A Brief Survey of Reinforcement Learning

A modern beamer theme

Giancarlo Frison

November 27, 2018

Table of contents

1. Introduction

2. Genetic Programming

3. Multi-armed Bandit

4. Markov Decision Process

5. Neural Networks in RL

6. Actor-Critic

7. Deep Q-Learning

Introduction

What is Reinforcement Learning

[5]

Reinforcement learning is an area of machine learning concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward.

Challenges in RL

RL vision is about creating systems that are capable of learning how to adapt in the real world, solely based on trial-and-error.

Challenges:

- The optimal policy must be inferred by interacting with the environment. The only learning signal the agent receives is the reward.
- Agents must deal with long-range time dependencies: Often the consequences of an action only materialise after many transitions of the environment [4].

What it is Not

Although RL can induce to an optimization, there are major differences within:

- Supervised learning

What it is Not

Although RL can induce to an optimization, there are major differences within:

- Supervised learning
- Mathematical optimization

What it is Not

Although RL can induce to an optimization, there are major differences within:

- Supervised learning
- Mathematical optimization
- Genetic programming

Genetic Programming

Genetic Programming

GP is a technique whereby computer programs are encoded as a set of genes that are then modified using an evolutionary algorithm.

- A number of chromosomes are randomly created.
- Each chromosome is evaluated through a fitness function.
- Best ones are selected, the others are disposed.
- Chromosomes could be breded among the selected for a new generation.
- Offsprings are randomly mutated.
- Repeat until the score threshold is reached [**?**].

Crossover

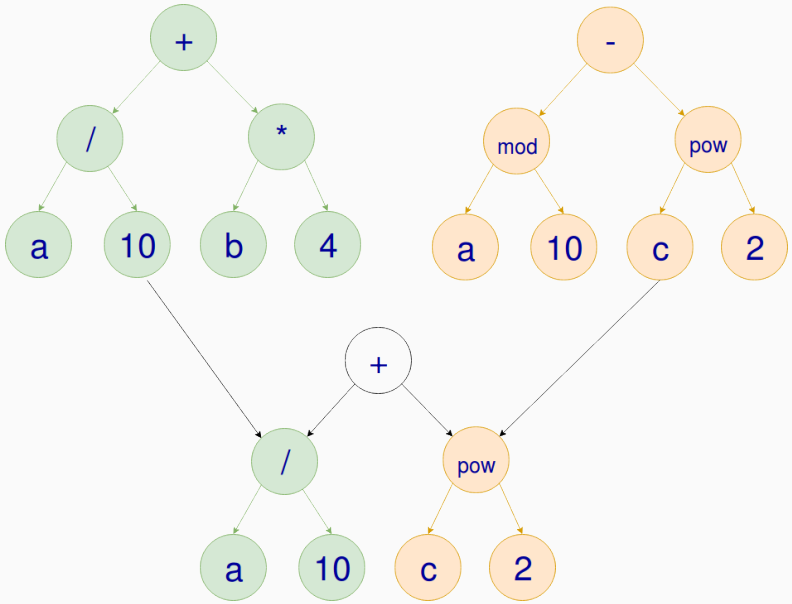


Figure 1: The “breeding” is called crossover. The chromosome (in this case an AST), is merged between two individuals for searching a better function.

Strenghts on many local minima

GP may overtake gradient-based algorithms when the solution space has many local minima

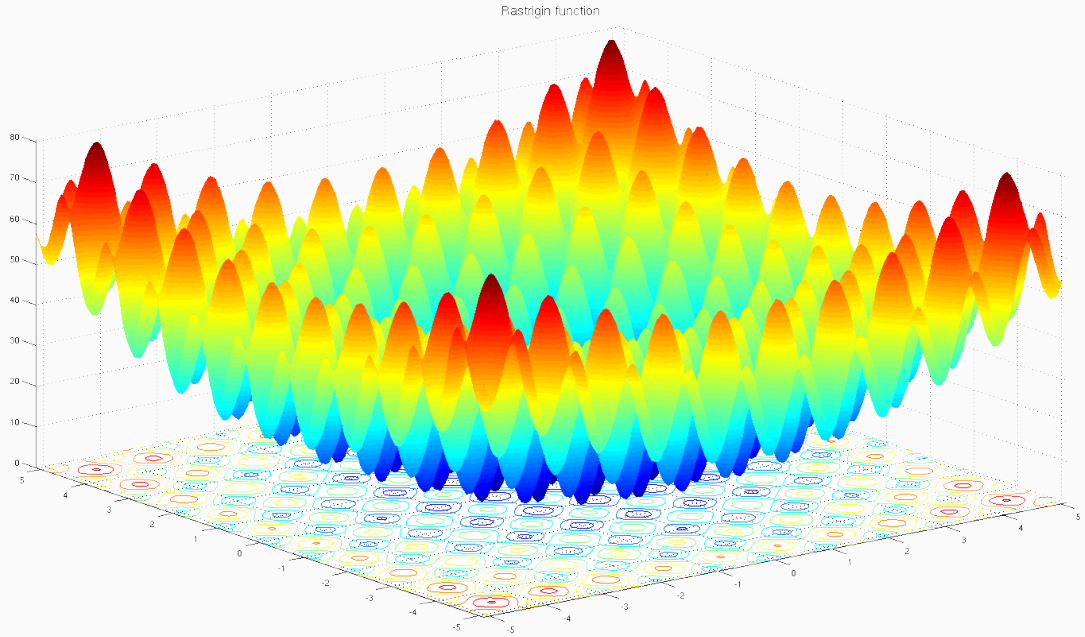


Figure 2: Rastrigin function

Strenghts on no-differentiable functions

$f(x)$ is DIFFERENTIABLE when it has always a finite derivative along the domain.

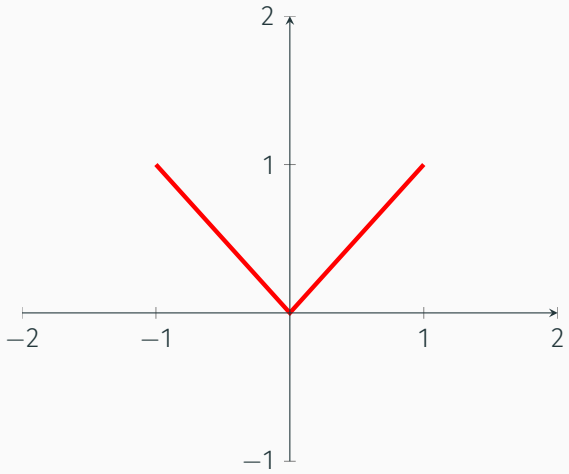


Figure 3: $f(x) = |x|$ has no derivative at $x = 0$.

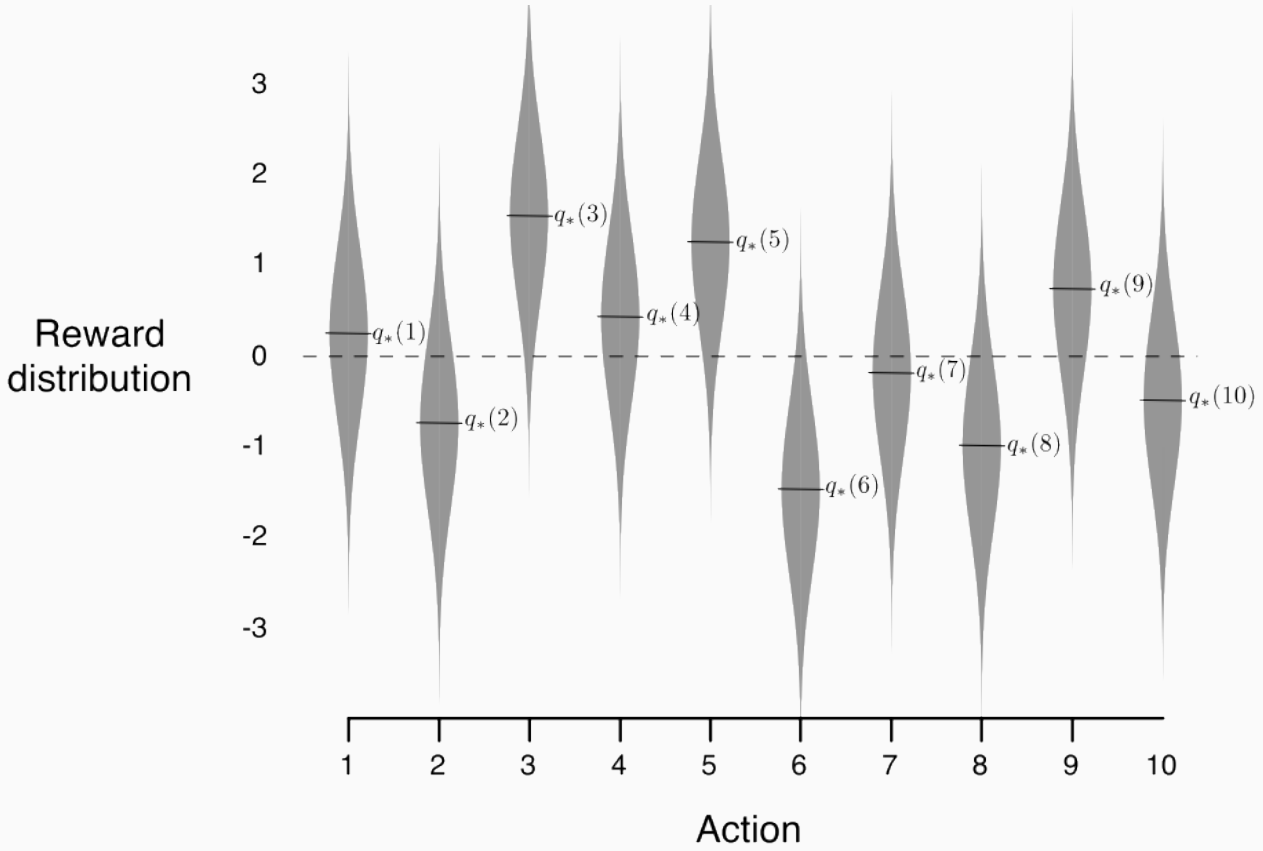
GP is not sensitive on *latent* no-differentiable functions.

Multi-armed Bandit

Multi-armed Bandit

The multi-armed bandit problem has been the subject of decades of intense study in statistics, operations research, electrical engineering, computer science, and economics [7]

Q Value's Action



An example bandit problem [4]. Obtained measures after repeated pullings with 10 arms.

Q Value's Action

$\mathbf{Q_n}$ is the estimated value of its action after n selections.

$$Q_{n+1} = \frac{R_1 + R_2 + ... + R_n}{n}$$

A more scalable formula, updates the average with incremental and small constant:

$$Q_{n+1} = Q_n + \frac{1}{n}(R_n - Q_n)$$

General expression of the badint algorithm at the fundation of RL. *Target* could be considered the reward R by now.

$$\textsf{NewEstimate} = \textsf{OldEstimate} + \textsf{StepSize}(\textsf{Target} - \textsf{OldEstimate})$$

Gambler's Dilemma

When pulled, an arm produces a random payout drawn independently of the past. Because the distribution of payouts corresponding to each arm is not listed, the player can learn it only by experimenting.

Exploitation
Earn more money by exploiting arms that yielded high payouts in the past.

Exploration
Exploring alternative arms may return higher payouts in the future.

Markov Decision Process

Definition of MDP

Figure 4: 2015 DARPA Robotics Challange [3]

Despite as in bandits, MDP formalizes the decision making (Policy π) in sequential steps, aggregated in Episodes.

Actions and States

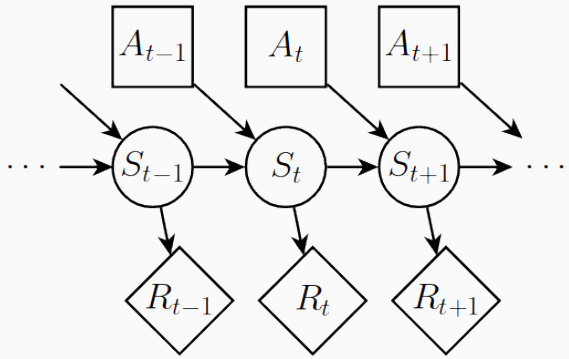


Figure 5: Model representation of MDP [2]

MDP strives to find the best π to all possible states. In Markov processes, the selected action depends **only** on current state.

How to evaluate an Agent?

Given that:

- Policy π defines a particular way of acting.
- $v_\pi(s)$ is the expected return from s following π *thereafter*.
- $q_\pi(s, a)$ is the $v_\pi(s)$ taking action a
- For maximizing future rewards, only the **max** Q_π is considered.
- γ is the rewards' discount factor.

A recursive algorithm could be indentified, known as the Bellman equation. Iteratevely computes the value Q from the terminal state:

$$Q_t(s, a) = R_{t+1} + \gamma max[v(S_{t+1})]$$

Grid World

🏁	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	🏁

- $A = up, down, right, left$
- Terminal states are the flagged boxes.
- $R_t = 0$ for terminal states.
- $R_t = -1$ for other states.

The problem is to define the best π. Value function is computed by iterative policy evaluation.

Grid World

Iteration	Calculated V_k	Policy π_k
k = 1	0 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 0	🏁 ← ? ? ↑ ? ? ? ? ? ? ↓ ? ? → 🏁
k = 2	0 -1.7 -2 -2 -1.7 -2 -2 -2 -2 -2 -2 -1.7 -2 -2 -1.7 0	🏁 ← ← ? ↑ ↵ ? ↓ ↑ ? ↱ ↓ ? → → 🏁
k = 3	0 -2.4 -2.9 -3 -2.4 -2.9 -3 -2.9 -2.9 -3 -2.9 -2.4 -3 -2.9 -2.4 0	🏁 ← ← ↰ ↑ ↵ ↰ ↓ ↑ ↱ ↱ ↓ ↳ → → 🏁

Neural Networks in RL

Beyond the Gridworld

In the Gridworld every state value v_t is stored in a table. The approach lacks scalability and is inherently limited to fairly low-dimension problems.

Figure 6: The state v_t might be a frame of a videogame [6]

Deep Reinforcement Learning

Deep learning enables RL to scale to decision-making problems that were previously intractable, i.e., settings with high-dimensional state and action spaces. [1]

- Universal function approximator
- Representation learning

Neural networks can learn the approximated value of V_s or $Q(s, a)$ for any given STATE/ACTION SPACE.

Actor-Critic

Deep Q-Learning

References i


K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath.
A brief survey of deep reinforcement learning.
arXiv preprint arXiv:1708.05866, 2017.


Giancarlo Frison.
First Steps on Evolutionary Systems.
2018.
link https://labs.cx.sap.com/2018/07/02/first-steps-on-evolutionary-systems/.


J. Zico Kolter.
Markov Decision Process.
http://www.cs.cmu.edu/afs/cs/academic/class/15780-s16/www/slides/mdps.pdf.

References ii


John Williams.
2015 darpa robotics challange.
Public domain.


P. Montague.
Reinforcement Learning: An Introduction, by Sutton, R.S. and Barto, A.G.
Trends in Cognitive Sciences, 3(9):360, 1999.


Omanomanoman.
CC BY-SA 4.0, https://commons.wikimedia.org/.


Valve Corporation.
A screenshot from the video game Dota 2.

References iii


Yamaguchi□□.
CC-BY-SA-3.0
(http://creativecommons.org/licenses/by-sa/3.0/)].

List of Figures i