St. Pölten University of Applied Sciences

/informatik & security

Reinforcement Learning

Studiengang: Data Science and Business Analytics

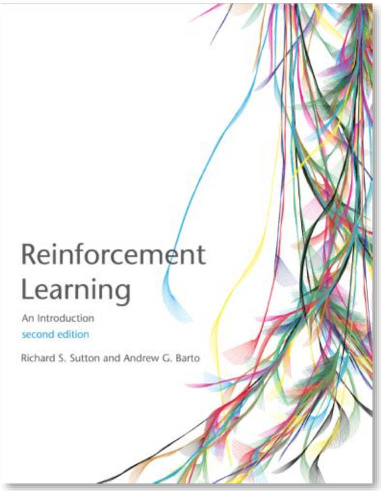
Vortragender: Sebastian Eresheim

Ort, Datum: St. Pölten, 05.12.2021

Organizational

Lecture is based on:

- *Reinforcement Learning: An Introduction*
  Richard Sutton**,** Andrew Barto
  www.incompleteideas.net/book/the-book.html



Overview

Machine Learning

Supervised Learning

Unsupervised Learning

Reinforcement Learning

Overview

Reinforcement Learning is motviated by learning:

- like a human
- via interaction (Trial and Error)
- „learning **what** to do!“ not „how to do it“.

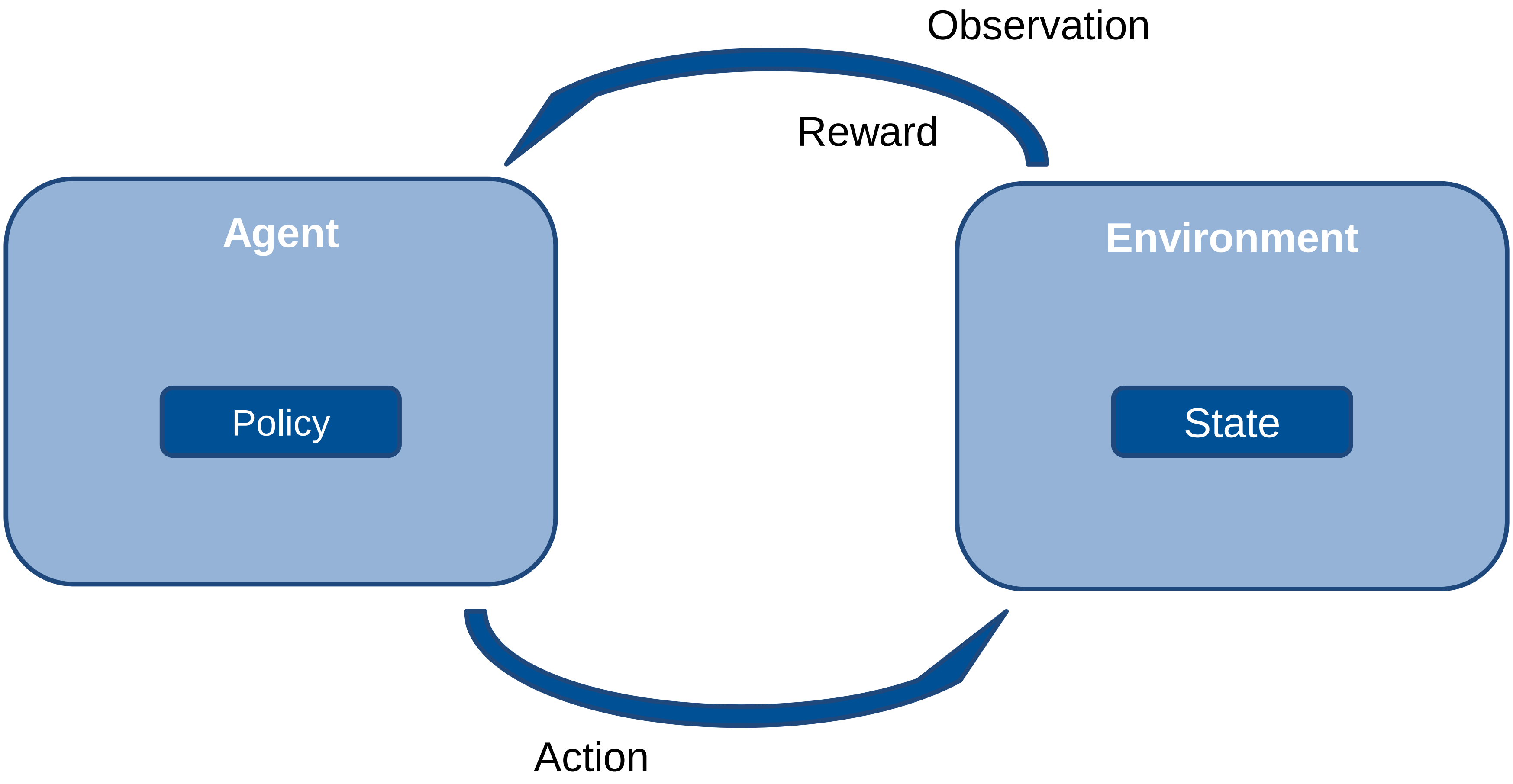
Overview

In reinforcement learning an agent learns how to map actions to observations in order to maximise a numerical reward, which it receives from an environment.

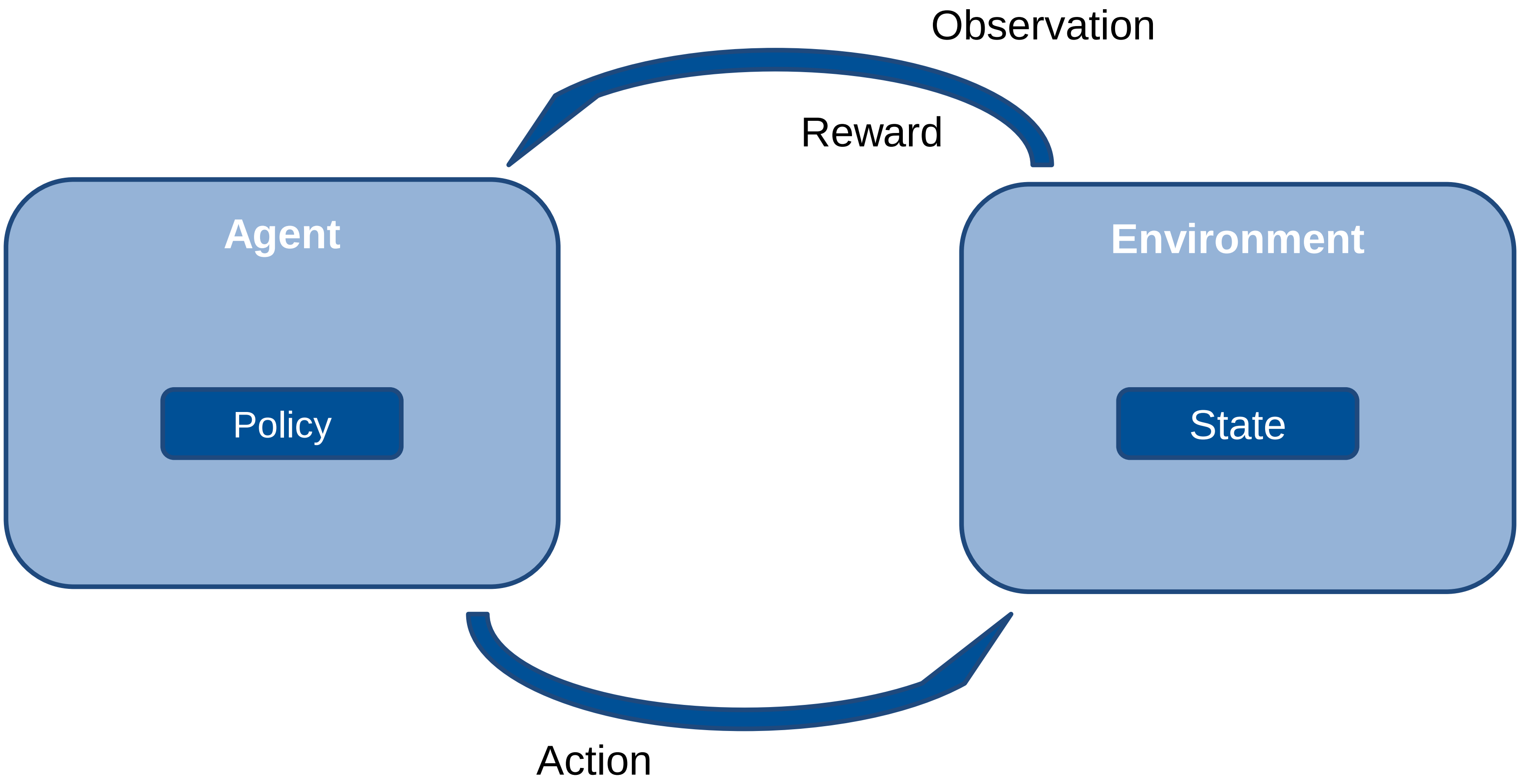
Overview

In reinforcement learning an **agent** learns how to map actions to observations in order to maximise a numerical reward, which it receives from an **environment**.

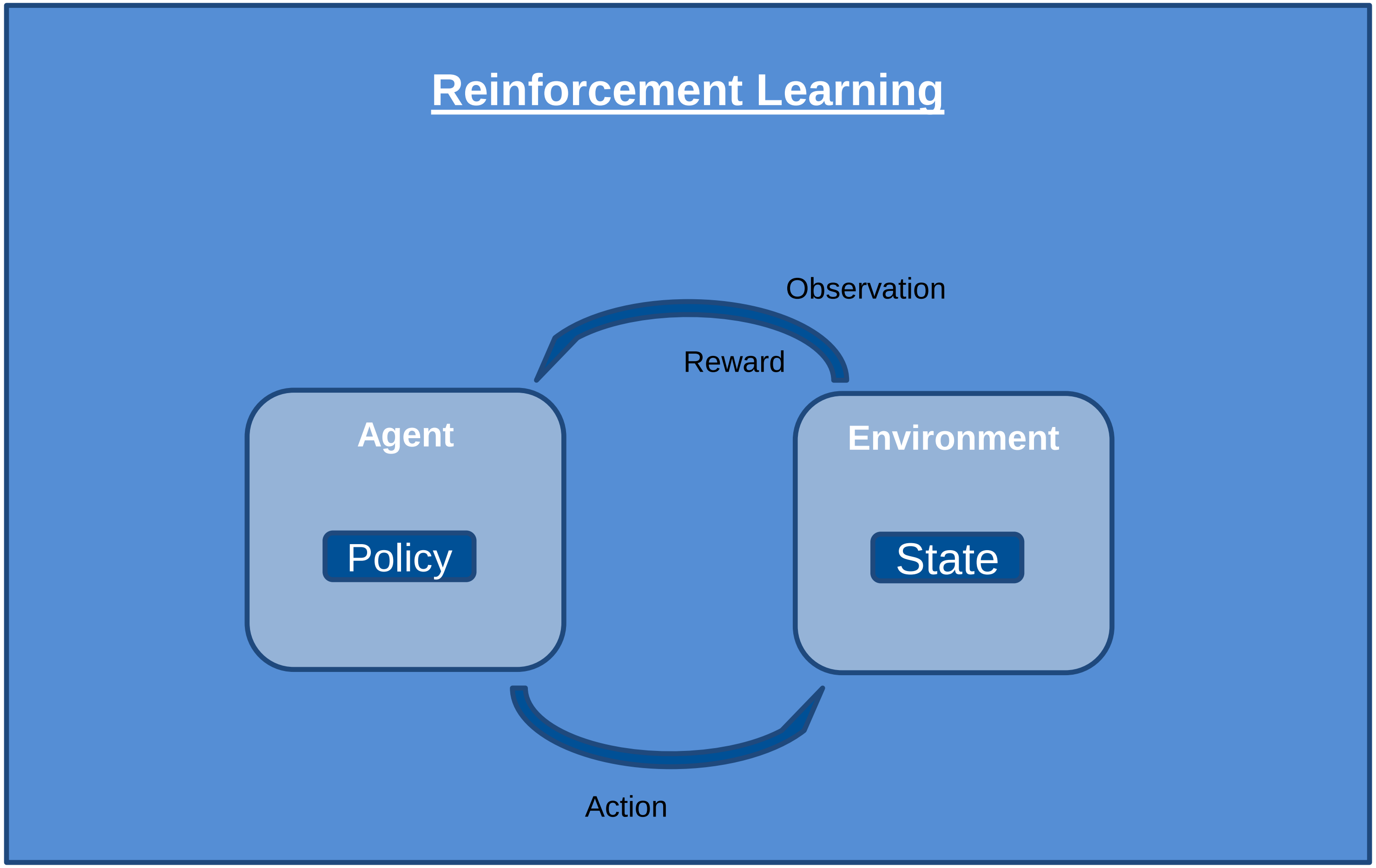
Preliminaries



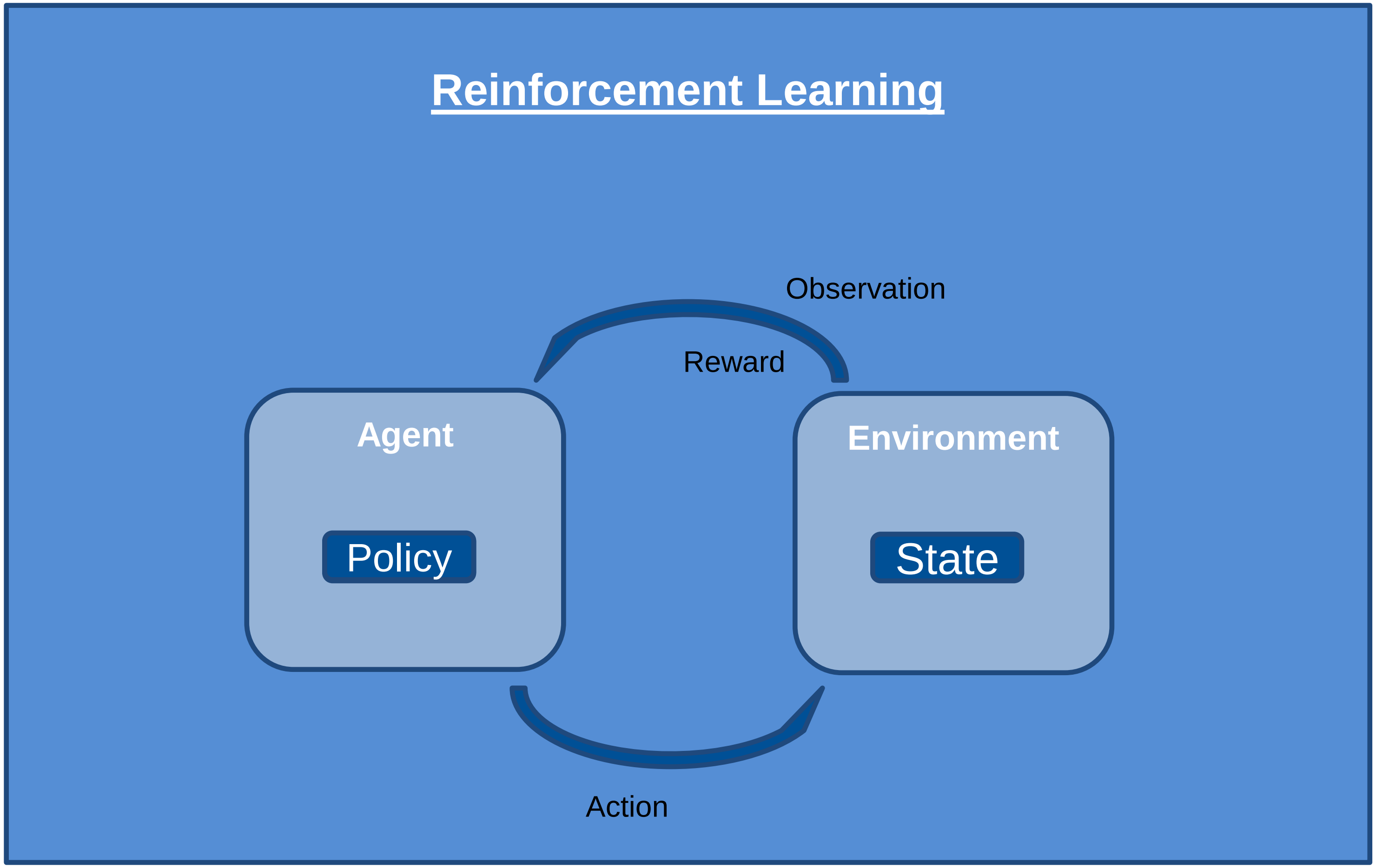
Preliminaries



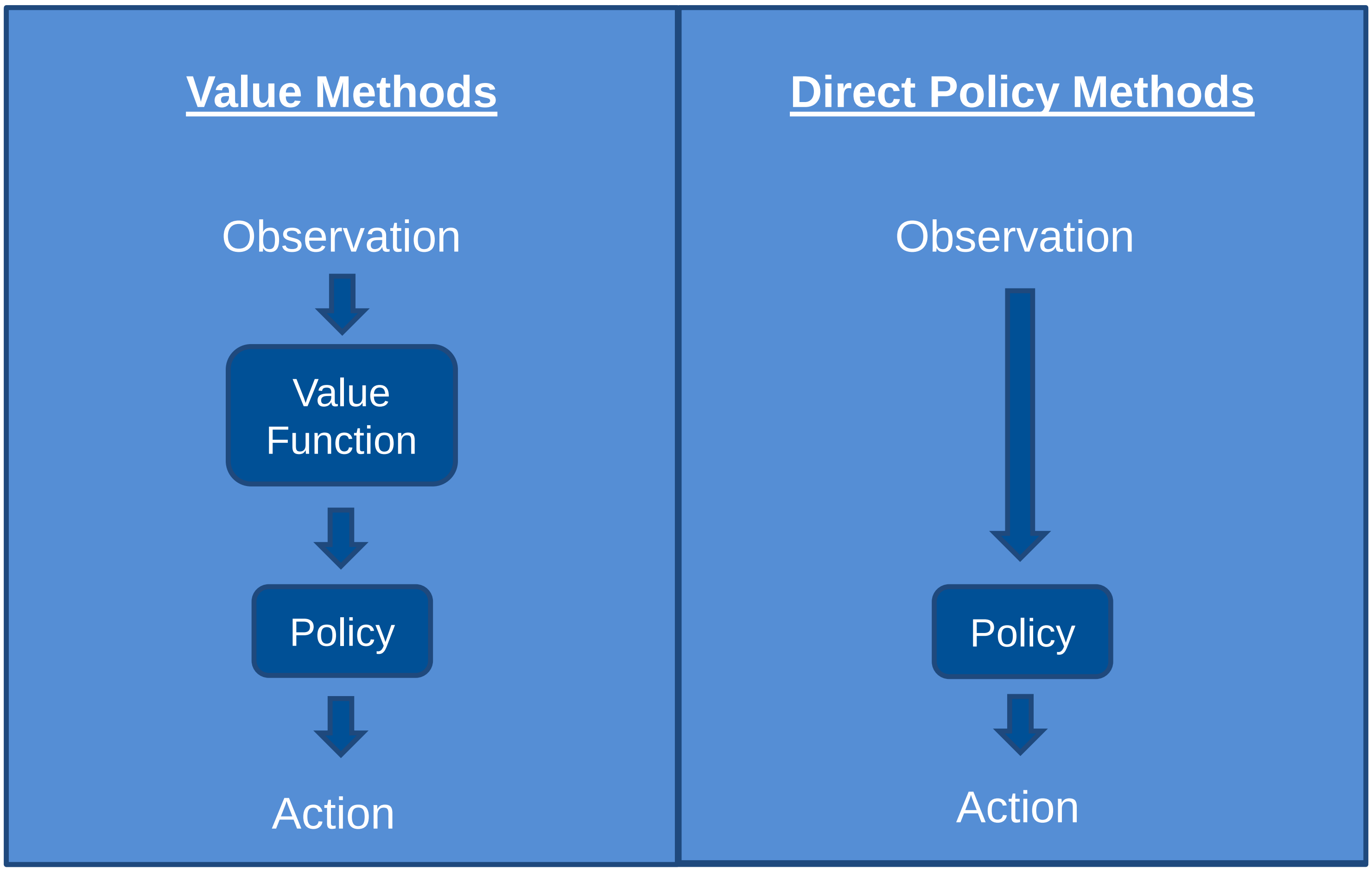
Preliminaries



Preliminaries



Preliminaries



Preliminaries

/informatik & security /fh/// st.pölten

Tabular Methods

Approximate Solution Methods

Direct Policy Methods

Overview - Problems

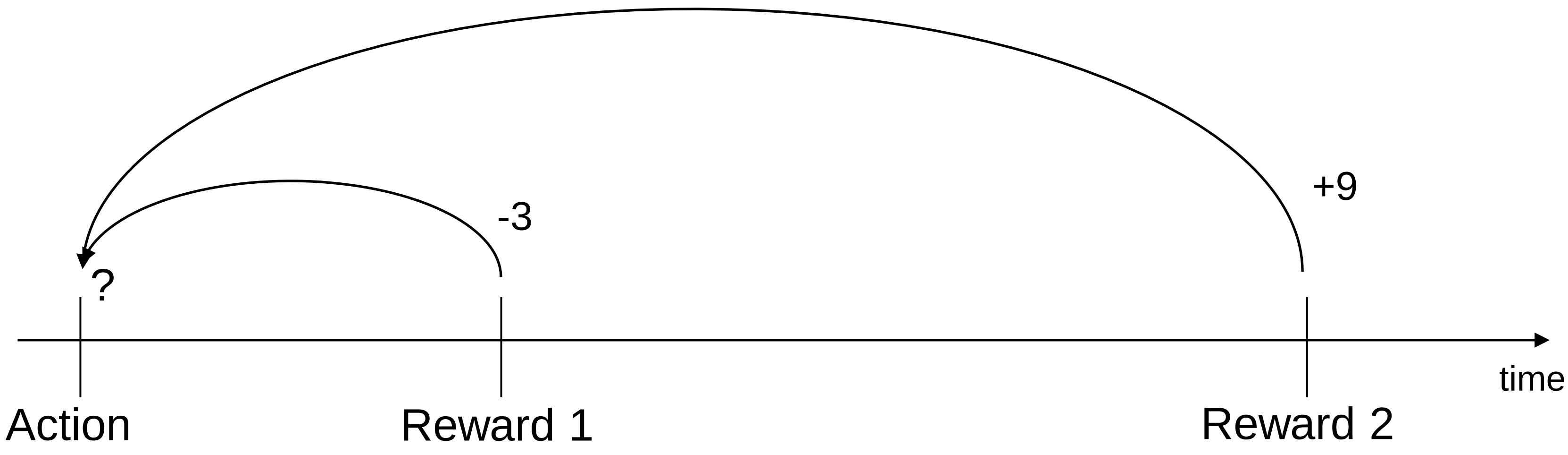
Exploitation

vs

Exploration

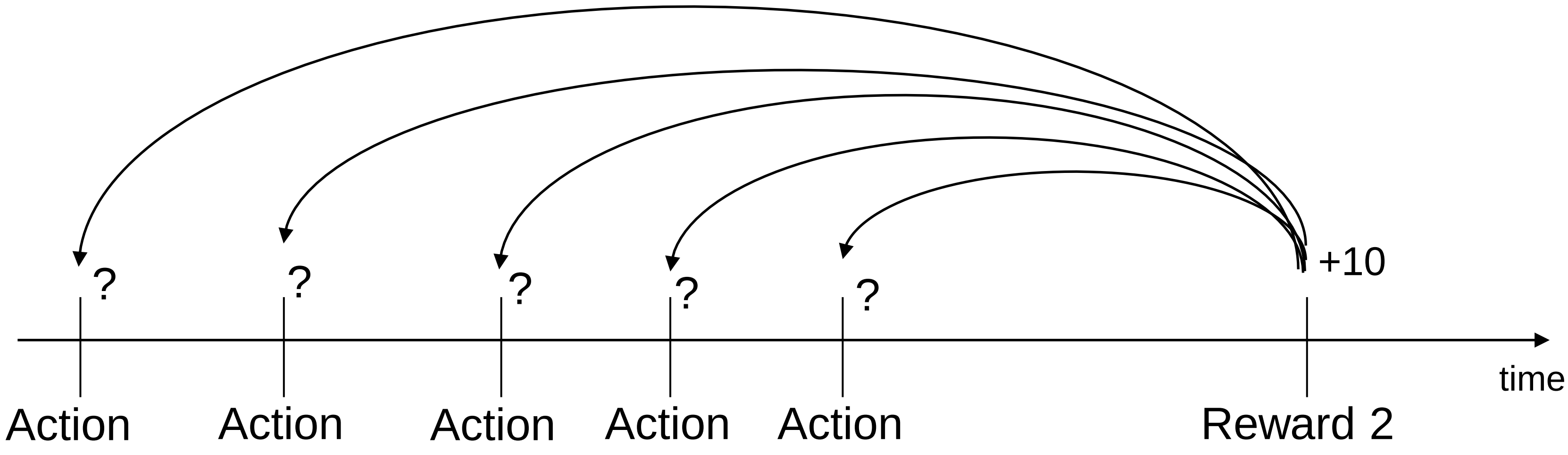
Overview - Problems

Delayed Reward



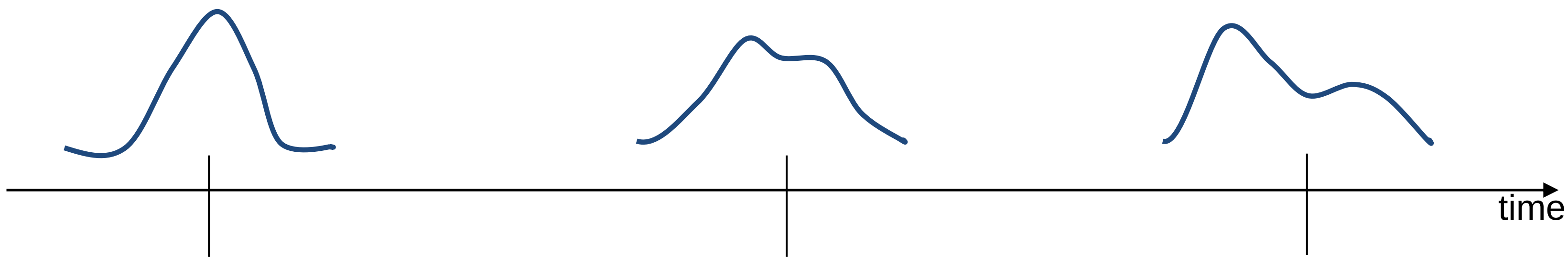
Overview - Problems

Credit Assignment Problem



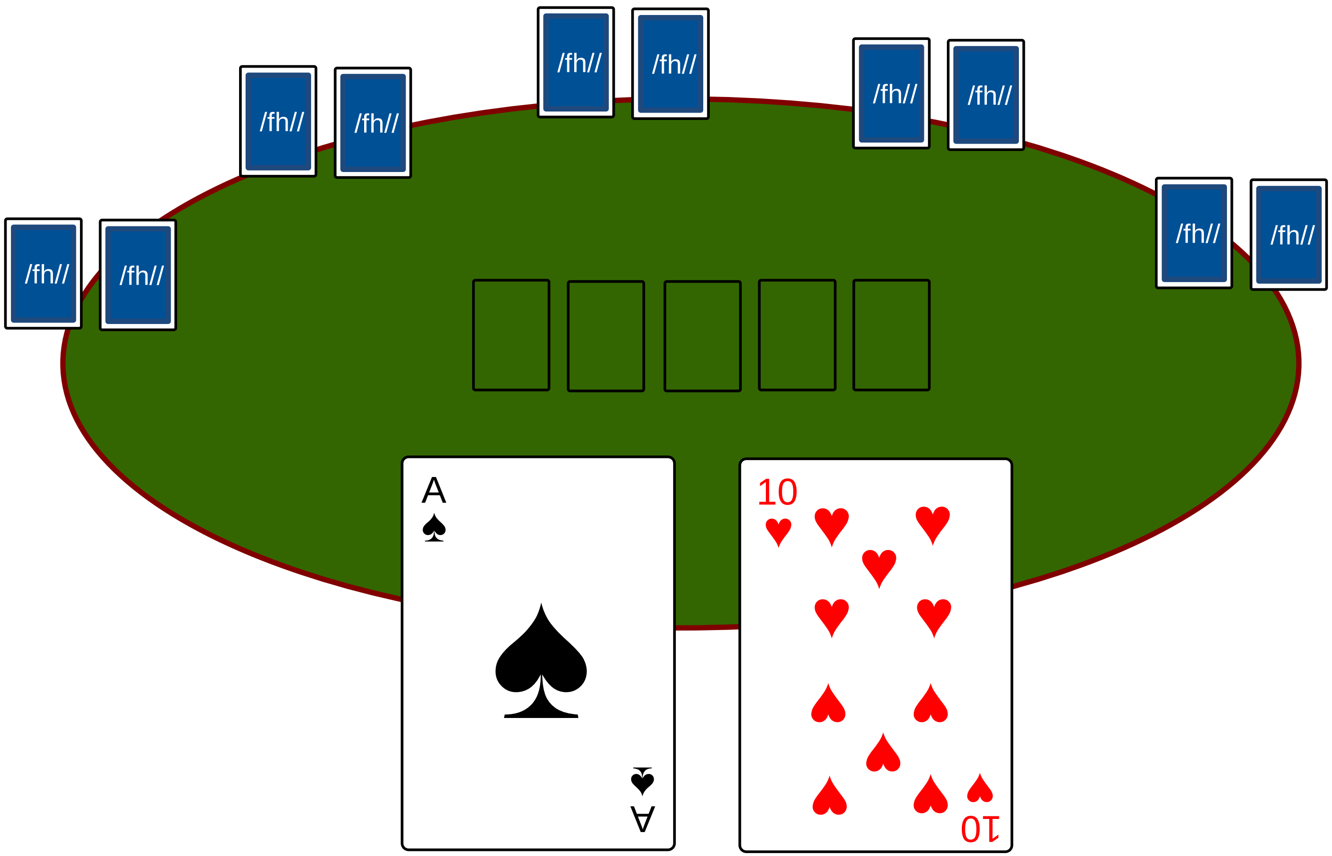
Overview - Problems

Non-Stationarity



Overview - Problems

Partial Observability



Overview - Problems

Multi-Agent

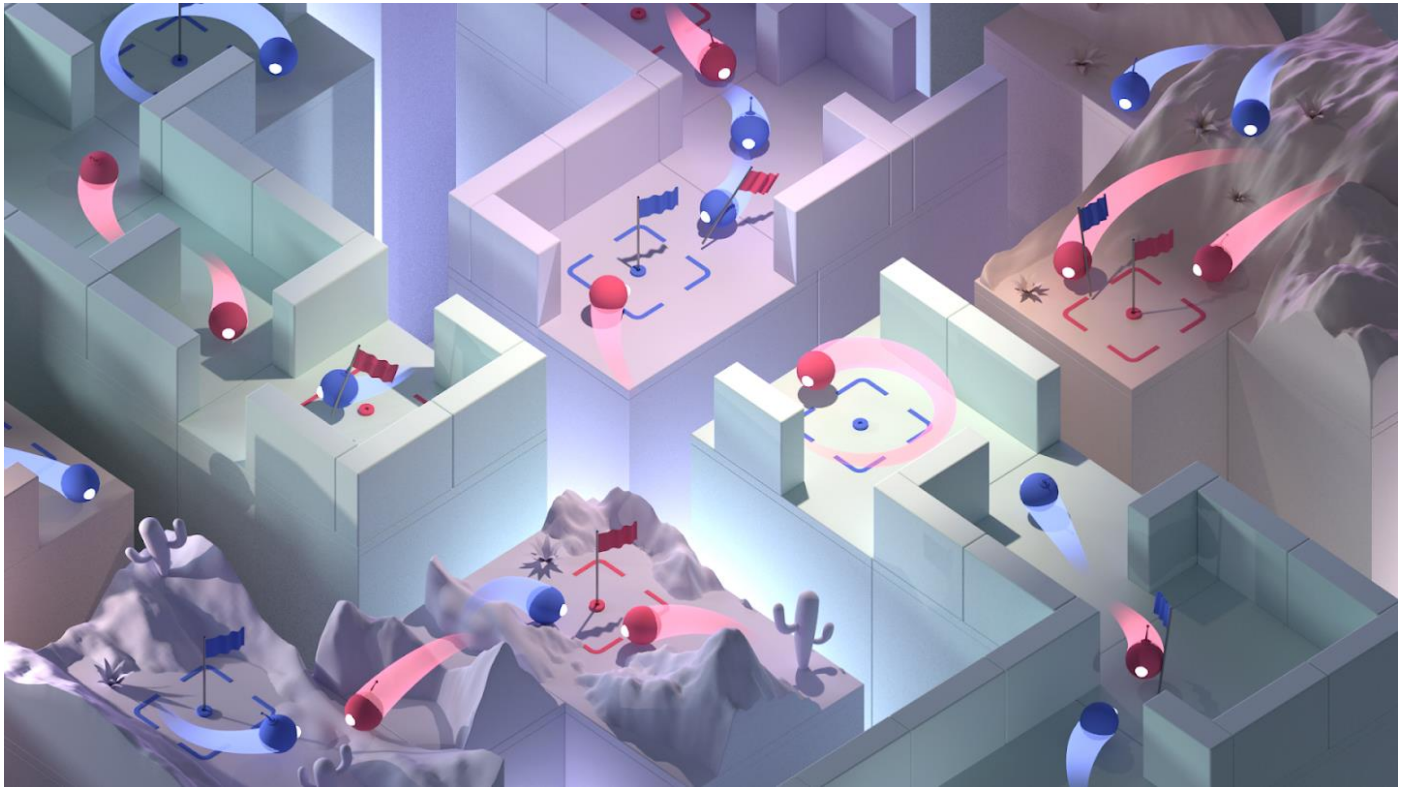
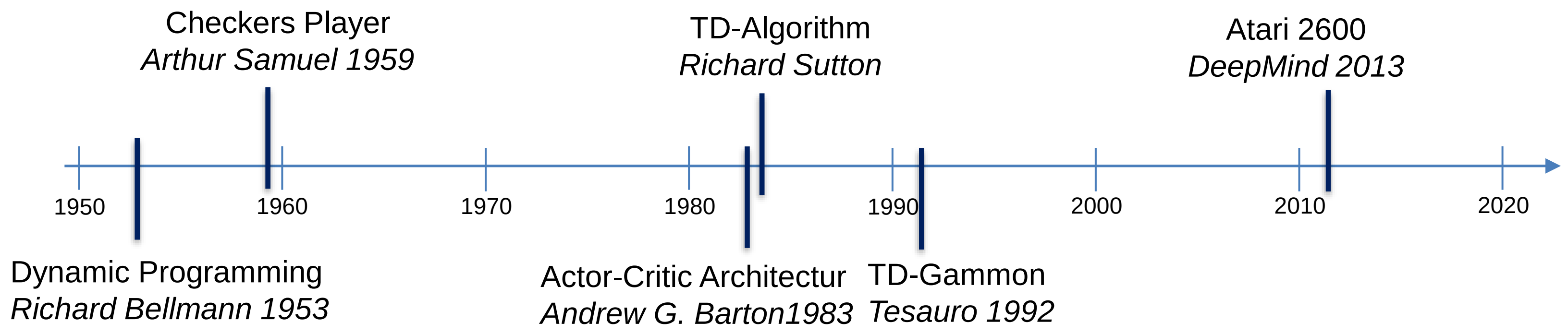
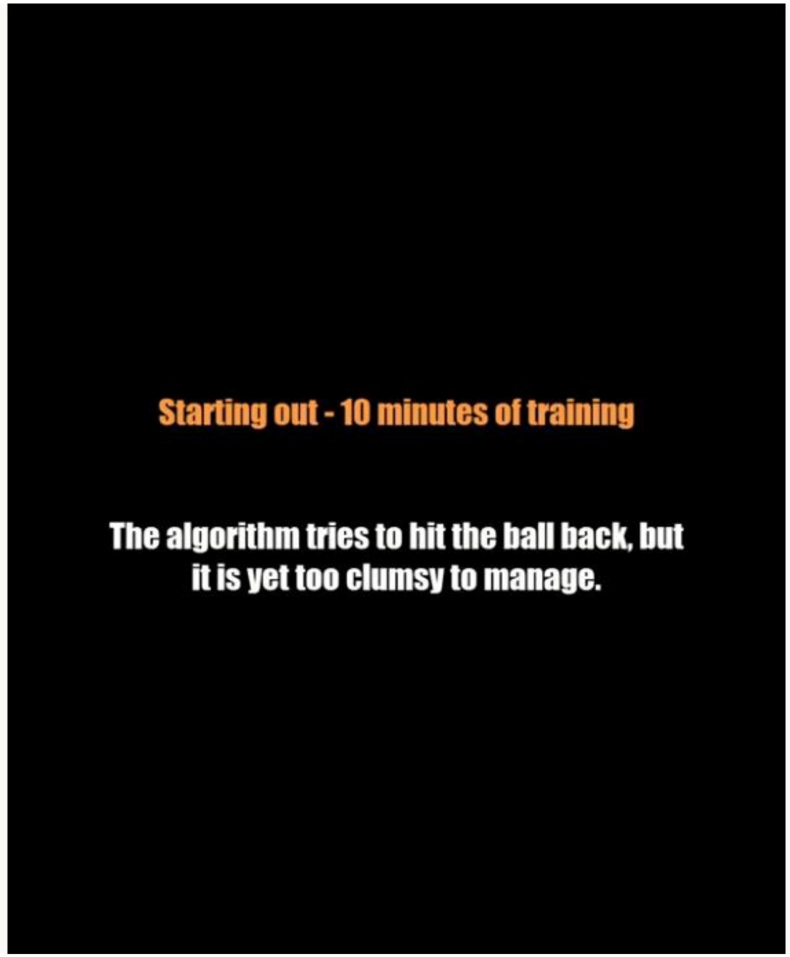


Image: https://deepmind.com/blog/article/capture-the-flag-science

Milestones of RL

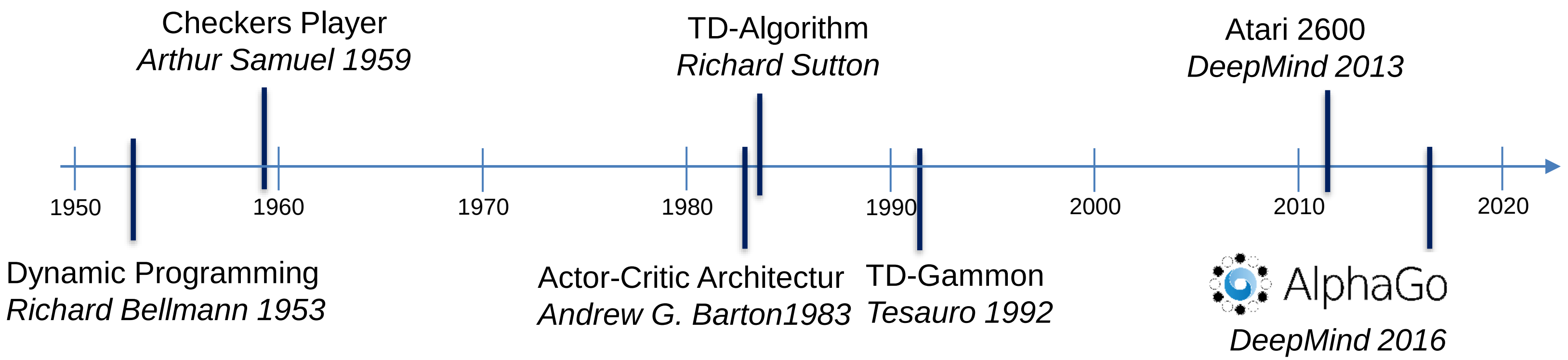


Atari Games



Source: Two Minute Papers https://www.youtube.com/watch?v=V1eYniJ0Rnk

Milestones of RL



AlphaGo – AlphaZero

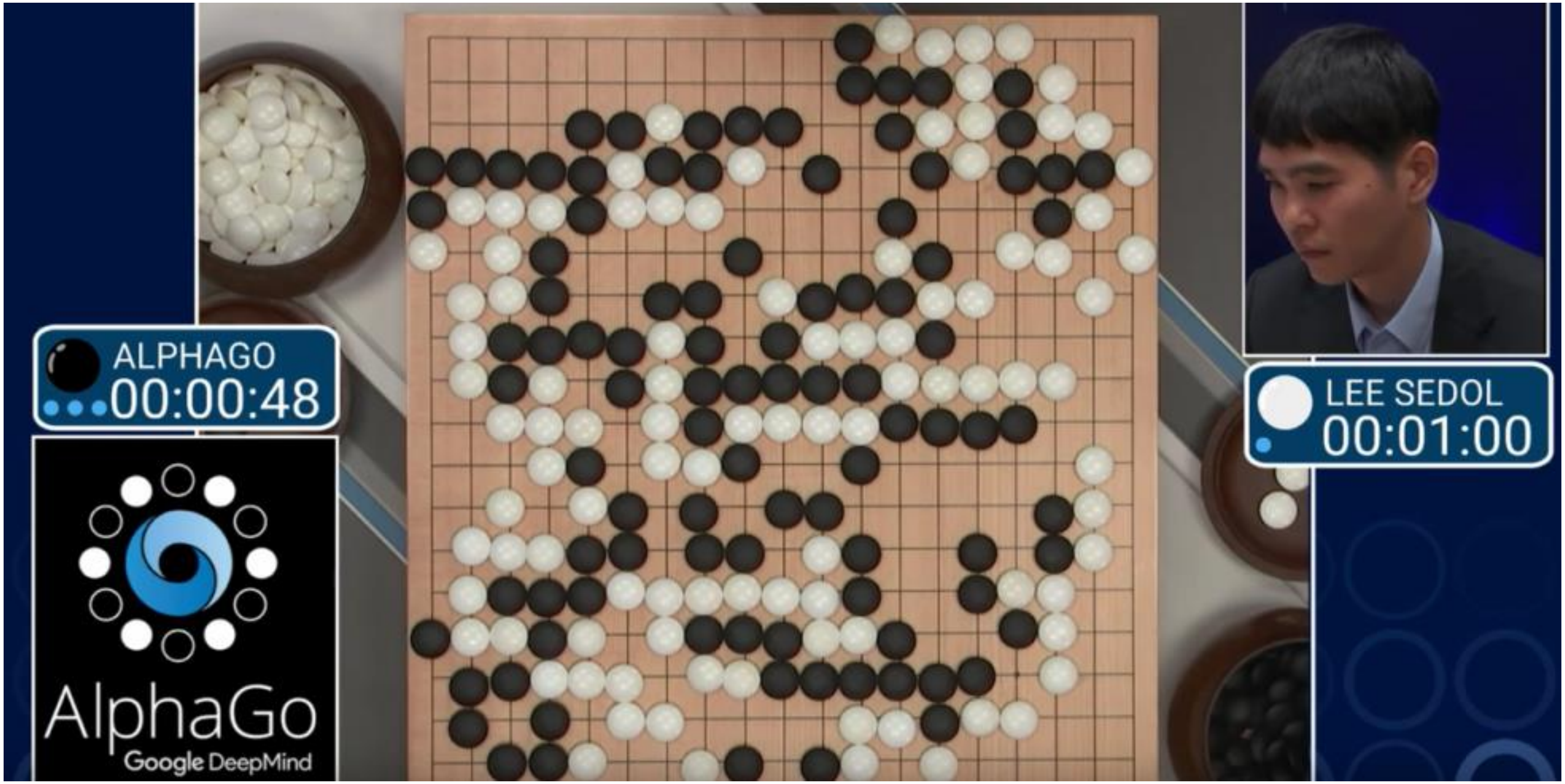
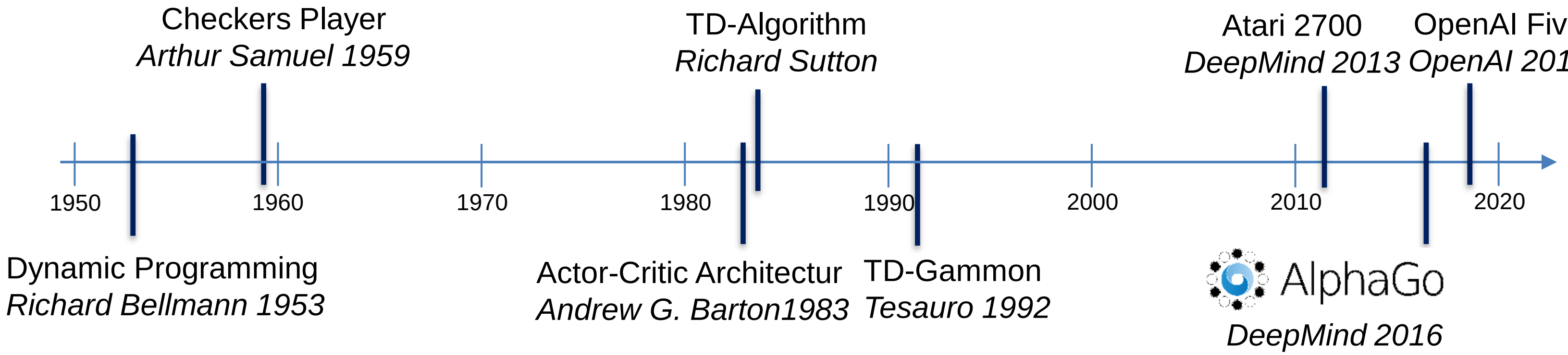


Image: https://cdn-images-1.medium.com/max/1600/0*J2HH4TDqGCuKbufS.png

Milestones of RL

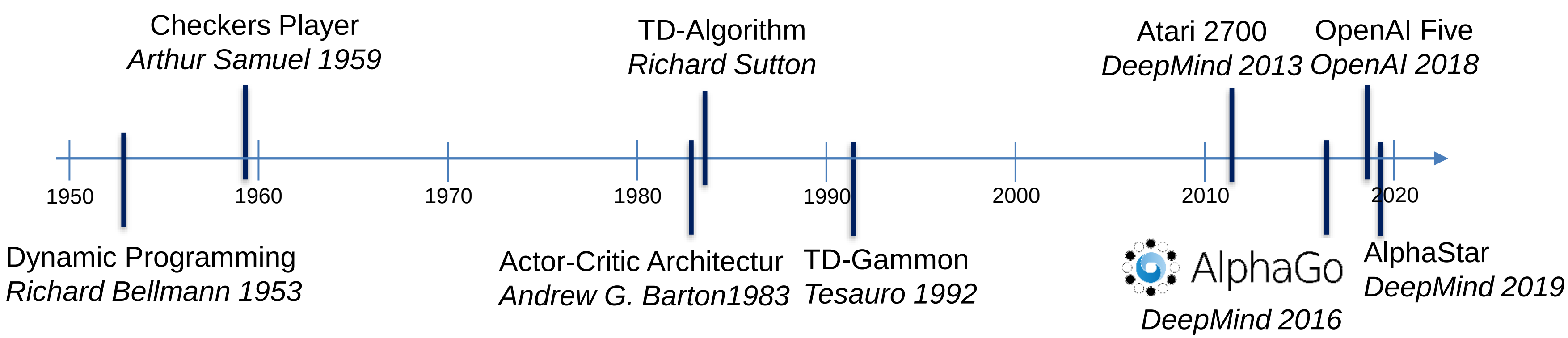


Open AI 5

/informatik & security

Source: Youtube https://www.youtube.com/watch?v=yBEidvm_tZQ

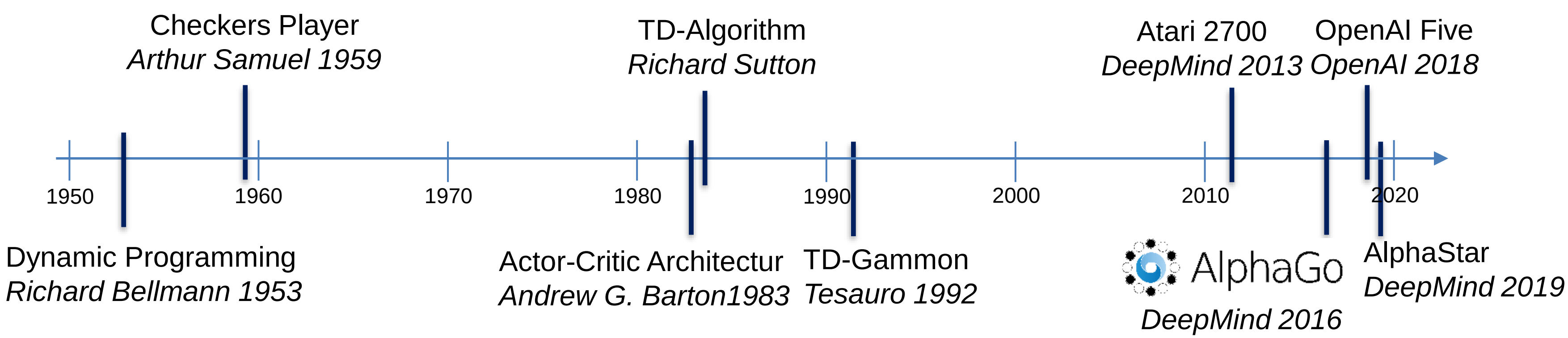
Milestones of RL



AlphaStar

Source: Youtube https://www.youtube.com/watch?v=cUTMhmVh1qs

Milestones of RL



Multi-armed Bandits

Exploitation

vs

Exploration

INTERACTIVE PART I

Multi-armed Bandits

n choices of **k** different actions

After each choice: numerical reward from stationary distribution

Objective: maximize total reward over n choices

Multi-armed Bandits

A_t… Action at timestep t

R_t… Reward at timestep t

<u>optimal action-value function:</u>

(for the multi-armed bandit problem)

$$q_*(a) \coloneqq \mathbb{E}[R_t | A_t = a]$$

We usually don‘t know that value!

$q_t(a)$… Estimate of $q_*(a)$ at timestep t

INTERACTIVE PART II

Multi-armed Bandits

/informatik & security /fh/// st.pölten

Estimator of action-value function:

$$q_t(a) := \frac{\text{sum of rewards when } a \text{ taken before } t}{\text{number of times } a \text{ taken before } t} = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{[A_i=a]}}{\sum_{i=1}^{t-1} \mathbb{1}_{[A_i=a]}}$$

Indicator function:

$$\mathbb{1}_{[A_i=a]}(a) := \begin{cases} 1 & \text{if } A_i = a \\ 0 & \text{if } A_i \neq a \end{cases}$$

complicated way of saying:
average of rewards

Action Selection Methods

/informatik & security

Random action selection method:

Randomly select one of the possible actions.

Action Selection Methods

Graph of a probability mass function (pmf) of the distribution:

No exploitation - Only exploration

Action Selection Methods

/informatik & security /fh/// st.pölten

<u>*Greedy* action selection method:</u>

Always select the action with the highest action-value according to the action-value function.

$$A_t = \underset{a}{\operatorname{argmax}}\, q_t(a)$$

Action Selection Methods

Graph of a probability mass function(pmf) of the distribution:

Only exploitation - no exploration

Action Selection Methods

/informatik & security /fh/// st.pölten

<u>*ε-Greedy* action selection method:</u>

In most $(1-\varepsilon)$ cases select the best action an in a small amount of cases select a random action.

$$A_t = \begin{cases} \underset{a}{\operatorname{argmax}}\, q_t(a) & \text{in } (100-\varepsilon)\ \%\ \text{cases} \\ \text{random action} & \text{in } \varepsilon\ \%\ \text{cases} \end{cases}$$

Mostly exploitation - small exploration

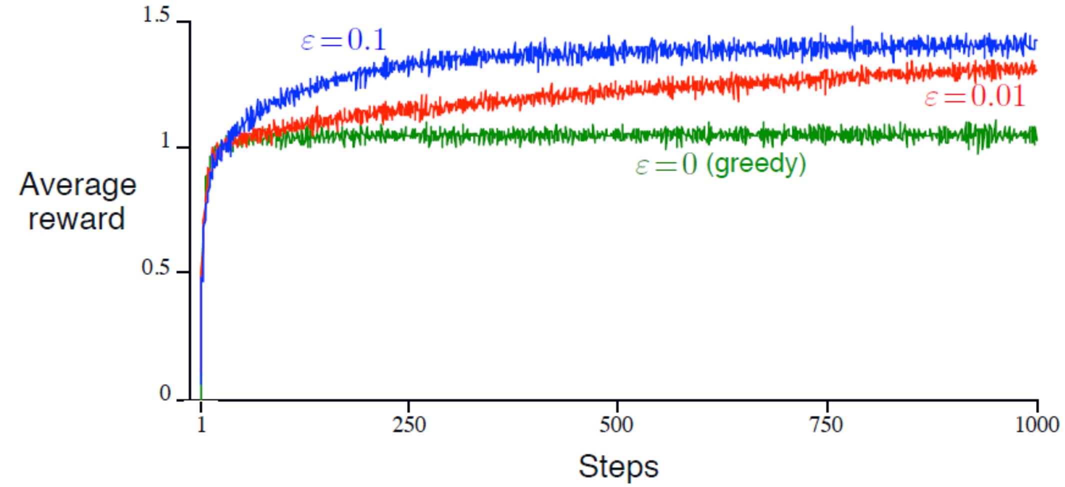
Action Selection Methods

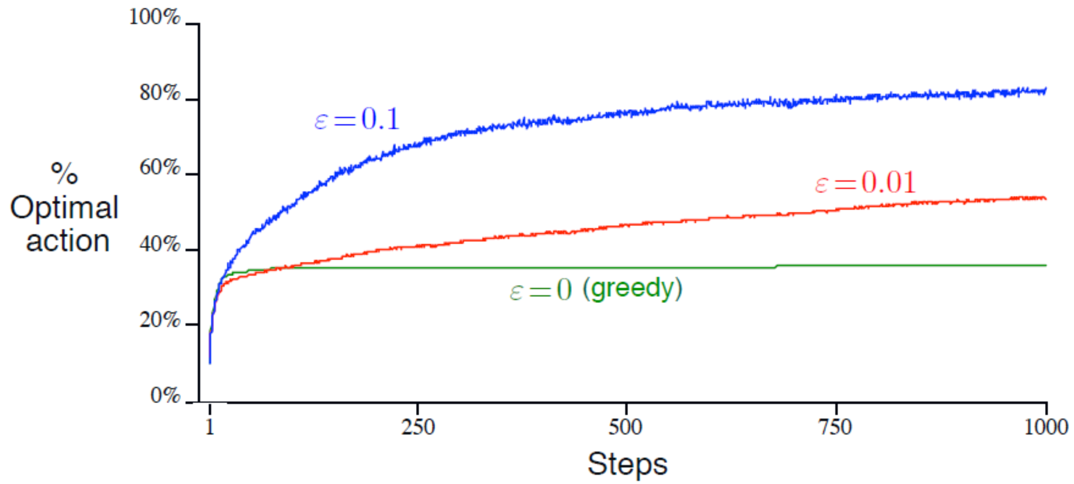
Graph of a probability mass function (pmf) of the distribution:

Mostly exploitation - small exploration

Multi-armed Bandits

/informatik & security





Incremental Implementation

/informatik & security /fh/// st.pölten

Simplified: only one action

$$q_n = \frac{R_1 + R_2 + \cdots + R_n}{n}$$

$$\begin{aligned}
q_n &= \frac{1}{n}\sum_{i=1}^{n} R_i \\
&= \frac{1}{n}(R_n + \sum_{i=1}^{n-1} R_i) \\
&= \frac{1}{n}(R_n + (n-1)\frac{1}{n-1}\sum_{i=1}^{n-1} R_i) \\
&= \frac{1}{n}(R_n + (n-1)Q_{n-1}) \\
&= \frac{1}{n}(R_n + nQ_{n-1} - Q_{n-1}) \\
&= Q_{n-1} + \frac{1}{n}(R_n - Q_{n-1})
\end{aligned}$$

Incremental Implementation

/informatik & security /fh/// st.pölten

$$q_{n+1} = q_n + \frac{1}{n+1}(R_n - q_n)$$

$$NewEstimate \leftarrow OldEstimate + StepSize\underbrace{[Target - OldEstimate]}_{\text{Error}}$$

Incremental Implementation

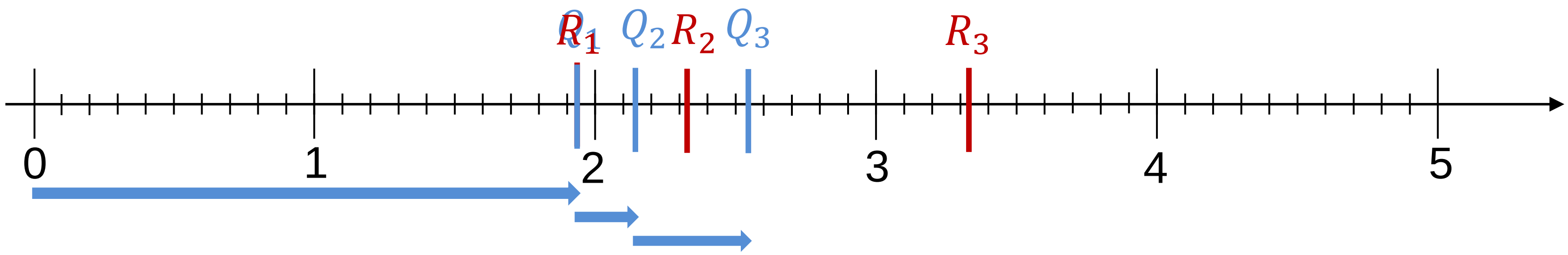
/informatik & security

$$NewEstimate \leftarrow OldEstimate + StepSize\underbrace{[Target - OldEstimate]}_{\text{Error}}$$

$$R_1 = 1{,}93 \quad Q_1 = 0 + \frac{1}{1}[1{,}93 - 0] = 1{,}93$$

$$R_2 = 2{,}35 \quad Q_2 = 1{,}93 + \frac{1}{2}[2{,}35 - 1{,}93] = 2{,}14$$

$$R_3 = 3{,}32 \quad Q_3 = 2{,}14 + \frac{1}{3}[3{,}32 - 2{,}14] = 2{,}53$$



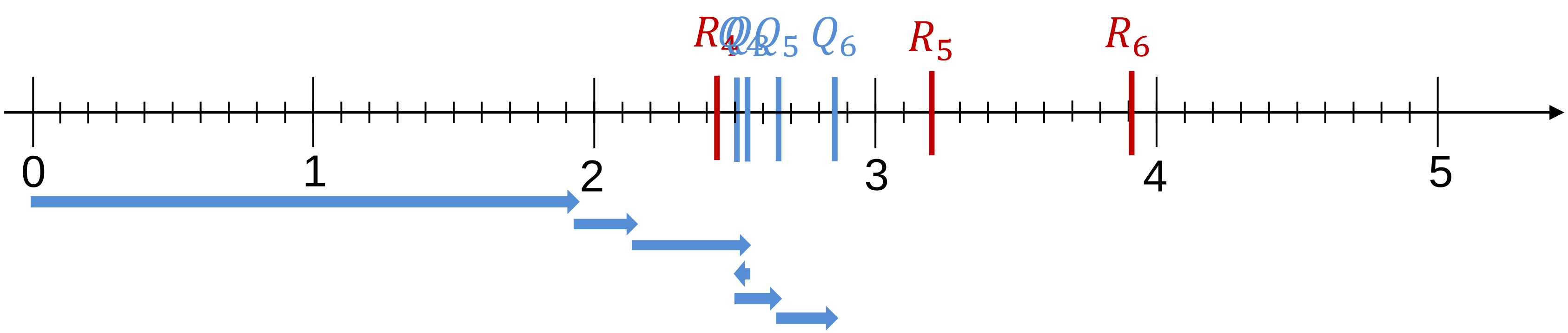
Incremental Implementation

$$NewEstimate \leftarrow OldEstimate + StepSize\underbrace{[Target - OldEstimate]}_{\text{Error}}$$

$$R_4 = 2{,}43 \quad Q_4 = 2{,}53 + \frac{1}{4}[2{,}43 - 2{,}53] = 2{,}505$$

$$R_5 = 3{,}28 \quad Q_5 = 2{,}505 + \frac{1}{5}[3{,}28 - 2{,}505] = 2{,}66$$

$$R_6 = 3{,}91 \quad Q_6 = 2{,}66 + \frac{1}{6}[3{,}91 - 2{,}66] = 2{,}868$$



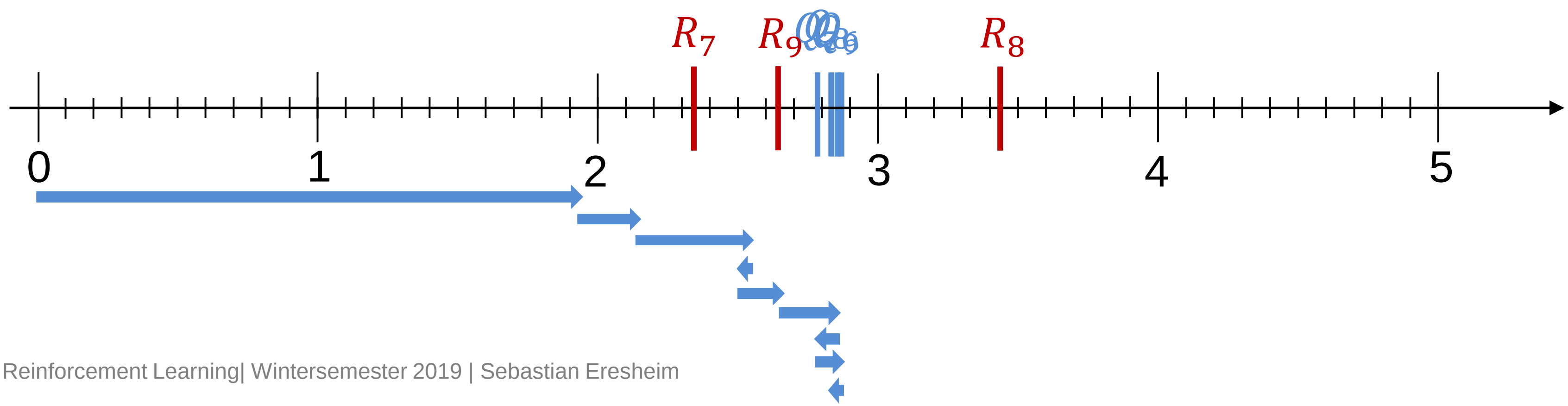
Incremental Implementation

$$NewEstimate \leftarrow OldEstimate + StepSize[\underbrace{Target - OldEstimate}_{\text{Error}}]$$

$$R_7 = 2{,}34 \quad Q_7 = 2{,}868 + \frac{1}{7}[2{,}34 - 2{,}868] = 2{,}793$$

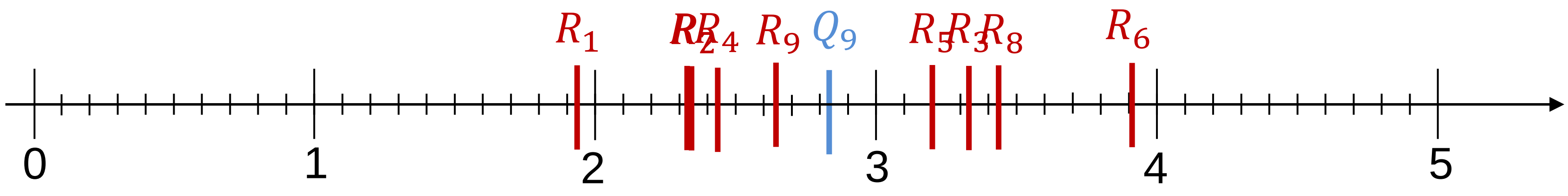
$$R_8 = 3{,}42 \quad Q_8 = 2{,}793 + \frac{1}{8}[3{,}42 - 2{,}793] = 2{,}871$$

$$R_9 = 2{,}64 \quad Q_9 = 2{,}871 + \frac{1}{9}[2{,}64 - 2{,}871] = 2{,}845$$

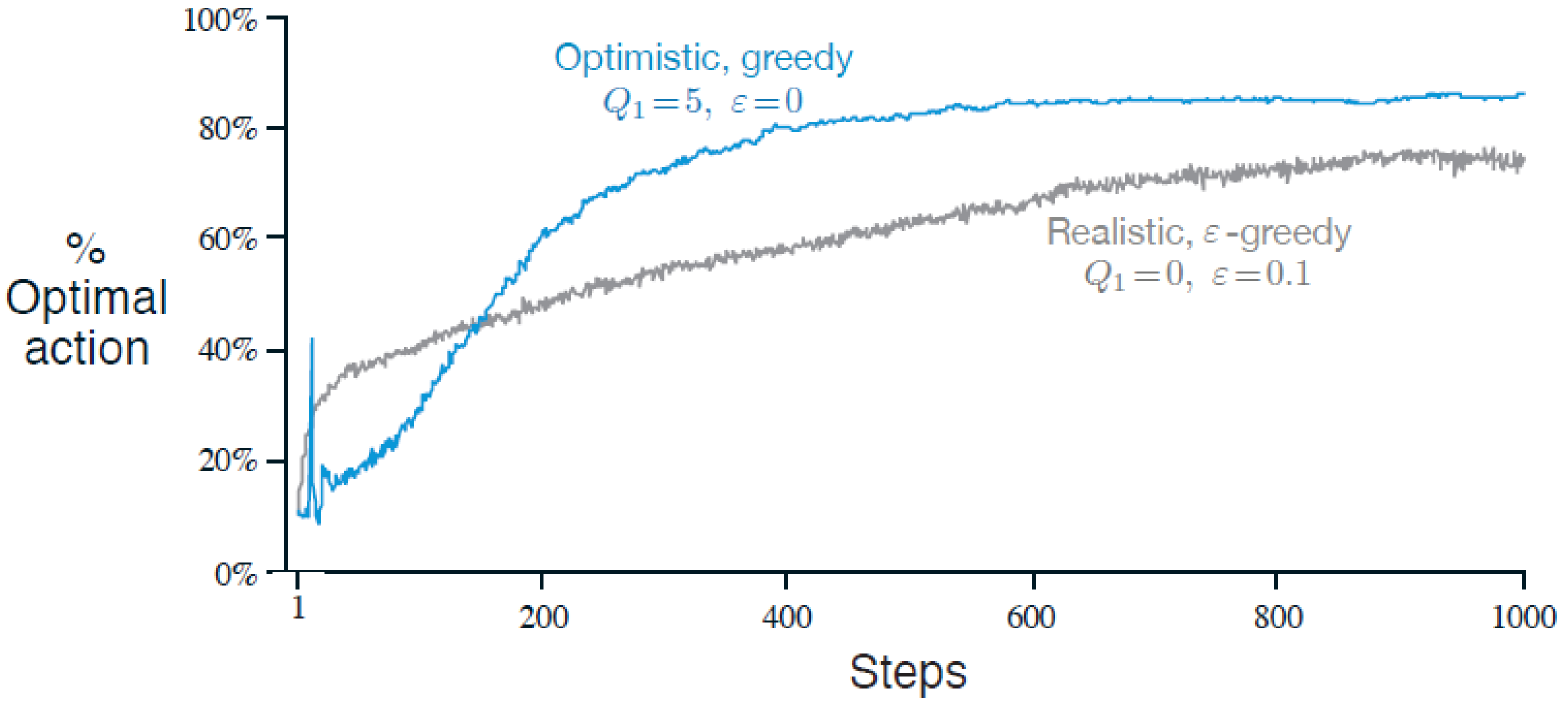


Incremental Implementation

$$NewEstimate \leftarrow OldEstimate + StepSize\underbrace{[Target - OldEstimate]}_{\text{Error}}$$



Optimistic initial values



Nonstationary Problem

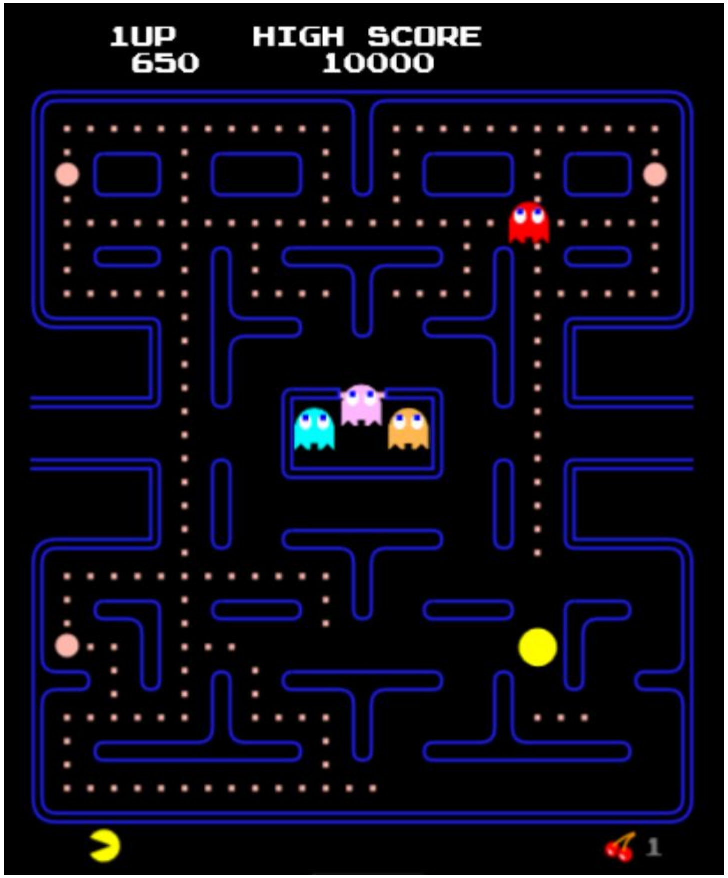
/informatik & security /fh/// st.pölten

Constant step-size Parameter $\alpha \epsilon (0,1]$ $\quad Q_{n+1} := Q_n + \alpha(R_n - Q_n)$

$$
\begin{aligned}
Q_{n+1} &= Q_n + \alpha(R_n - Q_n) \\
&= \alpha R_n + (1-\alpha)Q_n \\
&= \alpha R_n + (1-\alpha)[\alpha R_{n-1} + (1-\alpha)Q_{n-1}] \\
&= \alpha R_n + (1-\alpha)\,\alpha R_{n-1} + (1-\alpha)^2 Q_{n-1} \\
&= \alpha R_n + (1-\alpha)\,\alpha R_{n-1} + (1-\alpha)^2 \alpha R_{n-2} + \cdots + (1-\alpha)^{n-1}\alpha R_1 + (1-\alpha)^n Q_1 \\
&= (1-\alpha)^n Q_1 + \sum_{i=1}^{n} \alpha(1-\alpha)^{n-i} R_i
\end{aligned}
$$

Contextual Bandits

- Synonym for the full reinforcement learning problem
- Multiple situations in contrast to only one
- Action also affects the next situation not only the reward (Return)

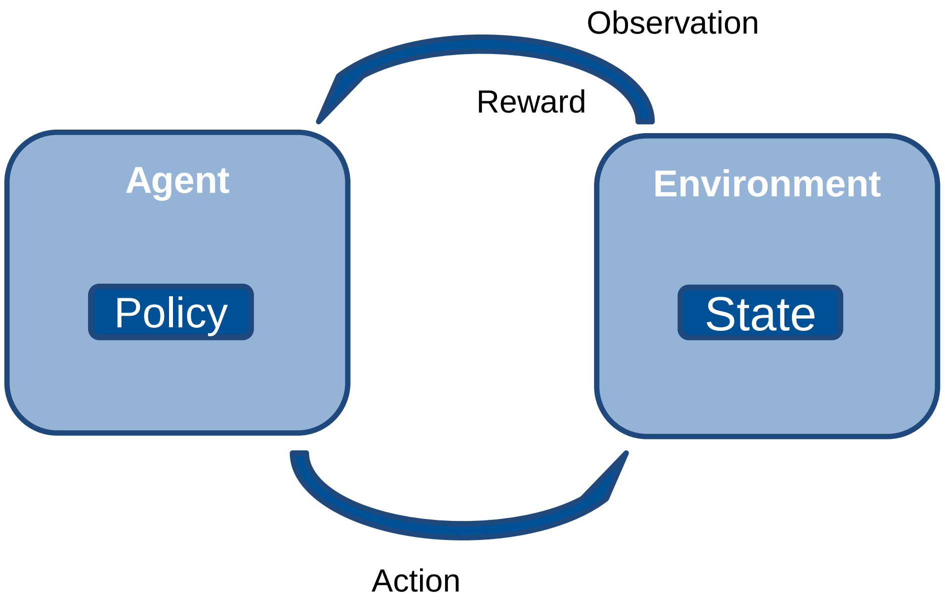


Multi-armed Bandits

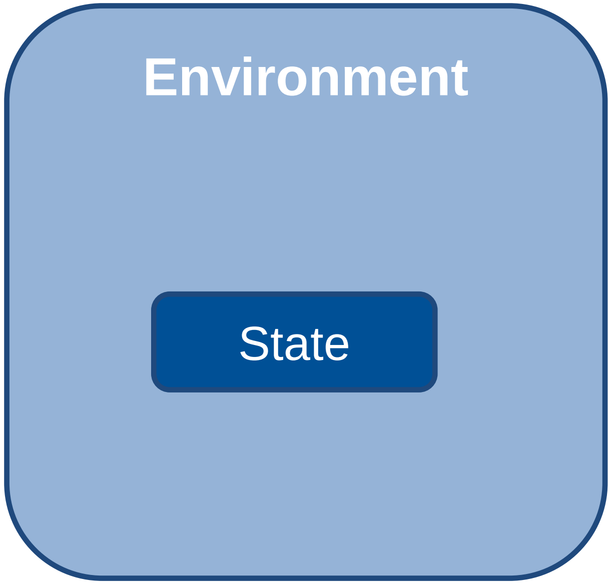
/informatik & security /fh/// st.pölten

- In which subfields can RL be devided into? What are the major differences?
- What is the mulit-armed bandit problem? What is the difference to the full RL problem?
- What is an acition-value function? How is it‘s estimator defined?
- What is the incremental notation of an action-value function estimator?
- Which action selecton methods exist?

MDP

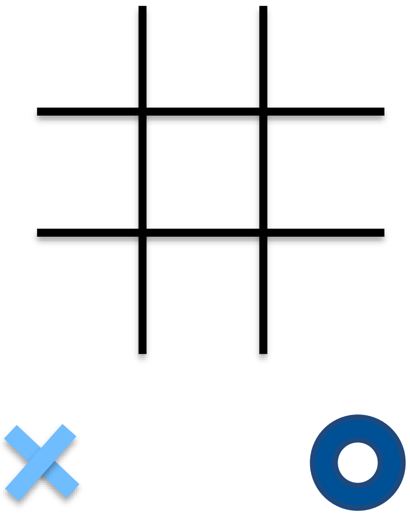

Observation
Reward
Agent
Policy
Environment
State
Action

MDP



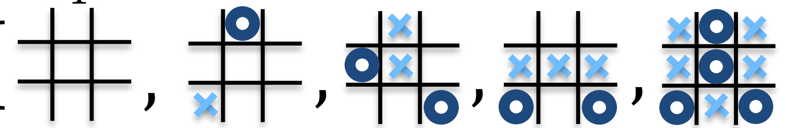
MDP – Example

Tic-Tac-Toe

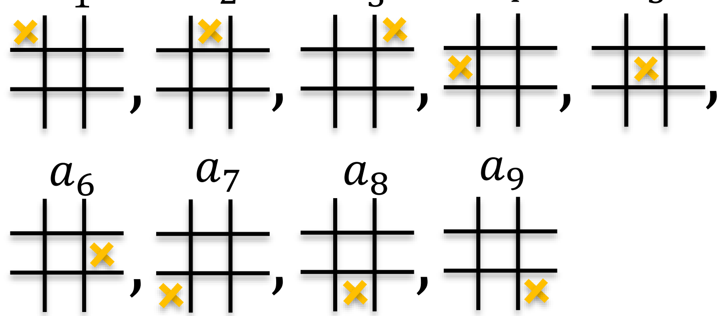


MDP – Example

State Space: $\mathcal{S} = \{ s_1, s_2, s_3, s_4, s_5, \ldots \}$

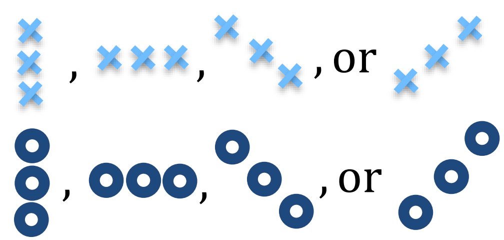


Action Space: $\mathcal{A} = \{ a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9 \}$



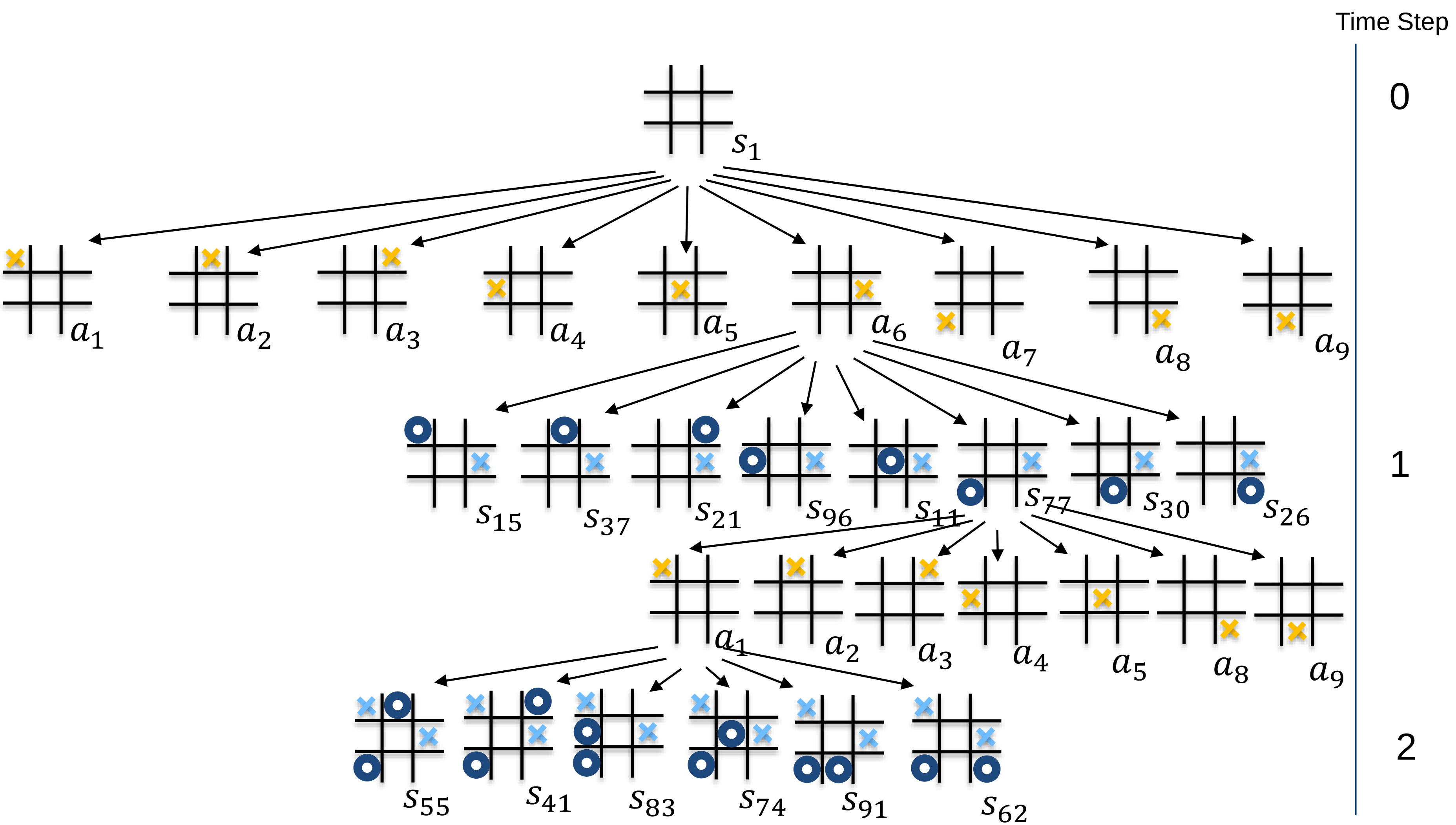
Reward Function:

$$\begin{cases} +1 & \text{if } \ldots, \ldots, \ldots, \text{or } \ldots \\ -1 & \text{if } \ldots, \ldots, \ldots, \text{or } \ldots \\ 0 & \text{else} \end{cases}$$

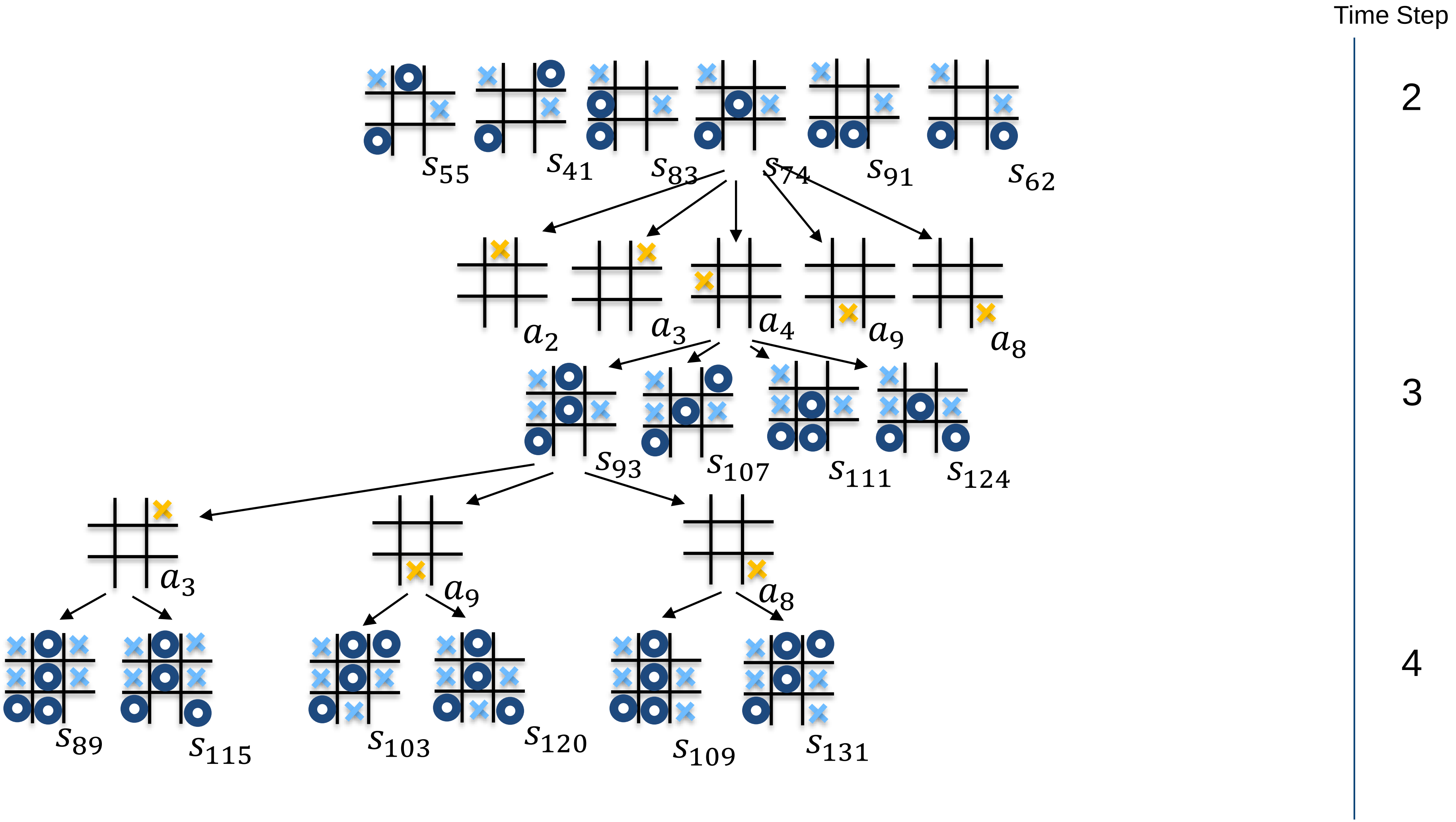


MDP – Example

/informatik & security

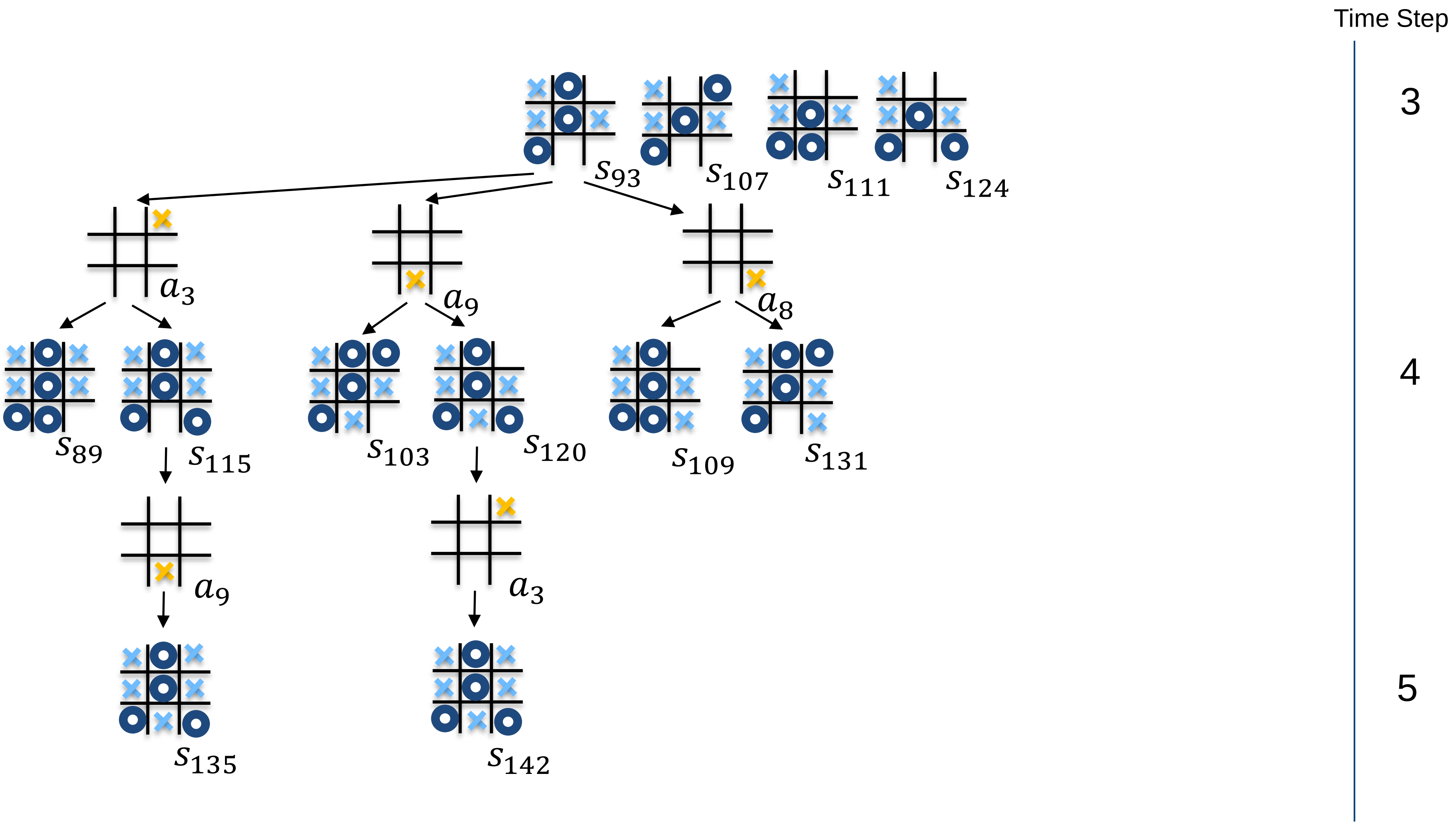


MDP – Example



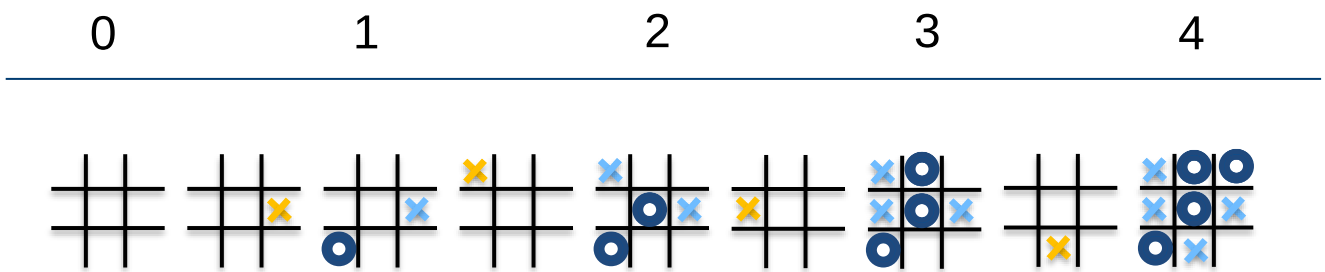
s_{55} s_{41} s_{83} s_{74} s_{91} s_{62}
a_2 a_3 a_4 a_9 a_8
s_{93} s_{107} s_{111} s_{124}
a_3 a_9 a_8
s_{89} s_{115} s_{103} s_{120} s_{109} s_{131}
Time Step t
2
3
4

MDP – Example

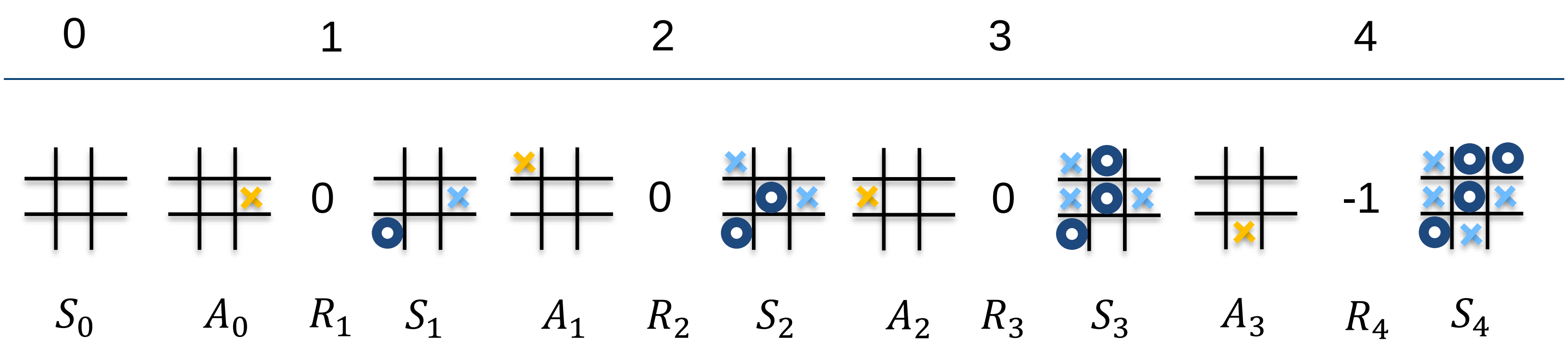


Time Step t
3
4
5
s_{93}
s_{107}
s_{111}
s_{124}
a_3
a_9
a_8
s_{89}
s_{115}
s_{103}
s_{120}
s_{109}
s_{131}
a_9
a_3
s_{135}
s_{142}

MDP – Example



MDP – Example

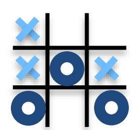
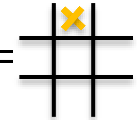


MDP – Example

S_t, A_t, R_t are random variables

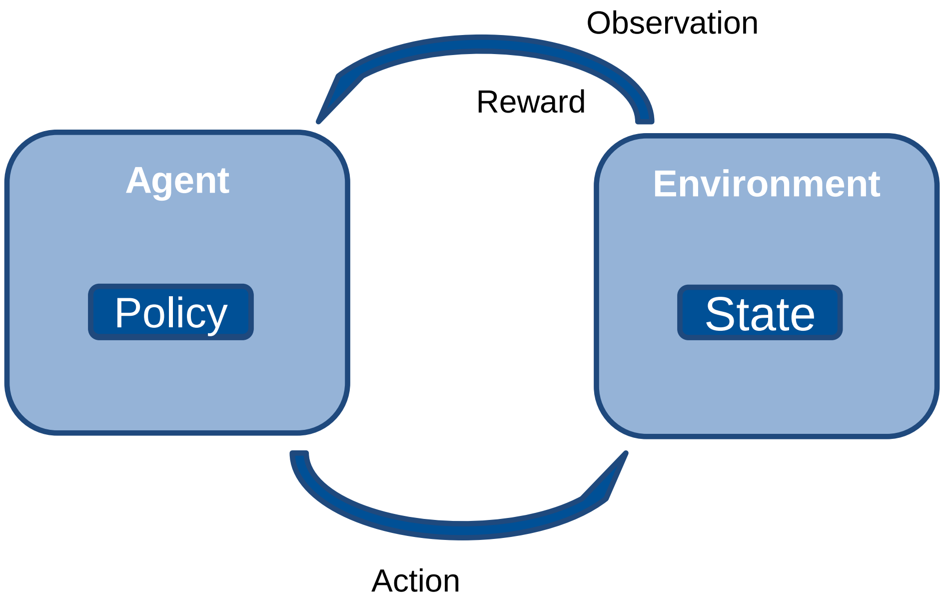
Like dice rolls or coin flips, they can have different results for separate executions.

$S_3 = s_{124}$ $A_2 = a_2$

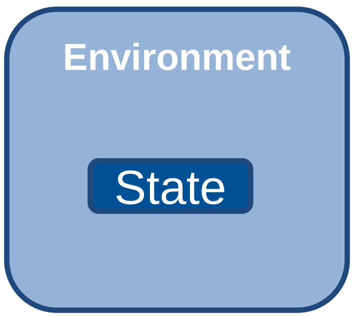
$S_3 =$  $A_2 =$ 

$$\mathbb{p}(S_{t+1} = s_{124} | S_t = s_{75}, A_t = a_4) = ?$$

MDP


Observation
Reward
Agent
Policy
Environment
State
Action

MDP – Definition



$\approx$

<u>Markov Decision Process</u>

State Space: $\mathcal{S} = \{s_1, s_2, \dots, s_N\}$

Action Space: $\mathcal{A} = \{a_1, a_2, \dots, a_M\}$

Set of Rewards: $\mathcal{R} \subset \mathbb{R}$

State-Transition Probability Function: $p: \mathcal{S} \times \mathcal{S} \times \mathcal{A} \rightarrow [0,1]$

Reward Function: $R: \mathcal{R} \times \mathcal{S} \times \mathcal{A} \rightarrow [0,1]$

MDP – Transition Function

State-Transition Probability Function: $p: \mathcal{S} \times \mathcal{S} \times \mathcal{A} \rightarrow [0,1]$

$$p(s', s, a) \coloneqq \mathbb{p}(S_{t+1} = s' | S_t = s, A_t = a)$$

- Defines how the <u>environment</u> behaves
- Function is usually not known
- Tic-tac-toe: contains rules of the games, as well as the opponents behavior

MDP – Reward Function

/informatik & security

Reward Function: $R: \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$

$$R(s, a, s')$$

- Defines what the environment rewards
- Function is usually designed by a human and therefore known
- Tic-tac-toe: contains rules of the games, as well as the opponents behavior

MDP – Markov Property

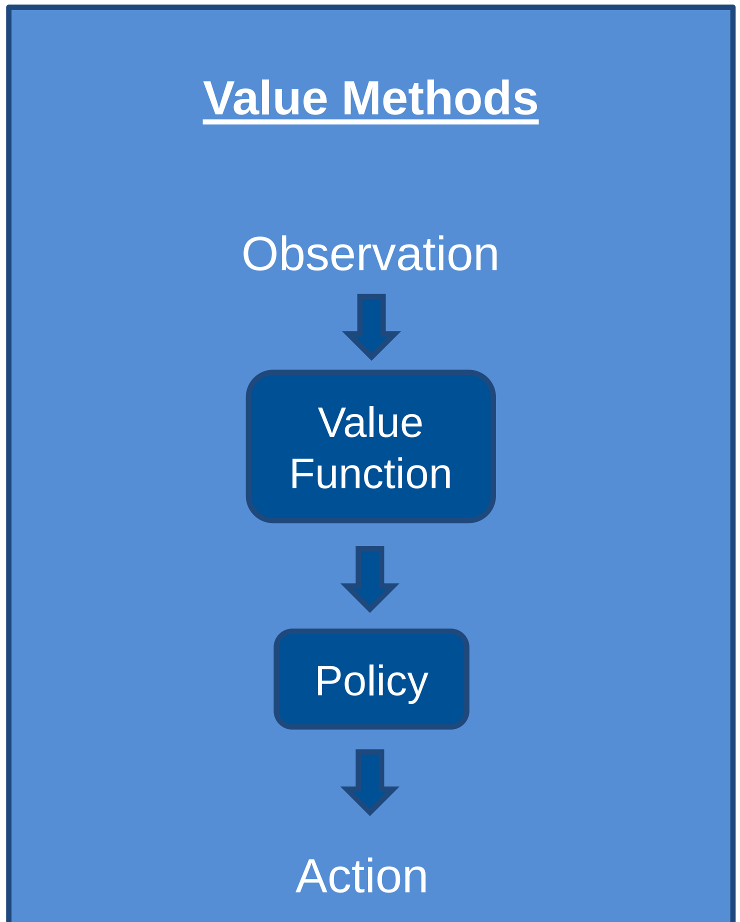
/informatik & security /fh/// st.pölten

The probability of each possible value of S_t and R_t depends **only** on the **immediately preceding** state and action and **not on earlier** states and actions.

„The future only depends on the present and not on the past“

$$\mathbb{p}(S_{t+1} = s' \,| S_t = s, A_t = a) =$$
$$\mathbb{p}(S_{t+1} = s' \,| S_t = s, A_t = a, \ldots, S_0 = s'', A_0 = a'',)$$

VF



VF – Policy

/informatik & security /fh/// st.pölten

Policy: $\pi: \mathcal{S} \times \mathcal{A} \rightarrow [0,1]$

- defines how the agent behaves
- Rows are probability functions over the action space
- Example: random policy

π	a_1	a_2	a_3	a_4
s_1	0.11	0.8	0.06	0.03
s_2	0.01	0.1	0.7	0.19
s_3	1.0	0.0	0.0	0.0
s_4	0.65	0.32	0.01	0.02
s_5	0.25	0.25	0.25	0.25
s_6	0.4	0.1	0.2	0.4

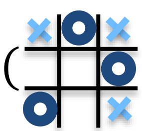
VF – Value Functions

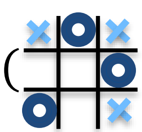
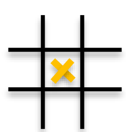
/informatik & security /fh/// st.pölten

Value Functions: $V: \mathcal{S} \rightarrow \mathbb{R}$

$Q: \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$

- indicate how „good“ a state or an action given a state is

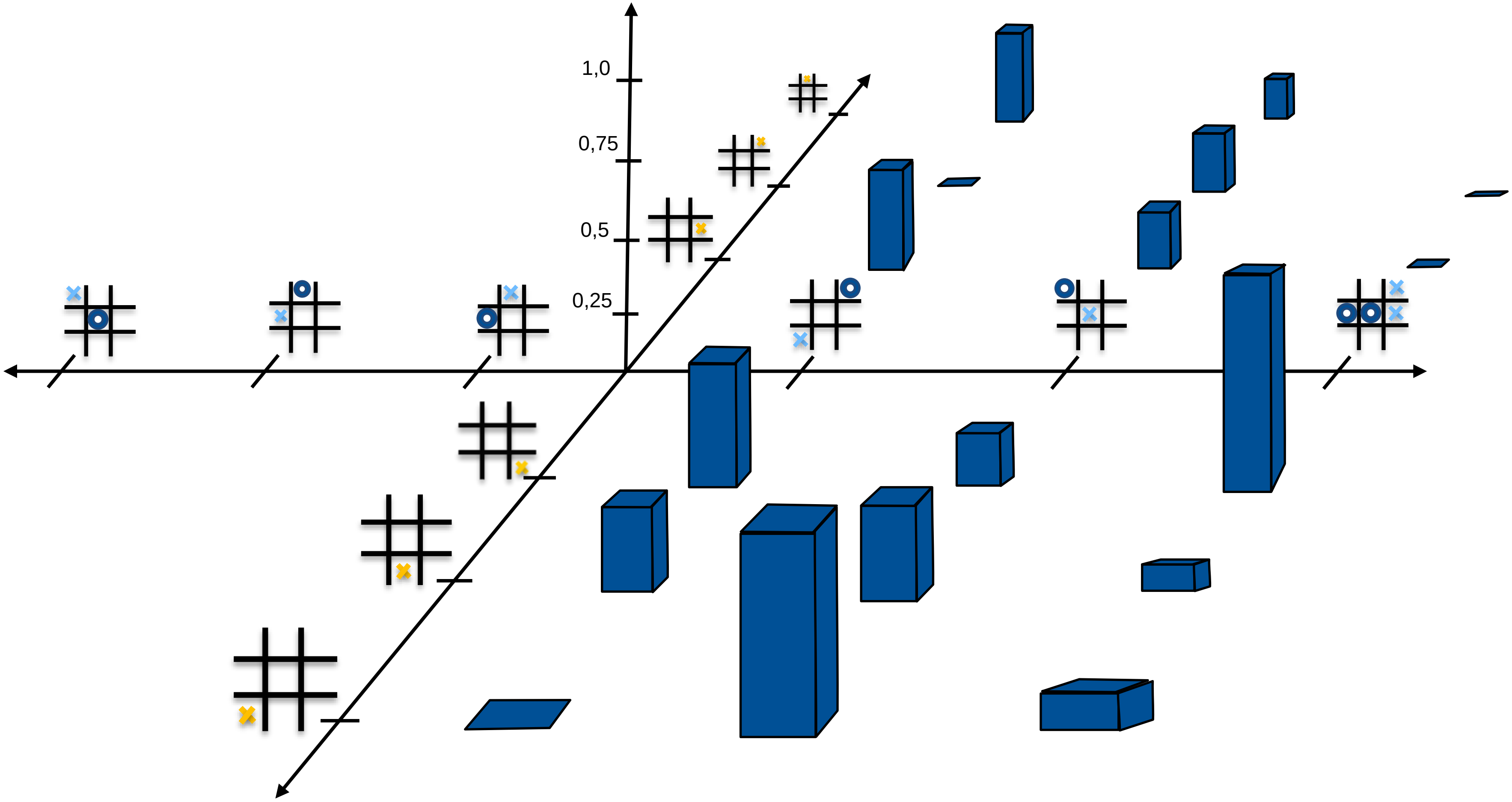
$V($  $) = 0.64$

$Q($ ,  $) = 1.0$

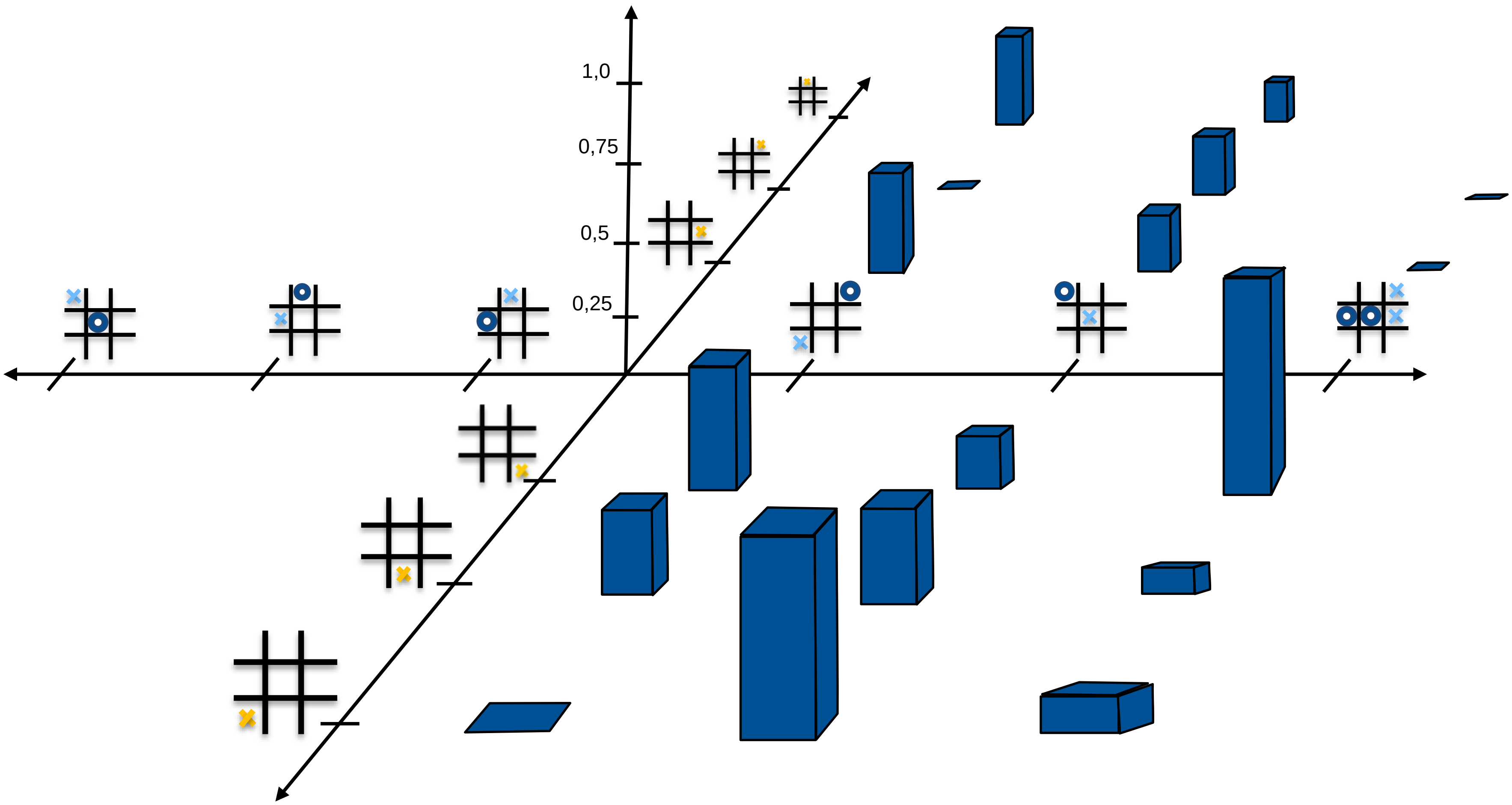
	V
$\boldsymbol{s_1}$	2.3
$\boldsymbol{s_2}$	-6.8
$\boldsymbol{s_3}$	10.0
$\boldsymbol{s_4}$	-5.4
$\boldsymbol{s_5}$	4.1
$\boldsymbol{s_6}$	-0.4

Q	$\boldsymbol{a_1}$	$\boldsymbol{a_2}$	$\boldsymbol{a_3}$	$\boldsymbol{a_4}$
$\boldsymbol{s_1}$	-0.1	4.3	0.3	1.8
$\boldsymbol{s_2}$	7.5	3	8.2	-5.7
$\boldsymbol{s_3}$	0.0	10.0	0.0	0.0
$\boldsymbol{s_4}$	6.5	0.32	0.01	-0.2
$\boldsymbol{s_5}$	3.0	0.21	-7.2	0.25
$\boldsymbol{s_6}$	4.5	0.1	-2.0	0.4

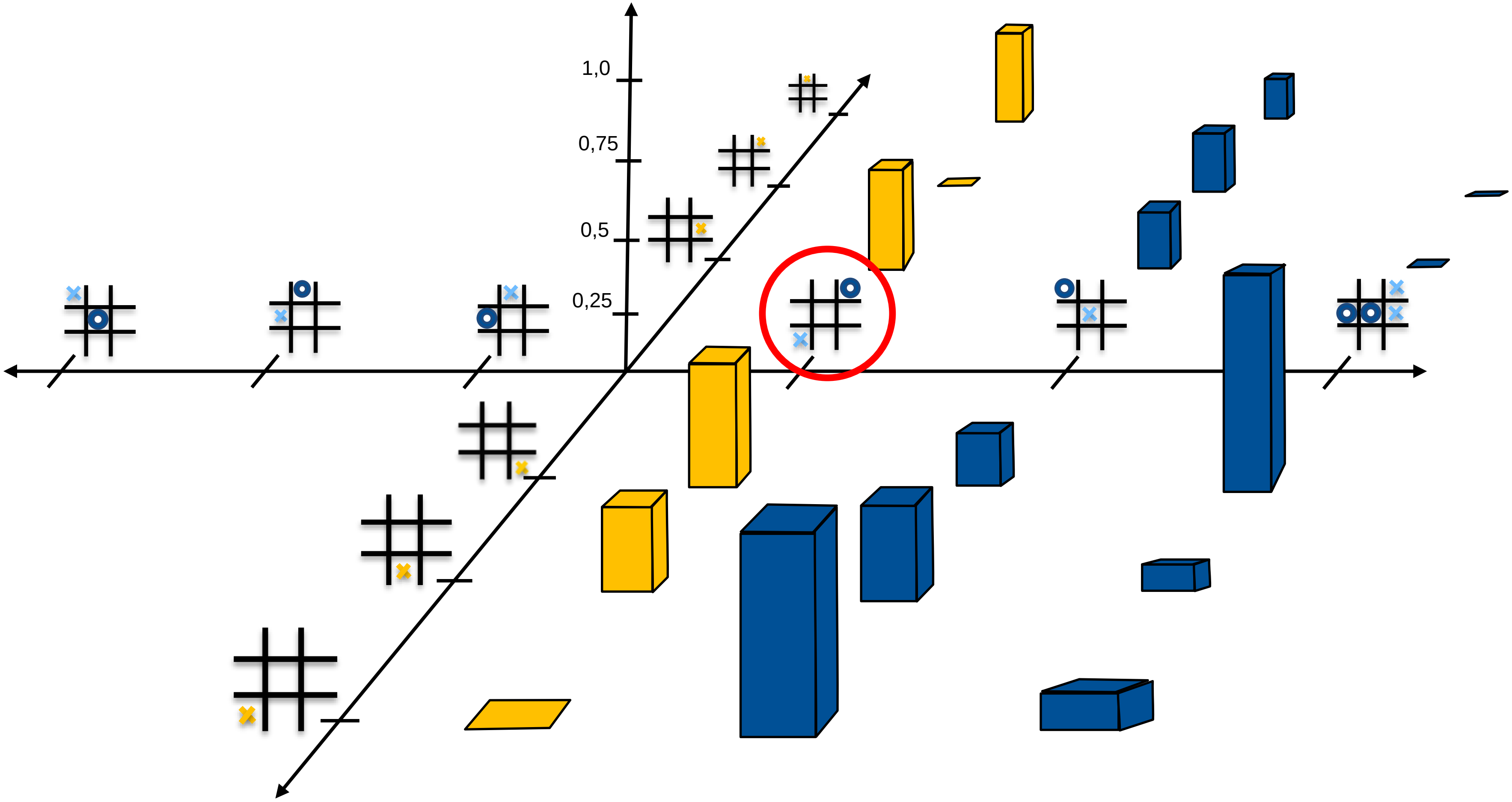
Value Function



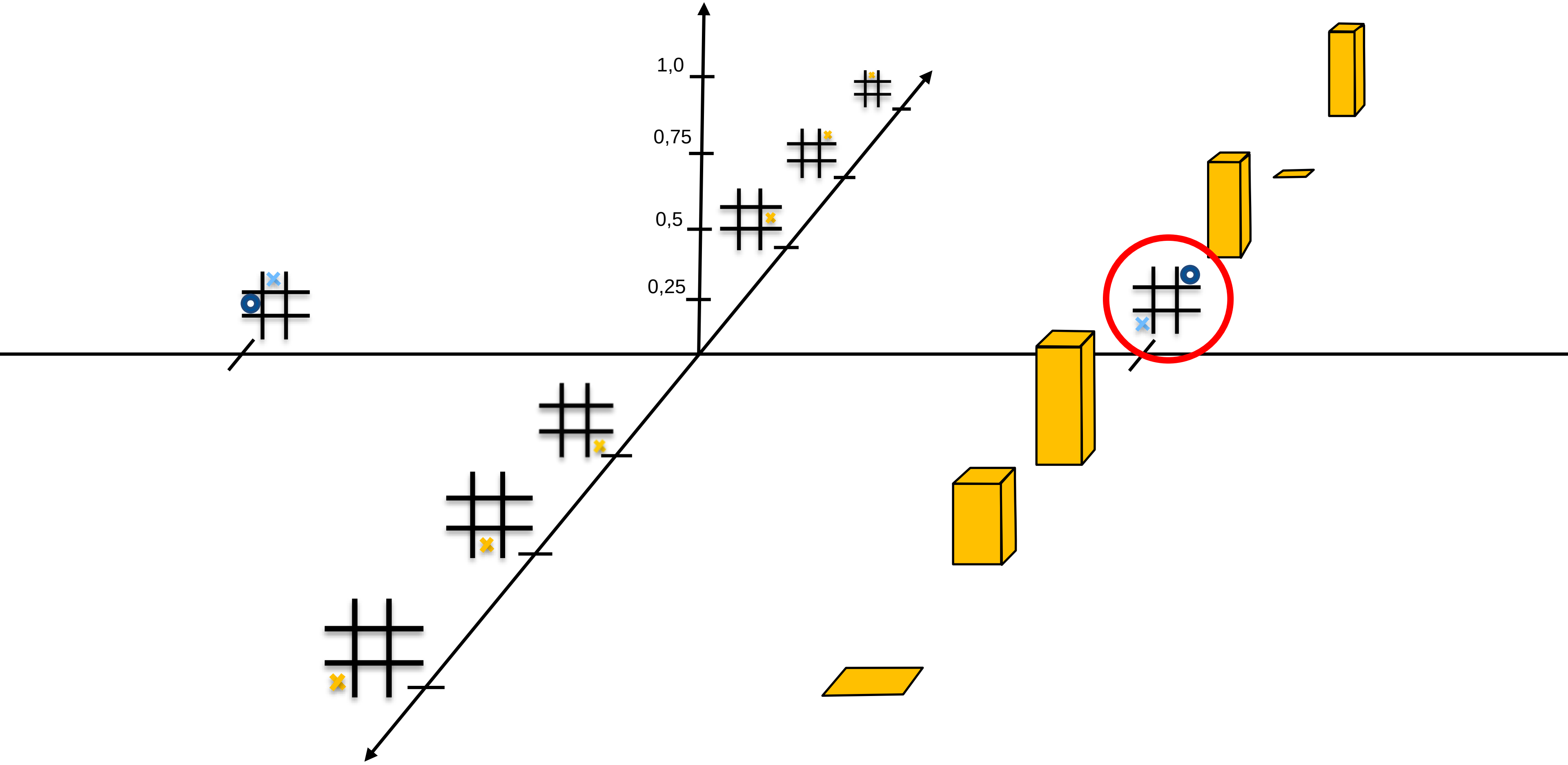
Value Function



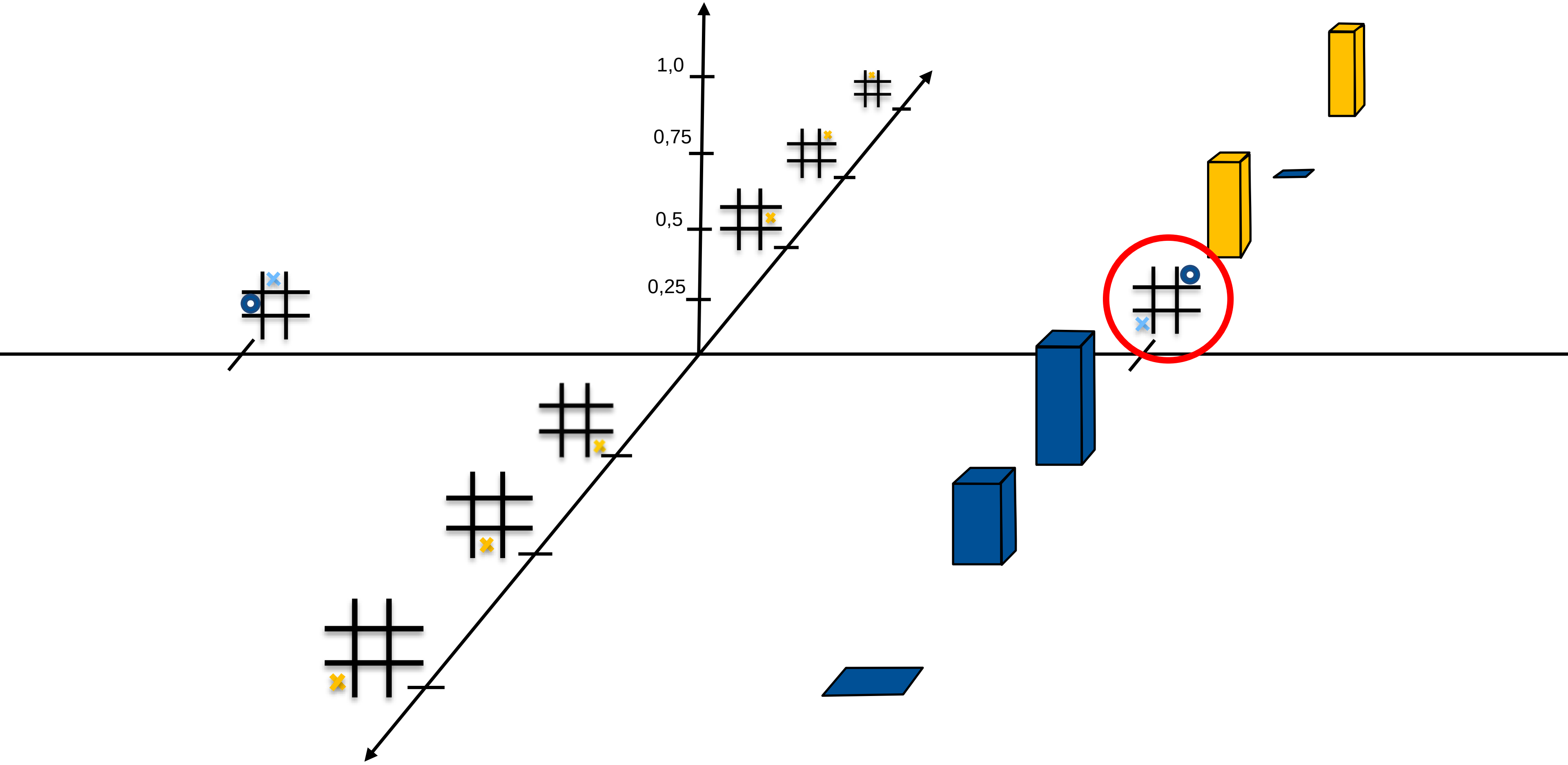
Value Function



Value Function

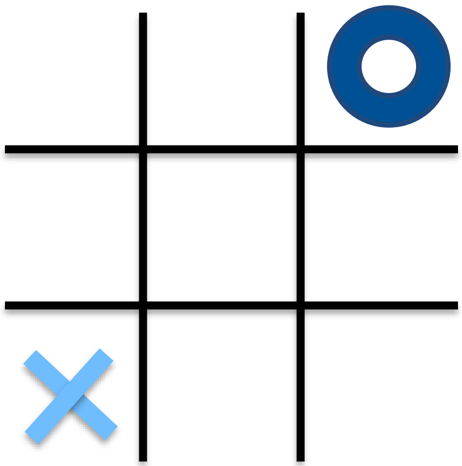


Value Function

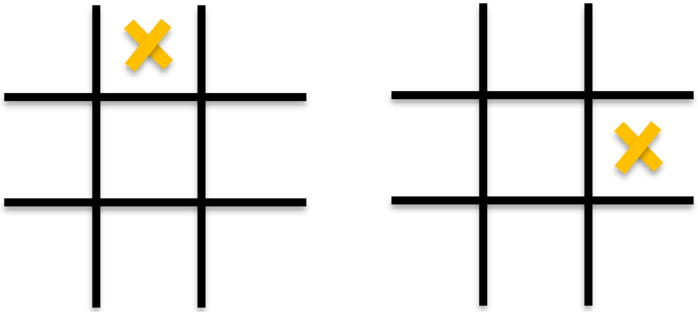


Value Function

State:



Greedy Actions:



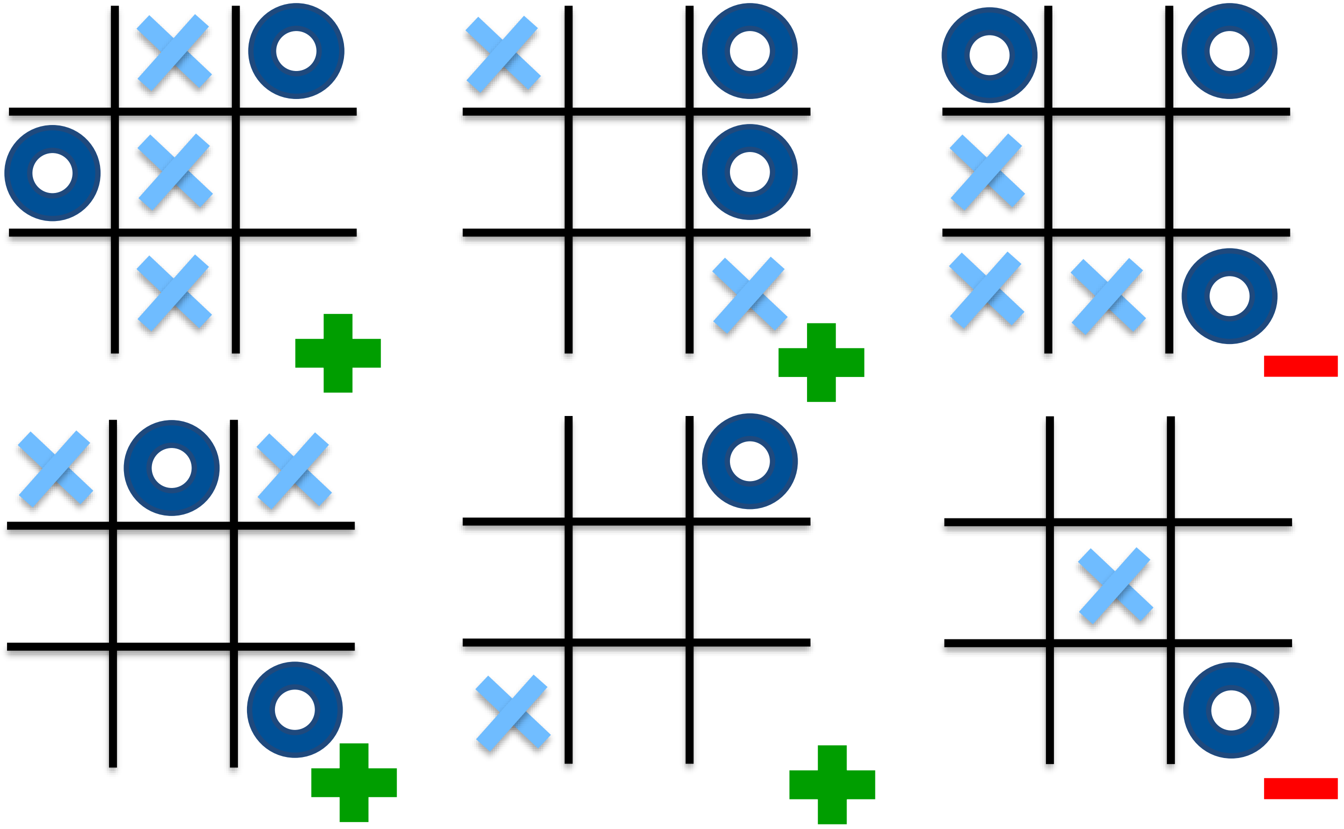
VF – Motivation

You are: X

Your opponent is: O

It is your turn!

VF – Motivation



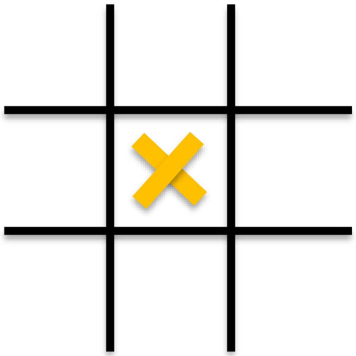
VF – Motivation

/informatik & security /fh/// st.pölten

You are:

Your opponent is:

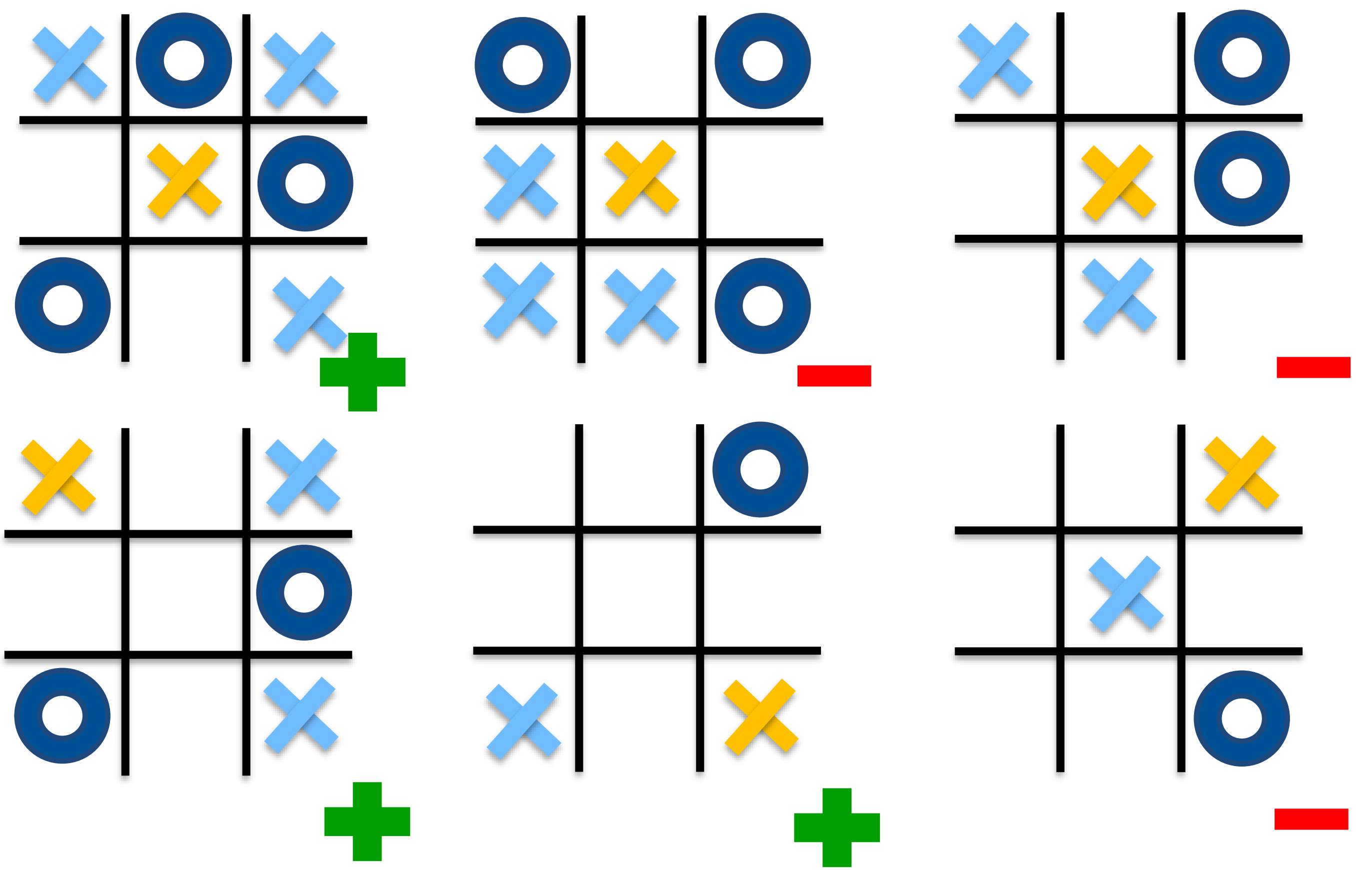
It is your turn!



indicates the next move

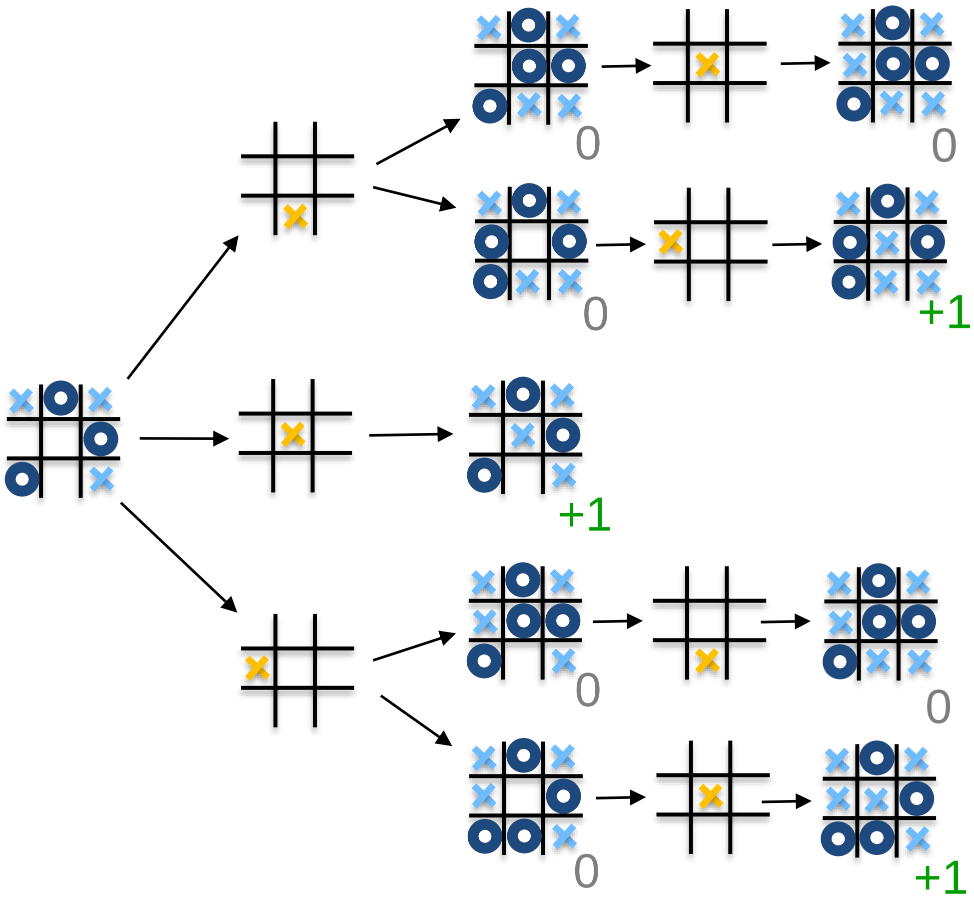
VF – Motivation

/informatik & security /fh/// st.pölten

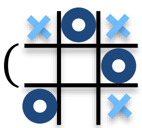


Which Value Function to choose?

VF



Naive approach:
5 outcomes
3 positive

=> $V($  $) = \frac{3}{5} = 0.6$

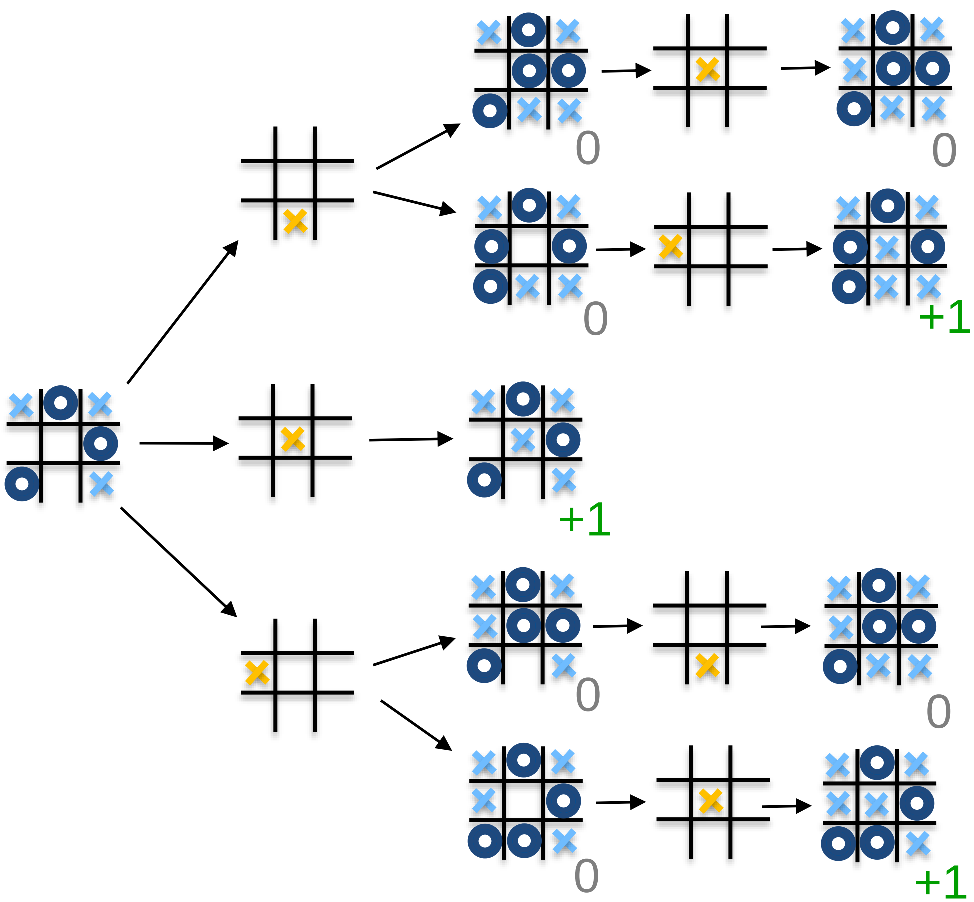
What if intermediate reward is given?

=> Need to add all reward until the end

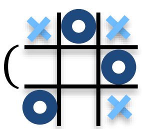
$$G_t := \sum_{n=t+1}^{T} \gamma^n R_n$$

is also a random variable

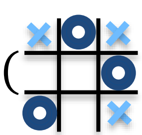
VF



Naive approach:
5 outcomes
3 positive

=> $V($  $) = \frac{3}{5} = 0.6$

Implies equal probability for all outcomes.
What if agent only chooses actions from the second column if possible?

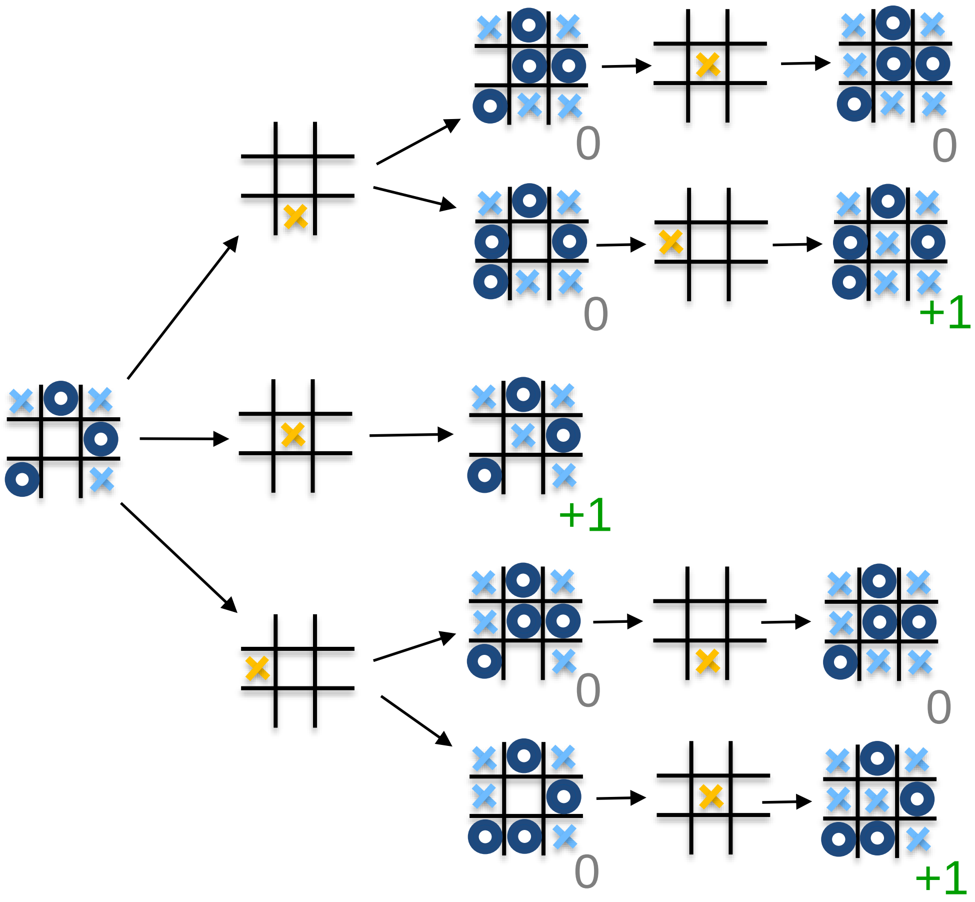
=> $V($  $) = \frac{2}{3} = 0.667$

Value function depends on current policy (which we know)

=> $V_\pi(s)$

VF

/informatik & security /fh/// st.pölten



Value function also depends on the environment (which we don‘t know)

=> Sampling

Sampling targets:

$$V_\pi(s) \approx \mathbb{E}_\pi[G_t | S_t = s]$$

$$Q_\pi(s, a) \approx \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$

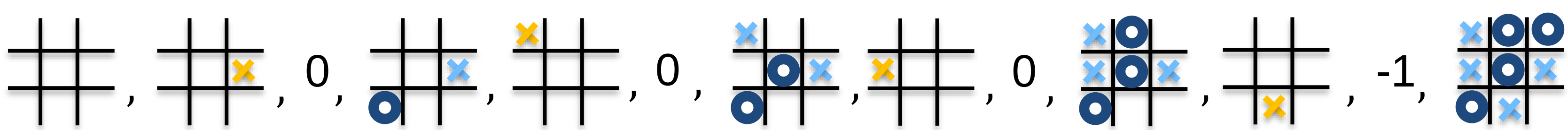
VF – Monte Carlo Method

1) Initialize with random policy and 0 for value funciton $Q(s, a)$

Q	$\boldsymbol{a_1}$	$\boldsymbol{a_2}$	$\boldsymbol{a_3}$	$\boldsymbol{a_4}$
$\boldsymbol{s_1}$	0.0	0.0	0.0	0.0
$\boldsymbol{s_2}$	0.0	0.0	0.0	0.0
$\boldsymbol{s_3}$	0.0	0.0	0.0	0.0
$\boldsymbol{s_4}$	0.0	0.0	0.0	0.0
$\boldsymbol{s_5}$	0.0	0.0	0.0	0.0
$\boldsymbol{s_6}$	0.0	0.0	0.0	0.0

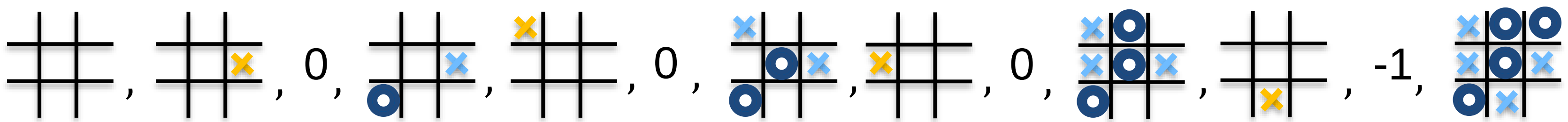
π	$\boldsymbol{a_1}$	$\boldsymbol{a_2}$	$\boldsymbol{a_3}$	$\boldsymbol{a_4}$
$\boldsymbol{s_1}$	0.25	0.25	0.25	0.25
$\boldsymbol{s_2}$	0.25	0.25	0.25	0.25
$\boldsymbol{s_3}$	0.25	0.25	0.25	0.25
$\boldsymbol{s_4}$	0.25	0.25	0.25	0.25
$\boldsymbol{s_5}$	0.25	0.25	0.25	0.25
$\boldsymbol{s_6}$	0.25	0.25	0.25	0.25

2) Play episode with current policy π



VF – Monte Carlo Method

2) Play episode with current policy π



3) Iterate episode from back to front, calculate the return G_t and update the value function

4) Make policy ϵ-greedy with regard to value function

5) continue with step 2)