Assignment 02

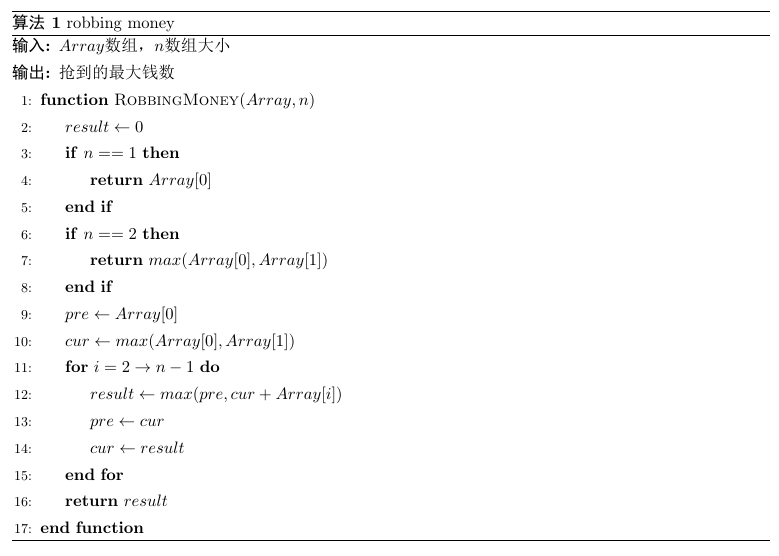
肖阳 201918008629001

# 1 Robbing Money

1. (a)Assume that there are N houses along the street, I use to represent the maximum amount of money I can rob at one night and to represent the amount of money in each house. If N equal to 1, I only have one house to rob , so . If N equal to 2, I will rob the house which has larger amount of money because I can’t rob both at one night, so . Considering more common cases, the maximum amount of money I can get from i houses determined by whether I rob the house. If I rob the house ,, or because I can’t rob two adjacent houses.

**DP EQUATION:**

(b) **Pseudo-code**



(c) **Prove the correctness**

Assume that OPT(i) have a better value OPT’(i), and I rob the house so that . Since has a different choice, . What’s more, is greater than . That is a contradiction to the definition of OPT(i). So the correctness is proved!

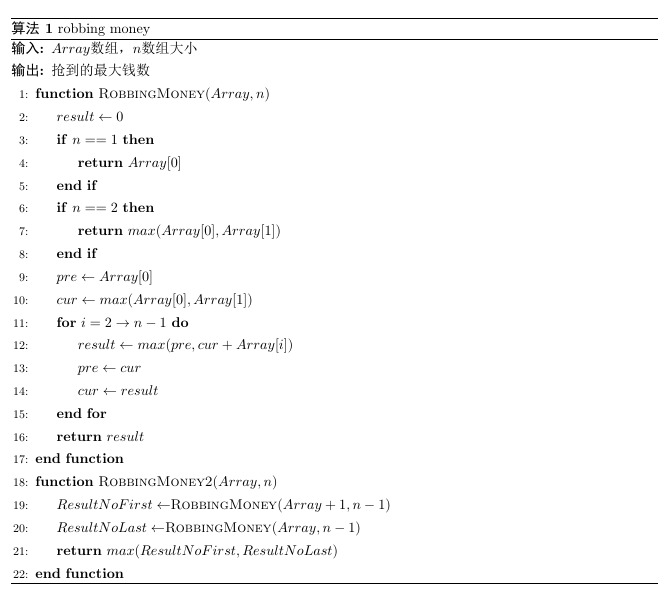
(d) **Time complexity**: O(n) **Space complexity**: O(1)

1. (a) The dissimilarity between this question to the last one is that the first house and the last house can’t be robbed at one night, so that the question can be splited into two situation: (1)The first house was not robbed. OPT(N) can be solved by the same way in Q1 with A[2:N].(2)The last house was not robbed. OPT(N) can be solved by the same way in Q1 with A[1:N-1].

**DP EQUATION:**

means the maximum amount of money I can rob from the house to the house.

(b) **Pseudo-code**



(c)**Prove the correctness**

The question contains the two special cases of Q1, without the first house or without the last house .This will not affect the correctness of the original algorithm.

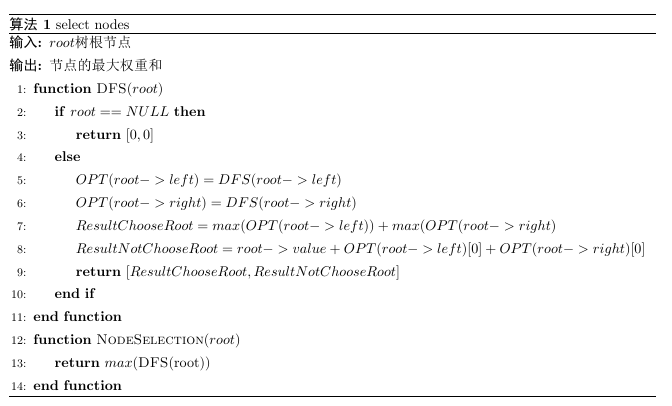
(d)**Time complexity**: O(n) **Space complexity**: O(1)

# 2 Node Selection

1. Assume R is the root of any subtree of the entire tree. represents the maximum sum of weight I can get from the subtree whose root of R. So there are only two situation: choose R or not choose R. If R is chosen,.If R is not chosen ,that means its child can be chosen, .if R is null, then .

**DP EQUATION:**

1. Pseudo-code



1. **Prove the correctness**

Assume that have a better value, and I chosen so that .

Since has the different value,

. is larger than , that is to say is larger than . This is a contradiction of the definition of . The prove complete!

1. Every node in the tree will be visited only once , so time complexity of the algorithm is O(n).

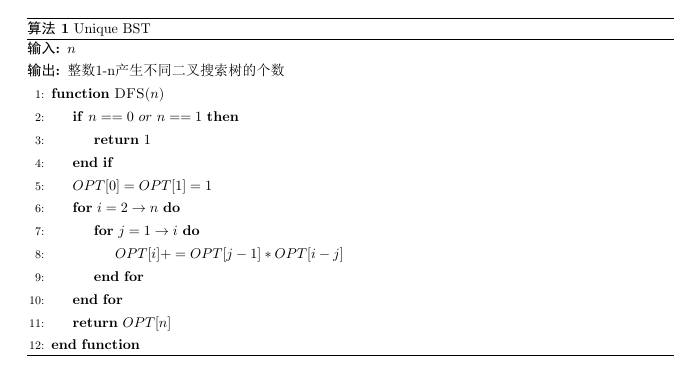
# Unique Binary Search Tree

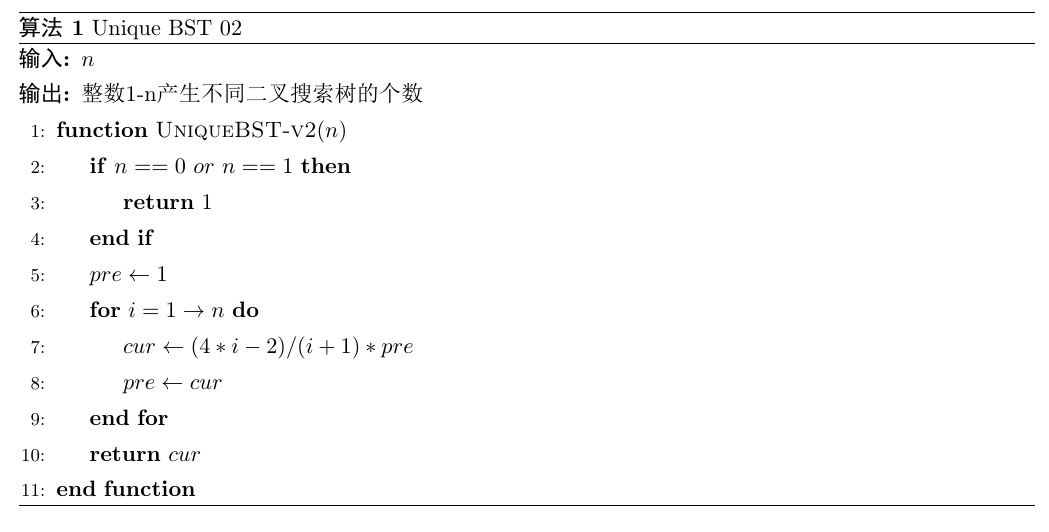
1. Assume OPT(N) is the number of unique binary search tree that 1~N can generate, F(i,N) is the number of unique binary search tree whose root is that 1~N can generate. So . What’s more , any BST whose root is *i* can be separated into two parts, left subtree is generated by number 1~(*i*-1) , right subtree is generated by number (i+1)~n. So the number of unique BST whose root is *i* equal to the number of unique left subtree times the number of unique right subtree, that is .

**DP EQUATION:**

What’s more , is a catalan number, so the equation can be transformed into

1. Pseudo-code





1. **Prove the correctness**

If n equal to 0,which means there are no positive integer to generate BST, we can only get a empty tree, OPT[0]=1; If n equal to 1, which means there are one integer 1 to generate BST, we can get a BST with one node, OPT[1]=1. Given N, there will be N situations of the root, So contain every cases of BST generated by N integer, and

has already been proved to be right.

1. Unique BST 01: Time complexity : O(n^2) Space complexity: O(n)

Unique BST 02: Time complexity : O(n) Space complexity: O(1)