**Exercise 5.2**

Consider the problem of solving two 8-puzzles: there are two 8-puzzle boards, you can make a move on either one, and the goal is to solve both.

a. Give a complete problem formulation.

b. How large is the reachable state space? Give an exact numerical expression.

c. Suppose we make the problem adversarial as follows: two players take turns moving; a coin is flipped to determine the puzzle on which to make a move in that turn: heads puzzle one; tails puzzle two. The winner is the first to solve either puzzle. Which algorithm can be used to choose a move in this setting?

d. In the adversarial problem, will one player eventually win if both play perfectly?

1. **· 初始状态：**两个任意的8-puzzle状态

**· 后继函数：**在未解决的8-puzzle中随机选择一个进行空格的移动，包括Left、Right、Up或Down动作，但如果空格位于边缘或角落，则不是所有动作可用。

**· 目标测试：**两个8-puzzle的滑块是否都到达目标位置

**· 动作代价：**每次移动滑块代价都为1。

1. 单个8-puzzle的可达状态数为=181,440

÷2是因为通过合法移动（滑动数字到相邻空格）无法改变排列的​**​逆序数奇偶性​**

双个8-puzzle的可达状态数为 =1814402 = 32,921,779,200

1. 可以使用**期望极小化极大算法**
2. 这取决于初始状态的情况：

如果初始时两个8-puzzle都只差一步解决，那么优先抛硬币的玩家获胜。

如果初始时只有一个8-puzzle差一步解决，那么获胜者将是抛硬币最先抛中该puzzle的玩家。

如果初始时两个8-puzzle都比较复杂（需要大于一步解决）时，假设某一时刻玩家1抛中了一个只剩两步解决的puzzle，如果他选择靠近目标一步，那么对手获胜的概率将是1/2+1/4+1/16……=2/3，自己获胜概率为1/3，所以最佳行动应该为远离目标一步，按这样考虑，表现完美的玩家之间不可能分出胜负。

**Exercise 5.7**

Prove the following assertion:

For every game tree, the utility obtained by max using minimax decisions against a suboptimal min will never be lower than the utility obtained playing against an optimal min.

Can you come up with a game tree in which max can do still better using a suboptimal strategy against a suboptimal min?

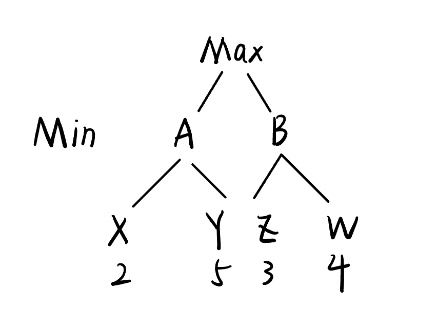
**证明**：考虑一个MAX节点，其有n个子MIN节点，这n个MIN节点的子节点均为叶节点。

设第i个MIN节点的叶节点中最低效用为Ci\*，而第i个MIN节点选择了一个次低效用Ci，则有Ci>=Ci\*。

设MAX节点基于最优MIN节点最优策略的效用值为uopt=max(C1\*,……，Cn\*），基于次优MIN节点次优策略的效用值为usub=max(C1,……，Cn）。

由于Ci>=Ci\*，对i=1,……,n均成立，所以必定有usub>=uopt 。**得证**。

**构造**：



如图，规定MAX和MIN需要同时做出行动，在双方都行动之前不知道对方会如何行动。

由于MIN层采取次优策略，A节点会选择X=5，B节点会选择Z=4。

若MAX采取最优策略，MAX节点认为MIN采取最优策略，即A选择X=2，B选择Z=3，

那么MAX会选择B。然而实际上MAX的选择得到的效用为4。

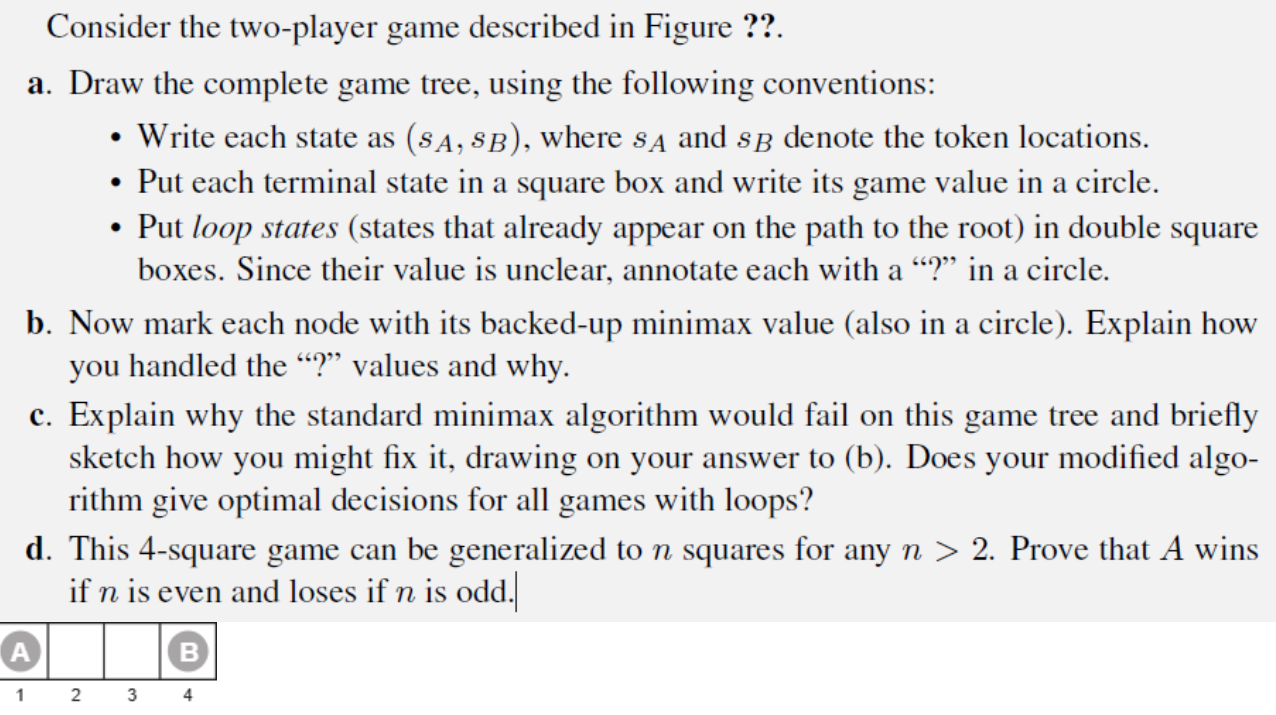
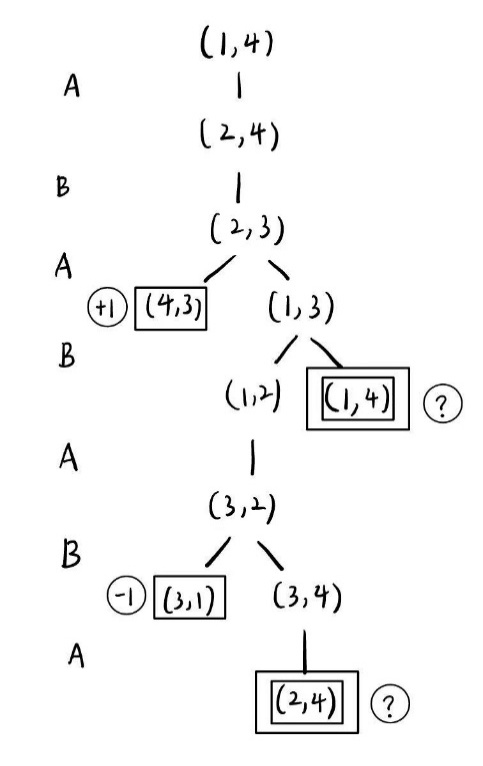
若MAX采取次优策略，在同上猜测的情况下，MAX会选择A。实际上MAX的选择得到的效用为5。

可见MIN采取次优策略时，MAX采取次优策略的收益可能高于采取最优策略的收益。

**Exercise 5.8**

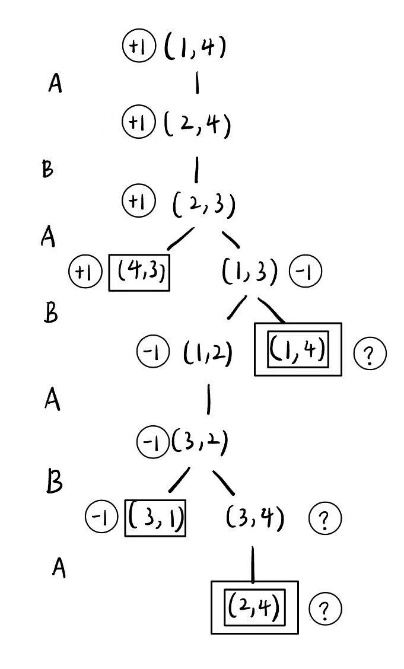
Player A moves first. The two players take turns moving, and each player must move his token to an open adjacent space in either direction. If the opponent occupies an adjacent space, then a player may jump over the opponent to the next open space if any. (For example, if A is on 3 and B is on 2, then A may move back to 1.) The game ends when one player reaches the opposite end of the board. If player A reaches space 4 first, then the value of the game to A is +1; if player B

 reaches space 1 first, then the value of the game to A is −1.



a.

b. 如果一个节点的后继值均为“？”，那么它的值也是“？”；如果不全为“？”，根据当前的行动者选择获胜值，即A选择+1，B选择 -1。



c．标准的极小极大算法是深度优先的，在该问题中会陷入无限循环。

修复方法：记录所有走过的状态，每次生成一个后继状态时，与记录的状态一一比较，若重复，则标记为“？”并停止生成该重复节点的后继。对“？”值的处理同b. 。

d. 证明（归纳法）：

n=3时，显然先行动的A会输掉。

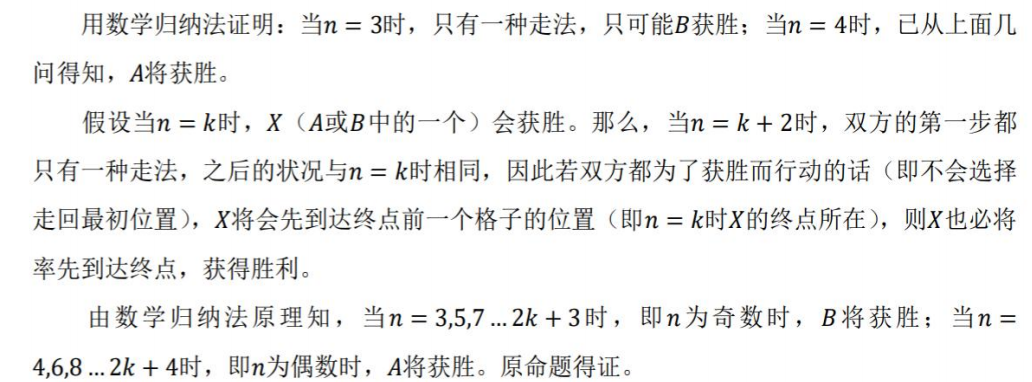
n=4时，按照上面画出的博弈树，A有必胜策略。

假设n<=2k-1时所有n为偶数的游戏，A会获胜；所有n为奇数的游戏，B会获胜。

n=2k时，考虑到A、B双方的第一步都只能互相靠近，第一次行动后必有A在方块2，B在方块n-1。考虑到子游戏[2, 2k-1]中，A是必胜的，即A会先于B到达2之前到达n-1，因此也会先于B到达n，那么A不会选择回到1，否则B到1的距离会小于A到n的距离而导致B必胜。如果B选择回到n，同理A也是必胜的。

n=2k+1时，同上分析，但在子游戏 [2, 2k]中B是必胜的，所以这种情况下B必胜。

综上，由数学归纳法，命题得证。



**Exercise 5.13**

Describe how the minimax and alpha–beta algorithms change for two-player, non-zero-sum games in which each player has a distinct utility function and both utility functions are known to both players.

If there are no constraints on the two terminal utilities, is it possible for any node to be pruned by alpha–beta?

What if the player’s utility functions on any state sum to a number between constants −k and k, making the game almost zero-sum?

**1.双人非零和博弈、独立效用函数的算法调整：**

设两个玩家的效用函数分别为 U1(s) 和 U2(s)

**(1)极小化极大算法的调整**

节点数据：每个节点需要同时记录两个玩家的效用值 (U1,U2)

节点行为：MAX节点（玩家1）：选择最大化 U1(s)的行动

MIN节点（玩家2）：选择最大化 U2(s)的行动

策略调整：两玩家不再是单纯的对抗关系，可能还有合作关系，需考虑对方的效用函数对自

身决策的影响

**(2) α-β剪枝算法的调整**

**剪枝条件**： **Max节点**（玩家1）：仅当某个子节点的 U1值高于当前已知的最高 U1值时，保留该分支；否则剪枝。

**Min节点**（玩家2）：仅当某个子节点的 U2值低于当前已知的最高 U2值时，保留该分支；否则剪枝。

**2. 无约束终端效用时的剪枝可能性：**

如果两个玩家的终端效用函数无任何约束（即 U1和 U2完全独立），传统的α-β剪枝可能无法生效**，因为无法通过单一效用值的上下界直接推断其他玩家的选择。**

**3. 近似零和博弈约束下的的剪枝可能性**

改变约束条件：U1∈[−k−U2, k−U2]

剪枝规则调整：**Max节点**（玩家1）：更新**α**=*当前已知的****U1下限***，通过**U1≥α**剪枝。

**Min节点**（玩家2）：更新 **β**=*当前已知的****U1上限***，通过 **U1≤β**剪枝。