REINFORCEMENT LEARNING

Single-state RL

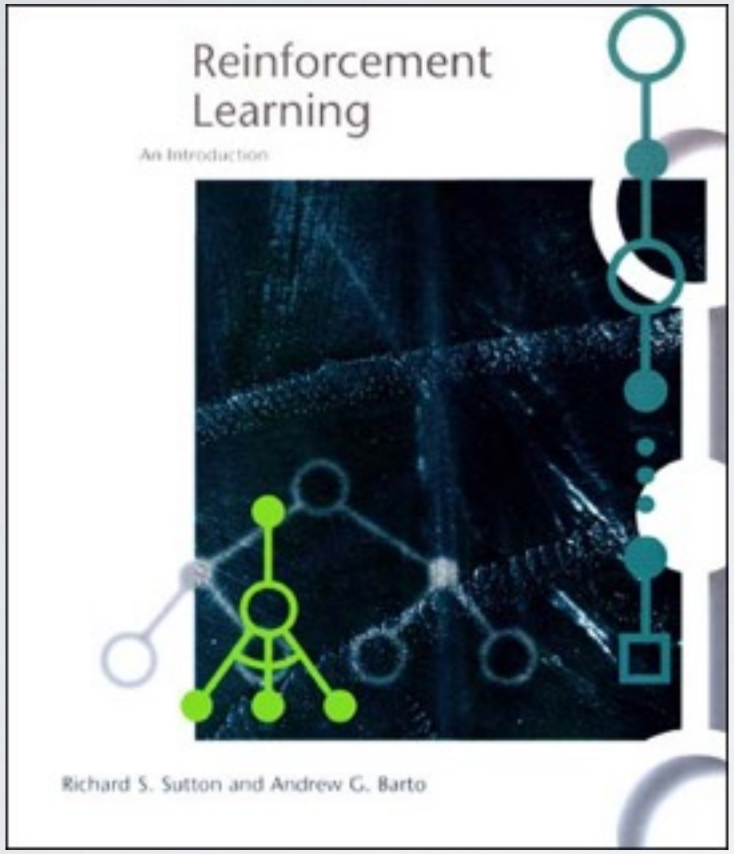
COORDINATES

- Prof. Dr. Yann-Michaël De Hauwere
- Office: VUB - Campus Etterbeek, Building G, 10th floor, Room 10G720
- email: ydehauwe@vub.ac.be
- http://ai.vub.ac.be

SCHEDULE

Date	Description
~~18/09/2014~~	~~No course this day~~
~~25/09/2014~~	~~Game theory basics~~
~~2/10/2014~~	~~Mixed strategies and Nash algorithms~~
~~9/10/2014~~	~~Extensive form games and their equilibria~~
~~16/10/2014~~	~~Evolutionary game theoy~~
~~23/10/2014~~	~~Evolution of cooperation~~
30/10/2014	N-armed bandits (stateless reinforcement learning)
6/11/2014	Graphical games
13/11/2014	Reinforcement learning and MDPS
20/11/2014	No course this day
27/11/2014	Sparse Interactions
4/12/2014	Project preparation time
11/12/2014	Selfish load balancing
18/12/2014	
25/12/2014	Winter break
1/01/2015	
Exam: Article + presentation of group project	

BIBLIOGRAPHY



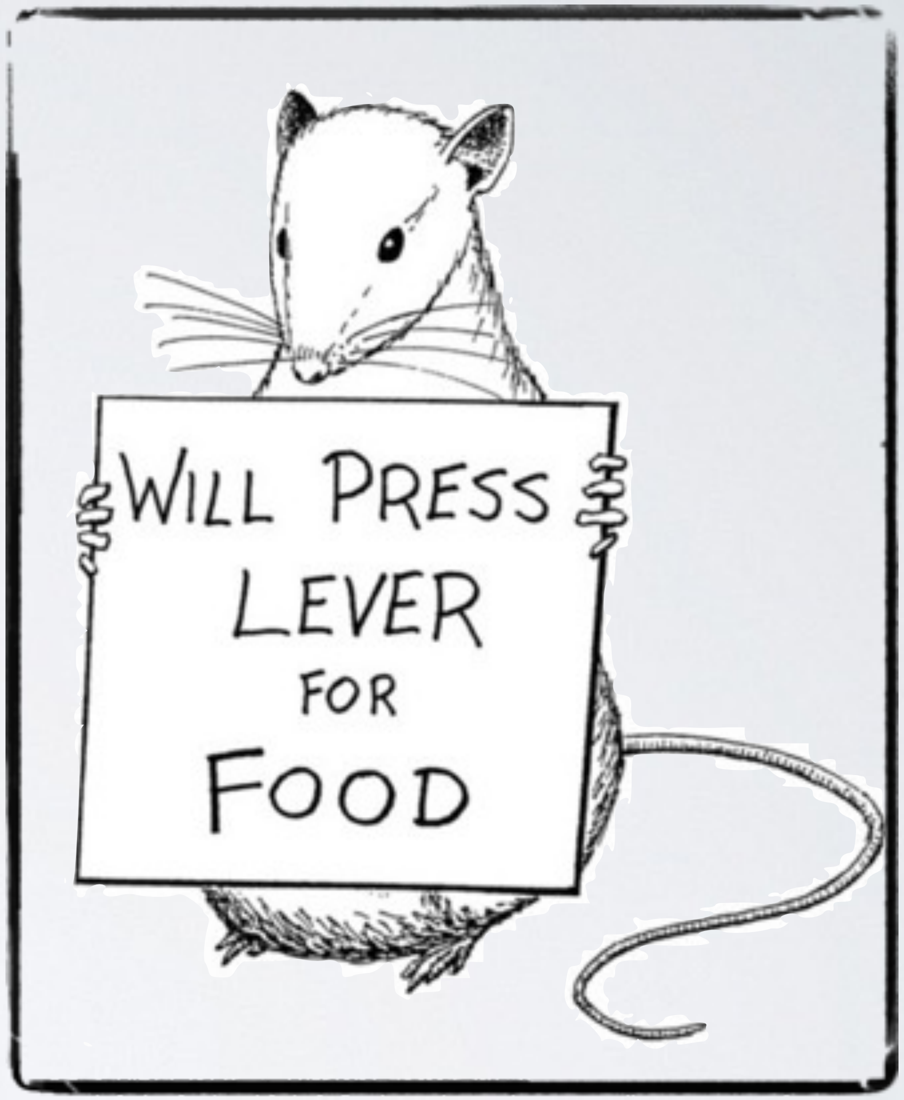
Reinforcement Learning: an introduction

R.S. Sutton and A.G. Barto

Available for free online

PART I

Reinforcement Learning
Introduction



WHY REINFORCEMENT LEARNING?

Based on ideas from psychology

- Edward Thorndike's **law of effect**

  Satisfaction strengthens behaviour, discomfort weakens it

- B.F. Skinner's principle of reinforcement

  Skinner Box: train animals by providing (positive) feedback

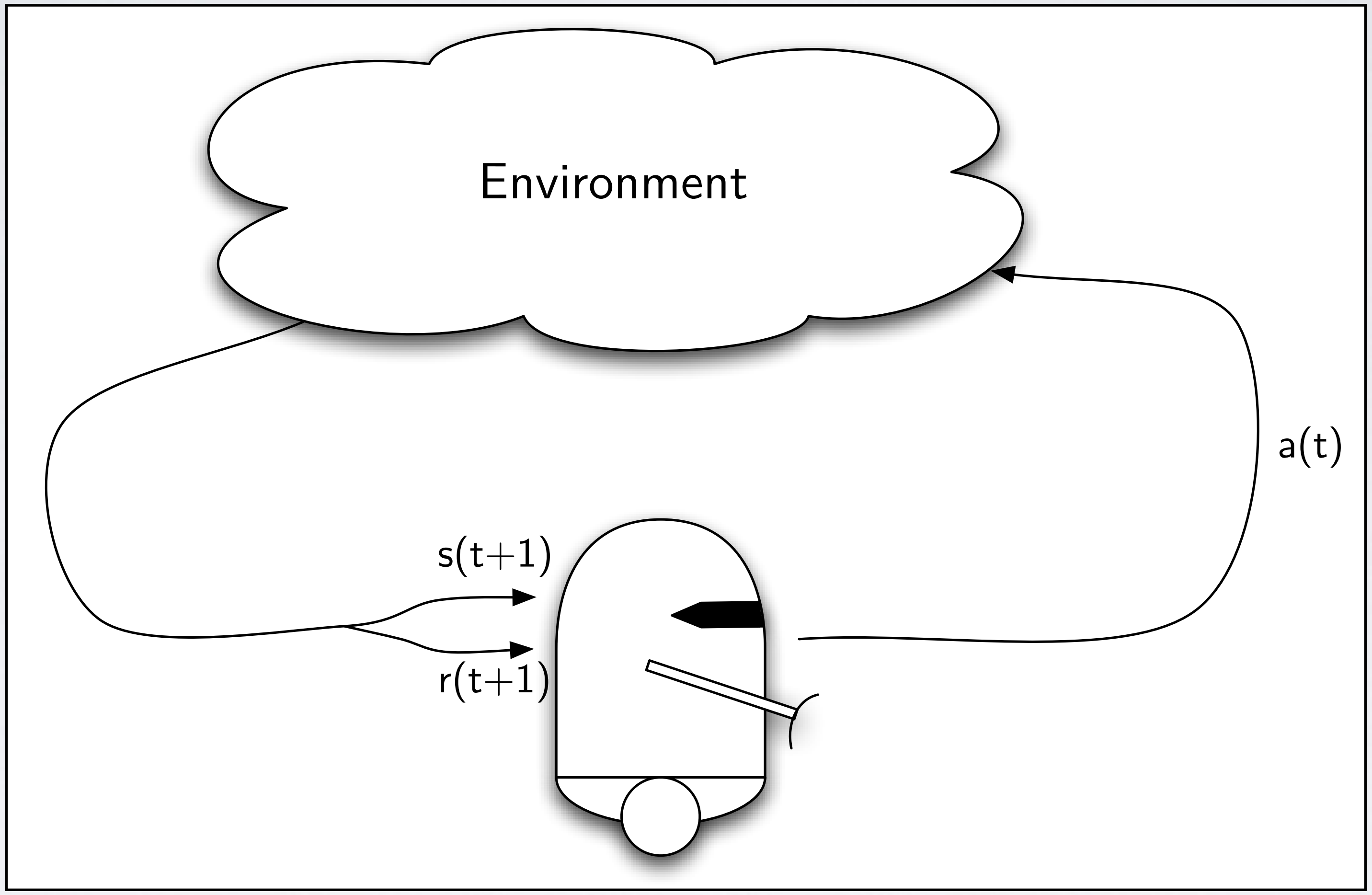
Learning by interacting with the environment

WHY REINFORCEMENT LEARNING?

Control learning

- Robot learning to dock on battery charger
- Learning to choose actions to optimize factory output
- Learning to play Backgammon/other games
- ….

THE RL SETTING



- Learning from interactions
- Learning what to do **- how to map situations to actions -** so as to maximize a numerical reward signal

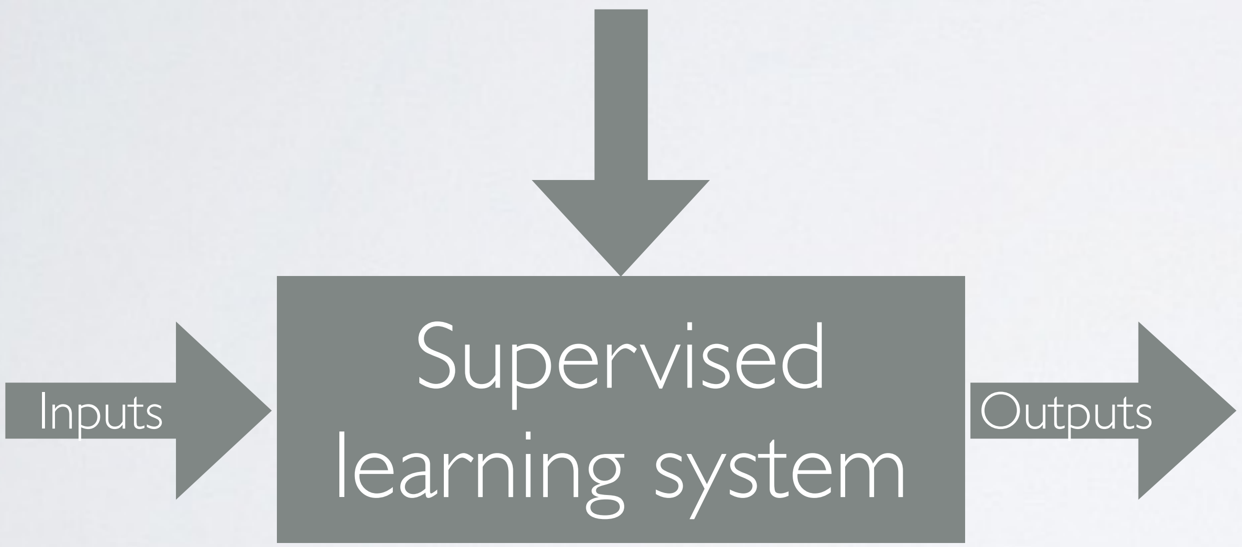
KEY FEATURES OF RL

- Learner is **not** told which action to take
- Trial-and-error approach
- Possibility of **delayed reward**
  - Sacrifice short term gains for greater long-term gains
- Need to balance **exploration** and **exploitation**
- Possible that states are only partially observable
- Possible needs to learn multiple tasks with same sensors
- In between **supervised** and **unsupervised** learning

SUPERVISED VS UNSUPERVISED

Supervised learning

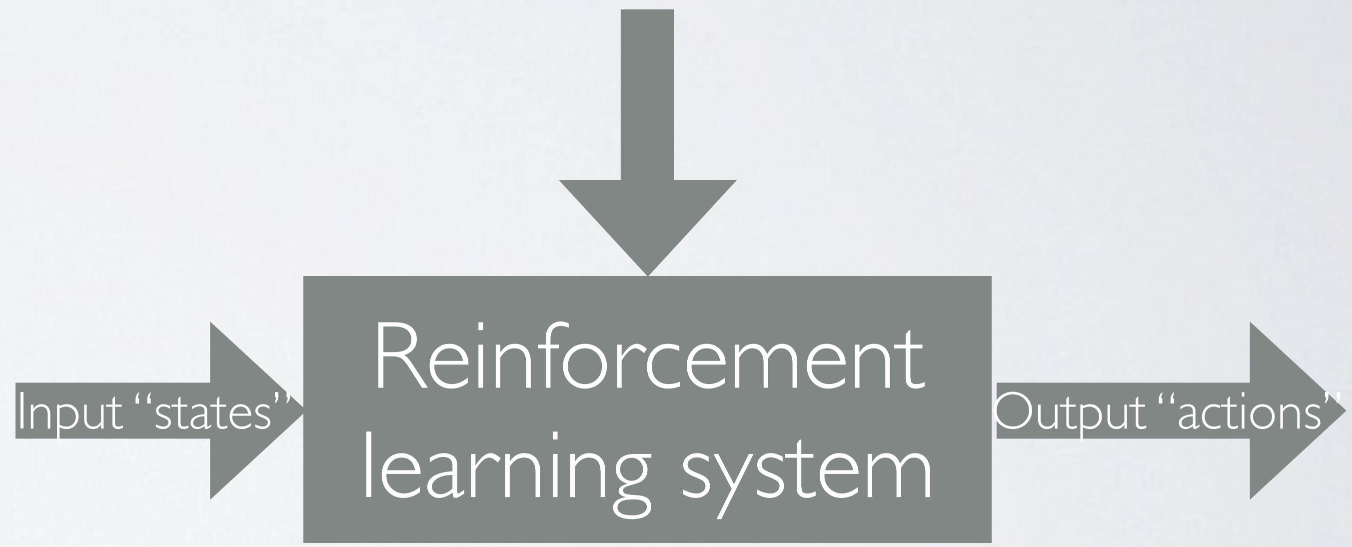
Training info = desired (target) outputs



Error = (target output - actual output)

Unsupervised learning

Training info = evaluations

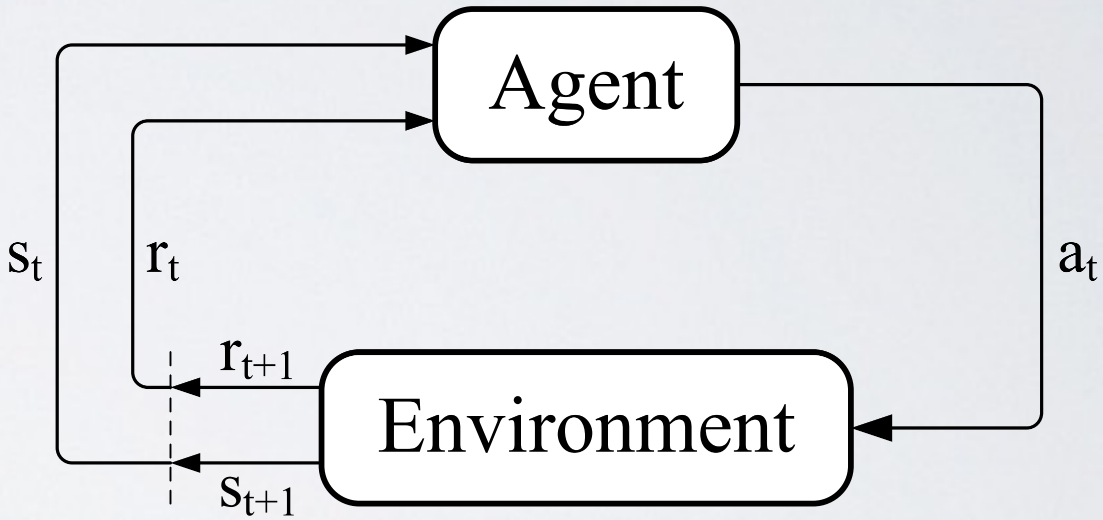


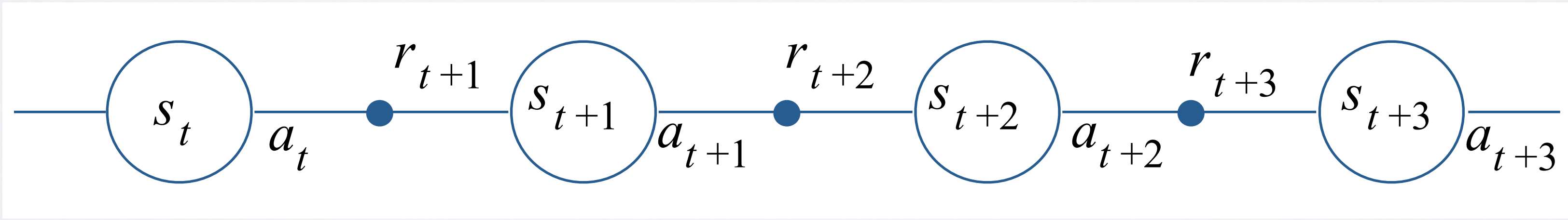
Objective: get as much reward as possible

THE AGENT-ENVIRONMENT INTERFACE

Agent interacts at discrete time steps $t = 0, 1, 2, \ldots$

- Observes state $s_t \in S$
- Selects action $a_t \in A(s_t)$
- Obtains immediate reward $r_{t+1} \in \mathbb{R}$
- Observes resulting state s_{t+1}





ELEMENTS OF RL

- Time steps need not refer to fixed intervals of real time
- **Actions** can be
  - low level (voltage to motors)
  - high level (go left, go right)
  - "mental" (shift focus of attention)
- **States** can be
  - low level "sensations" (temperature, (x,y) coordinates)
  - high level abstractions, symbolic
  - subjective, internal ("surprised", "lost")
- The **environment** is not necessarily known to the agent

ELEMENTS OF RL

- **State transitions** are
  - changes to the internal state of the agent
  - changes in the environment as a result of the agent's action
  - can be nondeterministic
- **Rewards** are
  - goals, subgoals
  - duration
  - …

LEARNING HOW TO BEHAVE

- The agent's **policy** π at time t is
  - a mapping from states to action probabilities
  - $\pi_t(s, a) = P(a_t = a | s_t = s)$

- Reinforcement learning methods specify **how** the agent changes its policy as a result of experience
- Roughly, the agent's goal is to **get as much reward** as it can over the long run

THE OBJECTIVE

- Use **discounted return** instead of total reward

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

- where $\gamma \in [0, 1]$ is the **discount factor** such that

shortsighted $\quad 0 \leftarrow \gamma \rightarrow 1 \quad$ farsighted

GOALS AND REWARDS

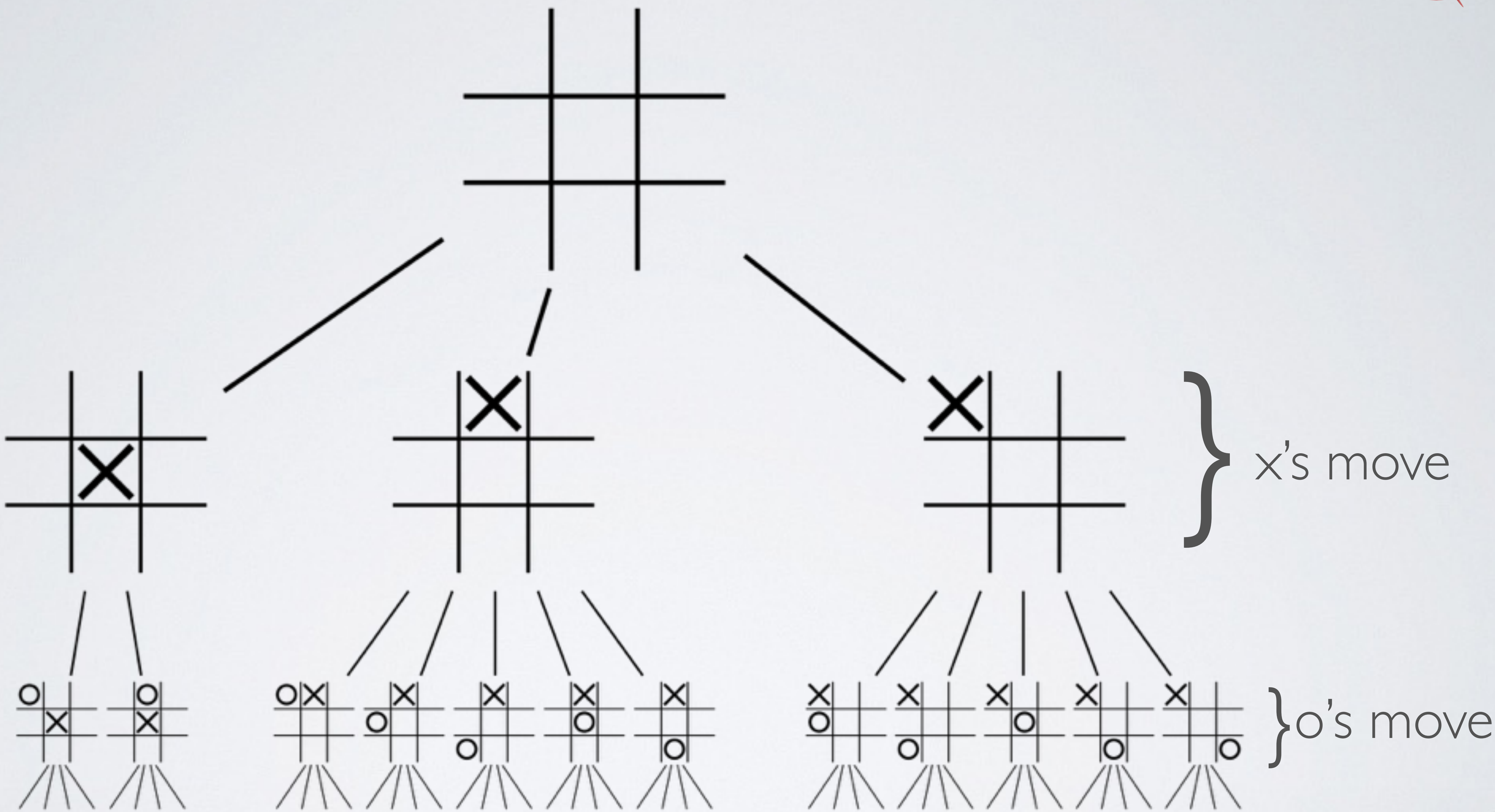
- Is a scalar reward signal an adequate notion of a goal?

- A goal should specify **what** we want to achieve, **not how** to achieve it

- A goal must be outside the agent's direct control, thus outside the agent

- The agent must be able to measure success:
  - explicitly
  - frequently during its lifespan

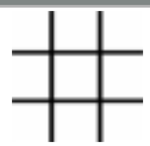
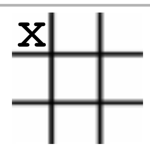
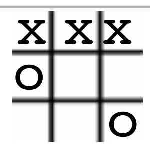
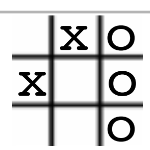
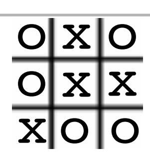
EXAMPLE: TIC-TAC-TOE



Assume an imperfect oponent: he/she makes mistakes

EXAMPLE:TIC-TAC-TOE

1. Make a table with one entry per state

State	V(s)	Estimated probability of winning
	.5	?
	.5	?
. . .	. . .	
	1	win
. . .	. . .	
	0	loss
. . .	. . .	
	0	draw

2. Now play lots of games

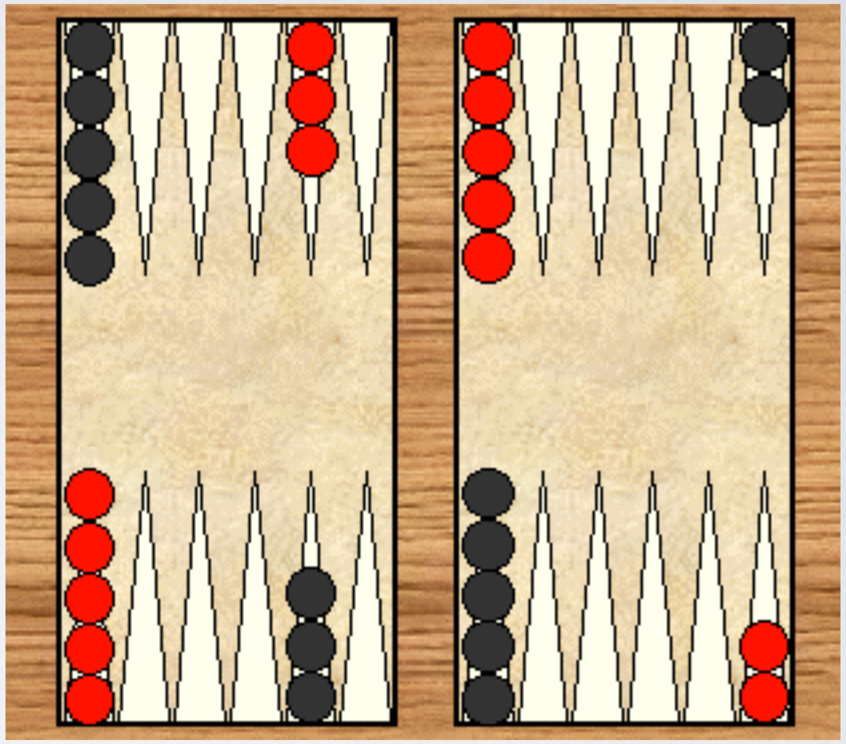
To pick moves:
look ahead one step

Pick the next state with the highest probability of winning

But 10% of the time pick a move at random = exploration

EXAMPLE: BACKGAMMON

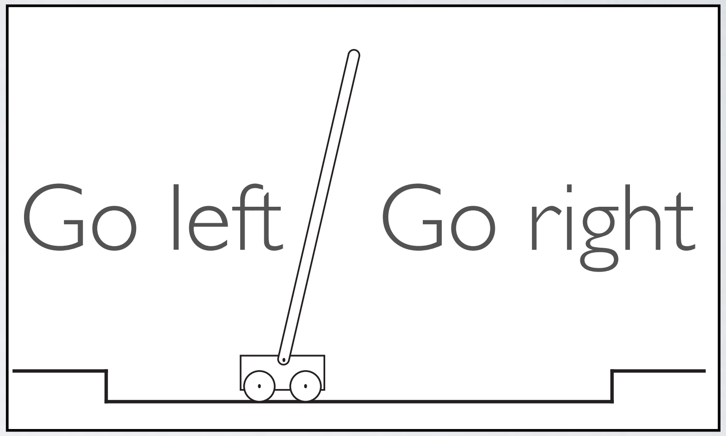
- Learn to play backgammon
- Immediate reward:
  - +100 if win
  - -100 if lose
  - 0 for all other states



- Trained by playing 1.5 million games against itself
  Now approximately equal to best human player

EXAMPLE: POLE BALANCING

- An **epsiodic** task where episode ends upon failure:
  - reward = +1 for each step
  - return = # steps before failure



- A **continuing** task with discounted return:
  - reward = -1 upon failure
  - return $= -\gamma^k$, for *k* steps before failure

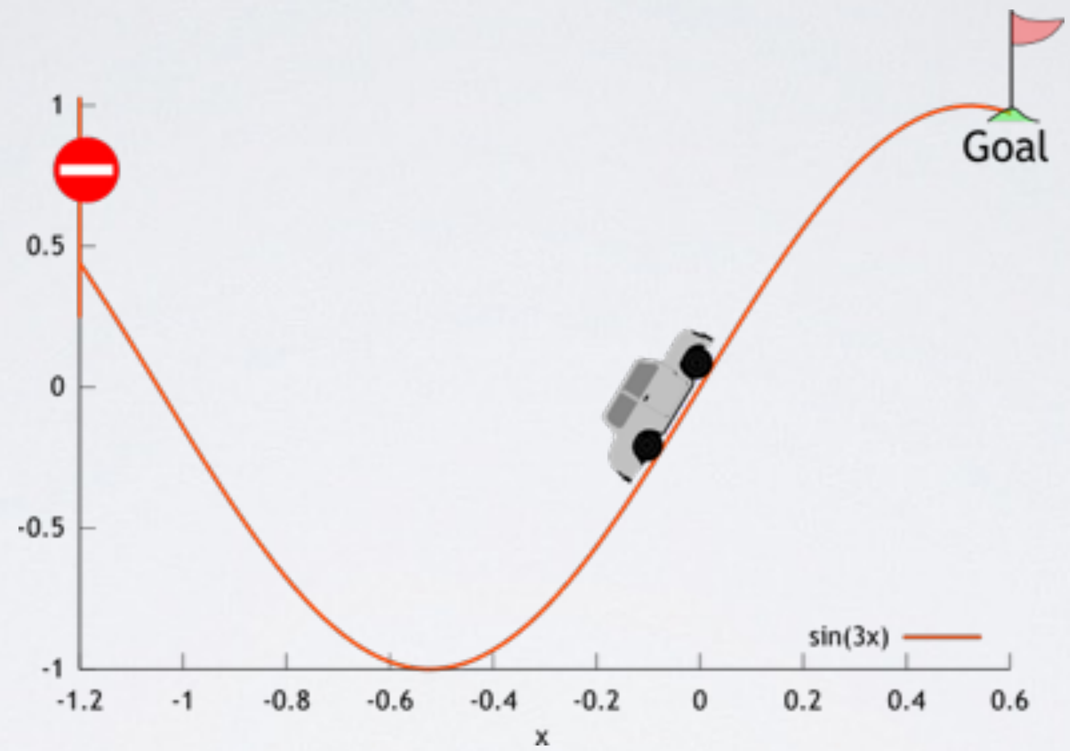
- Return is maximized by avoiding failure as long as possible

$$\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

EXAMPLE: POLE BALANCING

EXAMPLE: MOUNTAIN CAR

- Get to the top of the hill as quickly as possible (actions: Left or Right)

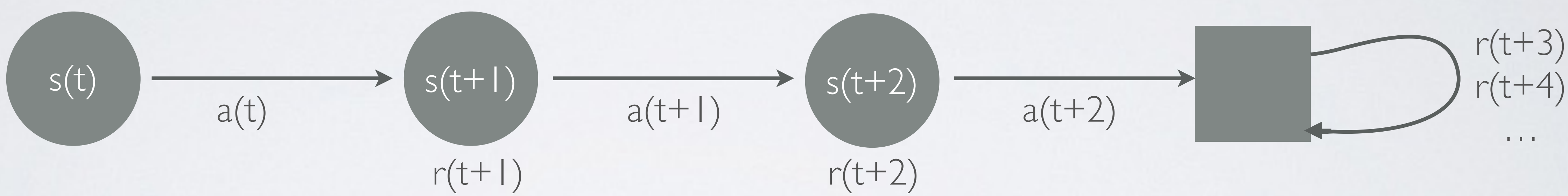


reward = -1 for each step **not** at top of hill
return = -#steps before reaching top of hill

- Return is maximized by minimizing # steps

A UNIFIED NOTATION

- Think of each episode as ending in an absorbing state that always produces reward of zero:



$$R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

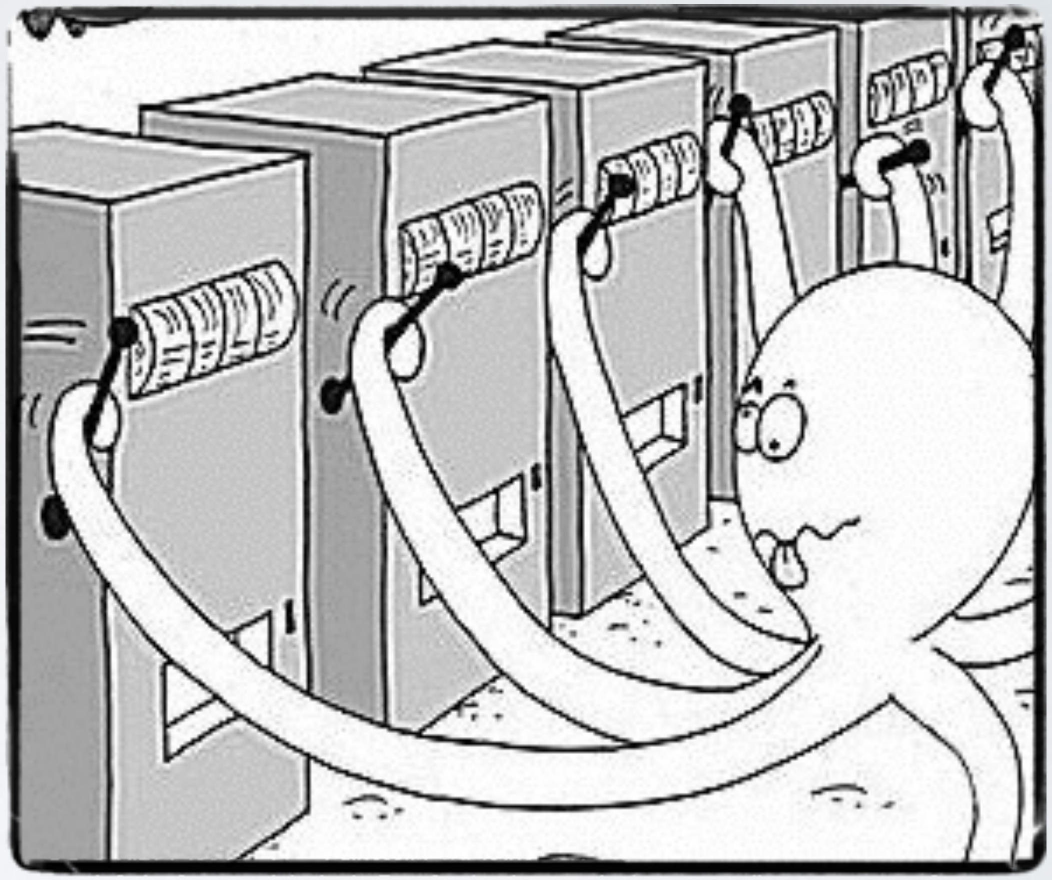
- γ can only be 1 if a zero reward absorbing state is reached

OTHER EXAMPLES

- **Robocup Soccer Teams** Stone & Veloso, Reidmiller et al.
  World's best player of simulated soccer, 1999; Runner-up 2000
- **Inventory Management** Van Roy, Bertsekas, Lee & Tsitsiklis
  10-15% improvement over industry standard methods
- **Dynamic Channel Assignment** Singh & Bertsekas, Nie & Haykin
  World's best assigner of radio channels to mobile telephone calls
- **Elevator Control** Crites & Barto
  (Probably) world's best down-peak elevator controller
- **Many Robots**
  navigation, bi-pedal walking, grasping, switching between skills...
- **TD-Gammon and Jellyfish** Tesauro, Dahl
  World's best backgammon player

PART II

Single state RL



EVALUATIVE FEEDBACK

ch2. Sutton & Barto

Evaluating actions vs **instructing** by giving correct actions

- **Pure evaluative feedback** depends solely on the action taken
- **Pure instructive feedback** depends not at all on the action taken

Supervised learning = instructive, RL = evaluative

Associative vs **Non-assoctive**

- **Associative:** inputs mapped to outputs;
  learn the best output for each input
- **Non-associative:** "learn" (find) one best output

n-Armed bandit (at least how we treat it) is:

- Non-associative
- Evaluative feedbak

THE N-ARMED BANDIT PROBLEM

Choose repeatedly from *n* actions; each choice is called a play

After each play a_t, you get a reward r_t, where

$$E\{r_t|a_t\} = Q^*(a_t)$$

These are unknown action values
Distribution of r_t depends only on a_t

Objective is to **maximize the reward** in the long run, e.g. over 1000 plays

To solve the n-armed bandit problem, you must **explore** a variety of actions and **exploit** the best of them

EXPLORATION/EXPLOITATION DILEMMA

- Suppose you form estimates

$$Q_t(a) \approx Q^*(a)$$

- The greedy action at *t* is a_t

$$a_t^* = \underset{a}{\operatorname{argmax}}\, Q_t(a)$$

$$a_t = a_t^* \rightarrow exploitation$$

$$a_t \neq a_t^* \rightarrow exploration$$

- Constant exploration = bad idea
- Constant exploitation = bad idea
- Stop exploration = bad idea
- Reduce exploration = good idea (maybe)

ACTION/VALUE METHODS

Methods that adapt action-value estimates and nothing else, e.g. suppose by the *t*-th play, action *a* has been chosen k_a times, producing rewards $r_1, r_2, \ldots, r_{k_a}$, then

$$Q_t(a) = \frac{r_1, r_2, \ldots, r_{k_a}}{k_a}$$ **Sample average**

$$\lim_{k_a \to \infty} Q_t(a) = Q^*(a)$$

INCREMENTAL IMPLEMENTATION

The average of the first *k* rewards is $Q_k = \frac{r_1, r_2, \ldots, r_k}{k}$

We could keer a running sum and count for this…

$$Q_{k+1} = Q_k + \frac{1}{k+1}[r_{k+1} - Q_k]$$

This is a common form for update rules:

NewEstimate = OldEstimate + StepSize [Target - OldEstimate]

RANDOM EXPLORATION

- Simplest form of action selection
- Very good for exploration
- Very bad for exploitation

$$a_t = \text{random action}$$

ϵ -GREEDY EXPLORATION

- The simplest way to balance exploration and exploitation

- Greedy action selection

$$a_t = a_t^* = \underset{a}{\operatorname{argmax}}\, Q_t(a)$$

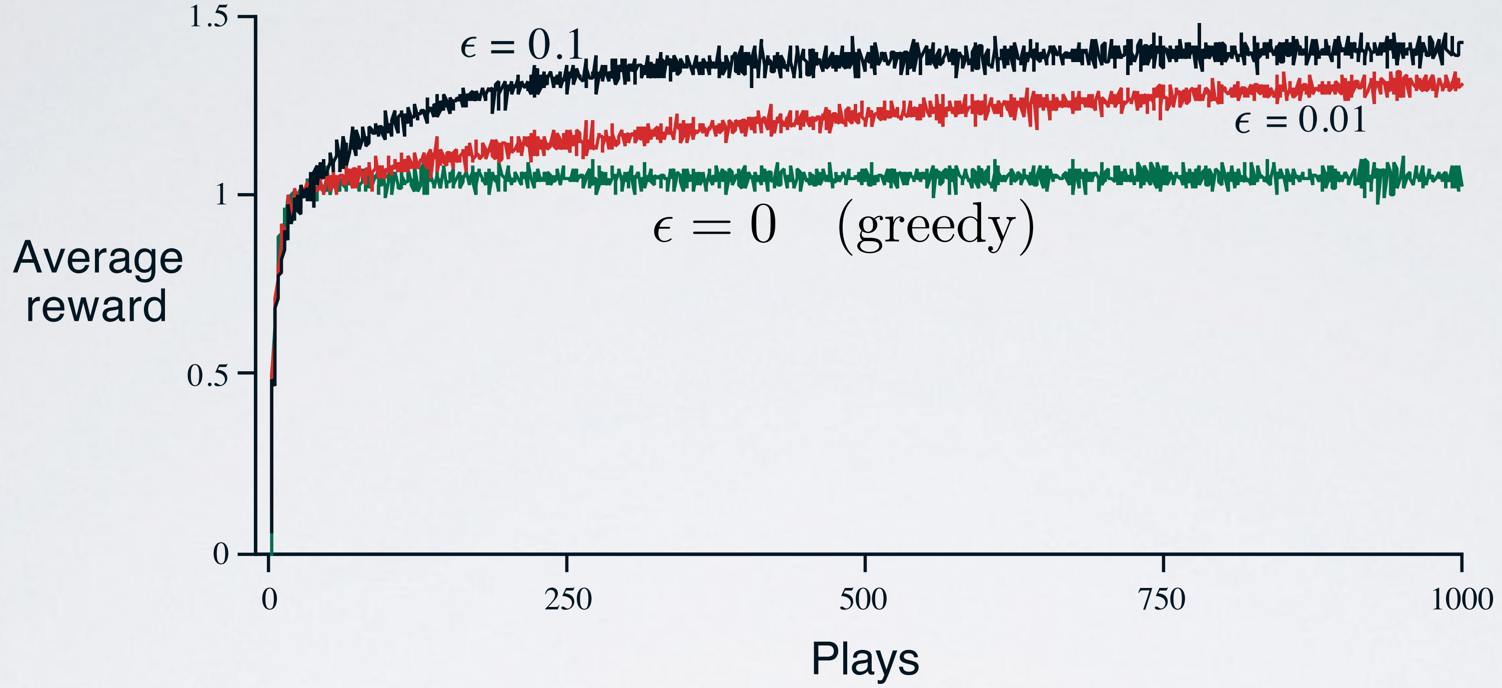
- ϵ-greedy action selection

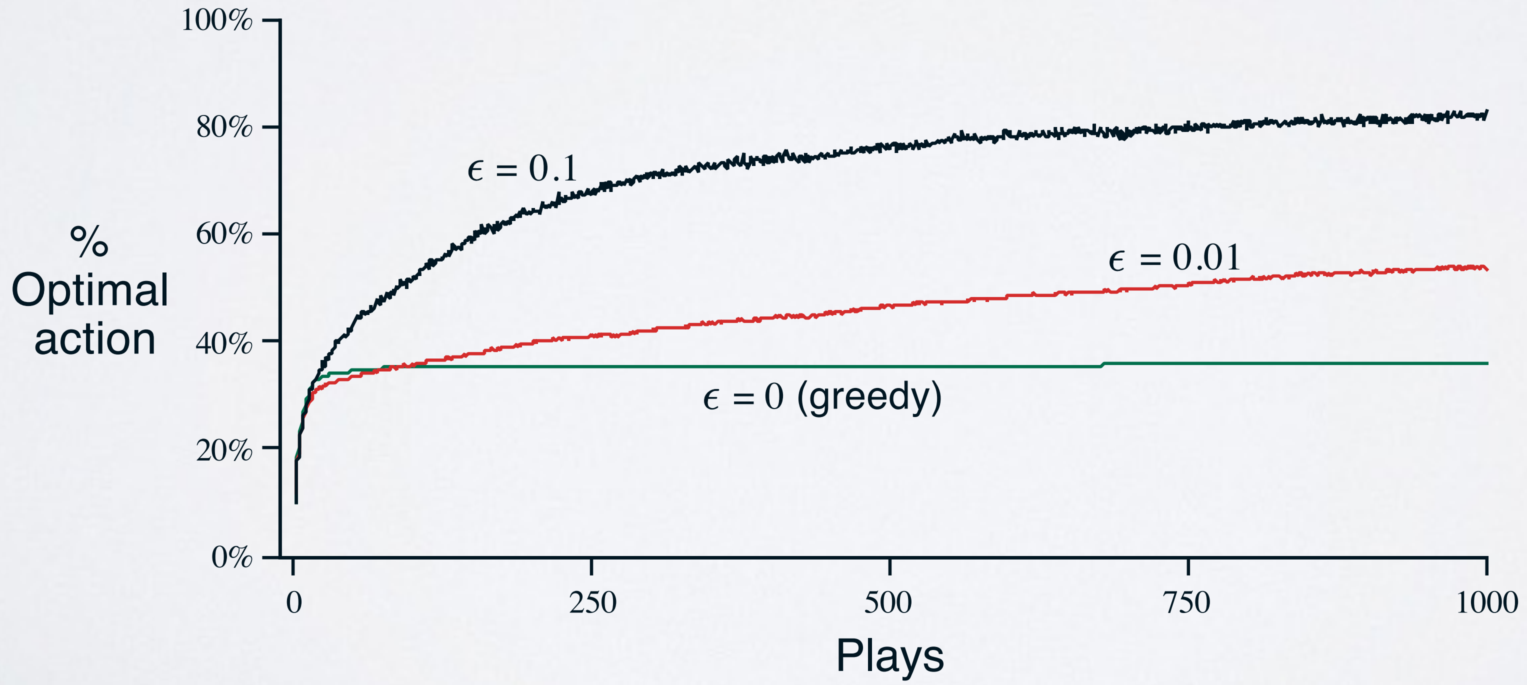
$$a_t = \begin{cases} a_t^* & \text{with probability } 1 \text{ - } \epsilon \\ \text{random action} & \text{with probability } \epsilon \end{cases}$$

10-ARMED TESTBED

- n = 10, so 10 possible actions
- Each $Q^*(a)$ is chosen randomly from a normal distribution $\eta(0, 1)$
- Each r_t is also normally distributed: $\eta(Q^*(a_t), 1)$
- 1000 plays
- Repeat the whole thing 2000 times and average the results

ϵ -GREEDY ON 10-ARMED TESTBED





SOFTMAX ACTION SELECTION

- Softmax action selection methods grade action probabilities by estimated values
- The most common softmax uses a Gibbs or Boltzmann distribution:

Choose action *a* on play *t* with probability

$$\frac{e^{Q_t(a)/\tau}}{\sum_{b=1}^{n} e^{Q_t(b)/\tau}}$$

Where τ is the computational temperature

TRACKING A NONSTATIONARY PROBLEM

- Choosing Q_k to be a sample average is appropriate in a stationary problem (i.e. $Q^*(a)$ does not change over time)

$$Q_{k+1} = Q_k + \frac{1}{k+1}[r_{k+1} - Q_k]$$

- In a non-stationary problem:

$$Q_{k+1} = Q_k + \alpha[r_{k+1} - Q_k]$$

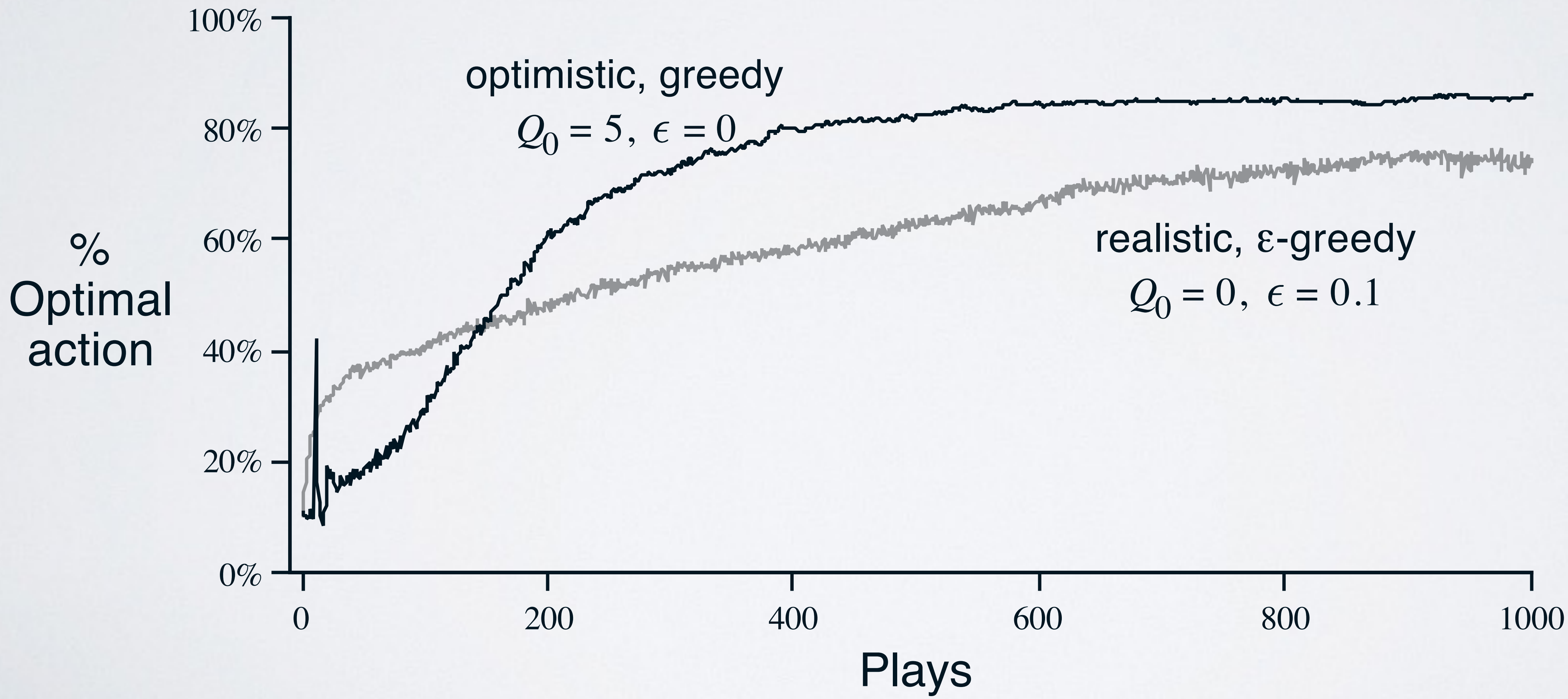
for *constant* $\alpha, 0 < \alpha \leq 1$

$$= (1-\alpha)^k Q_0 + \sum_{i=1}^{k} \alpha(1-\alpha)^{k-i} r_i$$

= exponential recency-weighted average

OPTIMISTIC INITIAL VALUES

- All methods so far depend on $Q_0(a)$, i.e. they are biased
- Suppose we initialize the action values optimistically
  $Q_0(a) = 5 \qquad \forall a$



TO REMEMBER

- N-armed bandits are single stage or stateless
- Simple methods, often used in practice
  f.i. Zone Heating
- Exploration/Exploitation dilemma

NEXT LECTURES

- Graphical games
- RL in multi-stage settings (including normal form games)
- Multi-Agent Reinforcement Learning