Planning and Acting in Partially Observable Stochastic Domains

Leslie Pack Kaelbling and Michael L. Littman and Anthony R. Cassandra (1998). Planning and Acting in Partially Observable Stochastic Domains, *Artificial Intelligence,* **101***:*99-134.

Introduction

Consider the problem of a robot navigating in a large office building. The robot can move from hallway intersection to intersection and can make local observations of its world. Its actions are not completely reliable, however. Sometimes, when it intends to move, it stays where it is or goes too far; sometimes, when it intends to turn, it overshoots. It has similar problems with observation. Sometimes a corridor looks like a corner; sometimes a T-junction looks like an L-junction. How can such an error-plagued robot navigate, even given a map of the corridors?

Introduction 2

In general, the robot will have to remember something about its history of actions and observations and use this information, together with its knowledge of the underlying dynamics of the world (the map and other information) to maintain an estimate of its location. Many engineering applications follow this approach, using methods like the Kalman filter … expressed as an ellipsoid or normal distribution in Cartesian space. This approach will not do for our robot, though. Its uncertainty may be **discrete**: it might be almost certain that it is in the north-east corner of either the fourth or the seventh floors, though it admits a chance that it is on the fifth floor, as well.

Introduction 3

Then, given an uncertain estimate of its location, the robot has to decide what actions to take. In some cases, it might be sufficient to ignore its uncertainty and take actions that would be appropriate for the most likely location. In other cases, it might be better for the robot to take actions for the purpose of gathering information, such as searching for a landmark or reading signs on the wall. In general, it will take actions that fulfill **both** purposes **simultaneously**.

Introduction 4

Partially Observable Markov Decision Process (POMDP)

To solve problems of choosing optimal actions in partially observable stochastic domains

- Essentially a planning problem: given a complete and correct model of the world dynamics and a reward structure, find an optimal way to behave.
- But: stochastic domains → depart from traditional AI planning model
  - Rather than taking plans to be sequences of actions (which may only rarely execute as expected): *mappings from situations to actions that specify the agent's behavior no matter what may happen*

Introduction 5

Partially Observable Markov Decision Process (POMDP)

No distinction drawn between actions taken to change the state of the world and actions taken to gain information

- Thus, optimal performance involves something akin to a “value of information” calculation
  - *Amount of information an action provides*
  - *Amount of reward an action produces*
  - *How an action changes the state of the world*

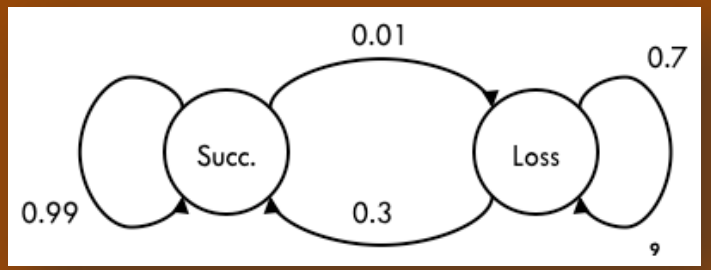
Background

Many real world applications are both non-deterministic and partially observable

- Autonomous robot control
  - Imperfect actions, observations
- Decision Support Systems
  - Incomplete knowledge of user's cognitive states
  - Local view of external environment
- Industrial Control Systems
  - Noisy sensors
  - Undiagnosed faults

Background: Markov Chains

- Markov Chains to model the environment
  - Handle non-determinism with probabilities
- 2 Tuple
  - $S = \{s\}$ set of states
  - $P(s' \mid s)$ state transitions
- Application: Network Loss
  - (Nguyen et al., 1996)



Background: Hidden Markov Model

- Hidden Markov Model (HMM) to model the environment
  - Incomplete knowledge of states
- 4 Tuple
  - $S = \{s\}$ set of states
  - $P(s' \mid s)$ state transitions
  - $\Omega = \{o\}$ set of observations
  - $O(s, o) = P(o \mid s)$ observation function
- Application: Bioinformatics
  - DNA sequences
  - (Durbin et al., 1998)

ATCGTCATTGCCATATACC
GGCTATCATTTCCCAGGA
TCTTACCTTTAAGTCTATAC

Background: Recap 1

	Fully Observable	**Partially Observable**
States	Markov Chain	Hidden Markov Model
Actions		

Markov Decision Processes (MDPs)

A Markov decision process can be described as a tuple $\langle \mathcal{S}, \mathcal{A}, T, R \rangle$, where

- $\mathcal{S}$ is a finite set of states of the world;
- $\mathcal{A}$ is a finite set of actions;
- $T : \mathcal{S} \times \mathcal{A} \to \Pi(\mathcal{S})$ is the *state-transition function*, giving for each world state and agent action, a probability distribution over world states (we write $T(s, a, s')$ for the probability of ending in state s', given that the agent starts in state s and takes action a); and
- $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, giving the expected immediate reward gained by the agent for taking each action in each state (we write $R(s, a)$ for the expected reward for taking action a in state s).

MDPs 2

Although there may be a great deal of uncertainty about the effects of an agent's actions, there is never any uncertainty about the agent's current state

- it has complete and perfect perceptual abilities **(MDP only!)**

The next state and the expected reward depend only on the previous state and the action taken

- even if we were to condition on additional previous states, the transition probabilities and the expected rewards would remain the *same*
- **The Markov property**—the state and reward at time $t + 1$ is dependent only on the state at time t and the action at time t.

MDPs: Acting Optimally

Maximizing some measure of the long-run reward received

- E.g., **finite-horizon optimality**
  - Maximizing expected sum of immediate rewards of *k* steps
  - But, what is *k*?
- E.g., **infinite-horizon discounted** model
  - Summing the rewards over the infinite lifetime of the agent, but discount them geometrically using a discount factor
  - *Why discounted?*

$$E\left[\sum_{t=0}^{k-1} r_t\right]$$

$$E\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$$

MDPs: Acting Optimally 2

Policy = Description of the behavior of an agent

Two kinds of policies: **stationary** and **non-stationary**

- A stationary policy is a situation-action mapping that specifies for each state, an action to be taken.
  - **The choice of action depends only on the state and is independent of the time step.**

$$\pi : \mathcal{S} \rightarrow \mathcal{A},$$

- A non-stationary policy is a sequence of situation-action mappings, indexed by time

indexed by time. The policy π_t is to be used to choose the action on the t^{th}-to-last step as a function of the current state, s_t. In the finite-horizon model,

MDPs: Acting Optimally 3

Horizon's impact on optimal policy

- **Finite**: typically non-stationary as the way an agent chooses its actions on the last step of its life is generally going to be very different from the way it chooses them when it has a long life ahead of it
- **Infinite**: typically stationary as an agent has a constant expected amount of time remaining → no reason to change action strategies

MDPs: Acting Optimally 4

Finite-Horizon

Evaluate a policy based on the long-run value that the agent expects to gain from executing it

$$V_{\pi,t}(s) = R(s, \pi_t(s)) + \gamma \sum_{s' \in \mathcal{S}} T(s, \pi_t(s), s') V_{\pi,t-1}(s')$$

- Immediate reward
- Discount factor
- Transition probability from *s* to *s'*
- Value of *s' (but now only with remaining t-1 steps)*

Recursively defined

MDPs: Acting Optimally 4

Infinite-Horizon Discounted

Evaluate a policy based on the expected discounted sum of future reward agent expects to gain from executing it

$$V_\pi(s) = R(s, \pi(s)) + \gamma \sum_{s' \in S} T(s, \pi(s), s') V_\pi(s')$$

- Immediate reward
- Discount factor
- Transition probability from *s* to *s'*
- Value of *s'*

Recursively defined as well

MDPs: Acting Optimally 5

Greedy Policy

For the finite horizon:

$$\pi_V(s) = \underset{a}{\operatorname{argmax}} \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s,a,s') V(s') \right]$$

Policy obtained by, at every step, taking the action that maximizes expected immediate reward plus the expected discounted value of the next state, as measured by *V*

MDPs: Acting Optimally 6

Optimal Policy

What is the optimal finite-horizon policy, π^*? The agent's last step is easy: it should maximize its final reward. So

$$\pi_1^*(s) = \operatorname*{argmax}_a R(s,a) \quad .$$

The optimal policy for the t^{th} step, π_t^*, can be defined in terms of the optimal $(t-1)$-step value function $V_{\pi_{t-1}^*,t-1}$ (written for simplicity as V_{t-1}^*):

$$\pi_t^*(s) = \operatorname*{argmax}_a \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s,a,s') V_{t-1}^*(s') \right] \quad ;$$

V_{t-1}^* is derived from π_{t-1}^* and V_{t-2}^*.

Note: Optimal policy = a sequence of actions that at each step maximizes the immediate rewards plus the discounted expected gain in value for the next time interval

MDPs: Acting Optimally 7

Optimal Policy's Value

$$V^*(s) = \max_a \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s,a,s') V^*(s') \right]$$

Note: An optimal policy π^* is just a greedy policy with respect to V^*

MDPs: Computing an Optimal Policy

$V_1(s) := 0$ for all s
$t := 1$
loop
  $t := t + 1$
  loop for all $s \in \mathcal{S}$ and for all $a \in \mathcal{A}$
    $Q_t^a(s) := R(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s,a,s') V_{t-1}(s')$
    $V_t(s) := \max_a Q_t^a(s)$
  end loop
until $|V_t(s) - V_{t-1}(s)| < \epsilon$ for all $s \in \mathcal{S}$

Table 1
The value iteration algorithm for finite state space MDPs.

MDPs: Summary

- 4 Tuple
  - $S = \{s\}$ set of environment states
  - $A = \{a\}$ set of possible actions
  - $T(s, a, s') = P(s' \mid s, a)$ next state function
  - $R(s, a)$ reward function
- Autonomous robot example
  - S set of locations
  - A movements (N, S, E, W, X)
  - T movement to new location
  - R inverse distance to goal

MDP: Recap 2

	Fully Observable	**Partially Observable**
States	Markov Chain	Hidden Markov Model
Actions	Markov Decision Process	

Partial Observability

For MDPs we can compute the optimal policy π and use it to act by simply executing $\pi(s)$ for current state s. **What happens if the agent is no longer able to determine the state it is currently in with complete reliability?**

A naïve approach would be for the agent to map the most recent observation directly into an action without remembering anything from the past

- Performing the same action in every location that looks the same—hardly a promising approach

Better? Adding randomness to the agent's behavior: a policy can be a mapping from **observations to probability distributions over actions**

- **Randomness allows the agent to sometimes choose different actions in different locations with the same appearance**

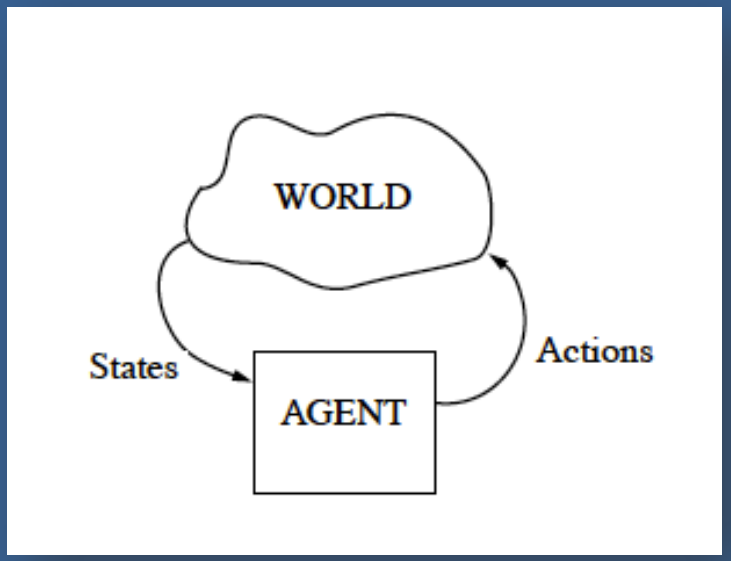
POMDP

A partially observable Markov decision process can be described as a tuple $\langle \mathcal{S}, \mathcal{A}, T, R, \Omega, O \rangle$, where

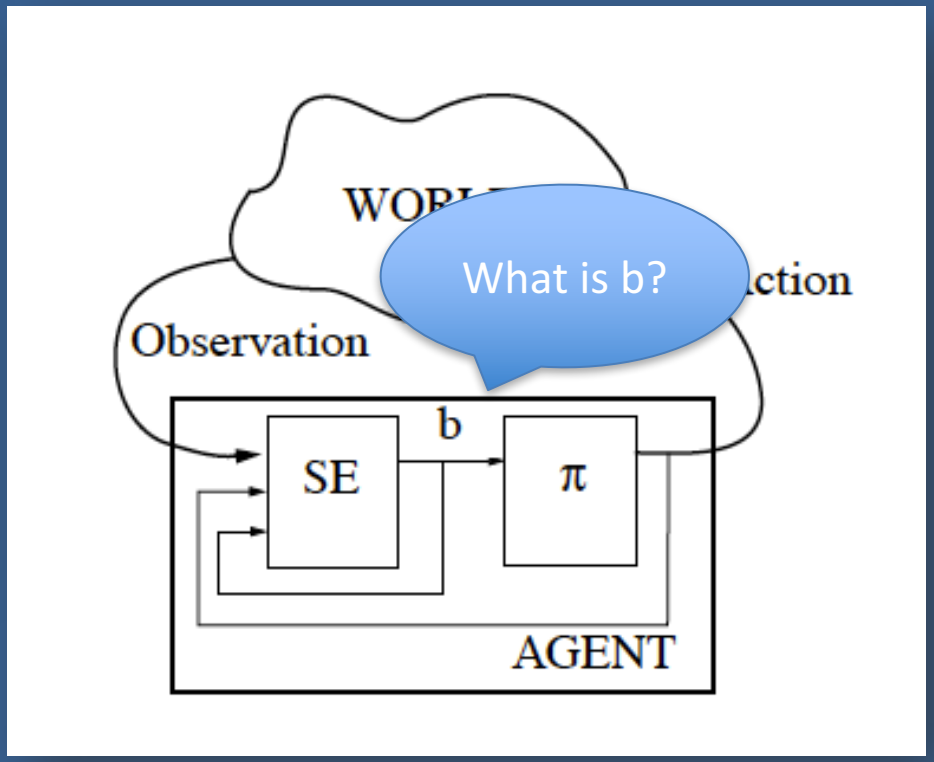
- $\mathcal{S}$, $\mathcal{A}$, T, and R describe a Markov decision process;
- Ω is a finite set of observations the agent can experience of its world; and
- $O : \mathcal{S} \times \mathcal{A} \to \Pi(\Omega)$ is the *observation function*, which gives, for each action and resulting state, a probability distribution over possible observations (we write $O(s', a, o)$ for the probability of making observation o given that the agent took action a and landed in state s').

Note: uncertain about observation → observation function to capture the probability distribution that if an agent observes *o*, what is the probability that it is because of performing *a* to state *s*?

POMDP vs. MDP



MDP



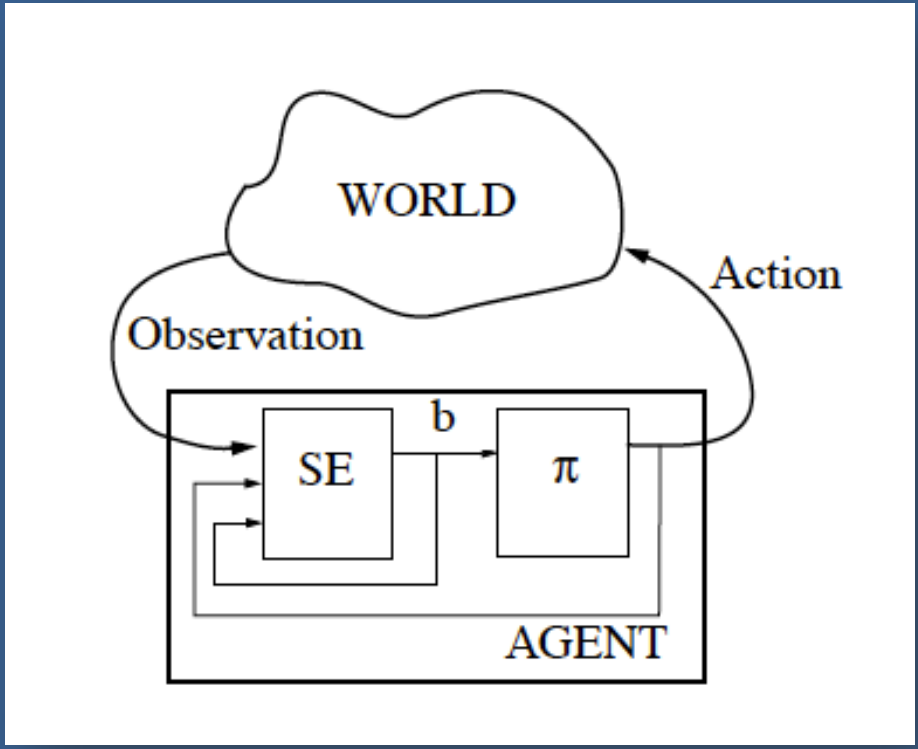
POMDP = state estimator (SE) + policy (π)

A POMDP is an MDP in which the agent is **unable** to observe the current state. Instead, it makes an observation based on the action and resulting state using an estimator. The agent's goal remains to maximize expected discounted future reward.

POMDP: Problem Structure

The agent makes observations and generates actions

- An internal **belief state**, *b*, that summarizes its previous experience (is this still Markov?)
- **SE**: responsible for updating the belief state based on the last action, the current observation, and the previous belief state



- ***π***: responsible for generating actions, but as a function of *the agent's belief state instead of the state of the world*

POMDP: Belief States

One choice might be the most probable state of the world, given the past experience.

- Plausible … is it sufficient in general?
- In order to act **effectively**, an agent must take into account its own degree of uncertainty and reason with that uncertainty
- E.g., If lost or confused, might be appropriate for an agent to take sensing actions

Probability distributions over state of the world

- Encode an agent's subjective probability about the state of the world and provide a basis for acting under uncertainty
- Sufficient statistic for the past history and initial belief state of the agent: given the agent's current belief state, no additional data about its past actions or observations would supply any further information about the current state of the world
  - **The process over belief states is Markov!**

POMDP: Computing Belief States

A belief state b is a probability distribution over $\mathcal{S}$. We let $b(s)$ denote the probability assigned to world state s by belief state b. The axioms of probability require that $0 \leq b(s) \leq 1$ for all $s \in \mathcal{S}$ and that $\sum_{s \in \mathcal{S}} b(s) = 1$. The state estimator must compute a new belief state, b', given an old belief state b, an action a, and an observation o. The new degree of belief in some state s', $b'(s')$, can be obtained from basic probability theory as follows:

$$
\begin{aligned}
b'(s') &= \Pr(s'|o, a, b) \\
&= \frac{\Pr(o|s', a, b) \Pr(s'|a, b)}{\Pr(o|a, b)} \\
&= \frac{\Pr(o|s', a) \sum_{s \in \mathcal{S}} \Pr(s'|a, b, s) \Pr(s|a, b)}{\Pr(o|a, b)} \\
&= \frac{O(s', a, o) \sum_{s \in \mathcal{S}} T(s, a, s') b(s)}{\Pr(o|a, b)}
\end{aligned}
$$

Bayes Theorem

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$$

The denominator, $\Pr(o|a, b)$, can be treated as a normalizing factor, independent of s', that causes b' to sum to 1. The state estimation function $SE(b, a, o)$ has as its output the new belief state b'.

POMDP: Example

Fig. 3. Simple POMDP to illustrate belief state evolution.

Four states (*S*)

- One goal state (star)

Two possible observations (*O*)

- One is always made when the agent is in state 1, 2, or 4
- One is made when the agent is in the goal state (3)

Two possible actions (*A*)

- EAST, WEST

Transitions (*T*)

- If no movement is possible in a particular direction, then the agent stays put
- The above actions succeed with p = 0.9
- When actions fail (i.e., with p = 0.1), the movement is in the opposite direction

POMDP: Example, Cont'd

Fig. 3. Simple POMDP to illustrate belief state evolution.

$$b'(s') = \frac{O(s', a, o) \sum_{s \in S} T(s, a, s') b(s)}{\Pr(o|a, b)}$$

- **Its initial belief state is [0.333 0.333 0 0.333]**
- If the agent takes action EAST and does **not** observe the goal, then the new belief state becomes?
- Note that *O*(*s'*, *a*, *o*) = 1 if non-goal state, and 0 if goal state.

	S1	S2	S3	S4
T = 0	0.333	0.333	0	0.333
T = 1	?	?	?	?
T = 2				

POMDP: Example, Cont'd

Fig. 3. Simple POMDP to illustrate belief state evolution.

$$b'(s') = \frac{O(s', a, o) \sum_{s \in S} T(s, a, s') b(s)}{\Pr(o|a, b)}$$

b'(s1) = O(s1, EAST, o)[T(s1,EAST,s1)b(s1) + T(s2,EAST,s1)b(s2) + T(s3,EAST,s1)b(s3) + T(s4,EAST,s1)b(s4)]*

= 1*[0.1*0.333 + 0.1*0.333 + 0*0 + 0*0.333]

= 0.0667 (un-normalized)

	S1	S2	S3	S4
T = 0	0.333	0.333	0	0.333
T = 1	0.0667 (un-normalized)	?	?	?
T = 2				

POMDP: Example, Cont'd

Fig. 3. Simple POMDP to illustrate belief state evolution.

$$b'(s') = \frac{O(s', a, o) \sum_{s \in S} T(s, a, s') b(s)}{\Pr(o|a, b)}$$

b'(s2) = O(s2, EAST, o)[T(s1,EAST,s2)b(s1) + T(s2,EAST,s2)b(s2) + T(s3,EAST,s2)b(s3) + T(s4,EAST,s2)b(s4)]*

= 1*[0.9*0.333 + 0.0*0.333 + 0.1*0 + 0*0.333]

= 0.2997 (un-normalized)

	S1	S2	S3	S4
T = 0	0.333	0.333	0	0.333
T = 1	0.0667 (un-normalized)	0.2997 (un-normalized)	?	?
T = 2				

POMDP: Example, Cont'd

Fig. 3. Simple POMDP to illustrate belief state evolution.

$$b'(s') = \frac{O(s', a, o) \sum_{s \in S} T(s, a, s') b(s)}{\Pr(o|a, b)}$$

b'(s3) = O(s3, EAST, o)[T(s1,EAST,s3)b(s1) + T(s2,EAST,s3)b(s2) + T(s3,EAST,s3)b(s3) + T(s4,EAST,s3)b(s4)]*

= 0*[0.0*0.333 + 0.1*0.333 + 0.1*0 + 0.9*0.333]

= 0

	S1	S2	S3	S4
T = 0	0.333	0.333	0	0.333
T = 1	0.0667 (un-normalized)	0.2997 (un-normalized)	0	?
T = 2				

POMDP: Example, Cont'd

Fig. 3. Simple POMDP to illustrate belief state evolution.

$$b'(s') = \frac{O(s', a, o) \sum_{s \in S} T(s, a, s') b(s)}{\Pr(o|a, b)}$$

b'(s4) = O(s4, EAST, o)[T(s1,EAST,s4)b(s1) + T(s2,EAST,s4)b(s2) + T(s3,EAST,s4)b(s3) + T(s4,EAST,s4)b(s4)]*

= 1*[0.0*0.333 + 0.0*0.333 + 0.9*0 + 0.9*0.333]

= 0.2997 (un-normalized)

	S1	S2	S3	S4
T = 0	0.333	0.333	0	0.333
T = 1	0.0667 (un-normalized)	0.2997 (un-normalized)	0	0.2997 (un-normalized)
T = 2				

POMDP: Example, Cont'd

Fig. 3. Simple POMDP to illustrate belief state evolution.

$$b'(s') = \frac{O(s', a, o) \sum_{s \in S} T(s, a, s') b(s)}{\Pr(o|a, b)}$$

Now to normalize:

- Sum the values Pr(o|a,b) = 0.0667 + 0.2997 + 0 + 0.2997 = 0.6667

	S1	S2	S3	S4
T = 0	0.333	0.333	0	0.333
T = 1	0.1	0.45	0	0.45
T = 2				

POMDP: Example, Cont'd

Fig. 3. Simple POMDP to illustrate belief state evolution.

$$b'(s') = \frac{O(s', a, o) \sum_{s \in S} T(s, a, s') b(s)}{\Pr(o|a, b)}$$

If we take action EAST again

How the belief states evolve depends on our choice of actions

- How should we explore our actions?

	S1	S2	S3	S4
T = 0	0.333	0.333	0	0.333
T = 1	0.1	0.45	0	0.45
T = 2	0.1	0.167	0	0.736

POMDP: Finding an Optimal Policy

The policy component of a POMDP agent must map the current belief state into action. Because the belief state is a sufficient statistic, the optimal policy is the solution of a continuous-space "belief MDP." It is defined as follows:

- $\mathcal{B}$, the set of belief states, comprise the state space;
- $\mathcal{A}$, the set of actions, remains the same;
- $\tau(b, a, b')$ is the state-transition function, which is defined as

$$\begin{aligned}\tau(b, a, b') &= \Pr(b'|a, b) \\ &= \sum_{o \in \Omega} \Pr(b'|a, b, o) \Pr(o|a, b) \quad ,\end{aligned}$$

where

$$\Pr(b'|b, a, o) = \begin{cases} 1 \text{ if } SE(a, b, o) = b' \\ 0 \text{ otherwise} \end{cases} ;$$

Estimating …

and

- $\rho(b, a)$ is the reward function on belief states, constructed from the original reward function on world states:

$$\rho(b, a) = \sum_{s \in \mathcal{S}} b(s) R(s, a) \quad .$$

POMDP: Finding an Optimal Policy

The reward function: the agent appears to be rewarded for merely believing that it is in good states

- However, because the state estimator is constructed from a correct observation and transition model of the world, the belief state represents the **true occupation probabilities** for all states s
  - And therefore, the reward function represents the true expected reward to the agent

Reward is now based on belief + action, not state + action

$$\rho(b, a) = \sum_{s \in \mathcal{S}} b(s) R(s, a)$$

POMDP: Finding an Optimal Policy

Policy Tree

When an agent has one step remaining, all it can do is take a single action

With two steps to go, it can take an action, make an observation, then take another action, perhaps depending on the previous observation

In general, an agent's non-stationary *t*-step policy can be represented by a policy tree

POMDP: Finding an Optimal Policy

Policy Tree 2

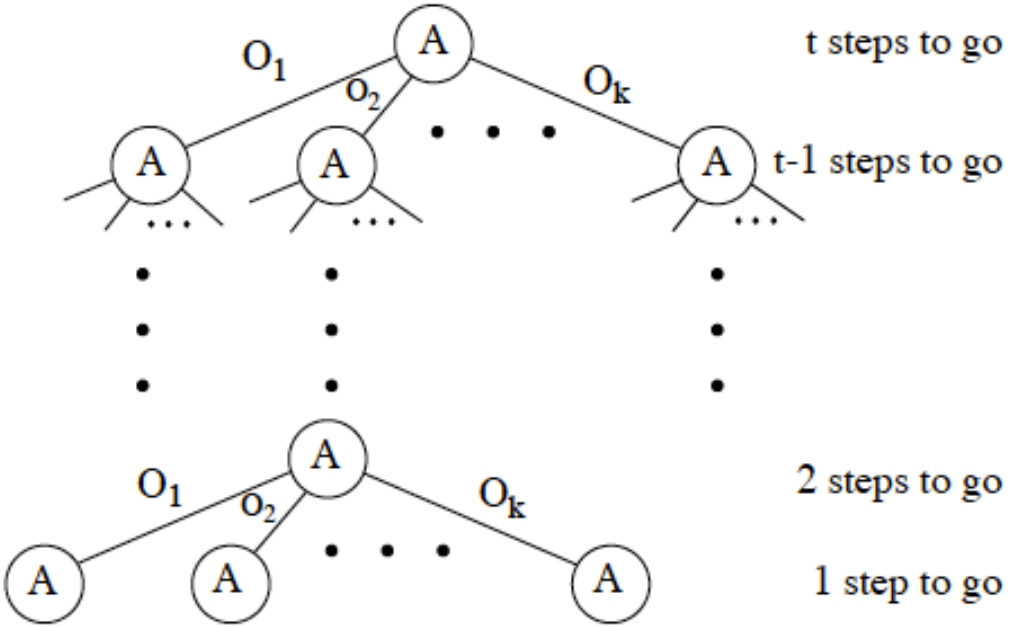


Fig. 4. A t-step policy tree.

POMDP: Finding an Optimal Policy

Policy Tree 3

In the simplest case, p is a 1-step policy tree (a single action). The value of executing that action in state s is

$$V_p(s) = R(s, a(p)) \ ,$$

where $a(p)$ is the action specified in the top node of policy tree p. More generally, if p is a t-step policy tree, then

$$\begin{aligned} V_p(s) &= R(s, a(p)) + \gamma \quad \text{Expected value of the future} \\ &= R(s, a(p)) + \gamma \sum_{s' \in \mathcal{S}} \Pr(s'|s, a(p)) \sum_{o_i \in \Omega} \Pr(o_i|s', a(p)) V_{o_i(p)}(s') \\ &= R(s, a(p)) + \gamma \sum_{s' \in \mathcal{S}} T(s, a(p), s') \sum_{o_i \in \Omega} O(s', a(p), o_i) V_{o_i(p)}(s') \ , \end{aligned}$$

where $o_i(p)$ is the $(t-1)$-step policy subtree associated with observation o_i at the top level of a t-step policy tree p. The expected value of the future is computed by first taking an expectation over possible next states, s', then considering the value of each of those states. The value depends on which policy subtree will be executed which, itself, depends on which observation is made. So, we take another expectation, with respect to the possible observations, of the value of executing the associated subtree, $o_i(p)$, starting in state s'.

POMDP: Finding an Optimal Policy

Policy Tree 4

Because the agent will never know the exact state of the world, it must be able to determine the value of executing a policy tree, p, from some belief state b. This is just an expectation over world states of executing p in each state:

$$V_p(b) = \sum_{s \in \mathcal{S}} b(s) V_p(s) \quad .$$

It will be useful, in the following exposition, to express this more compactly. If we let $\alpha_p = \langle V_p(s_1), \ldots, V_p(s_n) \rangle$, then $V_p(b) = b \cdot \alpha_p$.

POMDP: Finding an Optimal Policy

Policy Tree 5

Now we have the value of executing the policy tree *p* in every possible belief state. To construct an optimal *t*-step policy, however, it will generally be necessary to execute ***different policy trees from different initial belief states***. Let *P* be the finite set of ***all*** *t*-step policy trees. Then

$$V_t(b) = \max_{p \in \mathcal{P}} b \cdot \alpha_p \quad .$$

That is, the optimal t-step value of starting in belief state b is the value of executing the best policy tree in that belief state.

The Tiger Problem

Imagine an agent standing in front of two closed doors

- Behind one door: a **tiger**; behind the other: a **large reward**
- If the agent opens the door with the tiger, then a large penalty is received.
- Or the agent can listen, in order to gain some information about the location of the tiger.
  - not free
  - not entirely accurate
    - There is a chance that the agent will hear a tiger behind the left-hand door when the tiger is really behind the right-hand door, and vice versa.

The Tiger Problem 2

States *S*:

- { s0, s1 } s0 = tiger is on the left, s1 = tiger is on the right

Actions *A*:

- { OPEN-LEFT, OPEN-RIGHT, LISTEN }

Rewards *R*:

- Open correct door = + 10, Open wrong door = -100, Cost of listening = -1

Two possible observations *O*:

- { TIGER-LEFT, TIGER-RIGHT }

Immediately after the agent opens a door and receives a reward/penalty, the problem **resets**, randomly relocating the tiger behind one of the two doors

The Tiger Problem 2

Transitions *T*:

- The LISTEN action does ***not*** change the state of the world.
  - T(s0, LISTEN, s0) = 1
  - T(s0, LISTEN, s1) = 0
  - T(s1, LISTEN, s0) = 0
  - T(s1, LISTEN, s1) = 1
- The OPEN-LEFT and OPEN-RIGHT actions cause a transition to world state s0 with probability 0.5 and to state s1 with probability 0.5 (after observation & reward)
  - essentially resetting the problem
  - T(s0, OPEN-LEFT, s0) = 0.5, T(s0, OPEN-LEFT, s1) = 0.5
  - T(s1, OPEN-LEFT, s0) = 0.5, T(S1, OPEN-LEFT, s1) = 0.5
  - T(s0, OPEN-RIGHT, s0) = 0.5, T(s0, OPEN-RIGHT, s1) = 0.5
  - T(s1, OPEN-RIGHT, s0) = 0.5, T(S1, OPEN-RIGHT, s1) = 0.5

The Tiger Problem 3

Observations *O*:

- When the world is in s0, LISTEN → Observation of TIGER-LEFT with probability 0.85 and observation of TIGER-RIGHT with probability 0.15; and vice versa for s1
  - O(s0, LISTEN, TIGER-LEFT) = 0.85, O (s0, LISTEN, TIGER-RIGHT) = 0.15
  - O(s1, LISTEN, TIGER-LEFT) = 0.15, O (s1, LISTEN, TIGER-RIGHT) = 0.85
- With no knowledge of the world state, OPEN-LEFT → either observation with probability 0.5; OPEN-RIGHT → either observation with probability 0.5
  - Essentially any observation without the listen action is **uninformative**
  - O(s0, OPEN-LEFT, TIGER-LEFT) = 0.5, O (s0, OPEN-LEFT, TIGER-RIGHT) = 0.5
  - O(s0, OPEN-RIGHT, TIGER-LEFT) = 0.5, O (s0, OPEN-RIGHT, TIGER-RIGHT) = 0.5
  - O(s1, OPEN-LEFT, TIGER-LEFT) = 0.5, O (s1, OPEN-LEFT, TIGER-RIGHT) = 0.5
  - O(s1, OPEN-RIGHT, TIGER-LEFT) = 0.5, O (s1, OPEN-RIGHT, TIGER-RIGHT) = 0.5

The Tiger Problem 4

Rewards *R*:

- R(s0, LISTEN) = -1
- R(s1, LISTEN) = -1
- R(s0, OPEN-LEFT) = -100
- R(s1, OPEN-LEFT) = +10
- R(s0, OPEN-RIGHT) = +10
- R(s1, OPEN-RIGHT) = -100

The Tiger Problem
Finite Horizon Policies

If only one step

- If the agent believes with high probability that the tiger is on the left → best action = OPEN-RIGHT; and vice versa
- But what if the agent is highly uncertain about the tiger's location?
  - Best action = LISTEN (-1)
  - Why?
    - Guessing correctly = +10; guessing incorrectly = -100
    - Expected reward = (-100 + 10)/2 = -45!

The Tiger Problem
Finite Horizon Policies 2

Horizon = 1 step

Fig. 10. The optimal situation-action mapping for $t = 1$ for the tiger problem shows that each of the three actions is optimal for *some* belief state.

The belief interval is specified in terms of *b*(s0) only since *b*(s1) = 1-***b***(s0)

The Tiger Problem
Finite Horizon Policies 3

If only two steps

- LISTEN? Or opening a door?
- Opening a door at t = 2, the tiger would be randomly placed behind one of the doors → agent's belief state = (0.5, 0.5)
  - Then with only one step left = LISTEN would be the best action
- Best action = LISTEN (-1)

The Tiger Problem

Finite Horizon Policies 4

Horizon = 2 steps

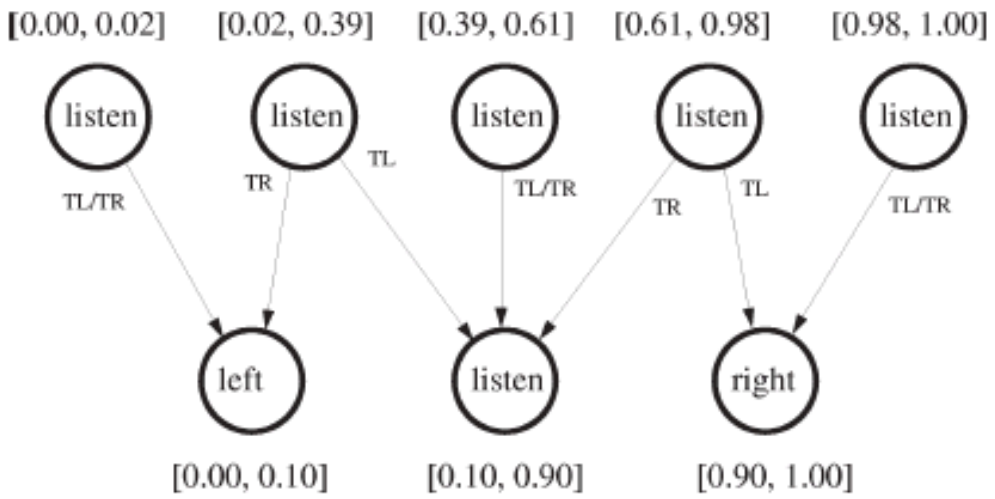


Fig. 12. The optimal nonstationary policy for $t = 2$ illustrates belief state transformations from $t = 2$ to $t = 1$. It consists of five separate policy trees.

The Tiger Problem

Finite Horizon Policies 5

Horizon = 3 steps

The optimal nonstationary policy for $t = 3$ also consists solely of policy trees with the **LISTEN** action at their roots. If the agent starts from the uniform belief state, $b = (0.5, 0.5)$, listening once does not change the belief state enough to make the expected value of opening a door greater than that of listening. The argument for this parallels that for the $t = 2$ case.

The Tiger Problem
Finite Horizon Policies 6

Horizon = 4 steps

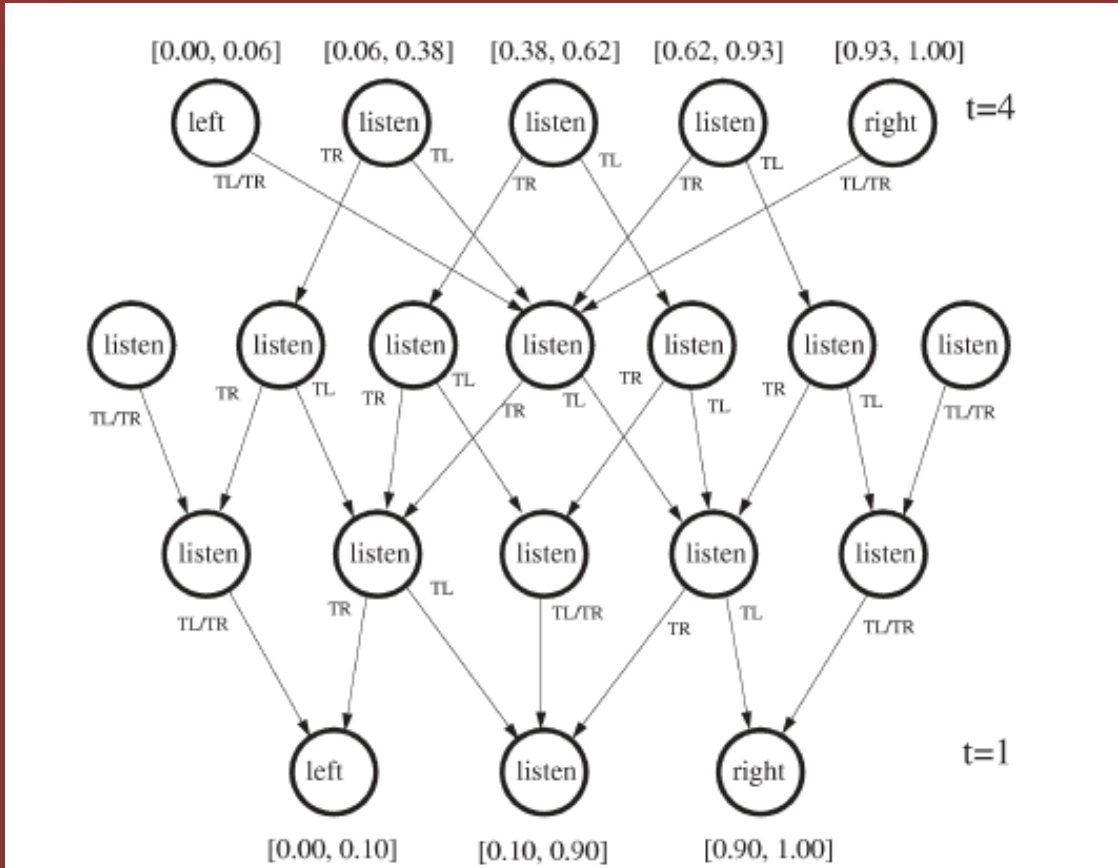


Fig. 13. The optimal nonstationary policy for $t = 4$ has a rich structure.

The Tiger Problem
Finite Horizon Policies 7

Policy trees

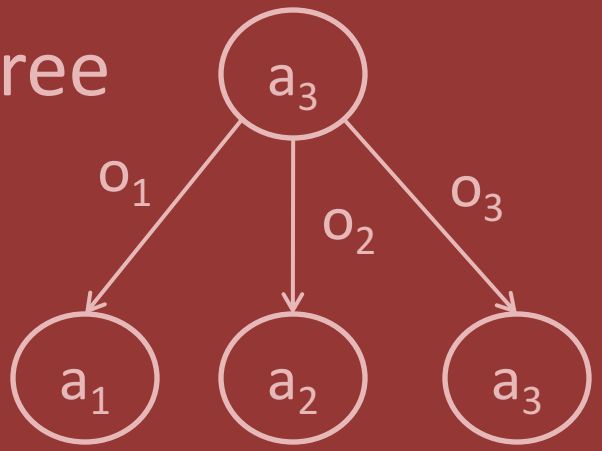
Choose actions based on observations

- Conditional plans

Define value function over trees

- Expected utility of following tree
- Optimize to build plan

One tree per state



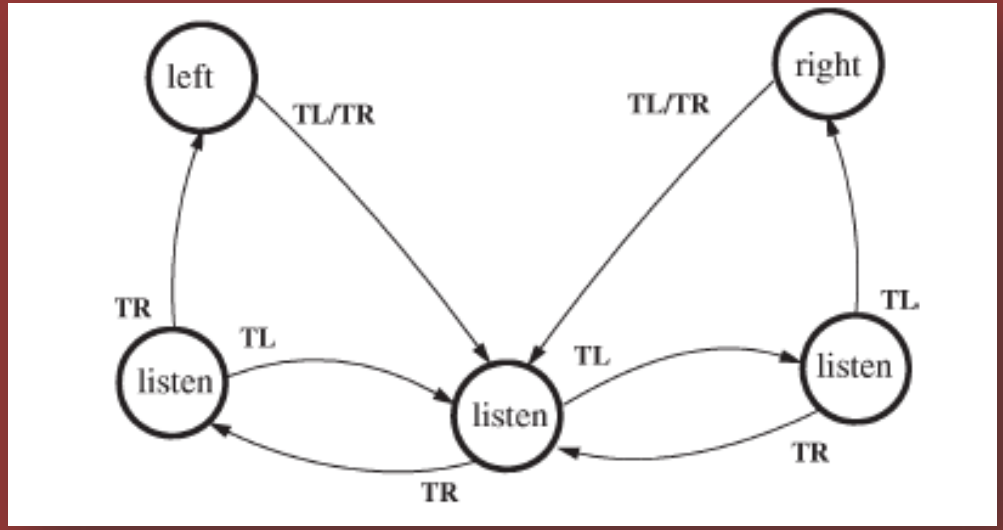
The Tiger Problem
Discounted Infinite Horizon

Finite Approximation

- Approximate with many step policy tree
- Stop when adding a new step offers little improvement

Plan Graph

- Policy tree converges to stationary policy
- Similar to Finite State Machines/Automata



Discussion

- Problem: difficult to plan in partially observable, nondeterministic environments
- Solution: state estimation, probabilistic transitions, utility maximization
- MDPs work great if environment is fully observable
- POMDPs handle “hidden” states but are harder to solve

Acknowledgments

Professor Adam Eck of Oberlin College, for his discussions and several slides