免用第三变量的两变量值交换
Swap of Two Variables without the Third Variable

徐金建，汤　彬
Xu Jinjian，Tang Bin
（东华理工学院，江西临川　344000）
（East China Institute of Technology，Linchuan，Jiangxi，344000，China）

摘要：针对采用第三变量两变量值的交换占用多余内存的不足，提出算术运算法、布尔型变量的交换法、异或运算法和异或运算取反法等 4 个免用第三个临时变量的两变量值交换方法，并分析这些方法的优缺点。

关键词：变量交换　第三变量　异或运算
中图法分类号：TP301.6

Abstract：To solve the problem of taking surplus of memory in swapping two variables using the third variable，the methods of arithmetic algorithm，swapping of boolean type variables，XOR and negated XOR are suggested. The advantages and disadvantages of these methods are discussed.

Key words：swap of variables，the third varcable，XOR

1　采用第三变量法

在程序设计中实现两变量值的交换时，常规的做法是定义第三个变量，作为程序中的临时变量。例如：给出变量 a 和 b，那么再定义另一个变量 c 作为临时变量，通过以下的形式来实现：

```
sub swap(a,b)
    c=a
    a=b
    b=c
end sub
```

这种方法可以简单地实现 2 个变量值的交换，它的优点是适用任意数据类型，缺点是运行该段程序时，计算机给临时变量开辟一个的存储空间。有时这会占用较多的内存，如果内存资源非常有限，那么这种变量交换法是不可取的。

2　算术运算法

在一定条件约束的情况下，通过算术运算方法来现两变量值的交换，该方法可以节省的内存资源。

```
sub swap(a,b)
    a=a+b
    b=a-b
    a=a-b
end sub
```

算术运算法实现变量交换的做法比较巧妙，它可以解决内存资源紧张的问题，便该方法不能全面解决问题，但它存在以下缺点：

(1)不适合布尔型变量。

在程序设计中，布尔型的变量值只存在 2 种，即 True 和 False，它们不可以进行算术运算，只可以进行逻辑运算。

(2)数值运算溢出。

由于该方法引入了算术运算，这就可能出现数值运算的溢出问题。例如，假如计算机内部采用 1 个字节表示整型数，那么整型数的范围是 $(-2^7 \sim 2^7-1)$ 即 $(-128 \sim 127)$，当 2 个变量做运算时得到的结果超出了 $(-128 \sim 127)$，会产生溢出的现象，导致 2 个变量值交换后与原值不同。

(3)浮点数的限制。

在程序设计中，有些浮点数是不准确的，只能是一种近似值。在表示这样的浮点数时，计算机将自动地切掉后面超出它所能表示的精度范围内的浮点数，导致不可避免的误差。例如，程序设计中浮点数的规律如下：

$2^{-1}=0.5$

$2^{-2}=0.25$

$2^{-3}=0.125$

……

如果 2 个变量的值分别为 $a=10.3$ 和 $b=7.5$，

2003-09-27 收稿。

按照这种表示方法，虽然变量 b 的值 7.5 可精确地在程序设计中表示出来，但是变量 a 的值 10.3 在程序设计中是无法精确表示出来的。也就是说，$a+b$ 的值不会象常规那样等于 17.8。程序设计中，计算出的这个和可能是略大于 17.8 也可能是略小于 17.8，但不可能确切的等于 17.8。说明算术运算法来实现两变量值的交换会导致交换后的变量值有差异。

通过算术运算法分析可知，算术运算法虽然可以在不定义第三变量实现两变量之间的交换，但存在一定的局限性。

3 布尔型变量的交换法

布尔型变理的变量与整数、浮点数不同，由于布尔型变量不可以进行算术运算，只能进行逻辑运算，只能通过以下方法实现 2 个布尔型变量值的交换：

```
if(! a&&b)=true          case(1)
if(a&&! b)=true          case(2)
if(a&&b)=true            case(3)
if(! a&&! b)=true        case(4)
```

当发生第三或第四种情况时，a 和 b 的值是相同的，2 个变量的值不需要进行交换，而当发生第一或第二种情况时，a 和 b 的值是不同。由于布尔型变量的取值只存在 2 种可能 true 或 false，因此可以通过下列来实现两变量的值交换：

```
if(! a&&b)=true then
{a=! a      b=! b}
if(a&&! b)=ture then
{a=a!       b=! b}
```

同样，在这种情况下实现两布尔型变量值的交换不引进第三变量，但以上的实现方法增加判断过程，增加程序的执行时间。所以说，这种方法的程序运行性能并未改善，特别是当 1 个大的程序当中存在大量这样的状态语句时，情况更甚，大大的增加程序的执行时间。

4 异或运算法

由 $(a\odot b)$ 设 $a,b\in\{0,1\}$，2 个 1 位二进制数 a，b 的异或 $a\odot b$ 的真值如表 1 所示。

表 1 异或运算真值[1]

a	b	$a\odot b$
1	1	0
1	0	1
0	1	1
0	0	0

4.1 布尔型变量的异或运算

从表 1 中可以看出，if(! (a xor b))=true，此时布尔型变量 a 和 b 的值相同，不需要对它们进行交换，对其它情况：if(a xor b)=true，此时布尔型变量 a 和 b 的值是不同，交换 $(a=!a,b=!b)$

```
if (a xor b)=true   then
{a=! a,b=! b}
```

在实际的应用过程中，只需要 1 个条件状态语句即可以实现两个布尔变量的值交换。比算术运算法节省了 3 条 if 语句，减少了程序运行的时间，提高了效率。

4.2 整数型变量的异或运算

$(x$ xor $y)x,y\in$ 整数，x,y 的按位异或 x xor y 是对相应的二进制数按位进行 $\odot$ 运算，其结果仍转换为十进制数．如果相应的二进制位数不等，较小的数前面的空白按零处理．例如，给出 2 个变量 a 和 b，$a=13$，$b=9$，免用第三变量实现这两个变量的值交换为：

13 xor 9 的异或过程：

a=13 xor 9=(! 13 &&. 9) or (13 &&. ! 9)=$(0010)_2=(2)_{10}$

b=2 xor 9=(! 2 &&. 9) or (2 &&. ! 9)=$(1011)_2=(13)_{10}$

a=2 xor 13 =(! 2 &&. 13) or (2 &&. ! 13)=$(1001)_2=(9)_{10}$

这种方法实现了整数间的互换，对于浮点数和布尔型变量同样成立。此时得到变量值交换的函数如下：

```
sub swap(a,b)
  a=a xor b
  b=a xor b
  a=a xor b
end sub
```

异或运算法比较巧妙，适用于整数、字符型、浮点型和布尔型，但不能用于复杂的数据类型，如字符串类型。

（下转第 63 页）

30 Benoit Baudry, Yves Le Traon, Gerson Sunyé. Testability analysis of a UML class diagram. in Proceedings of IEEE Metrics02, Ottawa, Canada, 2002. 54～65.

31 Jeff Offutt, Aynur Abdurazik. Generating tests from UML specifications. in Proceedings of UML99, Fort Collins, 1999. 416～429.

32 Kim Y, Carlson C R. Scenario based integration testing for object-oriented software development. in Proceedings of the Eighth Asian Test Symposium, Shanghai, China, 1999. 283～288.

33 Briand L, Labiche Y. A UML-Based approach to system testing. Software and Systems Modeling Springer, 2002, 1(1): 10～42.

34 刘 敏, 金茂忠, 刘 超. 基于 UML 活动图模型生成测试场景的设计. 计算机工程与应用, 2002, 12: 122～124.

35 张 楣, 刘 超, 孙昌爱. 基于 UML 活动图模型的测试用例生成技术研究. 北京航空航天大学学报, 2001, 27(4): 433～437.

36 Ben Lieberman. UML activity diagrams versatile roadmaps for understanding system behaviour. http://www.therationalegdge.com/content/apr_01/t_UML_bl.html.

37 Jeremiah Wittevrongel, Frank Maurer. Using UML to partially automate generation of scenario-based test drivers. in Proceedings of OOIS '01, University of Calgary, Canada, 2001. 303～306.

38 Jeremiah Wittevrongel, Frank Maurer. SCENTOR: scenario-based testing of E-business applications. in Proceedings of 10th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, Massachusetts, 2001. 41～48.

（责任编辑：邓大玉）

（上接第 56 页）

5 异或运算取反法

经过分析异或运算法可知，实现两变量的交换时也可采用如下方法：

```
sub swap(a,b)
  a=not(a xor b)
  b=not(a xor b)
  a=not(a xor b)
end sub
```

该方法与异或方法实质上是一样的，虽然比上面的方法要多计算一个过程，但不失为一种好方法，该方法也适合于整数、字符型、浮点型和布尔型，但不能用于其复杂的数值类型，如字符串类型。

6 结束语

以上这种方法都是免用第三变量来实现两变量的交换，减少了内存资源和提高了执行效率。但这几个方法仍然存在一些不足之处，对一些复杂的数据类型仍然行不通。这就要求我们在设计程序的，针对不同的变量类型，采用不同的方法，实现内存资源的优化和提高整个程序的执行速度，使程序达到最优化的目的。

参考文献：

1 白中英. 数字逻辑与数字系统. 第 2 版. 北京：人民邮电出版社，1999. 9.

（责任编辑：黎贞崇）