RL 8: Value Iteration and Policy Iteration

Michael Herrmann

University of Edinburgh, School of Informatics

07/02/2014

Last time: Eligibility traces: TD(λ)

Determine the δ error:

$$\delta_{t+1} = r_t + \gamma \hat{V}_t(s_{t+1}) - \hat{V}_t(s_t)$$

Update all states the agent has visited recently:

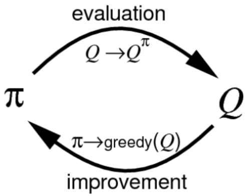
$$\hat{V}_{t+1}(s) = \hat{V}_t(s) + \eta \delta_{t+1} e_{t+1}(s)$$

Update the eligibility traces to make sure that the update becomes weaker and weaker since the state has been visited the last time:

$$e_{t+1}(s) = \begin{cases} 1 + \gamma \lambda e_t(s) & \text{if } s = s_t \\ \gamma \lambda e_t(s) & \text{if } s \neq s_t \end{cases}$$

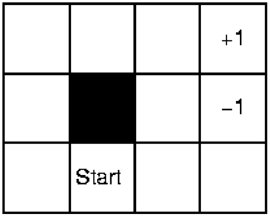
Parameters:

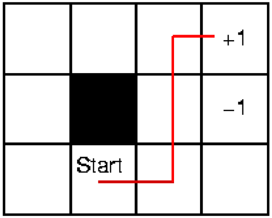
- $\gamma \lesssim 1$ discount factor
- $\eta \gtrsim 0$ learning rate
- $\lambda \lesssim 1$ trace decay parameter



Example from Stachnis/Burgard

A robot in a small grid world with “reflecting walls

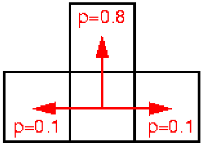


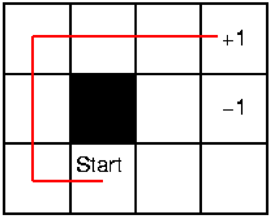


Reward:
$r = \pm 1$ as shown or
$r = -0.04$ per step

Optimal path from
Start to Goal
avoiding the trap.

Same example in a non-deterministic world





Robot does not exactly perform the desired action

If cost per step is low it pays to choose a safer path

The optimal policy in a non-deterministic problem

Optimal policy

$$\pi^*(i) = \arg\max_a \sum_j M_{ij}^a V(j)$$

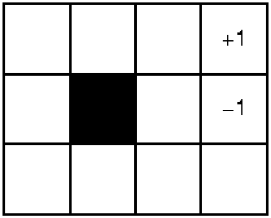
M_{ij}^a: Probability of reaching state j form state i with action a.

$V(j)$: Value of state j.

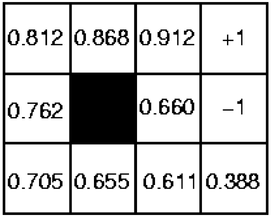
Given the value function and the transition probabilities (Markovian!), we can easily calculate the optimal policy

We know already a way to approximate the value function.

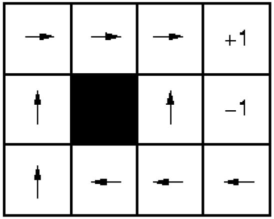
Value Iteration and Optimal Policy



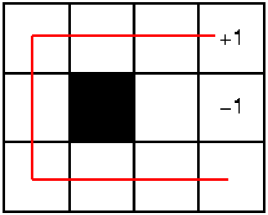
1. Given environment



2. Calculate state values



3. Extract optimal policy



4. Execute actions

Optimality

Note that state (3,2) has higher value than (2,3), but policy of (3,3) points to (2,3).

Policy is not the gradient of the value function!

$$\pi^* (i) = \arg\max_a \sum_j M_{ij}^a V(j)$$

Policy converges faster than the values of the value function.

For the policy to converge it is sufficient that the ***relations*** between the values are correct.

Can the we compute the optimal policy in a faster way?

Policy Iteration

```
function PolicyIteration($M$,r)
  do
    $V \leftarrow$ValueDetermination($\pi$, $V$, $M$, r)
    converged $\leftarrow$ TRUE
    for all state $i$ do
      if $\max_a \sum_j M_{ij}^a V(j) > \sum_j M_{ij}^{\pi(i)} V(i)$
      then $P(i) \leftarrow \arg\max_a \sum_j M_{ij}^a V(j)$
      converged $\leftarrow$ FALSE
    end
  while not converged
return $\pi$
```

Policy iteration with MC

A variant of the previous

- Initialise $\mathcal{Q}(s,a)$, $\pi(s)$ arbitrary $\forall s,a$; $R_{\text{List}}(s,a)$ empty list
- Repeat
    - Generate an episode using exploring starts and π
    - for each pair s, a in the episode
        - $R :=$ return following the first occurrence of s and a
        - Append R to $R_{\text{List}}(s,a)$
        - $\mathcal{Q}(s,a) :=$ average over $R_{\text{List}}(s,a)$
    - $\forall s$ in the episode: $\pi(s) := \arg\max_a \mathcal{Q}(s,a)$

Remarks

- Value determination possible by iteration, but sometimes also by direct computation:

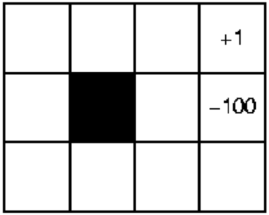
$$V(i) = r(i) + \gamma \sum_j M_{ij}^{\pi(i)} V(i) \quad \forall i \in S$$

  is a system of linear equations with dimension $|S|$:

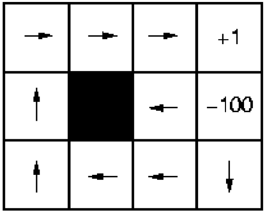
$$V = (I - \gamma M^{\pi})^{-1} r$$

  The system does not necessarily have a solution. Theory, therefore, often assumes $\gamma < 1$.
- For real-time applications even MDPs are hard to compute: Try deterministic approximations.
- Value iteration:
  Try to get a better evaluation, but use the best available policy
- Policy iteration:
  Try to get a better policy, using the currently best evaluation

Value iteration vs. Policy Iteration: Example



Problem



Solution (policy iteration)

Given the (0.1, 0.8, 0.1)-probability model it is optimal to try to move from (4,3) and (3,2) by bumping to the walls.

Then, entering the trap at (4,2) has probability 0.

Convergence

- Policy improvement still works if evaluation is done with MC

$$\begin{aligned} Q^{\pi_k}\left(s, \pi_{k+1}(s)\right) &= Q^{\pi_k}\left(s, \arg\max_a Q^{\pi_k}(s,a)\right) \\ &= \max_a Q^{\pi_k}(s,a) \\ &\geq Q^{\pi_k}\left(s, \pi_k(s)\right) \\ &= V^{\pi_k}(s) \end{aligned}$$

- The expected reward for π_{k+1} not worse than π_k
- We have to assume that the value function has stabilised, i.e. an infinite number of episodes

Conclusions

“Therefore in practice, value iteration should never be used. Implementation of modified policy iteration requires little additional programming effort yet attains superior convergence". Puterman, 1994

The policy iteration is polynomial in the problems size, but Leslie Pack Kaelbling remarks that: “In the worst case the number of iterations grows polynomially in $(1-\gamma)^{-1}$, so the convergence rate slows considerably as the discount factor approaches 1.”

(M. L. Littman, T. L. Dean, and L. P. Kaelbling. On the complexity of solving Markov decision problems. Proc. 11th Ann. Conf. on Uncertainty in AI, 1995.)

Philosophical arguments: E. B. Baum asks in *What is Thought?* (MIT, 2004) “What it fundamentally wrong with value iteration?”

Yet, value iteration is a straight-forward generalisation of the deterministic case. It may be more robust in dynamic problems, for higher uncertainty, or strong randomness.

Combined Value-Policy Iteration
E. Pashenkova (1996)

1. Perform Value Iteration and compute policy at each step of VI
2. IF no change in policy on two successive steps, fix the policy and perform one step of Policy Iteration:
    1. Value Determination finding precise values for the fixed policy;
    2. policy evaluation
    3. IF no change in policy, return it as an optimal policy, ELSE go to 1.

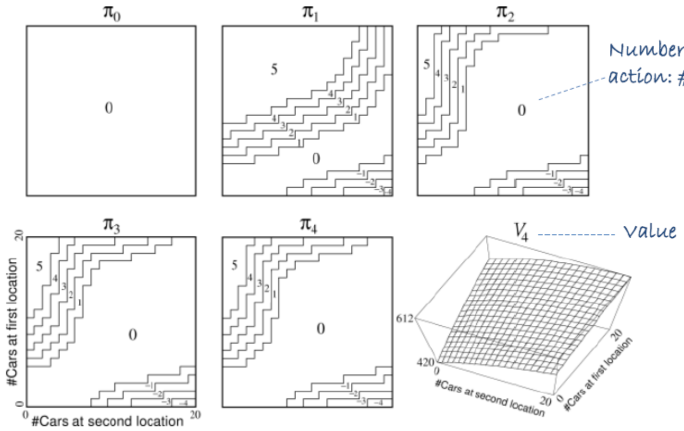
Example: Jack's car rental (4.2 in S+B)

- \$10 for each car rented (must be available when request received)
- Two locations, maximum of 20 cars at each
- Cars returned and requested randomly
  - Poisson distribution, n returns/requests with probability $\frac{\lambda^n}{n!}e^{-\lambda}$
  - Location 1: Average requests = Average returns = 3
  - Location 2: Average requests = 4, Average Returns = 2
- Can move up to 5 cars between locations overnight (costs \$2 each)

Problem setup:

- States, actions, rewards?
- Transition probabilities?

Jack's car rental: Solution



Jack's car rental: Discussion

Difficult to adapt the solution to different conditions, e.g.

- Suppose first car moved is free but all other transfer cost \$2
    - From location 1 to location 2 (not other direction!)
    - Because an employee would anyway go in that direction, by bus
- Suppose only 10 cars can be parked for free at each location
    - More than 10 incur fixed cost of \$4 for using an extra parking lot

For more information see: cns.upf.edu/dani/materials/jack.pdf

Many slides are adapted from web resources associated with Sutton and Barto's Reinforcement Learning book

. . . before being used by Dr. Subramanian Ramamoorthy in this course in the last three years.

Please check again the more formal considerations in the book *Algorithms for Reinforcement Learning* by C. Szepesvari, Chapters 2.1 and 4.1.

The first example today was adapted form " Autonomous Mobile Systems: The Markov Decision Problem Value Iteration and Policy Iteration by Cyrill Stachniss and Wolfram Burgard. These authors acknowledge: Russell & Norvig: AI – A Modern Approach (Chapter 17, pages 498ff)-

Reinforcement learning (just a few general remarks)

Inspired by behaviorist psychology, reinforcement learning is an area of machine learning in computer science, concerned with how an agent ought to take actions in an environment so as to maximize some notion of cumulative reward

en.wikipedia.org/wiki/Reinforcement_learning

- Choose actions to move to states which are as good as possible
- Quality of states is measured by the expected future discounted reward
- Expectation is taken w.r.t. to a fixed policy

Law of Effect: "Responses that produce a satisfying effect in a particular situation become more likely to occur again in that situation, and responses that produce a discomforting effect become less likely to occur again in that situation."
(Gray, Peter. ''Psychology", Worth, NY. 6th ed. pp 108–109, from wikipedia)

Historical roots: The law of effect

"Connectionism" (E. Thorndike, 1911):

- "satisfying state of affairs" leads to reinforcement of the association between action and this state
- "annoying state of affairs" leads to weakening of the association between action and this state

Remarks:

- Consequences of behaviour determine what is learnt and what is not
- Thorndike introduced animal studies for verifying predictions made from his theory. He also was among the first to apply psychological principles in the area of teaching (*active learning*)
- *Connectionism* implies modelling of higher brain functions as the emergent processes of interconnected networks of simple units. Thorndike provided the first working model.

http://www.lifecircles-inc.com/Learningtheories/behaviorism/Thorndike.html

Behaviourism

has been disparaged for focusing exclusively on behaviour, refusing to consider what was going on inside the head of the subject.

- RL shares with behaviourism
  - its origins in animal learning theory
  - its focus on the interface with the environment
  - states and actions (or: stimuli and responses)
  - the idea that *representations* aren't needed to define optimality
- In the end it all comes down to the actions taken and the states perceived.
- RL *of course* is all about the algorithms and processes going on inside the agent.
- For example, RL (in ML) often considers the construction of internal models of the environment within the agent, which is *far* outside the scope of behaviourism

adapted from http://webdocs.cs.ualberta.ca/~sutton/RL-FAQ.html#behaviorism, emphasis changed

Learning

Psychology

- Non-associative learning: single stimulus (habituation or sensitisation)
- Associative learning
  - two stimuli (classical conditioning):
    A neutral stimulus causes a response. After learning, a conditioned stimulus causes a similar response (unsupervised or Pavlovian learning)
  - stimulus-response (operant conditioning, reinforcement learning)

Machine learning

- Unsupervised learning
- Supervised learning
- Reinforcement learning

Classical condition: Rescorla & Wagner (1972)

Two assumptions:

- learning is driven by error (formalise notion of surprise)
- summations of predictors is linear

Change in value is proportional to the difference between actual and predicted outcome

$$\Delta V_X^{n+1} = \alpha_X \beta (\lambda - V_{\text{tot}})$$

ΔV_X change in the strength of association of (CS) X

$$V_X^{n+1} = V_X^n + \Delta V_X^{n+1}$$

$\alpha_X \in [0, 1]$ salience of the CS, $\beta \in [0, 1]$ rate parameter for the US

λ is the maximum conditioning possible for the US

V_{tot} is the total associative strength of all CS

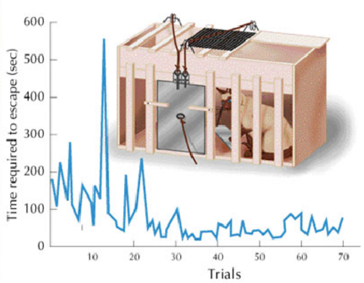
en.wikipedia.org/wiki/Rescorla%E2%80%93Wagner_model, www.princeton.edu/~yael/

Instrumental conditioning

Thorndike "Animal intelligence: an experimental study of the associative processes in animals" (PhD thesis)

Tested hungry cats in "puzzle boxes"

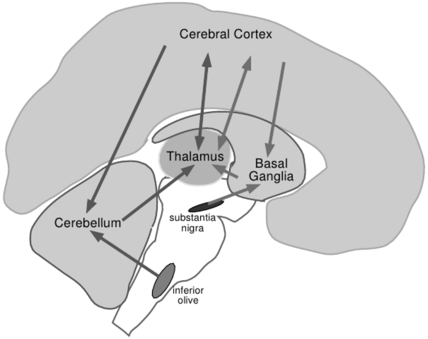
Definition for learning: Time to escape

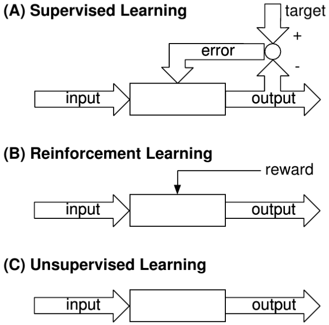


Gradual learning curves, did not look like 'insight' but rather trial and error

www.princeton.edu/~yael/

Learning paradigms and the brain





- Cerebellum: Supervised learning
- Basal ganglia: Reinforcement learning
- Cerebral cortex: Unsupervised learning

Doya, Kenji. What are the computations of the cerebellum, the basal ganglia and the cerebral cortex?. Neural networks 12.7 (1999): 961-974.