一种新的数组排序法

罗石麟,唐晓兵
(空军工程大学 导弹学院,陕西 三原 713800)

摘 要:提出一种新的数组排序法,分析了该算法在最坏情况下的计算复杂度。该算法比C. A. R. Hoare算法要快,操作简便。

关键词:数组排序;C. A. R. Hoare排序法;计算复杂度
中图分类号:TP301.6　　**文献标识码**:A　　**文章编号**:1009-3516(2002)02-0071-03

将某一给定数组中的数按其大小严格排序,是许多科学部门经常要遇到的问题。当数组中的数很多时,或者某一计算过程要反复进行排序运算时,或者在实时性要求很高的操作中需要排序时,排序方法的计算复杂度便显得尤为重要,所以对排序方法的研究时而有之。1962年英国的C. A. R. Hoare提出一种采用递归策略的快速排序法,由于它完成排序的平均计算复杂度仅为 $O(n\log_2 n)$,因而被誉为"20世纪最好的10个算法"之一[1]。本文提出一种数组排序算法,在最坏情况下其计算复杂度最多不超过 $O(n\log_2 n)$。

1 数组排序问题

用 $a_i \propto a_j \rightarrow a_k$ 表示将 a_i 与 a_j 比较大小,得到其中大数为 $a_k \subset \{a_i, a_j\}$。现设有一 n 个数的数组

$$\{a_1, a_2, \cdots, a_n\} \tag{1}$$

1)作比较如下:$a_1 \propto a_2 \rightarrow a_{k1}$,$a_{k1} \propto a_3 \rightarrow a_{k2}$,$\cdots$,$a_{k_{n-2}} \propto a_n \rightarrow a_{k_{n-1}} = \max\limits_{1 \le i \le n} \{a_i\}$ 共比较 $n-1$ 次。此时数组中最大数已沉底,不必再列入比较。

2)去除最大数 $a_{k_{n-1}}$ 剩余的数仍记为 $\{a_1, a_2, \cdots, a_{n-1}\}$,重复上述做法,得第二大数,并排入倒数第二,不必再列入比较,这里共比较 $n-2$ 次。在比较过程中,把遇到的等值数视为一个数,一同往下比较(比较其中一个即可),这样等值数自然排到了一起。

如此做下去,完成全部排序,最多共比较

$$(n-1)+(n-2)+\cdots+2+1=\frac{[(n-1)+1](n-1)}{2}=\frac{n(n-1)}{2} \tag{2}$$

次。不过,这样的运算次数是平凡的,方法不可取。

2 数组排序方法的改进

现在我们把数组(1)分成两组:Ⅰ及Ⅱ。为了方便起见,不妨暂设 n 为偶数,将(1)分为相等个数的两组,每组$\frac{n}{2}$个数。对每一组数,按上节所述方法排序,则每组需比较

$$\frac{\frac{n}{2}(\frac{n}{2}-1)}{2}=\frac{n(n-1)}{8} \tag{3}$$

收稿日期:2001-09-18
作者简介:罗石麟(1939-),男,湖南长沙人,教授,主要从事防空运筹学、算法设计与分析等研究.

次,两组共需比较$\frac{n(n-1)}{4}$次。

两组数分别完成排序后,将这两个排序队按如下方法合并为一个排序队:

记 $$\text{Ⅰ}: a_1^{(1)}, a_2^{(1)}, \cdots, a_{\frac{n}{2}}^{(1)}$$

$$\text{Ⅱ}: a_1^{(2)}, a_2^{(2)}, \cdots, a_{\frac{n}{2}}^{(2)}$$

都是大数排在后,小数排在前。将两组中最大数作比较,不失一般性,设 $a_{\frac{n}{2}}^{(1)} \leqslant a_{\frac{n}{2}}^{(2)}$,则

$$a_{\frac{n}{2}}^{(1)} \propto a_{\frac{n}{2}}^{(2)} \rightarrow a_{\frac{n}{2}}^{(2)}$$

再作 $a_{\frac{n}{2}-1}^{(1)} \propto a_{\frac{n}{2}-1}^{(2)}$,如此比较下去,直到 $a_{\frac{n}{2}}^{(1)}$ 在Ⅱ组中的序位确定为止,设为

$$a_1^{(2)}, a_2^{(2)}, \cdots, a_{\frac{n}{2}-l}^{(2)}, a_{\frac{n}{2}}^{(1)}, a_{\frac{n}{2}-l+1}^{(2)}, \cdots, a_{\frac{n}{2}}^{(2)}$$

继续作 $a_{\frac{n}{2}-1}^{(1)} \propto a_{\frac{n}{2}-l}^{(2)}, \cdots$。显然,如此合并完成排序最多需比较 $n-1$ 次[2]。这是因为二组数合并排序后需要比较次数最多的最坏顺序为

$$a_1^{(1)}, a_1^{(2)}, \cdots, a_{\frac{n}{2}-1}^{(1)}, a_{\frac{n}{2}-1}^{(2)}, a_{\frac{n}{2}}^{(1)}, a_{\frac{n}{2}}^{(2)} \tag{4}$$

此时,除首尾两数 $a_1^{(1)}$ 和 $a_{\frac{n}{2}}^{(2)}$ 以外,其它的数都要与相关的数比较两次后才能定位,故全部排序完共需比较次数为

$$2\left(\frac{n}{2}-1\right)+1=n-1 \tag{5}$$

综上所述,这样分两组分别比较后合并排序共需比较次数最多为

$$\frac{n(n-2)}{4}+(n-1)=\frac{n^2+2n-4}{4} \tag{6}$$

可见,此方法比上节的方法比较次数少,事实上:

$$\frac{n(n-1)}{2}-\frac{n^2+2n-4}{4}=\frac{n^2-4n+4}{4}=\frac{(n-2)^2}{4} \tag{7}$$

几乎要少一半的比较次数。

3 一种新的数组排序法

下面我们采用极端的方式,对数组(1)每任取两个数划为一组,共约$\frac{n}{2}$组。若 n 为奇数,则将 a_n 暂搁置一边,作为单独一个特殊数组,等到下述适当时机与某一组数进行比较合并。

现不妨设 n 为偶数,则数组(1)共分为$\frac{n}{2}$组,设为

$$\{a_1, a_2\}, \{a_3, a_4\}, \cdots, \{a_{n-1}, a_n\}$$

各组内两两比较排序,共需比较$\frac{n}{2}$次。然后将这些数组按上节方法两两比较排序合并。若此时$\lfloor \frac{n}{2} \rfloor$为奇数,则留下最后一组与 a_n 合并;否则,仍将 a_n 置一边待适当时机列入比较合并排序。若$\lfloor \frac{n}{2} \rfloor$为偶数,则合并时将合并成$\lfloor \frac{n}{4} \rfloor$组,每组最多需要比较 3 次。

现同时考虑可能的奇、偶两种情况,视为$\lceil \frac{n}{4} \rceil$组,则上述合并时需要比较次数不多于 $3\lceil \frac{n}{4} \rceil$次。

进一步,再依上法两两合并,每两组合并最多需比较 7 次,……,如此合并下去,直到完成数组(1)的排序,这样,显然对数组(1)排序的总比较次数不超过

$$A_1=\lceil \frac{n}{2} \rceil+3\lceil \frac{n}{4} \rceil+7\lceil \frac{n}{8} \rceil+\cdots+(2^{\lfloor \log_2 n \rfloor}-1)\lceil \frac{n}{2^{\lfloor \log_2 n \rfloor}} \rceil \tag{8}$$

上式共$[\log_2 n]-1$项,其中每一项内的向上取整均为考虑到出现奇数个数组的情况,又显然

$$A_1<(\lfloor \log_2 n \rfloor-1)n<n\log_2 n$$

所以本文提出的排序算法在最坏情况下其复杂度为 $O(n\log_2 n)$。

对于本文算法的平均时间复杂度,我们在 Athon600 MHz 微机上用 C 语言编程,选取 n = 1 000,2 000,5 000,8 000,10 000,12 000,每次随机产生 n 个数组元素,分别用本文算法和 Hoare 算法进行排序,各运行 1 000 次,统计出其平均排序时间,结果如表 1 所示。

表 1　两种算法的平均排序时间对比

数组大小 N	1 000	2 000	5 000	8 000	10 000	12 000
Hoare 算法(ms)	0.76	1.33	3.78	6.24	8.12	9.77
本文算法(ms)	0.60	1.10	3.45	5.60	7.30	8.85

由此可以看出,本文算法比 Hoare 算法速度稍快。

从计算复杂度看,C. A. R. Hoare 的快速排序法的平均性态为 $O(n\log_2 n)$,C. A. R. Hoare 算法也正是因其平均性态好而闻名的,但每次分割的支点最好为中位数,而搜索中位数并非一件容易的事,需要额外的"簿记"计算量,在最坏情况下,其计算复杂度可能变为 $O(n^2)$[3~4]。而本文的算法即使在最坏情况下其复杂度也仅为 $O(n\log_2 n)$,且不需要其它"簿记"操作,算法实现简单易行,不失为一个较好的排序算法。

参考文献:

[1] CIPRA. B A. The Best of the 20th Century: Editors Name Top 10 Algorithms[J]. SIAM NEWS, 2000, 33(4): 1 - 3.
[2] 张文明. 电子计算机软件:算法设计与分析[M]. 三原:空军导弹学院,1992.
[3] 顾立尧. 算法设计分析的理论与方法[M]. 上海:上海交通大学出版社. 1989.
[4] 卢开澄. 计算机算法导引——设计与分析[M]. 北京:清华大学出版社. 1996

(编辑:田新华)

A New Algorithm for Array Sorting

LUO Shi - lin,　TANG Xiao - bing
(The Missile Institute, Air Force Engineering University, Sanyuan 713800, China)

Abstract: This paper presents a new algorithm for array sorting and an analysis of its complexity of calculation in the worst situation. This algorithm is simple in operation and can run more quickly than C. A. R Hoare's Quichsort algorithm.

Keywords: array sorting; C. A. R. Hoare algorithm; complexity of calculation