**常用排序算法时间复杂度和空间复杂度**

转载 2015年05月17日 20:50:08

* 标签：
* [排序算法](http://so.csdn.net/so/search/s.do?q=%E6%8E%92%E5%BA%8F%E7%AE%97%E6%B3%95&t=blog) /
* [插入排序](http://so.csdn.net/so/search/s.do?q=%E6%8F%92%E5%85%A5%E6%8E%92%E5%BA%8F&t=blog) /
* [希尔排序](http://so.csdn.net/so/search/s.do?q=%E5%B8%8C%E5%B0%94%E6%8E%92%E5%BA%8F&t=blog) /
* [快速排序](http://so.csdn.net/so/search/s.do?q=%E5%BF%AB%E9%80%9F%E6%8E%92%E5%BA%8F&t=blog) /
* [归并排序](http://so.csdn.net/so/search/s.do?q=%E5%BD%92%E5%B9%B6%E6%8E%92%E5%BA%8F&t=blog)
* 11930

摘自维基百科： http://zh.wikipedia.org/wiki/%E6%8E%92%E5%BA%8F%E7%AE%97%E6%B3%95#.E7.A8.B3.E5.AE.9A.E6.80.A7

在[计算机科学](http://zh.wikipedia.org/wiki/%E8%AE%A1%E7%AE%97%E6%9C%BA%E7%A7%91%E5%AD%A6" \o "计算机科学" \t "_blank)所使用的排序算法通常被分类为：

* 计算的[时间复杂度](http://zh.wikipedia.org/wiki/%E8%A8%88%E7%AE%97%E8%A4%87%E9%9B%9C%E6%80%A7%E7%90%86%E8%AB%96" \o "计算复杂性理论" \t "_blank)（最差、平均、和最好性能），依据列表（list）的大小(n)。一般而言，好的性能是[O](http://zh.wikipedia.org/wiki/%E5%A4%A7O%E7%AC%A6%E5%8F%B7" \o "大O符号" \t "_blank)(n log n)，且坏的性能是O(n2)。对于一个排序理想的性能是[O](http://zh.wikipedia.org/wiki/%E5%A4%A7O%E7%AC%A6%E5%8F%B7" \o "大O符号" \t "_blank)(n)。仅使用一个抽象关键比较运算的排序算法总平均上总是至少需要O(n logn)。
* 存储器使用量（以及其他电脑资源的使用）
* 稳定性：**稳定排序算法**会让原本有相等键值的纪录维持相对次序。也就是如果一个排序算法是**稳定**的，当有两个相等键值的纪录R和S，且在原本的列表中R出现在S之前，在排序过的列表中R也将会是在S之前。
* 依据排序的方法：插入、交换、选择、合并等等。

稳定性[[编辑](http://zh.wikipedia.org/w/index.php?title=%E6%8E%92%E5%BA%8F%E7%AE%97%E6%B3%95&action=edit&section=2)]

当相等的元素是无法分辨的，比如像是整数，稳定性并不是一个问题。然而，假设以下的数对将要以他们的第一个数字来排序。

(4, 1) (3, 1) (3, 7)（5, 6）

在这个状况下，有可能产生两种不同的结果，一个是让相等键值的纪录维持相对的次序，而另外一个则没有：

(3, 1) (3, 7) (4, 1) (5, 6) （維持次序）

(3, 7) (3, 1) (4, 1) (5, 6) （次序被改變）

不稳定排序算法可能会在相等的键值中改变纪录的相对次序，但是稳定排序算法从来不会如此。不稳定排序算法可以被特别地实现为稳定。作这件事情的一个方式是人工扩充键值的比较，如此在其他方面相同键值的两个对象间之比较，（比如上面的比较中加入第二个标准：第二个键值的大小）就会被决定使用在原先数据次序中的条目，当作一个同分决赛。然而，要记住这种次序通常牵涉到额外的空间负担。

排序算法列表[[编辑](http://zh.wikipedia.org/w/index.php?title=%E6%8E%92%E5%BA%8F%E7%AE%97%E6%B3%95&action=edit&section=3)]

在这个表格中，n是要被排序的纪录数量以及k是不同键值的数量。

稳定的排序[[编辑](http://zh.wikipedia.org/w/index.php?title=%E6%8E%92%E5%BA%8F%E7%AE%97%E6%B3%95&action=edit&section=4)]

* [冒泡排序](http://zh.wikipedia.org/wiki/%E5%86%92%E6%B3%A1%E6%8E%92%E5%BA%8F)（bubble sort）— O(n2)
* [鸡尾酒排序](http://zh.wikipedia.org/wiki/%E9%B8%A1%E5%B0%BE%E9%85%92%E6%8E%92%E5%BA%8F)（cocktail sort）—O(n2)
* [插入排序](http://zh.wikipedia.org/wiki/%E6%8F%92%E5%85%A5%E6%8E%92%E5%BA%8F)（insertion sort）—O(n2)
* [桶排序](http://zh.wikipedia.org/wiki/%E6%A1%B6%E6%8E%92%E5%BA%8F)（bucket sort）—O(n);需要O(k)额外空间
* [计数排序](http://zh.wikipedia.org/wiki/%E8%AE%A1%E6%95%B0%E6%8E%92%E5%BA%8F)（counting sort）—O(n+k);需要O(n+k)额外空间
* [归并排序](http://zh.wikipedia.org/wiki/%E5%BD%92%E5%B9%B6%E6%8E%92%E5%BA%8F)（merge sort）—O(n log n);需要O(n)额外空间
* 原地[归并排序](http://zh.wikipedia.org/wiki/%E5%BD%92%E5%B9%B6%E6%8E%92%E5%BA%8F)— O(n2)
* [二叉排序树](http://zh.wikipedia.org/wiki/%E4%BA%8C%E5%8F%89%E6%8E%92%E5%BA%8F%E6%A0%91)排序（binary tree sort）— O(n log n)期望时间; O(n2)最坏时间;需要O(n)额外空间
* [鸽巢排序](http://zh.wikipedia.org/wiki/%E9%B8%BD%E5%B7%A2%E6%8E%92%E5%BA%8F)（pigeonhole sort）—O(n+k);需要O(k)额外空间
* [基数排序](http://zh.wikipedia.org/wiki/%E5%9F%BA%E6%95%B0%E6%8E%92%E5%BA%8F)（radix sort）—O(n·k);需要O(n)额外空间
* [侏儒排序](http://zh.wikipedia.org/w/index.php?title=%E4%BE%8F%E5%84%92%E6%8E%92%E5%BA%8F&action=edit&redlink=1)（gnome sort）— O(n2)
* [图书馆排序](http://zh.wikipedia.org/w/index.php?title=%E5%9B%BE%E4%B9%A6%E9%A6%86%E6%8E%92%E5%BA%8F&action=edit&redlink=1)（library sort）— 时间复杂度通常是O(n log n),需要(1+ε)n额外空间

不稳定的排序[[编辑](http://zh.wikipedia.org/w/index.php?title=%E6%8E%92%E5%BA%8F%E7%AE%97%E6%B3%95&action=edit&section=5)]

* [选择排序](http://zh.wikipedia.org/wiki/%E9%81%B8%E6%93%87%E6%8E%92%E5%BA%8F)（selection sort）—O(n2)
* [希尔排序](http://zh.wikipedia.org/wiki/%E5%B8%8C%E5%B0%94%E6%8E%92%E5%BA%8F)（shell sort）—O(n log2 n)如果使用最佳的现在版本
* [Clover排序算法](http://zh.wikipedia.org/w/index.php?title=Clover%E6%8E%92%E5%BA%8F%E7%AE%97%E6%B3%95&action=edit&redlink=1)（Clover sort）—O(n)期望时间，O（n^2/2）最坏情况
* [梳排序](http://zh.wikipedia.org/wiki/%E6%A2%B3%E6%8E%92%E5%BA%8F)— O(n log n)
* [堆排序](http://zh.wikipedia.org/wiki/%E5%A0%86%E6%8E%92%E5%BA%8F)（heap sort）—O(n log n)
* [平滑排序](http://zh.wikipedia.org/w/index.php?title=%E5%B9%B3%E6%BB%91%E6%8E%92%E5%BA%8F&action=edit&redlink=1)（smooth sort）— O(n log n)
* [快速排序](http://zh.wikipedia.org/wiki/%E5%BF%AB%E9%80%9F%E6%8E%92%E5%BA%8F)（quick sort）—O(n log n)期望时间, O(n2)最坏情况;对于大的、乱数列表一般相信是最快的已知排序
* [内省排序](http://zh.wikipedia.org/wiki/%E5%86%85%E7%9C%81%E6%8E%92%E5%BA%8F)（introsort）—O(n log n)
* [耐心排序](http://zh.wikipedia.org/w/index.php?title=%E8%80%90%E5%BF%83%E6%8E%92%E5%BA%8F&action=edit&redlink=1)（patience sort）—O(n log n + k)最坏情况时间，需要额外的O(n + k)空间，也需要找到[最长的递增子序列](http://zh.wikipedia.org/w/index.php?title=%E6%9C%80%E9%95%B7%E7%9A%84%E9%81%9E%E5%A2%9E%E5%AD%90%E5%BA%8F%E5%88%97&action=edit&redlink=1" \o "最长的递增子序列（页面不存在）" \t "_blank)（longest increasing subsequence）

不实用的排序[[编辑](http://zh.wikipedia.org/w/index.php?title=%E6%8E%92%E5%BA%8F%E7%AE%97%E6%B3%95&action=edit&section=6)]

* [Bogo排序](http://zh.wikipedia.org/wiki/Bogo%E6%8E%92%E5%BA%8F)— O(n × n!)，最坏的情况下期望时间为无穷。
* [Stupid排序](http://zh.wikipedia.org/w/index.php?title=Stupid%E6%8E%92%E5%BA%8F&action=edit&redlink=1)—O(n3);递归版本需要O(n2)额外存储器
* [珠排序](http://zh.wikipedia.org/wiki/%E7%8F%A0%E6%8E%92%E5%BA%8F)（bead sort）— O(n) or O(√n),但需要特别的硬件
* [煎饼排序](http://zh.wikipedia.org/w/index.php?title=%E7%85%8E%E9%A4%85%E6%8E%92%E5%BA%8F&action=edit&redlink=1)—O(n),但需要特别的硬件
* [臭皮匠排序](http://zh.wikipedia.org/wiki/%E8%87%AD%E7%9A%AE%E5%8C%A0%E6%8E%92%E5%BA%8F)（stooge sort）算法简单，但需要约n^2.7的时间

平均时间复杂度[[编辑](http://zh.wikipedia.org/w/index.php?title=%E6%8E%92%E5%BA%8F%E7%AE%97%E6%B3%95&action=edit&section=7)]

平均时间复杂度由高到低为：

* [冒泡排序](http://zh.wikipedia.org/wiki/%E5%86%92%E6%B3%A1%E6%8E%92%E5%BA%8F)O(n2)
* [选择排序](http://zh.wikipedia.org/wiki/%E9%80%89%E6%8B%A9%E6%8E%92%E5%BA%8F)O(n2)
* [插入排序](http://zh.wikipedia.org/wiki/%E6%8F%92%E5%85%A5%E6%8E%92%E5%BA%8F)O(n2)
* [希尔排序](http://zh.wikipedia.org/wiki/%E5%B8%8C%E5%B0%94%E6%8E%92%E5%BA%8F)O(n1.25)
* [堆排序](http://zh.wikipedia.org/wiki/%E5%A0%86%E6%8E%92%E5%BA%8F)O(n log n)
* [归并排序](http://zh.wikipedia.org/wiki/%E5%BD%92%E5%B9%B6%E6%8E%92%E5%BA%8F)O(n log n)
* [快速排序](http://zh.wikipedia.org/wiki/%E5%BF%AB%E9%80%9F%E6%8E%92%E5%BA%8F)O(n log n)
* [基数排序](http://zh.wikipedia.org/wiki/%E5%9F%BA%E6%95%B0%E6%8E%92%E5%BA%8F)O(n)

说明：虽然完全逆序的情况下，快速排序会降到选择排序的速度，不过从概率角度来说（参考信息学理论，和概率学），不对算法做编程上优化时，快速排序的平均速度比堆排序要快一些。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **名称** | **数据对象** | **稳定性** | **时间复杂度** | | **空间复杂度** | **描述** |
| **平均** | **最坏** |
| [冒泡排序](http://zh.wikipedia.org/wiki/%E6%B0%A3%E6%B3%A1%E6%8E%92%E5%BA%8F) | 数组 | http://upload.wikimedia.org/wikipedia/commons/thumb/f/fb/Yes_check.svg/15px-Yes_check.svg.png | O(n^2) | | O(1) | （无序区，有序区）。从无序区通过交换找出最大元素放到有序区前端。 |
| [选择排序](http://zh.wikipedia.org/wiki/%E9%80%89%E6%8B%A9%E6%8E%92%E5%BA%8F) | 数组 | http://upload.wikimedia.org/wikipedia/commons/thumb/a/a2/X_mark.svg/15px-X_mark.svg.png | O(n^2) | | O(1) | （有序区，无序区）。在无序区里找一个最小的元素跟在有序区的后面。对数组：比较得多，换得少。 |
| 链表 | http://upload.wikimedia.org/wikipedia/commons/thumb/f/fb/Yes_check.svg/15px-Yes_check.svg.png |
| [插入排序](http://zh.wikipedia.org/wiki/%E6%8F%92%E5%85%A5%E6%8E%92%E5%BA%8F) | 数组、链表 | http://upload.wikimedia.org/wikipedia/commons/thumb/f/fb/Yes_check.svg/15px-Yes_check.svg.png | O(n^2) | | O(1) | （有序区，无序区）。把无序区的第一个元素插入到有序区的合适的位置。对数组：比较得少，换得多。 |
| [堆排序](http://zh.wikipedia.org/wiki/%E5%A0%86%E6%8E%92%E5%BA%8F) | 数组 | http://upload.wikimedia.org/wikipedia/commons/thumb/a/a2/X_mark.svg/15px-X_mark.svg.png | O(n\log n) | | O(1) | （最大堆，有序区）。从堆顶把根卸出来放在有序区之前，再恢复堆。 |
| [归并排序](http://zh.wikipedia.org/wiki/%E5%BD%92%E5%B9%B6%E6%8E%92%E5%BA%8F) | 数组 | http://upload.wikimedia.org/wikipedia/commons/thumb/f/fb/Yes_check.svg/15px-Yes_check.svg.png | O(n\log n) | | O(n) +O(\log n) ，如果不是从下到上 | 把数据分为两段，从两段中逐个选最小的元素移入新数据段的末尾。可从上到下或从下到上进行。 |
| 链表 | O(1) |
| [快速排序](http://zh.wikipedia.org/wiki/%E5%BF%AB%E9%80%9F%E6%8E%92%E5%BA%8F) | 数组 | http://upload.wikimedia.org/wikipedia/commons/thumb/a/a2/X_mark.svg/15px-X_mark.svg.png | O(n\log n) | O(n^2) | O(\log n) ,O(n) | （小数，枢纽元，大数）。 |
| [希尔排序](http://zh.wikipedia.org/wiki/%E5%B8%8C%E5%B0%94%E6%8E%92%E5%BA%8F) | 数组 | http://upload.wikimedia.org/wikipedia/commons/thumb/a/a2/X_mark.svg/15px-X_mark.svg.png | O(n\log^2n) | O(n^2) | O(1) | 每一轮按照事先决定的间隔进行插入排序，间隔会依次缩小，最后一次一定要是1。 |
|  | | | | | |  |
| [计数排序](http://zh.wikipedia.org/wiki/%E8%AE%A1%E6%95%B0%E6%8E%92%E5%BA%8F) | 数组、链表 | http://upload.wikimedia.org/wikipedia/commons/thumb/f/fb/Yes_check.svg/15px-Yes_check.svg.png | O(n+m) | | O(n+m) | 统计小于等于该元素值的元素的个数i，于是该元素就放在目标数组的索引i位（i≥0）。 |
| [桶排序](http://zh.wikipedia.org/wiki/%E6%A1%B6%E6%8E%92%E5%BA%8F) | 数组、链表 | http://upload.wikimedia.org/wikipedia/commons/thumb/f/fb/Yes_check.svg/15px-Yes_check.svg.png | O(n) | | O(m) | 将值为i的元素放入i号桶，最后依次把桶里的元素倒出来。 |
| [基数排序](http://zh.wikipedia.org/wiki/%E5%9F%BA%E6%95%B0%E6%8E%92%E5%BA%8F) | 数组、链表 | http://upload.wikimedia.org/wikipedia/commons/thumb/f/fb/Yes_check.svg/15px-Yes_check.svg.png | O(k\times n) | O(n^2) |  | 一种多关键字的排序算法，可用桶排序实现。 |

* + 均按从小到大排列
  + k代表数值中的"数位"个数
  + n代表数据规模
  + m代表数据的最大值减最小值