文章编号:2095-7386(2016)03-0068-06
DOI:10.3969/j.issn.2095-7386.2016.03.013

一种由遍历序列构造二叉树的改进算法

王防修[1],刘春红[2]
(1.武汉轻工大学 数学与计算机学院,湖北 武汉 430023;2.九州通医药集团物流有限公司,湖北 武汉 430040)

摘 要:针对现有构造二叉树的算法无法适用于具有相同元素的遍历序列,提出了一种解决该问题的递归算法。该种算法以现有的递归算法为基础,通过引入遍历序列的标志序列,依据标志序列中元素之间的关系,从理论上证明了三种由遍历序列构造二叉树的算法都具有递归性。根据遍历序列构造二叉树的递归原理,设计了三种不同的由遍历序列构造二叉树的递归算法。通过算例仿真表明,使用笔者设计的算法可为具有相同元素的遍历序列构造二叉树。

关键词:先序遍历;中序遍历;后序遍历;标志序列;递归算法
中图分类号:TP 391 **文献标识码**:A

An improved algorithm for constructing two binary trees by traversing sequences

WANG Fang-xiu[1], *LIU Chun-hong*[2]
(1. School of Mathematics and Computer Science, Wuhan Polytechnic University, Wuhan 430023, China;
2. Jointown Pharmaceutical Group Logistics Co., Ltd. Wuhan 430040, China)

Abstract: In view of the present algorithm can not be applied to construct the binary tree by using the traversal sequence which has the same elements, this paper presents a recursive algorithm to solve the problem. Based on the existing recursive algorithm, this algorithm introduces the symbol sequence of the traversal sequence. According to the relationship among the elements in the symbol sequence, the three algorithms are proved theoretically to be recursive for using the traversal sequences to construct the binary tree. Based on the recursive principle of constructing the two binary tree by the travel sequences, three different recursive algorithms are designed to construct the binary tree. Simulation results show that the algorithm can be used to construct the two binary tree by using the traversal sequences in which there are the same elements.

Key words: preorder traversal; inorder traversal; postorder traversal; flag sequence; recursive algorithm

1 引言

作为一种典型的层次结构,二叉树[1]在解决现实生活中的一些实际问题时起着非常重要的作用。因此,用遍历序列构造二叉树一直是人们关注的热点问题[2-6]。通过对二叉树遍历序列性质的研究,出现了一序列由遍历序列构造二叉树的递归[7-8]和非递归算法[9-12]。然而,无论是递归还是非递归算法,

收稿日期:2016-04-08.
作者简介:王防修(1973-),男,副教授,E-mail:wfx323@126.com.
基金项目:国家自然科学基金资助项目(61179032).

都要求遍历序列中不能有相同元素出现,否则算法无能为力。显然,这种要求会大大限制现有算法的使用范围。事实上,很多情况下的序列都会有相同元素出现,比如常见的用二叉树表示的算术表达式就是典型的存在相同运算符号和运算对象的遍历序列。因此,笔者通过为遍历序列引入标志序列,对现有递归算法进行了改进,使得改进后的算法对遍历序列没有任何条件限制,即无论序列中有无重复元素,都能用本算法构造二叉树。

2　用遍历序列建立二叉树的数学原理

定理 1　如果已知一棵二叉树的先序遍历序列和中序遍历序列,则可用递归算法建立该二叉树。

证明 设 $X = x_i x_{i+1} \cdots x_j$ 和 $Y = y_k y_{k+1} \cdots y_l$ 分别是二叉树 T 的先序遍历序列和中序遍历序列。考虑到序列中可能存在相同元素,故需要为序列中的每个元素赋予一个不同标志,以示与其它元素的区别,故又设 $A = a_i a_{i+1} \cdots a_j$ 和 $B = b_k b_{k+1} \cdots b_l$ 分别是先序遍历序列 X 和中序遍历序列 Y 的元素标志序列,由于 X 和 Y 的长度相等,则有 $j - i + 1 = l - k + 1$,即 $j - i = l - k$。

由二叉树的遍历序列性质可知 $\{x_i, x_{i+1}, \cdots, x_j\} = \{y_k, y_{k+1}, \cdots, y_l\}$ 和 $\{a_i, a_{i+1}, \cdots, a_j\} = \{b_k, b_{k+1}, \cdots, b_l\}$,即 X 和 Y 是由相同的元素组成的符号序列,而 A 和 B 也是由相同的元素组成的标志序列。根据二叉树先序遍历序列的特点,元素 x_i 是二叉树 T 的根结点。由于 B 中存在元素 $b_m = a_i$,故从中序遍历序列 Y 的角度看,y_m 是二叉树 T 的根结点。因此,中序遍历 Y 及其标志序列 B 可以进行如下式(1)划分:

$$\begin{cases} Y = (y_k \cdots y_{m-1})\, y_m (y_{m+1} \cdots y_l), \\ B = (b_k \cdots b_{m-1})\, b_m (b_{m+1} \cdots b_l). \end{cases} \tag{1}$$

由中序遍历序列的性质可知,由于 y_m 是二叉树的根结点,故子序列 $y_k \cdots y_{m-1}$ 和 $b_k \cdots b_{m-1}$ 分别是 T 的左子树的中序遍历序列和标志序列,而 $y_{m+1} \cdots y_l$ 和 $b_{m+1} \cdots b_l$ 分别是 T 的右子树的中序遍历序列和标志序列。由二叉树的先序遍历序列的性质可知,一定存在位置 p,使先序遍历序列 X 及其标志序列 A 可以进行如下式(2)划分:

$$\begin{cases} X = x_i (x_{i+1} \cdots x_p)(x_{p+1} \cdots x_j), \\ A = a_i (a_{i+1} \cdots a_p)(a_{p+1} \cdots a_j). \end{cases} \tag{2}$$

其中 $x_{i+1} \cdots x_p$ 和 $a_{i+1} \cdots a_p$ 分别是 T 的左子树的先序遍历序列和标志序列,而 $x_{p+1} \cdots x_j$ 和 $a_{p+1} \cdots a_j$ 分别是 T 的右子树的先序遍历序列和标志序列。要得到这两个先序遍历子序列及其标志序列,必须由已知条件求出位置 p。由于先序遍历子序列和中序遍历子序列的长度相等,则有:

$$\{x_{i+1}, \cdots, x_p\} = \{y_k, \cdots, y_{m-1}\}. \tag{3}$$

$$\{x_{p+1}, \cdots, x_j\} = \{y_{m+1}, \cdots, y_l\}. \tag{4}$$

由式(3),式(4)可知子序列的长度相等,故有:

$$p - (i + 1) + 1 = m - 1 - k + 1. \tag{5}$$

$$j - (p + 1) + 1 = l - (m + 1) + 1. \tag{6}$$

由式(5),式(6)分别得位置 $p = m + i - k$ 和位置 $p = m + j - l$。

由于 $j - i = l - k$,由式(5)和式(6)可得两个位置 p 的值相等,故该位置是唯一的。根据求得的位置 p,可以得到两个子序列 $x_{i+1} \cdots x_p$ 和 $x_{p+1} \cdots x_j$。因此,$x_{i+1} \cdots x_p$ 和 $y_k \cdots y_{m-1}$ 分别表示二叉树 T 的左子树的先序遍历子序列和中序遍历子序列,而 $x_{p+1} \cdots x_j$ 和 $y_{m+1} \cdots y_l$ 分别表示二叉树 T 的右子树的先序遍历子序列和中序遍历子序列。同理,由 $x_{i+1} \cdots x_p$,$a_{i+1} \cdots a_p$ 和 $y_k \cdots y_{m-1}$,$b_k \cdots b_{m-1}$ 可求出 T 的左子树,而由 $x_{p+1} \cdots x_j$,$a_{p+1} \cdots a_j$ 和 $y_{m+1} \cdots y_l$,$b_{m+1} \cdots b_l$ 可以求出 T 的右子树。因此,由先序遍历和中序遍历构造二叉树实际上是一个递归过程。

递归子结构为:如果 $m > k$,则由 $x_{i+1} \cdots x_p$,$a_{i+1} \cdots a_p$ 和 $y_k \cdots y_{m-1}$,$b_k \cdots b_{m-1}$ 递归建立 T 的左子树。如果 $m < l$,则由 $x_{p+1} \cdots x_j$,$a_{p+1} \cdots a_j$ 和 $y_{m+1} \cdots y_l$,$b_{m+1} \cdots b_l$ 递归建立 T 的右子树。

递归终止条件为:如果 $m = k$,则 T 无左子树。如果 $m = l$,则 T 无右子树。

定理 2　如果已知一棵二叉树的中序遍历序列和后序遍历序列,则可用递归算法建立该二叉树。

证明　设 $Y = y_i y_{i+1} \cdots y_j$ 和 $Z = z_k z_{k+1} \cdots z_l$ 分别是二叉树 T 的中序遍历序列和后序遍历序列,用 $B = b_i b_{i+1} \cdots b_j$ 和 $C = c_k c_{k+1} \cdots c_l$ 分别表示 Y 和 Z 的元素标志序列,则由 Y 和 Z 的长度相等得 $j - i = l - k$。由一棵二叉树的中序遍历和后序遍历之间的关系如式(7):

$$\{y_i, y_{i+1}, \cdots, y_j\} = \{z_k, z_{k+1}, \cdots, z_l\}. \tag{7}$$

设 Y 和 Z 是由相同的元素组成的不同序列。由后序遍历序列的性质可知,元素 z_l 是二叉树 T 的根结点。由于 B 中存在元素 $b_m = c_l$,故从中序遍历序列 Y 的角度看,y_m 是二叉树 T 的根结点。因此,中序遍历序列 Y 及其标志序列可以进行如下式(8)划分:

$$\begin{cases} Y = (y_i y_{m-1}) y_m (y_{m+1} \cdots y_j), \\ B = (b_i \cdots b_{m-1}) b_m (b_{m+1} \cdots b_j). \end{cases} \tag{8}$$

由中序遍历序列的性质可知,子序列 $y_i \cdots y_{m-1}$ 和 $b_i \cdots b_{m-1}$ 分别是 T 的左子树的中序遍历序列和标志序列,而 $y_{m+1} \cdots y_j$ 和 $b_{m+1} \cdots b_j$ 分别是 T 的右子树的中序遍历序列和标志序列。根据二叉树的后序遍历序列的性质可知,后序遍历序列 Z 及其标志序列可以进行如下式(9)划分:

$$\begin{cases} Z = (z_k \cdots z_p)(z_{p+1} \cdots z_{l-1}) z_l, \\ C = (c_k \cdots c_p)(c_{p+1} \cdots c_{l-1}) c_l. \end{cases} \tag{9}$$

其中 $z_k \cdots z_p$ 和 $c_k \cdots c_p$ 分别是 T 的左子树的后序遍历序列和标志序列,而 $z_{p+1} \cdots z_{l-1}$ 和 $c_{p+1} \cdots c_{l-1}$ 分别是 T 的右子树的后序遍历序列和标志序列。由中序遍历和后序遍历之间的关系如式(10)和式(11):

$$\{z_k, \cdots, z_p\} = \{y_i, \cdots, y_{m-1}\}, \tag{10}$$

$$\{z_{p+1}, \cdots, z_{l-1}\} = \{y_{m+1}, \cdots, y_j\}. \tag{11}$$

由于子序列的长度相等,则有:

$$p - k + 1 = m - 1 - i + 1. \tag{12}$$

$$(l - 1) - (p + 1) + 1 = j - (m + 1) + 1. \tag{13}$$

由式(12)和式(13)分别得 $p = k + m - i - 1$,和 $p = l + m - j - 1$。

由于 $j - i = l - k$,可以得到这两个 p 的值是相等的。同理,由中序遍历子序列 $y_i \cdots y_{m-1}$,$b_i \cdots b_{m-1}$ 和后序遍历子序列 $z_k \cdots z_p$,$c_k \cdots c_p$ 可以得到二叉树 T 的左子树。由中序遍历子序列 $y_{m+1} \cdots y_j$,$b_{m+1} \cdots b_j$ 和后序遍历子序列 $z_{p+1} \cdots z_{l-1}$,$c_{p+1} \cdots c_{l-1}$ 可以得到二叉树 T 的右子树。因此,由二叉树的先序遍历序列和中序遍历序列构造二叉树也是一个递归过程。

递归子结构为:如果 $m > i$,则由 $y_i \cdots y_{m-1}$,$b_i \cdots b_{m-1}$ 和 $z_k \cdots z_p$,$c_k \cdots c_p$ 递归建立 T 的左子树。如果 $m < j$,则由 $y_{m+1} \cdots y_j$,$b_{m+1} \cdots b_j$ 和 $z_{p+1} \cdots z_{l-1}$,$c_{p+1} \cdots c_{l-1}$ 递归建立 T 的右子树。

递归终止条件为:如果 $m = i$,则 T 无左子树。如果 $m = j$,则 T 无右子树。

定理3 如果已知一棵二叉树的先序遍历序列和后序遍历序列,并且该二叉树没有出度为1的结点,则可用递归算法建立该二叉树。

证明 设 $X = x_i x_{i+1} \cdots x_j$ 和 $Z = z_k z_{k+1} \cdots z_l$ 分别是二叉树 T 的先序遍历序列和后序遍历序列,又设 $A = a_i a_{i+1} \cdots a_j$ 和 $C = c_k c_{k+1} \cdots c_l$ 分别是 Y 和 Z 的标志序列。由 X 和 Z 的长度相等得出 $j - i = l - k$。由先序遍历序列和后序遍历序列的关系可知,元素 $x_i = z_l$,它们是二叉树 T 的根结点。由于 C 中存在元素 $c_m = a_{l+1}$。因此,后序遍历序列 Z 可以划分为:

$$\begin{cases} Z = (z_k \cdots z_m)(z_{m+1} \cdots z_{l-1}) z_l, \\ C = (c_k \cdots c_m)(c_{m+1} \cdots c_{l-1}) c_l. \end{cases} \tag{14}$$

由后序遍历序列的性质可知,子序列 $z_k \cdots z_m$ 和 $c_k \cdots c_m$ 分别是 T 的左子树的后序遍历序列和标志序列,而 $z_{m+1} \cdots z_{l-1}$ 和 $c_{m+1} \cdots c_{l-1}$ 分别是 T 的右子树的后序遍历序列和标志序列。根据二叉树的先序遍历序列的性质可知,先序遍历序列 X 及其标志序列可以进行如下划分:

$$\begin{cases} X = x_i (x_{i+1} \cdots x_p)(x_{p+1} \cdots x_j), \\ A = a_i (a_{i+1} \cdots a_p)(a_{p+1} \cdots a_j). \end{cases} \tag{15}$$

其中 $x_{i+1} \cdots x_p$ 和 $a_{i+1} \cdots a_p$ 分别是 T 的左子树的先序遍历序列和标志序列,而 $x_{p+1} \cdots x_j$ 和 $a_{p+1} \cdots a_j$ 分别是 T 的右子树的先序遍历序列和标志序列。显然可得式(16)和式(17):

$$\{z_k, \cdots, z_m\} = \{x_{i+1}, \cdots, x_p\}. \tag{16}$$

$$\{z_{m+1}, \cdots, z_{l-1}\} = \{x_{p+1}, \cdots, x_j\}. \tag{17}$$

由于子序列的长度相等,则有:

$$p - i - 1 = m - k. \tag{18}$$

$$(l - 1) - (m + 1) = j - (p + 1). \tag{19}$$

由式(18)和式(19)分别得 $p = i + m - k + 1$ 和 $p = j + m - l + 1$。

由于 $j - i = l - k$,可得这两个 p 的值是相等的。因此,由先序遍历子序列 $x_{i+1} \cdots x_p$,$a_{i+1} \cdots a_p$ 和后序遍历子序列 $z_k \cdots z_m$,$c_k \cdots c_m$ 可以得到二叉树 T 的左子树。由先序遍历子序列 $x_{p+1} \cdots x_j$,$a_{p+1} \cdots a_j$ 和后序遍历子序列 $z_{m+1} \cdots z_{l-1}$,$c_{m+1} \cdots c_{l-1}$ 可以得到二叉树 T 的右子树。因此,由二叉树的先序遍历序列和后序遍历序列构造二叉树也是一个递归过程。

递归子结构为:如果 $m > i$,则 $x_{i+1} \cdots x_p$,$a_{i+1} \cdots a_p$ 和 $z_k \cdots z_m$,$c_k \cdots c_m$ 递归建立 T 的左子树。如果 $m < j$,则由 $x_{p+1} \cdots x_j$,$a_{p+1} \cdots a_j$ 和 $z_{m+1} \cdots z_{l-1}$,$c_{m+1} \cdots c_{l-1}$ 递归建立 T 的右子树。

递归终止条件为:如果 $i = j$,则 T 是叶子结点,即 T 既无无左子树又无右子树。

3 由遍历序列建立二叉树的算法设计

3.1 由先序遍历序列和中序遍历序列构造二叉树

为方便算法设计,不妨设建立二叉树的递归函数为 $T = f(X, A, Y, B, i, j, k, l)$,则其递归过程可以描述如下:

(1)由先序遍历序列 $X = x_i x_{i+1} \cdots x_j$ 可知 X 中的第一个元素 x 是二叉树 T 的根结点。

(2)从标志序列 B 中查找到 $b_m = a_i$。

(3)如果 $m = k$,则根结点 T 没有左孩子;否则,二叉树的根结点 T 的左孩子是其左子树的根结点,若设其左孩子为 T_l,则有

$$T_l = f(X,A,Y,B,i+1,m+i-k,k,m-1). \quad (20)$$

或者

$$T_l = f(X,A,Y,B,i+1,m+j-l,k,m-1). \quad (21)$$

(4)如果 $m = l$,则二叉树的根结点 T 没有右孩子;否则,根结点 T 的右孩子是其右子树的根结点。若设 T 的右孩子为 T_r,则有

$$T_r = f(X,A,Y,B,m+i-k+1,j,m+1,l). \quad (22)$$

或者

$$T_r = f(X,A,Y,B,m+j-l+1,j,m+1,l). \quad (23)$$

(5)当递归过程结束时,则得到一个根结点为 T 的二叉树。

3.2　由中序遍历序列和后序遍历序列构造二叉树

为方便算法描述,设建立二叉树的递归函数为 $T = g(Y,B,Z,C,i,j,k,l)$,其递归过程描述如下:

(1)后序遍历序列 $Z = z_k z_{k+1} \cdots z_l$ 中的元素 z_i 是二叉树 T 的根结点;

(2)从标志序列 B 中找到 $b_m = c_l$。

(3)如果 $m = i$,则二叉树 T 没有左子树;否则,二叉树 T 的左子树的根结点为

$$g(Y,B,Z,C,i,m-1,k,m+k-i-1). \quad (24)$$

或者

$$g(Y,B,Z,C,i,m-1,k,l+m-j-1). \quad (25)$$

(4)如果 $m = j$,则二叉树 T 没有右子树;否则,二叉树 T 的右子树的根结点为

$$g(Y,B,Z,C,m+1,j,k+m-i,l-1). \quad (26)$$

或者

$$g(Y,B,Z,C,m+1,j,l+m-j,l-1). \quad (27)$$

3.3　由先序遍历序列和后序遍历序列建立无出度为 1 的二叉树

为方便算法描述,设建立二叉树的递归函数为 $T = h(X,A,Z,C,i,j,k,l)$,则其递归过程描述如下:

(1)先序遍历序列 $X = x_i x_{i+1} \cdots x_j$ 中的元素 x_i 是二叉树 T 的根结点。

(2)从标志序列 C 中找到 $c_m = a_{l+1}$。

(3)如果 $j = i$ 或 $l = k$,则二叉树 T 没有左子树和右子树;否则,二叉树 T 的左子树的根结点为

$$h(X,A,Z,C,i+1,i+m-k+1,k,m). \quad (28)$$

或者

$$h(X,A,Z,C,i+1,j+m-l+1,k,m). \quad (29)$$

二叉树 T 的右子树的根结点为

$$h(X,A,Z,C,i+m-k+2,j,m+1,l-1). \quad (30)$$

或者

$$h(X,A,Z,C,j+m-l+2,j,m+1,l-1). \quad (31)$$

4　算法仿真

算例　试用上述三种算法构造如图 1 所示的算术表达式的二叉树。

解　从二叉树中可以看出,树中既有相同的运算对象(运算对象 a 和 b 都出现两次),又有相同的运算符号(其中“+”出现三次,“*”出现两次)。因此,用传统算法不能建立该二叉树。

为了将二叉树中的相同元素区分开来,不妨给树中每个元素赋予一个不同的整数标志。图 2 给出了二叉树中每个元素对应的整数标志。

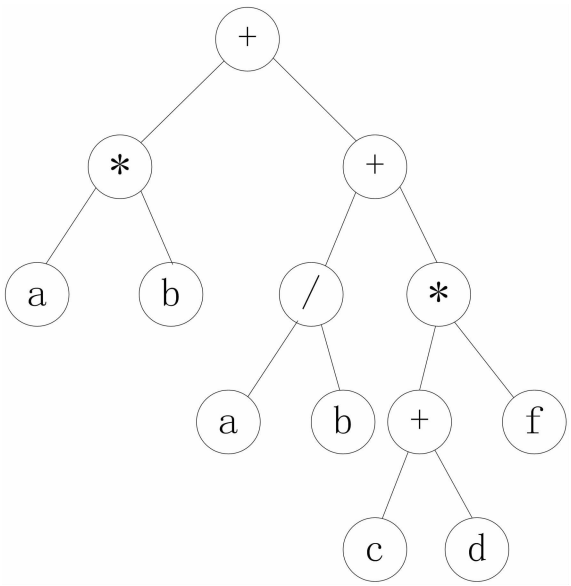

图 1　表达式 a * b + (a/b + (c + d) * f) 的二叉树

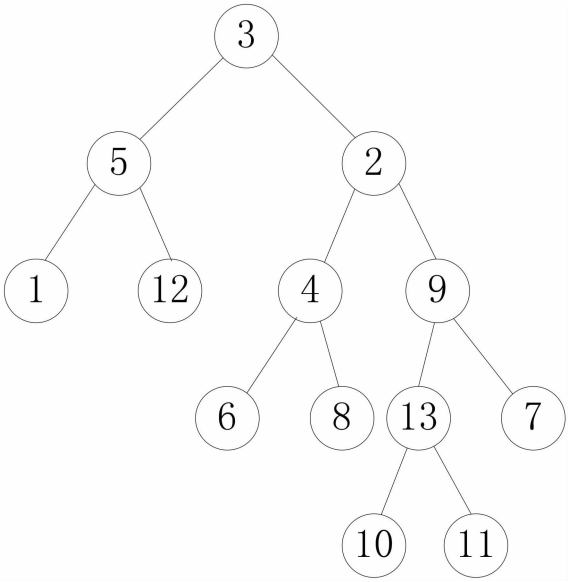

图 2 二叉树中元素的对应标志

因此,由图 1 和图 2 可以得到如表 1 所示的遍历序列和标志序列。

表 1 二叉树的遍历序列和标志序列

i	x_i	a_i	y_i	b_i	z_i	c
1	+	3	a	1	a	1
2	*	5	*	5	b	12
3	a	1	b	12	*	5
4	b	12	+	3	a	6
5	+	2	a	6	b	8
6	/	4	/	4	/	4
7	a	6	b	8	c	10
8	b	8	+	2	d	11
9	*	9	c	10	+	13
10	+	13	+	13	f	7
11	c	10	d	11	*	9
12	d	11	*	9	+	2
13	f	7	f	7	+	3

在表 1 中,$X=x_1\cdots\cdots x_{13}$和$A=a_1\cdots\cdots a_{13}$分别表示二叉树的先序遍历序列及其标志序列,$Y=y_1\cdots\cdots y_{13}$和$B=b_1\cdots\cdots b_{13}$分别表示二叉树的中序遍历序列及其标志序列,而$Z=z_1\cdots\cdots z_{13}$和$C=c_1\cdots\cdots c_{13}$分别表示二叉树的后序遍历序列及其标志序列。

方法一 根据算法 2.1,由先序遍历序列及其标志序列 X,A 和中序遍历序列及其标志序列 Y,B 构造二叉树的过程如下:

(1)由 $f(X,A,Y,B,1,13,1,13)$ 得根结点为 $x_1(+)$ 和 $m=4$。

(2)由于 $m=4$,故由 $f(X,A,Y,B,2,4,1,3)$ 可以得到 $x_1(+)$ 的左孩子为 $x_2(*)$ 和 $m=2$,而由 $m=2$ 可以得到 $x_2(*)$ 的左孩子 $x_3(a)$ 和右孩子 $x_4(b)$。同样,由于 $m=4$,由 $f(X,A,Y,B,5,13,5,13)$ 可以得到 $x_1(+)$ 的右孩子为 $x_5(+)$ 和 $m=8$。

(3)当 $m=8$ 时,由 $f(X,A,Y,B,6,8,5,7)$ 可以得到 $x_5(+)$ 的左孩子为 $x_6(/)$,以及 $x_6(/)$ 的左孩子 $x_7(a)$ 和右孩子 $x_8(b)$。由 $f(X,A,Y,B,9,13,9,13)$ 可以得到 $x_5(+)$ 的右孩子 $x_9(*)$ 和 $m=12$,而 $x_9(*)$ 的右孩子为 $x_{13}(f)$。

(4)当 $m=12$ 时,由 $f(X,A,Y,B,10,12,9,11)$ 可以得到 $x_9(*)$ 的左孩子为 $x_{10}(+)$,而 $x_{10}(+)$ 的左孩子为 $x_{11}(c)$ 和右孩子为 $x_{12}(d)$。

方法二 根据算法 2.2,由中序遍历序列及其标志序列 Y,B 和后序遍历序列及其标志序列 Z,C 构造二叉树的过程如下:

(1)由 $g(Y,B,Z,C,1,13,1,13)$ 得到根结点为 $z_{13}(+)$ 和 $m=4$。

(2)当 $m=4$ 时,由 $g(Y,B,Z,C,1,3,1,3)$ 得到 $z_{13}(+)$ 的左孩子为 $z_3(*)$ 和 $m=4$,由 $g(Y,B,Z,C,5,13,4,12)$ 得到 $z_{13}(+)$ 的右孩子为 $z_{12}(+)$ 和 $m=8$。

(3)当 $m=2$ 时,由 $g(Y,B,Z,C,1,1,1,1)$ 得到 $z_3(*)$ 的左孩子 $z_1(a)$,而由 $g(Y,B,Z,C,3,3,2,2)$ 得到 $z_3(*)$ 的右孩子 $z_2(b)$。

(4)当 $m=8$ 时,由 $g(Y,B,Z,C,5,7,4,6)$ 得到 $x_{12}(+)$ 的左孩子 $z_6(/)$ 和 $m=6$,而由 $g(Y,B,Z,C,9,13,7,11)$ 得到 $z_{12}(+)$ 的右孩子 $z_{11}(*)$ 和 $m=12$。

(5)当 $m=6$ 时,由 $g(Y,B,Z,C,5,5,4,4)$ 得到 $z_6(/)$ 的左孩子 $z_4(a)$,而由 $g(Y,B,Z,C,7,7,5,5)$ 得到 $z_6(/)$ 的右孩子 $z_5(b)$。

(6)当 $m=12$ 时,由 $g(Y,B,Z,C,9,11,7,9)$ 得到 $z_{11}(*)$ 的左孩子 $z_9(+)$ 和 $m=10$,而由 $g(Y,B,Z,C,13,13,10,10)$ 得到 $z_{11}(*)$ 的右孩子 $z_{10}(f)$。

(7)当 $m=10$ 时,由 $g(Y,B,Z,C,9,9,7,7)$ 得到 $z_9(+)$ 的左孩子 $z_7(c)$,由 $g(Y,B,Z,C,11,11,8,8)$ 得到 $z_9(+)$ 的右孩子 $z_8(d)$。

方法三 根据算法 2.3,由先序遍历序列及其标志序列 X,A 和后序遍历序列及其标志序列 Z,C 构造二叉树的过程如下:

(1)由 $h(X,A,Z,C,1,13,1,13)$ 得到根结点为

$x_1(+)$ 和 $m = 3$。

(2)当 $m = 3$ 时,由 $h(X,A,Z,C,2,4,1,3)$ 得到 $x_1(+)$ 的左孩子为 $x_2(*)$ 和 $m = 1$,由 $h(X,A,Z,C,5,13,4,12)$ 得到 $x_1(+)$ 的右孩子为 $x_5(+)$ 和 $m = 6$。

(3)当 $m = 1$ 时,由 $h(X,A,Z,C,3,3,1,1)$ 得到 $x_2(*)$ 的左孩子 $x_3(a)$,而由 $h(X,A,Z,C,4,4,2,2)$ 得到 $x_2(*)$ 的右孩子 $x_4(b)$。

(4)当 $m = 6$ 时,由 $h(X,A,Z,C,6,8,4,6)$ 得到 $x_5(+)$ 的左孩子 $x_6(/)$ 和 $m = 4$,而由 $h(X,A,Z,C,9,13,7,11)$ 得到 $x_5(+)$ 的右孩子 $x_9(*)$ 和 $m = 9$。

(5)当 $m = 4$ 时,由 $h(X,A,Z,C,7,7,4,4)$ 得到 $x_6(/)$ 的左孩子 $x_7(a)$,而由 $h(X,A,Z,C,8,8,5,5)$ 得到 $x_6(/)$ 的右孩子 $x_8(b)$。

(6)当 $m = 9$ 时,由 $h(X,A,Z,C,10,12,7,9)$ 得到 $x_9(*)$ 的左孩子 $x_{10}(+)$ 和 $m = 7$,而由 $h(X,A,Z,C,13,13,10,10)$ 得到 $x_9(*)$ 的右孩子 $x_{13}(f)$。

(7)当 $m = 7$ 时,由 $h(X,A,Z,C,11,11,9,9)$ 得到 $x_{10}(+)$ 的左孩子 $x_{11}(c)$,由 $h(X,A,Z,C,12,12,8,8)$ 得到 $x_{10}(+)$ 的右孩子 $x_{12}(d)$。

5　结束语

如果遍历序列中存在相同元素,则现有算法无法根据该遍历序列构造二叉树。笔者通过引入标志序列对现有的递归算法进行了改进,使得改进后的算法适用于任何遍历序列(无论该序列有无相同元素)。算法仿真表明,该算法对具有相同元素的序列构造二叉树行之有效。由于本算法对遍历序列没有任何限制性要求,故其适用范围得到大大延伸。虽然递归算法结构清晰,方便算法设计,但是递归算法运行效率较低,其耗费的计算时间和占用的存储空间都比非递归算法要多,故本文所提问题的非递归算法是接下来的研究方向。此外,由层次遍历序列和其它遍历序列构造二叉树的算法尚未被研究,故也是未来的研究方向。

参考文献:

[1] 严蔚敏,吴伟民. 数据结构[M]. 北京:清华大学出版社,1992:125-131.

[2] Xiang L,Lawi A,Ushijima K. On constructing a binary tree from its traversals[J]. Research Reports on Information Science and Electrical Engineering of Kyushu University,2000, 5(1):13-18.

[3] Mikinen E . Constructing a binary tree efficiently from its traversals[J]. International Journal of Computer Mathematics, 2007,75(2):143-147.

[4] 唐自立. 基于遍历序列的构造树的算法[J]. 苏州大学学报(自然科学版),2011,27(3):26-29.

[5] 唐自立. 由先序序列和结点的左孩子情况构造严格二叉树的高效算法[J]. 南通大学学报(自然科学版),2013,12(1):9-13.

[6] 唐自立. 由后序序列和结点的双亲情况构造严格二叉树的非递归算法[J]. 南通职业大学学报,2014,28(4):93-98.

[7] 刘璐. 由遍历序列构造二叉树的非递归算法实现[J]. 衡水学院学报,2009,11(4):37-40.

[8] 王防修,周康. 基于二叉排序树的二叉树建立[J]. 武汉工业学院学报,2013,32(3):53-57.

[9] 李丽姝. 利用遍历序列还原二叉树算法的研究与实现[J]. 电大理工,2010,242(1):53-54.

[10] 赵刚,李昆. 由遍历序列确定二叉树的算法[J]. 南昌航空大学学报,2010,24(1):55-59.

[11] 朱涛. 基于遍历序列重构二叉结构树的分析[J]. 红河学院学报,2013,11(2):27-30.

[12] 化志章. 基于遍历序列恢复二叉树的新解法及其证明[J]. 江西师范大学学报,2013,37(3):268-272.