# CSE 373

## April 12th – Trees

# ASSORTED MINUTIAE

- **HW2 due tonight**
  - Wrong submissions
- **Static functions for your test cases**
- **HW3 out tonight**
  - Dictionaries – LL, Array, BST
  - Empirical testing
- **Regrade system up by Friday**

# LECTURE STYLE

- **Too fast**
  - More time-in-class examples
  - Lots of material to cover, but it isn’t doing any good if no one understands
- **Too mathy**
  - A bit tougher, abstract concepts are the most important part of the course
  - More physical examples

# TODAY

- Review of Dictionaries
- BSTs as dictionaries
- Analysis of BSTs
- Tree traversals

# DICTIONARY

- **Data is inserted in <Key,Value> pairs.**
- **Keys must be comparable**
- **Implements three functions:**
  - Insert (key, value)
  - Find (key)
  - Delete (key)
- **Monday, we discussed 4 implementations**

# IMPLEMENTATIONS

- **Simple implementations**

| | `insert` | `find` | `delete` |
|---|---|---|---|
| Unsorted linked-list | *O(1)** | *O(n)* | *O(n)* |
| Unsorted array | *O(1)** | *O(n)* | *O(n)* |
| Sorted linked list | *O(n)* | *O(n)* | *O(n)* |
| Sorted array | *O(n)* | *O(log n)* | *O(n)* |

* Unless we need to check for duplicates

# IMPLEMENTATIONS

- **Other implementations?**
  - Binary Search Tree (BST)
  - Sort based on keys (which have to be comparable)
  - How do we implement this?
  - What changes need to be made?
    - *Discuss how your 143 BST is different from a dictionary BST*
    - *Consider particularly how the BST Node changes*

# IMPLEMENTATIONS

- **BST Node:**
  - Before:
    - Node left
    - Node right
    - Value data
  - Now?
    - Node left
    - Node right
    - Key k
    - Value v

# IMPLEMENTATIONS

- **BST Changes:**
  - Insert(), find() and remove() remain similar
  - Key is the primary comparison
  - Value is attached to the key
  - **Dictionary fact: All values have an associated key**
  - **For now, assume all keys are unique, i.e. each key only has one value**

# IMPLEMENTATIONS

- **BST Analysis:**
  - What is our time for the three functions?
    - Insert()? Delete()? Find()?
    - *Take 5 minutes to discuss*
    - *Consider average and worst-case.*
    - *What are the inputs for average and worst-case?*

# IMPLEMENTATIONS

- **BST Analysis:**
  - Insert():
    - Worst case: O(n)
    - Average case: O(height)
      - What is “average” data?
    - Best case: O(log n)

# AVERAGE CASE

- **Interesting concept**
  - Average to the user?
  - Average among all possible inputs?
- **Random data trials**
  - Produce random test cases and observe the result
    - Timer?

# AVERAGE CASE

- **Timing cases**
  - Advantages and disadvantages
    - + Actual runtime performance
    - – Can be skewed
    - + Easy to implement
    - – Difficult to ascertain asymptotic growth

# AVERAGE CASE

- **HW3 will have you do timing cases**
  - Many runs will reduce hardware uncertainty
  - Running at many sizes will make trends more apparent
  - Demonstrate some real implementation behavior

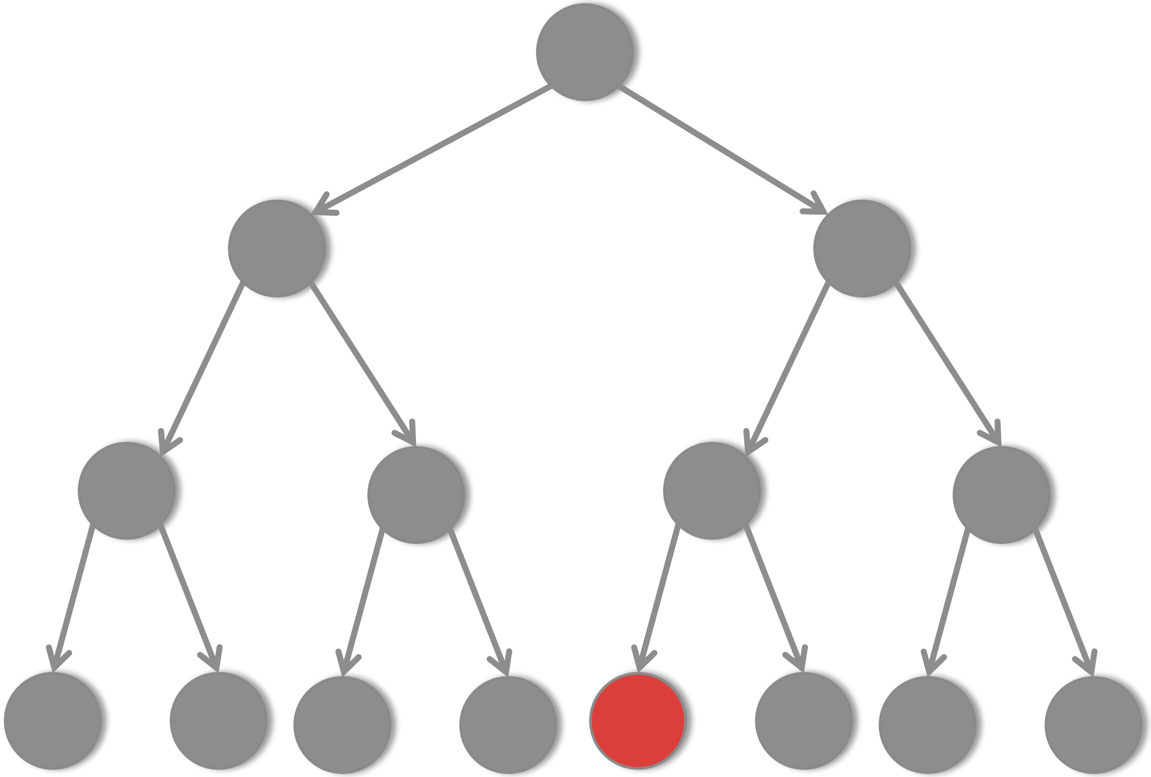
# BINARY SEARCH TREE

- **Back to BST**
  - How do we test average case for a dictionary?
  - We want varied input, without repititions
  - One solution:
    - Create a bunch of keys in a range
    - Select without replacement
    - Add into the dictionary
  - “Shape” of a dictionary is determined by insert() order, so ordering is critical.

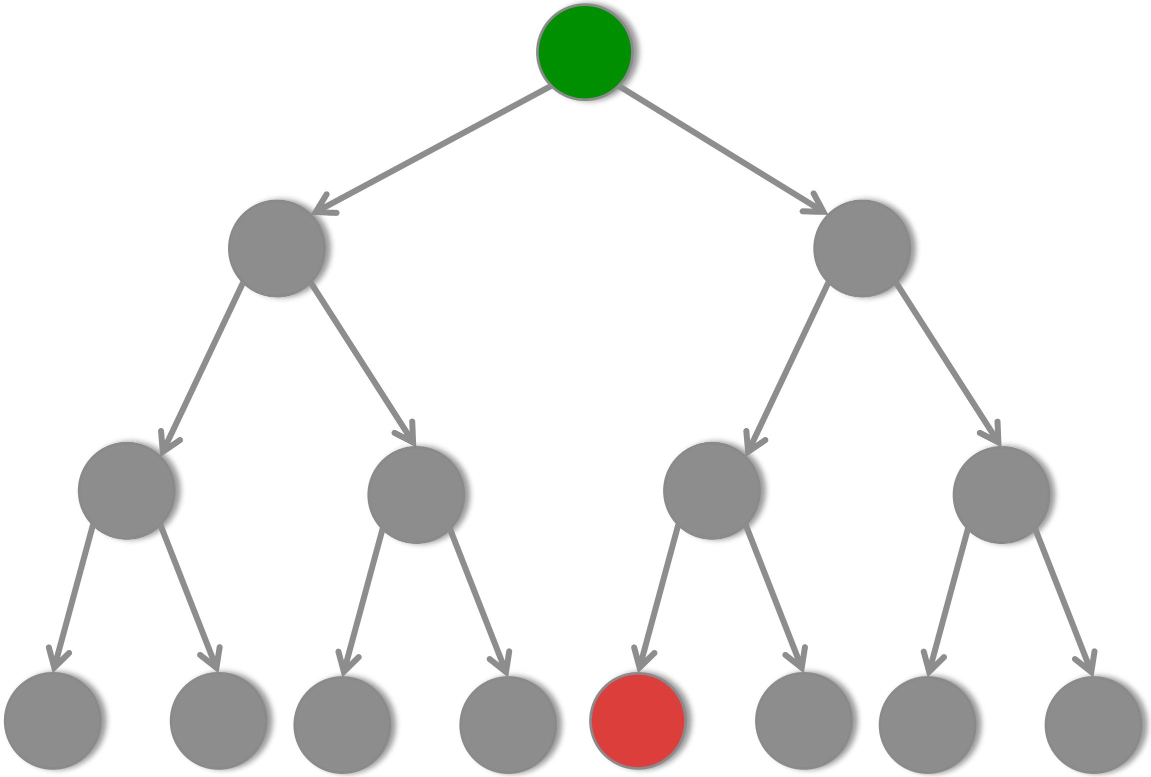
# INTRO FOR NEXT WEEK

- **Tree traversals**
  - How do we search through a tree?
  - Multiple ways? What if it isn’t a search tree?

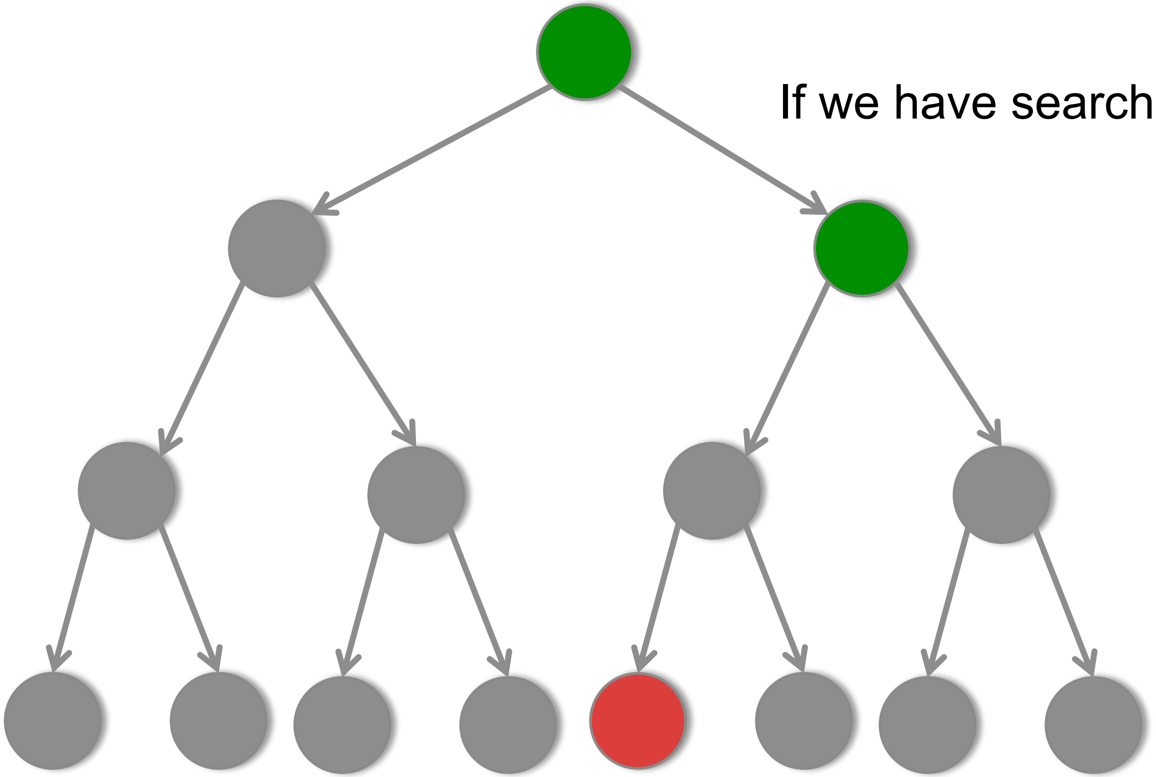
# INTRO FOR NEXT WEEK



# INTRO FOR NEXT WEEK

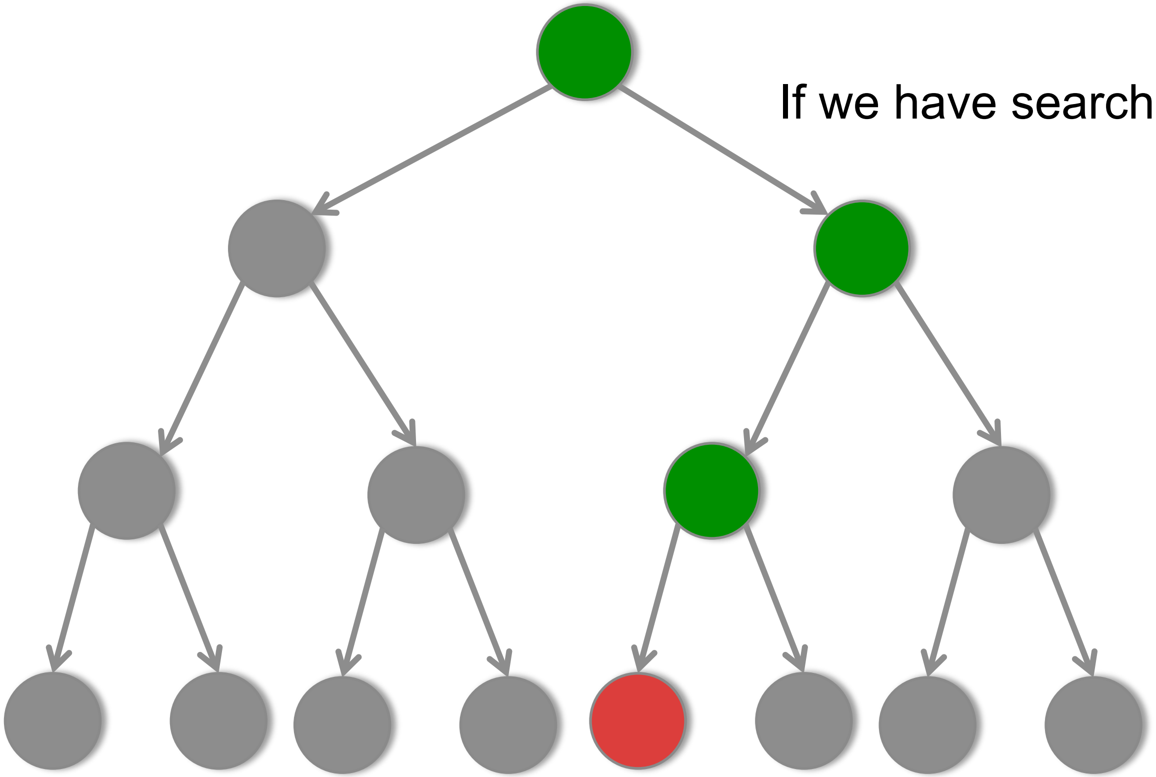


# INTRO FOR NEXT WEEK



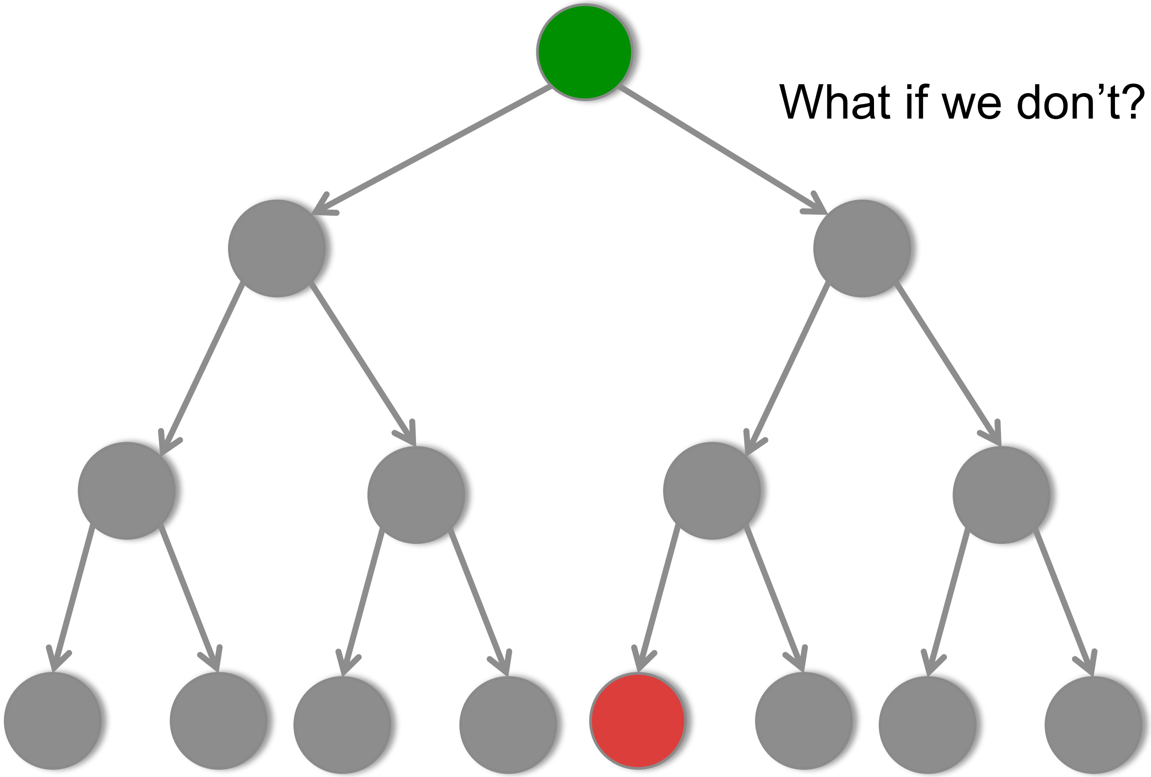
If we have search!

# INTRO FOR NEXT WEEK



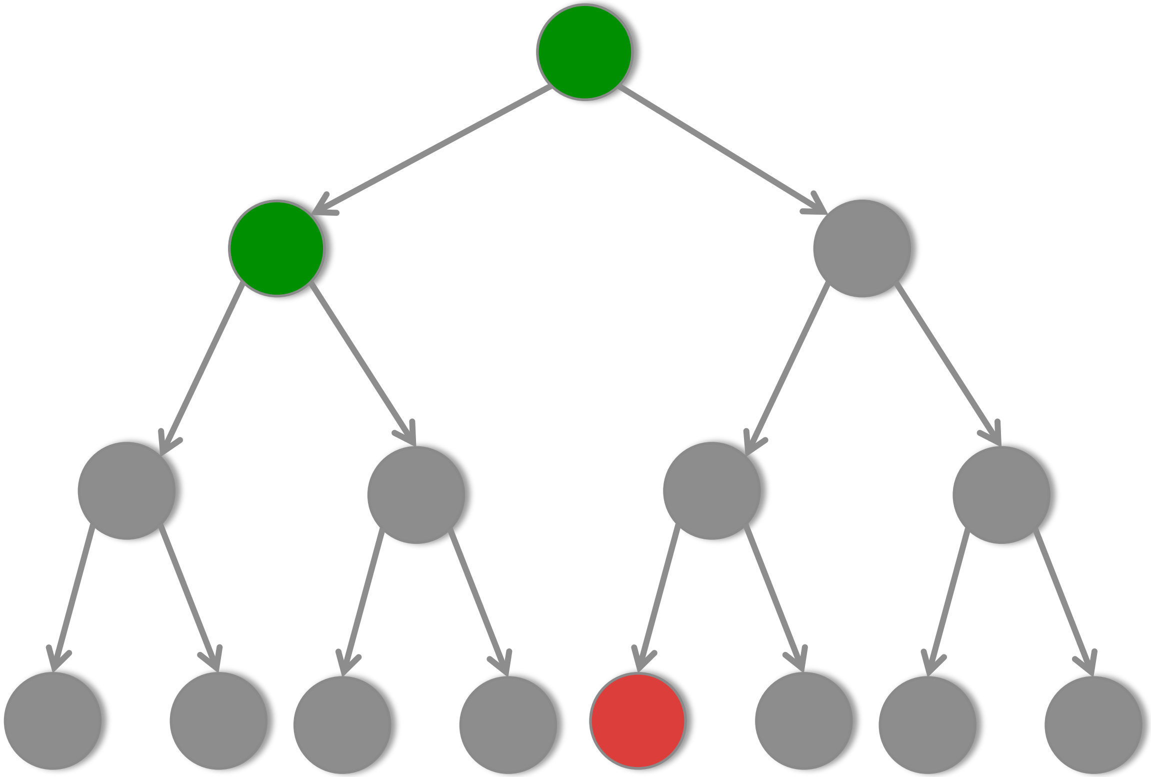
If we have search!

# INTRO FOR NEXT WEEK

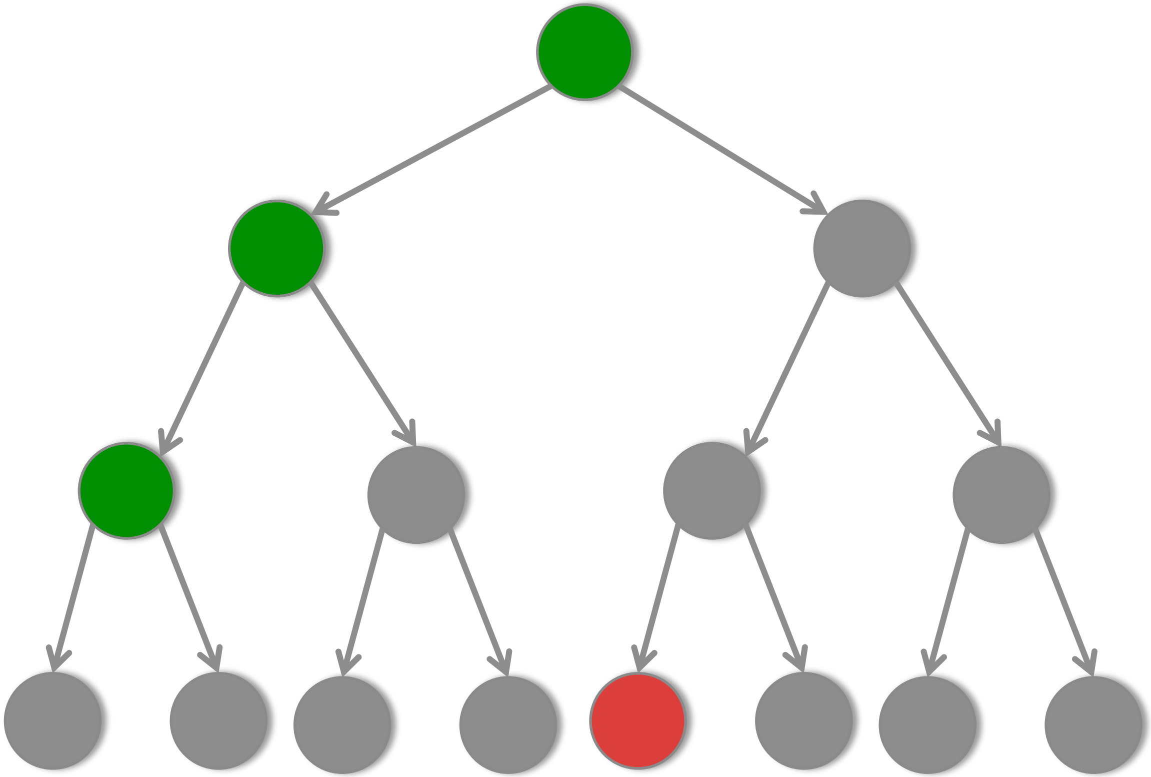


What if we don’t? Say it’s a heap

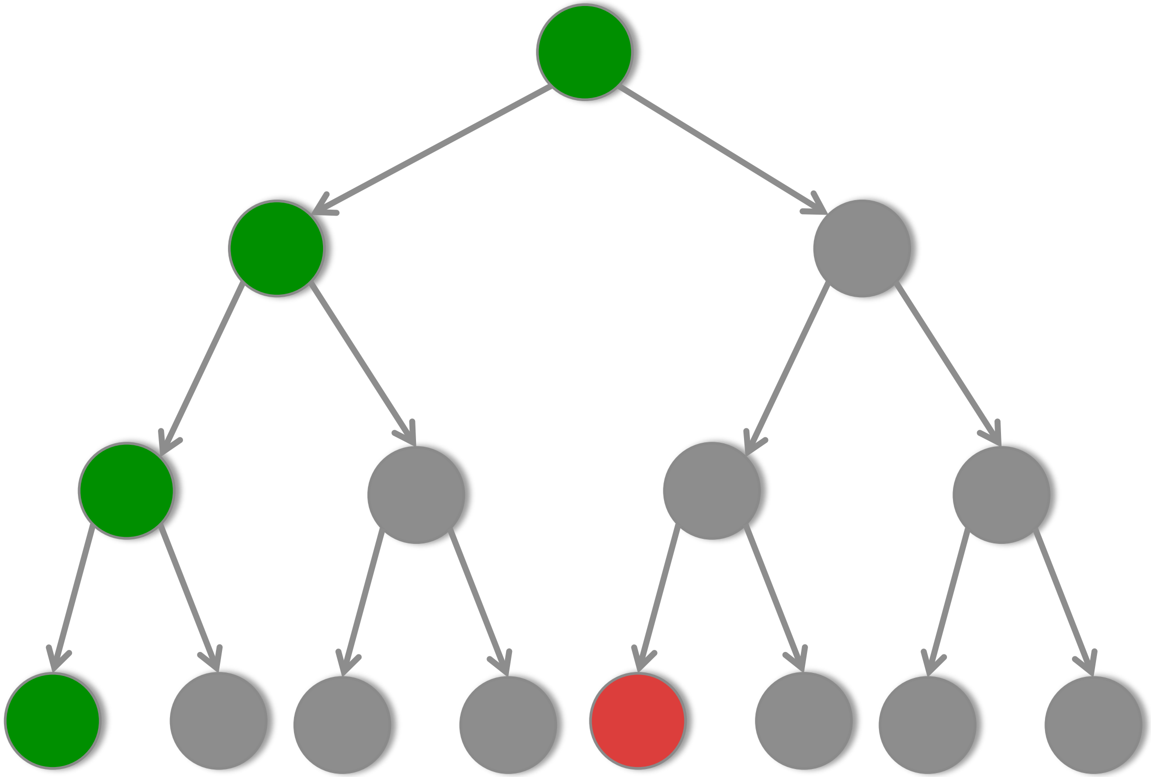
# INTRO FOR NEXT WEEK



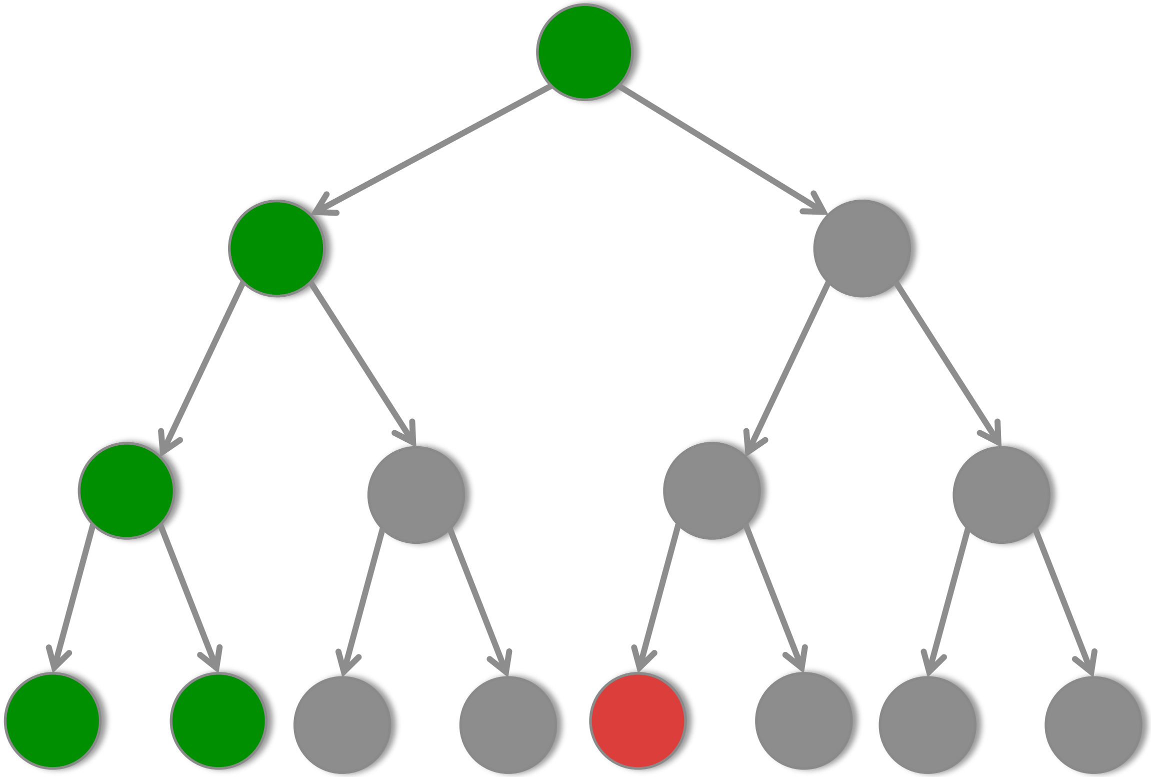
# INTRO FOR NEXT WEEK



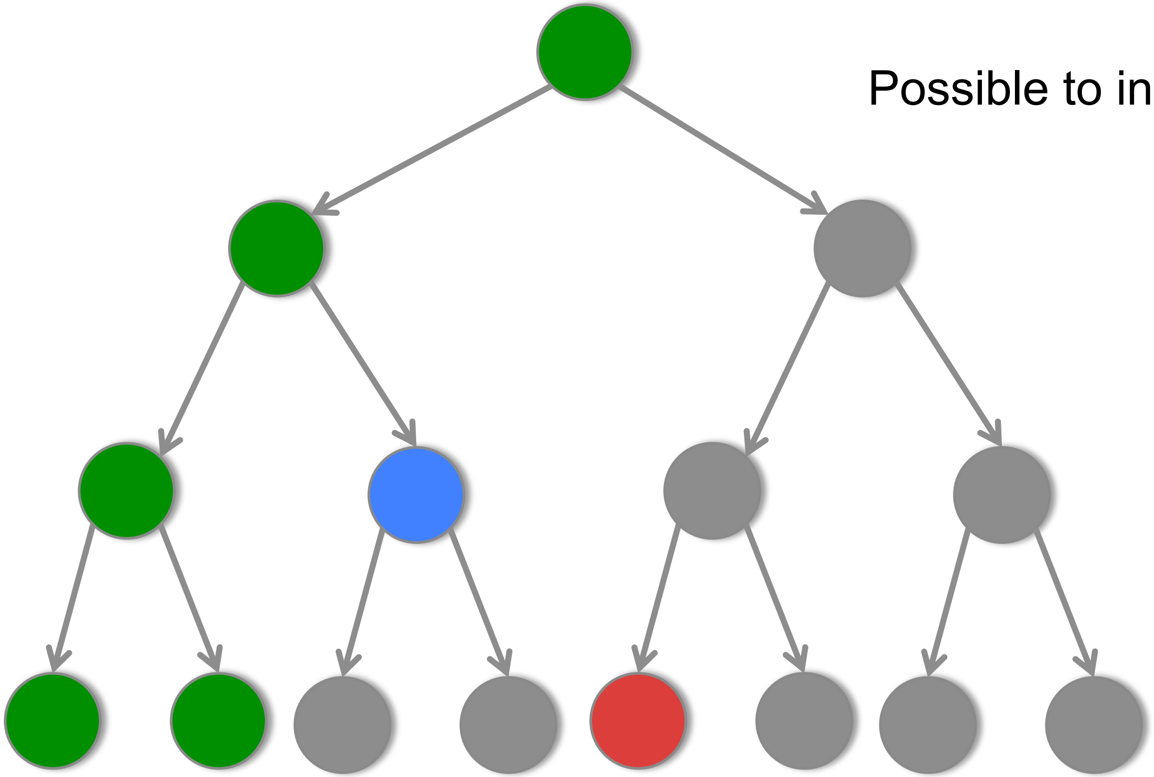
# INTRO FOR NEXT WEEK



# INTRO FOR NEXT WEEK

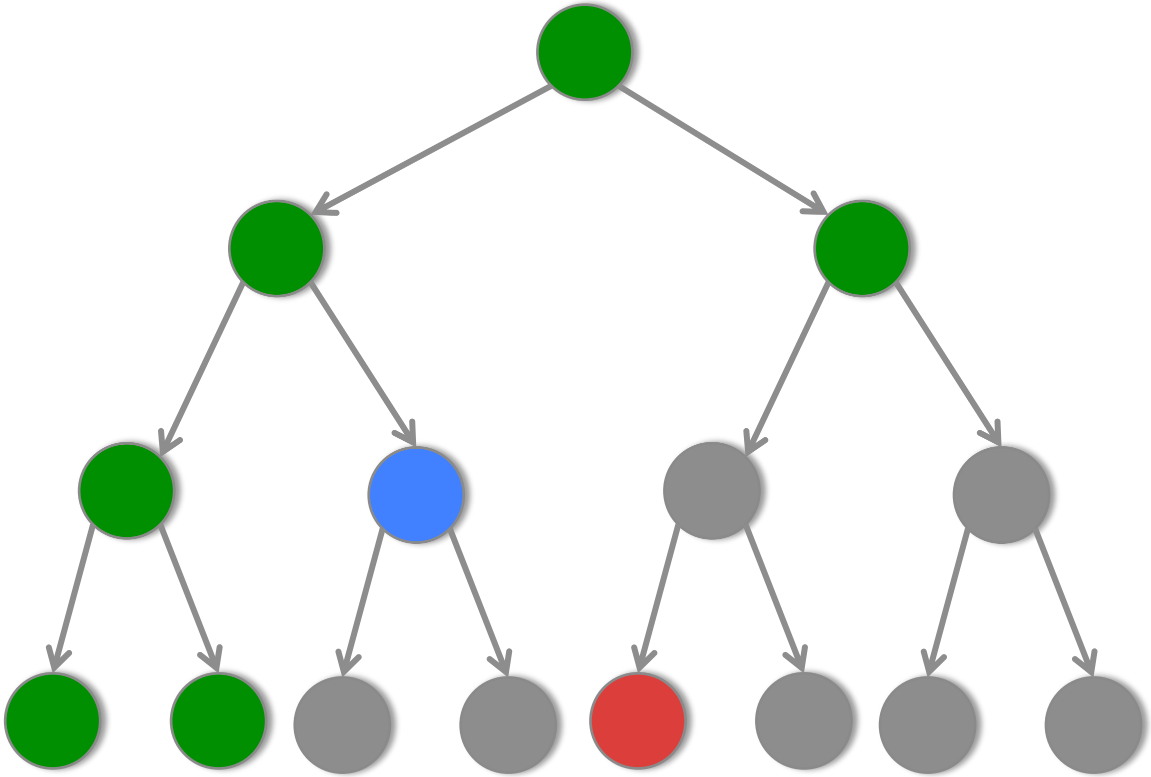


# INTRO FOR NEXT WEEK

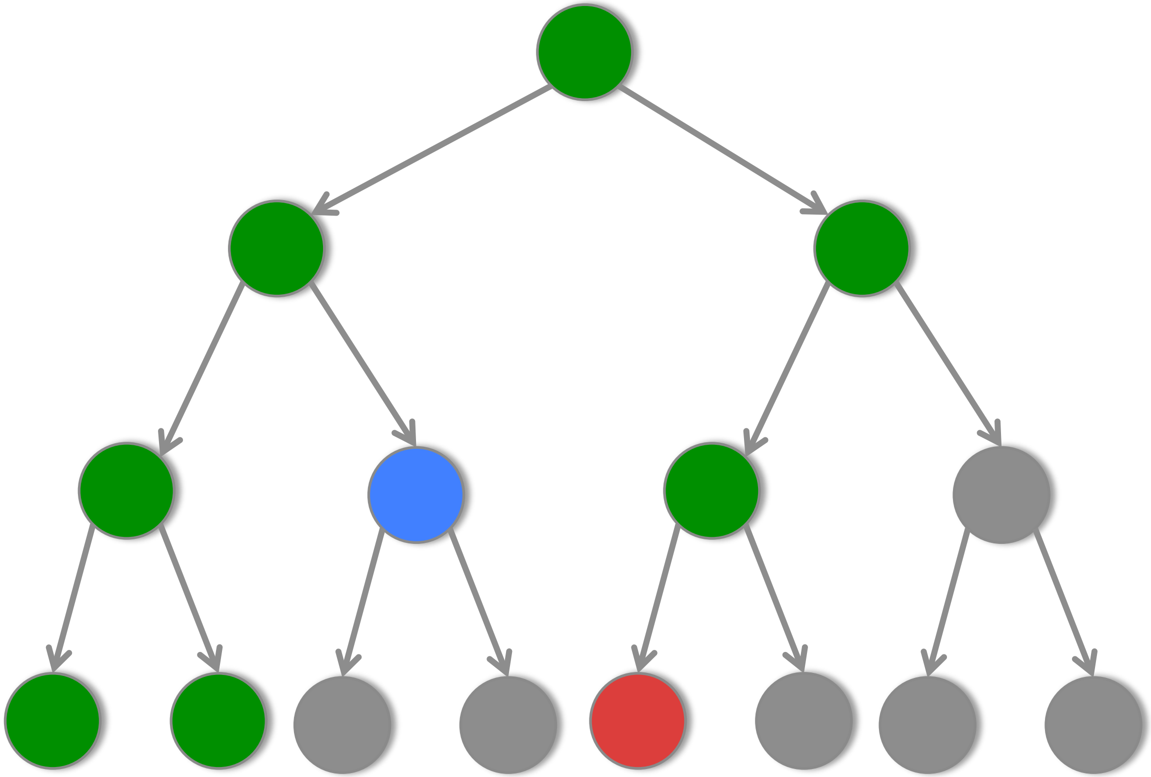


Possible to interrupt early

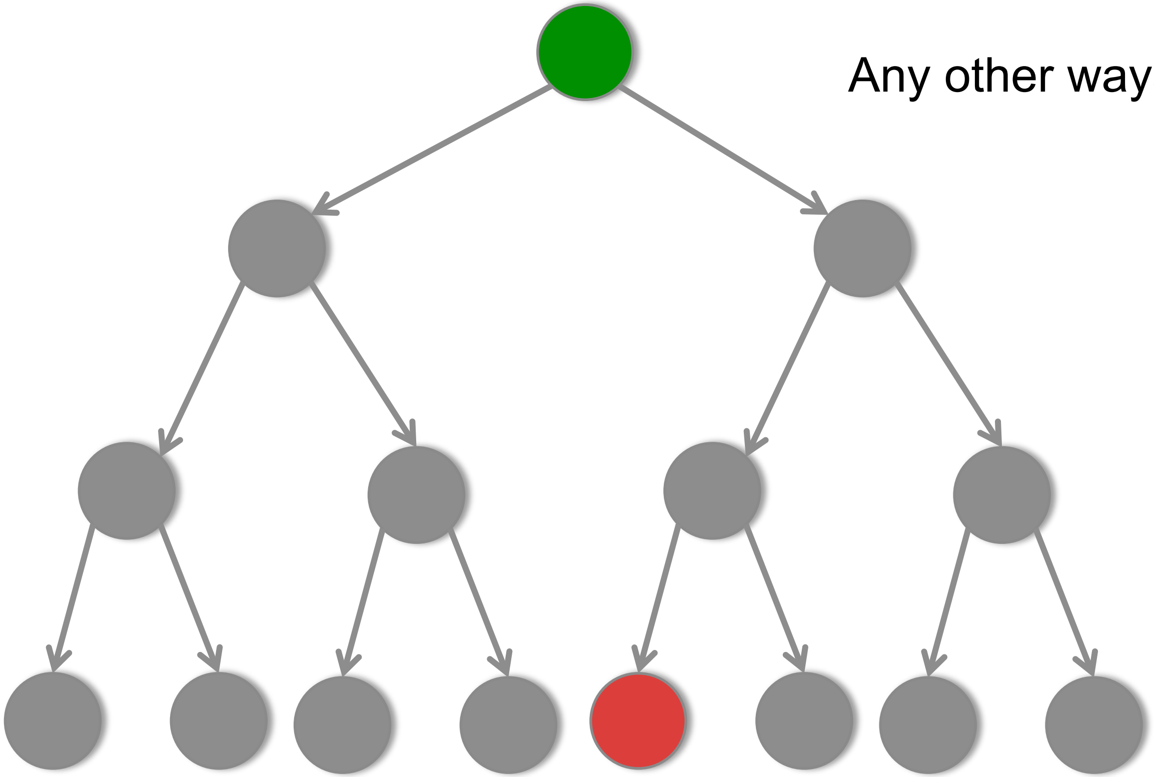
# INTRO FOR NEXT WEEK



# INTRO FOR NEXT WEEK

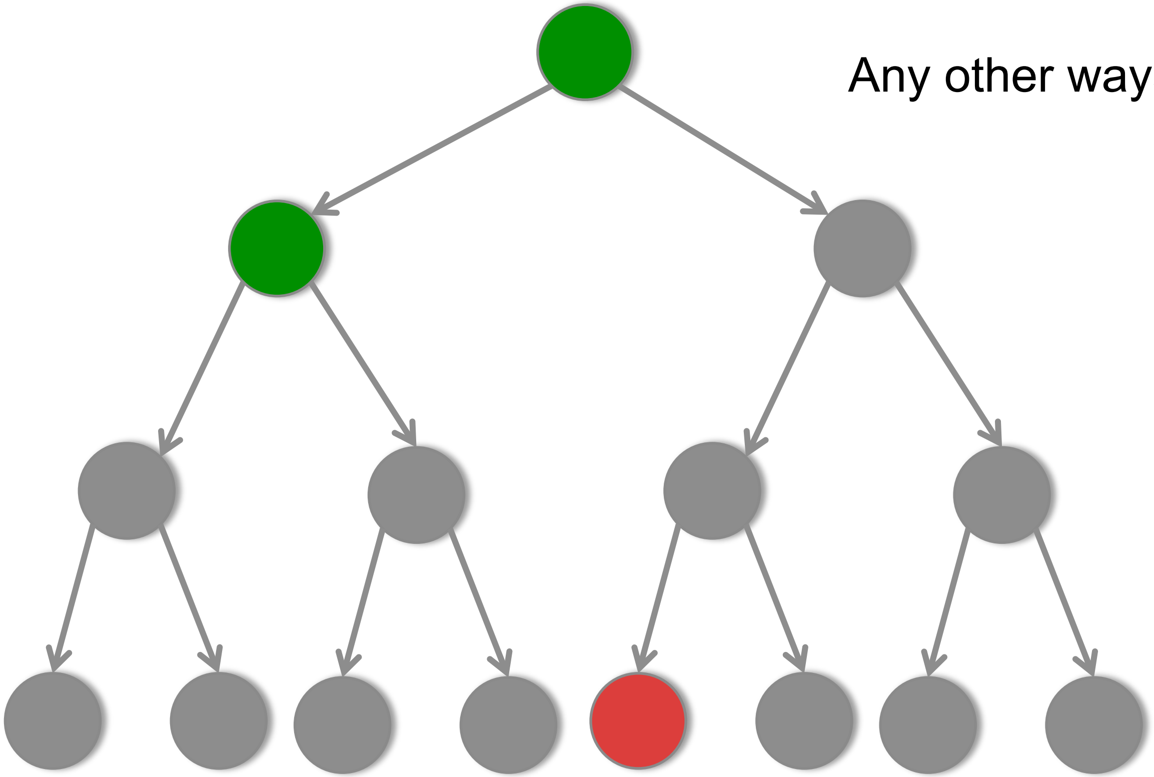


# INTRO FOR NEXT WEEK



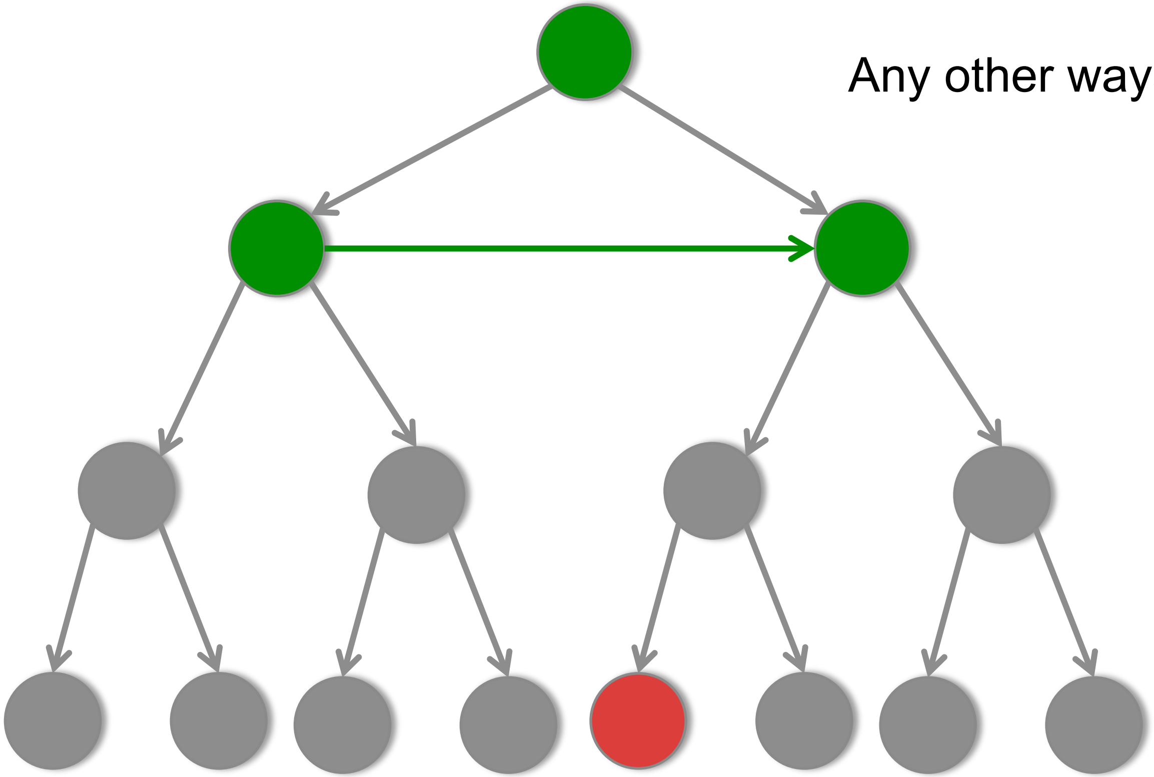
Any other ways?

# INTRO FOR NEXT WEEK



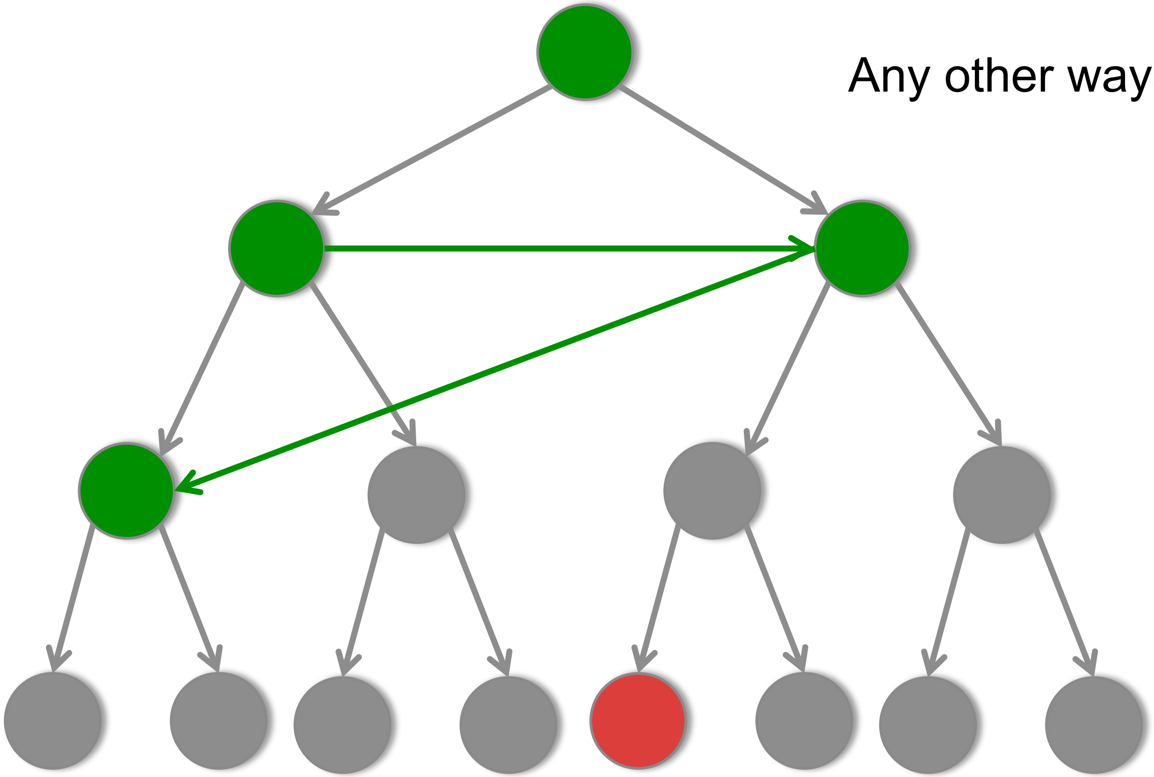
Any other ways?

# INTRO FOR NEXT WEEK



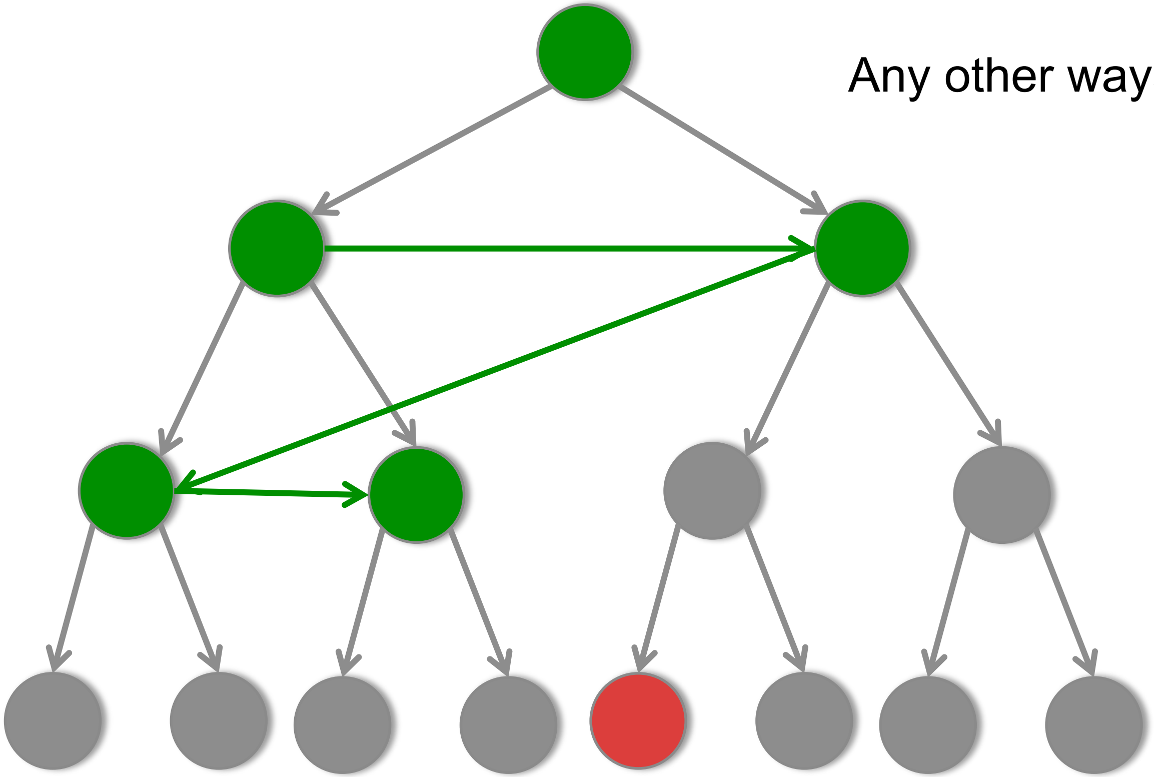
Any other ways?

# INTRO FOR NEXT WEEK



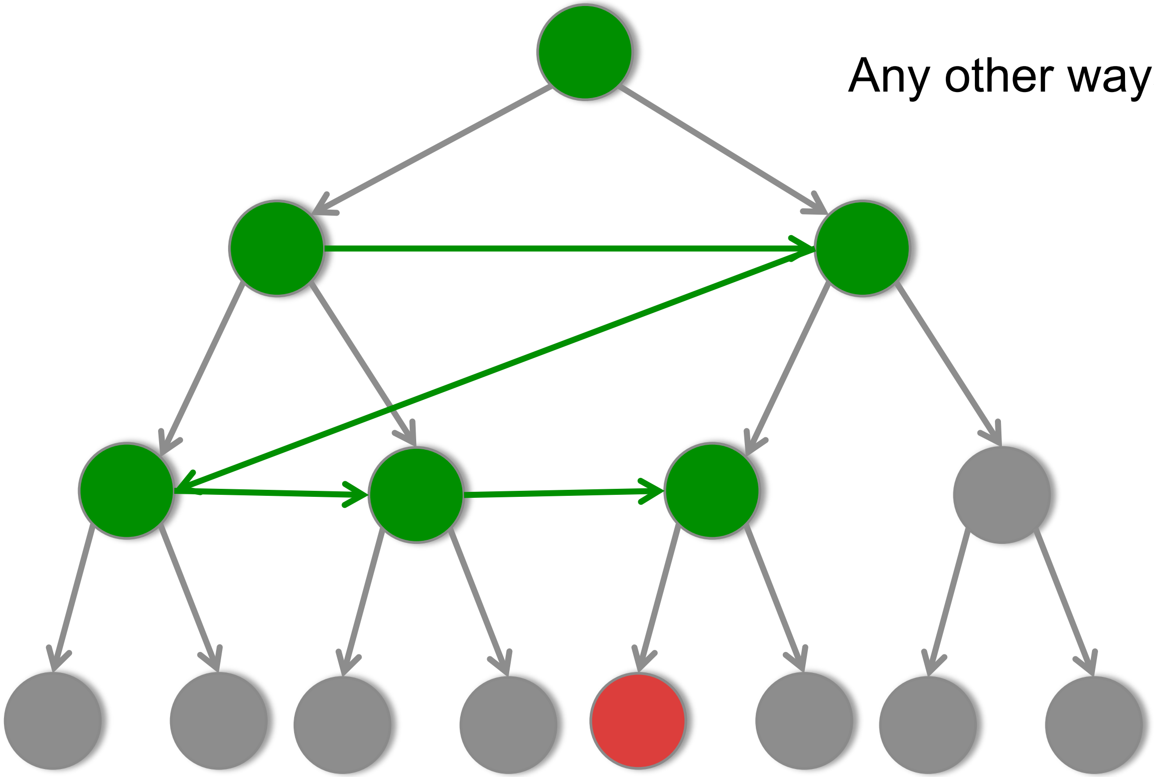
Any other ways?

# INTRO FOR NEXT WEEK



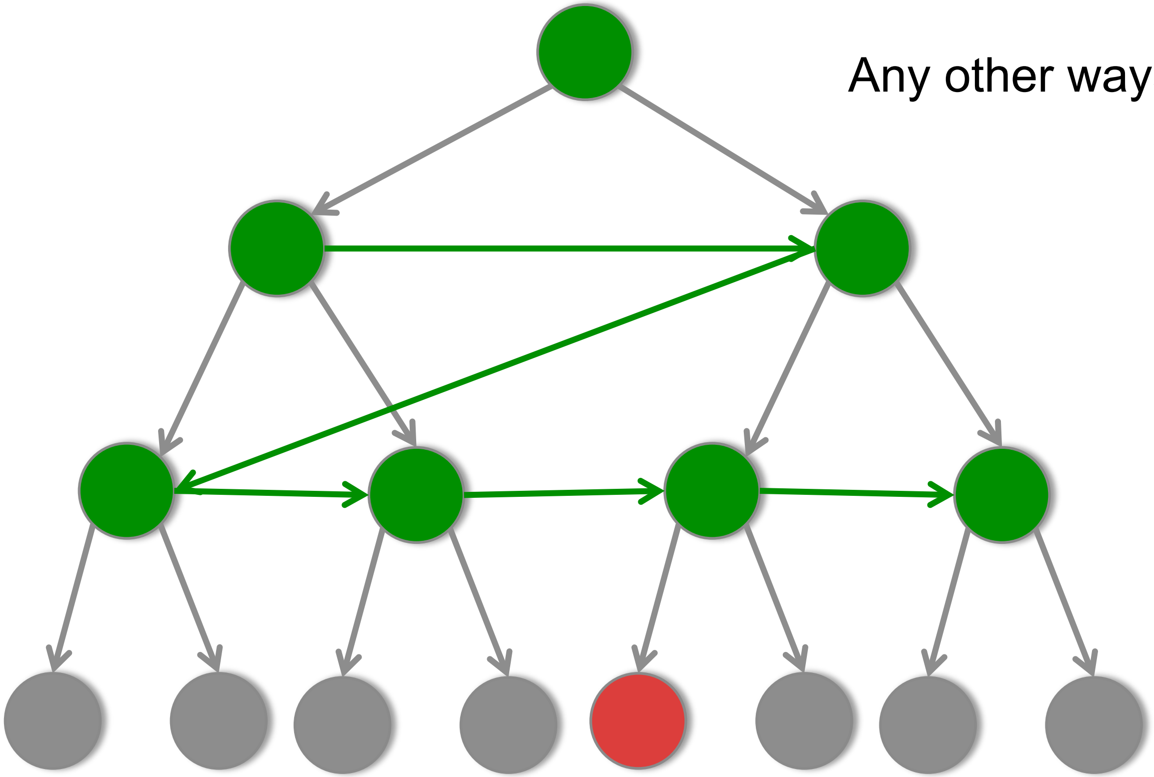
Any other ways?

# INTRO FOR NEXT WEEK



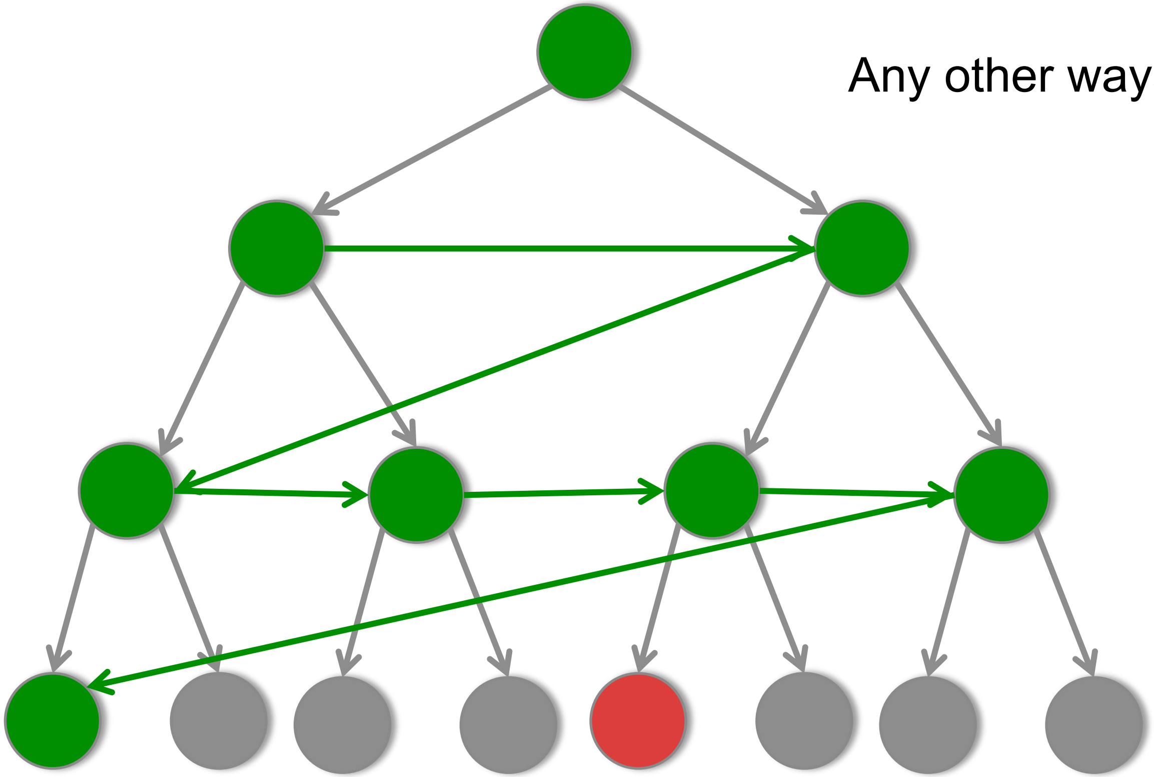
Any other ways?

# INTRO FOR NEXT WEEK



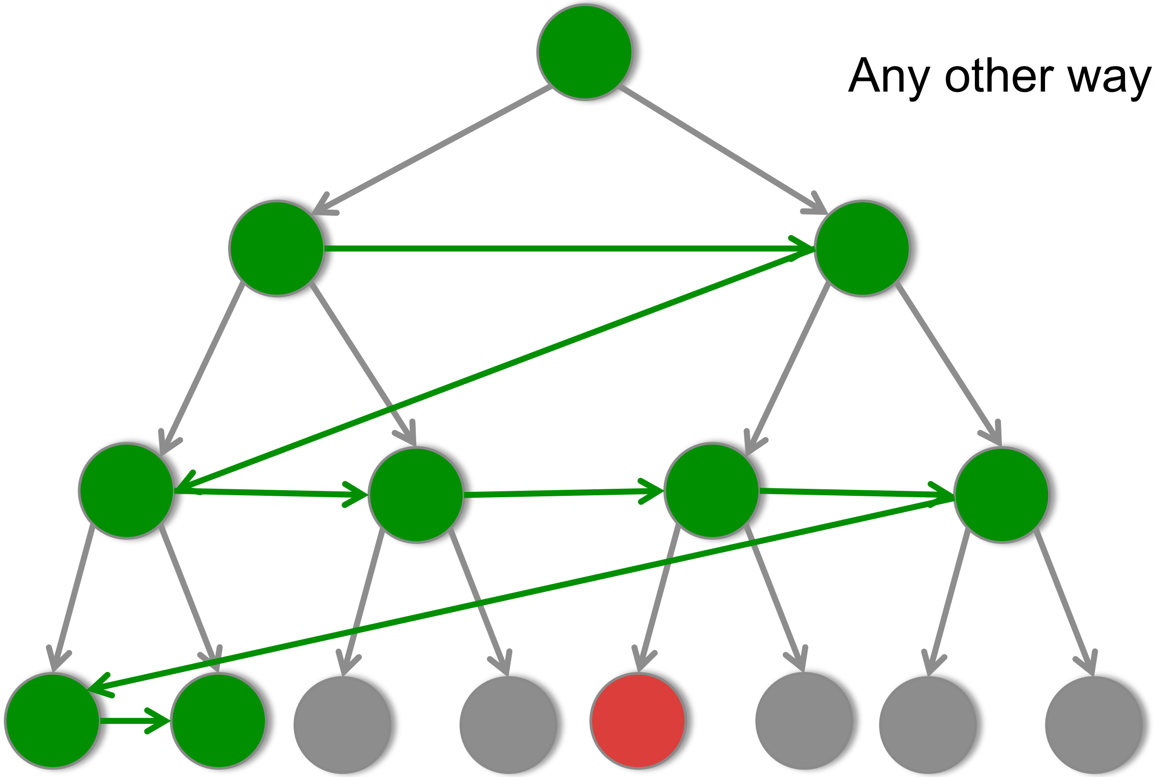
Any other ways?

# INTRO FOR NEXT WEEK



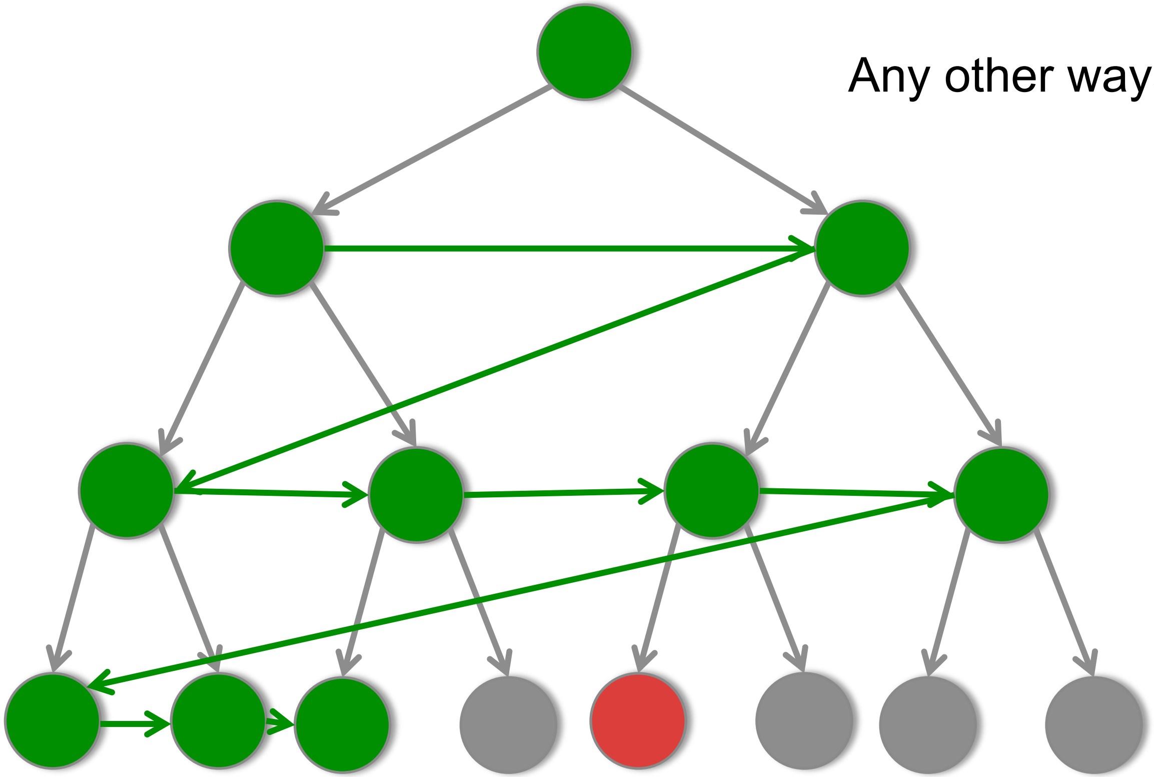
Any other ways?

# INTRO FOR NEXT WEEK



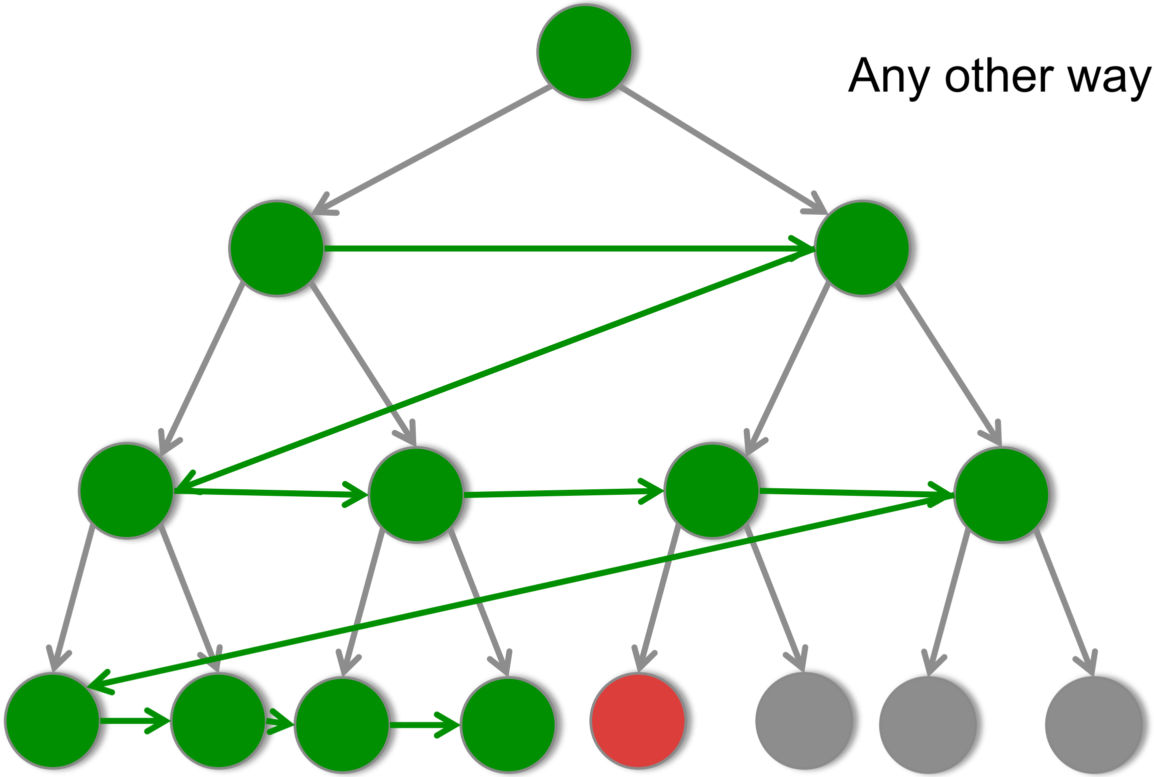
Any other ways?

# INTRO FOR NEXT WEEK



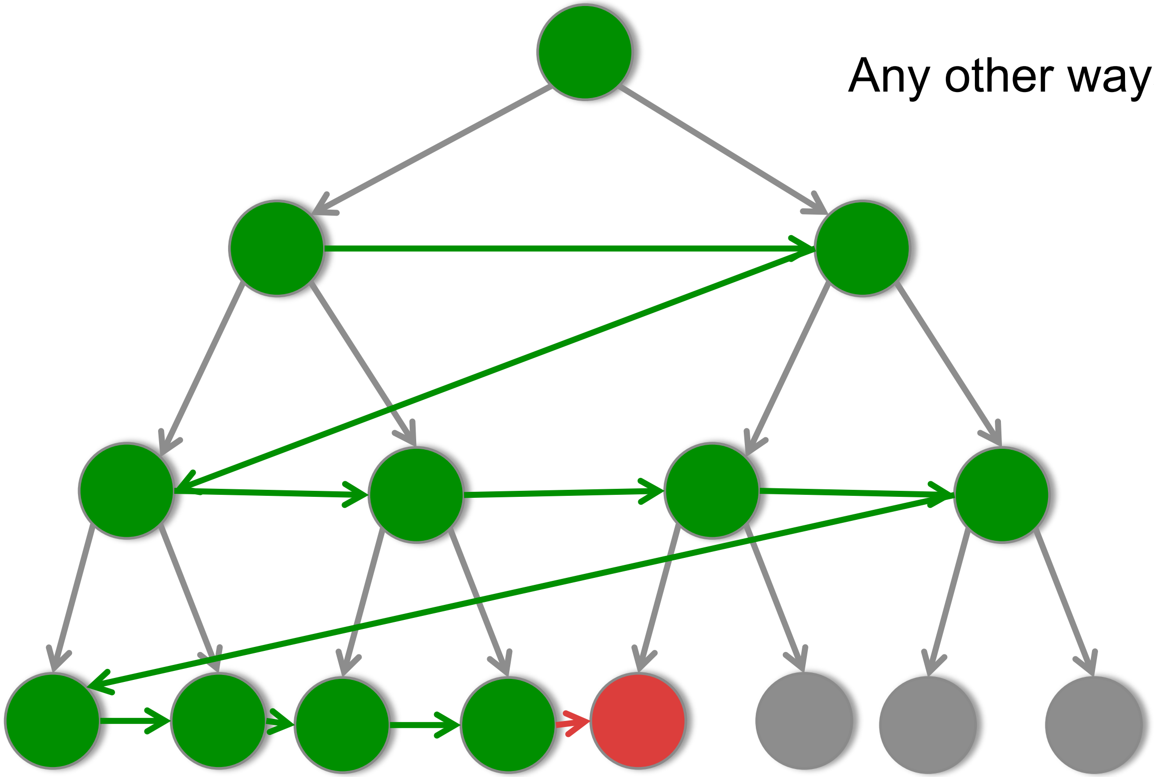
Any other ways?

# INTRO FOR NEXT WEEK



Any other ways?

# INTRO FOR NEXT WEEK



Any other ways?