算法实例复杂度分析

**题目：**求出最大子列和

4种解法

解法1：暴力求解，复杂度

（1）确定左端点，移动右端点，求得此序列最大子列和

（2）移动左端点，确定前两序列最大子列和

（3）不断向右移动左端点，直到找到整个数组得最大子列和

三层嵌套循环，故时间复杂度为

解法2：暴力求解优化解法，复杂度

int sum=0;

for(int i=0;i<n;i++)

int basesum=0; //左端移动时基础和清零

for(int j=i;j<n;j++){

basesum=basesum+a[j];//右端移动时在前一次计算基础上加上a[j]

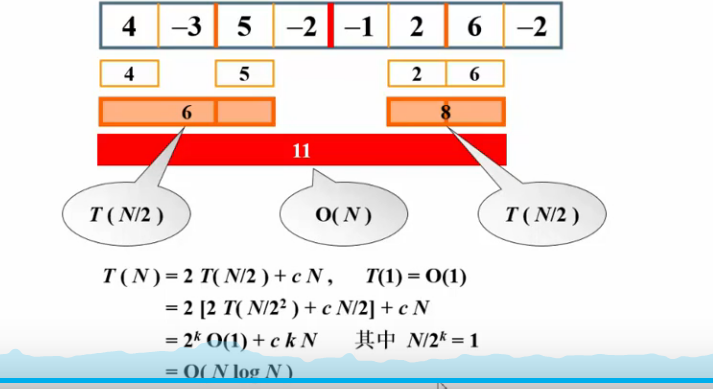
if(basesum>sum){

sum=basesum; //基础和大于当前最大和就把基础和作为新的最大和

}

}

解法3：分而治之，递归求解，复杂度



（1）思路：将数组从中间一分为二，分别计算左边最大和、右边最大和以及跨越边界的最大子列和，三者进行比较，找到最大值就是最终结果。

（2）求解方式

左/右：同样一分为二，直到分解为只有一个元素。

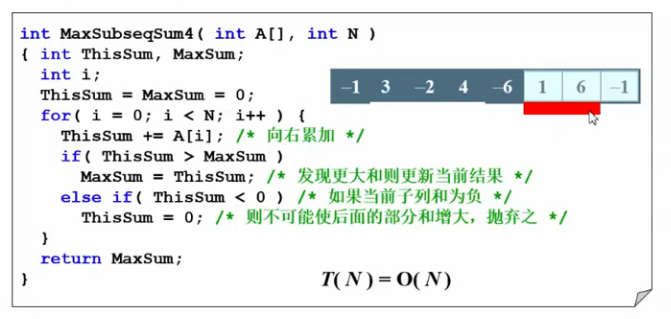
跨越边界：从中间往左右扫描，中间起的左边最大和与右边最大和相加。

（3）复杂度分析

分解可得)+C\*n=

分解到，*T(n)=，*时间复杂度为nlogn

解法4：在线处理，时间复杂度O(n)



（1）思路：

int sum=0;

int thissum=0;

for(int i=0;i<n;i++){

thissum+=a[i];

if(thissum>sum){

sum=thissum; //当前和大于总和，当前和成为新的最大和

}

else if(thissum<0){

thissum=0; //当前和不能起增加作用，则舍弃

}

}

（2）在线处理：即时地处理每个输入数据