

Problem 2044: Count Number of Maximum Bitwise-OR Subsets

Problem Information

Difficulty: Medium

Acceptance Rate: 89.62%

Paid Only: No

Tags: Array, Backtracking, Bit Manipulation, Enumeration

Problem Description

Given an integer array `nums`, find the **maximum** possible **bitwise OR** of a subset of `nums` and return **the** number of different non-empty subsets with the maximum bitwise OR.

An array `a` is a **subset** of an array `b` if `a` can be obtained from `b` by deleting some (possibly zero) elements of `b`. Two subsets are considered **different** if the indices of the elements chosen are different.

The bitwise OR of an array `a` is equal to $a[0] \text{ OR } a[1] \text{ OR } \dots \text{ OR } a[a.length - 1]$ (**0-indexed**).

Example 1:

Input: `nums = [3,1]` **Output:** `2` **Explanation:** The maximum possible bitwise OR of a subset is 3. There are 2 subsets with a bitwise OR of 3: - `[3]` - `[3,1]`

Example 2:

Input: `nums = [2,2,2]` **Output:** `7` **Explanation:** All non-empty subsets of `[2,2,2]` have a bitwise OR of 2. There are $2^3 - 1 = 7$ total subsets.

Example 3:

Input: `nums = [3,2,1,5]` **Output:** `6` **Explanation:** The maximum possible bitwise OR of a subset is 7. There are 6 subsets with a bitwise OR of 7: - `[3,5]` - `[3,1,5]` - `[3,2,5]` - `[3,2,1,5]` -

[2,5] - [2,1,5]

****Constraints:****

*`1` <= nums.length <= 16` *`1` <= nums[i] <= 105`

Code Snippets

C++:

```
class Solution {
public:
    int countMaxOrSubsets(vector<int>& nums) {

    }
};
```

Java:

```
class Solution {
    public int countMaxOrSubsets(int[] nums) {

    }
}
```

Python3:

```
class Solution:
    def countMaxOrSubsets(self, nums: List[int]) -> int:
```