Planning and Acting in Partially Observable Stochastic Domains

Lesli P. Kaelbling, Michael L.Littman, Anthony R.Cassandra

CS594 Automated decision making

course presentation

Professor: Piotr.

Hui Huang

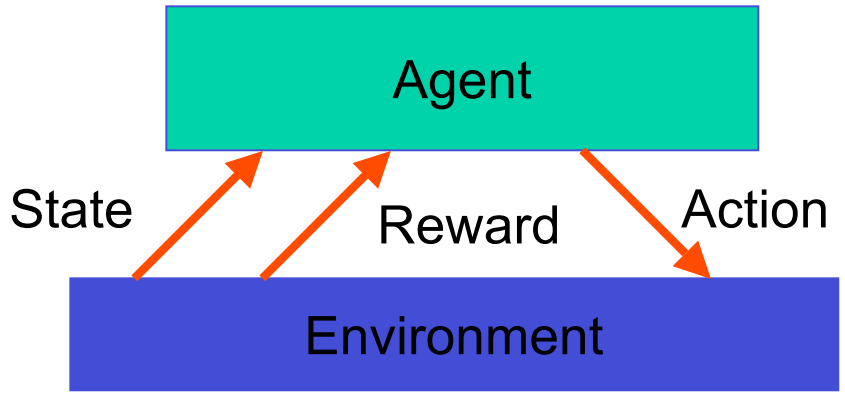
University of Illinois, Chicago

Oct 30, 2002

Outline

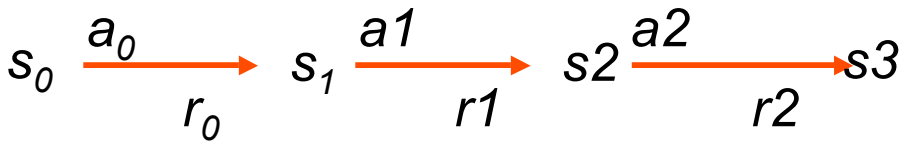
- MDP & POMDP Basic
- POMDP ≡ Belief Space MDP
  - Policy Tree
  - Piecewise Linear Value Function
- Tiger Problem

MDP Model



Process:

- *Observe* state s_t in S
- Choose action at in A_t
- Receive immediate reward r_t
- State changes to s_{t+1}



MDP Model <S, A, T, R>

Policy & Value Function

- Which action, *a*, should the agent take?
  - In MDPs:
    - Policy is a mapping from *state* to action, π: S $\rightarrow$ A

- Value Function $V_{\pi t}(S)$ given a policy π
  - The expected sum of reward gained from starting in state s executing non-stationary policy π for t steps.

- Relation
  - Value function is evaluation for policy based no the long-run value that agent expects to gain from execute the policy.

Optimization (MDPs)

- Recursively calculate expected long-term reward for each *state/belief:*

$$V_t^*(s) = \max_a \left[R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V_{t-1}^*(s') \right]$$

- Find the action that maximizes the expected reward:

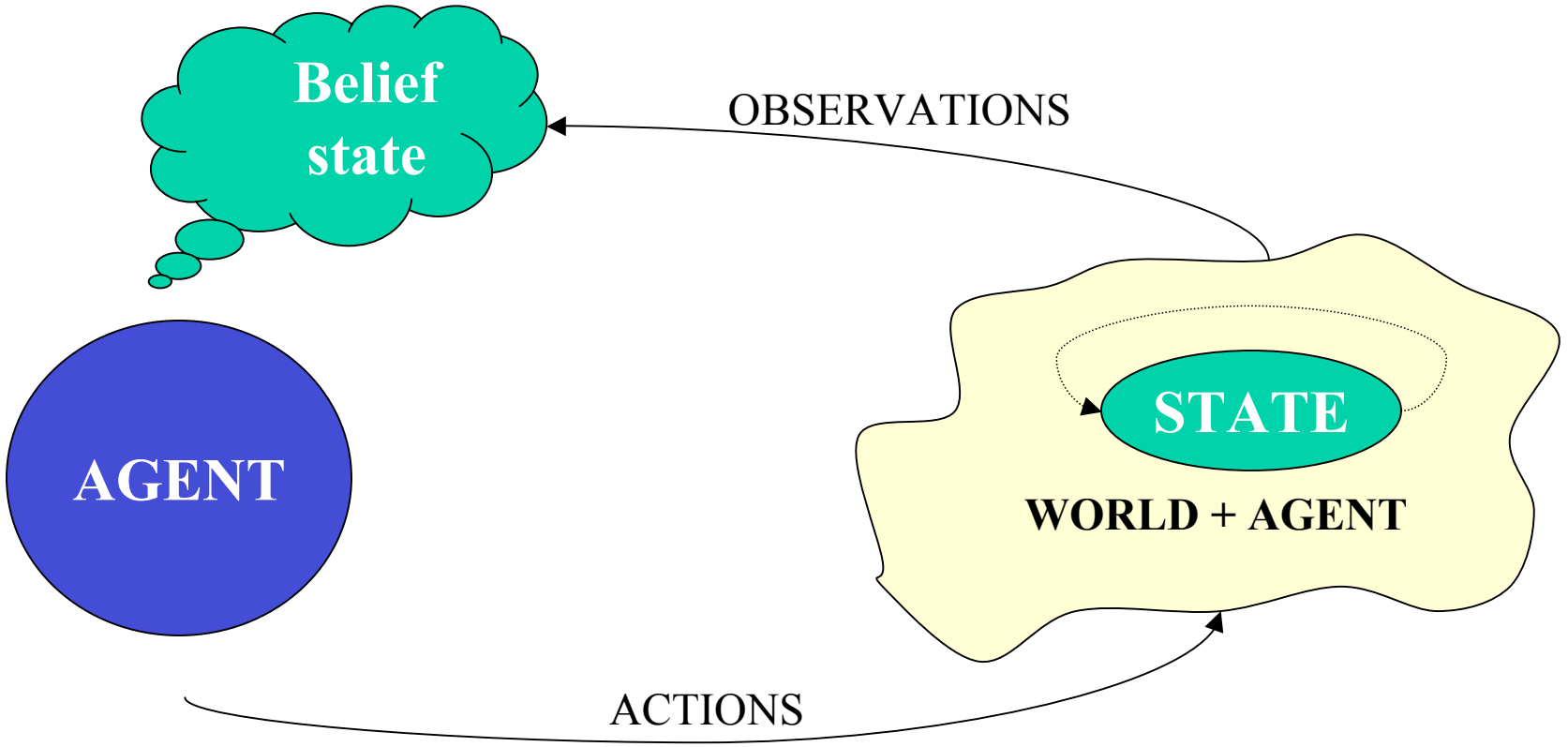
$$\pi_t^*(s) = \arg\max_a \left[R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V_{t-1}^*(s') \right]$$

POMDP: UNCERTAINTY

Case #1: Uncertainty about the action outcome

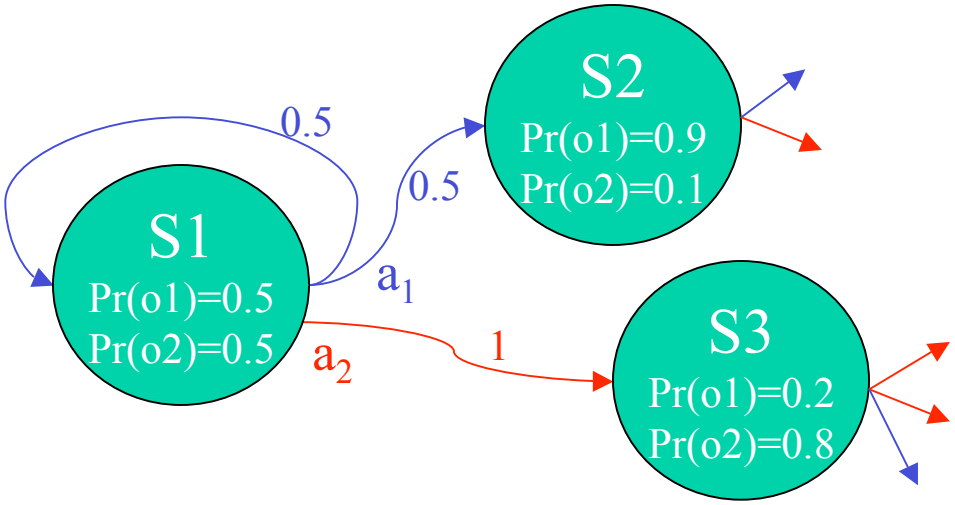
Case #2: Uncertainty about the world state due to imperfect (partial) information

A broad perspective



GOAL = Selecting appropriate actions

What are POMDPs?



Components:

Set of states: s∈S

Set of actions: a∈A

Set of observations: o∈Ω

POMDP parameters:

Initial belief: $b_0(s)=Pr(S=s)$

Belief state updating: $b'(s')=Pr(s'|o, a, b)$

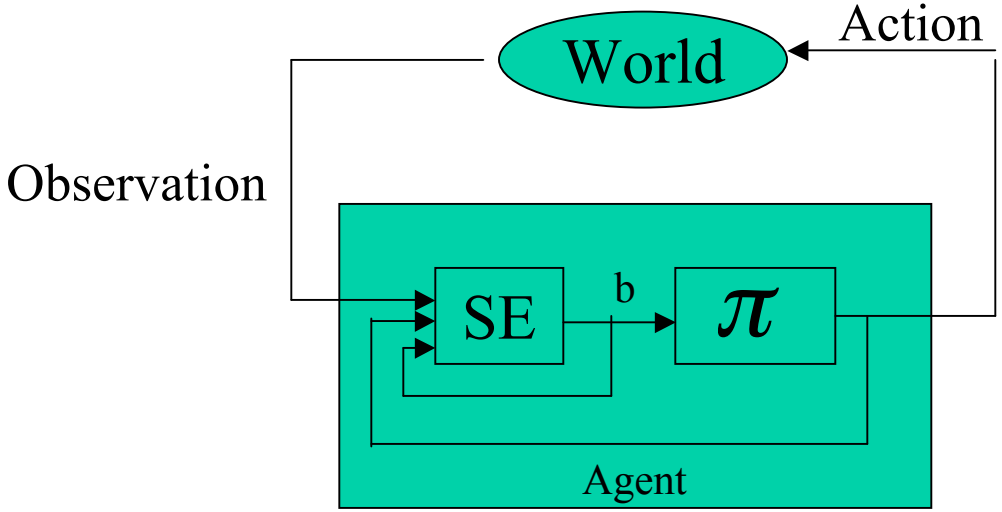
Observation probabilities: $O(s',a,o)=Pr(o|s',a)$

Transition probabilities: $T(s,a,s')=Pr(s'|s,a)$

Rewards: $R(s,a)$

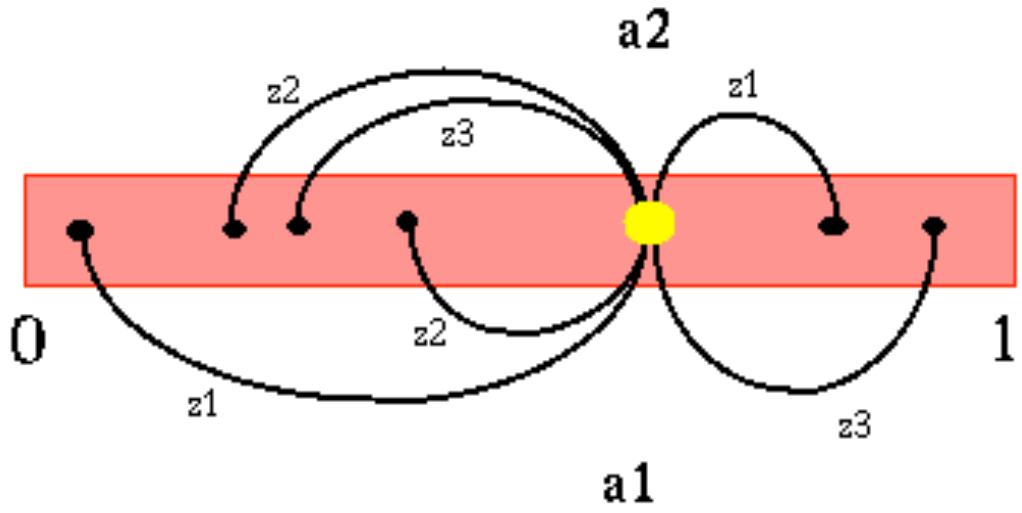
} MDP

Belief state



- Probability distributions over states of the underlying MDP
- The agent keeps an internal ***belief state,*** b, that summarizes its experience. The agent uses a ***state estimator***, SE, for updating the belief state b' based on the last action a_{t-1}, the current observation o_t, and the previous belief state b.
- Belief state is a sufficient statistic (it satisfies the Markov proerty)

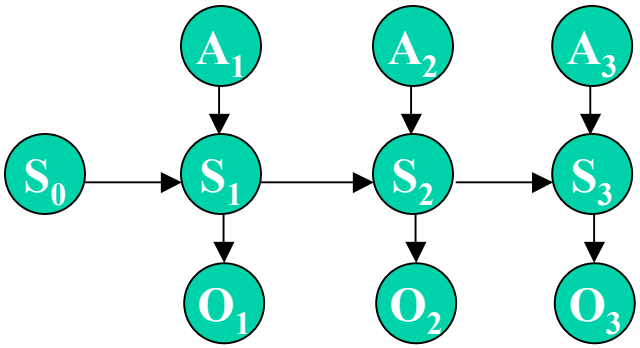
1D belief space for a 2 state POMDP



$$b'(s_j) = \mathrm{P}(s_j \mid o,a,b) = \frac{P(o \mid s_j,a)\sum_{s_i \in S} P(s_j \mid s_i,a)b(s_i)}{\sum_{s_j \in S} P(o \mid s_j,a)\sum_{s_i \in S} P(s_j \mid s_i,a)b(s_i)}$$

POMDP ≡ Continuous-Space Belief MDP

- a POMDP can be seen as a continuous-space "belief MDP", as the agent's belief is encoded through a continuous "belief state".



- We may solve this belief MDP like before using value iteration algorithm to find the optimal policy in a continuous space. However, some adaptations for this algorithm are needed.

Belief MDP

- The policy of a POMDP maps the current belief state into an action. As the belief state holds all relevant information about the past, the optimal policy of the POMDP is the the solution of (continuous-space) belief MDP.
- A belief MDP is a tuple <B, A, ρ, P>:

  B = infinite set of belief states

  A = finite set of actions

  ρ(b, a) = $\sum_{s \in S} b(s) R(s,a)$ (reward function)

  P(b'|b, a) = $\sum_{o \in O} P(b' \mid b, a, o) P(o \mid a, b)$ (transition function)

  Where P(b'|b, a, o) = 1 if SE(b, a, o) = b',

  P(b'|b, a, o) = 0 otherwise;

Thinking(Can we solving this Belief MDP?)

- The Bellman equation for this belief MDP is

$$V^*(b) = \max_{a \in A} \left[\sum_{s \in S} b(s) R(s,a) + \gamma \sum_{o \in O} \Pr(o \mid b, a) V^*(b_o^a) \right]$$

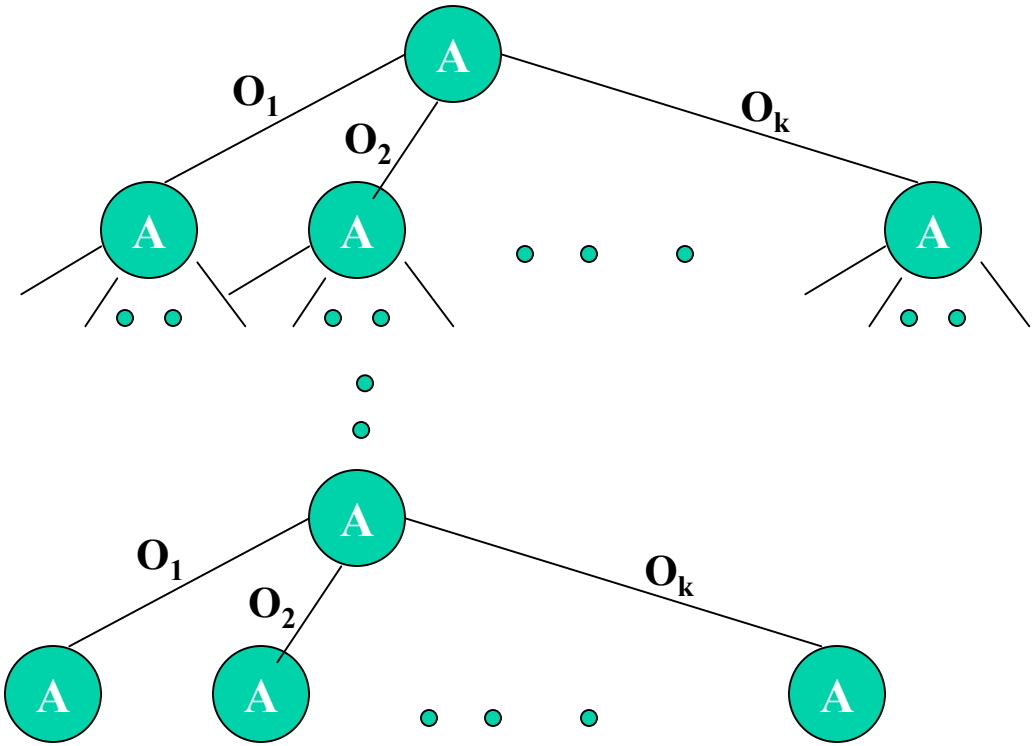
- In general case: Very Hard to solve continuous space MDPs. Unfortunately, DP updates cannot be carried out because there are uncountably many of belief states. One cannot enumerate every equation of value function. To conduct DP steps implicitly, we need to discuss the properties of value functions.

- Some special properties can be exploited first to simplify this problem
  - Policy Tree
  - Piecewise linear and convex property

- And then find an approximation algorithm to construct the optimal t-step discounted value function over belief space using value iteration…

Policy Tree

- With one step remaining, the agent must take a single action. With 2 steps to go, it takes an action, make an observation, and makes the final action. In general, an agent t-step policy can be represented as a policy tree.



T steps to go

T-1 steps to go

2 steps to go

(T=2)

1 step to go
(T=1)

Value Function for policy tree p

- If p is one-step policy tree, then the value of executing that action in state s is

  Vp(s) = R(s, a(p)).

- More generally, if p is a t-step policy tree, then

  Vp(s) = R(s, a(p)) + r (Expected value of the future)

  $$= R(s, a(p)) + r \sum_{s' \in S} T(s' \mid s, a(p), s') \sum_{o_i \in \Omega} T(s', a(p), o_i) V_{o_i(p)}(s')$$

- Thus, Vp(s) can be thought as a *vector* associated with the policy trees p since its dimension is the same as the number of states. We often use notation αp to refer to this vectors.

$$\alpha_p = \langle V_p(s_1), V_p(s_2), \ldots, V_p(s_n) \rangle$$

Value Function Over Belief Space

- As the exact world cannot be observed, the agent must compute an expectation over world states of executing policy tree p from belief state b:

$$\text{Vp(b)} = \sum_{s \in S} b(s) V_p(s)$$

- If we let $\alpha_p = \langle V_p(s_1), V_p(s_2), ..., V_p(s_n) \rangle$, then

$$V_p(b) = b\ \alpha_p$$

- To construct an optimal t-step policy, we must maximize over all t-step policy trees P:

$$V_t(b) = \max_{p \in P} b\ \alpha_p$$

- As Vp(b) is linear in b for each p∈P, Vt(b) is the upper surface of those functions. That is, Vt(b) is piecewise linear and convex.

Illustration: Piecewise Linear Value Function

- Let $V_{p1}V_{p2}$ and V_{p3} be the value functions induced by policy trees p1, p2, and p3. Each of these value functions are the form

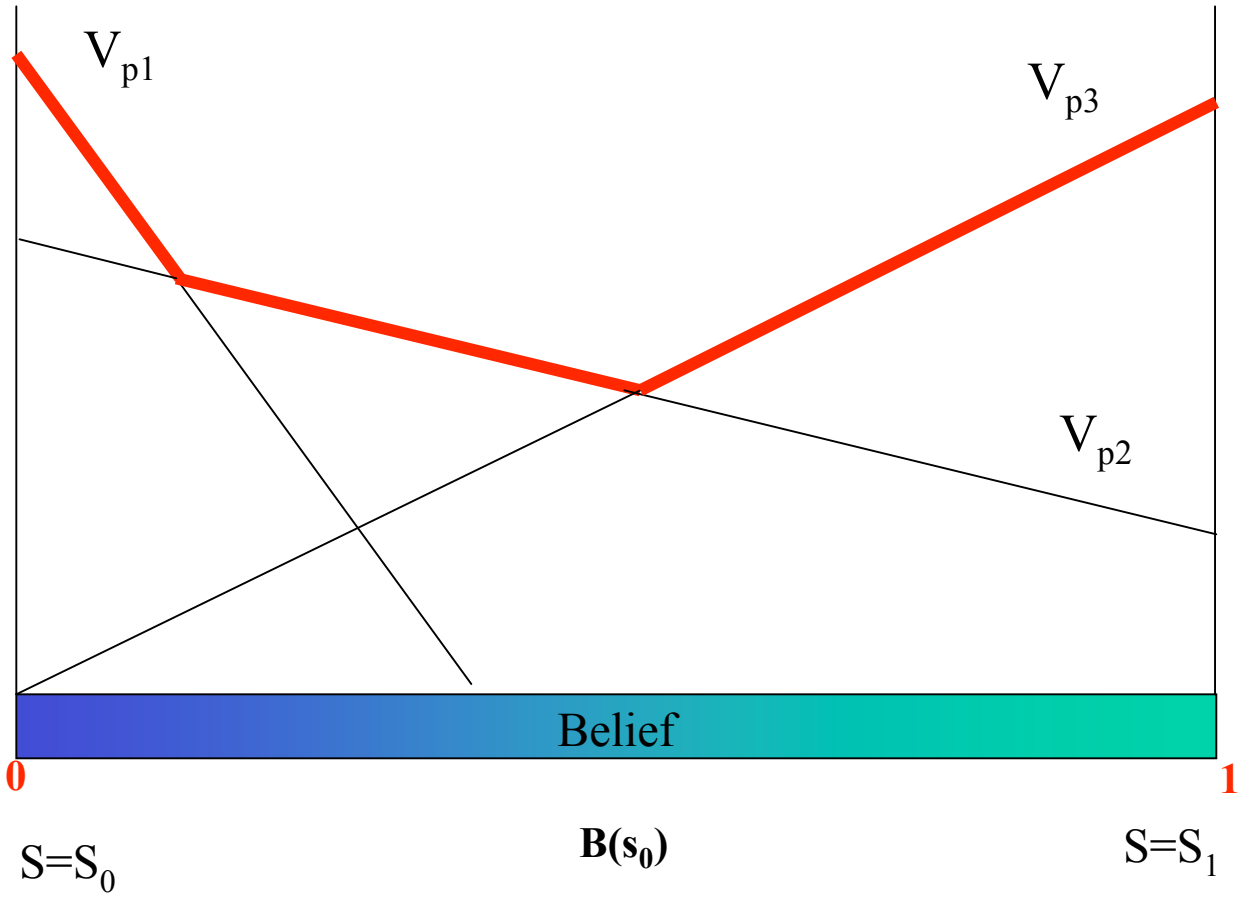
$$V_{p_i}(b) = b\ \alpha_{p_i}$$

- Which is a multi-linear function of b. Thus, for each value function can be shown as a line, plane, or hyperplane, depending on the number of states, and the optimal t-step value

- $$V_t(b) = \max_{p_i \in P} b\ \alpha_{p_i}$$

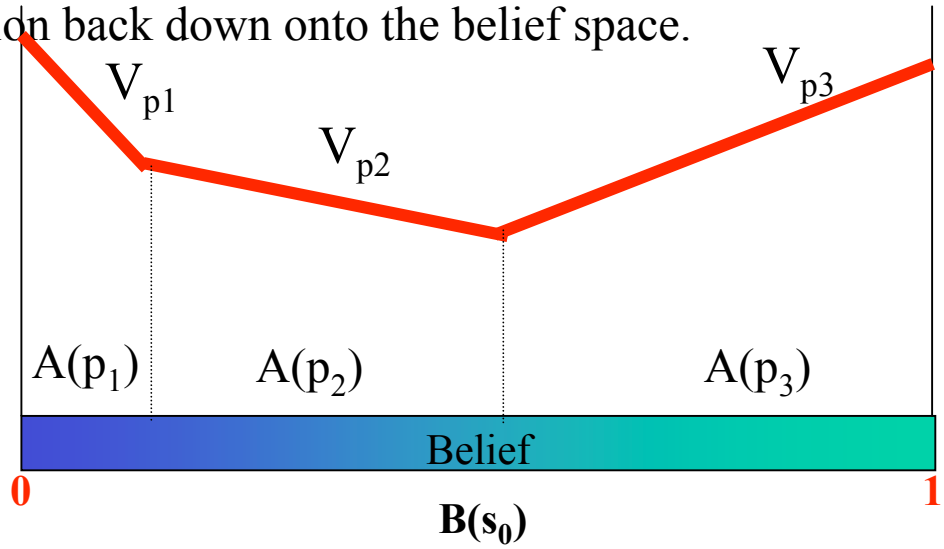
- Example– in the case of two states (b(s1) = 1- b(s2)) – be illustrated as the upper surface in two dimensions:

Picture: Optimal t-step Value Function



Optimal t-Step Policy

- The optimal t-step policy is determined by projecting the optimal value function back down onto the belief space.

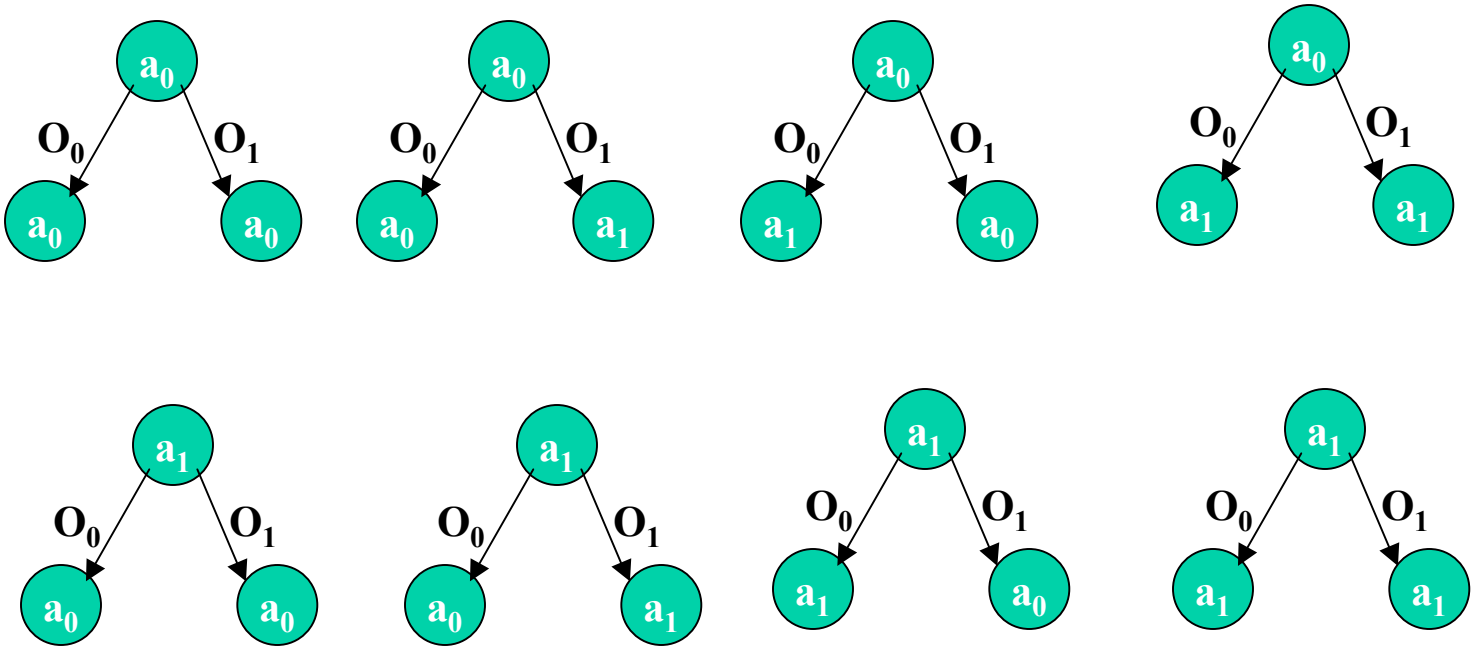


- The projection of the optimal t-step value function yields a partition into regions, within each of which there is a single policy tree, p, such that is maximal over the entire region. The optimal action in that region a(p), the action in the root of the policy tree p.

First Step of Value Iteration

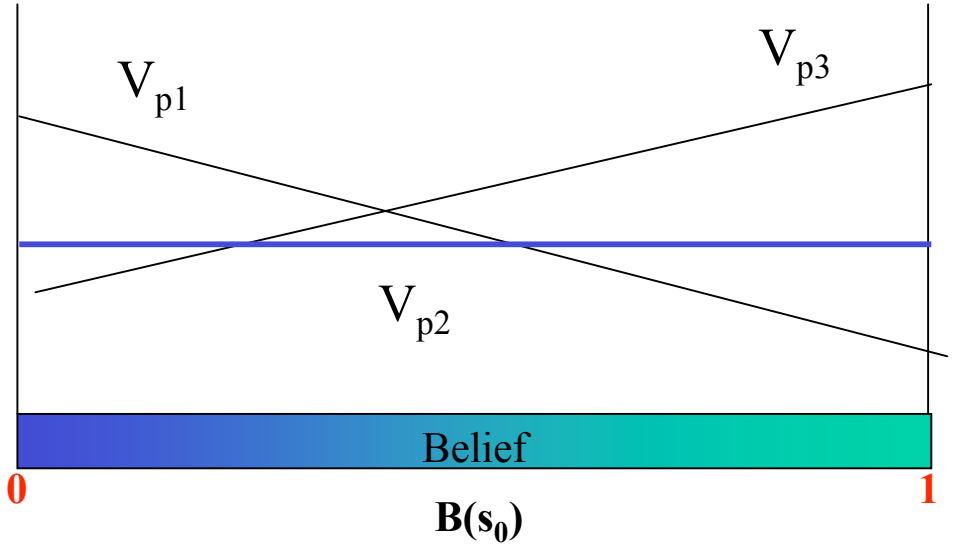
- One-step policy trees are just actions: 

- To do a DP backup, we evaluate every possible 2-step policy tree



Pruning

- Some policy trees are dominated and are never useful



- They are pruned and not considered on the next step.
- The key idea is to prune before evaluating (Witness and Incremental Pruning do this)

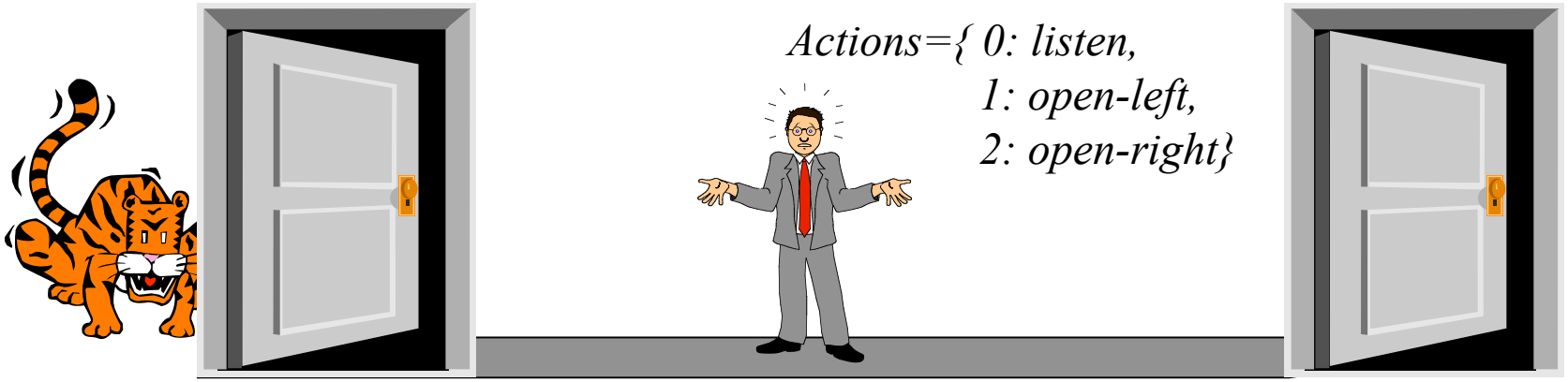
Value Iteration (Belief MDP)

- Keep doing backups until the value function doesn't change much anymore
- In the worst case, you end up considering every possible policy tree
- But hopefully you will converge before this happens

A POMDP example: The tiger problem

S0
"tiger-left"
Pr(o=TL | S0, listen)=0.85
Pr(o=TR | S1, listen)=0.15

S1
"tiger-right"
Pr(o=TL | S0, listen)=0.15
Pr(o=TR | S1, listen)=0.85



Actions={ 0: listen,
1: open-left,
2: open-right}

Reward Function

- *Penalty for wrong opening: -100*
- *Reward for correct opening: +10*
- *Cost for listening action: -1*

Observations

- *to hear the tiger on the left (TL)*
- *to hear the tiger on the right(TR)*

Tiger Problem (Transition Probabilities)

.

Prob. (LISTEN)	Tiger: left	Tiger: right
Tiger: left	1.0	0.0
Tiger: right	0.0	1.0

Prob. (LEFT)	Tiger: left	Tiger: right
Tiger: left	0.5	0.5
Tiger: right	0.5	0.5

Prob. (RIGHT)	Tiger: left	Tiger: right
Tiger: left	0.5	0.5
Tiger: right	0.5	0.5

Tiger Problem (Observation Probabilities)

.

Prob. (LISTEN)	O: TL	O: TR
Tiger: left	0.85	0.15
Tiger: right	0.15	0.85

Prob. (LEFT)	O: TL	O: TR
Tiger: left	0.5	0.5
Tiger: right	0.5	0.5

Prob. (LEFT)	O: TL	O: TR
Tiger: left	0.5	0.5
Tiger: right	0.5	0.5

Tiger Problem (Immediate Rewards)

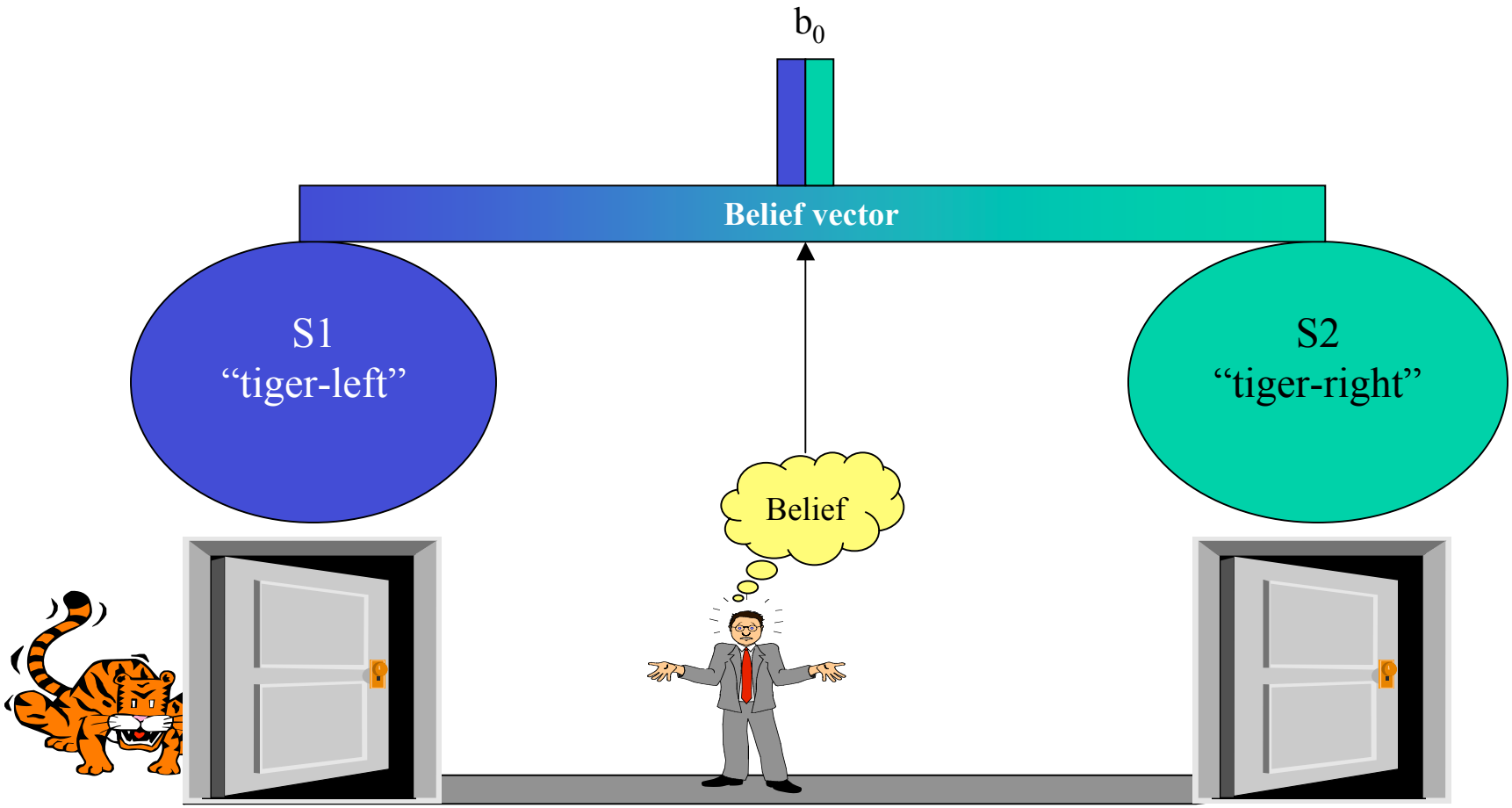
.

Reward (LISTEN)	
Tiger: left	-1
Tiger: right	-1

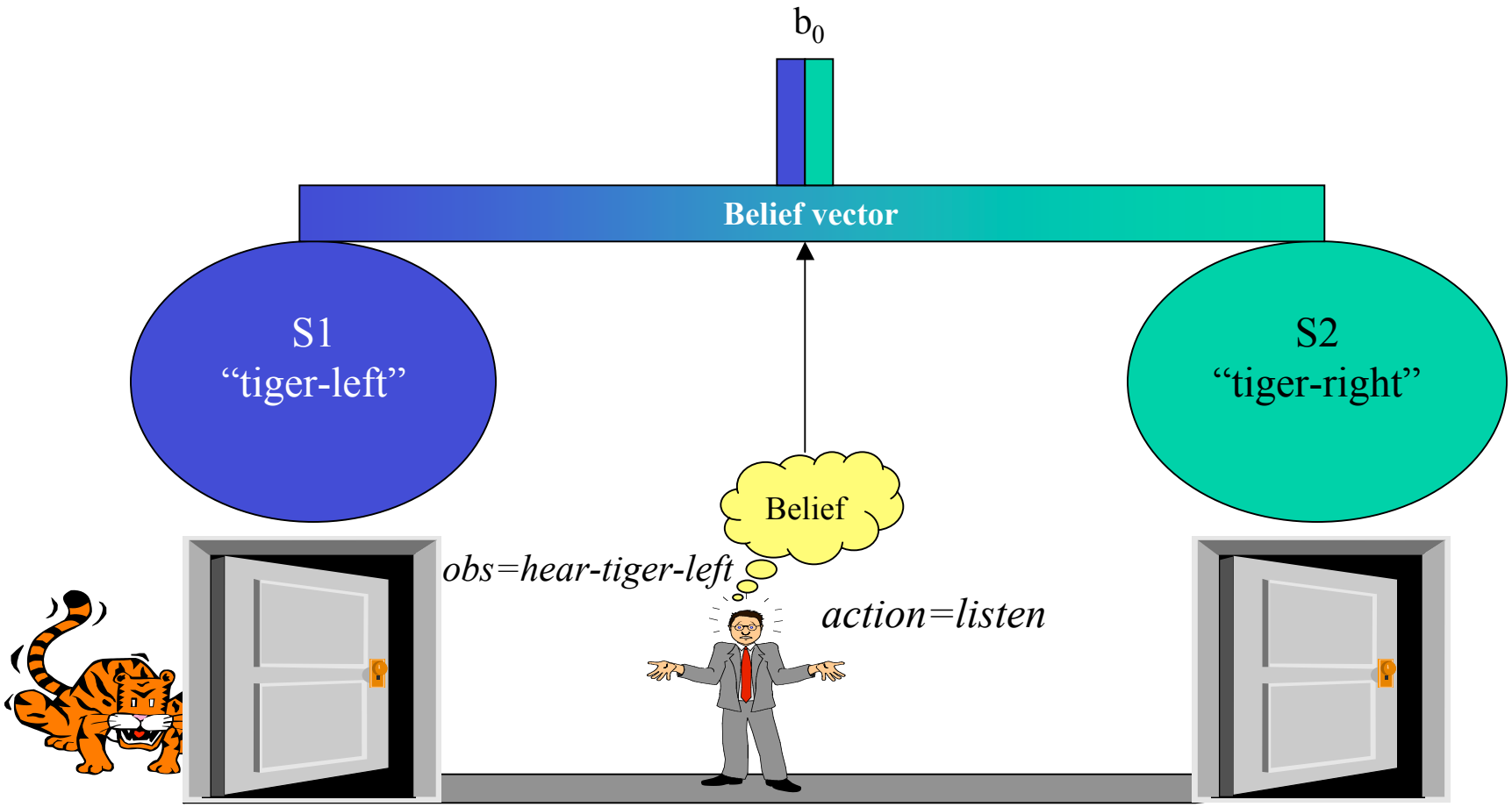
Reward (LEFT)	
Tiger: left	-100
Tiger: right	+10

Reward (RIGHT)	
Tiger: left	+10
Tiger: right	-100

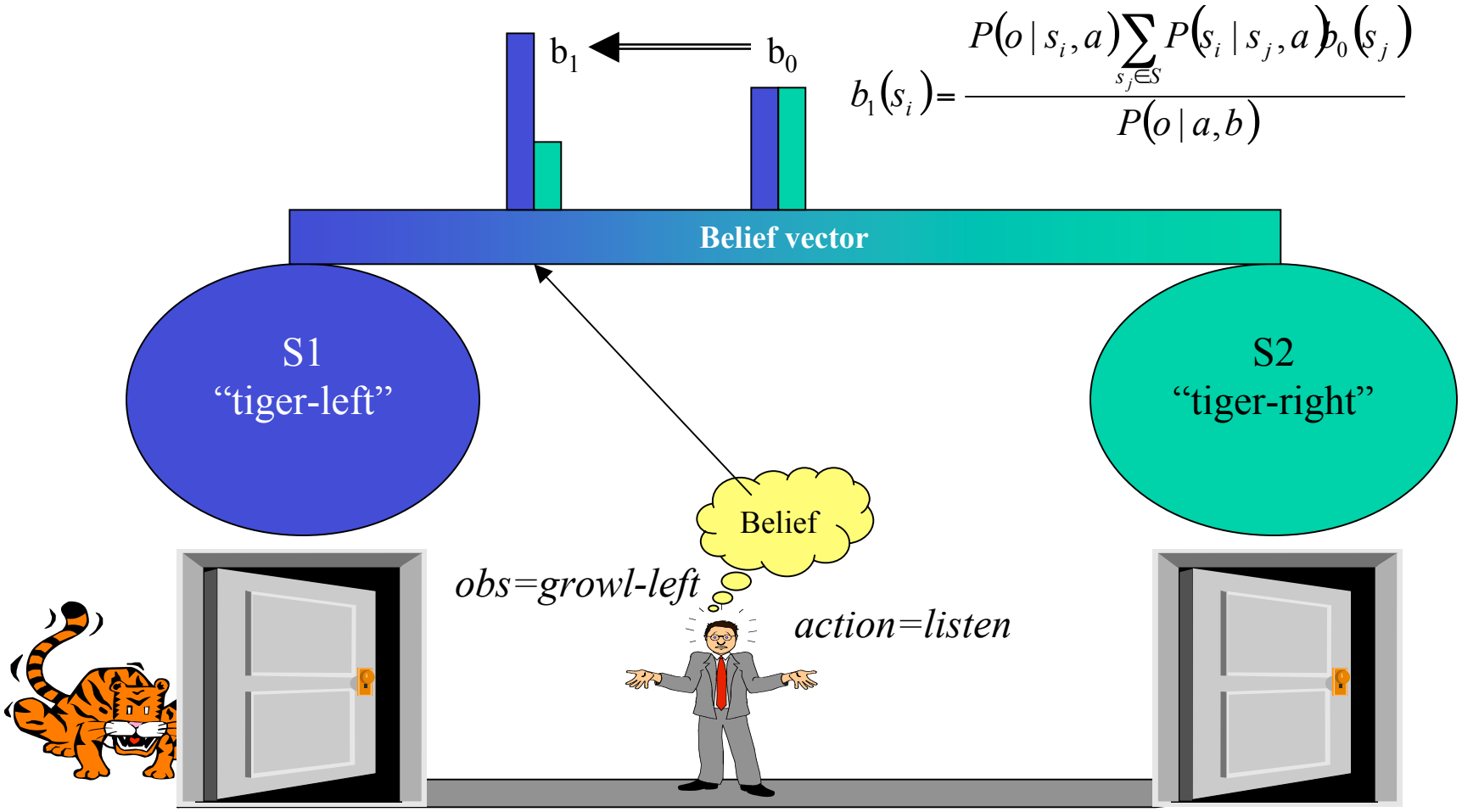
The tiger problem: State tracking



The tiger problem: State tracking

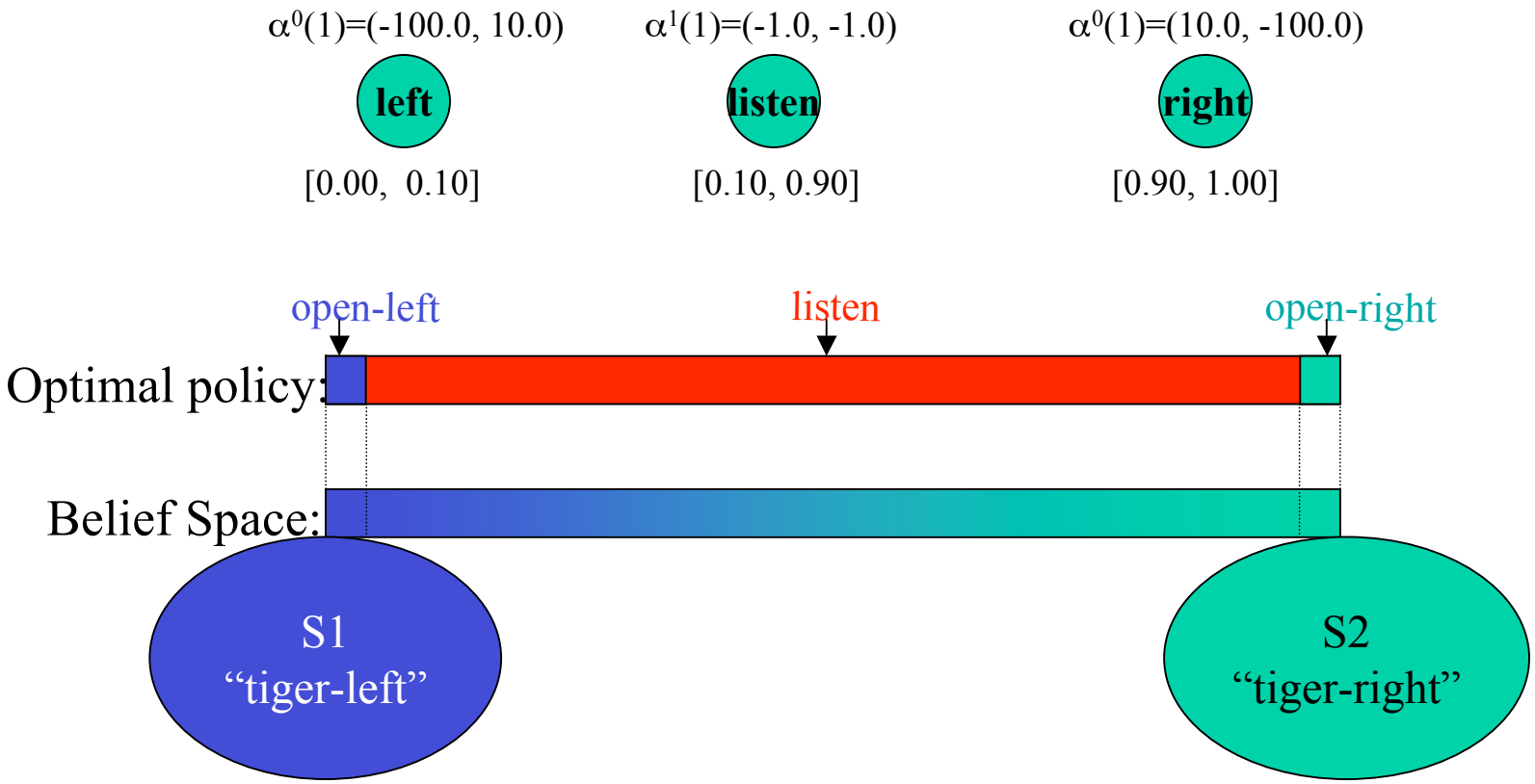


The tiger problem: State tracking



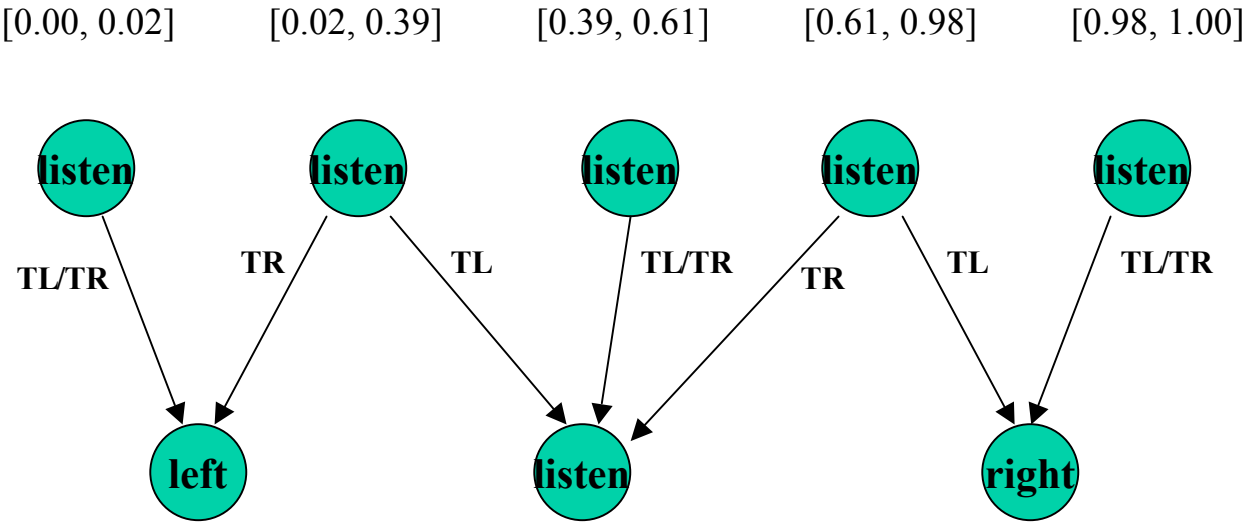
Tiger Example Optimal Policy t=1

- Optimal Policy for t=1



Tiger Example Optimal Policy for t=2

- For t=2



Reference

- ***Planning and acting in partially Observable stochastic domains*** Leslie P. Kaelbling
- ***Optimal Policies for partially Observable Markov Decision Processes*** Anthony R.Cassandra 1995
- ***Planning and control in stochastic domains with imperfect information*** Milos Hauskrecht
- ***Hierarchical Methods for Planning under Uncertainty*** Joelle Pineau
- ***POMDP's tutorial in Tony's POMDP Page*** http://www.cs.brown.edu/research/ai/pomdp/
- ***Partial Observability*** Dan Bernstein 2002

Thank You. ☺