一种 $O(n+n\log_2 m)$ 时间复杂度的排序算法*

A Sorting Algorithm with $O(n+n\log_2 m)$ Time Complexity

鲁晓明　　钟　诚
Lu Xiaoming　Zhong Cheng

(广西大学计算机与信息工程学院　南宁　530004)
(College of Computer and Information Engineering, Guangxi University, Nanning, 530004)

摘要　通过分析任意输入的 n 个数据的组成特性，设计一种 $O(n+n\log_2 m)$ 时间复杂度的排序算法，m 为原始输入数据序列中有序/逆有序的子序列个数，$1\leqslant m\leqslant n/2$。此排序算法的时间复杂性结果与输入数据的概率分布假设无关。

关键词　排序　算法复杂性　数据置换
中图法分类号　TP301.6

Abstract　By analyzing the characteristics of the n given input data, a sorting algorithm with $O(n+n\log_2 m)$ time complexity is presented, where m is the number of the sorted or conversely sorted sub-sequences in the original input sequence, $1\leqslant m\leqslant n/2$. The time complexity result is independent of the hypothesis of the probability distribution of the input data.

Key words　sorting, algorithm complexity, data swap

排序是计算机科学中最基本、最重要的研究问题之一[1]。由于排序问题不但本身十分有趣，而且具有十分重要的理论意义和广泛的应用价值，所以40多年它一直吸引着许多学者从串行或并行处理的角度不断探索并且取得了大量的极富创造性的成果[1~3]。

数据排序过程本质上是序列置换的过程。为了加快数据排序的速度，必须研究如何减少数据置换(交换)的次数。本文从分析原始输入数据组成的特性出发，找出其包含的有序子序列或者逆有序子序列，利用这些子序列的有序性，设计一个数据交换次数较少的排序算法。

1　算法的设计与分析

对于任意给定的2个数据 x 和 y，要么 $x\leqslant y$，要么 $x\geqslant y$。通过观察和分析，可以发现对于

2002-06-16 收稿。
* 国家"十五"863 计划课题(2001AA111004)的资助项目。

任意的输入数据序列 $A=\{a_1,a_2,\cdots,a_n\}$，其数据组成特征可以归纳为以下 3 种情形：

(1)$a_i \geqslant a_{i+1}(i=1,2,\cdots,n-1)$，即 A 为逆有序序列，其结构如图 1(a) 所示；

(2)$a_i \leqslant a_{i+1}(i=1,2,\cdots,n-1)$，即 A 为有序序列，其结构如图 1(b) 所示；

(3) 更一般的情形是序列 A 既非有序也非逆有序，但是总可以将 A 分成若干个子序列 A_1，$A_2,\cdots,A_m$，$1\leqslant m\leqslant n/2$，使得子序列 $A_j(j=1,2,\cdots,m)$ 一定满足上述(1)、(2)两种情况之一，其结构如图 1(c) 和(d) 所示。

假定将数据序列 A 按升序排列，则排序算法的思想如下：

第 1 步：对输入序列 A 进行扫描，寻找出满足情况(1) 或(2) 的子序列 $A_j,j=1,2,\cdots,m$，这样从整体来看，序列 A 就是一种升降起伏的波浪形的序列。我们把处于峰顶和谷底的数据分别称为峰点(子序列的最大元素) 和谷点(子序列的最小元素)；

第 2 步：依次扫描子序列 $A_j,j=1,2,\cdots,m$，若 A_j 为逆有序子序列，则将其调整为有序子序列，从而获得 m 个有序序列；

第 3 步：循环调用两路归并算法[1]，将 m 个有序子序列归并成一个长度为 n 的完整的有序序列。

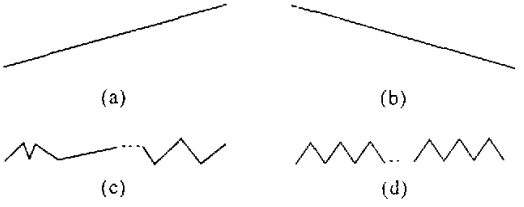

图 1 输入数据序列 A 的组成特征

(a) 有序的输入序列；(b) 逆有序的输入序列；(c) 具有峰顶点和峰谷点的一般输入序列；(d) 错落有致的输入序列。

我们将上述算法形式描述如下：

```
Algorithm  LZ-Sorting(A,n)
Begin
  m=0;//m 记有序/逆有序子序列的个数
(1) for i=2 to n do //找峰点和谷点
  if A[i-1]≤A[i] and A[i]>A[i+1] then
    begin
      m=m+1;B[m].p=1;//记峰点位置
      B[m].d=0;//表示找到一个有序子序列
    end
  else if A[i-1]≥A[i] and A[i]<A[i+1] then
  begin
    m=m+1;B[m].p=i;//记谷点位置
    B[m].d=1;//表示找到一个逆有序子序列
  end
(2) if m=1 and B[m].d=1 or m>1 then //将逆有序子序列调整为有序子序列
  begin
    B[0].p=0;
      for j=1 to m do
        if B[j].d=1 then
        for k=1 to do
```

```
            begin
              x=A[B[j-1].p+k];
              A[B[j-1].p+k]=A[B[j].p-(k-1)];
              A[B[j].p-(k-1)]=x;
            end;
      end;
    (3)if m>1 then //归并 m 个有序子序列成为一个长度为 n 的有序序列
        for  i=1  to  log2m  do
          for  j=1  to  m-1  STEP  2  do
            begin
            调用两路归并算法归并子序列 X[B[j-1].p+1..B[j].p]和 X[B[j].p+1..B[j+
1].p];
              B[j-1].p=B[j-1].p+1;
              B[j].p=B[j+1].p;
              m=m/2;
            end;
        End.
```

下面分析该算法的时间复杂度。设序列 A 的长度为 n，m 个子序列的长度分别记为 $L_1, L_2, \cdots, L_m$，则 $n=\sum_{i=1}^{m} L_i$。显然，算法第(1)步扫描数据序列所需的时间为 $T_s(n)=O(n)$，第(2)步将序列 A 中的逆有序子序列调整成有序子序列所需的时间最多为 $T_u(n)=O(n)$。当输入序列 A 为有序或者逆有序序列时，算法无需执行第(3)步，这时排序所需的时间为 $O(n)$。

对于一般情形的输入序列，算法需要执行第(3)步的子序列归并工作。因为共有 m 个有序子序列，所以算法需要进行 $\log_2 m$ 遍的归并工作。在每遍归并过程中，参与归并的数据个数为 n，所以每遍归并所需的时间为 $O(n)$。这样，算法步(3)所需的时间为

$$T_m(n)=\sum_{j=1}^{\log_2 m} O(n)=O(n\log_2 m).$$

因此，整个排序算法所需的时间为：

$$\begin{aligned} T(n) &= T_s c(n)+T_s t(n)+T_m(n) \\ &= O(n)+O(n)+O(n\log_2 m) \\ &= O(n+n\log_2 m). \end{aligned}$$

表1给出了本文排序算法的时间复杂度与已知的主要排序算法的时间复杂度的比较结果。

表 1　排序算法时间复杂度的对比

算　法	平均时间	最坏时间
Quick 排序	$O(n\log_2 n)$	$O(n^2)$
Heap 排序	$O(n\log_2 n)$	$O(n\log_2 n)$
归并排序	$O(n\log_2 n)$	$O(n\log_2 n)$
本文的排序算法	$O(n\log_2 n), 1\leqslant m\leqslant n/2$	$O(n\log_2(n/2))$

2　结束语

我们从连绵不断、此起彼伏的群山轮廓以及时涨时落的股市交易数据曲线得到启迪，通过分析任意输入的 n 个数据的

组成特性，设计一种时间复杂度为$O(n+n\log_2 m)$的排序算法，m为原始输入数据序列中有序/逆有序的子序列个数，$1 \leqslant m \leqslant n/2$。此排序时间复杂性结果与输入数据的概率分布假设无关。在最坏情形下，本文的排序算法的时间复杂度与现有的排序方法相同。但是，由于在大多数情形下m是小于n的，所以本文给出的排序算法在工程实践中将更有优势。

本文算法的设计思想和冒泡排序、基数排序算法的设计思想有着异曲同工之妙——它们都来源于对客观事物和现实生活的观察与思考。

下一步的工作是将本文的算法并行化，并探讨其在各种并行计算环境下的实现。

参考文献

1 Knuth D E. The art of computer programming: sorting and searching, vol 3. 2nd ed. Reading, Mass.: Addison-Wesley, 1998.

2 苏德富，钟 诚. 计算机算法设计与分析. 北京：电子工业出版社，2001.

3 陈国良. 并行算法的设计与分析. 北京：高等教育出版社，1994.

（责任编辑：黎贞崇）

（上接第150页）

论打好基础，它将在数据挖掘、综合评判等方面有着重要的应用.

参考文献

1 Yager R R, Filve D P. Approximate clustering via the mountain method. IEEE Transactions on Systems Man and Cybernetics, 1994, (24): 1279～1284.

2 Dunn J C A. Fuzzy Relative of the ISODATE process and its use in detecting compact well-separated clusters. Cybern, 1973, 3(3): 32～57.

3 Ng R, Han J E. Efficient and effective clustering methods for spatial data Minning. Department of Computer Science, University of Birtish Columbia, 1994.

4 Chiu S L. Fuzzy model identification based on cluster estimation. Journal of Intelligent and Fuzzy Systems, 1994, 2(3): 54～58.

5 张智星，孙春在，水谷英二[日]著. 神经一模糊和软计算. 西安：西安交通大学出版社，2000.

（责任编辑：黎贞崇）