COMP 532
Machine Learning and BioInspired Optimization

Lecture 3 Reinforcement Learning

Dr. Shan Luo

Department of Computer Science

shan.luo@liverpool.ac.uk

Module Overview

Focus of the module: **learning agents** that interact with an **initially unknown world**.

- Parallel Problem Solving from Nature (2)
- Single-Agent RL (6)
- Multi-Agent RL (6)
- Swarm Intelligence (4)
- Deep Learning (6)
- Artificial Immune Systems (3)
- DNA Computing (3)

Today's lecture and next 5
Single Agent Reinforcement Learning

Reinforcement Learning is learning what to do – how to map situations to actions – so as to maximize a numerical reward signal.

Roots of Reinforcement Learning

- Origins from:
  - Mathematical psychology (early 10's)
  - Control theory (early 50's)

- Mathematical psychology:
  - Edward Thorndike: research on animals via puzzle boxes
  - Bush & Mosteller: developed one of the first models of learning behavior

Roots of Reinforcement Learning

Edward Thorndike:

research on animals via puzzle boxes; "Law of effect”

Thorndike"Animal intelligence: an experimental study of the associative processes in animals." The Psychological Review: Monograph Supplements 2.4 (1898): i.

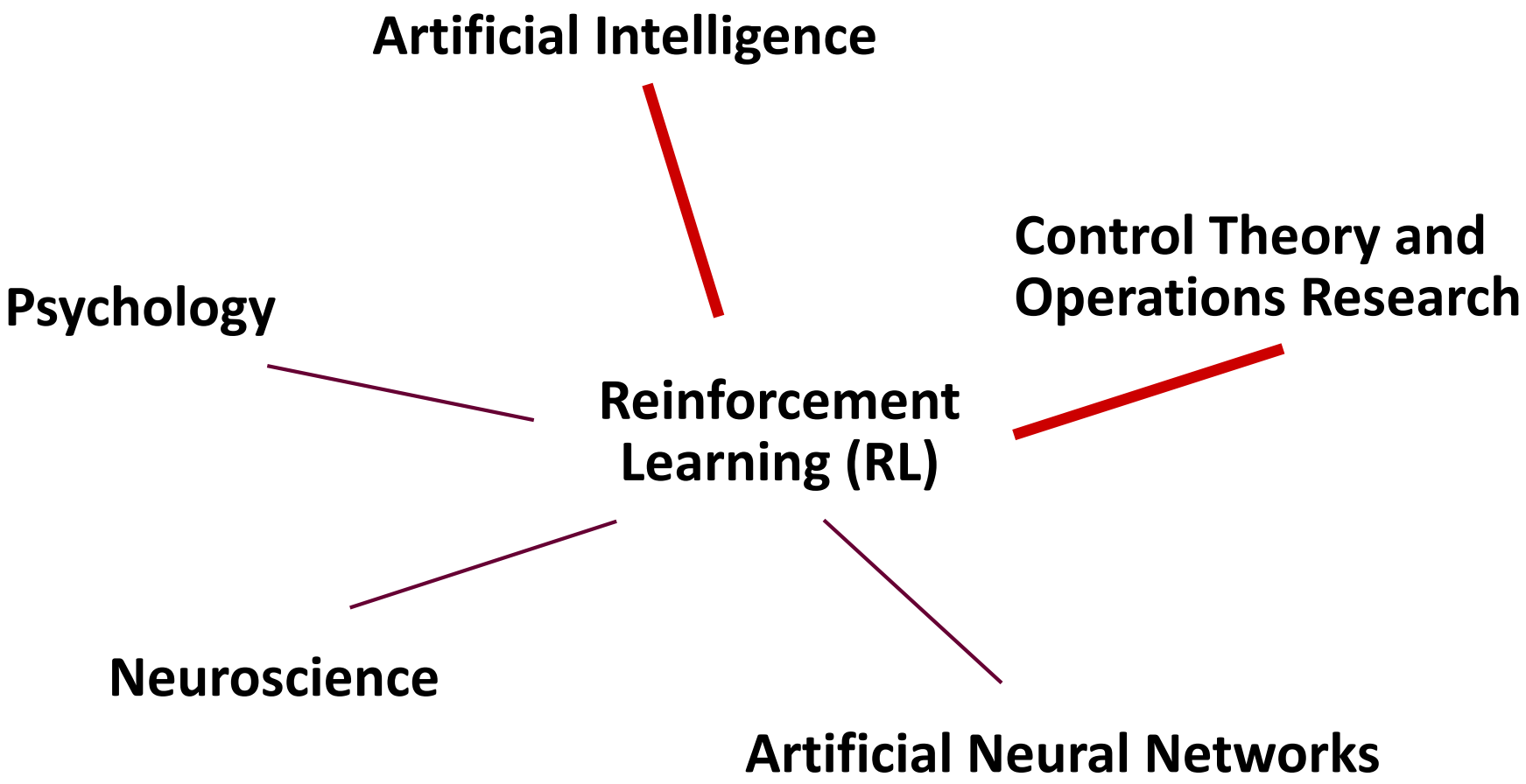
https://www.youtube.com/watch?v=BDujDOLre-8&t=68s

Roots of Reinforcement Learning

- Origins from:
  - Mathematical psychology (early 10's)
  - Control theory (early 50's)

- Control theory:
  - Richard Bellman:
    Stability theory of Differential Equations
    How to design an optimal controller?
  - Inventor Dynamic Programming:
    solving optimal control problems by solving the Bellman equations!

Roots of Reinforcement Learning

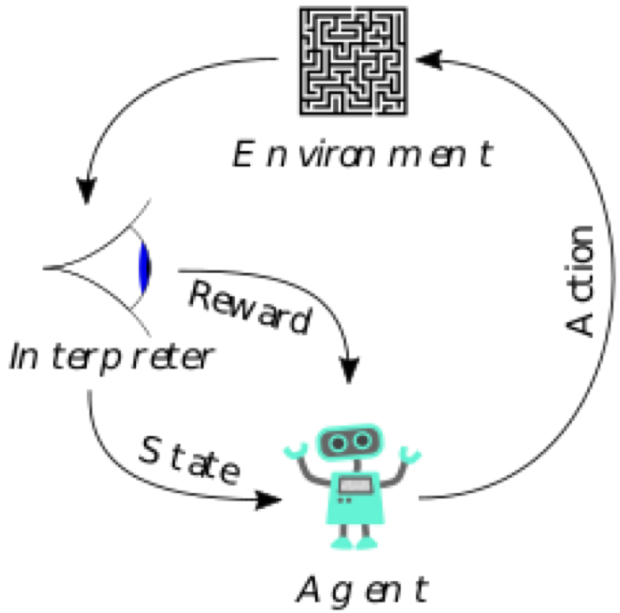


What is Reinforcement Learning?

- An approach to Artificial Intelligence
- Learning from interaction
- Goal-oriented learning
- Learning about, from, and while interacting with an external environment
- Learning what to do — **how to map situations to actions** — so as to maximize a numerical reward signal

Complete Agent

- Temporally situated
- Continual learning and planning
- Agent is to ***affect*** the environment
- Environment is stochastic and uncertain



Environment types

- Deterministic vs. stochastic
  - If the next state of the environment is (not) completely determined by the current state. (If the environment is deterministic except for the actions of other agents, then the environment is strategic)
    - e.g., taxi driving is stochastic (can't predict traffic, a tire may blow out), crossword maze is deterministic
- Fully observable vs. partially observable
  - Agent's sensors give access to the (in)complete state of the environment at each time point
    - e.g., chess is fully observable, taxi driving is partially observable

Environment types

- Episodic vs. sequential
  - The agent's experience is (not) divided into atomic "episodes". Decisions do not depend on previous decisions/actions.
    - e.g., taxi driving is sequential, maze running is episodic
- Dynamic vs. static
  - The environment is (un)changed while an agent is deliberating.
    - e.g., taxi driving is dynamic, poker is static

Environment types

- Discrete vs. continuous
  - A limited number of distinct, clearly defined states and actions.
    - e.g., taxi driving is continuous, poker is discrete
- Single agent vs. multi-agent
  - An agent operating by itself in an environment.
    - e.g., taxi driving is multi-agent, crossword puzzle is single-agent

Environment types

	Chess with a clock	Chess without a clock	Taxi driving
Fully observable	Yes	Yes	No
Deterministic	Strategic	Strategic	No
Episodic	No	No	No
Static	Semi	Yes	No
Discrete	Yes	Yes	No
Single agent	No	No	No

The real world is (of course) partially observable, stochastic, sequential, dynamic, continuous, multi-agent

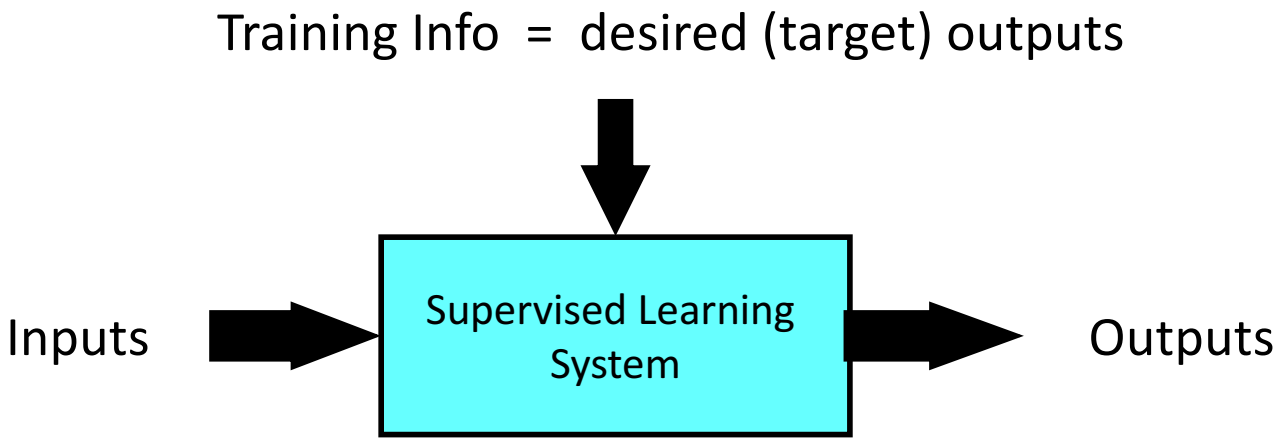
Key Features of RL

- Learner is not told which actions to take
- Trial-and-Error search
- Possibility of delayed reward
  - Sacrifice short-term gains for greater long-term gains
- The need to ***explore*** and ***exploit***
- Considers the whole problem of a goal-directed agent interacting with an uncertain environment

Examples of Reinforcement Learning

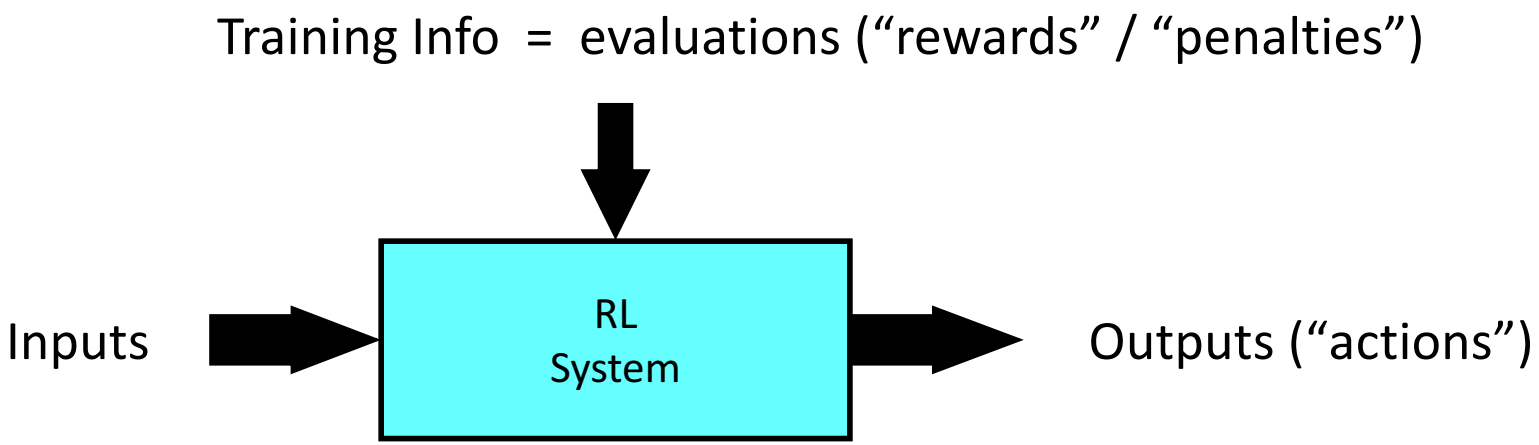
- Robocup Soccer Teams Stone & Veloso, Reidmiller et al.
  - World's best player of simulated soccer, 1999; Runner-up 2000
- Inventory Management Van Roy, Bertsekas, Lee & Tsitsiklis
  - 10-15% improvement over industry standard methods
- Dynamic Channel Assignment Singh & Bertsekas, Nie & Haykin
  - 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
- Goolge DeepMind Alpha Go
  - Beat the world's champion

Supervised Learning



Error = (target output – actual output)

Reinforcement Learning



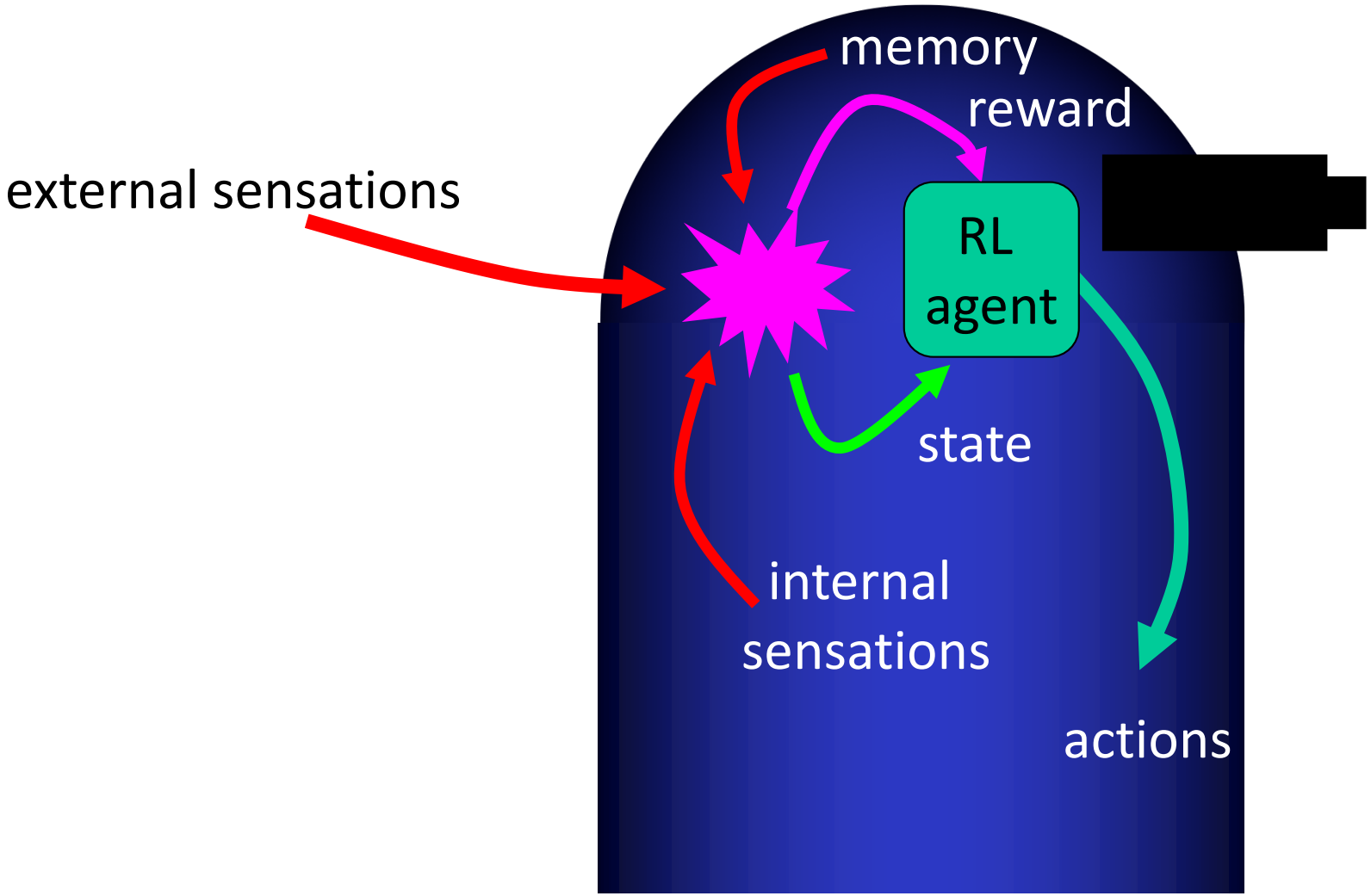
Objective: get as much reward as possible

Today

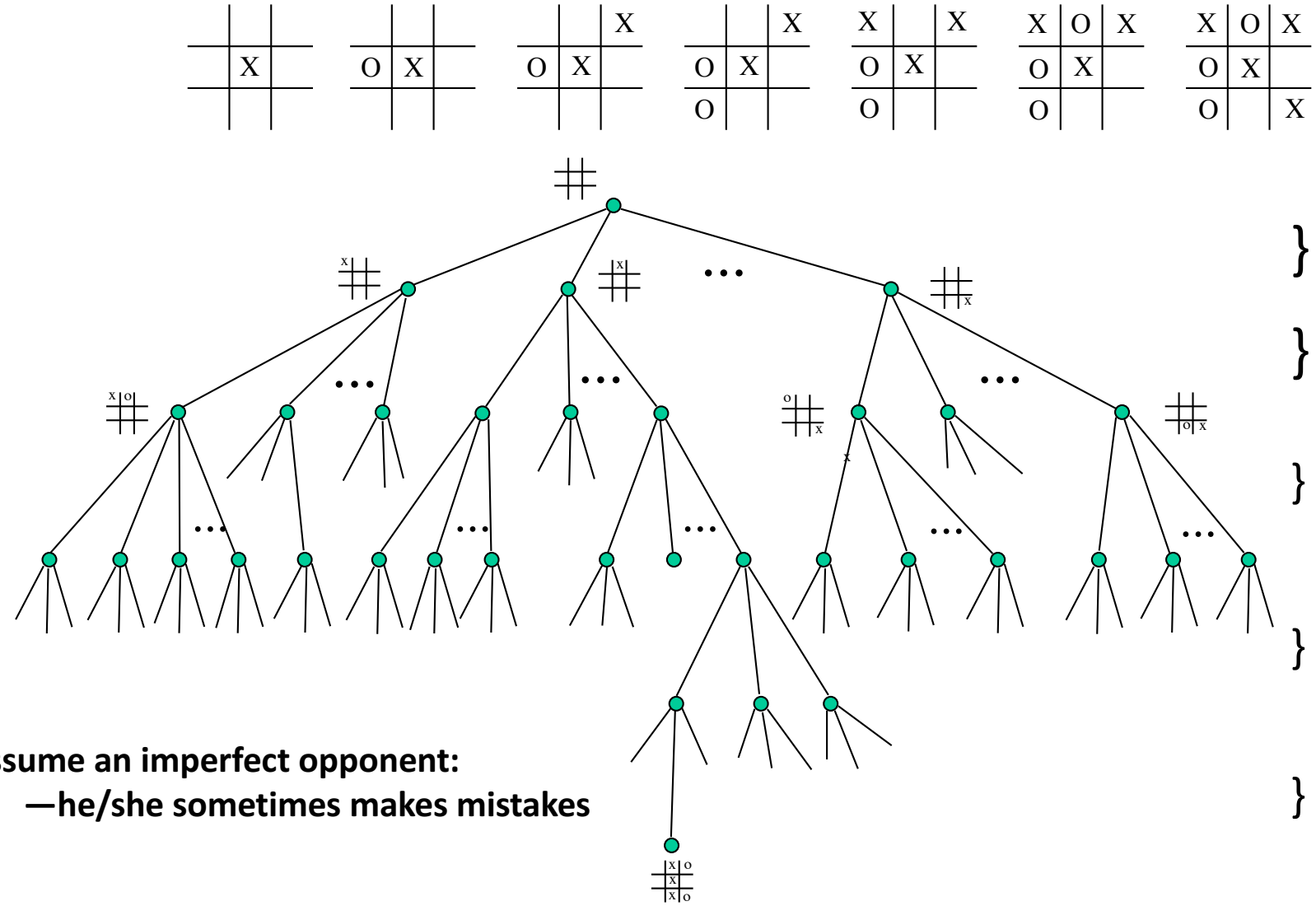
- Give an overview of the whole RL problem…
  - Before we break it up into parts to study individually
- Introduce the cast of characters
  - Experience (reward)
  - Policies
  - Value functions
  - Models of the environment
- Tic-Tac-Toe example
- Thought questions

Elements of RL

- **Policy**: what to do.
  - A policy defines the learning agent's way of behaving at a given time.
- **Reward**: what is good
  - The reward signal indicates what is good in an immediate sense.
- **Value**: what is good because it *predicts* reward
  - A value function specifies what is good in the long run.
- **Model**: what follows what
  - The behaviour of the environment, that allows inferences to be made about how the environment will behave.


memory
reward
external sensations
RL
agent
state
internal
sensations
actions

An Extended Example: Tic-Tac-Toe



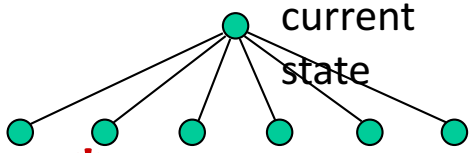
Assume an imperfect opponent:
—he/she sometimes makes mistakes

An RL Approach to 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

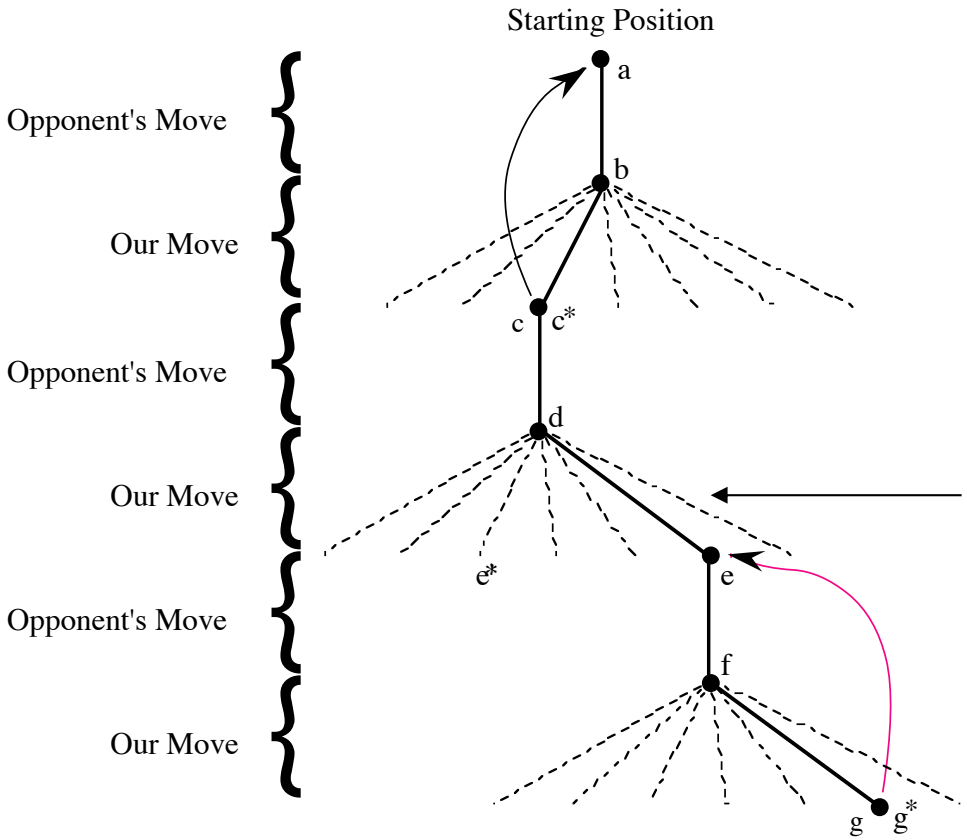
2. Now play lots of games.
To pick our moves,
look ahead one step:



Just pick the next state with the highest estimated prob. of winning — the largest $V(s)$; a ***greedy*** move.

But 10% of the time pick a move at random; an ***exploratory move***.

RL Learning Rule for Tic-Tac-Toe



"Exploratory" move

s – the state before our greedy move

s' – the state after our greedy move

We increment each $V(s)$ toward $V(s')$ – a ***backup***:

$$V(s) \leftarrow V(s) + \alpha\left[V(s') - V(s)\right]$$

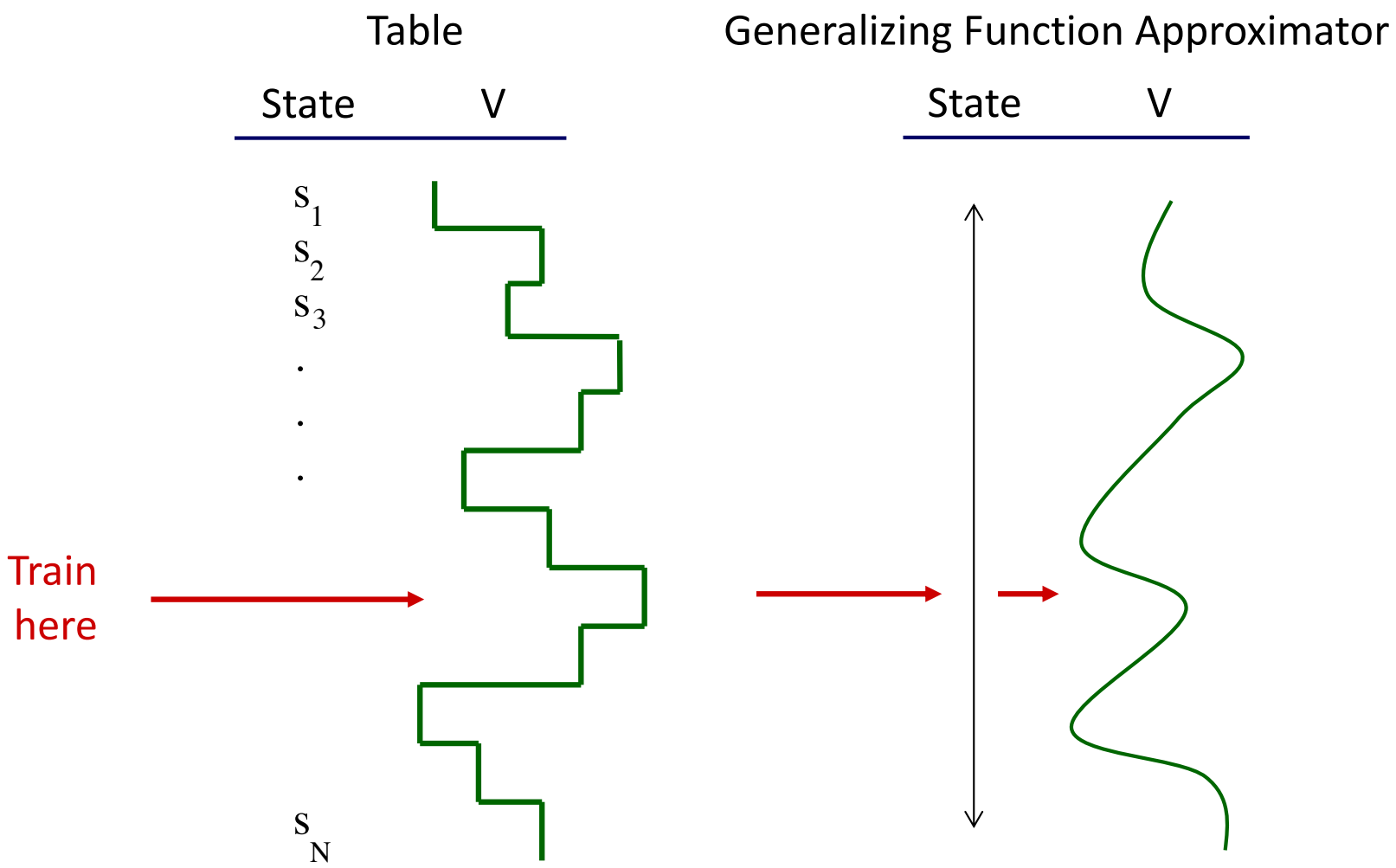
a small positive fraction, e.g., $\alpha = .1$

the ***step - size parameter***

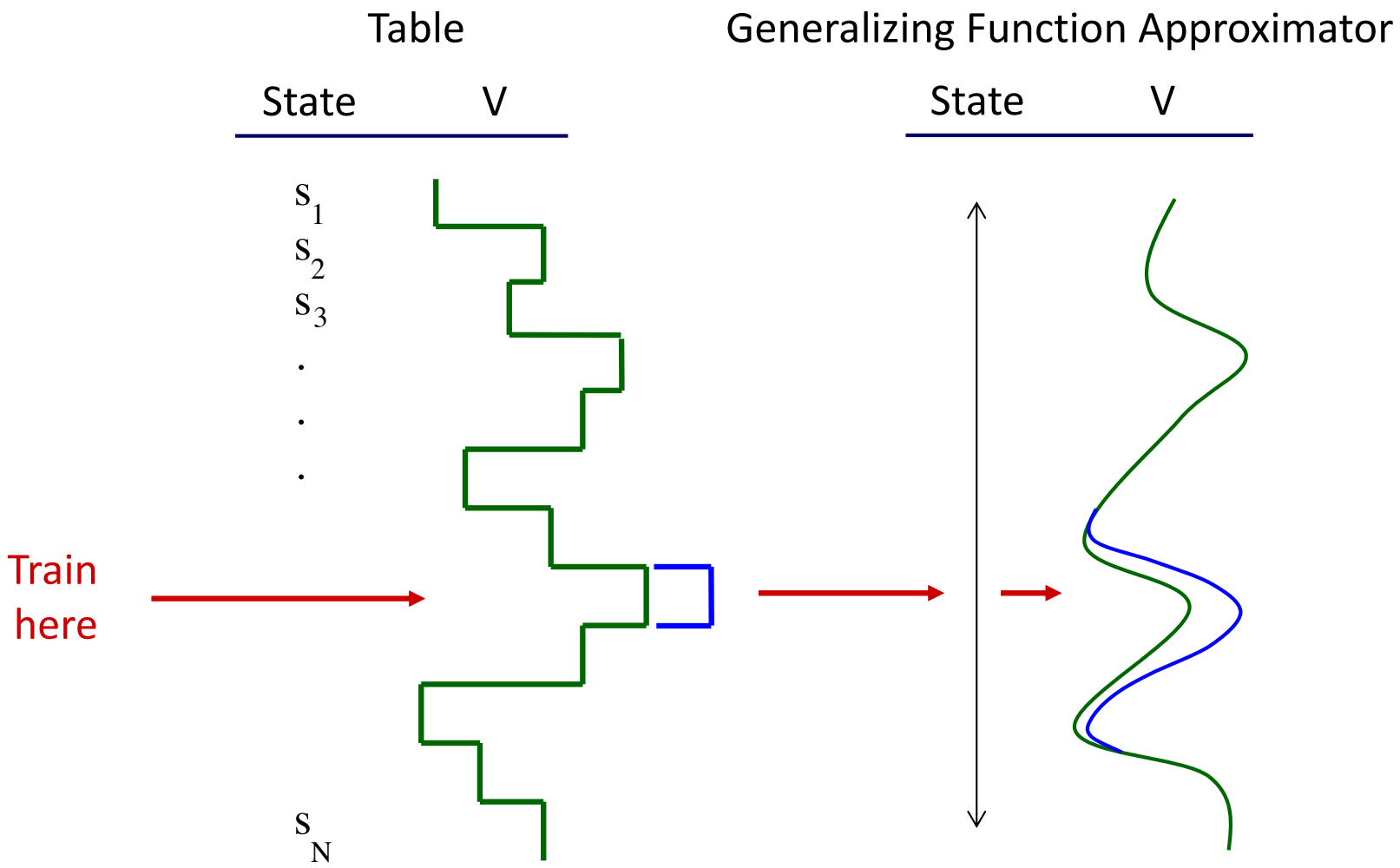
How can we improve this T.T.T. player?

- Take advantage of symmetries
  - representation/generalization
  - How might this backfire?
- Do we need “random” moves? Why?
  - Do we always need a full 10%?
- Can we learn from “random” moves?
- Can we learn offline?
  - Pre-training from self play?
  - Using learned models of opponent?
- . . .

e.g. Generalization



e.g. Generalization



How is Tic-Tac-Toe Too Easy?

- Finite, small number of states
- One-step look-ahead is always possible
- State completely observable
- . . .

Home tasks

- Practice to think in a RL way, what are the four elements? tic-tac-toe, backgammon, poker, chess, crossword puzzle, k-armed bandit, etc.

- Reinforcement learning book