MCMT Project
Mixing Time in Reinforcement Learning

Shun Zhang

December 7, 2014

What is Reinforcement Learning

- Learning with feedback, or sequential decision making.
- Defined on *Markov Decision Process*,
- which is an extension to Markov chain.

Definitions

Definition

Markov Chain is a two-element tuple (Ω, P), where

- Ω is the state space.
- P is the transition probability. $P : \Omega \times \Omega \to \mathbb{R}$.

Definition

Markov Decision Process is a four-element tuple (Ω, A, P, R), where

- Ω is the state space.
- A is an action set.
- P is the transition probability. $P : \Omega \times A \times \Omega \to \mathbb{R}$.
- R is the reward upon reaching a state. $R : \Omega \to \mathbb{R}$. [2]

Definitions

Definition

A policy in a MDP is a mapping $\pi : \Omega \to A$.

Definition

The utility of a state s following policy π is:
$U(s) = \mathrm{E}(R(s) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots)$, where $\gamma \in (0, 1]$, is a discounting factor.

Objective

- The goal of the agent is to maximize the accumulated rewards it gets.
- The agent should take the action that leads to the maximum expected utilities, that is $\mathrm{argmax}_a \sum_{s'} P(s, a, s')(R(s') + \gamma U(s'))$.
- Explore or Exploit Dilemma (ϵ-greedy, Upper bound confidence interval).

Learning algorithms: Value Iteration

Theorem

The utility of the optimal policy satisfies
$U(s) = \max_a \sum_{s'} P(s,a,s')(R(s') + \gamma U(s'))$ *for all* s.

To update, we can use the following rule. This is a bootstrapping way to solve the nonlinear equations above.

$$U(s) \leftarrow \max_a \sum_{s'} P(s,a,s')(R(s') + \gamma U(s')) \tag{1}$$

Learning algorithms: Q-learning

Definition

$Q(s,a) = \sum_{s'} P(s,a,s')(R(s') + \gamma U(s'))$.

Note that by definition, $U(s) = \max_a Q(s,a)$. The Q function can be computed by the following update rule.

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha(R(s') + \gamma \max_{a'} Q(s',a')) \quad (2)$$

where $\alpha \in (0,1)$.

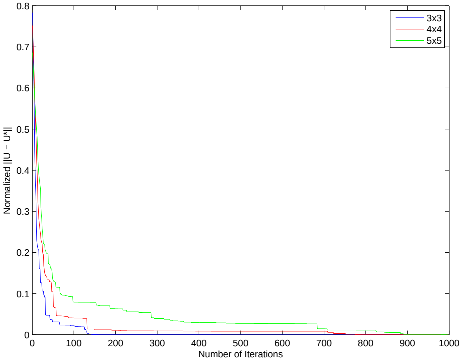
Theorem

Following the update rule Equation 2, Q eventually converges to Q^*.

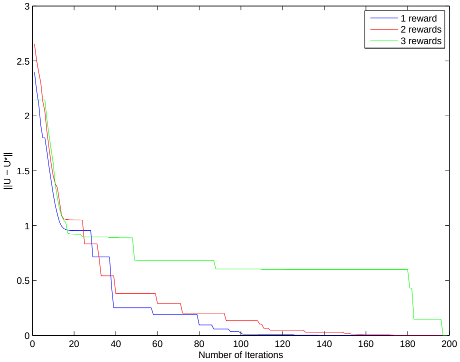
Experiments

One measure of the learning performance is to compare the current utility with that of the optimal solution, that is $\sum_s ||U(s) - U^*(s)||$ or $\sum_{s,a} ||Q(s,a) - Q^*(s,a)||$ for $s \in \Omega$.

Experiments



Experiments



E^3 Algorithm

Kearns et al. proposed an algorithm Explicit Explore or Exploit (E^3)[1] that uses the mixing time of the domain. The convergence time of the learning algorithm can be polynomially bounded by the mixing time of the transition function with high probability.

E^3 Algorithm

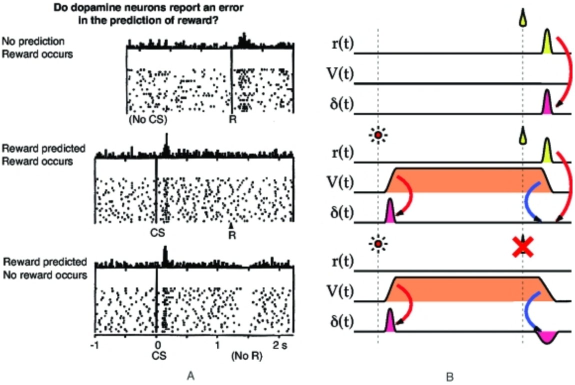
Theorem

Let $U(i)$ denote the value function for the policy with the optimal expected discounted return in M. Then there exists an algorithm A, taking inputs ϵ, δ, N and $U(i)$,such that the total number of actions and computation time taken by A is polynomial in $1/\epsilon, 1/\sigma, N$,the mixing time of the transition function, and the maximum reward. With probability at least $1-\delta$, A will halt in a state i, and output a policy such that following such policy, $U(i) \geq U^*(i) - \epsilon$.

Sketch of E^3 Algorithm

- Initially, the set S of known states is empty.
- Any time the current state is not in S, the algorithm performs random walk.
- Any state that is visited *enough times* in the random walk is marked as *known*, and not considered for exploration in the future.

Conclusion



Do dopamine neurons report an error
in the prediction of reward?
No prediction
Reward occurs
(No CS)
R
Reward predicted
Reward occurs
CS
R
Reward predicted
No reward occurs
-1
0
CS
1
(No R)
2 s
A
r(t)
V(t)
δ(t)
B

[1] Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49(2-3):209–232, 2002.

[2] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. Cambridge Univ Press, 1998.