TheTreeAndMan

Topcoder TCO 015, Q1B, 1000-Pointer

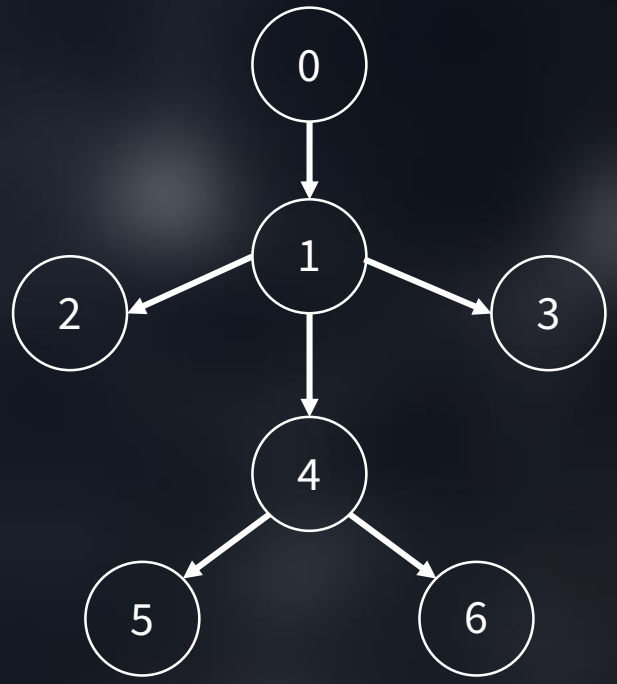
Clara Nguyen - COSC 594 - 09/20/2018

The Problem

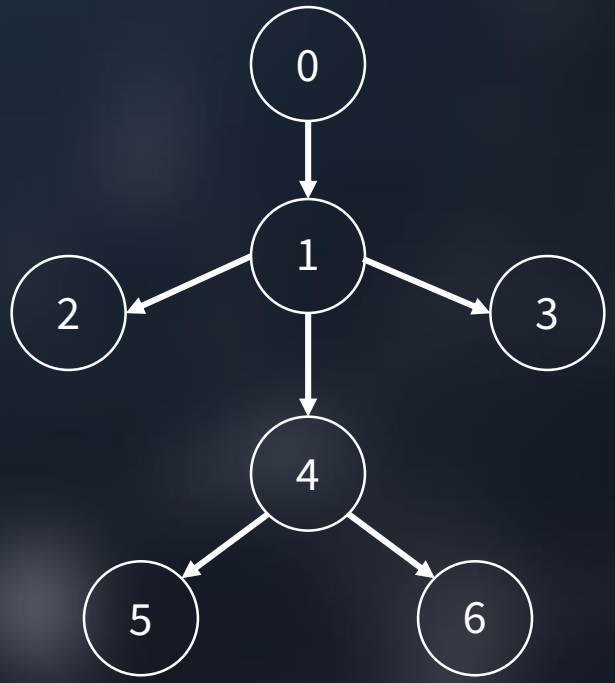
- Given a tree where nodes have a parent and any number of children.
- Find all possible combinations of “men” in the graph.

What is a man?

- Features a head, upper body, arms, lower body, and legs.
- Head, arms, body, and legs can be of any size.
- 2 men are different if they have a different sets of nodes in the graph.

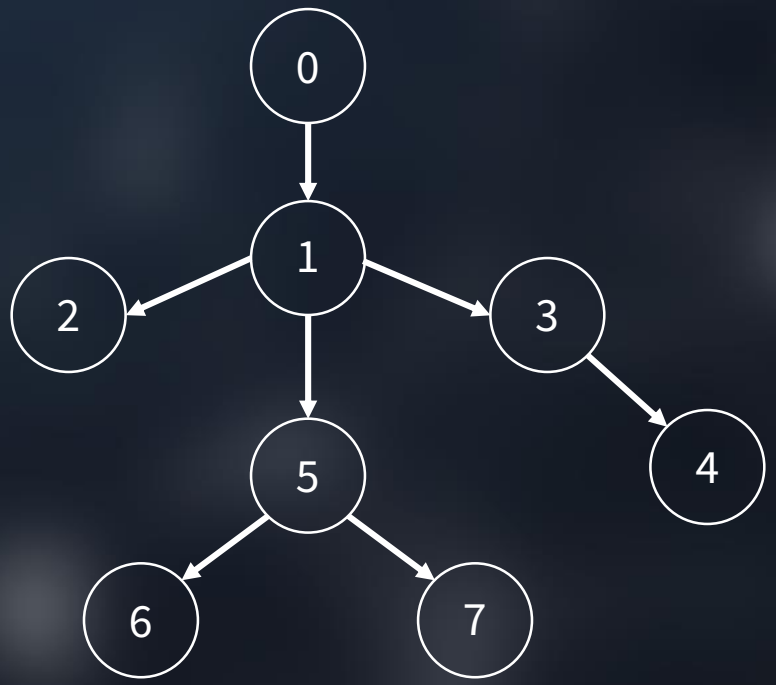


What is a man? (Example 1)



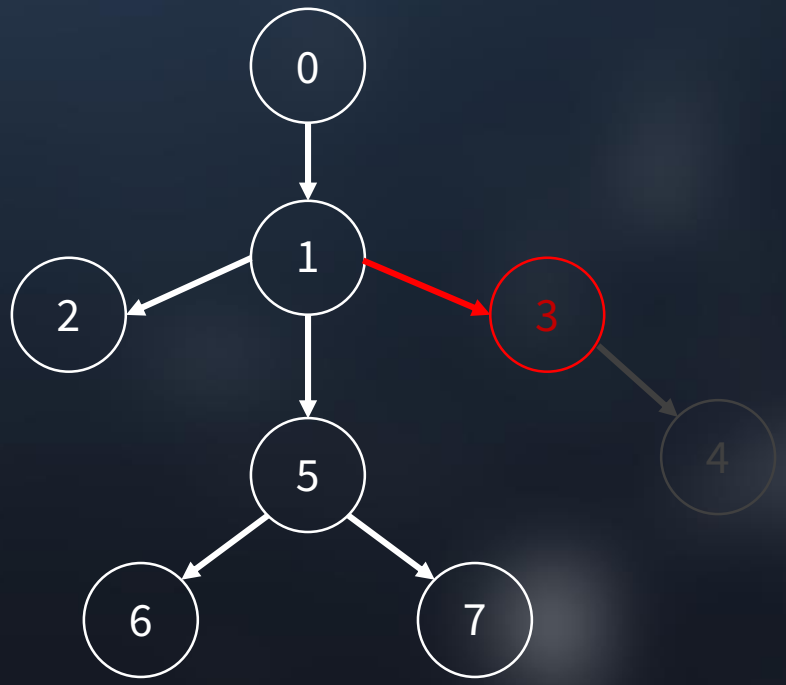
Combinations: 1

What is a man? (Example 2)

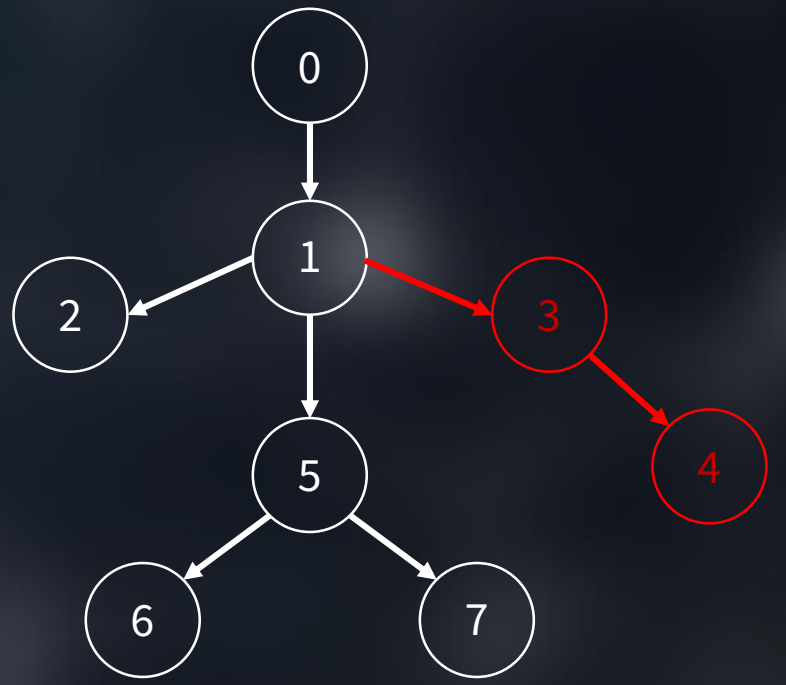


Combinations: 2

What is a man? (Example 2)

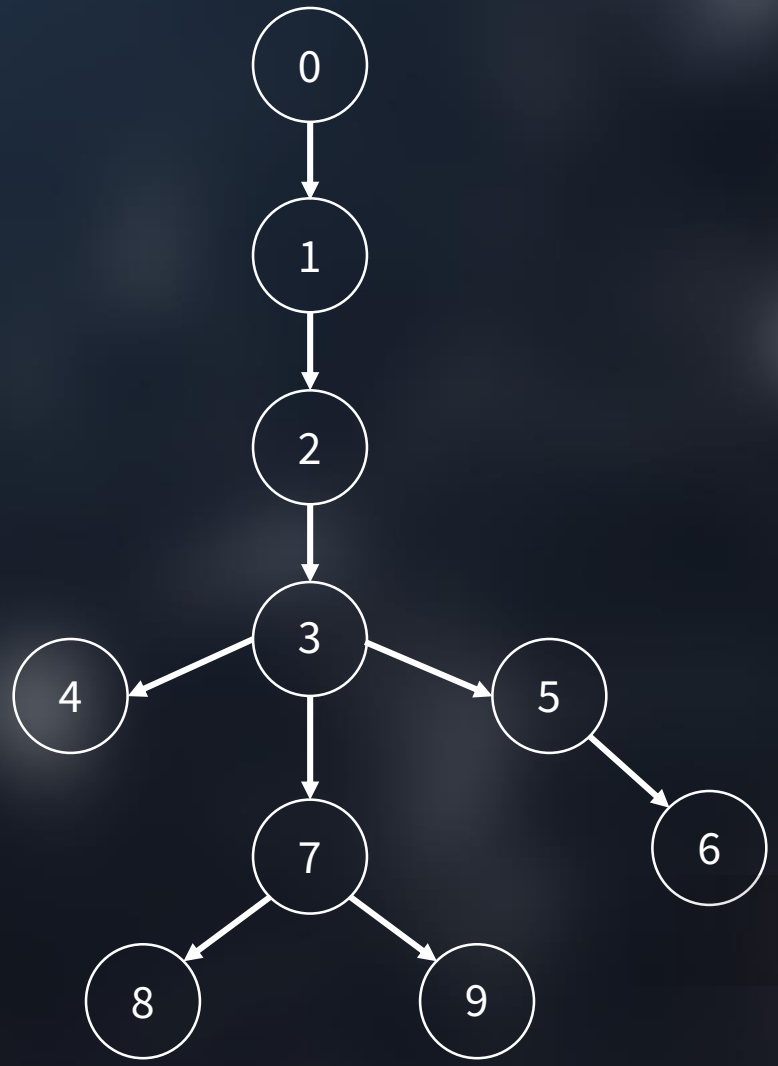


[0, 1, 2, 3, 5, 6, 7]



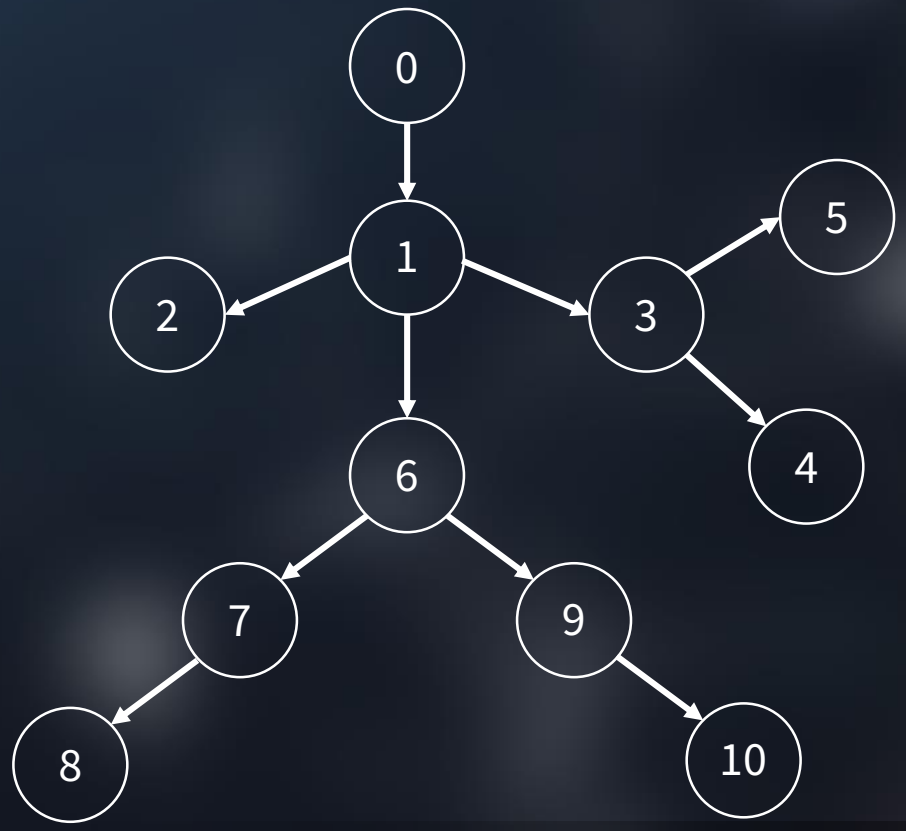
[0, 1, 2, 4, 5, 6, 7]

What is a man? (Example 3)



Combinations: 6

What is a man? (Example 4)



Combinations: 17

Problem Constraints

- **Class Name:** `TheTreeAndMan`
- **Method:** `solve`
- **Parameter:**

`parent`	`vector<int>`	Node parent indices

- **Return Value:** `int`
- **Constraints:**
  - N is between 2 and 2000 (inclusive)
  - Parent nodes contain exactly N – 1 elements.
  - For each valid node, its parent has to have an index between 0 and i

Method 1: Brute Force

Method 1: Brute Force

- Use a set to hold all combos and prevent duplicates.
- Traverse all possible combinations for valid body nodes:
- 1 Head, 1 Shoulder, 2 Arms, 1 Hip, 2 Legs. $O(N)$ each.
- The set’s size will be the number of valid men in the graph.

Method 1: Time Complexity

- $O(N)$ - Create the graph
- $O(N)$ - Traverse the graph for a head.
  - $O(N)$ - Traverse the graph for a upper body
    - $O(N)$ - Traverse the graph for a lower body
      - $O(N)$ - Traverse the graph for left arm
        - $O(N)$ - Traverse the graph for right arm
          - $O(N)$ - Traverse the graph for left leg
            - $O(N)$ - Traverse the graph for right leg
              - $O(\log N)$ - Check if set exists

Total Time: $O(N^7 \log N)$ Too slow!

Method 2: Dynamic Programming

Method 2: Dynamic Programming

- Much faster than Brute Force.
- Use recursion to traverse all combinations, then add them together.
- More mathematical approach.

Observations

- The problem doesn’t ask for all possible sets. Only combinations.
- Use math to compute combinations when at nodes.
  - $combinations = leg\ combos \times arm\ combos \times head\ combos$
- Use memoization to reduce repetitive recursive calls.

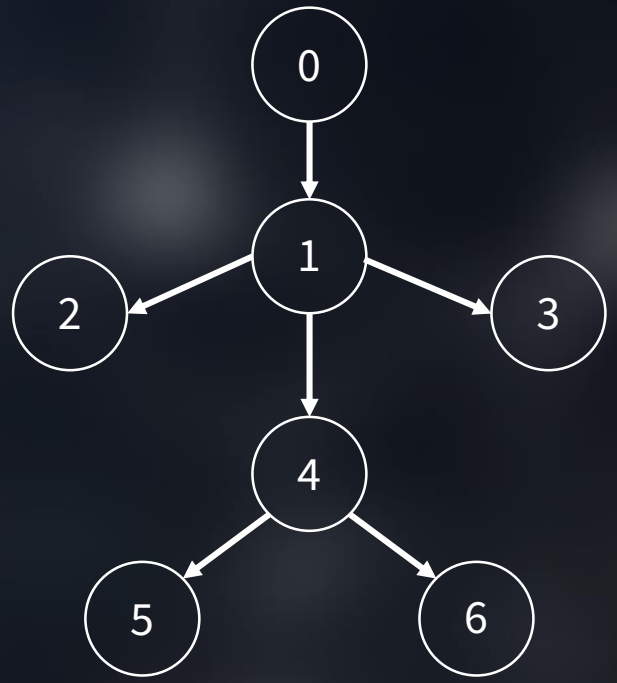
Method 2: Dynamic Programming

- Have 4 functions:
  - **Man_Leaf(n)** – Number of nodes reachable from node **n**
  - **Man_Legs(n)** – Number of leg combos in node **n**
  - **Man_Trunk(n)** – Number of arm combos × leg combos from node **n**
  - **Man(n)** – Number of men reachable from node **n**

Method 2: Man_Leaf(n)

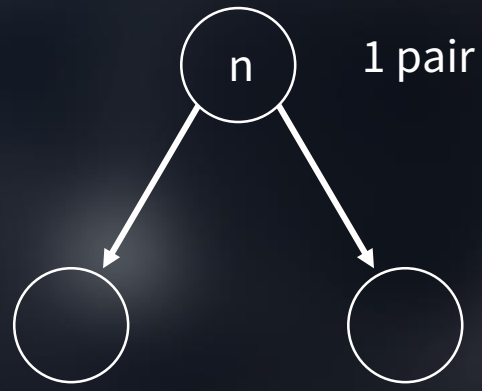
- Finds number of nodes reachable from node **n** (inclusively)

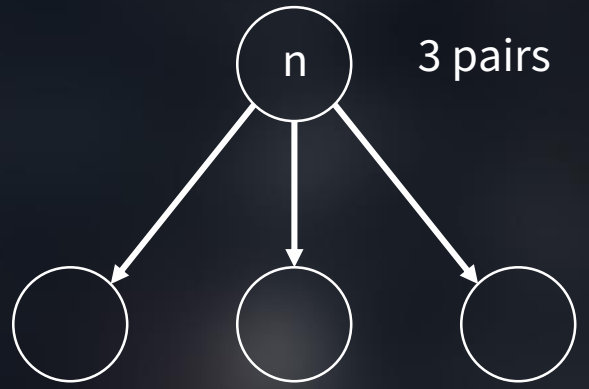
- **Man_Leaf(0)** = 7
- **Man_Leaf(4)** = 3



Method 2: Man_Legs(n)

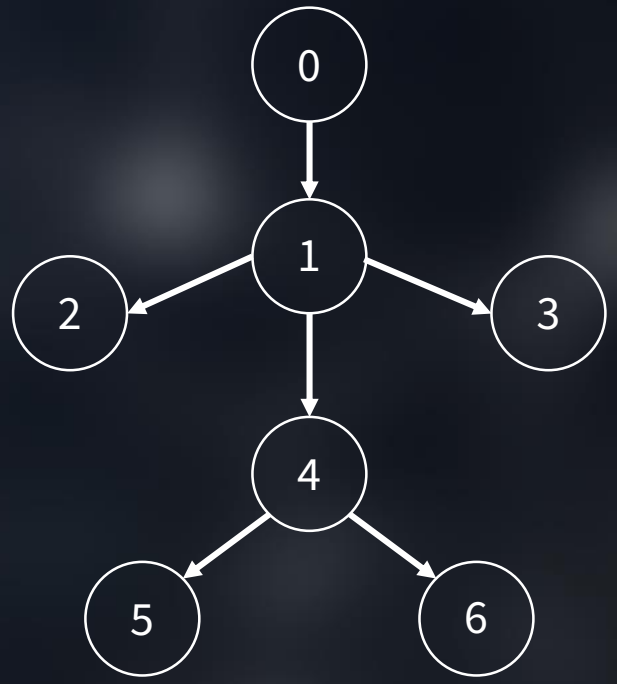
- Looks for the number of leg configurations reachable from **n**
- Recursively goes down and calls itself.





Method 2: Man_Legs(n)

- Looks for the number of leg configurations reachable from **n**
- Recursively goes down and calls itself.

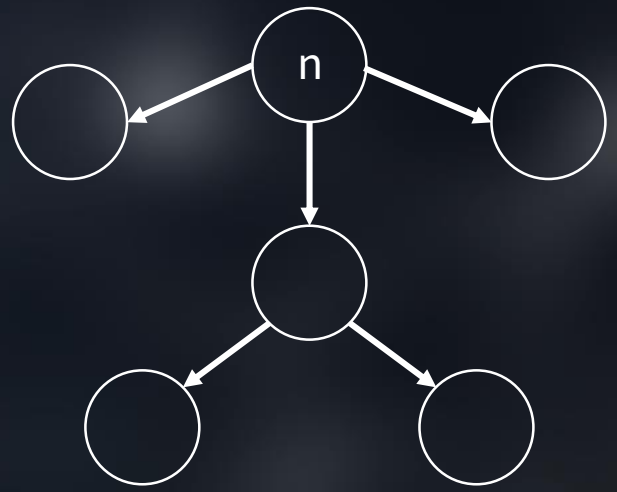


- **Man_Legs(4)** = 1
- **Man_Legs(2)** = 0
- **Man_Legs(1)** = 8

Method 2: Man_Trunk(n)

- Looks for the number of configurations with

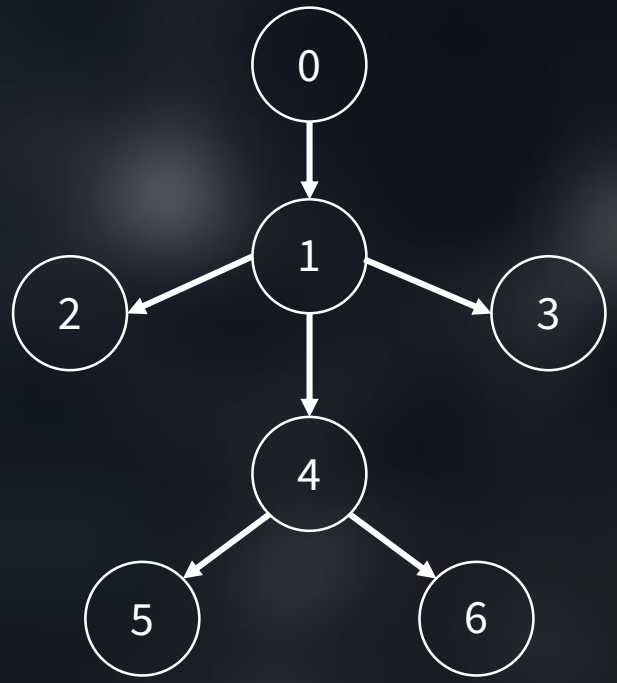
  2 arms and a node of legs from **n**.



Method 2: Man_Trunk(n)

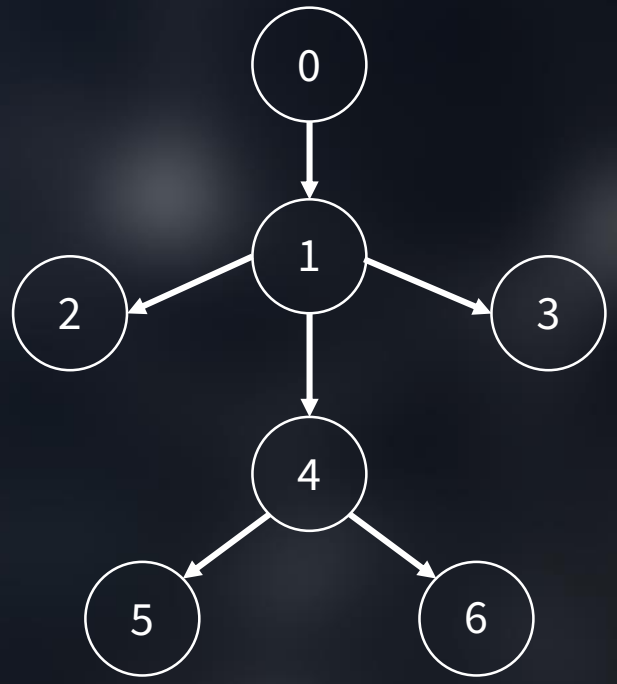
- Looks for the number of configurations with 2 arms and a node of legs from **n**.

- **Man_Trunk(1)** – 1
- Every other node – 0



Method 2: Man(n)

- Looks for the number of configurations like the one on the right, from **n**
- Recursive. Calls itself on all children.

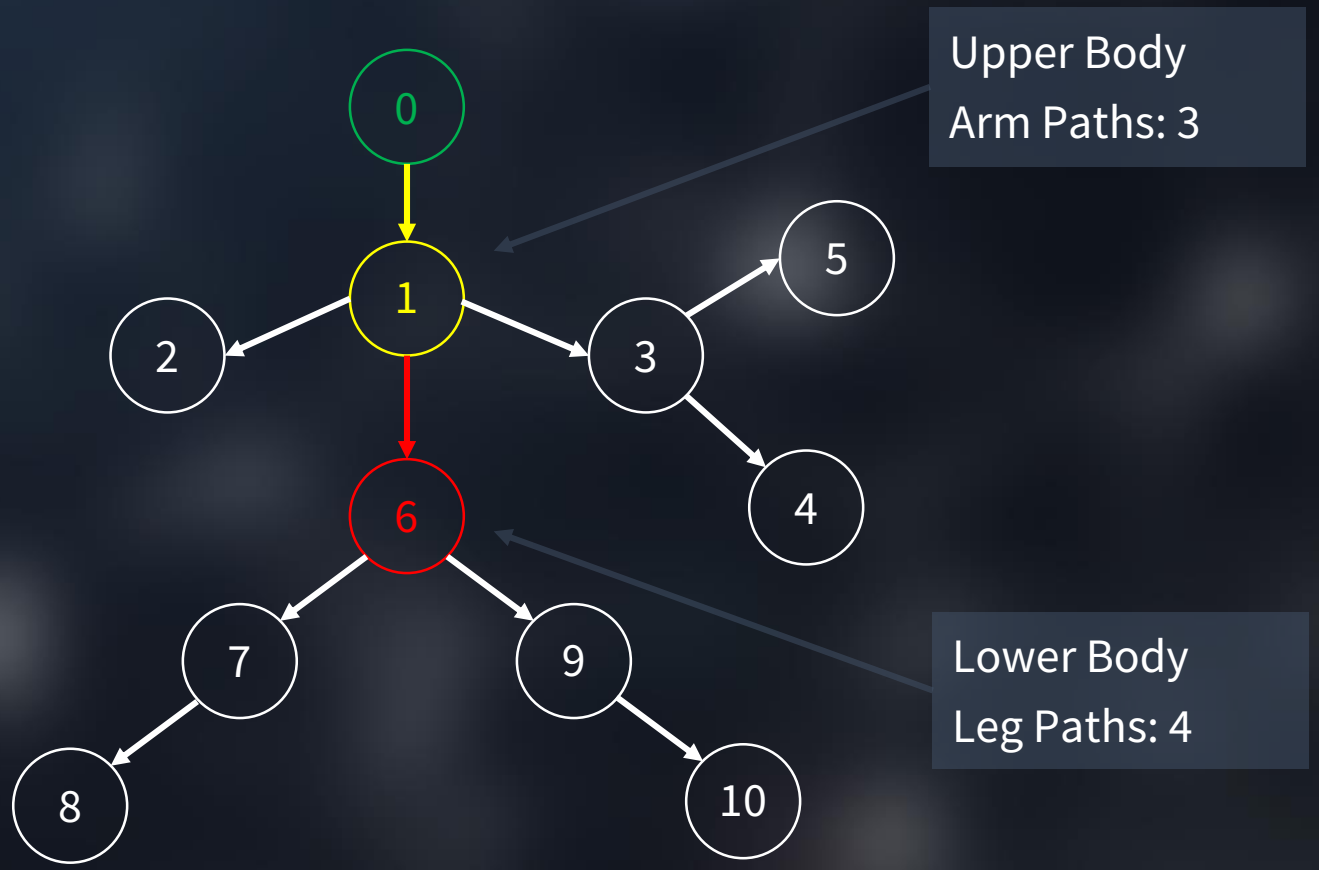


- **Man(0)** – 1
- Every other node – 0

Method 2: Example

- Body 1
  - Head Combos: 1
  - Arm Combos: 3
  - Leg Combos: 4

$$c_1 = 1 \times 3 \times 4 = 12$$

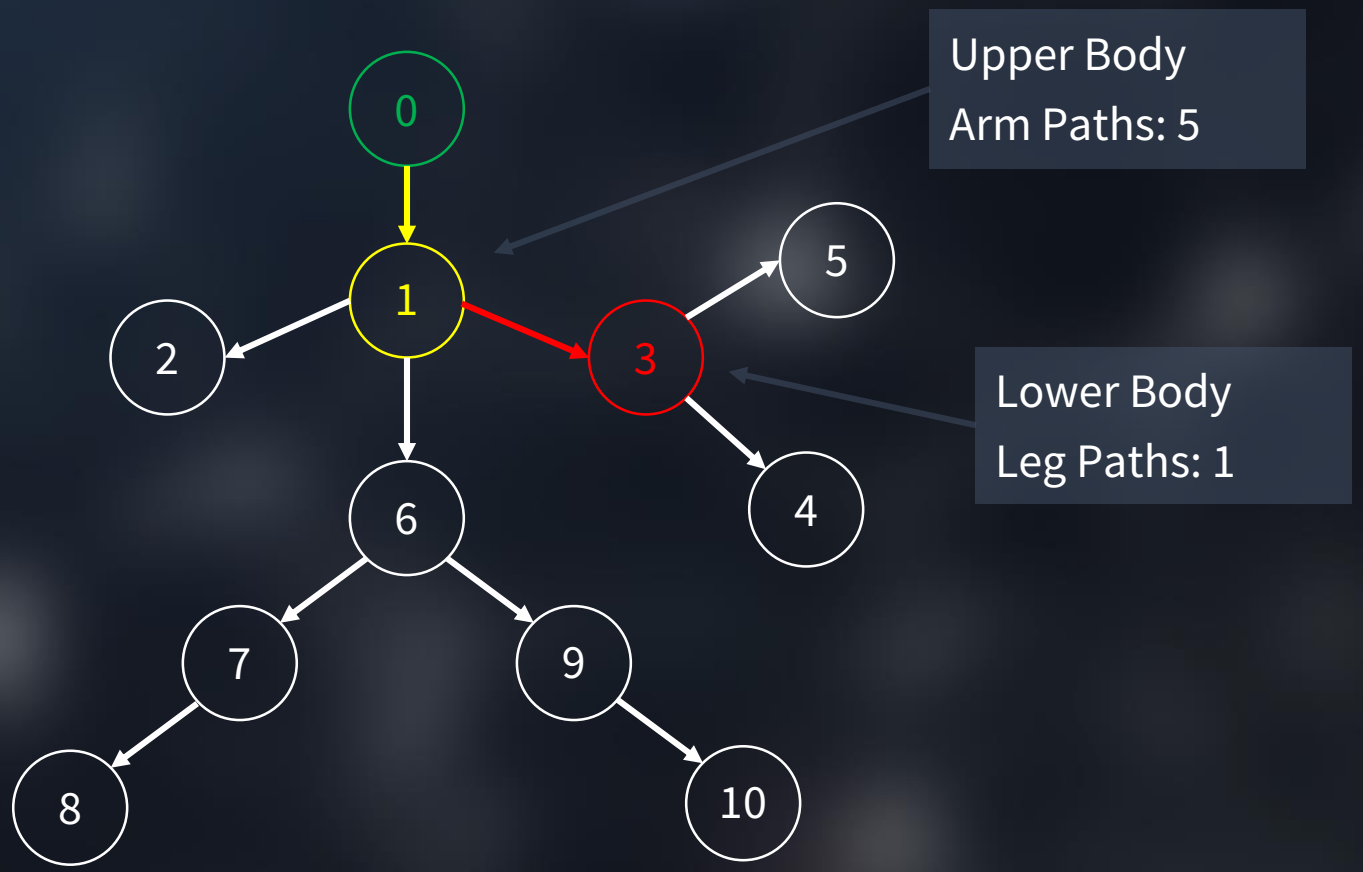


Method 2: Example

- Body 2
  - Head Combos: 1
  - Arm Combos: 5
  - Leg Combos: 1

$$c_1 = 1 \times 3 \times 4 = 12$$
$$c_2 = 1 \times 5 \times 1 = 5$$

$$c_1 + c_2 = 17$$



Method 2: Time Complexity

Setting up

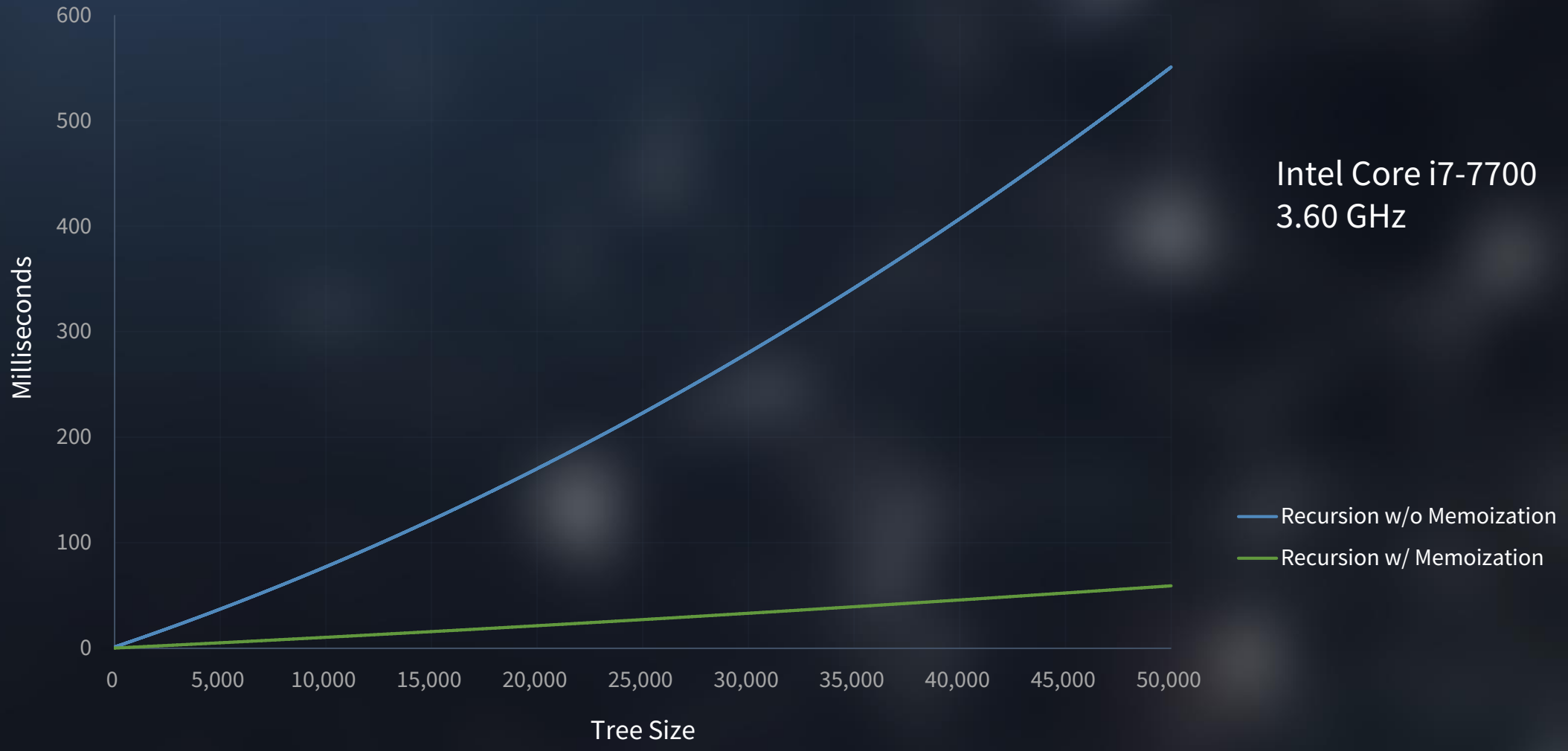
- $O(N)$ - Create the graph
- $O(N)$ - Pre-compute all leaf values for all nodes.

Functions

- $O(N)$ - Man()
- $O(N^3)$ - Man_Trunk()
- $O(N^2)$ - Man_Legs()
- $O(N)$ - Man_Leaf()

Overall: $O(N^3)$

Performance Comparison



One more thing…

Method 3: The $O(N \log N)$ Approach

- This can be done in $O(N \log N)$.
  - Traverse the tree backwards, from bottom to top.
  - Store depth and magic probability number in each node.
  - Make probability equations constant time.
  - Skip recursion entirely. Use multiplication to figure it out.

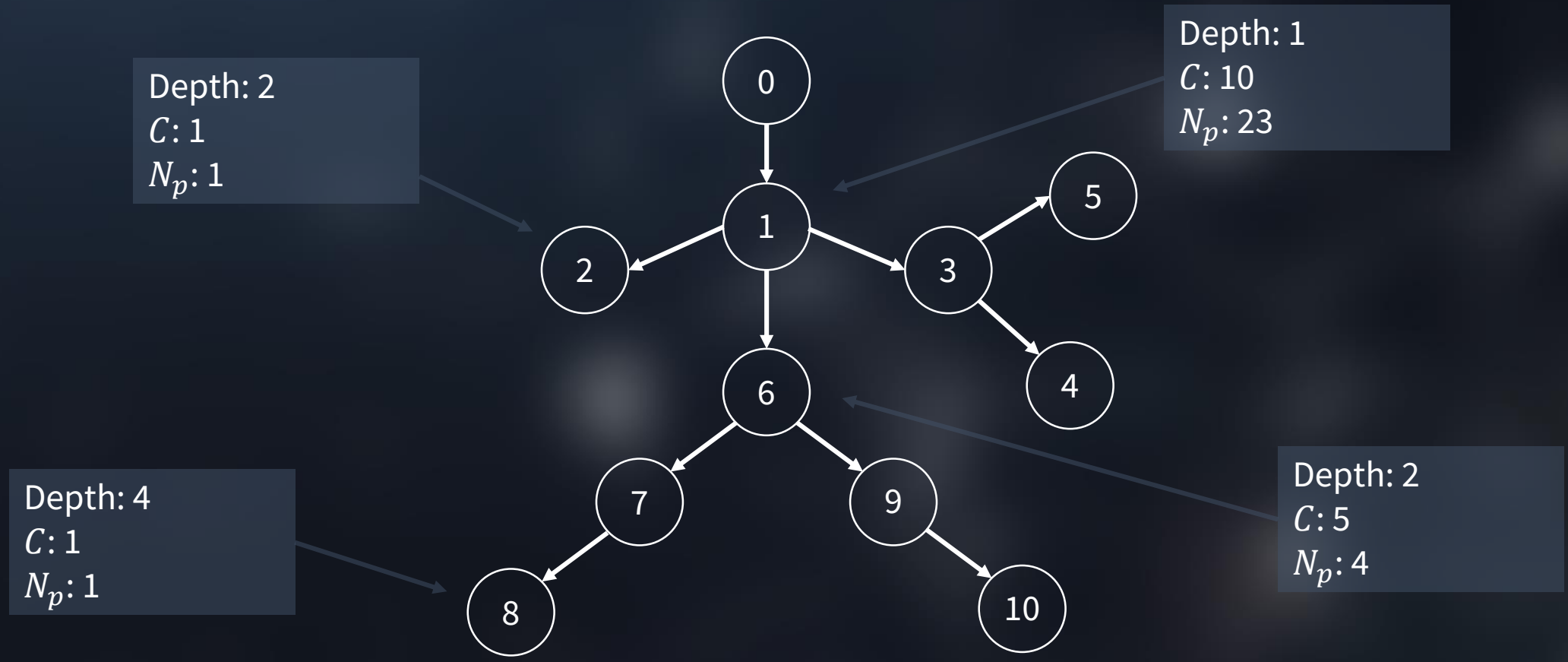
Method 3: Observations

- There are N children in each node, but only 1 parent.
- Don’t traverse nodes. Multiply combinations instead.
- Number of head combinations is equal to depth.
- Store a magic number (N_p) that we can subtract probabilities from.

$$N_p = \sum_{i=A+1}^{B} N_{i,c} \times C \text{ , where } C = N_{f,c} \text{ and increments by } N_{i,c}$$

Method 3: The $O(N \log N)$ Approach

- Each node will keep track of depth and all possible combinations.



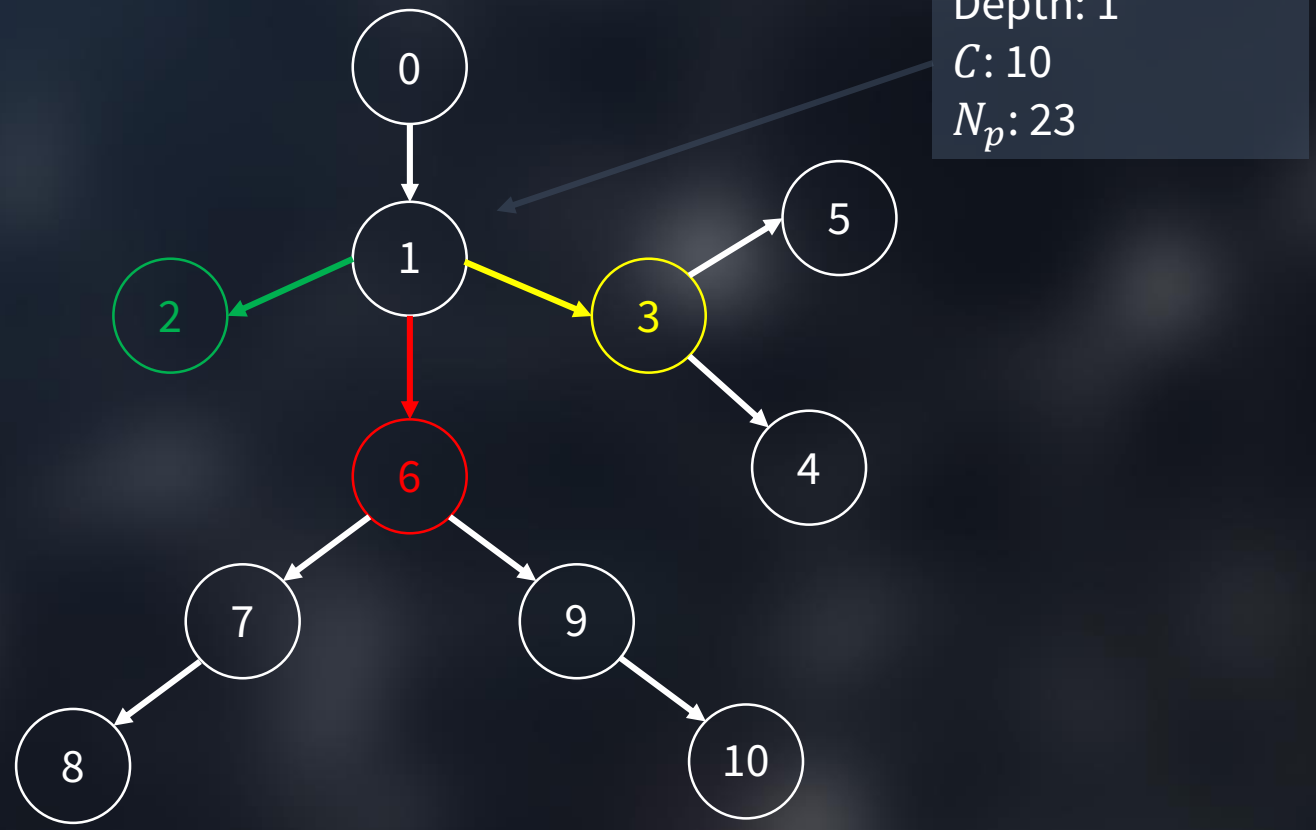
Method 3: The $O(N \log N)$ Approach

- Node 1's N_p is 23 because:

$$N_p = \sum_{i=A+1}^{B} N_{i,c} \times C$$

- Node 2 has 1 set.
- Node 3 has 3 sets.
- Node 6 has 5 sets.

$$N_p = (3 \times 1) + (5 \times 4) = 23$$

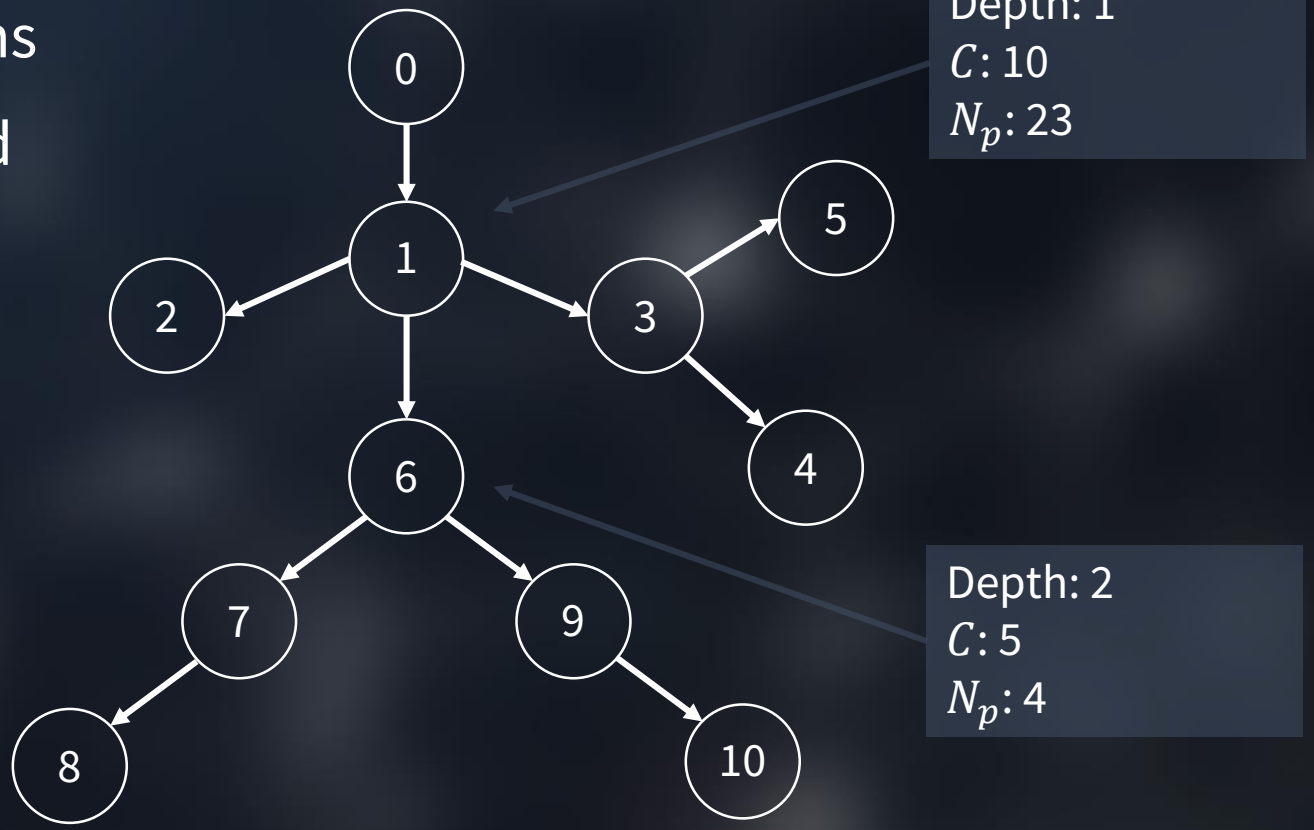


Method 3: The $O(N \log N)$ Approach

- Compute the arm combinations assuming Node 6 is the hip and Node 1's N_p is 23:

$$N_p - (C_1 - 1 - C_6) \times C_6$$

$$23 - (10 - 1 - 5) \times 5 = 3$$



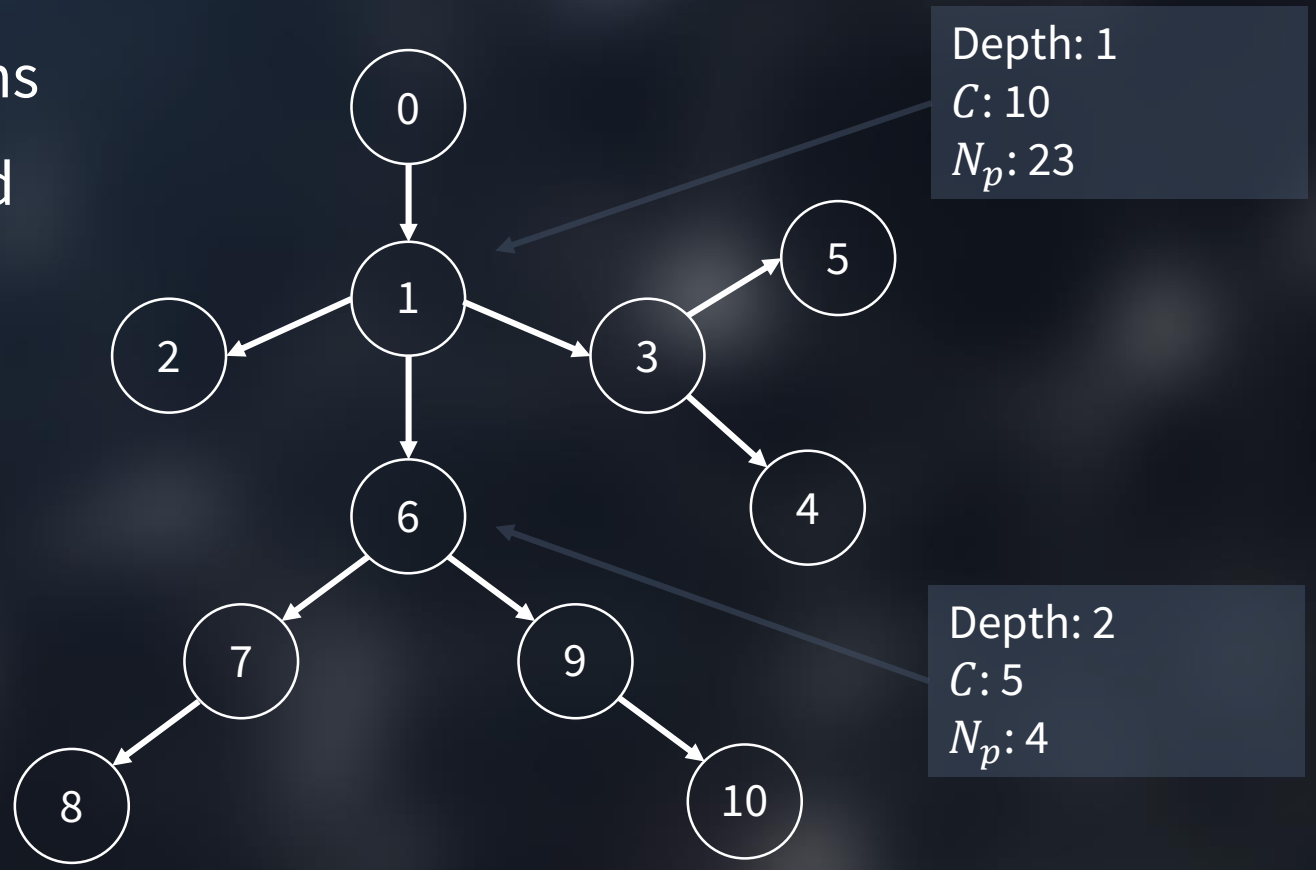
Method 3: The $O(N \log N)$ Approach

- Compute the arm combinations assuming Node 6 is the hip and Node 1's N_p is 23:

$$N_p - (C_1 - 1 - C_6) \times C_6$$

$$23 - (10 - 1 - 5) \times 5 = 3$$

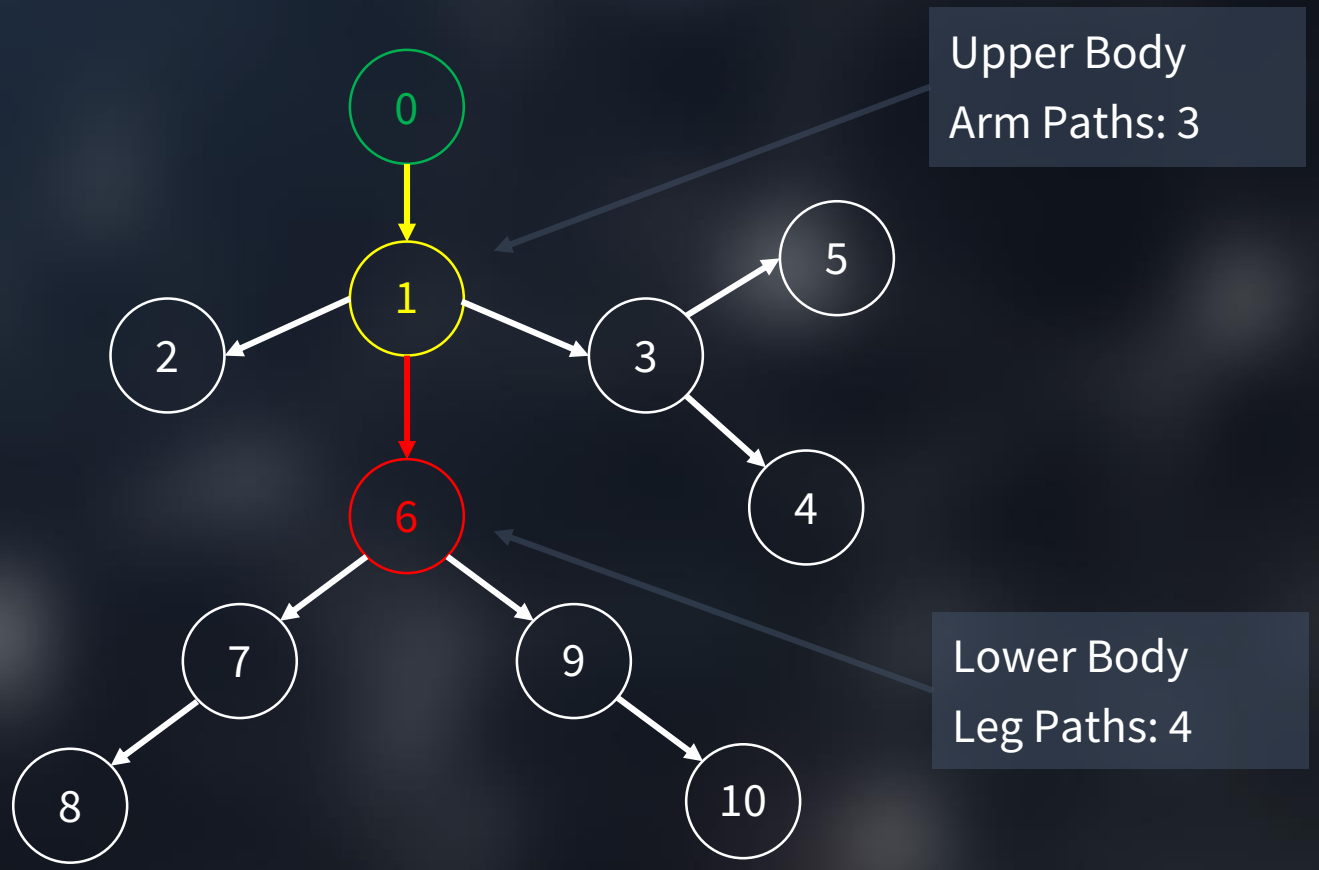
- Proof by Cosmo. It just works.



Method 3: Example

- Body 1
  - Head Combos: 1
  - Arm Combos: 3
  - Leg Combos: 4

$$c_1 = 1 \times 3 \times 4 = 12$$

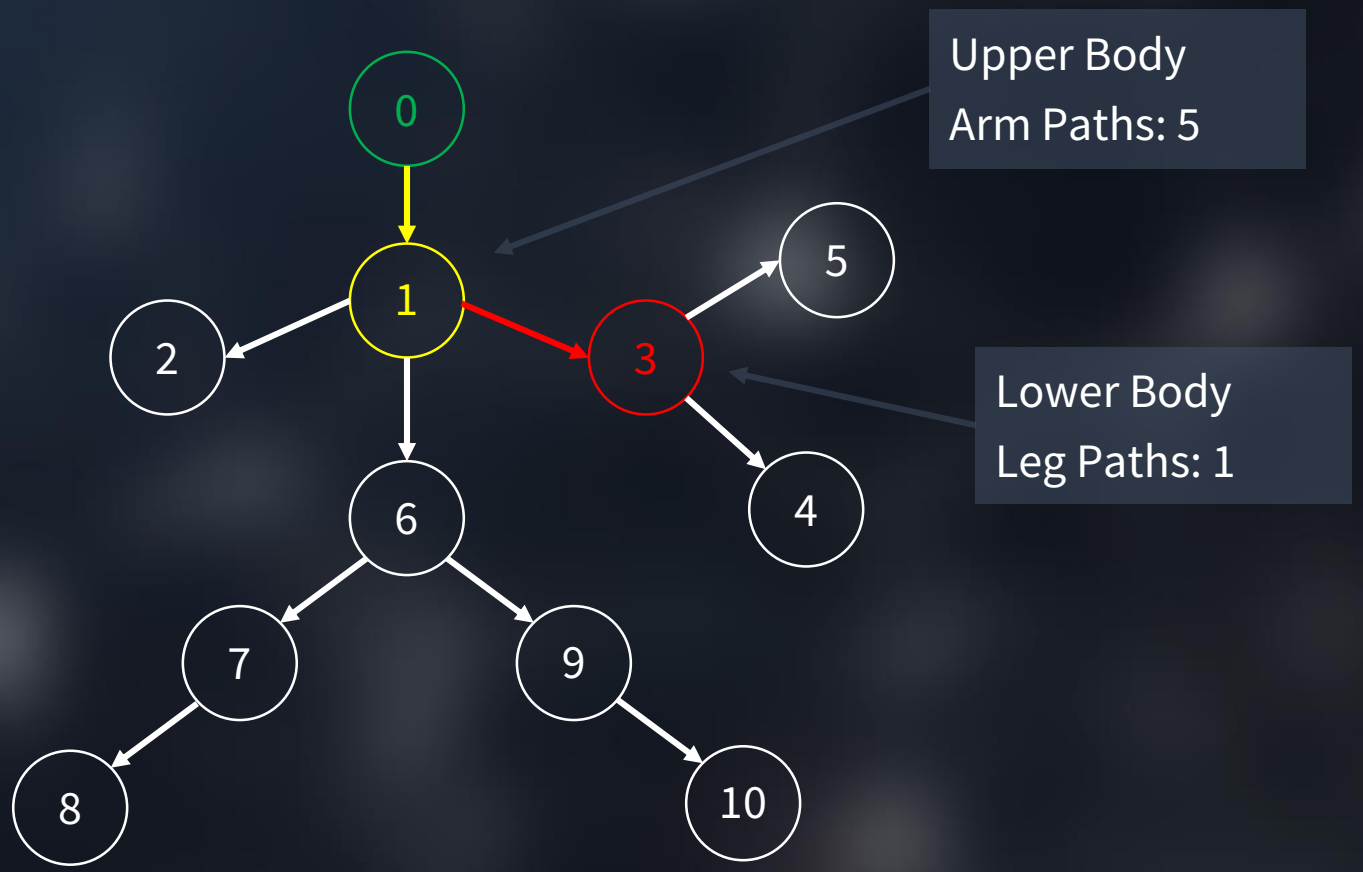


Method 3: Example

- Body 2
  - Head Combos: 1
  - Arm Combos: 5
  - Leg Combos: 1

$$c_1 = 1 \times 3 \times 4 = 12$$
$$c_2 = 1 \times 5 \times 1 = 5$$

$$c_1 + c_2 = 17$$



Method 3: Time Complexity

- $O(N \log N)$ - Create the graph and probability values
- $O(N)$ - Make a lookup table of combinations
- $O(N)$ - Traverse the list from N – 1 to 0.
  - $O(1)$ - Compute the combinations of legs
  - $O(\log N)$ - Go up and find an upper body
    - $O(1)$ - Compute the combinations of arms and heads
  - $O(1)$ - Multiply combinations of legs, arms, and heads

Total Time: $O(N \log N)$

Performance Comparison

How did the Topcoders do?

- 444 Topcoders opened the problem.
- 114 (25.68%) have submitted a solution.
- 90 (78.95%) of the submissions were correct.
- Overall Percentage was 20.27%
- Best time was 12:00
- Average Correct Time was 31:28

TheTreeAndMan

Topcoder TCO 015, Q1B, 1000-Pointer

Clara Nguyen - COSC 594 - 09/20/2018