Reinforcement Learning
Lecture 3

Gillian Hayes

15th January 2007

Learning from Interaction

– with environment

– to achieve some goal

- Baby playing. No teacher. Sensorimotor connection to environment.
  - Cause – effect
  - Action – consequences
  - How to achieve goals
- Learning to drive car, hold conversation, etc.
  - Environment's response affects our subsequent actions
  - We find out the effects of our actions later

Simple Learning Taxonomy

- Supervised Learning
  - "Teacher" provides required response to inputs. Desired behaviour known. "Costly"
- Unsupervised Learning
  - Learner looks for patterns in inputs. No "right" answer
- Reinforcement Learning
  - Learner not told <u>which</u> actions to take, but gets reward/punishment from environment and adjusts/learns the action to pick next time.

Reinforcement Learning

Learning a mapping from situations to actions in order to maximise a scalar reward/reinforcement signal

Exploration/Exploitation Tradeoff

High rewards from trying previously-well-rewarded actions – **EXPLOITATION**

BUT

Some actions we've not tried before might be better – **EXPLORATION**

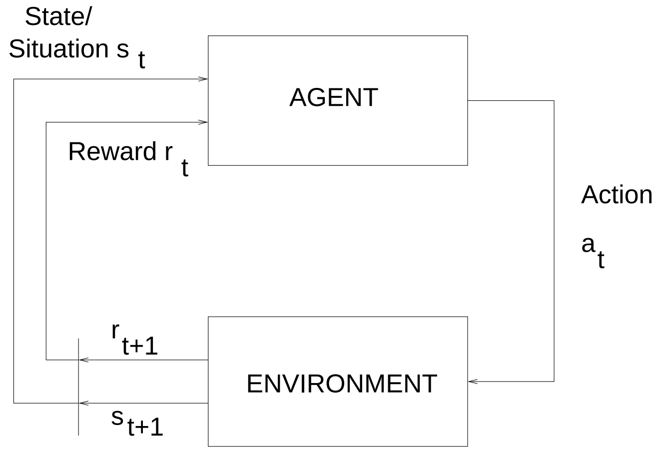
MUST DO BOTH

Especially if task stochastic, try each action many times per situations to get reliable estimate of reward.

Examples

- Animal learning to find food and avoid predators
- Robot trying to learn how to dock with charging station
- Backgammon player learning to beat opponent
- Football team trying to find strategies to score goals
- Infant learning to feed itself with spoon
- Cornet player learning to produce beautiful sounds
- Temperature controller keeping FH warm while minimising fuel consumption

Framework



Agent in situation/state s_t chooses action a_t
World changes to situation/state s_{t+1}
Agent perceives situation s_{t+1} and gets reward r_{t+1}

Telling the agent what to do is its

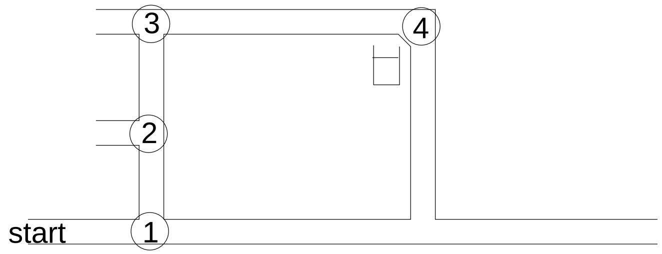
POLICY $\pi_t(s, a) = Pr\{a_t = a | s_t = s\}$

Given the situation at time t is s, the policy gives the probability the agent's action will be a.

For example: $\pi_t(s, \text{stay}) = 0.5$, $\pi_t(s, \text{go}) = 0.5$.

Reinforcement learning $\Rightarrow$ Get/find/learn the policy

Example Policy: Find the Coffee Machine



$\pi(1) \Rightarrow$ turn left	or	$\pi(1, \text{turn left}) \Rightarrow 1$
$\pi(2) \Rightarrow$ straight on		$\pi(1, \text{straight on}) \Rightarrow 0$
$\pi(3) \Rightarrow$ turn right		$\pi(1, \text{turn right}) \Rightarrow 0$
$\pi(4) \Rightarrow$ go through door		$\pi(1, \text{go through door}) \Rightarrow 0$
		etc.

Example Policy: Bandit Problem

10 arms, Q table gives the Q value for each arm

ϵ-greedy policy:

$\pi(s, a'|a' = \arg\max_a Q(a)) = 1 - \epsilon + \frac{\epsilon}{|A|}$

else

$\pi(s, a) = \frac{\epsilon}{|A|}, |A| = 10$

Choose the action with the highest Q value 1 - ϵ of the time – this action looks to be the best. It's the **greedy** action.

The remaining ϵ of the time choose over all the $|A|$ actions equally (which of course includes the best-looking action too).

Value Functions

• How desirable is it to be in a certain state?

What is its *value*?

$$V(s) \rightarrow \text{Value}$$

Value is (an estimate of) the expected future reward from that state

• Value vs. reward Long-term vs. immediate

⇒ Want actions that lead to states of high value, not necessarily high immediate reward

• Learn policy via learning value – when we know the values of states we can choose to go to states of high value (cf. GAGP discover policy directly)

• Genotypical vs. phenotypical learning? (GAGP vs. RL)

General Reinforcement Learning Algorithm

1. Initialise learner's internal state (e.g. Q values, other statistics)

2. Do for a long time
   - Observe current world state s
   - Choose action a using the policy
   - Execute action a
   - Let r be immediate reward, s' new world state
   - Update internal state based on s, a, r, s', previous internal state

3. Output a policy based on, e.g. learnt Q values and follow it

Requirements for an RL Algorithm

We need:

- Decision on what constitutes an internal state
- Decision on what constitutes a world state
- Sensing of a world state
- Action-choice mechanism (policy) based usually on
- an evaluation (of current world and internal state) function
- A means of executing the action
- A way of updating the internal state

The Environment vs the Learner

Environment (possibly/often simulated) provides

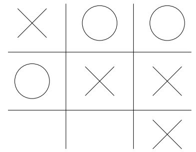
- Transition function describing the probability that a given action will take you from one world state to another – often called **the model** in the literature
- A reward function which says how much reward you will get for carrying out an action and ending up in a particular state – often also included in **the model**. (Definitely mention this in **exam questions** asking you about the model!)

You the simulator designer must specify these. If using the real world, they are given.

But of course from the learner's point of view, the learner has to discover what these are while exploring the world.

Example – Noughts and Crosses

See Sutton and Barto Section 1.4 and Figure 1.1.



Construct a player to play against an **imperfect** opponent

For each board state, set up $V(s)$ – estimate of probability of winning from that state

XXX	$V(s) = 1$
OOO	$V(s) = 0$
Rest	$V(s) = 0.5$ initially

Play many games

Move selection

- mostly pick move leading to state with highest V
- sometimes explore

Value adjustment

- back-up value of states after non-exploratory moves to states preceding moves
- e.g. $V(s_k) = V(s_k) + \alpha[V(s_{k+1}) - V(s_k)]$

Reduce α over time

V converges to probabilities of winning – optimal policy

Jargon

Policy $\pi(s, a)$	Decision on what action to do in that state
Reward function	Defines goal, and good and bad experience for learner
Value function	Predicts reward. Estimate of total future reward
Model of the environment	Maps states and actions onto states $S \times A \rightarrow S$. If in state s_1 we take action a_1, it predicts s_2 (and sometimes reward r_2). Not all agents use models.

Reward function and environmental model fixed external to agent.

Policy, value function, estimate of model adjusted during learning.