《离散数学》课程实验报告5 最优2元树在通信编码中的应用

**1.项目要求**

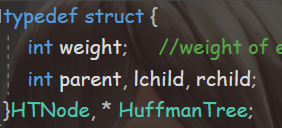
用户给出一组通信符号的个数和使用频率，要求输出各个通信符号对应的前缀码及哈夫曼编码。

**2.实验原理**

设2元树T有t片树叶v1,v2,…, vt, 树叶的权分别为w1, w2, …, wt, 称为T的权, 将每片树叶的权值和深度的乘积的总和记作W(T)。在所有t片树叶中, 带权w1, w2, …, wt 的 2元树中, 权最小的2元树称为最优2元树（哈夫曼树）。

在生成前缀码时，根据已构建好的哈夫曼树，对每个分支点, 若关联2条边, 则给左边标0, 右边标1; 若只关联1条边, 则可以给它标0(看作左边), 也可以标1(看作右边). 将从树根到每一片树叶的通路上标的数字组成的字符串记在树叶处, 所得的字符串构成一个前缀码。

**3.数据结构的设计**

本项目主要使用的数据结构为哈夫曼树。

树节点结构体（HTNode）有4个数据成员，weight存储该节点的权重，parent、lchild、rchild分别存储该节点的父节点、左孩子、右孩子在数组中的索引。

树（HuffmanTree）被定义为一个指向HTNode的指针，整个树将通过动态内存申请的方式建立。

**4.实验方法**

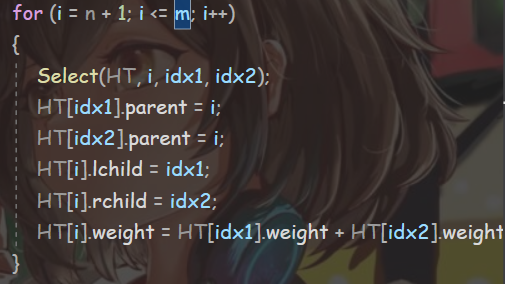
整个程序可以大致分为四部分：主函数，建立哈夫曼树，选择两个较小的节点和根据哈夫曼树输出叶节点的编码。

**4.1.主函数**

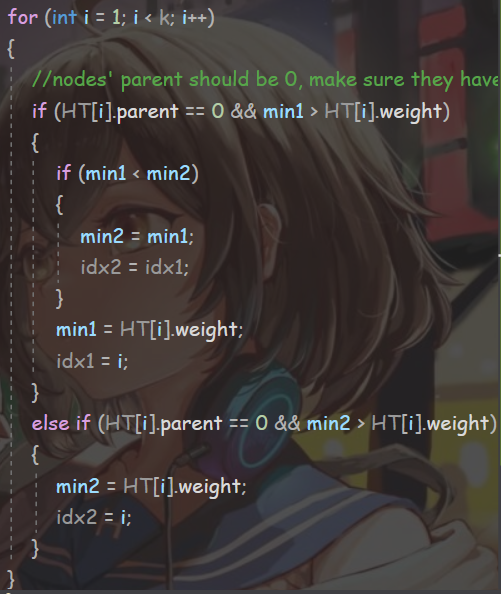
主函数负责读取用户输入，调用相关函数，计算结果并输出。

**4.2.建立哈夫曼树**

建立哈夫曼树通过函数void CreatHuffmanTree(HuffmanTree& HT, int n)实现。其中参数HT为待建立的哈夫曼树，参数n为哈夫曼树叶节点的个数。

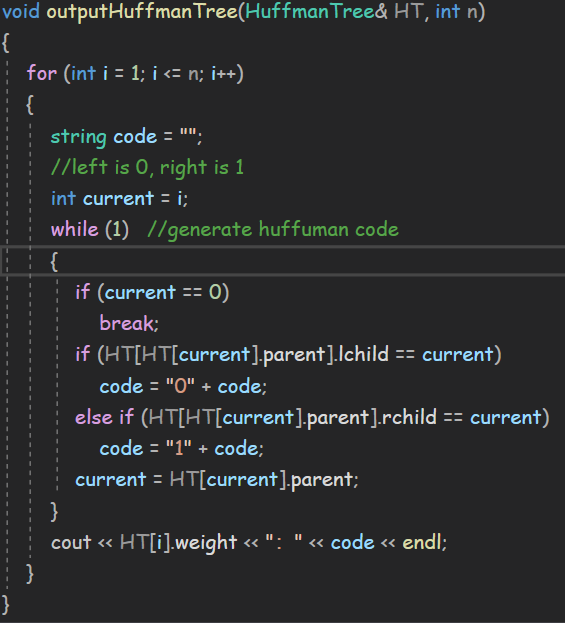
一棵哈夫曼树如果叶子节点有n个，则总节点数为2\*n-1，令m=2\*n-1方便调用，申请一段内存，用数组的方式存储哈夫曼树。在本题中，初始的叶子节点就是用户输入的分割完成后每块木头的长度，对应哈夫曼树的第1~n个元素。初始化哈夫曼树时将每个节点的父节点、左孩子、右孩子索引均置为0。从数组索引为n+1的位置开始建立哈夫曼树的非叶子节点，通过Select函数返回当前状态下权值最小的两个节点的索引，修改这两个节点对应的父节点的索引，修改父节点左右孩子节点的索引和权值，父节点的权值就是两个孩子节点的权值之和，关键代码段如右图。

**4.3.选择当前状态下两个较小的节点**

该功能通过函数void Select(HuffmanTree& HT, int k, int& idx1, int& idx2)实现。其中参数HT为待操作的哈夫曼树，参数k指的是要从哈夫曼树对应数组的第1~k个节点中寻找两个权值最小的节点，参数idx1和idx2为待返回的两个权值最小的节点的索引。

首先定义两个最小权值变量min1和min2，并将他们置为一个极大的数。值得一提的是，在选择最小节点之前必须保证该节点是可选择的，也就是保证该节点尚未被并入树中，这一点可以通过验证该节点是否有父节点的索引来判断，若该节点的数据成员parent值为0，则说明该节点尚未并入树中，是可选择的节点。然后遍历第1~k个节点，若发现权值比min1或min2小的节点，迭代更新min1、min2、idx1和idx2即可。

**4.4.输出哈夫曼编码**

该功能通过函数outputHuffmanTree(HuffmanTree& HT, int n)实现。每个叶节点的哈夫曼编码用一个字符串型变量code存储。遍历每一个叶节点（叶节点在数组HT中的索引为1~n），对每一个叶节点向上回溯，直到回溯到根节点为止，若该节点是其父节点的左孩子，则在已经生成的编码前添加字符“0”；反之，若该节点是其父节点的左孩子，则在已生成的编码前添加字符“1”。处理完一个叶节点后，即在控制台中输出节点的权重和对应的编码，其重要代码段如下：

**5.心得体会**

通过这次大作业，我进一步巩固了离散数学中与最优二元树和前缀码相关的知识。在计算机中以代码形式实现了从哈夫曼树的建立到哈夫曼编码生成的整个过程。一开始我本来想对老师给出的代码稍作改动，改正错误并使其更加健壮，但我发现给出的样例代码和我的思路几乎完全不一样，在对样例代码的测试过程中我也发现它只能输入13个通信符号的频率，不论输入的通信符号的频率个数比13多或少，输出都不符合要求。此外，每个通信符号的编码都会多出一个0。于是我决定以我自己的思路重构代码，在我的代码中可以灵活的根据输入的n值申请相应大小的内存空间，用户可以自由地选择要输入的通信符号频率的个数，也增加了程序的健壮性。实验效果可见下一页。

