

# MONTE CARLO TREE SEARCH

ECS170 Spring 2018  
Josh McCoy, @deftjams

# Too Many Paths

Space complexity:  
 $O(\text{moves} * \text{depth})$  or  $O(bm)$

# Monte Carlo Tree Search

- Alpha-Beta Minimax is good for modest branching factor with good heuristic.
- MCTS: no evaluation function/heuristic!
  - Do this many times:
    - Simulate with random moves
    - Score game outcome and keep stats
    - Play move with best stats.


# The Tree Part

Statistics tree construction.

Used as a guide to focus on the most promising parts of the search tree.

Simulations determine utility/value.

# MCTS High Level Process

- 
1. Selection
  2. Expansion
  3. Play-out
  4. Backpropagation

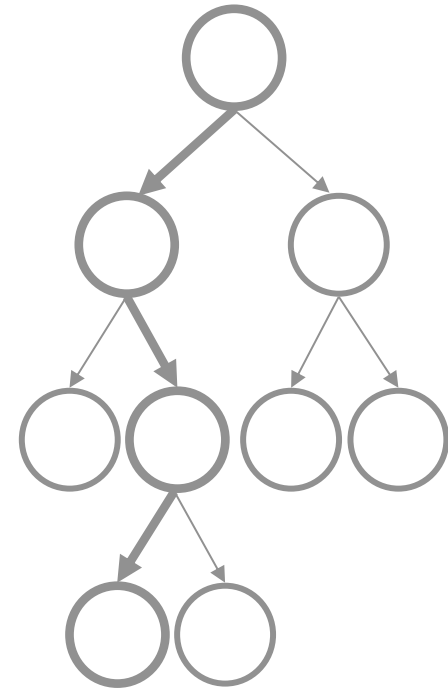
# MCTS High Level Process

1. Selection

2. Expansion

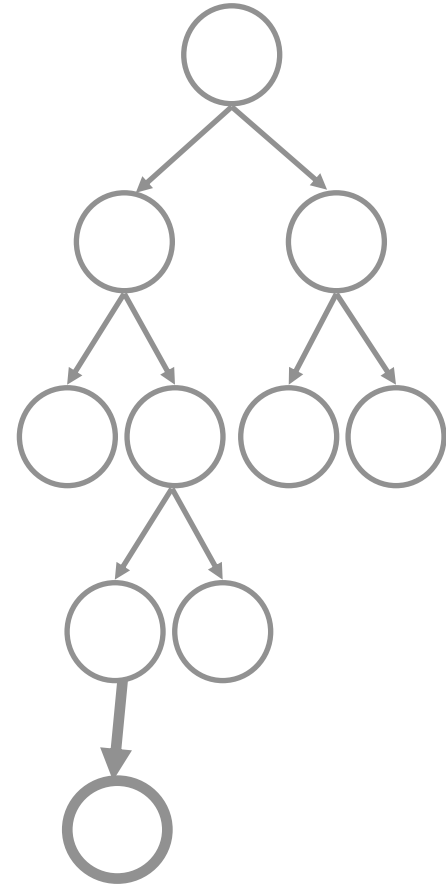
3. Play-out

4. Backpropagation



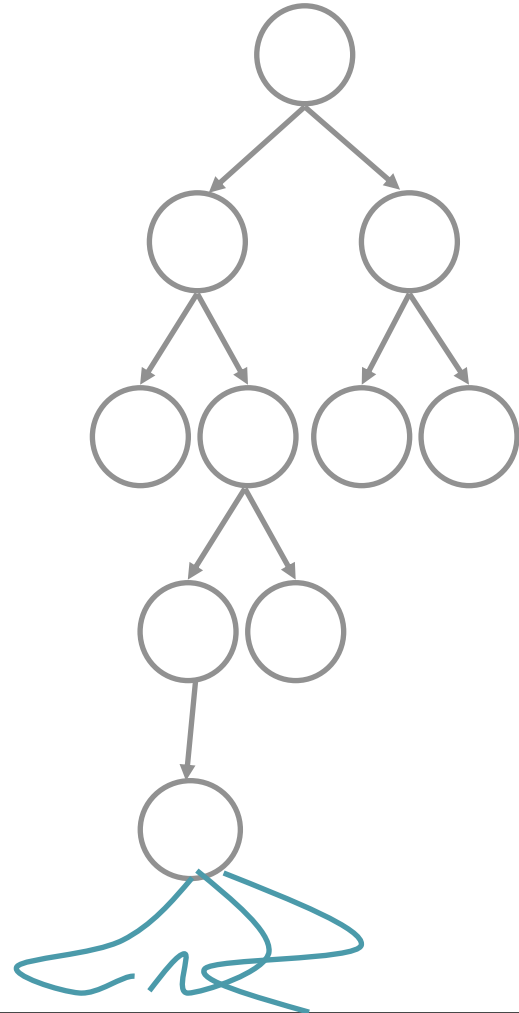
# MCTS High Level Process

1. Selection
2. Expansion
3. Play-out
4. Backpropagation



# MCTS High Level Process

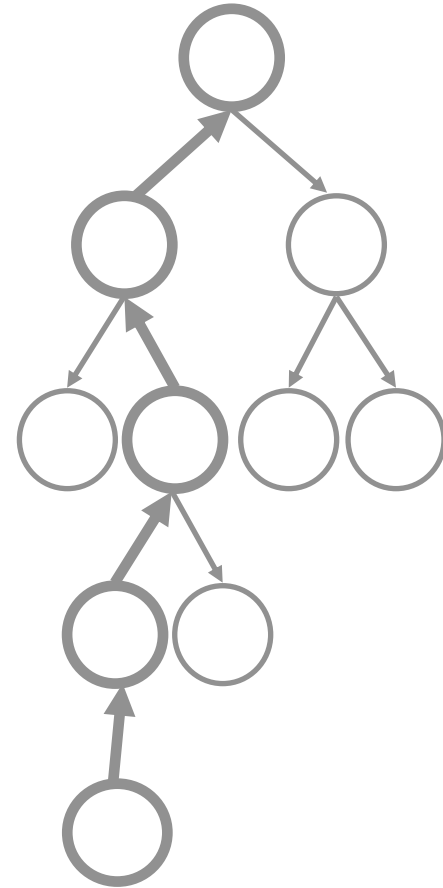
- 1. Selection
2. Expansion
3. Play-out
4. Backpropagation





# MCTS High Level Process

- 1. Selection
2. Expansion
3. Play-out
4. Backpropagation



# Upperbound Confidence

## Multi-armed Bandit



$P(A \text{ wins}) = 0.4$



$P(B \text{ wins}) = 0.55$



$P(C \text{ wins}) = 0.7$



$P(D \text{ wins}) = 0.3$

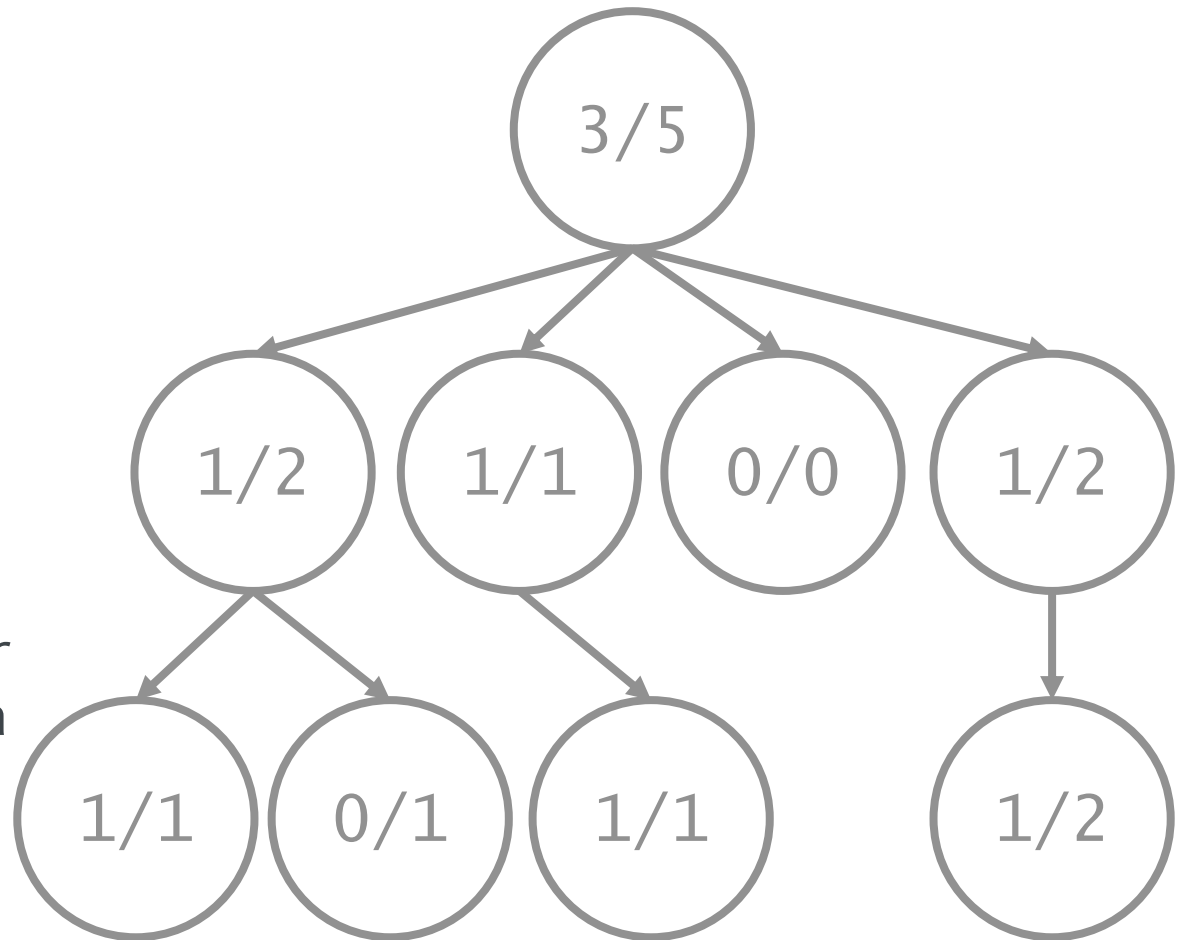
Each play has win/lose binary result.

# Selection

We need:

- Tree of win statistics
- Selection policy

(still alternating player and opponent at each depth)



# Upper Bound Confidence

The diagram illustrates the Upper Bound Confidence equation:  $v_i = x_i + \sqrt{\frac{\ln(N)}{n_i}}$ . Each term is annotated with a colored box and a line pointing to it:  $v_i$  is green and labeled 'value/utility';  $x_i$  is purple and labeled 'mean reward';  $N$  is red and labeled 'total trials';  $n_i$  is blue and labeled 'trials for subtree'.

$$v_i = x_i + \sqrt{\frac{\ln(N)}{n_i}}$$

- Higher value estimate is better
- Gives an estimate of true value.
- More trials makes better estimate

`v_i = x_i + math.sqrt(math.log(N)/n_i)`

# Upper Bound Confidence

$$v_i = x_i + \sqrt{\frac{\ln(N)}{n_i}}$$

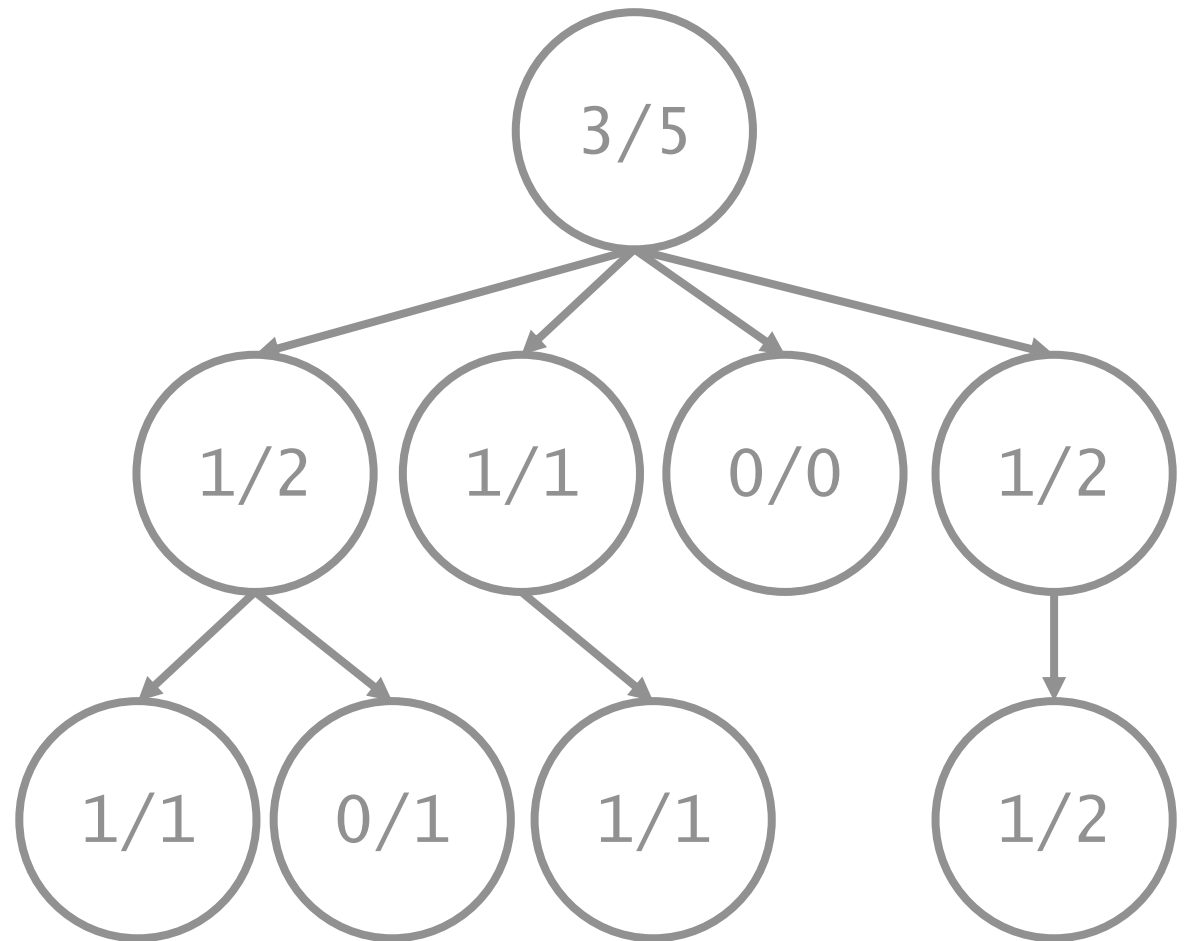
The diagram illustrates the Upper Bound Confidence formula with color-coded labels and boxes:

- $v_i$  (green) is labeled "value/utility" (green box).
- $x_i$  (purple) is labeled "mean reward" (purple box).
- $\ln(N)$  (red) is labeled "total trials" (red box).
- $n_i$  (blue) is labeled "trials for subtree" (blue box).

- Confidence interval is large when  $n_i$  is small, shrinks in proportion to  $\sqrt{n_i}$ 
  - High uncertainty about move, larger **exploration** term
- Explore: if trials is less than total trials.
- Tolerant to adversarial situations.

# Selection

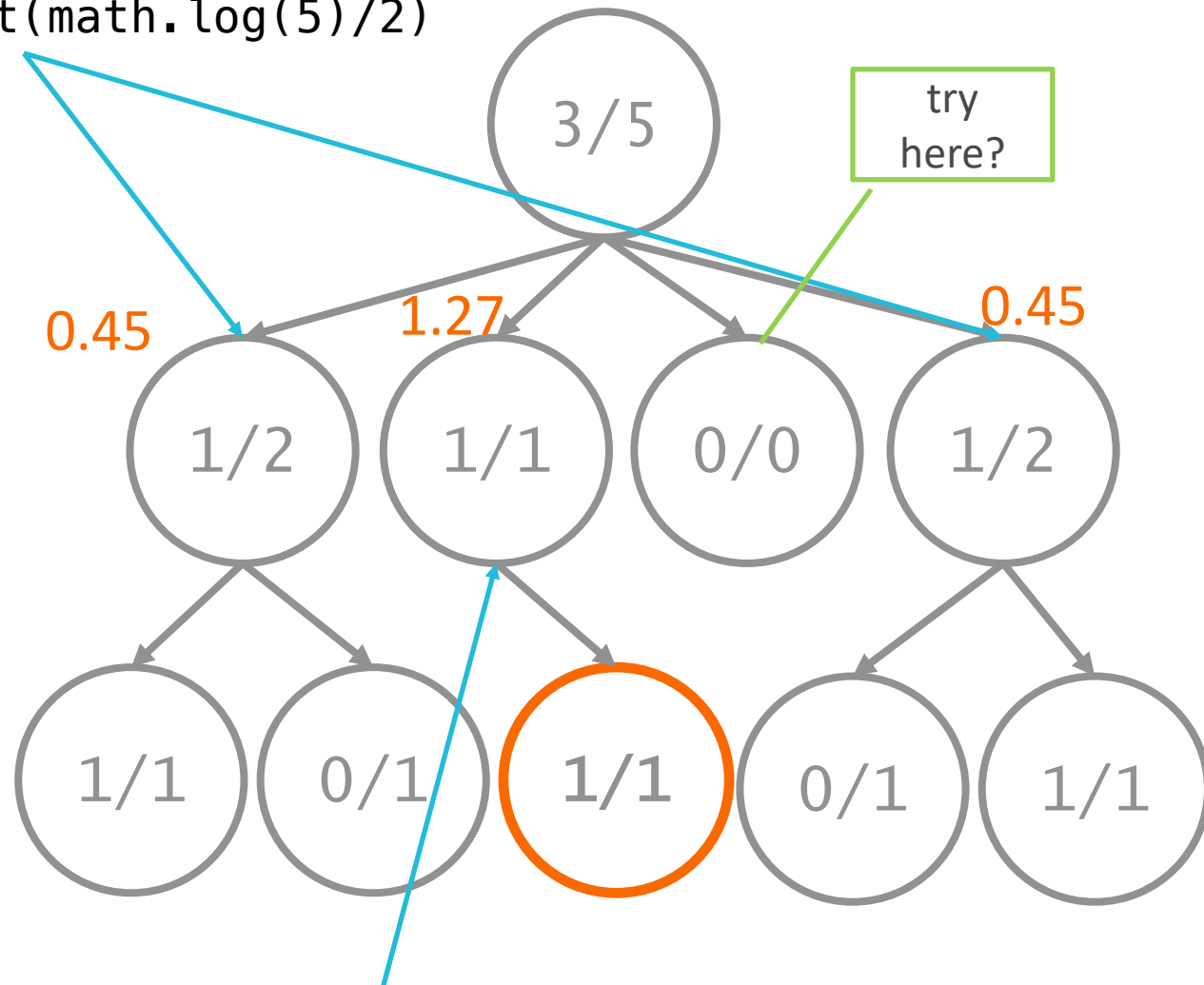
Select from UCB  
value



# Selection

$$1 * 1/2 * \text{math.sqrt}(\text{math.log}(5)/2)$$

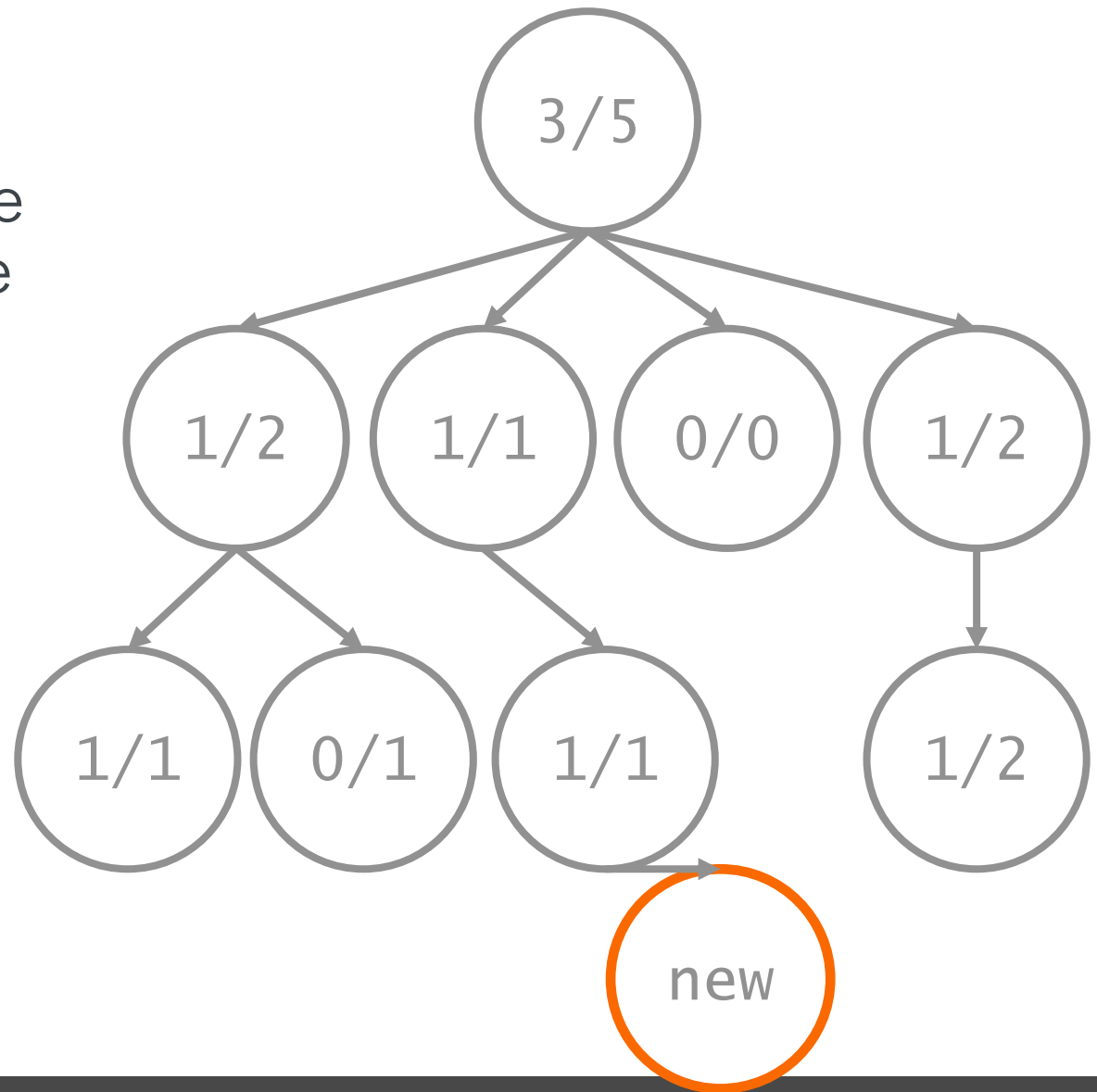
Select from UCB  
value



$$1 * 1/2 * \text{math.sqrt}(\text{math.log}(5)/2)$$

# Expansion

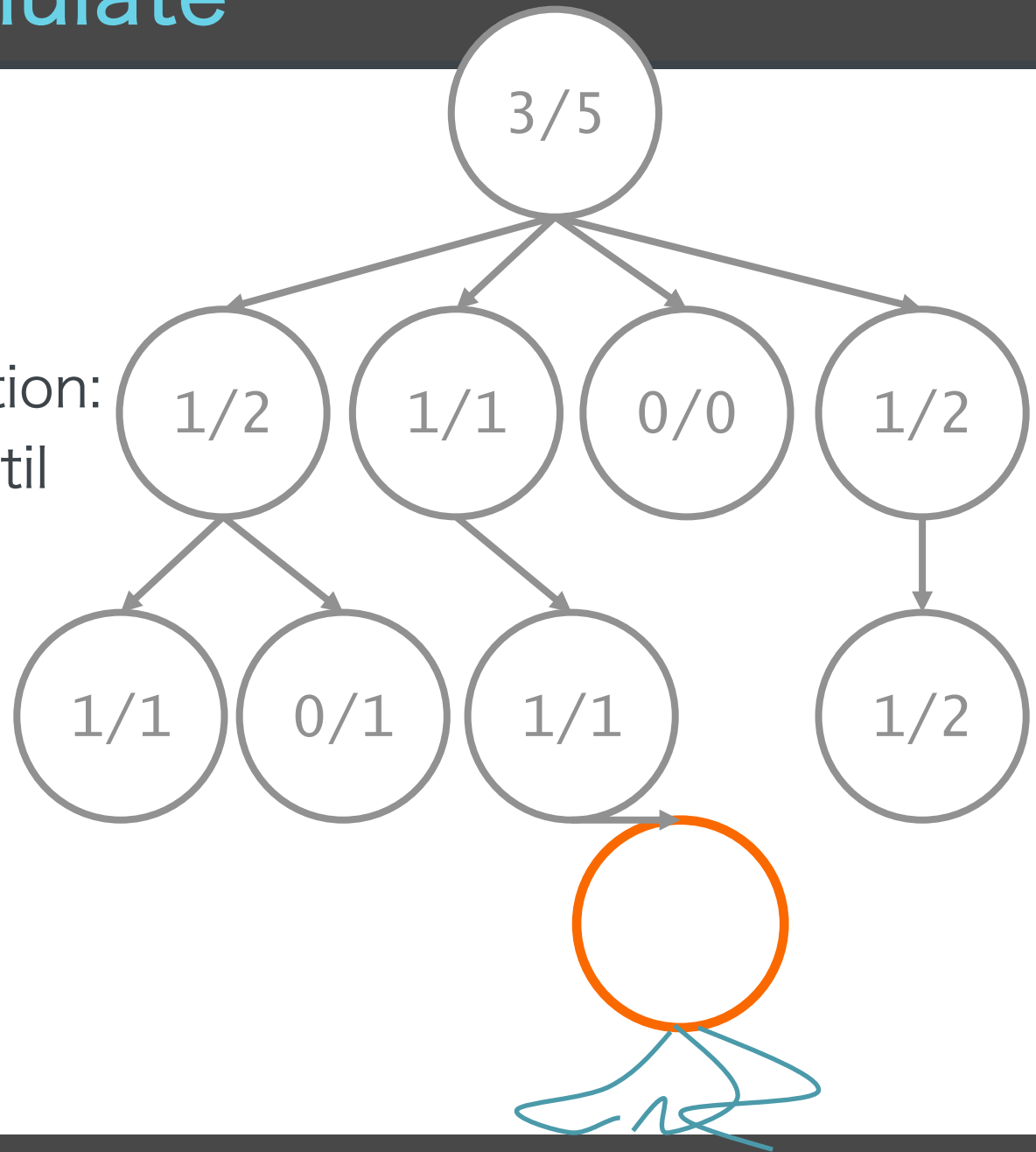
Create a new node on the search tree based off UCB1.





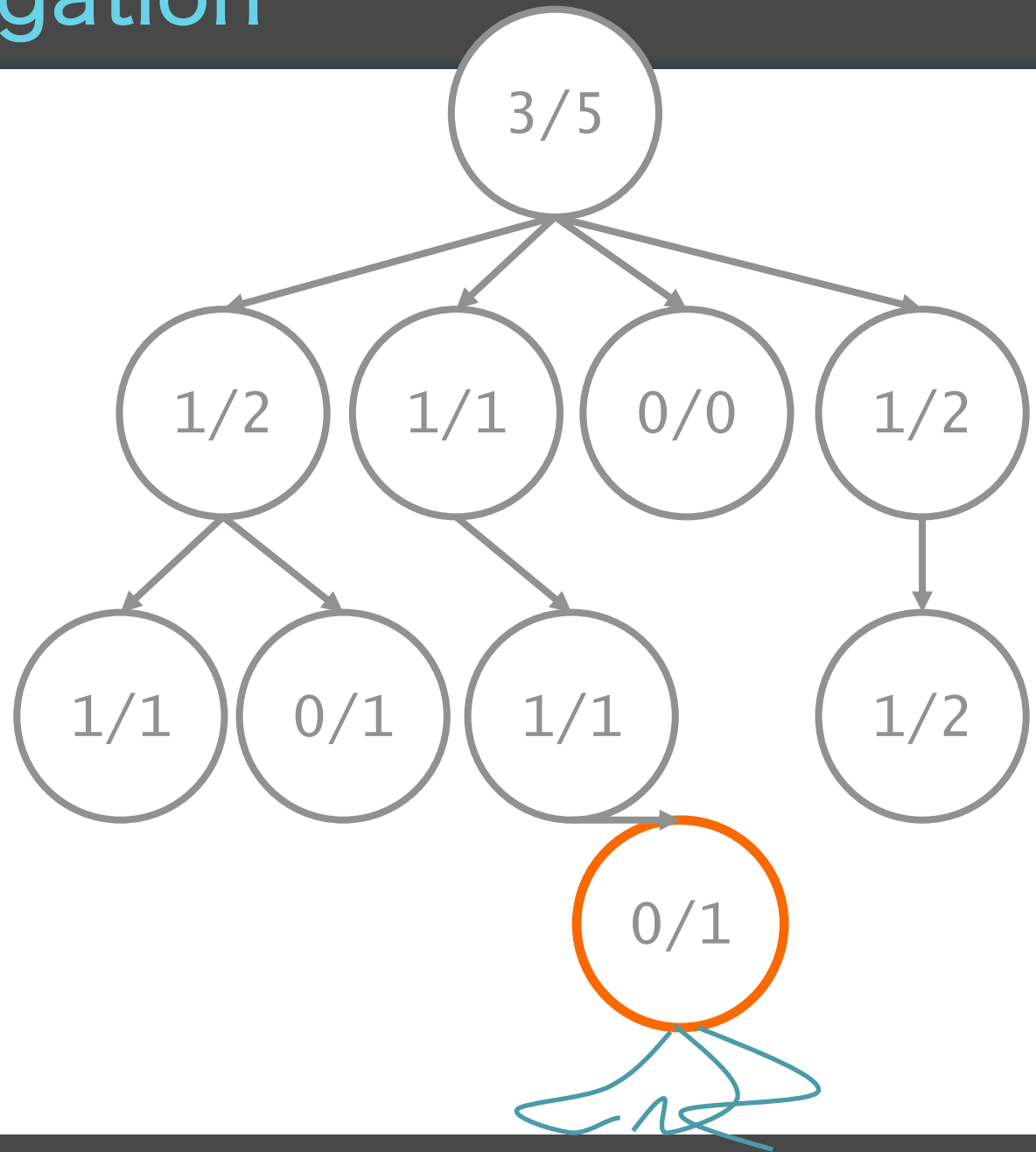
# Playout/Simulate

Stochastic simulation:  
random moves until  
payout.



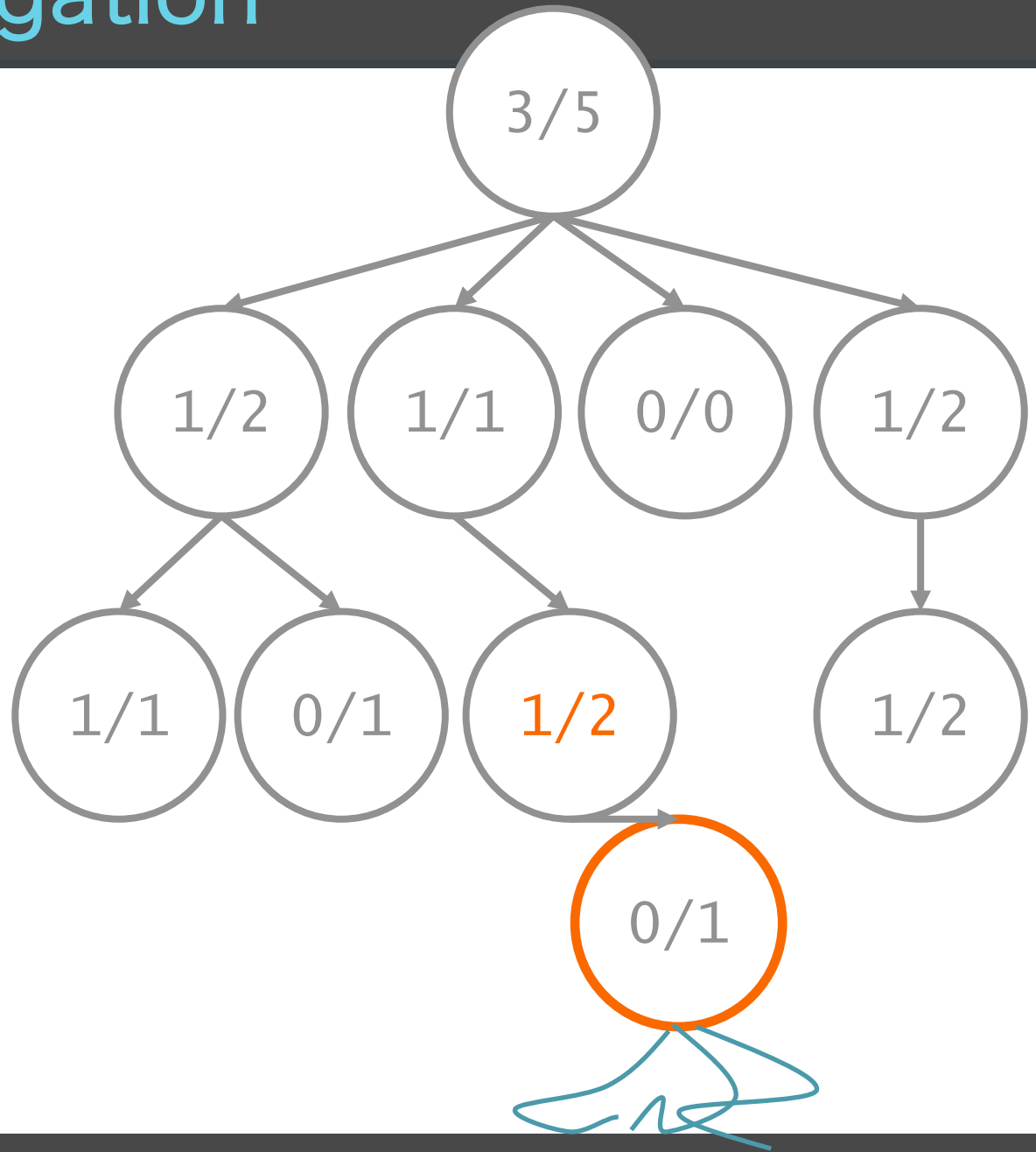
# Back Propagation

Update the tree  
based off  
simulation results.



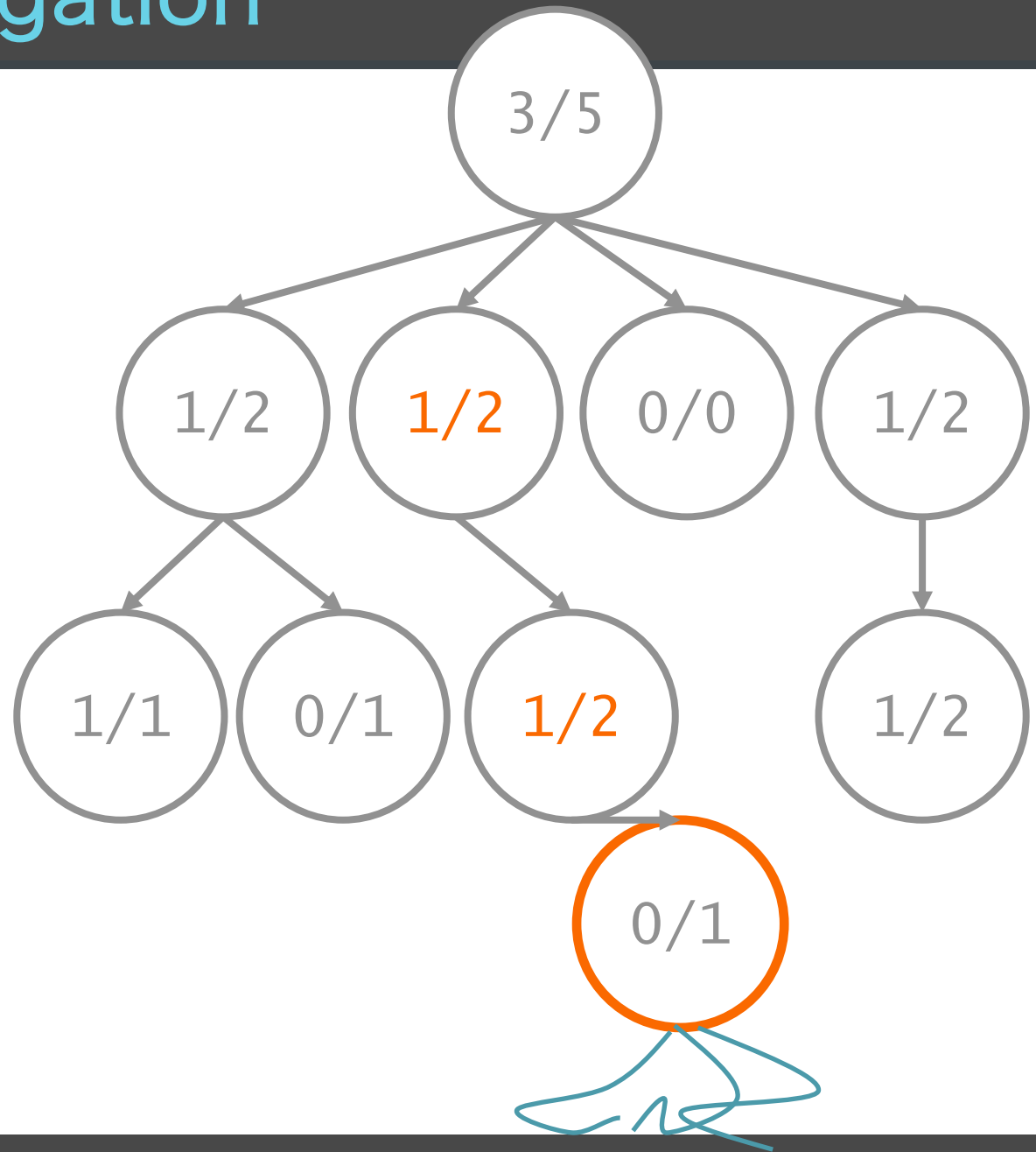
# Back Propagation

Update the tree  
based off  
simulation results.



# Back Propagation

Update the tree  
based off  
simulation results.



# Back Propagation

Update the tree  
based off  
simulation results.

