Intelligent Systems (AI-2)

Computer Science cpsc422, Lecture 6

Jan, 22, 2021

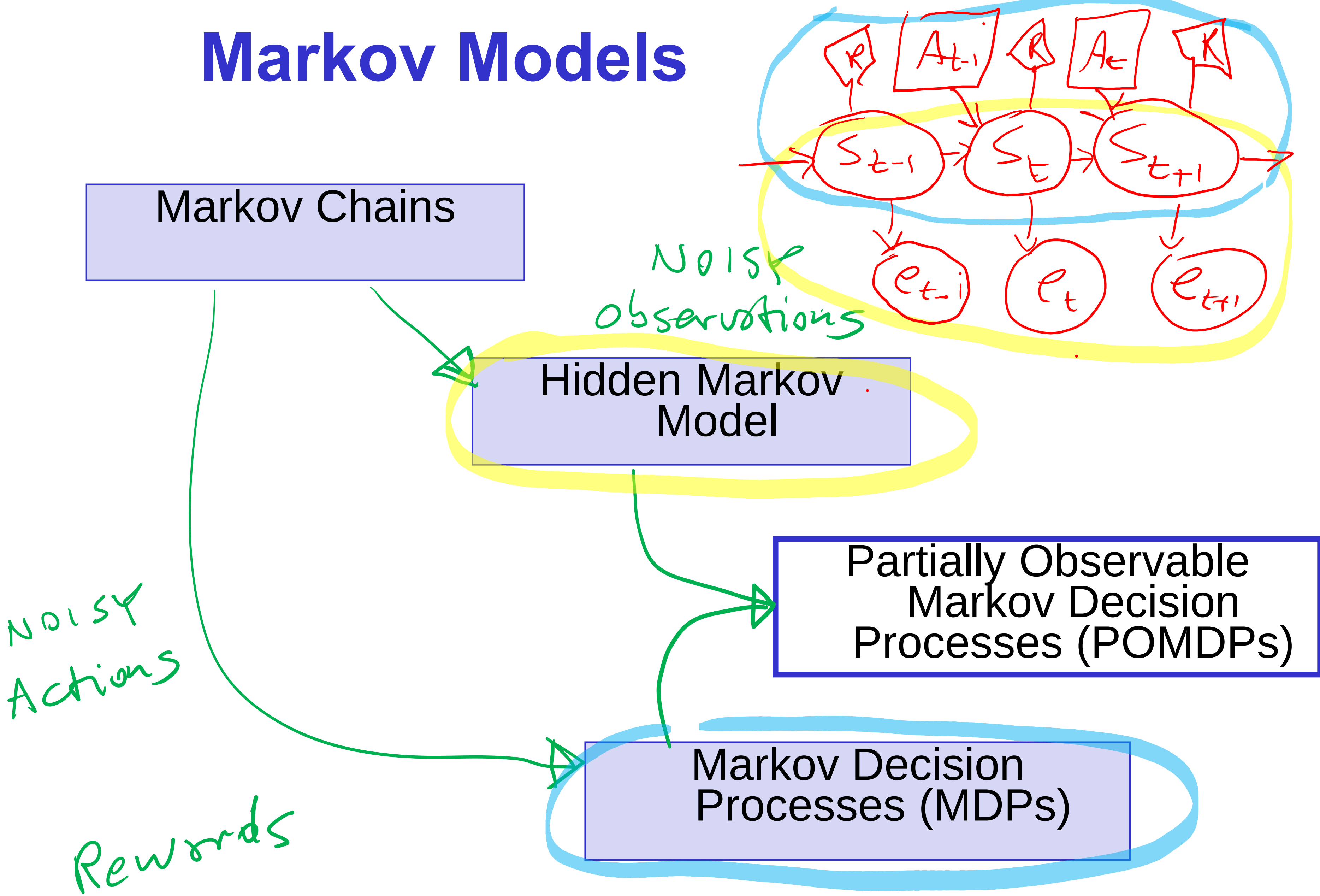
Slide credit POMDP: C. Conati and P. Viswanathan

Lecture Overview

Partially Observable Markov Decision Processes

- Summary
  - Belief State
  - Belief State Update
- Policies and Optimal Policy

Markov Models



Observation Model

- As in HMM, the agent can learn something about its actual state by *sensing* the environment:
  - **Sensor Model *P(e|s):*** probability of observing the evidence **e** in state ***s***

- A POMDP is fully specified by
  - Reward function: ***R(s)*** (we'll forget about *a* and *s'* for simplicity)
  - Transition Model: ***P(s' |a,s)***
  - Observation model: ***P(e|s)***

- Agent's belief state is updated by computing the conditional probability distribution over all the states given the sequence of observations and actions so far

State Belief Update

- We just saw *filtering* for HMM?
  - Compute conditional probability distribution over states at time t given all observations so far

$$\boldsymbol{P}(\boldsymbol{X}_t \mid \boldsymbol{e}_{0:t}) = \alpha \boldsymbol{P}(\boldsymbol{e}_t \mid \boldsymbol{X}_t) \sum_{x_{t-1}} \boldsymbol{P}(\boldsymbol{X}_t \mid \boldsymbol{x}_{t-1}) \, P(\boldsymbol{x}_{t-1} \mid \boldsymbol{e}_{0:t-1})$$

Filtering at time *t-1*

Inclusion of new evidence (sensor model)

Propagation to time *t*

- State belief update is similar but includes actions
  - If the agent has current belief state ***b(s)***, performs action ***a*** and then perceives evidence ***e***, the new belief state ***b'(s')*** is

$$b'(s') = \alpha P(e \mid s') \sum_{s} P(s' \mid a, s) b(s)$$

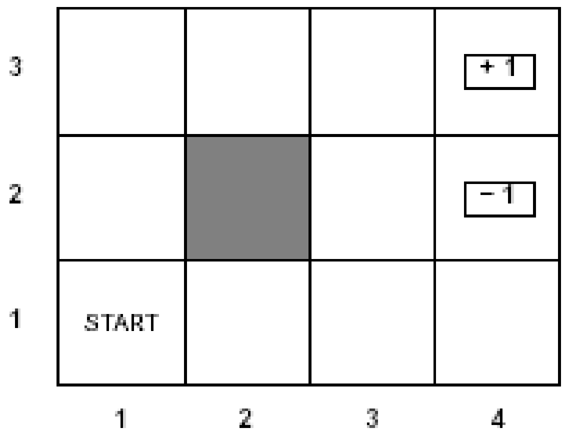
Inclusion of new evidence: Probability of perceiving ***e*** in ***s'***

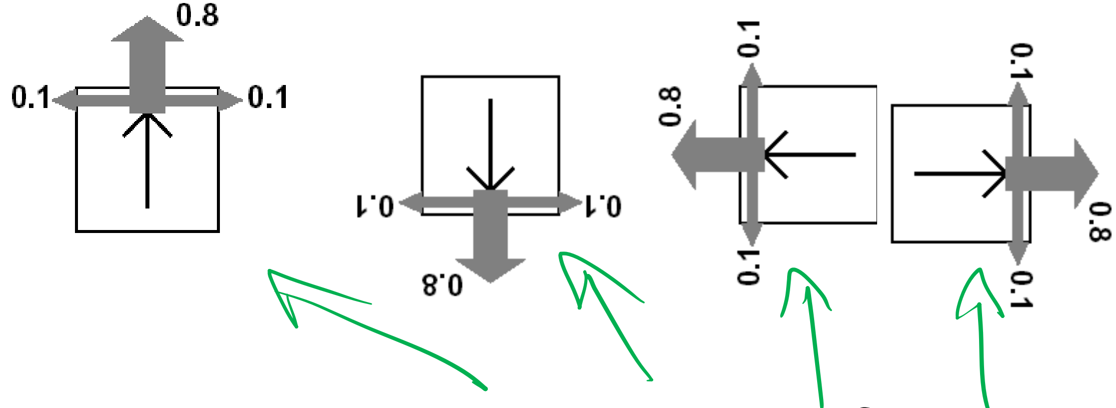
Sum over all the states that can take to **s'** after performing **a**

Filtering at time t-1: State belief based on all observations and actions up to t-1

Propagation at time t: Probability of transition to **s'** given **s** and **a**

Grid World Actions Reminder





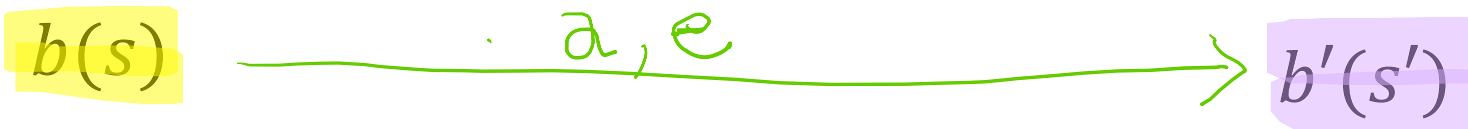
Agent moves in the above grid via actions *Up, Down, Left, Right*

Each action has:

- 0.8 probability to reach its intended effect
- 0.1 probability to move at right angles of the intended direction
- If the agents bumps into a wall, it says there

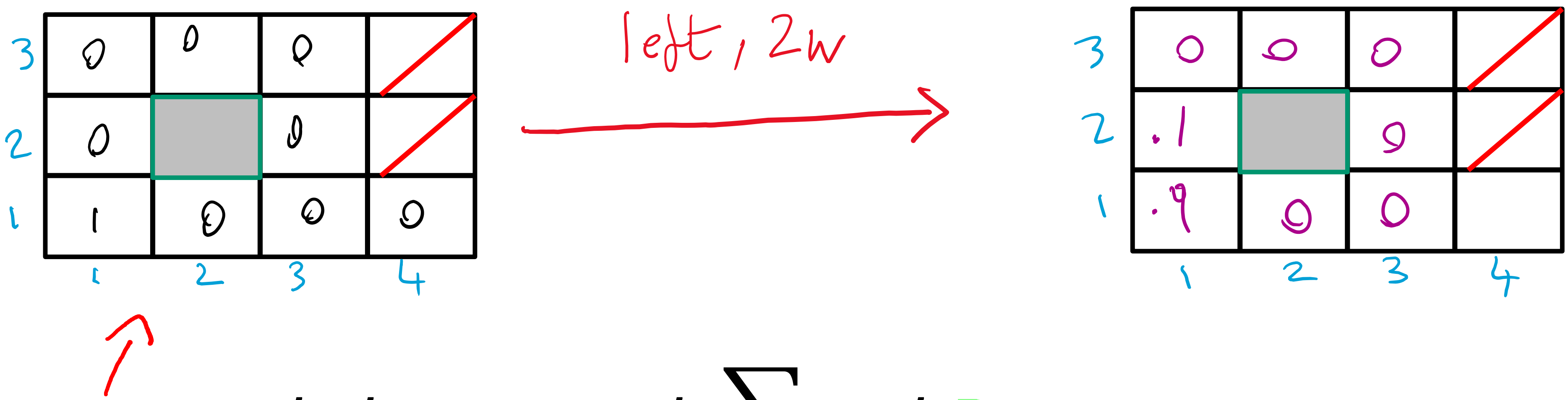
Belief Update

- When the agent performs action ***a*** in belief state ***b***, and then receives observation ***e***, it can compute the new belief state ***b'***



E = # of walls {1w, 2w}

A = {Up, Down, Left, Right}



$$b'(s') = \alpha P(e|s') \sum_{s} P(s'|s,a)\, b(s)$$

Example (no observation)

- What is the belief state after agent performs action *left* in the initial situation?

- The agent has no information about its position
  - Only one fictitious observation: *no observation*
  - *P(no observation | s) = 1* for every *s*

0.111	0.111	0.111	0.000
0.111		0.111	0.000
0.111	0.111	0.111	0.111

Let's instantiate $b'(s') = \alpha P(e \mid s') \sum_{s} P(s' \mid a, s) b(s)$

- For state (1,1) (action a = *left*)

$$b'(1,1) = \alpha\left[P((1,1) \mid (1,1), left) b(1,1) + P((1,1) \mid (2,1), left) b(2,1) + \ldots\right]$$

What is missing to get the correct answer?

A. $P((1,1) \mid (1,2), down) b(1,2)$

B. $P((1,1) \mid (1,3), left) b(1,3)$

C. $P((1,1) \mid (1,2), left) b(1,2)$

Example (no observation)

(column, row)

- Back to the grid world, what is the belief state after agent performs action *left* in the initial situation?
- The agent has no information about its position
  - Only one fictitious observation: *no observation*
  - *P(no observation | s) = 1* for every *s*

	1	2	3	4
3	0.111	0.111	0.111	0.000
2	0.111		0.111	0.000
1	0.111	0.111	0.111	0.111

- Let's instantiate $b'(s') = \alpha P(e \mid s') \sum_{s} P(s' \mid a, s) b(s)$
- For state (1,1) (action a = *left*)

$$b'(1{,}1) = \alpha\left[P((1{,}1) \mid (1{,}1), left) b(1{,}1) + P((1{,}1) \mid (2{,}1), left) b(2{,}1) + \ldots\right]$$

What is missing to get the correct answer?

A. $P((1{,}1) \mid (1{,}2), down) b(1{,}2)$

B. $P((1{,}1) \mid (1{,}3), left) b(1{,}3)$

C. $P((1{,}1) \mid (1{,}2), left) b(1{,}2)$

Example

- Back to the grid world, what is the belief state after agent performs action *left* in the initial situation?

0.111	0.111	0.111	0.000
0.111		0.111	0.000
0.111	0.111	0.111	0.111

- The agent has no information about its position
  - Only one fictitious observation: *no observation*
  - *P(no observation | s) = 1* for every *s*
- Let's instantiate $b'(s') = \alpha P(e \mid s') \sum_{s} P(s' \mid a, s) b(s)$

$$b'(1,1) = \alpha\left[P((1,1) \mid (1,1), left) b(1,1) + P((1,1) \mid (1,2), left) b(1,2) + P((1,1) \mid (2,1), left) b(2,1)\right]$$

$$b'(1,2) = \alpha\left[P((1,2) \mid (1,1), left) b(1,1) + P((1,2) \mid (1,2), left) b(1,2) + P((1,2) \mid (1,3), left) b(1,3)\right]$$

....

- Do the above for every state to get the new belief state

After five *Left* actions

0.111	0.111	0.111	0.000
0.111		0.111	0.000
0.111	0.111	0.111	0.111

0.300	0.010	0.008	0.000
0.221		0.059	0.012
0.371	0.012	0.008	0.000

bouncing 5 times with prob 10^{-1}

10^{-5}

?

tiny but not 0.

Belief State and its Update

Let's assume perfect sensor

$$b'(s') = \alpha P(e|s') \sum_s P(s'|s,a)\, b(s)$$

$b(s)$ —— a, e ——> $b'(s')$

left, 2w

$b(1,1)=1$

$$b'(1,1) = \alpha P(2w|(1,1)) \begin{bmatrix} P((1,1)|(1,1), left)b(1,1) + \\ P((1,1)|(1,2), left)b(1,2) + \\ P((1,1)|(2,1), left)b(2,1) \end{bmatrix} = .9$$

$\frac{.9}{.9+.1} = .9$

$b(1,2)=0$

$$b'(1,2) = \alpha P(2w|(1,2)) \begin{bmatrix} P((1,2)|(1,1), left)b(1,1) + \\ P((1,2)|(1,2), left)b(1,2) + \\ P((1,2)|(1,3), left)b(1,3) \end{bmatrix} = .1$$

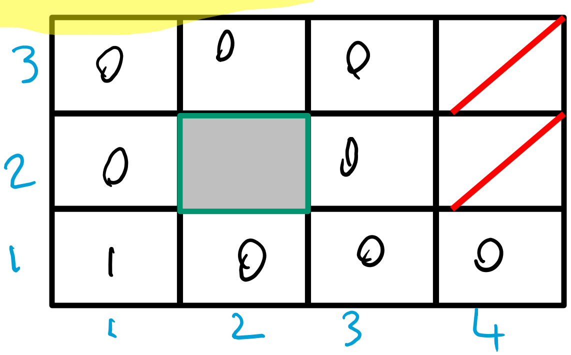
$\frac{.1}{.9+.1} = .1$

$b(2,1)=0$

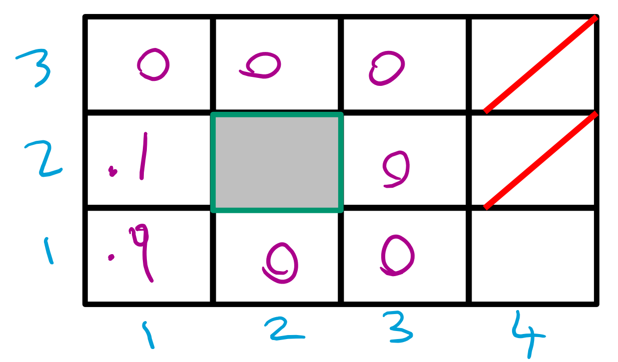
$$b'(2,1) = \alpha P(2w|(2,1))[\ldots\ldots] \quad 0 \quad = 0$$

$b(4,3)=0$

$$b'(4,3) = \alpha P(2w|(4,3))[\ldots\ldots] \quad 0$$



(column, row)



Belief Update: Example1

- The sensor that perceives the number of adjacent walls in a location has a 0.1 probability of error
  - P(2w|s) = 0.9 ; P(1w|s) = 0.1 if *s* is non-terminal and not in third column
  - P(1w|s) = 0.9 ; P(2w|s) = 0.1 if *s* is non-terminal and in third column
- Try to compute the new belief state if agent **moves *left*** and then **perceives 1 adjacent wall**

0.111	0.111	0.111	0.000
0.111		0.111	0.000
0.111	0.111	0.111	0.111

$$b'(s') = \alpha \; P(e \mid s') \sum_s P(s' \mid a, s) b(s)$$

$$b'(1,1) = \alpha \; X \left[P((1,1) \mid (1,1), left) b(1,1) + P((1,1) \mid (1,2), left) b(1,2) + P((1,1) \mid (2,1), left) b(2,1) \right]$$

X should be equal to ?

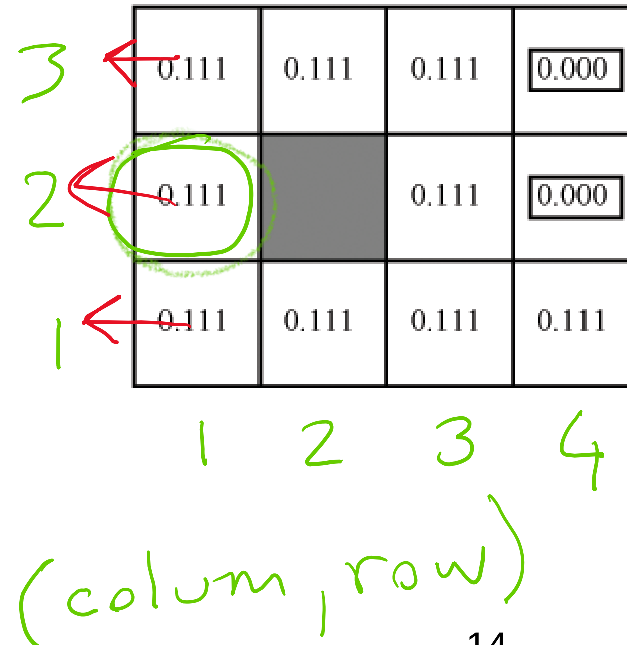
A. 0.1   **B.** 0.2   **C.** 0.9

Belief Update: Example 2

- Let's introduce a sensor that perceives the number of adjacent walls in a location with a 0.1 probability of error
  - P(2w|s) = 0.9 ; P(1w|s) = 0.1 if *s* is non-terminal and not in third column
  - P(1w|s) = 0.9 ; P(2w|s) = 0.1 if *s* is non-terminal and in third column
- Try to compute the new belief state if agent **moves *right*** and then **perceives 2 adjacent wall**

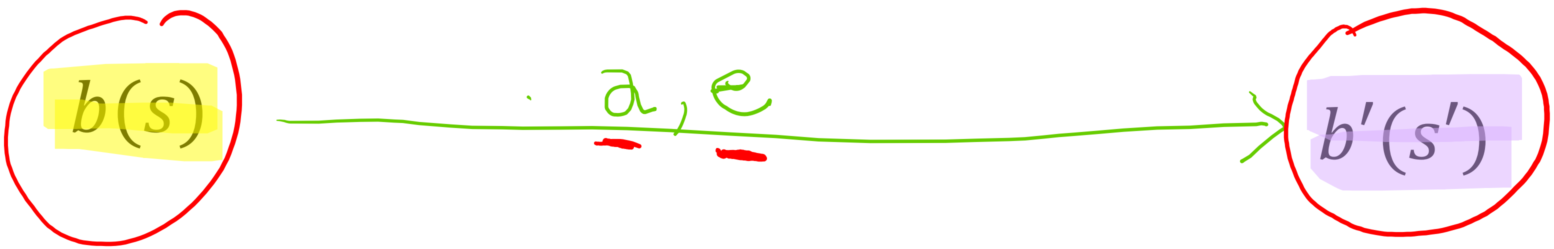
$$b'(s') = \alpha \; P(e \mid s') \sum_{s} P(s' \mid a, s) b(s)$$

$$b'(1,2) = \alpha \; P(2w \mid (1,2)) \times \left[\begin{array}{l} P((1,2) \mid (1,1), right) b(1,1) + \\ P((1,2) \mid (1,2), right) b(1,2) + \\ P((1,2) \mid (1,3), right) b(1,3) \end{array} \right]$$



Belief State and Belief Update: Summary

- When the agent performs action ***a*** in belief state ***b***, and then receives observation ***e***, it can compute the new belief state ***b'***



$$b'(s') = \alpha P(e|s') \sum_{s} P(s'|s,a)\, b(s)$$

- ***deterministic transition from one belief state to another***

Optimal Policies in POMDs ?

- Theorem (Astrom, 1965):
  - The optimal policy in a POMDP is a function $\pi^*(b)$ where b is the belief state (probability distribution over states)
- That is, $\pi^*(b)$ is a function from belief states (probability distributions) to actions
  - It does *not* depend on the actual state the agent is in
  - Good, because the agent does not know that, all it knows are its beliefs!
- Decision Cycle for a POMDP agent
  - Given current belief state *b*, execute $a = \pi^*(b)$
  - Receive observation *e*
  - compute : $b'(s') = \alpha P(e \mid s') \sum_{s} P(s' \mid s, a) b(s)$
  - Repeat

How to Find an Optimal Policy?

?

- Turn a POMDP into a corresponding MDP and then solve that MDP
- Generalize VI to work on POMDPs
- Develop Approx. Methods
  - Point-Based VI
  - Look Ahead

Finding the Optimal Policy: State of the Art

- Turn a POMDP into a corresponding MDP and then apply VI: only small models
- Generalize VI to work on POMDPs
  - 10 states in1998
  - 200,000 states in 2008-09 (now: always behind approx. methods)
- Develop Approx. Methods 2009 - now
  - Point-Based VI and Look Ahead
  - Even 50,000,000 states
    http://www.cs.uwaterloo.ca/~ppoupart/software.html

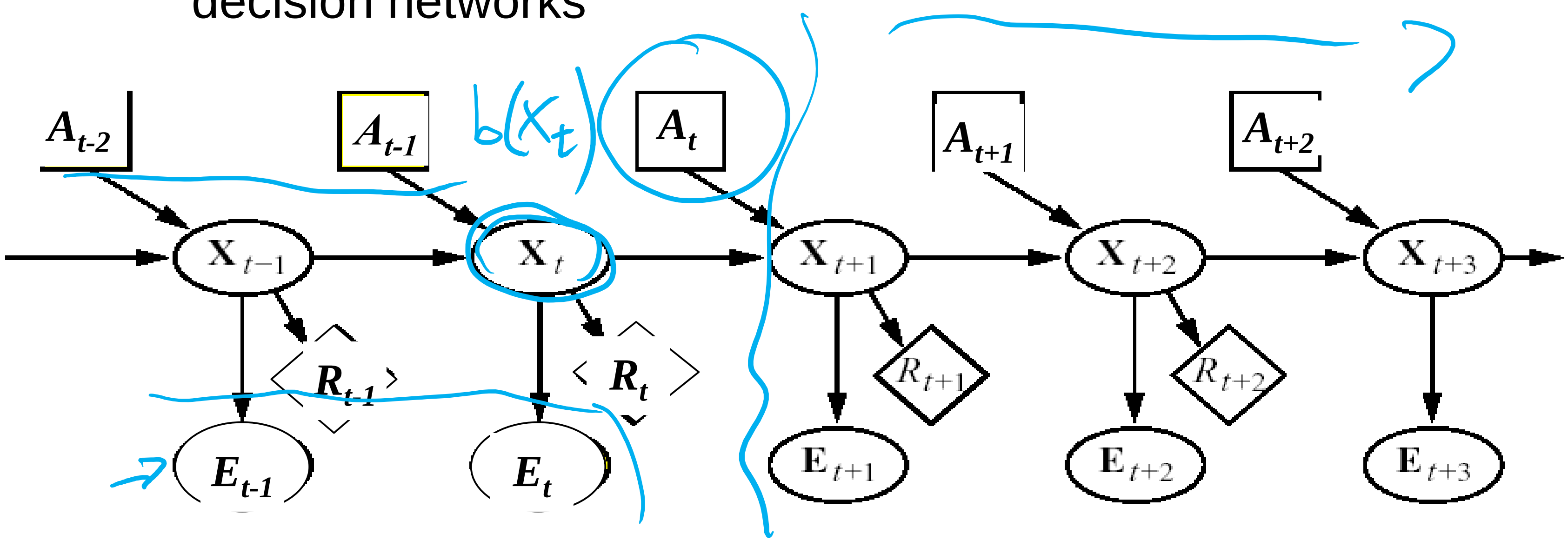
Recent Method: Point-based Value Iteration (not required)

- Find a solution **for a sub-set of all states**
- Not all states are necessarily reachable
- Generalize the solution to all states
- Methods include: PERSEUS, PBVI, and HSVI and other similar approaches (FSVI, PEGASUS)

Dynamic Decision Networks (DDN)

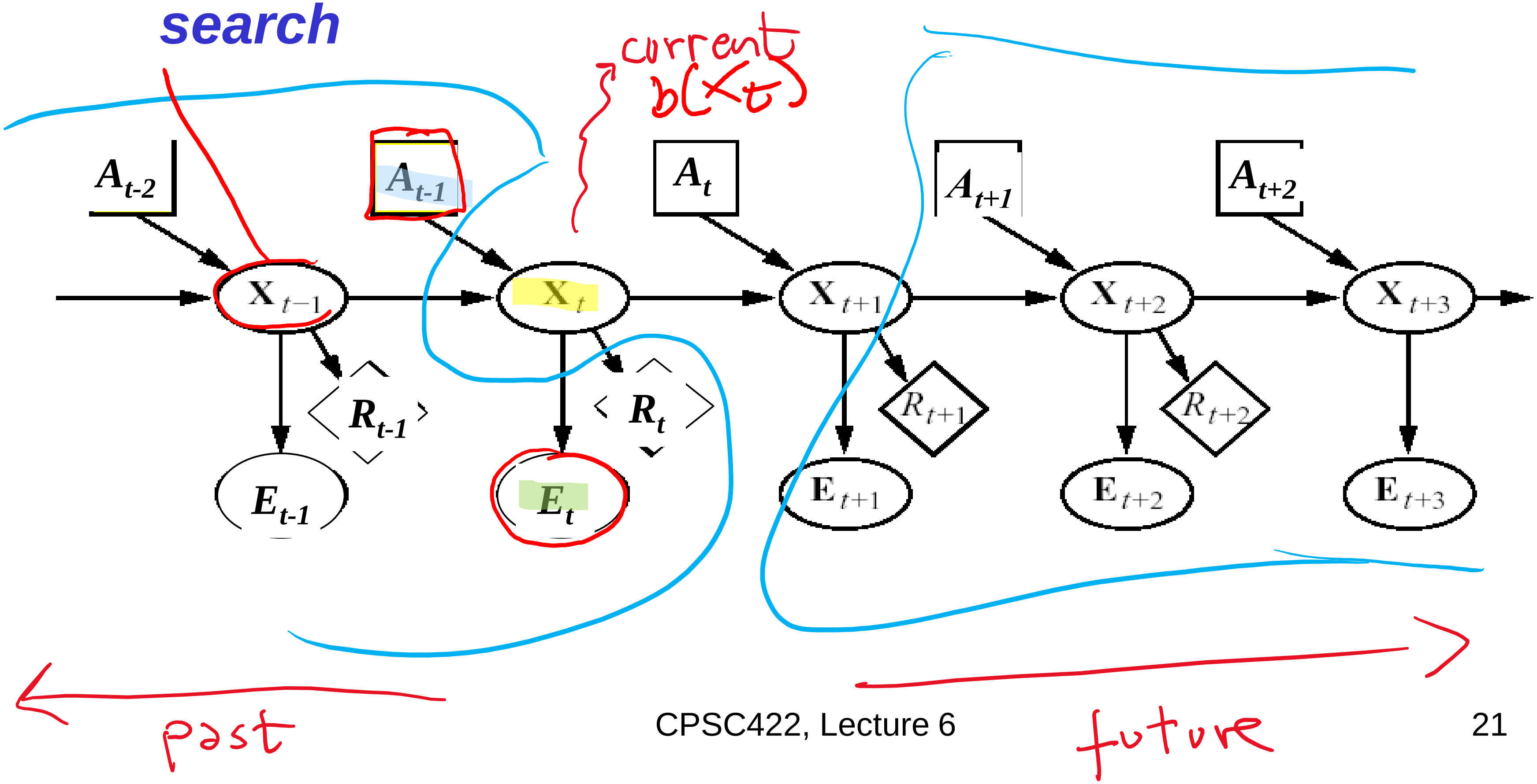
- Comprehensive approach to agent design in partially observable, stochastic environments

- Basic elements of the approach
  - Transition and observation models are represented via a Dynamic Bayesian Network (DBN).
  - The network is extended with decision and utility nodes, as done in decision networks

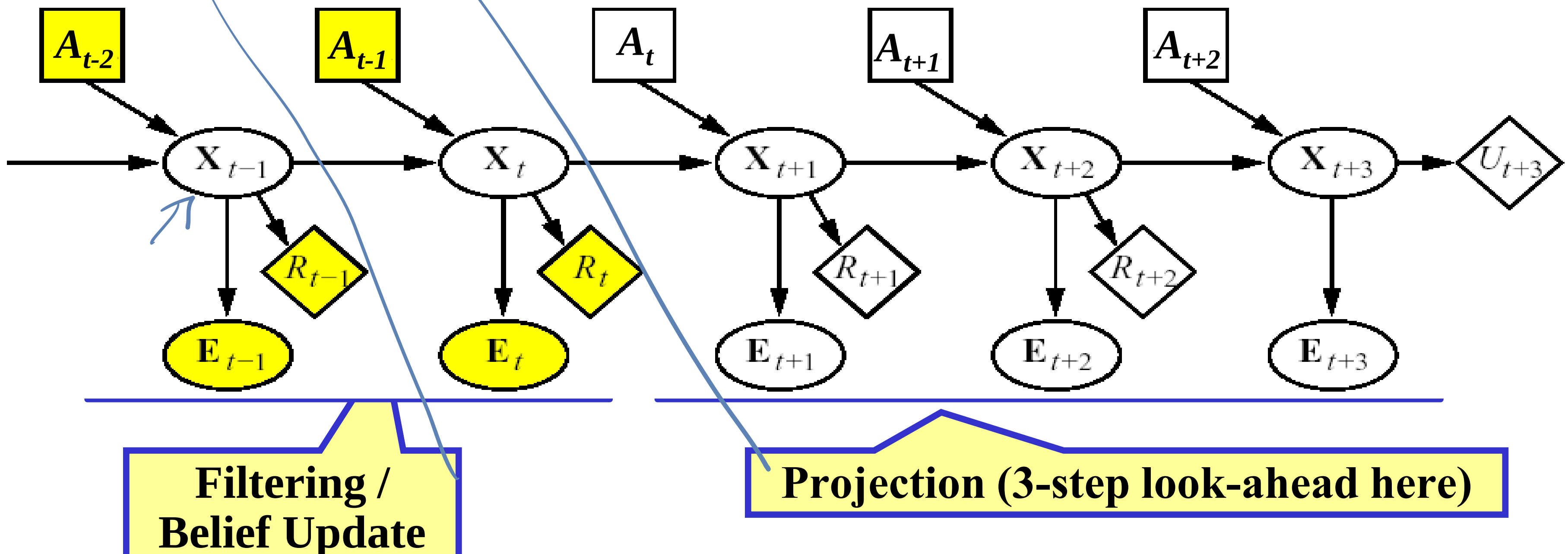


Dynamic Decision Networks (DDN)

- The Belief Update algorithm is used to incorporate each new percept and the action to update the belief state X_t
- Decisions are made by projecting forward possible action sequences and choosing the best one: ***look ahead search***



Dynamic Decision Networks (DDN)



- Nodes in yellow are known (evidence collected, decisions made, local rewards)
- Agent needs to make a decision at time t (A_t node)
- Network unrolled into the future for 3 steps
- Node U_{t+3} represents the utility (or expected optimal reward V^*) in state X_{t+3}
  - i.e., the reward in that state and all subsequent rewards
  - Available only in approximate form (from another approx. method)

Look Ahead Search for Optimal Policy

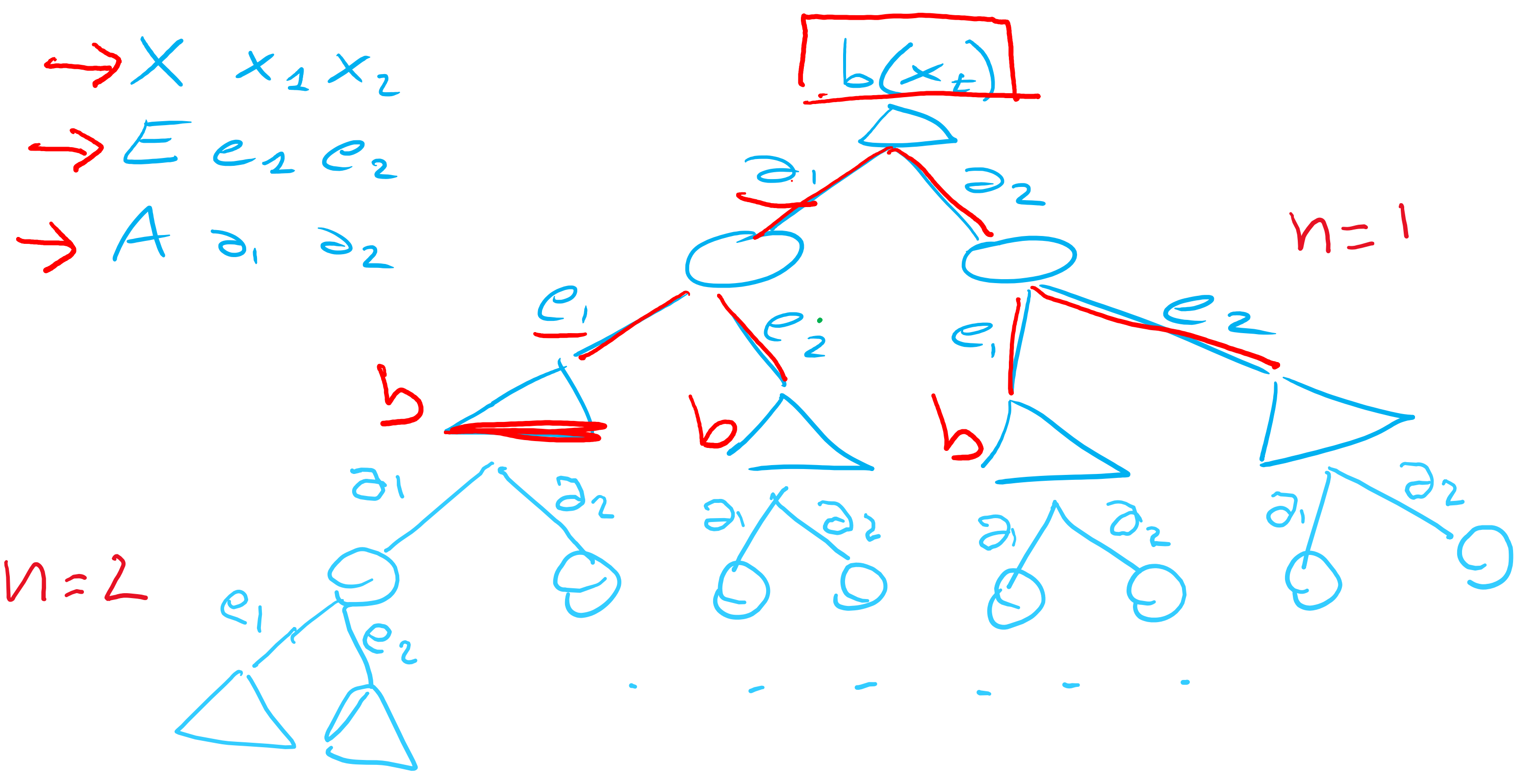
General Idea:

- **Expand the decision process for n steps into the future, that is**
  - "Try" all actions at every decision point
  - Assume receiving all possible observations at observation points
- **Result: tree of depth *2n+1*** where
  - every branch represents one of the possible sequences of *n* actions and n observations available to the agent, and the corresponding belief states
  - The leaf at the end of each branch corresponds to the *belief state* reachable via that sequence of actions and observations – use filtering/belief-update to compute it
- **"Back Up" the utility values of the leaf nodes** along their corresponding branches, **combining it with the rewards** along that path
- **Pick the branch with the highest expected value**

Look Ahead Search for Optimal Policy

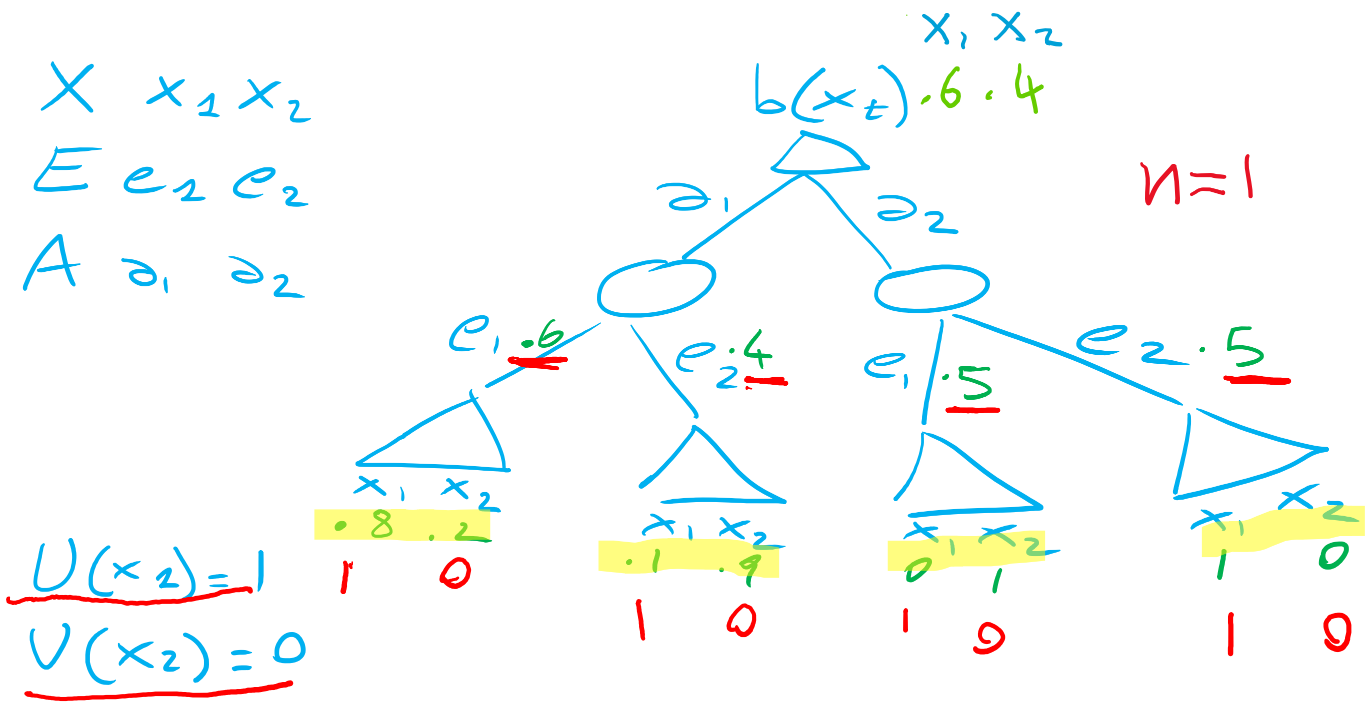
General Idea:

- **Expand the decision process for n steps into the future, that is**
  - "Try" all actions at every decision point
  - Assume receiving all possible observations at observation points



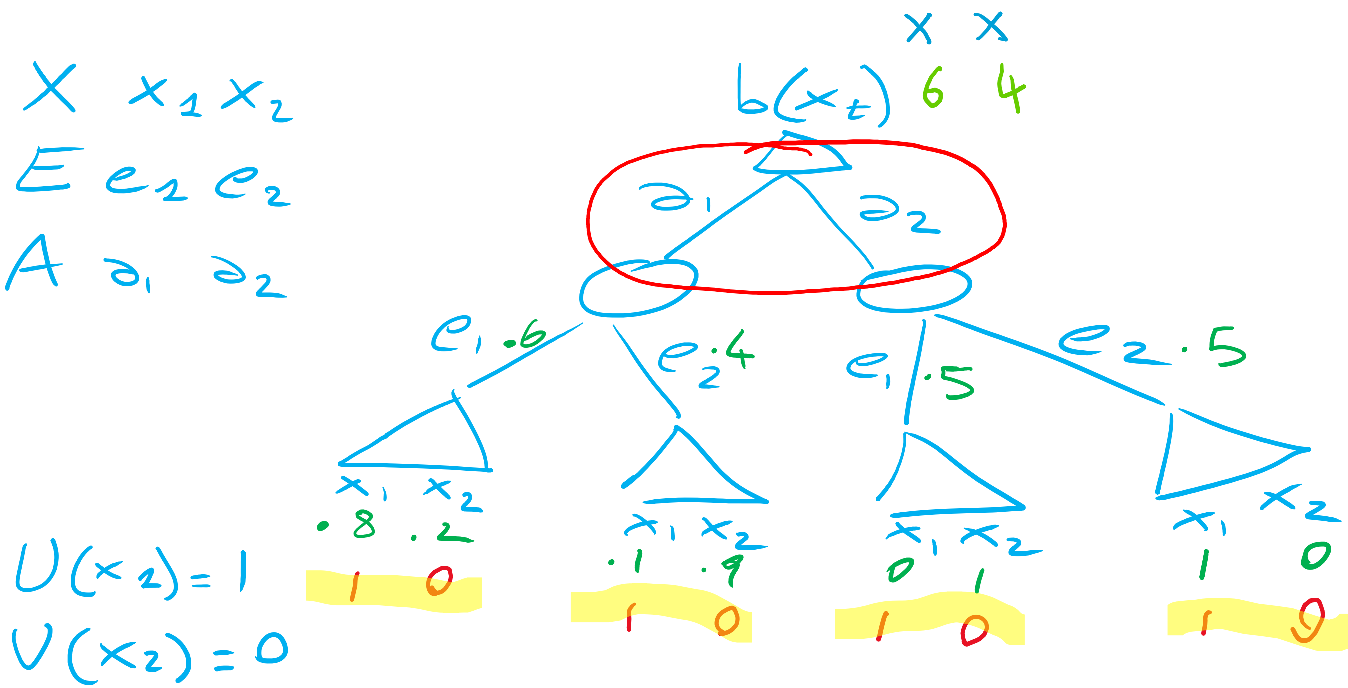
Look Ahead Search for Optimal Policy

- **Result: tree of depth 2*n*+1** where
  - every branch represents one of the possible sequences of *n* actions and n observations available to the agent, and the corresponding belief states
  - The leaf at the end of each branch corresponds to the *belief state* reachable via that sequence of actions and observations – use belief-update to compute it

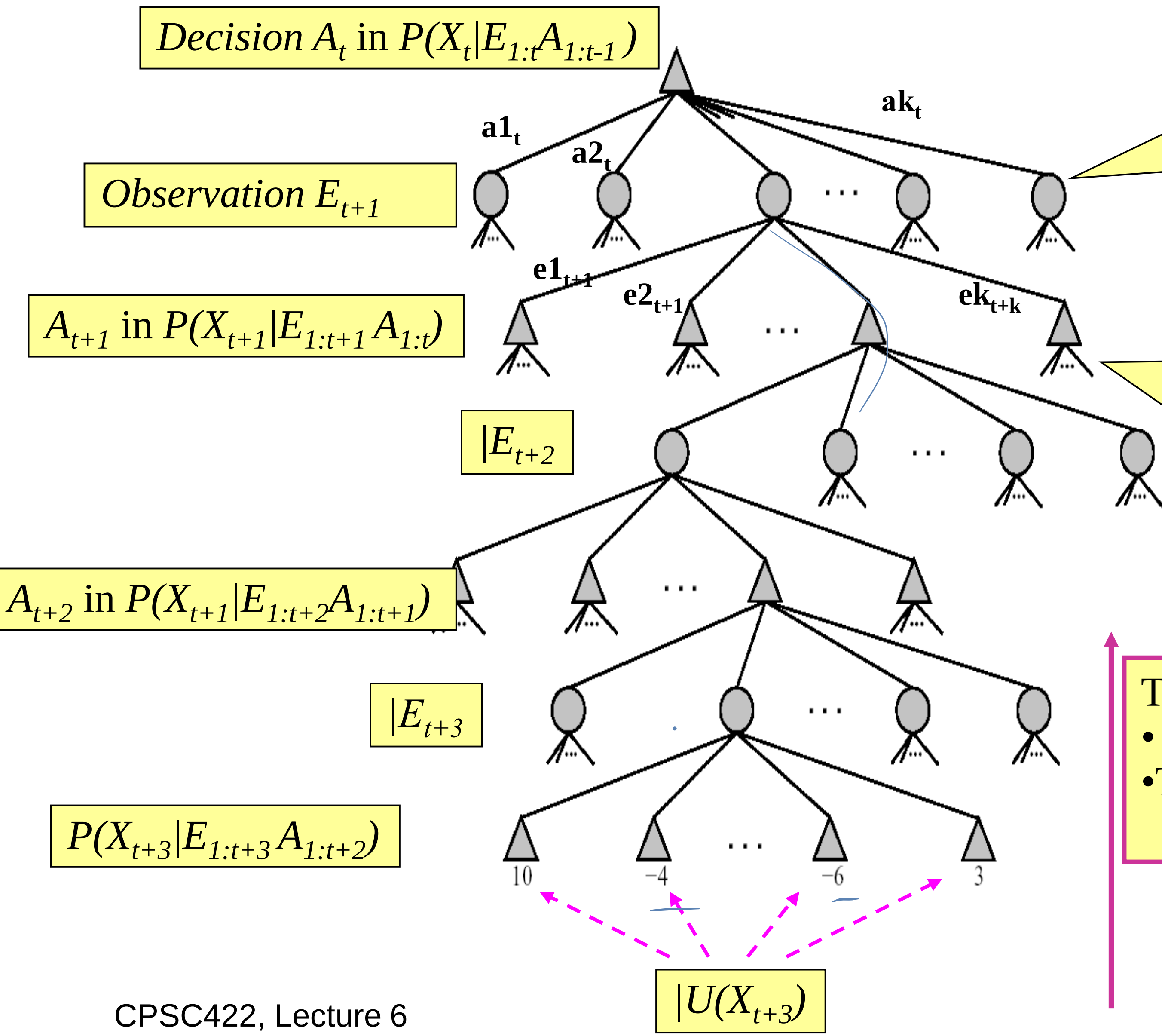


Look Ahead Search for Optimal Policy

- **"Back Up" the utility values of the leaf nodes** along their corresponding branches, **combining it with the rewards** along that path
- **Pick the branch with the highest expected value**



Look Ahead Search for Optimal Policy



These are chance nodes, describing the probability of each observation

Belief states are computed via any filtering algorithm, given the sequence of actions and observations up to that point

To back up the utilities
- take average at chance points
- Take max at decision points

on whiteboard

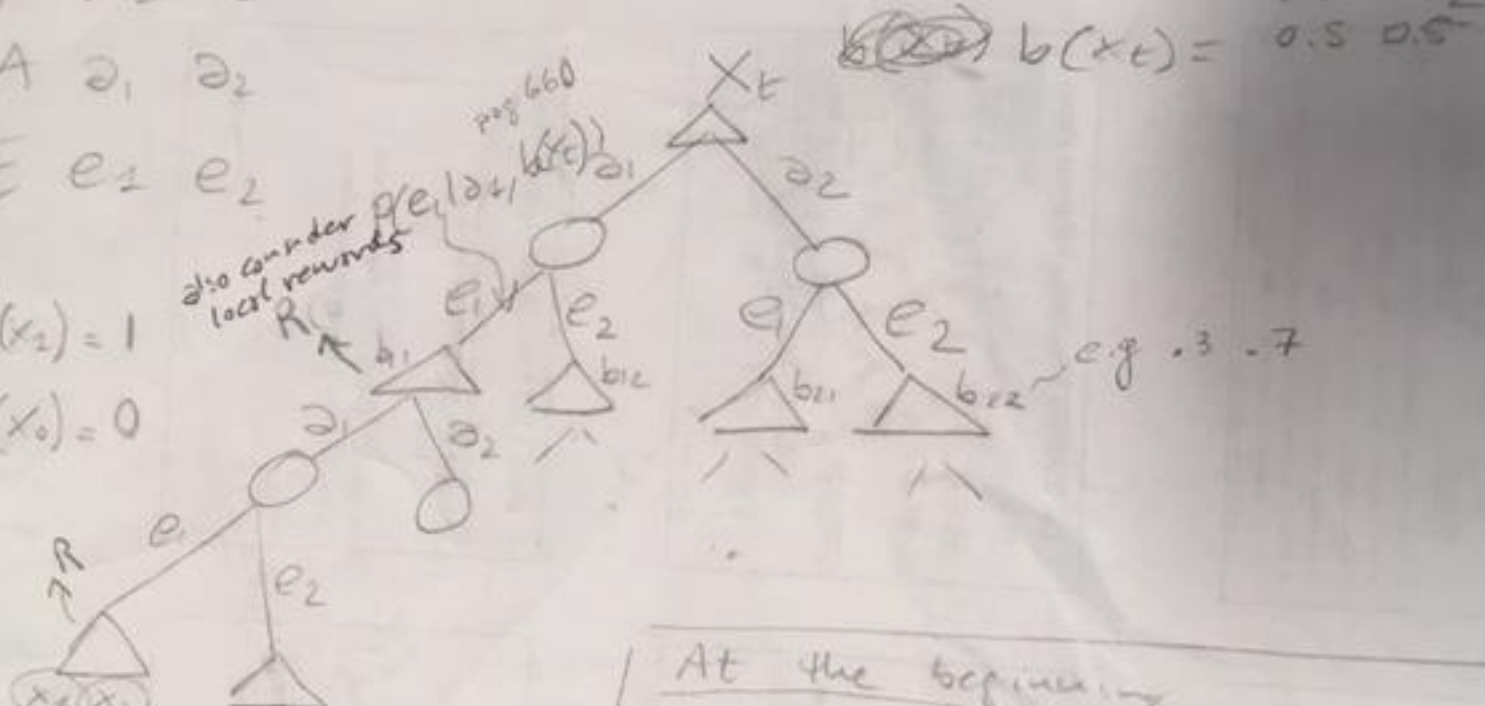
X x_1 x_2

A a_1 a_2

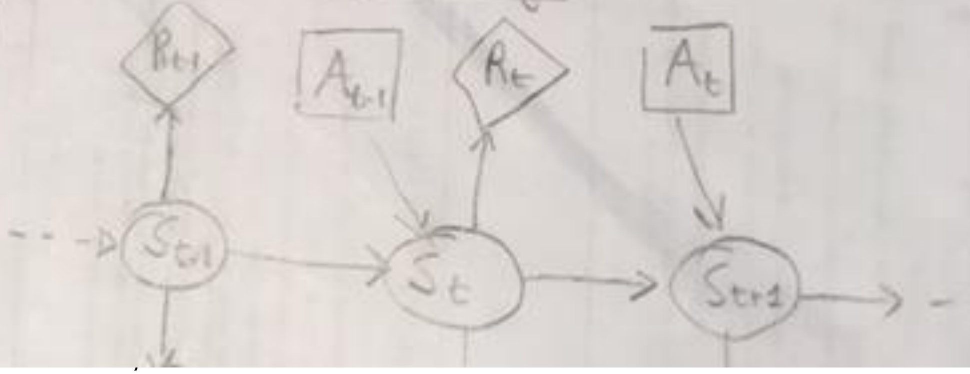
E e_1 e_2

$U(x_1) = 1$

$U(x_0) = 0$



At the beginning



P(

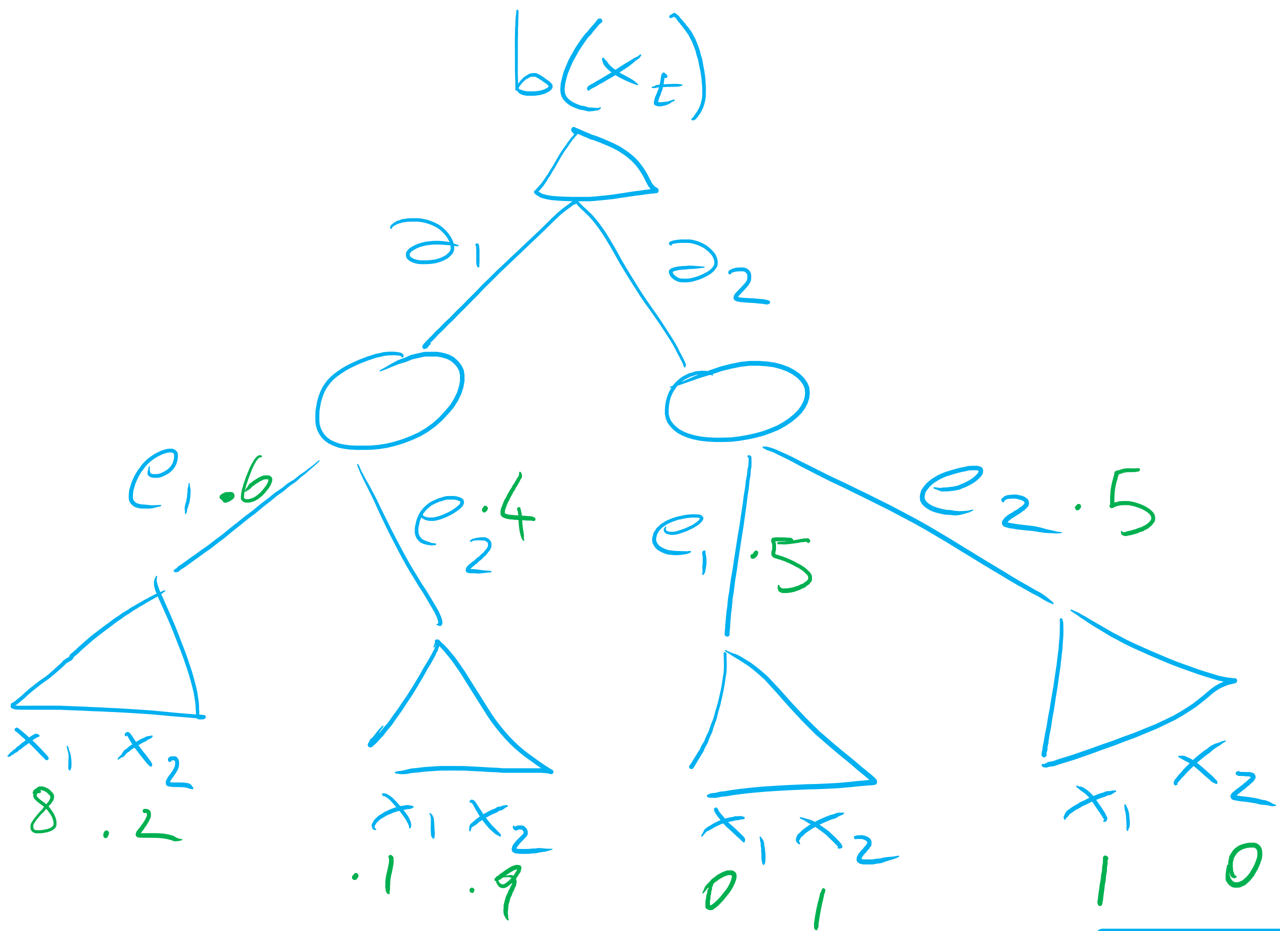
R(

P(

X x_1 x_2

E e_1 e_2

A a_1 a_2



$U(x_1) = 1$

$U(x_2) = 0$

- Best action at time t?

A. a_1

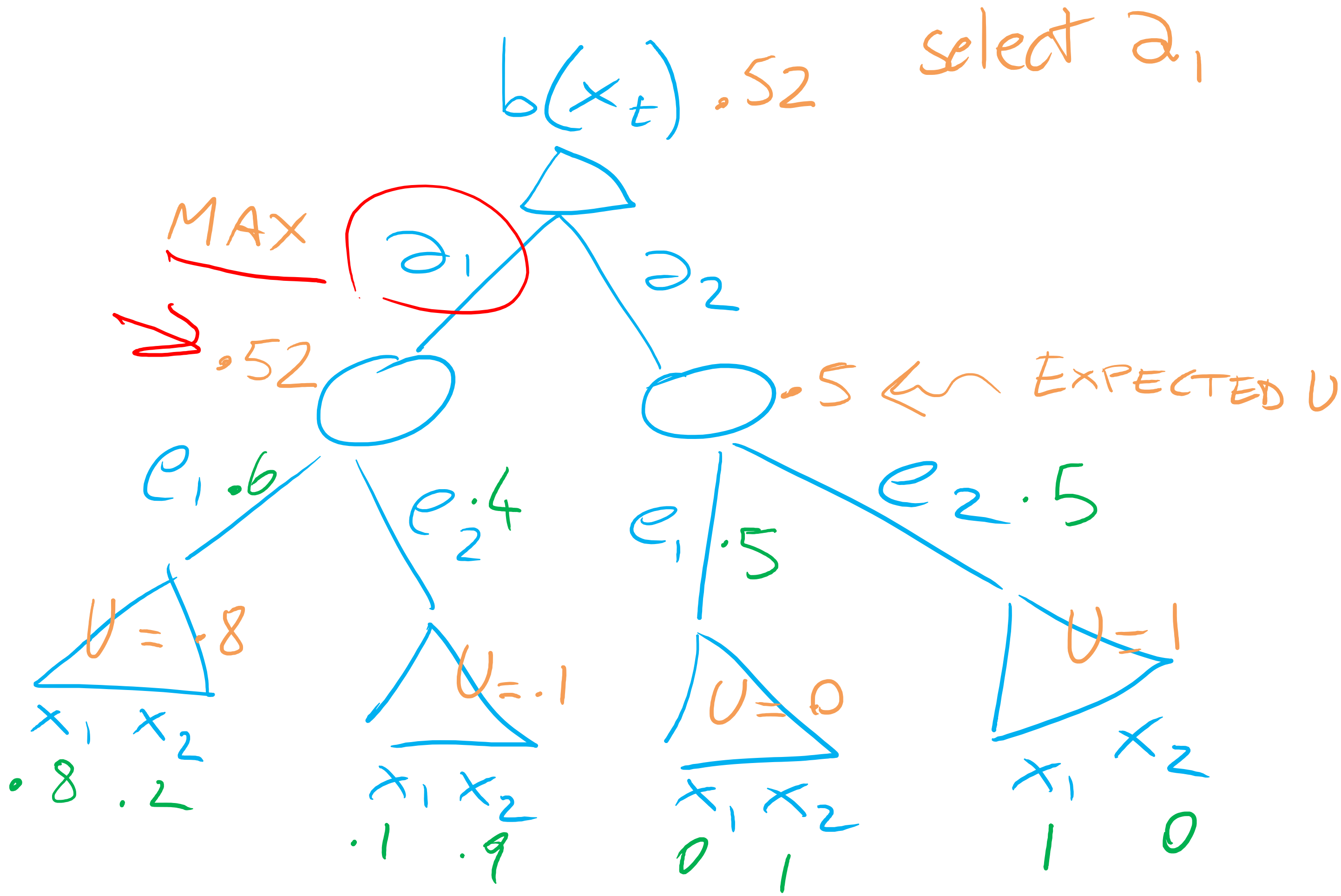
B. a_2

C. *indifferent*

X x_1 x_2

E e_1 e_2

A a_1 a_2



$U(x_1) = 1$

$U(x_2) = 0$

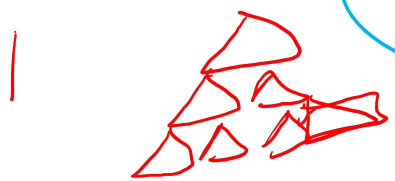
Look Ahead Search for Optimal Policy

➢ What is the time complexity for exhaustive search at depth *d*, with |A| available actions and |E| possible observations?

A. $O(d * |A| * |E|)$

B. $O(|A|^d * |E|^d)$

C. $O(|A|^d * |E|)$



d times

|A|*|E|*|A|*|E|*|A| . . . . . .

- Would Look ahead work better when the discount

factor is?

A. *Close to 1*

B. *Not too close to 1*

Some Applications of POMDPs……

- Jesse Hoey, Tobias Schröder, Areej Alhothali (**2015**), Affect control processes: Intelligent affective interaction using a POMDP, *AI Journal*
- S Young, M Gasic, B Thomson, J Williams (**2013**) POMDP-based Statistical Spoken Dialogue Systems: a Review, *Proc IEEE*,
- J. D. Williams and S. Young. Partially observable Markov decision processes for spoken dialog systems. *Computer Speech & Language*, 21(2):393–422, **2007**.
- S. Thrun, et al. Probabilistic algorithms and the interactive museum tour-guide robot Minerva. *International Journal of Robotic Research*, 19(11):972–999, **2000**.
- A. N.Rafferty,E. Brunskill,Ts L. Griffiths, and Patrick Shafto. Faster teaching by POMDP planning. In *Proc. of Ai in Education*, pages 280–287, **2011**
- P. Dai, Mausam, and D. S.Weld. Artificial intelligence for artificial artificial intelligence. In *Proc. of the 25th AAAI Conference on AI* , **2011**. [intelligent control of workflows]

- **Nan Ye, Adhiraj Somani, David Hsu and Wee Sun Lee (2017) "DESPOT: Online POMDP Planning with Regularization", Volume 58, pages 231-266**PDF | PostScript | doi:10.1613/jair.5328
  Appendix - Errata

- The partially observable Markov decision process (POMDP) provides a principled general framework for planning under uncertainty, but solving POMDPs optimally is computationally intractable, due to the "curse of dimensionality" and the "curse of history". To overcome these challenges, we introduce the Determinized Sparse Partially Observable Tree (DESPOT), a sparse approximation of the standard belief tree, for online planning under uncertainty. A DESPOT focuses online planning on a set of randomly sampled scenarios and compactly captures the "execution" of all policies under these scenarios. We show that the best policy obtained from a DESPOT is near-optimal, with a regret bound that depends on the representation size of the optimal policy. Leveraging this result, we give an anytime online planning algorithm, which searches a DESPOT for a policy that optimizes a regularized objective function. Regularization balances the estimated value of a policy under the sampled scenarios and the policy size, thus avoiding overfitting. The algorithm demonstrates strong experimental results, compared with some of the best online POMDP algorithms available. It has also been incorporated into an autonomous driving system for real-time vehicle control. The source code for the algorithm is available online

Another "famous" Application

Learning and Using POMDP models of Patient-Caregiver Interactions During Activities of Daily Living

Goal: Help Older adults living with cognitive disabilities (such as Alzheimer's) when they:

- **forget the proper sequence of tasks** that need to be completed
- **they lose track of the steps** that they have already completed.

Source: *Jesse Hoey UofT 2007*

422 big picture

StarAI (statistical relational AI)

Hybrid: Det +Sto

Prob CFG **Parsing**

Prob Relational Models

Markov Logics

Deterministic | **Stochastic**

Query

Logics

First Order Logics

Ontologies

- Full Resolution
- SAT

Belief Nets

Approx. : Gibbs

Markov Chains and HMMs

Forward, Viterbi….

Approx. : Particle Filtering

Undirected Graphical Models

Markov Networks

Conditional Random Fields

Planning

Markov Decision Processes and Partially Observable MDP

- Value Iteration
- Approx. Inference

Reinforcement Learning

Applications of AI

Representation

Reasoning Technique

Learning Goals for today's class

You can:

- Define a **Policy** for a POMDP
- Define and trace Belief Update
- Define and trace Look Ahead Search for finding an (approximate) Optimal Policy
- Compute Complexity of Look Ahead Search

TODO for Mon

- **Read textbook chpt 12**
  - 12.1 **Reinforcement Learning**
  - 12.3 Temporal Differences
  - 12.4 Q-learning

- **Assignment 1** has been posted on Canvas today (due Wed Feb 3 3:30PM)
  - VInfo and VControl
  - MDPs (Value Iteration)
  - POMDPs (Belief Update)