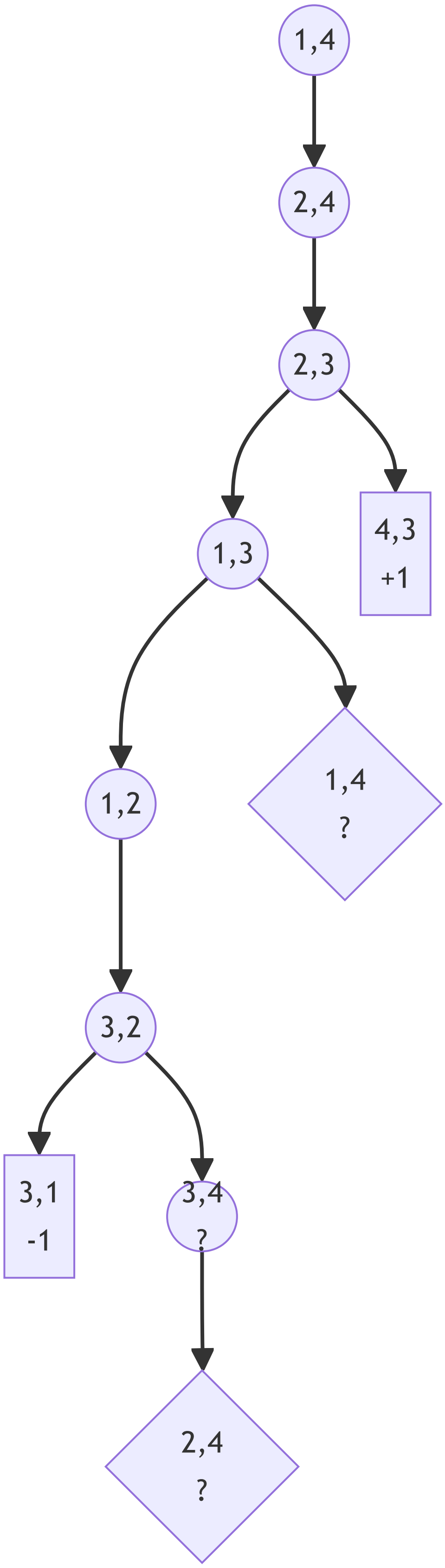
HW4

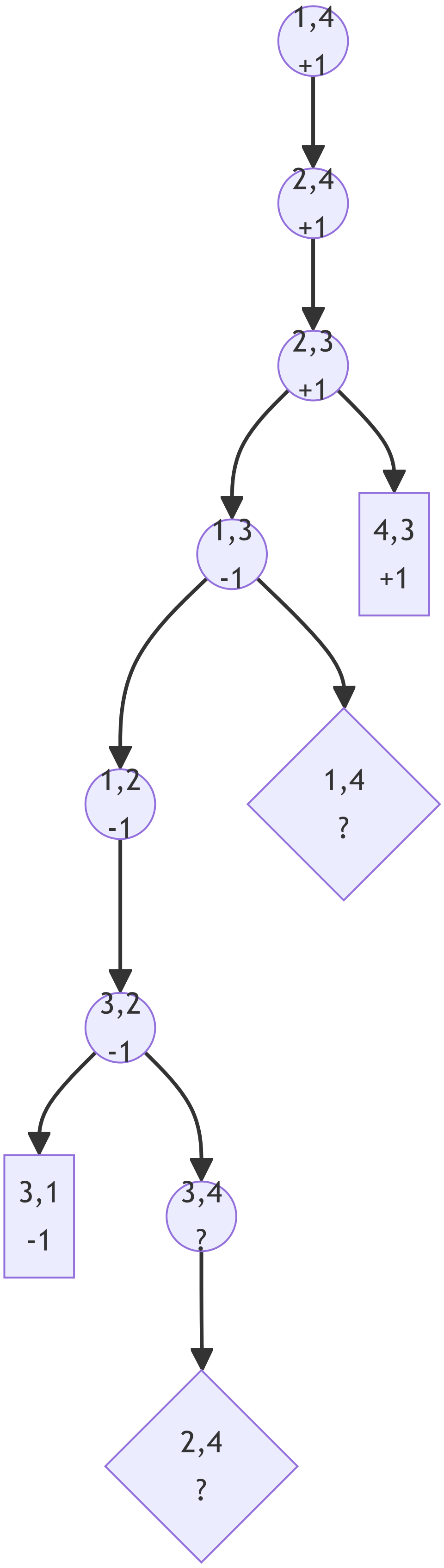
PB21111686_赵卓

5.8

- a.博弈树如下，菱形节点表示循环状态：



- b.
    - 标记极大极小值的博弈树如下：



- 对于该例，MINMAX计算公式中的“?”，如果当前结点s是MAX结点，那么MINMAX(s)=max(+1,?)=+1；如果当前结点s是MIN结点，那么MINMAX(s)=min(-1,?)=-1。这样处理的原因是，在该例中，“?”和+1或-1一起出现，不管“?”取什么值，结果都是确定的。

- c.

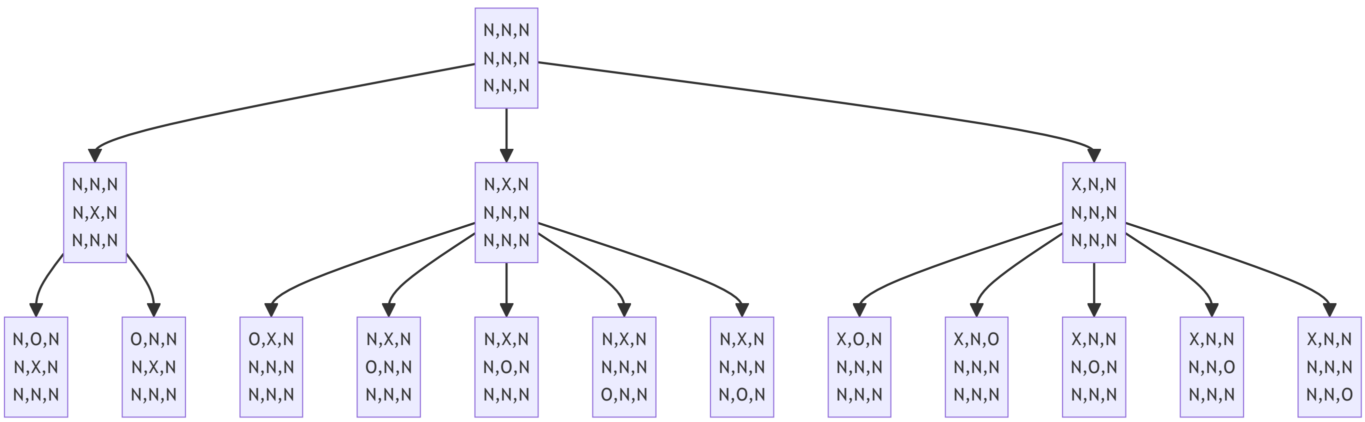
- 标准的极大极小算法默认不存在循环状态，每条路径都会到达终止状态得到评估值，以此反推得到极大极小值。但是题给情况存在循环，因此无法处理。对于循环状态则返回“?”，对于涉及“?”的MINMAX计算
- 对于改进方法，遇到循环状态则返回“?”，涉及“?”的MINMAX计算如b中所述。
- 显然这个方法无法处理所有具有循环状态的情况。这是因为本例较为特殊，对于MAX结点，“?”都是和+1一起出现；对于MIN结点，“?”都是和-1一起出现。而当对于MAX结点，“?”和-1一起出现而没有+1时，这个时候就要考虑“?”的取值了，MIN结点同理。另外，本例中只有+1和-1两个博弈值，对于多种博弈值的情况，则必须考虑“?”的具体取值而无法处理。

- d.证明如下：
现在我们假设A和B都很“聪明”，都以到达自己的目的地为最终目标，只会走当前的最佳选择。也就是说A能往右跳就往右跳，B能往左跳就往左跳，只有这样我们讨论A或B必胜才有意义。
在这种假设前提下我们讨论n为奇数和偶数的情况：
  - 若n=2k(k为自然数)
  则A在k，B在k+1时，AB相遇，且下一步A走。因此下两步A会选择往右跳到k+2上，B会向左跳到k上。这时A距离目的地还有k-2步，而B还有k-1步。且下一步A跳，因此A一定先到目的地，A必胜。
  - 若n=2k+1(k为自然数)
  同上，A在k+1，B在k+2时，AB相遇。且下一步B走。因此下两步B会选择往左跳到k上，A会向右跳到k+2上。这时A距离目的地还有k-1步，B也有k-1步。但是下一步B跳，因此B一定先到目的地，B必胜。

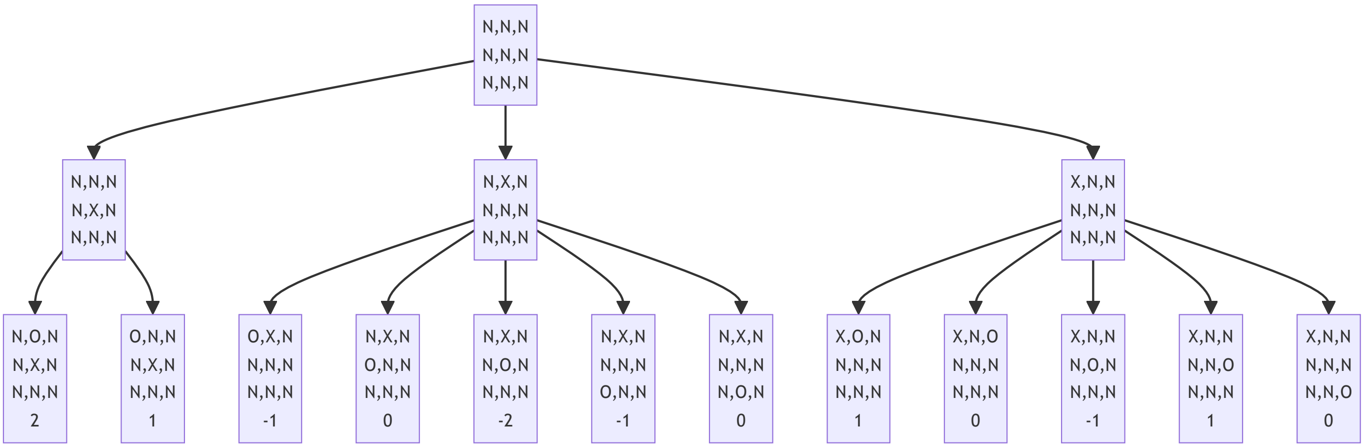
5.9

- a.可能的棋局数为9!局。

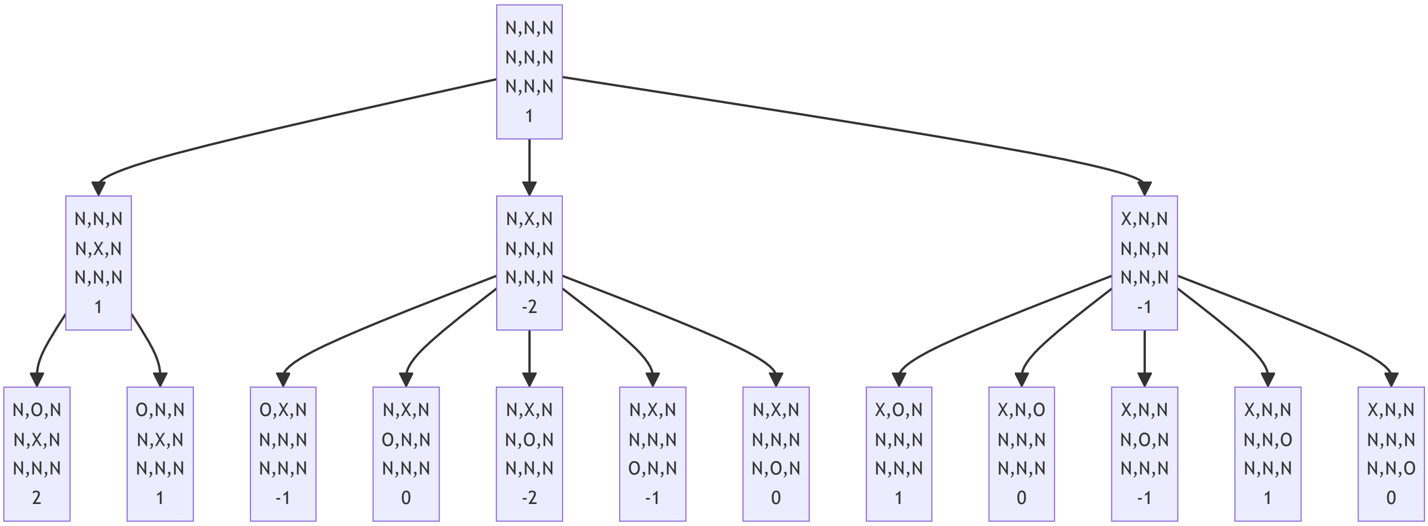
- b.假设X先走，N表示当前位置为空(NULL)，在考虑对称性的情况下，博弈树如下：



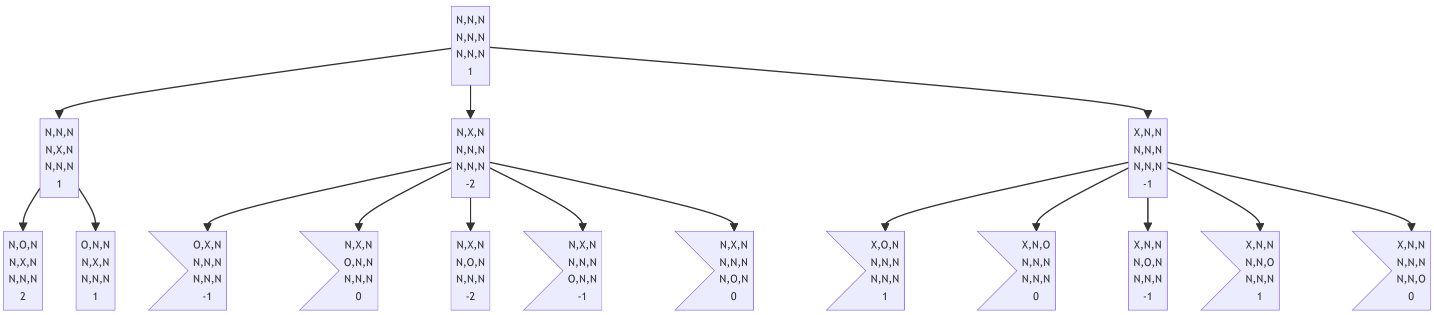
- c.评估函数值如下所示：



- d.深度为0和1的倒推值如下，由此可知最佳开始行棋是在中心落子：



- e.按照最优顺序，被剪枝的节点用右向旗帜标识，好像被“剪”去了：



5.13

- a.由题可得$n_2 = max(n_3, n_{31}, \ldots, n_{3b_3})$
  可得$n_1 = min(max(n_3, n_{31}, \ldots, n_{3b_3}), n_{21}, \ldots, n_{2b_2})$
  由此递推得$n_1 =$

$min(max(...max(min(n_j, n_{j_1}, ..., n_{jb_j}), n_{(j-1)1,...,n_{(j-1)b_{(j-1)}}})...), n_{21}, ..., n_{2b_2})$

- b.与a无本质差别
  可得$n_1 = min(l_2, n_2, r_2)$，$n_2 = max(l_3, n_3, r_3)$
  由此递推得$n_1 = min(l_2, max(...max(l_{(j-1)}, min(l_j, n_j.r_j), r_{j-1})...), r_2)$

- c.由b中的形式化表达可得，n_j不能超过$\{l_2, l_4 ..., l_j\}$

- d.若n_j是MIN结点，只需稍加修改：
  - d.a.$n_1 = min(max(...min(max(n_j, n_{j_1}, ..., n_{jb_j}), n_{(j-1)1,...,n_{(j-1)b_{(j-1)}}})...), n_{21}, ..., n_{2b_2})$
  - d.b.$n_1 = min(l_2, max(...min(l_{(j-1)}, max(l_j, n_j.r_j), r_{j-1})...), r_2)$
  - d.c.n_j不能小于$\{r_2, r_4 ..., r_j\}$