一种不用建造 Huffman 树的高效 Huffman 编码算法

李伟生 李 域 王 涛
(北京交通大学计算机与信息技术学院,北京 100044)

摘 要 Huffman 编码作为一种高效的不等长编码技术正日益广泛地在文本、图像、视频压缩及通信、密码等领域得到应用。为了更有效地利用内存空间、简化编码步骤和相关操作,首先研究了重建 Huffman 树所需要的信息,并提出通过对一类一维结构数组进行相关操作来获取上述信息的方法,然后利用这些信息,并依据提出的规范 Huffman 树的编码性质,便能直接得到 Huffman 编码。与传统的 Huffman 算法及近年来国内外文献中提出的改进算法相比,由于该方法不需要构造 Huffman 树,不仅使内存需求大大减少,而且编码步骤和相关操作更简洁,因而更利于程序的实现和移植。更重要的是,该算法思路为 Huffman 算法的研究和发展提供了新的途径。

关键词 规范 Huffman 树 结构数组 编码 编号
中图法分类号: TN918.81 **文献标识码**: A **文章编号**: 1006-8961(2005)03-0382-06

An Efficient Huffman Coding Algorithm of Non-creating Huffman Tree(NHTC)

LI Wei-sheng, LI Yu, WANG Tao
(*Institute of Computer & Information Technology, Beijing Jiaotong University, Beijing* 100044)

Abstract As an efficient and simple variable-length coding technique, Huffman codes are being widely used in text, image, video compression and so on. To reduce the requirement of the memory space, simplify encoding procedure, a layer information table(LIT) of Huffman tree is discussed to describe and reconstruct Huffman tree. An approach which adopts a special operating technique for a kind of structural array is designed to get data of LIT. A canonical Huffman tree(CAHT) is presented in this paper, and the existence, determination conditions and properties of encoding and decoding for CAHT are proved. Using LIT and encoding properties of CAHT, Huffman codes can be directly found. Comparing with traditional Huffman coding technique and other improved algorithms, the best advantage of the algorithm NHTC proposed in this paper is no need to create Huffman tree, so the operating process and coding procedure are more simplified and memory space requirement is reduced significantly.

Keywords canonical huffman tree, structural array, encode, serial number

1 引 言

自 1952 年提出 Huffman 编码以来,在过去的 50 年中,Huffman 算法一直得到国内外相关领域学者的关注,并取得了许多重要的进展。随着信息技术的发展,Huffman 编码作为高效的无损压缩的重要方法正日益广泛地在文本、图像、视频压缩及通信、密码等领域得到应用。

近年来,围绕提高 Huffman 算法的编码解码效率、有效利用存储空间、提高算法的压缩率等方面的研究已成为该领域的研究热点。文献[1]提出一种基于 Huffman 编码的进行无损图像压缩的方案,该方案旨在通过减少 Huffman 表的开销来获得高压缩

收稿日期:2004-01-15;**改回日期**:2004-08-02

第一作者简介:李伟生(1945 ~),男,教授,1968 年毕业于南开大学数学系,1988 ~ 1990 年赴德国工作研修。现主要研究方向为算法的设计与分析(图论算法、最优化算法、并行算法)、网络数据库,发表论文 80 余篇。E-mail: lws@computer.njtu.edu.cn

比。文献[2]提出由码长表(table of codelengths, TOCL)来构造单边 Huffman 树,并以此设计了简化的编码方案和树的簇集算法,从而不仅加速了对 Huffman 树中符号的检索,而且避免了由于 Huffman 树的稀疏性而造成的内存浪费。文献[3]由于利用树的父母-子女环存储结构来构造任意 k 元 Huffman 树和求 k 元 Huffman 编码,从而提高了空间利用率和加速了编码速度。文献[4]提出一种新的解码算法,其能一次性读入多位码流,以使绝大部分码字能一次性地读入,并立即解码;文献[5,6]分别提出高效的数组结构来重新表示 Huffman 树,并提出了相应的高效解码算法;文献[7]则在用于重建 Huffman 树的信息表中,除了保存必需的在 Huffman 树中依次出现的叶结点符号外,只保存环形叶结点的符号(带两个外部结点)的编码,由于环形结点数最多是叶结点的一半,所以存储空间需求低于上述几种数据结构;文献[8,9]提出从单边 Huffman 表(singleside grown Huffman table, SGHT)提取浓缩 Huffman 表(condensed Huffman table, CHT)的方法,同时采用了一种新的译码技术来使译码所需信息量进一步减少;文献[10]提出直接从码长表来对符号进行编码解码,在解码时同样采用(CHT)技术,从而不仅使算法速度明显加快,而且内存消耗明显减少;文献[11]提出一种算法,由于该算法使得编码后码流中 0、1 码元的分布概率(趋向)均等,因而改善了码流的信道传输性能。

但是笔者也注意到,上述改进算法和传统的 Huffman 算法一样,都必须以构造 Huffman 树为前提,而在具体实现时,又通常都要涉及到指针操作,有些算法还需要用到会造成内存很大浪费的递归技术。由于在编码、解码和对码字进行存储时,往往要频繁地对二进制码进行按位操作,这样又在一定程度上影响了算法的效率。为了更有效地利用内存空间,以便进一步简化编码步骤和相关操作,本文在对 Huffman 树及其构造算法进行深入分析的基础上,讨论了重建 Huffman 树所需要的关键信息,也即 Huffman 树中每一层上的叶结点数(层次表),同时对本文提出的规范 Huffman 树的确定条件和树中结点的序号、编号与编码的关系进行了研究,并按照建造 Huffman 树的算法思想,提出了通过对一类一维结构数组进行相应操作来获取层次表信息的算法。这样利用层次表和规范 Huffman 树的性质便能直接得到给定问题的 Huffman 编码。本文提出的方法由于不需要建造 Huffman 树,从而使内存需求大大减少,且编码步骤和相关操作更简洁快捷,也更利于程序的实现和移植。

2　Huffman 树层次表和规范 Huffman 树

2.1　Huffman 树的重建和层次表

为了说明本文的思路,先把有关概念和算法简述如下:

设二叉树 T 有分别带权为 $w_1,w_2,\cdots,w_t$ 的 t 片树叶,称 $W_T=\sum_{i=1}^{t}w_iL(v_i)$ 为 T 的权,其中 $L(v_i)$ 是根结点到叶结点 v_i 的通路长度。在所有的有 t 片树叶、分别带权为 $w_1,w_2,\cdots,w_t$ 的二叉树中,权最小的二叉树称为最优二叉树(Huffman 树)。设权重 $w_1\leqslant w_2\leqslant\cdots\leqslant w_t$,求最优二叉树的 Huffman 算法步骤如下:

(1)连接权为 w_1,w_2 的两片树叶,可得一分支点,赋权为 w_1+w_2。

(2)在 $w_1+w_2,w_3,\cdots,w_t$ 中选出两个最小的权,连接它们的顶点,即得到新的分支点和相应的权(权为两个顶点权之和)。

(3)重复步骤(2),直到形成 $t-1$ 个分支点,t 片树叶为止。

从 Huffman 树的定义和建造算法可知,建造 Huffman 树的关键是要确定待编码符号(叶结点)在树中的层次。若已求得一棵高度为 k 的 Huffman 树 T,则就可以计算出树中每一层上的叶子数,并可建立 Huffman 树层次表 $M=\{m_i,i=2,3,\cdots,k\}$。设根所在层次为第 1 层,其中 i 为树的层次,m_i 为第 i 层的叶子数。由 Huffman 树的定义可知,任一棵二叉树 T_1,如果它与 T 有相同的层次表,且只要把 T 的叶结点的权重按由小到大的次序从下到上、从左到右赋给树 T_1 的叶结点,则 T_1 也一定是 Huffman 树,同时任意交换同一层上叶结点的权重仍为 Huffman 树。由此就可以有各种不同的 Huffman 树。例如可以定义如下操作:在 Huffman 树的某一层上,若把一个非叶结点 u 的左、右子树剪下(Huffman 树中的非叶结点的度数一定为 2),分别作为同一层上某叶结点 v 的左、右子树,则称为从 u 到 v 的同层剪接。显然同层剪接不会改变树的层次表,不断进行上述变换就能得到各种形状不同的 Huffman 树。这就启示人们可以从中选取编码更为方便的 Huffman 树。

2.2 规范 Huffman 树及树中结点的编号

定义 1 设 T 是一棵 Huffman 树，T 中任一层上的结点排列顺序都是叶结点位于非叶结点的右侧，同一层上叶结点的权重从左向右按非降序排列，则称 T 为规范 Huffman 树。

给定叶结点的权重 $w_1 \leqslant w_2 \leqslant \cdots \leqslant w_n$，($n$ 为叶结点数)，设按 Huffman 算法建造的 Huffman 树为 T_1，高度为 k，若依次检查树 T_1 的第 i 层($i=1,2,\cdots,k-1$)，只要在某一层上有非叶结点(设为 u)在叶结点(设为 v)的右侧，则按 2.1 节定义的操作，从 u 到 v 进行同层剪接，直至每一层上所有叶结点均在非叶结点的右侧为止。设经过上述变换得到树 T，且 T 与 T_1 有相同的层次表。若把 T_1 树的叶结点的权重 $w_1 \leqslant w_2 \leqslant \cdots \leqslant w_n$ 由小到大按从下到上、从左到右的顺序依次赋给 T 的叶结点，则 T 就是规范 Huffman 树。由此得到以下定理：

定理 1 若给定叶结点的权重，则一定存在与之相应的规范 Huffman 树。

定义 2 一棵高度为 k，且有 2^k-1 个结点的二叉树称为满二叉树。若从满二叉树的根结点开始，自上而下、自左向右对树中的结点按 1,2,…连续编号，则该编号称为相应结点的序号。若在每一层从左到右分别按 0,1,2…对结点进行编号，则该编号称为结点在相应层的编号。

定义 3 任一棵规范 Huffman 树 T_i，都一定是一棵与其高度相同，且有相同根结点的满二叉树的子树，T_i 中任一结点的序号和编号定义为其在满二叉树中的序号和编号。

对满二叉树有如下熟知性质：

引理 1 设满二叉树中某结点序号为 m，则其左儿子序号为 $2m$，右儿子序号为 $2m+1$。

因为满二叉树中第 i 层的第 1 个结点序号为 2^{i-1}，所以有以下引理：

引理 2 满二叉树中第 i 层上($i=1,2,\cdots$)任一结点的编号加 2^{i-1} 就得到该结点的序号。

由上述引理容易得到以下引理：

引理 3 满二叉树中某结点的编号为 m，其左儿子编号为 $2m$，右儿子编号为 $2m+1$。

根据规范 Huffman 树与满二叉树中结点的序号和编号的一致性和上述引理，有以下定理：

定理 2 设 T 是高度为 k 的规范 Huffman 树，层次表为 M_i，其第 i 层上($i=1,2,\cdots,k$)最右边的结点编号为 R_i，最右边非叶结点编号为 N_i，则有

(1) $N_i = R_i - m_i$；

(2) $R_{i+1} = 2N_i + 1$；

因为 Huffman 树中除叶结点外，每个结点的度数一定为 2，所以第 2 层上最右边结点编号 $R_2=1$，若又已知规范 Huffman 树的层次表 M_i，则按定理 2 的公式就能从 R_2、m_2 开始，逐次递推求得树中每一层的 N_i 和 R_i，因而有如下推论：

推论 1 已知 Huffman 树的层次表，就能唯一确定一棵与之相应的规范 Huffman 树，进而可求得规范 Huffman 树上每一层上叶结点的编号。

3 Huffman 编码与结点序号的关系

在 Huffman 树中，从每个分支点引出的两条边上，左边的标 0，右边的标 1，而从树根到任一结点 v_i 的通路上各边的标号依次组成的 0、1 序列，则称为 v_i 的编码，而且树中所有叶结点编码的集合一定为{0、1}的二元最佳前缀码，也即 Huffman 编码。显然，由于树中结点的位置就决定了由根结点到该结点的通路，也就决定了该结点的编码，因而可以设法找出结点在树中的位置与编码的关系，以便求得一种更简便的编码方法。以下先讨论满二叉树中结点的编码与其序号的关系，然后推广到 Huffman 树的编码。如果满二叉树中任一结点的编码的求法仍遵循上述 Huffman 树中结点的编码规则，则依据规则显然有以下引理：

引理 4 满二叉树中第 i 层上($i=1,2,\cdots$)的结点编码长度为 $i-1$。

引理 5 在满二叉树中，第 p 层上任意两个结点 u 和 v，v 是位于 u 右侧的第 1 个结点，即

(1) 若 u 是其父结点的左儿子，则 u、v 的编码形式分别为 $a_1a_2\cdots a_{p-2}0$ 和 $a_1a_2\cdots a_{p-2}1$，其中 $a_1a_2\cdots a_{p-2}$ 是长度为 $p-2$ 的 0、1 序列；

(2) 若 u 是其父结点的右儿子，则 v 是其父结点的左儿子，v 的父结点也是位于 u 的父结点右侧的第 1 个结点，且 u、v 的编码形式分别为 $a_1a_2\cdots a_q01\cdots1$ 和 $a_1a_2\cdots a_q10\cdots0$，其中 $a_1a_2\cdots a_q$ 是离 u、v 最近的共同祖先结点 s 的编码，而且 u 的编码中最右边一位 0 的右侧 1 的位数和 v 的编码中最右边一位 1 的右侧 0 的位数均为 $p-q-2$。

证明 (1) 若 u 为父结点的左儿子，则 v 是 u 的兄弟结点，且一定为右儿子，由于它们有共同的父结点，显然有上述的编码形式；

(2) 设结点 u 的序号为 k,则由引理 1 可知,u 的父结点 u_1 的序号为 $(k-1)/2$,又显然由于 v 只可能为左儿子,其父结点 v_1 的序号为 $(k+1)/2$,所以 u_1 与 v_1 相邻。若 u_1 和 v_1 不是同一祖先结点 s 的左、右儿子,则 u_1 和 v_1 的编码关系类似于 u 和 v,依此类推,直至结点 s 为止,这样由 s 到 u 的通路上的编码为 $01\cdots1$(1 的位数为 $p-q-2$),由 s 到 v 的通路上的编码为 $10\cdots0$(0 的位数为 $p-q-2$),得证。

定理 3　满二叉树中,任一结点的序号的二进制表示去掉最高位,即为该结点的编码。

证明　不失一般性,设结点在第 p 层($p>1$),p 层最左边的结点序号为 2^{p-1},其二进制表示为 $100\cdots0$,其中 1 后面有 $p-1$ 个 0,结论成立。假设结点序号为 k 时结论成立,则证明结点序号为 $k+1$ 时也成立(以下简称为结点 k 和 $k+1$)。

(1) 若结点 k 为左儿子,此时结点 k 和结点 $k+1$ 为兄弟,且 k 和 $k+1$ 分别是偶数和奇数,则它们的二进制表示分别为 $1a_1a_2\cdots a_{p-2}0$ 和 $1a_1a_2\cdots a_{p-2}1$,由归纳假设可知,k 的编码为 $a_1a_2\cdots a_{p-2}0$,由引理 5(1)有 $k+1$ 的编码为 $a_1a_2\cdots a_{p-2}1$,得证。

(2) 若结点 k 是右儿子,则记结点 k 和结点 $k+1$ 的二进制表示分别为 $(k)_2$ 和 $(k+1)_2$,且有 $(k+1)_2=(k)_2+1$。由引理 5(2)和归纳假设知,结点 k 的编码有形式 $a_1a_2\cdots a_q011\cdots1$,$(k)_2=1a_1a_2\cdots a_q011\cdots1$,而结点 $k+1$ 的编码有形式 $a_1a_2\cdots a_q100\cdots0$,$(k+1)_2=1a_1a_2\cdots a_q011\cdots1+1=1a_1a_2\cdots a_q100\cdots0$,得证。

由引理 4 和引理 2 可以得到推论 2

推论 2　满二叉树中,第 i 层($i=1,2,\cdots$)上任一结点编号的二进制表示的左边增加若干位 0,使它成为 $i-1$ 位,就得到该结点的编码。

由定义 3 确定的规范 Huffman 树与满二叉树中结点的序号和编号的一致性可以得到定理 4。

定理 4　在高度为 k 的规范 Huffman 树中,若所有叶结点序号用二进制表示,则分别去掉其最高位组成的集合为 Huffman 编码。第 $i(i=1,2,\cdots)$ 层上叶结点的编号的二进制表示的左边分别增加若干位 0,使其成为 $i-1$ 位,则所有叶结点的这类二进制表示的集合为 Huffman 编码。

4　算法及计算实例

本节介绍不用建造 Huffman 树求 Huffman 编码的算法,并给出一个计算实例。

由第 2 节的讨论可知,依据 Huffman 树的层次表可以确定相应的规范 Huffman 树,并能计算出树中所有叶结点的编号。由于第 3 节已导出了规范 Huffmans 树的 Huffman 编码与叶结点编号的关系,所以只要求得 Huffman 树的层次表就能通过计算得到 Huffman 编码。

4.1　利用结构数组求 Huffman 树的层次表

设 $A[i]$ 是结构数组,$i=1,2,\cdots,n$,n 是待编码的符号数。A 的 4 个成员项 C_1、C_2、C_3、$link$ 分别为用于存放待编码符号、符号的权重、符号在 Huffman 树中的层次、链接下一个数组元素的下标。若有 $A[i_1].link=i_2$,$A[i_2].link=i_3,\cdots,A[i_m].link=0$,则称 $A[i_k]$ ($k=1,2,\cdots,m$)是以 $A[i_1]$ 为首结点,以 $A[i_m]$ 为尾结点的线性链。设有另一个以 $A[j_1]$ 为首结点的线性链,若令 $A[i_m].link=j_1$,则实现了由 $A[i_1]$ 到 $A[j_1]$ 的链接。

算法开始时,令 $A[i].link=0$,$A[i].C_3=1(i=1,2,\cdots,n)$,所有待编码符号的权重之和 $W=\sum_{i=1}^{n}A[i].C_2$,且有 $A[1].C_2\leqslant A[2].C_2\leqslant\cdots\leqslant A[n].C_2$,则算法步骤如下:

(1)在 $A[i](i=1,2,\cdots,n)$ 中选取权 C_2 最小的两个元素,设下标分别为 i_1 及 j_1,且有 $i_1<j_1$。令 $A[i_1].C_2=A[i_1].C_2+A[j_1].C_2$,$A[j_1].C_2=W$。

(2)分别使 $A[i_1]$、$A[j_1]$ 为首结点的线性链的每个元素的层数 C_3 加 1,并实现 $A[i_1]$ 和 $A[j_1]$ 的链接。

(3)重复步骤(1)和步骤(2),直至所有数组元素被链成一个线性链。

(4)根据各元素的层次求出第 i 层层次表 M_i,$i=2,3,\cdots,k$(k 为元素的最大层数)。

4.2　利用层次表求符号的编号

设 $M[i]$、$N[i]$、$R[i](i=2,3,\cdots,k)$ 分别表示规范 Huffman 树第 i 层上的叶结点数、最右边的非叶结点的编号、最右边的结点的编号。由定理 2 可知,以下算法用于计算规范 Huffman 树中,按从下到上、从左到右的顺序排列的叶结点的编号,并存入 $A[i].C_2$ 中($i=1,2,\cdots,n$)。

算法步骤如下:

(1)$i=2$,$R[i]=1$ (第 2 层最右边结点编号为 1);

(2)$N[i]=R[i]-M[i]$;

(3)当 $i<k$ 时,计算 $R[i+1]=2*N[i]+1$;

(4)$i=i+1$，当$i\leqslant k$重复步骤(2)～步骤(4)；

(5)for($j=1,i=k$; $i\geqslant 2$; $i--$)；

for($p=N[i]+1$; $p\leqslant R[i]$;$p++$)

$A[j++].C_2=p$。

4.3 算法实例

设有 8 个待编码的字符‘A’、‘C’、‘B’、‘F’、‘E’、‘G’、‘D’、‘H’，其权重分别为 3、5、7、8、11、14、23、29。按 4.1 节算法逐次计算的结果如下表所示：

(A,3,1,0) (C,5,1,0) (B,7,1,0) (F,8,1,0) (E,11,1,0) (G,14,1,0) (D,23,1,0) (H,29,1,0)

第 1 次:(A,8,2,2)(C,100,2,0)

第 2 次:(B,15,2,4)(F,100,2,0)

第 3 次:(A,19,3,2)(C,100,3,5)(E,100,2,0)

第 4 次:(B,29,3,4)(F,100,3,6)(G,100,2,0)

第 5 次:(A,42,4,2)(C,100,4,5)(E,100,3,7)(D,100,2,0)

第 6 次:(B,58,4,4)(F,100,4,6)(G,100,3,8)(H,100,2,0)

第 7 次:(A,100,5,2)(C,100,5,5)(B,100,5,4)(F,100,5,6)(E,100,4,7)(G,100,4,8)(D,23,3,3)(H,100,3,0)

计算所得到的层次表 $M[2]$、$M[3]$、$M[4]$、$M[5]$分别为 0、2、4。按 4.2 节算法可计算得到：

$$N[2]=1,\ R[2]=1$$
$$N[3]=1,\ R[3]=3$$
$$N[4]=1,\ R[4]=3$$
$$N[5]=-1,\ R[5]=3$$

由 4.2 节可计算得到(表中略去成员项 *link*)：

(A,0,5)(C,1,5) (B,2,5) (F,3,5) (E,2,4) (G,3,4) (D,2,3) (H,3,3)

根据定理 4 可直接得到如下编码表：

‘A’	‘C’	‘B’	‘F’	‘E’	‘G’	‘D’	‘H’
0000	0001	0010	0011	010	011	10	11

4.4 结果与讨论

在上例给定的条件下，按照传统 Huffman 算法构造的 Huffman 树和调整所得到的相应的规范 Huffman 树如图 1 和图 2 所示。

实验结果表明，本算法所得到的计算结果与对规范 Huffman 树按编码规则得到编码完全一致。

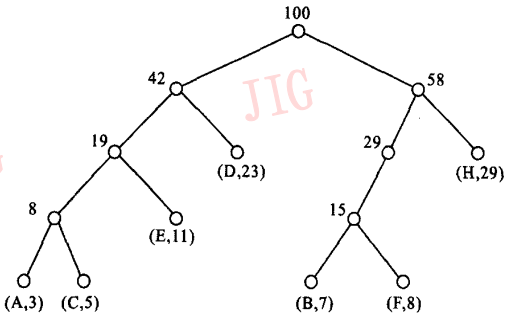

图 1 传统 Huffman 树

Fig. 1 Traditional Huffman tree

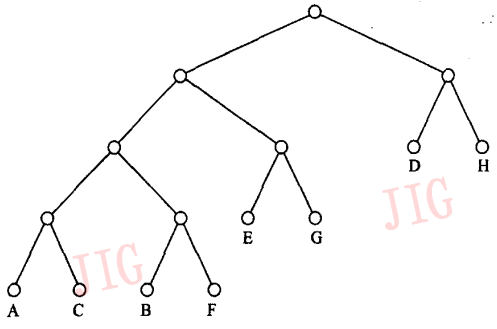

图 2 规范 Huffman 树

Fig. 2 Canonical Huffman tree

由于本算法不需要建造 Huffman 树，而是利用结构数组求 Huffman 树的层次表，所以算法的内存需求与现有算法相比大大减少。设待编码的符号数为 n，则本算法所需要的结构数组元素为 n，如果构造 Huffman 树，则其叶子数也为 n，非叶子结点数为 $n-1$，所以共需要 $2n-1$ 个结构类型的结点。另外在结构变量的定义中，在本算法的数组元素中，用于链接线性链的下标(结构的成员项)通常定义为一个字节的整型变量就能满足计算要求，而建造 Huffman 树则要求每个结点的成员项中需定义两个指针分别指向左、右儿子，如果是 16 位地址，则每个节点中仅指针成员项就需要 4 个字节。此外，由于本算法避开了指针操作，所以能在不支持指针的开发环境中实现，同时也有利于程序的移植。

在算法的步骤和实现方面，本算法除了与建造 Huffman 树一样，每次需要选取两个权重最小的元素外，其最主要的操作是线性链的元素增 1 操作，在得到层次表后，只要再通过简单运算就能得到节点的编号，并能立即得到符号的编码。在编码存储时，由于只需在相应编号的二进制表示前加若干个 0，

即可按码字直接存储,从而避免了对二进制数按位进行操作的繁琐,而且在建造 Huffman 树的算法中,除了与建树的有关操作外,传统的算法在对叶结点编码时,通常要引入叶结点到根节点进行上溯和下溯操作,有时还要引入递归过程,而文献[2,8]中的改进算法则在求得码长表 CLT 后,也需要从树的最低层开始,逐层对每个叶结点按所在层次和位置利用公式逐一计算来得到编码,计算中还要涉及到二进制数的按位操作和不同长度码字的存储等技术。显然本算法在计算步骤上要较其他算法简单易行。

5 结　论

本文在深入研究了 Huffman 算法的基础上,首选提出了 Huffman 树的层次表和规范 Huffman 树的概念,同时研究了规范 Huffman 树的编码性质,并提出了不用建造 Huffman 树求 Huffman 编码的方法和实现步骤。与现有算法相比,本算法不仅在内存需求上大大减少,而且计算步骤和程序实现更简单易行,也更利于程序的移植。此外,利用本文提出的规范 Huffman 树,还可以在减少 Huffman 编码表的长度,以提高压缩效率和设计更高效快捷的解码算法等方面进行进一步的研究和探索,并可望获得新的进展。最重要的是,本文提出的算法原理和设计思路为 Huffman 算法的进一步发展和改进提供了新的途径。

参考文献(References)

1 Hu Yu-chen, Chang Chin-chen. A new lossless compression scheme based on Huffman coding scheme for image compression[J]. Signal processing: Image Communication, 2000, **16**:367 ~ 372.

2 Reza Hashemian. Memory efficient and high-speed search Huffman coding[J]. IEEE Transactions on Communications, 1995, **43** (10): 2576 ~ 2581.

3 Wang Ling, Cheng Li. A new construction method for arbitrary Huffman trees with K elements [J]. Aeronautical Computer Technique, 1998, **28** (4): 12 ~ 15. [王玲, 陈莉. 任意 K 元 Huffman 树的新构造[J]. 航空计算技术, 1998, **28** (4): 12 ~ 15.]

4 Dong Pei-liang, Yu Ri-long, Liao Tian-kang, *et al*. A fast Huffman-decoding algorithm and its implementation of software and hardware [J]. Journal of Fudan University(Natural Science), 2002, **41**(2): 97 ~ 99. [董培良, 俞日龙, 廖天康等. 一种快速霍夫曼解码算法及其软硬件实现[J]. 复旦学报(自然科学版), 2002, **41** (2): 165 ~ 169.]

5 Chung Kuo-lian. Efficient Huffman decoding [J]. Information Processing Letters, 1997, **61**:97 ~ 99.

6 Chen Hong-chung, Wang Yue-li, Lan Yu-feng. A memory efficient and fast Huffman decoding algorithm [J]. Information Processing Letters, 1999, **69**:119 ~ 122.

7 Chowdhury Rezaul Alam, Kaykobad M, Irwin King. An efficient decoding technique for Huffman codes[J]. Information Processing Letters, 2002, **81**:305 ~ 308.

8 Reza Hashemian. Condensed table of Huffman coding, a new approach to efficient decoding [J]. IEEE Transactions on Communications, 2004, **52** (1):6 ~ 8.

9 Reza Hashemian. Condensed Huffman coding, a new efficient decoding technique[A]. In: The 2002 45th Midwest Symposium on Circuits and Systems[C], Tulsa Oklahoma, USA: IEEE Circuits and Systems Society and the School of Electrical and Computer Engineering at Oklahoma State University, 2003, **1**:228 ~ 231.

10 Reza Hashemian. Derect Huffman code and decoding using the table of code-lengths[A]. In: Pradip K. Srimani, edi. Proceeding of IEEE International Conference on Information Technology: Computers and Communications[C], Las Vegas, Nevada, USA: Computer Society of IEEE, 2003:237 ~ 241.

11 Lin Jia-yu, Liu Ying. A note on Huffman coding [J]. Acta Electronica Sinica, 2003, **31**(4):602 ~ 604. [林嘉宇, 刘莹. 关于 Huffman 编码的一个注记[J]. 电子学报, 2003, **31** (4):602 ~ 604.]