Q-Learning for history-based Reinforcement Learning

Mayank Daswani*, Peter Sunehag*, Marcus Hutter*, Australian National University. mayank.daswani@anu.edu.au

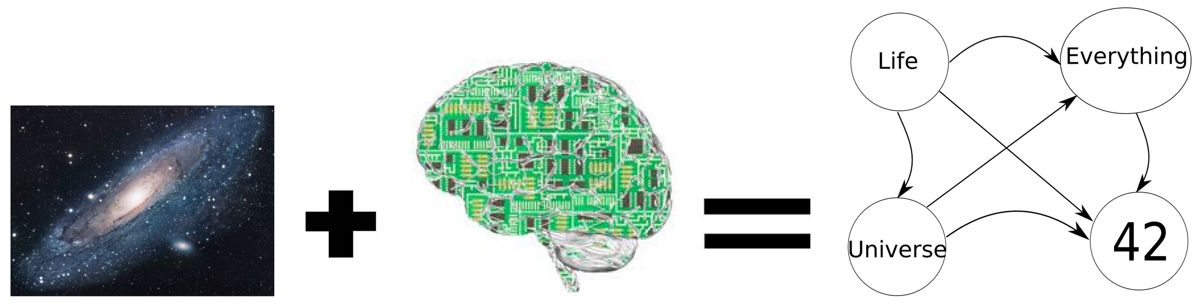
Summary.

- Feature reinforcement learning (FRL) maps non-Markov environments to a Markov state space.
- Scaling FRL for the original cost function is hard since it is model-based.
- By constructing a cost based on the value function, we can now apply linear function approximation to make the framework tractable for large spaces.
- Experimental results show that the algorithm is competitive with state-of-the-art methods.

1 The Problem

The general reinforcement learning (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.

2 Feature RL



Feature Reinforcement Learning aims to automatically reduce a complex real-world problem to a useful representation (MDP) i.e. 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,

$$\phi(h_t) = s_t$$

In order to select the best ϕ, we need a cost function on ϕ and a way to search over the space containing ϕ. The original cost function used is given below.

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

- 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.

3 Model-free Cost

- The above cost function is model-based and cannot be directly used in large observation spaces.
- A viable alternative is to use a model-free cost that can then be function approximated.

For each ϕ we define a Q-table based on the state space given by ϕ to be of the form $Q(\phi(h), a)$. We use the squared pathwise Q-learning error to find a suitable map $\phi : \mathcal{H} \to \mathcal{S}$ by selecting ϕ to minimise the following cost,

$$Cost_{QL}(\phi) = \min_{Q} \frac{1}{2} \sum_{t=1}^{n} (r_{t+1} + \gamma \max_{a} Q(\phi(h_{t+1}), a) - Q(\phi(h_t), a_t))^2 + Reg(\phi)$$

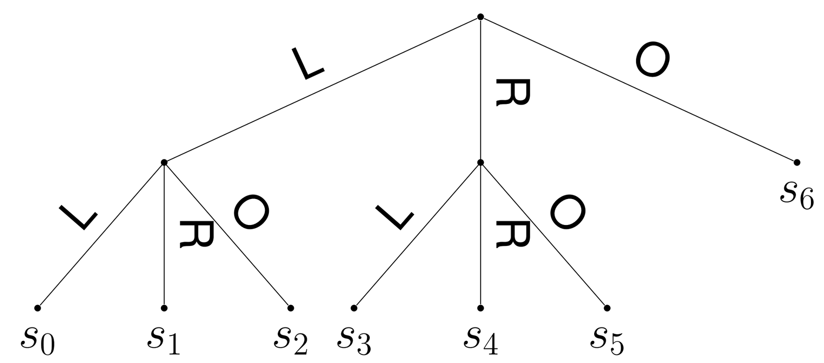
This can easily be extended to the linear function approximation setting by representing Q as linear function over some features. The stochastic search is then over these features.

4 Algorithm

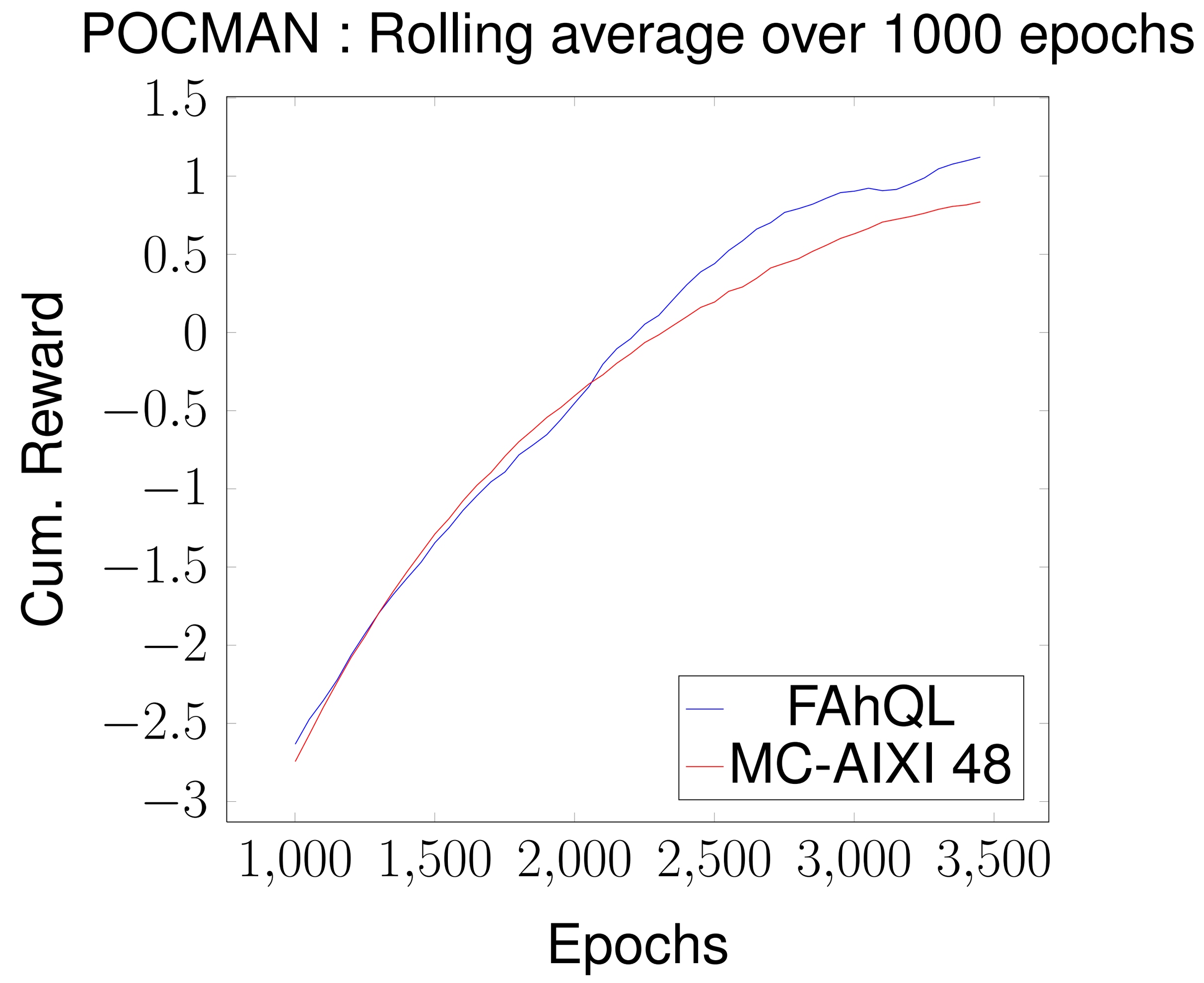
An intuitive explanation of the algorithm is as follows.

- We are given some environment which produces an observation and reward when given an action.
- Act randomly for a predefined amount of time.
- Loop the following.
  - Run simulated annealing on the existing history using the above Cost function to get ϕ.
  - Map the history sequence to a state sequence.
  - Use Q-learning on the state sequence to find a suitable policy.
  - Act according to this policy for some time while adding the resulting observations and rewards to the history.

5 Features



- For the tabular approach, we used suffix trees which allow us to uniquely map history sequences to state sequences.
- For the linear function approximation we defined a set of features known as event selectors (and a modified version called bit selectors).
- An event selector is a set of features ξ_j. Each feature ξ_j consists of a position m and an observation o. Feature ξ_j is on if the $(n-m)$-th position in the history has observation o.
- A bit selector is similar but picks out bits of the history instead of observations.



6 Experiments

- The algorithm was tested on three domains. Tiger, Cheesemaze and Partially Observable Pacman (Pocman).
- On Tiger and Cheesemaze we compared against the standard ϕMDP algorithm using suffix trees and event selectors.
- On Pocman, we compared against MC-AIXI using bit selectors.
- In the smaller environments, our algorithm performed competitively achieving optimal performance but being more feature-efficient in the function approximation case.
- On Pocman, we perform better than MC-AIXI and use 36 times less space and are twice as fast.

7 Conclusion

- The algorithm presented above can be viewed as an extension of Q-learning to the history-based setting.
- It allows us to scale the feature RL framework to large environments.

*Research School of Computer Science, College of Engineering & Computer Science, Australian National University

Australian National University

ANU College of Engineering & Computer Science