

Lecture Notes in Networks and Systems 1009


Herwig Unger
Marcel Schaible *Editors*

Advances in Real-Time and Autonomous Systems

Proceedings of the 15th International
Conference on Autonomous Systems

 Springer

Series Editor

Janusz Kacprzyk , *Systems Research Institute, Polish Academy of Sciences, Warsaw, Poland*

Advisory Editors

Fernando Gomide, *Department of Computer Engineering and Automation—DCA, School of Electrical and Computer Engineering—FEEC, University of Campinas—UNICAMP, São Paulo, Brazil*

Okyay Kaynak, *Department of Electrical and Electronic Engineering, Bogazici University, Istanbul, Türkiye*

Derong Liu, *Department of Electrical and Computer Engineering, University of Illinois at Chicago, Chicago, USA*

Institute of Automation, Chinese Academy of Sciences, Beijing, USA

Witold Pedrycz, *Department of Electrical and Computer Engineering, University of Alberta, Alberta, Canada*

Systems Research Institute, Polish Academy of Sciences, Warsaw, Canada

Marios M. Polycarpou, *Department of Electrical and Computer Engineering, KIOS Research Center for Intelligent Systems and Networks, University of Cyprus, Nicosia, Cyprus*

Imre J. Rudas, *Óbuda University, Budapest, Hungary*

Jun Wang, *Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong*

The series “Lecture Notes in Networks and Systems” publishes the latest developments in Networks and Systems—quickly, informally and with high quality. Original research reported in proceedings and post-proceedings represents the core of LNNS.

Volumes published in LNNS embrace all aspects and subfields of, as well as new challenges in, Networks and Systems.

The series contains proceedings and edited volumes in systems and networks, spanning the areas of Cyber-Physical Systems, Autonomous Systems, Sensor Networks, Control Systems, Energy Systems, Automotive Systems, Biological Systems, Vehicular Networking and Connected Vehicles, Aerospace Systems, Automation, Manufacturing, Smart Grids, Nonlinear Systems, Power Systems, Robotics, Social Systems, Economic Systems and other. Of particular value to both the contributors and the readership are the short publication timeframe and the worldwide distribution and exposure which enable both a wide and rapid dissemination of research output.

The series covers the theory, applications, and perspectives on the state of the art and future developments relevant to systems and networks, decision making, control, complex processes and related areas, as embedded in the fields of interdisciplinary and applied sciences, engineering, computer science, physics, economics, social, and life sciences, as well as the paradigms and methodologies behind them.

Indexed by SCOPUS, INSPEC, WTI Frankfurt eG, zbMATH, SCImago.

All books published in the series are submitted for consideration in Web of Science.

For proposals from Asia please contact Aninda Bose (aninda.bose@springer.com).

Herwig Unger · Marcel Schaible
Editors

Advances in Real-Time and Autonomous Systems

Proceedings of the 15th International
Conference on Autonomous Systems

 Springer

Editors

Herwig Unger
Lehrgebiet Kommunikationsnetze
FernUniversität in Hagen
Hagen, Germany

Marcel Schaible
Lehrgebiet Kommunikationsnetze
FernUniversität in Hagen
Hagen, Nordrhein-Westfalen, Germany

ISSN 2367-3370

ISSN 2367-3389 (electronic)

Lecture Notes in Networks and Systems

ISBN 978-3-031-61417-0

ISBN 978-3-031-61418-7 (eBook)

<https://doi.org/10.1007/978-3-031-61418-7>

© The Editor(s) (if applicable) and The Author(s), under exclusive license
to Springer Nature Switzerland AG 2024

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

If disposing of this product, please recycle the paper.

Preface

It is with great pleasure that we present the second edition of the “Real-time and Autonomous Systems” volume, published by Springer in English. This collection showcases the advancements made in this field by European and Thai scientists. The conference on “Autonomous Systems” has now reached its 15th iteration, with a brief interruption due to the COVID-19 pandemic.

Originating in 2008 as a modest workshop for PhD students, our conference has evolved into a cherished tradition. Since 2010, colleagues from diverse research domains have contributed articles on their ongoing work, unsolved scientific challenges, and research outcomes to our proceedings. The event was initially titled “Distributed Systems and Networks”, but in subsequent years, it was aptly renamed “Autonomous Systems” to encompass the broad spectrum of contributions. This term encapsulates self-contained and self-controlled entities that operate without external oversight in various scientific disciplines.

In 2019, we made the significant decision to rebrand the publication from “proceedings” to “almanac”, reflecting its evolving nature. Authors are now encouraged to contribute without the obligation to attend or present at the conference. Maintaining our commitment to openness, the almanac remains free from reviews or censorship, fostering a platform for unconventional ideas and perspectives. Distributed among conference participants, it serves as a catalyst for uninhibited discussions, emphasizing the importance of intellectual diversity in scientific discourse.

In 2023, we ventured into a collaborative publication by combining the proceedings of two conferences: the “Real-Time Systems” in Boppard and the “Autonomous Systems” in Majorca Island. Regrettably, we couldn’t sustain this collaboration due to the cancellation of the “Real-time Systems” meeting in 2023 owing to insufficient submissions. Despite challenges, the “Autonomous Systems” conference in 2023 flourished with over 40 participants, reminiscent of pre-pandemic times. Notable keynote speakers, including Wookey Lee (Korea), Wolfram Schiffmann (Germany), and Phayung Meesad (Thailand), enriched our program with insights into team synergy, emergency landing systems, and stock analysis using deep reinforcement learning. Stephan Pareigis (Germany) provided a comprehensive tutorial on Reinforcement Learning, while Kyan-doghere Kyamakya showcased groundbreaking results in “Intelligent Traffic Systems”. Additionally, for the first time a tutorial on the real-time, safety-related programming language PEARL by Wolfgang Halang is published in English in this volume.

Innovating our presentation format, we introduced open discussion sessions for major topics, fostering interactive exchanges among participants. This format, characterized by brief talks followed by small-group discussions involving specialists and non-specialists, received positive feedback, indicating a promising future for interactive conference sessions.

The editors hope this publication inspires and informs readers, sparking new ideas and expanding knowledge horizons. We extend a warm invitation to join us at the upcoming conference in Cala Millor, Spain, from October 22–27 2024, either as a participant or a contributor. Further details are available at <https://www.confautsys.org>.

We eagerly anticipate your engagement and contribution to our vibrant scientific community.

January 2024

Herwig Unger
Marcel Schaible

Organization

Program Committee

R. Baran, Hamburg
J. Bartels, Krefeld
M. Baunach, Graz
B. Beenen, Lüneburg
J. Benra, Wilhelmshaven
V. Cseke, Wedemark
R. Gumzej, Maribor
W.A. Halang, Hagen
H.H. Heitmann, Hamburg
P. Holleczeck, Erlangen
M.M. Kubek, Hagen
Z. Li, Hagen
R. Müller, Furtwangen
S. Pareigis, Hamburg
M. Pellkofer, Landshut
M. Schaible, München
G. Schiedermeier, Landshut
U. Schneider, Mittweida
D. Tutsch, Wuppertal
H. Unger, Hagen (Vorsitz)
C. Yuan, Köln
D. Zöbel, Koblenz

Internet Address of the Real-Time Systems Committee: www.real-time.de

CR Subject Classification (2001): C3, D.4.7

Contents

Algorithmic Foundations of Reinforcement Learning	1
<i>Stephan Pareigis</i>	
Autonomous Emergency Landing of an Aircraft in Case of Total Engine-Out	28
<i>Steffen Flämig and Wolfram Schiffmann</i>	
The Safety-Related Real-Time Language SafePEARL	43
<i>Wolfgang A. Halang</i>	
Meta-Learning for Time Series Analysis and/or Forecasting: Concept Review and Comprehensive Critical Comparative Survey	80
<i>Witesyavwirwa Vianney Kambale, Denis D'Ambrosi, Paraskevi Fasouli, and Kyandoghere Kyamakya</i>	
Ensemble Learning with Physics-Informed Neural Networks for Harsh Time Series Analysis	110
<i>Antoine Kazadi Kayisu, Paraskevi Fasouli, Witesyavwirwa Vianney Kambale, Pitshou Bokoro, and Kyandoghere Kyamakya</i>	
Language Meets Vision: A Critical Survey on Cutting-Edge Prompt-Based Image Generation Models	122
<i>Paraskevi Fasouli, Witesyavwirwa Vianney Kambale, and Kyandoghere Kyamakya</i>	
Blockchain and Beyond – A Survey on Scalability Issues	141
<i>Oliver Tominski and Martin Drebingner</i>	
Emotion-Aware Chatbots: Understanding, Reacting and Adapting to Human Emotions in Text Conversations	158
<i>Philip Kossack and Herwig Unger</i>	
Author Index	177



Algorithmic Foundations of Reinforcement Learning

Stephan Pareigis^(✉)

Department of Informatics, Hamburg University of Applied Sciences, Berliner Tor 7,
20099 Hamburg, Germany
stephan.pareigis@haw-hamburg.de

Abstract. A comprehensive algorithmic introduction to reinforcement learning is given, laying the foundational concepts and methodologies. Fundamentals of Markov Decision Processes (MDPs) and dynamic programming are covered, describing the principles and techniques for addressing model-based problems within MDP frameworks. The most significant model-free reinforcement learning algorithms, including Q-learning and actor-critic methods are explained in detail. A comprehensive overview of each algorithm's mechanisms is provided, forming a robust algorithmic and mathematical understanding of current practices in reinforcement learning.

Keywords: reinforcement learning · MDP · markov-decision process · dynamic programming · deep reinforcement learning · SARSA · Q-learning · DQN · REINFORCE · A2C · PPO · DDPG · SAC · policy gradient methods · exploration vs exploitation · sparse rewards · robotics · offline reinforcement learning

1 Introduction

The article provides a comprehensive introduction and overview over the algorithmic foundations of Reinforcement Learning (RL).

Reinforcement Learning forms an important part of artificial intelligence, characterized by learning optimal decision-making through interactions with dynamic environments. The field is characterized by numerous technological applications, including autonomous systems, strategic game playing, and complex decision-making processes.

Figure 1 illustrates the algorithmic and theoretical roots of reinforcement learning, its categorization within the area of artificial intelligence, and important fields of application. RL can be categorized as a distinct type of machine learning, next to supervised learning and unsupervised learning. Its algorithmic roots lie within optimal control theory and dynamic programming. Applications include problems in which sequential decisions have to be made in order to optimize a given objective function.

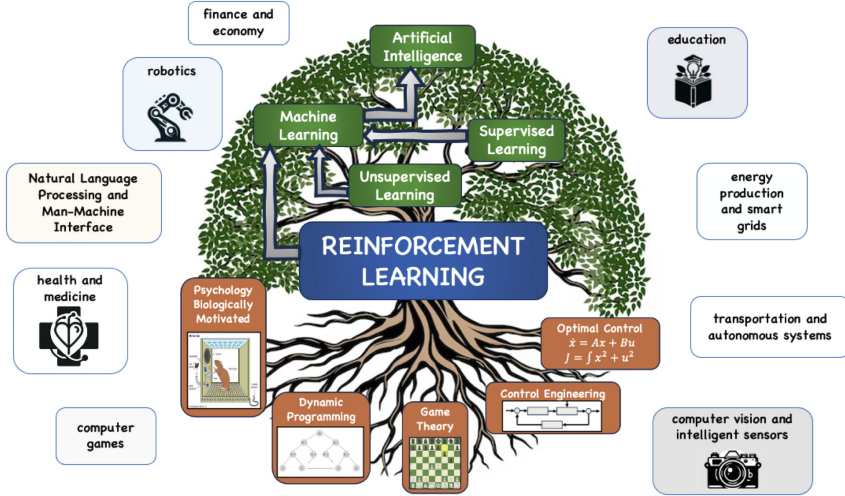


Fig. 1. The Roots of Reinforcement Learning: A visual map illustrating the interdisciplinary nature of Reinforcement Learning (RL). RL is considered a subfield of Artificial Intelligence. Its relationship with Machine Learning, Supervised and Unsupervised Learning, and connections to Dynamic Programming, Game Theory, and Control Engineering. The diagram further branches out to show RL’s diverse applications in sectors such as healthcare, finance, robotics, energy, education, computer vision, and more, underlining its profound impact across industries.

A comprehensive book on reinforcement learning is R. Sutton’s and A. Barto’s book *Reinforcement Learning: An Introduction* [1]. Other books which cover the basics of reinforcement learning including practical examples and modern research areas and applications are M. Lapan’s book on *Deep Reinforcement Learning* [2] and the workshop book from Palmas et al. [3]. [4] gives theoretical background on modern Deep RL methods like PPO and A2C.

Section 2 gives an introduction to the theoretical concepts of reinforcement learning. Section 2.1 covers the key concept of a Markov Decision Process (MDP). The problem setting is to find an optimal strategy for the MDP in form of a policy which optimizes the total reward. If full knowledge of the MDP is given, dynamic programming principles can be applied to obtain an optimal strategy. Section 2.2 explains how the Bellman Equation is used to iteratively approximate a solution.

If no model of the MDP is given, then exploration methods must be used to learn from the interactions with the MDP as covered in Sect. 3. The methods are based upon dynamic programming principles as described previously, and they can be divided into on-policy and off-policy methods.

When observation spaces are based on image inputs from cameras or are otherwise too large to handle, then artificial neural networks are used to approximate an optimal strategy. This area is referred to as deep reinforcement learning.

Obtaining a solution for the optimization problem can either be done based on the approximation of an action-value function as described in Sect. 3.2 or the approximation of a policy as explained in Sect. 3.3. A combination of value and policy based methods are actor-critic methods as described in Sect. 3.4.

If the action space is continuous, then specially designed methods are used as carried out in Sect. 3.5.

2 Reinforcement Learning Fundamentals

The theoretical foundation of reinforcement learning is based upon Markov Decision Processes (MDPs), which provide a mathematical framework for modeling decision-making in stochastic environments. The central objective is to identify an optimal decision-making strategy that maximizes a specified objective function. The chapter covers the definition of an MDP, derives a policy-induced trajectory distribution, introduces value functions, and the Bellman Equation. The chapter closes with dynamic programming principles which are used to obtain a value function given full knowledge of the underlying MDP.

2.1 Markov Decision Processes (MDPs)

The following basic definitions from stochastic processes are frequently used in the formulation of MDPs and Reinforcement Learning.

Definition. The *conditional probability* of an event A given an event B is defined as $P(A|B) := \frac{P(A \cap B)}{P(B)}$.

If events A and B are independent, then $P(A|B) = P(A)$. In this case, the probability that both events A and B occur can be calculated as

$$P(A \cap B) = P(A) \cdot P(B). \quad (1)$$

The product rule for independent events is essential for calculations in MDPs.

Definition. If X is a random variable and p is its probability distribution, then the *expected value* of X is defined as

$$\mathbb{E}[X] := \int x \cdot p(X = x) dx \quad \text{or} \quad \mathbb{E}[X] := \sum_x x \cdot p(x) \quad (2)$$

depending on whether X is continuous or discrete.

Elements of an MDP. An MDP is composed of the following components:

- **States (\mathcal{S}):** A state represents a specific situation in the environment, the physical world, or system to be controlled. A state space may be high dimensional like the raw input from a camera. It may also be low dimensional like a 2D grid or a feature space. An MDP models a system that transitions between different states.

- **Observations (O):** Ideally all states can be fully observed. In this case $\mathcal{S} = O$. If not all states of an MDP can be fully observed, then it is called a *partially observable MDP* or POMDP. The *observation space* is used in RL to describe the set of all possible observable input (sensor input, feature information, odometric information, etc.).
- **Actions (A):** Actions are the decisions or moves that an agent can take in a given state. The set of all possible actions in a state is denoted as $A(s)$.
- **Transitions \mathcal{P} :** The transition probability function

$$P : \mathcal{S} \times A \times \mathcal{S} \rightarrow [0, 1], \quad (s, a, s') \mapsto P(s'|s, a) \in [0, 1] \quad (3)$$

describes the probability of transitioning from one state to another given a particular action. It is often represented as $P(s'|s, a)$, denoting the probability of transitioning to state s' from state s by taking action a .

The Markov property of an MDP states that the transition probability to a state s_{t+1} is only dependent on the predecessor state s_t and action a_t , not on past states, i.e. a MDP is memory-less

$$p(s_{t+1}|s_t, a_t) = p(s_{t+1}|a_t, s_t, a_{t-1}, s_{t-1}, \dots).$$

- **Rewards R :** Each state-action-state triple receives a numerical reward

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

The immediate reward is denoted as $R(s'|s, a)$. The expected immediate reward in a state s with action a can be expressed as

$$\mathbb{E}[R(s, a)] = \sum_{s' \in \mathcal{S}} P(s'|s, a) \cdot R(s'|s, a) \quad (4)$$

- **Policy (π):** A policy is a strategy that the agent follows to choose actions in each state. It can be deterministic

$$\pi : \mathcal{S} \longrightarrow A, \quad \pi(s) = a \in A \quad (5)$$

or stochastic

$$\pi : A \times \mathcal{S} \longrightarrow [0, 1], \quad \pi(a|s) = p \in [0, 1]. \quad (6)$$

When a policy π is given, the expected immediate reward in a state s can be written as

$$\mathbb{E}[R(s)] = \sum_{s' \in \mathcal{S}, a \in A} \pi(a|s) \cdot P(s'|s, a) \cdot R(s'|s, a) \quad (7)$$

- **Realizing an MDP:** A sequence of states $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \dots, s_T)$ is called a realization of an MDP for some starting state s_0 , when a_t is chosen according to π , i.e.

$$a_t = \pi(s_t) \text{ or chosen from the distribution } a_t \sim \pi(\cdot|s_t) \quad (8)$$

and $r_t = R(s_{t+1}|s_t, a_t)$. τ is also called trajectory or path. T can be a finite or infinite time horizon.

- **Total Return:** The cumulated reward or total return G for some realization τ is defined as

$$G := \sum_{t=0}^{T-1} \gamma^t R_t, \quad R_t = R(s_{t+1}|s_t, a_t) \text{ along } \tau \quad (9)$$

for some discount factor $\gamma \in]0, 1]$. The goal in RL is to find a policy π^* which maximizes the expected total reward $\mathbb{E}_{\tau \sim \pi}[G]$. $\tau \sim \pi$ denotes a realization of a trajectory given a policy π .

The discount factor γ may be set to 1 for MDPs which terminate after a finite time. For infinite horizon MDPs a discount factor $\gamma < 1$ is necessary to ensure existence of G .

Policy Induced Trajectory Distribution $\tau \sim \pi$. For a given policy π every concrete trajectory $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \dots, s_T)$ has a certain probability. The probability distribution for trajectories for a fixed policy π will be derived in this section.

According to (3) the probability for ending up in state s_1 when starting in state s_0 with a given action a_0 is $p(s_1|s_0, a_0)$. If a_0 is chosen according to the given policy $a_0 \sim \pi(\cdot|s_0)$, then the probability for the sequence (s_0, a_0, s_1) is

$$P((s_0, a_0, s_1)) = \pi(a_0|s_0) \cdot P(s_1|s_0, a_0) \quad (10)$$

Policy Induced Trajectory Distribution. Using the memory-lessness of an MDP together with the product rule for independent probabilities (1), the probability for a specific trajectory τ_{s_0} starting in state s_0 can be calculated as the product of the probabilities in (10)

$$P_\pi(\tau) = P_\pi(s_0, a_0, \dots, s_T) = \prod_{t=0}^{T-1} \pi(a_t|s_t) \cdot P(s_{t+1}|s_t, a_t) \quad (11)$$

Equation (11) is called the *policy induced trajectory distribution* for $\tau \sim \pi$.

This probability can be used to calculate the expected value of the total return $\mathbb{E}_{\tau \sim \pi}[G]$ over all trajectories τ starting in state s_0 .

Equation (12A) uses the definition of the expected value. (12B) shows that $\mathbb{E}_{\tau \sim \pi}[G]$ may be expressed as expected values of local rewards, using the *occupancy measure* d^π .

Using (9) and (11) the expected total return for G_π can be calculated as

$$\mathbb{E}_{\tau \sim \pi}[G_\pi] \stackrel{A}{=} \sum_{\tau} P_\pi(\tau) \cdot G_\pi(\tau) \stackrel{B}{=} \sum_s d^\pi(s) \cdot \mathbb{E}[R(s)] \quad (12)$$

Occupancy Measure. The value d^π as introduced in Eq. (12) is called the *discounted occupancy measure*.

$$d^\pi(s) = \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{T-1} \gamma^t \cdot \mathbb{I}(s_t = s) \right]$$

where $\mathbb{I}(s_t = s)$ is 1 if the $s_t = s$. The value d^π expresses the discounted frequency of visiting state s given a policy π . Figure 2 explains the property (12B) in a simple example. The expected total return in state A can hence be expressed in two different ways.

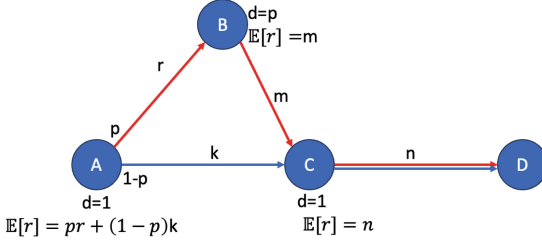


Fig. 2. The example illustrates Eq. (12) and shows a Markov Process which allows a red path $\tau_{\text{red}} = (A, B, C, D)$ and a blue path $\tau_{\text{blue}} = (A, C, D)$. The rewards are k, r, m, n as shown on the edges of the graph. The probability for τ_{red} is $P(\tau_{\text{red}}) = p \cdot 1 \cdot 1$. The probability for τ_{blue} is $P(\tau_{\text{blue}}) = (1-p) \cdot 1$. The expected total return for A is therefore $\mathbb{E}[G] = p \cdot (r + m + n) + (1-p) \cdot (k + n)$. Using the occupancy measure and expected one-step rewards in each state as shown in the image, alternatively the expected return can be calculated as $\mathbb{E}[G] = \sum_{s \in S} d(s) \cdot \mathbb{E}[r(s)] = 1 \cdot (pr + (1-p)k) + p \cdot m + 1 \cdot n$.

The following sections will deal with the expected total return, which will be called value function V , and methods for calculating it.

Value Functions. The objective is to find an optimal policy that maximizes the expected cumulative reward or total return. If in every state $s \in S$ a prediction of the expected total return G_s were known, then an optimal action a could be chosen which leads to a subsequent state in which the prediction of the expected total return is maximized. Such a prediction is called a state value function V .

State Value Function (V): The *state value function* $V(s)$ for a policy π in an MDP is defined as the expected total return starting from state s and following policy π thereafter. Mathematically, it can be expressed as:

$$V^\pi(s) := \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{T-1} \gamma^t R_t \mid S_0 = s \right] \quad (13)$$

In order to be able to calculate a greedy action or policy based on the state value function V^π , the transition properties of the MDP must be known. A greedy policy $\pi^{*,\pi}$ with respect to some value function V^π (based on a fixed policy π) is then calculated as

$$\pi^{*,\pi}(s) = \arg \max_{a \in A} \left[\sum_{s' \in S} P(s'|s, a) [R(s, a, s') + \gamma V^\pi(s')] \right] \quad (14)$$

In each state s the action $a^* = \pi^{*,\pi}(s)$ is chosen, which maximizes the expected total return when taking one step with a^* and continuing with policy π thereafter. Note that $\pi^{*,\pi}$ is generally not an optimal policy. It is a greedy policy based on the value function V^π of given policy π .

If the transition properties of the MDP are not known, then the action value function Q^π can serve as a prediction for the expected total return.

Action Value Function (Q): The *action-value function* $Q(s, a)$, for a policy π , in an MDP is defined as the expected return starting from state s , taking an action a , and thereafter following policy π . It can be formally written as:

$$Q^\pi(s, a) = \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{T-1} \gamma^k R_t \mid S_t = s, A_t = a \right] \quad (15)$$

The difference to the above is, that Q also depends on an action a . This way an optimal action can be chosen by taking the action a^* which maximizes the Q -value in the current state s :

$$a^* = \arg \max_{a \in A} Q(s, a) \quad (16)$$

Comparing (14) and (16) shows that in the latter case no transition information of the MDP is necessary in order to choose a greedy action in every state.

Bellman Equations. The Bellman Equations form an essential component of the algorithmic structure of dynamic programming and reinforcement learning. They express that the value of a state can be expressed as an immediate reward plus the value in the next state.

The **Bellman Expectation Equation** for the **state value function** V given a policy π is given by

$$V^\pi(s) = \sum_{a \in A} \pi(a|s) \cdot \sum_{s' \in S} P(s'|s, a) [R(s, a, s') + \gamma V^\pi(s')] \quad (17)$$

The **Bellman Expectation Equation** for the **action value function** Q given a policy π is given by

$$Q^\pi(s, a) = \sum_{s' \in S} P(s'|s, a) \cdot \left[R(s, a, s') + \gamma \sum_{a' \in A} \pi(a'|s') Q^\pi(s', a') \right] \quad (18)$$

The **Bellman Optimality Equation** for the **state value function** is:

$$V^*(s) = \max_{a \in A} \sum_{s' \in S} P(s'|s, a) \cdot [R(s, a, s') + \gamma V^*(s')] \quad (19)$$

The **Bellman Optimality Equation** for the **action value function** Q is given by:

$$Q^*(s, a) = \sum_{s' \in S} P(s'|s, a) \cdot [R(s, a, s') + \gamma \max_{a' \in A} Q^*(s', a')] \quad (20)$$

The Bellman Equations form a basis for the following algorithms.

2.2 Solving an MDP: Dynamic Programming

If the transition information of the MDP is known, then Eq. (17) can be applied iteratively to calculate an optimal policy π^* . The methods are explained in the following section.

Policy Iteration and Value Iteration. In **Policy Iteration** an optimal policy is directly calculated using the following two steps:

1. **Policy Evaluation:** Compute V^π for the current policy, iteratively applying the Bellman Expectation Equation (17).
2. **Policy Improvement:** Update the policy π to a greedy policy $\pi^{*,\pi}$ by selecting in every state the action which maximizes the expression in the Eq. (14).
3. Repeat until convergence.

Figure 3a shows the progression of the algorithm for a simple grid example. The heat maps in the top row show the value function, the bottom row shows the greedy policy with respect to the current value function.

In **Value Iteration** the optimal value function V^* is calculated directly using iterative application of the Bellman Optimality Equation (19). It can be said that policy evaluation and policy improvement are integrated into a single update, iterating until convergence as shown in Fig. 3b.

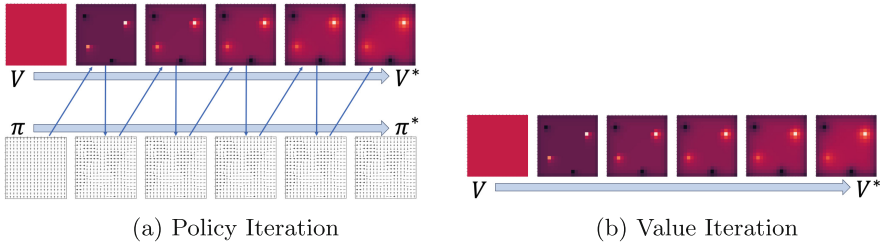


Fig. 3. In *policy iteration* (left) a policy is evaluated and then improved acting greedily. Often it is not necessary to calculate the value function precisely as the policy converges fast to an optimal policy. In *value iteration* (right) the optimal value function is directly approximated using the Bellman Optimality Equation (19).

Dynamic Programming Iteration Methods: There exist various methods for the sequence in which the states are updated when applying policy evaluation or value iteration. The methods are depicted in Fig. 4.

Synchronous Backups: In each iteration, the algorithm computes the new value for every state based on the old values (from the previous iteration) and updates them all at once at the end of the iteration. This method ensures that

