# 第五章 基数排序及树同构算法

一．思想：基数排序是不基于元素比较的排序方法，它是按组成关键字各位的值来实现排序的。

关键字可以看作是一个k元组其中每个的值域为0至。关键字的大小用字典序排序。即

当且仅当以下两个条件之一成立：

1.存在整数j，使得，而对所有;

2.且对任意。

二．算法：

PROCEDURE Radixsort;

/\*输入序列，其中是k元组，的值域为0至。输出序列，它是的某个排列，满足\*/

BEGIN

把依次放入队列QUEUE;

FOR j := k DOWNTO 1 DO

BEGIN

FOR l := 0 TO DO

将Q[l]置为空队列；

WHILE QUEUE非空 DO

BEGIN

设是QUEUE中第一个元素；

将从QUEUE移到队列尾；

END;

FOR l := 0 TO DO

将队列的内容接到QUEUE队列尾；

END

END;

三．算法分析：

上述算法在最坏情况下时间复杂度为。最外层循环每执行一次需时间，共循环k次。

四．不等长的关键字的基数排序

思想：本算法对n个不等长的串进行排序，每个串中的每一个成份的值域为0至。第i个串的长度为，。

算法的本质是先将n个串按串长递减的顺序排列。设是最长串的长度。用基数排序，先对长为的串的第位排序，然后对长度大于等于的串的第位进行排序，依此下去。

五．排序方法

输入：一系列串(元组)，每个串的每一个成份的值域为0至。且的长度，设。

输出：的一个排列, 满足：。

1.首先，对的每个l，建立一个链表，该链表中的元素是出现在串中第l个成份。我们首先对的每个i和l，构造有序对，然后用基数排序将所有这些偶对按字典序排序，建立一个表。从左到右扫描这个表，建立个有序链表

NONEMPTY[l] 。NONEMPTY[l]按顺序包含整数j，使得

对某个i成立。

2.确定每个串长，然后对，建立链表LENGTH[l], LENGTH[l]中包含所有长为l的串。

3.用基数排序对串的每个成份排序，从第位开始，第i遍扫描之后，QUEUE队列中只包含长度大于等于的那些串。这些串已经按从到位排序。NONEMPTY表中列出了每一遍扫描时，哪些桶要被使用。这样可以加速计算各个桶的连接。

六．算法：

PROCEDURE Radix\_Sort2;

BEGIN

1. 建立空队列QUEUE;
2. FOR j := 0 TO DO 建立空队列Q[j];
3. FOR l := DOWNTO 1 DO

BEGIN

1. 将LENGTH[l]队列连接到QUEUE队列头；
2. WHILE QUEUE非空 DO

BEGIN

1. 设是QUEUE队列的第一个串；
2. 将从QUEUE移到队列尾；

END;

1. FOR NONEMPTY[l]中每个整数j DO

BEGIN

1. 将队列Q[j]连接到QUEUE队列尾；
2. 将Q[j]置为空队列；

END

END

END;

七．算法分析：

Radix\_Sort2算法最坏情况下时间复杂度为, 其中

。

步骤1需时间建立对偶，并需时间将它们排序。类似地，步骤2也需要时间。

现在考虑步骤3。设有个串含有第i个成份，并设含有第i个成份的串中，第i个成份有个不同值。即NONEMPTY[i]的长度为。

考虑上述算法中第3行的一个固定的l值，行的循环需时间，行的循环需时间。第4行需常数时间。行一遍需时间。整个循环共需

因为

故行的循环共需时间。第1行需常数时间，第2行需时间。从而步骤3也需时间。

八．判定两棵树同构的算法

1.定义：两棵树称为是同构的，是说：存在一个双射函数，使得对任意，当且仅当。

2.算法：

(1)对的所有叶子赋整数0。

(2)归纳地，设的第层的所有顶点已被赋了整数值。设是在第层的所有顶点按它们的整数值非递减的顺序排列所得的表。是同上方法对应于的表。

(3)从左到右扫描，按以下动作为在第层的每个非叶结点赋一个整数元组：对中每个顶点v，把v的整数值作为赋给v的父亲元组的下一个分量。完成这一步后，在第i层的每个非叶结点w有一个元组，其中是按非递减顺序排列的整数。它们是赋给w的儿子的整数。设是按上述方法赋给第i层结点的元组序列。

(4)对重复步骤(3)，并设是这样得到的在第i层顶点所赋的元组序列。

(5)用不等长串基数排序算法将排序，分别得到。

(6)如果不同，则停止。这两棵树不同构。否则，把1赋给在第i层中具有第一个不同元组的那些顶点，把2赋给在第i层中具有中第2个不同元组的那些顶点，依此下去。这些整数赋给在第i层的顶点，按值不减的顺序，构造中那些顶点的表，然后把在第i层的叶子放到表的前端。同样方法，构造在第i层顶点的表。返回步骤(3)，同计算，赋给在第层顶点的元组。

(7)如果的根被赋予同样的元组，则同构。

3.例子：见图5.1

4.算法分析：

给第i层顶点赋整数所需的时间与第层顶点数成比例，处理所有层结点需时间，处理叶结点所需时间至多为O(n)。故整个算法所需时间为O(n).

九．无根树的同构算法

1.树的中心

设表示G中顶点u到顶点v的最短路径的长度，称为u到v的距离。

定理5.1：一棵树或者恰有一个中心，或者有两个相邻的中心。

2.无根树的同构判定算法

算法：

(1)输入两棵树;

(2)找出的中心集合；

(3)若, 或反之，则不同构；

(4)若, 则以为根，判定是否同构；

(5)若, 则以的根，分别以为的根，判定是否同构；

(6)在第(4),(5)步中，若有一次判定出同构，则同构；否则不同构。

3.找树的中心的算法

算法：

(1)输入一棵树T;

(2)找出T的所有叶子的集合;

(3)

(4)FOR i := 1 TO k DO

BEGIN

(5) 在T中删除，并将与相邻的点的度数减1；

(6) 若的度数为1，则将加入;

END;

(7)若T中只剩一个顶点或两个相邻顶点，则找到T的中心或，算法中止；

否则，，返回步骤3.