Make Costs of Paths Equal in a Binary Tree (/problems/make-costs-of-paths-equal-in-a-binary-tree/)

Submission Detail

1564 / 1564 test cases passed.

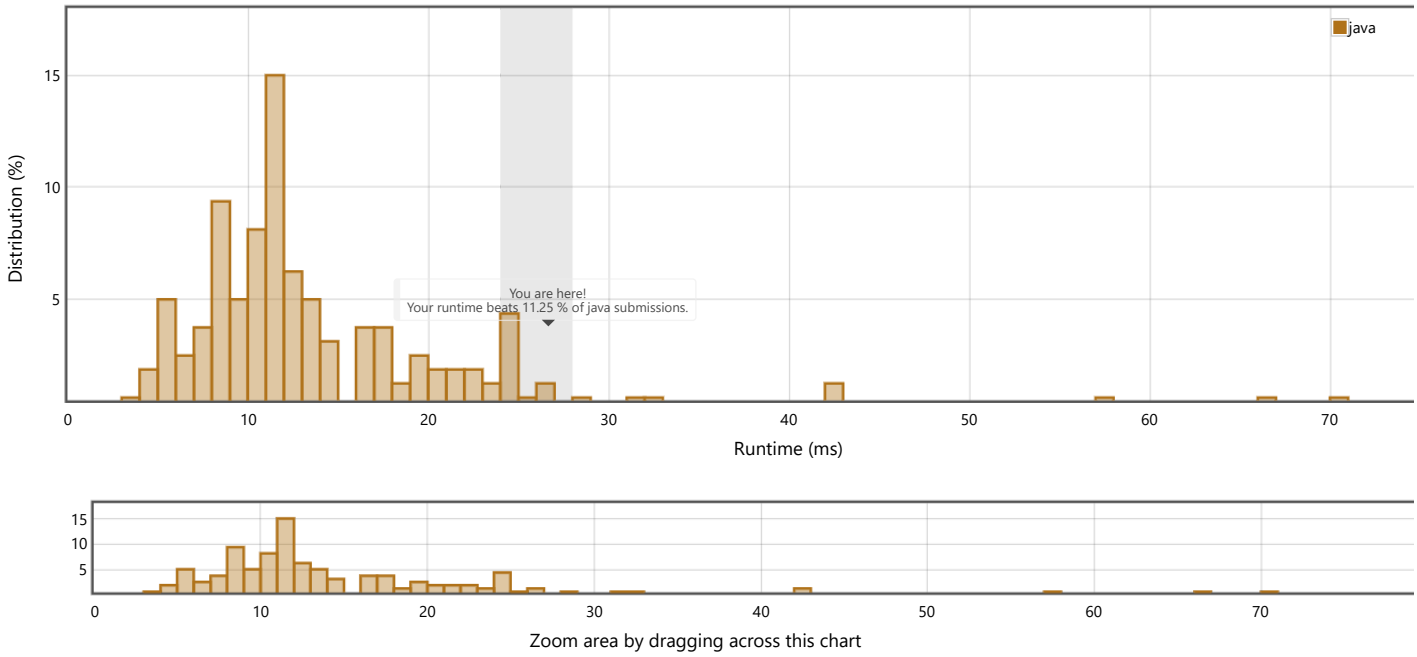
Status: **Accepted**

Runtime: **26 ms**

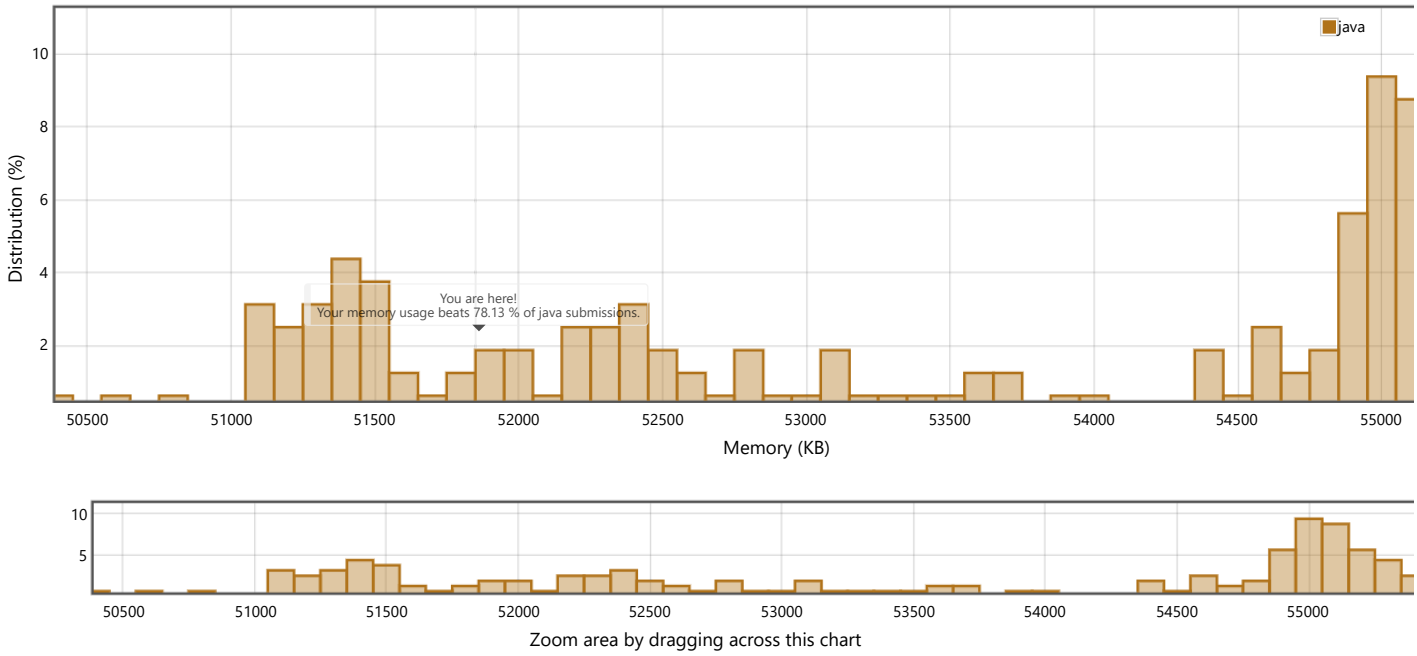
Memory Usage: **51.9 MB**

Submitted: **1 minute ago**

Accepted Solutions Runtime Distribution



Accepted Solutions Memory Distribution



Invite friends to challenge **Make Costs of Paths Equal in a Binary Tree**

Submitted Code: 1 minute ago

Language: java

Edit Code

```
1  class Solution {
```

```
    public void dfs(int numNodes, int node, int[] pathCosts, int[] nodeCosts, int currentPathCost) {
        if (node >= numNodes) {
            return;
        }
        currentPathCost += nodeCosts[node];
        pathCosts[node] = currentPathCost;
        dfs(numNodes, node*2+1, pathCosts, nodeCosts, currentPathCost);
        dfs(numNodes, node*2+2, pathCosts, nodeCosts, currentPathCost);
    }

    public int minIncrements(int numNodes, int[] nodeCosts) {
        int[] pathCosts = new int[numNodes];
        dfs(numNodes, 0, pathCosts, nodeCosts, 0);

        int half = numNodes / 2;
        int maxPathCost = 0;
        for (int i = half; i < numNodes; i++) {
            maxPathCost = Math.max(maxPathCost, pathCosts[i]);
        }
        int numIncrements = 0;
        for (int i = 0; i < numNodes; i++) {
            int left = i, right = i;
            while (left < half) {
                left = left*2+1;
                right = right*2+2;
            }
            int subMaxPathCost = 0;
            for (int j = left; j <= right; j++) {
                subMaxPathCost = Math.max(subMaxPathCost, pathCosts[j]);
            }

            int increment = maxPathCost - subMaxPathCost;
            numIncrements += increment;
            for (int j = left; j <= right; j++) {
                pathCosts[j] += increment;
            }
        }
        return numIncrements;
    }
}
```

Back to problem (/problems/make-costs-of-paths-equal-in-a-binary-tree/)