对 Wirth 一个不变式的修正

邹姝稚

(扬州大学工学院计算机科学与工程系,江苏 扬州 225009)

摘要:Wirth 在《算法 + 数据结构 = 程序》一书中关于‘对半检索’程序中给出了一个不变式,但该不变式是不完善的.本文对此予以了修正,并通过一系列定理及其证明进行了完整的论证.在论证过程中,也揭示出程序作为一种对象,其整体的一些性质.程序的这些特性,在一般测试时未必能被认识.因而程序中的‘不变式’及对‘不变式’的论证,也应是程序中最有价值的资料.

关键词:对半检索;不变式;程序正确性

中图分类号:TP301 文献标识码:A 文章编号:1000-1980(2000)06-0111-04

1 问题的提出

N.Wirth 在文献[1]中给出了一个如下的‘对半检索’程序:

```
Var  a :array[1… N] of T ; {N > 0}
⋮
i := 1 ; j := N ;
repeat
        k := (i + j) div 2 ;
  if x > a[k] then i := k + 1 else j := k - 1 ;
until (a[k] = x) ∨ (i > j) ;
```

其中数组 a 的元素已被排序.同时,Wirth 还给出了以下断言:

在 repeat 语句入口处相应的不变式是

$$\begin{cases} a[h] < x & h = 1,\cdots,i-1 \\ a[h] \geqslant x & h = j+1,\cdots,N \end{cases}$$

因此,若由于 $i > j$ 而终止程序,这就意味着对 $1 \leqslant h \leqslant N$,不存在 $a[h] = x$.

显然,循环不变式应该蕴涵程序的终止结论.但 Wirth 给出的不变式不能蕴涵终止结论,因为当 $h \in \{j+1,\cdots,N\}$ 时,可能含有 $a[h] = x$.这样,当程序终止时,不一定能够保证 $1 \leqslant h \leqslant N \supset a[h] \neq x$.因而这一不变式是不准确的,应改作:

$$\begin{cases} a[h] < x & \text{对 } h = 1,\cdots,i-1 \\ a[h] > x & \text{对 } h = j+1,\cdots,N \end{cases}$$

现对此进行论证.

2 论 证

2.1 框图

首先给出程序框图 1,并标明不变式及终止结论的断点 P,Q.此外,为便于论证,在框图中增加了三个断点 A,B 及 C.

收稿日期:1999-08-20

作者简介:邹姝稚(1964—),女,江苏扬州人,讲师,主要从事形式化与程序设计方法学研究.

2.2 定理

定理 1 在断点 A 处,总有 $i \leqslant j$.

证明 对循环次数 l 施行归纳穷举[2].

奠基 $l = 0$ 时,因为 $i := 1, j := N \{N > 0\}$,故 $i \leqslant j$ 成立.

归纳 $l = l_0 + 1$ 时,由于第 $l_0 + 1$ 轮循环是$(a[k] = x) \vee (i > j)$不成立而进入 A 的,故 $i \leqslant j$ 仍成立.

定理 2 在断点 A 处,以下断言只有一个成立.

$$i = k = j \tag{1}$$

$$i = k < j \tag{2}$$

$$i < k < j \tag{3}$$

证明 由定理 1,A 处总有$i \leqslant j$,这样变元 i, j 之间只有三种可能:$i = j, i + 1 = j, i + 2 \leqslant j$.经赋值 $k := (i + j)\ \mathrm{div}\ 2$ 后,分别得到式(1)、式(2)及式(3).

推论 3 在断点 A 处,总有 $i \leqslant k \leqslant j$.

定理 4 在断点 B 处,以下断言只有一个成立:

$$i > j \wedge i - 1 = j = k \wedge x > a[k] \tag{4}$$

$$i \leqslant j \wedge i = k + 1 \wedge x > a[k] \tag{5}$$

证明 如果在 A 点式(1)成立,经过测试 $x > a[k]$及赋值 $i := k + 1$,在 B 点式(4)成立.如果在 A 点式(2)或式(3)成立,仍经测试、赋值,因而式(5)成立.

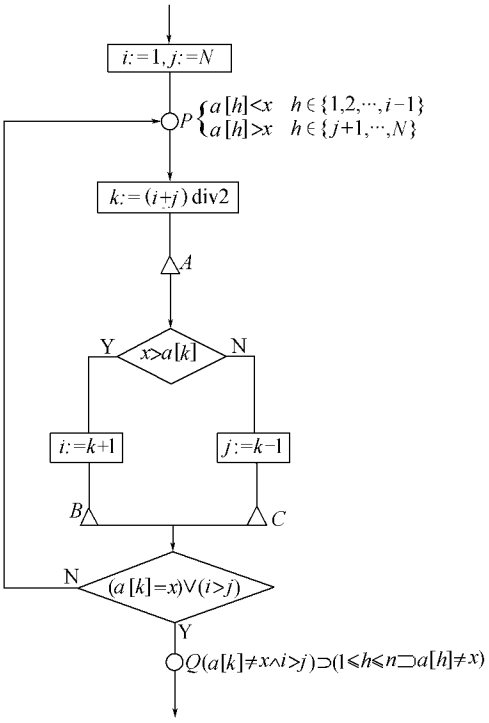

图 1 程序框图

Fig.1 Block diagram of the program

定理 5 在断点 C 处,以下断言只有一个成立:

$$i > j \wedge i = j + 1 = k \wedge x \leqslant a[k] \tag{6}$$

$$i \leqslant j \wedge j + 1 = k \wedge x \leqslant a[k] \tag{7}$$

证明 如果在 A 点式(1)或式(2)成立,因为测试 $x > a[k]$不成立,通过赋值 $j := k - 1$ 后,在 C 点式(6)成立.如果在 A 点式(3)成立,经过测试、赋值,因而式(7)成立.

定理 6 经测试$(a[k] = x) \vee (i > j)$计算进入循环,则

$$i = k + 1 \wedge x > a[k] \tag{8}$$

$$j + 1 = k \wedge x < a[k] \tag{9}$$

只有一个成立.

证明 如果计算由 B 点经测试进入循环,在 B 点只能式(5)成立,故式(8)成立.如果计算通过 C 点而循环,在 C 处只能$i \leqslant j \wedge j + 1 = k \wedge x < a[k]$,这样式(9)成立.对于不变式:

$$\begin{cases} a[h] < x & h \in \{1, 2, \cdots, i - 1\} \\ a[h] > x & h \in \{j + 1, \cdots, N\} \end{cases}$$

$\{1, 2, \cdots, i - 1\}$是小于 x 值的数组元素的下标集合,可称其为"小于 x 的下标集",而 $\{j + 1, \cdots, N\}$为"大于 x 的下标集".其中 i, j 随着循环次数 l 以及所执行的语句的变化而变化.

定理 7 对于任何一次循环 l 而言,在 P 点以下两式同时成立:

$$1 = i_0 \leqslant i_1 \leqslant \cdots \leqslant i_{l-1} \leqslant i_l \tag{10}$$

$$N = j_0 \geqslant j_1 \geqslant \cdots \geqslant j_{l-1} \geqslant j_l \tag{11}$$

证明 对在 P 点处循环次数 l 施行归纳.

奠基 $l = 0$ 时,因 $i_0 = 1, j_0 = N$,故式(10)、式(11)成立.

归纳 设 $l = m$ 时,式(10)、式(11)成立.当 $l = m + 1$ 时,可依照第 m 次循环经过 B 或 C 点而分别两种情形.

情形一 第 m 次循环过 B 点.对于程序中变元 i, j 而言,在这次循环中只有变元 i 经赋值而发生变化,所以到达 P 点时 $j_m = j_{m+1}$,故式(11)成立,即 $N = j_0 \geqslant j_1 \geqslant \cdots \geqslant j_m = j_{m+1}$.

对于变元 i,因为计算过 B 点式(8)成立,故在 P 点,开始第 $m + 1$ 轮循环时有 $i_{m+1} = k_m + 1 \wedge x >$

$a[k_m]$. 将 $k_m = i_{m+1} - 1$ 代入 $x > a[k_m]$, 得 $x > a[i_{m+1} - 1]$.

因而 $i_{m+1} - 1$ 属于"小于 x 的下标集",又因数组 a 的元素已排序(按由小至大的顺序),所以对于第 $m+1$ 轮循环而言,在 P 处成立 $a[h] < x, h \in \{1, 2, \cdots, i_{m+1} - 1\}$.

又因为第 m 次循环过 A 点时,由推论 3 有 $i_m \leqslant k_m \leqslant j_m$, 加之 $i_{m+1} = k_m + 1$. 故 $i_{m+1} > i_m$. 这样第 $m+1$ 轮循环在 P 点处式(10)成立,即 $1 = i_0 \leqslant i_1 \leqslant \cdots \leqslant i_m < i_{m+1}$.

情形二 第 m 次循环过 C 点. 这时只有变元 j 发生变化,同理可得 $i_m = i_{m+1}, 1 = i_0 \leqslant i_1 \leqslant \cdots \leqslant i_m = i_{m+1}$. 这时式(9)成立,故 $a[h] > x, h \in \{j_{m+1} + 1, \cdots, N\}$. 并且 $j_m > j_{m+1}$, 这样,式(11)成立,即 $N = j_0 \geqslant j_1 \geqslant \cdots \geqslant j_m > j_{m+1}$.

推论 8　在 P 点处,以下两个断言同时成立:

$$a[h] < x \qquad h \in \{1, 2, \cdots, i-1\} \tag{12}$$

$$a[h] > x \qquad h \in \{j+1, \cdots, N\} \tag{13}$$

定理 7 除保证式(12)、式(13)成立外,同时还提供了更丰富的程序性质:首先,每次循环过 P 点时,均引起下标集变化. 可以说循环的目的就是为了构造(扩大)下标集,也正因为每次下标集均有所"扩大",这样循环终止时自然可得出结论 Q. 其次,在式(10)、式(11)的"不等式链"中,具有同下标的变元 i, j 在各自的链中,如果一个变元是用等号与链连接,另一变元在自己的链中必然是用严格不等号连接.

定理 9　如果计算经过 B 点,且因 $a[k] \neq x \ \wedge \ i > j$ 而终止,那么对这轮计算:

(a)在 P 点处两个下标集分别为 $\{1, 2, \cdots, j-1\}$, $\{j+1, \cdots, N\}$.

(b)计算进入测试 $(a[k] = x) \vee (i > j)$ 时,变元 j, k 满足 $j = k$.

(c)计算终止时,在 Q 点 $x > a[j]$.

这样循环终止时,两个下标集分别为 $\{1, 2, \cdots, j-1, j\}$, $\{j+1, \cdots, N\}$.

证明:首先逆着本次计算过程,考察变元 i, j, k 在 B 点、A 点处所应该满足的关系. 因为计算是通过 B 点而终止的,所以在 B 只能式(4)成立,即 $i > j \wedge i - 1 = j = k \wedge x > a[k]$.

再由定理 4,计算在 A 点时只能式(1)成立,即 $i = j = k$.

证(a):因为计算经过赋值 $k := (i + j)$ div2,不改变 i, j 的值. 所以在 P 点 $i = j$, 这样 P 处的两个下标集为 $\{1, 2, \cdots, j-1\}$, $\{j+1, \cdots, N\}$.

证(b):经过赋值 $i := k + 1$, 计算到达 B 时,变元 i 值增大,但变元 j, k 不变,故 $j = k$.

证(c):计算终止时,以下关系成立 $j = k \wedge x > a[k]$.

将 $j = k$ 代入 $x > a[k]$, 得 $x > a[j]$. 表明计算终止是因为 $x > a[j]$, 从而 j 成为小于 x 的下标集中新加入的一个元素.

可以仿照定理 9 的证明,因而可得:

定理 10　如果计算经过 C 点且因 $a[k] \neq x \wedge i > j$ 而终止,那么对这轮计算:

(a)在 P 点处两个下标集分别为 $\{1, 2, \cdots, i-1\}$, $\{i+1, \cdots, N\}$.

(b)计算进入测试 $(a[k] = x) \vee (i > j)$ 时,变元 i, k 满足 $i = k$.

(c)计算终止时,在 Q 点 $x < a[i]$.

这样循环终止时,两个下标集分别为 $\{1, 2, \cdots, i-1\}$, $\{i, i+1, \cdots, N\}$. 由定理 9,10 易得:

推论 11　如果程序因 $a[h] \neq x \wedge i > j$ 而终止,那么在 Q 点有:$1 \leqslant h \leqslant N \supset a[h] \neq x$.

由此可得出结论,Wirth 的"对半检索"程序在 repeat 语句入口处的不变式断言:

$$a[h] < x \quad \text{对 } h = 1, \cdots, i-1$$

$$a[h] \geqslant x \quad \text{对 } h = j+1, \cdots, N$$

是不严格的,准确的表达应是:

$$a[h] < x \quad \text{对 } h = 1, \cdots, i-1$$

$$a[h] > x \quad \text{对 } h = j+1, \cdots, N$$

3　余　　论

本文所讨论的只是一个简单的程序,但从中我们得到如下的启示:

a.任何一个程序(算法)在没有经过严格测试和论证之前,是无法断定其正确性的.从本文来看,程序的论证过程比程序本身复杂许多,涉及的定理也不那么明显、自然,这正暗示了程序的正确性论证比想象的要难得多.如何寻求一种程序正确性论证的'形式方法'应该引起人们的注意.

b.程序中的断言及对断言的论证,应该是程序中最有价值的资料,这些资料反映了程序作为一种对象,其整体的性质.如本文的例子中,程序之所以需要循环,正是为了不断构造非 x 值的下标集,也正因为每次循环使下标集有所增长,故而采用的算法是高效的.此外,还可看到 P 和 Q 间的密切关系,如果数组中不含有 x 值,最终通过测试,致使两个下标集吻合.程序的这些特性,在一般测试乃至设计中,未必能被认识.只有在严格的论证中,才能得以披露.

c.软件复用技术将显著提高软件开发效率,避免重复劳动.程序作为软件重构技术中的可重用软构件,应对其进行某种'重用认证',以确认重用软件有所期望的性质.如果不能把握程序的整体特性而轻率地重用是十分危险的.据此,多为程序作些'定型',并寻找一些相应的'定理'应该是一项有意义的工作[3].

参考文献:

[1] Wirth N. 算法 + 数据结构 = 程序[M].北京 科学出版社,1984.20 ~ 21.
[2] Kleen S C. 元数学导论[M].北京 科学出版社,1984.201.
[3] Dijkstra E W. 结构程序设计[M].北京 科学出版社,1991.12 ~ 14.

Correction of One of the Wirth Invariants

ZOU Shu-zhi

(*College of Engineering*, *Yangzhou Univ.*, *Yangzhou* 225009, *China*)

Abstract: An invariant given by N. Wirth in Algorithm + Datastructure = Program has been proved more or less inaccurate in "Binary Research" Program. In this paper, the inaccuracy is modified on the basis of related principles. Furthermore, the properties of the complete program as an object are discussed, which are not recognized in general tests.

Key words: binary research; invariant; program correctness