中山大學
SUN YAT-SEN UNIVERSITY

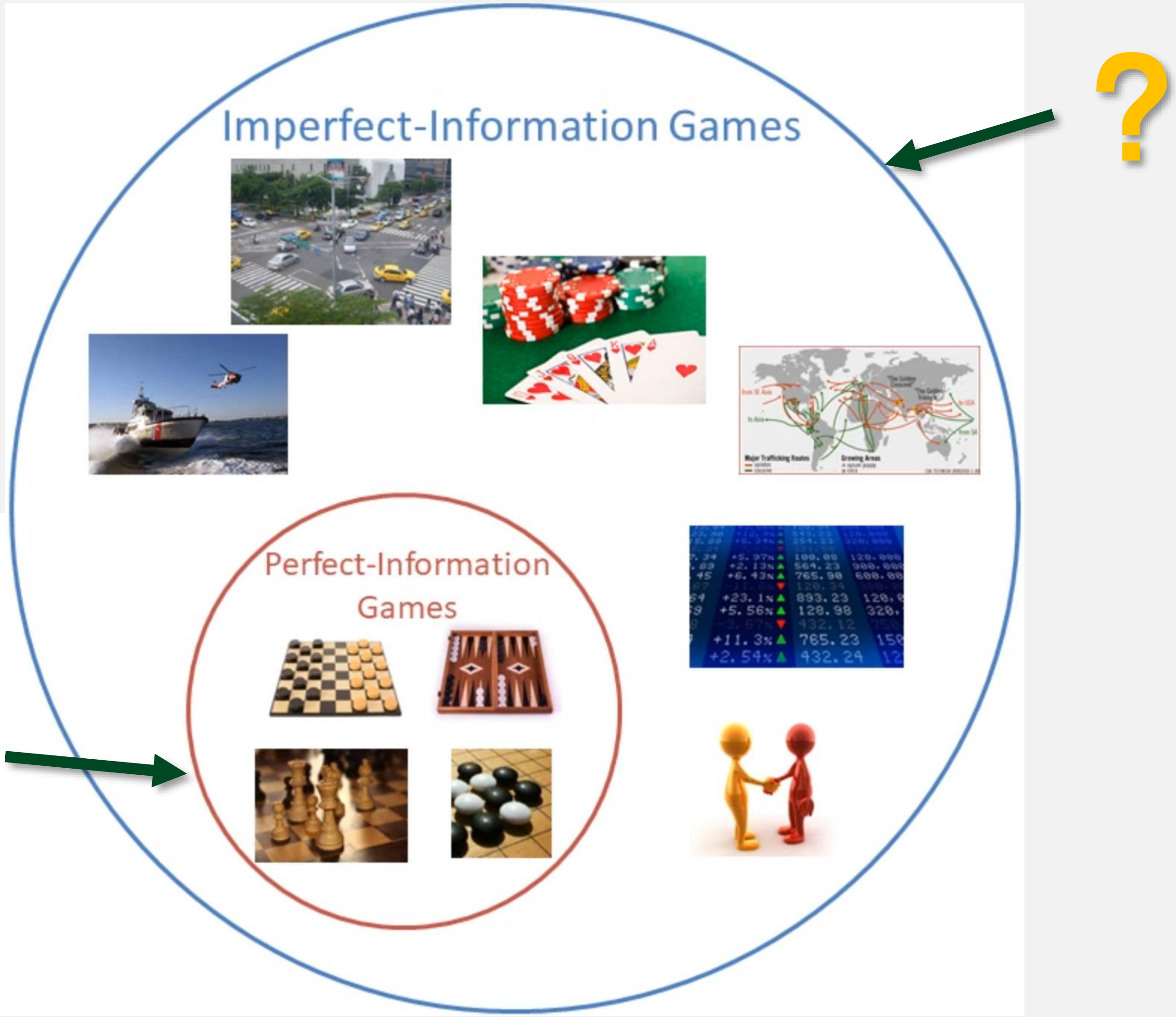
面向非完全信息博弈的强化学习

2022.5.31

目录

CONTENTS

- 非完全信息博弈广泛存在于人类社会之中，是人工智能发展的需要关注的核心问题之一。该类博弈的特例：完全信息博弈中的通用高效求解器已被研究者构建，而该针对问题本身的**突破性算法尚未出现。**




Chess
Shogi
Go
AlphaZero vs. Stockfish
AlphaZero vs. Elmo
AlphaZero vs. AG0
W:29.0% D:70.6% L:0.4%
W:2.0% D:97.2% L:0.8%
W:84.2% D:2.2% L:13.6%
W:98.2% D:0.0% L:1.8%
W:68.9% L:31.1%
W:53.7% L:46.3%

公共信息

私有信息

Part.01

博弈与强化学习

1.1 强化学习

observation

o_t

action

a_t

reward

r_t

将连续的时间尺度离散化

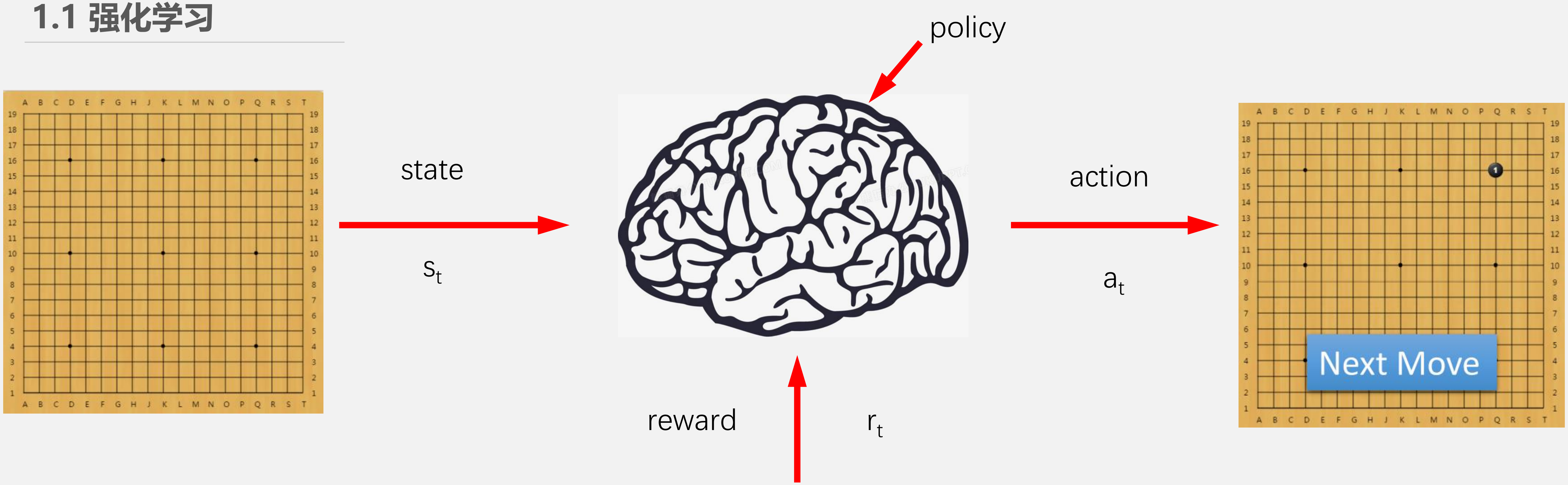
在每个时刻 t 有：

- 结合o_t，智能体执行动作a_t
- 环境处理该动作，执行内部逻辑
- 环境给出收益r_{t+1}、新观察值o_{t+1}
- 结合o_{t+1}，智能体执行动作a_{t+1}
- ……

在多数强化学习设定中，

认为observation 与 state 等价

1.1 强化学习

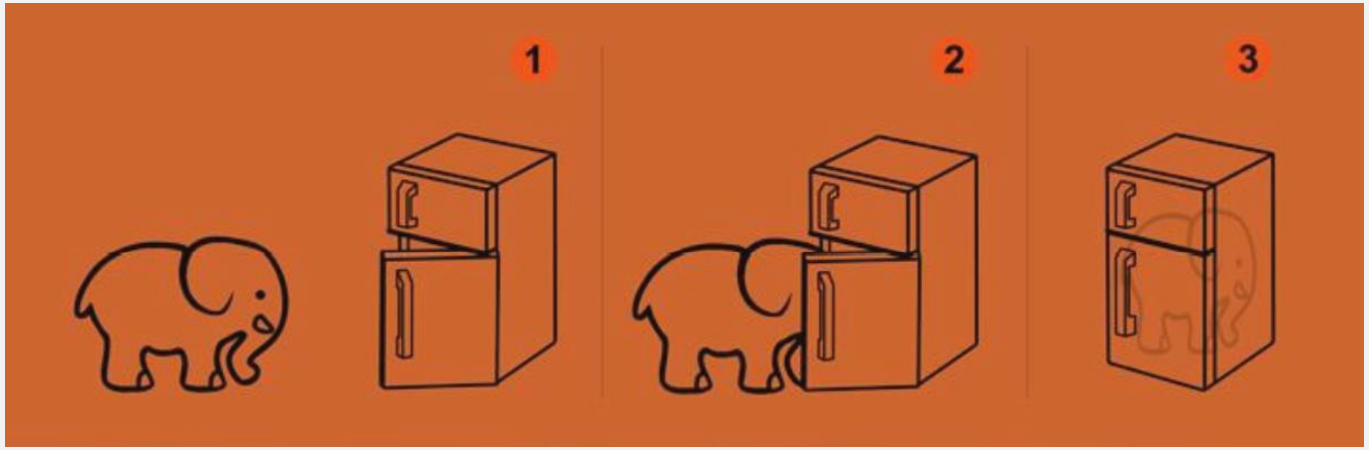


对于围棋：

- 多数时候：reward ＝ 0
- 认输或收官后气少：reward ＝ -1
- 对方认输或收官后气多：reward ＝ 1

1.2 获得策略

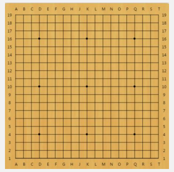
- 01 弄一个神经网络
- 02 选一个Loss函数
- 03 做梯度下降



深度学习太简单辣！！

1.2.1 获得策略——监督学习

人类顶级玩家经验数据集：



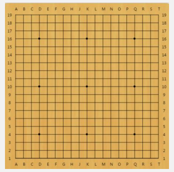
-> 走 (3, 4)

-> 走 (13, 6)

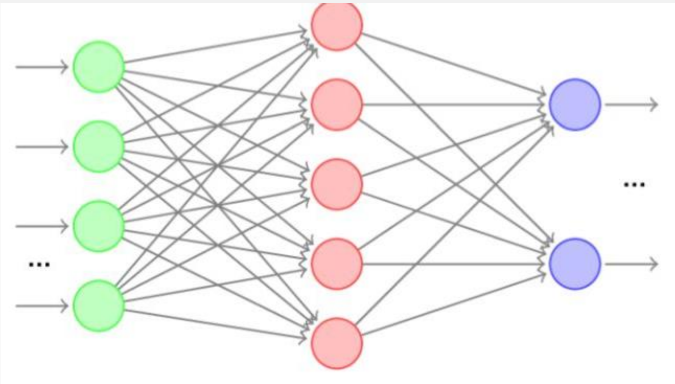
-> 走 (5, 5)

神经网络：泛化性好

输入



编码



输出

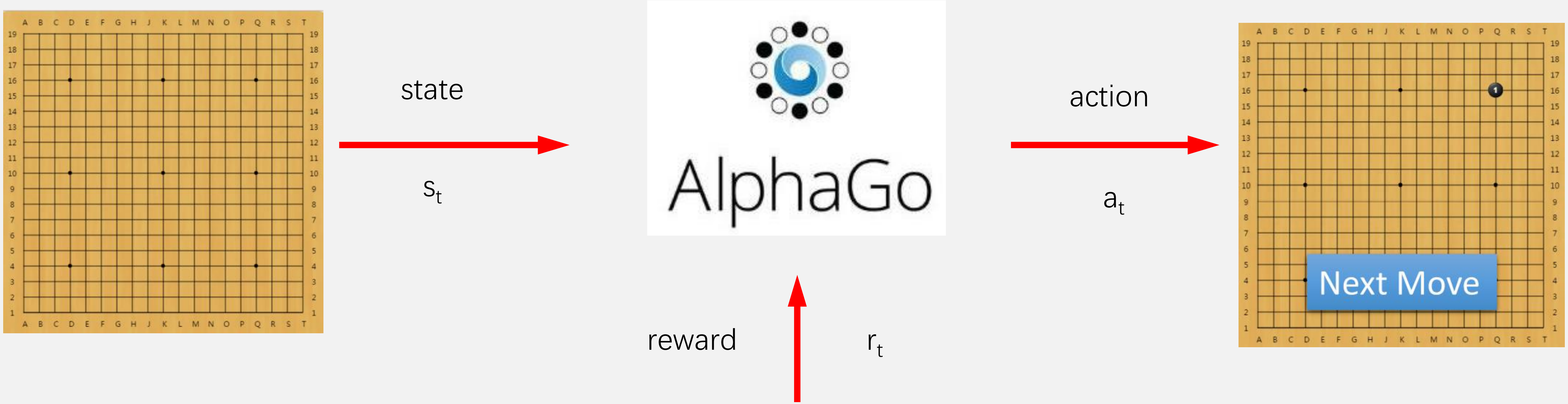
走(3, 4) 0.2

…

走(10, 8) 0.01

采取某动作的概率

1.2.2 获得策略——自博弈



Game Engine + Opponent AlphaGo

1.2.2 获得策略——自博弈


黑子
state
s_t
AlphaGo
action
a_t
Next Move
reward
r_t
Game Engine +
AlphaGo
白子

1.2.2 获得策略——自博弈


白子
state
s_t
AlphaGo
action
a_t
Next Move
reward
r_t
Game Engine +
AlphaGo
黑子

1.3.1 自博弈求解算法——MinMax

MinMax (DP)：最差情况下的最优选择

方块希望值尽量大，三角反之

-> 方块3选1

-> 三角3选1

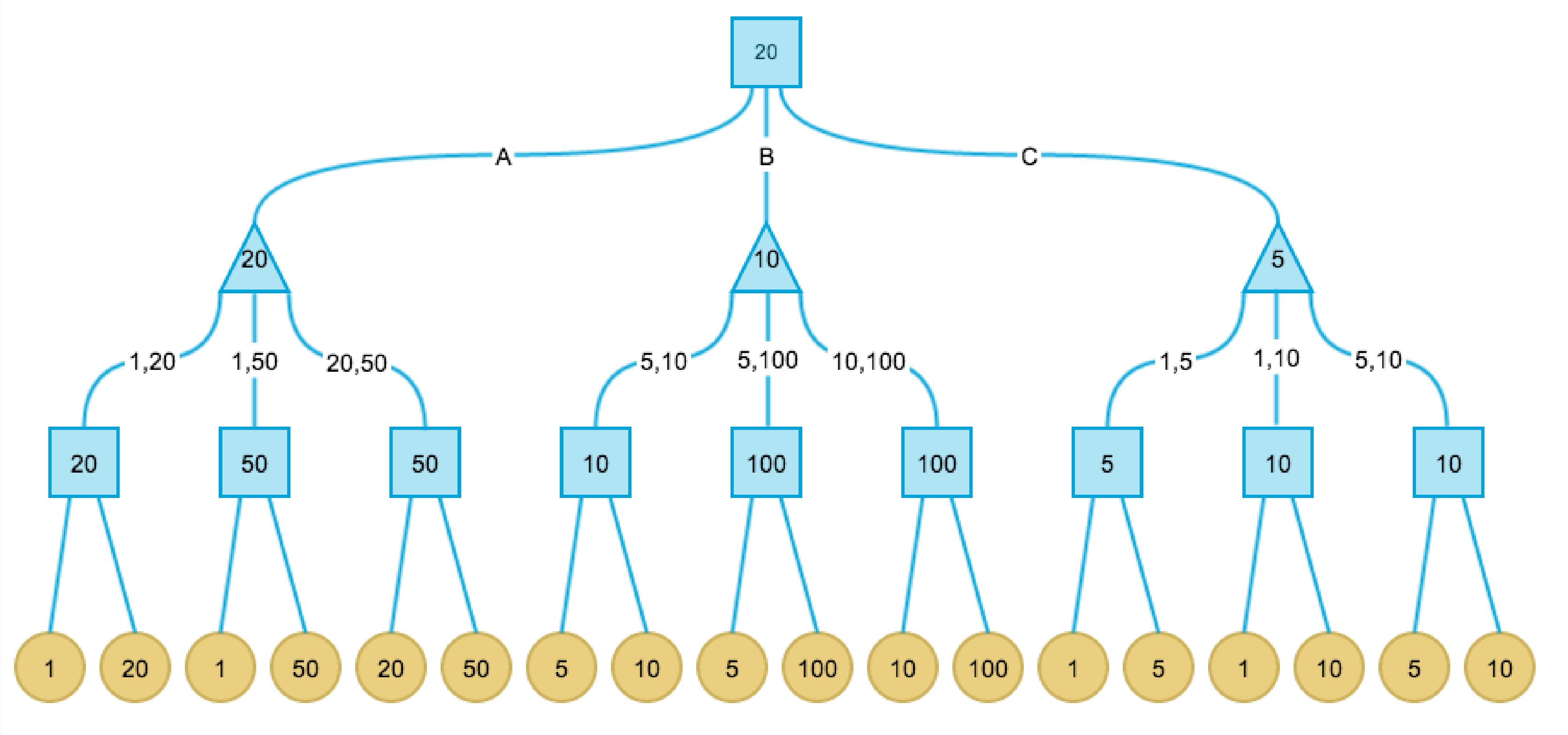
-> 方块2选1

-> 游戏结束

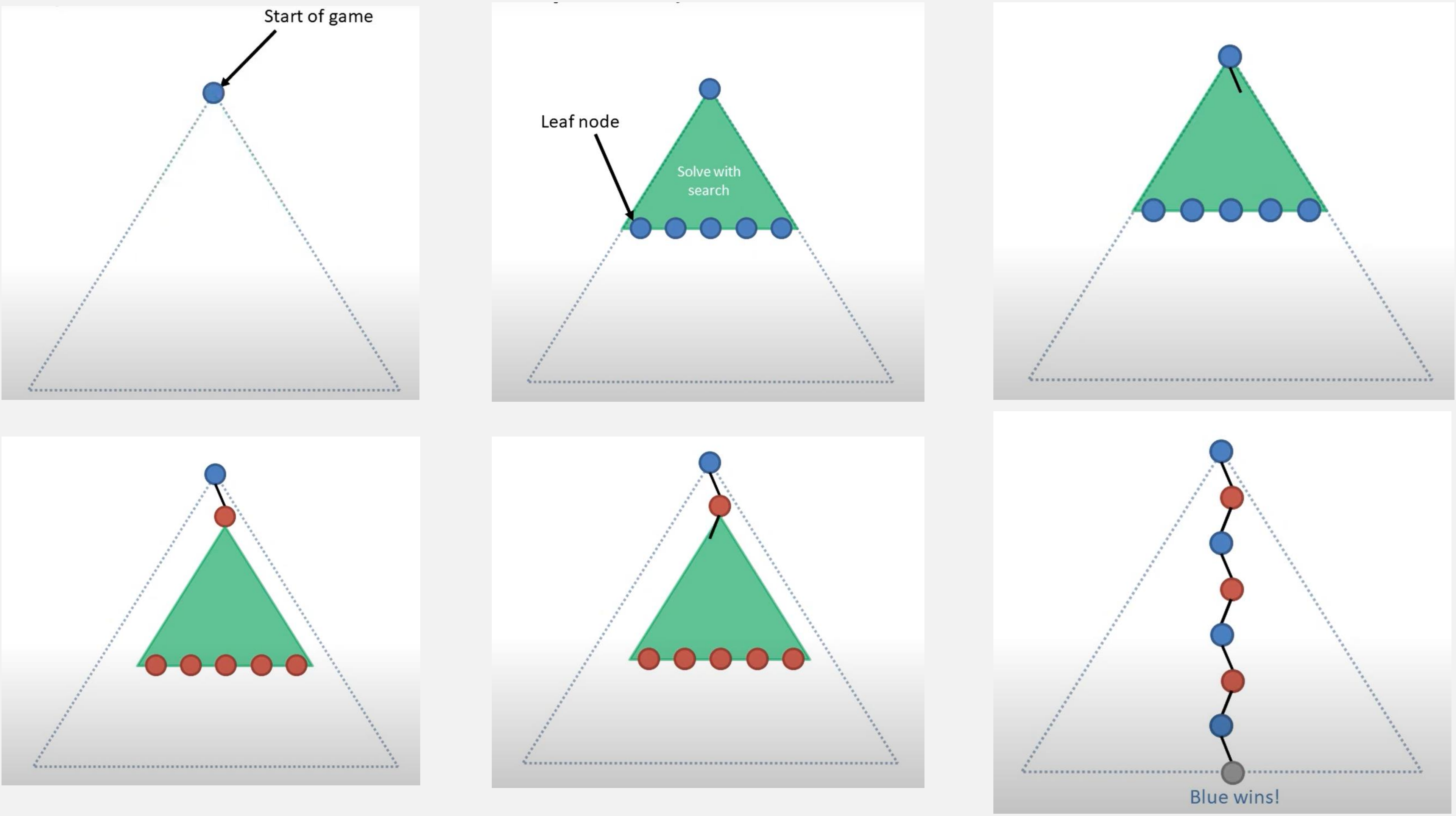
前提：假定对手在玩最优策略

某博弈的博弈树

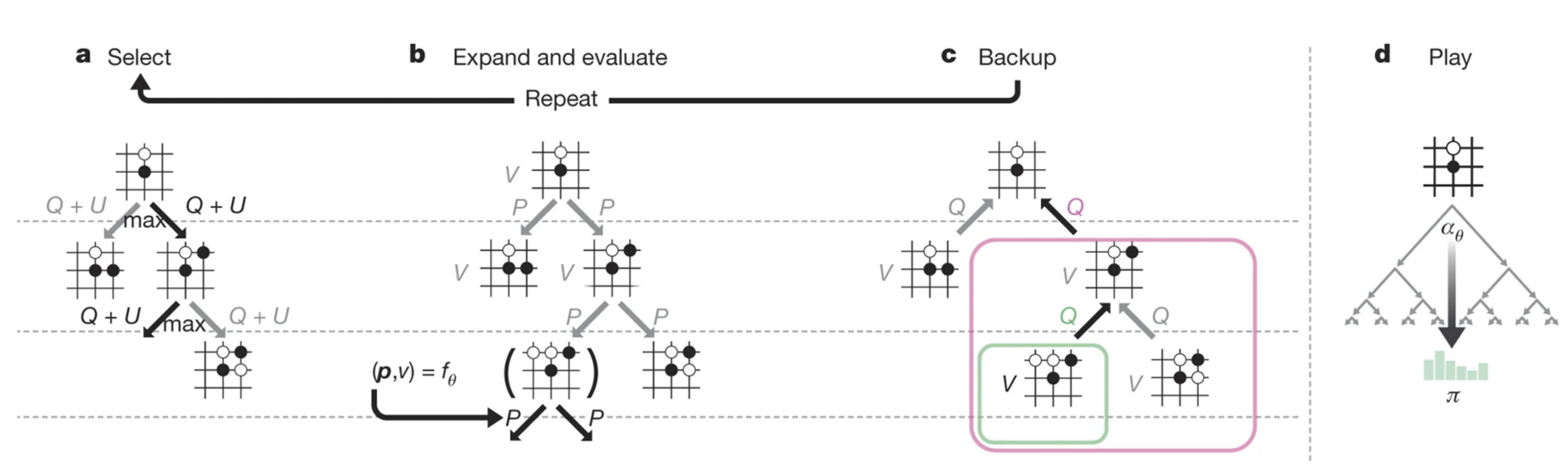
状态是方块/三角/圆，动作是线



1.3.2 自博弈求解算法——蒙特卡洛树搜索



1.3.2 自博弈求解算法——蒙特卡洛树搜索



$$score(a) \triangleq Q(a) + \frac{\eta}{1+N(a)} \cdot \pi(a|s;\theta)$$

蒙特卡洛/网络估计

Part.02

非完全信息博弈与强化学习

2.1 非完全信息博弈的例子

石头剪刀布下的自博弈：

P1：我打算一直出石头

P2：我的最优决策应该是布

P1：我发现他爱出布，那我以后多出剪刀

P2：我多出石头

P1：我布

P2：剪

P1：石头

……

猜硬币：P1抛硬币并观察结果后可以选择是否放弃游戏，在其选择继续游戏后P2可以选择是否放弃游戏。

任意一方玩家放弃，则其输一枚赌注。若两方均选择参与游戏，则硬币正面与反面向上分别对应P1、2赢得两枚赌注。

P1：正面向上继续，反面向上放弃

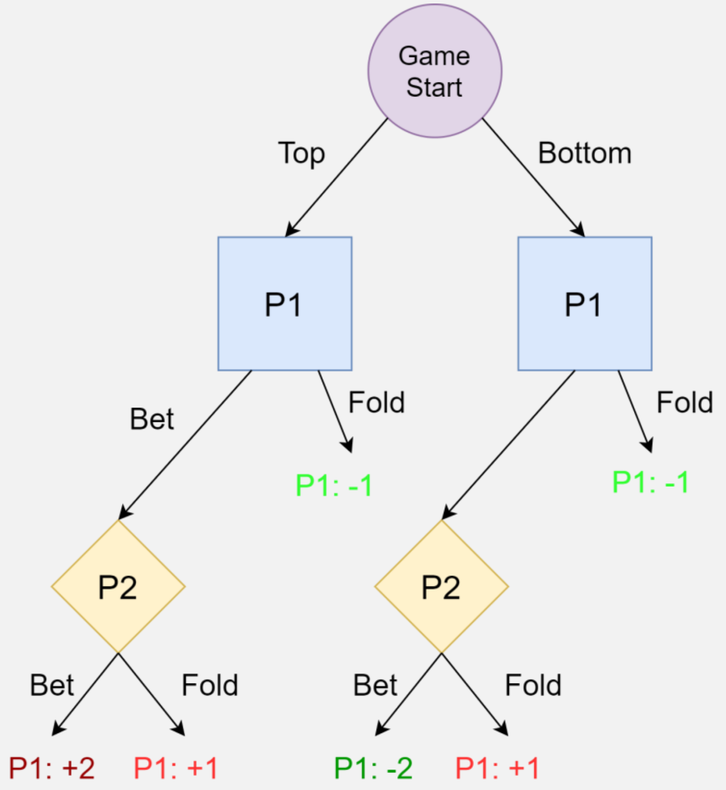
P2：P1继续则放弃，否则继续 $E(R_2) = 0$

P1：始终继续 $E(R_1) = 1$

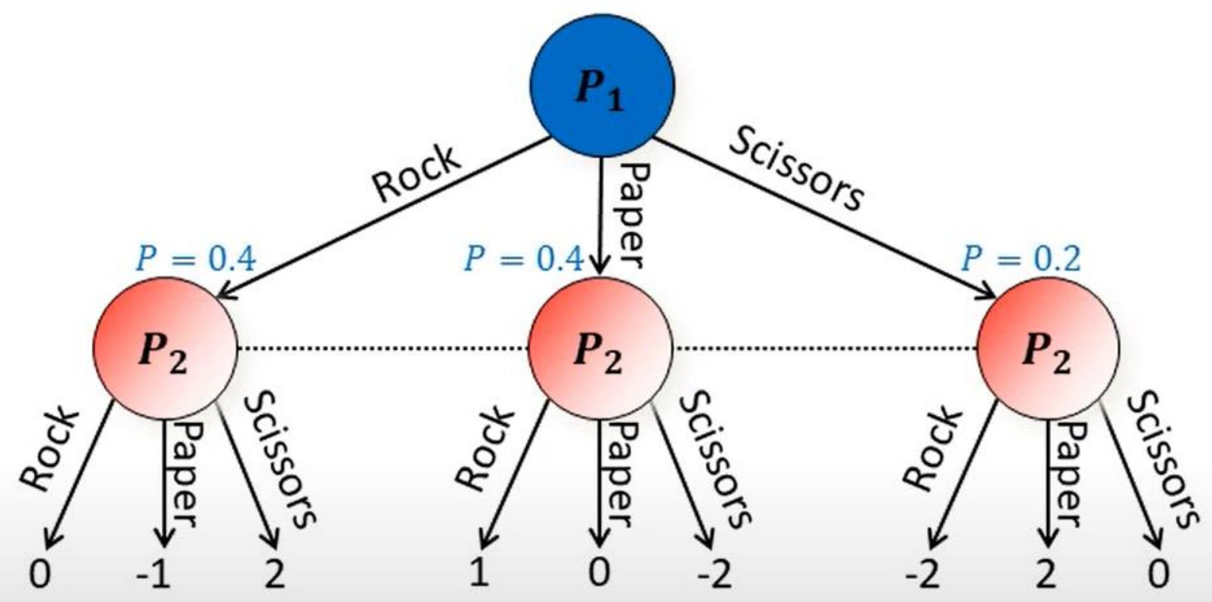
P2：始终继续 $E(R_2) = 0$

P1：正面向上继续，反面向上放弃 $E(R_1) = 1$

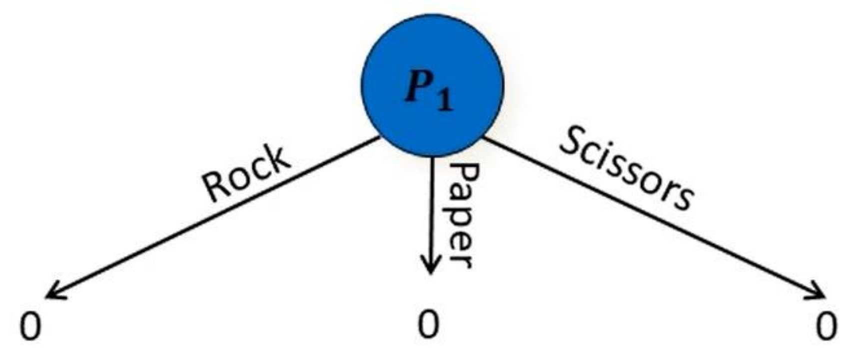
……



2.1 非完全信息博弈的例子



石头剪刀布变种，其最优策略如图



最优策略下收益值函数

2.2 非完全信息博弈难点

循环归因：最优**策略**取决于对手的**私有信息分布**，该信息由对手的**行为历史**推理得出。而对手的**行为历史**到其私有信息的分布取决于对手的**策略**，这又由对手对我方**私有信息**的分布的估计得出。

值函数失效：不同于基于值函数的（Value-Based） 强化学习，由值函数无法推知最优策略。

不能DP：自博弈相关性：无法区分对方真实状态，博弈树底层子问题求解后无法被上层利用

无法定义对方当前最优策略：非完全信息博弈中的最优策略不能保证某次必胜，但是要是拥有最高期望收益的，这就要求其能够应对对手策略改变

2.3 纳什均衡

纳什均衡 —— 一种泛化的最优策略描述：单个玩家策略对纳什均衡策略的偏离必将导致其**期望**收益的下降

在完全信息博弈中，纳什均衡退化为强化学习的优化目标

如果我认为对手的玩法离纳什均衡很远，我能不能离开自己的纳什均衡去利用他的失误?

——不行

1. 如果我离开了纳什均衡，被他发现，他来利用我，那保不齐谁赚的更多
2. 大多数纳什均衡策略都是非确定性策略，如果对手确实是在玩纳什均衡呢?

2.3 纳什均衡

囚徒乙 / 囚徒甲	供 认	拒 供
供 认	2年，2年	0年，5年
拒 供	5年，0年	0.5年，0.5年

假定囚徒间单独决策不合作

纳什均衡解：**均供认**

2.3 纳什均衡

石头剪刀布下的自博弈：

P1：我打算一直出石头

P2：我的最优决策应该是布

P1：我发现他爱出布，那我以后多出剪刀

P2：我多出石头

P1：我布

P2：剪

P1：石头

……

纳什均衡解：(1/3, 1/3, 1/3) **E(R) = 0**

猜硬币：P1抛硬币并观察结果后可以选择是否放弃游戏，在其选择继续游戏后P2可以选择是否放弃游戏。

任意一方玩家放弃，则其输一枚赌注。若两方均选择参与游戏，则硬币正面与反面向上分别对应P1、2赢得两枚赌注。

P1：正面向上继续，反面向上放弃

P2：P1继续则放弃，否则继续 **$E(R_2) = 0$**

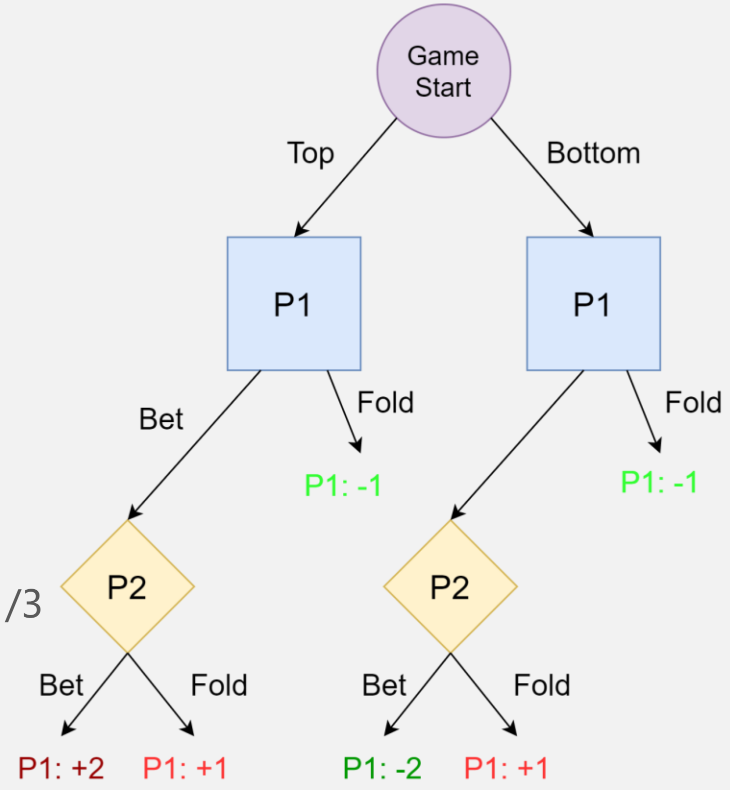
P1：始终继续 **$E(R_1) = 1$**

P2：始终继续 **$E(R_2) = 0$**

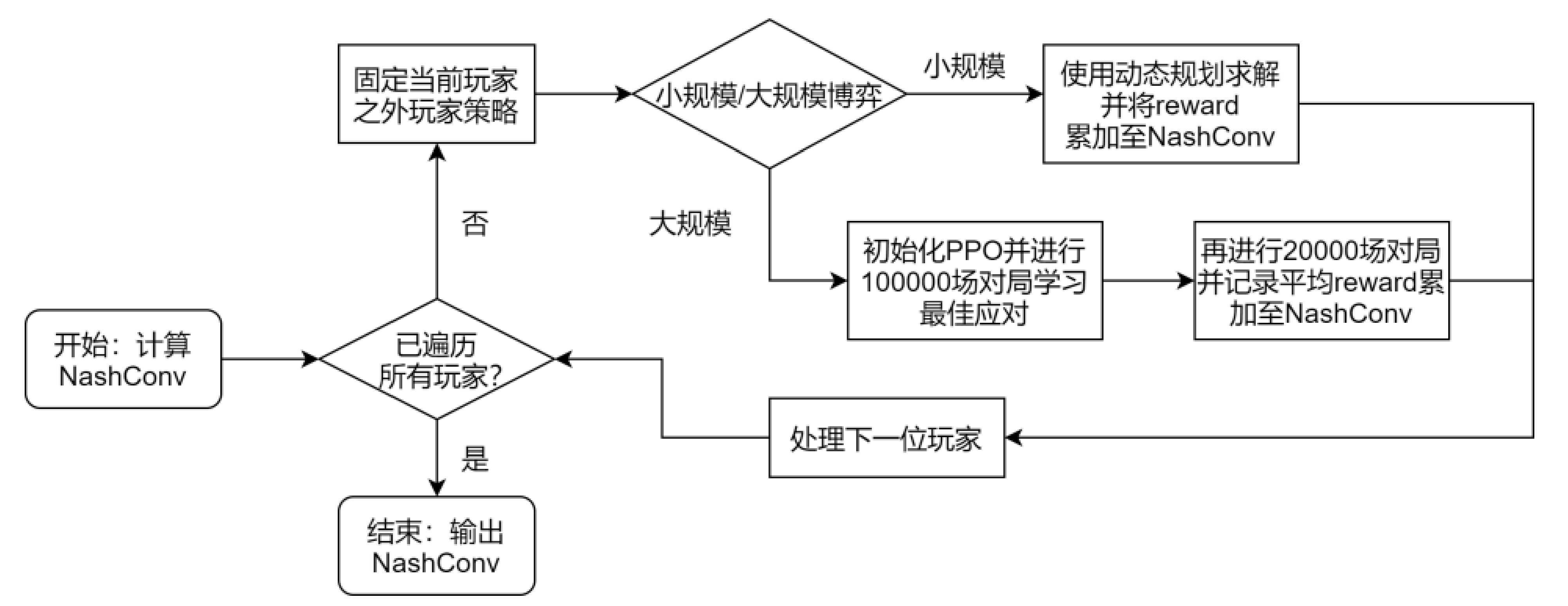
P1：正面向上继续，反面向上放弃 **$E(R_1) = 1$**

……

纳什均衡解：正面向上继续游戏，其他情形以1/3概率继续游戏 **$E(R_1) = 1/3$**

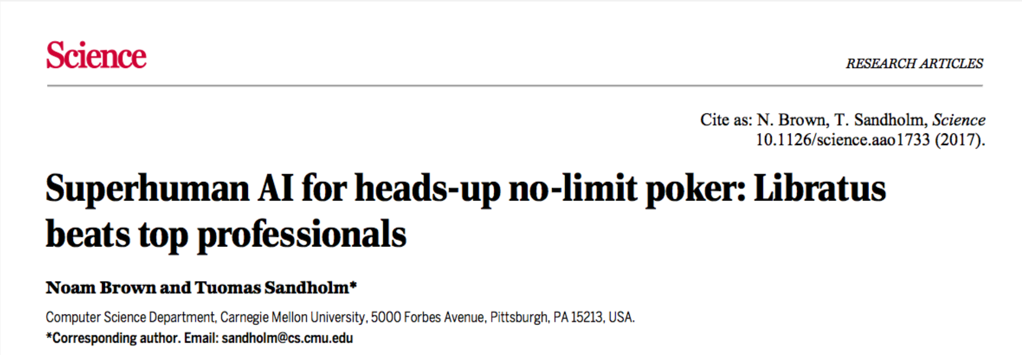


2.4 性能度量方法


开始：计算
NashConv
已遍历
所有玩家?
否
是
结束：输出
NashConv
固定当前玩家
之外玩家策略
小规模/大规模博弈
小规模
大规模
使用动态规划求解
并将reward
累加至NashConv
初始化PPO并进行
100000场对局学习
最佳应对
再进行20000场对局
并记录平均reward累
加至NashConv
处理下一位玩家

2.5 现有主要非完全信息博弈算法

1. **虚拟遗憾最小化（Counterfactual Regret Minimization, CFR）**：基于博弈理论，最强，但是需要模型，且泛化性差。
2. **基于最佳应对（固定对手策略后的MDP下的最优策略）**
   - **2.1 虚拟博弈**（Fictitious Play）：基于博弈理论，不断找最佳应对，把最佳应对策略做平均，可以无模型，慢。
   - **2.2 基于种群**（Population based Self-Play）：以特殊方法融合最佳应对策略。

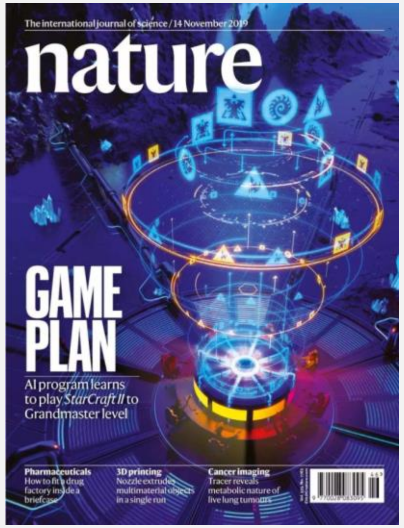

Science RESEARCH ARTICLES

Cite as: N. Brown, T. Sandholm, *Science* 10.1126/science.aao1733 (2017).

Superhuman AI for heads-up no-limit poker: Libratus beats top professionals

Noam Brown and Tuomas Sandholm*

Computer Science Department, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA.
*Corresponding author. Email: sandholm@cs.cmu.edu



2.5.1 现有主要非完全信息博弈算法——虚拟遗憾最小化（CFR）

虚拟遗憾最小化核心思想

$$u^t = \sum_{a \in A} \sigma^t(a) u^t(a)$$

期望收益

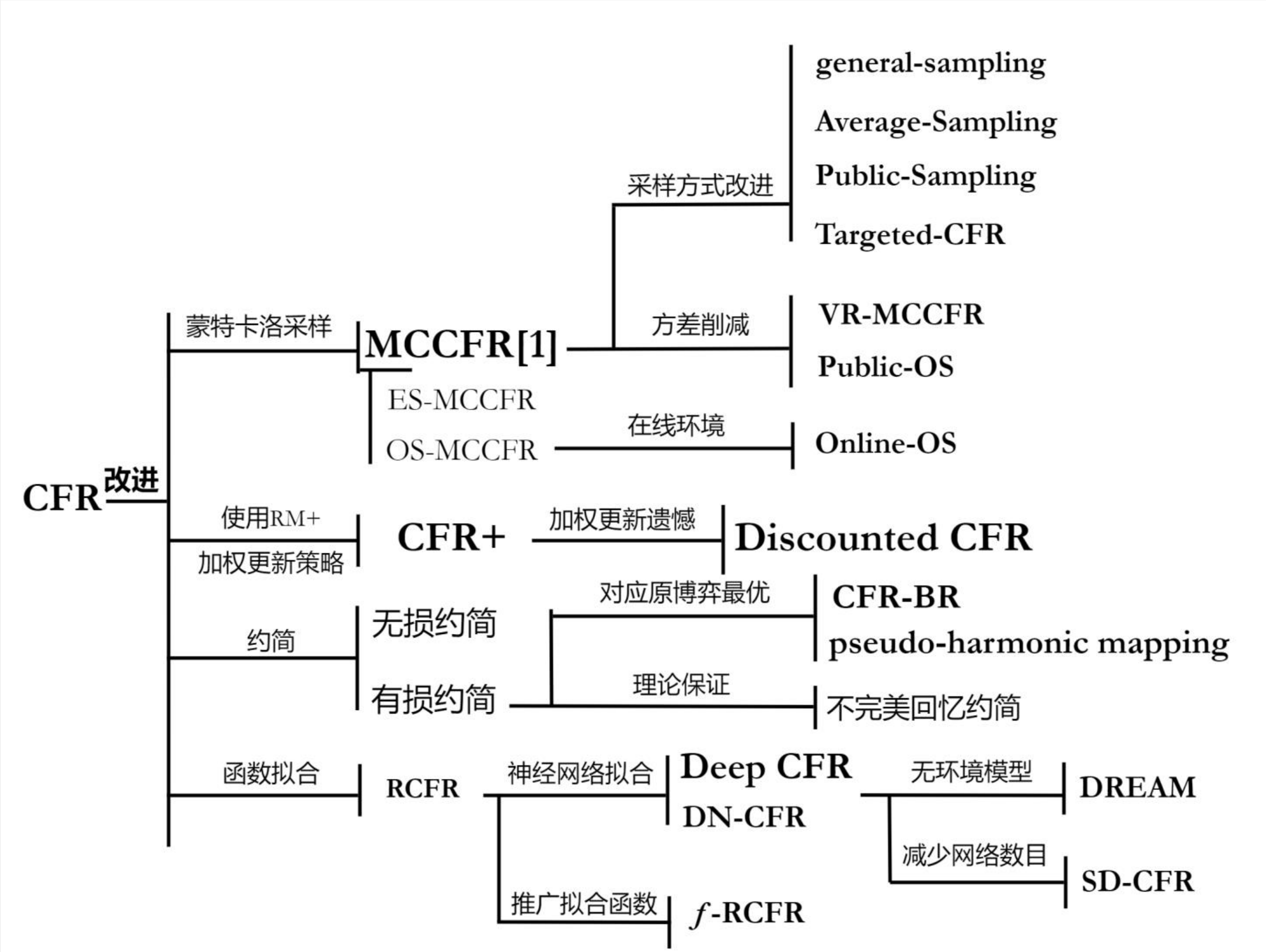
$$R^T(a) = \sum_{t=1}^{T} \left(u^t(a_t) - u^t \right)$$

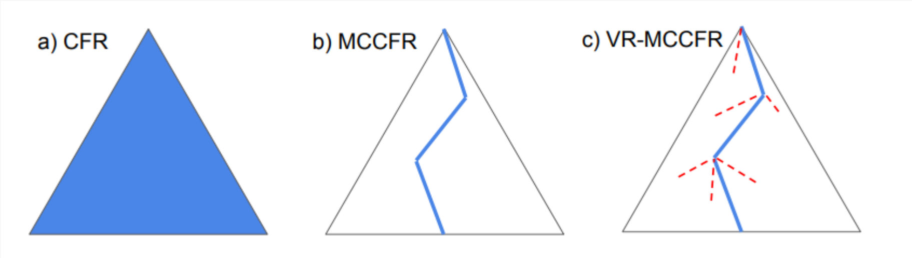
累计遗憾值

$$\sigma^{T+1}(a) = \begin{cases} \dfrac{R^{T,+}(a)}{\sum_a R^{T,+}(a)} & \text{if } \sum_a R^{T,+}(a) > 0 \\ \dfrac{1}{|A(I)|} & \text{otherwise} \end{cases}$$

遗憾值匹配
得到均衡解

2.5.1 现有主要非完全信息博弈算法——虚拟遗憾最小化（CFR）





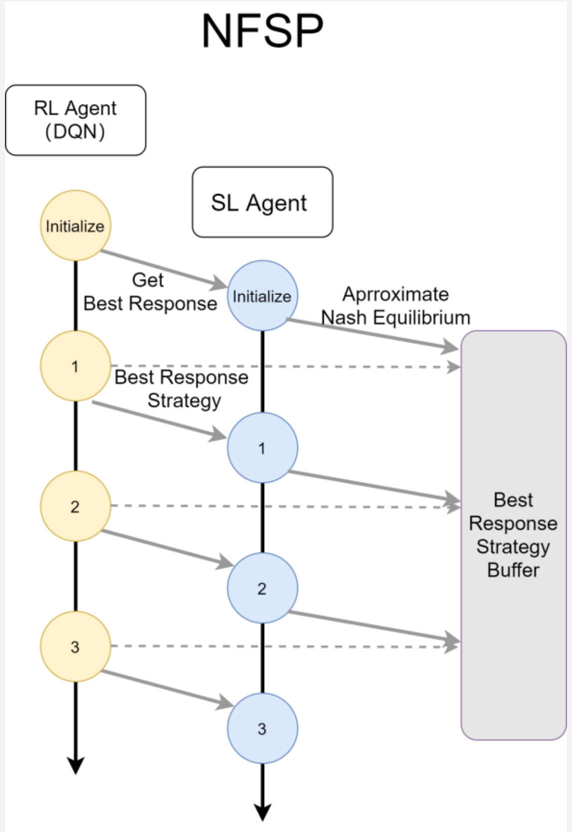
博弈环境	信息集数	信息集大小
德州扑克	10^{162}	10^{3}
德州扑克（约简）	10^{8}	10^{155}

2.5.2 现有主要非完全信息博弈算法——虚拟博弈

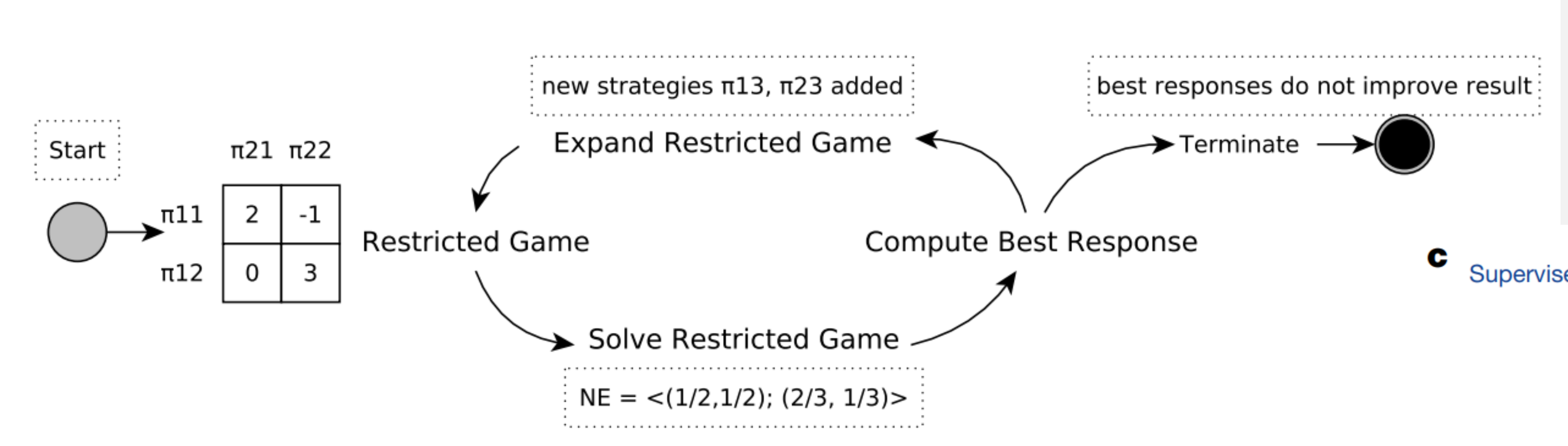
FP $\xrightarrow{\text{扩展式博弈}}$ **XFP** $\xrightarrow{\text{神经网络拟合}}$ **NFSP**

Algorithm 1 Neural Fictitious Self-Play (NFSP) with fitted Q-learning

Initialize game Γ and execute an agent via RUNAGENT for each player in the game
function RUNAGENT(Γ)
  Initialize replay memories $\mathcal{M}_{RL}$ (circular buffer) and $\mathcal{M}_{SL}$ (reservoir)
  Initialize average-policy network $\Pi(s, a \mid \theta^{\Pi})$ with random parameters θ^{Π}
  Initialize action-value network $Q(s, a \mid \theta^{Q})$ with random parameters θ^{Q}
  Initialize target network parameters $\theta^{Q'} \leftarrow \theta^{Q}$
  Initialize anticipatory parameter η
  for each episode **do**
    Set policy $\sigma \leftarrow \begin{cases} \epsilon\text{-greedy}(Q), & \text{with probability } \eta \\ \Pi, & \text{with probability } 1-\eta \end{cases}$
    Observe initial information state s_1 and reward r_1
    for $t = 1, T$ **do**
      Sample action a_t from policy σ
      Execute action a_t in game and observe reward r_{t+1} and next information state s_{t+1}
      Store transition $(s_t, a_t, r_{t+1}, s_{t+1})$ in reinforcement learning memory $\mathcal{M}_{RL}$
      if agent follows best response policy $\sigma = \epsilon$-greedy (Q) **then**
        Store behaviour tuple (s_t, a_t) in supervised learning memory $\mathcal{M}_{SL}$
      end if
      Update θ^{Π} with stochastic gradient descent on loss
        $\mathcal{L}(\theta^{\Pi}) = \mathbb{E}_{(s,a)\sim\mathcal{M}_{SL}}\left[-\log \Pi(s, a \mid \theta^{\Pi})\right]$
      Update θ^{Q} with stochastic gradient descent on loss
        $\mathcal{L}(\theta^{Q}) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{M}_{RL}}\left[\left(r + \max_{a'} Q(s', a' \mid \theta^{Q'}) - Q(s, a \mid \theta^{Q})\right)^2\right]$
      Periodically update target network parameters $\theta^{Q'} \leftarrow \theta^{Q}$
    end for
  end for
end function

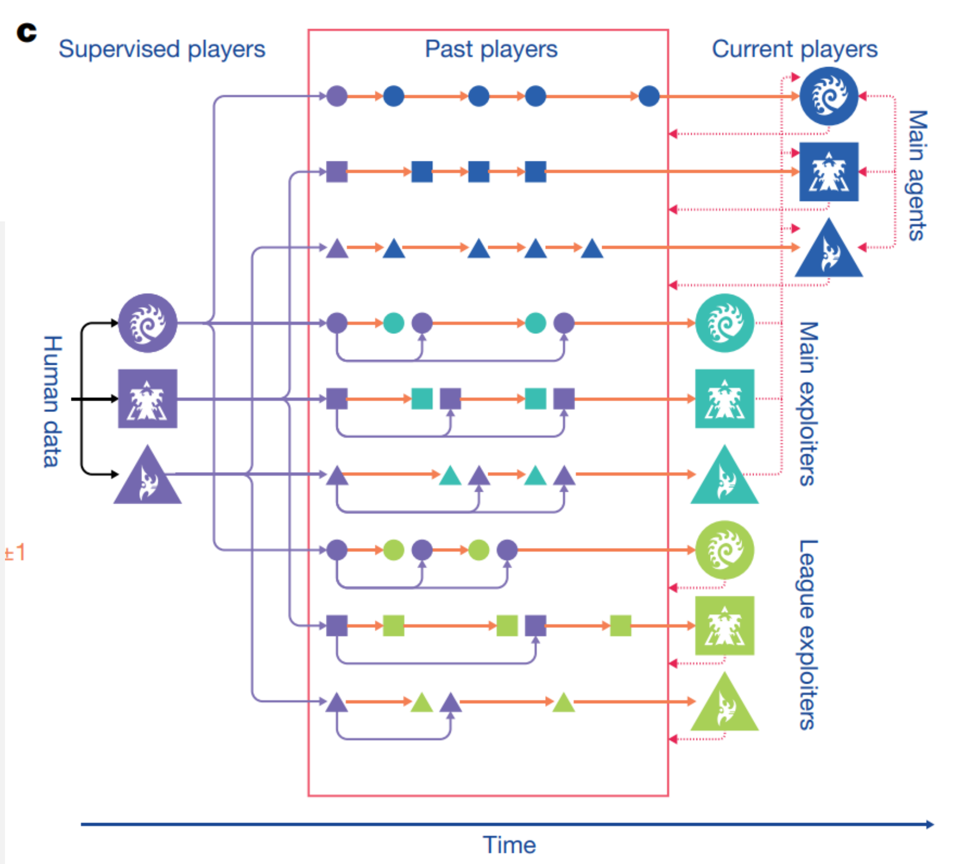


2.5.3 现有主要非完全信息博弈算法——基于种群



Algorithm 1: Policy-Space Response Oracles

input :initial policy sets for all players Π
Compute exp. utilities U^{Π} for each joint $\pi \in \Pi$
Initialize meta-strategies $\sigma_i = \textsc{Uniform}(\Pi_i)$
while *epoch* e *in* $\{1, 2, \cdots\}$ **do**
  for *player* $i \in [[n]]$ **do**
    for *many episodes* **do**
      Sample $\pi_{-i} \sim \sigma_{-i}$
      Train oracle π'_i over $\rho \sim (\pi'_i, \pi_{-i})$
    $\Pi_i = \Pi_i \cup \{\pi'_i\}$
  Compute missing entries in U^{Π} from Π
  Compute a meta-strategy σ from U^{Π}
Output current solution strategy σ_i for player i



2.6 展望

多人博弈待解决

算法解决的核心问题侧重点不同

慢，未出现高效分布式算法