3.7

3.7

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

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？

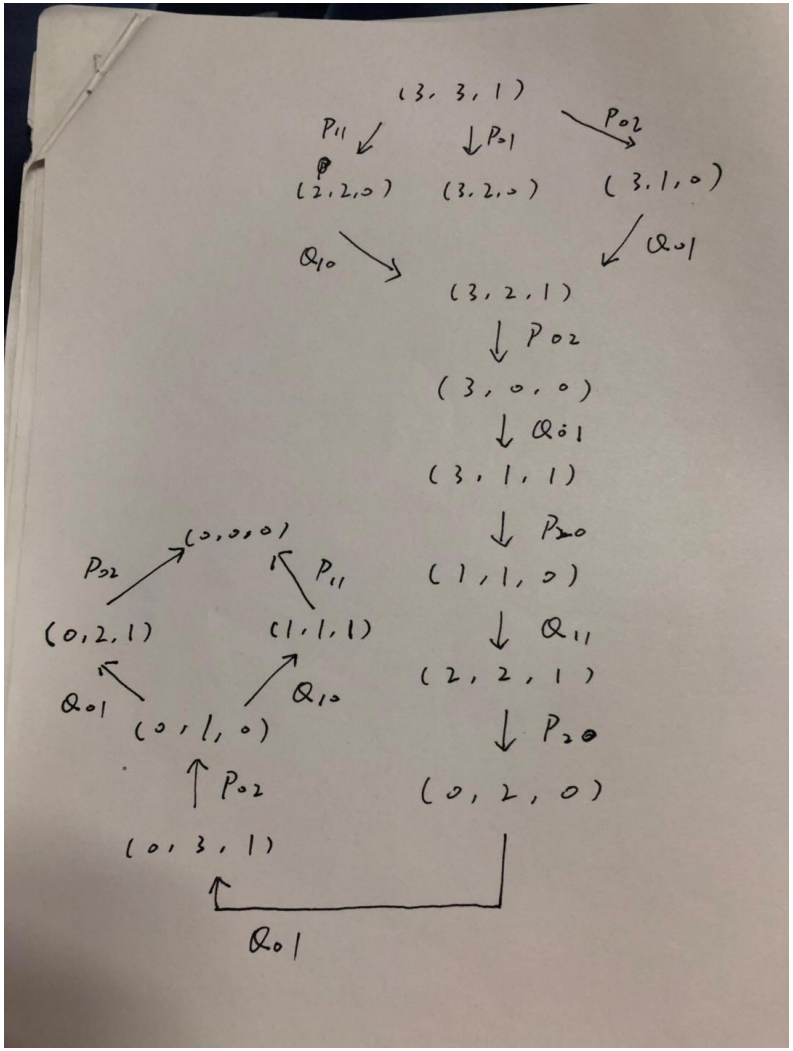
c. 这个问题的状态空间如此简单，你认为为什么人们求解它却很困难？

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。

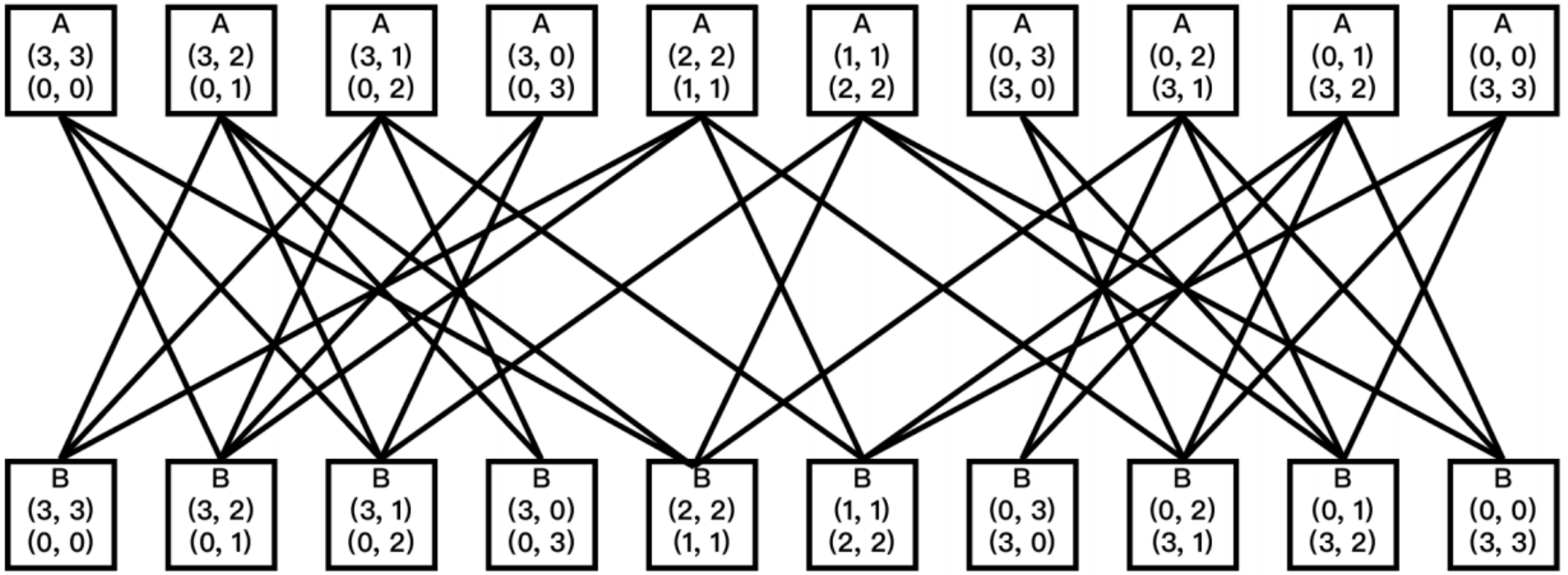
b. 最优解之一：(3,3,1)->(2,2,0)->…->(0,2,1)->(0,0,0)。检查重复状态可以避免进入死循环。

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。

3.9


(3, 3, 1)
P11
P01
P02
(2,2,0)
(3,2,0)
(3,1,0)
Q10
Q01
(3,2,1)
P02
(3,0,0)
Q01
(3,1,1)
P20
(1,1,0)
Q11
(2,2,1)
P20
(0,2,0)
(0,0,0)
P02
P11
(0,2,1)
(1,1,1)
Q01
Q10
(0,1,0)
P02
(0,3,1)
Q01

3.9


A (3, 3) (0, 0)
A (3, 2) (0, 1)
A (3, 1) (0, 2)
A (3, 0) (0, 3)
A (2, 2) (1, 1)
A (1, 1) (2, 2)
A (0, 3) (3, 0)
A (0, 2) (3, 1)
A (0, 1) (3, 2)
A (0, 0) (3, 3)
B (3, 3) (0, 0)
B (3, 2) (0, 1)
B (3, 1) (0, 2)
B (3, 0) (0, 3)
B (2, 2) (1, 1)
B (1, 1) (2, 2)
B (0, 3) (3, 0)
B (0, 2) (3, 1)
B (0, 1) (3, 2)
B (0, 0) (3, 3)

4.1（第三版3.23）

跟踪 A*搜索算法用直线距离启发式求解从 Lugoj 到 Bucharest 问题的过程。按顺序列出算法扩展的节点和每个节点的 f，g，h 值。

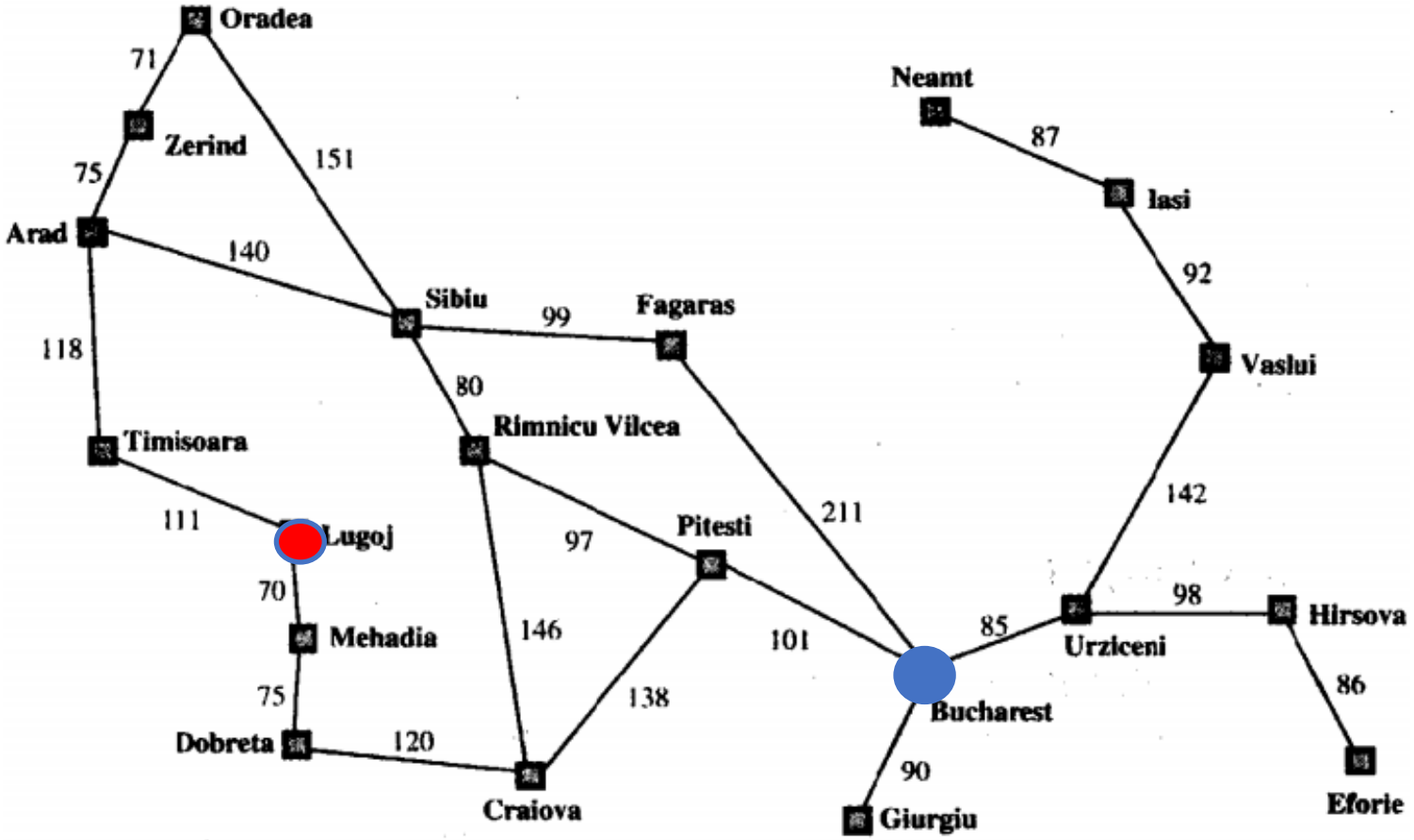


图 3.2 一个简化的部分罗马尼亚道路图

Arad	366	Mehadia	241
Bucharest	0	Neamt	234
Craiova	160	Oradea	380
Drobeta	242	Pitesti	100
Eforie	161	Rimnicu Vilcea	193
Fagaras	176	Sibiu	253
Giurgiu	77	Timisoara	329
Hirsova	151	Urziceni	80
Iasi	226	Vaslui	199
Lugoj	244	Zerind	374

图 3.22 h_{SLD} 的值——到 Bucharest 的直线距离

4.1

- 初始状态

  待扩展：Lugoj: 244=0+244

- 扩展Lugoj（g=0）

  待扩展：Mehadia: 311=70+241 Timisoara: 440=111+329

- 扩展Mehadia （g=70）

  待扩展：Timisoara: 440=111+329 Dobreta: 387=145+242

  Lugoj: 384=140+244

- 扩展Lugoj （g=140）

  待扩展： Timisoara: 440=111+329 Dobreta: 387=145+242

  Mehadia: 451=210+241 Timisoara: 580=251+329

4.1

- 扩展Dobreta（g=145）

  待扩展： Timisoara: 440=111+329　Mehadia: 451=210+241

  Timisoara: 580=251+329　Craiova: 425=256+160

  Mehadia: 461=220+241

- 扩展Craiova（g=256）

  待扩展： Timisoara: 440=111+329　Mehadia: 451=210+241

  Timisoara: 580=251+329　Mehadia: 461=220+241

  Dobreta: 627=385+242　Pitesti: 503=403+100

  Rimnicu: 604=411+193

- 扩展Timisoara（g=111）

  待扩展：Mehadia: 451=210+241　Timisoara: 580=251+329

  Mehadia: 461=220+241　Dobreta: 627=385+242

  Pitesti: 503=403+100　Rimnicu: 604=411+193

  Lugoj: 466=222+244　Arad: 595=229+366

- 扩展Mehadia（g=210）

  待扩展：Timisoara: 580=251+329　Mehadia: 461=220+241

  Dobreta: 627=385+242　Pitesti: 503=403+100

  Rimnicu: 604=411+193　Lugoj: 466=222+244

  Arad: 595=229+366　Lugoj: 524=280+244

  Dobreta: 527=285+242

- 扩展Mehadia（g=220）

  待扩展：Timisoara: 580=251+329　Dobreta: 627=385+242
  Pitesti: 503=403+100　Rimnicu: 604=411+193
  Lugoj: 466=222+244　Arad: 595=229+366
  Lugoj: 524=280+244　Dobreta: 527=285+242
  Lugoj: 534=290+244　Dobreta: 537=295+242

- 扩展Lugoj（g=222）

  待扩展：Timisoara: 580=251+329　Dobreta: 627=385+242
  Pitesti: 503=403+100　Rimnicu: 604=411+193
  Arad: 595=229+366　Lugoj: 524=280+244
  Dobreta: 527=285+242　Lugoj: 534=290+244
  Dobreta: 537=295+242　Mehadia: 533=292+241
  Timisoara: 662=333+329

- 扩展Pitesti（g=403）

  待扩展：
  Timisoara: 580=251+329
  Rimnicu: 604=411+193
  Lugoj: 524=280+244
  Lugoj: 534=290+244
  Mehadia: 533=292+241
  Burcharst: 504=504+0
  Craiova: 701=541+160
  Dobreta: 627=385+242
  Arad: 595=229+366
  Dobreta: 527=285+242
  Dobreta: 537=295+242
  Timisoara: 662=333+329
  Rimnicu: 693=500+193

- 拓展Burcharst: 结束

4.2(第三版3.25)

启发式路径算法是一个最佳优先搜索，它的目标函数是 $f(n)=(2-w)\,g(n)+w\,h(n)$。算法中 w 取什么值能保证算法是最优的？当 $w=0$ 时，这个算法是什么搜索？$w=1$ 呢？$w=2$ 呢？

- 当 w=0, $f(n)=2g(n)$, 此时扩展的是路径耗散最低的节点，为代价一致搜索。
- 当 w=1, $f(n)=g(n)+h(n)$, 为A*搜索
- 当 w=2, $f(n)=2h(n)$, $f(n)$完全由估计值决定，为贪婪最佳优先搜索。

- 改写$f(n)=(2-w)\left[g(n)+\frac{w}{2-w}h(n)\right]$，前面的系数$(2-w)$不影响扩展节点扩展的顺序。要保证算法最优，也就是保证 $\frac{w}{2-w}h(n)$ 不会高估耗散，即使得$\frac{w}{2-w}h(n)\le h(n)$, 得到$w\le 1$

设计一个启发函数，使它在八数码游戏中有时会估计过高，并说明它在什么样的特殊问题下会导致非最优解。（可以借助计算机的帮助。）证明：如果 h 被高估的部分从来不超过 c，A*算法返回的解的耗散比最优解的耗散多出的部分也不超过 c。

- 非最优的启发式：$g(n) = 3 \times \sum_{i=5}^{8} d_{\text{Manhattan}}(i)$ ，即5-8滑块的三倍曼哈顿距离和。

- 这个启发式在编号5-8的滑块离目标位置较远，1-4滑块较近时是容易高估的。一个特殊样例：

8	7	6
5	4	3
2	1	

目标状态

7	6	3
8		1
5	4	2

初始状态

4.6

- 证明耗散多出部分不超过c

1. 由于给定 h 被高估的部分从来不超过 c，那么$h(x) \leq h'(x) + c$. 设采用高估启发式函数返回的解目标节点是G_2, 最优目标节点是G_1.

2. 反设耗散多出的部分超过c，也即$g(G_2) - g(G_1) > c$, 此时$f(G_2) - f(G_1) = g(G_2) - g(G_1) + h(G_2) - h(G_1)$.

3. 由假设知$g(G_2) - g(G_1) > c$.

4. 由题设被高估的部分从来不超过c，$h(G_2) \leq h'(G_2) + c = c$，$h(G_1) \leq h'(G_1) + c = c$, 因此 $|h(G_2) - h(G_1)| \leq c$.

5. 因此$f(G_2) - f(G_1) = g(G_2) - g(G_1) + h(G_2) - h(G_1) > 0$. 从而在被高估的启发式函数下，$G_1$必然先于$G_2$被扩展，与返回解为$G_2$矛盾。所以返回的解与最优解之间的耗散差不会超过$c$.

4.7（第三版3.29）

证明如果一个启发式是一致的，它肯定是可采纳的。构造一个非一致的可采纳启发式。

- 设启发式$h(n)$是一致的，也就是说，对于每个节点n和通过任何行动 a 生成后继节点n'，有：$h(n) \leq c(n,a,n') + h(n')$, 即$h(n) - h(n') \leq c(n,a,n')$

令初始节点为a_0，a_i的后继为a_{i+1}，目标节点为a_N，那么有：

$$h(a_0) - h(a_1) \leq c(a_0, a[0][1], a_1)$$
$$h(a_1) - h(a_2) \leq c(a_1, a[1][2], a_2)$$
$$\cdots$$
$$h(a_{N-1}) - h(a_N) \leq c(a_{N-1}, a[N-1][N], a_N)$$

两边分别相加，得到$h(a_0) - h(a_N) \leq Actual\ cost(a_0, a_N)$. 从而可采纳

4.7

- 非一致的可采纳启发式：

  考虑如下路径a------b------c，其中a—b的实际耗散为2，b—c的实际耗散为2.

  若令$h(a) = 4,\ h(b) = 1,\ h(c) = 0$，由于没有高估实际耗散，$h$是可采纳的。但$h(a) = 4 > h(b) + cost(a, b) = 1 + 2$，所以非一致。

6.1

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。

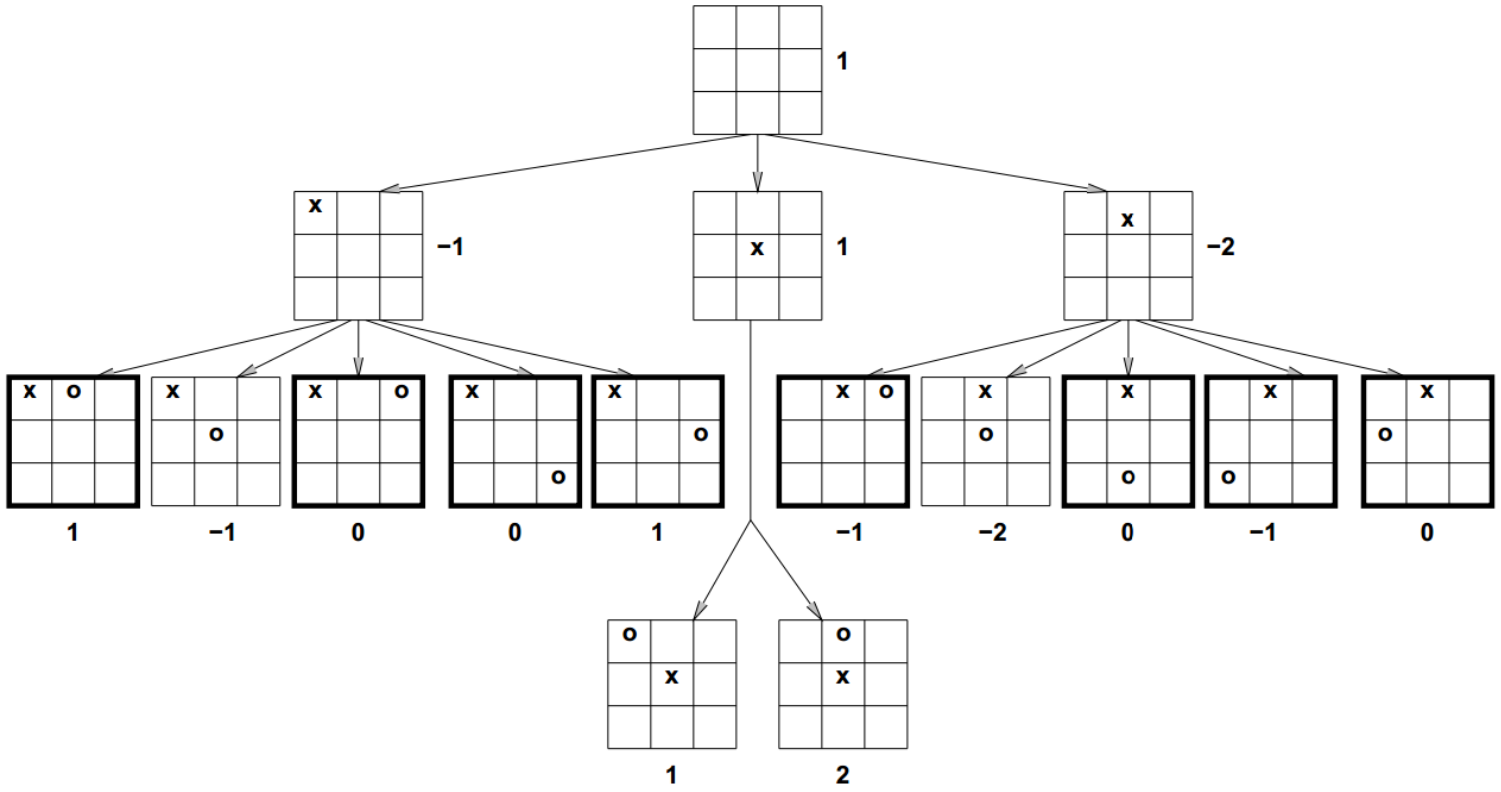
e. 假设节点*按对α-β剪枝的最优顺序*生成，圈出如果使用α-β剪枝将被剪掉的深度为 2 的节点。

6.1

a 需要分情况讨论，不能直接答3^9或者9！

b c d 如图

e 选择中间



6.3（第三版5.8）

5.8 考虑图 5.17 中描述的两人游戏。

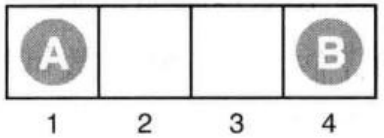


图 5.17 一个简单游戏的初始棋局

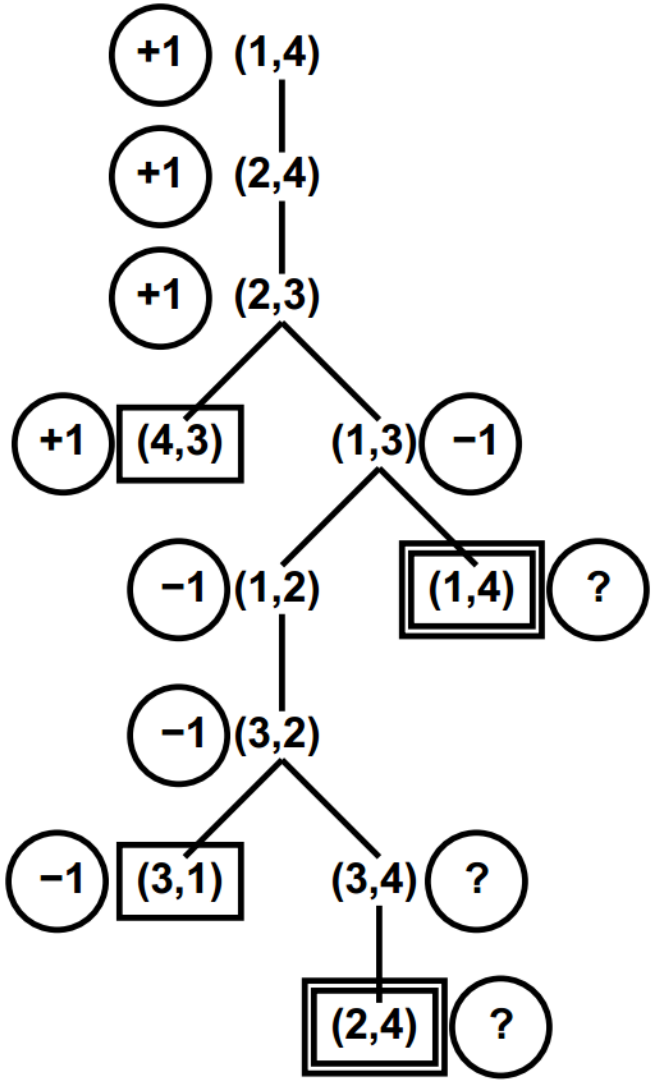
选手 *A* 先走。两个选手轮流走棋，每个人必须把自己的棋子移动到任一方向上的相邻空位中。如果对方的棋子占据着相邻的位置，你可以跳过对方的棋子到下一个空位。（例如，*A* 在位置 3，*B* 在位置 2，那么 A 可以移回 1。）当一方的棋子移动到对方的端点时游戏结束。如果 *A* 先到达位置 4，*A* 的值为+1；如果 *B* 先到位置 1，A 的值为−1。

a. 根据如下约定画出完整博弈树：
   - 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。
   - 每个终止状态用方框画出，用圆圈写出它的博弈值。
   - 把循环状态（在到根结点的路径上已经出现过的状态）画上双层方框。由于不清楚他们的值，在圆圈里标记一个“？”。

b. 给出每个结点倒推的极小极大值（也标记在圆圈里）。解释怎样处理“？”值和为什么这么处理。

c. 解释标准的极小极大算法为什么在这棵博弈树中会失败，简要说明你将如何修正它，在（b）的图上画出你的答案。你修正后的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，其中 $n > 2$。证明如果 n 是偶数 A 一定能赢，而 n 是奇数则 A 一定会输。



6.3（第三版5.8）

- b
  - MAX步：max{-1, ?} = ?, max{+1,?}=+1
  - MIN步：min{-1, ?} = -1, min{+1,?}=?
  - min{?, ?} = ?
- c
  - 导致死循环，如果一个节点下有两个win节点，没法处理
  - 方案：检测重复状态，用b的方案解决重复状态。
- d
  - 归纳法，n=3时，A输.n=4时，A赢。
  - n>4时，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。

6.5（第三版5.13）

5.13 请给出α-β剪枝正确性的形式化证明。要做到这一点需考虑图 5.18。问题为是否要剪掉结点 n_j，它是一个 MAX 结点，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，会发生剪枝。

a. n_1 的值是所有后代结点的最小值：$n_1 = \min(n_2, n_{21}, \cdots, n_{2\,b2})$。请为 n_2 找到类似的表达式，以得到用 n_j 表示的 n_1 的表达式。

b. 深度为 i 的结点 n_i 的极小极大值已知，l_i 是在结点 n_i 左侧结点的极小值（或者极大值）。同样，r_i 是在 n_i 右侧的未探索过的结点的极小值（或者极大值）。用 l_i 和 r_i 的值重写 n_1 的表达式。

c. 现在重新形式化表达式，来说明为了向 n_1 施加影响，n_j 不能超出由 l_i 值得到的某特定界限。

d. 假设 n_j 是 MIN 结点的情况，请重复上面的过程。

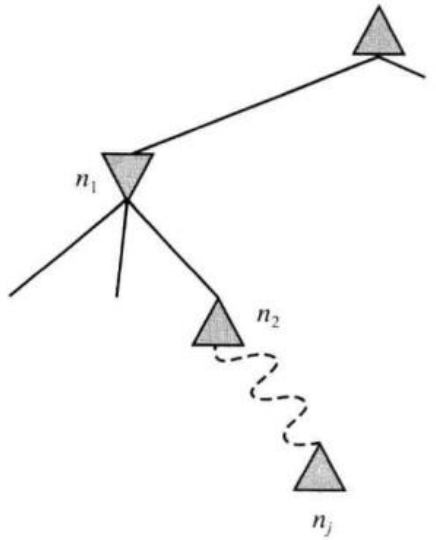


图 5.18　是否剪掉结点 n_j 时的情形

6.5（第三版5.13）

- a
  - n2 = max(n3,n31,... ,n3b3)
  - n1 = min(max(n3,n31,... ,n3b3),n21,... ,n2b2)
  - 拓展直到表达式包含nj
- b
  - n1 = min(l2, max(l3,n3,r3),r2)
  - n1 = min(l2,max(l3,min(...max(lj-1,min(lj,nj,rj),rj-1)...),r3),r2)
  - 继续扩展n3直到nj为止，最深的一层为min(lj,nj,rj)

- C
  - 由于nj为max结点，j为偶数
  - $n_{j-1} = \min(\text{nj}, \text{lj}, \text{rj})$ => nj<=lj (否则nj对nj-1无影响，也对n1无影响）
  - $\text{n}_{j-2} = \max(n_{j-1}, l_{j-1}, r_{j-1}) = \max(n_j, l_{j-1}, r_{j-1}) \Rightarrow n_j \geq l_{j-1}$
  - 由此不断递推得$\text{n}_j \leq l_{j-2}, n_j \leq l_{j-4}, \ldots$和$n_j \geq l_{j-1}, \ldots$
  - 最后$\max\{l_3, l_5, l_7, \ldots, l_{j-1}\} \leq n_j \leq \min\{l_2, l_4, \ldots, l_j\}$
- d
  - nj 是min结点,j为奇数
  - $n_{j-1} = \max(\text{nj}, \text{lj}, \text{rj})$ => nj>=lj
  - $\text{n}_{j-2} = \min(n_{j-1}, l_{j-1}, r_{j-1}) = \min(n_j, l_{j-1}, r_{j-1}) \Rightarrow n_j \leq l_{j-1}$
  - $\max\{l_3, l_5, l_7, \ldots, l_{j-2}, l_j\} \leq n_j \leq \min\{l_2, l_4, \ldots, l_{j-1}\}$

5.6

分别用回溯算法、前向检验算法、MRV 和最少约束值启发式算法手工求解图 5.2 中的密码算术问题。

$$O+O=R+10\cdot X_1$$
$$X_1+W+W=U+10\cdot X_2$$
$$X_2+T+T=O+10\cdot X_3$$
$$X_3=F$$

$$\begin{array}{r} T\ W\ O \\ +\ T\ W\ O \\ \hline F\ O\ U\ R \end{array}$$

	X_3	X_2	X_1	F	T	W	O	U	R
初始域	{0,1}	{0,1}	{0,1}	{0,1…,9}	{0,1,..,9}	{0,1,..,9}	{0,1,..,9}	{0,1,..,9}	{0,1,..,9}
After $X_3=1$	1	{0,1}	{0,1}	{1}	{0,1,..,9}	{0,1,..,9}	{0,1,..,9}	{0,1,..,9}	{0,1,..,9}
After $F=1$	1	{0,1}	{0,1}	1	{5,..,9}	{0,2,..,9}	{0,2,..,9}	{0,2,..,9}	{0,2,..,9}
After $X_2=0$	1	0	{0,1}	1	{5,..,9}	{0,2,3,4}	{0,2,4,6,8}	{0,2,..,9}	{0,2,..,9}

5.6

$$O + O = R + 10 \cdot X_1$$
$$X_1 + W + W = U + 10 \cdot X_2$$
$$X_2 + T + T = O + 10 \cdot X_3$$
$$X_3 = F$$

$$\begin{array}{r} T\ W\ O \\ +\ T\ W\ O \\ \hline F\ O\ U\ R \end{array}$$

	X_3	X_2	X_1	F	T	W	O	U	R
After $X_1 = 0$	1	0	0	1	{5,6,7}	{0,2,3,4}	{0,2,4}	{0,4,6,8}	{0,4,8}
After O=4	1	0	0	1	{7}	{0,3}	4	{0,6}	{8}
After T=7	1	0	0	1	7	{0,3}	4	{0,6}	{8}
After R=8	1	0	0	1	7	{0,3}	4	{0,6}	8
After W=3	1	0	0	1	7	3	4	{6}	8
After U=6	1	0	0	1	7	3	4	6	8

5.8

用 AC-3 算法说明弧相容对图 5.1 中所示问题能够检测出不完全赋值{*WA* = *red*, *V* = *blue*}的不相容。

- 证明：

  (SA,WA)消除不相容,SA={G,B}

  (SA,V)消除不相容,SA={G}

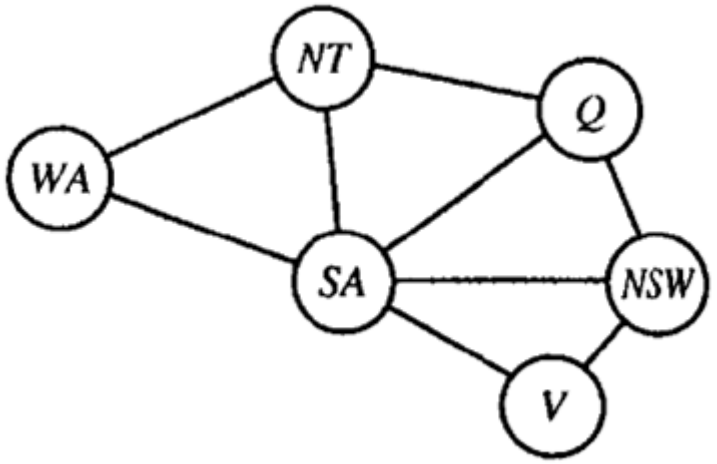
  同理，得到NT={B}, NSW={R}

  (Q,NT)消除不相容, Q={R,G}

  (Q,NSW)消除不相容, Q={G}

  (Q,SA)消除不相容, Q={}

  所以{WA=R, V=B}不相容



5.9

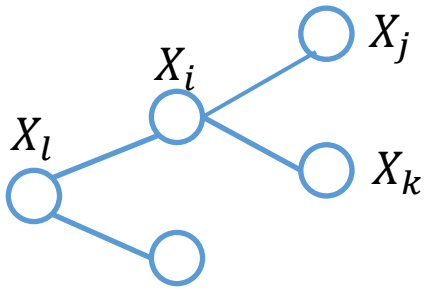
在树状结构 CSP 问题的最坏情况下运行 AC-3 算法的复杂度是多少？

- 假设有n个顶点，值域中最多有d个取值。树状结构下，弧的条数为$O(n)$，检验每条弧的复杂度为$O(d^2)$.
- 每条弧只需要检验一次，故总的复杂度为$O(nd^2)$.
- 如何做到每条弧只检验一次？

    1. 采用逆拓扑序进行检验
    2. 使用集合数据结构存储弧

5.9

- 逆拓扑序检验：



当检验弧(X_i, X_j)时，X_i值域发生变化，需要将弧(X_l, X_i)重新检验。同理，当检验弧(X_i, X_k)时，需要重新检验弧(X_l, X_i)。
在逆拓扑序下，只有当X_i与其所有子节点组成的弧检验完后，(X_l, X_i)才会被检验，而(X_l, X_i)同时也是此前不断重新加入检验集合中的弧。这就保证了在树中，一条边被检验完毕后，不再会被加入待检验集合。

- 使用集合数据结构存储弧（第三版教材P174）

最流行的弧相容算法是 AC-3（见图 6.3）。为使每个变量都是弧相容的，AC-3 算法维护一个弧相容队列（实际上，考虑的顺序并不重要，所以数据结构实际上是个集合，我们只是称之为队列）。首先，队列中包含 CSP 中的所有弧。AC-3 从队列中弹出弧(X_i, X_j)，首

当使用集合存储待检验弧，逆拓扑序中所有重新需要加入检验集合中的弧，早已经位于待检验集合中，无需加入。这就保证了每条弧只会被检验一次。

7.12 本习题将考察子句和蕴涵语句之间的关系。

a. 证明子句$(\neg P_1 \vee \ldots \vee \neg P_m \vee Q)$ 逻辑等价于蕴涵语句$(P_1 \wedge \ldots \wedge P_m) \Rightarrow Q$。

b. 证明每个子句（不管正文字的数量）都可以写成$(P_1 \wedge \ldots \wedge P_m) \Rightarrow (Q_1 \vee \ldots \vee Q_n)$的形式，其中 P 和 Q 都是命题符号。由这类语句构成的知识库是表示为**蕴涵范式**或称**科瓦尔斯基（Kowalski）范式**的。

c. 写出蕴涵范式语句的完整归结规则。

a. $(P_1 \wedge \cdots \wedge P_m) \Rightarrow Q$ 等价于 $\neg(P_1 \wedge \cdots \wedge P_m) \vee Q$（蕴含消去）

$\neg(P_1 \wedge \cdots \wedge P_m)$ 等价于 $(\neg P_1 \vee \cdots \vee \neg P_m)$（摩根律）

因此，$(P_1 \wedge \cdots \wedge P_m) \Rightarrow Q$ 等价于 $(\neg P_1 \vee \cdots \vee \neg P_m \vee Q)$

b. 一个子句会有一些正文字和负文字，先将它们排列成$(\neg P_1 \vee \cdots \vee \neg P_m \vee Q_1 \vee \cdots \vee Q_n)$，然后设$Q$ 为 $Q_1 \vee \cdots \vee Q_n$，则同a可以得到，$(P_1 \wedge \cdots \wedge P_m) \Rightarrow (Q_1 \vee \cdots \vee Q_n)$。

c. 对于原子语句p_i, q_i, r_i, s_i，其中$p_j = q_k$，

$$p_1 \wedge \cdots \wedge p_j \wedge \cdots \wedge \mathrm{p}_{n_1} \Rightarrow r_1 \vee \cdots \vee r_{n_2}$$

$$s_1 \wedge \cdots \wedge s_{n_3} \Rightarrow q_1 \vee \cdots \vee q_k \vee \cdots \vee q_{n_4}$$

$$\left(p_1 \wedge \cdots \wedge p_{j-1} \wedge p_{j+1} \wedge \cdots \wedge \mathrm{p}_{n_1} \wedge s_1 \wedge \cdots \wedge s_{n_3} \Rightarrow r_1 \vee \cdots \vee r_{n_2} \vee q_1 \vee \cdots \vee q_{k-1} \vee q_{k+1} \vee \cdots \vee q_{n_4}\right)$$

- 证明前向链算法的完备性

很容易看出，前向链接是**可靠**的：每个推理本质上是分离规则的一个应用。前向连接也是**完备**的：每个被蕴涵的原子语句都将得以生成。验证这一点的最简单方法是考察 *inferred* 表的最终状态（在算法到达**不动点**以后，不可能再出现新的推理）。该表对于在推理过程中参与推理的每个符号都包含 *true*，而所有其它的符号为 *false*。我们可以把该推理表看作一个逻辑模型；而且，*原始 KB 中的每个确定子句在该模型中都为真*。为了看到这一点，假定相反情况成立，也就是说某个子句 $a_1 \wedge \ldots \wedge a_k \Rightarrow b$ 在模型中为假。那么 $a_1 \wedge \ldots \wedge a_k$ 在模型中必须为真，b 在模型中必须为假。但这和我们的假设，即算法已经到达一个不动点相矛盾！因而，我们可以得出结论，在不动点推理的原子语句集定义了原始 *KB* 的一个模型。此外，被 *KB* 蕴涵的任一原子语句 q 在它的所有模型中必须为真，尤其是这个模型。因此，每个被蕴涵的语句 q 必定会被算法推断出来。

8.6 用一个没有矛盾的词汇表（需要你自己定义）在一阶逻辑中表示下列语句：

a. 某些学生在 2001 年春季学期上法语课。

b. 上法语课的每个学生都通过了考试。

c. 只有一个学生在 2001 年春季学期上希腊语课。

d. 希腊语课的最好成绩总是比法语课的最好成绩高。

e. 每个买保险的人都是聪明的。

f. 没有人会买昂贵的保险。

g. 有一个代理，他只卖保险给那些没有投保的人。

h. 镇上有一个理发师，他给所有不自己刮胡子的人刮胡子。

i. 在英国出生的人，如果其双亲都是英国公民或永久居住者，那么此人生来就是一个英国公民。

j. 在英国以外的地方出生的人，如果其双亲生来就是英国公民，那么此人血统上是一个英国公民。

k. 政治家可以一直愚弄某些人，也可以在某个时候愚弄所有人，但是他们无法一直愚弄所有的人。

$Takes(x, c, s)$: student x takes course c in semester s;
$Passes(x, c, s)$: student x passes course c in semester s;
$Score(x, c, s)$: the score obtained by student x in course c in semester s;
$x > y$: x is greater than y;
F and G: specific French and Greek courses (one could also interpret these sentences as referring to *any* such course, in which case one could use a predicate $Subject(c, f)$ meaning that the subject of course c is field f;
$Buys(x, y, z)$: x buys y from z (using a binary predicate with unspecified seller is OK but less felicitous);
$Sells(x, y, z)$: x sells y to z;
$Shaves(x, y)$: person x shaves person y
$Born(x, c)$: person x is born in country c;
$Parent(x, y)$: x is a parent of y;
$Citizen(x, c, r)$: x is a citizen of country c for reason r;
$Resident(x, c)$: x is a resident of country c;
$Fools(x, y, t)$: person x fools person y at time t;
$Student(x)$, $Person(x)$, $Man(x)$, $Barber(x)$, $Expensive(x)$, $Agent(x)$, $Insured(x)$, $Smart(x)$, $Politician(x)$: predicates satisfied by members of the corresponding categories.

a. Some students took French in spring 2001.
$\exists x \;\; Student(x) \wedge Takes(x, F, Spring2001)$.

b. Every student who takes French passes it.
$\forall x, s \;\; Student(x) \wedge Takes(x, F, s) \Rightarrow Passes(x, F, s)$.

c. Only one student took Greek in spring 2001.
$\exists x \;\; Student(x) \wedge Takes(x, G, Spring2001) \wedge \forall y \;\; y \neq x \Rightarrow \neg Takes(y, G, Spring2001)$.

d. The best score in Greek is always higher than the best score in French.
$\forall s \;\; \exists x \;\; \forall y \;\; Score(x, G, s) > Score(y, F, s)$.

e. Every person who buys a policy is smart.
$\forall x \;\; Person(x) \wedge (\exists y, z \;\; Policy(y) \wedge Buys(x, y, z)) \Rightarrow Smart(x)$.

f. No person buys an expensive policy.
$\forall x, y, z \;\; Person(x) \wedge Policy(y) \wedge Expensive(y) \Rightarrow \neg Buys(x, y, z)$.

g. There is an agent who sells policies only to people who are not insured.
$\exists x \;\; Agent(x) \wedge \forall y, z \;\; Policy(y) \wedge Sells(x, y, z) \Rightarrow (Person(z) \wedge \neg Insured(z))$.

h. There is a barber who shaves all men in town who do not shave themselves.
$\exists x \;\; Barber(x) \wedge \forall y \;\; Man(y) \wedge \neg Shaves(y, y) \Rightarrow Shaves(x, y)$.

i. A person born in the UK, each of whose parents is a UK citizen or a UK resident, is a UK citizen by birth.
$\forall x \;\; Person(x) \wedge Born(x, UK) \wedge (\forall y \;\; Parent(y, x) \Rightarrow ((\exists r \;\; Citizen(y, UK, r)) \vee Resident(y, UK))) \Rightarrow Citizen(x, UK, Birth)$.

j. A person born outside the UK, one of whose parents is a UK citizen by birth, is a UK citizen by descent.

$$\forall x \;\; Person(x) \wedge \neg Born(x, UK) \wedge (\exists y \;\; Parent(y, x) \wedge Citizen(y, UK, Birth)) \Rightarrow Citizen(x, UK, Descent).$$

k. Politicians can fool some of the people all of the time, and they can fool all of the people some of the time, but they can't fool all of the people all of the time.

$$\begin{aligned} &\forall x \;\; Politician(x) \Rightarrow \\ &\quad (\exists y \;\; \forall t \;\; Person(y) \wedge Fools(x, y, t)) \wedge \\ &\quad (\exists t \;\; \forall y \;\; Person(y) \Rightarrow Fools(x, y, t)) \wedge \\ &\quad \neg(\forall t \;\; \forall y \;\; Person(y) \Rightarrow Fools(x, y, t)) \end{aligned}$$

8.15　解释下面给出的 wumpus 世界中相邻方格的定义存在什么问题：

$$\forall x, y \quad Adjacent([x, y], [x+1, y]) \wedge Adjacent([x, y], [x, y+1])$$

1. 仅考虑上相邻格和右相邻格：Adjacent([1,2], [1,1])作为相邻格不符合定义
2. 忽略边界情况

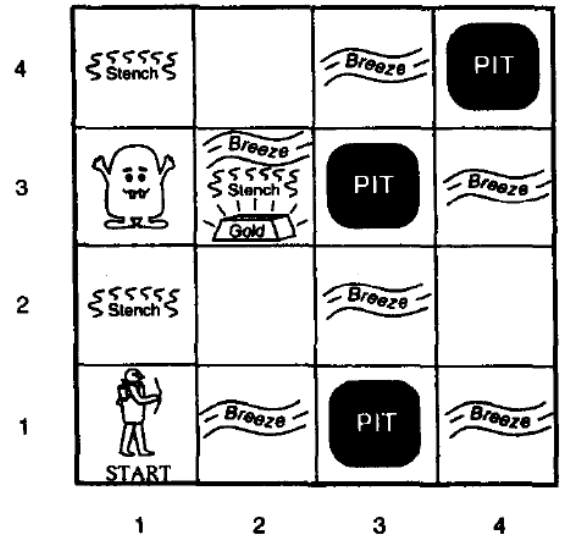


图 7.2　一个典型的 wumpus 世界。智能体位于左下角

8.15 There are several problems with the proposed definition. It allows one to prove, say, $Adjacent([1,1],[1,2])$ but not $Adjacent([1,2],[1,1])$; so we need an additional symmetry axiom. It does not allow one to prove that $Adjacent([1,1],[1,3])$ is false, so it needs to be written as

$$\forall s_1, s_2 \quad \Leftrightarrow \quad \ldots$$

Finally, it does not work as the boundaries of the world, so some extra conditions must be added.

9.3 假定知识库中只包括一条语句：$\exists x\ AsHighAs(x, Everest)$。下列那个语句是应用存在量词实例化以后的合法结果？

a. $AsHighAs(Everest, Everest)$

b. $AsHighAs(Kilimanjaro, Everest)$

c. $AsHighAs(Kilimanjaro, Everest) \wedge AsHighAs(BenNevis, Everest)$ （在两次应用之后）

a. 错误，Everest为对象名，不能再属于另一个对象

b. 正确

c. 错误，Kilimanjaro & BenNevis 应用两次实例化

第三版 p.269

只要 C_1 不在知识库的别处出现。基本上，存在语句说明存在某些满足条件的对象，实例化过程仅仅是给该对象命名。自然地，该名字不能已经属于另一个对象。数学提供了一个很好的例子：假设我们发

存在实例化比全称实例化更加复杂，它在推理中也承担了稍微有些不同的角色。全称实例化可以多次应用从而获得许多不同的结果，而存在实例化只能应用一次，然后存在量化语句就可以被抛弃。

9.4 对于下列每对原子语句，如果存在，请给出最一般合一置换：

a. $P(A, B, B), P(x, y, z)$

b. $Q(y, G(A, B)), Q(G(x, x), y)$

c. $Older(Father(y), y), Older(Father(x), John)$

d. $Knows(Father(y), y), Knows(x, x)$

a. {x/A, y/B, z/B}

b. 不存在，y = G(A, B), y = G(x, x)，所以需要x = A, x = B，矛盾。（两个语句对不同对象使用相同变量名则不能合一，通过重命名解决）

c. {x/John, y/John}

d. 不存在，需要x = y, x = Father(y)，矛盾。

9.9 写出下列语句的逻辑表示，使得它们适合应用一般化分离规则：

a. 马、奶牛和猪都是哺乳动物。

b. 一匹马的后代是马。

c. Bluebeard 是一匹马。

d. Bluebeard 是 Charlie 的父亲。

e. 后代和双亲是逆关系。

f. 每个哺乳动物都有一个双亲。

GMP（一般化分离规则）：

$$\frac{p_1', p_2', \ldots, p_n', (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{q\theta}$$

where $p_i'\theta = p_i\theta$ for all i

a. $Horse(x) \Rightarrow Mammal(x)$
$Cow(x) \Rightarrow Mammal(x)$
$Pig(x) \Rightarrow Mammal(x)$

b. $Offspring(x, y) \wedge Horse(y) \Rightarrow Horse(x)$

c. $Horse(Bluebeard)$

d. $Parent(Bluebeard, Charlie)$

e. $Offspring(x, y) \Rightarrow Parent(y, x)$
$Parent(x, y) \Rightarrow Offspring(y, x)$
(Note we couldn't do $Offspring(x, y) \Leftrightarrow Parent(y, x)$ because that is not in the form expected by Generalized Modus Ponens.)

f. $Mammal(x) \Rightarrow Parent(G(x), x)$ (here G is a Skolem function).

Each existential variable is replaced by a **Skolem** function of the enclosing universally quantified variables:

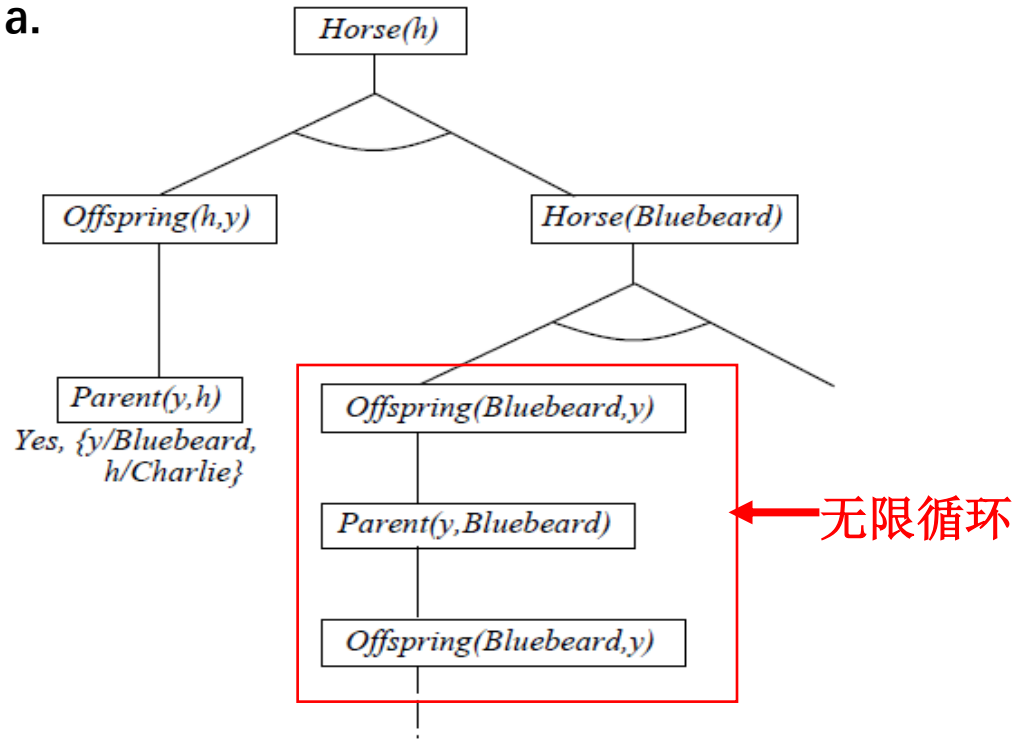
9.10 本习题中，我们将采用你在习题 9.9 中写出的语句，运用反向链接算法来回答一个问题。

a. 画出由一个穷举反向链接算法为查询 $\exists h$ *horse(h)*生成的证明树，其中子句按照给定的顺序进行匹配。

a. 你对于本领域注意到了什么？

b. 实际上从你的语句中得出了多少个 *h* 的解？

a.



a. $Horse(x) \Rightarrow Mammal(x)$
$Cow(x) \Rightarrow Mammal(x)$
$Pig(x) \Rightarrow Mammal(x)$

b. $Offspring(x, y) \wedge Horse(y) \Rightarrow Horse(x)$

c. $Horse(Bluebeard)$

d. $Parent(Bluebeard, Charlie)$

e. $Offspring(x, y) \Rightarrow Parent(y, x)$
$Parent(x, y) \Rightarrow Offspring(y, x)$
(Note we couldn't do $Offspring(x, y) \Leftrightarrow Parent(y, x)$ because that is not in the form expected by Generalized Modus Ponens.)

f. $Mammal(x) \Rightarrow Parent(G(x), x)$ (here G is a Skolem function).

a. 因为规则b：一匹马的后代是马

$$Offspring(x, y) \wedge Horse(y) \Rightarrow Horse(x)$$

得到无限循环

b. 容易证明：Bluebeard 和 Charlie 都是马

13.8 在一年一度的体检之后，医生告诉你一个好消息和一个坏消息。坏消息是你在一种严重疾病的测试中结果呈阳性，而这个测试的精度为 99%（即当你确实患这种病时，测试结果为阳性的概率为 0.99，而当你未患这种疾病时测试结果为阴性）。好消息是，这是一种罕见的病，在你这个年龄段大约 10000 人中才有 1 例。为什么“这种病很罕见”对于你而言是一个好消息？你确实患有这种病的概率是多少？

记患病为事件A，检测结果为阳性为事件B，那么$\mathrm{P}(B|A)=P(\bar{B}|\bar{A})=0.99$，$P(\mathrm{A})=0.0001$

$$\mathrm{P}(\mathrm{A}\,|\,\mathrm{B})=\frac{\mathrm{P}(\mathrm{B}|\mathrm{A})\mathrm{P}(\mathrm{A})}{\mathrm{P}(\mathrm{B}|\mathrm{A})\mathrm{P}(\mathrm{A})+\mathrm{P}(\mathrm{B}|\bar{A})\mathrm{P}(\bar{A})}=\frac{0.99\times0.0001}{0.99\times0.0001+0.01\times0.9999}=0.009804$$

设$P(A)=p$，则：

$$P(A|B)=\frac{0.99p}{0.99p+0.01(1-p)}=\frac{0.99}{0.98+0.01/p}$$

$P(A|B)$与p成正比，所以“这种病很罕见”是一个好消息。

如果连续测两次都是阳性，患病的概率？

13.11 假设给你一只装有 n 个无偏差硬币的袋子，并且告诉你其中 $n-1$ 个硬币是正常的，一面是正面而另一面是反面。不过剩余 1 枚硬币是伪造的，它的两面都是正面。

a. 假设你把手伸进口袋均匀随机地取出一枚硬币，把它抛出去，并发现硬币落地后正面朝上。那么你拿到伪币的（条件）概率是多少？

b. 假设你不停地抛这枚硬币，拿到它之后一共抛了 k 次而且看到 k 次正面向上。那么现在你拿到伪币的条件概率是多少？

c. 假设你希望通过把取出的硬币抛掷 k 次的方法来确定它是不是伪造的。如果 k 次抛掷后都是正面朝上，那么决策过程返回 FAKE（伪造），否则返回 NORMAL（正常）。这个过程发生错误的（无条件）概率是多少？

a. 记拿到假币为事件A，正面朝上为事件B，$P(A|B)=\frac{\frac{1}{n}\times 1}{\frac{1}{n}\times 1+\frac{n-1}{n}\times\frac{1}{2}}=\frac{2}{n+1}$

b. K次正面朝上为事件C，$P(C|A)=1, P(C|\bar{A})=\frac{1}{2^k}$，把上面的B换成C即可得到结果$\mathrm{P(A|C)}=\frac{2^k}{2^k+n-1}$

c. “发生错误”就是指$\mathrm{P}(\bar{C},\mathrm{A})+P(C,\bar{A})=0+\frac{1}{2^k}\times\frac{n-1}{n}$

13.18 文本分类是在文档所包含的文本基础上，把给定的文档分配到固定类别集合中某一个类别的任务。这个任务中常常用到朴素贝叶斯模型。在这些模型中，查询变量是文档类别，“结果”变量则是语言中每个词是否出现。我们假设文档中的词的出现都是独立的，其出现频率由文档类别确定。

a. 准确地解释当给定一组类别已经确定的文档作为“训练数据”时，这样的模型是如何构造的。

b. 准确地解释如何对新文档进行分类。

c. 这里独立性假设合理吗？请讨论。

$$P_{post} = P(Y|X) = \frac{P(Y)\prod_{i=1}^{d} P(x_i|Y)}{P(X)}$$

a. 模型由先验概率P(category)和条件概率P(word i|category)组成。对于每一类来说，利用所有文档中的属于类c的那一部分文档，来近似估计P(category=c)。类似的，用属于类c的那一部分文档中包含w单词i的文档来近似估计P(word i=true|category=c)。

b. 当得到一个新文档时，根据新文档中包含的词来判断文档是否包含某个词$word_i$，最后来计算条件概率$P(category=c|\cdots,w_i\cdots,w_j,\cdots)$

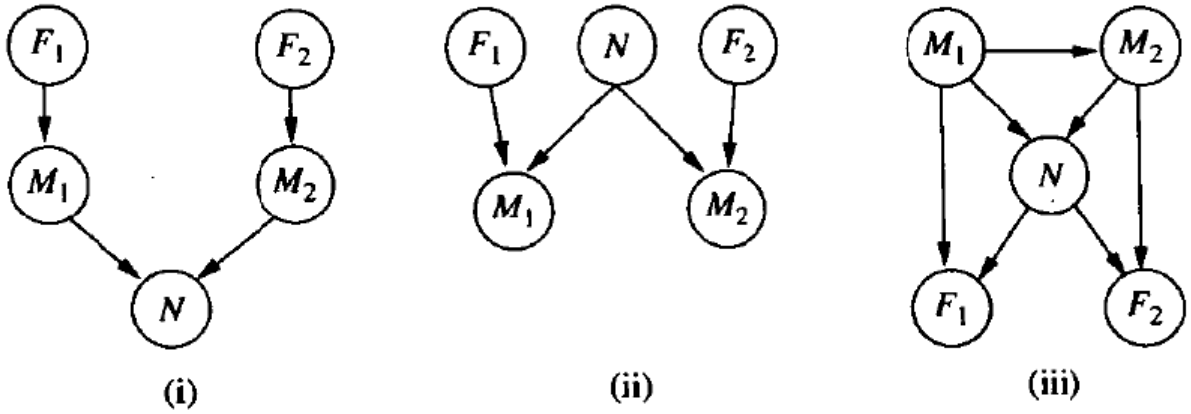
c. 不合理。因为实际文档中上下文（context）的单词间存在关联性。

14.3 两个来自世界上不同地方的宇航员同时用他们自己的望远镜观测了太空中某个小区域内恒星的数目 N。记这两个宇航员的测量结果分别为 M_1 和 M_2。通常，测量中会有不超过 1 颗恒星的误差，发生错误的概率 e 很小。而每台望远镜可能发生（发生的概率 f 更小一些）对焦不准确（分别记作 F_1 和 F_2），在这种情况下科学家会少数 3 颗甚至更多的恒星（或者说，当 N 小于 3 时，连一颗恒星都观测不到）。考虑图 14.19 所示的 3 种贝叶斯网络结构。

a. 这 3 种网络结构哪些是对上述信息的正确（但不一定是高效的）表示？

b. 哪一种网络结构是最好的？请解释。

c. 当 $N \in \{1, 2, 3\}$，$M_1 \in \{0, 1, 2, 3, 4\}$时，写出一个关于 $\mathbf{P}(M_1 \mid N)$的条件概率表。概率分布表里的每个条目都应该表达为参数 e 和/或 f 的一个函数。



(ii) F1, F2, N, M1, M2

(iii) M1, M2, N, F1, F2

a.写出联合概率的的公式，以（ii）为例：

$$P(F_1, F_2, N, M_1, M_2) = P(F_1)P(F_2)P(N)P(M_1|F_1, N)P(M_2|F_2, N)$$

（ii）和（iii）正确；（i）错误；

b.（ii）更好

c. 当 $N \in \{1, 2, 3\}$，$M_1 \in \{0, 1, 2, 3, 4\}$时，写出一个关于 $\mathbf{P}(M_1|N)$的条件概率表。
概率分布表里的每个条目都应该表达为参数 e 和/或 f 的一个函数。

$$\begin{aligned}\mathbf{P}(M_1|N) &= \mathbf{P}(M_1|N, F_1)\mathbf{P}(F_1|N) + \mathbf{P}(M_1|N, \neg F_1)\mathbf{P}(\neg F_1|N) \\ &= \mathbf{P}(M_1|N, F_1)\mathbf{P}(F_1) + \mathbf{P}(M_1|N, \neg F_1)\mathbf{P}(\neg F_1)\end{aligned}$$

	$N=1$	$N=2$	$N=3$
$M_1=0$	f + e(1-f)	f	f
$M_1=1$	(1-2e)(1-f)	e(1-f)	0.0
$M_1=2$	e(1-f)	(1-2e)(1-f)	e(1-f)
$M_1=3$	0.0	e(1-f)	(1-2e)(1-f)
$M_1=4$	0.0	0.0	e(1-f)

（1）分为对焦准确，对焦不准两种情况；

（2）多数一颗和少数一颗的概率都是e，正确的概率是（1-2e）；

注：另一种理解为多数一颗或少数一颗的概率是e/2，正确概率是（1-e），均可。

14.13 考虑图 14.22(ii)的网络，假设两个望远镜完全相同。$N \in \{1,2,3\}$，$M_1, M_2 \in \{0,1,2,3,4\}$，CPT 表和习题 14.12 所描述的一样。使用枚举算法（图 14.9）计算概率分布 $\mathbf{P}(N \mid M_1=2, M_2=2)$。

$$P(N|M1=2,M2=2)=\frac{1}{P(M1=2,M2=2)}P(N,M1=2,M2=2)$$
$$=\frac{1}{\mathrm{P}(M1=2,M2=2)}\sum_{F1,F2}P(N,M1=2,M2=2,F1,F2)$$
$$=\frac{1}{\mathrm{P}(M1=2,M2=2)}\sum_{F1,F2}P(N)P(F1)F(F2)P(M1=2|F1,N)P(M2=2|F2,N)$$

由于N取值不大于3，F1=true 或 F2=true时，相应的概率$P(M1=2|F1,N)$或$(M2=2|F2,N)$ 为零，因此上式总只有一项（F1=false, 且F2=false时) 非零

$$P(N|M1=2,M2=2)=\frac{1}{P(M1=2,M2=2)}P(N)P(F1=false)F(F2=false)P(M1=2|F1=false,N)P(M2=2|F2=false,N)$$
$$=\frac{1}{P(M1=2,M2=2)}P(N)(1-f)^2P(M1=2|F1=false,N)P(M2=2|F2=false,N)$$

$N=1$时，$P(N=1|M1=2,M2=2)=\frac{1}{P(M1=2,M2=2)}P(N=1)(1-f)^2e^2$

同理，

$$P(N=2|M1=2,M2=2)=\frac{1}{P(M1=2,M2=2)}P(N=2)(1-f)^2(1-2e)^2$$

$$P(N=3|M1=2,M2=2)=\frac{1}{P(M1=2,M2=2)}P(N=3)(1-f)^2e^2$$

14.7 这道习题是关于图 14.10 所示的变量消元算法的：

a. 第 14.4 节对如下查询应用了变量消元算法

$$\mathbf{P}(Burglary \mid JohnCalls = true, MaryCalls = true)$$

执行必要的计算，并检验计算结果的正确性。

$$
\begin{aligned}
& P(B|j,m) \\
&= \alpha P(B) \sum_e P(e) \sum_a P(a|b,e) P(j|a) P(m|a) \\
&= \alpha P(B) \sum_e P(e) \left[.9 \times .7 \times \begin{pmatrix} .95 & .29 \\ .94 & .001 \end{pmatrix} + .05 \times .01 \times \begin{pmatrix} .05 & .71 \\ .06 & .999 \end{pmatrix} \right] \\
&= \alpha P(B) \sum_e P(e) \begin{pmatrix} .598525 & .183055 \\ .59223 & .0011295 \end{pmatrix} \\
&= \alpha P(B) \left[.002 \times \begin{pmatrix} .598525 \\ .183055 \end{pmatrix} + .998 \times \begin{pmatrix} .59223 \\ .0011295 \end{pmatrix} \right] \\
&= \alpha \begin{pmatrix} .001 \\ .999 \end{pmatrix} \times \begin{pmatrix} .59224259 \\ .001493351 \end{pmatrix} \\
&= \alpha \begin{pmatrix} .00059224259 \\ .0014918576 \end{pmatrix} \\
&\approx \langle .284, .716 \rangle
\end{aligned}
$$

10次乘法，4次加法

4次乘法，2次加法

2次乘法

1次加法，2次除法

b. 统计计算中算术运算的次数，将其与枚举算法完成的运算次数进行比较。

c. 假设贝叶斯网络具有*链式*结构，即由一个布尔随机变量序列 $X_1, \ldots, X_n$ 构成，其中 $Parents(X_i) = \{X_{i-1}\}$，对于 $i = 2, \ldots, n$。请问通过枚举算法计算 $\mathbf{P}(X_1 \mid X_n = true)$的复杂度是什么？用变量消元算法呢？

b. 共有7次加法，16次乘法，2次除法

而枚举运算需要7次加法，22次乘法，2次除法

c.枚举：两个深度（n-2）完全二叉树，$O(2^n)$

变量消元：

$$
\begin{aligned}
&\mathbf{P}(X_1|X_n = true) \\
&= \alpha\mathbf{P}(X_1)\ldots\sum_{x_{n-2}} P(x_{n-2}|x_{n-3})\sum_{x_{n-1}} P(x_{n-1}|x_{n-2})P(X_n = true|x_{n-1}) \\
&= \alpha\mathbf{P}(X_1)\ldots\sum_{x_{n-2}} P(x_{n-2}|x_{n-3})\sum_{x_{n-1}} \mathbf{f}_{X_{n-1}}(x_{n-1}, x_{n-2})\mathbf{f}_{X_n}(x_{n-1}) \\
&= \alpha\mathbf{P}(X_1)\ldots\sum_{x_{n-2}} P(x_{n-2}|x_{n-3})\mathbf{f}_{\overline{X_{n-1}X_n}}(x_{n-2})
\end{aligned}
$$

最后一行中最后一项的计算只与x_{n-1}, x_{n-2}相关，而该过程的代价独立于n，为一常量，故复杂度为O(n)

- 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。

作业

- 已知正例点 $x_1=(1,2)^{\mathrm{T}}, x_2=(2,3)^{\mathrm{T}}, x_3=(3,3)^{\mathrm{T}}$，

  负例点 $x_4=(2,1)^{\mathrm{T}}, x_5=(3,2)^{\mathrm{T}}$， 试求Hard Margin SVM的最大间隔分离超平面和分类决策函数，并在图上画出分离超平面、间隔边界及支持向量。

假设超平面 $w^T x + b = 0$

$$\begin{cases} \min \frac{1}{2}\|w\|^2 \\ s.t.\ \ y_i(w^T x_i + b) \geq 1 \end{cases}$$

$$x_1 = \begin{pmatrix}1\\2\end{pmatrix} \quad x_2 = \begin{pmatrix}2\\3\end{pmatrix} \quad x_3 = \begin{pmatrix}3\\3\end{pmatrix}$$

$$x_4 = \begin{pmatrix}2\\1\end{pmatrix} \quad x_5 = \begin{pmatrix}3\\2\end{pmatrix} \quad y = (1,1,1,-1,-1)$$

- 拉格朗日对偶

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j ({x_i}^T x_j)$$

凹函数

$$\text{subject to } \alpha_i \geq 0 \,, \sum_{i=1}^{n} \alpha_i y_i = 0$$

$\alpha_1 + \alpha_2 + \alpha_3 - \alpha_4 - \alpha_5 = 0$ 代入目标函数，有

$$\max \ \ 2(\alpha_1 + \alpha_2 + \alpha_3) - \frac{1}{2}(4{\alpha_1}^2 + 2\,{\alpha_2}^2 + {\alpha_3}^2 + 2{\alpha_4}^2 + 4\alpha_1\alpha_2 + 6\alpha_1\alpha_4 + 2\alpha_2\alpha_3 + 2\alpha_3\alpha_4)$$

首先，求解上式各偏导/梯度=0

α_1 =-0.4<0，$\alpha_2 = 1.2$，$\alpha_3 = \alpha_4 = 0$

不满足约束条件，故最大值一定在可行域的边界处取到；又从上述目标函数可以直观地看出最大值处 α_4 必为0 （因为含有α_4的项为 $-{\alpha_4}^2 - 3\alpha_1\alpha_4 - \alpha_3\alpha_4$ ）

因此上式简化为

$$\max\ 2(\alpha_1 + \alpha_2 + \alpha_3) - \frac{1}{2}(4{\alpha_1}^2 + 2\,{\alpha_2}^2 + {\alpha_3}^2 + 4\alpha_1\alpha_2 + 2\alpha_2\alpha_3)$$

求解上式各偏导/梯度=0，无解；故α_1 , α_2 α_3至少有一个为0，即边界值

α_1 =0时， 求解偏导为0，可得 α_2 =0 ， α_3 =2，f=2

α_2=0时， 求解偏导为0，可得 α_1 =0.5 ， α_3 =2，f=2.5

α_3 =0时， 求解偏导为0，可得 α_1 =0 ， $\alpha_2 = 1$，f=1

$$\alpha_1 = \frac{1}{2} \qquad \alpha_2 = 0 \qquad \alpha_3 = 2 \quad \alpha_4 = 0 \qquad \alpha_5 = \frac{5}{2}$$

$$w = \sum \alpha_i y_i x_i = \begin{pmatrix} -1 \\ 2 \end{pmatrix} \qquad b = -2$$

决策函数：f(x)=sign(-$x_1 + 2x_2 - 2$)

- Training (i.e., finding the parameter w) can be done by maximizing the conditional log likelihood of training data $\{(\mathbf{x}_i, y_i)\}_{i=1:N}$

$$\max_{\mathbf{w}} \ell(\mathbf{w}) = \max_{\mathbf{w}} \sum_{i=1}^{N} \log p(y_i|\mathbf{x}_i; \mathbf{w})$$

or $$\min_{\mathbf{w}} J(\mathbf{w}) = \min_{\mathbf{w}} -\ell(\mathbf{w})$$

$$= \min_{\mathbf{w}} -\left[\sum_{i=1}^{N} y_i \log \sigma(\mathbf{w}^\top \mathbf{x}_i) + (1 - y_i) \log\,(1 - \sigma(\mathbf{w}^\top \mathbf{x}_i))\right]$$

- Want $\min_{\mathbf{w}} J(\mathbf{w})$

Repeat {

$$\mathrm{w}_j := \mathrm{w}_j - \alpha \frac{\partial}{\partial \mathrm{w}_j} J(\mathbf{w})$$

} (simultaneously update all w_j)

- A useful fact

$$\frac{\partial}{\partial z}\sigma(z) = \frac{\partial}{\partial z}\frac{1}{1+e^{-z}} = \underbrace{-\left(\frac{1}{1+e^{-z}}\right)^2}_{\partial\sigma/\partial(1+e^{-z})} \times \underbrace{-e^{-z}}_{\partial(1+e^{-z})/\partial z}$$

$$= \sigma^2(z)\left(\frac{1-\sigma(z)}{\sigma(z)}\right) = \sigma(z)(1-\sigma(z)).$$

- Compute $\frac{\partial}{\partial \mathrm{W}_j} J(\mathbf{w})$

$$\frac{\partial}{\partial W_j} J(W) = -\sum_{i=1}^{N} y_i \frac{1}{\sigma(z)}\sigma(z)(1-\sigma(z))x_i + (1-y_i)\frac{-1}{1-\sigma(z)}\sigma(z)(1-\sigma(z))x_i$$

$$= -\sum_{i=1}^{N} y_i(1-\sigma(z))x_i - (1-y_i)\sigma(z)x_i \qquad z = W^T x_i$$