

```

1  /*
2  给定一个二叉树，它的每个结点都存放着一个整数值。
3
4  找出路径和等于给定数值的路径总数。
5
6  路径不需要从根节点开始，也不需要在叶子节点结束，但是路径方向必须是向下的（只能从父节点到子节点）。
7
8  二叉树不超过1000个节点，且节点数值范围是 [-1000000,1000000] 的整数。
9
10 示例：
11
12  root = [10,5,-3,3,2,null,11,3,-2,null,1], sum = 8
13
14      10
15     /  \
16    5   -3
17   / \   \
18  3  2   11
19 / \   \
20 3 -2  1
21
22 返回 3。和等于 8 的路径有：
23
24 1. 5 -> 3
25 2. 5 -> 2 -> 1
26 3. -3 -> 11
27
28 来源：力扣（LeetCode）
29 链接：https://leetcode-cn.com/problems/path-sum-iii
30 著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。
31 */

```

分析:

- 任务要求遍历所有的节点及所有节点的分支,采取两重递归的话,栈开销预计不小;
- 方法一:BFS+DFS
 - BFS:遍历所有节点
 - DFS:遍历从当前节点开始的所有可能分支
- 方法二:DFS+DFS

方法一:C++_BFS+DFS

```

1  /**
2   * Definition for a binary tree node.
3   * struct TreeNode {
4   *     int val;
5   *     TreeNode *left;
6   *     TreeNode *right;
7   *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
8   * };

```

```

9      */
10     class solution
11     {
12     private:
13         int ret_val    = 0;
14         int target_sum = 0;
15
16         void __pathSum(TreeNode* node , int level_sum)
17         {
18             if(node==NULL)
19             {
20                 return ;
21             }
22
23             int temp = level_sum + node->val;
24
25             if(temp==target_sum)
26             {
27                 ret_val++;
28                 //return;
29             }
30             __pathSum(node->left,temp);
31             __pathSum(node->right,temp);
32         }
33     public:
34         int pathSum(TreeNode* root, int sum)
35         {
36             ret_val    = 0;
37             target_sum = sum;
38             queue<TreeNode*>    qtn    ;
39             TreeNode*          temp    ;
40             if(root!=NULL)
41             {
42                 qtn.push(root);
43                 while(!qtn.empty())
44                 {
45                     temp = qtn.front();
46                     qtn.pop();
47                     __pathSum(temp,0);
48                     if(temp->left)
49                     {
50                         qtn.push(temp->left);
51                     }
52                     if(temp->right)
53                     {
54                         qtn.push(temp->right);
55                     }
56                 }
57             }
58             return ret_val;
59         }
60     };
61
62     /*
63     执行结果:
64     通过
65     显示详情

```

```
67 执行用时 :40 ms, 在所有 C++ 提交中击败了49.50% 的用户
68 内存消耗 :14.5 MB, 在所有 C++ 提交中击败了93.90%的用户
69 */
```

方法二:C++_DFS+DFS

```
1  /**
2   * Definition for a binary tree node.
3   * struct TreeNode {
4   *     int val;
5   *     TreeNode *left;
6   *     TreeNode *right;
7   *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
8   * };
9   */
10 class Solution
11 {
12     private:
13         int ret_val = 0;
14         int target_sum = 0;
15
16         void __pathSum(TreeNode* node , int level_sum)
17         {
18             if(node==NULL)
19             {
20                 return ;
21             }
22
23             int temp = level_sum + node->val;
24
25             if(temp==target_sum)
26             {
27                 ret_val++;
28                 //return;
29             }
30             __pathSum(node->left,temp);
31             __pathSum(node->right,temp);
32         }
33
34
35         /* 根左右*/
36         void dfs(TreeNode* node)
37         {
38             if(node==NULL)
39             {
40                 return;
41             }
42
43             __pathSum(node,0);
44             dfs(node->left);
45             dfs(node->right);
46         }
47
48     public:
49         int pathSum(TreeNode* root, int sum)
50         {
```

```
51         ret_val      = 0;
52         target_sum    = sum;
53         dfs(root);
54         return ret_val;
55     }
56 };
57
58 /*
59 执行结果:
60 通过
61 显示详情
62 执行用时 :44 ms, 在所有 C++ 提交中击败了41.65% 的用户
63 内存消耗 :14.6 MB, 在所有 C++ 提交中击败了84.96%的用户
64 */
```

AlimyBreak
2019.09.15