρ. 根据本文对股票最大收益问题进行形式化推导的过程,可以得出循环不变式 $\rho{:}s = \text{max}income(m-1) \wedge (1 < m \leqslant n+1)$.

5)根据需要定义相关函数 max*income*. 在 Isabelle 中用 fun 命令定义 max*income* 函数.

```
fun maxincome::"nat⇒int list⇒int"
where
  "maxincome 0 p = (0)" |
  "maxincome n p = (∑i∈{1..(n-1)}.
(if((p! i) > (p! i-1)) then((p! i) - (p! i-1))
else 0))".
```

6)验证算法程序中语句的正确性. 证明循环不变式在每次循环执行前后均为真.

(i)

```
lemma wp1:"(n::nat) >1 ∧ m = 2 ∧ s = 0⇒
s = (maxincome(m-1::nat)p) ∧ m >1 ∧ m≤n"
  apply auto
  done.
```

(ii)

```
lemma wp2:"s = (maxincome(m - 1::nat)
p) ∧ m > 1 ∧ m ≤ n ∧ m < n⇒(maxincome m p) =
(maxincome m p) ∧Suc m >1 ∧Suc m≤n"
  apply auto
  done.
```

(iii)

```
lemma wp3:"s = (maxincome(m - 1::nat)
p) ∧ m >1 ∧ m≤n ∧ m > n⇒s = (maxincome n p)"
  apply auto
  done.
```

验证程序语句正确性的引理,都可以用 "apply auto" 进行求解,结果都是 "No subgoals!",命令 done 表示结束当前证明.

4.2 基因序列问题的自动化验证

当应用 Isabelle 系统进行程序的自动化验证时,前 4 步基本相似,这里就不多做赘述. 但应注意文件名应与定理名相同.

创建好文件并完成前 4 步证明过程之后,即可进行第 5 步证明过程.

根据需要定义相关函数 *marker*、max*j* 和 max*genome*.

```
fun marker::"nat⇒nat⇒int list⇒bool"
where
  "marker i j a = (if(∀m∈{i..j}.((a! m) =
0)) then true else false)"
fun maxj::"nat⇒int list⇒int "
where
  "maxj  0 a = (0)" |
  "maxj  j a = (∑p∈{0..j}.(if(marker p j
a) then 1 else 0))"
fun maxgenome::"nat⇒int list⇒int"
where
  "maxgenome 0 a = (0)" |
  "maxgenome k a = (max(maxj k a)(maxge-
nome(k-(1::nat))a))".
```

验证算法程序中语句的正确性. 证明循环不变式在每次循环执行前后均为真.

(i)

```
lemma wp1:"(n::nat)≥1 ∧ mj =0 ∧ mg =0 ∧
p = 1⇒mg = (maxgenome(p - 1::nat)a) ∧ mj =
(maxj(p-1::nat)a) ∧p≥1 ∧p≤n+1"
  apply auto
  done.
```

(ii)

```
lemma wp2:"mg = (maxgenome p a) ∧ mj =
(maxj p a) ∧p≥1 ∧p≤n ∧p<n⇒(max(maxj(Suc
p)a)mg) = (maxgenome(Suc p)a) ∧ Suc p≥1 ∧Suc
p≤n"
  apply auto
  done.
```

(iii)

```
lemma wp3:"mj = (maxj(p-1::nat)a) ∧
mg = (maxgenome(p-1::nat)a) ∧p≥1 ∧p≤n ∧
p>n⇒mg = (maxgenome n a)"
  apply auto
  done.
```

验证程序语句正确性的引理,都可以用 "apply

auto"进行求解,结果都是"No subgoals!",命令 done 表示结束当前证明.

5　结束语

本文使用精确的数学符号描述问题的程序规约,使用程序规约变换技术对程序规约进行等价变换,从而得到递推关系式和循环不变式,进而得到算法程序,在保证算法程序正确性的同时,清晰地展示了算法的设计思路与过程,提高了算法的可靠性. 已有研究表明形式化推导方法可应用于 3 类经典算法,即递归与分治策略、动态规划和贪心算法.

使用 Isabelle 定理证明器对算法程序进行自动化验证,不仅弥补了传统手工证明过程繁琐和易出错等不足,而且也提高了验证算法正确性的效率,使得验证过程更加直观严谨. 目前已经运用 Isabelle 定理证明器验证了许多算法程序,但是由于软件的复杂,使得部分算法的自动化验证过程仍然需要少量的人工干预. 笔者正在致力于图的有关算法的正确性证明,希望通过对各种算法程序的不断研究与证明,能够形成系统的算法自动化证明库,以供其他研究人员进行学习与研究.

6　参考文献

[1] 彭成,王盼卿. 软件形式化开发方法的选择策略研究[J]. 电子设计工程,2014,22(15):30-36.

[2] 游珍,薛锦云. 基于 Isabelle 定理证明器算法程序的形式化验证[J]. 计算机工程与科学,2009,31(10):85-89.

[3] 包丽梅,张玉春,张世铮. 关于形式化方法与软件可靠性的研究[J]. 内蒙古民族大学学报:自然科学版,2010,25(2):166-167.

[4] 苏竟秀,龙陈锋. Floyd 算法与 RAD 算法性能分析[J]. 计算机应用与软件,2015,32(2):116-119.

[5] 雷富兴,张来顺. 基于 Hoare 逻辑的过程调用的形式化方法[J]. 计算机工程与设计,2011,32(1):197-201.

[6] 杨黄磊,薛锦云. 一类单元赋值语句型循环不变式的开发方法研究[J]. 江西师范大学学报:自然科学版,2014,38(4):378-382.

[7] 刘峰涛,贺国光. 基于 L_Z 方法的宏观交通运输系统复杂性测度[J]. 哈尔滨工业大学学报,2008,40(12):2058-2061.

[8] 丁湘陵,王志刚. 基于 B 方法的体系结构描述语言的形式化研究[J]. 计算机工程与科学,2013,35(1):100-106.

[9] 朱威. 基于 PAR 方法的分形算法开发研究[D]. 河南:郑州大学信息工程大学,2014.

[10] 苏昭. 形式化 PAR 方法及其算法程序规约精化机理[J]. 江西科技学院学报,2014,9(3):53-57.

[11] 李婉璐. 软件工程中的形式化方法研究综述[J]. 数字技术与应用,2015(10):108-109.

[12] Lawrence C Paulson. A mechanised proof of Godel´s incompleteness theorems using nominal Isabelle[J]. J Autom Reasoning,2015(55):1-37.

[13] Li Yongjian,Pang Jun. Formalizing provable anonymity in Isabelle/HOL[J]. Formal Aspects of Computing,2015(27):255-282.

[14] Mike Stannett. Using Isabelle/HOL to verify first-order relativity theory[J]. J Autom Reasoning,2014(52):361-378.

[15] Filip Maric. Formal verification of a modern SAT solver by shallow embedding into Isabelle/HOL[J]. Theoretical Computer Science 2010,411(50):4333-4356.

The Formal Derivation of Algorithm and Automatic Verification Based on Isabelle

QI Leilei,YANG Qinghong*,YOU Ying

(College of Computer Information Engineering,Jiangxi Normal University,Nanchang Jiangxi 330022,China)

Abstract: The continuous development of trusted software has further promoted the in-depth study of formal methods. Two formal problems of practical application are introduced,and the formal deduction process of the algorithm is demonstrated by the formal method based on recursive relation. The Isabelle theorem prover is combined with the weakest predicate method of Dijkstra to validate the algorithm automatically,which avoid the problem that manual verification process is tedious and error prone. The research shows that the formal method based on recursive relation can not only improve the efficiency of the algorithm,but also ensure the correctness of the derivation process by mathematical transformation. So this process effectively guarantees the correctness of the algorithm.

Key words: formal methods;Isabelle theorem prover;automatic verification;formal derivation

(责任编辑:冉小晓)