人工智能基础 HW3

PB18111697 王章瀚

2021 年 4 月 23 日

5.9

a.

至多 9! 种, 但可能在这之前就已经分出胜负.

b.

如下图所示:

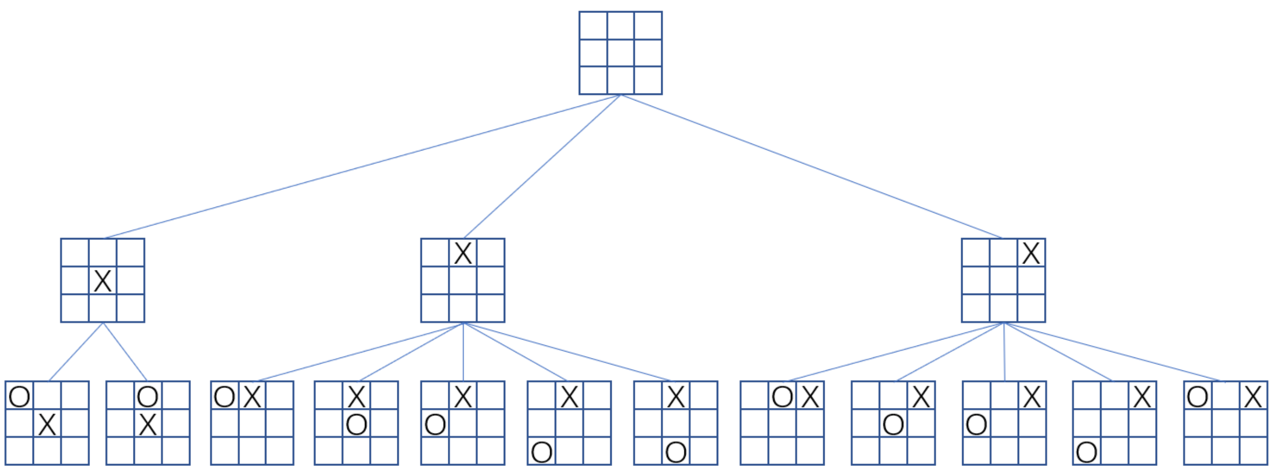

图 1: 5.9.b game tree starting from an empty board down to depth 2

c.

evaluation均已标在图中:

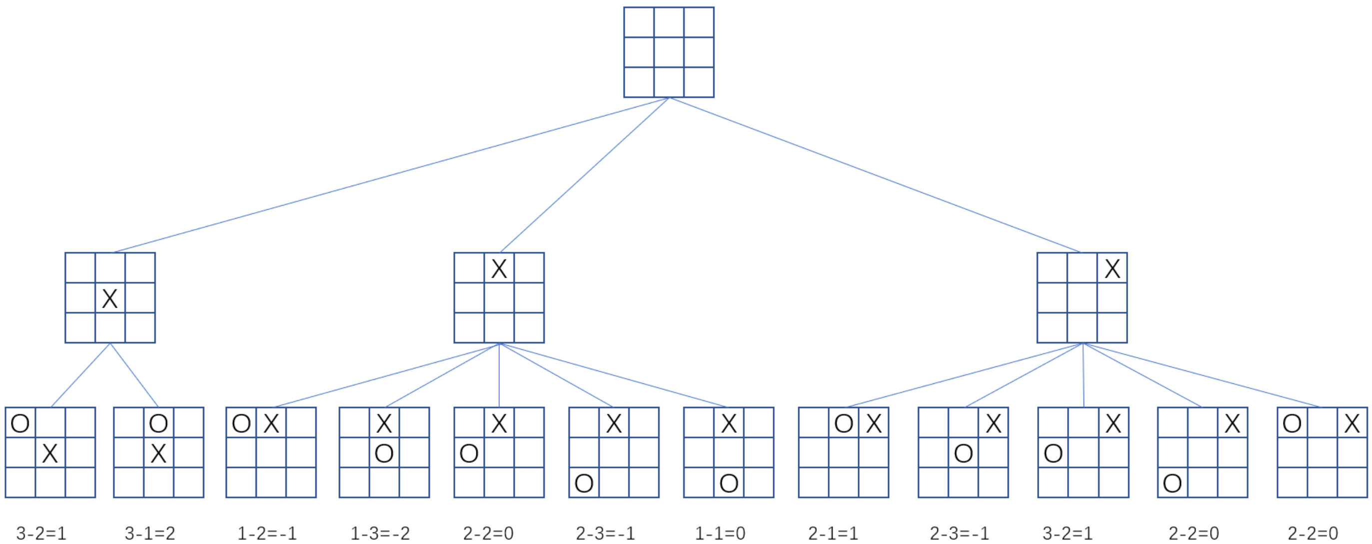

图 2: 5.9.c marked with evaluations

d.

使用 minmax 算法结果如下:

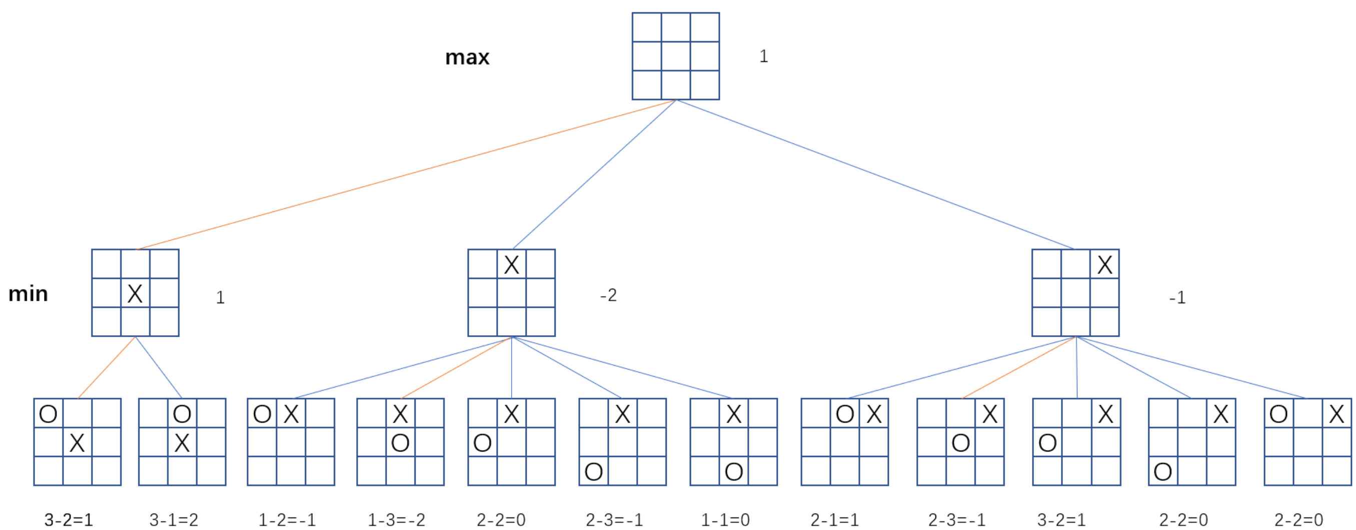

图 3: 5.9.d use minmax algorithm

因此最佳的初始选择是: 把 X 放在棋盘中间的位置.

e.

如果顺序是最优的, 那么将会有如下绿色框出的节点不会被测试:

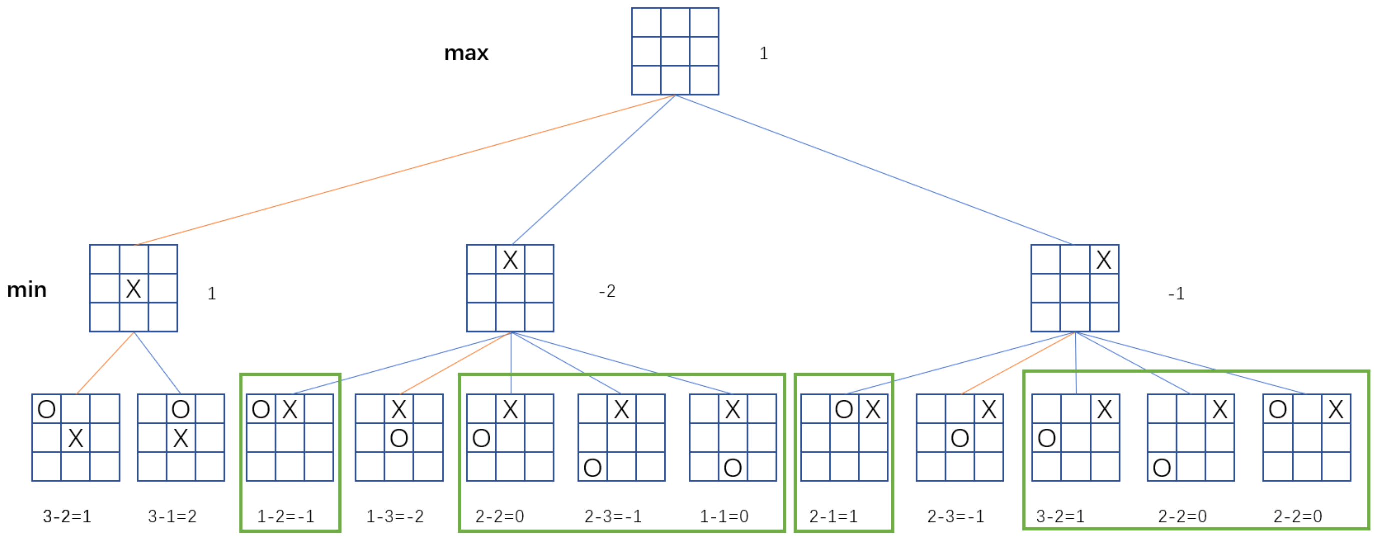

图 4: 5.9.e

5.8

a.

完整的 game tree 如下图所示:

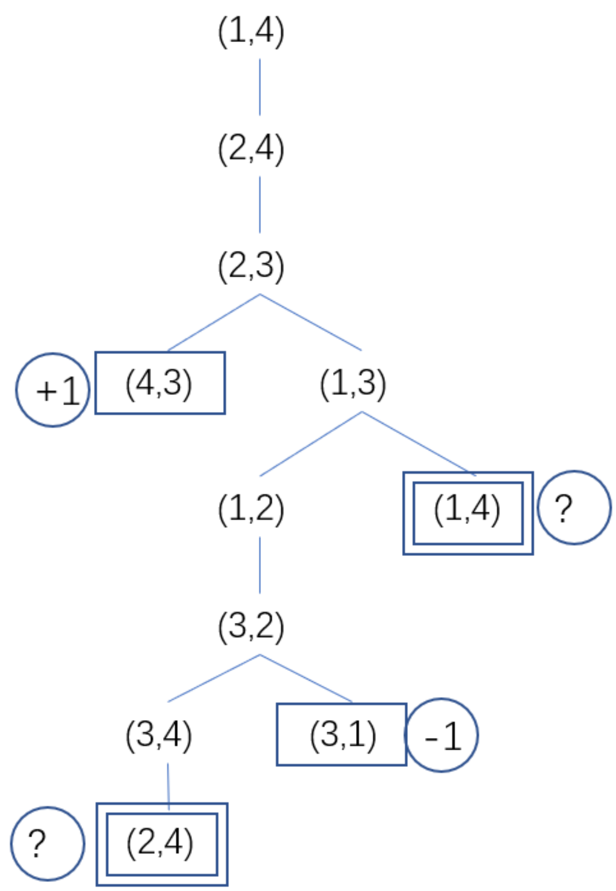

图 5: 5.8.a

b.

- 当面对 +1 和 ? 的时候, 显然选择 +1 的路径能赢, 即 max(+1,?)=+1, min(+1,?)=?
- 当面对 -1 和 ? 的时候, 选择 -1 一定不能赢, 但选择 ? 则还有希望能赢, 因此这时要选择 ?, 即 max(-1,?)=?, min(-1,?)=-1
- 当面对的子节点只有一个 ?, 就只能选它
- 其他组合情况未出现, 故不予定义

使用 minmax 算法后如下所示:

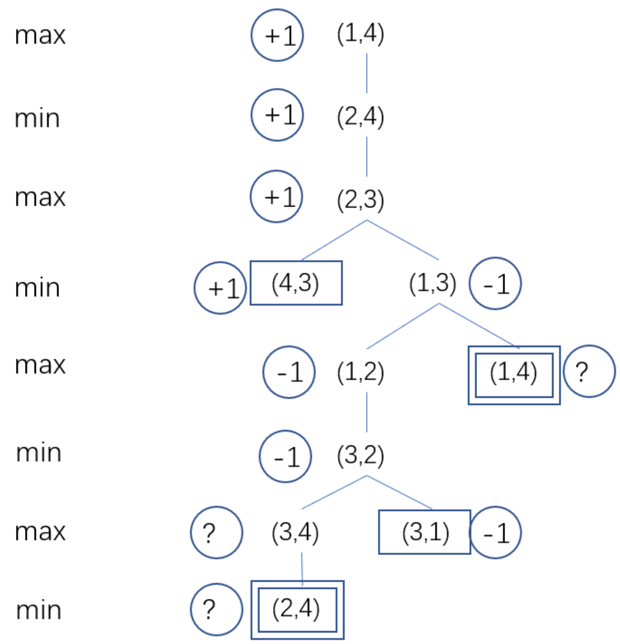

图 6: 5.8.b

c.

因为运用标准的 minmax 算法会导致陷入死循环. 解决方案是, 将已经访问过的状态放进一个集合里, 每检查一个新节点的时候要首先看一下是否是已经处理过的节点, 如果是的话, 就标记为 ?, 并按 (b) 所述方法处理 ?.

这种方法并不能区分所有带 loop 的游戏, 因为在本题中, loop 出现的时候能够很好地解释 ? 的语义, 并给出应该选择 ? 还是另一个节点, 类似于一种排除法. 但其他游戏则不一定能够精确地定义好 ? 的语义.

d.

可以考虑数学归纳法

1. 据前述, 当 $n = 4$, 先行者胜. 假设 $n \leq 2k$ 时, n 为偶数则先行者胜, n 为奇数则后行者胜.
2. 若 $n = 2k + 1$, 则先行者只能有一种走法, 那么变成 $n = 2k$ 的情况, 这时, 相当于对手先行的偶数情况, 所以一定是对手胜.
3. 若 $n = 2k + 2$, 则先行者走完相当于对手面临 $2k + 1$ 的局势, 因为是奇数, 所以对手一定会输, 也就是先行者胜

综上命题得证.

5.13

a.

对于 n_2:

$$n_2 = \max(n_3, n_{31}, \cdots, n_{3b_3})$$

进而 n_1 有:

$$n_1 = \min(\max(\min(..., n_{41}, n_{4b_4}), n_{31}, \cdots, n_{3b_3}), n_{21}, n_{2b_2})$$

其中最内层嵌套的 $\cdots$ 可以不断地用相应的 n_j 替代, 直到最底层 n_j.

b.

则有:

$$n_1 = \min(l_2, \max(l_3, ..., r_3), r_2)$$

其中最内层嵌套的 $\cdots$ 可以递归地写下去直到最底层的 $\max(l_j, n_j, r_j)$

c.

- 对于极小值(i 为偶数)的 l_i, 为了让 n_j 有效, 它必须满足 $n_j < l_i$, 否则不如选 l_i 对应的;
- 对于极大值(i 为奇数)的 l_i 也类似, $n_j > l_i$

而 n_j 是一个 max 节点, 因此一定要有 $\max(l_3, l_5, \cdots, l_{j-1}) < n_j < \min(l_2, l_4, \cdots, l_j)$

d.

则有

$$n_1 = \min(l_2, \max(l_3, ..., r_3), r_2)$$

其中一直写下去最内层是 $\min(l_j, n_j, r_j)$ 从而

- 对于极小值(i 为偶数)的 l_i, 为了让 n_j 有效, 它必须满足 $n_j < l_i$, 否则不如选 l_i 对应的;
- 对于极大值(i 为奇数)的 l_i 也类似, $n_j > l_i$

而 n_j 是一个 max 节点, 因此一定要有 $\max(l_3, l_5, \cdots, l_j) < n_j < \min(l_2, l_4, \cdots, l_{j-1})$