MCTS Extensions

2/15/17

The Monte Carlo Tree Search Algorithm

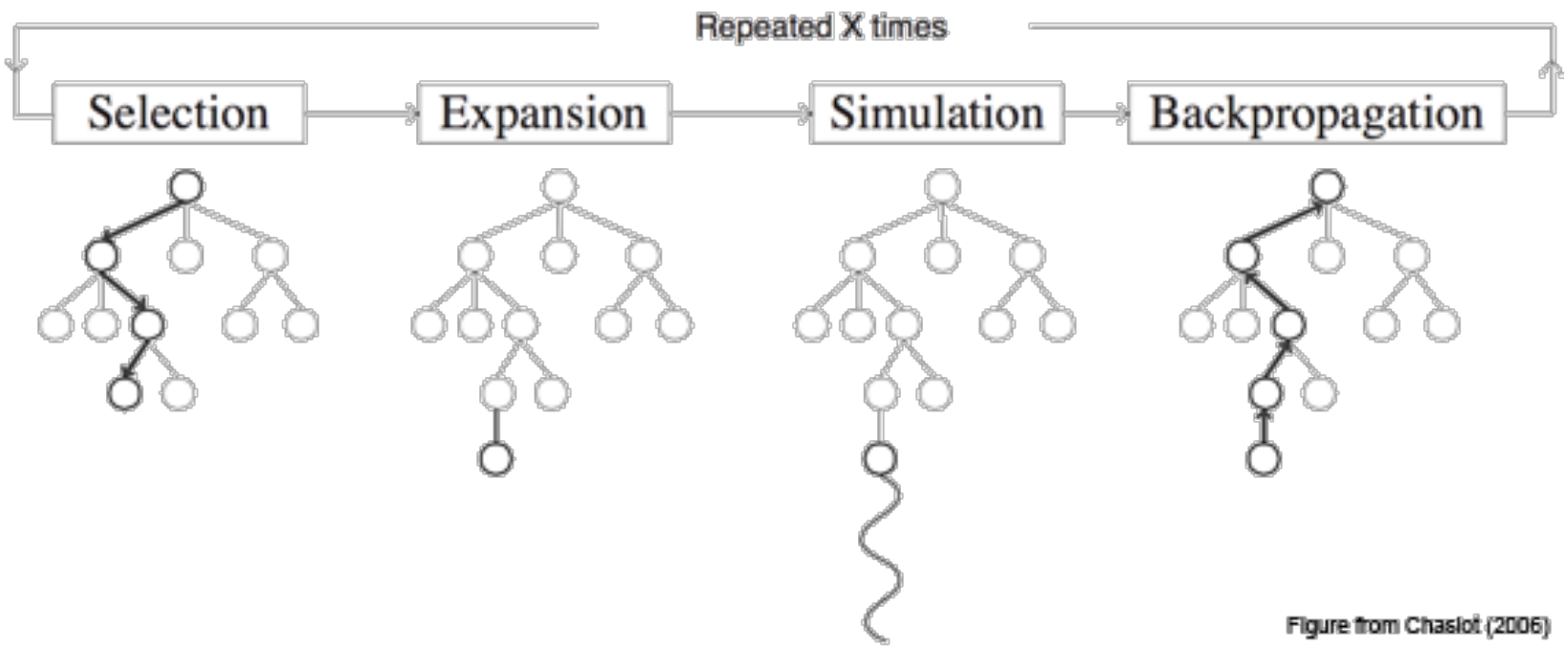
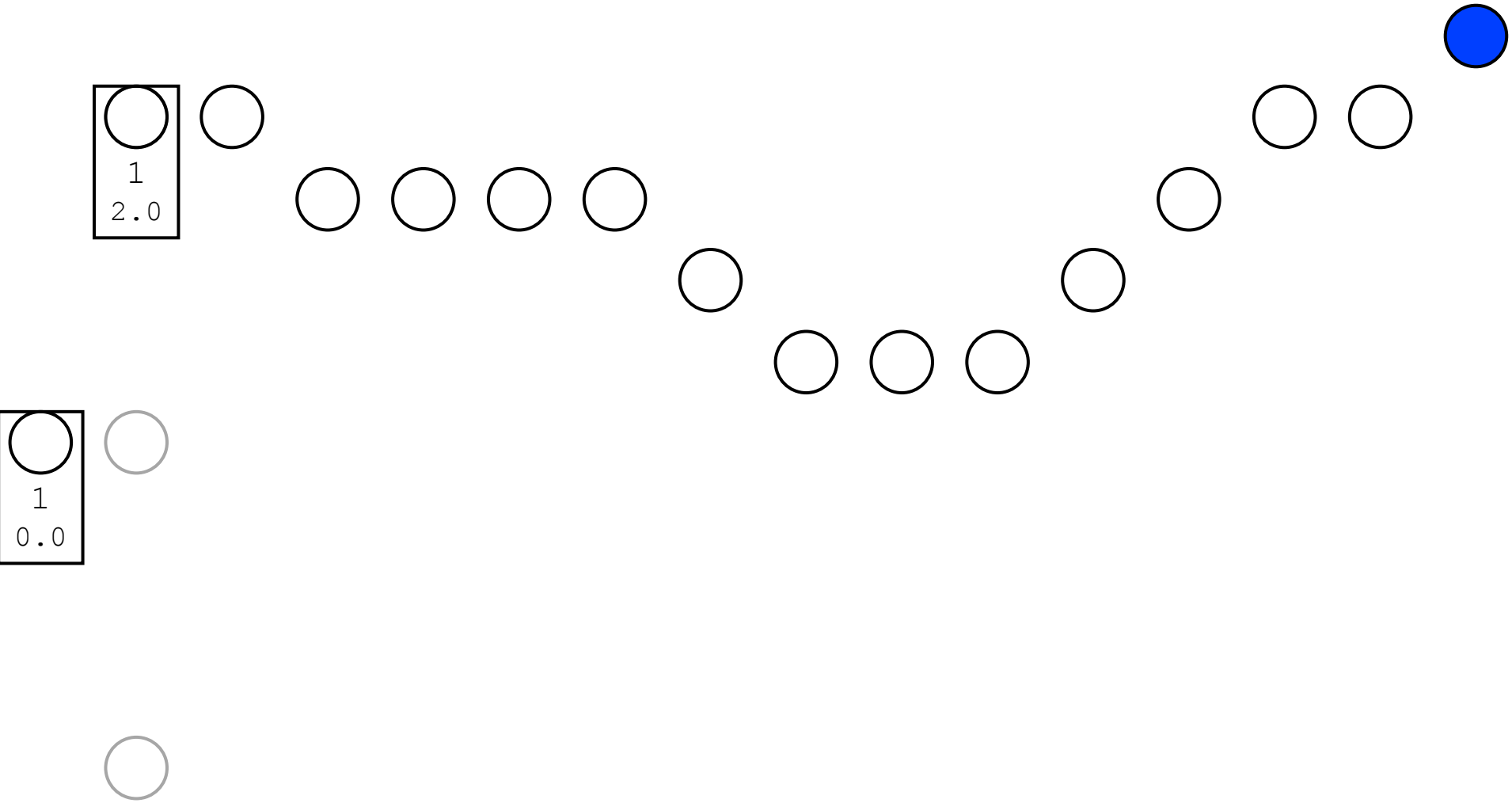


Figure from Chaslot (2006)

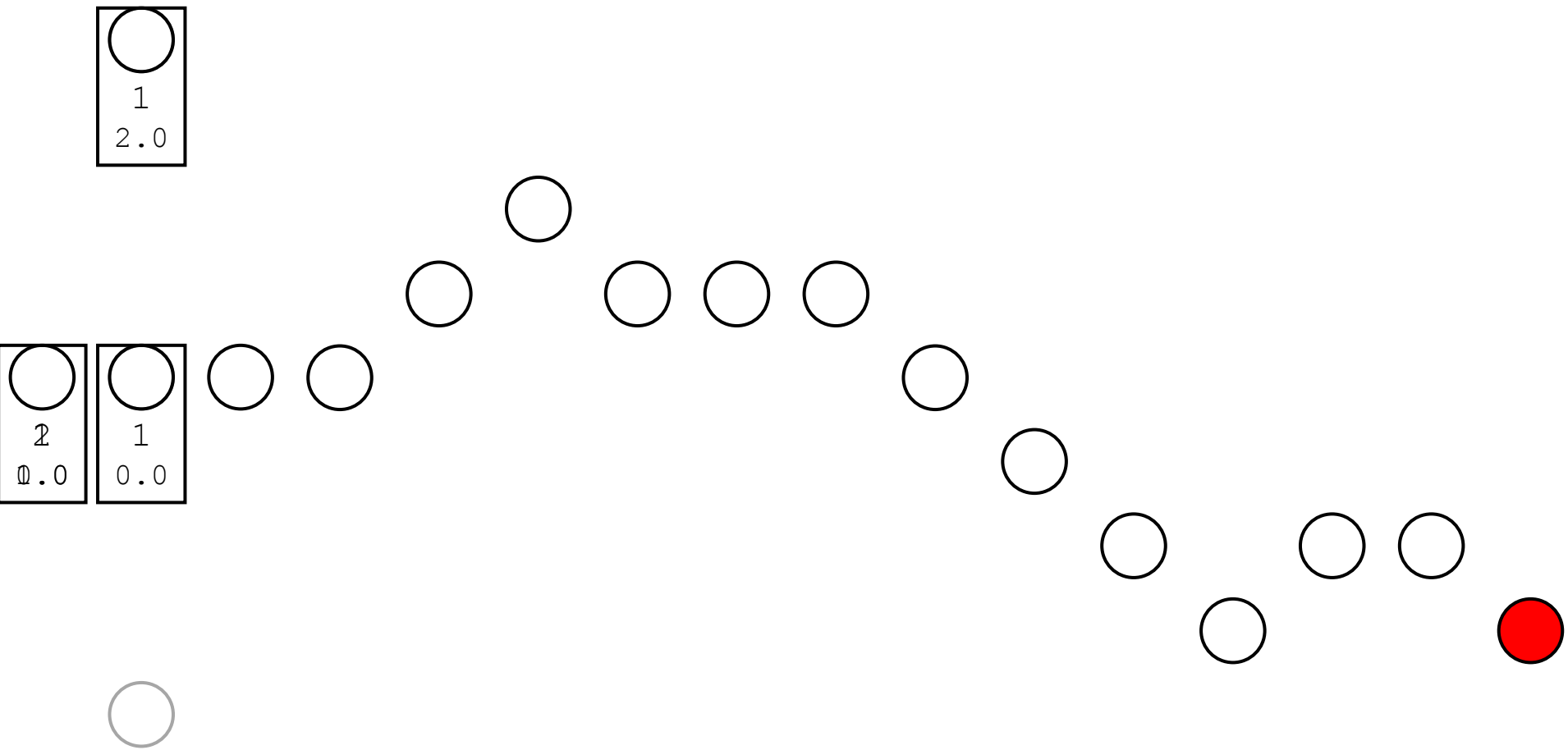
MCTS Pseudocode

```
for i = 1 : rollouts
   node = root
   init empty path
   # selection
   while all children expanded and node not terminal
      node = UCB_sample(node)
      add node to path
   # expansion
   if node not terminal
      node = expand(random unexpanded child of node)
      # simulation
      outcome = random_playout(node's state)
   # backpropagation
   for each node in the path
      update node’s value and visits
```

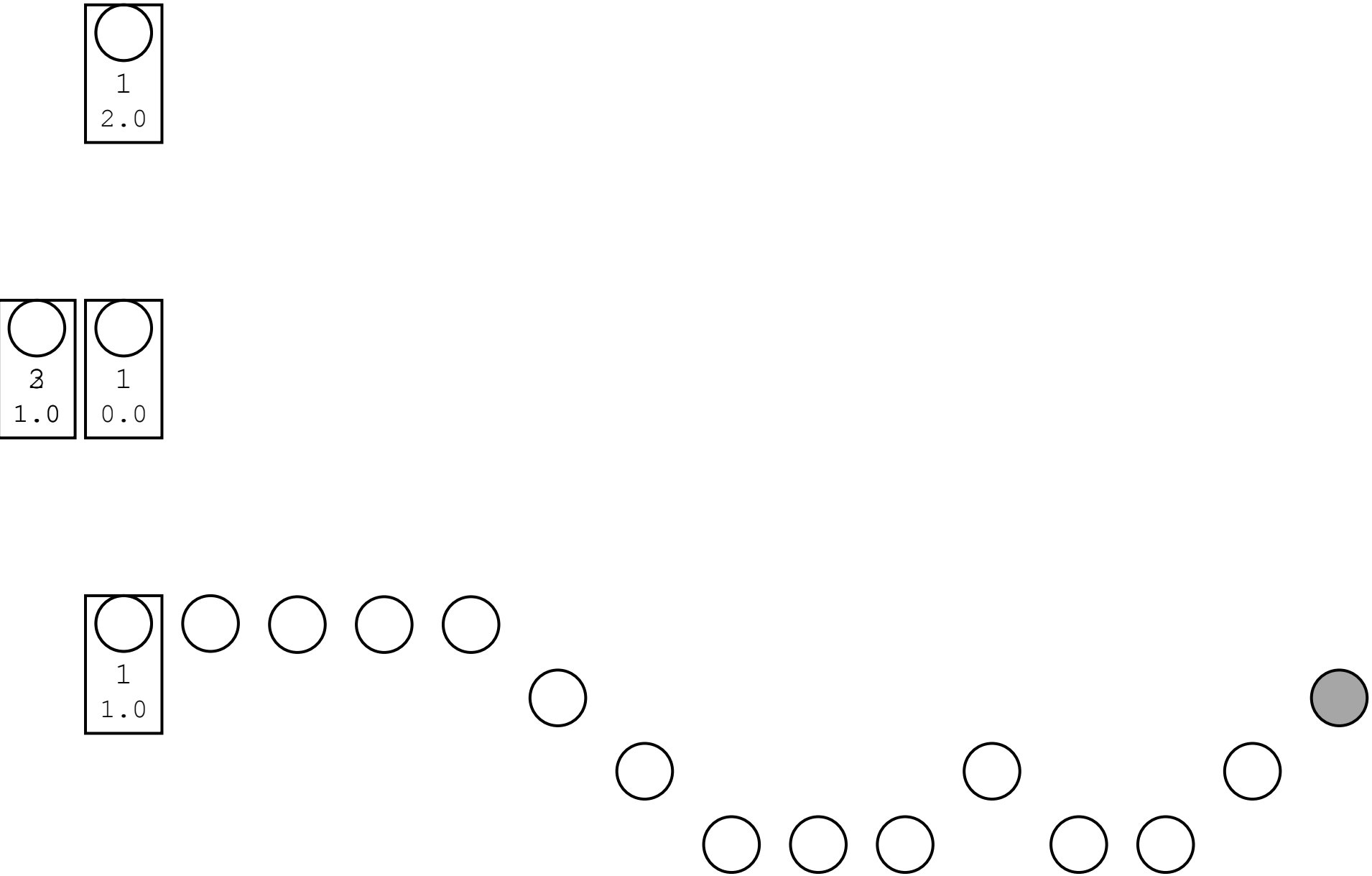
Selection Expansion Simulation Backpropagation



Selection Expansion Simulation Backpropagation



Selection Expansion Simulation Backpropagation



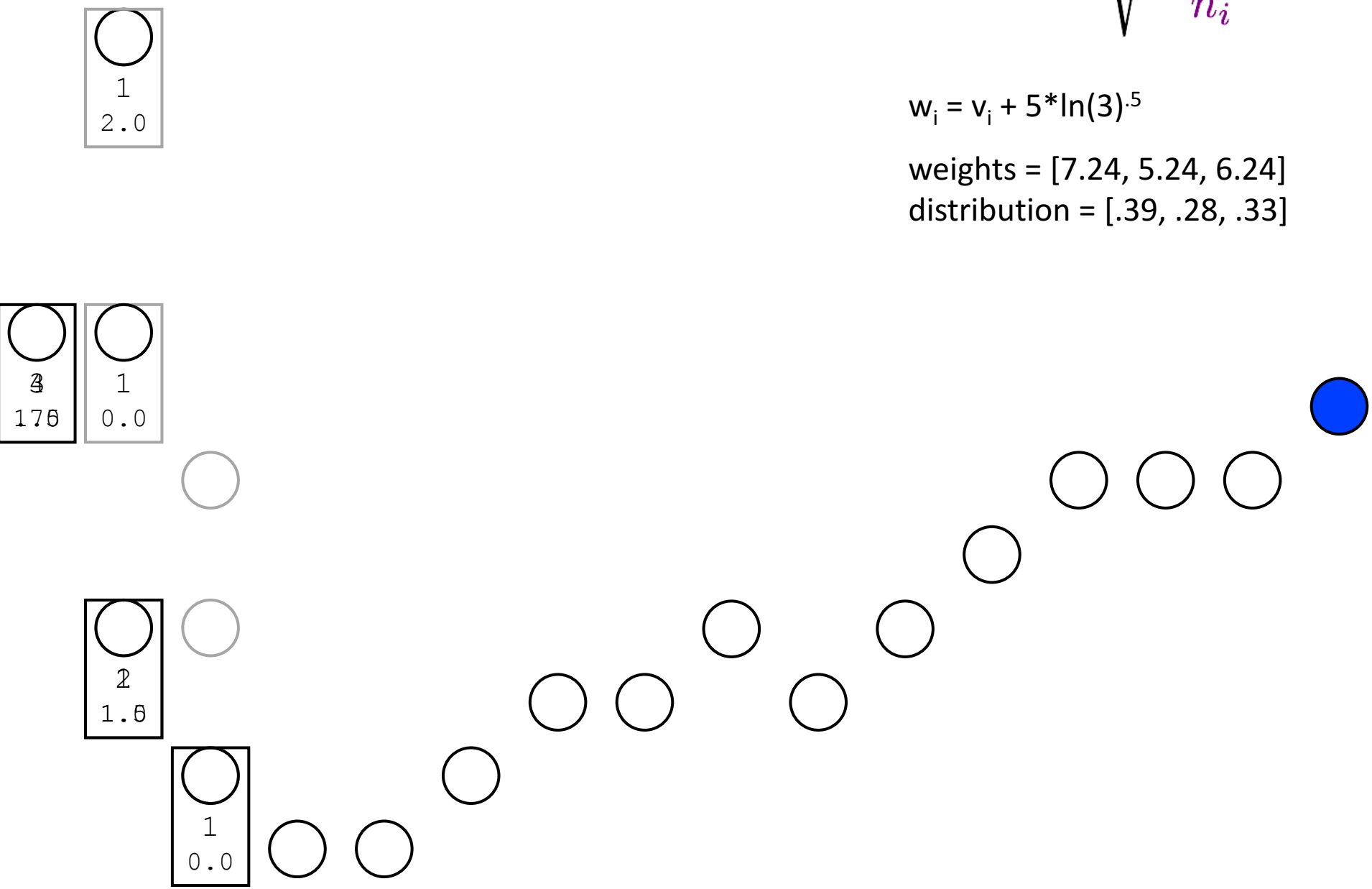
Selection Expansion Simulation Backpropagation

$$C = 5.0$$

$$v_i + C \times \sqrt{\frac{\ln(N)}{n_i}}$$

$w_i = v_i + 5*\ln(3)^{.5}$

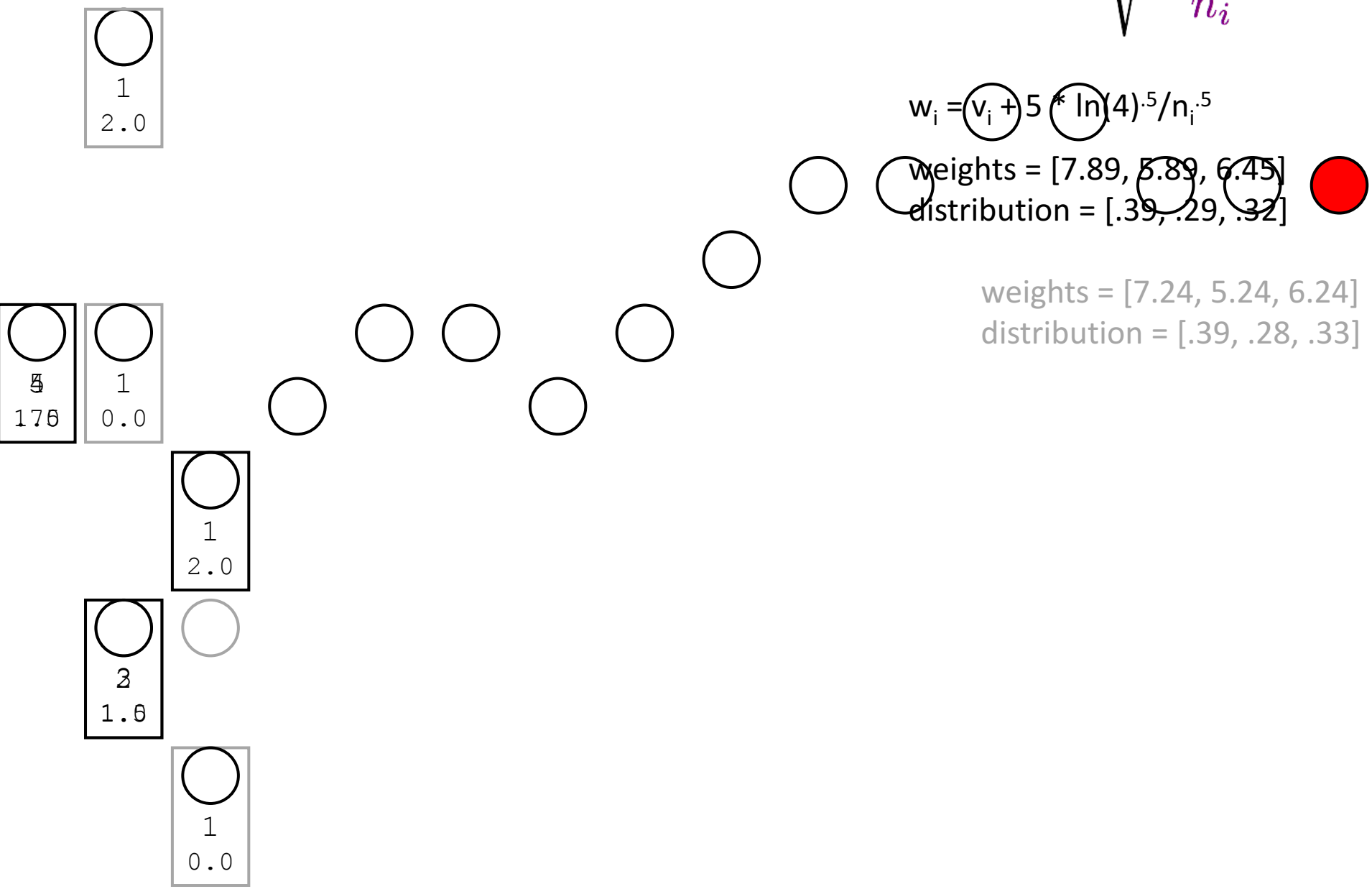
weights = [7.24, 5.24, 6.24]
distribution = [.39, .28, .33]



Selection Expansion Simulation Backpropagation

$$C = 5.0$$

$$v_i + C \times \sqrt{\frac{\ln(N)}{n_i}}$$



$w_i = v_i + 5 * \ln(4)^{.5}/n_i^{.5}$

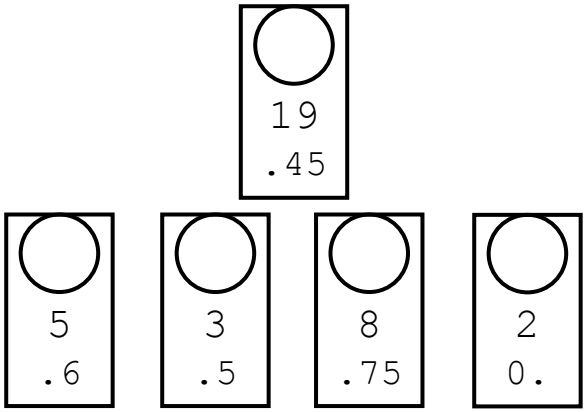
weights = [7.89, 5.89, 6.45]
distribution = [.39, .29, .32]

weights = [7.24, 5.24, 6.24]
distribution = [.39, .28, .33]

Exercise: construct the UCB distribution

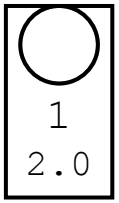
$$v_i + C \times \sqrt{\frac{\ln N}{n_i}}$$

$$C = 2$$



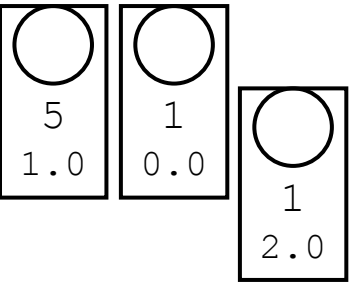

```
weights = [2.13, 2.48, 1.96, 2.43]

probs   = [0.24, 0.28, 0.22, 0.27]
```

How do we pick a move?



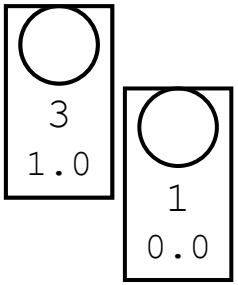
MCTS builds a tree, with visits and values for each node. How can we use this to pick a move?

- Pick the highest-value move.



- Pick the most-visited move.

- Can we do both?
  - Use some weighted combination.
  - Keep simulating until they agree.



Generalizing MCTS Beyond UCT

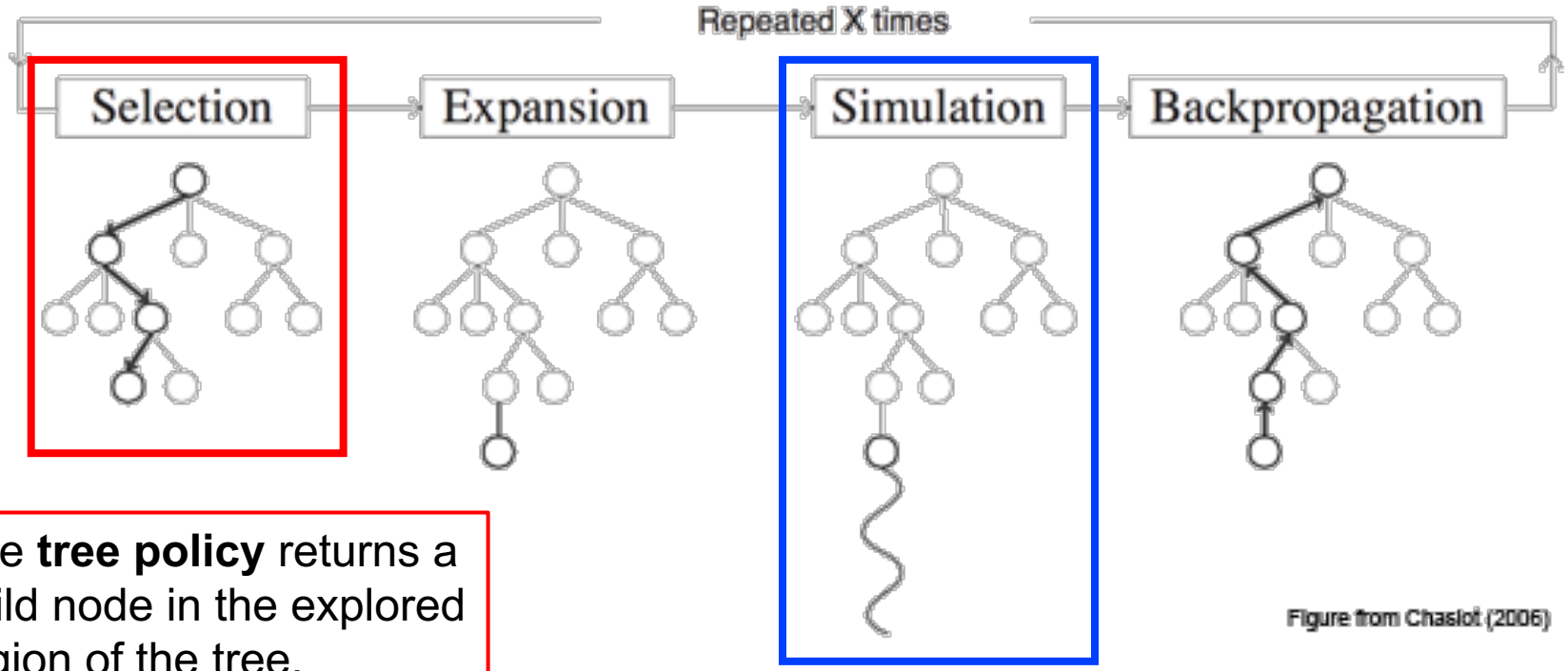


Figure from Chaslot (2006)

The **tree policy** returns a child node in the explored region of the tree.

UCT uses a tree policy that draws samples according to UCB.

The **default policy** returns a value estimate for a newly expanded node.

UCT uses a **default policy** that completes a uniform random playout.

Alternative tree policies

Requirement: The **tree policy** needs to trade off exploration and exploitation.

- Epsilon-greedy: pick a uniform random child with probability ε and the best child with probability (1-ε).
  - We'll see this again soon.
- Use UCB, but seed the tree within initial values.
  - From previous runs.
  - Using a heuristic.
- Other ideas?

Alternative default policies

Requirement: The **default policy** needs to run quickly and return a value estimate.

- Use the board evaluation heuristic from bounded minimax.
- Run multiple random rollouts for each expanded node.
- Other ideas?

Exercise: extend MCTS to these games

How can MCTS handle non-zero-sum games?

How can MCTS handle games with randomness?

Non-Zero-Sum Games

<u>Key idea</u>: store a value tuple with the average utility for each player.

- Each node now stores visits, children, and one value for each player.
- The agent who's making a decision will compute UCB weights using only *their component* of the value tuple.

Randomness in the Environment

This is what Monte Carlo simulations were made for!

- Whenever we hit a move-by-nature in the game tree, sample from nature's move distribution.
- We still need to track value and visits for the nature node, so that the parent can make its choices.