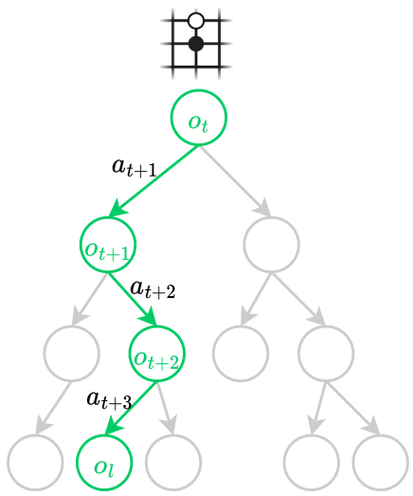
AlphaZero: A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play

A. MCTS in AlphaZero

a. Selection

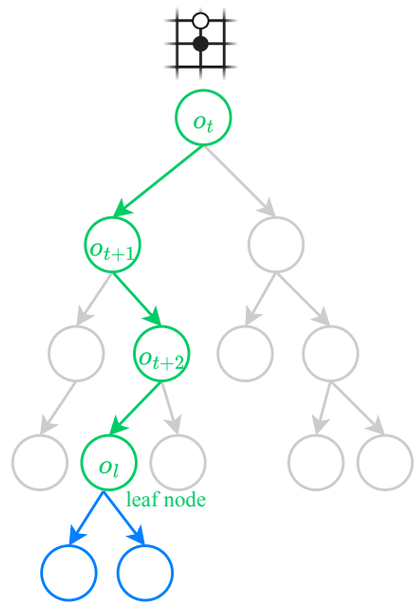


In each node, select action a_k according to the **UCB** score:

$a_k = \arg\max_a \left[Q(o,a) + cp(o,a) \cdot \frac{\sqrt{\sum_b N(o,b)}}{1+N(o,a)} \right]$

The selection function is applied recursively until a **leaf** node is reached.

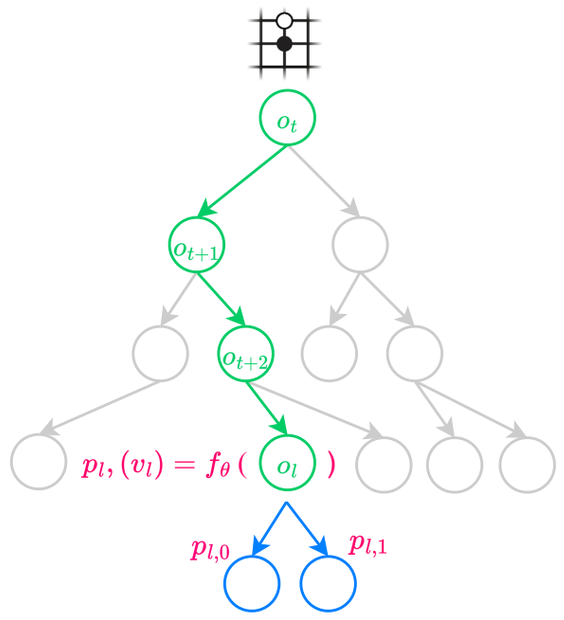
b. Expansion



The **leaf node** o_l is added to the search tree. Each edge (o_l, a) from the newly expanded node is initialized to:

$\{N(o_l,a) = 0, Q(o_l,a) = 0, P(o_l, a = P_l)\}$

c. Evaluation

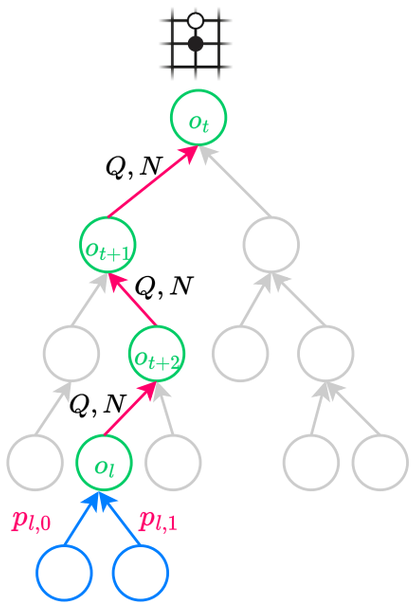


The leaf node is evaluated by the neural network :

$p_l, (v_l) = f_\theta(o_l)$;

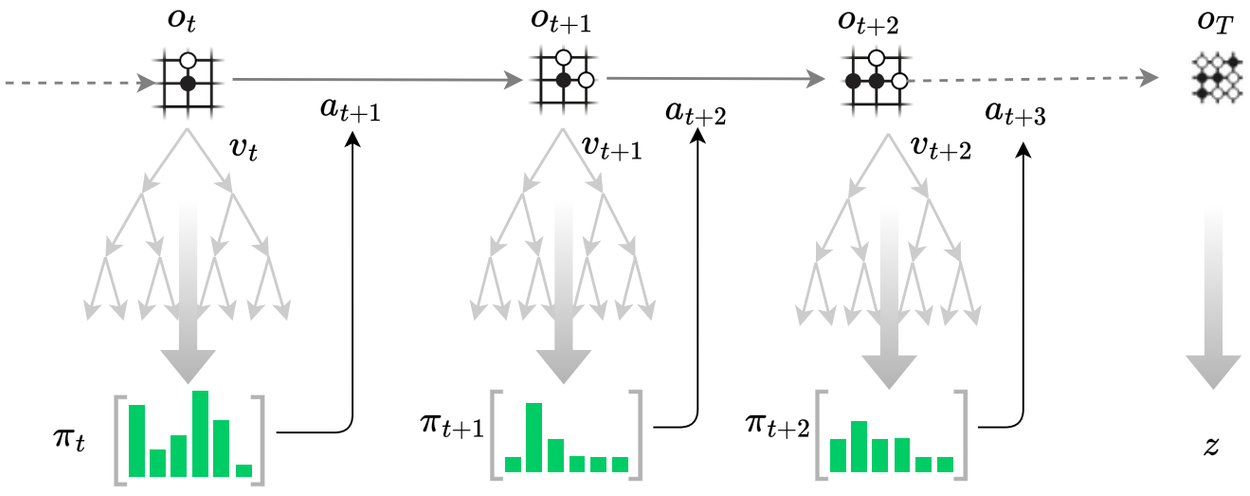
the vector of policy priors P_l are stored in the outgoing edges from o_l.

d. Backpropagation



Action values Q are updated to track the mean value of all evaluations $\{v\}$ in the subtree below that action. N are updated to track how many times the action has been on the whole MCTS simulations.

B. Acting （Self-play）



C. Training

$$l_t(\theta) = l^v\left(z_t, v_t\right) + l^p\left(\pi_t, \mathbf{p}_t\right) + c\|\theta\|^2$$

where, z_t is the game reward from the **perspective of the current player** and π_{t+k} is the MCTS searched policy at timestep t .

v_t and $\mathbf{p}_t$ is the predicted value and policy from the neural network f_θ.

l^v is MSE loss, l^p is cross-entropy loss.