Monte Carlo

2018年江苏省研究生“数据挖掘与人工智能”暑期学校

AlphaGo原理与实现

朱天驰、刘杰
www.mcarlo.com
蒙特卡洛科技™·致力于提供垂直行业人工智能解决方案

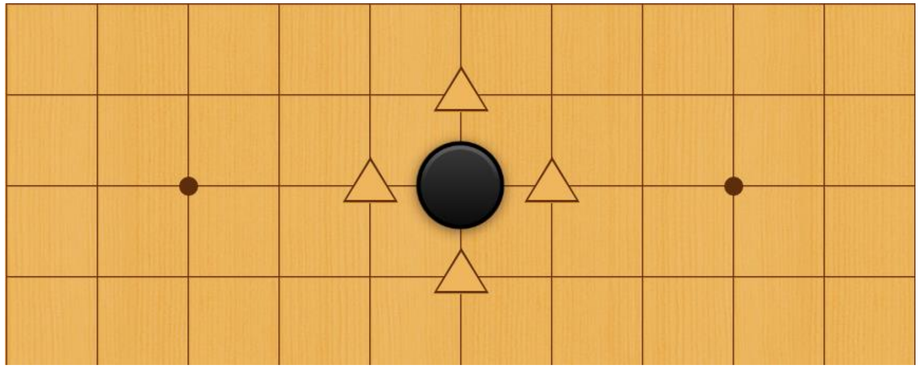
Monte Carlo
蒙　特　卡　洛

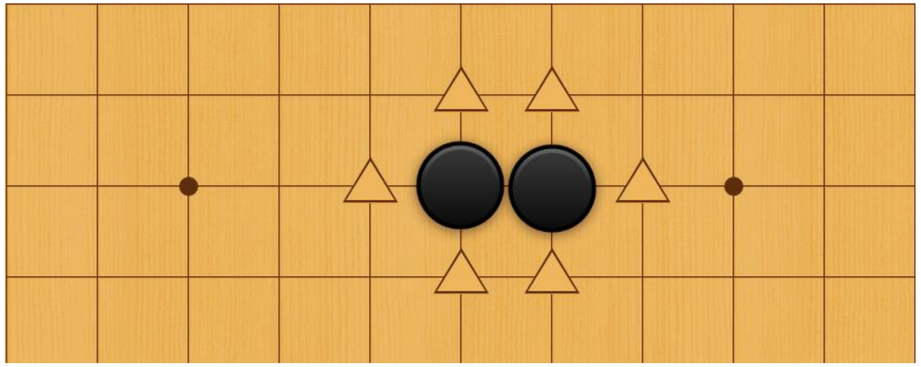
1

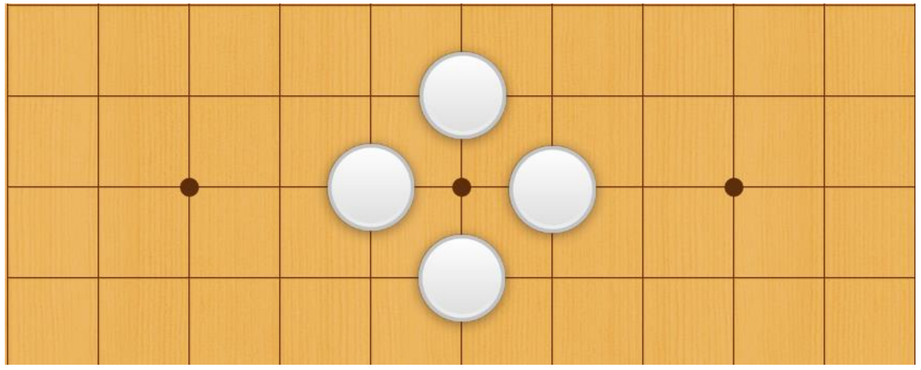
AlphaGo的定义

重要论文、名词来源、辉煌战绩

1. 2015-10 AlphaGo Fan
   2016-1 AlphaGo =《Mastering the game of Go with deep neural networks and tree search》
2. 2016-3 AlphaGo Lee
3. 2017-1 AlphaGo Master
   2017-10 **AlphaGo Zero** =《Mastering the Game of Go without Human Knowledge》
4. 2017-12 **AlphaZero** =《Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm》







Monte Carlo
蒙 特 卡 洛

2

规则

英文论文中的日文

子：stone

气：liberty

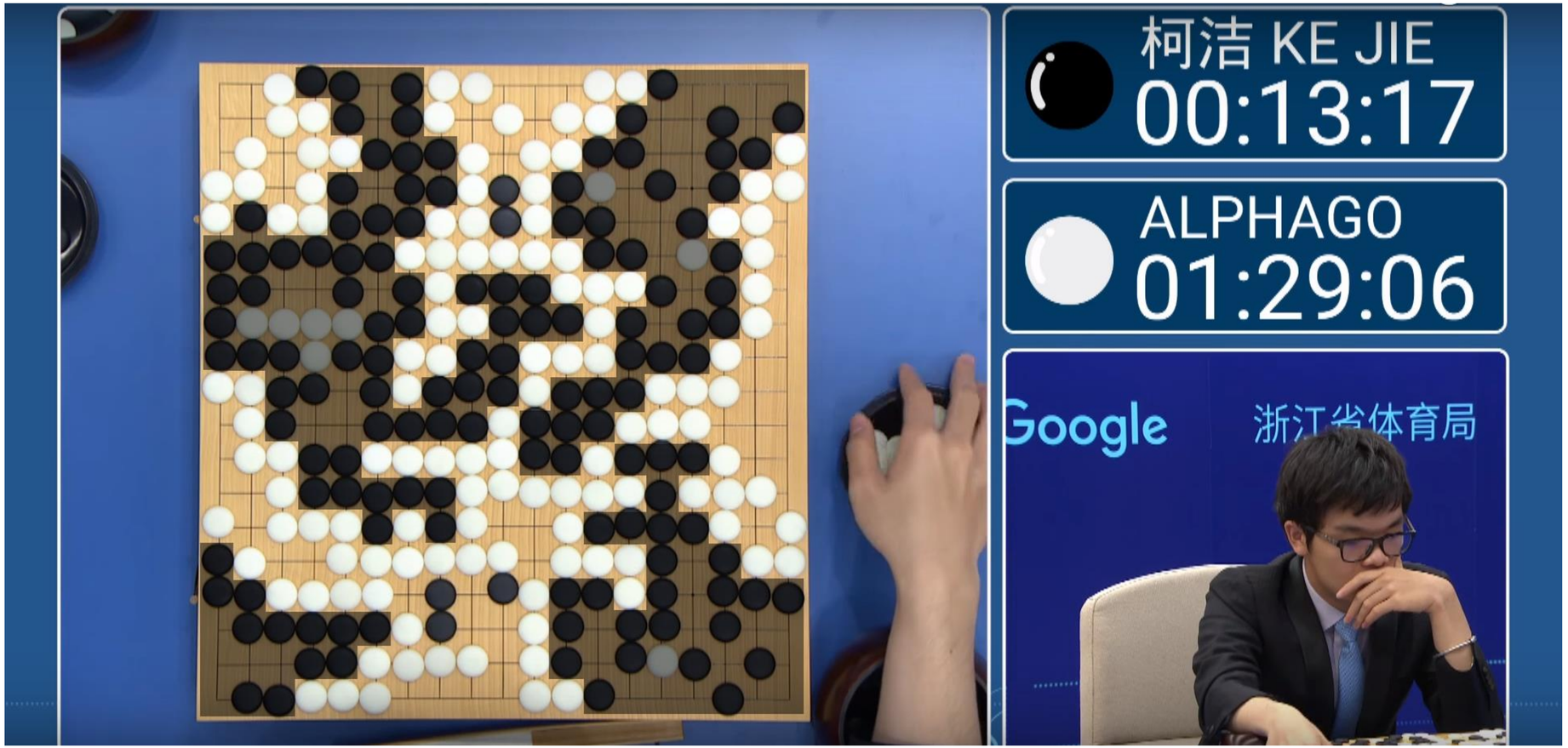
长：nobi

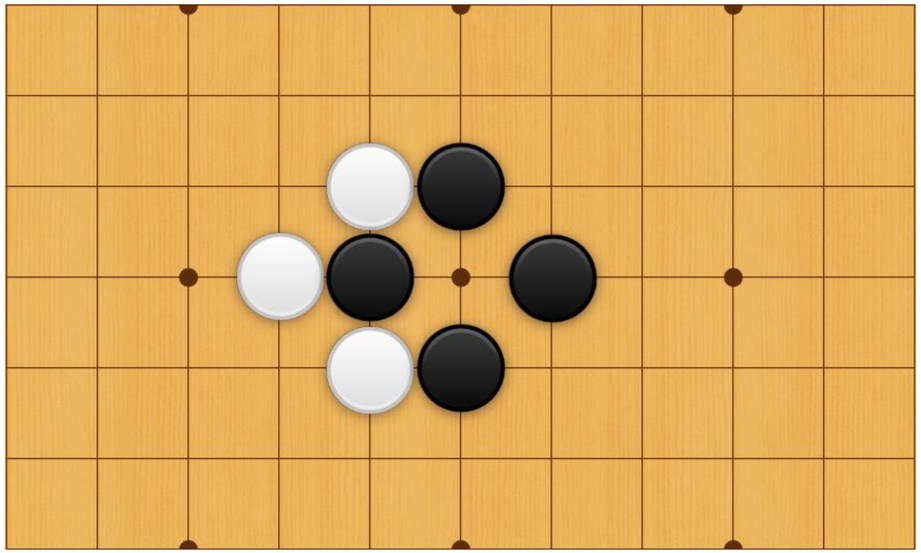
打吃/提：atari / ~~take~~

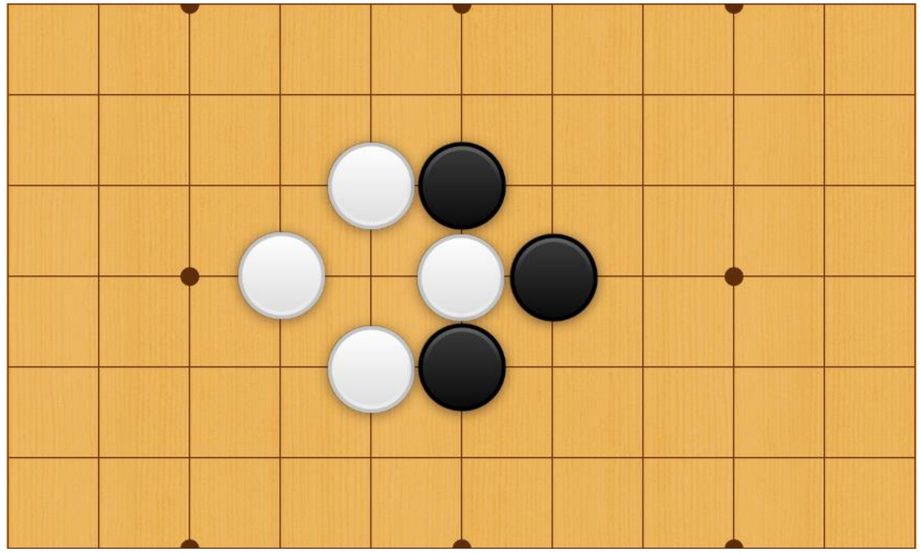
眼/空：~~eye~~ / territory

2 规则

黑184子
需要$\frac{361}{2}+\frac{7.5}{2}=184.25$
先手/贴目：~~sente~~ / komi







2

规则

英文论文中的日文

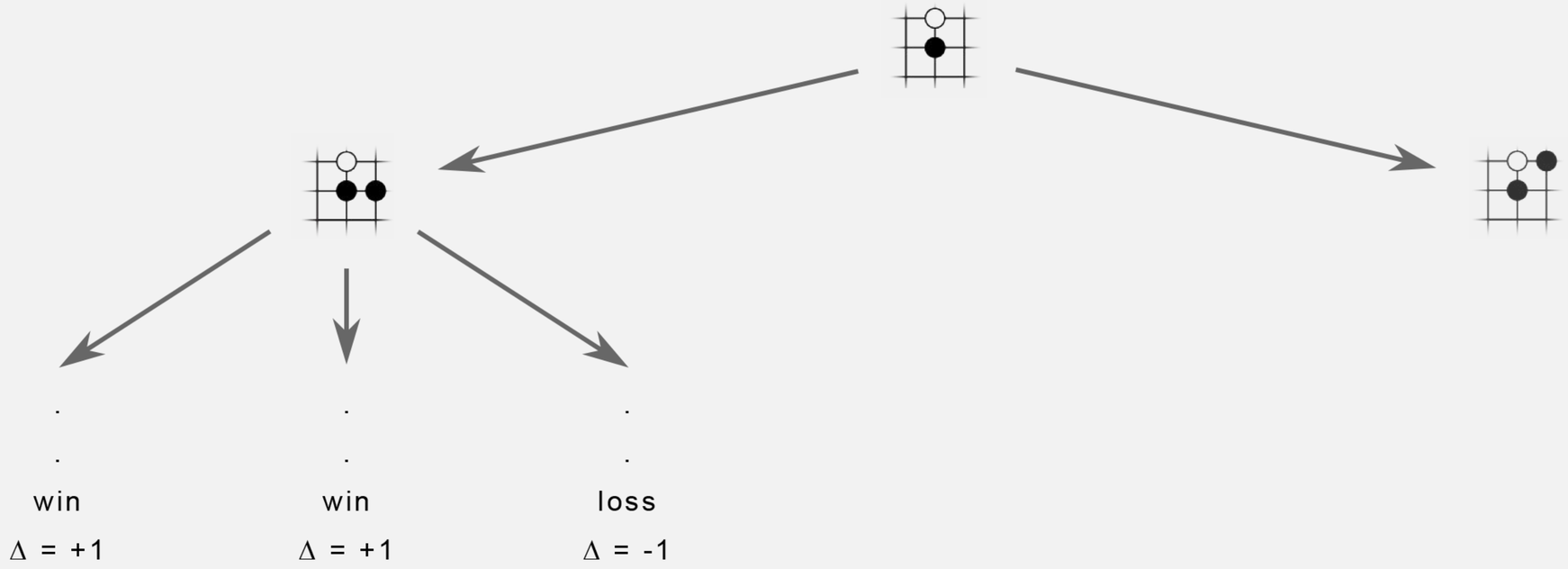
劫：ko

定式：joseki

Monte Carlo
蒙 特 卡 洛

一个朴素的想法

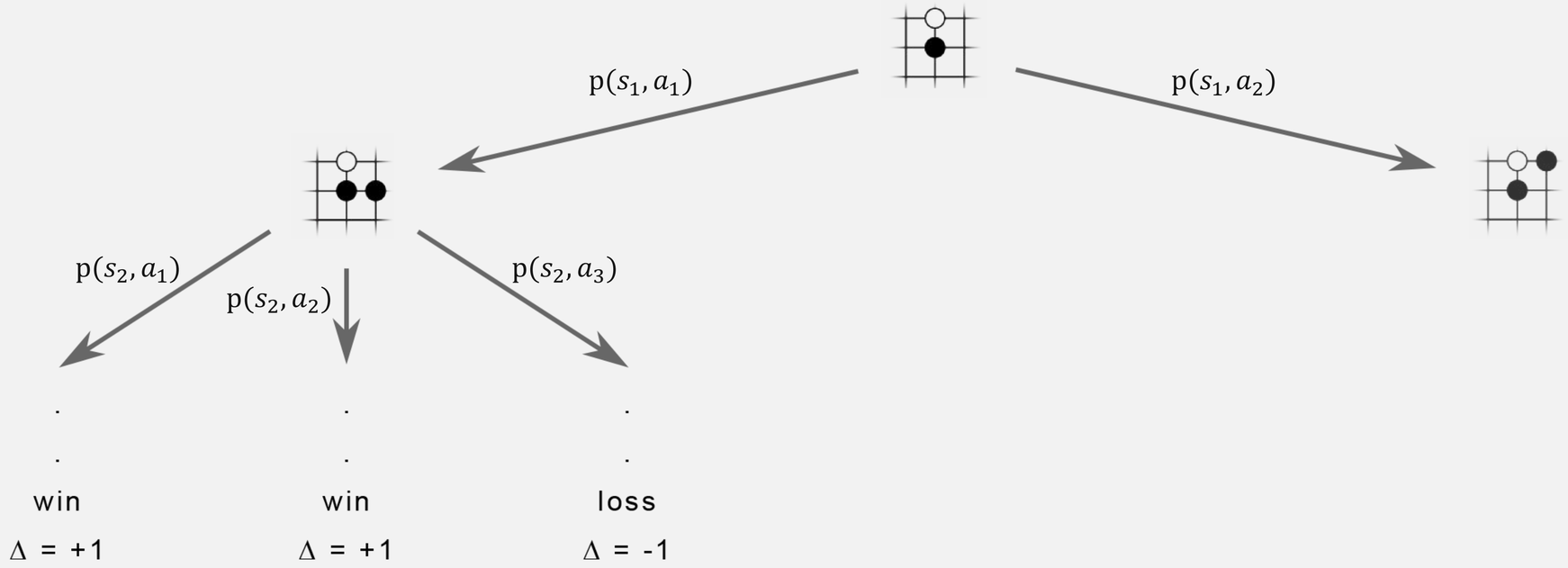
10^{170}次广度优先遍历不可能。深度优先：探索 $N_{simulation}$次，每次从root开始选择$a \sim DefaultPolicy$直到term。整条路径上记录 $\mathrm{N}(s_t, a_t) \leftarrow \mathrm{N}(s_t, a_t) + 1$，$\mathrm{W}(s_t, a_t) \leftarrow \mathrm{W}(s_t, a_t) + \Delta$，最后$return = \underset{a}{\mathrm{argmax}} \frac{W(s_{root}, a)}{N(s_{root}, a)}$。



Monte Carlo
蒙 特 卡 洛

改进1：剪枝

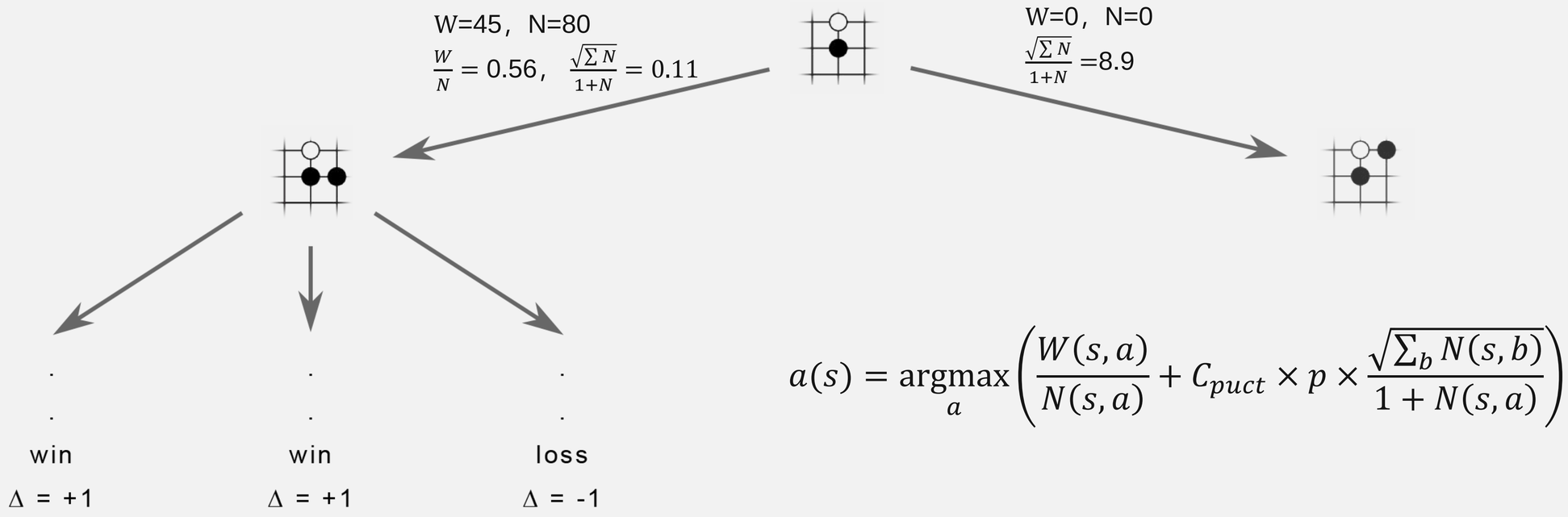
获得 $(s,a) \rightarrow p(s,a)$ 的映射，在每个盘面上，对所有合法a按照归一化概率 $\frac{p(s,a)}{\sum_b p(s,b)}$ 向下探索，一共探索 $N_{simulation}$次。p高时重复推演，p低时几乎不推演。



Monte Carlo
蒙 特 卡 洛

改进2：减少浪费/防止错判

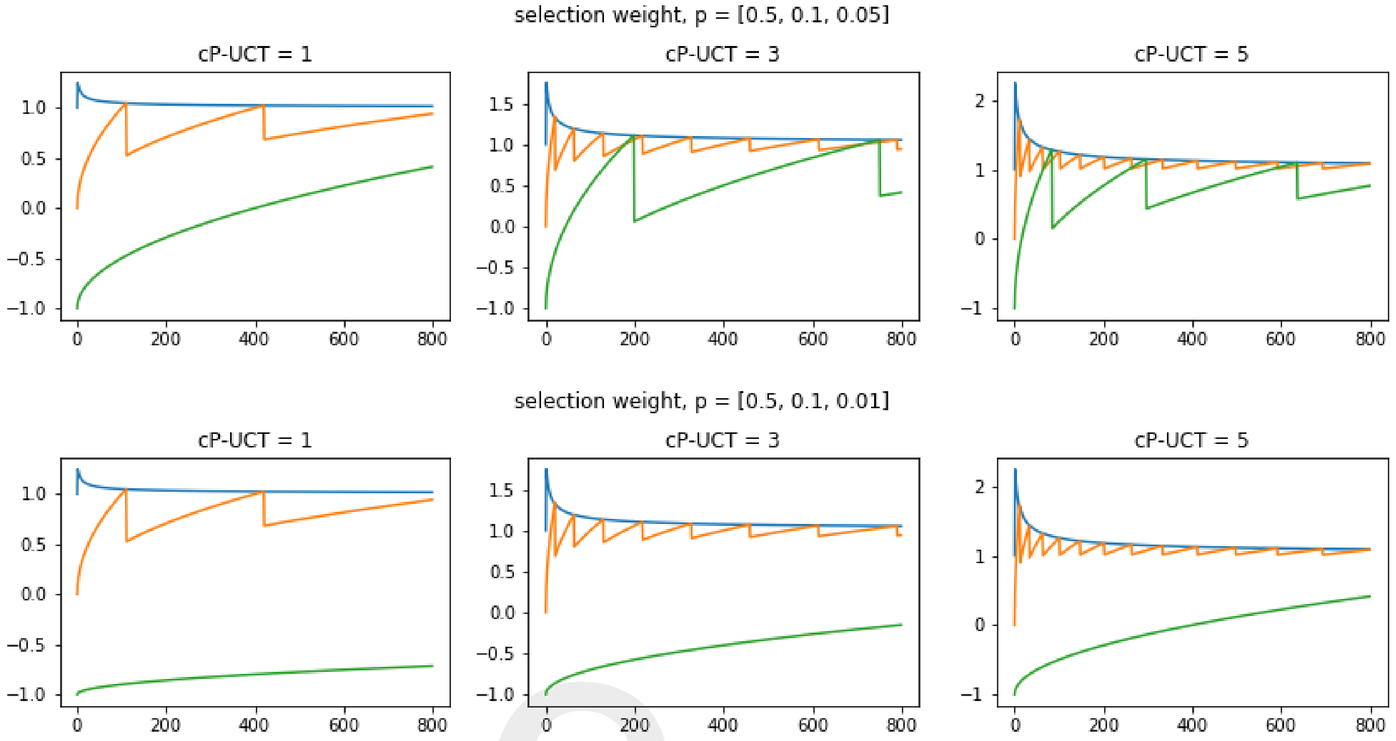
UCB = Upper Confidence Bounds，探索次数越多越“自信”，就越不需要再探索。



$$a(s) = \underset{a}{\operatorname{argmax}}\left(\frac{W(s,a)}{N(s,a)} + C_{puct} \times p \times \frac{\sqrt{\sum_b N(s,b)}}{1+N(s,a)}\right)$$

改进2：减少浪费/防止错判

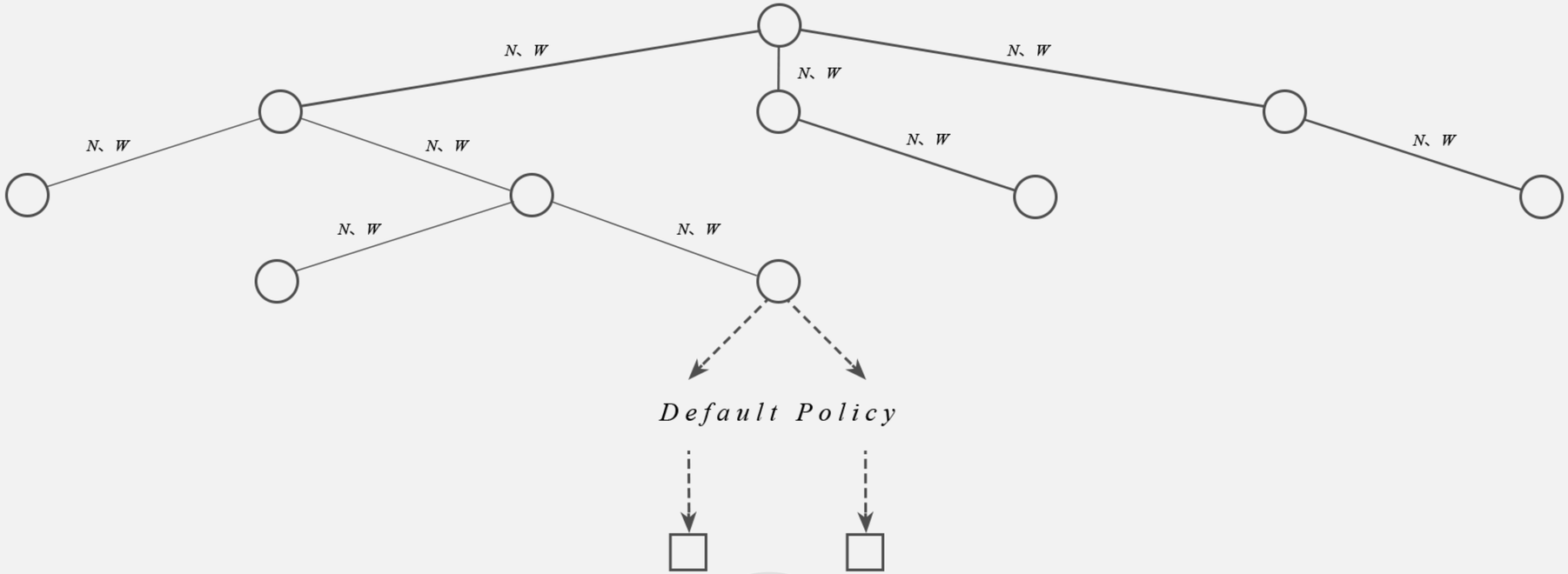
$$a(s) = \underset{a}{\operatorname{argmax}}\left(\frac{W(s,a)}{N(s,a)} + C_{puct} \times p \times \frac{\sqrt{\sum_b N(s,b)}}{1+N(s,a)}\right)$$



Monte Carlo
蒙 特 卡 洛

改进3：减少只visit过一次的node

区分tree policy和default policy。其中tree部分每次只增长一层（记录N、W），其余部分只记录结果，不记录N、W。



Monte Carlo
蒙 特 卡 洛

MCTS = Monte Carlo Tree Search

function UCTSEARCH(s_0)
  create root node v_0 with state s_0
  while within computational budget **do**
    $v_l \leftarrow$ TREEPOLICY(v_0)
    $\Delta \leftarrow$ EVALUATE($s(v_l)$)
    BACKUP(v_l, Δ)
  return a(BESTCHILD($v_0, 0$))

可rollout到term ➔

function TREEPOLICY(v)
  while v is nonterminal **do**
    if v not expanded **then**
      return EXPAND(v)
    else
      $v \leftarrow$ BESTCHILD(v, Cp)
  return v

选leaf有2种情况 ➔

function BESTCHILD(v, c)

$$\textbf{return} \quad \underset{v' \in \text{children of } v}{\arg\max} \frac{W(v')}{N(v')} + c\,p\,\frac{\sqrt{N(v)}}{1+N(v')}$$

function EXPAND(v)
  foreach $a \in A(s(v))$ **do**
    add a new child v' to v
      with $s(v') = f(s(v), a)$
        $a(v') = a$
        $N(s(v), a) = 0$
        $W(s(v), a) = 0$
        $p(s(v), a) = p$
  return v

function BACKUP (v, Δ)
  while v is not null **do**
    $N(v) \leftarrow N(v) + 1$
    $W(v) \leftarrow W(v) + \Delta$
    $\Delta \leftarrow -\Delta$
    $v \leftarrow$ parent of v

← 如果黑win，则白loss

MCTS = Monte Carlo Tree Search

function UCTSEARCH(s_0)
  create root node v_0 with state s_0
  while within computational budget **do**
    $v_l \leftarrow$ TREEPOLICY(v_0)
    $\Delta \leftarrow$ EVALUATE($s(v_l)$)
    BACKUP(v_l, Δ)
  return a(BESTCHILD($v_0, 0$))

可rollout到term ➔

function TREEPOLICY(v)
  while v is nonterminal **do**
    if v not expanded **then**
      return EXPAND(v)
    else
      $v \leftarrow$ BESTCHILD(v, Cp)
  return v

选leaf有2种情况 ➔

function BESTCHILD(v, c)

$$\textbf{return} \quad \underset{v' \in \text{children of } v}{\arg\max} \frac{W(v')}{N(v')} + c\,p\,\frac{\sqrt{N(v)}}{1+N(v')}$$

function EXPAND(v)
  foreach $a \in A(s(v))$ **do**
    add a new child v' to v
      with $s(v') = f(s(v), a)$
        $a(v') = a$
        $N(s(v), a) = 0$
        $W(s(v), a) = 0$
        $p(s(v), a) = p$
  return v

function BACKUP (v, Δ)
  while v is not null **do**
    $N(v) \leftarrow N(v) + 1$
    $W(v) \leftarrow W(v) + \Delta$
    $\Delta \leftarrow -\Delta$
    $v \leftarrow$ parent of v

← 如果黑win，则白loss

Monte Carlo
蒙 特 卡 洛

MCTS = Monte Carlo Tree Search

function UCTSEARCH(s_0)
  create root node v_0 with state s_0
  while within computational budget **do**
    $v_l \leftarrow$ TREEPOLICY(v_0)
    $\Delta \leftarrow$ EVALUATE($s(v_l)$)
    BACKUP(v_l, Δ)
  return a(BESTCHILD($v_0, 0$))

可rollout到term ➔

function TREEPOLICY(v)
  while v is nonterminal **do**
    if v not expanded **then**
      return EXPAND(v)
    else
      $v \leftarrow$ BESTCHILD(v, Cp)
  return v

选leaf有2种情况 ➔

function BESTCHILD(v, c)

$$\textbf{return} \quad \underset{v' \in \text{children of } v}{\arg\max} \frac{W(v')}{N(v')} + c\,p\,\frac{\sqrt{N(v)}}{1+N(v')}$$

function EXPAND(v)
  foreach $a \in A(s(v))$ **do**
    add a new child v' to v
      with $s(v') = f(s(v), a)$
        $a(v') = a$
        $N(s(v), a) = 0$
        $W(s(v), a) = 0$
        $p(s(v), a) = p$
  return v

function BACKUP (v, Δ)
  while v is not null **do**
    $N(v) \leftarrow N(v) + 1$
    $W(v) \leftarrow W(v) + \Delta$
    $\Delta \leftarrow -\Delta$
    $v \leftarrow$ parent of v

← 如果黑win，则白loss

MCTS = Monte Carlo Tree Search

function UCTSEARCH(s_0)
  create root node v_0 with state s_0
  while within computational budget **do**
    $v_l \leftarrow$ TREEPOLICY(v_0)
    $\Delta \leftarrow$ EVALUATE($s(v_l)$)
    BACKUP(v_l, Δ)
  return a(BESTCHILD($v_0, 0$))

可rollout到term ➔

function TREEPOLICY(v)
  while v is nonterminal **do**
    if v not expanded **then**
      return EXPAND(v)
    else
      $v \leftarrow$ BESTCHILD(v, Cp)
  return v

选leaf有2种情况 ➔

function BESTCHILD(v, c)
  return $\underset{v' \in \text{children of } v}{\arg\max} \frac{W(v')}{N(v')} + c\,p\,\frac{\sqrt{N(v)}}{1+N(v')}$

function EXPAND(v)
  foreach $a \in A(s(v))$ **do**
    add a new child v' to v
      with $s(v') = f(s(v), a)$
        $a(v') = a$
        $N(s(v), a) = 0$
        $W(s(v), a) = 0$
        $p(s(v), a) = p$
  return v

function BACKUP (v, Δ)
  while v is not null **do**
    $N(v) \leftarrow N(v) + 1$
    $W(v) \leftarrow W(v) + \Delta$
    $\Delta \leftarrow -\Delta$
    $v \leftarrow$ parent of v

← 如果黑win，则白loss

MCTS = Monte Carlo Tree Search

function UCTSEARCH(s_0)
  create root node v_0 with state s_0
  while within computational budget **do**
    $v_l \leftarrow$ TREEPOLICY(v_0)
    $\Delta \leftarrow$ EVALUATE($s(v_l)$)
    BACKUP(v_l, Δ)
  return a(BESTCHILD($v_0, 0$))

可rollout到term ➔

function TREEPOLICY(v)
  while v is nonterminal **do**
    if v not expanded **then**
      return EXPAND(v)
    else
      $v \leftarrow$ BESTCHILD(v, Cp)
  return v

选leaf有2种情况 ➔

function BESTCHILD(v, c)
  return $\underset{v' \in \text{children of } v}{\arg\max} \frac{W(v')}{N(v')} + c\,p\,\frac{\sqrt{N(v)}}{1+N(v')}$

function EXPAND(v)
  foreach $a \in A(s(v))$ **do**
    add a new child v' to v
      with $s(v') = f(s(v), a)$
        $a(v') = a$
        $N(s(v), a) = 0$
        $W(s(v), a) = 0$
        $p(s(v), a) = p$
  return v

function BACKUP (v, Δ)
  while v is not null **do**
    $N(v) \leftarrow N(v) + 1$
    $W(v) \leftarrow W(v) + \Delta$
    $\Delta \leftarrow -\Delta$
    $v \leftarrow$ parent of v

← 如果黑win，则白loss

Monte Carlo
蒙 特 卡 洛

MCTS = Monte Carlo Tree Search

function UCTSEARCH(s_0)
  create root node v_0 with state s_0
  while within computational budget **do**
    $v_l \leftarrow$ TREEPOLICY(v_0)
    $\Delta \leftarrow$ EVALUATE($s(v_l)$)
    BACKUP(v_l, Δ)
  return a(BESTCHILD($v_0, 0$))

可rollout到term ➔

function TREEPOLICY(v)
  while v is nonterminal **do**
    if v not expanded **then**
      return EXPAND(v)
    else
      $v \leftarrow$ BESTCHILD(v, Cp)
  return v

选leaf有2种情况 ➔

function BESTCHILD(v, c)

$$\textbf{return} \quad \underset{v' \in \text{children of } v}{\arg\max} \frac{W(v')}{N(v')} + c\,p\,\frac{\sqrt{N(v)}}{1+N(v')}$$

function EXPAND(v)
  foreach $a \in A(s(v))$ **do**
    add a new child v' to v
      with $s(v') = f(s(v), a)$
        $a(v') = a$
        $N(s(v), a) = 0$
        $W(s(v), a) = 0$
        $p(s(v), a) = p$
  return v

function BACKUP (v, Δ)
  while v is not null **do**
    $N(v) \leftarrow N(v) + 1$
    $W(v) \leftarrow W(v) + \Delta$
    $\Delta \leftarrow -\Delta$
    $v \leftarrow$ parent of v

← 如果黑win，则白loss

MCTS = Monte Carlo Tree Search

function UCTSEARCH(s_0)
  create root node v_0 with state s_0
  while within computational budget **do**
    $v_l \leftarrow$ TREEPOLICY(v_0)
    $\Delta \leftarrow$ EVALUATE($s(v_l)$)
    BACKUP(v_l, Δ)
  return a(BESTCHILD($v_0, 0$))

可rollout到term ➔

function TREEPOLICY(v)
  while v is nonterminal **do**
    if v not expanded **then**
      return EXPAND(v)
    else
      $v \leftarrow$ BESTCHILD(v, Cp)
  return v

选leaf有2种情况 ➔

function BESTCHILD(v, c)
  return $\underset{v' \in \text{children of } v}{\arg\max} \frac{W(v')}{N(v')} + c\,p\,\frac{\sqrt{N(v)}}{1+N(v')}$

function EXPAND(v)
  foreach $a \in A(s(v))$ **do**
    add a new child v' to v
      with $s(v') = f(s(v), a)$
        $a(v') = a$
        $N(s(v), a) = 0$
        $W(s(v), a) = 0$
        $p(s(v), a) = p$
  return v

function BACKUP (v, Δ)
  while v is not null **do**
    $N(v) \leftarrow N(v) + 1$
    $W(v) \leftarrow W(v) + \Delta$
    $\Delta \leftarrow -\Delta$
    $v \leftarrow$ parent of v

← 如果黑win，则白loss

Monte Carlo
蒙 特 卡 洛

怎样学习$p(s,a) \rightarrow p(s,a;\theta)$

function UCTSEARCH(s_0)
  create root node v_0 with state s_0
  while within computational budget **do**
    $v_l \leftarrow$ TREEPOLICY(v_0)
    $\Delta \leftarrow$ EVALUATE($s(v_l)$)
    BACKUP(v_l, Δ)
  return a(BESTCHILD($v_0, 0$))

function TREEPOLICY(v)
  while v is nonterminal **do**
    if v not expanded **then**
      return EXPAND(v)
    else
      $v \leftarrow$ BESTCHILD(v, Cp)
  return v

function BESTCHILD(v, c)
  return $\underset{v' \in \text{children of } v}{\arg\max} \frac{W(v')}{N(v')} + c\,p\sqrt{\frac{N(v)}{1+N(v')}}$

function EXPAND(v)
  foreach $a \in A(s(v))$ **do**
    add a new child v' to v
      with $s(v') = f(s(v), a)$
        $a(v') = a$
        $N(s(v), a) = 0$
        $W(s(v), a) = 0$
        $p(s(v), a) = p$
  return v

function BACKUP(v, Δ)
  while v is not null **do**
    $N(v) \leftarrow N(v) + 1$
    $W(v) \leftarrow W(v) + \Delta$
    $\Delta \leftarrow -\Delta$
    $v \leftarrow$ parent of v

Monte Carlo
蒙 特 卡 洛

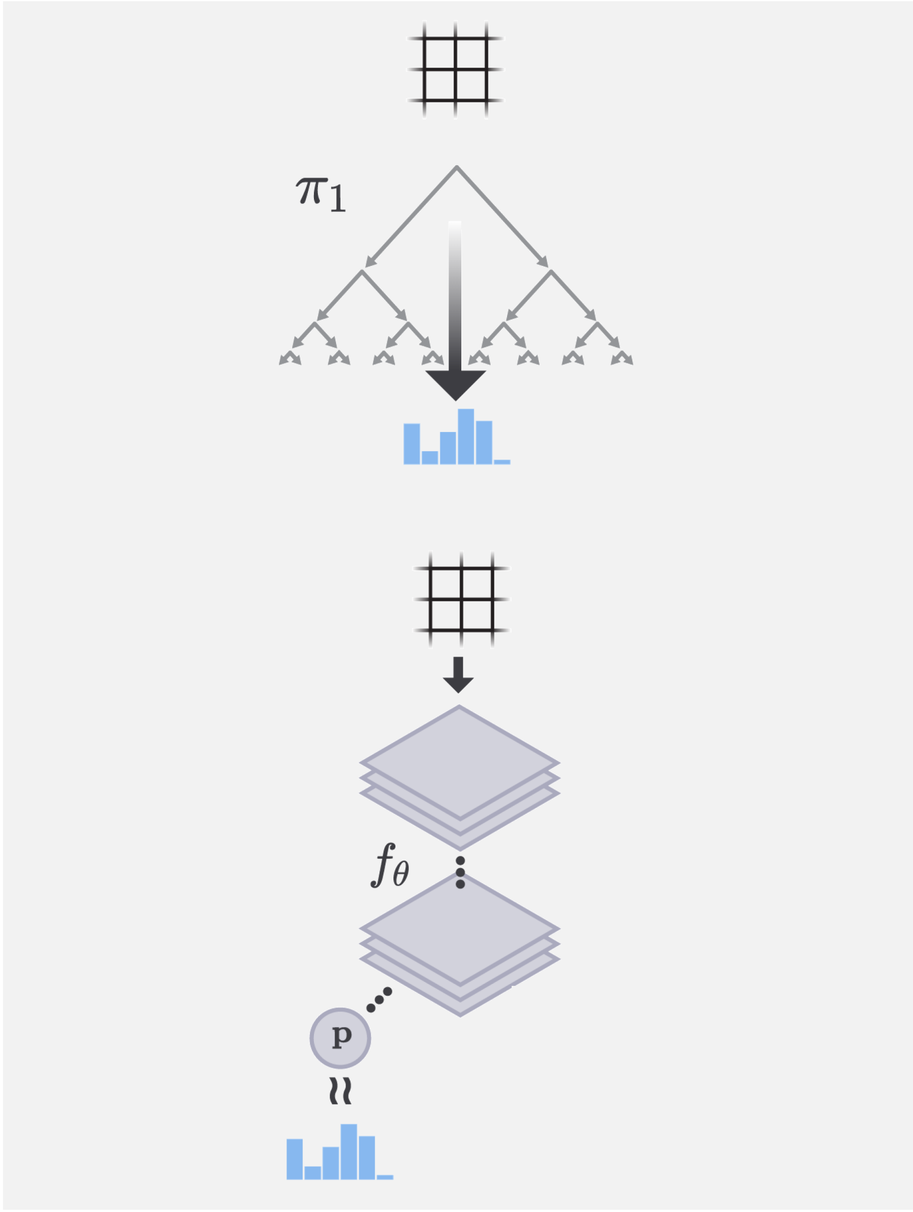
4

怎样学习
$p(s, a; \theta)$

Classification问题的变形

return $\underset{a}{\text{argmax}}\, \text{BESTCHILD}(v_0, 0)$

$$\mathrm{p}(a|s_{root}) \rightarrow \frac{N(s_{root}, a)^{1/\tau}}{\sum_b N(s_{root}, b)^{1/\tau}}$$



Evaluate是什么

function UCTSEARCH(s_0)
  create root node v_0 with state s_0
  while within computational budget **do**
    $v_l \leftarrow$ TREEPOLICY(v_0)
    $\Delta \leftarrow$ EVALUATE$(s(v_l))$
    BACKUP(v_l, Δ)
  return a(BESTCHILD($v_0, 0$))

function TREEPOLICY(v)
  while v is nonterminal **do**
    if v not expanded **then**
      return EXPAND(v)
    else
      $v \leftarrow$ BESTCHILD(v, Cp)
  return v

function BESTCHILD(v, c)
  return $\underset{v' \in \text{children of } v}{\arg\max} \frac{W(v')}{N(v')} + c\,P\,\frac{\sqrt{N(v)}}{1+N(v')}$

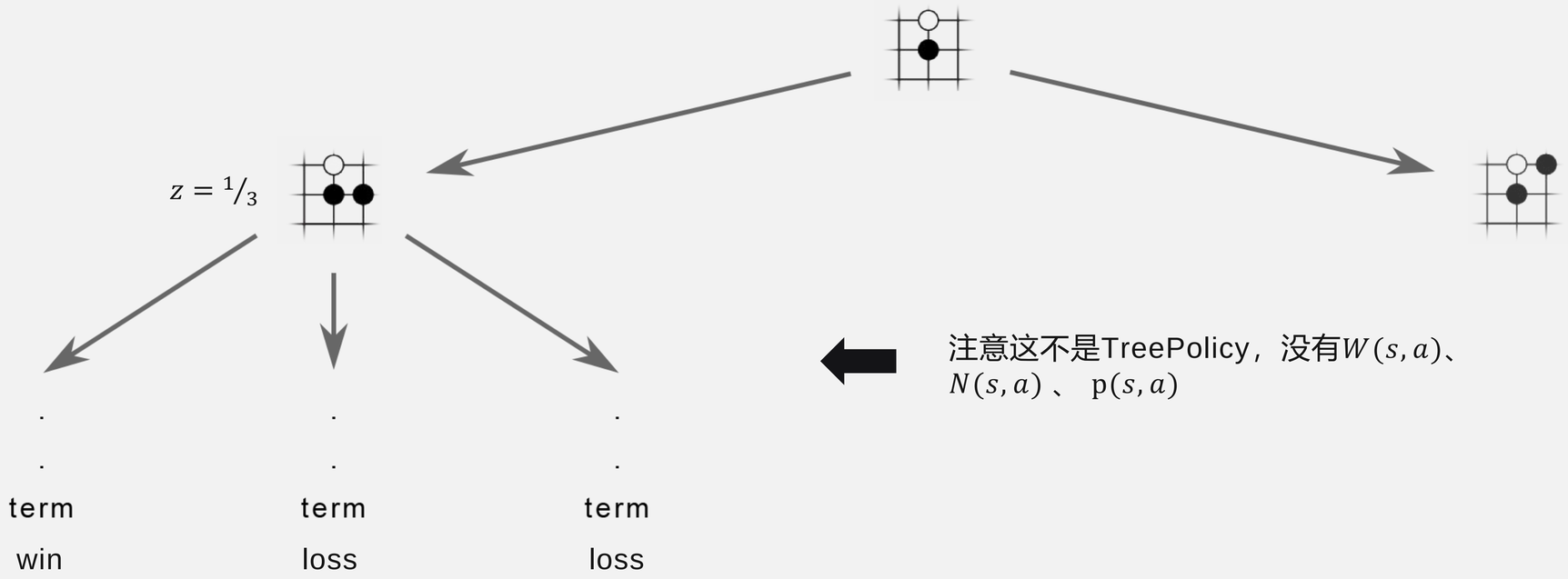
function EXPAND(v)
  foreach $a \in A(s(v))$ **do**
    add a new child v' to v
      with $s(v') = f(s(v), a)$
        $a(v') = a$
        $N(s(v), a) = 0$
        $W(s(v), a) = 0$
        $p(s(v), a) = p$
  return v

function BACKUP(v, Δ)
  while v is not null **do**
    $N(v) \leftarrow N(v) + 1$
    $W(v) \leftarrow W(v) + \Delta$
    $\Delta \leftarrow -\Delta$
    $v \leftarrow$ parent of v

Monte Carlo
蒙 特 卡 洛

Evaluate是什么

回顾rollout的方案。



注意这不是TreePolicy，没有$W(s,a)$、$N(s,a)$ 、 $\mathrm{p}(s,a)$

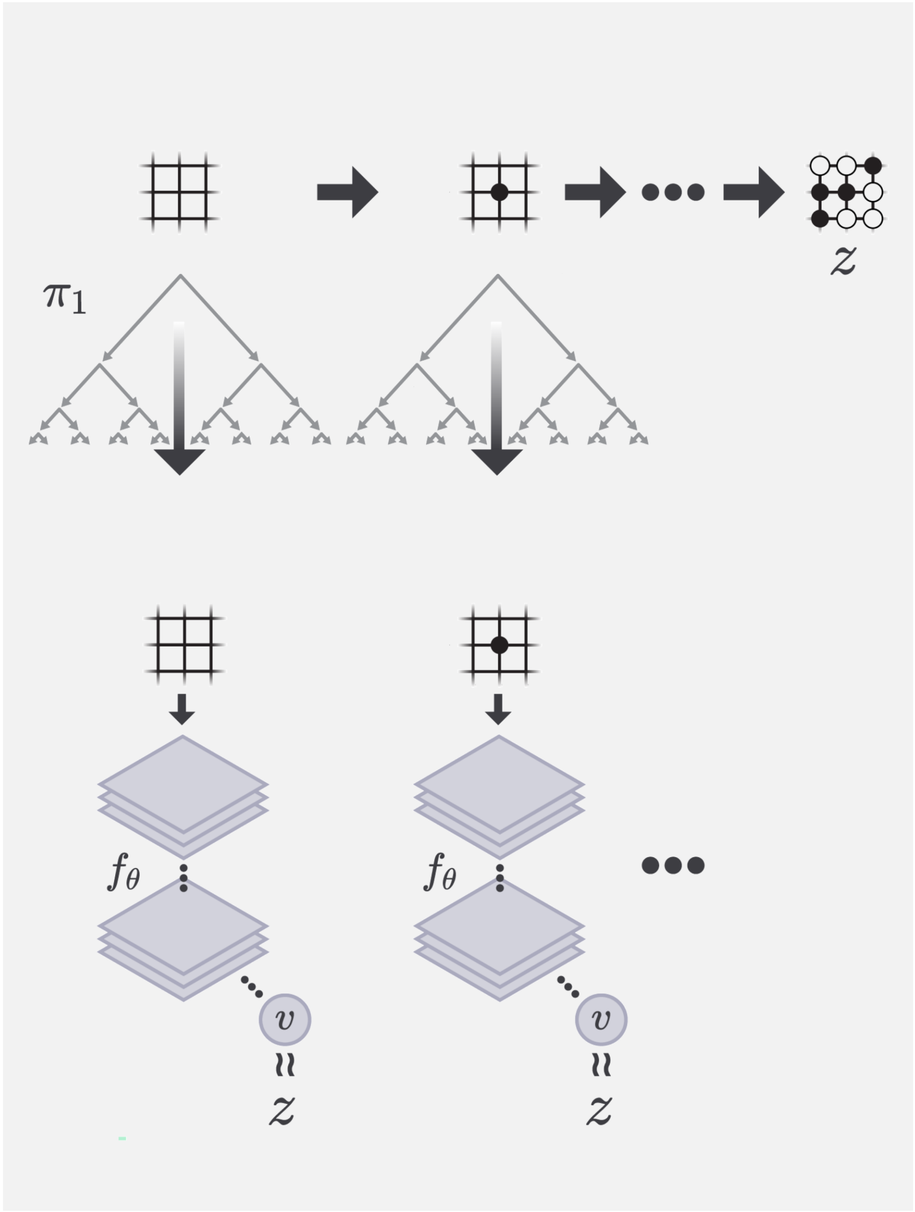
Monte Carlo
蒙 特 卡 洛

5

怎样学习 $v(s;\theta)$

纯regression问题

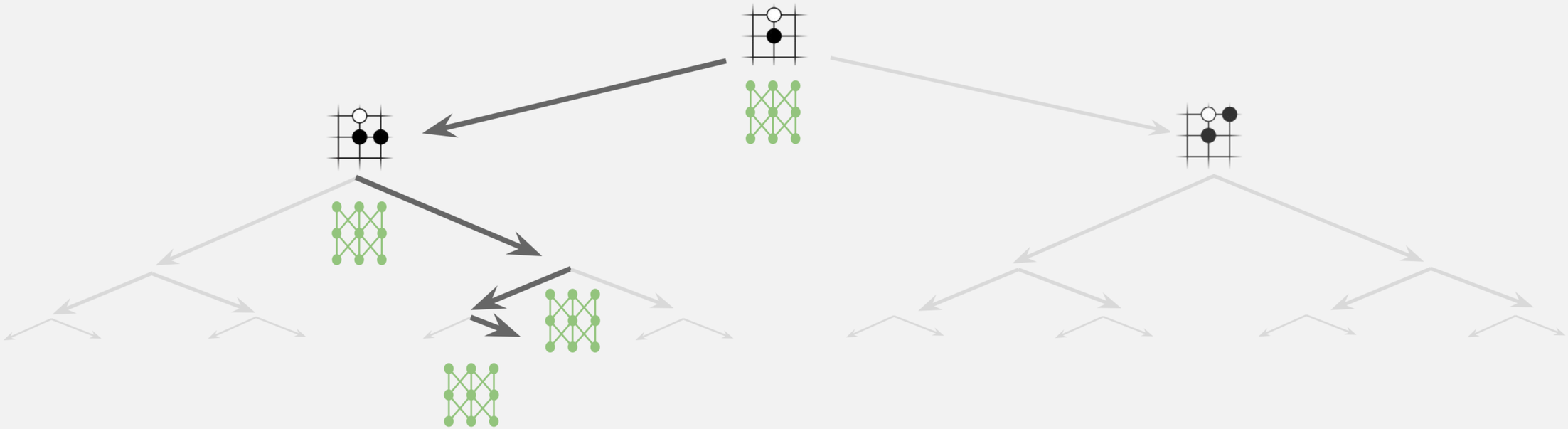
$v(s) \to z$



Monte Carlo
蒙 特 卡 洛

p与v的小结

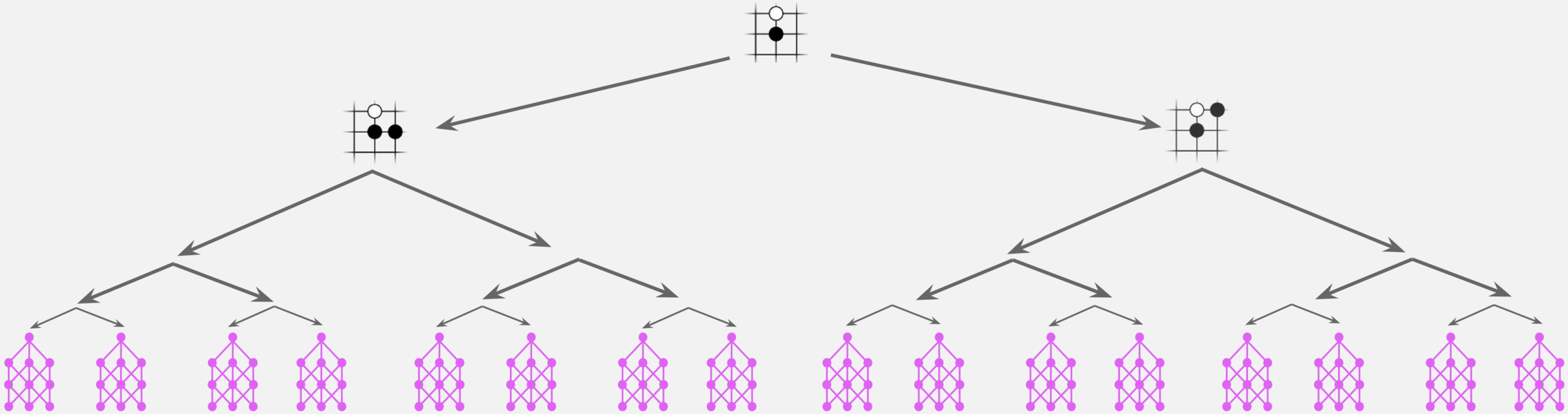
人是怎么下围棋的?

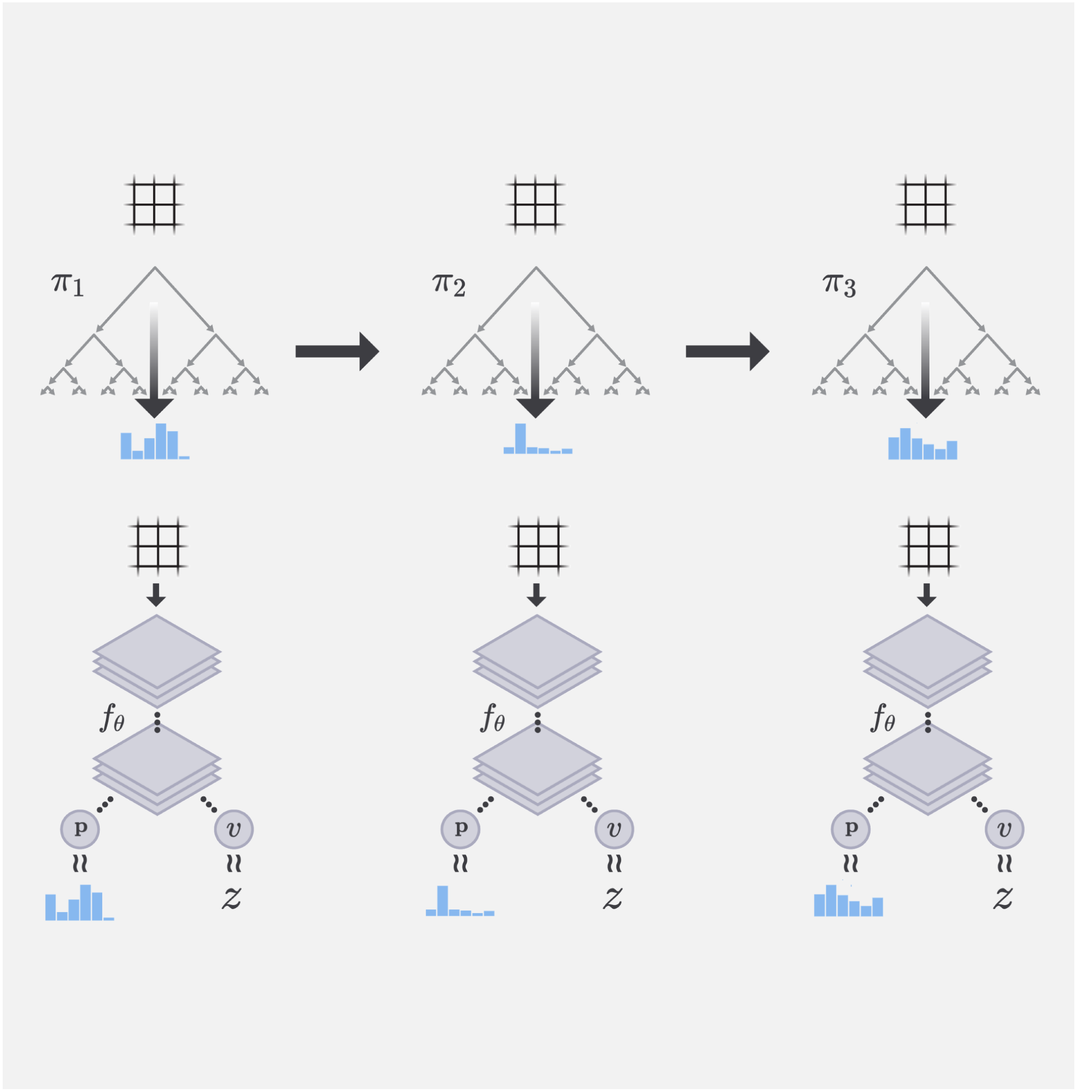


Monte Carlo
蒙 特 卡 洛

p与v的小结

人是怎么下围棋的？与人的思维过程相似，是合理的AI方案。





7

RL

Self-play实现的水平增强

用π_1执行MCTS（强化）产生样本，训练得到π_2。假设π_2能够拟合强化后的agent，则下次用π_2执行MCTS会得到更强的agent。

Monte Carlo
蒙 特 卡 洛

7

RL的哲学

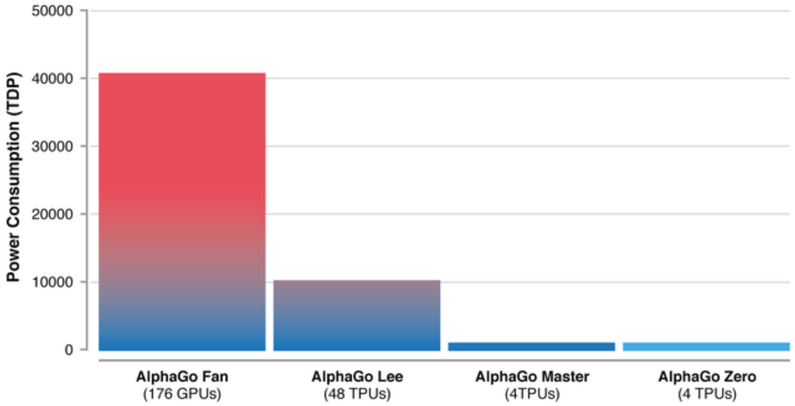
Tabula rasa之梦

人生来是一块白板，没有与生俱来的心智，知识要靠经验和感官获得。

Monte Carlo
蒙 特 卡 洛

7

RL的代价



每局游戏 = 1600 $MCTS\ search \times 200\ moves$

RL需要700k batch $\times$ 4096 $^{moves}/_{batch}$

1080Ti + E5 12 cores的性能 = $13\ ^{s}/_{move}$

batch = 8，filter = 256，block = 40

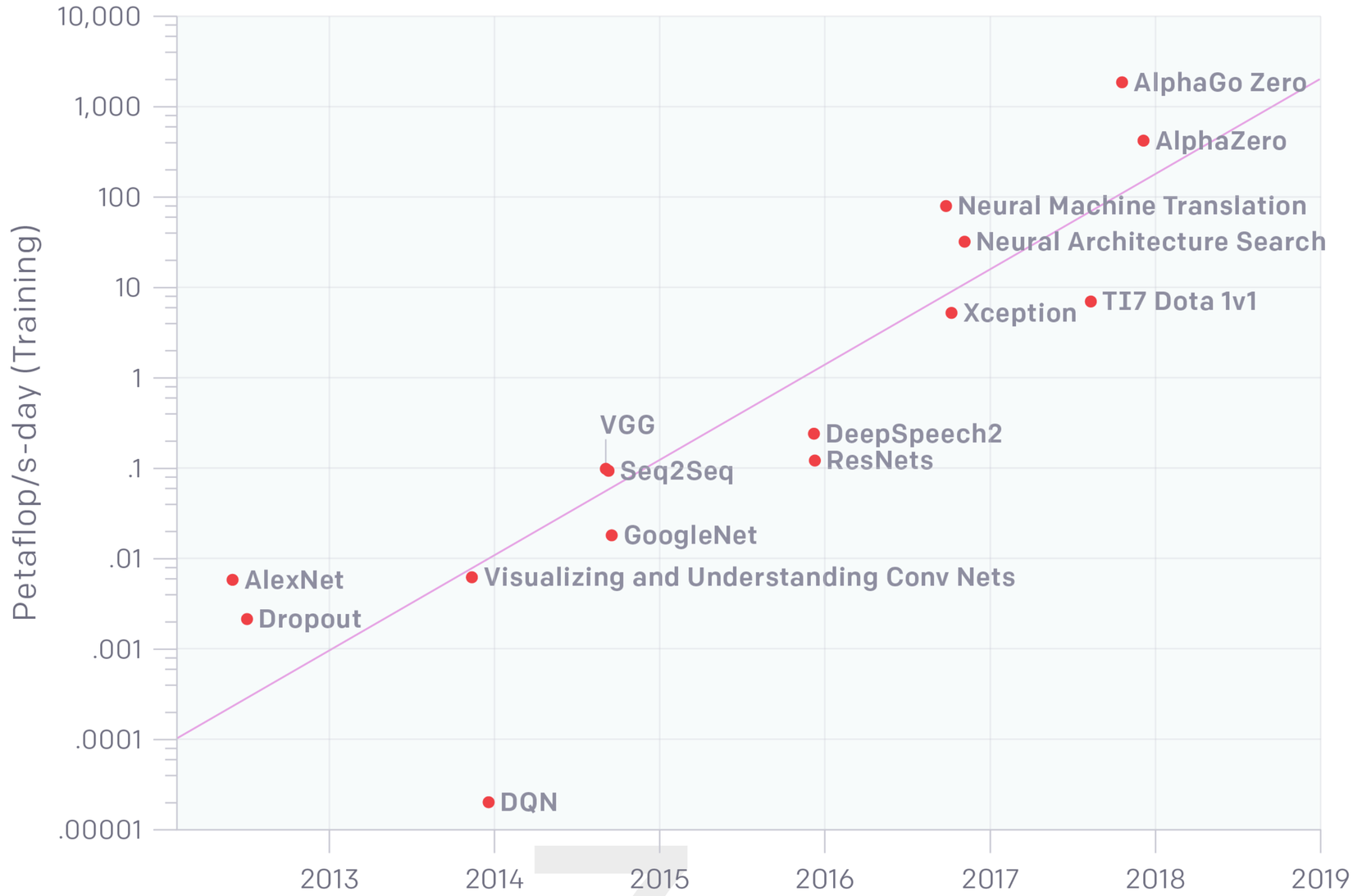
总时间= $13\ ^{s}/_{move} \times 700k\ batch \times 4096\ ^{move}/_{batch} = 1182\ years$

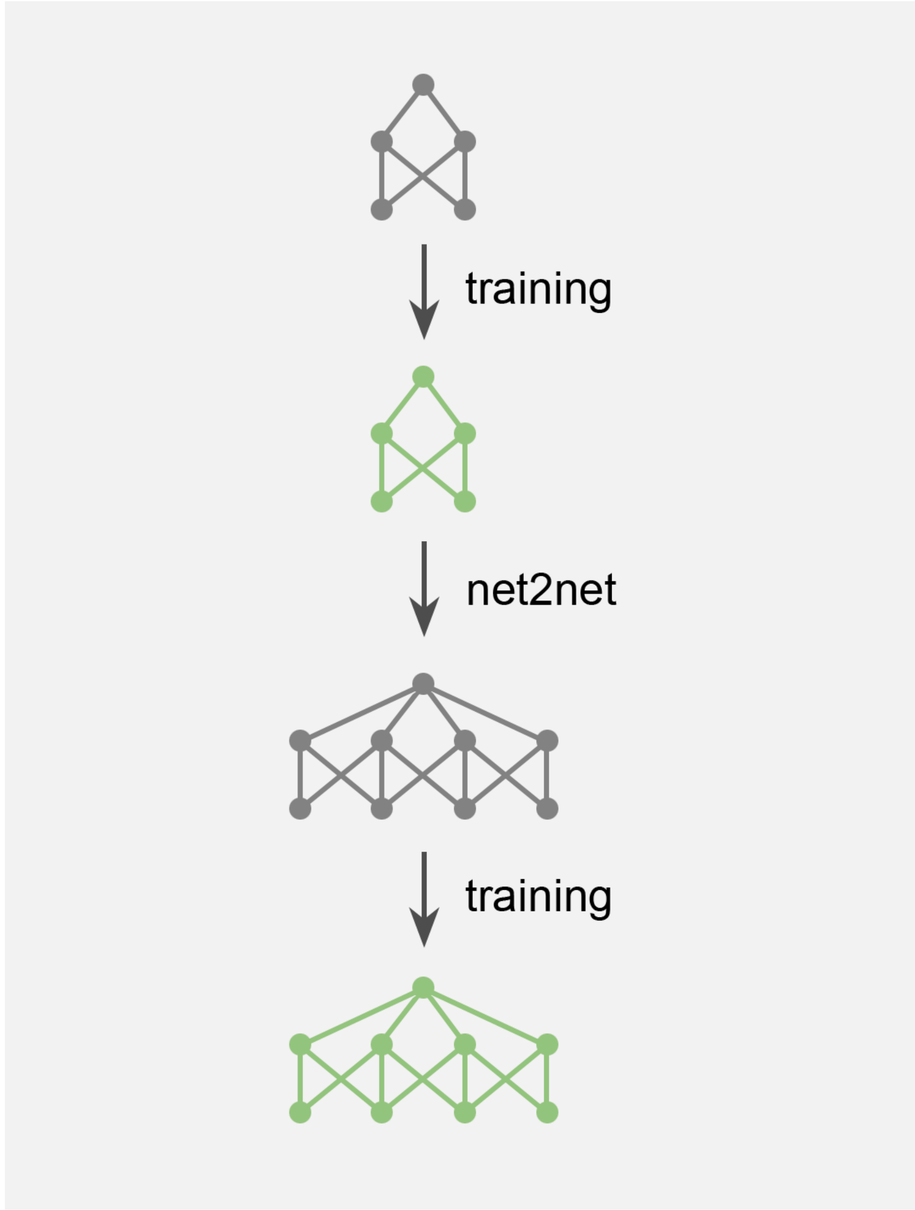
在Google Cloud租用TPU超过5,000,000人民币

误会：4TPU是做什么的?

产生训练数据（prediction）需要多久?

Monte Carlo
蒙 特 卡 洛





Monte Carlo
蒙 特 卡 洛

7

net2net

一种优雅的迁移学习方案

传统流程：抛弃旧model，从samples重新训练新model

新流程：快速训练旧model，扩展为新model，继续训练

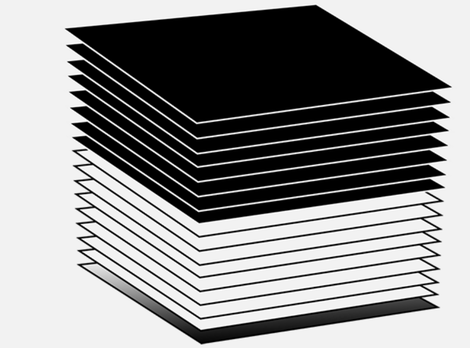
Monte Carlo
蒙 特 卡 洛

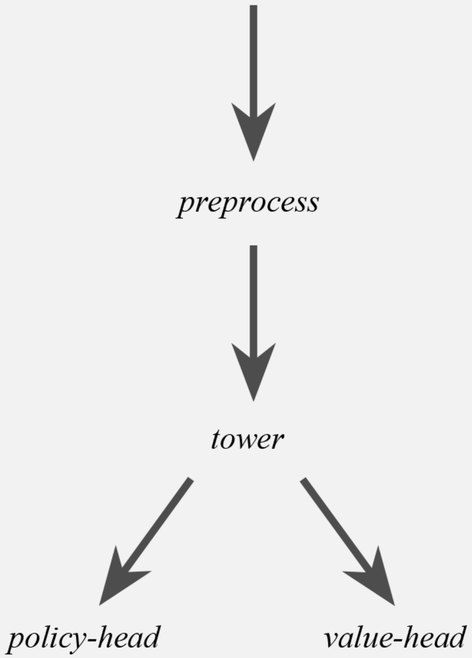
8

网络结构

输入与输出

输入：361 × ((p1 stone + p2 stone) × 8 + 当前color)
color：1表示轮到黑棋
preprocess/tower：conv2d 3x3x256、BN、ReLU
tower：20或40
插曲：color plane够用吗?





Monte Carlo
蒙 特 卡 洛

8

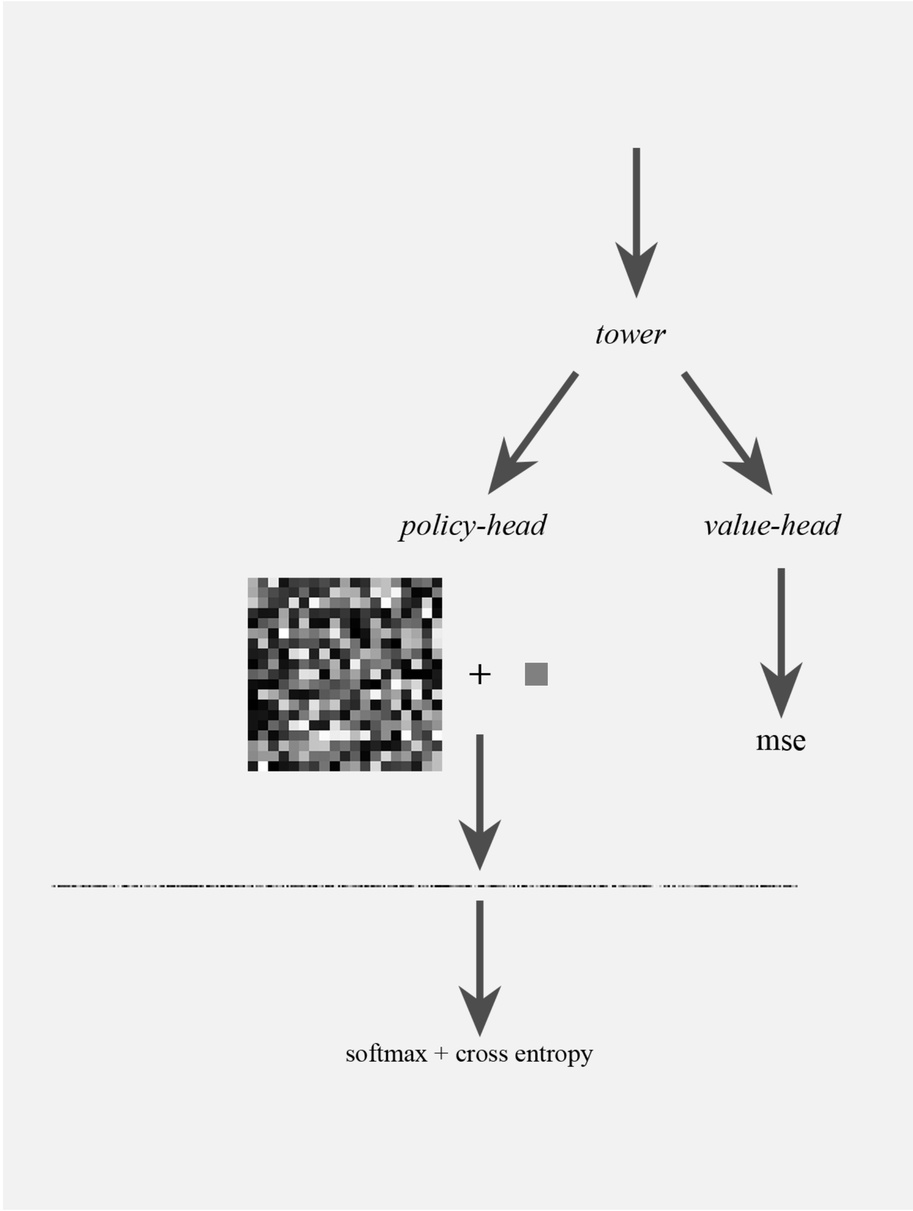
网络结构

输入与输出Z

policy-head输出：361 + 1

value-head输出：1

loss = p loss + v loss + L2



Monte Carlo
蒙 特 卡 洛

Thanks!

欢迎对游戏人工智能技术感兴趣的同学加入我们！

朱天驰