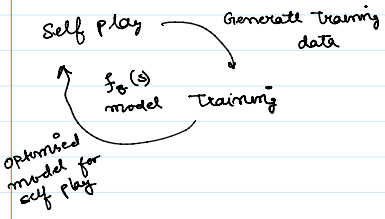
Alpha-zero

04 March 2023 14:02

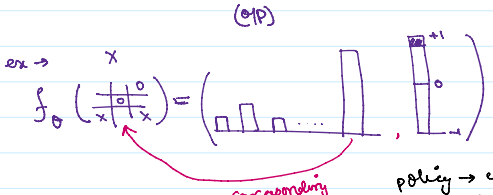


MODEL :-

$f_\theta(s) = (p, v)$

- state (i/p)
- policy (o/p)
- value

ex →



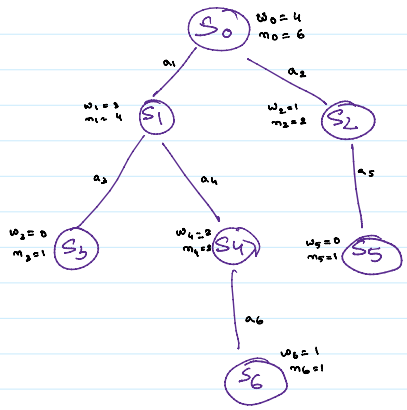
corresponding atoms place

policy → choose which state is the best

value → overall state is good or not.

{ $f(\text{best}) \in [-1, 1)$ }

MONTE CARLO TREE SEARCH



w → no. of times we were able to win

n → total no. of times that state has been visited

{ out of n times, we won w times }

for each root node, check the child node. The child node with higher win rate (w/n) is more potential

ex → for ex → $a_1 : \frac{3}{4}$

$a_2 : \frac{1}{2}$

$a_2 >> a_1$

1) selection | walk down till leaf node

2) Expansion | create new node

3) simulation | play randomly

4) Backpropagation

which direction to walk down?

they follow UCB formula

$$\frac{w_i}{n_i} + c\sqrt{\frac{\ln(N_i)}{n_i}} \quad \Big| \; c = 2$$

- $\frac{w_i}{n_i}$ → highest winning ratio
- $\sqrt{\frac{\ln(N_i)}{n_i}}$ → child that has been visited the least no. of times

ex : $a_1 = \frac{3}{4} + 2\sqrt{\frac{\ln(6)}{4}}$

≈ 2.0089

$a_2 = \frac{1}{2} + 2\sqrt{\frac{\ln(6)}{2}}$

$= 2.0393$

So, $a_2 >> a_1$

↓

walk down w/a a_2

How monte carlo TS adapt to α-0 algorithm →

1) include policy and value

↳ no random future

i) Include policy and value → no random future iteration

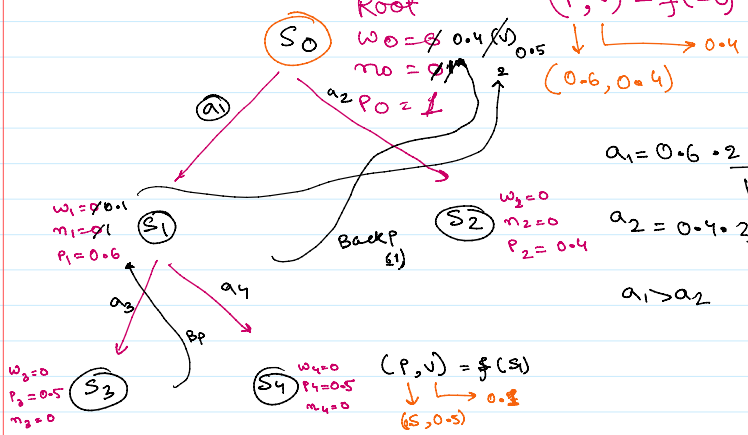
$$UBC = \left(\frac{w_i}{n_i}\right) + p_i \, c \, \frac{\sqrt{N_i}}{1+n_i}$$

iff $n_i > 0$

p_i → children with higher policy gets selected

(*) Simulation step skipped

ii) Create all possible nodes rather than a single node in expansion phase.



$a_1 = 0.6 \cdot \frac{2\sqrt{1}}{1+0} = 1.2$

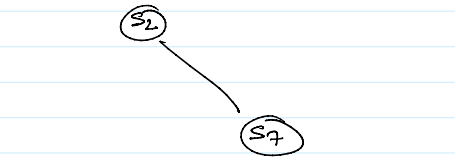
$a_2 = 0.4 \cdot \frac{2\sqrt{1}}{1+0} = 0.8$

$a_1 > a_2$

s_0, $a_2 >> a_1$

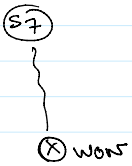
walk down with a_2

Expansion:



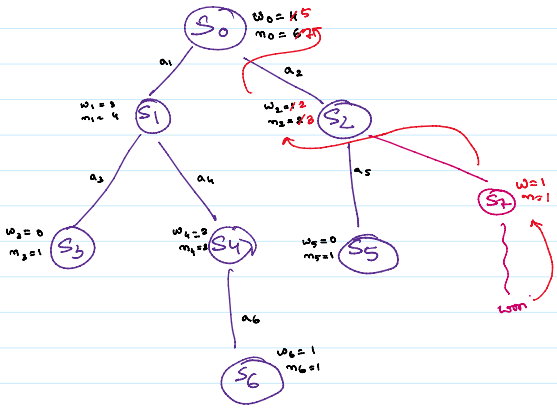
Create new node
$w_7 = 0$
$n_7 = 0$

Simulation → play until game over.
random play



Back propagation

Since we won the game, we won 1 time out of 1 time we came in s_7



Self play

X: → Perform MCTS → π {mcts-distribution}

{0.13, 0.09, 0.12, 0.09, 0.08, 0.10, 0.16, 0.10, 0.13} $z = 1$ → Reward / final result of player

O:

{0.14, 0.06, 0.11, 0.13, 0.23, 0.07, 0.0, 0.12, 0.14} $z = -1$

....

Draw / one player win

Training

1) s, π, z = Sample | Take sample from training data

2) $f_\theta(s) = (p, v)$ | Get o/p from model

3) minimize the difference between the policy p and MCTS distribution π

minimize value and final reward difference.

$L = (z - v)^2 - \pi^T \log p + c\|\theta\|^2$ Minimize loss by backpropagation