第1,4,10次作业

第一次作业

3.7 给出下列问题的初始状态、目标测试、后继函数和耗散函数。选择精确得足以实现的形式化。

a. 只用四种颜色对平面地图染色，要求每两个相邻的地区不能染成相同的颜色。

b. 一间屋子里有一只 3 英尺高的猴子，屋子的房顶上挂着一串香蕉，离地面 8 英尺。屋子里有两个可叠放起来、可移动、可攀登的 3 英尺高的箱子。猴子很想得到香蕉。

c. 有一个程序，当送入一个特定文件的输入记录时会输出“不合法的输入记录”。已知每个记录的处理独立于其它记录。要求找出哪个记录不合法。

d. 有三个水壶，容量分别为 12 加仑、8 加仑和 3 加仑，还有一个水龙头。可以把壶装满或者倒空，从一个壶倒进另一个壶或者倒在地上。要求量出刚好 1

a. Initial state: No regions colored.
Goal test: All regions colored, and no two adjacent regions have the same color.
Successor function: Assign a color to a region.
Cost function: Number of assignments.

b. Initial state: As described in the text.
Goal test: Monkey has bananas.
Successor function: Hop on crate; Hop off crate; Push crate from one spot to another; Walk from one spot to another; grab bananas (if standing on crate).
Cost function: Number of actions.

c. Initial state: considering all input records.
Goal test: considering a single record, and it gives “illegal input” message.
Successor function: run again on the first half of the records; run again on the second half of the records.
Cost function: Number of runs.
Note: This is a **contingency problem**; you need to see whether a run gives an error message or not to decide what to do next.

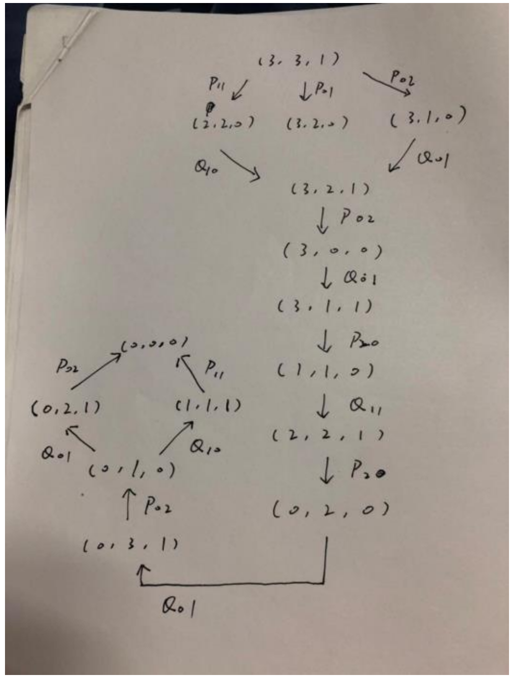
d. Initial state: jugs have values $[0, 0, 0]$.
Successor function: given values $[x, y, z]$, generate $[12, y, z]$, $[x, 8, z]$, $[x, y, 3]$ (by filling); $[0, y, z]$, $[x, 0, z]$, $[x, y, 0]$ (by emptying); or for any two jugs with current values x and y, pour y into x; this changes the jug with x to the minimum of $x + y$ and the capacity of the jug, and decrements the jug with y by by the amount gained by the first jug.
Cost function: Number of actions.

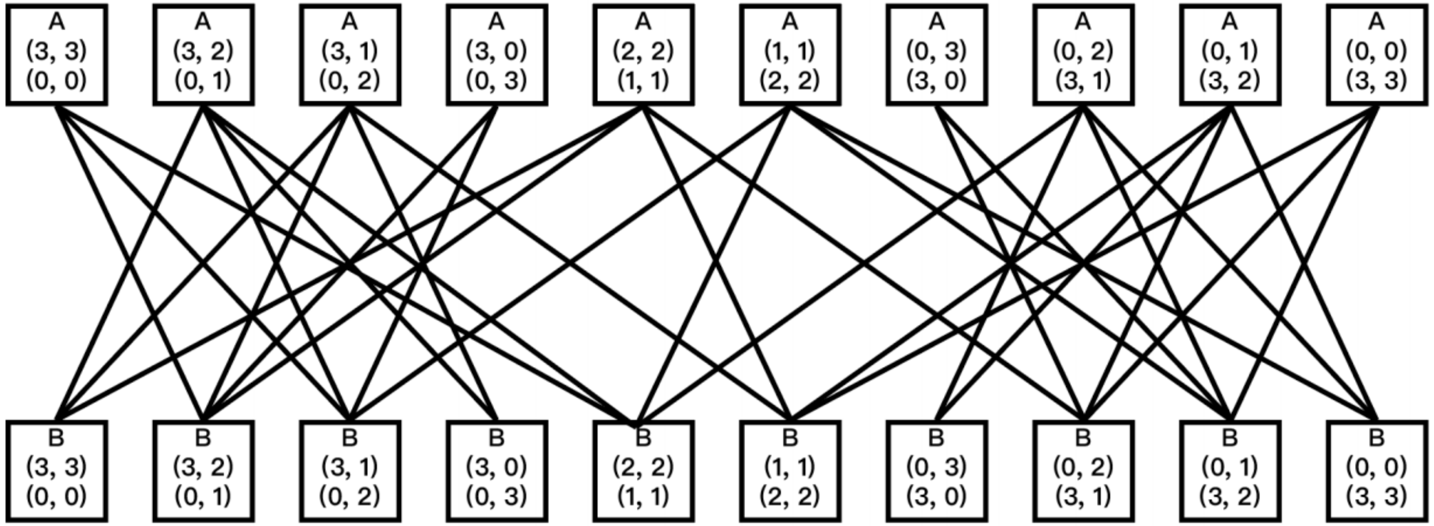
3.9 **传教士和野人**问题通常描述如下：三个传教士和三个野人在河的一边，还有一条能载一个人或者两个人的船。找到一个办法让所有的人都渡到河的另一岸，要求在任何地方野人数都不能多于传教士的人数（可以只有野人没有传教士）。这个问题在 AI 领域中很著名，因为它是第一篇从分析的观点探讨问题形式化的论文的主题（Amarel，1968）

a. 精确地形式化该问题，只描述确保该问题有解所必需的特性。画出该问题的完全状态空间图。

b. 用一个合适的搜索算法实现和最优地求解该问题。检查重复状态是个好主意吗？

c. 这个问题的状态空间如此简单，你认为为什么人们求解它却很困难？





(b)

```
BFS(G)
    // G: the graph of states and actions;
    // G.adj[u]: a set of nodes that can be reached from u by an action;
    // frontier: a FIFO queue;
    // if no solution can be found, return -1; else return cost
    // the optimal path can be infered from u.parent
    cost = 0
    for each node u in G
        u.visited = false
    initial_node.visited = true
    EnQueue(initial_node, frontier)
    while !isEmpty(frontier)
        cost = cost + 1
        u = DeQueue(frontier)
        for each node v in G.f[u] && !v.visited
            if v == goal_node
                return cost
            v.visited = true;
            EnQueue(v, frontier)
    return -1
```

一种可行的方案是：去2个野人，回1个野人，去2个野人，回1个野人，去2个传教士，回1个传教士和1个野人，去2个传教士，回1个野人，去2个野人，回1个野人，去2个野人。

最低路径耗散为11。

a. 初始状态是3个传教士和3个野人以及一条船在岸上，另一岸为空。目标状态是3个传教士和3个野人都到达另一岸。耗散函数为1。后继函数为移动1个或者2个人以及一条船到另一岸去。Pij为左岸向右岸输送i个传教士，j个野人，Qij为反方向。例如（2,1,0）表示在左岸有2个传教士，1个野人，船在右岸。

b. 最优解之一：(3,3,1)->(2,2,0)->···->(0,2,1)->(0,0,0)。由于问题规模有限，可以不必检查重复状态。

c. 几乎所有的移动要么是非法的，要么就需要返回到上一状态。因此状态空间规模将会变大。

第四次作业

Chapter 5

6.1 这道习题以井字棋（圈与十字游戏）为例子，练习博弈中的基本概念。我们定义 X_n 为恰好有 n 个 X 而没有 O 的行、列或者对角线的数目。同样 O_n 为正好有 n 个 O 的行、列或者对角线的数目。效用函数给 $X_3 = 1$ 的局面赋值+1，给 $O_3 = 1$ 的局面赋值−1。所有其它终止局面效用值为 0。对于非终止局面，我们使用线性的评价函数，定义为 $Eval(s) = 3X_2(s) + X_1(s) - (3O_2(s) + O_1(s))$。

a. 估算大约总共有多少种可能的井字棋局面？

b. 考虑到对称性，给出从空棋盘开始到深度为 2 的完整博弈树（例如，在棋盘上一个 X 和一个 O 的局面）。

c. 标出深度为 2 的所有局面的评价值。

d. 使用极小极大值算法标出深度为 1 和 0 的局面的回传值，并根据这些值选出最佳的起始步。

e. 假设节点*按对α-β剪枝的最优顺序*生成，圈出如果使用α-β剪枝将被剪掉的深度为 2 的节点。

a 不是9!

3X2O, X win: $8*C_6^2$

3X3O, O win: $8*C_6^3-12$

4X3O, X win: $8*6*C_5^3-12*C_3^1$

......

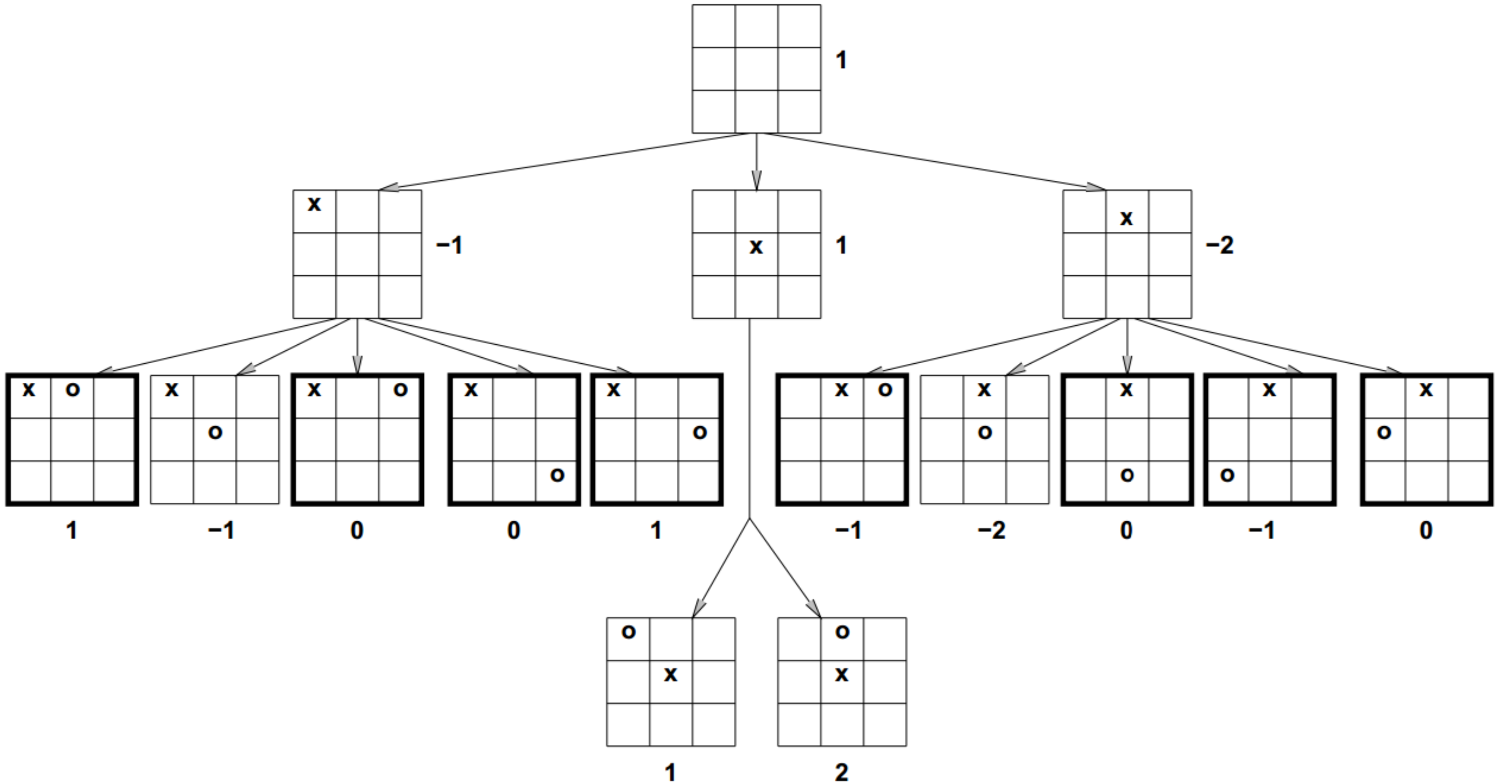
X can win in any of the following ways, as can O:

```
XXX  ...  ...  X..  .X.  ..X  X..  ..X
...  XXX  ...  X..  .X.  ..X  .X.  .X.
...  ...  XXX  X..  .X.  ..X  ..X  X..
```

http://www.mathrec.org/old/2002jan/solutions.html chapter 5

b c d如图

e 最优情况是，先走中间的树得到最大值为1，随后两边都能减去4个



Chapter 5

6.3 考虑图 6.14 中描述的两人游戏。

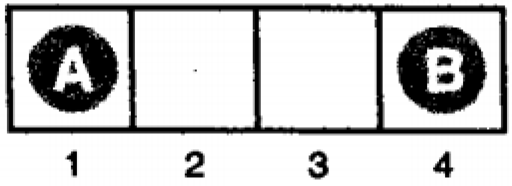
a. 用下面的约定，画出完整的博弈树：

- 每个状态用(S_A, S_B)表示，其中 S_A 和 S_B 表示棋子的位置。
- 每个终止状态外面画方框，并在圆圈里写出它的博弈值。
- 把*循环状态*（已经在到根节点的路径上出现过的状态）放在双方框内。由于不清楚怎样给循环状态赋值，在圆圈里标记一个“？”。

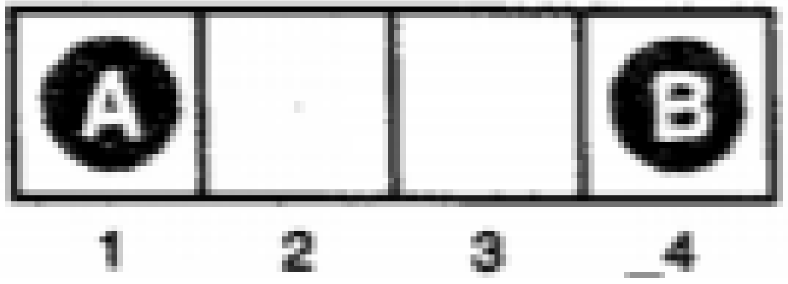
b. 现在给每个节点标记回传的极小极大值（也标记在圆圈里）。解释怎样处理“？”值和为什么。

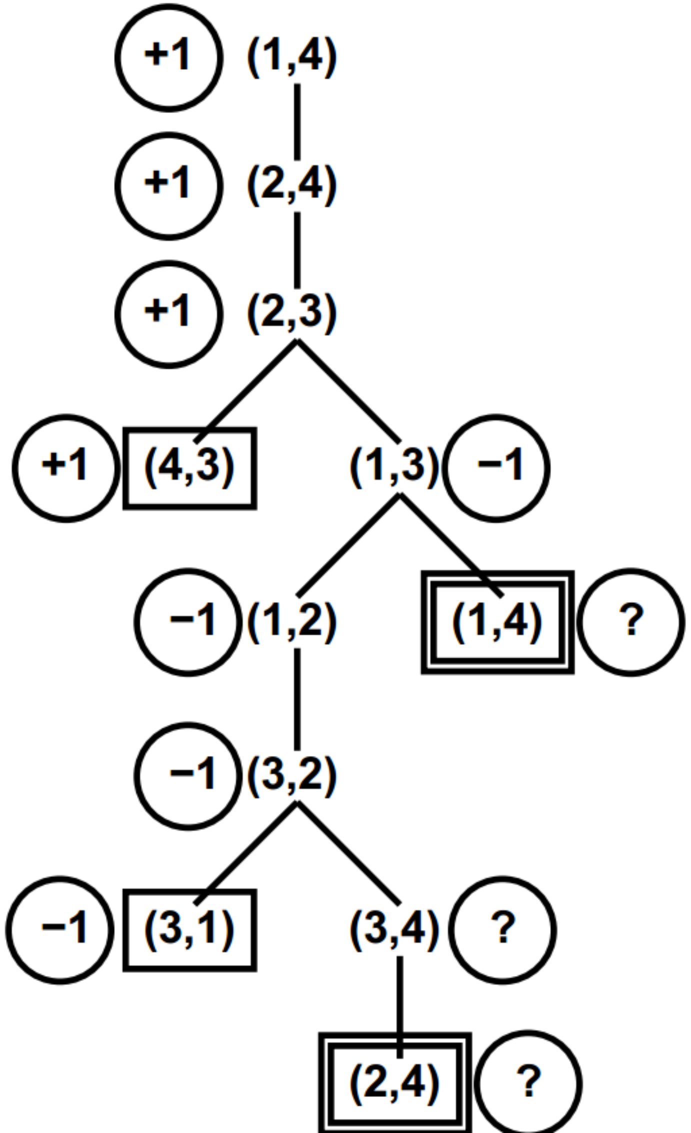
c. 解释标准的极小极大值算法为什么在这棵博弈树中会失败，简要地勾画你可能如何改进它，并在问题（b）的图上画出你的答案。你的改进算法对于所有包含循环的游戏都能给出最优决策吗？

d. 这个 4-方格游戏可以推广到 n 个方格，对于任意 $n > 2$。证明如果 n 是偶数，A 一定能赢；而 n 是奇数，A 一定会输。



Chapter 5





Chapter 5

b

MAX步：max{-1, ?} = ?, max{+1,?}=+1

MIN步：min{-1, ?} = -1, min{+1,?}=?

min{?, ?} = ?

c 标准的极大极小算法是深度优先搜索，会导致死循环。
用b的方案解决重复状态。
无法给出所有循环的最优方案，因为无法比较？和？

d 归纳法，n=3时，A输.n=4时，A赢。n>4时，由于前面两步是固定的，问题可以规约为n-2的问题。因此n为偶数时，A赢；n为奇数时，A输。

chapter 5

6.5 给出一个关于α-β剪枝正确性的形式化证明。要做到这个，考虑图 6.15 中的情况。问题为是否要剪掉节点 n_j，它是一个 MAX 节点，也是 n_1 的一个后代。基本的思路是当且仅当 n_1 的极小极大值可以被证明独立于 n_j 的值时，才剪枝。

a．n_1 的值由下面公式给出

$$n_1 = \min(n_2, n_{21}, \ldots, n_2 b_2)$$

为 n_2 找到类似的表达式，因此得到用 n_j 表示 n_1 的表达式。

b．节点 n_i 的极小极大值已知，l_i 是在节点 n_i 左侧深度为 i 的节点的极小值（或者极大值）。同样，r_i 是在 n_i 右侧深度为 i 的未探索过的节点的极小值（或者极大值）。用 l_i 和 r_i 的值重写你的 n_1 表达式。

c．现在调整你的表达式来说明为了影响 n_1，n_j 必须不超出由 l_i 值得到的一个特定界限。

d．对于 n_j 是 MIN 节点的情况重复上面的过程。

Chapter 5

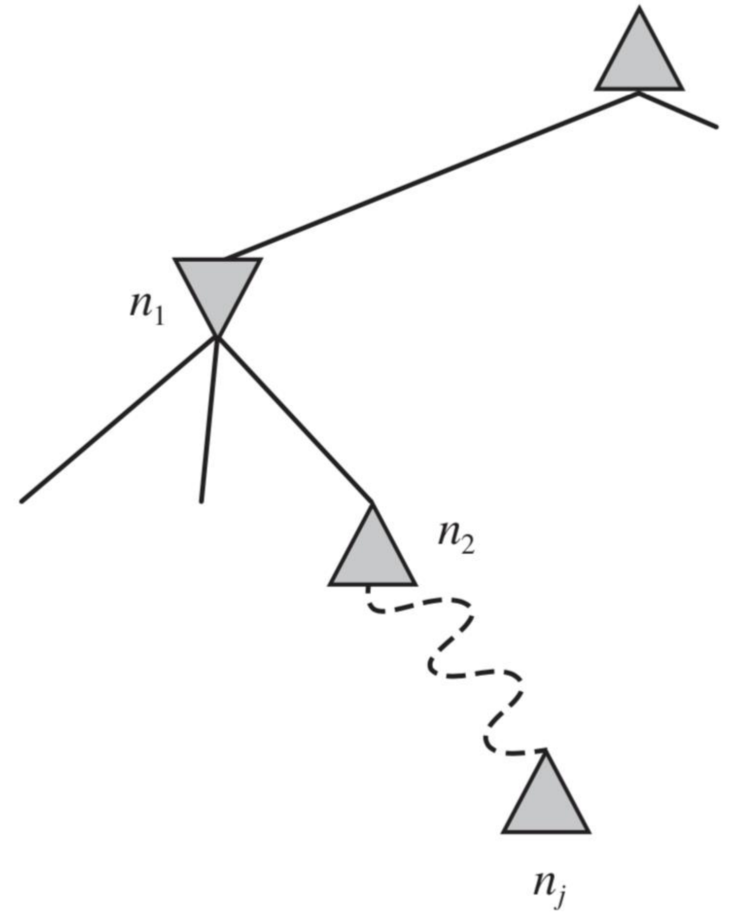


Figure 5.18 Situation when considering whether to prune node n_j.

chapter 5

a.

$$n_2 = max(n_3, n_{31}, \ldots, n_{3b_3})$$
$$n_1 = min(max(min(\ldots(min(n_j, n_{j1}, \ldots, n_{jb_j}), \ldots)\ldots), n_{31}, n_{3b_3}), n_{21}, \ldots, n_{2b_2})$$

依次类推，替代n3, ... 直到包含nj

b

$$n_1 = \min(l_2, \max(l_3, n_3, r_3), r_2)$$

$$n_1 = min(l_2, max(l_3, min(\ldots max(l_{j-1}, min(l_j, n_j, r_j), r_{j-1})\ldots), r_3), r_2)$$

继续扩展n3直到nj为止，最深的一层为 $\min(l_j, n_j, r_j)$

Chapter 5

c.

如果$n_j > l_j, min(l_j, n_j, r_j)$与$n_j$无关，那么$n_1$也与$n_j$无关；

如果$n_j > l_{j-2}$，

$min(l_j, n_j, r_j) \neq n_j$时，$n_1$与$n_j$无关；

$min(l_j, n_j, r_j) = n_j$时，$max(l_{j-1}, min(l_j, n_j, r_j), r_{j-1}) \geq n_j > l_{j-2}$，

$min(l_{j-2}, max(l_{j-1}, min(l_j, n_j, r_j), r_{j-1}), r_{j-2})$与$n_j$无关

n_j 只要大于任意一个下标为偶数的l_j,就不会对n_1造成影响，综上$n_j>min(l_2,l_4,,,l_j)$时对n_1无影响

d

n_j 只要小于所有下标为奇数的l_j,就不会对n_1造成影响，综上$n_j<max(l_3,l_5,,,l_j)$时对n_1无影响

第十次作业

- 试证明对于不含冲突数据（即特征向量完全相同但标记不同）的训练集，必存在与训练集一致（即训练误差为0）的决策树

- 反证法。假设不存在与训练集一致的决策树，那么训练集训练得到的决策树上至少有一个节点存在无法划分的多个数据（即，若节点上没有冲突数据的话，则必然能够将数据划分开）。这与前提“不含冲突数据”相矛盾，因此必有与训练集一致的决策树。