南京大学 人工智能学院
SCHOOL OF ARTIFICIAL INTELLIGENCE, NANJING UNIVERSITY

第六部分：部分可观察环境中的序贯决策系统

章宗长

2021年6月16日

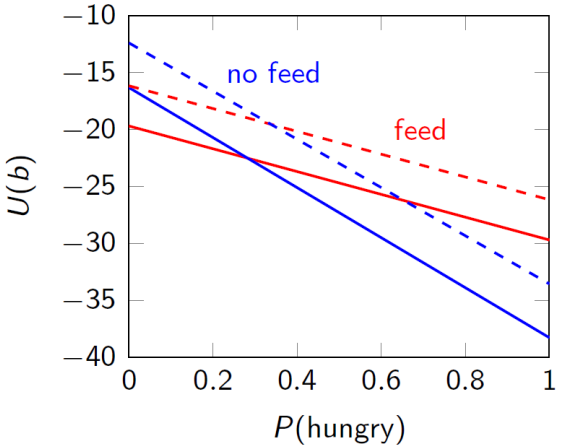
内容安排

- 部分可观察的MDP
- 信念状态更新
- 精确求解方法
- **离线方法**
- 在线方法
- 应用案例：飞行器避碰系统

完全可观察的近似：QMDP

假设所有状态的不确定性在下一个时刻消失

- 创建一个阿尔法向量的集合
  - 每个行动对应一个向量
  - 每个向量由完全可观察MDP的$Q(s,a)$构成



虚线：用QMDP求得的值函数

- 使用值迭代来计算阿尔法向量
  - 初始化：对所有s，$\alpha_a^{(0)}(s) = 0$
  - 迭代：

$$\alpha_a^{(k+1)}(s) = R(s,a) + \gamma \sum_{s'} T(s' \mid s,a) \max_{a'} \alpha_{a'}^{(k)}(s')$$

每轮迭代的运算次数：$O(|\mathcal{A}|^2|\mathcal{S}|^2)$

提供了最优值函数的上界，即对所有$\mathbf{b}$，有$\max_a \boldsymbol{\alpha}_a^\top \mathbf{b} \geq U^*(\mathbf{b})$

- 信念状态$\mathbf{b}$的值函数：$\max_a \boldsymbol{\alpha}_a^\top \mathbf{b}$，近似最优行动：$\arg\max_a \boldsymbol{\alpha}_a^\top \mathbf{b}$

快速获知界

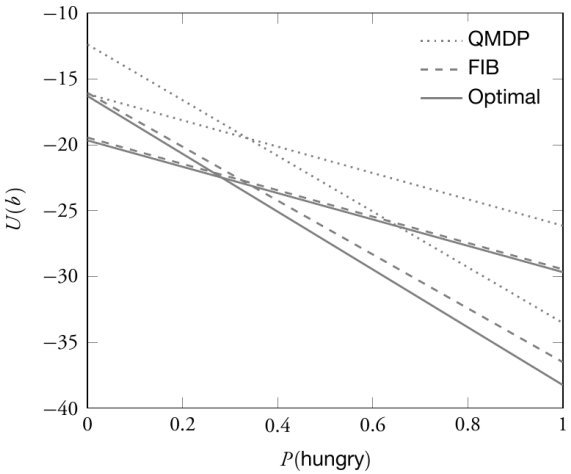
- 快速获知界（Fast Informed Bound, FIB）
  - 与QMDP类似，也计算每个行动的阿尔法向量

- FIB在一定程度上考虑部分可观察性：

$$\alpha_a^{(k+1)}(s) = R(s,a) + \gamma \sum_o \max_{a'} \sum_{s'} O(o \mid s', a) T(s' \mid s, a) \alpha_{a'}^{(k)}(s')$$

每轮迭代的运算次数：$O(|\mathcal{A}|^2|\mathcal{S}|^2|\mathcal{O}|)$

- FIB可以获得比QMDP更接近最优值函数的上界



最优值函数及其上界：QMDP & FIB

基于点的值迭代

点：信念状态、信念点

- 更新在信念状态空间中，与有限个点关联的阿尔法向量
- 信念点的集合：$B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$
- 与它们关联的阿尔法向量的集合：$\Gamma = \{\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_n\}$
- 给定这些阿尔法向量，可以估计在任一信念点$\mathbf{b}$处的值函数：

$$U^{\Gamma}(\mathbf{b}) = \max_{\alpha \in \Gamma} \boldsymbol{\alpha}^{\top}\mathbf{b} = \max_{\alpha \in \Gamma} \sum_{s} \alpha(s) b(s)$$

- 初始化在Γ中的阿尔法向量，使得对所有$\mathbf{b}$，有$U^{\Gamma}(\mathbf{b}) \leq U^{*}(\mathbf{b})$
- 一种计算下界的方法：初始化n个阿尔法向量的所有元素为

$$\max_{a} \sum_{t=0}^{\infty} \gamma^{t} \min_{s} R(s,a) = \frac{1}{1-\gamma} \max_{a} \min_{s} R(s,a)$$

基于点的值迭代（续）

- 基于初始的阿尔法向量集合，可以更新在**b**处的值函数：

$$U(b) \leftarrow \max_a \left[R(b,a) + \gamma \sum_o P(o \mid b,a) U(b') \right]$$

$$P(o \mid b,a) = \sum_s b(s) \sum_{s'} O(o \mid s',a) T(s' \mid s,a)$$

$$b'(s') = \frac{O(o \mid s',a)}{P(o \mid b,a)} \sum_s T(s' \mid s,a) b(s)$$

- 结合这几个方程，可以得到更新公式：

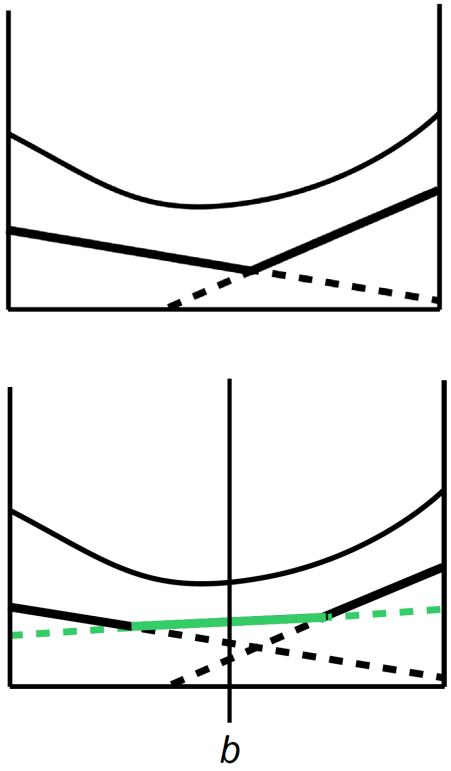
$$U(b) \leftarrow \max_a \left[R(b,a) + \gamma \sum_o \max_{\alpha \in \Gamma} \sum_s b(s) \sum_{s'} O(o \mid s',a) T(s' \mid s,a) \alpha(s') \right]$$

基于点的值迭代（续）

- 除了简单地更新**b**处的值，还可以计算**b**处的阿尔法向量：

Algorithm 6.4 Backup belief

```
1: function BackupBelief(Γ, b)
2:     for a ∈ A
3:         for o ∈ O
4:             b′ ← UpdateBelief(b, a, o)
5:             α_{a,o} ← arg max_{α∈Γ} α⊤b′
6:         for s ∈ S
7:             α_a(s) ← R(s,a) + γ Σ_{s′,o} O(o | s′, a) T(s′ | s, a) α_{a,o}(s′)
8:     α ← arg max_{α_a} α_a⊤ b
9:     return α
```



用这些阿尔法向量近似信念状态空间中任意点的函数值

- 基于点的值迭代算法更新n个信念状态的阿尔法向量，直至收敛

小结：离线方法

- 获得最优值函数上界的方法
  - 完全可观察的近似：QMDP
    - 假设所有状态的不确定性在下一个时刻消失
  - 快速获知界：FIB
    - 在一定程度上考虑部分可观察性
- 获得最优值函数下界的方法：基于点的值迭代
  - 基本做法：选择有限个信念点，更新与其相关联的选择

内容安排

- 部分可观察的MDP
- 信念状态更新
- 精确求解方法
- 离线方法
- **在线方法**
- 应用案例：飞行器避碰系统

使用近似值函数前向看

- 在线使用一步前向看的策略来改进离线计算出的策略
- 如果b是当前信念状态，则一步前向看的策略为：

$$\pi(b) = \arg\max_a \left[R(b,a) + \gamma \sum_o P(o \mid b,a) U(\text{UpdateBelief}(b,a,o)) \right]$$

- 用阿尔法向量表示的**近似值函数**
  - 用QMDP、FIB、基于点的值迭代等方法估计
  - 用滚轮（rollout）策略估计

在观察空间很大时，这一策略很有用

- 使用采样来代替对所有可能的观察求和
  - 对每个行动，独立地调用 $\text{SampleObservation}(b,a)$ 来产生n个观察，然后计算

$$\pi(b) = \arg\max_a \left[R(b,a) + \gamma \frac{1}{n} \sum_{i=1}^{n} U(\text{UpdateBelief}(b,a,o_{a,i})) \right]$$

使用近似值函数前向看（续）

- 算法6.9：用滚轮策略来估计近似值函数
  - 改进了算法4.10，可以处理部分可观察性
  - 在产生式模型G的输出中，多了观察o
  - 使用的是单一的滚轮策略

Algorithm 6.9 Rollout evaluation

```
function ROLLOUT(b, d, π_0)
    if d = 0
        return 0
    a ~ π_0(b)
    s ~ b
    (s', o, r) ~ G(s, a)
    b' ← UPDATEBELIEF(b, a, o)
    return r + γROLLOUT(b', d − 1, π_0)
```

- 改进：使用一个滚轮策略的集合
  - 并行地评价它们
  - 使用有最大值的策略来估计特定信念状态处的值
- 在很多问题上，一步前向看可以显著地改进策略的性能

前向搜索

- 把一步前向搜索推广至多步前向搜索（算法6.10）
  - 当$d = 0$时，不选择行动，回报为$U(b)$
  - 当$d > 0$时，计算每一个行动的值，返回最好的行动和它的关联值

Algorithm 6.10 Forward search online policy

```
function SelectAction(b, d)
    if d = 0
        return (NIL, U(b))
    (a*, u*) ← (NIL, −∞)
    for a ∈ A
        u ← R(b, a)
        for o ∈ O
            b′ ← UpdateBelief(b, a, o)
            (a′, u′) ← SelectAction(b′, d − 1)
            u ← u + γ P(o | b, a) u′
        if u > u*
            (a*, u*) ← (a, u)
    return (a*, u*)
```

通过下式计算一个行动a的值：

$$R(b,a)+\gamma\sum_{o}P(o\mid b,a)U_{d-1}(\text{UpdateBelief}(b,a,o))$$

- 时间复杂度为$O\left(|\mathcal{S}|^2\times(|\mathcal{A}|\times|\mathcal{O}|)^d\right)$
- 若改为采样n个观察，则复杂度变为$O\left(|\mathcal{S}|^2\times(|\mathcal{A}|\times n)^d\right)$

分支限界搜索

- POMDP版本的分支限界搜索
  - 使用QMDP或FIB来计算上界函数$\bar{Q}$
  - 使用盲策略的值函数作为下界函数$\underline{U}$

盲策略：不论当前信念状态是什么，都选择相同的行动

这个下界函数有$|\mathcal{A}|$个阿尔法向量，计算如下：

$$\alpha_a^{(k+1)}(s) = R(s,a) + \gamma \sum_{s'} T(s' \mid s,a)\alpha_a^{(k)}(s')$$

其中 $\alpha_a^{(0)} = \min_s R(s,a)/(1-\gamma)$

Algorithm 6.11 Branch and bound online policy

```
function SelectAction(b, d)
    if d = 0
        return (NIL, U(b))
    (a*, u) ← (NIL, −∞)
    for a ∈ A
        if Q̄(b, a) ≤ u
            return (a*, u)
        u ← R(b, a)
        for o ∈ O
            b′ ← UpdateBelief(b, a, o)
            (a′, u′) ← SelectAction(b′, d − 1)
            u ← u + γ P(o | b, a) u′
        if u > u
            (a*, u) ← (a, u)
    return (a*, u)
```

行动应该按上界降序排列：如果行动a_i在a_j之前被评价，则$\bar{Q}(b,a_i) \geq \bar{Q}(b,a_j)$

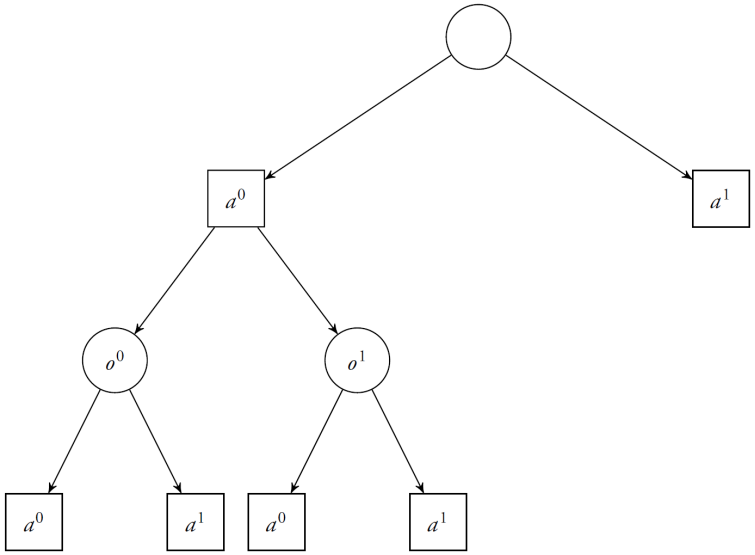
需要遍历观察空间和更新信念状态

- 上下界之差越小，可裁剪的搜索区域越多，计算时间越少
- 最坏时间复杂度与前向搜索相同

蒙特卡洛树搜索

- POMDP版本的蒙特卡洛树搜索
  - 主要的不同：计数和值与历史关联，而不是状态
- 历史
  - 过去行动和观察的序列
  - 如果有2个行动（a^0和a^1）和2个观察（o^0和o^1），则一个可能的历史：$h = a^0 o^1 a^1 o^1 a^0 o^0$
- 历史树
  - 根结点：从当前信念状态b开始的空历史
  - 树的各层在行动结点和观察结点之间交替
  - 在算法执行过程中，历史树将扩展

在历史树中，与每个行动结点关联的是历史-行动对的值估计$Q(h,a)$和计数$N(h,a)$，其中h由根结点到该结点的路径决定



示例：蒙特卡洛树搜索中的历史树

蒙特卡洛树搜索（续）

Algorithm 6.12 Monte Carlo tree search

```
1:  function SELECTACTION(b, d)
2:      h ← ∅
3:      loop
4:          s ~ b
5:          SIMULATE(s, h, d)
6:      return arg max_a Q(h, a)
7:  function SIMULATE(s, h, d)
8:      if d = 0
9:          return 0
10:     if h ∉ T
11:         for a ∈ A(s)
12:             (N(h, a), Q(h, a)) ← (N_0(h, a), Q_0(h, a))
13:         T = T ∪ {h}
14:         return ROLLOUT(s, d, π_0)
15:     a ← arg max_a Q(h, a) + c sqrt(log N(h) / N(h, a))
16:     (s', o, r) ~ G(s, a)
17:     q ← r + γ SIMULATE(s', hao, d − 1)
18:     N(h, a) ← N(h, a) + 1
19:     Q(h, a) ← Q(h, a) + (q − Q(h, a)) / N(h, a)
20:     return q
```

Anytime：循环可以随时终止，终止后返回某一解

如果循环次数足够多，则算法会收敛至最优行动

在算法执行的过程中，更新一组值估计$Q(h,a)$和计数$N(h,a)$，$N(h)=\sum_a N(h,a)$

- 初始化
  - 初始化N_0、Q_0和滚轮策略π_0时，可以使用先验知识
  - 不必在每步决策后重新初始化
- 在下一个时间步，与所选行动和实际观察关联的观察结点变为根结点

小结：在线方法

- 使用近似值函数前向看
  - 估计近似值函数：滚轮策略、QMDP、FIB、基于点的值迭代
  - 计算一步前向看策略：观察空间求和、采样
- 前向搜索：多步前向看
- 分支限界搜索
  - 前向搜索的一种扩展
  - 使用最优值函数的上下界使得前向搜索更高效
- 蒙特卡洛树搜索
  - 计数和值与历史关联，而不是状态

内容安排

- 部分可观察的MDP
- 信念状态更新
- 精确求解方法
- 离线方法
- 在线方法
- **应用案例：飞行器避碰系统**

应用案例：飞行器避碰系统

- **TCAS vs. ACAS X**
- 避碰问题的建模和离线求解
- 实时执行和性能评价

空中相撞事件

- 1956年，在科罗拉多大峡谷，飞机空中相撞造成128人死亡
- 1978年，在加州圣地亚哥市，飞机空中相撞造成144人死亡
- 1986年，在加州喜瑞都市，飞机空中相撞造成82人死亡
- …

交通警报和避碰系统（TCAS）

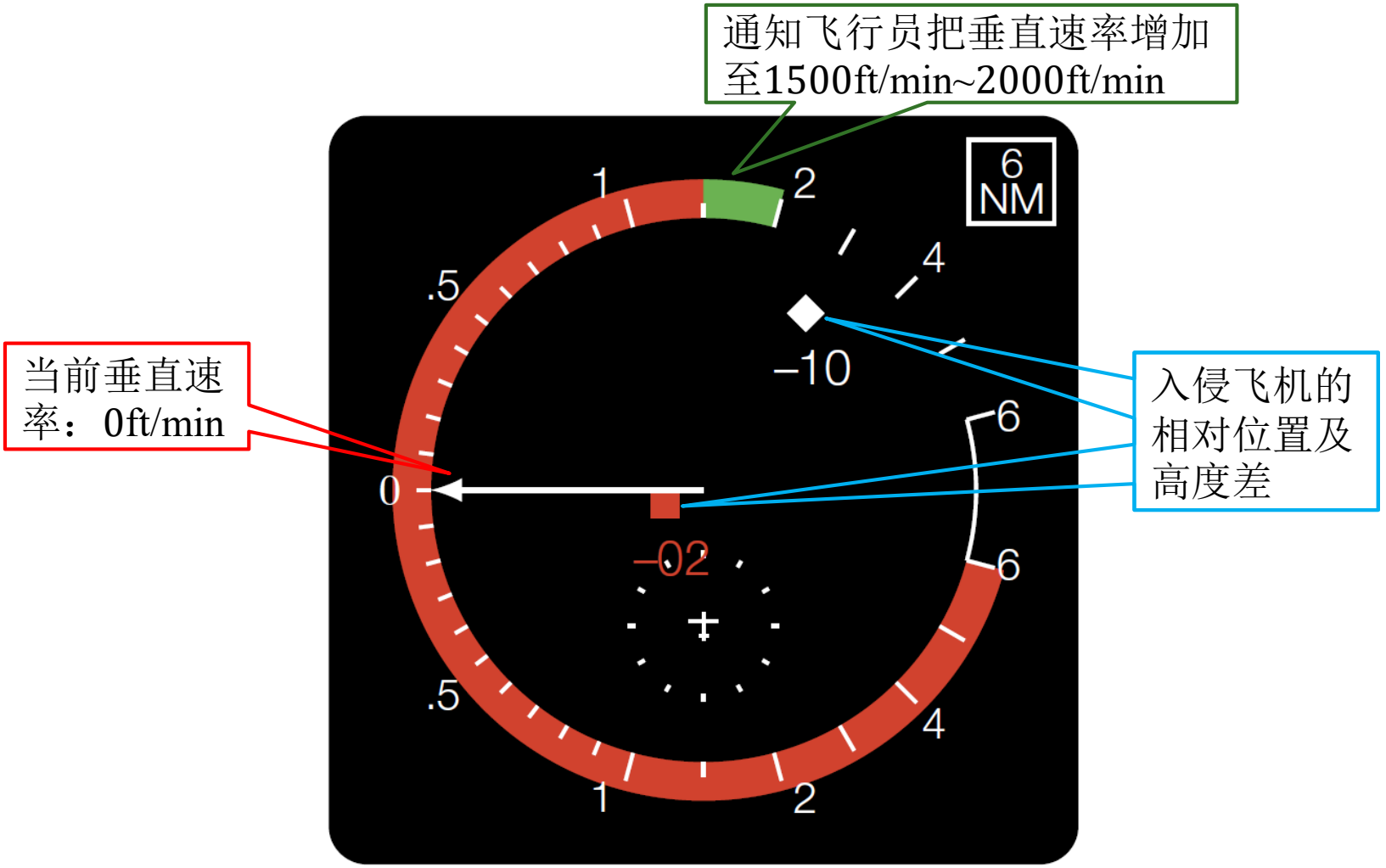
- 灯塔避碰系统（BCAS）
  - 仅能用于低密度空域
- 交通警报和避碰系统（TCAS）
  - 基于BCAS，改进在于能用在高密度空域中
  - 广泛用于各类商用飞机中
- TCAS提供两类警报
  - 交通咨询（Traffic Advisory, TA）
    - 如“Traffic, Traffic”
    - 提醒飞行员有入侵的飞机，做好避碰准备
  - 决议咨询（Resolution Advisory, RA）
    - 指示飞行员上升或下降以保持安全距离

交通警报和避碰系统（续）

- TCAS仅提供不同垂直操作的咨询：
  - 不要上升或下降
  - 限制上升或下降至500、1000、2000ft/min
  - 水平飞行
  - 以1500ft/min上升或下降
  - 保持当前的垂直速率

- TCAS把何时报警和发出何种咨询表示为一个大的规则集合
  - 启发式参数设置：补偿传感器噪声、误差及飞机反应的差异性
  - 控制何时加强、减弱、逆转咨询
  - 处理有多架飞机同时入侵的情况

交通警报和避碰系统（续）



TCAS的显示板

交通警报和避碰系统（续）

- TCAS的局限性：
  - 基于一个确定的模型设计，限制了系统的鲁棒性
  - 不支持安全性和新空域的操作需求
    - 卫星导航：允许飞机飞得更近
    - 新的飞机类型：无人机
  - 针对大飞机设计
  - 若要用到小飞机上，需要重新设计，成本高昂

飞行器避碰系统（ACAS X）

- 飞行器避碰系统（ACAS X）
  - 可能成为有人驾驶和无人驾驶的飞机的下一代国际标准
  - 把避碰问题用POMDP建模，然后用动态规划求解最优策略
  - 仿真研究表明，这种方法能带来安全性和操作性能的显著提升

- ACAS X有监控和咨询逻辑上的改进
  - 使用一种即插即用的监控架构，以支持基于GPS数据的监控和提供新的传感器模式
  - 使用一张用空域模型优化得到的数值表来表示逻辑，能为新用户类型（如小飞机）提供避碰保护
    - 改进了鲁棒性，支持新需求，减少了不必要的警报
    - 简化了开发过程，减少了实现和维护成本

应用案例：飞行器避碰系统

- TCAS vs. ACAS X
- **避碰问题的建模和离线求解**
- 实时执行和性能评价

决议咨询集合

三类咨询sense：向上、向下、水平飞行
加强（strengthening）：同一类，但速率加快
减弱（weakening）：同一类，但速率减慢
逆转（reverse）：从一类到另一类

- ACAS X的决议咨询集合
  - 与表中仅一处不同，即把MCL和MDES合成了行动“maintain”

Name	Vertical Rate Range Min (ft/min)	Max (ft/min)	Aural Annunciation
COC	−∞	∞	Clear of Conflict (or nothing)
DNC2000	−∞	2000	Monitor Vertical Speed
DND2000	−2000	∞	Monitor Vertical Speed
DNC1000	−∞	1000	Monitor Vertical Speed
DND1000	−1000	∞	Monitor Vertical Speed
DNC500	−∞	500	Monitor Vertical Speed
DND500	−500	∞	Monitor Vertical Speed
DNC	−∞	0	Level-off, Level-off (or Monitor Vertical Speed)
DND	0	∞	Level-off, Level-off (or Monitor Vertical Speed)
MDES	−∞	current	Maintain Vertical Speed, Maintain
MCL	current	∞	Maintain Vertical Speed, Maintain
DES1500	−∞	−1500	Descend, Descend
CL1500	1500	∞	Climb, Climb
SDES1500	−∞	−1500	Descend, Descend NOW, Descend, Descend NOW
SCL1500	1500	∞	Climb, Climb NOW, Climb, Climb NOW
SDES2500	−∞	−2500	Increase Descent, Increase Descent
SCL2500	2500	∞	Increase Climb, Increase Climb

目标垂直速率范围

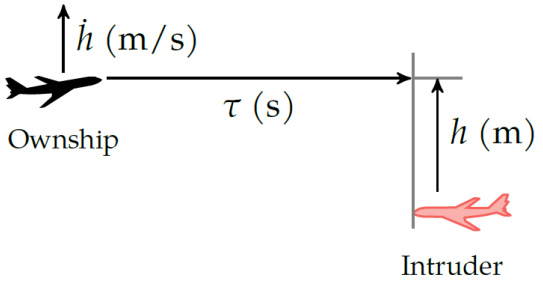
矫正性的：Level-off, Level-off
预防性的：Monitor Vertical Speed

矫正性咨询

- 如果当前垂直速率在目标范围内，则咨询是预防性的；否则，咨询是矫正性的

状态变量

- h：入侵飞机与我方飞机的高度差
- $\dot{h}_0$：我方飞机的垂直速率
- $\dot{h}_1$：入侵飞机的垂直速率
- τ：潜在的碰撞时间
- s_{adv}：当前咨询
- s_{res}：是否飞行员正响应咨询



相对简单
状态变量较少，使得用动态规划计算最优咨询不太困难

咨询响应模型

- 下一个时间步的咨询响应：
  - 飞行员总对COC响应
  - 一旦飞行员响应，在咨询过程中会持续响应
  - 飞行员对初始咨询的平均响应延迟为5s
  - 飞行员对逆转的平均响应延迟为5s
  - 飞行员对加强或减弱的平均响应延迟为3s

$$P(s'_{\text{res}} = \text{true} \mid s_{\text{adv}}, s_{\text{res}}, a) = \begin{cases} 1 & \text{if } a = \text{COC} \\ 1 & \text{if } s_{\text{res}} = \text{true and } s_{\text{adv}} = a \\ 1/(1+5) & \text{if } s_{\text{adv}} = \text{COC and } a \neq \text{COC} \\ 1/(1+5) & \text{if } s_{\text{adv}} \text{ and } a \text{ are opposite sense} \\ 1/(1+3) & \text{if } s_{\text{adv}} \text{ and } a \text{ are same sense} \end{cases}$$

响应延迟服从几何分布

- 如果每步的响应概率为$\frac{1}{k+1}$，则平均响应时间为k

状态转移模型

- 我方飞机的加速度$\ddot{h}_0$：
  - 若没在响应咨询，则从均值为0，标准差为3ft/s²的高斯分布中采样
  - 若正在响应咨询，则选择加速度使得垂直速率回到咨询所要求的范围
- 入侵飞机的加速度$\ddot{h}_1$：
  - 从均值为0，标准差为3ft/s²的高斯分布中采样
- 有了$a, s'_{\text{res}}, \ddot{h}_0$和$\ddot{h}_1$后，用下式更新状态：

一个时间步：$\Delta t = 1\text{s}$

$$\begin{bmatrix} h \\ \dot{h}_0 \\ \dot{h}_1 \\ \tau \\ s_{\text{adv}} \\ s_{\text{res}} \end{bmatrix} \leftarrow \begin{bmatrix} h + \dot{h}_1(\Delta t) + \frac{1}{2}\ddot{h}_1(\Delta t)^2 - \dot{h}_0(\Delta t) - \frac{1}{2}\ddot{h}_0(\Delta t)^2 \\ \dot{h}_0 + \ddot{h}_0(\Delta t) \\ \dot{h}_1 + \ddot{h}_1(\Delta t) \\ \tau - 1 \\ a \\ s'_{\text{res}} \end{bmatrix}$$

奖赏函数

- 不同事件的奖赏
  - 考虑了安全性和操作需求
  - 这里的奖赏函数仅依赖于6个状态变量和当前行动
- 交叉咨询
  - 当入侵飞机在下方（上方），发出向下（向上）的咨询
  - 导致两架飞机在高度上相交，应尽可能避免

高度差　　垂直速率差

Reward	Separation (ft)	Closure (ft/min)	Event
-1	≤ 175		$\tau \leq 0$
-1			Maintain advisory with $\dot{h}_0 < 1500$ ft/min
-1			Prohibited advisory transitions
-1			Preventive crossing advisory
-0.1	> 650	< 2000	Corrective advisory
-3×10^{-2}	> 1000	< 4000	Corrective advisory
-1×10^{-2}	> 650	< 2000	Preventive advisory
-1×10^{-2}	> 500		Crossing advisory
-8×10^{-3}			Reversal
-5×10^{-3}			Strengthening
-1×10^{-3}			Weakening
-1.5×10^{-3}		> 3000	Non-MVS/LOLO
-2.3×10^{-3}		< 3000	Any advisory
-5×10^{-4}		> 3000	MVS/LOLO
$-4 \times 10^{-4} \times \Delta\dot{h}$			Crossing advisory when $\lvert\dot{h}_0\rvert > 500$ ft/min and $\dot{h}_0$ is in opposite direction of advisory
-4×10^{-4}			Maintain
-1×10^{-4}			MVS/LOLO
$-3 \times 10^{-5} \times \Delta\dot{h}$			Any advisory
-1×10^{-5}			Corrective advisory
1×10^{-9}			COC

$$\Delta\dot{h} = \min(|h_{\min} - \dot{h}_0|, |h_{\max} - \dot{h}_0|)$$

我方飞机达到咨询所要求的范围在垂直速率上所需的最小变化

动态规划

- 先假设完全可观察性，可以用动态规划来计算值函数U^*，使用Bellman最优方程：

$$U^*(s) = \max_a \left(R(s,a) + \gamma \int T(s' \mid s,a) U^*(s') \, ds' \right)$$

因为加速度$\ddot{h}_0$和$\ddot{h}_1$为连续随机变量，所以T为概率密度函数

- 用采样方法对$\ddot{h}_0$和$\ddot{h}_1$离散化，并用网格方法把连续的状态空间离散化，Bellman最优方程变为：

$$U^*(s) = \max_a \left(R(s,a) + \gamma \sum_{s'} T(s' \mid s,a) U^*(s') \right)$$

高斯-赛德尔值迭代

- 用高斯-赛德尔（Gauss-Seidel）值迭代方法求解

$$U(s) \leftarrow \max_a \left(R(s,a) + \gamma \sum_{s'} T(s' \mid s,a) U(s') \right)$$

- 由$U^*(s)$计算$Q^*(s,a)$，存放在一张查找表中

- 用局部近似（如多线性插值）法估计连续状态空间中任意状态-行动对的最优Q值

QMDP离线规划

- 考虑状态变量的部分可观察性
- 系统估计信念状态b，并使用它来选择最优的行动：

QMDP $\pi^*(b) = \arg\max_a \int b(s)Q^*(s,a)ds$

- 为了高效计算，把信念状态表示为关联权重$w^{(1)}, \ldots, w^{(n)}$的一组样本$s^{(1)}, \ldots, s^{(n)}$：

$$\pi^*(b) = \arg\max_a \sum_i w^{(i)} Q^*(s^{(i)}, a)$$

信念状态因子化及更新

- 对信念状态b因子化：

飞行员响应，用贝叶斯规则更新

$$b(s)=b(h,\dot{h}_0,\dot{h}_1,\tau,s_{\text{adv}},s_{\text{res}})=b(h,\dot{h}_0,\dot{h}_1)b(\tau)b(s_{\text{adv}},s_{\text{res}})$$

传感器误差，用粒子滤波器估计和更新

- $b(\tau)$：潜在碰撞时间
  - 离线计算这个分布

$$D_0(s)=\begin{cases}1 & \text{如果另一架飞机在横向距离内}\\ 0 & \text{否则}\end{cases}$$

$$D_k(s)=\sum_{s'}T(s,s')D_{k-1}(s')$$

其他飞机在第k秒飞进一个横向距离内的概率

给定状态s，$\tau\geq 40$的概率为$1-\sum_{k=0}^{39}D_k(s)$

  - 保存在一张查找表中

应用案例：飞行器避碰系统

- TCAS vs. ACAS X
- 避碰问题的建模和离线求解
- **实时执行和性能评价**

实时执行

- ACAS X的实时执行需要完成如下过程：
  - 估计信念状态
  - 通过对查找表中的条目插值来计算关联的状态-行动值
  - 考虑更多影响成本的变量，修正状态-行动值
  - 处理多架飞机同时入侵的情况
  - 产生交通警报

在线成本

- 离线计算得到的查找表仅是一个6个变量的函数
  - 不想让要存储的查找表过大
- 在线计算时，需要考虑更多因素
  - 高度禁止：阻止在低于某一高度时发出咨询
  - 咨询切换或重启：惩罚频繁地切换或重启咨询
  - 初始化：阻止在系统开启的数秒内发出咨询
  - 多次逆转：阻止多次逆转
  - 坏的转移：阻止TCAS不允许的咨询序列

成本无限高

成本较低

多个威胁

- TCAS
  - 独立地确定每架入侵飞机的最优咨询
  - 依靠一个相对复杂的规则集合来结合这些咨询
  - 输出一个咨询给飞行员

- ACAS X
  - 针对不同的入侵飞机提供不同保护模式
    - 使用不同的查找表和不同的状态估计参数
  - 结合与不同入侵飞机关联的状态-行动值
  - 输出一个咨询给飞行员

交通警报

- 咨询（警报）
  - 决议咨询（Resolution Advisory, RA）
  - 交通咨询（Traffic Advisory, TA）
- 前面设计ACAS X时，聚焦在如何产生RA上
- TA在视觉获取和让飞机准备响应RA等方面，扮演重要角色
- 有多种方法产生优化的TA，考虑如下因素：
  - 允许适当地前置时间
  - 避免骚扰警报
  - …

交通警报（续）

- ACAS X中采用的方法
  - 不扩大或修改当前的查找表
  - 基于“无警报”行动的值来产生TA
    - 碰撞的可能性越大，无警报的值越低
- 仅设置一个阈值会带来频跳的问题
- 为了防止频跳，ACAS X采用了两种方法
  - 要求TA至少持续8s
  - 采用3个阈值：开启阈值（on threshold）、挤压阈值（squeeze threshold）、关闭阈值（off threshold）

交通警报（续）

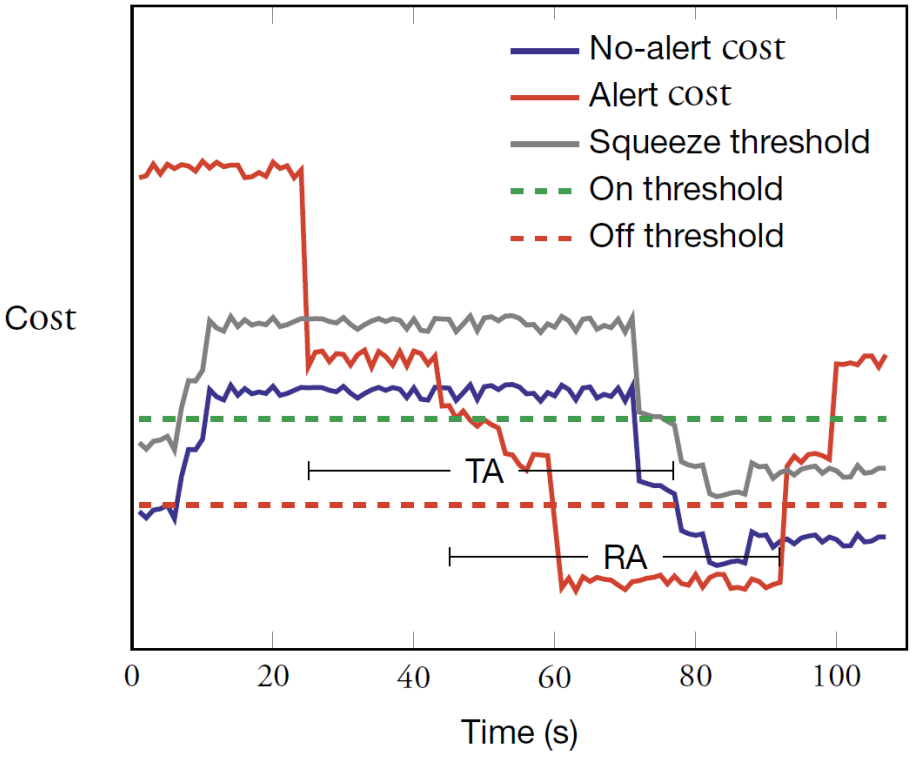
- 开启阈值和关闭阈值是常数值
- 挤压阈值是无警报值的常数偏置（offset）

当警报成本低于无警报成本与挤压阈值之和，且无警报成本高于开启阈值时，发出TA

当无警报成本低于关闭阈值时，关闭TA

当警报成本低于无警报成本时，发出RA

当警报成本高于无警报成本时，关闭RA



性能评价

- 参数调整
- 安全分析
- 操作适宜性
- 操作可接受性

参数调整

- 系统参数
  - 控制系统的行为，不需要设计者调整
- 设计参数
  - 需要设计者调整
  - 设计过程的复杂度与设计参数的数目呈指数关系
  - 评价每个设计点需要数百万次仿真

设计点：一组设计参数的具体值

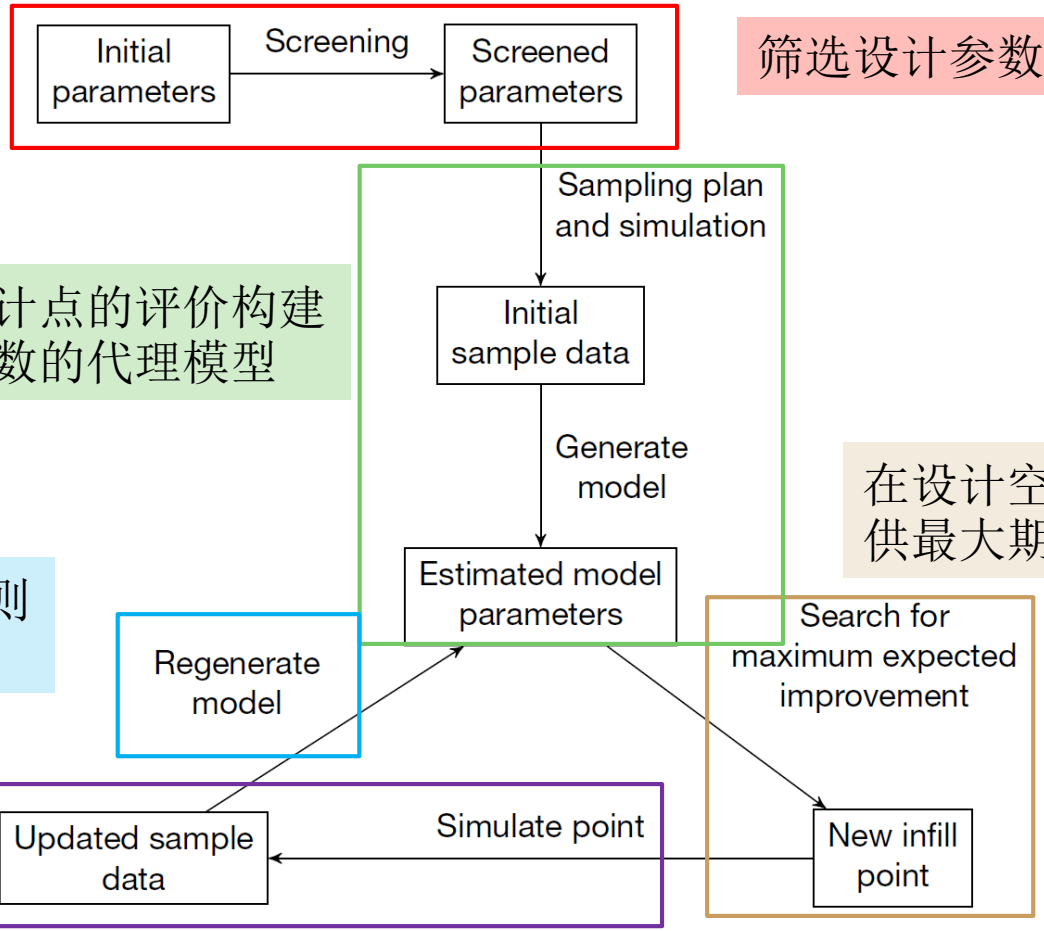
给定一个设计点的仿真结果，
如何确定下一个设计点？

- 自动机器学习（AutoML）：自动化调设计参数的过程

基于代理模型的优化过程

- 定义在设计空间上的效用函数u
  - 写成不同度量的加权和形式
  - 用效用（偏好）启发式来确定权重，使之与专家偏好一致
- 筛选设计参数集合，选出对系统性能影响较大的一组参数
- 用以往设计点的评价来构建一个效用函数的代理模型
- 用代理模型优化方法在设计空间中搜索效用最大的点
  - 根据代理模型，在设计空间中搜索能提供最大期望改进的点
  - 用仿真来评价所选点，用贝叶斯规则来更新模型
  - 重复这一过程，直至找到性能最好的点

基于代理模型的优化过程（续）



安全分析

- 美国雷达数据集
  - 规模：393077次相遇数据
  - 采集时间：2007.12 – 2008.12
  - 采集设备：130个雷达
    - 更新率为4.5s的短距雷达和更新率为12s的长距雷达
- 学习一个空中相遇模型，使用它来估计碰撞风险
- 风险率（risk ratio）：

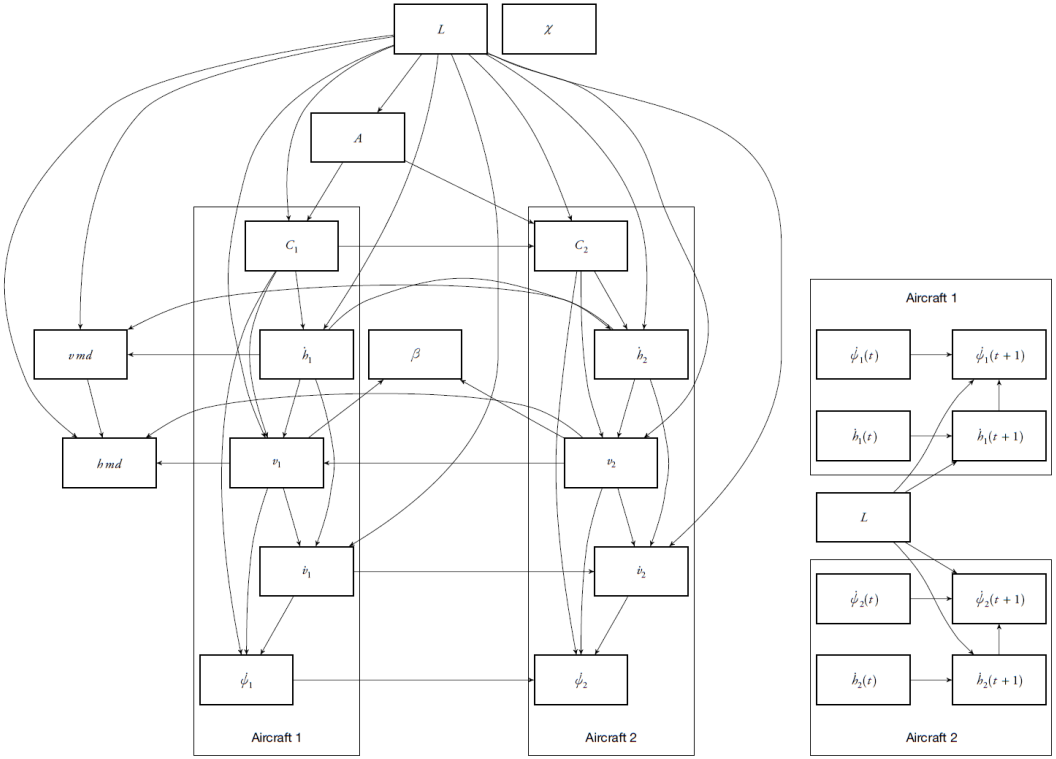
$$\frac{\text{有避碰系统的}NMAC}{\text{无避碰系统的}NMAC}$$

$NMAC$：几乎空中碰撞的概率

- 估计$NMAC$的方法：
  - 直接从模型中采样，计算平均值作为无偏估计
  - 重要性采样：减少方差
    - 交叉熵方法：找到合适的重要性采样分布

空中相遇模型

- 基于美国雷达数据集，使用贝叶斯网络结构学习方法，得到如下的初始和转移贝叶斯网络模型



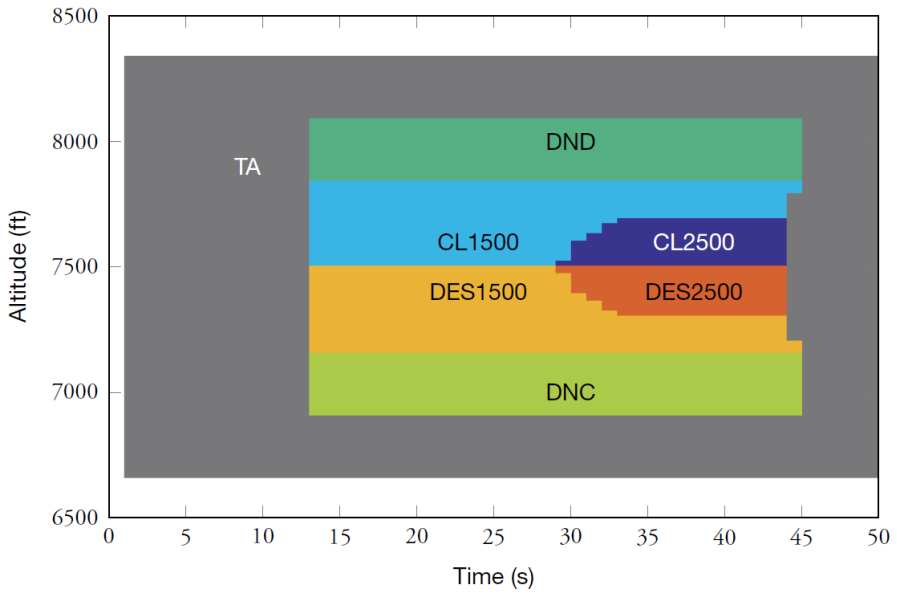
(a) Initial distribution.　　(b) Transition distribution.

- 使用贝叶斯参数学习方法，得到模型参数

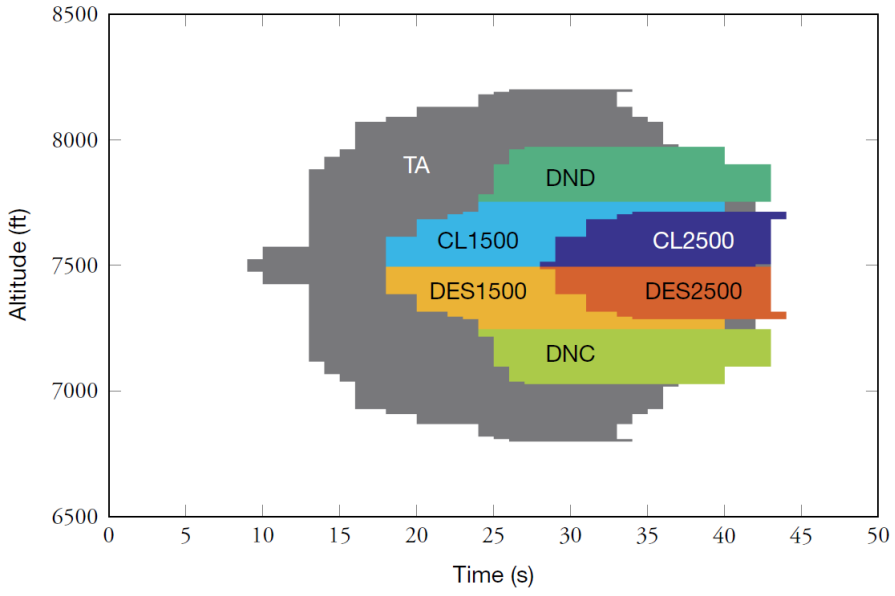
操作适宜性

- ACAS X vs. TCAS
  - 报警率低30%
  - 更少的警报区域

TCAS



ACAS X

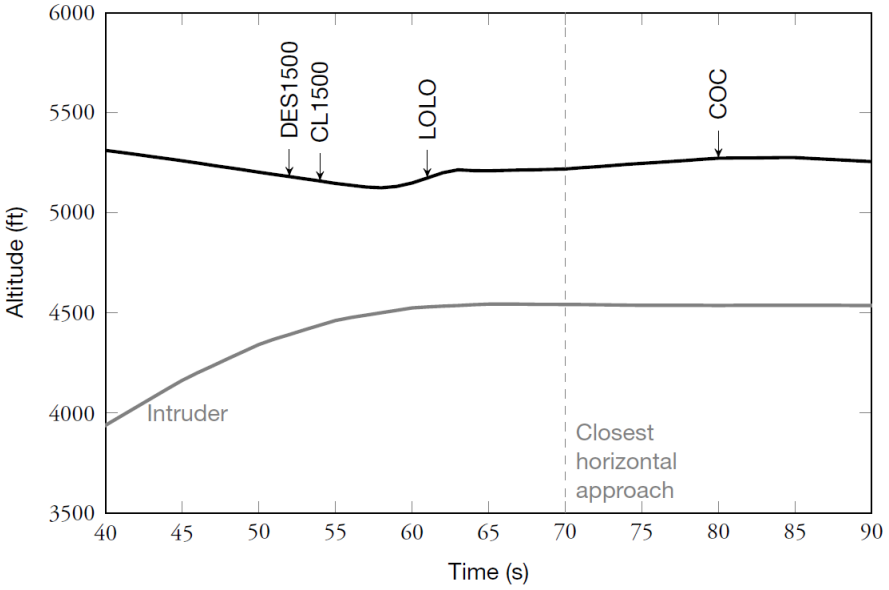


操作可接受性

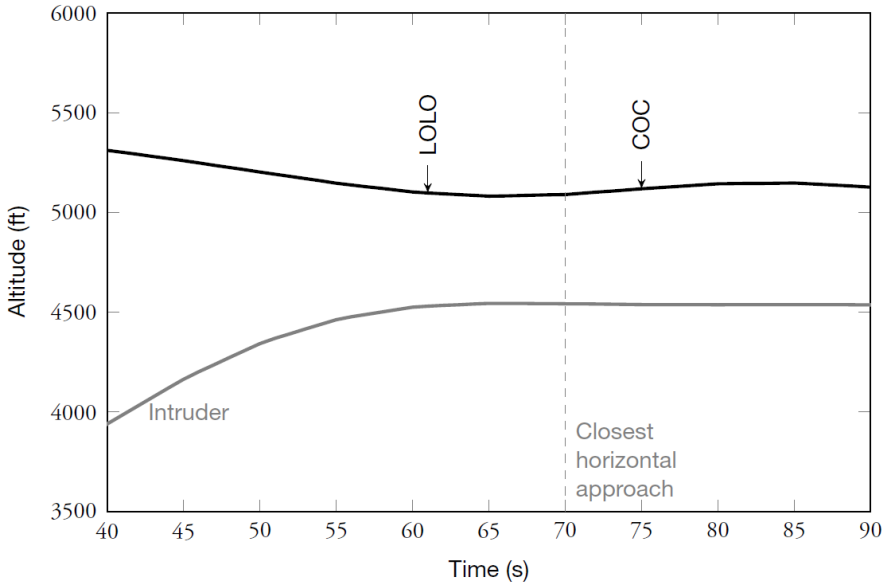
- TCAS
  - 交叉下降报警→逆转爬升报警→水平飞行报警→冲突解除报警
- ACAS X
  - 水平飞行报警→冲突解除报警

没有交叉或逆转警报

TCAS



ACAS X



课后练习6.6

- 离线求解一个POMDP与在线求解一个POMDP有何不同？各有什么优缺点？QMDP、FIB、基于点的值迭代有何不同？各有什么优缺点？

课后练习6.7

- 假设行动空间$\mathcal{A}=\{a^1,a^2\}$，信念状态$b=\begin{bmatrix}0.5\\0.5\end{bmatrix}$，奖赏总为1，观察函数为$O(o^1|\,a^1)=0.8$和$O(o^1|\,a^2)=0.4$，折扣因子$\gamma=0.9$。我们有一个阿尔法向量$\alpha=\begin{bmatrix}-3\\4\end{bmatrix}$，并用它表示近似值函数。请使用深度为1的前向搜索方法计算$U(b)$。

在计算中可能使用到的数据如下：

a	o	UpdateBelief(b,a,o)
a^1	o^1	$[0.3,0.7]^\top$
a^2	o^1	$[0.2,0.8]^\top$
a^1	o^2	$[0.5,0.5]^\top$
a^2	o^2	$[0.8,0.2]^\top$