BAT算法面试题(十一)--最长的斐波那契子序列的长度(动态规划法)

一.面试题目

如果序列 `X_1,X_2,...,X_n` 满足下列条件,就说它是 斐波拉契式的:

- `n >= 3`
- 对于所有 `i+2 <= n` ,都有 `X_i + X_{i+1} = X_{i+2}` ;

给定一个严格递增的正整数数组形成序列.找到A中最长的斐波拉契式子序列的长度.如果一个不存在,返回0.比如,子序列是从原序列A中派生出来的.它从A中删除任意数量的元素.而不改变其元素的顺序.例如 `[3,5,8]` 是 `[3,4,5,6,7,8]` 的子序列.

二.案例

案例(1)

- **输入**: `[1,2,3,4,5,6,7,8]`
- **输出**: **5**
- **原因**: 最长的斐波波拉契式子序列: `[1,2,3,5,8]`

案例(2)

- **输入**: `[1,3,7,11,12,14,18]`
- **输出**: 3
- **原因**: 最长的斐波波拉契式子序列: `[1,11,12],[3,11,14],[7,11,18]`

三.解决方案-- 使用Set(集合)暴力法

- **思路**

将斐波拉契式的子序列中的2个连续项A[i],A[j] 视为单个结点(i,j).整个子序列是这些连续结点的之间的路径.例如,对于斐波拉契式的子序列, `(A[1] = 2,A[2] = 3,A[4]`

`= 5,A[7] = 8,A[10] = 13)`,结点的路径就为`(1,2)<->(2,3)<->(4,7)<->(7,10).`

这样做的目的,只有当`A[i]+A[j] == A[k]`时.两结点`(i,j)和(j,k)`才是连贯的.我们需要这个信息才能知道它们之间是可以连通的.

- **算法**

设`longest[i,j]`是结束在`[i,j]`的最长路径.那么如果`(i,j)`和`(j,k)`是连通的,`longest[j,k] = longest[i,j]+1`.由于`i`是由`A.index(A[k] - A[j])`唯一确定的.我们在`i`检查每组`j < k`,并相应更新`longest[j,k]`

四.代码

```
class Solution {
public:
    int lenLongestFibSubseq(vector<int>& A) {
        int N = A.size();
        unordered_map<int, int> index;
        for (int i = 0; i < N; ++i)
            index[A[i]] = i;

        unordered_map<int, int> longest;
        int ans = 0;
        for (int k = 0; k < N; ++k)
            for (int j = 0; j < k; ++j) {
                if (A[k] - A[j] < A[j] && index.count(A[k] - A[j])) {
                    int i = index[A[k] - A[j]];
                    longest[j * N + k] = longest[i * N + j] + 1;
                    ans = max(ans, longest[j * N + k] + 2);
                }
            }

        return ans >= 3 ? ans : 0;
    }
};
```

五.复杂度分析

- **时间复杂度**: `O(N^2)`,其中N指的是A的长度
- **空间复杂度**: `O(NlogM)`,其中M是A中的最大的元素.

六.学习建议

- 先了解基本思路
- 在带着数据,理解代码的执行