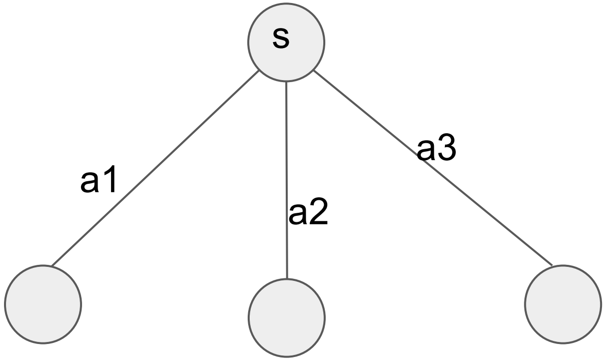
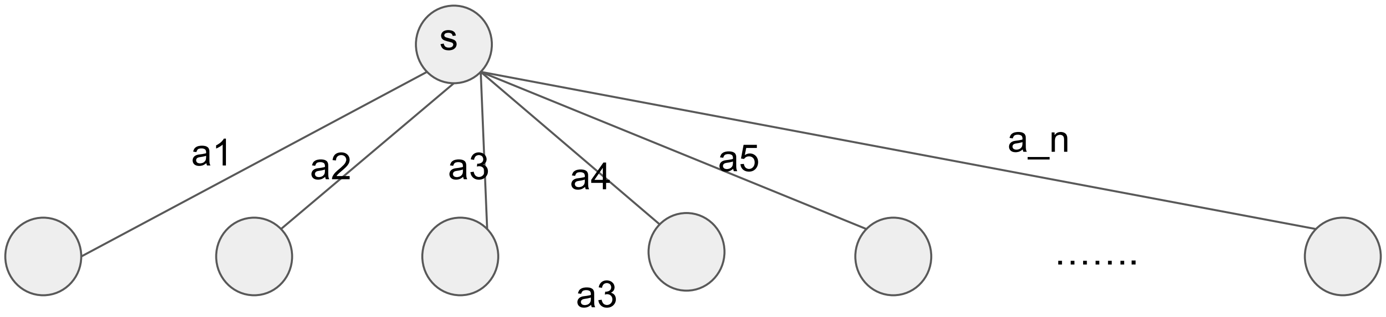
AlphaGo Zero

Monte Carlo Tree Search



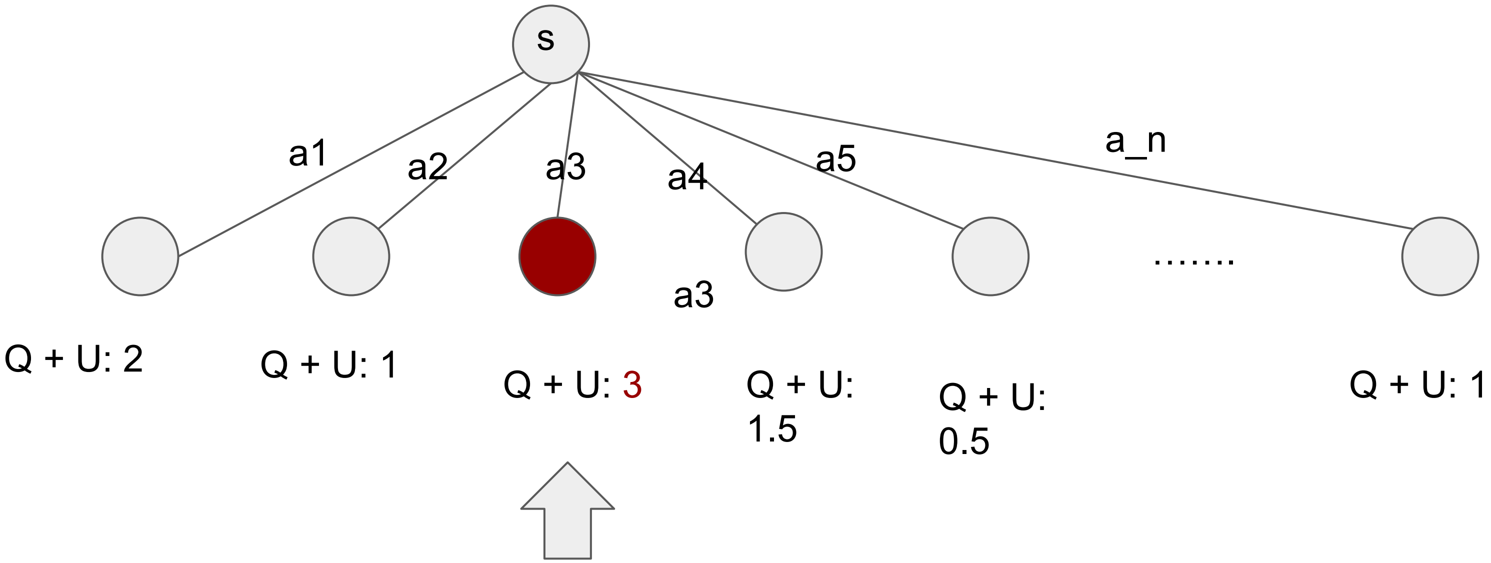
Each node contains:
P(s,a): a prior probability to take action a at state s
N(s,a): number of visits from state s to take action a
Q(s,a): an action value {from state s to take action a} Policy Network will calculate this.
U(s,a): = P(s,a)/(1+B(s,a))

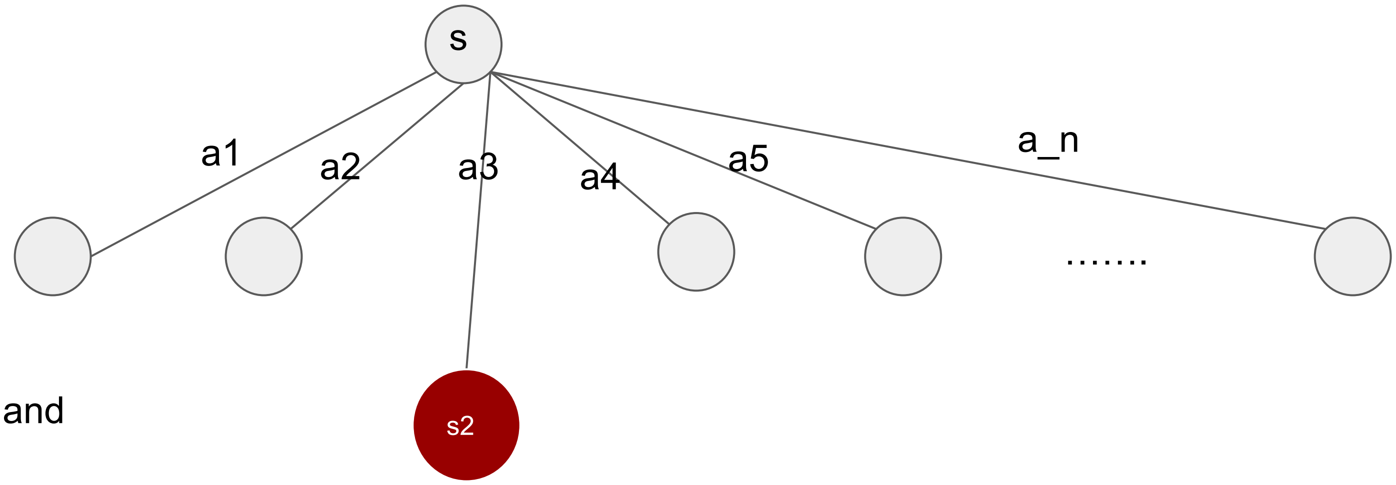


Initialize the tree

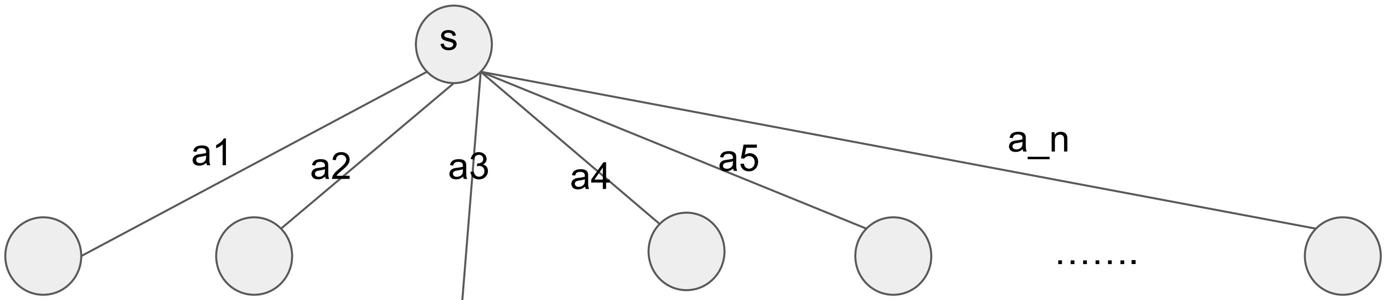
Select:

Max Q(s,a) + s(s,a)

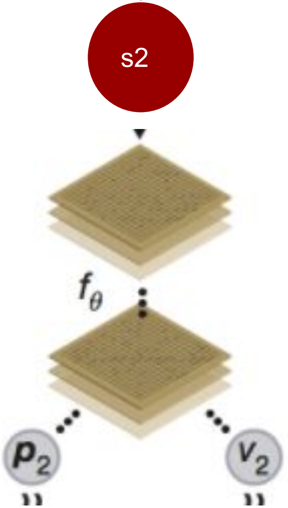

s
a1
a2
a3
a4
a5
a_n
a3
…….
Q + U: 2
Q + U: 1
Q + U: 3
Q + U: 1.5
Q + U: 0.5
Q + U: 1



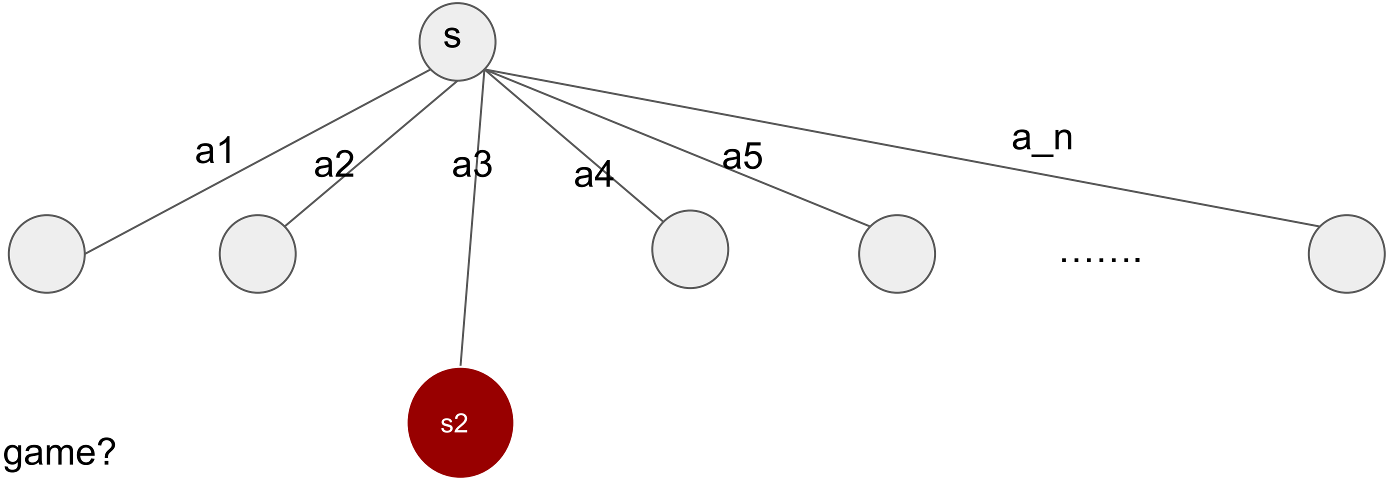
Move to this node, and
check if it is a leaf



Put s2 into policy network(for example CNN, ResNet):
f(s2) -> p2:action_probs,
v2:leaf_value

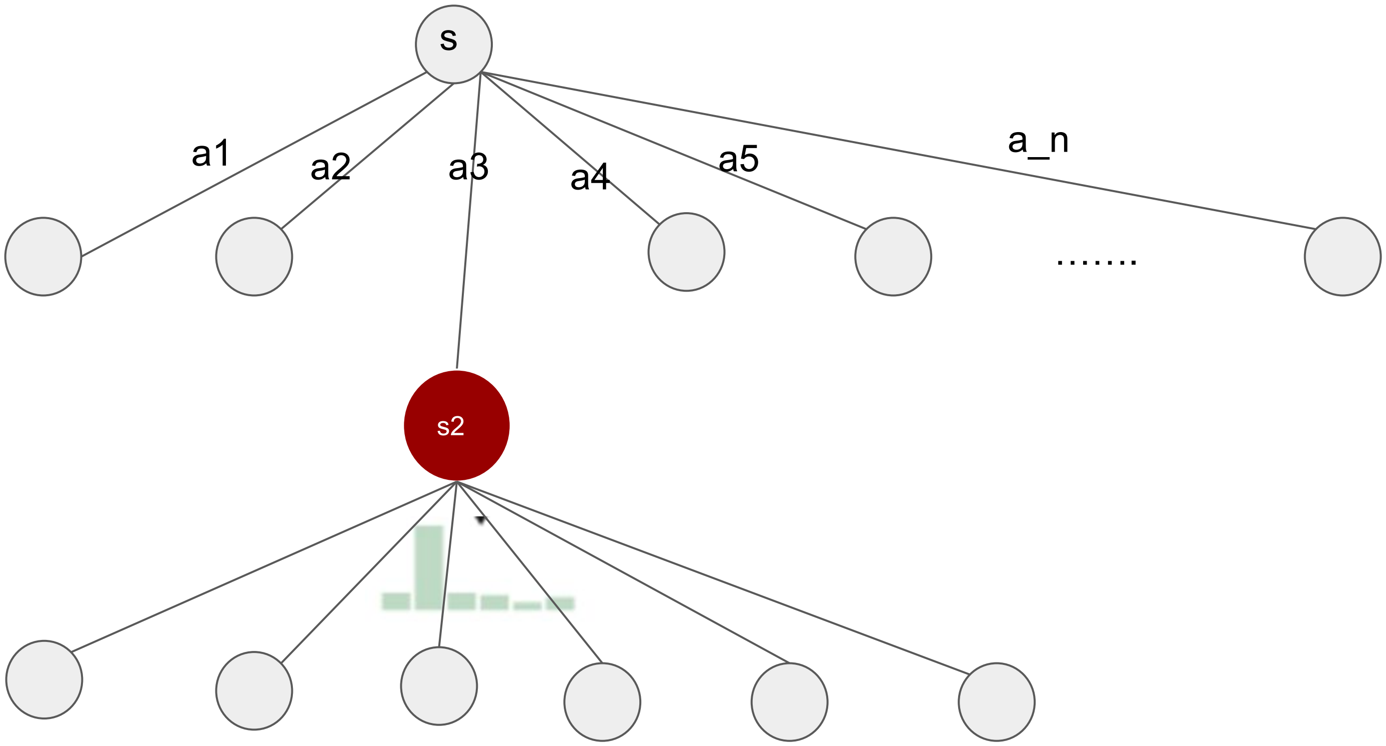


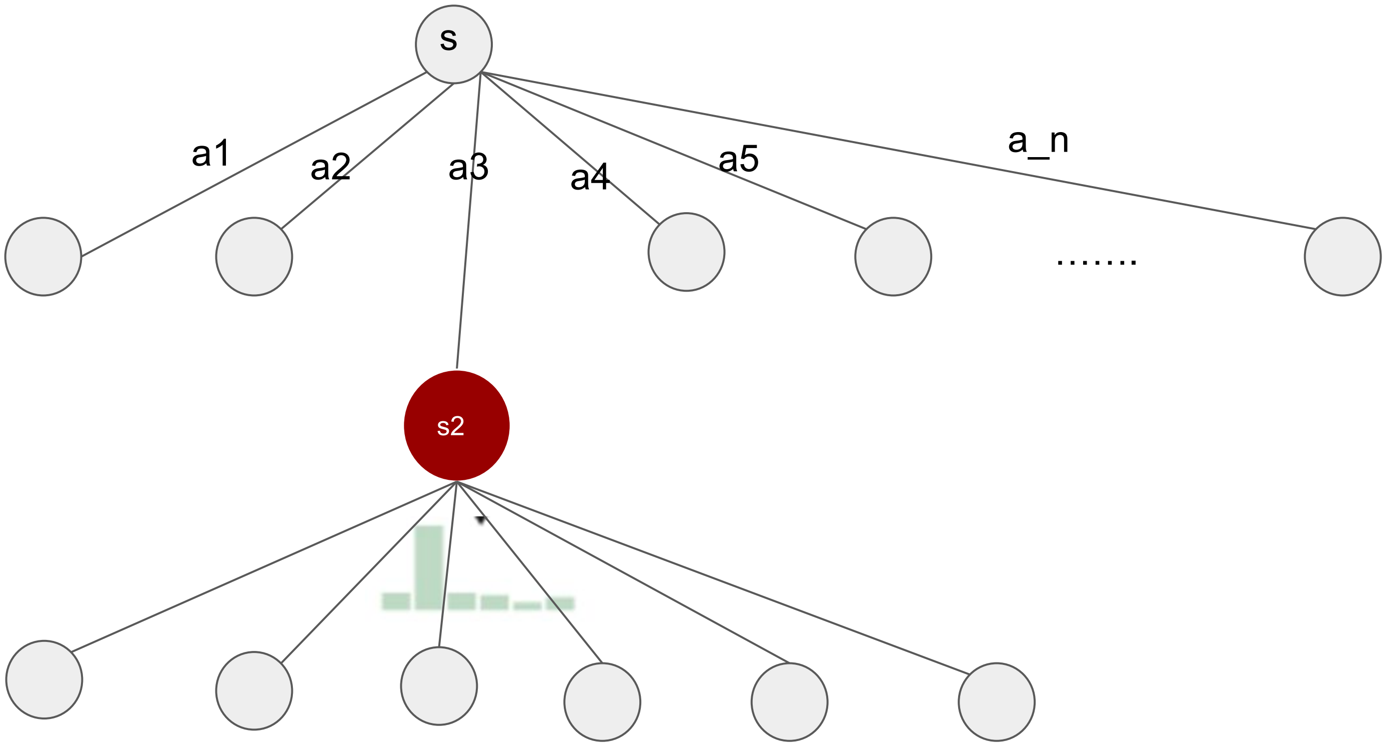
p2 is like a list of action and their prior probability.



Check:
is s2 the end of the game?
No

Expand s2:

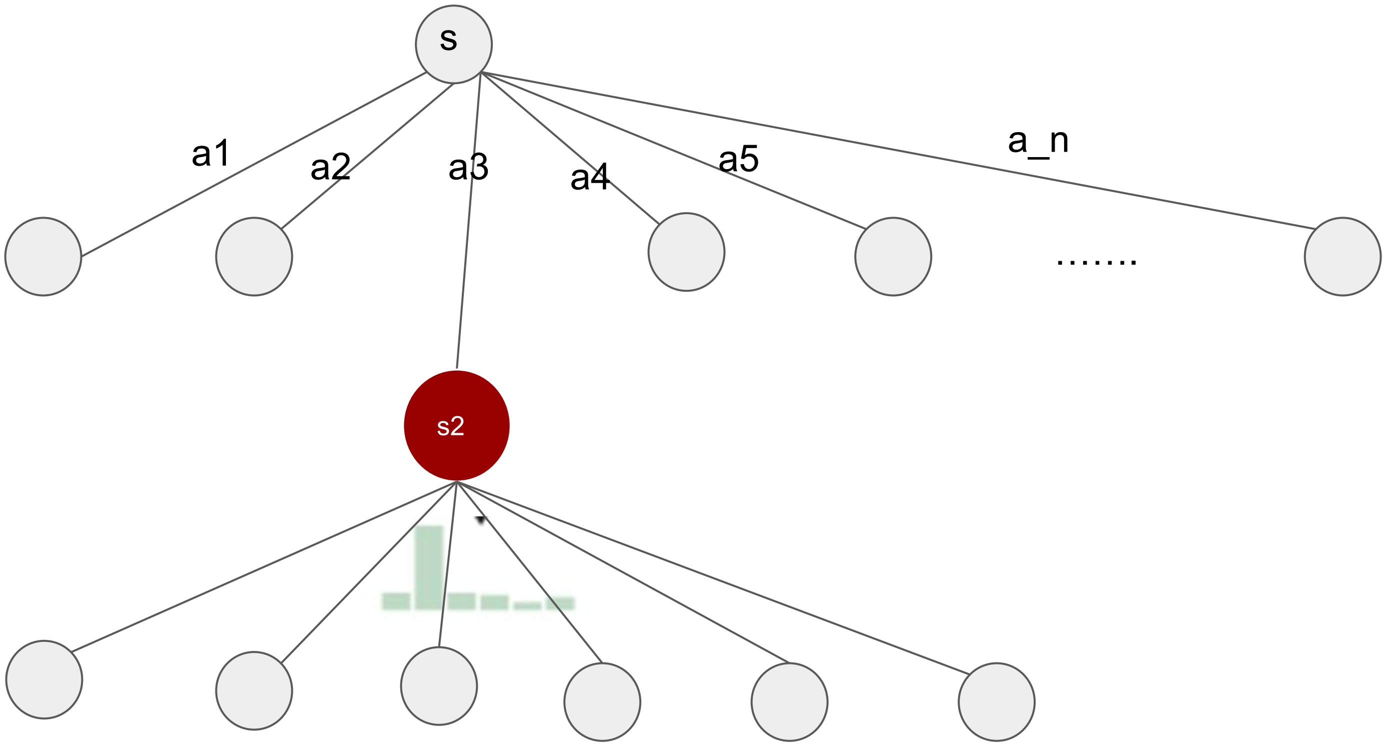




Update:

N(s,a) += 1
Q(s,a) +=
leaf_value -
Q(s,a)/N(s,a)

Update starting
from the root



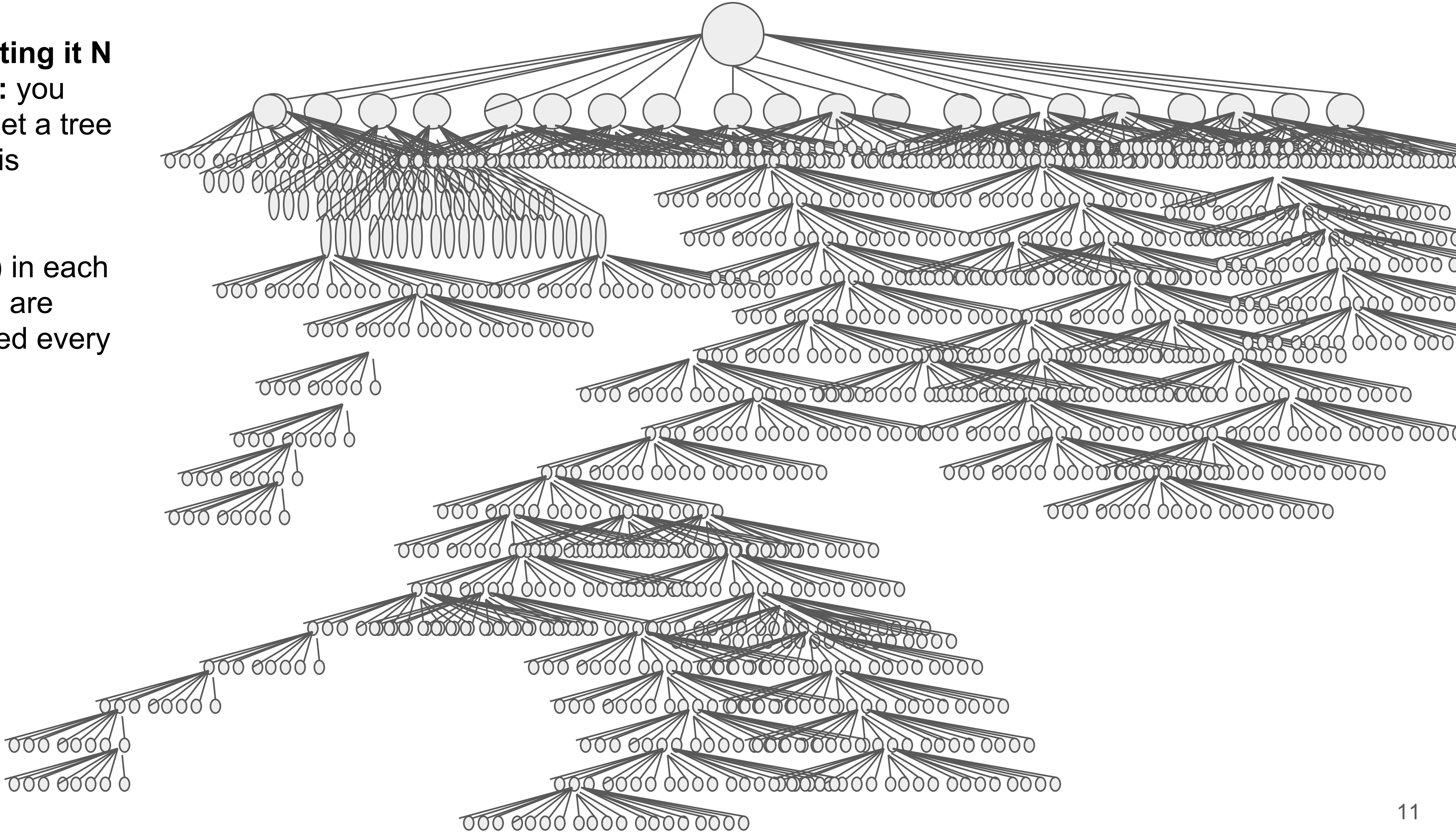
Repeat the above actions:
N times

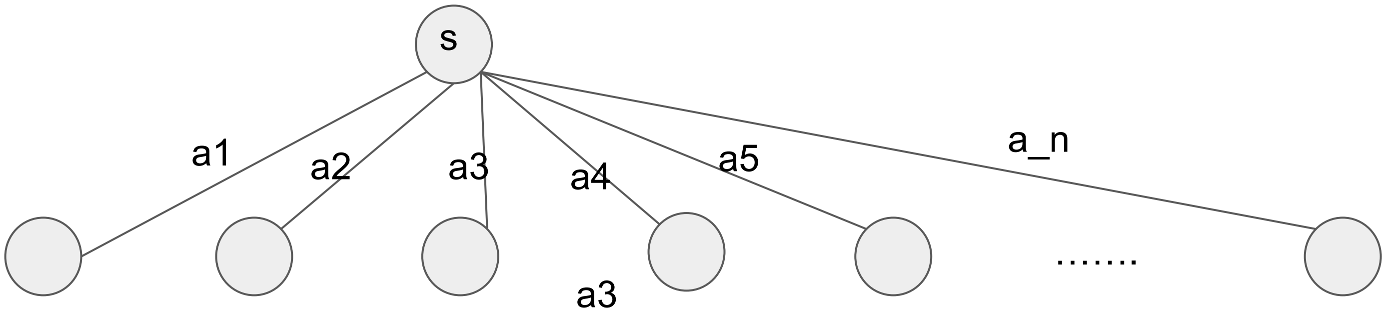
N depends on your computational resource

Every time, you start from the initial root.

After repeating it N times: you may get a tree like this

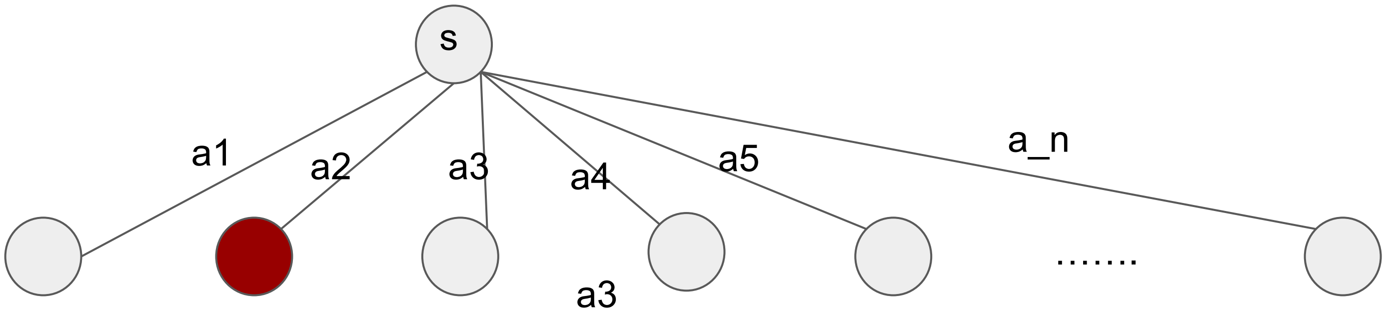
N(s,a) in each nodes are updated every time





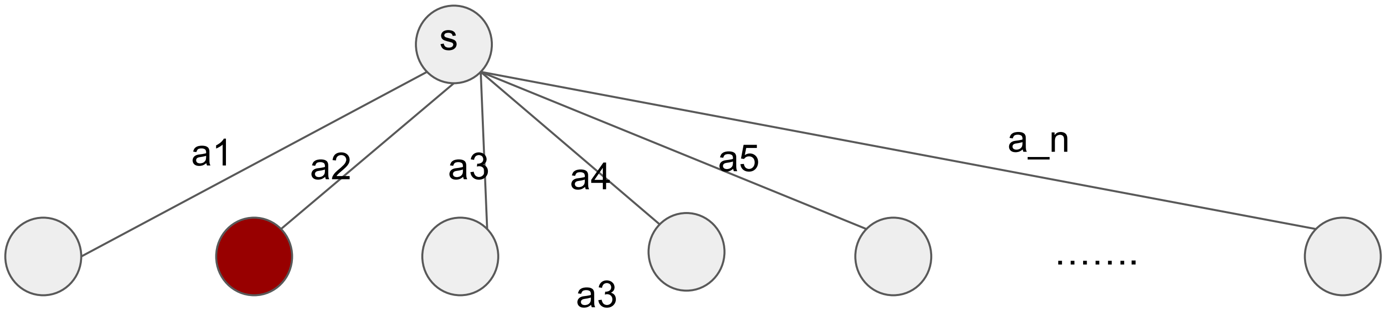
Calculate
action probabilities(pi2)

act_probs = softmax(N(s,a))



Randomly choose a move based on the **probability distribution(pi 2)** you get from the **last page.** This pi2 is **search probabilities.**

Move to the next state.



Now starting from the red node,
repeating step 6 -13.

Then move to the next node.

Do this until the game is terminated.

Now the tree search is completed.

The game is terminated.

We need to train the policy network.

In each step in the game, we have search probability pi, and probability produced by the policy network.

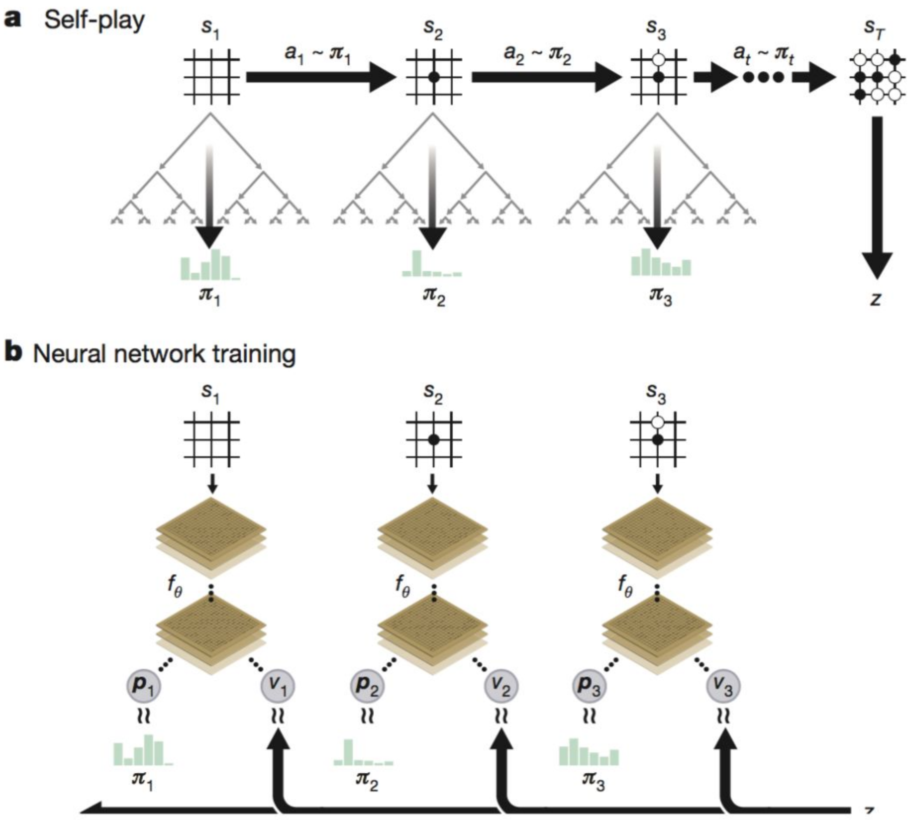
We have v produced by the network.

We have z which is the real win or loss result at each node.

We want to make
pi and p similar
make v and z similiar

So the loss function is:
max similarity of pi and p
minimize error between v and z

(check note)



Repeat the above steps M iteration