Reinforcement Learning

on an example

Overview

Machine Learning

Supervised

- classification
- regression

Unsupervised

- clustering
- generative modelling

Reinforcement

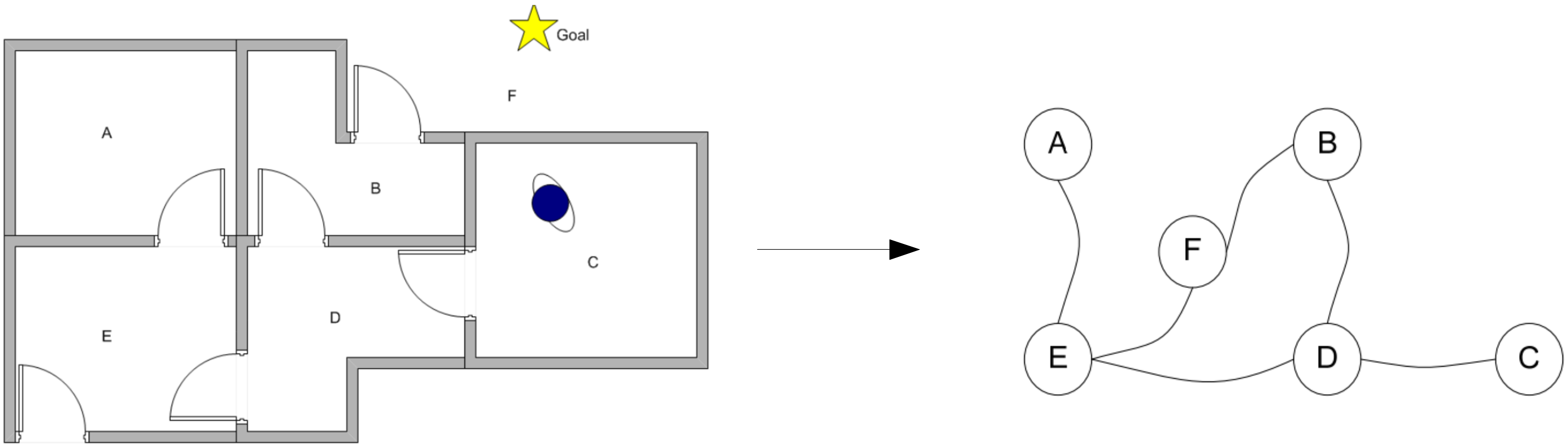
- no examples
- reward

https://deepmind.com/research/publications/playing-atari-deep-reinforcement-learning/

Theory Q-Learning:

- **exploration vs. exploitation dilemma** : example K-Armed-Bandit Problem
- **Markov-Decision-Process:**
  - s0 : Initial state
  - A(s) : all possible actions from state s
  - P(s‘|s,a) : probability to get from state s to s‘
  - R(s,a,s‘) : reward from state s to s‘
- **Q-Learning:**
  - find best policy π(s) to get best reward
  - model free, no Information about P(s‘|s,a)
  - Agent learns the value of state action pairs (Q-values)
  - **update function:** $Q(s,a) = Q(s,a) + \alpha(R(s) + \gamma \max Q(s',a') - Q(s,a))$
    - Q(s,a) : old Q-value or 0 (not defined)
    - α : 0→1 learning rate
    - R(s) : reward
    - γ : 0→1 discount factor, near 0 no reward in future, near 1 high reward
    - maxQ(s‘,a‘): maximum Q-value from state s‘ is next action a‘

Q-Learning
on an example


Goal
F
A
B
C
D
E
A
B
F
E
D
C

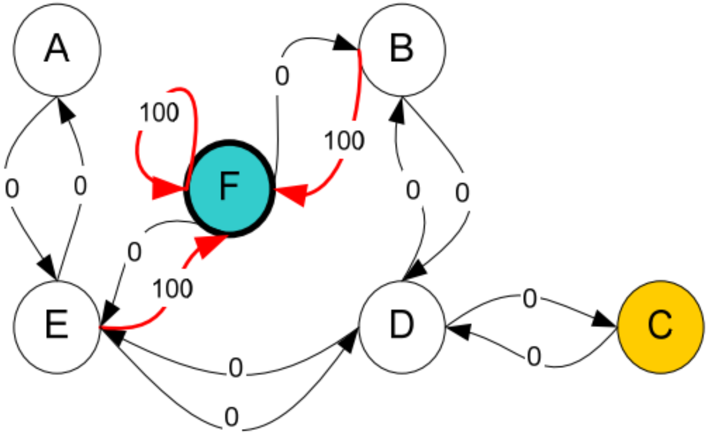
Q Learning
Given: State diagram with a goal state (represented by matrix R)
Find: Minimum path from any initial state to the goal state (represented by matrix Q)

Q Learning Algorithm goes as follow

1. Set parameter γ, and environment reward matrix **R**
2. Initialize matrix **Q** as zero matrix
3. For each episode:
   - A. Select random initial state
   - B. Do while not reach goal state
     - a. Select one among all possible actions for the current state
     - b. Using this possible action, *consider* to go to the next state
     - c. Get maximum Q value of this next state based on all possible actions
     - d. Compute
       $$\mathbf{Q}(state, action) = \mathbf{R}(state, action) + \gamma \cdot Max\left[\mathbf{Q}(next\ state, all\ actions)\right]$$
     - e. Set the next state as the current state

     End Do

   End For



$$
\mathbf{R} = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array}
\begin{array}{c}
\begin{array}{cccccc} \quad\quad\quad\quad state \backslash action \quad A & B & C & D & E & F \end{array} \\
\begin{bmatrix}
- & - & - & - & 0 & - \\
- & - & - & 0 & - & 100 \\
- & - & - & 0 & - & - \\
- & 0 & 0 & - & 0 & - \\
0 & - & - & 0 & - & 100 \\
- & 0 & - & - & 0 & 100
\end{bmatrix}
\end{array}
$$

$$
\mathbf{Q} = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array}
\begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\
\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}
$$

$$
\mathbf{R} = \begin{array}{c} state \backslash action \\ A \\ B \\ C \\ D \\ E \\ F \end{array}
\begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\
\begin{bmatrix} - & - & - & - & 0 & - \\ - & - & - & 0 & - & \boxed{100} \\ - & - & - & 0 & - & - \\ - & 0 & 0 & - & 0 & - \\ 0 & - & - & 0 & - & 100 \\ - & 0 & - & - & 0 & 100 \end{bmatrix} \end{array}
$$

Q(B,F)

$$
\longrightarrow
$$

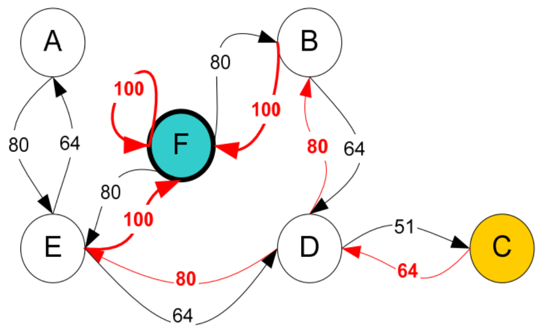
$$
\mathbf{Q} = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array}
\begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\
\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}
$$

By random selection, we select to go to F as our action

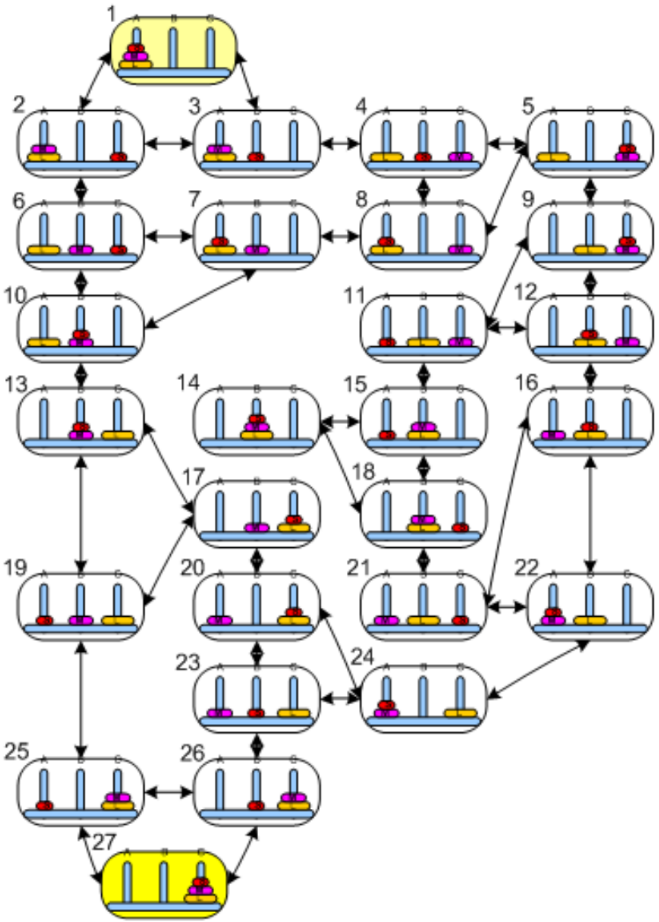
$$
\mathbf{Q}(state, action) = \mathbf{R}(state, action) + \gamma \cdot Max\left[\mathbf{Q}(next\ state, all\ actions)\right]
$$

$$
\mathbf{Q}(B,F) = \mathbf{R}(B,F) + 0.8 \cdot Max\left\{\mathbf{Q}(F,B), \mathbf{Q}(F,E), \mathbf{Q}(F,F)\right\} = 100 + 0.8 \cdot 0 = 100
$$

$$
\mathbf{Q} = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array}
\begin{array}{c} \textit{state} \backslash \textit{action} \quad A \quad B \quad C \quad D \quad E \quad F \\
\begin{bmatrix}
- & - & - & - & 400 & - \\
- & - & - & 320 & - & 500 \\
- & - & - & 320 & - & - \\
- & 400 & 256 & - & 400 & - \\
320 & - & - & 320 & - & 500 \\
- & 400 & - & - & 400 & 500
\end{bmatrix}
\end{array}
$$



Towers of Hanoi


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27

```
-     setRMatrix();
-     hanoiGame.setLambda(lambda);
-     for ( int i = 0; i < rounds; i++ ) {
          String res = hanoiGame.learn();
          System.out.print(".");
      //          System.out.print(res);
      }
-     System.out.print(hanoiGame.bestMoves());
```

https://github.com/sky4walk/HanoiTowersSolver

R-Matrix:

-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0	-1	0	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0	0	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	0	-1	0	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	0	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	0	-1	-1	-1	-1	0	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	0	-1	0	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	0	0	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	0	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	0	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	0	-1	0	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	0	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	0	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	0	0	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0	-1	-1	-1	0	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	0	-1	-1	0	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	0	-1	0	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0	0	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	0	100
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0	-1	100
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	100

Q-Matrix:

0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	3	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	3	0	1	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	3	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	3	3	0	0	0	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	3	0	0	6	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	3	0	3	0	0	0	0	12	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	6	0	0	0	0	0	0	12	0	25	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	3	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	3	0	0	1	0	0	0	3	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0	6	0	25	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	12	0	0	0	0	0	25	12	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	6	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	12	0	0	0	12	0	0	0	0	0	0	0	50	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	0	0	0	0	0	25	12	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	0	3	0	0	0	6	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	0	0	0	0	6	0	0	12	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	0	0	0	12	0	50	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	0	6	25	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	25	0	0	0	0	0	0	50	100
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	25	0	50	0	100
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Ablauf:

```
1)
  +     |     |
 -+-    |     |
--+--   |     |
-----------------
```

```
2)
  |     |     |
 -+-    |     |
--+--   |     +
-----------------
```

```
6)
  |     |     |
  |     |     |
--+--  -+-    +
-----------------
```

```
10)
  |     |     |
  |     +     |
--+--  -+-    |
-----------------
```

```
13)
  |     |     |
  |     +     |
  |    -+-  --+--
-----------------
```

```
19)
  |     |     |
  |     |     |
  +    -+-  --+--
-----------------
```

```
25)
  |     |     |
  |     |    -+-
  +     |   --+--
-----------------
```

```
27)
  |     |     +
  |     |    -+-
  |     |   --+--
-----------------
```