数据结构

概念

- 数据 -- 凡是可以输入计算机并被计算机处理的东西.联系人,学生,整形数据,字符串,浮点数等等.
- 结构 -- 数据与数据之间的特定关系
- 逻辑关系
    - 线性结构(队列)
    - 非线性(树型结构, 图型机构)
- 物理关系
    - 顺序存储(数据元素与数据元素之间的地址是连续的)
    - 链式存储(数据元素与数据元素之间的地址是不连续的)

时间复杂度

算法

算法即为解决问题的步骤（解决问题的有限执行序列）

好的算法

耗时少，占用空间少

- 算法评估：
    1. 数据量相同
    2. 算法语句执行时间默认相同
    3. 评估算法的语句执行次数

大O记法

> 在计算机科学]中，算法的**时间复杂度**（Time complexity）是一个函数，它定性描述该算法的运行时间。这是一个代表算法输入值的字符串的长度的函数。时间复杂度常用大O符号表述，不包括这个函数的低阶项和首项系数。使用这种方式时，时间复杂度可被称为是渐近的，亦即考察输入值大小趋近无穷时的情况。

使用

大O符号在分析算法效率的时候非常有用。举个例子，解决一个规模为n的问题所花费的时间（或者所需步骤的数目）可以表示为：$T(n) = 4n^2 - 2n + 2$。当n增大时，n^2项将开始占主导地位，而其他各项可以被忽略。举例说明：当$n = 500$，$4n^2$项是$2n$项的1000倍大，因此在大多数场合下，省略后者对表达式的值的影响将是可以忽略不计的。

进一步看，如果我们与任一其他级的表达式比较，n^2项的系数也是无关紧要的。例如：一个包含n^3或n^2项的表达式，即使$T(n) = 1,000,000 \cdot n^2$，假定$U(n) = n^3$，一旦$n$增长到大于1,000,000，后者就会一直超越前者 $T(1,000,000) = 1,000,000^3 = U(1,000,000)$。

这样，针对第一个例子

$$T(n) = 4n^2 - 2n + 2$$

，大O符号就记下剩余的部分，写作：

$$T(n) \in \mathrm{O}(n^2)$$

或

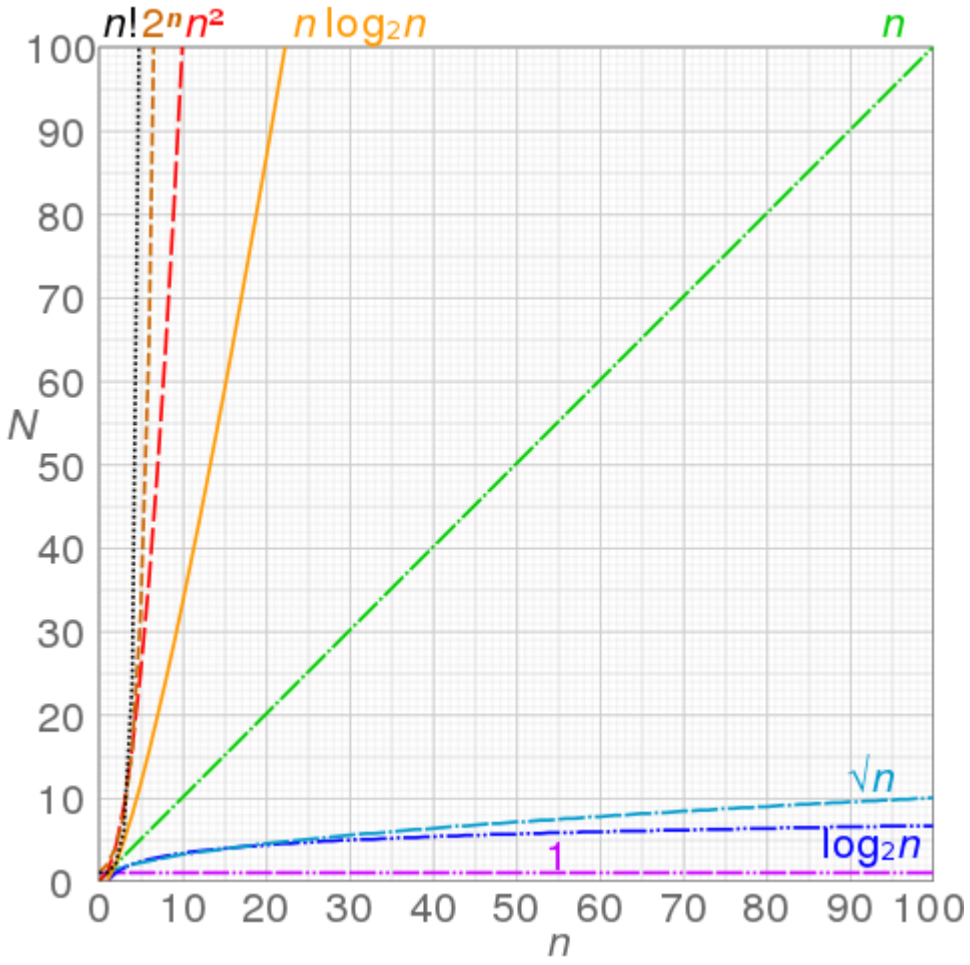
$$T(n) = \mathrm{O}(n^2)$$

并且我们就说该算法具有n^2阶（平方阶）的**时间复杂度**。

常见时间复杂度列表

名称	$T(n)$	算法举例
常数	$O(1)$	判断一个二进制数的奇偶
对数	$O(\log n)$	二分搜索
线性	$O(n)$	无序数组搜索
线性对数	$O(n \log n)$	快速排序
平方	$O(n^2)$	冒泡排序
指数	$O(2^{n^\epsilon})$，对任意的ε > 0	.....
阶乘	*O*(*n*!)	.....

常见时间复杂度比较


n!
2ⁿn²
n log₂n
n
√n
log₂n
1
N
n
0
10
20
30
40
50
60
70
80
90
100