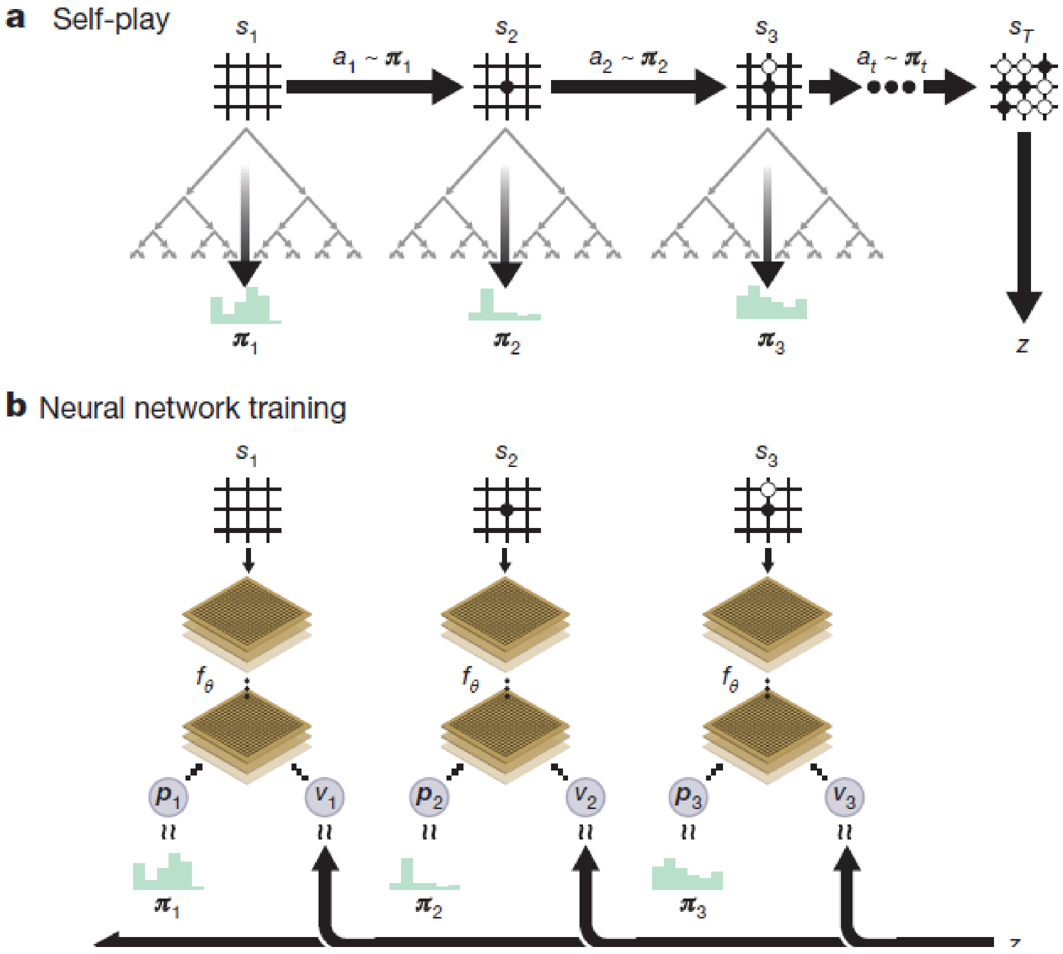
AlphaZero in Gomoku

Hongming Zhang

Bo Xu’s Workgroup@CASIA

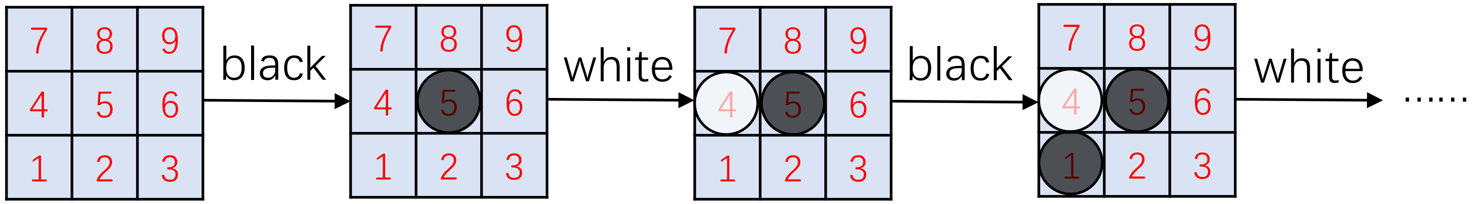
2018.11.30

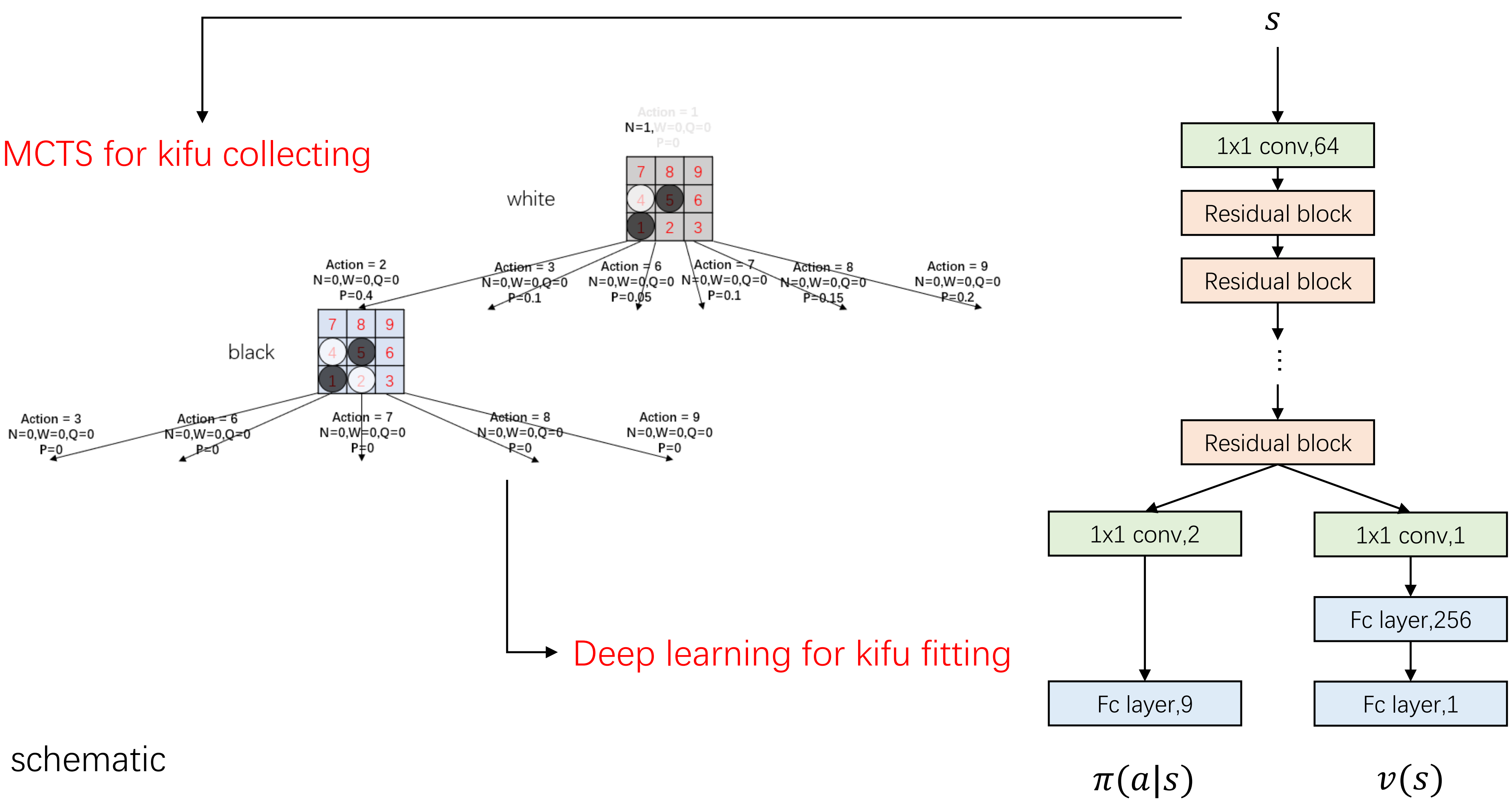
- Self-play reinforcement learning in AlphaGo Zero



- This powerpoint explains AlphaGo Zero algorithm with Gomoku.

- To simplify,
  we take the 3x3 Gomoku for instance.
- The board here is only 3x3, the one that get 3 in a row will win the game.
- Numbers in the board denote actions, black and white denote two players.





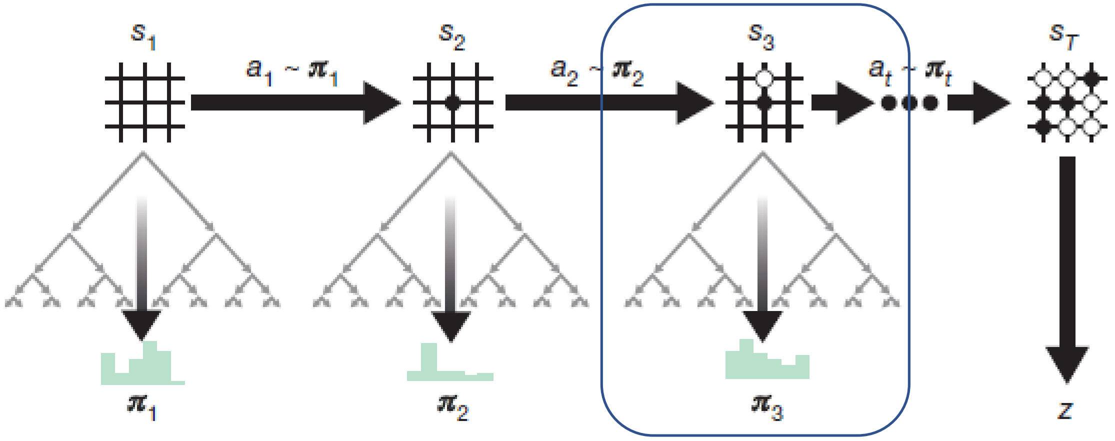
- Algorithm schematic

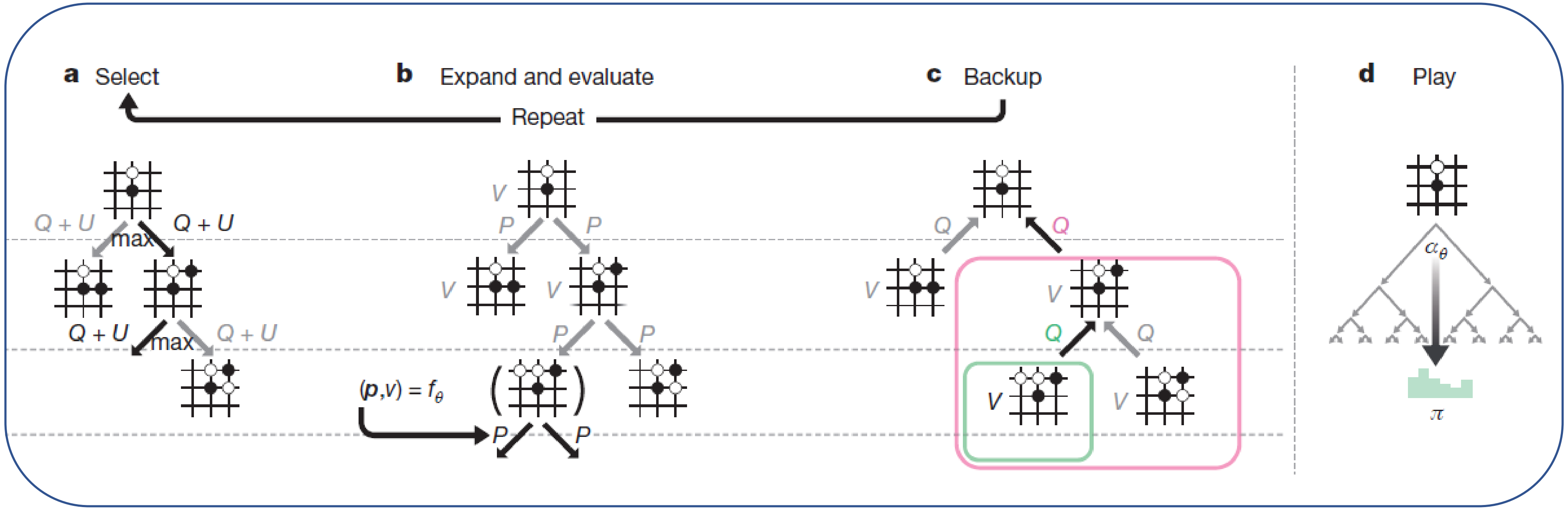
Self-play

➢ MCTS in AlphaGo Zero

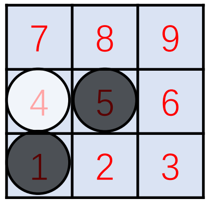
- Select
- Expand and evaluate
- Backup

- Play



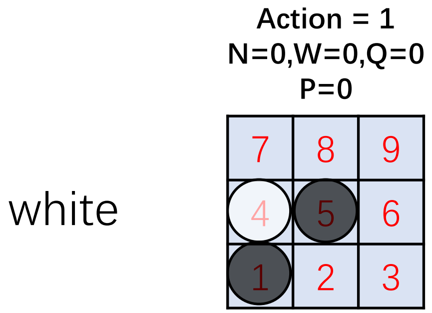


- Monte Carlo Tree Search in Gomoku



Real board state

- We assume the game start from this state.
- So the root node in the tree is from here!

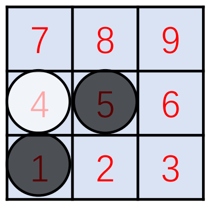


Tree from this node.

The information here is

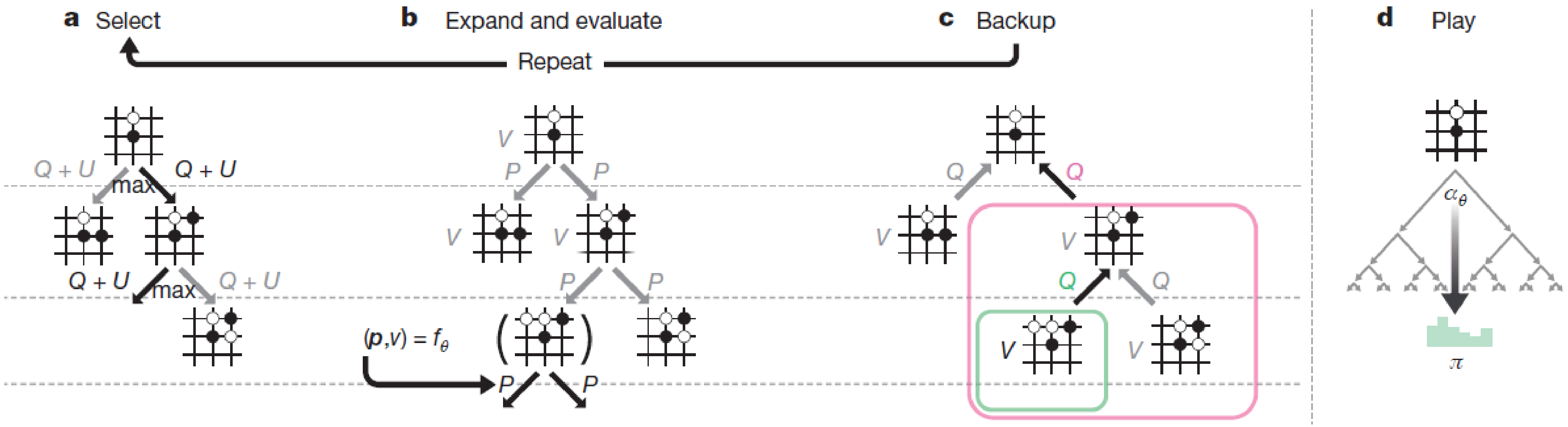
- Action : action to arrive this state/node
- N : this state's/node's visit time
- W : this state's/node's total value
- Q : this state's/node's mean value

MCTS



Real board state

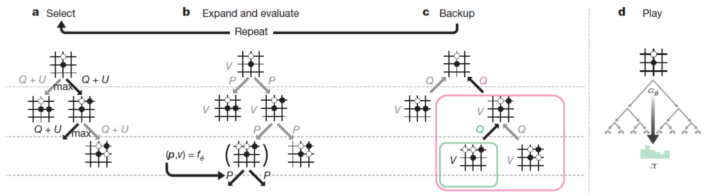
Begin!



MCTS

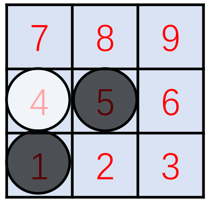
Select

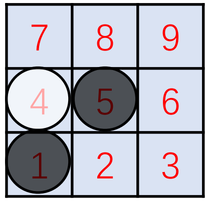
- If the node is in tree, we can select an action to go to the next node/state.
- If the node is leaf node, there is no node to go and we should expand the children nodes.



Action = 1
N=0,W=0,Q=0
P=0

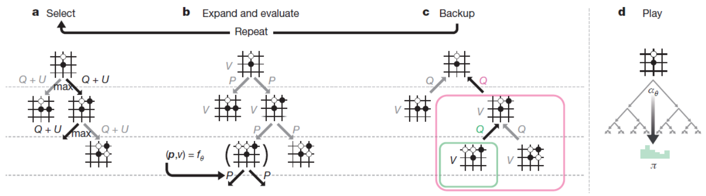
white

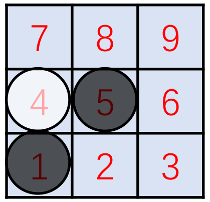




Real board state

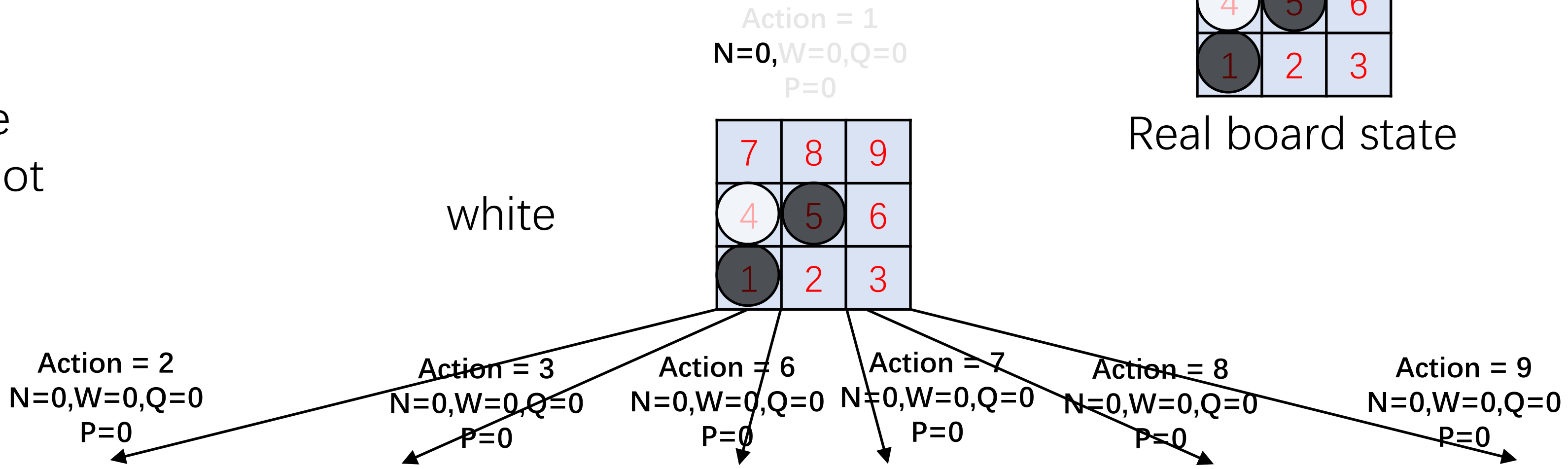
Expand and evaluate





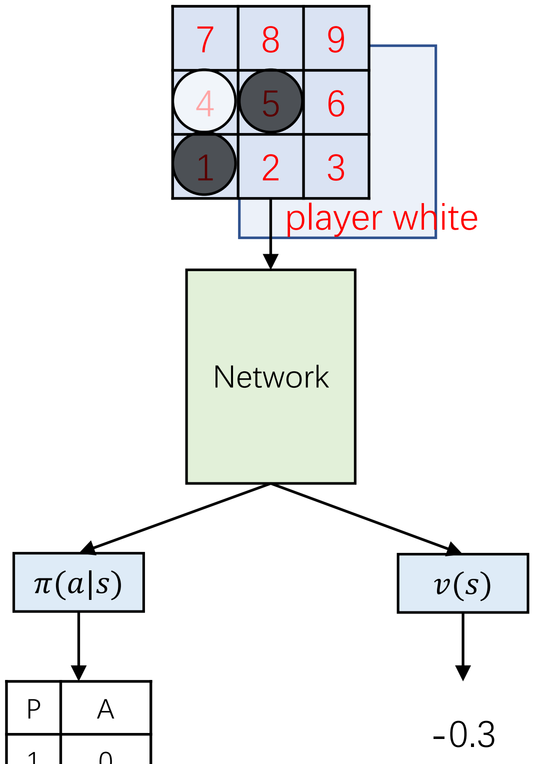
Real board state

- We assumed this was root node
  So the parameters in grey will not be used!

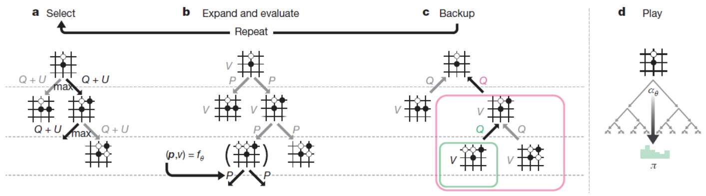


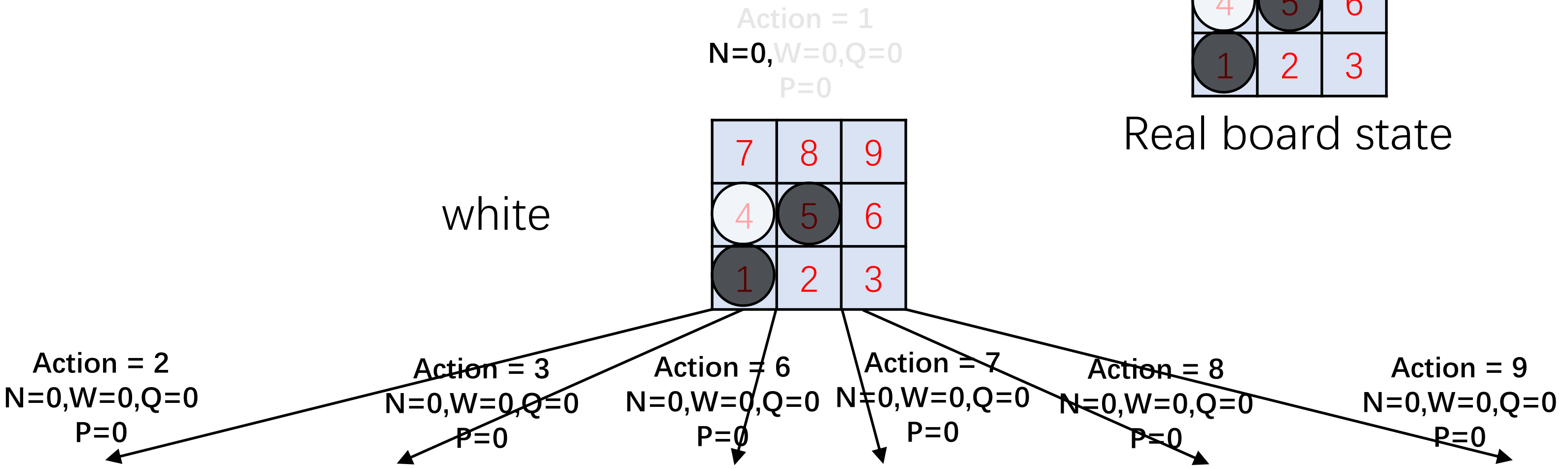
Expand and evaluate

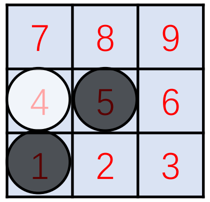
MCTS



P	A
1	0
2	0.4
3	0.1
4	0
5	0
6	0.05
7	0.1
8	0.15
9	0.2



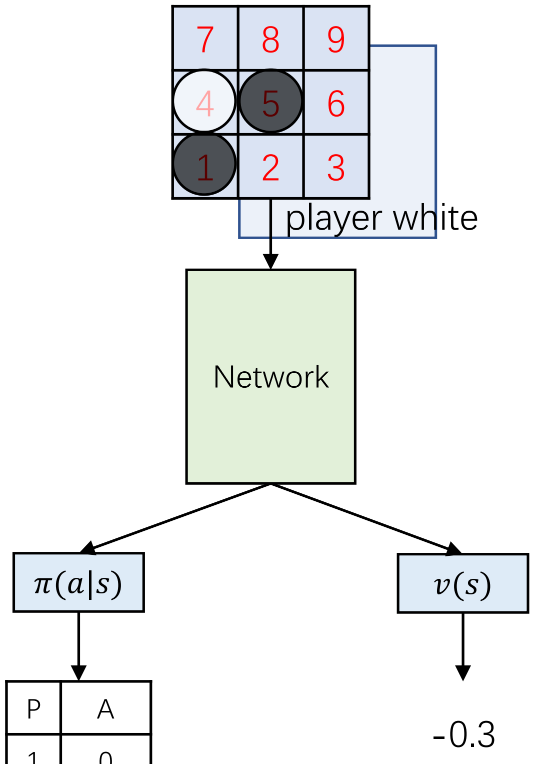




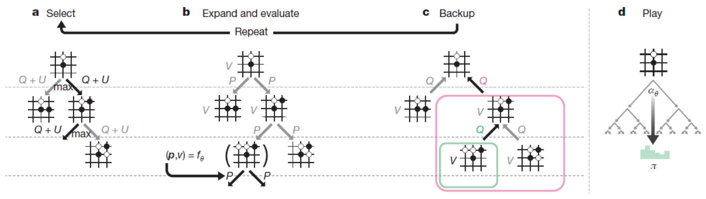
Real board state

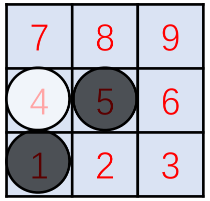
MCTS

Expand and evaluate

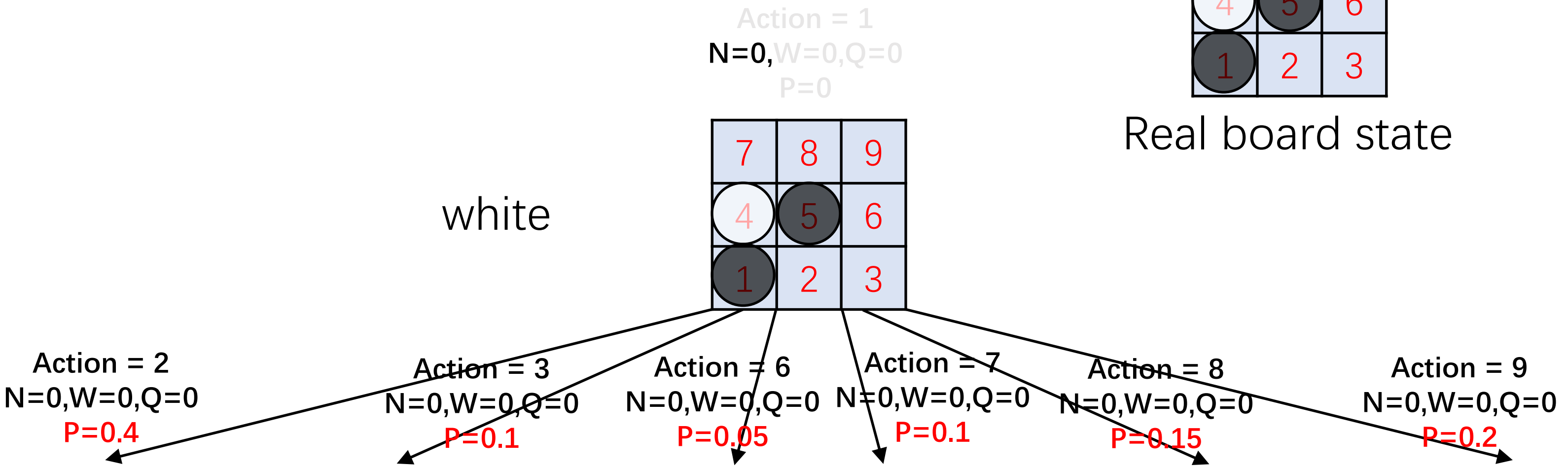


P	A
1	0
2	0.4
3	0.1
4	0
5	0
6	0.05
7	0.1
8	0.15
9	0.2



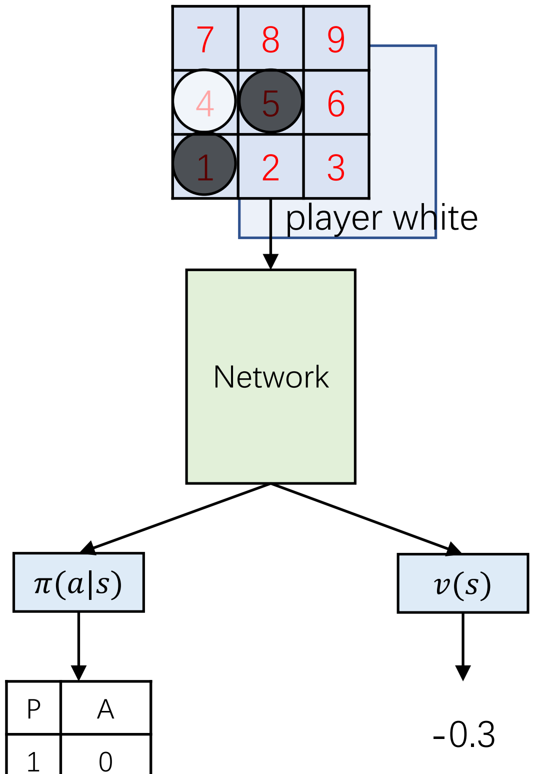


Real board state

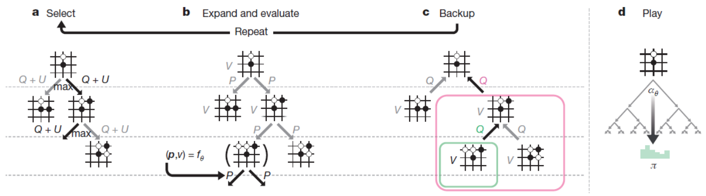


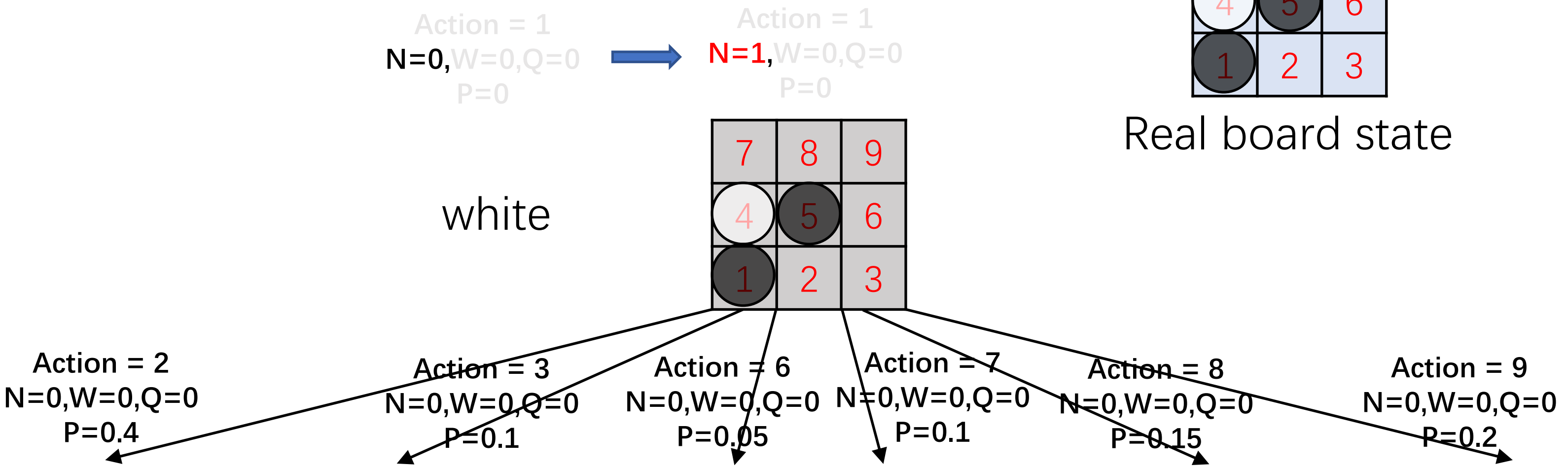
Backup

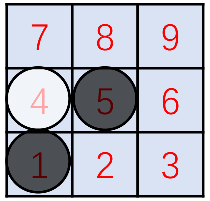
MCTS



P	A
1	0
2	0.4
3	0.1
4	0
5	0
6	0.05
7	0.1
8	0.15
9	0.2



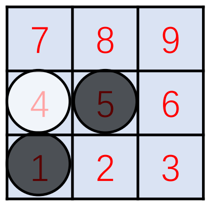




Real board state

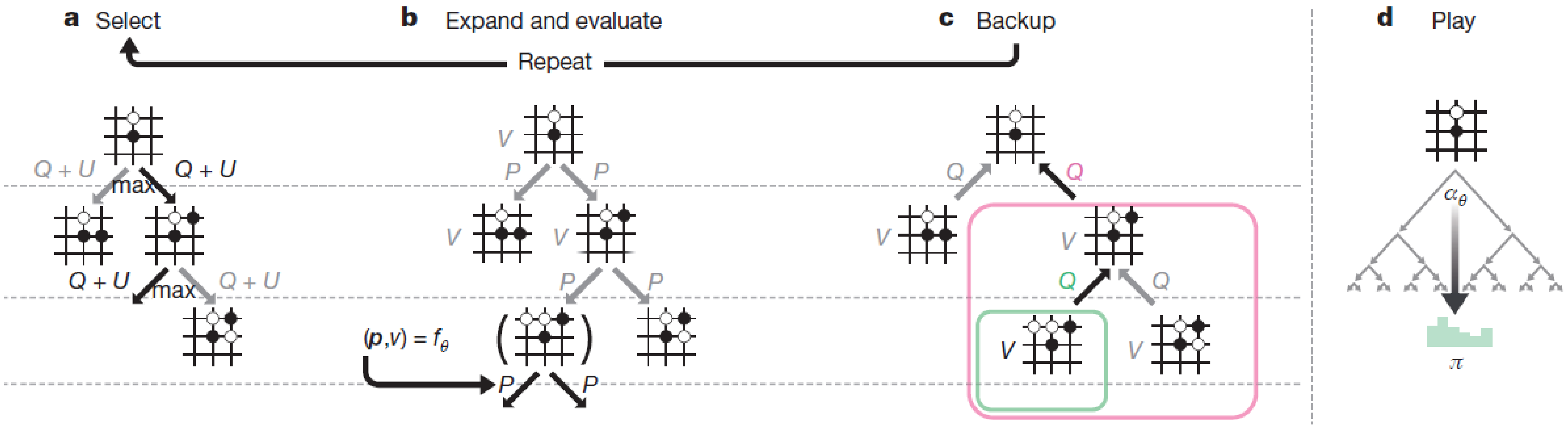
$$N(s,a) = N(s,a) + 1$$
$$W(s,a) = W(s,a) + v(s)$$
$$Q(s,a) = \frac{W(s,a)}{N(s,a)}$$

- We assumed this was root node before. So no need to backup value here!
- But we need to change the visit time from N to N+1 for later use.

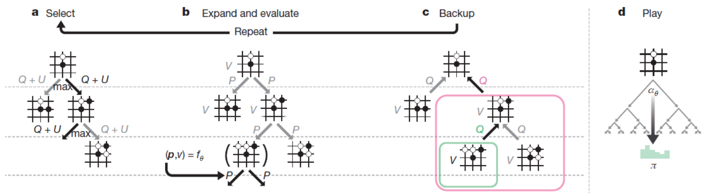


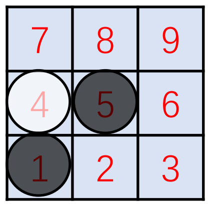
Real board state

Done！
And one more !



Select



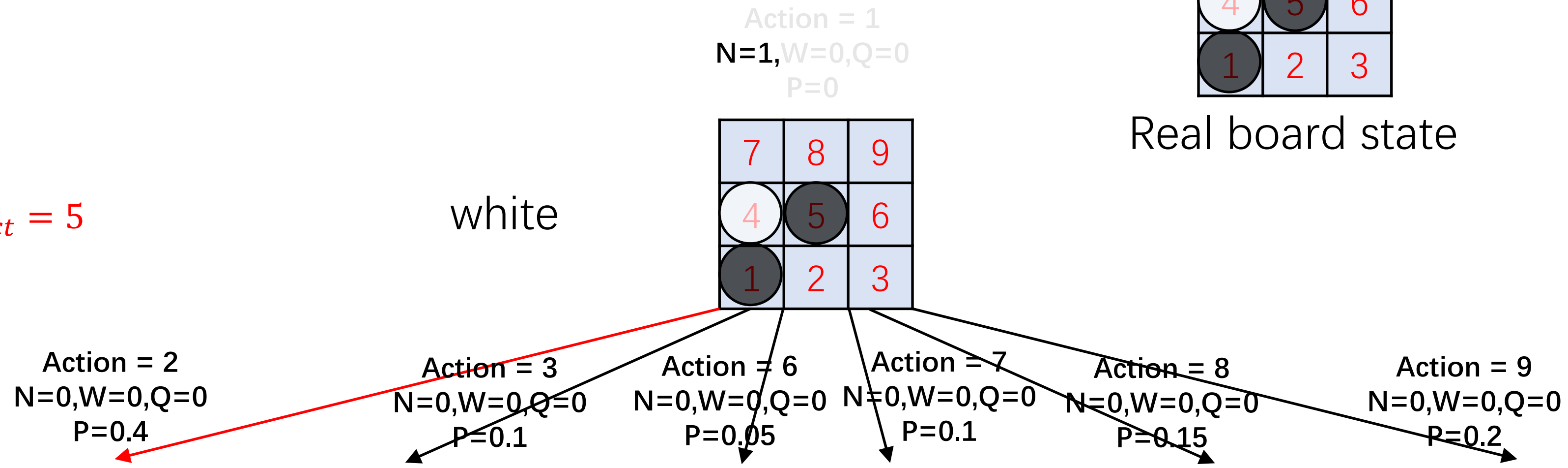


Real board state

$$a = argmax_a(Q(s,a) + U(s,a))$$

Exploration : $U(s,a) = c_{puct}P(s,a)\frac{\sqrt{\sum_b N(s,b)}}{1+N(s,a)}, c_{puct} = 5$

Exploitation : $Q(s,a) = \frac{W}{N}$

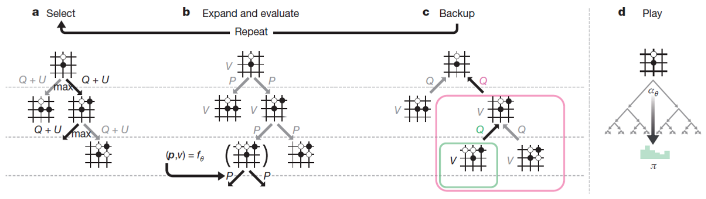


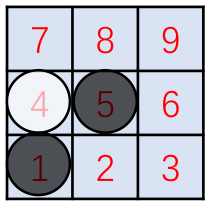
Select

$$a = argmax_a(Q(s,a) + U(s,a))$$

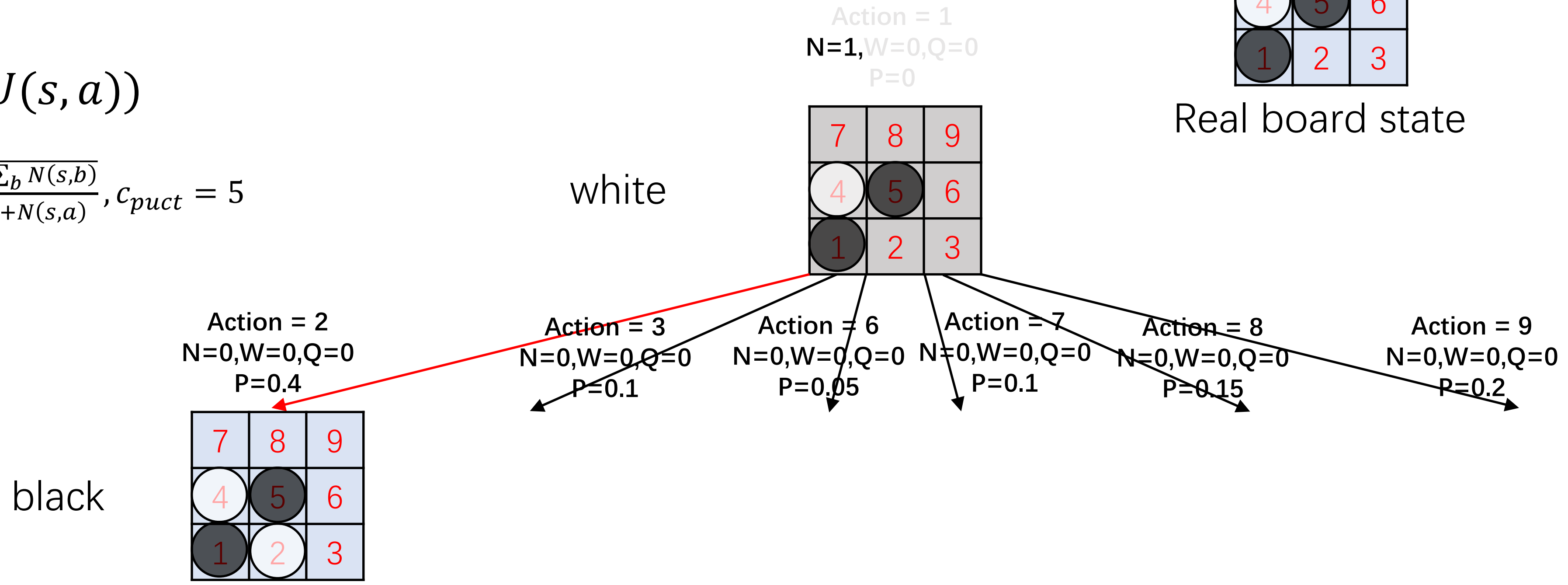
Exploration : $U(s,a) = c_{puct} P(s,a) \frac{\sqrt{\sum_b N(s,b)}}{1+N(s,a)}, c_{puct} = 5$

Exploitation : $Q(s,a) = \frac{W}{N}$



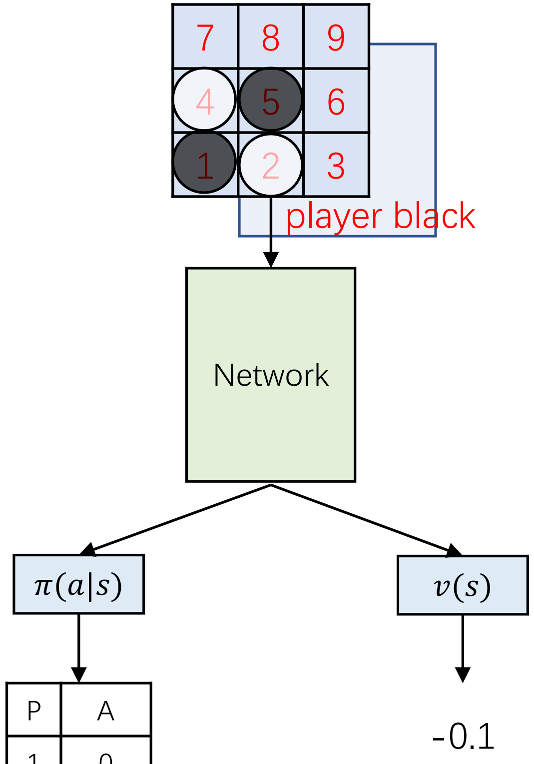


Real board state

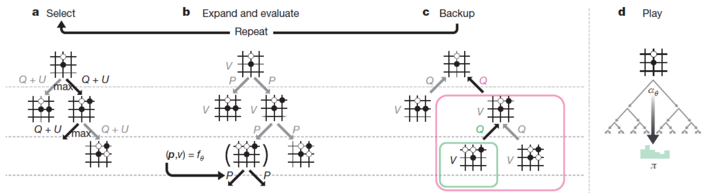


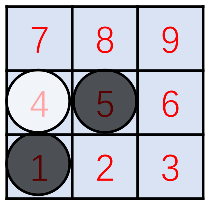
MCTS

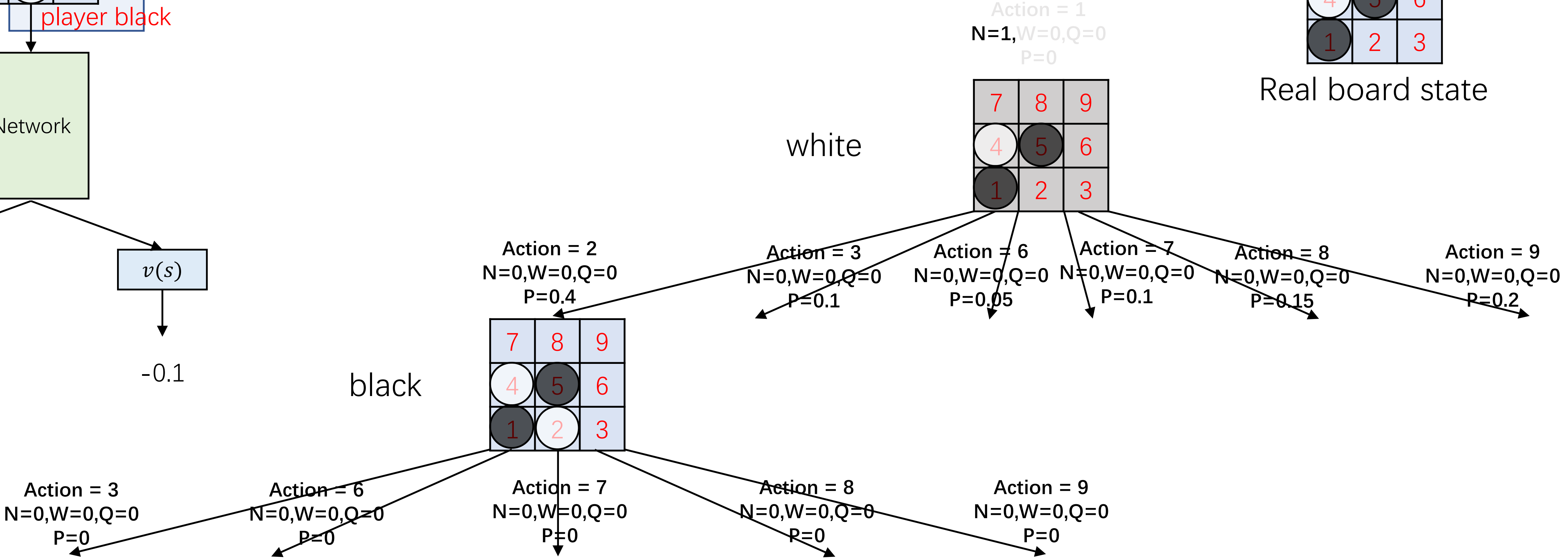
Expand and evaluate



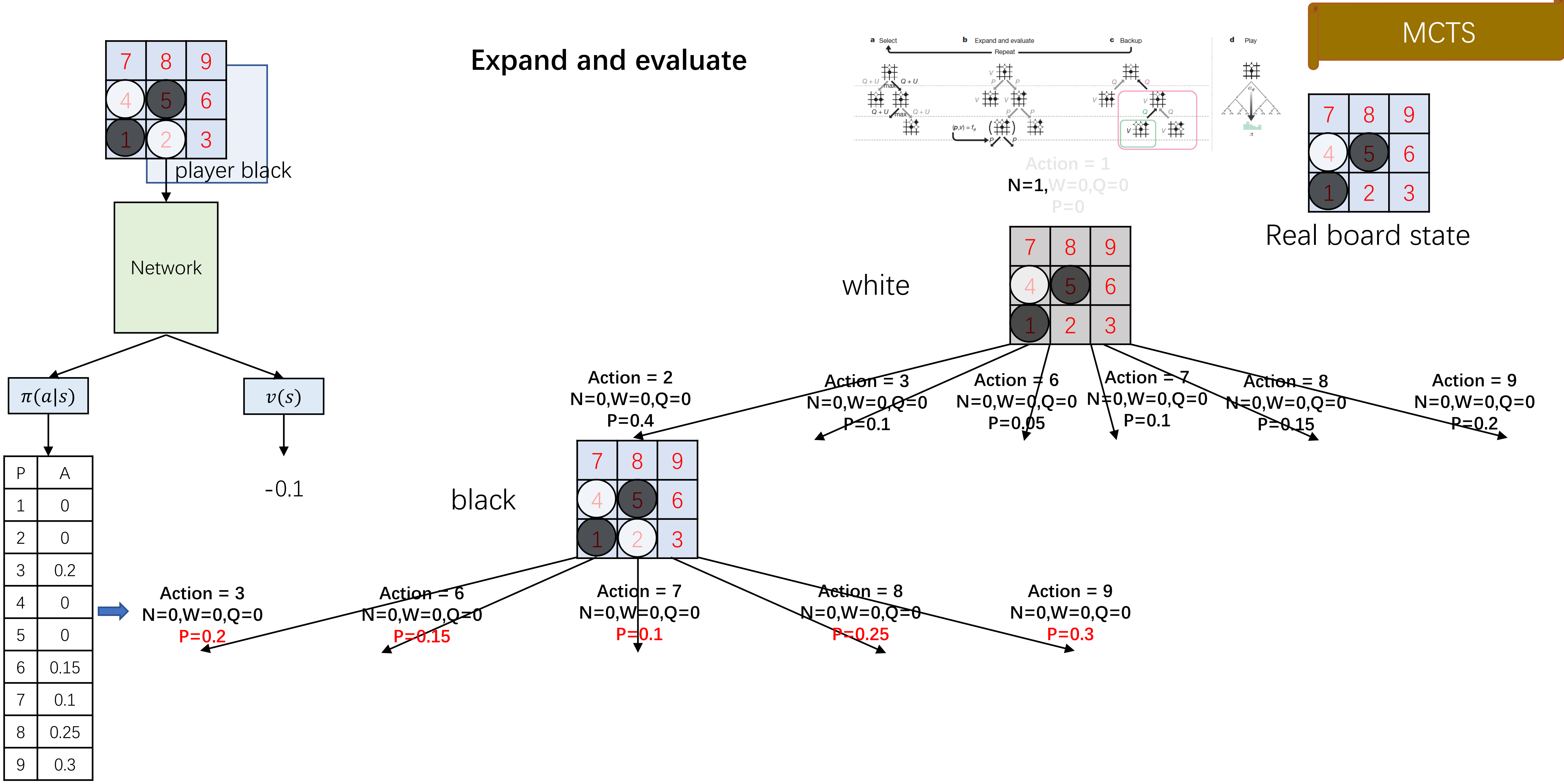
P	A
1	0
2	0
3	0.2
4	0
5	0
6	0.15
7	0.1
8	0.25
9	0.3





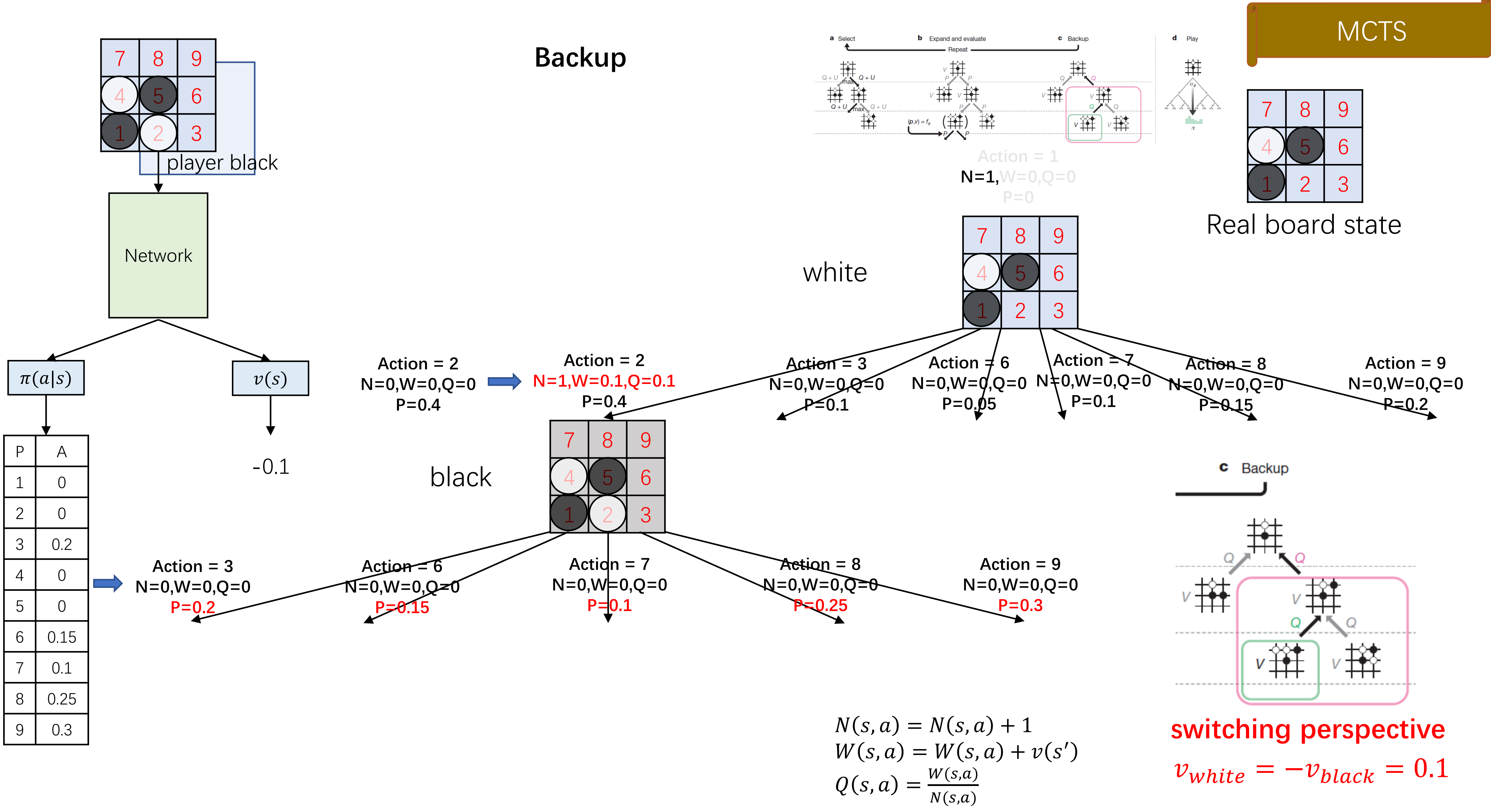


Expand and evaluate



P	A
1	0
2	0
3	0.2
4	0
5	0
6	0.15
7	0.1
8	0.25
9	0.3

Backup

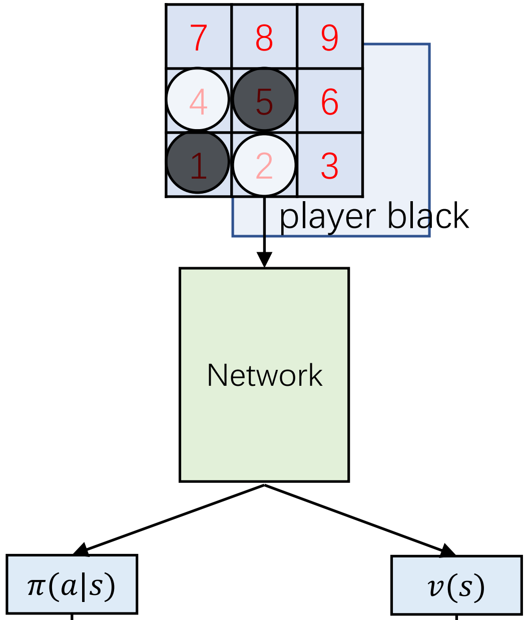


P	A
1	0
2	0
3	0.2
4	0
5	0
6	0.15
7	0.1
8	0.25
9	0.3

$$N(s,a) = N(s,a) + 1$$
$$W(s,a) = W(s,a) + v(s')$$
$$Q(s,a) = \frac{W(s,a)}{N(s,a)}$$

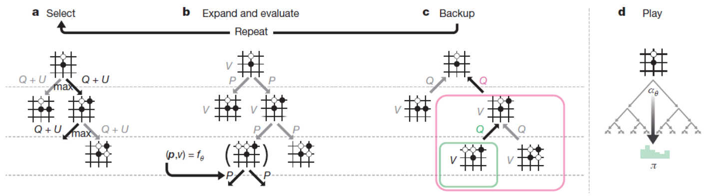
$$v_{white} = -v_{black} = 0.1$$

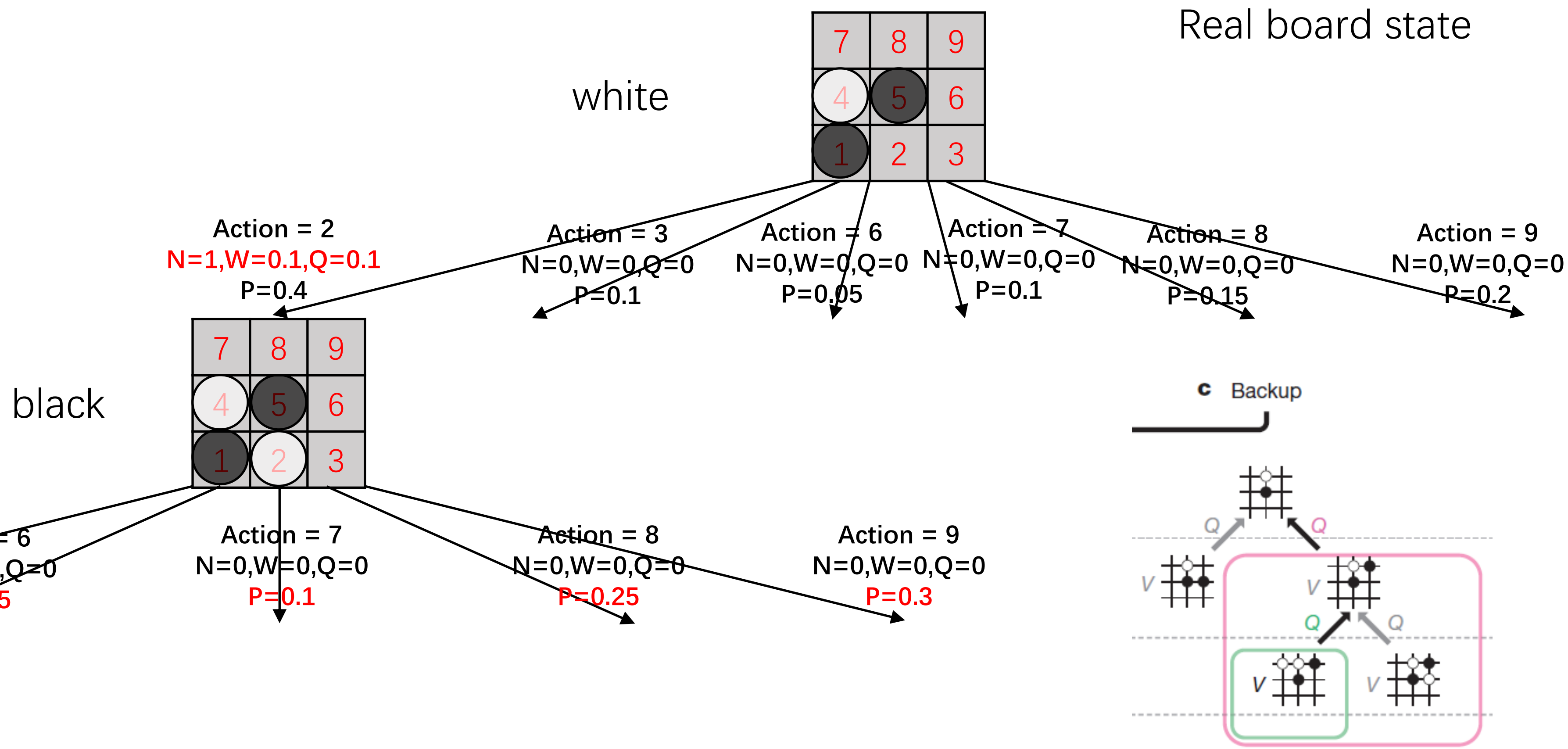
Backup

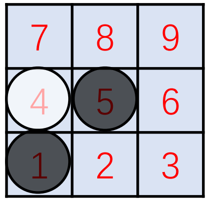


P	A
1	0
2	0
3	0.2
4	0
5	0
6	0.15
7	0.1
8	0.25
9	0.3

-0.1





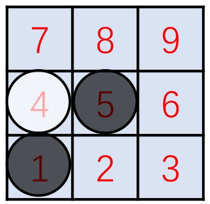


Real board state

switching perspective

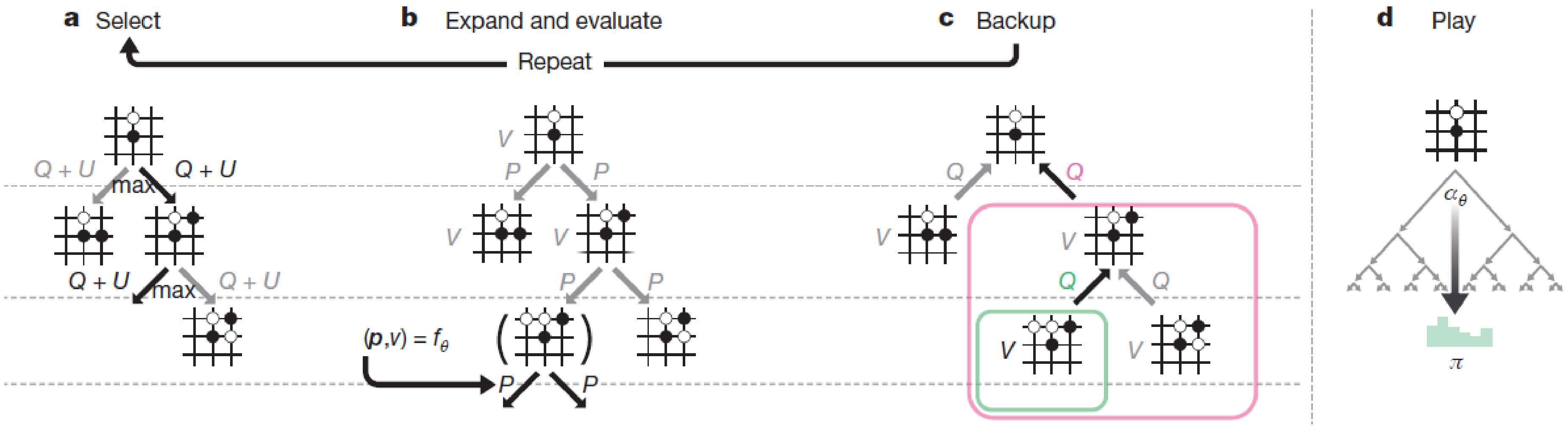
$$v_{white} = -v_{black} = 0.1$$

$$N(s,a) = N(s,a) + 1$$
$$W(s,a) = W(s,a) + v(s')$$
$$Q(s,a) = \frac{W(s,a)}{N(s,a)}$$

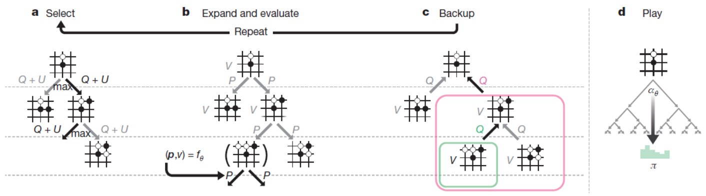


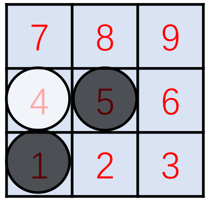
Real board state

Done！
And one more !



Select



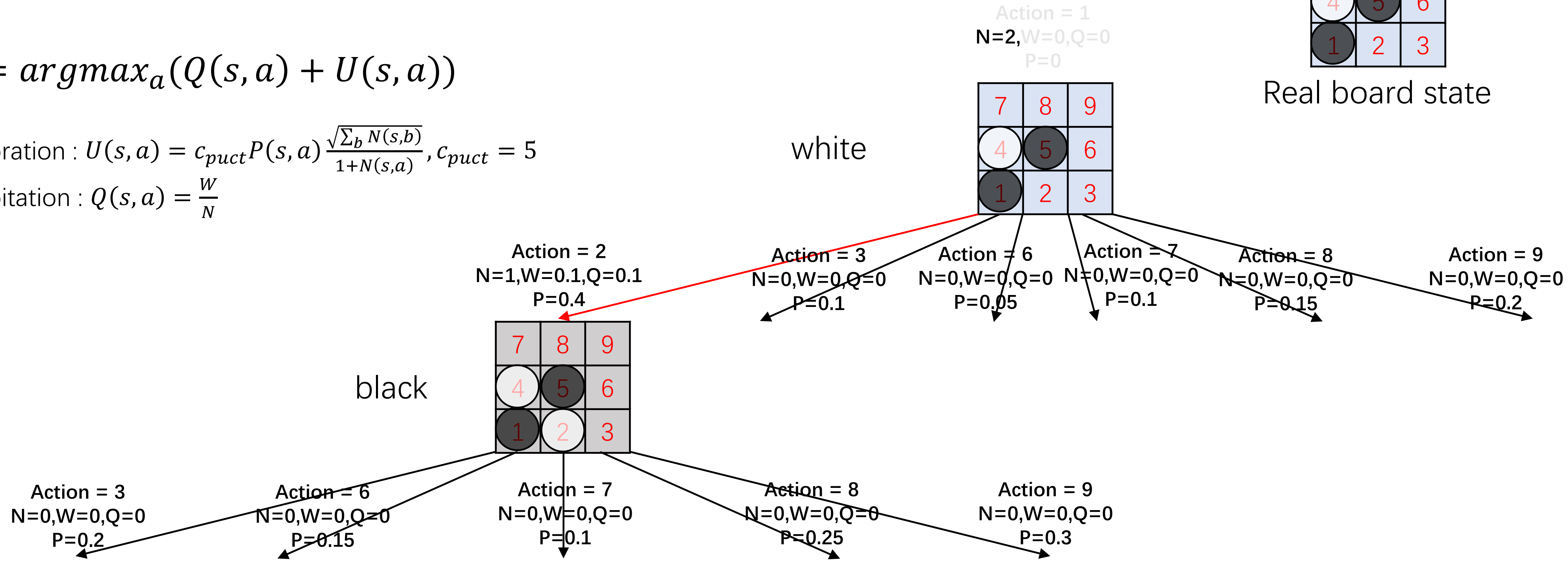


Real board state

$$a = argmax_a(Q(s,a) + U(s,a))$$

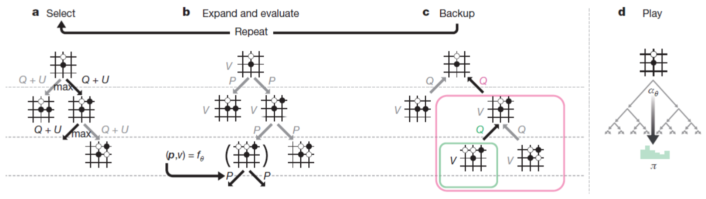
Exploration : $U(s,a) = c_{puct}P(s,a)\frac{\sqrt{\sum_b N(s,b)}}{1+N(s,a)}, c_{puct} = 5$

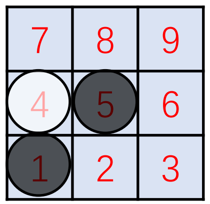
Exploitation : $Q(s,a) = \frac{W}{N}$



MCTS

Select



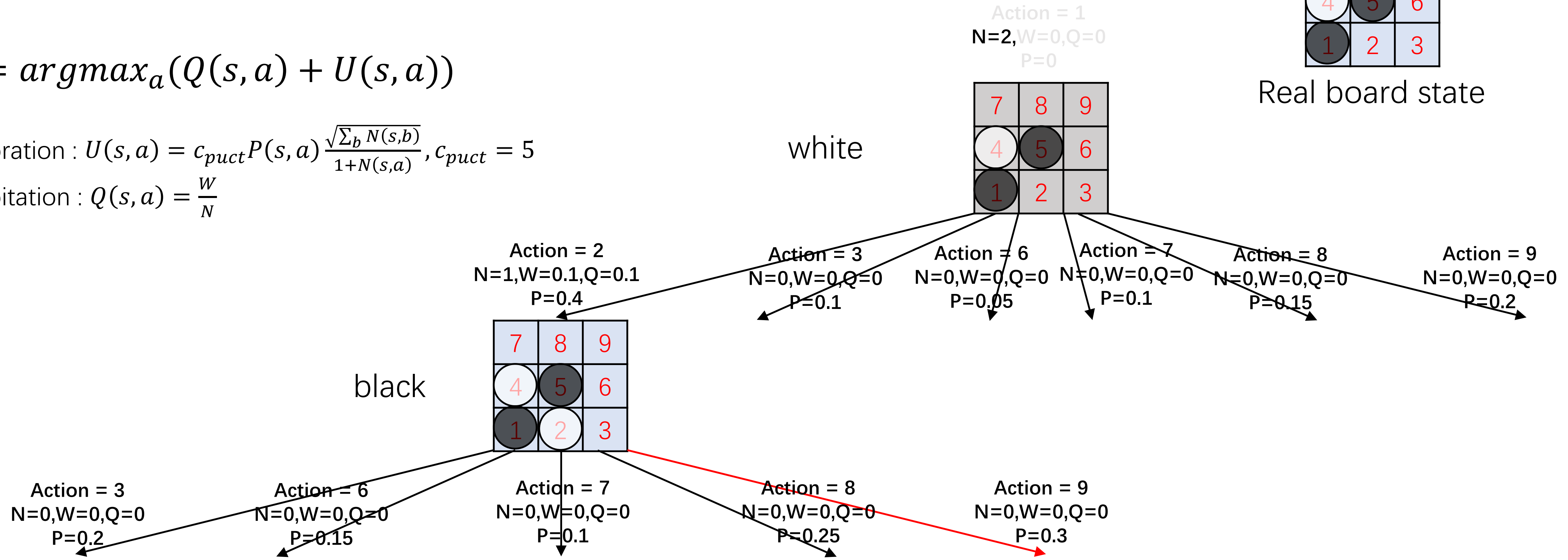


Real board state

$$a = argmax_a(Q(s,a) + U(s,a))$$

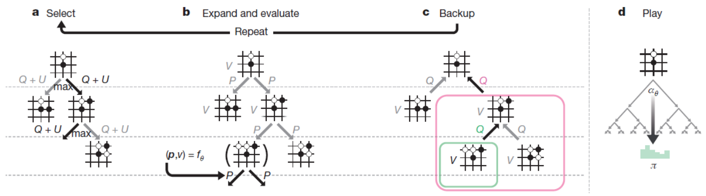
Exploration : $U(s,a) = c_{puct}P(s,a)\frac{\sqrt{\sum_b N(s,b)}}{1+N(s,a)}, c_{puct} = 5$

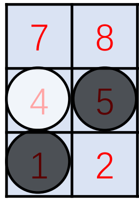
Exploitation : $Q(s,a) = \frac{W}{N}$



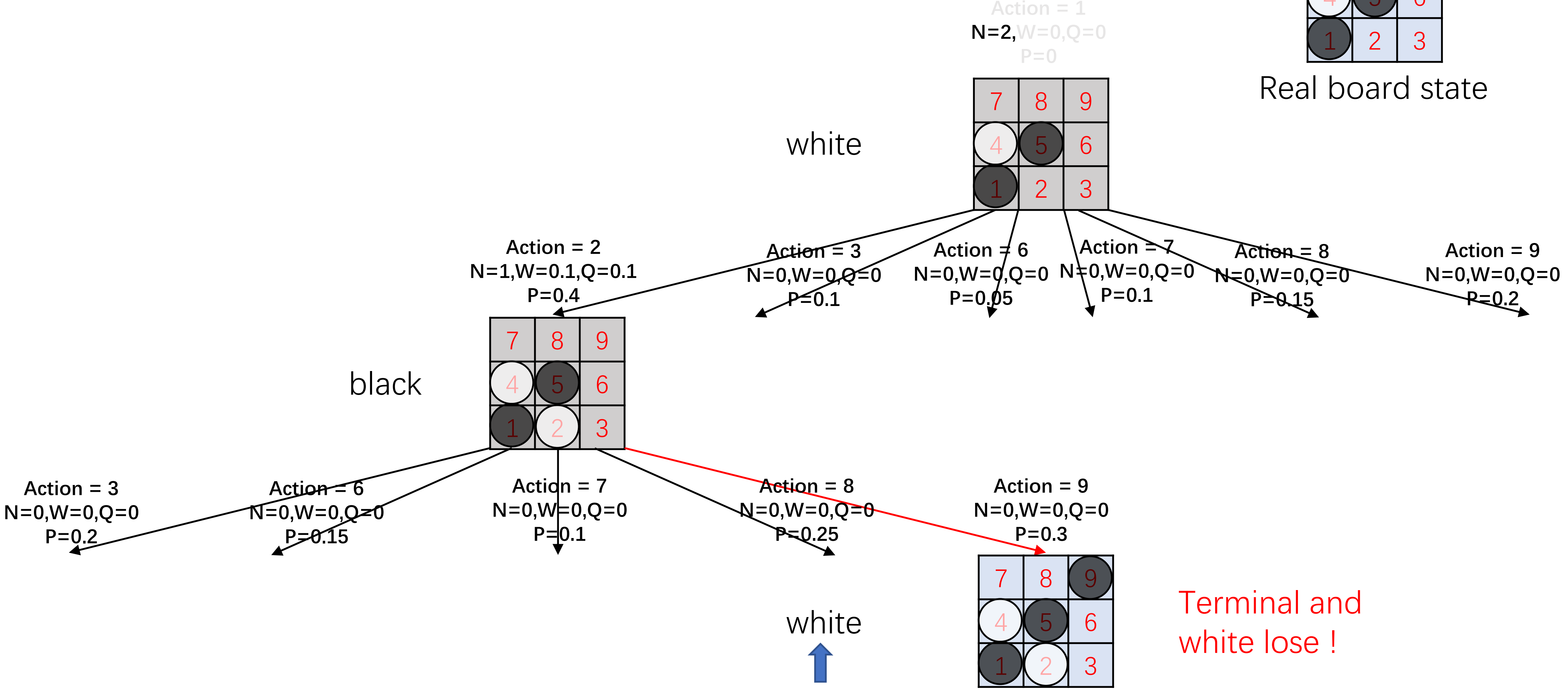
Expand and evaluate

MCTS





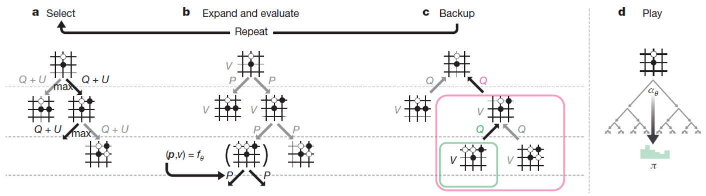
Real board state

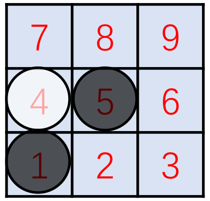


Reward = -1

- Note that we always get the reward from the current perspective.

Backup



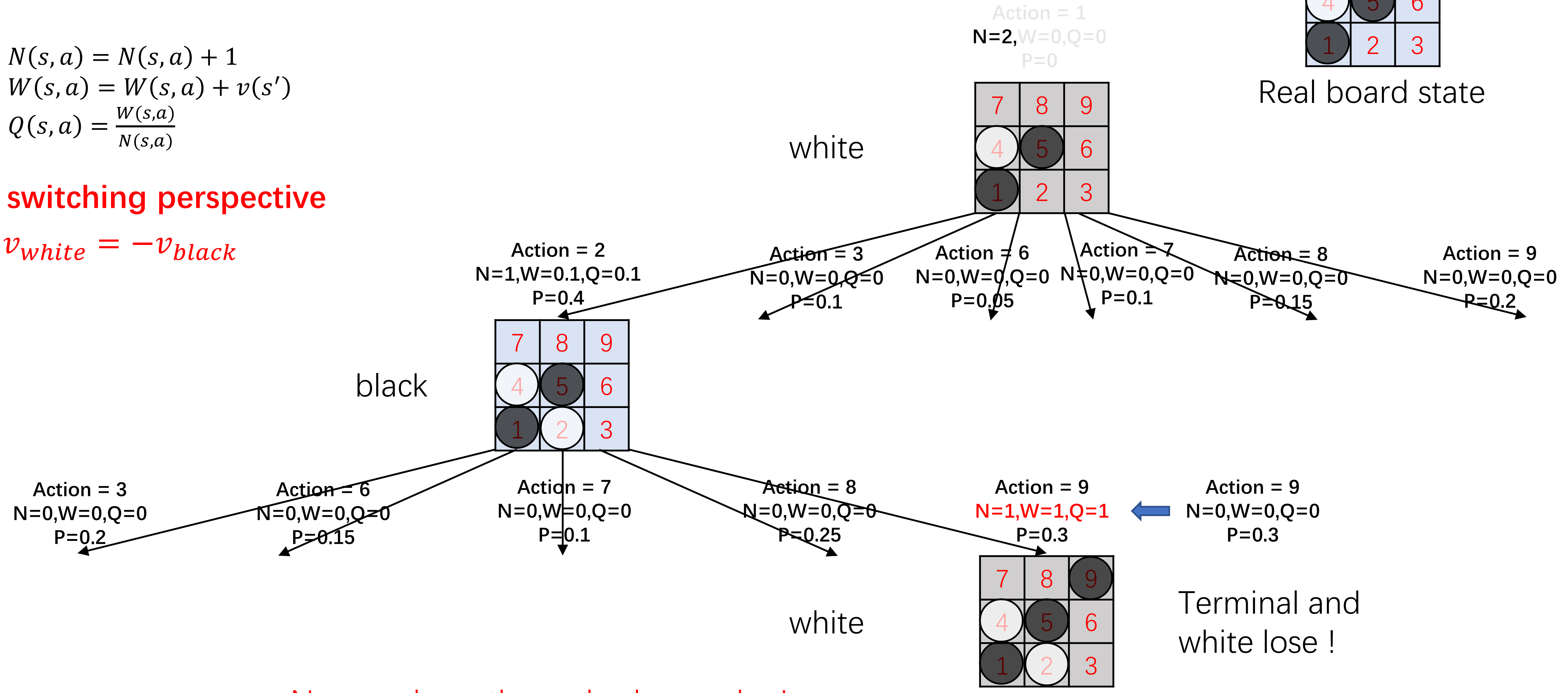


Real board state

$$N(s,a) = N(s,a) + 1$$
$$W(s,a) = W(s,a) + v(s')$$
$$Q(s,a) = \frac{W(s,a)}{N(s,a)}$$

switching perspective

$$v_{white} = -v_{black}$$

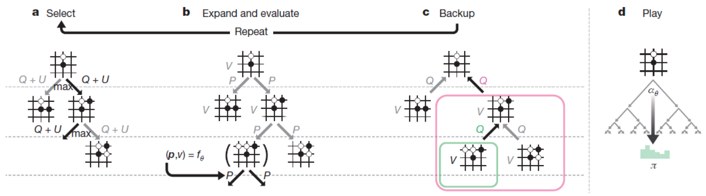


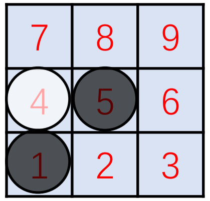
- Now we have the real value so don't need a predictive value from network.

Reward = -1

Backup

MCTS





Real board state

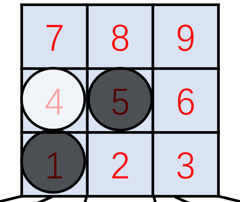
$$N(s,a) = N(s,a) + 1$$
$$W(s,a) = W(s,a) + v(s')$$
$$Q(s,a) = \frac{W(s,a)}{N(s,a)}$$

switching perspective

$$v_{white} = -v_{black}$$

Action = 1
N=2,W=0,Q=0
P=0

white



Action = 2
N=1,W=0.1,Q=0.1 → N=2,W=-0.9,Q=-0.45
P=0.4

Action = 3
N=0,W=0,Q=0
P=0.1

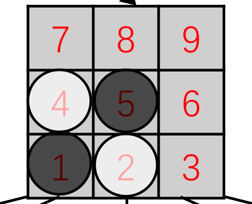
Action = 6
N=0,W=0,Q=0
P=0.05

Action = 7
N=0,W=0,Q=0
P=0.1

Action = 8
N=0,W=0,Q=0
P=0.15

Action = 9
N=0,W=0,Q=0
P=0.2

black



Action = 3
N=0,W=0,Q=0
P=0.2

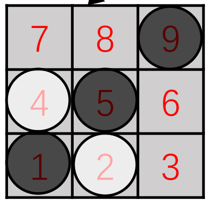
Action = 6
N=0,W=0,Q=0
P=0.15

Action = 7
N=0,W=0,Q=0
P=0.1

Action = 8
N=0,W=0,Q=0
P=0.25

Action = 9
N=1,W=1,Q=1
P=0.3

white

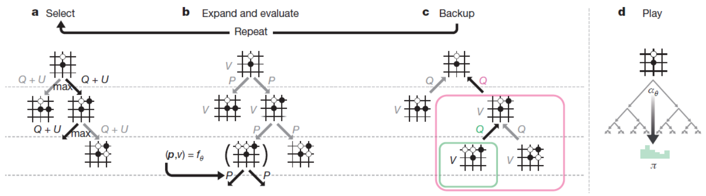


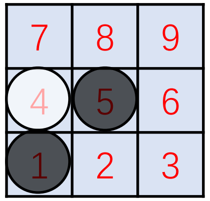
Terminal and white lose !

Reward = -1

- Now we have the real value so don't need a predictive value from network.

Backup



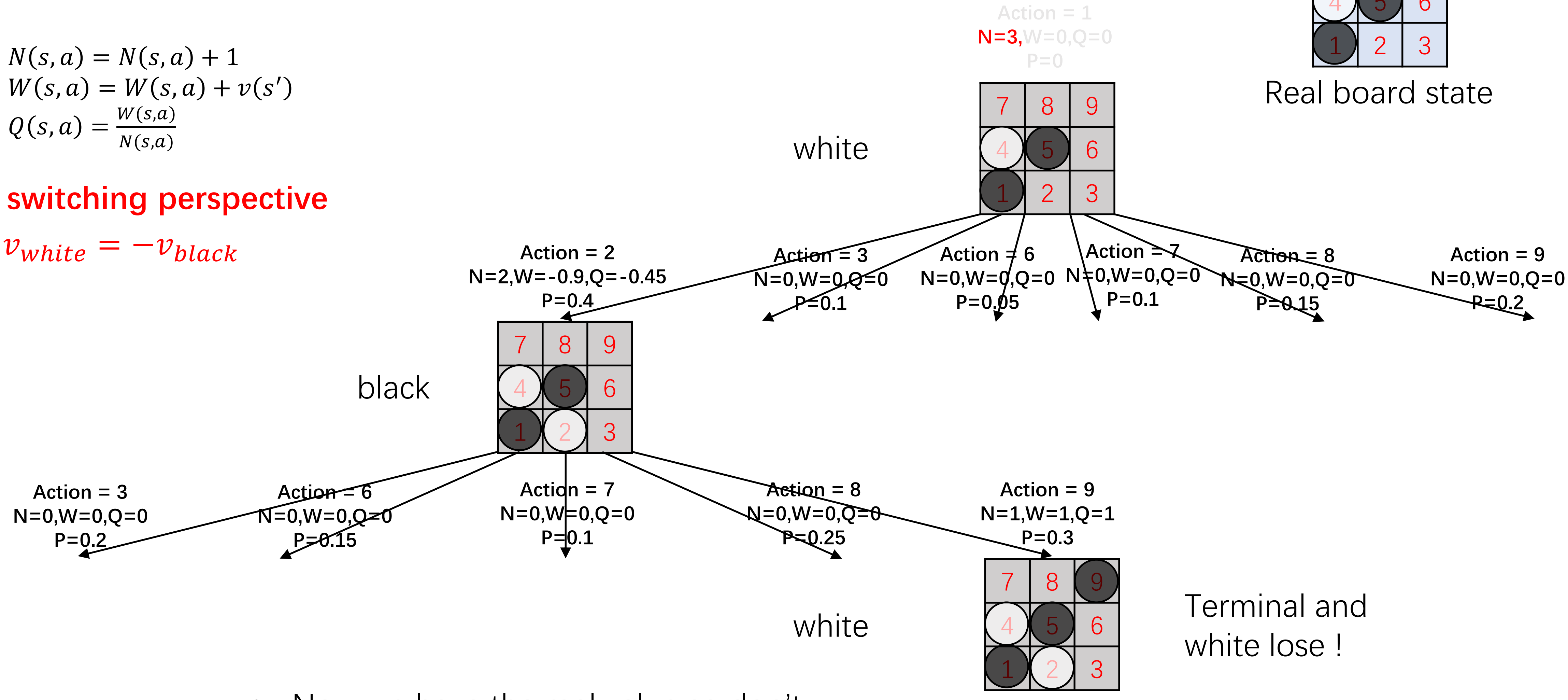


Real board state

$$N(s,a) = N(s,a) + 1$$
$$W(s,a) = W(s,a) + v(s')$$
$$Q(s,a) = \frac{W(s,a)}{N(s,a)}$$

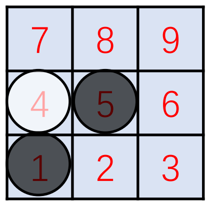
switching perspective

$$v_{white} = -v_{black}$$



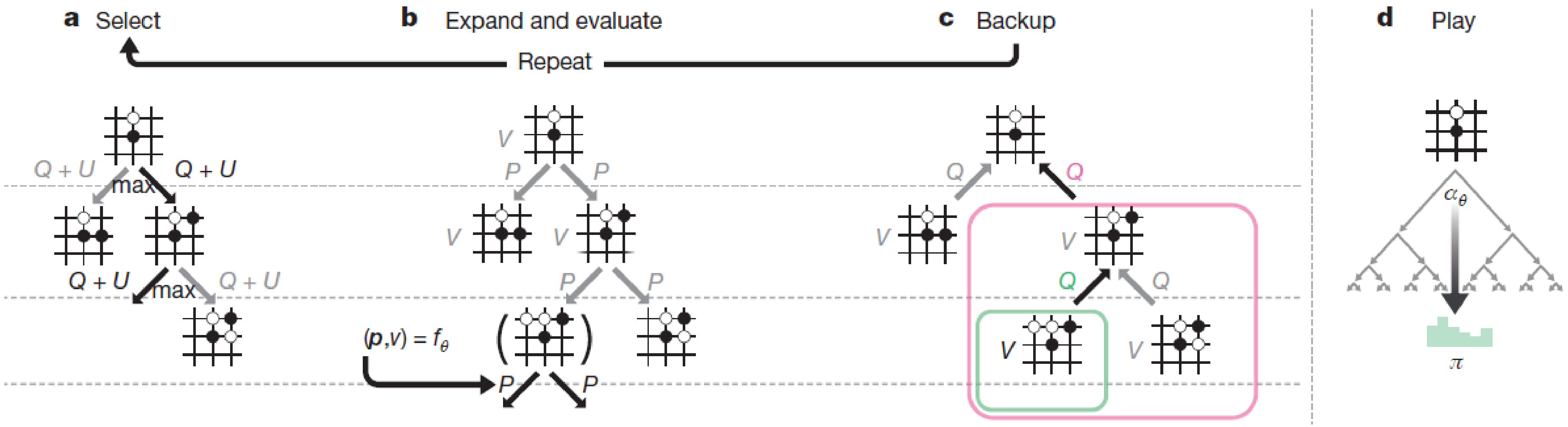
Reward = -1

- Now we have the real value so don't need a predictive value from network.

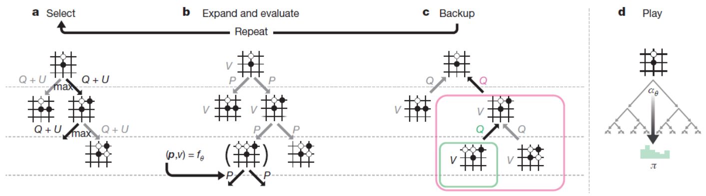


Real board state

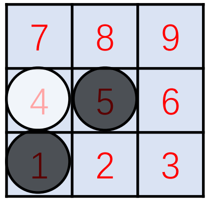
Done！
And one more !



Select



MCTS

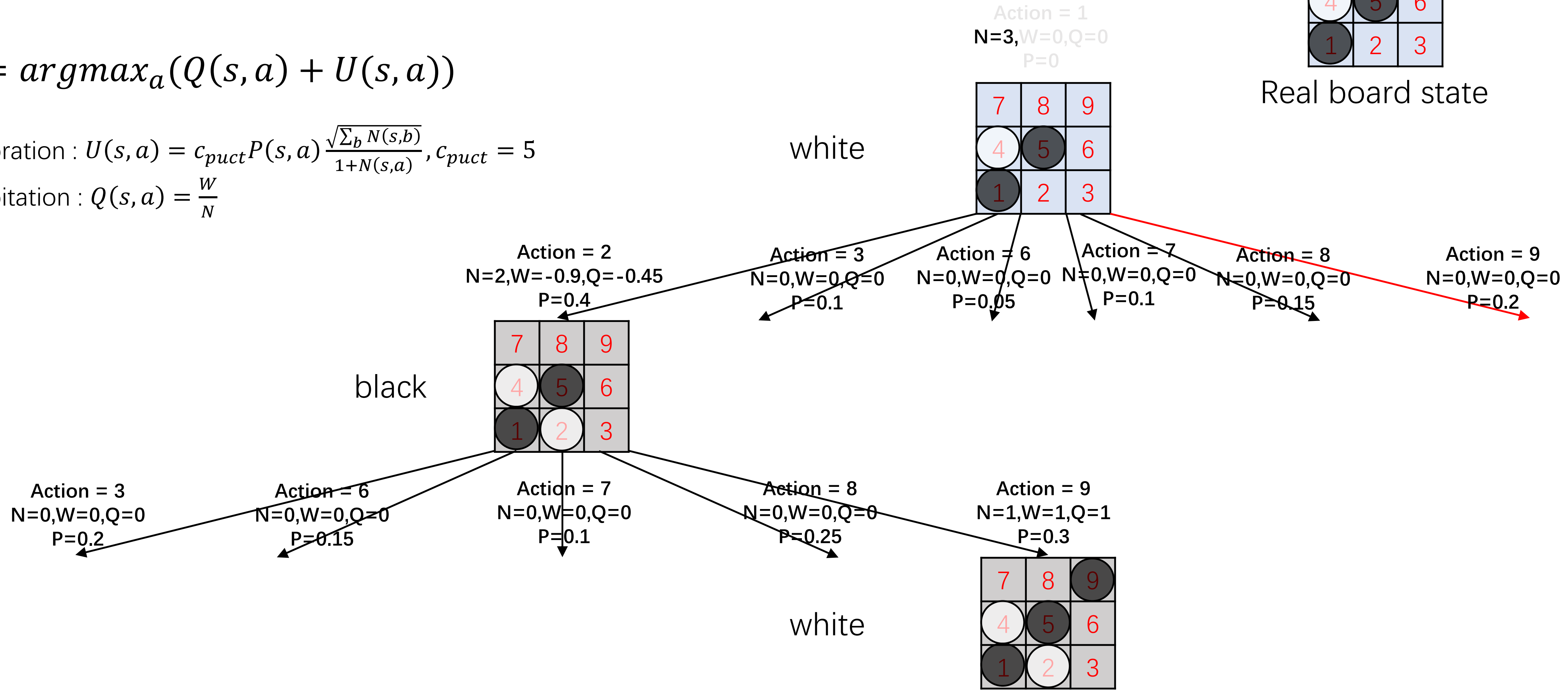


Real board state

$$a = argmax_a(Q(s,a) + U(s,a))$$

Exploration : $U(s,a) = c_{puct}P(s,a)\frac{\sqrt{\sum_b N(s,b)}}{1+N(s,a)}, c_{puct} = 5$

Exploitation : $Q(s,a) = \frac{W}{N}$



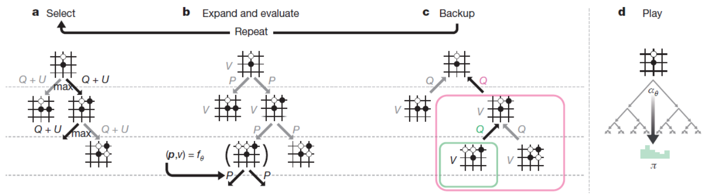
MCTS

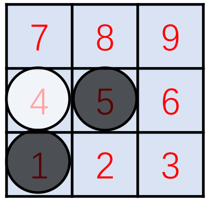
Select

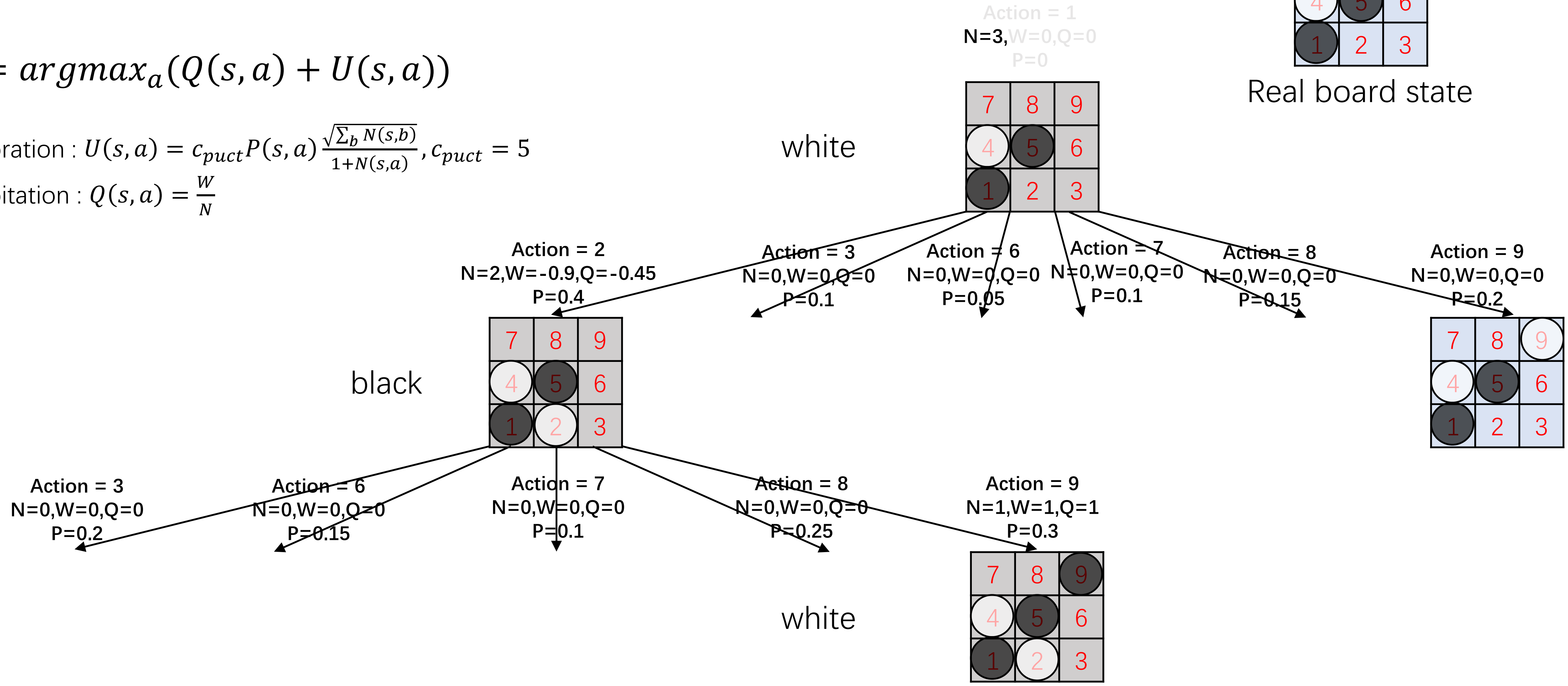
$$a = argmax_a(Q(s,a) + U(s,a))$$

Exploration : $U(s,a) = c_{puct}P(s,a)\frac{\sqrt{\sum_b N(s,b)}}{1+N(s,a)}, c_{puct} = 5$

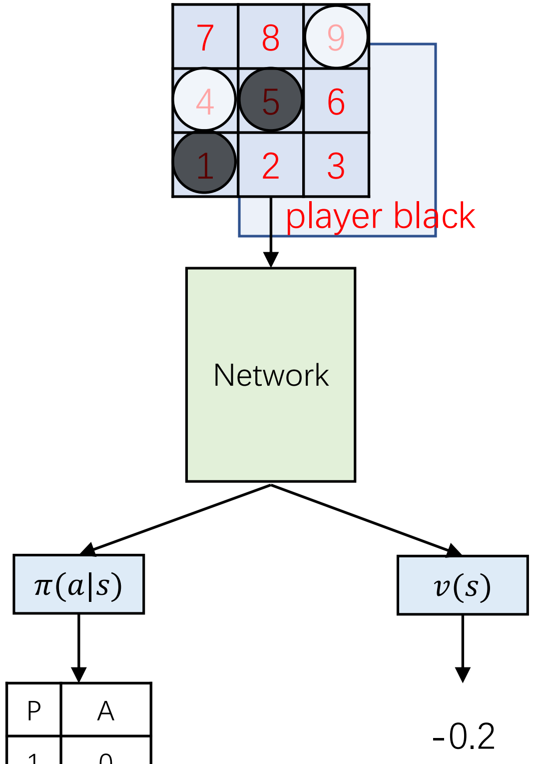
Exploitation : $Q(s,a) = \frac{W}{N}$



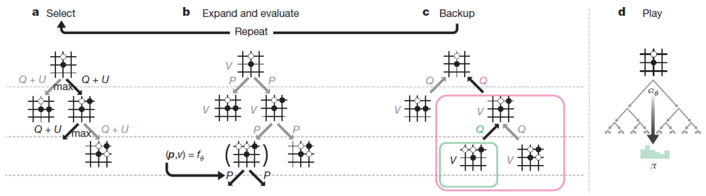


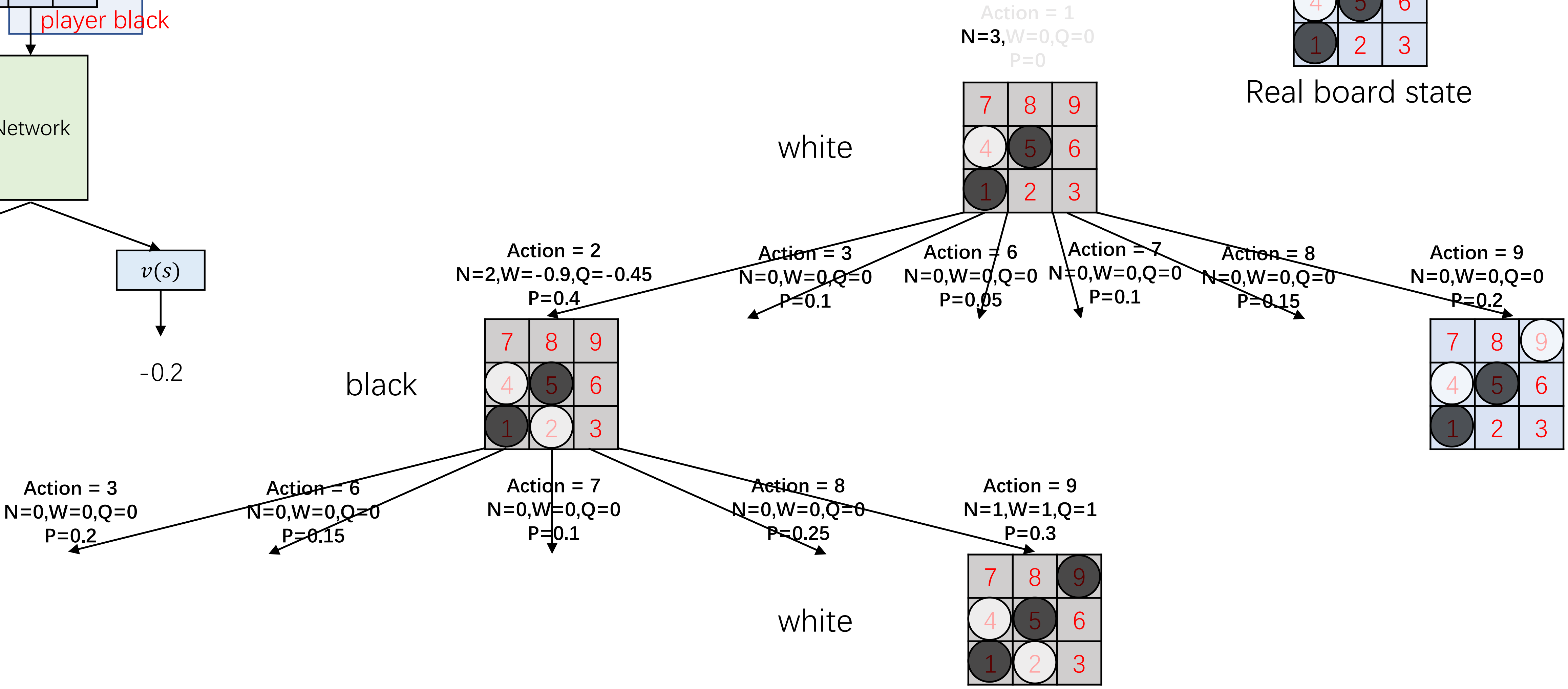


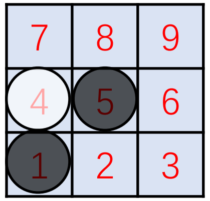
Expand and evaluate



P	A
1	0
2	0.25
3	0.2
4	0
5	0
6	0.1
7	0.2
8	0.25
9	0



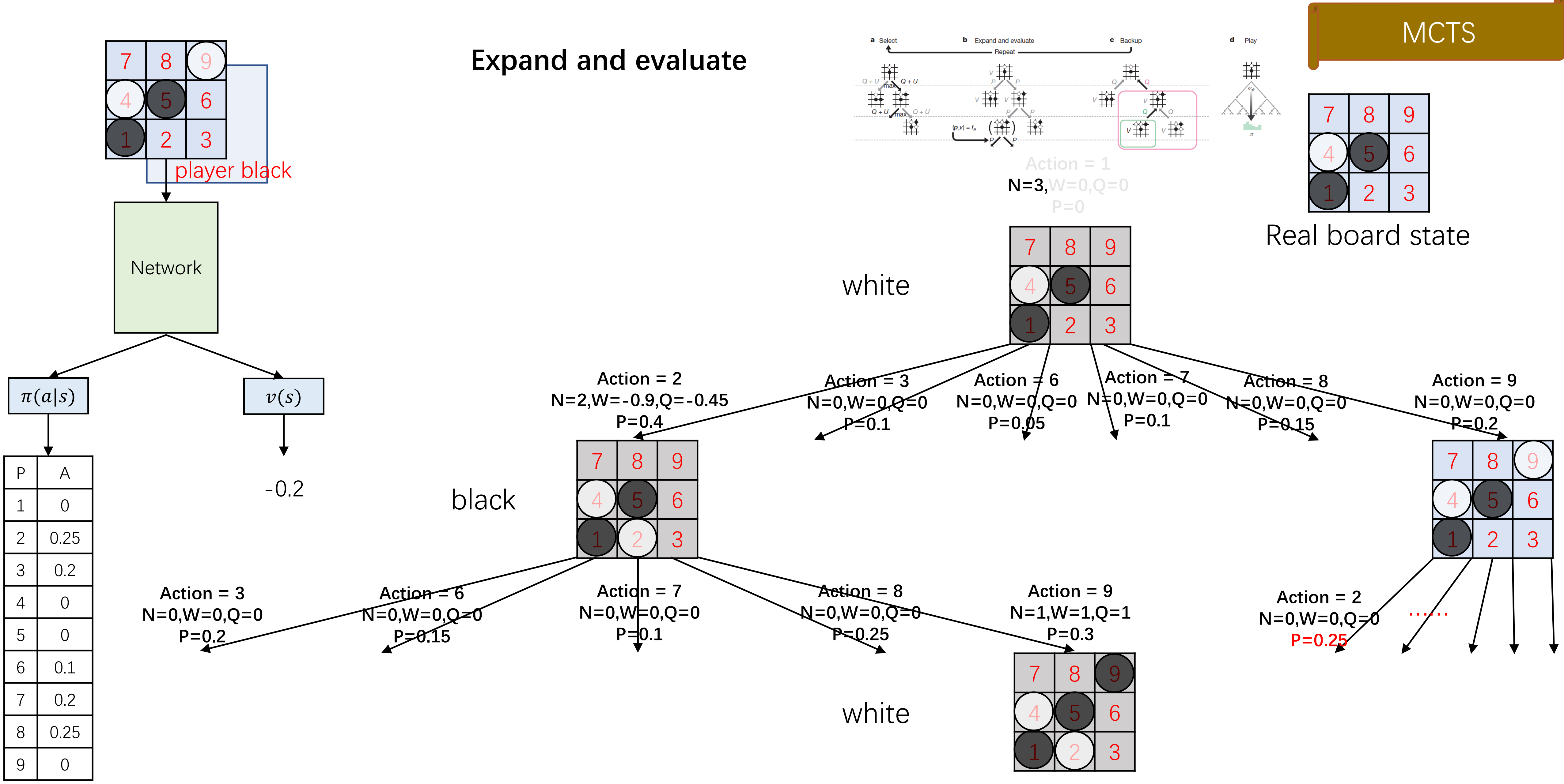




Real board state

Expand and evaluate

MCTS



P	A
1	0
2	0.25
3	0.2
4	0
5	0
6	0.1
7	0.2
8	0.25
9	0

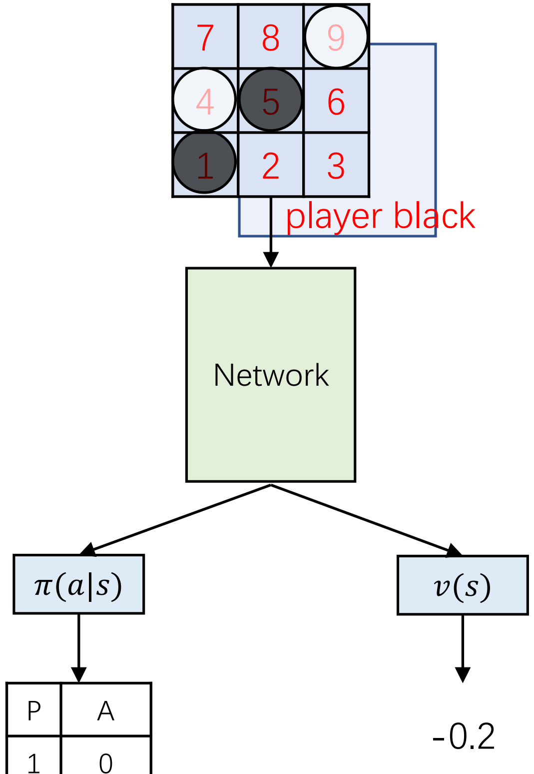
MCTS

Backup

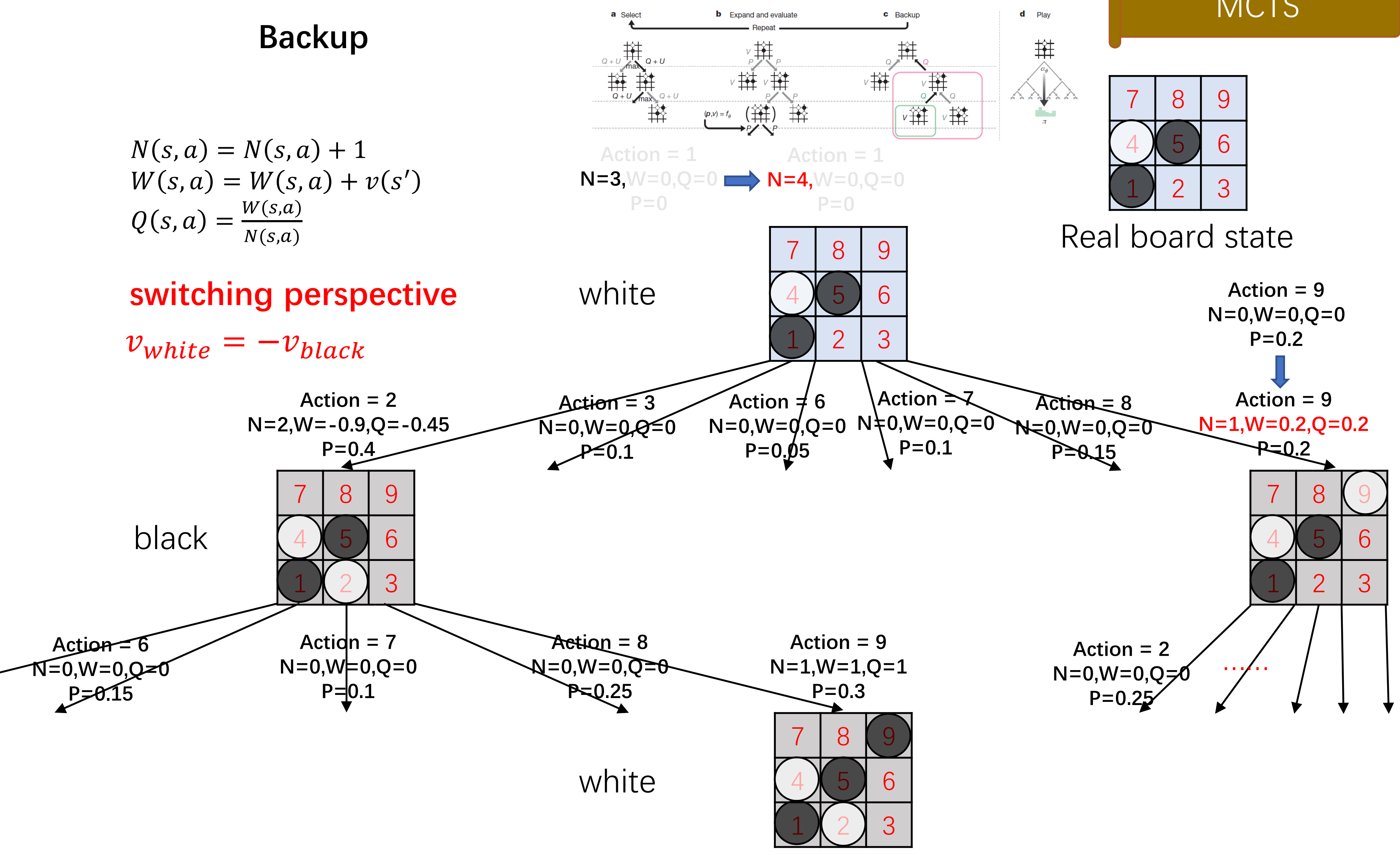
$$N(s,a) = N(s,a) + 1$$
$$W(s,a) = W(s,a) + v(s')$$
$$Q(s,a) = \frac{W(s,a)}{N(s,a)}$$

switching perspective

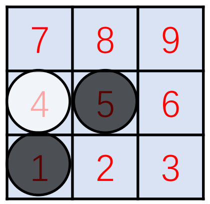
$$v_{white} = -v_{black}$$



P	A
1	0
2	0.25
3	0.2
4	0
5	0
6	0.1
7	0.2
8	0.25
9	0

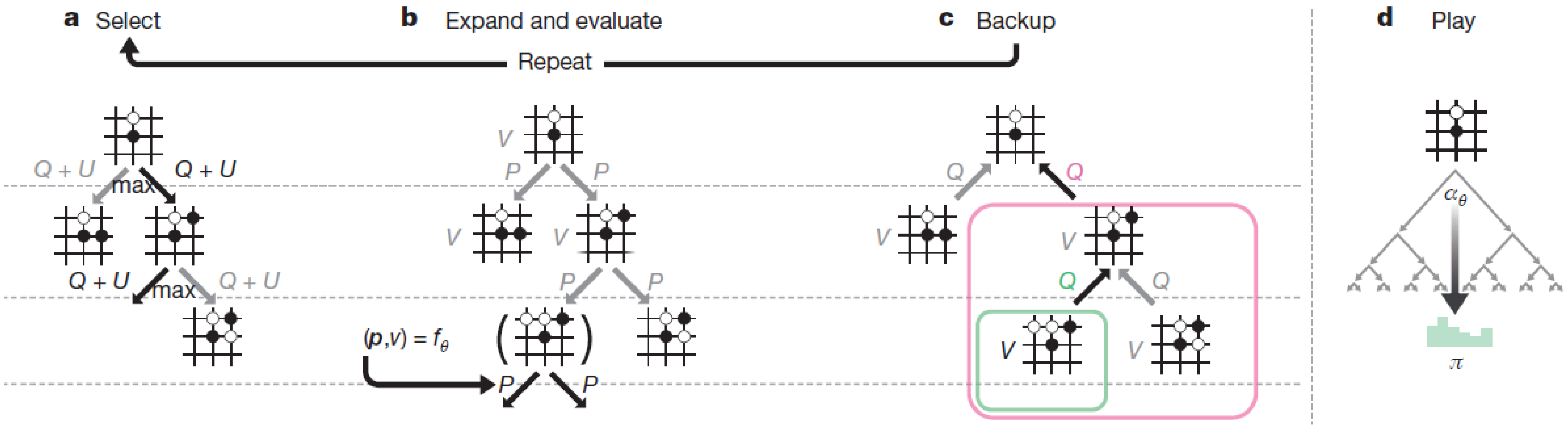


MCTS

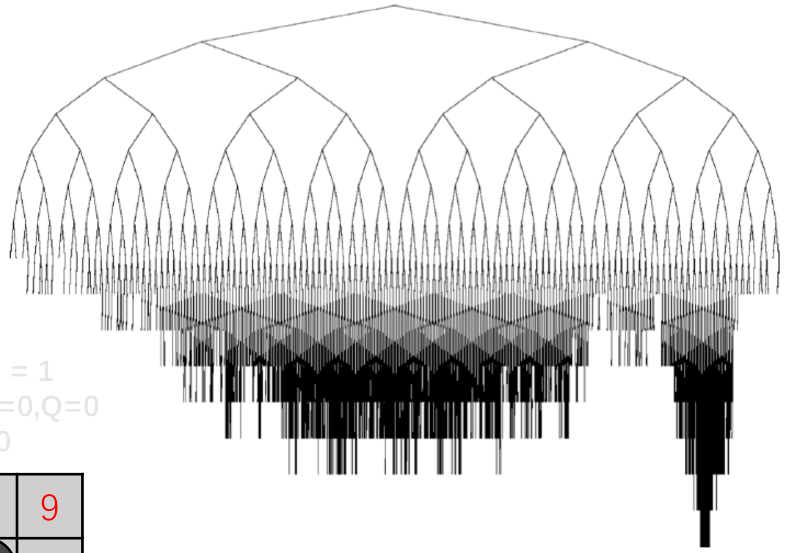


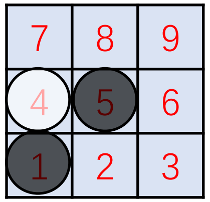
Real board state

Done !

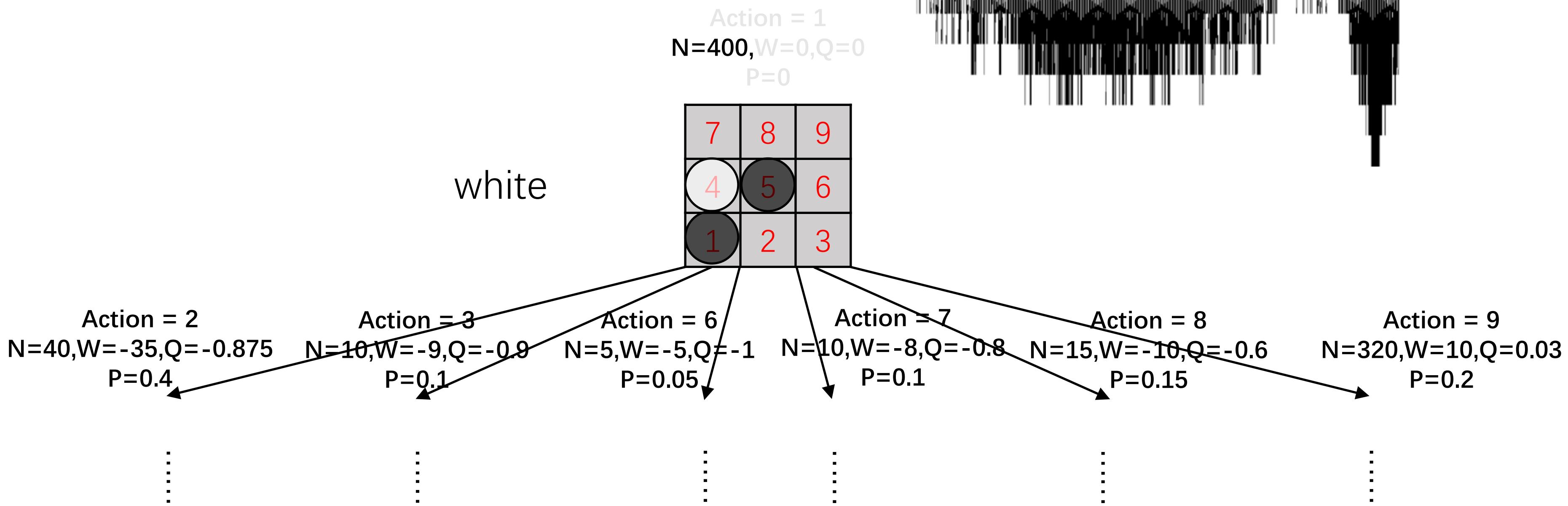


- Suppose we have repeated 400 times





Real board state

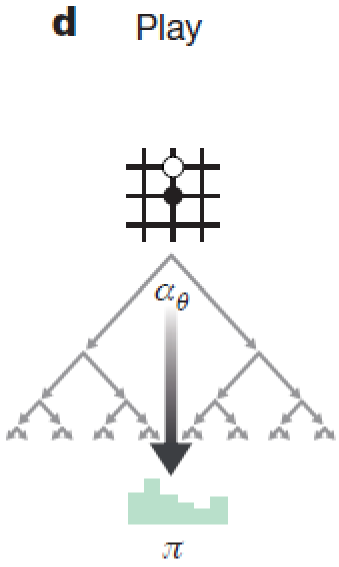


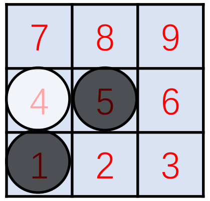
Play

- The move is chosen by calculating the probability

$$\pi(a|s) = \frac{N(s,a)^{1/\tau}}{\sum_b N(s,b)^{1/\tau}},$$

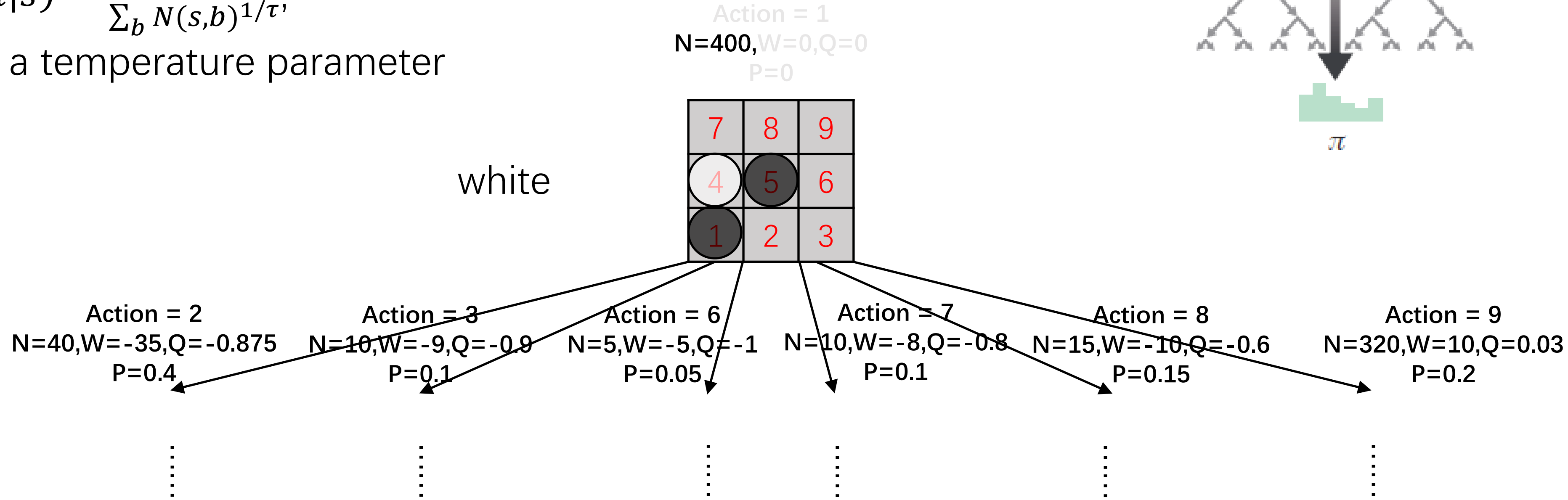
τ is a temperature parameter



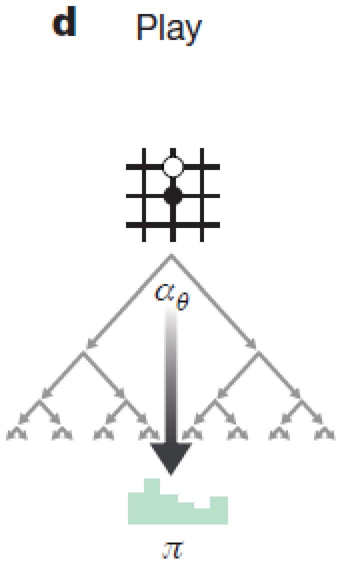


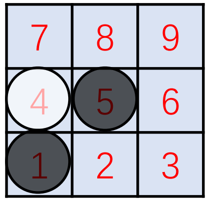
Real board state

white



Play





Real board state

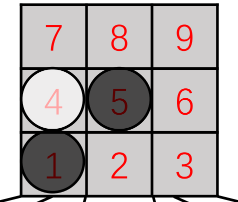
- The move is chosen by calculating the probability

$$\pi(a|s) = \frac{N(s,a)^{1/\tau}}{\sum_b N(s,b)^{1/\tau}},$$

τ is a temperature parameter

Action = 1
N=400, W=0,Q=0
P=0

white



Action = 2
N=40,W=-35,Q=-0.875
P=0.4

Action = 3
N=10,W=-9,Q=-0.9
P=0.1

Action = 6
N=5,W=-5,Q=-1
P=0.05

Action = 7
N=10,W=-8,Q=-0.8
P=0.1

Action = 8
N=15,W=-10,Q=-0.6
P=0.15

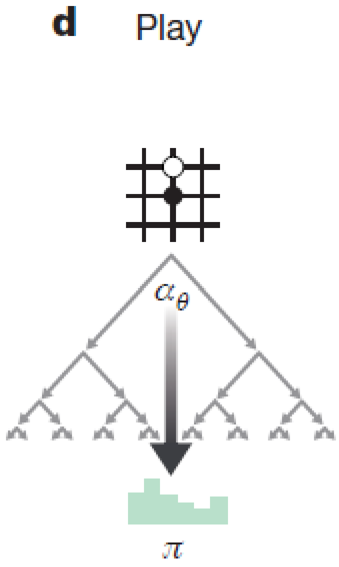
Action = 9
N=320,W=10,Q=0.03
P=0.2

⋮

If we set $\tau = 1$, then

$\pi(2|s)$ $\frac{40}{400}$ $\pi(3|s)$ $\frac{10}{400}$ $\pi(6|s)$ $\frac{5}{400}$ $\pi(7|s)$ $\frac{10}{400}$ $\pi(8|s)$ $\frac{15}{400}$ $\pi(9|s)$ $\frac{320}{400}$

Play



7	8	9
4	5	6
1	2	3

Real board state

- The move is chosen by calculating the probability

$$\pi(a|s) = \frac{N(s,a)^{1/\tau}}{\sum_b N(s,b)^{1/\tau}},$$

τ is a temperature parameter

Action = 1
N=400, W=0,Q=0
P=0

white

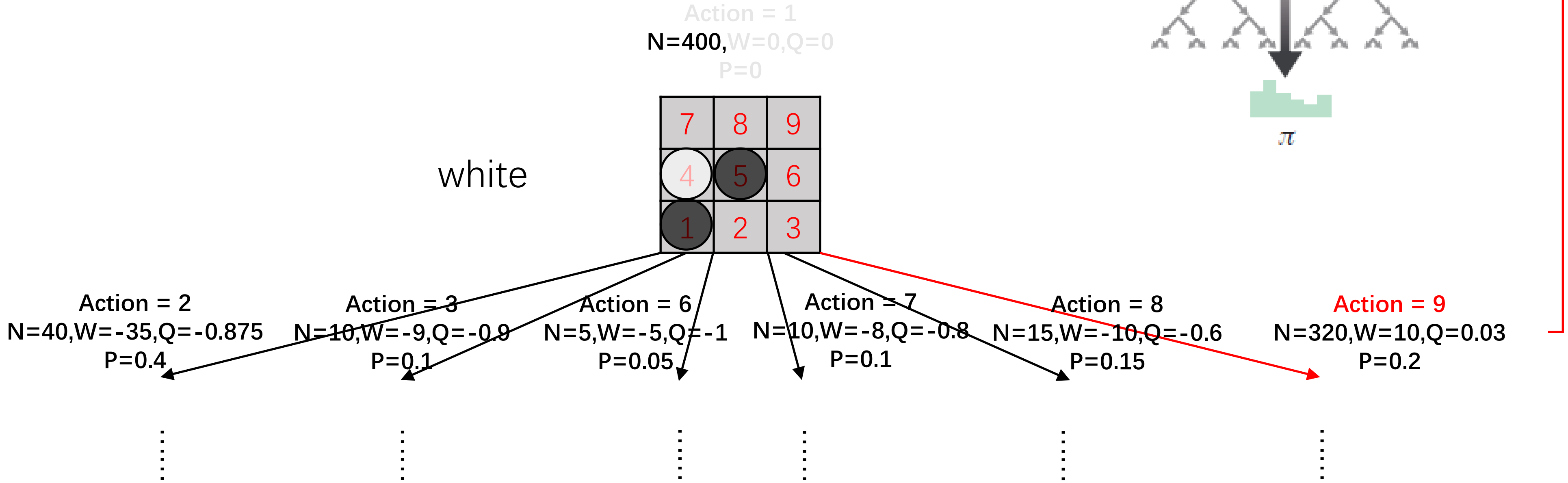
7	8	9
4	5	6
1	2	3

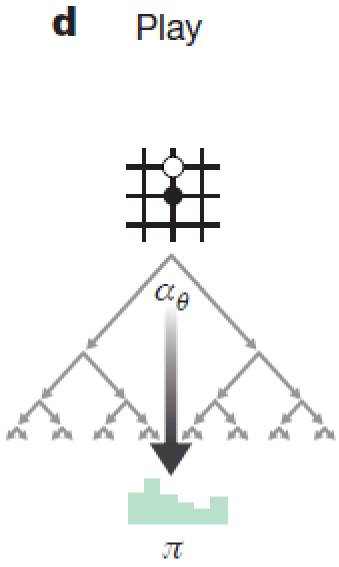
Action = 2	Action = 3	Action = 6	Action = 7	Action = 8	Action = 9
N=40,W=-35,Q=-0.875	N=10,W=-9,Q=-0.9	N=5,W=-5,Q=-1	N=10,W=-8,Q=-0.8	N=15,W=-10,Q=-0.6	N=320,W=10,Q=0.03
P=0.4	P=0.1	P=0.05	P=0.1	P=0.15	P=0.2
⋮	⋮	⋮	⋮	⋮	⋮
$\pi(2\|s)$	$\pi(3\|s)$	$\pi(6\|s)$	$\pi(7\|s)$	$\pi(8\|s)$	$\pi(9\|s)$
$\frac{40}{400}$	$\frac{10}{400}$	$\frac{5}{400}$	$\frac{10}{400}$	$\frac{15}{400}$	$\frac{320}{400}$

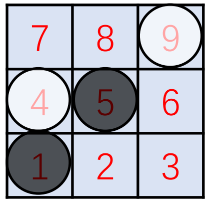
If we set $\tau = 1$, then

Self-play

Play



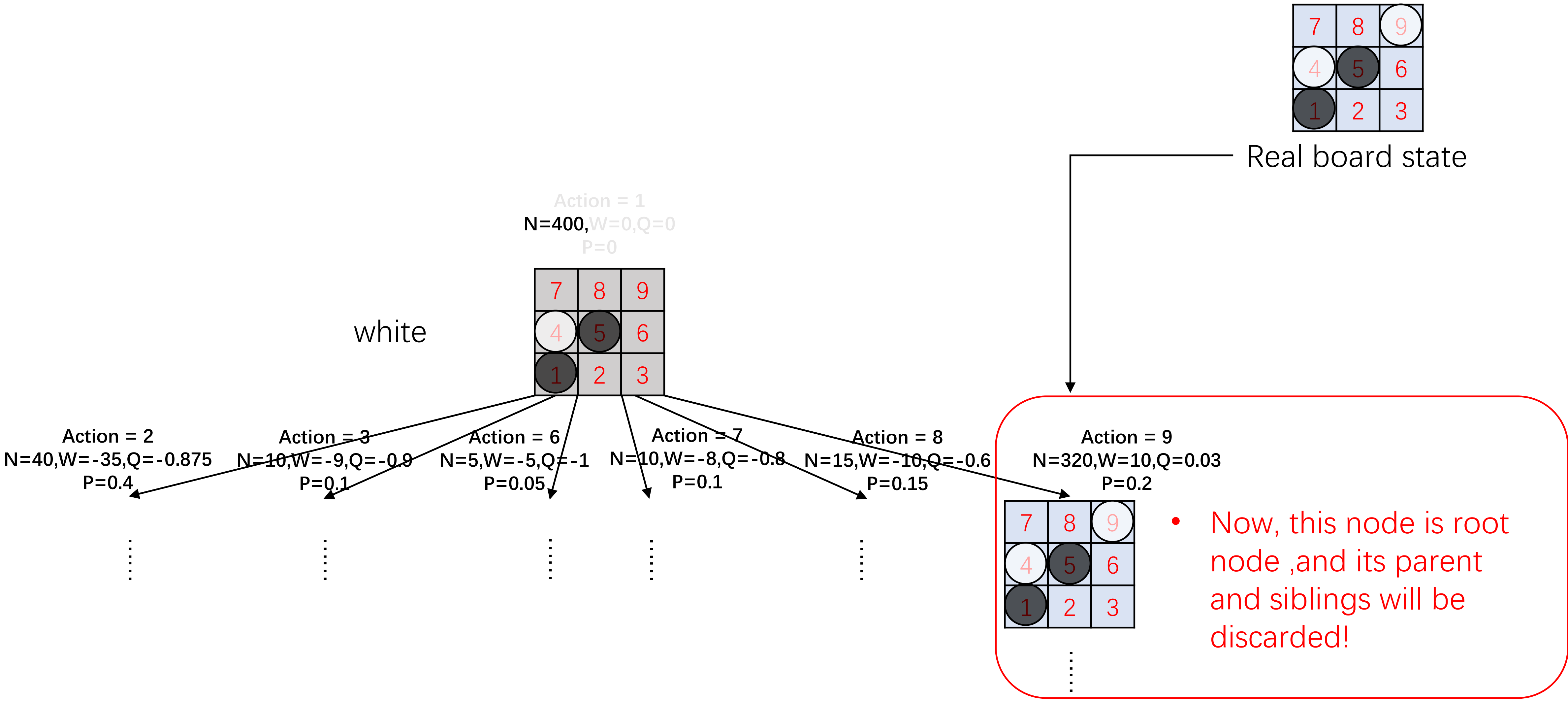




Real board state

Do move in real board!

Self-play



- Now, this node is root node ,and its parent and siblings will be discarded!

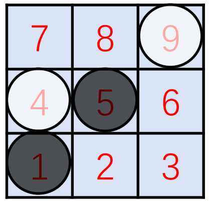
Self-play MCTS

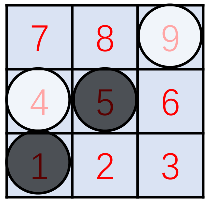
- Monte Carlo Tree Search in Gomoku

  - The tree search algorithm will start from here again!
  - And after every real move, the search algorithm will be repeated until game is over!

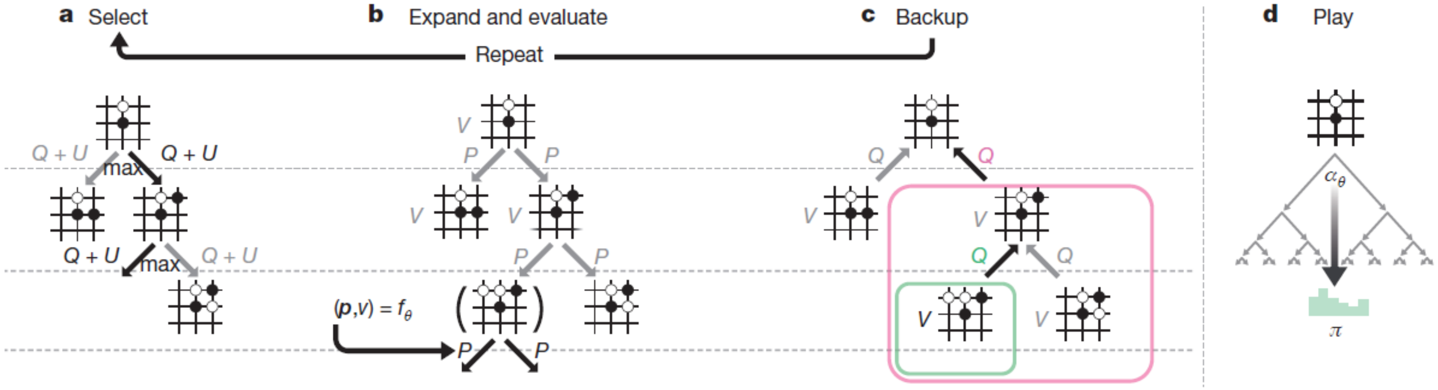
black

Action = 9
N=320,W=10,Q=0.03
P=0.2

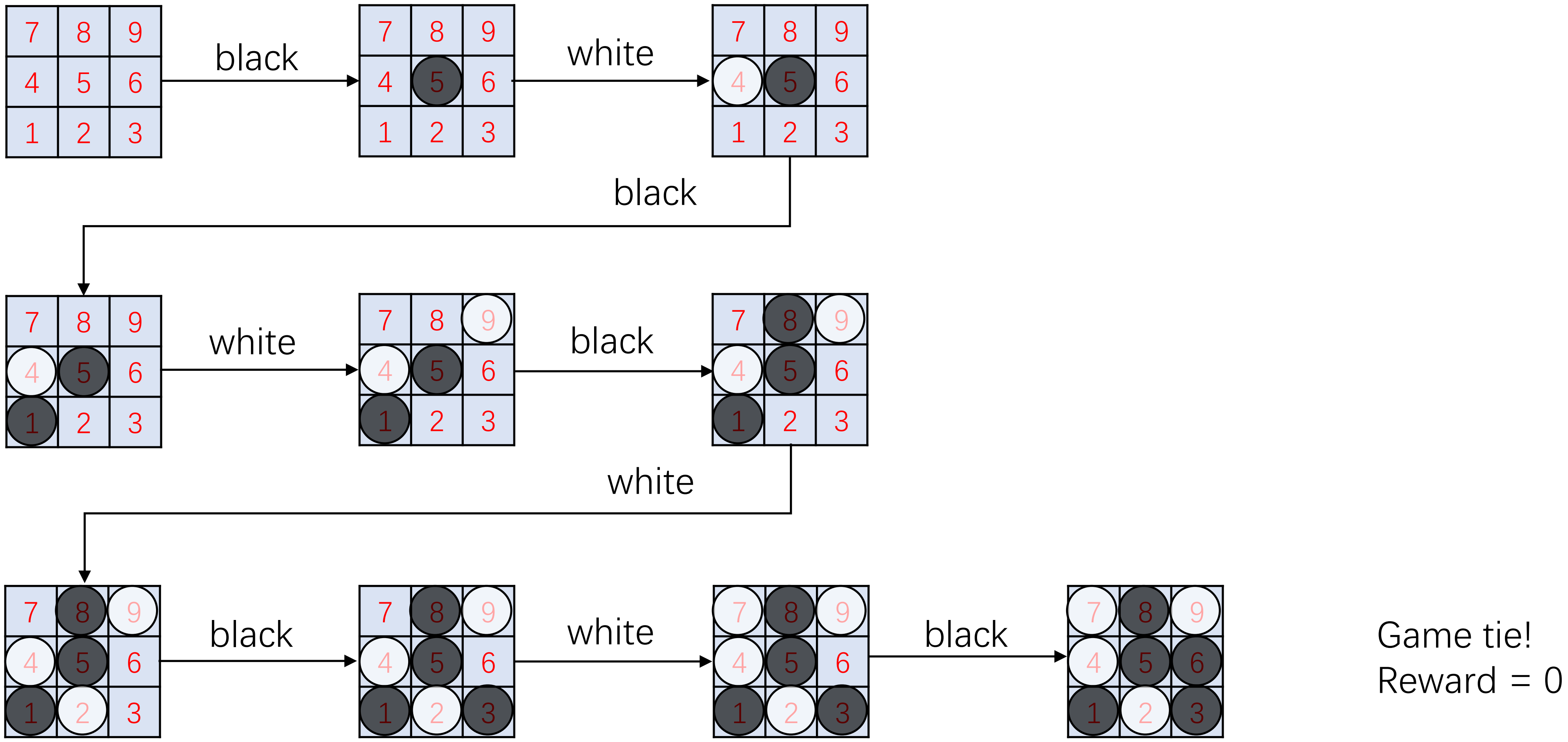




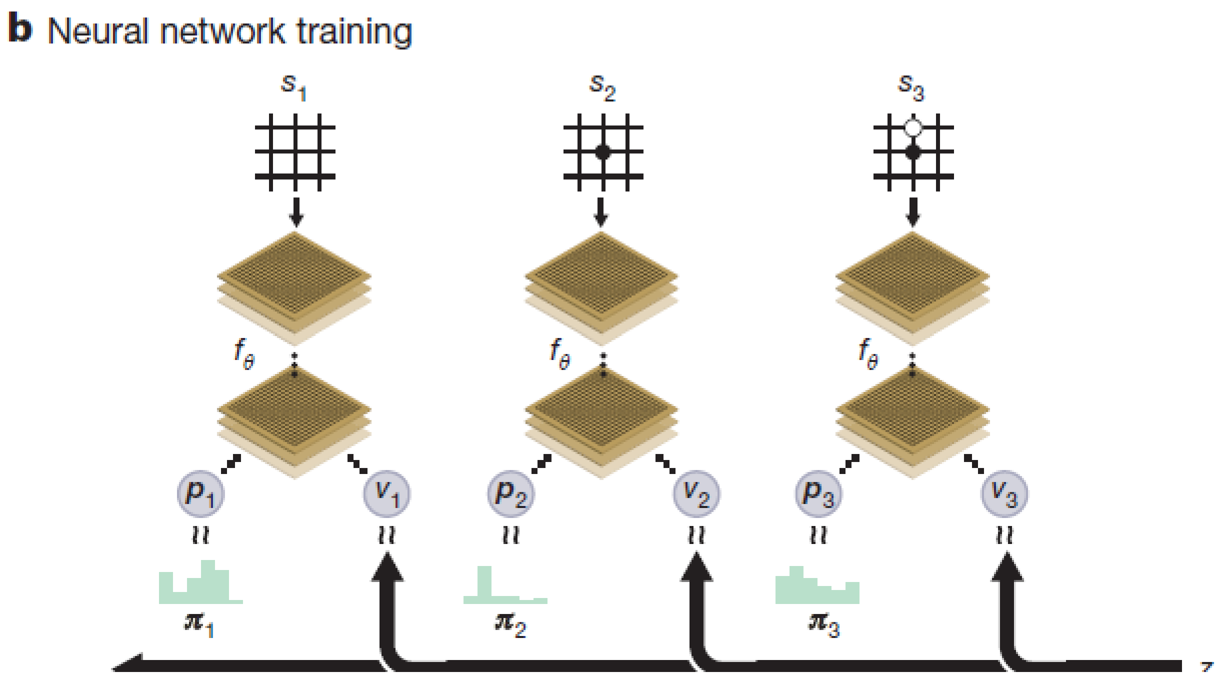
Real board state



- One game is done!

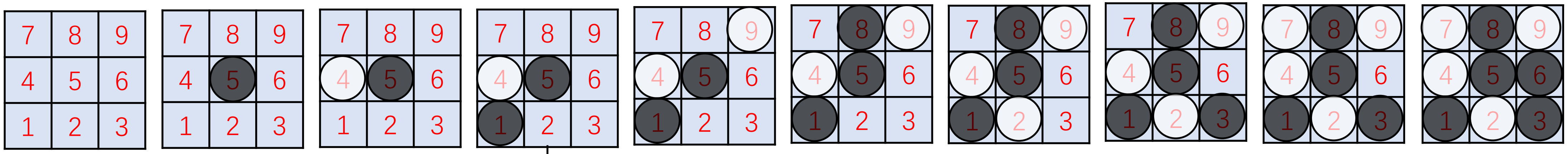


- After self-play for #C games,
  Neural network training starts.



Network training

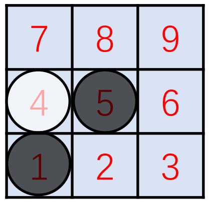
- We get the data : kifu



$$[(s_1, \pi(a|s_1) = \frac{N(s_1,a)^{\frac{1}{\tau}}}{\sum_b N(s_1,b)^{\frac{1}{\tau}}}, reward = 0),$$

$$\cdots\cdots,$$

$$(s_{10}, \pi(a|s_{10}) = \frac{N(s_{10},a)^{\frac{1}{\tau}}}{\sum_b N(s_{10},b)^{\frac{1}{\tau}}}, reward = 0)]$$

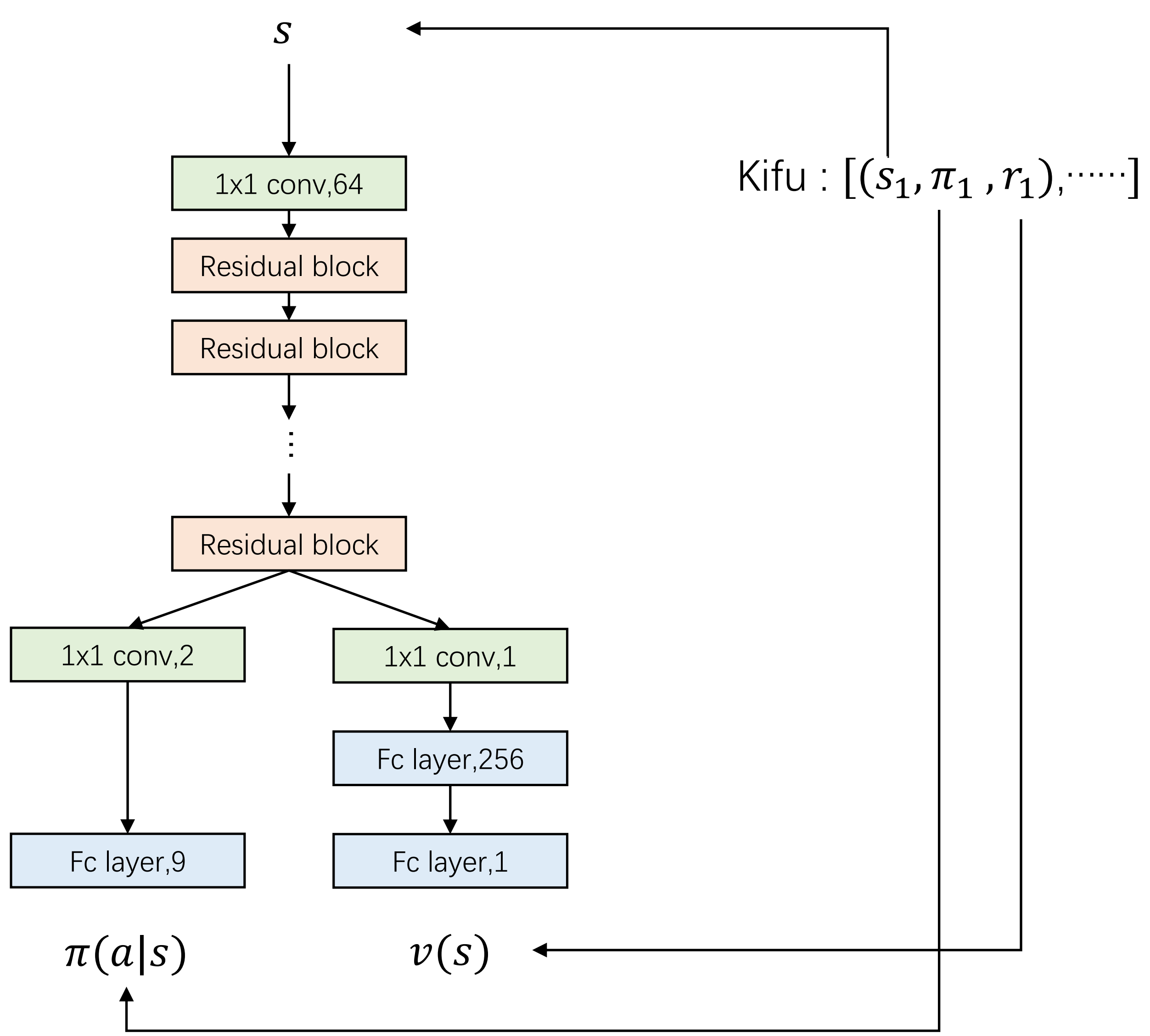
$s =$ 

$\pi(a|s) =$

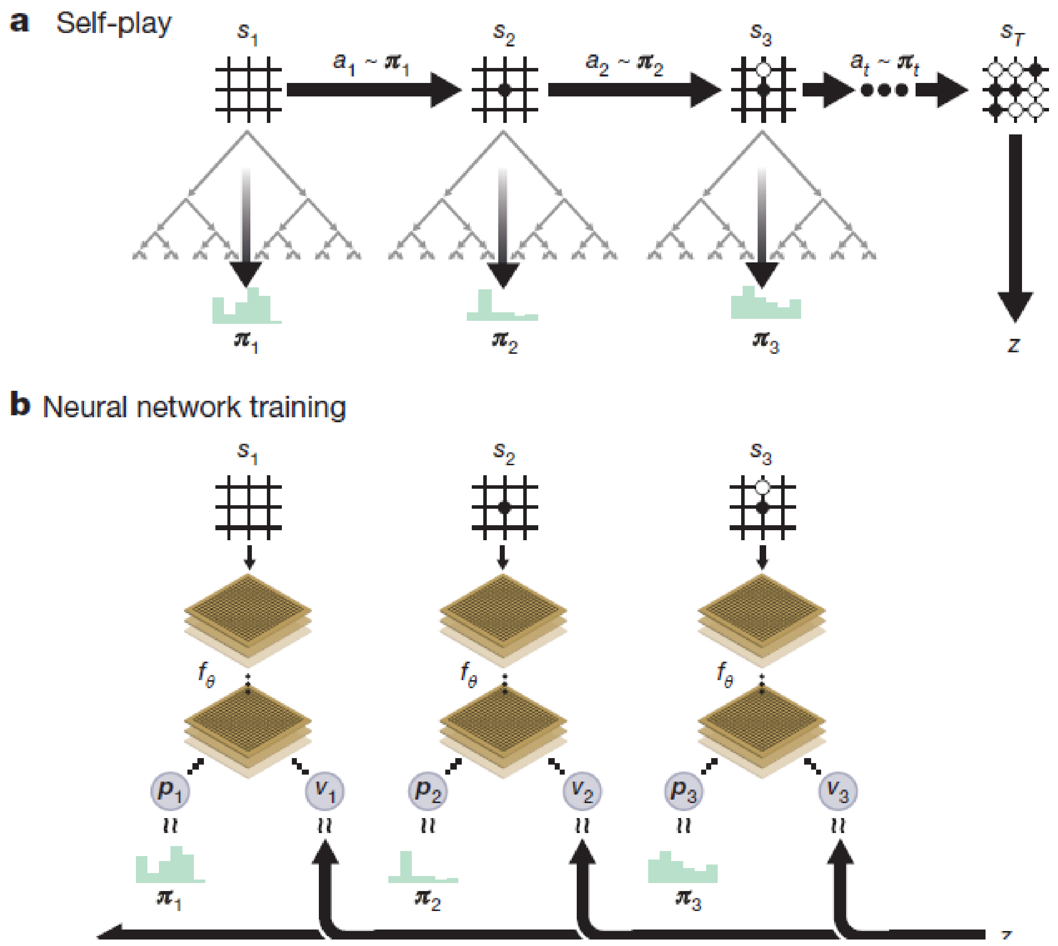
$\pi(1\|s)$	$\pi(2\|s)$	$\pi(3\|s)$	$\pi(4\|s)$	$\pi(5\|s)$	$\pi(6\|s)$	$\pi(7\|s)$	$\pi(8\|s)$	$\pi(9\|s)$
$\frac{0}{400}$	$\frac{40}{400}$	$\frac{10}{400}$	$\frac{0}{400}$	$\frac{0}{400}$	$\frac{5}{400}$	$\frac{10}{400}$	$\frac{15}{400}$	$\frac{320}{400}$

$reward = 0$

- Use the kifu to train the network!



- Repeat the whole process over and over
- The network can be more accurate and its strength will improve gradually!



Thanks !