链滴

算法分析（二）最大子序列和问题

作者：Angonger

原文链接：https://ld246.com/article/1501158520529

来源网站：链滴

算法分析（二）最大子序列和问题

标签（空格分隔）： Java 算法

给定一个整数序列，a₀,a₁,a₂,.....a_n(以为负数),求其中最大的子序列和。如果所有整数都是负数，那么最大子序列和为0。

<table>
<tr>
<th colspan="4">前半部分</th><th colspan="4">后半部分</th>
</tr>
<tr>
<td>4</td>
<td>-3</td>
<td>5</td>
<td>-2</td>
<td>-1</td>
<td>2</td>
<td>6</td>
<td>-2</td>
</tr>
</table>

1. 解法一（最容易想到的）： O(N ³)

遍历所有的子序列，找出最大子序列

```
public static int maxSubSum(int a[]) {
        int maxSum = 0;
        for (int i = 0; i < a.length; i++) {//N次 决定子序列的起始位
            for (int j = i; j < a.length; j++) {//N次 决定子序列的结束位
                int thisSum = 0;
                for (int k = i; k <= j; k++) {//N次 累加判断
                    thisSum += a[k];
                    if (thisSum > maxSum) {
                        maxSum = thisSum;
                    }
                }
            }
        }
        return maxSum;
    }
```

2. 解法二（从某个位置开始的子序列，求最大）： O(N ²)

```
    public static int maxSubSum2(int a[]) {
        int maxSum = 0;
        for (int i = 0; i < a.length; i++) {//N次 起始位
            int thisSum = 0;
            for (int j = i; j < a.length; j++) {//N次 剩下的进行累加
                thisSum += a[j];
```

```
            if (thisSum > maxSum) {
                maxSum = thisSum;
            }
        }
    }
    return maxSum;
}
```

3. 解法三（分治策略（divide-and-conquer）O(Nlog(N))）

最大子序列和必定出现在三个位置之一（前半部分、后半部分、横跨前半部分后半部分）。分别求这种位置的最大子序列和。

```
private static int maxSubSum3(int a[], int leftIndex, int rightIndex) {
    if (leftIndex == rightIndex) {// 基准情况
        if (a[leftIndex] > 0) {
            return a[leftIndex];
        } else {
            return 0;
        }
    }
    int midIndex = (leftIndex + rightIndex) / 2;
    int maxLeftSubSum = maxSubSum3(a, leftIndex, midIndex);
    int maxRightSubSum = maxSubSum3(a, midIndex + 1, rightIndex);
    int maxLeftBaseSum = 0;
    int maxRightBaseSum = 0;
    for (int i = midIndex; midIndex >= leftIndex; i--) {// 前半部分
        maxLeftBaseSum += a[i];
        if (maxLeftBaseSum > maxLeftSubSum) {
            maxLeftSubSum = maxLeftBaseSum;
        }
    }
    for (int i = midIndex + 1; midIndex <= rightIndex; i++) {// 后半部分
        maxRightBaseSum += a[i];
        if (maxRightBaseSum > maxRightSubSum) {
            maxRightSubSum = maxRightBaseSum;
        }
    }
    return max(maxLeftSubSum, maxRightSubSum, maxLeftSubSum + maxRightSubSum);

}

private static int max(int i, int j, int k) {
    int temp = i;
    if (temp < j) {
        temp = j;
    }
    if (temp < k) {
        temp = k;
    }
    return temp;
}
```

4. END

参考以下内容按自己理解写了一遍，有错误请回复。不胜感激！

最大子序列和算法问题
《数据结构与算法分析》
12