

Q-learning for history-based reinforcement learning

Mayank Daswani, Peter Sunehag,
Marcus Hutter

Research School of Computer Science
CECS

15th Nov 2013



Outline

Introduction

Background

- Traditional RL

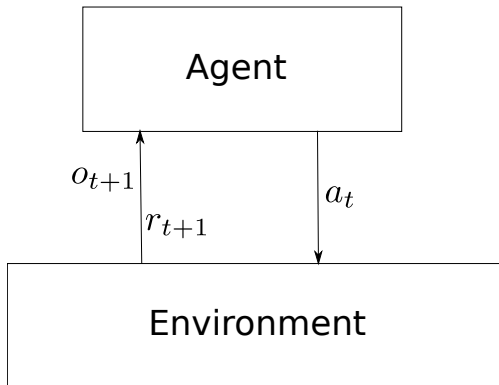
- Feature Reinforcement Learning

Model-free Cost

Experiments

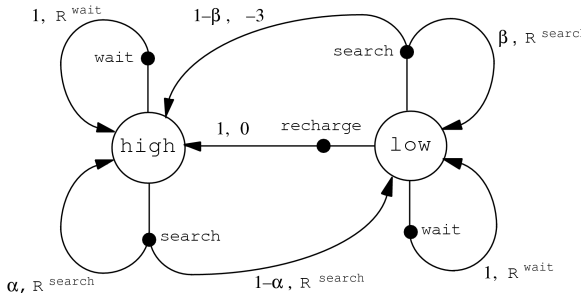
Conclusion

The general RL problem



An agent acts in an unknown environment and receives observations and rewards in *cycles*. The agent's task is to act so as to receive as much reward as possible.

Traditional RL



Source : Reinforcement Learning : Sutton and Barto.

In traditional reinforcement learning, the environment is considered to be a Markov Decision Process (MDP).

Traditional RL

Given an MDP representation a value function can be defined which says how good it is to be in a particular state. Formally, a (action) value function is the expected future discounted reward sum i.e.

$$Q^\pi(s, a) = E_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s, a_t = a \right]$$

where π is the current policy. The Bellman equation tells us that this is in fact

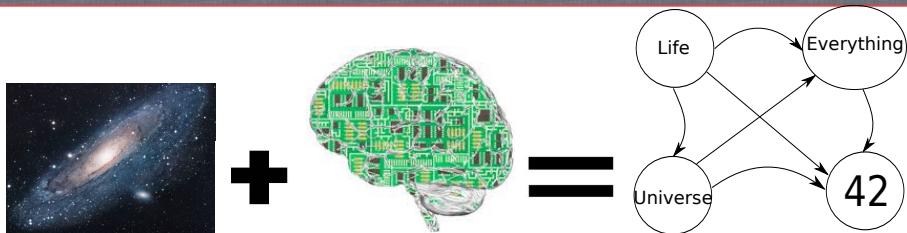
$$Q^\pi(s, a) = \sum_a \pi(s, a) \sum_{s'} \mathcal{P}_{ss'}^a \left[\mathcal{R}_{ss'}^a + \gamma \sum_{a'} \pi(s', a') Q^\pi(s', a') \right]$$

Model-based versus model-free RL

There are two broad approaches to solving unknown MDPs.

- Model-based RL approximates the (unknown) transition probabilities and reward distribution of the MDP.
- Model-free RL attempts to directly estimate the value function itself.

Feature Reinforcement Learning



Feature RL aims to automatically reduce a complex real-world problem to a useful (computationally tractable) representation (MDP).

Formally we create a map ϕ from an agent's history to an MDP state. ϕ is then a function that produces a relevant summary of the history.

$$\phi(h_t) = s_t$$

Feature Markov Decision Process (Φ MDP)

In order to select the best ϕ , we need a cost function and a way to search over the space containing ϕ .

The original cost proposed is,

$$Cost(\phi|h) = CL(s_{1:n}^\phi|a_{1:n}) + CL(r_{1:n}|s_{1:n}^\phi, a_{1:n}) + CL(\phi)$$

In order to calculate these code lengths we need to have the transition and reward counts, effectively the *model* for the MDP.

Φ MDP : Choosing the right ϕ

- A global stochastic search (e.g. simulated annealing) is used to find the ϕ with minimal cost.
- Traditional RL methods can then be used to find the optimal policy given the minimal ϕ .

Algorithm 1: A high-level view of the generic Φ MDP algorithm.

Input : Environment $Env()$;

Initialise ϕ ;

Initialise history with observations and rewards from $t = init_history$
random actions;

Initialise M to be the number of timesteps per epoch;

while true do

$\phi = SimulAnneal(\phi, h_t)$;

$s_{1:t} = \phi(h_t)$;

$\pi = FindPolicy(s_{1:t}, r_{1:t}, a_{1:t-1})$;

for $i = 1, 2, 3, \dots M$ **do**

$a_t \leftarrow \pi(s_t)$;

$o_{t+1}, r_{t+1} \leftarrow Env(h_t, a_t)$;

$h_{t+1} \leftarrow h_t a_t o_{t+1} r_{t+1}$;

$t \leftarrow t + 1$;

end

end

Motivation

- Scale the feature reinforcement learning framework to deal with large environments using function approximation.



Scaling up Feature RL

Following the model-based and model-free dichotomy, there are two ways to scale up feature RL.

- In the model-based case, we can search for factored MDPs instead. This involves an additional search over the temporal structure of the factored MDP.
- In the model-free case, we can use function approximation. But first we need a model-free cost!

Q-learning

A particular model-free method is Q-learning. It is an off-policy, temporal difference method that converges asymptotically under some mild assumptions.

It uses the update rule

$$Q(s, a) \leftarrow Q(s, a) + \alpha_t \Delta_t$$

where Δ_t is the temporal difference

$$\Delta_t = r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)$$

Q-learning Cost

We can define a cost based on the Q-learning error over the history so far,

$$Cost_{QL}(Q) = \frac{1}{2} \sum_{t=1}^n (\Delta_t)^2$$

This is similar to the loss used for regularised least-squares fitted Q-iteration. Now we can extend this cost to the history-based setting.

Q-learning in history-based RL

We can use the cost to find a suitable map $\phi : \mathcal{H} \rightarrow \mathcal{S}$ by selecting ϕ to minimise the following cost,

$$\begin{aligned} \text{Cost}_{QL}(\phi) = \min_Q & \frac{1}{2} \sum_{t=1}^n (r_{t+1} + \gamma \max_a Q(\phi(h_{t+1}), a) - Q(\phi(h_t), a_t))^2 \\ & + \text{Reg}(\phi) \end{aligned}$$

Extension to linear FA

This cost also easily extends to the linear function approximation case where we approximate $Q(h_t, a_t)$ by $\xi(h_t, a_t)^T \mathbf{w}$ where $\xi : \mathcal{H} \times \mathcal{A} \rightarrow \mathbb{R}^k$ for some $k \in \mathbb{R}$.

$$\text{Cost}_{QL}(\xi) = \min_{\mathbf{w}} \frac{1}{2} \sum_{t=1}^n \left(r_{t+1} + \gamma \max_a \xi(h_{t+1}, a)^T \mathbf{w} - \xi(h_t, a_t)^T \mathbf{w} \right)^2 + \text{Reg}(\xi)$$

Feature maps

We need to define the feature map that takes histories to states in both the tabular and function approximation cases.

- In the tabular case, we use suffix trees to map histories to states.
- In the function approximator case we define a new feature class of event selectors. A feature ξ_i checks the $n - m$ position in the history (h_n) for an observation-action pair (o, a).

If the history is $(0, 1), (0, 2), (3, 4), (1, 2)$ then a event-selector checking 3 steps in the past for the observation-action pair $(0, 2)$ will be turned on.

Relation to existing TD-based approaches

- This work resembles recent regularised TD-based function approximation methods.
- The key differences are in the regulariser and in the use of simulated annealing to find suitable feature sets.
- The problem setting.

Experimental results: Cheesemaze



2	3	4	3	
1		1		1
0				0

Figure: Cheese Maze Domain

Experimental results: Cheesemaze

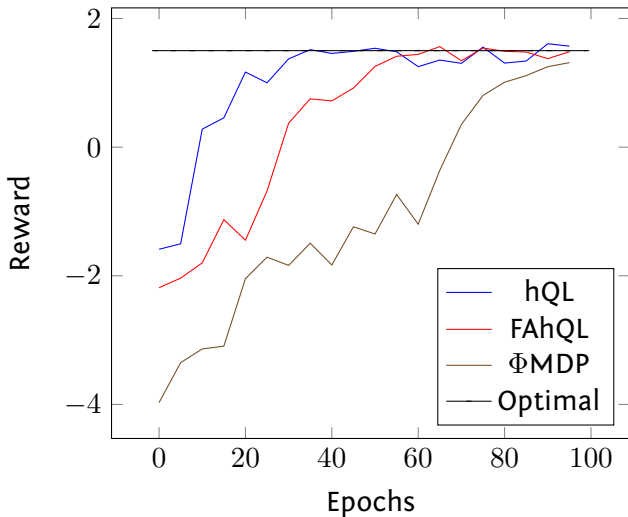


Figure: Comparison between hQL, FAhQL and Φ MDP on Cheese Maze

Domain : Tiger

- You must choose between 2 doors.
- One has a tiger behind it and the other a pot of gold.
- You can listen for the tiger's growl, but the resulting observation is only accurate 85% of the time.

Experimental Results : Tiger

Comparison between hQL, FAhQL and Φ MDP on Tiger

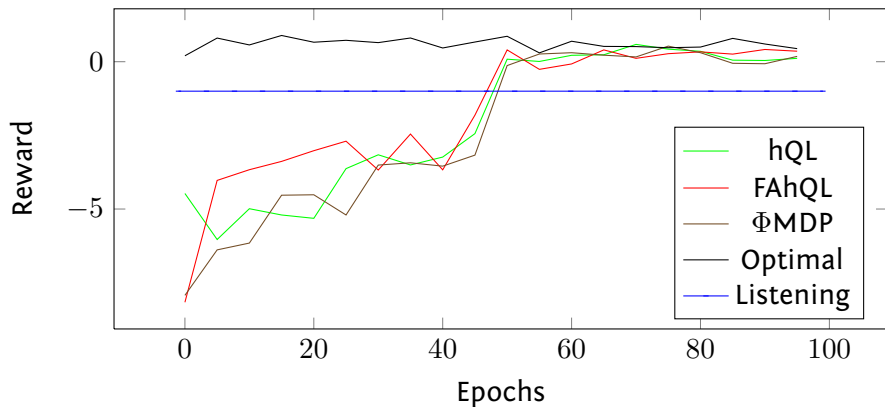
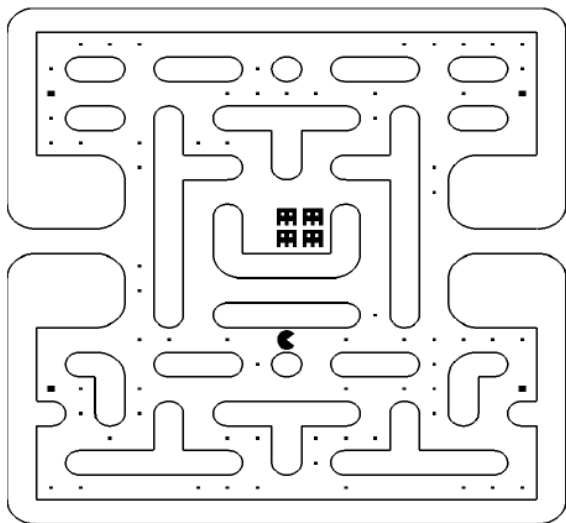


Figure: Comparison between hQL, FAhQL and Φ MDP on Tiger

Domain : POCMAN



Experimental Results : POCMAN

POCMAN : Rolling average over 1000 epochs

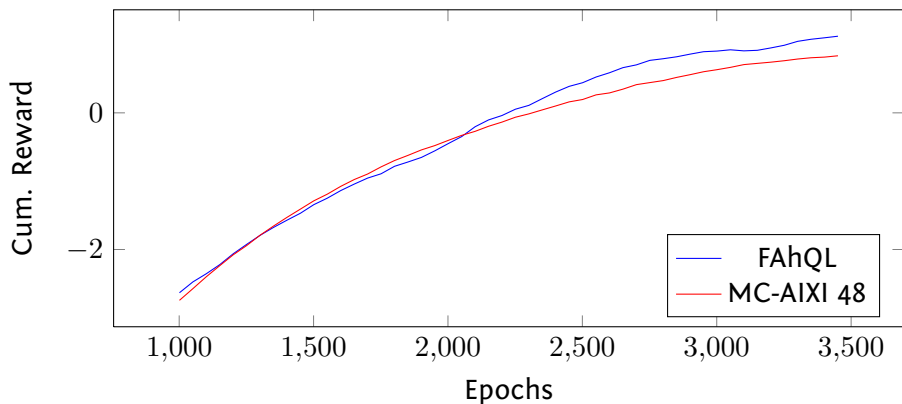


Figure: MC-AIXI vs hQL on Pocman

Computation used : POCMAN

Table: Computational comparison on Pocman

Agent	Cores	Memory(GB)	Time(hours)	Iterations
MC-AIXI 96 bits	8	32	60	$1 \cdot 10^5$
MC-AIXI 48 bits	8	14.5	49.5	$3.5 \cdot 10^5$
FAhQL	1	0.4	17.5	$3.5 \cdot 10^5$

Conclusions/Future Work

- We introduced a model-free cost to the Feature RL framework which allows for scaling to large environments.
- The resulting algorithm can be viewed as an extension of Q-learning to the history-based setting.

Problems/Future Work

- It does not deal with the exploration-exploitation problem. It uses ϵ -greedy exploration.
- The extension to function approximation should be made sound by using methods like Greedy-GQ to avoid divergence.
- Current work is using this as a feature construction method to learn how to play ATARI games.



Questions?