**算法复杂度**

　　同一问题可用不同[算法](http://baike.baidu.com/view/7420.htm)解决，而一个算法的质量优劣将影响到算法乃至程序的效率。算法分析的目的在于选择合适算法和改进算法。一个算法的评价主要从时间复杂度和空间复杂度来考虑。

　　1、时间复杂度

　　（1）时间频度

　　一个算法执行所耗费的时间，从理论上是不能算出来的，必须上机运行测试才能知道。但我们不可能也没有必要对每个算法都上机测试，只需知道哪个算法花费的时间多，哪个算法花费的时间少就可以了。并且一个算法花费的时间与算法中语句的执行次数成正比例，哪个算法中语句执行次数多，它花费时间就多。一个算法中的语句执行次数称为语句频度或时间频度。记为T(n)。

　　（2）时间复杂度

　　在刚才提到的时间频度中，n称为问题的规模，当n不断变化时，时间频度T(n)也会不断变化。但有时我们想知道它变化时呈现什么规律。为此，我们引入时间复杂度概念。

　　一般情况下，算法中基本操作重复执行的次数是问题规模n的某个函数，用T(n)表示，若有某个辅助函数f(n),使得当n趋近于无穷大时，T（n)/f(n)的极限值为不等于零的常数，则称f(n)是T(n)的同数量级函数。记作T(n)=O(f(n)),称O(f(n)) 为算法的渐进时间复杂度，简称时间复杂度。

　　在各种不同算法中，若算法中语句执行次数为一个常数，则时间复杂度为O(1),另外，在时间频度不相同时，时间复杂度有可能相同，如T(n)=n2+3n+4与T(n)=4n2+2n+1它们的频度不同，但时间复杂度相同，都为O(n2)。

　　按数量级递增排列，常见的时间复杂度有：

　　常数阶O(1),对数阶O(log2n),线性阶O(n),

　　线性对数阶O(nlog2n),平方阶O(n2)，立方阶O(n3),...，

　　k次方阶O(nk),指数阶O(2n)。随着问题规模n的不断增大，上述时间复杂度不断增大，算法的执行效率越低。

　　2、空间复杂度

　　与时间复杂度类似，空间复杂度是指算法在计算机内执行时所需存储空间的度量。记作:

　　S(n)=O(f(n))

　　我们一般所讨论的是除正常占用内存开销外的辅助存储单元规模。讨论方法与时间复杂度类似，不再赘述。