

2.2 线性表的顺序表示

顺序表的定义

线性表的顺序存储又称顺序表

用一组地址连续的存储单元依次存储线性表中的数据元素

逻辑上相邻的两个元素在物理位置上也相邻

静态分配 数组的大小和空间已经固定，一旦空间占满，再加入新的数据将会产生溢出，程序就会崩溃

动态分配 存储数组的空间是在程序执行过程中通过动态存储分配语句分配的

一旦数据空间占满，就另外开辟一块更大的存储空间，用以替换原来的存储空间

注意 动态分配仍然是顺序存储结构，物理结构没有变化，依然是随机存取方式，只是分配的空间大小可以在运行时决定

随机访问，即通过首地址和元素序号可在时间O(1)内找到指定的元素

特点 存储密度高，每个结点只存储数据元素 与链表形成对比

逻辑上相邻的元素物理上也相邻，当执行插入和删除操作时，需要移动大量元素

顺序表上基本操作的实现

插入操作

由于顺序表的中元素的物理位置是相邻的，所以当插入新元素的时候就需要对表中的元素进行整体移动

最好情况：在表尾插入(i = n + 1),元素后移语句将不执行，时间复杂度为O(1)

最坏情况：在表头插入(i = 1),元素后移语句将执行n次，时间复杂度为O(n)

假设 p_i ($p_i = 1/(n+1)$)是在第i个位置上插入一个结点的概率

则在长度为n的线性表中插入一个结点时，移动结点的平均次数为

$$\sum_{i=1}^{n+1} p_i (n-i+1) = \sum_{i=1}^{n+1} \frac{1}{n+1} (n-i+1) = \frac{1}{n+1} \sum_{i=1}^{n+1} (n-i+1) = \frac{1}{n+1} \frac{n(n+1)}{2} = \frac{n}{2}$$

时间复杂度为 O(n)

删除操作

即为移动元素，对想要删除的元素进行覆盖

最好情况：删除表尾元素(i = n)，无须移动元素，时间复杂度为O(1)

最坏情况：删除表头元素(i = 1),需移动除第一个元素外的所有元素，时间复杂度为O(n)

假设 p_i ($p_i = 1/n$)是在第i个位置上删除一个结点的概率

则在长度为n的线性表中删除一个结点时，删除结点的平均次数为

$$\sum_{i=1}^n p_i (n-i) = \sum_{i=1}^n \frac{1}{n} (n-i) = \frac{1}{n} \sum_{i=1}^n (n-i) = \frac{1}{n} \frac{n(n-1)}{2} = \frac{n-1}{2}$$

时间复杂度为 O(n)

按值查找(顺序查找)

最好情况:查找的元素就在表头,仅需比较一次,时间复杂度为 O(1)

最坏情况:查找的元素在表尾(或不存在)时,需要比较n次,时间复杂度为O(n)

假设 p_i ($p_i = 1/n$)是查找的元素在第i ($1 \leq i \leq L.length$)个位置上的概率

则在长度为n的线性表中查找值为e的元素所需比较的平均次数为

$$\sum_{i=1}^n p_i \times i = \sum_{i=1}^n \frac{1}{n} \times i = \frac{1}{n} \frac{n(n+1)}{2} = \frac{n+1}{2}$$

时间复杂度为O(n)