希尔排序

希尔排序(缩小增量法)

基本思想：分割成若干个较小的子文件，对各个子文件分别进行直接插入排序，当文件达到基本有序时，再对整个文件进行一次直接插入排序。

对待排记录序列先作“宏观”调整，再作“微观”调整。

“宏观”调整，指的是，“跳跃式”的插入排序。

4. 排序过程：

首先将记录序列分成若干子序列，

然后分别对每个子序列进行直接插入排序，

最后待基本有序时，再进行一次直接插入排序

例如：将 n 个记录分成 d 个子序列：

{ R[1]，R[1+d]，R[1+2d]，…，R[1+kd] }

{ R[2]，R[2+d]，R[2+2d]，…，R[2+kd] }

…

{ R[d]，R[2d]，R[3d]，…，R[kd]，R[(k+1)d] }

其中，d 称为增量，它的值在排序过程中从大到小逐渐缩小，直至最后一趟排序减为 1。

例 初始：	49	38	65	97	76	13	27	48	55	4
取d1=5 一趟分组：	49	38	65	97	76	13	27	48	55	4
一趟排序：	13	27	48	55	4	49	38	65	97	76
取d2=3 二趟分组：	13	27	48	55	4	49	38	65	97	76
二趟排序：	13	4	48	38	27	49	55	65	97	76
取d3=1 三趟分组：	13	4	48	55	27	49	38	65	97	76
三趟排序：	4	13	27	38	48	49	55	65	76	97

算法描述

Ch8_3.txt

```
void shellsort(JD r[],int n,int d[], int T)
{  int i,j,k;
    JD x;
    k=0;
    //循环每一趟进行分组，组内进行简单插入排序
}
```

```
while(k<T)
  {  for(i=d[k]+1;i<=n;i++)
      //i为未排序记录的位置
      {  x=r[i];
          j=i-d[k];//j为本组i前面的记录位置
         while((j>0)&&(x.key<r[j].key))
          //组内简单插入排序
         {  r[j+d[k]]=r[j];
             j=j-d[k];
         }
         r[j+d[k]]=x;
      }
      k++;
}
```

希尔排序特点

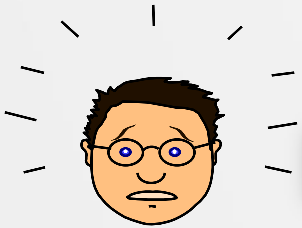
- 子序列的构成不是简单的“逐段分割”，而是将相隔某个增量的记录组成一个子序列
- 希尔排序可提高排序速度，因为
  - 分组后n值减小，n^2更小，而$T(n)=O(n^2)$,所以T(n)从总体上看是减小了
  - 关键字较小的记录跳跃式前移，在进行最后一趟增量为1的插入排序时，序列已基本有序
- 增量序列取法
  - 无除1以外的公因子
  - 最后一个增量值必须为1

最坏复杂度分析:

【定理】使用希尔增量的最坏时间复杂度为 $\Theta(N^2)$.

〖Example〗 A bad case:

	1	9	2	10	3	11	4	12	5	13	6	14	7	15	8	16
8-sort	1	9	2	10	3	11	4	12	5	13	6	14	7	15	8	16
4-sort	1	9	2	10	3	11	4	12	5	13	6	14	7	15	8	16
2-sort	1	9	2	10	3	11	4	12	5	13	6	14	7	15	8	16
1-sort	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16



P增量对不互质（relatively prime）. 因此小的增量可能没有效果

Hibbard’s 增量序列:

$h_k = 2^k - 1$ ---- 持续增量没有公共因子.

【Theorem】使用Hibbard’s 增量的最坏时间复杂度 $\Theta(N^{3/2})$.

希尔排序算法本身很简单，但复杂度分析很复杂. 他适合于中等数据量大小的排序（成千上万的数据量）.

Conjectures:

$T_{avg-Hibbard}(N) = O(N^{5/4})$

Sedgewick’s best sequence is {1, 5, 19, 41, 109, … } in which the terms are either of the form $9\times4^i - 9\times2^i + 1$ or $4^i - 3\times2^i + 1$. $T_{avg}(N) = O(N^{7/6})$ and $T_{worst}(N) = O(N^{4/3})$.

表 26－2　希尔排序算法的性能

时间复杂度			空间复杂度	稳定性	复杂性
平均情况	最坏情况	最好情况			
$O(n\log n)\sim O(n^2)$	$O(n\log n)\sim O(n^2)$		$O(1)$	不稳定	较复杂