

# Analysis Algorithm

## [Example] Sum from i to j

### § 3 Compare the Algorithms

[[Example]] Given (possibly negative) integers  $A_1, A_2, \dots, A_N$ , find the maximum value of  $\sum_{k=i}^j A_k$ .

#### Algorithm 1

```
int MaxSubsequenceSum ( const int A[ ], int N )
{
    int ThisSum, MaxSum, i, j, k;
    /* 1*/ MaxSum = 0; /* initialize the maximum sum */
    /* 2*/ for( i = 0; i < N; i++ ) /* start from A[ i ] */
    /* 3*/     for( j = i; j < N; j++ ) { /* end at A[ j ] */
    /* 4*/         ThisSum = 0;
    /* 5*/         for( k = i; k <= j; k++ )
    /* 6*/             ThisSum += A[ k ]; /* sum from A[ i ] to A[ j ] */
    /* 7*/         if ( ThisSum > MaxSum )
    /* 8*/             MaxSum = ThisSum; /* update max sum */
    /* 9*/     } /* end for-j and for-i */
    return MaxSum;
}
```

$T(N) = O(N^3)$

- 记忆前项和

## Algorithm 2

```

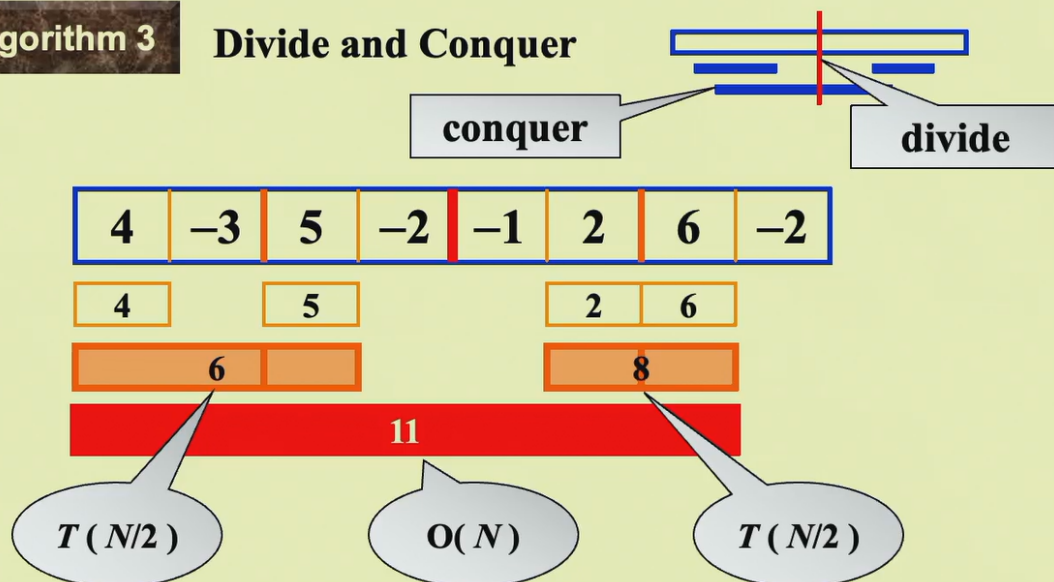
int MaxSubsequenceSum ( const int A[ ], int N )
{
    int ThisSum, MaxSum, i, j;
    /* 1*/   MaxSum = 0; /* initialize the maximum sum */
    /* 2*/   for( i = 0; i < N; i++ ) { /* start from A[ i ] */
    /* 3*/       ThisSum = 0;
    /* 4*/       for( j = i; j < N; j++ ) { /* end at A[ j ] */
    /* 5*/           ThisSum += A[ j ]; /* sum from A[ i ] to A[ j ] */
    /* 6*/           if ( ThisSum > MaxSum )
    /* 7*/               MaxSum = ThisSum; /* update max sum */
    /* 8*/       } /* end for-j */
    } /* end for-i */
    return MaxSum;
}

```

- Divide and Conquer (分治法)

### 8.3 Compare the Algorithms

## Algorithm 3 Divide and Conquer



$$\begin{aligned}
 T(N) &= 2T(N/2) + cN, \quad T(1) = O(1) \\
 &= 2[2T(N/2^2) + cN/2] + cN \\
 &= 2^k O(1) + c k N \quad \text{where } N/2^k = 1 \\
 &= O(N \log N)
 \end{aligned}$$

Also true for  
 $N \neq 2^k$

如何解 $T(N)$ : 直接带入

- Online Algorithm

**Algorithm 4****On-line Algorithm**

```
int MaxSubsequenceSum( const int A[ ], int N )
{
    int ThisSum, MaxSum, j;
    /* 1*/ ThisSum = MaxSum = 0;
    /* 2*/ for ( j = 0; j < N; j++ ) {
    /* 3*/     ThisSum += A[ j ];
    /* 4*/     if ( ThisSum > MaxSum )
    /* 5*/         MaxSum = ThisSum;
    /* 6*/     else if ( ThisSum < 0 )
    /* 7*/         ThisSum = 0;
    } /* end for-j */
    /* 8*/ return MaxSum;
}
```

-1	3	-2	4	-6	1	6	-1
----	---	----	---	----	---	---	----

$$T(N) = O(N)$$

**A[ ]** is scanned **once** only.

负值不可能产生更大的和，因此置零重新计算

意义：

1. 快
2. 空间复杂度低
3. 随时停止即为当前最佳结果

## [Example] Binary search

```
int BinarySearch ( const ElementType A[ ],
                  ElementType X, int N )
{
    int Low, Mid, High;
    /* 1*/ Low = 0; High = N - 1;
    /* 2*/ while ( Low <= High ) {
    /* 3*/     Mid = ( Low + High ) / 2;
    /* 4*/     if ( A[ Mid ] < X )
    /* 5*/         Low = Mid + 1;
        else
    /* 6*/         if ( A[ Mid ] > X )
    /* 7*/             High = Mid - 1;
        else
    /* 8*/             return Mid; /* Found */
    } /* end while */
    /* 9*/ return NotFound; /* NotFound is defined as -1 */
}
```

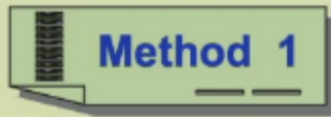
Home work:  
Self-study Euclid's Algorithm  
and Exponentiation

辗转相除法

指数算法 (分治+递归)

# Check your Analysis

## § 5 Checking Your Analysis

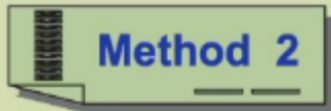


When  $T(N) = O(N)$ , check if  $T(2N)/T(N) \approx 2$

When  $T(N) = O(N^2)$ , check if  $T(2N)/T(N) \approx 4$

When  $T(N) = O(N^3)$ , check if  $T(2N)/T(N) \approx 8$

... ..



When  $T(N) = O(f(N))$ , check if

$$\lim_{N \rightarrow \infty} \frac{T(N)}{f(N)} \approx \text{Constant}$$

Read the example given on p.28 (Figures 2.12 & 2.13).