Daily Coding Problem #47

Problem

This problem was asked by Facebook.

Given a array of numbers representing the stock prices of a company in chronological order, write a function that calculates the maximum profit you could have made from buying and selling that stock once. You must buy before you can sell it.

For example, given [9, 11, 8, 5, 7, 10], you should return 5, since you could buy the stock at 5 dollars and sell it at 10 dollars.

Solution

The brute force solution here is to iterate over our list of stock prices, and for each price, find the largest profit you can make from selling that stock price (with the formula future - current), and keep track of the largest profit we can find. That would look like this:

```
def buy_and_sell(arr):
    max_profit = 0
    for i in range(len(arr) - 1):
        for j in range(i, len(arr)):
            buy_price, sell_price = arr[i], arr[j]
            max_profit = max(max_profit, sell_price - buy_price)
    return max_profit
```

This would take $O(N^2)$. Can we speed this up?

The maximum profit comes from the greatest difference between the highest price and lowest price, where the higher price must come after the lower one. But if we see a high price x and then a higher price y afterwards, then we can always discard x. So, if we keep track of the highest price in the future for each variable, we can immediately find how much profit buying at that price can make.

That means we can look at the array backwards and always keep track of the highest price we've

seen so far. Then, at each step, we can look at the current price and check how much profit we would have made buying at that price by comparing with our maximum price in the future. Then we only need to make one pass!

```
def buy_and_sell(arr):
    current_max, max_profit = 0, 0
    for price in reversed(arr):
        current_max = max(current_max, price)
        potential_profit = current_max - price
        max_profit = max(max_profit, potential_profit)
    return max_profit
```

This runs in O(N) time and O(1) space.

© Daily Coding Problem 2019

Privacy Policy

Terms of Service

Press