053 Maximum Subarray

2018年3月29日 18:52

Question:

Find the contiguous subarray within an array (containing at least one number) which has the largest sum.
For example, given the array [-2,1,-3,4,-1,2,1,-5,4],
the contiguous subarray [4,-1,2,1] has the largest sum = 6.

More practice:

If you have figured out the O(*n*) solution, try coding another solution using the divide and conquer approach, which is more subtle.

来自 <https://leetcode.com/problems/maximum-subarray/description/>

给定一个序列（至少含有 1 个数），从该序列中寻找一个连续的子序列，使得子序列的和最大。
例如，给定序列 [-2,1,-3,4,-1,2,1,-5,4]，
连续子序列 [4,-1,2,1] 的和最大，为 6。

扩展练习:

若你已实现复杂度为 O(*n*) 的解法，尝试使用更为精妙的分治法求解。

Solution for Python3:

```
class Solution:
    def maxSubArray(self, nums):
        """
        :type nums: List[int]
        :rtype: int
        """
        for i in range(1,len(nums)):
            if nums[i - 1] > 0:
                nums[i] += nums[i - 1]
        return max(nums)
```

Solution for C++:

```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        for (int i = 1; i < nums.size(); i++) {
           if (nums[i - 1] > 0) {
               nums[i] += nums[i - 1];
           }
        }
       return *max_element(nums.begin(), nums.end());;
    }
};
```

Appendix:

Python 内建函数之max/min():

max/min 后直接跟序列会返回此序列的最大/最小值

max(iterable, key, default) 求迭代器的最大值，其中iterable 为迭代器，max会for i in … 遍历一遍这个迭代器，然后将迭代器的每一个返回值当做参数传给key=func 中的func(一般用lambda表达式定义)，然后将func的执行结果传给key，然后以key为标准进行大小的判断。

```
d1 = {'name': 'a', 'price': 3}
d2 = {'name': 'b', 'price': 2}
d3 = {'name': 'c', 'price': 1}
l1 = [d1, d2, d3]

a = max(l1, key=lambda x: x['name'])
print(a) #{'name': 'c', 'price': 1}

b = max(l1, key=lambda x: x['price'])
print(b) #{'name': 'a', 'price': 3}
```

C++: auto index= max_element(iter.begin(), iter.end())返回的是迭代器，加'*'才能获取到该迭代器指向的值：*index。