第二章 对抗搜索

- 对抗搜索：博弈

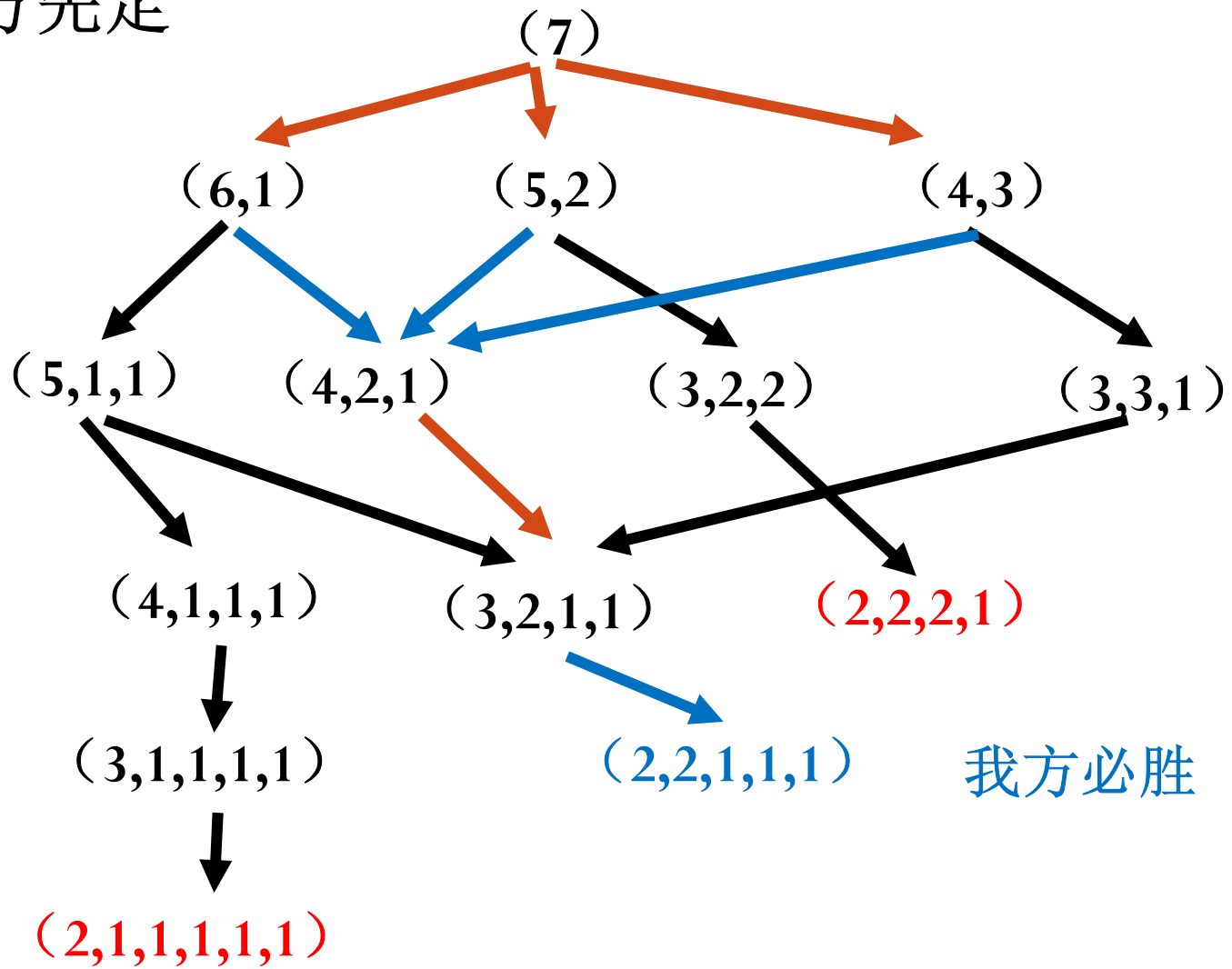
- 博弈问题
- 极小极大方法
- α-β剪枝
- 蒙特卡洛博弈方法

2.1 博弈问题

- 博弈问题
  - 双人
  - 一人一步
  - 双方信息完备
  - 零和

分钱币问题

对方先走


(7)
(6,1)
(5,2)
(4,3)
(5,1,1)
(4,2,1)
(3,2,2)
(3,3,1)
(4,1,1,1)
(3,2,1,1)
(2,2,2,1)
(3,1,1,1,1)
(2,2,1,1,1)
我方必胜
(2,1,1,1,1,1)

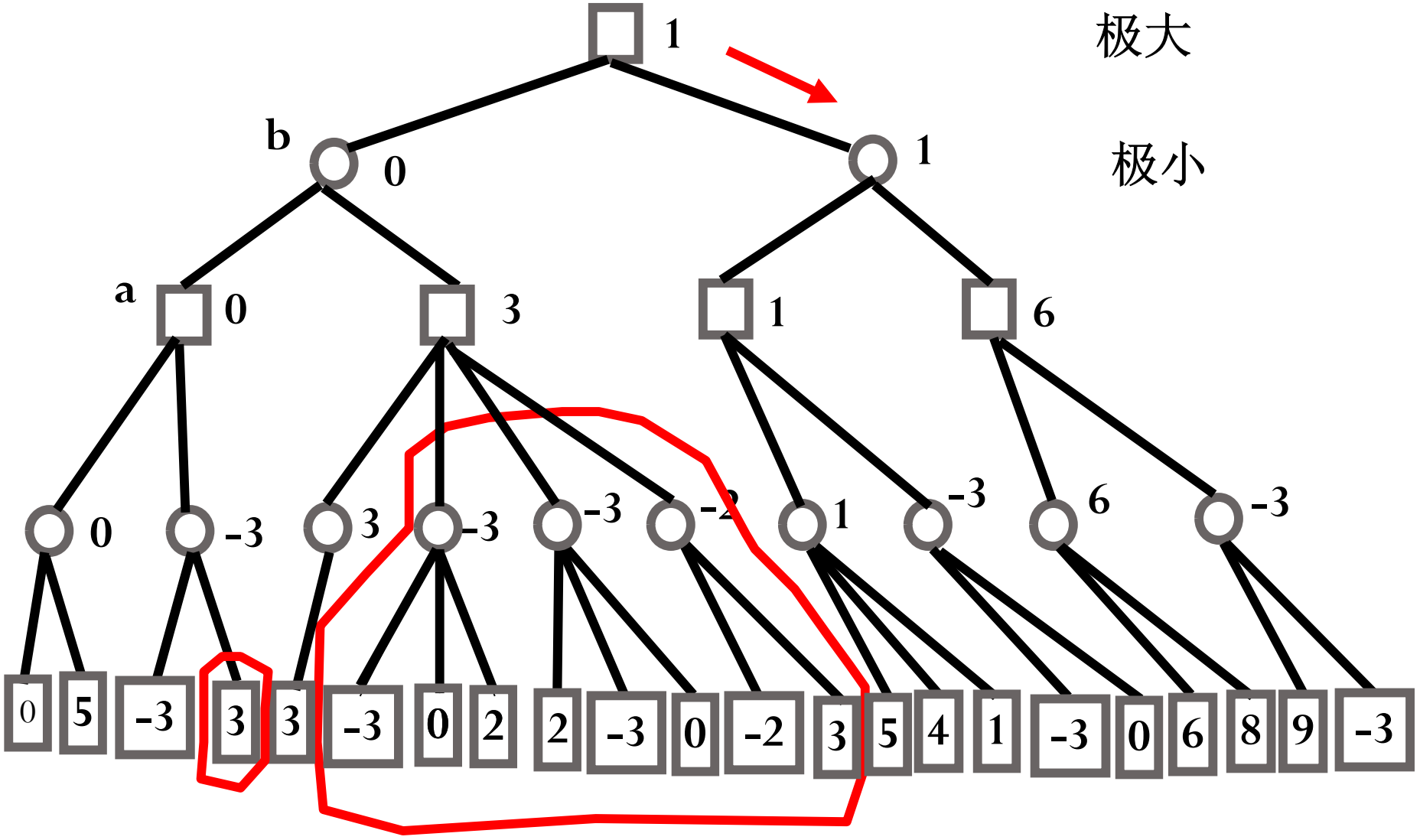
中国象棋

- 一盘棋平均走50步，总状态数约为10^{161}。
- 假设1毫微秒走一步，约需10^{145}年。

- 宇宙年龄：$1.38*10^{10}$

- 结论：不可能穷举。

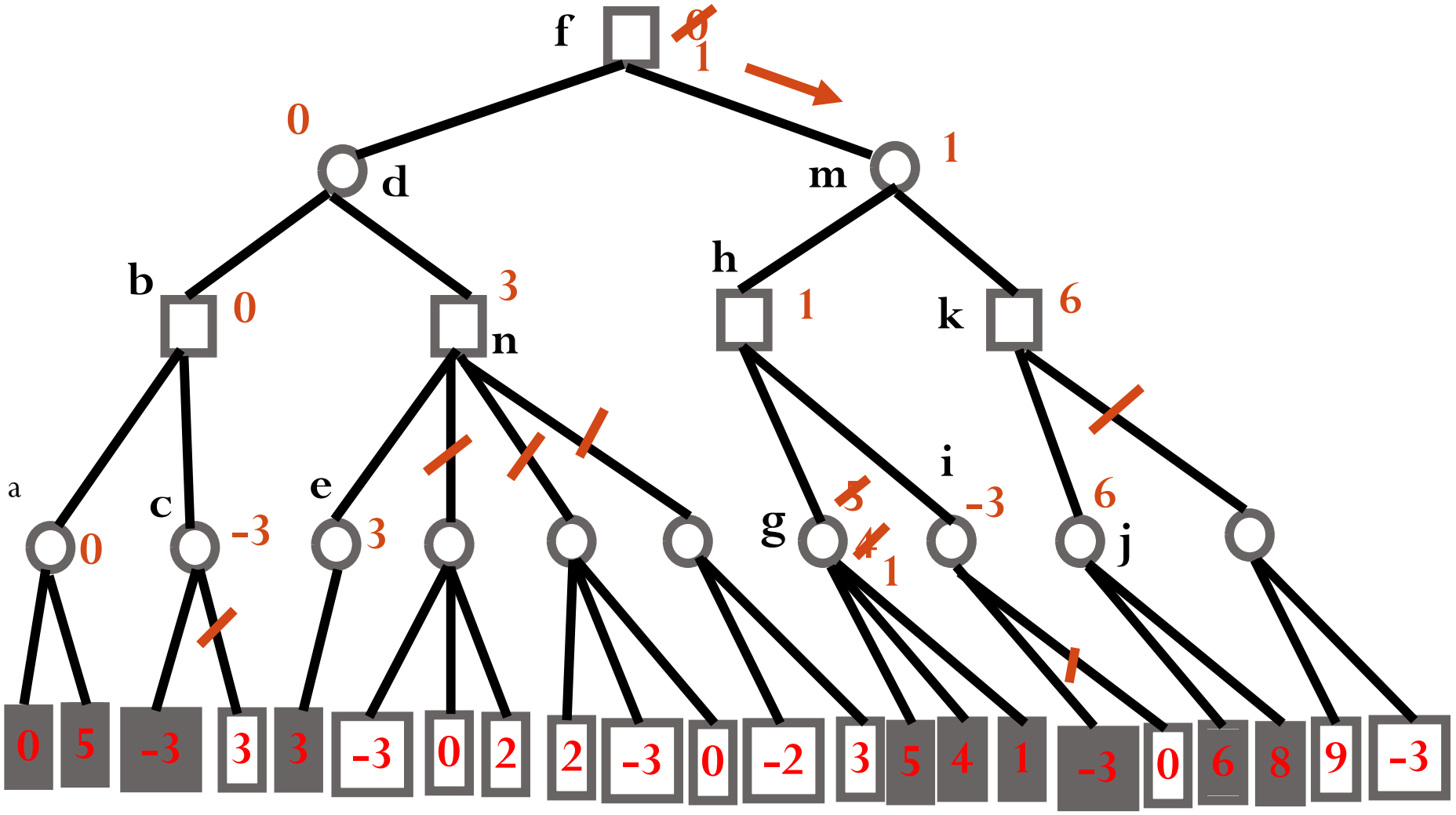
2.2 极小极大过程

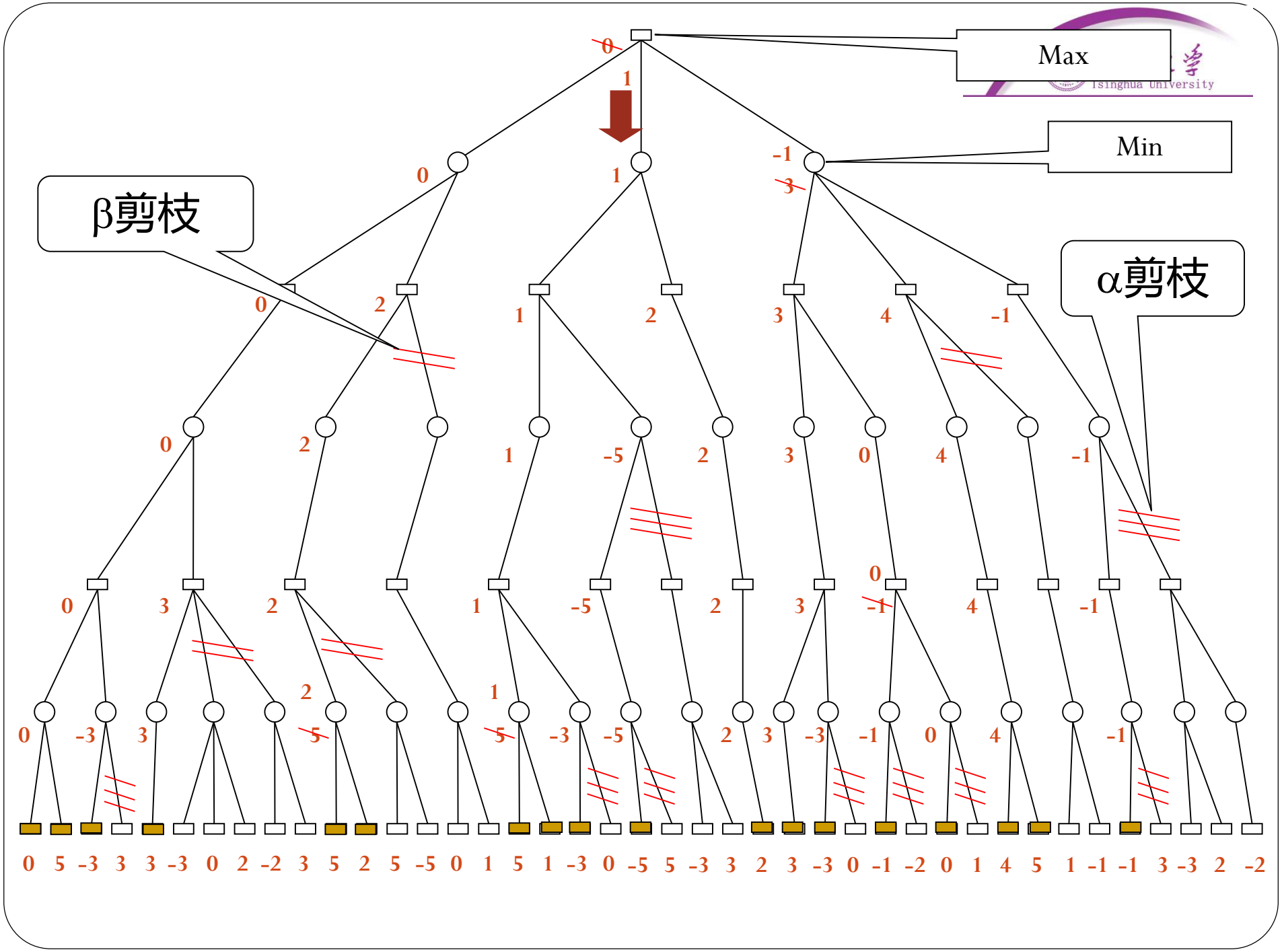


2.3 α-β剪枝

- 极大节点的下界为α。
- 极小节点的上界为β。
- 剪枝的条件：
  - 后辈节点的β值≤祖先节点的α值时， α剪枝
  - 后辈节点的α 值≥祖先节点的β值时， β剪枝
- 简记为：
  - 极小≤极大，剪枝
  - 极大≥极小，剪枝

α-β剪枝（续）




Max
Min
β剪枝
α剪枝
0 5 -3 3 3 -3 0 2 -2 3 5 2 5 -5 0 1 5 1 -3 0 -5 5 -3 3 2 3 -3 0 -1 -2 0 1 4 5 1 -1 -1 3 -3 2 -2

练习题

- 上述例题自己独立做一遍。

2.4 蒙特卡洛博弈方法

- 为什么α-β剪枝方法在围棋上失效？
  - α-β剪枝方法存在的问题
    - 依赖于局面评估的准确性
  - 局面评估问题
    - 大量专家知识
    - 知识的统一性问题
    - 人工整理

围棋落子模型

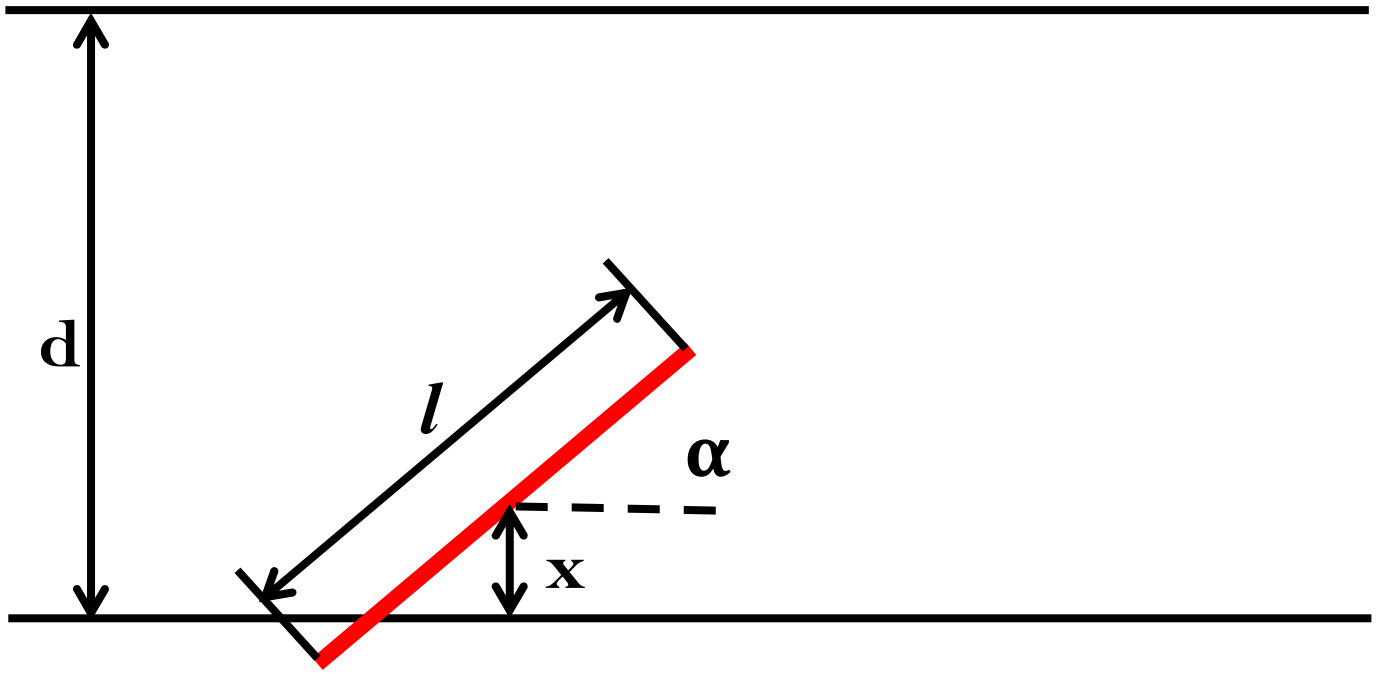
- 围棋对弈过程可看做一个马尔科夫过程：
- 五元组：{T，S，A(i)，P(a|i)，r(i,a)}
  - T：决策时刻
  - S：状态空间，S={i}
  - A(i)：可行动集合（可落子点）
  - P(a|i)：状态i下选择行动a的概率
  - r(i,a)：状态i下选择行动a后课获得的收益

蒙特卡洛方法

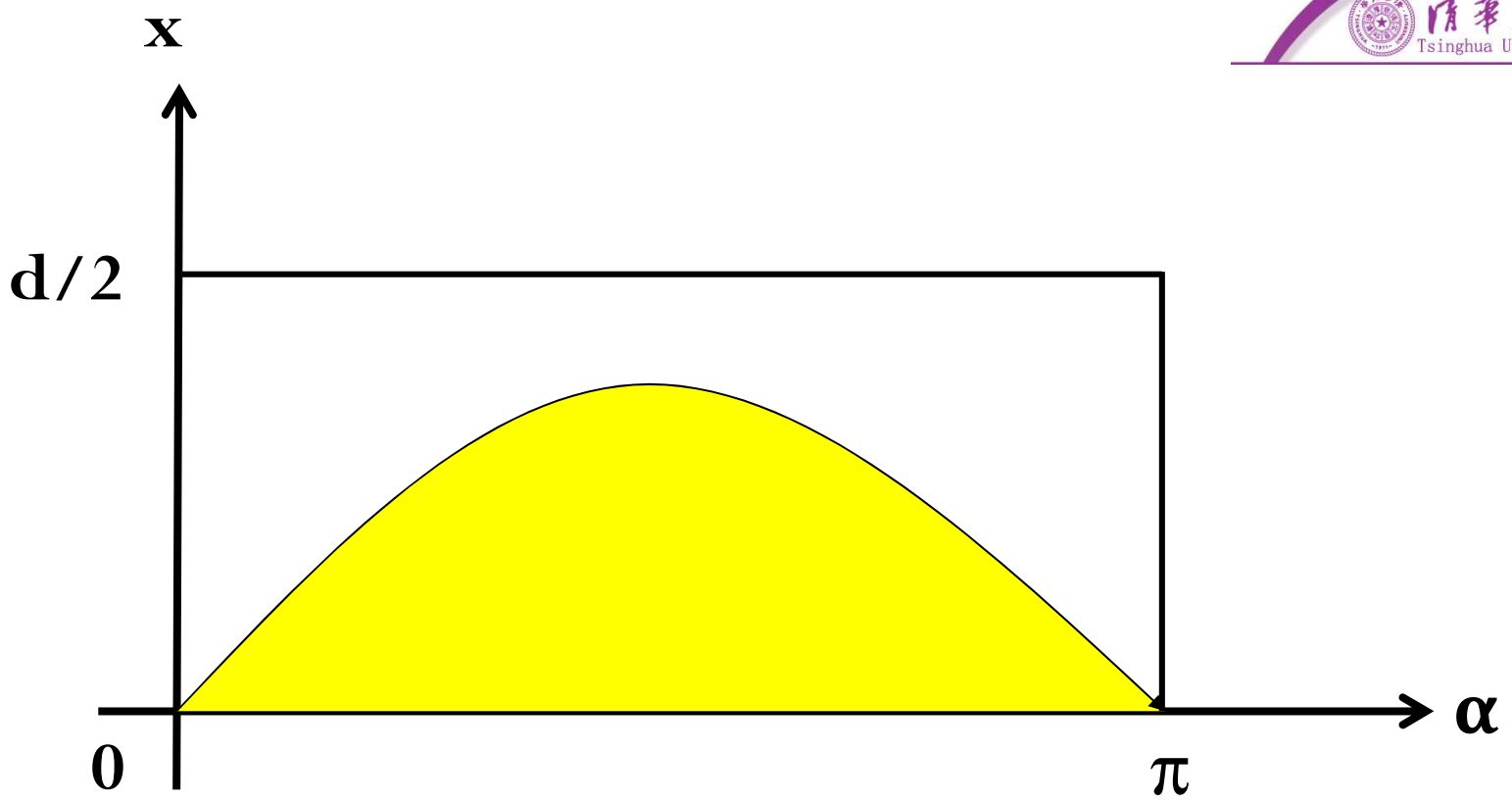
- 二十世纪40年代中期S.M.乌拉姆和J.冯·诺伊曼提出的一种随机模拟方法
  - 多重积分
  - 矩阵求逆
  - 线性方程组求解
  - 积分方程求解
  - 偏微分方程求解
  - 随机性问题模拟

蒲丰投针问题

- 1777年法国科学家蒲丰提出一种计算π的方法：
- 取一张白纸，在上面画上许多条间距为d的等距平行线，另取一根长度为l（l<d）的针，随机地向该纸上投掷针，并记录投掷次数n以及针与直线相交的次数m，据此计算π值。



- (x, α)决定了针的位置
- 针与直线的相交条件：$x \le (l/2)\cdot\sin\alpha$
- 其中：$x \in [0, d/2], \alpha \in [0, \pi]$



- 黄颜色部分与长方形面积之比即为针与直线相交的概率

$$P = \frac{\int_0^{\pi} \frac{l}{2}\sin\alpha \, d\alpha}{\frac{d}{2}\pi} = \frac{2l}{\pi d}$$

$$\pi = \frac{2l}{Pd} \approx \frac{2nl}{md}$$

n： 投掷次数

m： 针与直线相交的次数

蒙特卡洛评估

- 从当前局面的所有可落子点中随机选择一个点落子
- 重复以上过程
- 直到胜负可判断为止
- 经多次模拟后，选择胜率最大的点落子

蒙特卡洛树搜索

- 解决马尔科夫决策问题的有效方法之一
- 基本思想与特点：
  - 将可能出现的状态转移过程用状态树表示
  - 从初始状态开始重复抽样，逐步扩展树中的节点
  - 某个状态再次被访问时，可以利用已有的结果，提高了效率
  - 在抽样过程中可以随时得到行为的评价

蒙特卡洛树搜索的步骤

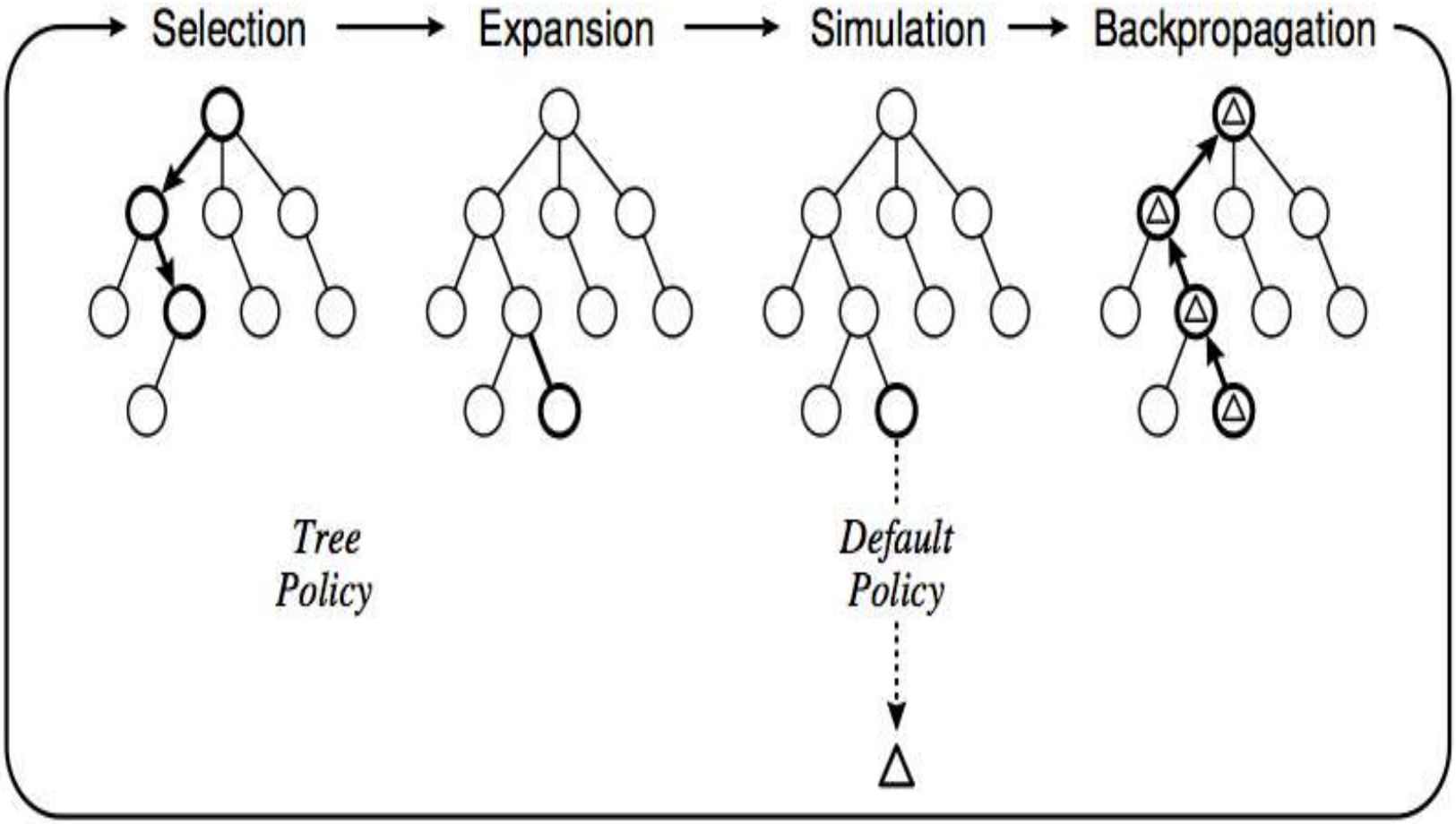
- 选择
  - 从根节点出发自上而下地选择一个落子点
- 扩展
  - 向选定的点添加一个或多个子节点
- 模拟
  - 对扩展出的节点用蒙特卡洛方法进行模拟
- 回溯
  - 根据模拟结果依次向上更新祖先节点估计值

更新过程

- 设n_i为当前要模拟的节点，△为模拟获得的收益
- 对n_i及其祖先的模拟次数加1
- n_i的收益加△
- 更新n_i的同类祖先节点的收益

（这里节点的类型按照极大极小节点划分）

蒙特卡洛树搜索流程


Selection
Expansion
Simulation
Backpropagation
Tree Policy
Default Policy

选择落子点的策略

- 两方面的因素：
  - 对尚未充分了解的节点的探索
  - 对当前具有较大希望节点的利用

清華大學 Tsinghua University

多臂老虎机模型

多臂老虎机模型

- 1952年Robbins提出的一个统计决策模型
- 多臂老虎机
  - 多臂老虎机拥有k个手臂，拉动每个手臂所获得的收益遵循一定的概率且互不相关，如何找到一个策略，使得拉动手臂获得的收益最大化
- 用于解决蒙特卡洛树搜索中选择落子点的问题

信心上限算法UCB1

- **function UCB1**
- **for each 手臂j:**
- 访问该手臂并记录收益
- **end for**
- **while 尚未达到访问次数限制 do:**
- 计算每个手臂的UCB1信心上界I_j
- 访问信心上界最大的手臂
- **end while**

$$I_j = \overline{X}_j + \sqrt{\frac{2\ln(n)}{T_j(n)}}$$

- 其中：
- $\overline{X}_j$ 是手臂j所获得回报的均值
- ***n***是到当前这一时刻为止所访问的总次数
- $T_j(n)$ 是手臂j到目前为止所访问的次数

- 上式考虑了“利用”和“探索”间的平衡

信心上限树算法UCT

- 将UCB1算法应用于蒙特卡洛树搜索中，用于选择可落子点
  - 可落子点不是随机选择，而是根据UCB1选择信心上限值最大的节点
  - 实际计算UCB1时，加一个参数c进行调节：

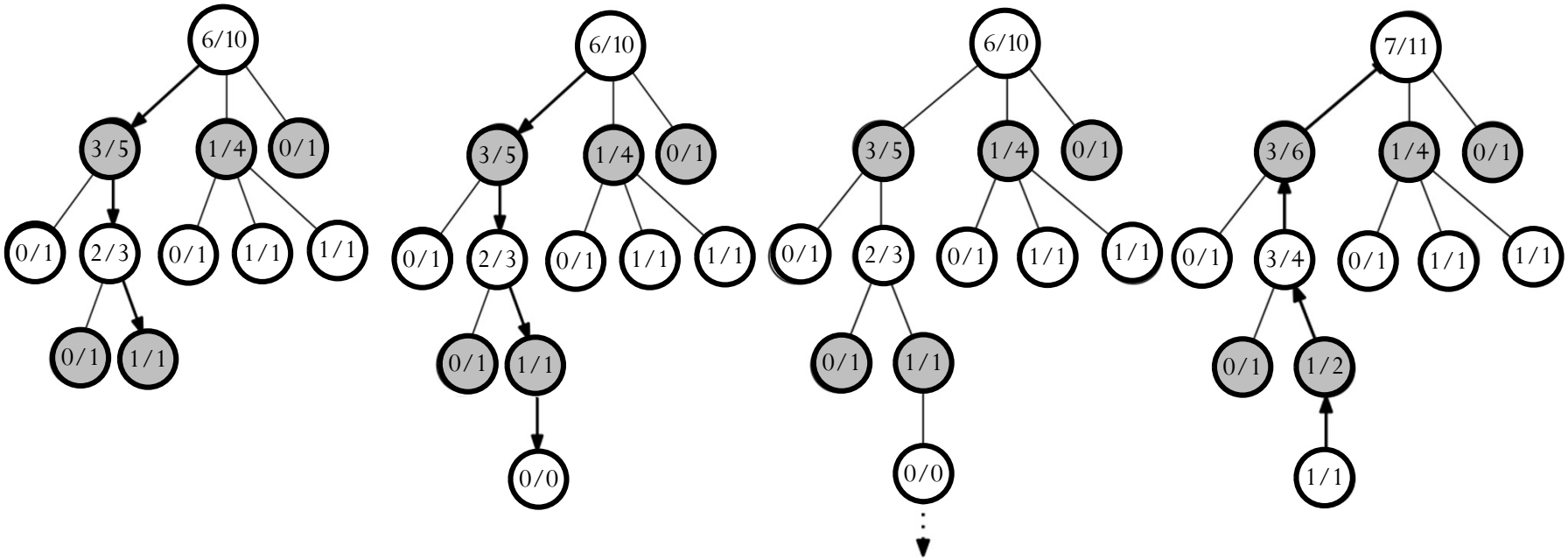
$$I_j = \overline{X}_j + c\sqrt{\frac{2\ln(n)}{T_j(n)}}$$

- 引入符号:
- v: 节点，包含以下信息:
  - s(v): v对应的状态
  - a(v): 来自父节点的行为
  - Q(v): 随机模拟获得的收益
  - N(v): v的总访问次数

- 信心上限树算法（UCT）
- function UCTSEARCH(s_0)
- 以状态s_0创建根节点v_0;
- while 尚未用完计算时长 do:
- v_1 = TREEPOLICY(v_0);
- △ = DEFAULTPOLICY($s(v_1)$);
- BACKUP(v_1, △);
- end while
- return a(BESTCHILD(v_0, 0));

- 全部算法的伪代码，请见课程资料

UCT算法示例

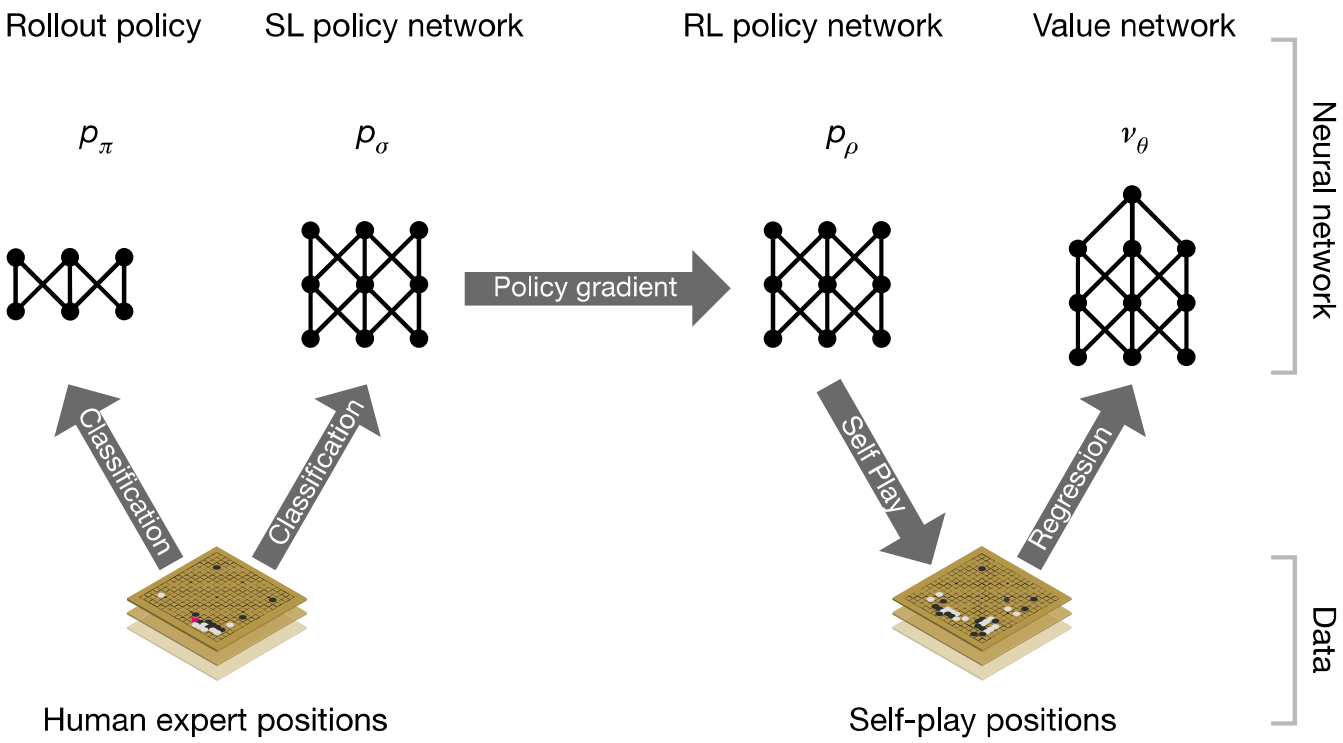


节点：获胜次数/模拟总次数

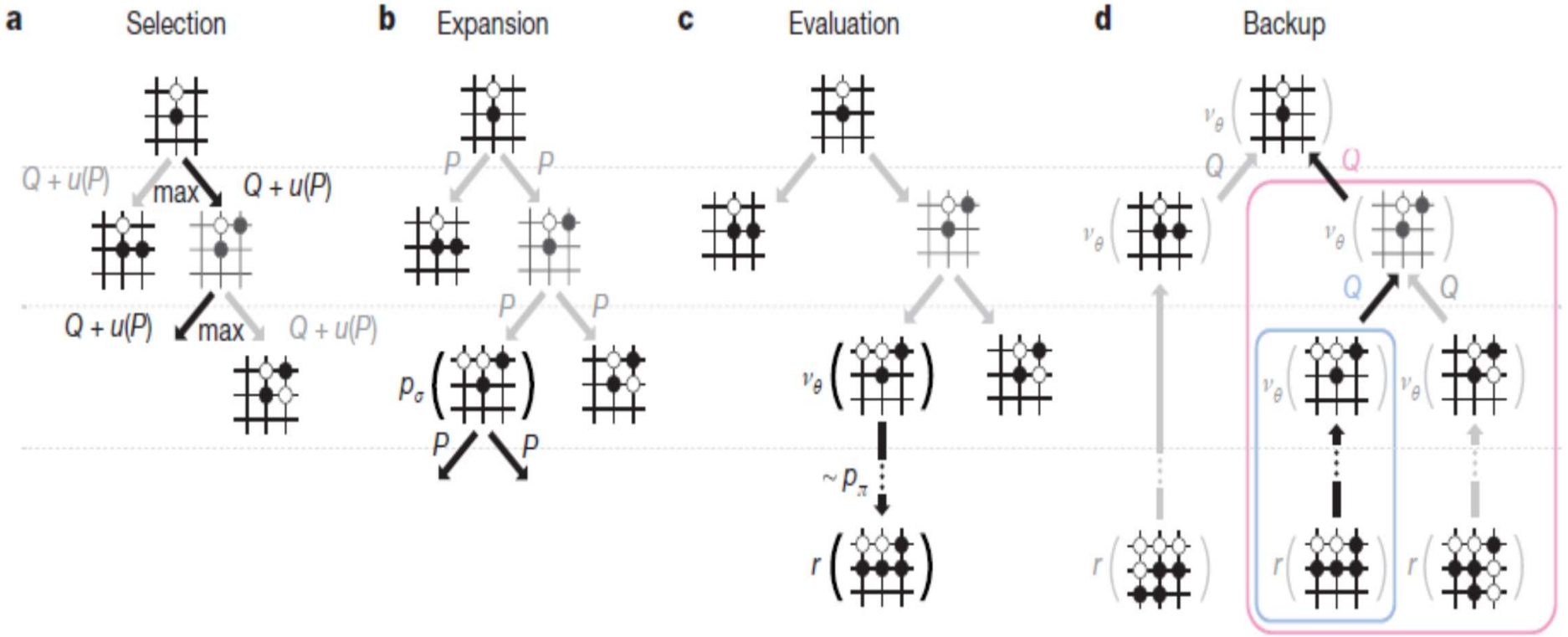
获胜次数是从本节点角度说的

假设c=0，即 $I_j = \overline{X}_j$

AlphaGo


Rollout policy
SL policy network
RL policy network
Value network
p_π
p_σ
p_ρ
v_θ
Policy gradient
Classification
Classification
Self Play
Regression
Neural network
Data
Human expert positions
Self-play positions

AlphaGo


a
Selection
Q + u(P)
max
Q + u(P)
Q + u(P)
max
Q + u(P)
b
Expansion
P
P
P
P
P
P
c
Evaluation
~ p_π
r
d
Backup
Q
Q
Q
Q
r
r
r

AlphaGo

- 利用策略网络缩小搜索的范围
- 将估值网络的结果结合到信心上限的计算中
- 一个节点被模拟一定次数后才扩展
- 最终选择模拟次数最多的节点为最佳走步

小结

- 极小极大方法
- α-β剪枝
  - 剪枝的目的
  - 什么情况下可以剪枝
- 蒙特卡洛树搜索
  - 采用蒙特卡洛树搜索的目的
  - 蒙特卡洛树搜索过程