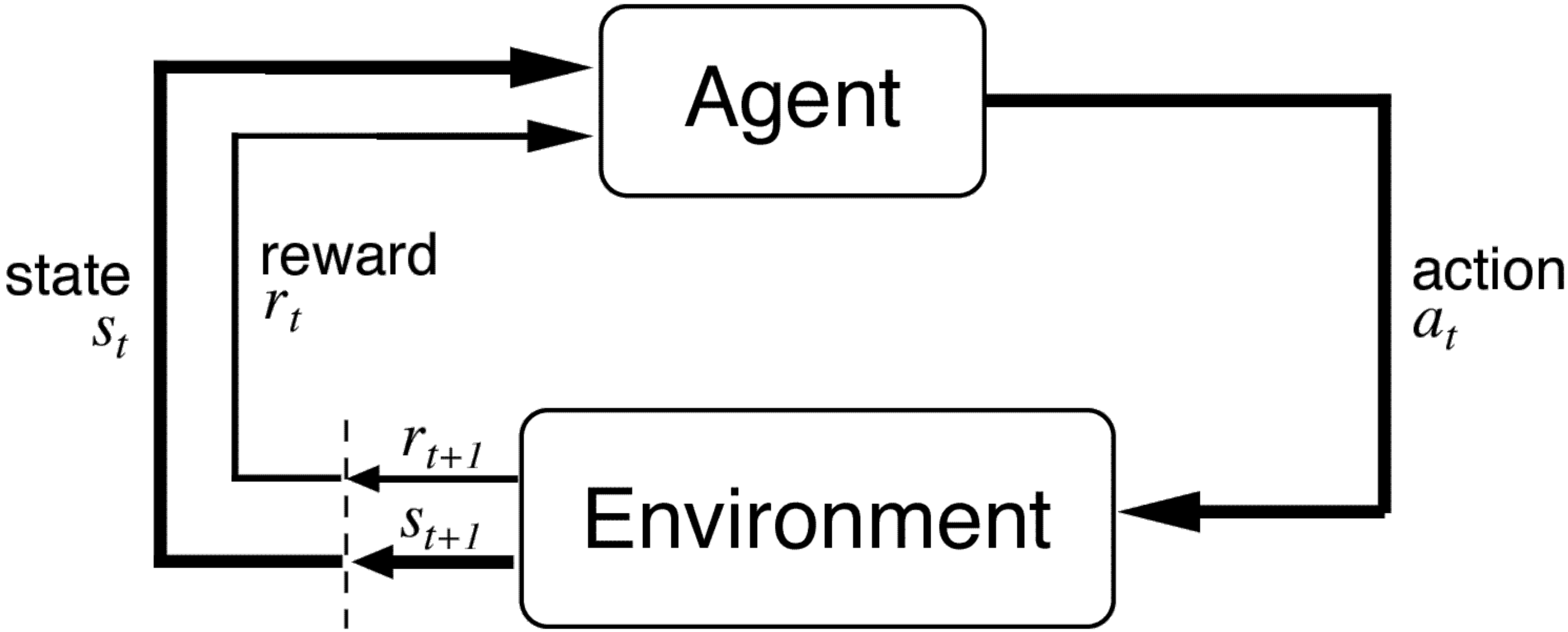
Partial observability

Beyond MDPs

So far, we have assumed we interact with MDP



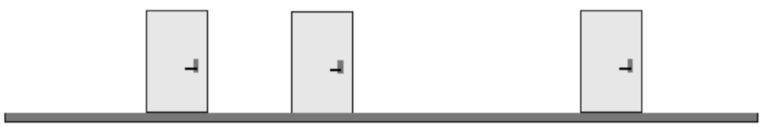
In particular, assumes agent has access to true *state*

Figures: Sutton&Barto, RL:AI

Beyond MDPs

What if we don’t have access to internal state?

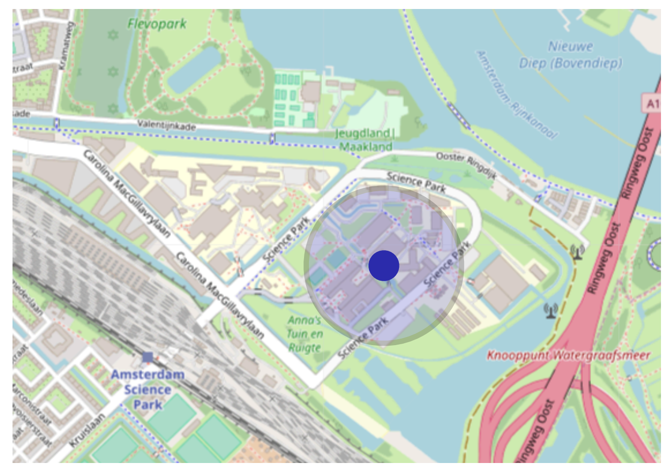
Aliasing



“I am in front of a door -
but I don’t know which one”

“Which direction is the ball going?”

Noise



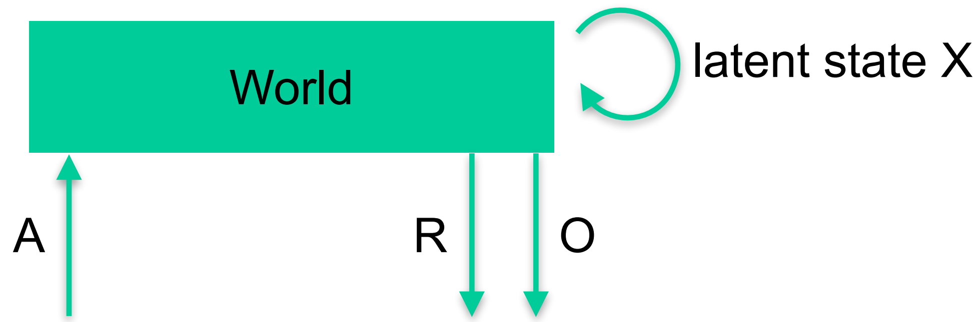
“My GPS tells me where I am,
but it can be off a bit”

openstreetmap.org

Beyond MDPs

The information we get about the state are **observations**

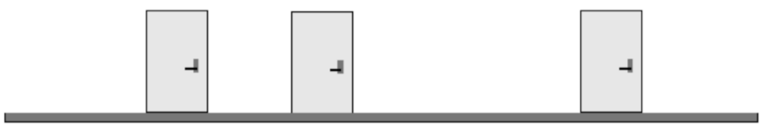
Interaction with world can now be phrased in terms of actions and observations



In addition to rewards r(x',a) and transitions p(x'|a,x) the world now also has an observation function p(o' | a,x')

Beyond MDPs

Aliasing



“I am in front of a door -
but I don’t know which one”

Observation can be seen as feature of latent state

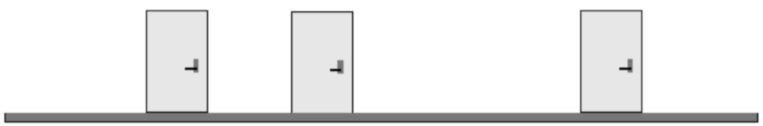
Thus, we could use the observation as if it were a feature, and use the techniques from lecture 5 and 6

But we can do better!

Beyond MDPs

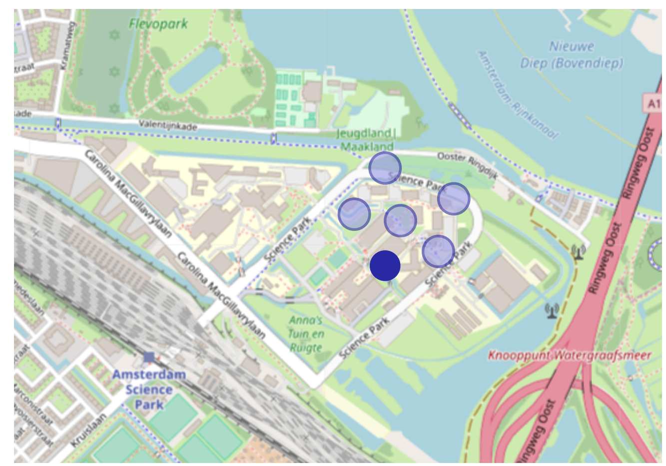
In either case, there is information in the history of interactions

Aliasing



“I was in front of a door, went two steps right, and am again in front of a door”

Noise



“I didn’t move, and these were the measured locations”

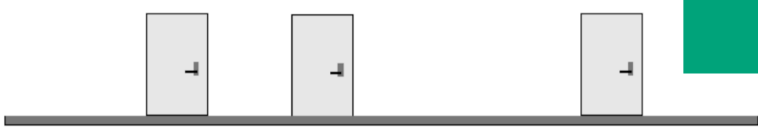
openstreetmap.org

Beyond MDPs

In either case, there is information in the history of interactions

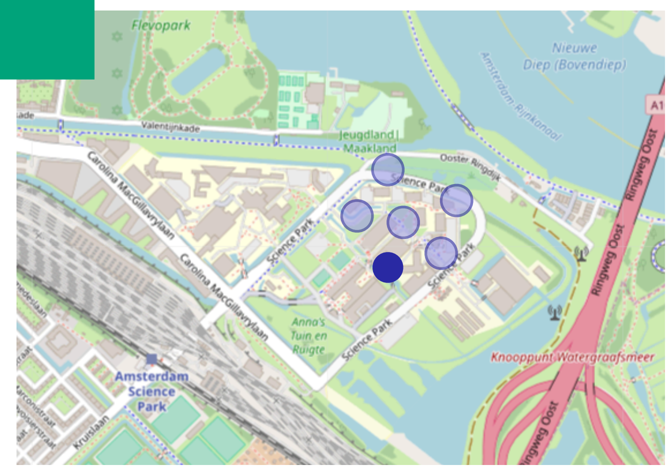
Where am I?

Aliasing



“I was in front of a door, went two steps right, and am again in front of a door”

Noise



“I didn’t move, and these were the measured locations”

openstreetmap.org

Histories and Markov functions

For an optimal policy, should use all information available

$$H_t \doteq A_0, O_1, \ldots, A_{t-1}, O_t$$

So there is an optimal policy of the form

$$A_t = \pi(H_t)$$

but full histories are annoying to work with!

Ht: all the info before time step t. it includes all actions and all observations
we use all previous history to decide the best actions at time t. pie: a func eg
LSTM or neural network.

Histories and Markov functions

here, features of history

we have to satisfy 2 properties:

1) compact: the smaller we can compress the info, the better. eg. in tabular method, small amount of state action pair, is better than infinite state-action pair. because easier to compute.

2) capture all relevant info: we dont want to lose relevant info when compressing info.

question: in which case, the feature on one history can be allowed to = feature of other history.
if Prob (O_t+1| Ht=h, At=a) = Prob (O_t+1| Ht=h', At=a)
then f(h)=f(h')
under both histories, if we get the same distribution of observations (= the predicted observation are the same), then these 2 histories are equivalent. means: then i am willing to map two diff histories to the same description. means: i wont loose any info if i do the mapping.

We can extract *fe*

$$f(H_t)$$

Two desired prop

$f(H_t)$ should be c

$f(H_t)$ should cap

Histories and Markov functions

We can extract *features* from a *history*

$$f(H_t)$$

if the above property is true for every observation and action, then St is called markov state.
St is set of extract feature from history. This state St does not have any ambiguity.

Two desired properties:

$f(H_t)$ should be *compact* (low-d) summary of history

$f(H_t)$ should capture all relevant information…

$$f(h) = f(h') \Rightarrow \Pr\{O_{t+1} = o | H_t = h, A_t = a\} = \Pr\{O_{t+1} = o | H_t = h', A_t = a\}$$

When true ($\forall o, a$), $S_t = f(H_t)$ is a *Markov state*.

Let's look at an example!

Terminology

Note that some terms are used slightly different than so far:

a,r	Same as in MDP
x	Latent 'true' state
o	Observation
s	Internal representation, represents knowledge about x. Used by policy & value function

Terminology

Note that sor are used slightly different than so far:

action and reward.

a,r	Same as in MDP
x	Latent 'true' state
o	Observation
s	Internal representation, represents knowledge about x. Used by policy & value function

which state we are in. s is input of policy and value func

In MDP, these are all the same!

$s = o = x$

if env is at a state , then we observe this state. we give the same state as input to compute policy and value

o is input copy of s
s is a combination of states in the past time steps.

Example: Tiger problem

true location.

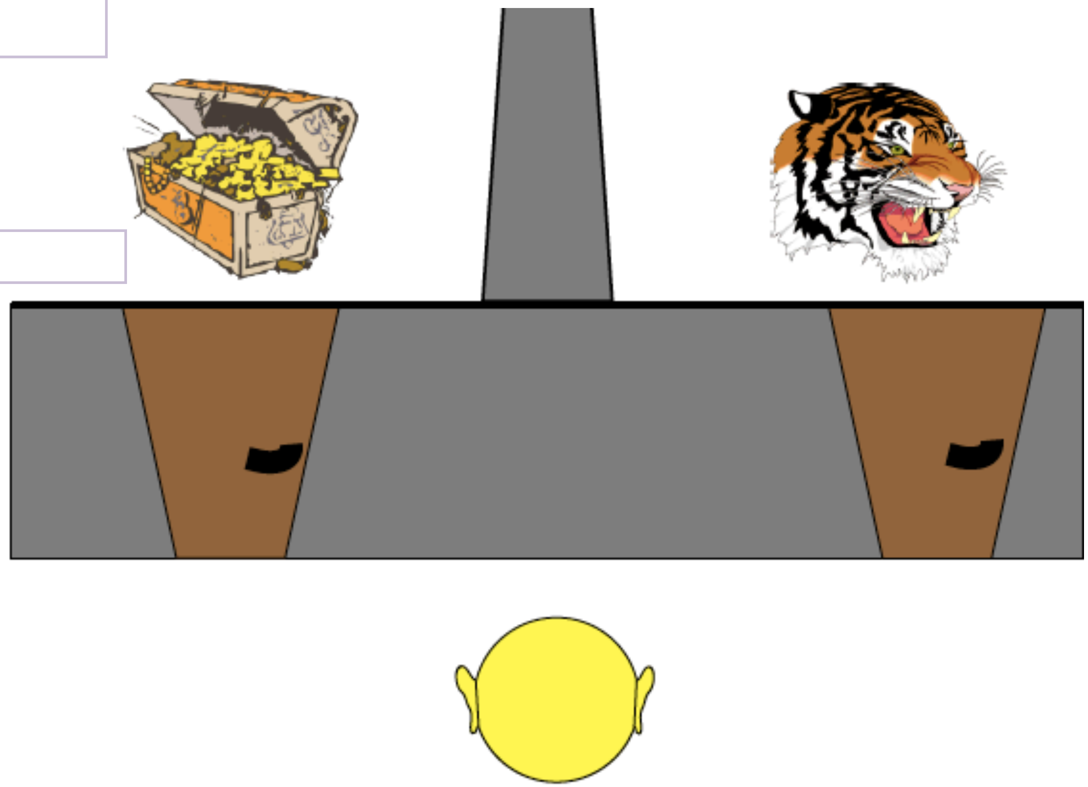
- **States**: left / rig...
  (50% prob.)

- **Actions**: Open left,
  open right, listen

obv

- **Observation**:
  Hear left, Hear right

- **Transitions**: static,
  but opening resets.



- **Rewards**:
  - correct door +10,
  - wrong door -100
  - listen -1

- **Observations** are correct 85% of the time.
  - P(HearLeft | Listen, State=left) = 0.85
  - P(HearRight | Listen, State=left) = 0.15

Thanks F. Oliehoek & S. Whiteson

Hist

assume we give us 2 times to hear the sound. if 1 time we hear tiger sound coming from let, 1 time coming from right, then we are 50% sure that tiger is located behind left door, 50% behind right door. thus we record
[total nb of tiger sound that we hear from left, total nb of tiger sound that we hear from right]

multiple histories have the same internal state:
if we hear 3 times, the following histories have the same internal state = [nb of HL=2, nb of HR=1]
History1= HL, HL, HR
History2= HL, HR, HL
History3= HR, HL, HL
all the histories give the same prob that tiger locates at left door.

another thing: if we hear HL again in time step 4, then [3,1]

In Tige

Only n

Thus, [

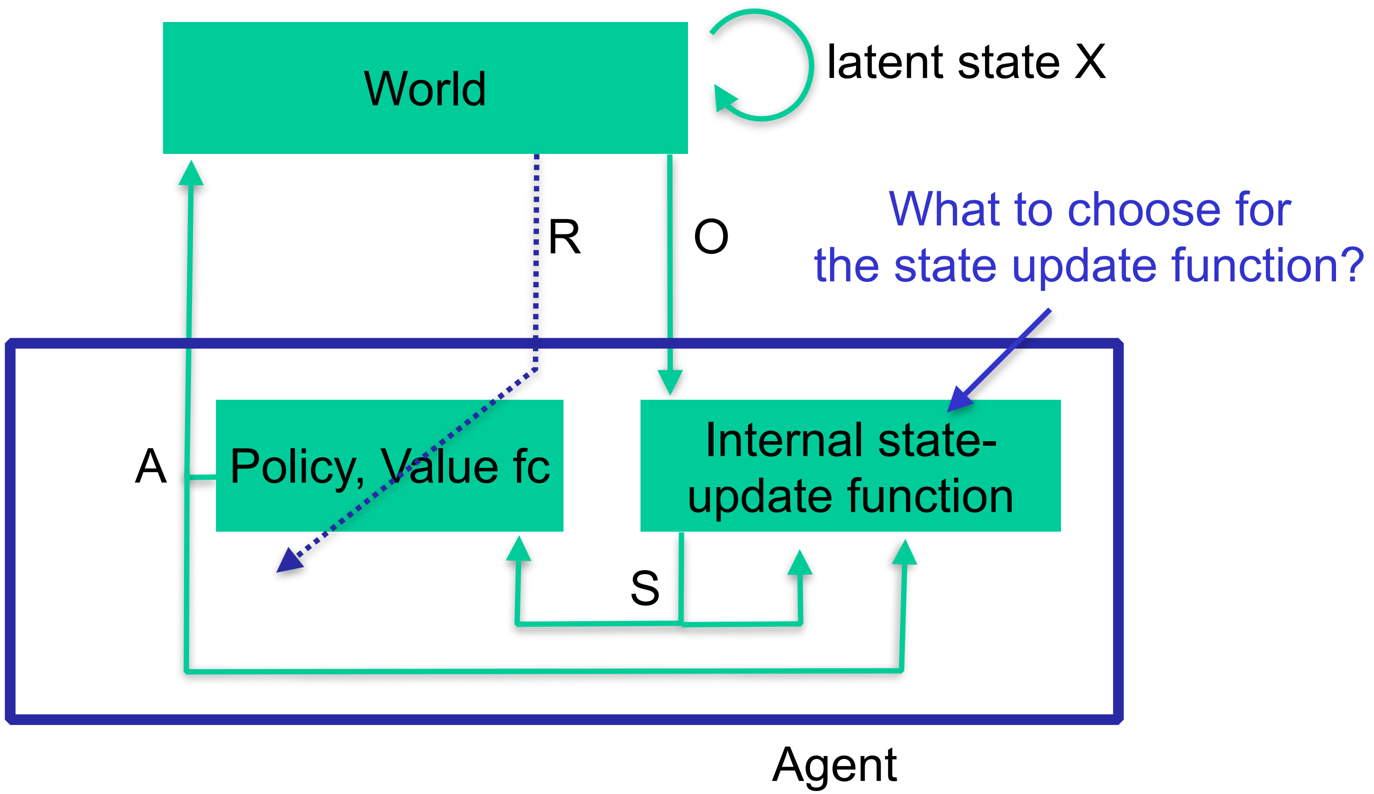
Since a

this is a

Also, when we add an observation, *easy to update* state:
just add to the totals.

Histories and Markov functions

This suggests a way to approach the problem



What kind of internal state to use?

First attempt: internal state S = H?

what if we define internal state S = whole history

this is a markov func, because: only when 2 histories are equivalent, then they will have the same state representation. if 2 histories are diff, then state representations are diff.

Advantages:

- Extremely simple
- Clearly a Markov function

Disadvantages

- Not *compact*, need to remember all of history
- Tabular policy needs to represent all possible

    Impossible in continuing problems

    Inefficient in episodic problems (e.g. HR HL … tiger problem)

when sequence become longer and longer, tabular does not work

history1=HR HL 与 history2 = HL HR have the same state representation.but tabular methods treat these 2 as diff.

What kind of internal state to use?

because 1st attempt fails, so we try a diff method.

Seconc Try to figure out what the latent state X is…

but we're likely never to be sure! (consider tiger problem)

Let's define internal s to contain p(x|h) for all x. Use Bayes' rule

likelihood

prior

$$p(x'|h') = p(x'|o', a, h) = \frac{p(o'|x', a, h)\,p(x'|a, h)}{p(o'|a, h)}$$

evidence

s' is not a guess of true latent state of the world. s' is a prob distri over all possible true latent state of the world.
in tiger case, s'=a vector=[prob (tiger locates at left), prob (tiger locates at right)] given the history. we want to update this vector recursively. we want to use the past internal state and the new external obs, to get new vector prob.

new history h' is splited into 2 parts: obs and action at current time step + all previous history h.

s' is t

What kind of i se?

likelihood: obv is only dependent on true state and what action we take. obv at current time step is indep of h, so remove h

$$p(x'|h') = \frac{p(o'|x',a,\cancel{h})\,p(x'|a,h)}{p(o'|a,h)}$$

x_curve' =for all possible real state.
sum over x_curve' = p (hear tiger from left| true state is tiger stands on left)+ p(hear tiger from left| true state is tiger stands on right)

expand into

x: all possible previous states at time t-1.
x' current state at time t.
p(x|h): prob over the state that we might be in, given history so far. internal state. =s(x)

$$p(x'|h') = \frac{p(o'|x',a)\sum_x p(x'|x,a)p(x|h)}{\sum_{\tilde{x}'} p(o'|\tilde{x}',a)\sum_x p(\tilde{x}'|x,a)p(x|h)}$$

(normalizer)

=s'(x) **observation model** **transition model** **=s(x)** **old belief** ?

all possible real state. this is sum of prob distribution. sum must =1

p(o'|x') = given tiger is on the left, what is prob that we hear tiger is on the left

What kind of internal state to use?

Updating these ... es
known observa...

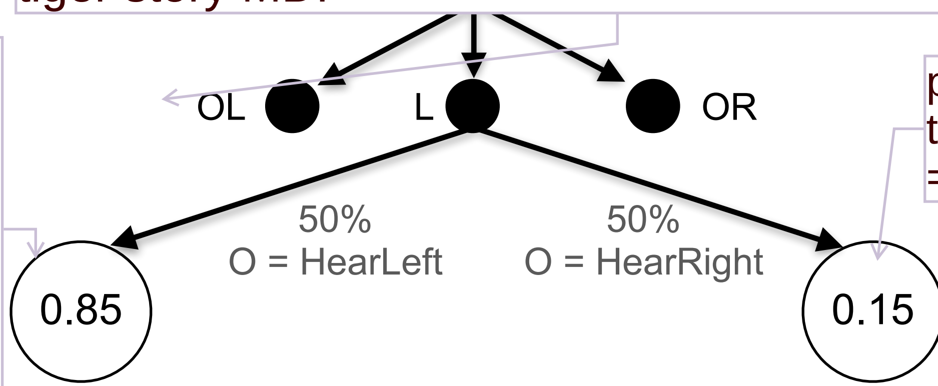
we correctrize the his...
(s) = [prob that tiger a...
because sum =1, so ...
that tiger is on left

G... a
distribution mod...

my belief state says: 50% prob that tiger is on the left. if i take action: open left door, then game is over. if i take action: listen to the tiger sound, then i might hear left, also might hear right. if i hear left, then i will update my belief. means: i increase "tiger is on the left" prob from 0.5 to 0.85. if i hear right, then the prob that "tiger is on the left" goes down, drop from 0.5 to 0.15

transition is interpretated as transit from 0.5 to 0.85. or ... 0.15. then we take action according to the ... we call this : a belief state MDP . it is inside a tiger story MDP

prob that
tiger is on left
=0.5



prob that
tiger is on left
=0.85
or i believe
that the prob
that i hear
tiger on the
left is 85%

P...

prob that
tiger is on left
=0.15

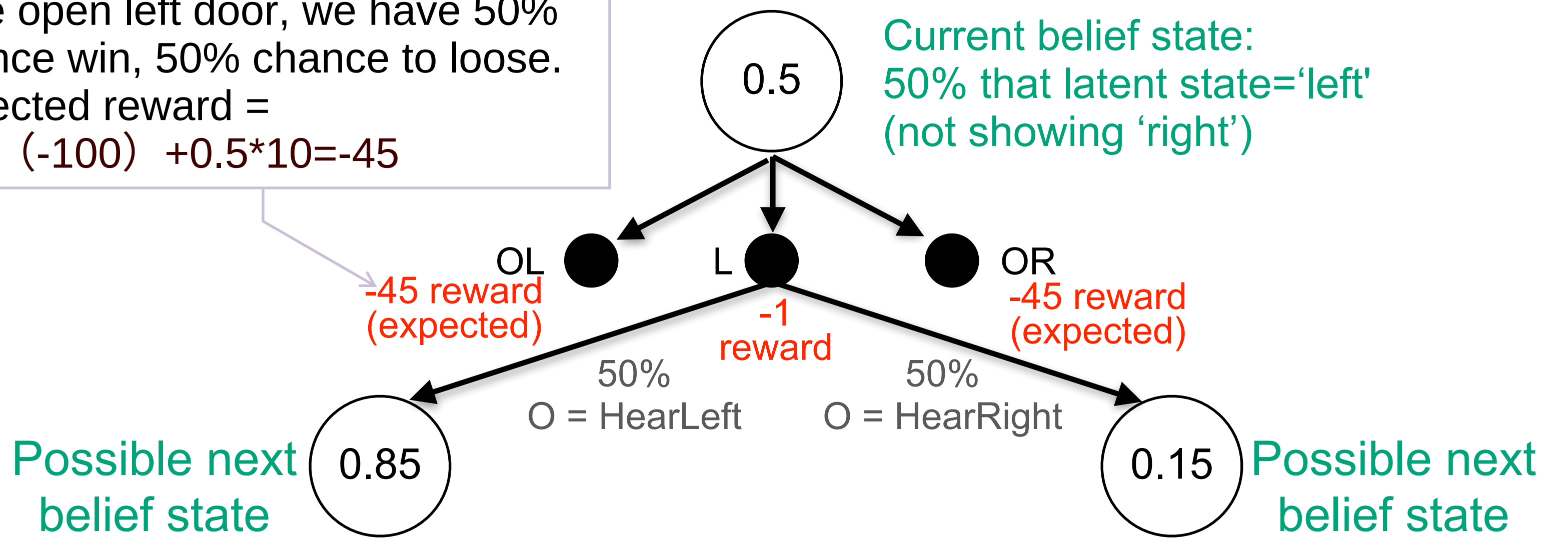
Possible next
belief state

What kind of internal state to use?

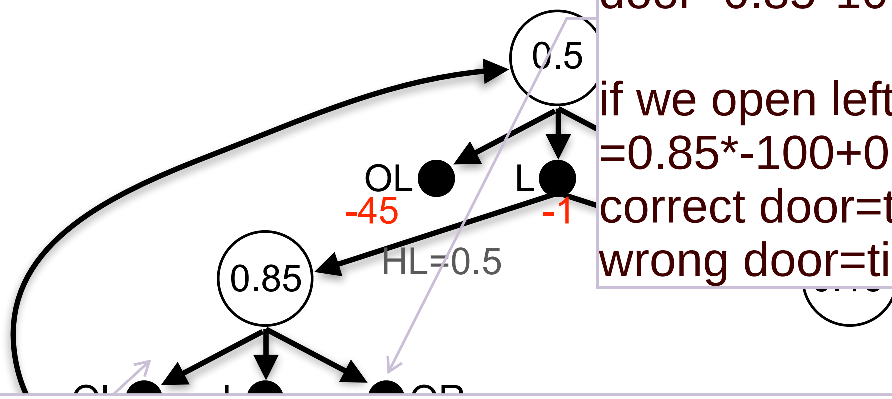
Updating these internal states (called *belief states*) requires known observation and transition model

Given the observation and transition model, we can obtain a *distribution model* of transitions in the belief-space MDP

if we open left door, we have 50% chance win, 50% chance to loose.
expected reward =
0.5*（-100）+0.5*10=-45



Tiger problem & beli



if we listen once, we hear left, we open right door, we have 85% chance get treasure, 15% chance to get tiger.
expected reward = reward of open right door=0.85*10+0.15*-100=-6.5

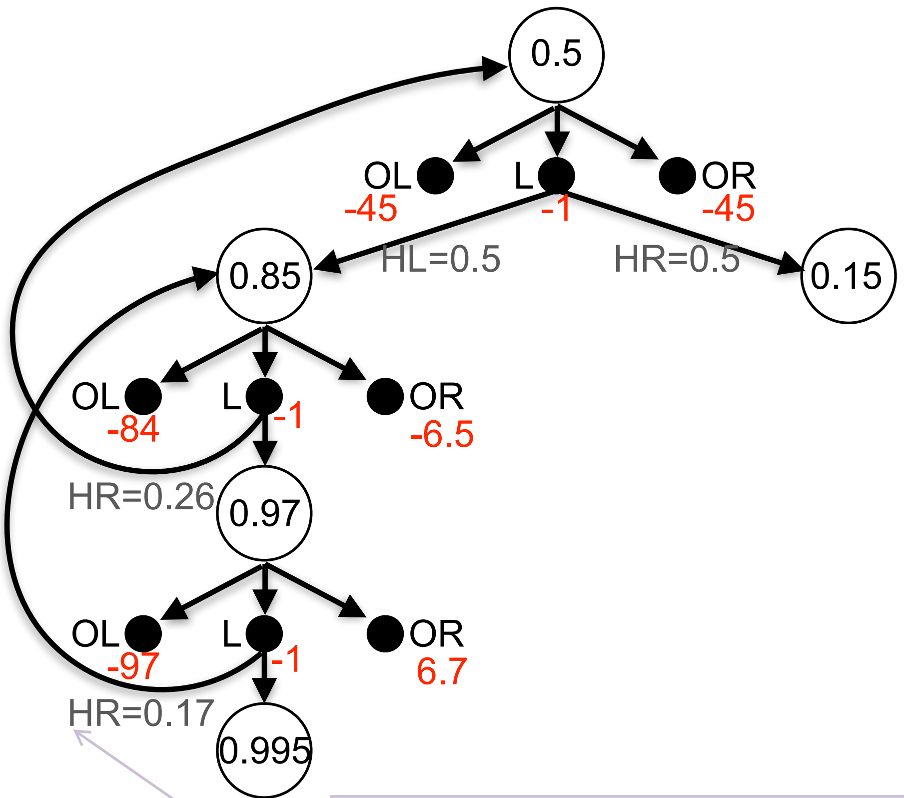
if we open left door, expected reward =0.85*-100+0.15*(10)=83.5=84
correct door=treasure = +10
wrong door=tiger= -100

prob that tiger is truely on right =0.15
if tiger is right, with 85% prob we will correctly hear it from right

HR=the chance that i hear tiger from right = tiger is left & we hear right + tiger is right & we hear right =0.85*0.15+0.15*0.85=0.26

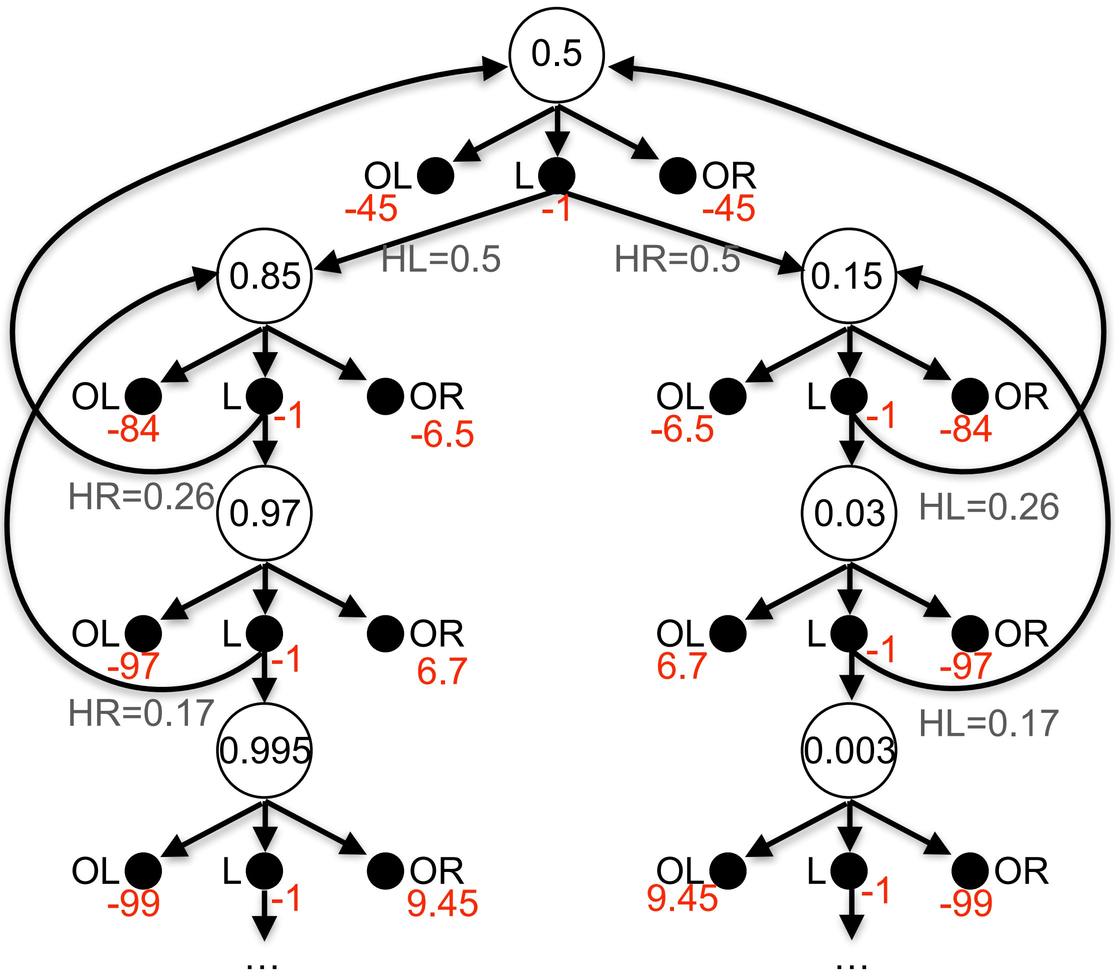
when belief state =0.85, if we chooes to listen, then with prob 0.26 we hear from right, then we get 听到left一次，听到right一次，we will back to belief state =0.5. Or if i hear left, (i have ... prob to hear left), then i am more sure that tiger is left, so i update belief state (=prob that i think tiger is left) to 0.97

Tiger problem & belief states



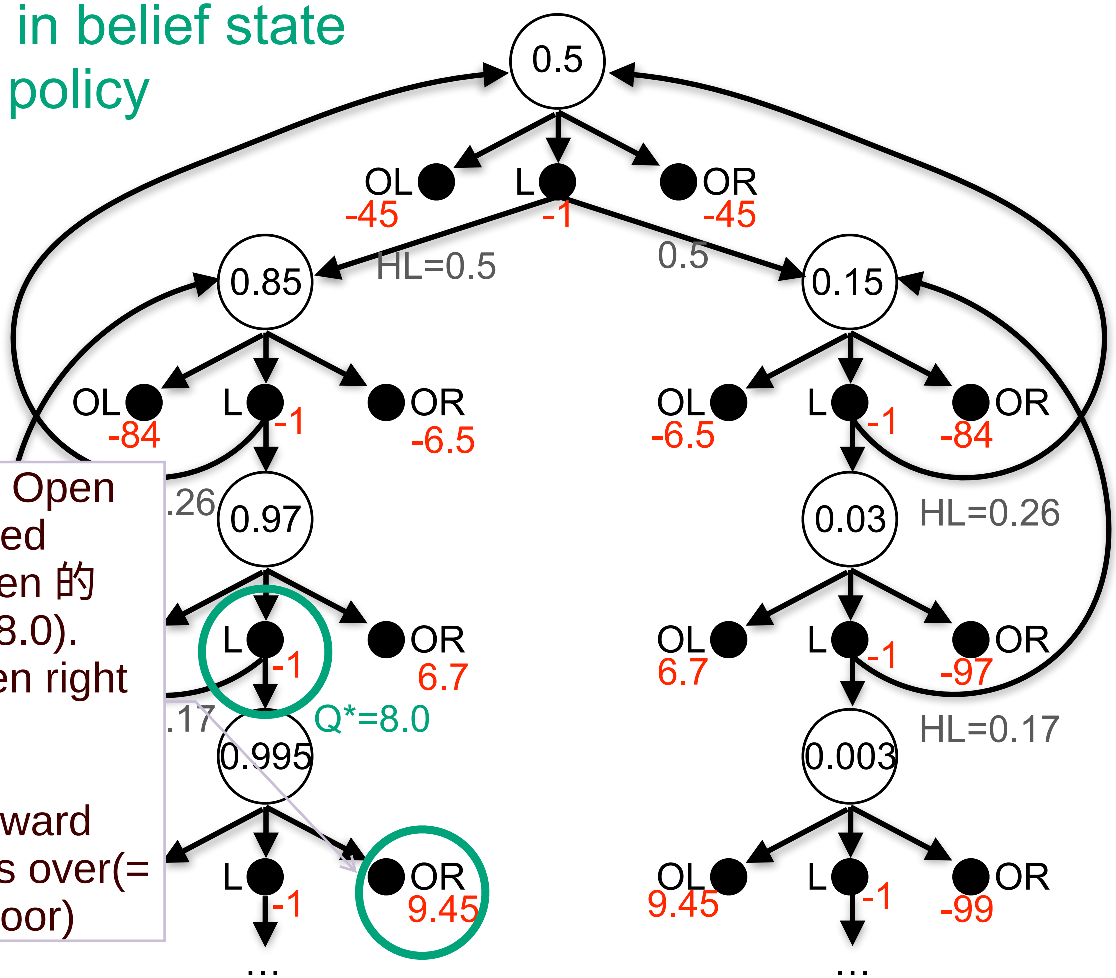
we have 0.17 prob that we hear L, L, R. this is equivalent to hear L only once. so we go back

Tiger problem & belief states



Tiger problem & belief states

Planning (DP) in belief state yields optimal policy + Q function (e.g. value iteration)

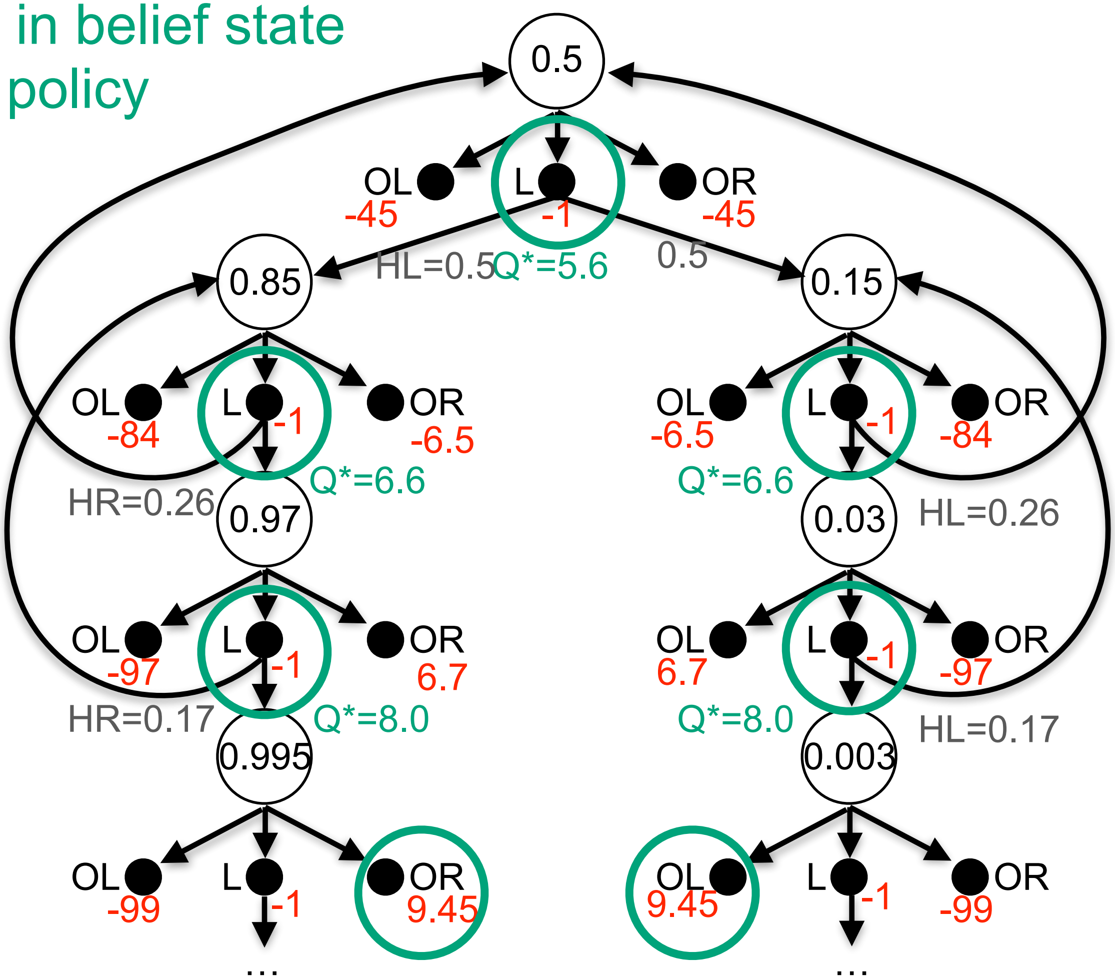


we keep listen, until Open right door 的 expected reward(=9.45) > listen 的 expected reward (=8.0). then we decide: open right door, stop listening

Q func: expected reward from now till game is over(= i decide to open a door)

Tiger problem & belief states

Planning (DP) in belief state yields optimal policy + Q function (e.g. value iteration)



What kind of internal state to use?

This *belief state* approach is the classical approach for *POMDPs* (partially

concrete meaning: the belief state means prob that we think tiger is on left

relatively compact: if there are 10 discrete states, then s is a 10 dim vector.

Advantages:

- Concrete meaning of state as belief (probability) in underlying state
- Relatively compact: s has as many dimensions as x has states
- State can be updated recursively without memorising history

Disadvantage:

- Underlying models are needed!
- Underlying model difficult to learn…
- Only for discrete state spaces

i only have a sequence of actions and obv, it is difficult to know the exact true state.

What kind of internal state to use?

Third attempt: predictions. Remember

$$f(h) = f(h') \Rightarrow \Pr\{O_{t+1} = o | H_t = h, A_t = a\} = \Pr\{O_{t+1} = o | H_t = h', A_t = a\}$$

Define internal state as probability of next observation?

$$f(h) = \begin{bmatrix} f_{o_1 a_1}(h) \\ f_{o_2 a_1}(h) \\ \vdots \\ f_{o_1 a_2}(h) \\ \vdots \end{bmatrix}$$

here, Prob (hear left| History=h, A = 'listen')
previous, Belief State = Prob(Tiger is on left |History=h)

$$f_{oa}(h) := \Pr\{O_{t+1} = o | H_t = h, A_t = a\}$$

‘by definition’ this fulfils the Markov criterion above

What kind of internal state to use?

We can also consider longer tests, e.g.

$$\tau = a_1 o_1 a_2 o_2 a_3 o_3$$

and define the probability that a test “succeeds”:

$$p(\tau|h) \doteq \Pr\{O_{t+1} = o_1, O_{t+2} = o_2, O_{t+3} = o_3 | H_t = h, A_t = a_1, A_{t+1} = a_2, A_{t+2} = a_3\}$$

prob of these 3 obs, given the 3 actions and h

It can be proven that for special sets of ‘core tests’ τ_1, τ_2, …τ_d the vector [p(τ_1|h), p(τ_2|h), … , p(τ_d|h)] is a Markov state

this is more compact

What kind of internal state to use?

These are called *predictive state representations*

Example:

- In the tiger problem, if we don't know $p(o\,|\,x)$, we cannot calculate the belief state (In other problems, I might never know what x was)
- However, all information can be captured by two 'tests' (or even one)

  $p(\text{HL}\,|\,h, L)$

  $p(\text{HR}\,|\,h, L)$
- These probabilities can be learned from data
  (e.g. naively with a LSTM classifier, but there are smarter ways…)

What kind of internal state to use?

These are called *predictive state representations*

Advantages:

- Test probabilities learnable from data
- As compact or more so than belief states
- Can still be updated recursively

Disadvantage

- Still limited to ‘tabular’ setting (but there are extensions)

Back to approximations

Alternative: re-introduce approximation

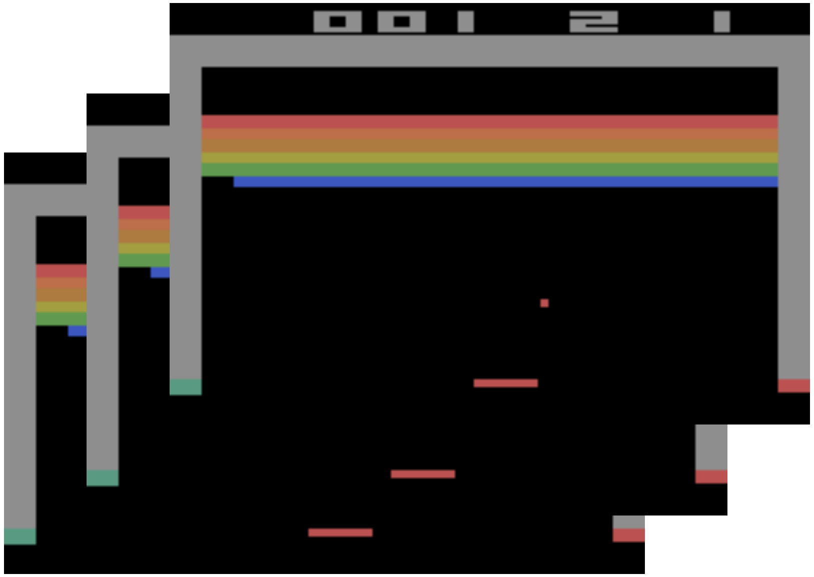
Use non-Markov state:

- Use last observation as internal state?
  S = O
- Better: Use k most recent observations (+actions?) as internal state
  $S=(O_{t-k}\ A_{t-k}\ \ldots\ O_{t-1}\ A_{t-1}\ O_t)$

Can be seen as ‘features’ from the history

Back to approximations

Example: Frame stacking from Atari paper (Mnih et al., 2013)



O_t
O_{t-1}
O_{t-2}

Wikipedia

Back to approximations

k most recent observations and/or actions can be used as internal state

Advantages

- Very simple to define and use

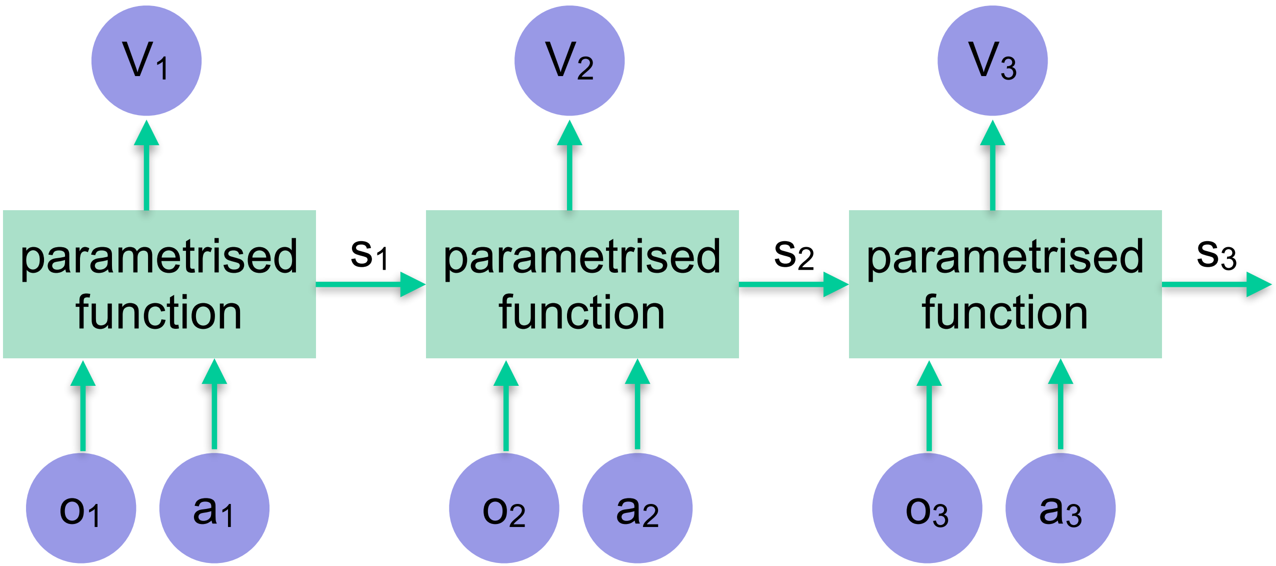
Disadvantage

- Could be very suboptimal if we need a memory of more than k steps
- Potentially not very compact
- Potentially Non-Markov

Back to approximations

One insight of DQN was that features can be learned -

Learn end-to-end in partial observable settings?

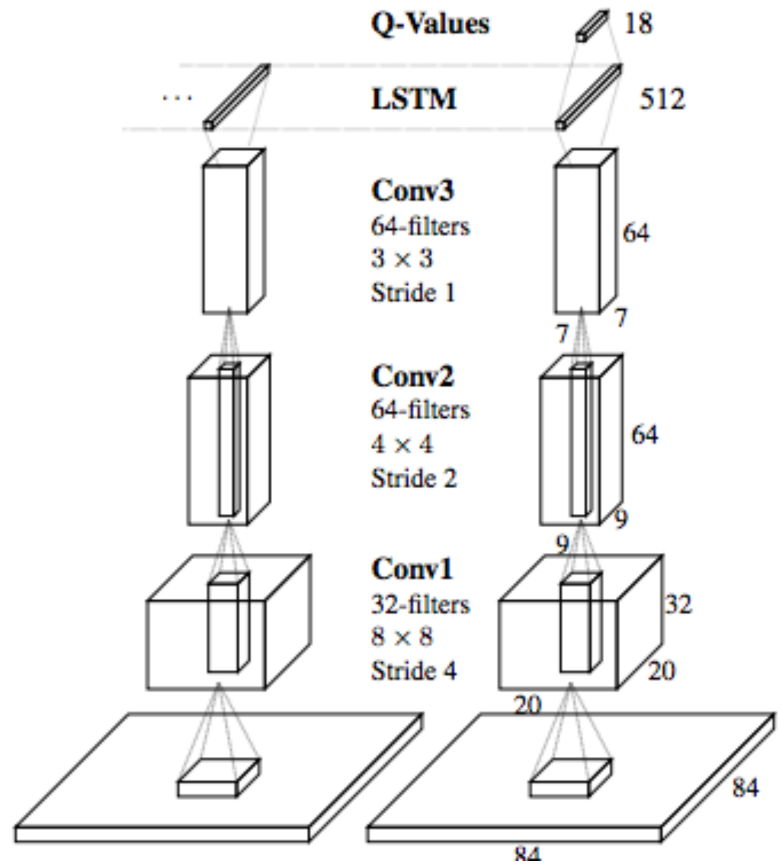


Back to approximations

Example: Deep Recurrent Q-Learning

Combination of Conv-layers and recurrent (LSTM) layers

Trained using prediction loss on target Q-network



Hausknecht & Stone, Deep Recurrent Q-learning for partially observable MDPs, AAAI 2015 fall symposium

Back to approximations

End-to-end learned states

Advantages

- Conceptually simple & ties in to DL methodology
- Compared to stacking, no fixed ‘k’, s can depend on full history
- Can adjust compactness (up to a point…)

Disadvantage

- RNN learning can be tricky in practice (local optima, train time)
- Potentially Non-Markov

Comparison

Exact methods

- **Full history**
  Not compact…
- **Belief state**
  Easy to interpret
  Requires known model
  (tricky to learn from data)
- **Predictive state**
  Model learnable from data
  Most compact

Approximate methods

- **Recent observation(s)**
  Easy
  Lose long-term dependencies
- **End-to-end learning**
  Quite general
  RNN learning can be tricky,
  requires much data…

Conclusion

Partial observable MDP’s do not have all relevant information from history in the observations

Thus, an internal state has to be extracted from the history

Trade-off between various factors:

- Compactness
- Markov property
- Interpretability
- Computational complexity of updates, learning
- Ease of implementation

What you should know

What is a state update function and why do we need it?

What are the advantages and disadvantages of the discussed state update functions?