

Reinforcement Learning mit adaptiver Steuerung von Exploration und Exploitation

Dissertation zur Erlangung des Doktorgrades Dr. rer. nat.
der Fakultät für Ingenieurwissenschaften und Informatik
der Universität Ulm

vorgelegt von
Michel Tokic
aus Dresden



Universität Ulm
Institut für Neuroinformatik
2013

Danksagungen

Meinem Doktorvater Günther Palm möchte ich herzlich dafür danken, dass er mich während dem Anfertigen dieser Arbeit mit zahlreichen fruchtbaren Gesprächen und seiner ausdauernden Geduld begleitete. Ebenso ein herzliches Dankeschön an Martin Riedmiller für seine konstruktiven Anmerkungen und die Bereitschaft das Korreferat zu übernehmen. Großer Dank gebührt auch Birgit Allgaier, Richard Cubek, Friedhelm Schwenker und Benjamin Stähle, die mir in den letzten Zügen dieser Arbeit halfen, manch holprigen Satz deutlicher darzustellen. Ein großes Dankeschön auch an Holger Voos und Wolfgang Ertel, die mich in ihren Forschungsgruppen teilhaben ließen und mir hierdurch die Mitarbeit in der Wissenschaft ermöglichten. Philipp Ertle bin ich unheimlich dankbar für die unzähligen Gespräche in den letzten Jahren, egal ob in unserem Arbeitszimmer oder auf einer unserer schönen Dienstreisen. Schlussendlich auch ein herzliches Dankeschön an Julia, Nino, Laila, Delphine und Mama, insbesondere für ihre Unterstützung bezüglich meiner Berufswahl, die in den letzten Jahren oftmals viel Verständnis am Wochenende und zur Ferienzeit erforderte.

Amtierende Dekanin: **Prof. Dr. Tina Seufert**
Erstgutachter: **Prof. Dr. Günther Palm**
Zweitgutachter: **Prof. Dr. Martin Riedmiller**
Tag der Promotion: **01. Oktober 2013**

Zusammenfassung (Deutsch)

Anhand berechenbarer Modelle für Reinforcement Learning (RL) kann intelligentes Verhalten, z. B. für autonome Roboter, durch sensomotorische Interaktion erlernt werden (Sutton und Barto, 1998). Die Art und Weise des Lernens ist neurobiologisch und psychologisch motiviert, wobei ein künstlicher Lernagent von seiner Umgebung *Reward* für den Nutzen getätigter Aktionen erhält. Hieraus ergibt sich das natürliche Lernziel, den kumulierten Reward zu optimieren.

Um dieses ehrgeizige Ziel erreichen zu können, muss die Aktionswahl intelligent gesteuert werden. Hierbei ist es von Bedeutung, dass neben der Wahl von *Exploitationsaktionen*, die bereits erworbenes Wissen über die Umgebung ausnutzen, auch *Explorationsaktionen* getätigt werden, welche die Dynamik der Umgebung erkunden. Dabei dürfen jedoch nicht zu viele Explorationsaktionen getätigt werden, um niedrigen Reward durch schlechte Aktionen zu vermeiden; gleichzeitig aber auch nicht zu wenige, um das Wissen über die langfristige Auswirkung von Aktionen möglichst präzise abschätzen zu können. Hierdurch entsteht das bekannte *Dilemma zwischen Exploration und Exploitation* (Thrun, 1992).

Diese Dissertation stellt neue Explorationsstrategien für modellfreies Reinforcement Learning in diskreten Aktionsräumen vor. Es wird das Ziel verfolgt, die Explorationsrate eines Lernagenten nicht global vom Experimentator festlegen zu lassen, sondern durch *Meta-Learning*, auf Basis des Lernfortschritts, zu adaptieren. In diesem Zusammenhang wird gezeigt, dass zur Zielerreichung zwei Grundelemente wichtig sind: (1) der TD-Fehler, welcher den Unterschied zwischen der Prädiktion und Observation von Reward angibt; (2) die Lernrate zur Gewichtung des TD-Fehlers in der Wertefunktion. Letztere ist wichtig, um das Lernen mit stochastischen Rewards zu ermöglichen, bei welchen die Prädiktion selten mit der Observation übereinstimmt. In diesem Zusammenhang werden Wertunterschiede als Maß für die *Sicherheit über die Auswirkung von Aktionen* verwendet, die beim Aktualisieren (Neuschätzen) der Wertefunktion entstehen. In einem weiteren Ansatz werden „stochastische Neuronen“ eingesetzt, um das Explorationsverhalten nicht nur lokal, sondern auch global zu steuern. Ebenso werden die technischen Beiträge in den Kontext der Neurobiologie eingeordnet, in welcher die folgenden Neurotransmitter eine wichtige Rolle spielen: Dopamin (TD-Fehler), Acetylcholin (Lernrate) und Norepinephrin (Explorationsparameter) (Doya, 2002).

Da das Explorationsverhalten nicht explizit vom Experimentator gesteuert wird, sondern vom Inneren des Lernagenten, sind die Ergebnisse dieser Arbeit ein wichtiger Schritt in Richtung vollautonome Lernagenten. Besondere Anforderungen ergeben sich daher durch: *partiell beobachtbare Umgebungen*, *stochastische Rewards* und *nichtstationäre Prozesse*. Die entwickelten Explorationsstrategien verfolgen das Ziel, diese Anforderungen möglichst *robust* zu adressieren.

Abstract (Englisch)

Using computational models of reinforcement learning (RL), intelligent behavior based on sensorimotor interactions can be learned (Sutton and Barto, 1998). The way of learning is inspired from neurobiology and psychology, where an artificial agent performs actions within its environment that responds with a *reward* signal describing the action's utility. Therefore, the natural objective is optimizing the cumulative reward over time.

For achieving this ambitious goal, actions must be selected in an intelligent manner. In this context, it is important to select actions exploiting learned knowledge so far (short-term optimization), but also actions exploring the dynamics of the environment (long-term optimization). One of the main problems in deciding between both is that excessive selections of exploratory actions often lead to negative rewards, because of not being goal directed. On the other hand, exploiting uncertain environment knowledge may lead to sub-optimal action selections, because of underestimating optimal actions. As a result, the *dilemma between exploration and exploitation* arises (Thrun, 1992).

This dissertation presents new action-selection policies for model-free reinforcement learning with discrete actions. The aim is to learn exploration meta-parameters based on learning progress rather than selecting them from hand by the experimenter. It is shown that two basic elements are important: (1) the TD error, which is the difference between the prediction and observation of rewards; and (2) the learning rate for weighting the TD error in the value function. The latter one is important for learning with stochastic rewards, where the observation rarely equals the prediction. In this context, value differences are interpreted as a measure for the *certainty about the consequence of actions*, which arise at times of updating the value function. Furthermore, "stochastic neurons" are deployed for adapting exploratory behavior not only locally, but also in a global man-

ner. Furthermore, the technical contributions of this thesis are sort into the context of neurobiology, in which the following neurotransmitters play an important role: dopamine (TD error), acetylcholine (learning rate) and norepinephrine (exploration parameter) (Doya, [2002](#)).

Because exploratory behavior is explicitly not controlled by the experimenter, but adapted by the interior of the agent, the results of this work are an important step towards the development of fully autonomous learning agents. Hence, requirements to be respected arise from: *partially-observable environments, stochastic rewards and non-stationary processes*. The developed action-selection policies aim at addressing these requirements being *robust*.

Inhaltsverzeichnis

Zusammenfassung (Deutsch)	II
Zusammenfassung (Englisch)	III
Abbildungsverzeichnis	IX
Glossar	XIII
1 Einleitung	1
1.1 Zielsetzung dieser Arbeit	3
1.2 Wissenschaftlicher Beitrag	4
1.3 Aufbau dieser Arbeit	5
2 Berechenbare Modelle für Reinforcement Learning	7
2.1 Die Agent-Umgebung-Interaktion	7
2.2 Lernprobleme	8
2.2.1 Episodische Lernprobleme	9
2.2.2 Kontinuierliche Lernprobleme	9
2.2.3 Gemeinsame Notation	10
2.3 Markovsche Entscheidungsprozesse	10
2.4 Wertefunktionen	12
2.5 Optimale Wertefunktionen	15
2.6 Modellbasiertes Lernen	16
2.6.1 Strategie-Evaluierung	17
2.6.2 Strategie-Verbesserung	17
2.6.3 Strategie-Iteration	18
2.6.4 Wert-Iteration	18
2.7 Modellfreies Lernen	19
2.7.1 Q-learning	20
2.7.2 Sarsa	21
2.8 Analogien zur Verhaltenspsychologie und Neurobiologie . .	22

2.9	Diskussion	24
3	Related Work: Exploration und Exploitation	26
3.1	Die Greedy-Strategie	27
3.1.1	Die ε -Greedy-Strategie	28
3.1.2	Die decreasing- ε -Strategie	29
3.1.3	Die ε -first-Strategie	30
3.2	Die Softmax-Strategie	31
3.3	Meta-Lernstrategien	33
3.3.1	Reward basierte Adaption	33
3.3.2	TD-Fehler basierende Adaption	35
3.3.3	Aktionswert basierte Adaption	36
3.4	Die Reinforcement-Comparison-Strategie	37
3.5	Die Pursuit-Strategie	38
3.6	Counter-Based-Exploration-Strategien	39
3.6.1	Counter-Based-Exploration mit Abschwächung	40
3.6.2	Die Error-/Counter-Based-Exploration-Strategie	41
3.6.3	Die Recency-Based-Exploration-Strategie	41
3.7	Die Strategie der optimistischen Initialwerte	42
3.8	Diskussion	42
4	VDBE-Strategien	46
4.1	Die Idee	47
4.2	Die VDBE-Strategie	47
4.3	Alternative Aktivierungsfunktionen	49
4.4	Konvergenzanalyse	52
4.4.1	Große Zustandsräume	52
4.4.2	Konvergenz in deterministischen Umgebungen	53
4.4.3	Konvergenz in stochastischen Umgebungen	55
4.5	Das n -armige Banditenproblem	58
4.5.1	Untersuchung	58
4.5.2	Ergebnisse	59
4.6	Die VDBE-Softmax-Strategie	61
4.7	Das Bandit-World-Problem	62
4.7.1	Untersuchung	63
4.7.2	Ergebnisse	63
4.8	Diskussion	64

5	Problemdimensionen & Experimente	66
5.1	Lernprobleme im Reinforcement Learning	66
5.2	Klassifikation von Lernproblemen	69
5.3	Das Cliff-Walking-Problem	70
5.3.1	Experimentelle Untersuchung	72
5.3.2	Ergebnisse	72
5.3.3	Diskussion	75
5.4	Der Laufroboter	75
5.4.1	Die Architektur	76
5.4.2	Partielle Beobachtbarkeit	78
5.4.3	Experimentelle Untersuchung	79
5.4.4	Ergebnisse	81
5.5	Das diskrete Mountain-Car-Problem	83
5.5.1	Partielle Beobachtbarkeit	84
5.5.2	Experimentelle Untersuchung	85
5.5.3	Ergebnisse	85
5.6	Diskussion	86
6	Parameteradaption mit stochastischen Neuronen	88
6.1	Das Lernproblem	89
6.2	Globale Adaption	90
6.3	Lokale Adaption	94
6.4	Das Dynamic-Cliff-Problem	95
6.4.1	Experimentelle Untersuchung	96
6.4.2	Ergebnisse	97
6.5	Das Mountain-Car-Problem mit zwei Zielzuständen	97
6.5.1	Experimentelle Untersuchung	100
6.5.2	Ergebnisse	101
6.6	Diskussion	102
7	Psychologische und neurobiologische Aspekte	104
7.1	Psychologische Aspekte	104
7.1.1	Klassisches Konditionieren	105
7.1.2	Operantes Konditionieren	106
7.1.3	Fazit	107
7.2	Neurobiologische Aspekte	108
7.2.1	Unsupervised Learning in der Großhirnrinde	108
7.2.2	Reinforcement Learning in den Basalganglien	110
7.2.3	Supervised Learning im Kleinhirn	111
7.2.4	Steuerung von Exploration/Exploitation	111

7.3	Einordnung der technischen Beiträge in die Neurobiologie .	112
7.3.1	VDBE-Strategien	113
7.3.2	REINFORCE-Strategien	114
7.3.3	Interaktion mit dem Kleinhirn	114
7.4	Diskussion	115
8	Schlussbetrachtung und Ausblick	117
8.1	Beiträge zur Zielsetzung dieser Arbeit	118
8.2	Limitierungen	121
8.3	Ausblick	122
	Literaturverzeichnis	124
A	Plots	136
	Cliff-Walking-Problem mit $\alpha = 0.2$	137
	Cliff-Walking-Problem mit $\alpha = 0.8$	138
	Laufroboter	139
	Bandit-World-Problem	140
	Mountain-Car-Problem	141
	Dynamic-Cliff-Problem	142
	Dynamic-Cliff- sowie Mountain-Car-Problem	143
	Mountain-Car-Problem mit zwei Zielzuständen	144
B	Lebenslauf	145
C	Veröffentlichungen	148

Abbildungsverzeichnis

2.1	Interaktions-Schema des Agenten mit seiner Umgebung (nach Sutton und Barto (1998)).	8
2.2	Reward-Diskontierung nach Gleichung (2.5): Ein Diskontierungsfaktor in Höhe von $\gamma = 0.9$ führt dazu, dass der hohe Reward zum Zeitpunkt $t = 9$, im Return R_t stärker berücksichtigt wird, woraus resultiert, dass dieser mit $R_t = 0.062$ positiv ist, obwohl zunächst negative Rewards erhalten werden. Ein niedrigerer Diskontierungsfaktor in Höhe von $\gamma = 0.5$ berücksichtigt hingegen den hohen Reward fast gar nicht, weswegen der Return mit $R_t = -0.093$ ein negativer Wert ist (Abb. in Anlehnung an (Doya, 2002)).	10
3.1	Die ε -Greedy-Strategie in Abhängigkeit unterschiedlicher Explorationsraten ε . Man beachte die gleichverteilte Aktionswahl derjenigen Aktionen, die nicht den höchsten Q -Wert besitzen. Man beachte ebenso, dass für $\varepsilon = 1$ alle Aktionen mit selbiger Wahrscheinlichkeit gewählt werden würden.	29
3.2	Zeitlicher Abfall der Explorationsrate bei ε -first- sowie decreasing- ε -Strategien.	30
3.3	Die Softmax-Strategie in Abhängigkeit unterschiedlicher Temperaturen τ . Man beachte das erfolgreiche Unterdrücken der Aktion mit dem Q -Wert -200	32
4.1	Vergleich verschiedener Aktivierungsfunktionen $f(s, a, \sigma)$ einer VDBE-Strategie: (a) zeigt das Verhalten einer Boltzmann-Aktivierung für unterschiedliche Belegungen der inversen Empfindlichkeit σ ; (b) zeigt hingegen das Verhalten für $\sigma = 1$ für die Boltzmann-Aktivierung (Gleichung 4.2), die exponentielle Aktivierung (Gleichung 4.5) sowie die hyperbolischen Aktivierung (Gleichung 4.6).	51

4.2	Vergleich des Lernverhaltens in Abhängigkeit unterschiedlicher Lernraten: (a) für deterministische Rewards und (b) für stochastische Rewards.	56
4.3	Ergebnisse des n -armiges Banditen-Problems (Tokic, 2010). Vergleich des mittleren Rewards pro Aktion für (a) ε -Greedy, (b) Softmax und (c) VDBE. Abbildung (d) zeigt für VDBE die Explorationsrate $\varepsilon(s)$, in Abhängigkeit der inversen Sensibilität σ	60
4.4	Bandit-World-Problem: Skizze nach Tokic und Palm (2011). .	63
5.1	Cliff-Walking-Problem: Der <i>safe path</i> wird von Sarsa mit stochastischen Strategien gelernt, hingegen der (gefährlichere) <i>optimal path</i> von Q -learning. Zustand S markiert den Startzustand, hingegen G den Zielzustand. Skizze nach (Sutton und Barto, 1998).	70
5.2	Cliff-Walking-Problem: Modellierte Umgebung im Gridworld-Editor der Teaching-Box (Ertel u. a., 2009; Tokic u. a., 2010a).	73
5.3	Laufroboter: (a) zeigt das Modell nach Kimura u. a. (1997) sowie (b) den entwickelten Hardware-Roboter (Tokic und Bou Ammar, 2012; Tokic u. a., 2009, 2010a).	77
5.4	Laufroboter: Das 5×5 Zustandsraum-Modell (links) sowie eine zyklische Strategie zur Vorwärtsbewegung (rechts). Zustände innerhalb des Zyklus sind mit \odot markiert.	77
5.5	Laufroboter: Der Gridworld-Editor der Teaching-Box (Tokic u. a., 2010a): (a) zeigt die aufgezeichneten Rewards von einem Linoleumboden und (b) die dazugehörige Wertefunktion, welche durch Wert-Iteration mit $\gamma = 0.99$ erlernt wurde. Rote Pfeile in (a) kennzeichnen eine suboptimale Strategie, hingegen grüne und schwarze Pfeile in (a)+(b) die tatsächlich optimale Strategie. Der grüne Zyklus in (b) besitzt einen mittleren Reward pro Aktion in Höhe von $\bar{r}_{t+1} = \frac{17+20-6-8}{4} = 5.75$	80
5.6	Mountain-Car-Problem: Skizze nach (Sutton und Barto, 1998). .	83

6.1	Modell eines stochastischen REINFORCE-Neurons im Verbund eines neuronalen Netzwerkes. Die Eingabe des Neurons kann anhand von vorgeschalteten Neuronen (mit evtl. andersartiger Aktivierungsfunktion) bestimmt werden, oder durch tabellarische Speicherung. Die Ausgabe (Aktivierung) des Neurons wird anhand einer Normalverteilung bestimmt, $a_e \sim \mathcal{N}(\mu, \sigma)$, und als kontinuierliche Aktion in der Umgebung (Williams, 1992) oder als Explorationsparameter (Tokic und Palm, 2012a,b) ausgeführt. Die Umgebung liefert die Bestärkung in Form von ρ zurück, wodurch die Parameter μ und σ , auf Basis von $\bar{\rho}$ und ρ , nach REINFORCE adaptiert werden.	91
6.2	Beispiel einer REINFORCE-Adaptierung: (a) Anpassung der Normalverteilung in Richtung von a_e , falls $\rho > \bar{\rho}$. (b) Anpassung der Normalverteilung in entgegengesetzter Richtung, falls $\rho < \bar{\rho}$	93
6.3	Dynamic-Cliff-Problem: Schematische Darstellung.	96
6.4	Dynamic-Cliff-Problem: Reward- sowie Klippenabsturz-Bandbreite in Abhängigkeit des untersuchten Parameterbereichs (Tokic u. a., 2012), für konstante Belegungen des Explorationsparameters (ohne REINFORCE).	98
6.5	Dynamic-Cliff-Problem: Entwicklung des Mittelwerts und der Standardabweichung für durch REINFORCE adaptierte Basisstrategien. Man beachte die Dynamik der Exploration zum Zeitpunkt von Nichtstationaritäten.	99
6.6	Mountain-Car-Problem mit zwei Zielzuständen: Skizze. . . .	100
7.1	Drei grundlegende Lernparadigmen: (A) Supervised Learning durch Fehlerpropagierung; (B) Reinforcement Learning durch ein Reward-Signal; (C) Unsupervised Learning durch Statistiken basierend auf dem Input-Signal.	109
A.1	Cliff-Walking-Problem: Return für Lernrate $\alpha = 0.2$ über 1000 Experimente gemittelt.	137
A.2	Cliff-Walking-Problem: Return für Lernrate $\alpha = 0.8$ über 1000 Experimente gemittelt.	138
A.3	Laufroboter: Return über 2000 Experimente gemittelt.	139
A.4	Bandit-World-Problem: Return über 500 Experimente gemittelt.	140

A.5	Mountain-Car-Problem: Return über 2000 Experimente gemittelt (und geglättet). Aufgrund von schlechten Ergebnissen liegen Kurven mit ** gekennzeichneten Parametern außerhalb des Darstellungsbereichs.	141
A.6	Dynamic-Cliff-Problem: Return der Basisstrategien über 500 Experimente gemittelt. Aufgrund von schlechten Ergebnissen liegen Kurven mit ** gekennzeichneten Parametern außerhalb des Darstellungsbereichs.	142
A.7	Dynamic-Cliff- sowie Mountain-Car-Problem mit zwei Zielzuständen: Return für REINFORCE-adaptierte Basisstrategien.	143
A.8	Mountain-Car-Problem mit zwei Zielzuständen: Return der Basisstrategien über 200 Experimente gemittelt. Aufgrund von schlechten Ergebnissen liegen Kurven mit ** gekennzeichneten Parametern außerhalb des Darstellungsbereichs.	144

Glossar

Reinforcement Learning (Kapitel 2 und 7)

CS	Konditionierter Stimuli im Rescorla-Wagner-Modell.
E	Erwartungswert einer Zufallsvariablen.
E_π	Erwartungswert einer Zufallsvariablen in Abhängigkeit der Strategie π .
$Pr(\cdot \cdot)$	Bedingte Wahrscheinlichkeit.
$Q(s, a)$	Geschätzter Wert von Aktion a in Zustand s .
Q^*	Optimale Wertefunktion für Zustand-/Aktionspaare.
Q^π	Wertefunktion für Zustand-/Aktionspaare unter der Strategie π .
$Q^n(s, a)$	Geschätzter Wert von Aktion a in Zustand s nach einem Update, durch z. B. Q -learning oder Sarsa.
$Q_j(s, a)$	Geschätzter Wert von Aktion a in Zustand s zur j -ten Update-Iteration.
R_t	Kumulierter und ggf. diskontierter Reward (Return) ab Zeitschritt t .
T	Zeitpunkt beim Erreichen des Zielzustands.
US	Unkonditionierter Stimuli im Rescorla-Wagner-Modell.
$V(CS)$	Wert des prädizierten Verhaltens für Stimulus CS (Rescorla-Wagner Modell).
$V(s)$	Geschätzter Wert von Zustand s .
V^*	Optimale Wertefunktion für Zustände.
V^π	Wertefunktion für Zustände unter der Strategie π .
Δ	Temporal-Difference Fehler (TD-Fehler).
α	Lernrate (Schrittweitenparameter, engl. step-size parameter).
ϵ_j	Fehler in Q_j in Bezug auf Q^* .
η	Lernrate im Rescorla-Wagner-Modell.
γ	Diskontierungsfaktor.
$\mathcal{A}(s)$	Die Menge aller Aktionen in Zustand s .
$\mathcal{A}^*(s)$	Die Menge aller Aktionen mit dem höchsten Q -Wert in Zustand s .

\mathcal{P}	Menge aller Transitionswahrscheinlichkeiten.
$\mathcal{P}_{ss'}^a$	Transitionswahrscheinlichkeit von Zustand s nach Zustand s' , im Falle, dass Aktion a gewählt wird.
\mathcal{R}	Menge aller Rewards.
$\mathcal{R}_{ss'}^a$	Reward für den Übergang von Zustand s nach Zustand s' , im Falle, dass Aktion a gewählt wird.
\mathcal{S}	Die Menge aller Zustände ohne Zielzustände.
\mathcal{S}^+	Die Menge aller Zustände mit Zielzuständen.
π	Die Strategie des Agenten (Mapping von Zuständen auf Aktionen).
π'	Eine verbesserte Strategie.
$\pi(s, a)$	Auswahlwahrscheinlichkeit für Aktion a in Zustand s .
θ	Ein Schwellwert.
a	Eine Aktion.
a^*	Aktion mit dem höchsten Q -Wert in Zustand s . Im Falle, dass es mehrere Aktionen mit gleichem Q -Wert gibt, wird eine zufällige aus dieser Menge gewählt.
r_{t+1}	Der Reward zum Zeitpunkt $t + 1$.
s	Ein Zustand (Abkürzung für s_t).
s'	Ein Folgezustand (Abkürzung für s_{t+1}).
t	Ein diskreter Zeitschritt.

Explorationsverfahren (Kapitel 3)

Δ	TD-Fehler Δ .
$\Phi(s, a)$	Aufsummierter Wert aller möglichen Folgezustände s' nach Ausführung von Aktion a in Zustand s , unter Berücksichtigung der Transitionswahrscheinlichkeit $\mathcal{P}_{ss'}^a$.
$\alpha(t)$	Lernrate α zum Zeitpunkt t .
α_{Δ}	Lernrate für den gleitenden Mittelwert $\bar{\Delta}$.
$\bar{\Delta}$	Gleitender Mittelwert über den TD-Fehler Δ .
β	Eine Lernrate.
$\epsilon(t)$	Aktivitätsterm zum Zeitpunkt t (Kombination aus mid- und long-term Reward).
$\gamma(t)$	Diskontierungsfaktor γ zum Zeitpunkt t .
λ	Ein Abschwächungsfaktor.
$\bar{r}(s)$	Gleitender Mittelwert über Rewards aller Aktionen in Zustand s (Referenz-Reward).
ρ	Eine Lernrate.
τ	Globaler Temperatur-Parameter.
$\tau(t)$	Temperatur-Parameter τ zum Zeitpunkt t .

τ_{LT}	Lernrate für $r_{LT}(t)$.
τ_{MT}	Lernrate für $r_{MT}(t)$.
ε	Globale Explorationsrate.
ζ	Zeitstempel t des letzten Besuchs von Zustand s .
a_ϵ	Lernrate des Reward-Terms im Aktivitätsterm $\epsilon'(t)$.
$c(s)$	Ein Counter über wie oft Zustand s bislang durchlaufen wurde.
$p(s, a)$	Aktionspräferenz für Aktion a in Zustand s .
$r(t)$	Unmittelbarer Reward für das Tätigen einer Aktion (entspricht r_{t+1}).
$r_{LT}(t)$	Long-term Reward (gleitender Mittelwert über $r_{MT}(t)$).
$r_{MT}(t)$	Mid-term Reward (gleitender Mittelwert über den unmittelbaren Reward).

VDBE-Strategien (Kapitel 4)

$\delta(s)$	Lernrate für $\varepsilon(s)$.
σ	Inverse Empfindlichkeit.
$\varepsilon(s)$	Explorationsrate ε in Zustand s .
ξ	Eine Gleichverteilte Zufallszahl.
$f(s, a, \sigma)$	Aktivierungsfunktion.

REINFORCE (Kapitel 6)

R	Episodischer Return.
R_{greedy}	Episodischer Return einer Greedy-Strategie.
R_{random}	Episodischer Return einer Random-Strategie.
Δ	Differenz zwischen zwei Zeitschritten.
α	Lernrate für die Anpassung von θ .
$\bar{\rho}$	<i>Reinforcement Baseline</i> bei REINFORCE-Algorithmen (durchschnittliche Belohnung).
$\mathcal{N}(\mu, \sigma)$	Normalverteilung mit Mittelwert μ und Standardabweichung σ .
μ	Mittelwert einer Normalverteilung.
$\nabla_{\theta} \rho$	Gradient von ρ an der Stelle θ .
ρ	Belohnung für das Wählen des Explorationsparameters a_e .
σ	Standardabweichung einer Normalverteilung.
θ	Parametervektor der zu adaptierenden Verteilung ($\sigma + \mu$).
a_e	Explorationsparameter auf Basis der Normalverteilung: $a_e \sim \mathcal{N}(\mu, \sigma)$.
e	<i>Characteristic Eligibility</i> bei REINFORCE-Algorithmen.
g	Wahrscheinlichkeitsdichtefunktion.

Kapitel 1

Einleitung

Verhalten anhand sensomotorischer Interaktion zu erlernen, ist eine Schlüsseltechnologie für Problemstellungen, die für einen Ingenieur anhand klassischer Programmierung sehr schwer zu lösen sind. Interdisziplinär fließen im *Reinforcement Learning* (RL) Aspekte aus der Psychologie, den Neurowissenschaften, der Informatik und der Mathematik zusammen, um das Verhalten von künstlichen Agenten möglichst optimal zu erlernen (Sutton und Barto, 1998). Ein Agent erhält für getätigte Aktionen *Reward* von seiner Umgebung, welcher als Belohnungs- bzw. Bestrafungssignal zu verstehen ist. Das Ziel ist, den kumulierten Reward über die Zeit zu maximieren, was durch Prädiktion zukünftiger Rewards, auf Basis von bislang erlerntem Wissen, erfolgt. Derartig kognitives Lernen ermöglicht das Finden von Lösungen für komplexe Problemstellungen, z. B. Strategien für Brettspiele (Eck und Wezel, 2008; Faußer und Schwenker, 2010; Tesauro, 2002; Thrun, 1995), nichtlineare motorische Basisfähigkeiten humanoider Roboter (Peters und Schaal, 2008) oder das Optimieren technischer Systeme (Hans, 2007; Hans u. a., 2008).

Um den Reward eines Agenten zu maximieren, muss dessen Aktionswahl intelligent gesteuert werden. Zum Einen ist es wichtig, eine bestimmte Anzahl von zufälligen *Explorationsaktionen* zu wählen, durch welche die Umgebung erkundet wird. Dieses trägt zum Wissen über die langfristige Auswirkung von Aktionen bei, kann jedoch aufgrund des Zufalls ebenso negativen Reward einbringen. Deswegen ist es zum Anderen ebenso wichtig *Exploitationsaktionen* zu wählen, die erworbenes Wissen über die Umgebung ausnutzen, mit dem Ziel möglichst viel Reward zu erhalten. Man

beachte jedoch, dass ein frühzeitiges Ausnutzen von unzureichend explorierten Aktionen auch dazu führen kann, dass die tatsächlich optimale Aktion unterschätzt wird, und demnach nicht der höchstmögliche Reward erzielt wird. Hierbei entsteht das bekannte *Dilemma zwischen Exploration und Exploitation*, welches wir aus Entscheidungsfindungen unseres alltäglichen Lebens kennen: z. B. welchen Partner wir wählen, welche Forschungsprojekte wir durchführen oder welche Eissorte uns am besten schmeckt (Cohen u. a., 2007). All diese Entscheidungsfindungen erfordern ein vorheriges Erkunden von Alternativen, bevor wir uns an eine einzige Option langfristig binden.

In der Modellierung derartig kognitiven Verhaltens ist der Übergang von Exploration nach Exploitation eine große Herausforderung. Grundlegend wird davon ausgegangen, dass je öfters Aktionen getätigt werden, das Wissen über deren Auswirkung sicherer wird. Werden jedoch Exploitationsaktionen auf Basis von unvollständigem Wissen getätigt, kann dies dazu führen, dass suboptimale Aktionen gewählt werden, da die Auswirkung von optimalen Aktionen vom Agenten unterschätzt wird. Auf der anderen Seite kann jedoch das Wählen einer Explorationsaktion negativen Reward einbringen, da der Agent nicht wissen kann, welche Auswirkung eine bislang noch nicht getätigte Aktion besitzt; zum Beispiel wenn wir bemerken dass Paranüsse allergische Symptome in unserem Körper auslösen. Ebenso gibt es aber auch Aktionssequenzen mit vergleichsweise niedrigem Reward, die jedoch in Zustände führen können, in denen ein relativ hoher Reward erhalten wird. Durch die zeitliche Verzögerung wird dies erst nach vielen Aktionen bemerkt; zum Beispiel das Mattsetzen des gegnerischen Königs in einer Schachpartie, was nicht nur durch die zuletzt getätigte Aktion ermöglicht wurde. Die Problematik von verzögerten Rewards wird diesbezüglich von Sutton als das *Temporal Credit-Assignment Problem* bezeichnet (Sutton, 1984). Zum Lösen dieses Problems muss der Reward daher auch auf (u. U. sehr viele) zuvor getätigte Aktionen zurückpropagiert werden, um die entsprechende Aktionssequenz zur Zielerreichung erlernen zu können. Ebenso muss ein Tier von Zeit zu Zeit nach neuen Futterquellen Ausschau halten, da diese in der Natur typischerweise nicht-stationär sind. Zusammengefasst sieht man deutlich, dass die Balance zwischen Exploration und Exploitation in hohem Maße für die Maximierung des Rewards entscheidend ist.

Stand der Forschung in der RL-Literatur ist, dass unterschiedliche Ansätze zur Steuerung von Exploration und Exploitation existieren, jedoch bis

heute keine allgemeingültige, optimale Balance existiert (Cohen u. a., 2007). Einige Ansätze verfolgen das Ziel, in diskreten Zustands- und Aktionsräumen das Explorationsproblem mit Aktionszählern sehr effizient zu lösen (Auer, 2002; Auer u. a., 2002; Thrun, 1992; Thrun und Möller, 1992), sind jedoch in Zustandsräumen mit kontinuierlichen Dimensionen unpraktikabel (Nissen, 2007; Rummery und Niranjan, 1994), da praktisch unendlich viele Explorationsdaten approximiert werden müssen, deren Funktion nicht konvergent ist. Im Gegensatz dazu balancieren Strategien wie ϵ -Greedy und Softmax das Verhältnis von Exploration und Exploitation eher suboptimal (Thrun, 1992), sind jedoch in großen Zustandsräumen praktikabler, da sie keine zusätzlichen Explorationsdaten benötigen. Für die Softmax-Strategie existieren neurobiologische Hinweise, dass im menschlichen Gehirn das Wählen von Explorations- und Exploitationsaktionen in ähnlicher Weise stattfindet (Daw u. a., 2006). Derartig globale Ansätze haben jedoch das Problem, dass die Steuerung von Exploration und Exploitation auf einem globalen Explorationsparameter basiert, welcher vom Experimentator von Hand festgelegt werden muss. Mit Hinblick auf das Lernen bei Menschen und Tieren stellen wir jedoch fest, dass die Steuerung autonom, vom Inneren eines Individuums heraus erfolgt, insbesondere auf Basis von Unsicherheit und erwartetem Reward (Aston-Jones und Cohen, 2005). Eine derartige Steuerung in berechenbare Modelle umzusetzen, nimmt sich diese Arbeit als Zielsetzung.

1.1 Zielsetzung dieser Arbeit

Diese Arbeit verfolgt das Ziel, robuste Explorationsstrategien für modellfreies Reinforcement Learning in diskreten Aktionsräumen zu entwickeln. Inspiriert vom Vorbild der Natur sollen diese auf den gesammelten Erfahrungen der sensomotorischen Interaktion basieren. Hierfür wird der *Temporal-Difference Fehler* (TD-Fehler) als ein wesentliches Grundelement in Betracht gezogen, welcher in der Neurobiologie mit Dopamin assoziiert wird (Schultz u. a., 1993; Schultz, 1998; Schultz u. a., 1997). Aus Sicht des Reinforcement Learning gibt dieser den Unterschied zwischen der *Prädiktion* und der *Observation* von Reward an (Sutton und Barto, 1998). Stimmen beide überein, d. h. der TD-Fehler geht gegen Null, so kann das erlernte Wissen über die langfristige Auswirkung von Aktionen als *sicher* interpretiert werden. In diesem Fall sollen bevorzugt Exploitationsaktionen getätigt werden, die das erlernte Wissen über zukünftige Rewards ausnutzen. Ein hoher Betragswert des TD-Fehlers gibt hingegen an, dass die Observation bislang unzureichend vom Agenten prädiziert wird. Das fehlende Wis-

sen über die Umgebung sollte in diesem Fall durch eine bevorzugte Wahl von Explorationsaktionen hinzugelernt werden. Unter diesem Aspekt ergeben sich folgende interessante Forschungsfragen für meine Dissertation:

Nichtstationaritäten: Wie kann man mit nichtstationären Prozessen umgehen, bei denen sich aufgrund von Umgebungsänderungen Rewards unvorhergesehen ändern?

Stochastische Rewards: Wie kann ein auf dem TD-Fehler basierendes Verfahren mit stochastischen Rewards umgehen, bei welchen die Prädiktion so gut wie nie mit der Observation übereinstimmt?

Robustheit: Wie robust verhält sich das Verfahren bei Variationen des Explorationsparameters?

Partielle Beobachtbarkeit: Ist es möglich mit partiell beobachtbaren Umgebungen umzugehen, in welchen der Agent die exakte Zustandskoordinaten nicht kennt?

Neurobiologische Plausibilität: Liefern die Beiträge dieser Arbeit ebenso Rückschlüsse auf das Analogon in der Neurobiologie?

1.2 Wissenschaftlicher Beitrag

Der wissenschaftliche Beitrag dieser Arbeit sind neue Explorationsstrategien, welche die Explorationsrate auf Basis des Lernfortschritts adaptieren, anstatt von einem Experimentator manuell festgelegt zu werden (Tokic, 2010; Tokic und Palm, 2011, 2012a,b; Tokic u. a., 2012). Ein derartiges Lernverhalten spiegelt das Lernen in natürlichen Organismen wider, in welchen die Steuerung von Exploration und Exploitation in ähnlicher Art und Weise stattfindet. In der Literatur wird dieser Prozess auch als *Meta-Lernen* bezeichnet, bei welchem gelernt wird, wie gelernt werden soll (Doya, 2002; Ishii u. a., 2002; Kobayashi u. a., 2009; Schweighofer und Doya, 2003).

Die vorgestellte VDBE-Strategie erweitert die ε -Greedy-Strategie dahingehend, dass nicht eine globale Explorationsrate ε von Hand belegt werden muss, sondern ein lokaler, zustandsabhängiger Meta-Parameter $\varepsilon(s)$ gelernt wird (Tokic, 2010). Der Lernprozess wird durch den TD-Fehler angetrieben, welcher den Unterschied zwischen dem prädizierten und tatsächlich observierten Reward angibt (Sutton und Barto, 1998). In diese Arbeit dient der TD-Fehler als Informationsquelle zur Erkennung von Umge-

bungsänderungen sowie zur Messung der Sicherheit über die langfristige Auswirkung von Aktionen.

In einem aufbauenden Ansatz wurde das Steuern eines Explorationsparameters anhand des Modells eines stochastischen Neuron untersucht (Tokic und Palm, 2012a,b), wofür das von Williams (1992) vorgestellte REINFORCE-Verfahren zum Einsatz kommt. Dieses ist ein Gradient-Following-Verfahren, welches den Performanz-Gradienten nicht direkt berechnet, sondern anhand der ersten partiellen Ableitung der Aktivierungsfunktion schätzt. Das Verfahren, welches ursprünglich für Reinforcement Learning mit kontinuierlichen Aktionen angedacht war, wird zur Steuerung eines kontinuierlichen Explorationsparameters in diskreten Aktionsräumen eingesetzt und mit verschiedenen Explorationsstrategien kombiniert. Das Ziel war ursprünglich, den VDBE-Parameter zur Steuerung der Empfindlichkeit auf den TD-Fehler zu adaptieren. In den durchgeführten Experimenten stellte sich jedoch heraus, dass dieselbe Vorgehensweise direkt auch auf andere Explorationsstrategien übertragen werden kann, weswegen ebenso die direkte Steuerung von ε -Greedy (ohne VDBE) und Softmax in Betracht gezogen wurde. Zusätzlich wird untersucht, wie sich das Lernverhalten bei einer lokalen, zustandsbasierten Steuerung, gegenüber dem einer globalen, episodischen Steuerung unterscheidet. Letztere hat den Vorteil, dass Explorationsdaten (drei Skalare pro Zustand) nur in Startzuständen benötigt werden, was insbesondere zu einer hohen Speichereffizienz für Lernprobleme mit wenigen oder nur einem Startzustand führt.

Im Kern dieser Arbeit werden die soeben angedeuteten Explorationsstrategien untersucht, gegenübergestellt und mit ihren individuellen Vor- und Nachteilen verglichen. In mehreren Experimenten aus verschiedenen Lernproblemklassen wird empirisch untersucht, wie sich der Reward bei Variationen des Explorationsparameters verhält. Hierbei werden die modellfreien Temporal-Difference-Lernverfahren Q -learning (Watkins, 1989) und Sarsa (Rummery und Niranjan, 1994) verwendet, welche auch als plausible berechenbare Modelle für das Lernen in der Hirntheorie gelten (Dayan, 2009; Niv, 2009; Niv u. a., 2006; Watkins, 1989).

1.3 Aufbau dieser Arbeit

Der Aufbau dieser Arbeit ist wie folgt: In Kapitel 2 werden berechenbare Modelle für Reinforcement Learning vorgestellt, mit welchen es möglich ist optimales Verhalten zu erlernen. Das darauf folgende Kapitel 3 be-

schäftigt sich mit der Explorationsproblematik und stellt verwandte Arbeiten aus der Literatur vor. Kapitel 4 stellt die in dieser Arbeit entwickelten VDBE-Explorationsstrategien vor, wodurch die Balance zwischen Exploration und Exploitation, auf Basis des Lernfortschritts, adaptiert werden kann. Das anschließende Kapitel 5 gibt eine Übersicht über mögliche Lernproblemdimensionen, an welchen die vorgestellten Explorationsstrategien evaluiert werden. Ferner werden in Kapitel 6 Explorationsstrategien untersucht, die zur Steuerung stochastische Neuronen verwenden, für die globale oder lokale Steuerung eines Meta-Parameters. In Kapitel 7 werden Parallelen zwischen der technischen Sichtweise von Reinforcement Learning gegenüber der Neurobiologie und Verhaltenspsychologie aufgezeigt und ebenso die technischen Beiträge dieser Arbeit spekulativ in die Neurobiologie eingeordnet. Abschließend werden die erhaltenen Erkenntnisse in Kapitel 8 reflektiert sowie Ausblicke über mögliche Folgearbeiten gegeben.

Kapitel 2

Berechenbare Modelle für Reinforcement Learning

In diesem Kapitel werden berechenbare Modelle für *Reinforcement Learning* vorgestellt, mit welchen intelligentes Verhalten, auf Basis von Interaktion und Rewards, erlernt werden kann. Die Darstellung und verwendete Notation richten sich nach dem Standard-Werk „*Reinforcement Learning: An Introduction*“ von Sutton und Barto (1998).

2.1 Die Agent-Umgebung-Interaktion

Eine schematische Darstellung des Reinforcement Learning ist in Abbildung 2.1 aufgezeigt, in welcher ein Agent mit seiner Umgebung, in einer Sequenz von diskreten Zeitschritten, $t = 0, 1, 2, 3, \dots, T$, interagiert (Sutton und Barto, 1998). Zu jedem Zeitschritt t wird der Zustand der Umgebung $s_t \in \mathcal{S}$ aus einer endlichen Menge von möglichen Zuständen \mathcal{S} wahrgenommen. Auf dieser Grundlage wählt der Agent eine Aktion a_t aus, die aus der endlichen Menge $\mathcal{A}(s_t)$ von möglichen Aktionen in Zustand s_t entstammt, $a_t \in \mathcal{A}(s_t)$. Einen Zeitschritt später erhält der Agent einen Reward $r_{t+1} \in \mathbb{R}$ von seiner Umgebung und befindet sich in einem Folgezustand s_{t+1} . Der Reward gibt darüber Auskunft *wie zielführend* die zuletzt getätigte Aktion war, was beispielsweise ein negativer Wert für schlechte Aktionen ist, hingegen ein positiver Wert für zielführende Aktionen. Durch einen anschließenden Lernschritt können mit Hilfe des Rewards gute Aktionen bestärkt, hingegen schlechte Aktionen vermieden werden.

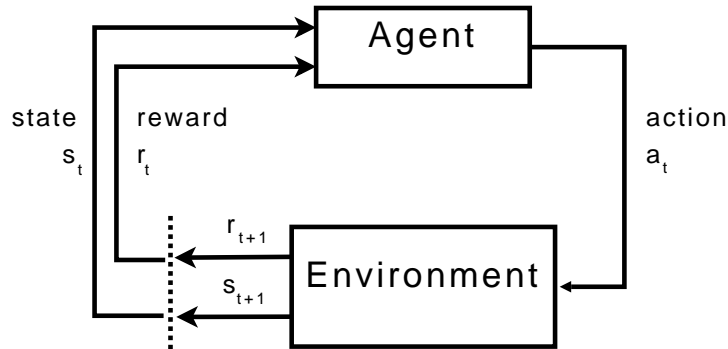


Abbildung 2.1: Interaktions-Schema des Agenten mit seiner Umgebung (nach Sutton und Barto (1998)).

Zu jedem Zeitschritt besitzt der Agent eine Abbildung von Zuständen S auf Aktionen, was als *Strategie* π bezeichnet wird:

$$\pi : S \rightarrow \mathcal{A} . \quad (2.1)$$

Die Werte $\pi(s, a)$ sind probabilistische Auswahlwahrscheinlichkeiten für das Wählen von Aktion a in Zustand s :

$$\pi_t(s, a) = Pr(a_t = a | s_t = s) . \quad (2.2)$$

Das Ziel ist diese Werte zu optimieren. Hierfür werden Aktionen mit einer hohen Auswahlwahrscheinlichkeit bestärkt, wenn diese zu gutem Reward führen; hingegen schlechte Aktionen mittels einer niedrigeren Auswahlwahrscheinlichkeit in Zukunft vermieden. Für RL mit diskreten Aktionen ist üblicherweise die Summe aller Auswahlwahrscheinlichkeiten in einem beliebigen Zustand $s \in S$:

$$\sum_{a \in \mathcal{A}(s)} \pi(s, a) = 1 . \quad (2.3)$$

2.2 Lernprobleme

Grundlegend werden Lernprobleme in episodische und kontinuierliche Lernprobleme klassifiziert. Dies ist notwendig, da beachtet werden muss, wie der kumulierte Reward R_t definiert wird, welcher auch als *Return* bezeichnet wird.

2.2.1 Episodische Lernprobleme

Als *episodisch* werden Lernprobleme bezeichnet, die einen zeitlich begrenzten Horizont $\{0, 1, \dots, T\}$ besitzen, wodurch die Höhe des kumulierten Rewards

$$R_t = r_{t+1} + r_{t+2} + \dots + r_T \quad (2.4)$$

begrenzt wird. Beispielsweise ist dies bei Zielzuständen gegeben, in denen die aktuelle Lernepisode terminiert (z. B. das Ende eines Brettspieles). Es könnte aber auch eine maximale Anzahl von aufeinanderfolgenden Aktionen sein, welche die maximale Länge einer Lernepisode künstlich begrenzt. Wichtig ist jedenfalls die Annahme, dass ein begrenzter Horizont existiert. Die Zustandsmenge inklusive Zielzuständen wird in diesem Fall als \mathcal{S}^+ bezeichnet, hingegen die Menge ohne Zielzustände als \mathcal{S} .

2.2.2 Kontinuierliche Lernprobleme

Ein *kontinuierliches* Lernproblem besitzt keinen zeitlich begrenzten Horizont. Stattdessen führt der Agent kontinuierlich Aktionen in seiner Umgebung aus, wie beispielsweise beim Regeln einer Temperatur. In Gleichung (2.4) besteht jedoch das Problem, dass der Return für $t \rightarrow \infty$ den Wert $R_t = \infty$ annehmen kann. Zur Lösung dieses Problems verwendet man das Abschwächungsprinzip (Discounting), wodurch weiter in der Zukunft erhaltene Rewards in R_t je niedriger gewichtet werden, desto später sie vom Agenten empfangen werden (Sutton und Barto, 1998). Hierdurch wird die eigentlich unendliche Summe durch einen endlichen Wert begrenzt:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \quad (2.5)$$

Der Parameter γ gibt einen Diskontierungsfaktor im Intervall $[0, 1)$ an. Je kleiner γ gewählt wird, desto niedriger werden zukünftige Rewards in R_t gewichtet, wodurch unmittelbare Rewards mehr Bedeutung erhalten (vgl. Abbildung 2.2). Analog werden für hohe Belegungen in der Zukunft erhaltene Rewards stärker gewichtet. Folglich stellt dieser Parameter sozusagen die Entfernung eines *Weitblicks* in die Zukunft dar, da ein Reward k Zeitschritte in der Zukunft mit γ^{k-1} in R_t gewichtet ist.

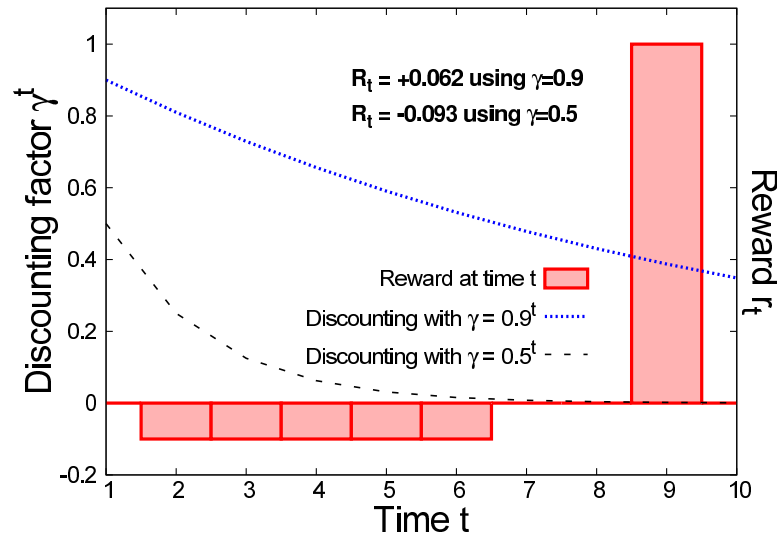


Abbildung 2.2: Reward-Diskontierung nach Gleichung (2.5): Ein Diskontierungsfaktor in Höhe von $\gamma = 0.9$ führt dazu, dass der hohe Reward zum Zeitpunkt $t = 9$, im Return R_t stärker berücksichtigt wird, woraus resultiert, dass dieser mit $R_t = 0.062$ positiv ist, obwohl zunächst negative Rewards erhalten werden. Ein niedrigerer Diskontierungsfaktor in Höhe von $\gamma = 0.5$ berücksichtigt hingegen den hohen Reward fast gar nicht, weswegen der Return mit $R_t = -0.093$ ein negativer Wert ist (Abb. in Anlehnung an (Doya, 2002)).

2.2.3 Gemeinsame Notation

Um im Folgenden eine gemeinsame Notation für sowohl episodische als auch kontinuierliche Lernprobleme zu erhalten, können die Gleichungen (2.4)+(2.5) miteinander kombiniert werden (Sutton und Barto, 1998):

$$R_t = \sum_{k=0}^T \gamma^k r_{t+k+1} . \quad (2.6)$$

Es wird erlaubt, dass die Belegungen $T = \infty$ und $\gamma = 1$ möglich sind, jedoch nicht miteinander auftreten dürfen. Folglich erhält man eine gemeinsame Notation für beide Problemklassen, welche zusätzlich auch deren Parallelen ausdrückt.

2.3 Markovsche Entscheidungsprozesse

Als wichtiges Lernelement benötigt ein Agent eine Beschreibung der Zustände seiner Umgebung, um möglichst effizient erlerntes Wissen mit der

aktuellen Situation zu assoziieren, passende Aktionen zu wählen und neues Wissen wieder hinzuzufügen. Im Folgenden wird daher nun definiert, welche Anforderungen an eine Zustandsbeschreibung gestellt werden, damit sie effizient zum Lernen verwendet werden kann. Von besonderer Bedeutung ist hierbei die *Markov-Eigenschaft*, da bei deren Gegebenheit optimales Verhalten erlernt werden kann (Bellman, 1957).

Als wichtiger Bestandteil muss die Zustandsbeschreibung (sensorische) Informationen enthalten, welche dem Agenten nur über die augenblicklichen Ereignisse Auskunft geben; also beispielsweise nur aus den offenen Karten beim Black-Jack-Kartenspiel bestehen oder nur aus den aktuellen Positionen der Spielsteine eines Brettspiels. Die Zustandsbeschreibung sollte jedoch nicht enthalten, wie der Agent in diesen Zustand gekommen ist, wodurch einerseits die Beschreibung möglichst kompakt gehalten wird, ebenso aber auch das erlernte Wissen über die Umgebung. Im Allgemeinen wird eine derartige Beschreibung auch als *Markov-Eigenschaft* bzw. *Gedächtnislosigkeit* bezeichnet. Formal darf die Wahrscheinlichkeit für den Folgezustand s' und den Reward r zum Zeitpunkt $t + 1$, nur von Zustand s und der gewählten Aktion a zum Zeitpunkt t abhängig sein:

$$Pr \{s_{t+1} = s', r_{t+1} = r \mid s_t, a_t\} , \quad (2.7)$$

für alle s', r, s_t und a_t .

Ein Lernproblem, dass die Markov-Eigenschaft erfüllt, wird *Markov Decision Process* (MDP, dt. Markovscher Entscheidungsprozess) genannt. Sollten (idealerweise) der Zustands- und Aktionsraum endlich sein, so ist das Lernproblem ein *endlicher MDP*. Dieser ist durch seinen Zustands- und Aktionsraum sowie der Ein-Schritt-Dynamik definiert. Für beliebige Zustands-Aktions-Paare gilt, dass die Wahrscheinlichkeit für Folgezustand s' nur von Zustand s und der gewählten Aktion a abhängig ist:

$$\mathcal{P}_{ss'}^a = Pr \{s_{t+1} = s' \mid s_t = s, a_t = a\} . \quad (2.8)$$

Dabei wird \mathcal{P} als die Menge der *Transitionswahrscheinlichkeiten* bezeichnet. Der Erwartungswert des Rewards, E , ist hingegen abhängig von Zustand s , Aktion a sowie des Folgezustands s' :

$$\mathcal{R}_{ss'}^a = E \{r_{t+1} \mid s_t = s, a_t = a, s_{t+1} = s'\} . \quad (2.9)$$

Die Mengen \mathcal{P} und \mathcal{R} beschreiben die wichtigsten Dynamikaspekte eines MDPs, wobei u. a. folgende Spezialfälle existieren:

Deterministische Umgebungen:

Ein wichtiger Spezialfall sind *deterministische Umgebungen* (Thrun, 1992). Für diese gilt, dass die Zustandsübergangs- und Rewardfunktion völlig vorhersagbar sind. Hierbei muss gelten, dass für jeden Zustand genau ein Folgezustand s' existiert, für den $\mathcal{P}_{ss'}^a = 1$ gilt, sowie für alle andere Zustände $y \neq s'$: $\mathcal{P}_{sy}^a = 0$. Demnach sind Brettspiele wie Schach, Dame oder Othello deterministische MDPs, da jede Aktion in einen *deterministischen* Folgezustand führt.

Ergodische MDPs:

Eine weitere Eigenschaft von MDPs ist die *Ergodizität* (Thrun, 1992). Diese wird einem MDP zugesprochen, falls für alle Zustände $s \in \mathcal{S}$ eine Wahrscheinlichkeit größer Null existiert, einen beliebigen Folgezustand $s' \in \mathcal{S}$, ggf. über mehrere Zwischenzustände, zu erreichen. Demnach ist das Brettspiel Othello nicht ergodisch, da eroberte Ecksteine vom Gegner nicht mehr zurückerobert werden können. Hingegen besitzt das Kartenspiel „Uno“ die ergodische Eigenschaft, da es möglich ist, wieder in die Spielanfangssituation zu gelangen (wenn auch über sehr viele Schritte).

Partiell beobachtbare MDPs:

Partiell beobachtbare MDPs haben als Besonderheit, dass Zustandsinformationen nur zum Teil beobachtbar sind, so wie es bei vielen realistischen Entscheidungsproblemen der Fall ist (Russell und Norvig, 2009, S. 658ff.). Dem Agenten fehlen Informationen in der Wahrnehmung des Zustandssignals, wodurch es nicht möglich ist die exakte Position im Zustandsraum zu bestimmen, und folglich eine optimale Aktionswahl erschwert. In der Praxis trifft dies oftmals auf autonome Roboter zu, die sich nur relativ anhand von (evtl. verrauschten) Sensoren lokalisieren können (z. B. Laserscanner oder Kameras), jedoch nicht die exakte Koordinate in der Welt kennen.

2.4 Wertefunktionen

Viele Algorithmen zum Lösen von Markovschen Entscheidungsprozessen basieren auf dem Schätzen einer Wertefunktion für Zustände oder Zustands-Aktions-Paare (Sutton und Barto, 1998). Die Wertefunktion sagt dabei aus, *wie wertvoll* es für den Agenten ist in einem bestimmten Zustand zu sein oder eine Aktion in einem bestimmten Zustand auszuwäh-

len. Mit der Bezeichnung *wertvoll* ist gemeint, wie viel zukünftigen Reward der Agent erhält, falls er beginnend in Zustand s Aktion a wählt. Da der kumulierte Reward R_t jedoch ebenso von den gewählten Aktionen in Folgezuständen abhängig ist, werden auf Grund dessen Wertefunktionen in Abhängigkeit von Aktionsauswahlstrategien π definiert.

Wie bereits erwähnt definiert Strategie π eine Abbildung von Zuständen auf Aktionen $\pi : \mathcal{S} \rightarrow \mathcal{A}$. Im Folgenden wird deshalb der *Wert* von Zustand s , bei Befolgung von Strategie π , als $V^\pi(s)$ bezeichnet. Dieser gibt den kumulierten und diskontierten Reward an, den der Agent für das Folgen von π , beginnend in s , erwartet zu erhalten. Formal wird solch eine *Zustandswertfunktion* $V^\pi(s)$ definiert als:

$$\begin{aligned} V^\pi(s) &= E_\pi \{ R_t \mid s_t = s \} \\ &= E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{k+t+1} \mid s_t = s \right\}. \end{aligned} \quad (2.10)$$

E_π gibt an, dass R_t ein Erwartungswert ist, unter der Bedingung, dass der Agent, beginnend in Zustand s , Strategie π folgt und t ein beliebiger Zeitschritt ist. Es gilt anzumerken, dass für episodische Lernprobleme der Wert eines Zielzustandes (sollte einer existieren) den Wert 0 besitzt.

Analog kann auch eine *Wertfunktion für Zustands-Aktions-Paare* Q^π angegeben werden. Diese beschreibt den erwarteten, kumulierten und diskontierten Reward für das Folgen von Strategie π , beginnend in Zustand s mit Aktion a :

$$\begin{aligned} Q^\pi(s, a) &= E_\pi \{ R_t \mid s_t = s, a_t = a \} \\ &= E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{k+t+1} \mid s_t = s, a_t = a \right\}. \end{aligned} \quad (2.11)$$

Die Wertefunktionen V^π und Q^π werden im Reinforcement Learning durch Interaktion mit der Umgebung geschätzt. Wenn beispielsweise Strategie π gefolgt wird und dabei Mittelwerte über alle zukünftigen Rewards gespeichert werden (für jeden besuchten Zustand), so konvergieren diese Werte zu $V^\pi(s)$ unter der Annahme, dass Zustand s unendlich oft durchlaufen wird. Ähnlich konvergiert die Zustands-Aktions-Wertfunktion $Q^\pi(s, a)$ unter der Annahme, dass für jede Aktion in s Mittelwerte gespeichert werden (Sutton und Barto, 1998).

Sollte ein Lernproblem aus sehr vielen Zuständen bestehen oder sogar kontinuierliche Zustandsdimensionen besitzen, so ist das Approximieren der Wertefunktion in tabellarischer Form nicht mehr praktikabel. Stattdessen wird oftmals die Wertefunktion als eine parametrisierte Funktion gelernt, deren Parameter, auf Basis der observierten Rewards, durch ein Gradientenverfahren adaptiert werden (Boyan und Moore, 1995; Sutton, 1996).

Eine grundlegende Eigenschaft von Wertefunktionen ist die rekursive Verbindung von Zuständen zueinander. Für beliebige Strategien und Zustände hält folgende Konsistenzbedingung zwischen dem Wert $V^\pi(s)$ eines Zustandes und dem Wert $V^\pi(s')$ eines möglichen Folgezustandes:

$$\begin{aligned}
V^\pi(s) &= E_\pi \{ R_t \mid s_t = s \} \\
&= E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{k+t+1} \mid s_t = s \right\} \\
&= E_\pi \left\{ r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{k+t+2} \mid s_t = s \right\} \\
&= \sum_a \pi(a|s) \sum_{s'} \mathcal{P}_{ss'}^a \left[\mathcal{R}_{ss'}^a + \gamma E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{k+t+2} \mid s_{t+1} = s' \right\} \right] \\
&= \sum_a \pi(a|s) \sum_{s'} \mathcal{P}_{ss'}^a [\mathcal{R}_{ss'}^a + \gamma V^\pi(s')] , \tag{2.12}
\end{aligned}$$

wobei $a \in \mathcal{A}(s)$ eine Aktion aus der Menge von möglichen Aktionen in Zustand s entstammt, hingegen Folgezustand s' aus der Menge von möglichen Zuständen \mathcal{S} ; beziehungsweise \mathcal{S}^+ bei episodischen Lernproblemen (Sutton und Barto, 1998).

Gleichung (2.12) wird als *Bellman-Gleichung* der Wertefunktion V^π bezeichnet, welche die Verbindung zwischen dem Wert eines Zustandes sowie dem Wert eines möglichen Folgezustandes ausdrückt. Dabei wird über alle möglichen Aktionen $a \in \mathcal{A}(s)$ in Zustand s gemittelt, unter der Berücksichtigung der jeweiligen Auswahlwahrscheinlichkeiten $\mathcal{P}_{ss'}^a$. Die Bellman-Gleichung hat sich in der Praxis als Basis vieler Lernverfahren etabliert, wodurch RL-Probleme auf effektivere Art und Weise gelöst werden können, als durch die oben beschriebene Mittelung von erhaltenen Rewards (Sutton und Barto, 1998).

2.5 Optimale Wertefunktionen

Das Ziel im Reinforcement Learning ist es, eine Strategie π zu finden, die möglichst viel Reward einbringt, was dadurch erreicht werden kann, indem verschiedene Strategien miteinander verglichen werden. Dabei ist es möglich für endliche MDPs den Begriff einer *optimalen Strategie* zu definieren (Sutton und Barto, 1998). In diesem Zusammenhang wird eine Strategie π' genau dann als besser oder gleich gut wie Strategie π bezeichnet, wenn der erwarteter Reward von π' größer oder gleich groß ist, wie der von Strategie π , d. h. $\forall s \in \mathcal{S} : V^{\pi'}(s) \geq V^{\pi}(s)$. Da es Strategien gibt, die besser oder gleich gut sind wie alle anderen Strategien, wird diese besondere Art von Strategie als *optimale Strategie* π^* bezeichnet. Ihr unterliegt ebenso die *optimale Zustandswertfunktion* V^* :

$$V^*(s) = \max_{\pi} V^{\pi}(s) , \quad (2.13)$$

für alle $s \in \mathcal{S}$. Analog wird die *optimale Wertefunktion für Zustands-Aktions-Paare* $Q^*(s, a)$ definiert als:

$$Q^*(s, a) = \max_{\pi} Q^{\pi}(s, a) , \quad (2.14)$$

für alle $s \in \mathcal{S}$ sowie alle $a \in \mathcal{A}(s)$. Für beliebige Zustands-Aktions-Paare (s, a) liefert diese Funktion den erwarteten, kumulierten und diskontierten Reward für das Wählen von Aktion a in Zustand s , wodurch einer optimalen Strategie gefolgt werden kann. Ebenso ist es möglich Q^* in Analogie zu V^* zu definieren:

$$Q^*(s, a) = E \{ r_{t+1} + \gamma V^*(s_{t+1}) \mid s_t = s, a_t = a \} . \quad (2.15)$$

Da V^* die Wertefunktion einer Strategie ist, muss auch diese die Konsistenzbedingung der Bellman-Gleichung (2.12) für alle Zustände erfüllen. Da es insbesondere die optimale Wertefunktion ist, kann die Konsistenzbedingung von V^* in eine besondere Form, ohne Bezug auf eine bestimmte Strategie, überführt werden, die auch als *Bellmansche Optimalitätsgleichung* bekannt ist. Diese drückt aus, dass der Wert eines Zustandes beim Folgen

einer optimalen Strategie, gleich dem Wert der besten Aktion in Zustand s entspricht (Sutton und Barto, 1998):

$$\begin{aligned}
 V^*(s) &= \max_{a \in \mathcal{A}(s)} Q^{\pi^*}(s, a) \\
 &= \max_a E_{\pi^*} \{R_t \mid s_t = s, a_t = a\} \\
 &= \max_a E_{\pi^*} \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s, a_t = a \right\} \\
 &= \max_a E_{\pi^*} \left\{ r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s, a_t = a \right\} \\
 &= \max_a E \{r_{t+1} + \gamma V^*(s_{t+1}) \mid s_t = s, a_t = a\} \tag{2.16} \\
 &= \max_{a \in \mathcal{A}(s)} \sum_{s'} \mathcal{P}_{ss'}^a [\mathcal{R}_{ss'}^a + \gamma V^*(s')] . \tag{2.17}
 \end{aligned}$$

Die letzten beiden Gleichungen stellen dabei zwei Formen der Bellmanschen Optimalitätsgleichung für Zustandswertefunktionen dar. Analog sind diese für Zustands-Aktions-Wertefunktionen:

$$\begin{aligned}
 Q^*(s, a) &= E \left\{ r_{t+1} + \gamma \max_{a' \in \mathcal{A}(s')} Q^*(s_{t+1}, a') \mid s_t = s, a_t = a \right\} \\
 &= \sum_{s'} \mathcal{P}_{ss'}^a \left[\mathcal{R}_{ss'}^a + \gamma \max_{a' \in \mathcal{A}(s')} Q^*(s', a') \right] . \tag{2.18}
 \end{aligned}$$

Sobald also die optimale Wertefunktion bekannt ist, ist es möglich einer optimalen Strategie zu folgen, indem in jedem Zustand die Aktion mit dem höchsten Q -Wert gewählt wird. In diesem Zusammenhang wurde von Littman u. a. (1995) gezeigt, dass die Lernkomplexität von V^* proportional zu $1/(1 - \gamma)$ ist.

2.6 Modellbasiertes Lernen

Lernverfahren aus dem Bereich *Dynamische Programmierung* (DP) können verwendet werden, wenn ein vollständiges Modell der Umgebung zur Verfügung steht (Bellman, 1957; Bertsekas, 1987; Sutton und Barto, 1998). Auf Basis der beiden Mengen \mathcal{R} und \mathcal{P} , die einen MDP vollständig beschreiben, werden durch iteratives Annähern Zwischenlösungen V berechnet, auf deren Basis neue Zwischenlösungen V' berechnet werden. In einem iterativen Prozess führt dies zur optimalen Wertefunktion V^* , anhand welcher die optimale Strategie π^* abgeleitet werden kann. Derartige Lernverfahren,

welche Schätzungen auf Basis von vorherigen Schätzungen verbessern, bezeichnet man daher auch als *bootstrapping*-Verfahren.

2.6.1 Strategie-Evaluierung

Als Basis vieler DP-Verfahren wird über eine *Strategie-Evaluierung* die Wertefunktion V^π der zugrunde liegenden Strategie π berechnet, um auf dieser Grundlage die Strategie anschließend zu verbessern (Sutton und Barto, 1998). Die Berechnung der Wertefunktion erfolgt als Fixpunktiteration nach Gleichung (2.12) für jeden Zustand s :

$$V_{k+1}(s) = \sum_a \pi(s, a) \sum_{s'} \mathcal{P}_{ss'}^a [\mathcal{R}_{ss'}^a + \gamma V_k(s')] \quad , \quad (2.19)$$

für alle $s \in \mathcal{S}$, wobei $V_0(s)$ mit einem beliebigen Wert initialisiert wird¹. Für $k \rightarrow \infty$ konvergiert $V_k \rightarrow V^\pi$, unter der Bedingung, dass V^π existiert. Sollte V^π existieren, so ist es in der Praxis ausreichend den Lernprozess abubrechen, sobald der größte Fehler eines Zustands, gemessen beim Aktualisieren der Wertefunktion, unterhalb einer kleinen positiv gewählten Schwelle θ liegt:

$$\max_{s \in \mathcal{S}} |V_{k+1}(s) - V_k(s)| < \theta \quad , \quad \text{mit } \theta \gtrsim 0 \quad . \quad (2.20)$$

2.6.2 Strategie-Verbesserung

Besitzt man die Wertefunktion V^π einer zugrunde liegenden Strategie π , so ist es möglich einen Strategie-Verbesserungsschritt durchzuführen (Sutton und Barto, 1998). Als Ergebnis erhält man eine möglicherweise bessere Strategie π' , wobei gilt, dass $\pi' \geq \pi$. Um π' zu erhalten wird von nun an nicht mehr der bisherigen Strategie π gefolgt, sondern gierig Aktion a gewählt, welche den erwarteten Reward maximiert:

$$\pi'(s) = \arg \max_a Q^\pi(s, a) \quad (2.21)$$

$$= \arg \max_a E_\pi \{ r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s, a_t = a \} \quad (2.22)$$

$$= \arg \max_a \sum_{s'} \mathcal{P}_{ss'}^a [\mathcal{R}_{ss'}^a + \gamma V^\pi(s')] \quad . \quad (2.23)$$

¹Üblicherweise wird $V_{k=0} = 0$ verwendet, da dies der Information entspricht, dass noch kein Wissen vorliegt.

Hierbei gilt:

$$\sum_s \sum_a \pi'(s, a) Q^\pi(s, a) \geq \sum_s \sum_a \pi(s, a) Q^\pi(s, a) \quad (2.24)$$

Folglich ist es sinnvoller der verbesserten Strategie π' zu folgen, da der Agent hierdurch mehr Reward erhält.

2.6.3 Strategie-Iteration

Sobald eine Strategie π anhand von V^π verbessert wurde zu π' ist es möglich die dazugehörige Wertefunktion $V^{\pi'}$ zu berechnen, um daraus eine noch bessere Strategie π'' ableiten zu können (Sutton und Barto, 1998). In einem alternierenden Prozess bezeichnet man dies als *Strategie-Iteration*, welche aus einer Sequenz von Strategie-Evaluierungs- und Verbesserungsschritten (\xrightarrow{E} bzw. \xrightarrow{I}) besteht:

$$\pi_0 \xrightarrow{E} V^{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} V^{\pi_1} \xrightarrow{I} \pi_2 \xrightarrow{E} \dots \xrightarrow{I} \pi^* \xrightarrow{E} V^{\pi^*}, \quad (2.25)$$

und schlussendlich zur optimalen Strategie π^* führt. Jede durch diesen Prozess neu erzeugte Strategie π' ist von der Performanz besser oder gleich gut, wie alle bisherigen Strategien. Da ein endlicher MDP aus einer endlichen Anzahl von Strategien besteht, konvergiert diese Form von Strategie-Iteration in endlicher Zeit zur optimalen Strategie.

2.6.4 Wert-Iteration

Ein Nachteil von Strategie-Iteration ist, dass für jede Iteration eine komplette Strategie-Evaluierung durchgeführt werden muss, die ihrerseits ein mehrmaliges Iterieren über alle Zustände des Zustandsraumes erfordert, um aus π die dazugehörige Wertefunktion zu berechnen. Da in der Praxis oftmals die größte Verbesserung nach einem Strategie-Evaluationsschritt über alle Zustände auftritt, kann man basierend auf diesem Teilergebnis eine Strategie-Verbesserung direkt anschließen, ohne dabei die Konvergenzeigenschaft zu verlieren. Dieser Spezialfall wird als *Wert-Iteration* bezeichnet (vgl. Algorithmus 1), bei welcher iterativ über alle Zustände ein Strategie-Evaluationsschritt erfolgt, mit anschließender Strategie-Verbesserung (Bertsekas, 1987; Puterman und Shin, 1978; Tokic u. a., 2009). In der Praxis wird das Verfahren abgebrochen sobald die größte Wertänderung, gemessen über alle Zustandsaktualisierungen einer Iteration, unterhalb einer kleinen, positiv gewählten Schwelle θ liegt.

Algorithm 1 WERT ITERATION NACH (SUTTON UND BARTO, 1998)

- 1: Initialize V arbitrarily, e.g. $V(s) = 0$ for all $s \in \mathcal{S}^+$
 - 2: **repeat**
 - 3: $\Delta \leftarrow 0$
 - 4: **for each** $s \in \mathcal{S}$ **do**
 - 5: $v \leftarrow V(s)$
 - 6: $V(s) \leftarrow \max_a \sum_{s'} \mathcal{P}_{ss'}^a [\mathcal{R}_{ss'}^a + \gamma V(s')]$
 - 7: $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
 - 8: **end for**
 - 9: **until** $\Delta < \theta$ (a small positiv number)
 - 10: Output a deterministic policy, π , such that

$$\pi(s, a) = \begin{cases} \frac{1}{|A^*(s)|} & \text{if } \arg \max_a \sum_{s'} \mathcal{P}_{ss'}^a [\mathcal{R}_{ss'}^a + \gamma V(s')] , \\ 0 & \text{otherwise.} \end{cases}$$
-

2.7 Modellfreies Lernen

Algorithmen aus dem Bereich *Dynamische Programmierung* setzen ein vollständiges Modell der Umgebung voraus, in Form der beiden Mengen \mathcal{R} und \mathcal{P} . Da diese jedoch bei Realwelt-Problemen schwer im Voraus zu erhalten sind, war einer der wichtigsten Durchbrüche im Reinforcement Learning die Entwicklung von Temporal-Difference Learning (TD-Learning) (Sutton, 1988; Watkins, 1989), wodurch optimales Verhalten ohne vollständigem Modell erlernt werden kann. TD-Lernverfahren wie Q -learning (Watkins, 1989) und Sarsa (Rummery und Niranjan, 1994) approximieren die Q -Funktion direkt auf Basis der unmittelbaren Observation des Agenten (Reward + Folgezustand), weshalb diese Form von Lernen besser das biologische Vorbild nachbildet (Dayan, 2009; Doya, 2008).

Aufgrund von fehlenden Informationen über die Umgebung benötigen derartige Lernverfahren eine stochastische Komponente in der Aktionsauswahl, die sicherstellt, dass das Wissen über die Umgebung möglichst präzise gelernt wird. Der Reward des Agenten hängt hierbei signifikant davon ab:

1. wie viele Explorationsaktionen getätigt werden, welche die Umgebung erkunden,
2. wie viele Exploitationsaktionen getätigt werden, welche das bereits erlernte Wissen über die Umgebung ausnutzen.

Aufgrund der Fülle von Verfahren zur Steuerung zwischen Exploration und Exploitation wird dieses Thema im anschließenden Kapitel 3 genauer beleuchtet. Im Folgenden werden nun zwei gängige Verfahren für Temporal-Difference Learning vorgestellt, die in Kapitel 4-6 an mehreren Lernproblemen angewandt werden.

2.7.1 Q -learning

Bei Q -learning wird die Observation des Agenten, bestehend aus dem erhaltenen Reward r_{t+1} sowie dem Folgezustand s' , dazu verwendet, um eine Wertefunktion für Zustands-Aktions-Paare $Q(s, a)$ zu schätzen (Watkins, 1989). Dabei wird nach dem Ausführen einer Aktion a_t in Zustand s_t , die Schätzung $Q(s_t, a_t)$ wie folgt aktualisiert (vgl. Algorithmus 2):

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \underbrace{\alpha}_{\text{Lernrate}} \underbrace{\left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right]}_{\text{TD-Fehler } \Delta}, \quad (2.26)$$

wobei $0 < \alpha \leq 1$ eine positive Lernrate angibt. Durch diese wird bestimmt, wie stark der bisherige Q -Wert in Richtung des TD-Fehlers Δ angepasst wird (George und Powell, 2006; Powell, 2007).

Algorithm 2 Q -LEARNING nach (Sutton und Barto, 1998)

- 1: Initialize $Q(s, a)$ arbitrarily, e.g. $Q(s, a) = 0$ for all s, a
 - 2: **for** each episode **do**
 - 3: Initialize start state s
 - 4: **repeat**
 - 5: Select action a according to $\pi(s, a)$
 - 6: Take action a , observe reward r and successor state s'
 - 7: $b^* \leftarrow \operatorname{argmax}_{b \in \mathcal{A}(s')} Q(s', b)$
 - 8: $\Delta \leftarrow r + \gamma Q(s', b^*) - Q(s, a)$
 - 9: $Q(s, a) \leftarrow Q(s, a) + \alpha \Delta$
 - 10: $s \leftarrow s'$
 - 11: **until** s is terminal state
 - 12: **end for**
-

Der TD-Fehler Δ besteht aus dem unmittelbar erhaltenen Reward r_{t+1} , dem mit $\gamma \in [0, 1)$ abgeschwächten höchsten Q -Wert des Folgezustandes $\max_a Q(s_{t+1}, a_t)$ sowie der bisherigen Schätzung $Q(s_t, a_t)$ über den Wert der getätigten Aktion. Hohe Lernraten gleichen den Q -Wert schneller an

das Verhalten der Umgebung an, können jedoch zu Fluktuationen führen, falls die Umgebung stochastische Rewards an den Agenten liefert. Für niedrige Lernraten werden Fluktuationen hingegen verringert, jedoch im Gegenzug die Lernzeit verlängert (vgl. Abbildung 4.2).

In (Watkins, 1989) und (Watkins und Dayan, 1992) wird gezeigt, dass die Q -Funktion gegen Q^* konvergiert falls:

- alle Transitionen des Zustandsraumes unendlich oft durchlaufen werden,
- es sich um einen deterministischen MDP handelt,
- die unmittelbaren Rewards beschränkt sind, d. h. eine positive Konstante c existiert, für die gilt: $(\forall s, a) |\mathcal{R}_{s,s'}^a| \leq c$.

2.7.2 Sarsa

Eine weitere Möglichkeit zum Erlernen der Wertefunktion durch Interaktion bietet der Sarsa-Algorithmus², welcher ursprünglich unter dem Namen *Modified Connectionist Q-Learning* (MCQ-L) vorgestellt wurde (Rummery und Niranjan, 1994). Im Unterschied zu Q -learning basiert dieses Verfahren direkt auf der Strategie des Agenten, wobei für das Update nicht der maximale Q -Wert des Folgezustandes verwendet wird, sondern der Q -Wert der tatsächlich gewählten Aktion im Folgezustand $Q(s_{t+1}, a_{t+1})$ (vgl. Algorithmus 3):

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \right] . \quad (2.27)$$

Durch diesen Unterschied ist es möglich Wissen über das Explorationsverhalten in der Wertefunktion abzubilden, um hierdurch Aktionen zu vermeiden, die im Folgezustand, bei zu hoch eingestellter Explorationsrate, einen übermäßig hoch negativen Reward einbringen können. Diese subtile Unterscheidung im TD-Fehler, über die zu berücksichtigende Information im Folgezustand s' , führt dazu, dass man Q -learning als ein *off-policy* Verfahren bezeichnet, hingegen Sarsa als *on-policy* Verfahren. Dabei gilt anzumerken, dass beide Lernverfahren unter Verwendung der Greedy-

²Sarsa ist die Abkürzung für das 5-Tupel $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})$ der verwendeten Variablen des Lernverfahrens.

Algorithm 3 SARSA nach (Sutton und Barto, 1998)

```

1: Initialize  $Q(s, a)$  arbitrarily, e.g.  $Q(s, a) = 0$  for all  $s, a$ 
2: for each episode do
3:   Initialize start state  $s$ 
4:   Choose  $a$  from  $s$  using policy  $\pi(s, a)$  derived from  $Q$ 
5:   repeat
6:     Take action  $a$ , observe reward  $r$  and successor state  $s'$ 
7:     Choose  $a'$  and  $s'$  using policy  $\pi(s', a')$  derived from  $Q$ 
8:      $\Delta \leftarrow r + \gamma Q(s', a') - Q(s, a)$ 
9:      $Q(s, a) \leftarrow Q(s, a) + \alpha \Delta$ 
10:     $s \leftarrow s'; a \leftarrow a';$ 
11:   until  $s$  is terminal state
12: end for

```

Strategie (vgl. Abschnitt 3.1) dieselben Ergebnisse erzielen, da deterministisch, auf Basis des bislang gelernten Wissens, die Aktion mit dem höchsten Q -Wert gewählt wird. Für eine Veranschaulichung dessen sei auf das Cliff-Walking-Beispiel in Kapitel 5.3 verwiesen.

Für Sarsa gibt es für den Fall von stochastischen Strategien, die zufällige Aktionen zur Erkundung des Zustandsraumes verwenden, keine Konvergenzgarantie. Dies ist darauf zurückzuführen, dass der Q -Wert für Aktion a in Zustand s auf die unterschiedlichen Werte der stochastischen Aktionen a' im Folgezustand s' angepasst wird. Nichtsdestotrotz kann gerade bei Umgebungen mit hohem Gefahrenpotential Sarsa einen höheren Reward erreichen, wie etwa im untersuchten Cliff-Walking-Problem in Kapitel 5.3. Darüber hinaus ist ein Verfahren, welches die Eigenschaften von Q -learning und Sarsa miteinander kombiniert, unter dem Namen Q -SARSA(λ) zu finden (Nissen, 2007).

2.8 Analogien zur Verhaltenspsychologie und Neurobiologie

Elektrophysiologische Aufnahmen vom Gehirn nicht-menschlicher Primaten (Morris u. a., 2006) und Ratten (Roesch u. a., 2007) zeigen, dass doperminerge Neuronen tatsächlich ein TD-Fehler ähnliches Signal übertragen. Dieses Signal ähnelt eher dem beim Lernen von Zustands-Aktions-Werten $Q(s, a)$, jedoch weniger dem von Zustandswerten $V(s)$ (Niv, 2009). Dabei

deuten die Ergebnisse von Morris u. a. auf eine Lernregel analog zu Q -learning hin, hingegen die von Roesch u. a. auf eine Ähnlichkeit zu Sarsa. Ob diese Feststellungen zu bedeuten haben, dass im Gehirn tatsächlich das Lernen nach dem Schema von Abbildung 2.1 stattfindet, oder eher nach dem Schema eine Aktor-Kritiker-Architektur (Sutton, 1984), ist derzeit eine offene Forschungsfrage (Niv, 2009; Niv u. a., 2006).

Bei genauer Betrachtung lassen sich im TD-Modell weitere verhaltenspsychologische Analogien herstellen (Sutton und Barto, 1998, S. 8), insbesondere zu primären Verstärkern (z. B. Hunger stillen) und sekundären Verstärkern (z. B. Geld, mit welchem man sich Nahrung kaufen kann). In diesem Sinne ist im TD-Fehler Δ der unmittelbare Reward r_{t+1} als primärer Verstärker anzusehen, da bei Betätigung einer entsprechenden Aktion ein Bedürfnis direkt befriedigt werden kann. Als sekundärer Verstärker ist hingegen der Zustands-Aktions-Wert des Folgezustandes³ anzusehen, welcher nur durch den Erhalt von primären Verstärkern entstehen kann. Die Werte im Folgezustand zeigen sozusagen den Weg, wie man zum primären Verstärker gelangt, was bei entsprechend hoch eingestelltem Discounting-Parameter γ über sehr viele Aktionen propagiert werden kann.

Bezüglich des Diskontierens von Reward kann eine Analogie zur Verhaltensökonomie hergestellt werden. In diesem Zusammenhang beobachtete Chung (1965) bei Tauben, dass die Diskontierung negativ-exponentiell, zeitlich konsistent erfolgt, was über das Abschwächen mit einem konstanten Faktor γ modelliert werden kann (in Analogie zu Abbildung 2.2). Etwas später wurde diese Feststellung von Ainslie (1974) jedoch zur zeitlich inkonsistenten, hyperbolischen Diskontierung korrigiert. Ein bekanntes Experiment ist hierbei die Entscheidung zwischen zwei Aktionen mit jeweils unterschiedlicher Höhe des Rewards in Abhängigkeit der Zeit. Dies kann z. B. Geld bei Menschen sein (Green u. a., 1994) oder Futter bei Tieren (Ainslie, 1974; Chung und Herrnstein, 1967). Die Wahl der ersten Aktion führt zu einem geringen unmittelbaren Reward; hingegen die zweite Aktion zu einem höheren Reward, der jedoch zeitlich verzögert ist. Der Proband muss sich entscheiden, ob er entweder die sichere, unmittelbare Belohnung nimmt, oder mit einem gewissen Grad an Selbstbeherrschung auf die größere Belohnung wartet, die jedoch mit Unsicherheit verbunden ist. Zum Beispiel ist die Sterbewahrscheinlichkeit bei Menschen im hohen Alter als Unsicherheit zu verstehen (Green u. a., 1994), da z. B. ein 80-jähriger

³ $\arg \max_b Q(s', b)$ bei Q -learning bzw. $Q(s', a')$ bei Sarsa

Mensch eher dazu neigt einen Geldbetrag sofort anzunehmen, anstatt in zehn Jahren das zehnfache davon.

2.9 Diskussion

In diesem Kapitel wurden berechenbare Modelle für Reinforcement Learning vorgestellt und insbesondere das Lernen mit diskreten Aktionen betrachtet. Hierbei wird zwischen modellfreien und modellbasierten Lernverfahren unterschieden, jedoch mit beiden Verfahrensklassen optimales Verhalten erlernt werden. Die Grundlage hierfür bildet die Bellmansche Optimalitätsgleichung für Zustandswertefunktionen (2.17) oder Zustands-Aktions-Wertefunktionen (2.18), welche die vorgestellten Lernverfahren zu lösen anstreben. Modellbasierte Verfahren benötigen hierfür ein vollständiges Umgebungsmodell (Rewards + Transitionswahrscheinlichkeiten), hingegen modellfreie Verfahren nur sogenannte step-by-step Observationen der sensomotorischen Interaktion. Dieser Unterschied macht modellfreie Verfahren, wie Q -learning und Sarsa, neurobiologisch plausibel (Dayan, 2009; Niv, 2009), da Tiere und Menschen ebenso ohne vollständigem Modell lernen können.

Neben dem reinen modellbasierten bzw. modellfreien Lernen existieren auch hybride Lernverfahren, die z. B. das Umgebungsmodell während der Interaktion erlernen. Diesbezüglich wurde in (Tokic u. a., 2009) ein Laufroboter vorgestellt, welcher sich von selbst das Vorwärtsbewegen beibringt (vgl. Kapitel kap:laufroboterProblem). Der Reward ist die zurückgelegte Distanz, die nach jeder getätigten Aktion im Reward-Modell abgespeichert wird und unmittelbar im Anschluss ein Wert-Iterationsschritt für jeden Zustand, der insgesamt 25 Zustände, stattfindet. Als Resultat schafft es der Laufroboter, welcher durch einen ATmega32-Mikrocontroller mit 2kB RAM gesteuert wird, nach dem Einschalten eine effiziente Strategie zur Vorwärtsbewegung innerhalb von ca. 15-20 Sekunden zu erlernen. Rein modellfreie Lernfahren benötigen hierfür gewöhnlich länger (Tokic u. a., 2010a), da durch die Wert-Iteration das Wissen über zukünftige Rewards schneller an Nachbarzustände propagiert wird, in denen sich der Laufroboter jedoch nicht aktiv befand. Modellfreie Verfahren führen hingegen nur ein Update auf den Q -Wert der getätigten Aktion durch, was jedoch durch den Einsatz von Eligibility-Traces (Singh und Sutton, 1996) verbessert werden könnte (jedoch im Rahmen dieser Arbeit nicht explizit untersucht wurde).

Schlussendlich bleibt festzuhalten, dass beim modellfreien Lernen sowie beim Lernen mit unvollständigen Umgebungsmodellen, eine stochastische Komponente in der Aktionsauswahl benötigt wird. Diese ist wichtig, da die Wertefunktion möglichst akkurat approximiert und lokalen Minima entwichen werden soll. Wie die folgenden Kapitel zeigen werden, hängt die Optimierung des kumulierten Rewards signifikant von der Stochastik in der Aktionswahl ab (Exploration / Exploitation), ohne welcher oftmals nur suboptimales Verhalten erlernt wird.

Kapitel 3

Related Work: Exploration und Exploitation

Im modellfreien Reinforcement Learning stehen einem Agenten zwei Optionen für die Aktionsauswahl zur Verfügung. Eine der Möglichkeiten ist, dass sich der Agent erkundend verhält, wodurch er für die Wahl von zufälligen Explorationsaktionen, einen Wissenszuwachs darüber erhält, wie gut bzw. wie schlecht die jeweils getätigten Aktionen waren. Die zweite Möglichkeit ist, dass auf Basis von bereits erlerntem Wissen eine Exploitationsaktion gewählt wird, zur zielgerichteten Optimierung des Rewards.

Die Balance zwischen Exploration und Exploitation ist hierbei eine große Herausforderung. Einerseits verhindert zu viel Exploration die Maximierung des kumulierten Rewards, da zufällige Aktionen einen relativ niedrigen Reward einbringen können. Andererseits kann das Ausnutzen von unsicherem Wissen dazu führen, dass lediglich suboptimale Aktionen gewählt werden, was ebenfalls einer Maximierung des kumulierten Rewards entgegen strebt. Gleichzeitig sind nichtdeterministische Umgebungen eine weitere Herausforderung, zum Beispiel wenn stochastische Rewards empfangen werden, deren Verteilung auch nichtstationär sein kann. Für letzteres ist es wichtig, ein derartiges Umgebungsverhalten zu registrieren und durch eine bevorzugte Wahl von Explorationsaktionen die veränderte Umgebung neu zu erkunden. Typischerweise soll sich ein Agent zu Beginn des Lernprozesses eher erkundend verhalten, da zu diesem Zeitpunkt noch kein Umgebungswissen vorliegt.

Die Begriffserklärung soll zunächst an einem konkreten Beispiel, dem Lernverhalten eines Bettlers, verdeutlicht werden. Angenommen man würde den kumulierten Reward eines Bettlers als die eingenommen Spenden pro Zeiteinheit definieren, z. B. pro Tag oder pro Stunde, so ist dieser typischerweise *stochastisch*, da nicht immer am gleichen Ort, zur selbigen Tageszeit, dieselben Einnahmen erzielt werden (zurückzuführen auf unterschiedlich viele Spender mit jeweils unterschiedlicher Spendenhöhe). Als *stationär* würde man die Verteilung der Rewards bezeichnen, wenn deren Mittelwert zumindest konstant ist. Eine Aktion die hierbei erheblichen Einfluss auf den Mittelwert besitzt, ist u. a. die Platzwahl in der Stadt (entweder eher in der Innenstadt oder am Stadtrand, mit entsprechend variabler Anzahl an Passanten). Hingegen würde die Verteilung der Rewards als *nichtstationär* bezeichnet werden, wenn deren Mittelwert nicht konstant ist, z. B. wenn unvorhersehbare Ereignisse auftreten. Letztere können z. B. wetterbedingt sein oder durch eine Baustelle, wodurch potentielle Spender weiter entfernt vom Bettler entlanglaufen, diesen also nicht sehen können. Infolgedessen fällt der Reward pro Zeiteinheit entsprechend niedriger aus. Um die Summe der Spenden dennoch zu optimieren, sollte sich der Bettler demnach einen neuen Platz suchen, was entweder auf Basis seines bisherigen Wissens erfolgen kann (Exploitation) oder durch das Ausprobieren von bislang nicht genutzten Plätzen (Exploration).

Das Ziel dieses Kapitels ist es, geläufige, berechenbare Explorationsstrategien für modellfreies Reinforcement Learning vorzustellen. Im anschließenden Kapitel werden die eigenen Erweiterungen präsentiert, welche auf einigen Basisstrategien dieses Kapitels aufbauen.

3.1 Die Greedy-Strategie

Eine naheliegende Strategie zur Aktionsauswahl ist es, in Zustand s *greedy* (gierig) eine zufällige Aktion aus der Menge $\mathcal{A}^*(s)$, der bislang als optimal geschätzten Aktionen, zu wählen (Precup und Sutton, 1997; Riedmiller, 2005; Watkins, 1989):

$$\mathcal{A}^*(s) = \arg \max_a Q(s, a) . \quad (3.1)$$

Aktionen dieser Menge besitzen den höchsten, geschätzten Q -Wert in Zustand s , wodurch zu erwarten ist, dass durch eine Wahl deren der kumulierte Reward maximiert wird. In der Praxis verwendet man $\mathcal{A}^*(s)$ um Aus-

wahlwahrscheinlichkeiten für beliebige Aktionen $a \in \mathcal{A}(s)$ in Zustand s zu erhalten:

$$\pi(s, a) = \begin{cases} \frac{1}{|\mathcal{A}^*(s)|} & \text{falls } a \in \mathcal{A}^*(s) \\ 0 & \text{ansonsten} \end{cases} \quad (3.2)$$

Der gleichverteilte Zufall ist deswegen erlaubt, da es für den kumulierten Reward zunächst irrelevant ist, welche der geschätzten, optimalen Aktionen tatsächlich gewählt wird, da augenscheinlich alle Aktionen zu gleich hohem Reward führen. In der Praxis ist es jedoch wichtig die Aktion durch gleichverteilten Zufall zu wählen, da die Q -Werte lediglich Schätzungen darstellen, die evtl. noch mit Unsicherheit behaftet sein können oder Rewards für bestimmte Aktionen dieser Menge nichtstationär sind.

Die Greedy-Strategie nutzt in diesem Sinne das bislang erlernte Umgebungswissen gierig aus. Das Problem ist jedoch, dass dies unabhängig davon geschieht, ob andere Aktionen, $\mathcal{A}(s) \setminus \mathcal{A}^*(s)$, im aktuellen Lernprozess noch unterschätzt werden; also sprich zu mehr Reward führen können, als dies ihre momentane Schätzung angibt. Deswegen klingt eine derartige Auswahlstrategie auf den ersten Blick relativ einfach und naheliegend, ist jedoch oftmals ineffektiv (Tokic, 2010; Tokic u. a., 2010a). Zusätzlich kann nämlich das Problem entstehen, dass die Q -Funktion aufgrund fehlender Exploration in lokale Minima konvergiert. Die Folge ist, dass überwiegend suboptimale Aktionen gewählt werden, die den kumulierten Reward jedoch nicht maximieren.

3.1.1 Die ε -Greedy-Strategie

Eine kleine aber entscheidende Erweiterung zur Greedy-Strategie bietet die ε -Greedy-Strategie (Faußer und Schwenker, 2008, 2010; Varges u. a., 2009; Watkins, 1989). Auf Basis einer globalen, konstanten Explorationsrate, $\varepsilon \in [0, 1]$, wird bei dieser Strategie eine gleichverteilt zufällige Aktion gewählt, die unabhängig von der Wertefunktion ist (vgl. Abbildung 3.1):

$$\pi(s, a) = \begin{cases} \frac{1-\varepsilon}{|\mathcal{A}^*(s)|} + \frac{\varepsilon}{|\mathcal{A}(s)|} & \text{falls } a \in \mathcal{A}^*(s) \\ \frac{\varepsilon}{|\mathcal{A}(s)|} & \text{ansonsten} \end{cases} \quad (3.3)$$

Solch eine Heuristik versucht sicher zu stellen, dass die Wertefunktion nicht in einem lokalen Minimum konvergiert (Sutton und Barto, 1998). Der Parameter ε muss jedoch mit Bedacht vom Experimentator gewählt werden, um einerseits genügend Exploration sicherzustellen, jedoch andererseits nicht übermäßig viele Explorationsschritte zu tätigen. Für $\varepsilon > 0$ und

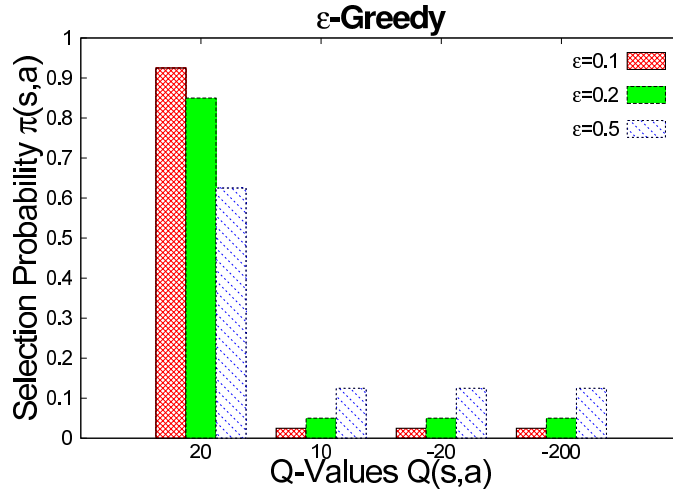


Abbildung 3.1: Die ϵ -Greedy-Strategie in Abhängigkeit unterschiedlicher Explorationsraten ϵ . Man beachte die gleichverteilte Aktionswahl derjenigen Aktionen, die nicht den höchsten Q -Wert besitzen. Man beachte ebenso, dass für $\epsilon = 1$ alle Aktionen mit selbiger Wahrscheinlichkeit gewählt werden würden.

$t \rightarrow \infty$ wird jede Aktion unendlich oft ausgewählt; so dass z. B. eine der Bedingungen erfüllt wird, damit für Q -learning $Q(s, a) \rightarrow Q^*(s, a)$ konvergiert (Watkins und Dayan, 1992). Folglich ist für $\epsilon > 0$ und $t \rightarrow \infty$ die Wahrscheinlichkeit größer gleich $1 - \epsilon$ im Limit eine optimale Aktion zu wählen.

3.1.2 Die decreasing- ϵ -Strategie

Einen Spezialfall der ϵ -Greedy-Strategie stellt die decreasing- ϵ -Strategie dar. Die Idee ist, den Lernprozess mit einer hohen Explorationsrate zu beginnen, welche über die Zeit verringert wird (Vermorel und Mohri, 2005). Die Begründung hierfür ist, dass mit zunehmender Anzahl an Interaktionen, das Wissen über die langfristige Auswirkung von Aktionen sicherer wird und daher eher ausgenutzt werden sollte. In der Praxis kann solch eine Steuerung über einen zusätzlichen Abschwächungsfaktor $\lambda \in (0, 1)$ realisiert werden, mittels welchem nach dem Tätigen einer Aktion die Explorationsrate etwas abgeschwächt wird:

$$\epsilon_{t+1} = \lambda \cdot \epsilon_t . \quad (3.4)$$

Für $t \rightarrow \infty$ konvergiert die Explorationsrate $\epsilon \rightarrow 0$ (vgl. Abbildung 3.2). Für nichtstationäre Rewards ist solch eine Steuerung jedoch ungeeignet. Das

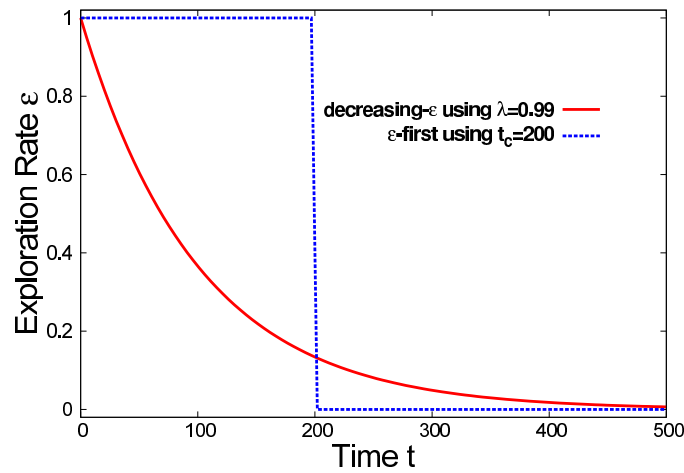


Abbildung 3.2: Zeitlicher Abfall der Explorationsrate bei ε -first- sowie decreasing- ε -Strategien.

Problem ist, dass für $\varepsilon \rightarrow 0$ kein aktives Neuerkunden mehr stattfindet, was jedoch bei Umgebungsänderungen notwendig wäre. Für deterministische Rewards oder stationäre Reward-Verteilungen ist dies hingegen eine interessante Strategie, welche jedoch ein experimentelles „fine-tuning“ des Abschwächungsfaktors λ benötigt (Grześ und Kudenko, 2010; Pérez-Urbe und Sanchez, 1998).

3.1.3 Die ε -first-Strategie

Eine ebenso auf ε -Greedy basierende Strategie, ist die ε -first-Strategie, welche Vermorel und Mohri (2005) am n -armigen Banditen-Problem untersuchen (vgl. Kapitel 4.5). Bei ε -first wird die Lernzeit in zwei Phasen unterteilt: Eine *Explorationsphase* sowie eine *Exploitationsphase* (vgl. Abbildung 3.2). Der Lernprozess beginnt mit der Explorationsphase, innerhalb welcher ausschließlich zufällige Aktionen gewählt werden ($\varepsilon = 1$), um möglichst viel Wissen über die Dynamik der Umgebung zu sammeln. Diese Phase wird auch *Random-Exploration* genannt, deren zeitliche Dauer vom Experimentator festgelegt werden muss. In der anschließenden Exploitationsphase, die zum Zeitpunkt t_c beginnt, wird hingegen per Greedy-Strategie ausschließlich eine der Aktionen mit dem höchsten Q -Wert gewählt ($\varepsilon = 0$), da angenommen wird, dass in der Explorationsphase genügt Wissen über die Umgebung gesammelt wurde.

Wie Vermorel und Mohri am Beispiel des n -armigen Banditen-Problem zeigen (vgl. Kapitel 4.5), ist diese Strategie lediglich für sehr kleine Zustands-

räume von Interesse. Der Hintergrund ist, dass mit zunehmender Anzahl an Zuständen und Aktionen, die benötigte Anzahl an mehrmaligen Durchläufen exponentiell ansteigt, um Wissen über in weiter Ferne liegende Zustände und Aktionen entsprechend zurückpropagieren zu können. Analog gilt dies ebenso für die decreasing- ε -Strategie, weswegen es für beide Strategien für große Zustands- und Aktionsräume schwierig ist, eine sinnvolle Belegung des jeweiligen Explorationsparameters zu finden. Daher finden beide Strategien relativ wenig Anwendung in der Praxis. Hingegen ist zu beobachten, dass ε -Greedy mit konstant niedrig gewählter Explorationsrate oftmals favorisiert wird, da der Explorationsparameter intuitiver zu wählen ist.

3.2 Die Softmax-Strategie

Trotz des Vorteils, dass die ε -Greedy-Strategie sehr effektiv und wenig rechenintensiv die Balance zwischen Exploration und Exploitation steuern kann, hat sie auch einen Nachteil. Das Problem ist, dass Explorationsaktionen gleichverteilt über alle möglichen Aktionen in Zustand s gewählt werden, unabhängig davon, ob bereits Wissen in Form von Q -Werten vorliegt, die darüber Auskunft geben, ob bestimmte Aktionen zu positivem oder eher negativem Reward führen. Wenn beispielsweise in einem Zustand 8 von 10 Aktionen einen Q -Wert von -100 besitzen, hingegen nur zwei Aktionen einen Q -Wert von 1, so ist es naheliegend diese Information ebenfalls bei der Wahl von Explorationsaktionen mitzuberücksichtigen.

Eine Möglichkeit dieses Verhalten zu verbessern bietet die sogenannte Softmax-Strategie (Bridle, 1990; Eck und Wezel, 2008; Kakvi, 2009; Kirkpatrick u. a., 1983). Bei dieser werden Auswahlwahrscheinlichkeiten anhand einer Gibbs-Boltzmann-Verteilung erstellt, auf Basis der bisher geschätzten Q -Werte. Aktionen mit relativ hohem Q -Wert erhalten eine hohe Auswahlwahrscheinlichkeit, andere Aktionen eine entsprechend niedrigere:

$$\pi(s, a) = \frac{e^{\frac{Q(s,a)}{\tau}}}{\sum_b e^{\frac{Q(s,b)}{\tau}}} \quad (3.5)$$

Der positive Parameter τ , auch *Temperatur-Parameter* genannt¹, dient zur Steuerung zwischen Exploration und Exploitation. Hohe Temperaturen

¹In der Literatur wird oftmals auch von der *inversen Temperatur* $\beta \equiv \frac{1}{\tau}$ gesprochen, d. h. Gleichung (3.5) ändert sich zu $\pi(s, a) = \frac{e^{\beta Q(s,a)}}{\sum_b e^{\beta Q(s,b)}}$ (Ishii u. a., 2002; Schweighofer und Doya, 2003)

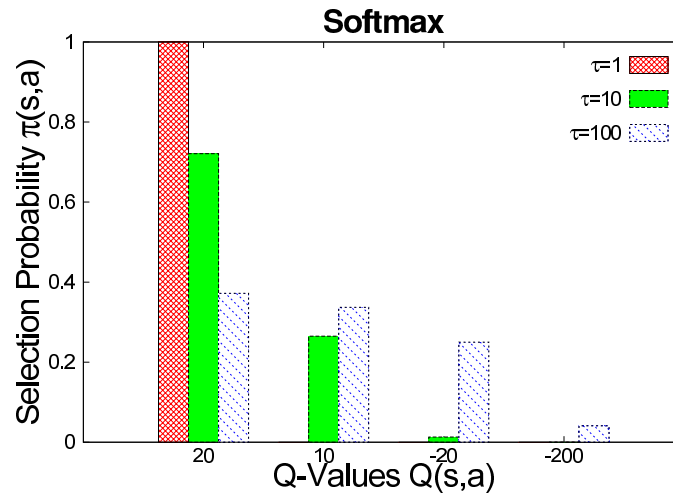


Abbildung 3.3: Die Softmax-Strategie in Abhängigkeit unterschiedlicher Temperaturen τ . Man beachte das erfolgreiche Unterdrücken der Aktion mit dem Q -Wert -200 .

$\tau \rightarrow \infty$ führen dazu, dass alle Aktionen in Zustand s mit (fast) selbiger Wahrscheinlichkeit gewählt werden, da der Quotient im Exponent der e -Funktion für alle Aktionen gegen 0 geht. Niedrige Temperaturen $\tau \gtrsim 0$ führen hingegen dazu, dass das Verhalten *Greedy* wird (vgl. Abbildung 3.3).

Ein Problem der Softmax-Strategie ist die Wahl des Temperatur-Parameters, durch welchen numerische Berechnungsprobleme auf Rechnern entstehen können. Beispielsweise ist in Java 1.6 SE ein 64-Bit Double-Wert definiert durch die Grenzen:

$$\begin{aligned}
 \text{Double.MIN_VALUE} &= 4.9 * 10^{-324} \\
 &\approx e^{1.589} * e^{-324 \ln(10)} \\
 &\approx e^{-744.448}
 \end{aligned}$$

$$\begin{aligned}
 \text{Double.MAX_VALUE} &= 1.7976931348623157 * 10^{308} \\
 &\approx e^{0.586} * e^{308 \ln(10)} \\
 &\approx e^{709.782}
 \end{aligned}$$

Für Gleichung (3.5) bedeutet dies, dass sich der Quotient $Q(s, a)/\tau$ im Intervall $(-744.45, 709.782)$ befinden muss, damit das Ergebnis nicht als $-\infty$ bzw. ∞ interpretiert wird. Da man in der Praxis jedoch nicht weiß, wel-

chen Wert die Q -Werte in einem Zustand annehmen werden, ist es oftmals schwierig geeignete Temperaturen für effizientes Lernen zu finden. Vereinfachen kann man diese Problematik, indem die Q -Werte eines Zustands in ein definiertes Intervall normalisiert werden, z. B. $[V_{\min}, V_{\max}] = [-1, 1]$, was die Wahl des Temperatur-Parameters etwas erleichtert (Tokic und Palm, 2011; Tokic u. a., 2012).

3.3 Meta-Lernstrategien

Wie bereits geschildert, ist es oftmals schwierig geeignete Temperaturen τ für effizientes Lernen zu finden, da hierfür u. a. Wissen über die Höhe der zu erlernenden Q -Werte benötigt wird. Aus diesem Grunde beschäftigen sich Wissenschaftler mit der Fragestellung, wie man einen Parameter auf Basis des Rewards bzw. des Lernfortschritts adaptieren kann. In der Literatur wird dies als *Meta-Lernen* bezeichnet, bei welchem der zu erlernende Parameter als *Meta-Parameter* bezeichnet wird. Kurz gefasst: es wird gelernt, *wie* gelernt werden soll. Wie die folgenden Studien zeigen, wird als Meta-Parameter nicht nur die Temperatur τ in Betracht gezogen, sondern ebenso die Lernrate α und der Diskontierungsfaktor γ .

3.3.1 Reward basierte Adaption

Schweighofer und Doya (2003) stellen ein Verfahren vor, welches nicht nur die Temperatur τ adaptiert, sondern gleichzeitig auch die Lernrate α sowie den Diskontierungsfaktor γ . Das Verfahren steuert diese Meta-Parameter global und zeitabhängig, wobei die grundlegende Funktionsweise auf dem SRV-Algorithmus (*Stochastic Real-Valued*) von Gullapalli (1990) basiert. Ein *SRV-Neuron* erzeugt einen stochastischen Output in Abhängigkeit eines Mittelwerts $\epsilon(t)$ mit Standardabweichung $\sigma(t)$, wobei letztere zur Exploration im Raum von $\epsilon(t)$ benötigt wird². Grundlegend wird ein Meta-Parameter auf Basis eines *mid-term rewards*

$$r_{MT}(t) = \left(1 - \frac{1}{\tau_{MT}}\right) r_{MT}(t-1) + r(t) \quad (3.6)$$

sowie eines *long-term rewards*

$$r_{LT}(t) = \left(1 - \frac{1}{\tau_{LT}}\right) r_{LT}(t-1) + r_{MT}(t) \quad (3.7)$$

²Ein darauf aufbauendes Verfahren wurde von Williams (1992) vorgestellt, welches in Kapitel 6 dieser Arbeit, zur Adaption von beliebigen Explorationsparametern, verwendet wird.

adaptiert und zum Zeitpunkt $t = 0$ mit $r_{MT}(0) = r_{LT}(0) = 0$ initialisiert. Beides sind fortlaufende Mittelwerte, wobei das Verfahren zwei zu wählende Lernraten τ_{MT} und τ_{LT} verwendet. Der Einfluss des unmittelbaren Rewards³ auf den mid-term Reward wird durch Lernrate τ_{MT} gesteuert, hingegen der des mid-term Rewards auf den long-term Reward durch Lernrate τ_{LT} .

Der Diskontierungsfaktor γ wird zum Zeitpunkt t wie folgt bestimmt:

$$\gamma(t) = 1 - e^{-\epsilon(t)} , \quad (3.8)$$

wobei der Aktivitätsterm $\epsilon(t)$, auf Basis der beiden Mittelwerte sowie eines Rauschterms $\sigma(t) \sim \mathcal{N}(0, v)$, gegeben ist durch:

$$\epsilon(t) = \epsilon'(t) + \sigma(t) \quad (3.9)$$

mit

$$\epsilon'(t) = \epsilon'(t-1) + \alpha_\epsilon (r_{MT}(t) - r_{LT}(t)) \sigma(t) . \quad (3.10)$$

α_ϵ gibt eine Lernrate für $\epsilon'(t)$ an, wobei letzteres zum Zeitpunkt $t = 0$ mit $\epsilon'(0) = 0$ initialisiert wird. Schweighofer und Doya schlagen vor, den Rauschterm jeweils nach $n \gg 1$ Aktionen neu zu wählen, da dies einer kleinen spontanen Änderung des tonischen Feuerns von derartigen SRV-Neuronen entspricht (biologische Analogie des Verfahrens).

In den Gleichungen (3.9) und (3.10) erkennt man für das Verfahren folgende Eigenschaften:

1. $\epsilon(t)$ wird höher, falls $r_{MT}(t) > r_{LT}(t)$,
2. $\epsilon(t)$ wird kleiner, falls $r_{MT}(t) < r_{LT}(t)$,
3. $\epsilon(t)$ bleibt konstant, falls $r_{MT}(t) \approx r_{LT}(t)$.

Da Schweighofer und Doya keine expliziten Lernregeln für die Lernrate α und Temperatur τ angeben, schlagen Kobayashi u. a. (2009) die Steuerung wie folgt vor:

$$\alpha(t) = e^{-\epsilon(t)} \quad (3.11)$$

$$\tau(t) = \frac{1}{e^{\epsilon(t)} - 1} . \quad (3.12)$$

³Schweighofer und Doya verwenden als Notation für den unmittelbaren Reward $r(t)$ anstatt r_{t+1} .

Die Lernrate $\alpha(t)$ und Temperatur $\tau(t)$ nehmen für $r_{MT}(t) > r_{LT}(t)$ hohe Werte an, da $\epsilon(t)$ ansteigt, hingegen niedrige Werte für $r_{MT}(t) < r_{LT}(t)$. Dieses Verhalten ist auf die Bedingung zurückzuführen, dass beide Parameter bei Umgebungsänderungen hohe Werte annehmen sollten, da falsche Prädiktionen mehr Exploration und schnelleres Lernen (durch hohe Lernraten) erfordern. Genau umgekehrt verhält es sich für den Diskontierungsfaktor $\gamma(t)$, welcher bei Änderungen in der Umgebung eher zu kurzfristiger Reward-Maximierung führen soll, d. h. die lokale Umgebung optimiert wird, anstatt die globale Maximierung anzustreben.

Zusammengefasst erhält man durch das Verfahren von Schweighofer und Doya eine Steuerung für drei Meta-Parameter, muss jedoch fünf Parameter von Hand belegen ($\tau_{MT}, \tau_{LT}, \alpha_\epsilon, v, n$), deren Werte nicht trivial zu bestimmen sind und stark von der Umgebungsdynamik abhängen. Ein weiterer Nachteil ist, dass die Steuerung in Abhängigkeit der Zeit erfolgt, was den Einsatz in Umgebungen mit zeitverzögerten Rewards oder stets gleich hohen Wegekosten erschwert. Beispielhaft sei diesbezüglich das in Kapitel 5.5 beschriebene Mountain-Car-Problem zu erwähnen, bei welchem der Agent für jede getätigte Aktion einen unmittelbaren Reward in Höhe von $r_{t+1} = -1$ erhält, und es hierbei das Ziel ist, die Summe dieser Wegekosten zu minimieren. Das Verfahren würde konsequenterweise dazu tendieren, dass der mid- und long-term Reward stets den Wert $r_{MT} = r_{LT} = -1$ annehmen, da der Reward keine für das Verfahren verwertbare Umgebungsinformation beinhaltet. Folglich bringt das Verfahren nur Vorteile in Umgebungen mit (nahezu) unmittelbaren Rewards, wie Schweighofer und Doya in einem ca. 8-10 Zustände großen MDP sowie dem Pendulum Swing-Up Task demonstrieren.

3.3.2 TD-Fehler basierende Adaption

Kobayashi u. a. (2009) schlagen ein Verfahren vor, welches dieselben Meta-Parameter (α , γ und τ) adaptiert, ebenso in globaler und zeitabhängiger Vorgehensweise. Das Verfahren differenziert sich jedoch von (Schweighofer und Doya, 2003), da der Absolutwert des TD-Fehlers $|\Delta(t)|$ zum Zeitpunkt t als Lerninformation verwendet wird (vgl. Gleichung 2.26), welcher nicht nur den unmittelbaren Reward, sondern auch den Q -Wert der getätigten Aktion sowie den der geschätzten optimalen Aktion im Folgezustand mitberücksichtigt. Dies hat zum Vorteil, dass Informationen von zukünftigen Zuständen und Rewards mit enthalten sind, wodurch das Verfahren auch für Umgebungen mit zeitverzögerten Rewards einsatzfähig wird.

Zur Steuerung wird ein gleitender Mittelwert $\bar{\Delta}$ über den Absolutwert des TD-Fehlers eingeführt:

$$\bar{\Delta}(t) = \left(1 - \frac{1}{\alpha_{\Delta}}\right) \bar{\Delta}(t-1) + \frac{1}{\alpha_{\Delta}} |\Delta(t)|, \quad (3.13)$$

wobei α_{Δ} eine positive Lernrate angibt.

Basierend auf $\bar{\Delta}$ werden die drei Meta-Parameter wie folgt adaptiert:

$$\alpha(t) = \frac{2}{1 + e^{-\bar{\Delta}(t)}} - 1, \quad (3.14)$$

$$\gamma(t) = \frac{2}{1 + e^{\bar{\Delta}(t)}}, \quad (3.15)$$

$$\tau(t) = e^{\bar{\Delta}(t)} - 1. \quad (3.16)$$

Das Verfahren verhält sich dabei wie folgt: Die Lernrate $\alpha(t)$ und Temperatur $\tau(t)$ nehmen niedrige Werte an, falls der gleitende Mittelwert des TD-Fehlers $\bar{\Delta}(t)$ niedrig ist, d. h. die Prädiktion des zukünftigen Rewards mit der aktuellen Beobachtung nahezu übereinstimmt. Hingegen werden hohe Werte angenommen, desto größer $\bar{\Delta}(t)$ ist, da die bislang gelernten Q -Werte Rewards der Umgebung noch unzureichend präzisieren. In Analogie zu (Schweighofer und Doya, 2003) verhält sich hingegen der Diskontierungsfaktor $\gamma(t)$ entgegengesetzt zur Lernrate $\alpha(t)$, da bei falschen Prädiktionen eher die lokale Umgebung optimiert werden soll.

3.3.3 Aktionswert basierte Adaption

Ferner stellen Ishii u. a. (2002) ein Verfahren vor, welches die Temperatur τ entweder zustandsbasiert oder in globaler Form adaptiert. Das Verfahren arbeitet modellbasiert, d. h. es wird zusätzlich zur Q -Funktion auch das Umgebungsmodell mitgelernt (Transitionswahrscheinlichkeiten $\mathcal{P}_{ss'}^a$), was auf den Einsatz in partiell beobachtbaren Umgebungen zurückzuführen ist. Generell wird für die Steuerung der Temperatur die *Konfidenz* eines Zustands verwendet, welche auf der Varianz von Q -Werten in Zustand s basiert. Auf die weiteren Details wird jedoch an dieser Stelle verzichtet, da diese Arbeit zum Ziel hat modellfreie Verfahren zu untersuchen. Es sei lediglich anzumerken, dass vom Experimentator drei Lernparameter von Hand vorgegeben werden: (1) eine konstante Temperatur τ_0 zur Skalierung der Q -Werte, (2) eine minimale Temperatur τ_r sowie (3) eine Lernrate α_r (in der globalen Variante).

3.4 Die Reinforcement-Comparison-Strategie

Die bisher beschriebenen Strategien setzen das Vorhandensein einer Wertefunktion voraus. Einen anderen Ansatz zur Steuerung des Explorationsverhaltens verfolgen Lernverfahren, die Aktionen mit dem Ergebnis von vorher getätigten Aktionen im selbigen Zustand vergleichen. Im Folgenden wird nun solch ein Verfahren vorgestellt, um ein Gesamtbild zu vermitteln.

Eine intuitive Annahme im Reinforcement Learning ist, dass Aktionen mit einem relativ hohen Reward häufiger gewählt werden sollen, im Gegensatz zu Aktionen, die einen relativ niedrigen Reward einbringen. Wie aber soll der Agent wissen, ob beispielsweise ein Reward in Höhe von "8" hoch oder niedrig ist? Eine Möglichkeit hierfür bietet das Vergleichen mit einem Referenz-Reward, z. B. dem gemittelten Reward der letzten Aktionen im selbigen Zustand. Lernverfahren, die auf dieser Idee basieren, werden daher auch *Reinforcement-Comparison*-Verfahren genannt, welche in der Praxis manchmal zu einem höheren Reward führen, gegenüber Lernverfahren, die auf Wertefunktionen basieren (Sutton und Barto, 1998, S. 41).

Reinforcement-Comparison-Verfahren verwenden keine Schätzungen einer Wertefunktion, sondern basieren nur auf dem unmittelbaren Reward r_{t+1} . Um zielgerichtet Aktionen wählen zu können, werden Aktionspräferenzen $p(s, a)$ verwaltet, die auf Basis eines zustandsabhängigen Referenz-Rewards $\bar{r}(s)$, auch *Baseline* genannt, adaptiert werden⁴. Die Aktionsauswahl findet z. B. durch Softmax statt, wodurch die Auswahlwahrscheinlichkeiten $\pi(s, a)$ für alle Aktionen im aktuellen Zustand erhalten werden:

$$\pi_t(s, a) = \frac{e^{p_t(s, a)}}{\sum_b e^{p_t(s, b)}} . \quad (3.17)$$

Nach dem Ausführen einer Aktion wird die zugehörige Aktionspräferenz in Abhängigkeit des Referenz-Rewards und des tatsächlich erhaltenen Rewards r_{t+1} adaptiert:

$$p_{t+1}(s, a) = p_t(s, a) + \beta [r_{t+1} - \bar{r}_t(s)] , \quad (3.18)$$

wobei $0 < \beta \leq 1$ eine Lernrate angibt. Aktionen, für die der unmittelbare Reward höher ist als der Referenz-Reward, bestärken die Aktionspräferenz $p(s, a)$ für Aktion a in Zustand s . Für getätigte Aktionen, bei denen der un-

⁴Sutton und Barto (1998, S. 41ff.) geben eine zustandslose Notation an, welche im Rahmen dieser Arbeit für mehrere Zustände generalisiert verwendet wird.

mittelbare Reward geringer ist als der Referenz-Reward, wird hingegen die Aktionspräferenz abgeschwächt. Schlussendlich wird nach jeder getätigten Aktion der Referenz-Reward wie folgt adaptiert:

$$\bar{r}_{t+1}(s) = \bar{r}_t(s) + \rho[r_{t+1} - \bar{r}_t(s)] , \quad (3.19)$$

wobei der Parameter $0 < \rho \leq 1$ ebenfalls eine positiv zu wählende Lernrate angibt.

3.5 Die Pursuit-Strategie

Ein weiter Ansatz, der die Idee von Aktionspräferenzen mit Lernverfahren für Wertefunktionen vereint, sind sogenannte Pursuit⁵-Verfahren (Sutton und Barto, 1998; Thathachar und Sastry, 1985). Die Idee ist, dass Aktionen bevorzugt werden sollen, die im Moment optimal in Bezug auf die geschätzte Wertefunktion erscheinen⁶. In der einfachsten Form sind hierbei die Auswahlwahrscheinlichkeiten $\pi(s, a)$ gleich denen der berechneten Aktionspräferenzen $p(s, a)$:

$$\pi(s, a) = p(s, a) .$$

Nach einer getätigten Aktion in Zustand s , werden dessen Aktionspräferenzen auf Basis der Wertefunktion adaptiert, um momentan optimal erscheinende Aktionen in Zukunft häufiger zu wählen. Hierfür wird nach einem Lernschritt, z. B. durch Q -learning oder Sarsa, zum Zeitpunkt $t + 1$ die Menge aller optimalen Aktionen in Zustand s bestimmt:

$$\mathcal{A}_t^*(s) = \arg \max_{a \in \mathcal{A}(s)} Q_{t+1}(s, a) . \quad (3.20)$$

Die Aktionspräferenzen der optimalen Aktionen, $a^* \in \mathcal{A}_t^*(s)$, werden mittels einer Lernrate $0 < \beta \leq 1$ mehr nach Richtung 1 erhöht:

$$\pi_{t+1}(s, a^*) = \pi_t(s, a^*) + \beta \left[\frac{1}{|\mathcal{A}_t^*(s)|} - \pi_t(s, a^*) \right] , \quad (3.21)$$

⁵Pursuit: engl. streben, verfolgen.

⁶Sutton und Barto (1998, S. 43f.) geben eine zustandslose Notation an, welche im Rahmen dieser Arbeit für mehrere Zustände generalisiert verwendet wird.

um die Wahl dieser für die Zukunft zu bestärken. Alle anderen Aktionspräferenzen, $a \in \mathcal{A}_t(s) \setminus \mathcal{A}_t^*(s)$, werden hingegen mehr nach Richtung 0 abgeschwächt:

$$\pi_{t+1}(s, a) = \pi_t(s, a) + \beta[0 - \pi_t(s, a)] . \quad (3.22)$$

Folglich ist die Summe aller Aktionspräferenzen in Zustand s zu jedem Zeitpunkt $\sum_a \pi(s, a) = 1$, falls diese zu Beginn des Lernprozesses mit $\pi(s, a) = \frac{1}{|\mathcal{A}(s)|}$ initialisiert werden.

3.6 Counter-Based-Exploration-Strategien

Eine Verfahrensfamilie, die das Explorationsverhalten basierend auf Aktionszählern steuert, wurde von Thrun (1992) vorgestellt. *Counter-Based-Exploration-Strategien* steuern die Exploration *gerichtet* und können unter bestimmten Gegebenheiten eine höhere Performanz erreichen, im Gegensatz zu *ungerichteten* Strategien, wie z. B. ϵ -Greedy oder Softmax. Mit der Bezeichnung *gerichtet* ist hierbei gemeint, dass Informationen über den Explorationsprozess mit einbezogen werden (die Aktionszähler). Im Gegensatz zu Softmax und ϵ -Greedy benötigen derartige Verfahren jedoch einen zusätzlichen Speicher für die Aktionszähler, was im Gegenzug ermöglicht die optimale Strategie π^* , unter bestimmten Gegebenheiten, in Polynomialzeit zu finden (Thrun, 1992).

In der einfachsten Variante wird jedem Zustand ein Aktionszähler $c(s)$ zugewiesen, welcher während des Lernprozesses misst, wie häufig Zustand s bislang durchlaufen wurde. Diese Information fließt in eine Bewertungsfunktion für Zustands-Aktions-Paare ein, $p(s, a)$, die sich aus einem *Explorations-* sowie einem *Exploitationsterm* zusammensetzt:

$$p(s, a) = \underbrace{\rho \cdot \Phi(a)}_{\text{Exploitationsterm}} + \underbrace{\frac{c(s)}{E[c|s, a]}}_{\text{Explorationsterm}} , \quad (3.23)$$

Der Exploitationsterm $\Phi(s, a)$ ist hierbei definiert als der gewichtete Wert möglicher Folgezustände, unter Berücksichtigung der Transitionswahrscheinlichkeiten $\mathcal{P}_{ss'}^a$ für das Wählen von Aktion a in Zustand s :

$$\Phi(a) = \sum_{s'} \mathcal{P}_{ss'}^a \cdot V(s') . \quad (3.24)$$

Hingegen gibt der Aktionszähler im Quotienten des Explorationsterms die gewichtete Summe der erwarteten Counter-Werte möglicher Folgezustände an:

$$E[c|s, a] = \sum_{s'} \mathcal{P}_{ss'}^a \cdot c(s') , \quad (3.25)$$

die durch das Wählen von Aktion a in Zustand s erreicht werden können. Hierfür werden im Explorationsterm zwei Spezialfälle definiert: $\frac{0}{0} = 0$ und $\frac{z}{0} = \infty$ für alle $z > 0$. Der konstante Parameter $\rho \geq 0$ wird zur Steuerung des Explorationsverhaltens verwendet und muss vom Experimentator mit Bedacht gewählt werden.

Schlussendlich erfolgt die Aktionsauswahl bei Counter-Based-Exploration nicht mehr zufällig, sondern deterministisch anhand der Greedy-Strategie, wobei zu jedem Zeitschritt zufällig eine der Aktionen in Zustand s gewählt wird, die $p(s, a)$ maximieren:

$$\begin{aligned} \mathcal{A}^*(s) &= \operatorname{argmax}_{a \in \mathcal{A}(s)} p(s, a) \\ \pi(s, a) &= \begin{cases} \frac{1}{|\mathcal{A}^*(s)|} & \text{falls } a \in \mathcal{A}^*(s) \\ 0 & \text{ansonsten} \end{cases} \end{aligned} \quad (3.26)$$

3.6.1 Counter-Based-Exploration mit Abschwächung

Aktionszähler allein enthalten keine Information darüber, *wann* ein möglicher Folgezustand zuletzt besucht wurde, was für effizientes Lernen jedoch u. U. von Bedeutung ist. Beispielsweise sollte bei zwei gleichwertigen Folgezuständen genau der Zustand bevorzugt werden, der am längsten nicht besucht wurde (Thrun, 1992). Die Begründung hierfür ist, dass die Zeitspanne größer ist, in welcher unvorhersehbare Ereignisse aufgetreten sein könnten. Um diese Information ebenfalls zu berücksichtigen wurde das Verfahren um einen positiven Abschwächungsparameter $\lambda \lesssim 1$ erweitert, durch welchen alle Aktionszähler nach jeder Aktion ein wenig abgeschwächt werden:

$$c(s) \leftarrow \lambda \cdot c(s) \quad \forall s \in \mathcal{S} . \quad (3.27)$$

In Gleichung (3.23) wird länger nicht besuchten Folgezuständen, bei ursprünglich gleichem Aktionszähler, eine höhere Priorität gegeben. Dies resultiert daraus, dass $E[c|s, a]$ für den am längsten nicht besuchten Zustand immer kleiner wird und demzufolge ein höherer Wert im Explorationsterm entsteht.

3.6.2 Die Error-/Counter-Based-Exploration-Strategie

Eine weitere Möglichkeit ist es, den Wertunterschied der letzten Wertaktualisierung ΔV_{last} als zusätzliche Heuristik mit in die Aktionsbewertung einzubeziehen (Thrun, 1992). Ein hoher Wertunterschied soll dazu führen, dass eine Aktion bevorzugter gewählt wird. Dieses Vorgehen wird dadurch begründet, dass die Aktion scheinbar in der Vergangenheit zu selten gewählt wurde und demnach ebenfalls Unsicherheiten in den Folgezuständen zu erwarten sind. Um diese Information ebenfalls mit einzubeziehen, wird Gleichung (3.23) um einen zusätzlichen *Error-Term* erweitert:

$$p(s, a) = \rho \cdot \Phi(a) + \frac{c(s)}{E[c|s, a]} + \underbrace{\beta \cdot E[\Delta V_{last}|s, a]}_{\text{Error-Term}}, \quad (3.28)$$

wobei $\beta > 0$ eine positive Lernrate angibt, welche die Fehlerheuristik in der Aktionsauswahl steuert. $E[\Delta V_{last}|s, a]$ gibt den erwarteten Wertunterschied an, welcher zu Beginn des Lernprozesses mit einem hohen Wert initialisiert wird. Bei mehreren Folgezuständen mit gleichem Counter-Wert $E[c|s, a]$, wird folglich demjenigen Folgezustand Priorität gegeben, dessen erwarteter Wertunterschied $E[\Delta V_{last}|s, a]$ am höchsten ist, d. h. in welchem die Umgebung bislang noch unzureichend erkundet wurde. Eine konnektionistische Variante dieser Idee ist zudem in (Thrun und Möller, 1992) zu finden.

3.6.3 Die Recency-Based-Exploration-Strategie

Eine strengere Vorgehensweise, welche die Limitierungen von rein auf Aktionszählern basierenden Strategien zu überwinden versucht, bietet die *Recency-Based-Exploration-Strategie* (Sutton, 1990; Thrun, 1992). Die Idee ist, dass bei der Aktionswahl Folgezustände bevorzugt werden sollen, die in der Vergangenheit seltener besucht wurden. Hierfür erhält jeder Zustand einen Neuigkeitswert $\zeta(s)$, womit die Anzahl an Zeitschritten seit dem letzten Durchlaufen von Zustand s gemessen wird⁷. Auf Basis dieses Wertes wird eine Evaluation über alle möglichen Aktionen durchgeführt:

$$p(s, a) = \rho \cdot \Phi(s, a) + \underbrace{\sqrt{E[\zeta|s, a]}}_{\text{Recency-Term}}, \quad (3.29)$$

⁷Üblicherweise der Zeitstempel t des letzten Besuches von Zustand s .

wobei $\rho \geq 0$ einen Parameter zur Gewichtung des Anteils der Wertefunktion darstellt. Der Term $E[\zeta|s, a]$ gibt den erwarteten Zeitstempel für das Wählen von Aktion a in Zustand s an. Die Aktionsauswahl findet ebenfalls deterministisch durch die Greedy-Strategie statt, auf Basis von $p(s, a)$.

Es gilt anzumerken, dass mit dieser Strategie Aktionen bestärkt werden, die ganz selten in der Vergangenheit gewählt wurden, jedoch erst kürzlich vom Agenten evtl. einmal. Der Recency-Term $\sqrt{E[\rho|s, a]}$ wurde bereits von Sutton (1990) als *exploration bonus*, in ähnlichem Kontext, bezeichnet.

3.7 Die Strategie der optimistischen Initialwerte

Eine einfache aber manchmal sehr effektive Heuristik, bietet die Strategie der *optimistischen Initialwerte* (Precup und Sutton, 1997; Sutton, 1996; Sutton und Barto, 1998). Die Idee ist, dass zu Beginn des Lernprozesses alle Q -Werte mit einem hohen Wert initialisiert werden, welcher weit über dem wahren Q^* -Wert liegen sollte. Während des Lernens verwendet der Agent die Greedy-Strategie, und lernt durch Interaktion die tatsächlichen Q -Werte in Abhängigkeit der erhaltenen Rewards. In der Praxis wurde dies beispielsweise von Sutton (1996) zum Erlernen von Strategien für die Puddle-World- und Mountain-Car-Probleme (vgl. Abschnitt 5.5) verwendet. Das Problem ist jedoch, dass es für einen Experimentator oftmals schwierig ist zu definieren, wie hoch die Initialwerte sein müssen, um effizient zum Lernen verwendet werden zu können. Zudem ist solch eine Strategie für nicht-deterministische Umgebungen ungeeignet, da sie nur auf den Initialwerten basiert, d. h. kein explizites Neuerkunden integrieren.

3.8 Diskussion

In diesem Kapitel wurden mehrere Strategien vorgestellt, für die zielgerichtete Aktionswahl im modellfreien Reinforcement Learning. Strategien wie ϵ -Greedy und Softmax benötigen hierfür nur die momentanen Informationen der Wertefunktion, um auf Basis deren sowie eines globalen Explorationsparameters, das Explorationsverhalten zu steuern. Reinforcement-Comparison sowie das Verfahren von Schweighofer und Doya (2003) steuern Exploration und Exploitation nur auf Basis des unmittelbaren Rewards r_{t+1} . Aufgrund dessen fließt in die Aktionsauswahl kein Wissen über zukünftige Aktionen mit ein, weshalb derartige Verfahren für Lernprobleme mit zeitlich verzögertem Reward eher ungeeignet sind. Pursuit-Strategien hingegen versuchen das Wissen über Folgezustände mit zu integrieren, in-

dem Aktionspräferenzen auf Basis der getätigten Aktion und des resultierenden Aktionswertes erstellt werden. Counter-Strategien versuchen hingegen selten gewählte Aktionen trotzdem hin und wieder in Betracht zu ziehen, da die Möglichkeit besteht, dass ein Aktionswert von geschätzten schlechten Aktionen noch unterschätzt wird.

Trotz dass in der RL-Literatur diverse Ansätze zur Steuerung von Exploration/Exploitation vorgestellt wurden, ist in der Literatur beobachtbar, dass ϵ -Greedy und Softmax oftmals die Strategien erster Wahl sind, wobei ϵ -Greedy scheinbar die beliebtere von beiden ist. Plausible Erklärungen hierfür können sein:

1. dass kein zusätzlicher Speicher für explorationsspezifische Daten benötigt wird,
2. und aufgrund dessen beide Verfahren geeignet für Lernprobleme mit kontinuierlichen Zustandsdimensionen sind.

Diesbezüglich äußerte sich Richard S. Sutton, einer der Pioniere im Reinforcement Learning, in einem Interview gegenüber der Zeitschrift KI (Heidrich-Meisner, 2009, S. 42):

„KI: Do you have rules of thumb when to use which approach?”

Richard S. Sutton: My rules of thumb are extremely simplistic—I guess because I tend to focus on the general problem of AI rather than particular applications. Thus, my rule of thumb is to use Sarsa if possible, and actor-critic methods when pressed. By actor-critic methods, I would now mean the newer policy-gradient methods, using in conjunction with an estimated state-value function. **I tend to use linear function approximation with constant step sizes and epsilon-greedy action selection. In other words, I use the simplest possible methods that I can understand very well. For my purposes, that is usually sufficient.**“

Darüber hinaus verglichen Vermorel und Mohri unterschiedliche Strategien am n -armigen Banditen-Problem (vgl. Abschnitt 4.5) mit der Schlussfolgerung (Vermorel und Mohri, 2005, S. 437):

„One remarkable outcome of our experiments is that the most naive approach, the ϵ -greedy strategy, proves to be often hard to beat.“

In der Zeitschrift *Nature* berichten hingegen Daw u. a., dass sich das menschliche Explorationsverhalten eher dem einer Softmax-Strategie ähnelt (Daw u. a., 2006, S. 876):

„We compared models by using the likelihood of the subjects' choices given their experience, optimized over free parameters. This comparison (Supplementary Tables 1 and 2) revealed strong evidence for value-sensitive (softmax) over undirected (ϵ -greedy) exploration. There was no evidence to justify the introduction of an extra parameter that allowed exploration to be directed towards uncertainty (softmax with an uncertainty bonus): at optimal fit, the bonus was negligible, making the model equivalent to the simpler softmax.“

Im Vergleich dazu argumentiert Thrun (1992), dass *ungerichtete* Strategien, wie z. B. ϵ -Greedy und Softmax, zu schlechterem Reward führen können. Hierbei zeigen (Thrun, 1992; Whitehead, 1991a,b), dass für MDPs mit deterministischer und ergodischer Eigenschaft, die optimale Strategie in Polynomialzeit gefunden werden kann. Der Nachteil wiederum ist, dass bei solchen Verfahren für jeden Zustand zusätzliche Informationen gespeichert werden müssen, die bei kontinuierlichen Zustandsdimensionen durch Funktionsapproximatoren schwer zu erlernen sind (Rummery und Niranjan, 1994, S. 4):

„Although various methods have been suggested for use in discrete state-space systems (Thrun 1992), they are not generally applicable to systems using continuous function approximators.“

Analog gilt dies auch für Strategien wie Reinforcement-Comparison sowie Pursuit.

Ein weiteres Problem von auf Aktionszählern basierenden Strategien (Auer, 2002; Auer u. a., 2002; Kocsis und Szepesvári, 2006; Thrun, 1992; Thrun

und Möller, 1992) ist der Mix aus erwartetem Reward und Counter-Werten in einer gemeinsamen Evaluierungsfunktion $p(s, a)$ (siehe Gleichung (3.23)). Dieser Mix kann gefährlich werden, wenn beispielsweise in einem Lernproblem nur ein sehr geringer Reward in Höhe von $r_{t+1} = 0.0001$, z. B. für das Betreten eines absorbierenden Endzustandes (Mitchell, 1997, S. 371–372), erhalten wird, hingegen für alle anderen Aktionen lediglich $r_{t+1} = 0$. Daraus resultiert, dass der Exploitationsterm stets ein sehr niedriger Wert ist und aufgrund dessen der Explorations-(Counter)-Term in der Evaluierungsfunktion $p(s, a)$ dominieren würde⁸; d. h. das erlernte Wissen über die Umgebung, die vom Exploitationsterm repräsentiert wird, praktisch keine Auswirkung auf die Aktionswahl besitzt. Natürlich könnte der Explorations-Term dementsprechend durch einen konstanten Faktor abgeschwächt werden (Auer, 2002; Auer u. a., 2002; Kocsis und Szepesvári, 2006), dessen Belegung jedoch das Wissen über die Reward-Funktion voraussetzt. Für stochastische Umgebungen autonomer Systeme, die beispielsweise den Reward über Sensoren messen (Tokic und Bou Ammar, 2012; Tokic u. a., 2009), sind solche Verfahren daher eher unpraktikabel.

Die genannten Argumente scheinen daher plausible Erklärungen dafür zu sein, weshalb einfache Strategien wie ε -Greedy und Softmax in der Praxis breite Anwendung finden; beispielsweise beim Erlernen von Spielstrategien:

ε -Greedy: 4-Gewinnt (Faußer und Schwenker, 2008), Tetris (Groß u. a., 2008) oder Dame (Faußer und Schwenker, 2010)

Softmax: Othello/Reversi (Eck und Wezel, 2008) oder Black-Jack (Kakvi, 2009).

Schlussendlich bleibt festzuhalten, dass alle in diesem Kapitel vorgestellten Explorationsstrategien mindestens einen Parameter besitzen, der über Erfolg oder Misserfolg beim Lernen entscheidet. Dieser muss vom Experimentator von Hand belegt werden, was in der Praxis viel Experimentierzeit in Anspruch nehmen kann. Aus diesem Grund stellen die nun folgenden Kapitel die eigenen Ansätze vor, welche das Ziel verfolgen, dieses Problem zu entschärfen.

⁸gg. dass ein Zeitschritt den Aktionszähler $c(s)$ um jeweils den Wert 1 inkrementiert.

Kapitel 4

VDBE-Strategien

Im letzten Kapitel wurden diverse Strategien vorgestellt, mit welchen das Verhältnis zwischen Exploration und Exploitation gesteuert werden kann. Hierbei stellte sich heraus, dass ε -Greedy und Softmax als *praxistaugliche* Strategien bezeichnet werden, da sie ohne zusätzliches Wissen über den Explorationsprozess auch in kontinuierlichen Zustandsräumen verwendet werden können. Darüber hinaus ist ε -Greedy in der Praxis oftmals die Methode der ersten Wahl (Heidrich-Meisner, 2009), welche bei richtiger Parameterbelegung als schwer zu schlagen gilt (Vermorel und Mohri, 2005).

Das nun folgende Kapitel stellt die sogenannten VDBE-Strategien vor, welche die Explorationsrate von ε -Greedy auf Basis des Wertunterschieds adaptieren. Der Wertunterschied, welcher sich aus dem Produkt zwischen der Lernrate α und des TD-Fehlers Δ zusammensetzt, ist ein natürliches Lernsignal, welches beim Update der Wertefunktion entsteht. Im Unterschied zum Verfahren von Kobayashi u. a. (2009), wird hierbei der Explorationsparameter zustandsbasiert adaptiert, was zum Ziel hat, nur in Zuständen mit Unsicherheit explorativ zu sein, und auf der angenommenen Markoveigenschaft der Umgebung basiert. Die beschriebene Steuerung des Explorationsverhaltens kann daher im Sinne von Abschnitt 3.3 ebenso als *Meta-Lernen* bezeichnet werden (Ishii u. a., 2002; Kobayashi u. a., 2009; Schweighofer und Doya, 2003).

Ferner wird durch eine zusätzliche Kombination mit der Softmax-Strategie, das Verfahren unempfindlicher gegenüber Rauschen in der Wertefunktion gestaltet (Tokic und Palm, 2011; Tokic u. a., 2012), was auf der MBE-Strategie von Wiering (1999) basiert. Im anschließenden Kapitel 5 werden

mehrere Untersuchungsergebnisse präsentiert, die durch die Anwendung der VDBE-Strategien in unterschiedlichen Lernproblemdimensionen gesammelt werden konnten. Hierbei wird ebenso der Frage nachgegangen, wie ein derartiges Verfahren mit stochastischen Rewards umgehen kann, was laut Aussage von Kobayashi u. a. (2009) bislang nicht untersucht wurde. Hierbei zeigt sich, dass zum Lösen dieser Problemstellung der TD-Fehler allein unzureichend ist, sondern ebenso die Lernrate α mit in Betracht gezogen werden muss.

4.1 Die Idee

Ein neuartiger Ansatz ist das ausschließliche Betrachten von Wertunterschieden zur Steuerung von Exploration und Exploitation (Tokic, 2010; Tokic und Palm, 2011; Tokic u. a., 2012). Die Idee ist, dass beim Erlernen der Wertefunktion Wertunterschiede als Maß dazu dienen können, die (Un)Sicherheit, über die Auswirkung von Aktionen in der Umgebung, zu beschreiben. Prinzipiell ist es möglich zu argumentieren, dass kleine Wertunterschiede als *sicheres Wissen* interpretiert werden können, da die Wertefunktion bereits den wahren Wert der getätigten Aktion gut prädiziert. Analog können große Wertunterschiede als *Unsicherheit* interpretiert werden, da die Observation des Rewards noch unzureichend von der Wertefunktion prädiziert wird.

Eine wichtige Annahme für solch ein Verfahren ist, dass die Wertefunktion unter bestimmten Gegebenheiten konvergiert¹, wodurch die Beträge der Wertänderungen mit zunehmender Sicherheit, über die Auswirkung von Aktionen, immer kleiner werden. In diesem Fall soll der Agent bevorzugt Exploitationsaktionen wählen, hierdurch also das bislang erworbene Wissen *gierig* ausnutzen. Kommt der Agent hingegen in unbekannte Zustände, oder ändert sich die Dynamik der Umgebung, so sind typischerweise die Beträge der Wertänderungen hoch, was folglich als Unsicherheit interpretiert wird. Der Agent sollte sich in diesem Fall also eher explorativ verhalten und neues Wissen über die Umgebung hinzulernen.

4.2 Die VDBE-Strategie

Der soeben beschriebene Ansatz wird in einem berechenbaren Modell umgesetzt, indem die ε -Greedy-Strategie zur „Value-Difference Based Explorati-

¹Beispielsweise durch Q -learning (Watkins und Dayan, 1992).

on“-Strategie (VDBE) erweitert wird (Tokic, 2010). Grundlegend wird hierbei jedem Zustand $s \in \mathcal{S}$ eine dynamische Explorationsrate zugewiesen:

$$\varepsilon(s) \in [0, 1] , \quad (4.1)$$

welche die Wahrscheinlichkeit für das Wählen einer zufälligen Aktion in Zustand s angibt. Die Explorationsrate ist somit nicht mehr ein globaler Parameter, sondern ein lokaler, zustandsabhängiger Wert. Dieses Vorgehen wird mit der angenommenen Markov-Eigenschaft der Umgebung begründet (vgl. Abschnitt 2.3), was zur effektiven Steuerung von Exploration und Exploitation beitragen soll. Beispielsweise ist es hierdurch möglich, nur in Bereichen des Zustandsraumes explorativ zu sein, in denen sich die Dynamik der Umgebung geändert hat oder das bislang erlernte Wissen noch mit Unsicherheit behaftet ist.

Nach dem Ausführen einer Aktion und dem darauffolgenden Lernschritt, z. B. durch Q -learning oder Sarsa, wird $\varepsilon(s)$ auf Basis des entstandenen Wertunterschiedes adaptiert. Hierfür werden die Q -Werte von Aktion a in Zustand s , vor dem Lernschritt $Q(s_t, a_t)$ sowie nach dem Lernschritt $Q^n(s_t, a_t)$, mittels einer Boltzmann-Verteilung aktiviert:

$$\begin{aligned} f(s, a, \sigma) &= \left| \frac{e^{\frac{Q(s,a)}{\sigma}}}{e^{\frac{Q(s,a)}{\sigma}} + e^{\frac{Q^n(s,a)}{\sigma}}} - \frac{e^{\frac{Q^n(s,a)}{\sigma}}}{e^{\frac{Q(s,a)}{\sigma}} + e^{\frac{Q^n(s,a)}{\sigma}}} \right| \\ &= \frac{1 - e^{\frac{-|Q^n(s,a) - Q(s,a)|}{\sigma}}}{1 + e^{\frac{-|Q^n(s,a) - Q(s,a)|}{\sigma}}} \\ &= \frac{1 - e^{\frac{-|\alpha \cdot \Delta|}{\sigma}}}{1 + e^{\frac{-|\alpha \cdot \Delta|}{\sigma}}} , \end{aligned} \quad (4.2)$$

wobei σ einen positiven Skalierungsparameter angibt, welcher als *inverse Empfindlichkeit* bezeichnet wird. Dieser muss in Abhängigkeit von der Höhe der zu erhaltenden Rewards gewählt werden, kann jedoch auch durch das in Kapitel 6 beschriebene Verfahren automatisch adaptiert werden. Den Einfluss dieses Parameter veranschaulicht Abbildung 4.1(a). Man erkennt dass niedrige Belegungen von σ die Aktivierung $f(s, a, \sigma)$ schnell gegen den Wert 1 streben lassen. Hingegen führen hohe Belegungen von σ dazu, dass exploratives Verhalten nur bei hohen Wertunterschieden entsteht. Sobald jedoch die Wertefunktion konvergiert, wodurch der TD-Fehler $\Delta \rightarrow 0$ geht, konvergiert ebenfalls die Aktivierung $f(s, a, \sigma) \rightarrow 0$.

Damit nicht nur der Wertunterschied der zuletzt getätigten Aktion Einfluss auf das Explorationsverhalten erhält, wird zusätzlich auch die Information des bisherigen Explorationsverhaltens mitberücksichtigt. Hierfür wird die bisherige Explorationsrate $\varepsilon(s)$ mit der Aktivierung $f(s_t, a_t, \sigma)$ kombiniert:

$$\varepsilon^n(s) \leftarrow \delta \cdot f(s_t, a_t, \sigma) + (1 - \delta) \cdot \varepsilon(s) . \quad (4.3)$$

Der Parameter $\delta \in [0, 1]$ steuert den Einfluss der zuletzt getätigten Aktion auf $\varepsilon(s)$. Hohe Belegungen von δ führen dazu, dass die zuletzt getätigte Aktion stärker in $\varepsilon(s)$ gewichtet wird. Niedrige Belegungen führen hingegen dazu, dass $\varepsilon(s)$ nur langsam an den Lernfortschritt angepasst wird. Dabei hat es sich in der Praxis bewährt, den Parameter δ mit dem Kehrwert der verfügbaren Aktionen in Zustand s zu belegen (Tokic, 2010; Tokic und Palm, 2011; Tokic u. a., 2012):

$$\delta = \frac{1}{\mathcal{A}(s)} . \quad (4.4)$$

Eine separate Speicherung von δ wird daher nicht benötigt, da dies zur Laufzeit anhand der endlichen Aktionsmenge $\mathcal{A}(s)$ erzeugt werden kann. Das Ergebnis in Gleichung (4.3) ist stets ein Wert im Intervall $[0, 1]$, wobei große Wertunterschiede $\varepsilon(s) \rightarrow 1$ streben lassen. Stimmt hingegen die Prädiktion mit der Observation überein, konvergiert $\varepsilon(s) \rightarrow 0$. Schlussendlich sei zur Veranschaulichung in Algorithmus 4 das Zusammenspiel zwischen Q -learning und der VDBE-Strategie exemplarisch aufgezeigt:

4.3 Alternative Aktivierungsfunktionen

Gleichung (4.2) ist selbstverständlich nur eines der möglichen Modelle, um aus dem Wertunterschied eine Explorationsrate im Intervall $[0, 1]$ zu erzeugen. Alternativ kann der Nenner in Gleichung (4.2) weggelassen werden, um eine rein *exponentielle Aktivierung* zu erhalten:

$$f(s, a, \sigma) = 1 - e^{\frac{-|\alpha \cdot \Delta|}{\sigma}} . \quad (4.5)$$

Das Ergebnis vom verbleibenden Zähler ist stets ein Wert im Intervall $[0, 1]$ (vgl. Abbildung 4.1(b)). Im Unterschied zu Gleichung (4.2), ist die exponentielle Aktivierung, bei selbigem Parameter σ , lediglich etwas empfindlicher für kleine Wertunterschiede $|\alpha \cdot \Delta|$.

Algorithm 4 Q -LEARNING WITH VDBE

```

1: Initialize  $Q$  arbitrarily, e.g.  $Q(s, a) = 0$  for all  $s, a$ 
2: for each episode do
3:   Initialize start state  $s$ 
4:   repeat
5:      $\xi \leftarrow \text{rand}(0..1)$ 
6:     if  $\xi < \varepsilon(s)$  then
7:        $a \leftarrow \text{rand}(\mathcal{A}(s))$ 
8:     else
9:        $a \leftarrow \text{argmax}_{b \in \mathcal{A}(s)} Q(s, b)$ 
10:    end if
11:    take action  $a$ , observe reward  $r$  and successor state  $s'$ 
12:     $a^* \leftarrow \text{argmax}_{b \in \mathcal{A}(s')} Q(s', b)$ 
13:     $\Delta \leftarrow r + \gamma Q(s', a^*) - Q(s, a)$ 
14:     $Q(s, a) \leftarrow Q(s, a) + \alpha \Delta$ 
15:     $\varepsilon(s) \leftarrow \delta \cdot \frac{1 - e^{-\frac{|\alpha \cdot \Delta|}{\sigma}}}{1 + e^{-\frac{|\alpha \cdot \Delta|}{\sigma}}} + (1 - \delta) \cdot \varepsilon(s)$ 
16:     $s \leftarrow s'$ 
17:  until  $s$  is terminal state
18: end for

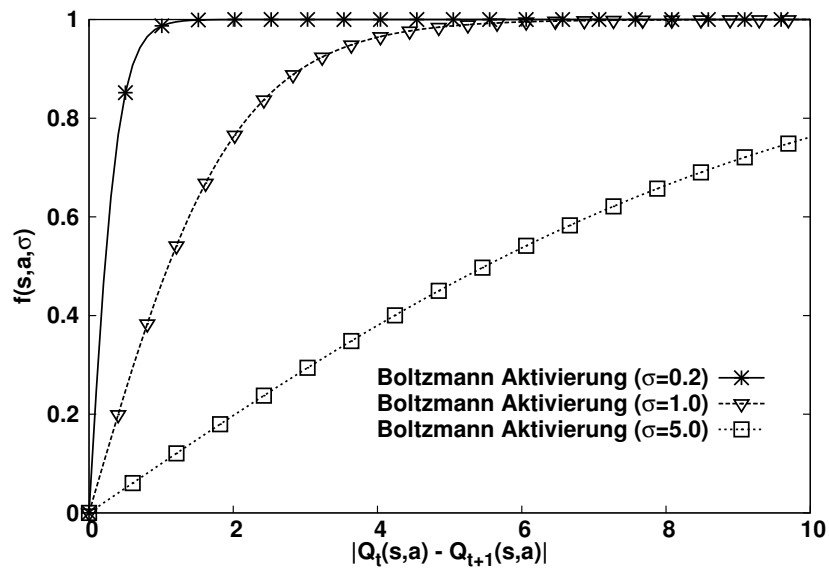
```

Anstelle einer exponentiellen Aktivierung (Gleichungen 4.2+4.5), wäre eine weitere Möglichkeit eine *hyperbolische Aktivierung*:

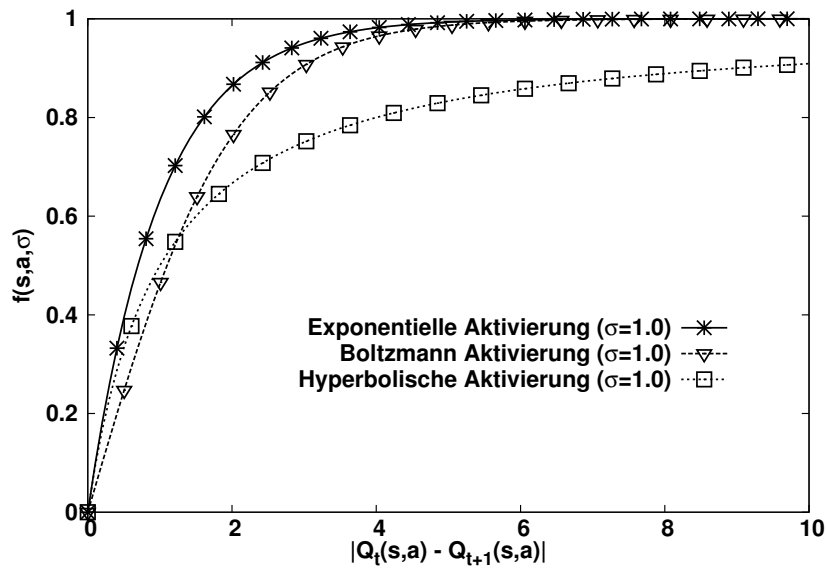
$$f(s, a, \sigma) = 1 - \frac{1}{1 + \frac{|\alpha \Delta|}{\sigma}} . \quad (4.6)$$

Im Unterschied zur exponentiellen Aktivierung, benötigt die Hyperbolische höhere Wertunterschiede, um den selben Wert der Aktivierung $f(s, a, \sigma)$ zu erreichen (vgl. Abbildung 4.1(b))². Hingegen verhält sich dieses Modell für kleinere Wertunterschiede eher wie eine exponentielle Aktivierung.

²vgl. http://en.wikipedia.org/wiki/Hyperbolic_discounting in Analogie zur exponentiellen bzw. hyperbolischen Reward-Diskontierung.



(a)



(b)

Abbildung 4.1: Vergleich verschiedener Aktivierungsfunktionen $f(s,a,\sigma)$ einer VDBE-Strategie: (a) zeigt das Verhalten einer Boltzmann-Aktivierung für unterschiedliche Belegungen der inversen Empfindlichkeit σ ; (b) zeigt hingegen das Verhalten für $\sigma = 1$ für die Boltzmann-Aktivierung (Gleichung 4.2), die exponentielle Aktivierung (Gleichung 4.5) sowie die hyperbolischen Aktivierung (Gleichung 4.6).

4.4 Konvergenzanalyse

Aus den Gleichungen (4.2)+(4.3) ist ersichtlich, dass das Explorationsverhalten von fünf Faktoren abhängig ist:

1. der Lernrate δ für $\varepsilon(s)$,
2. der inversen Empfindlichkeit σ ,
3. dem TD-Fehler Δ ,
4. der Lernrate α für Q -learning bzw. Sarsa,
5. sowie des Diskontierungsfaktors γ .

Da die ersten beiden Einflussfaktoren schon beim Vorstellen der VDBE-Strategien diskutiert wurden, beleuchtet der nun folgende Abschnitt den Einfluss von Δ , α und γ . Da diese Parameter vom Lernverfahren der Wertefunktion entstammen, hat dies zur Folge, dass die Konvergenz von VDBE-Strategien auch von der Konvergenz des verwendeten Lernverfahrens für die Q -Funktion abhängig ist, ebenso aber auch von der Repräsentation der zu erlernenden Wertefunktion. Im Folgenden wird daher kurz auf das Lernen in großen Zustandsräumen eingegangen, und anschließend auf das von TD-Lernverfahren in deterministischen und stochastischen Umgebungen.

4.4.1 Große Zustandsräume

In der Praxis versucht man das Komplexitätsproblem von großen Zustandsräumen dadurch in den Griff zu bekommen, indem Wertefunktionen nicht durch Tabellen approximiert werden, sondern als parametrisierte Funktion, z. B. anhand neuronaler Netzwerke (Boyan und Moore, 1995). Hierdurch erhofft man sich, dass durch Generalisierung erlerntes Wissen indirekt auch auf Nachbarzustände übertragen wird, was zu einer Reduktion der Lernzeit beitragen kann. Ebenso benötigt eine parametrisierte Repräsentation i. d. R. weniger Speicherplatz, da z. B. nur die Gewichte eines neuronalen Netzwerkes gespeichert werden müssen, jedoch nicht die Tabelle aller Aktionswerte. Ein häufig auftretendes Problem ist hierbei, dass die Approximation in lokalen Minima enden kann und demzufolge nicht exakt die tatsächlich optimale Strategie π^* von der approximierten Q -Funktion abgeleitet wird. Zusätzlich existiert bei neuronalen Netzwerken

das Problem, dass lokale Änderungen unerwünschte, globale Auswirkungen haben können, was wiederum zu Fluktuationen in der Wertefunktion führen kann. Diese Aspekte führen bei VDBE-Strategien zu einem evtl. ungewollten Anstieg der Explorationsrate.

Da das Lernen mit Funktionsapproximatoren mittlerweile zu einem eigenständigen Forschungsbereich angewachsen ist, sei an dieser Stelle auf grundlegende Arbeiten verwiesen, die sich mit diesem Thema tiefergehend beschäftigen (Boyan und Moore, 1995; Gordon, 1995; Precup u. a., 2001). Im Reinforcement Learning wird oftmals die Wertefunktion durch neuronale Netzwerke repräsentiert (Faußer und Schwenker, 2008; Flentge, 2005; Nissen, 2007; Riedmiller, 2005) oder Tile-Coding verwendet (Sutton, 1996; Timmer und Riedmiller, 2007). Bei letzterem wird anhand einer Diskretisierung versucht, dem Problem von ungewollten globalen Änderungen, die durch lokale Updates entstehen, entgegenzuwirken.

4.4.2 Konvergenz in deterministischen Umgebungen

Für Q -learning kann bewiesen werden, dass das Verfahren unter den folgenden drei Annahmen zur optimalen Strategie π^* konvergiert (Mitchell, 1997; Watkins, 1989; Watkins und Dayan, 1992):

1. Bei der Umgebung muss es sich um einen deterministischen MDP handeln (Stochastik wird in Abschnitt 4.4.3 betrachtet).
2. Die unmittelbaren Rewards müssen beschränkt sein, d. h. es muss eine positive Konstante c existieren, für die gilt: $(\forall s, a) |\mathcal{R}_{s,s'}^a| \leq c$.
3. Alle Zustandsübergänge werden unendlich oft durchlaufen.

Während die ersten beiden Annahmen noch relativ einfach erfüllt werden können, ist die dritte Annahme oftmals schwer zu erfüllen, insbesondere für Zustandsräume mit kontinuierlichen Zustandsdimensionen (vgl. Abschnitt 4.4.1).

Die Konvergenz für Q -learning wird dadurch bewiesen, indem man zeigt, dass der geschätzte Aktionswert mit dem größten Fehler, durch eine Neuschätzung diesen um mindestens einen Faktor $0 \leq \gamma < 1$ reduziert (Mitchell, 1997, S. 377-379). Der tatsächliche (wahre) Aktionswert wird als $Q^*(s, a)$ bezeichnet, zu welchem $Q(s, a)$ konvergiert. Q_j ist die Tabelle der geschätzten Aktionswerte nach j Aktualisierungen, welche zu Beginn mit

beliebigen, endlichen Werten initialisiert wird. In diesem Sinne gibt ϵ_j den maximalen Fehler in Q_j an:

$$\epsilon_j = \max_{s,a} |Q_j(s, a) - Q^*(s, a)| . \quad (4.7)$$

Für jeden Aktionswert $Q_j(s, a)$, der zur Iteration $j + 1$ neu geschätzt wird, ist die Höhe des Fehlers im neu geschätzten Aktionswert $Q_{j+1}(s, a)$:

$$\begin{aligned} |Q_{j+1}(s, a) - Q^*(s, a)| &= |(r + \gamma \max_{a'} Q_j(s', a')) - (r + \gamma \max_{a'} Q^*(s', a'))| \\ &= \gamma |\max_{a'} Q_j(s', a') - \max_{a'} Q^*(s', a')| \\ &\leq \gamma \max_{a'} |Q_j(s', a') - Q^*(s', a')| \\ &\leq \gamma \max_{s'', a'} |Q_j(s'', a') - Q^*(s'', a')| \\ |Q_{j+1}(s, a) - Q^*(s, a)| &\leq \gamma \epsilon_j . \end{aligned} \quad (4.8)$$

Die dritte Zeile folgt aus der zweiten Zeile, da für beliebige Funktionen f_1 und f_2 folgende Ungleichung gilt:

$$|\max_a f_1(a) - \max_a f_2(a)| \leq \max_a |f_1(a) - f_2(a)| . \quad (4.9)$$

In der vierten Zeile wurde eine neue Zustandsvariable s'' eingeführt, über welche die Maximierung stattfindet und für beliebige Zustände gilt. Der resultierende Ausdruck führt zur Definition des Fehlers ϵ_j nach Gleichung (4.7). Demnach ist der neu geschätzte Aktionswert $Q_{j+1}(s, a)$ für beliebige (s, a) höchstens γ mal den maximalen Fehler in Q_j , ϵ_j . Der größte Fehler in den Initialwerten ($\epsilon_{j=0}$) ist begrenzt, da die Aktionswerte von $Q_{j=0}(s, a)$ und $Q^*(s, a)$ ebenfalls begrenzt sind für beliebige s und a . Nach dem ersten Lernintervall, in welchem jedes Zustandsaktionspaar (s, a) besucht wird, ist der größte Fehler in der Tabelle höchstens $\gamma \epsilon_{j=0}$. Nach k Lernintervallen ist der Fehler höchstens $\gamma^k \cdot \epsilon_{j=0}$. Da jedes Zustandsaktionspaar unendlich oft besucht wird (Annahme 3), ist die Anzahl an Lernintervallen ebenfalls unendlich, wodurch $\epsilon_{j=0} \rightarrow 0$ geht für $j \rightarrow \infty$, und folglich Q_j gegen Q^* konvergiert. Für VDBE-Strategien in Kombination mit Q -learning bedeutet dies, dass der TD-Fehler Δ mit zunehmender Anzahl an Interaktionsschritten immer kleiner wird. Hieraus woraus resultiert, dass die Explorationsrate $\varepsilon(s) \rightarrow 0$ konvergiert.

Eine Konvergenzgarantie für Sarsa kann nicht direkt gegeben werden, sondern nur indirekt über die Strategie (Sutton und Barto, 1998). Der Hintergrund ist, dass Sarsa nicht die Aktion mit dem höchsten Q -Wert im Fol-

gezustand s' verwendet, sondern den Q -Wert der tatsächlich gewählten Aktion a_{t+1} im Folgezustand (vgl. Gleichungen 2.26+2.27). Bei stochastischen Strategien gibt es demzufolge keine Konvergenzgarantie, da Sarsa den Q -Wert der getätigten Aktion in Zustand s , auf den Q -Wert der getätigten Aktion im Folgezustand s' zu adaptieren strebt. Mit entsprechend hoch eingestellter Explorationsrate kann s' demnach stark variieren, mit der Folge, dass für VDBE die Explorationsrate nicht gegen $\varepsilon(s) \rightarrow 0$ konvergiert. Stattdessen ist die Höhe von $\varepsilon(s)$ von der Höhe des Wertunterschieds abhängig, ebenso aber auch von der gewählten inversen Sensibilität σ (Tokic und Palm, 2011; Tokic u. a., 2012). Falls die Strategie jedoch über die Zeit Greedy (Gleichung 3.2) und folglich deterministisch wird, so konvergiert auch die Q -Funktion von Sarsa unter den gleichen Bedingungen wie für Q -learning. Dies ist beispielsweise für die decreasing- ε -Strategie der Fall, bei welcher mit einer hohen Explorationsrate $\varepsilon \sim 1$ gestartet wird, welche über die Zeit gegen $\varepsilon \rightarrow 0$ konvergiert.

Somit könnte man sich natürlich fragen, welche Daseinsberechtigung Sarsa besitzt? Der Hintergrund ist, dass über dieses Lernverfahren die Stochastik einer Explorationsstrategie mitgelernt wird, da negative Kosten, aufgrund von Explorationsaktionen im Folgezustand, entstehen können. Dieses Wissen wird durch Sarsa auch an die vorhergehenden Aktionen zurückpropagiert, woraus resultiert, dass unter Umständen nicht die optimale Strategie π^* , mit den geringsten Wegekosten, gelernt wird. Anstelle dessen werden *sichere* Strategien gelernt, die Zustände meiden, in denen aufgrund von Explorationsaktionen hohe Kosten entstehen können (Sutton und Barto, 1998). In Abschnitt 5.3 wird dieses Verhalten anhand des Cliff-Walking-Problems verdeutlicht.

4.4.3 Konvergenz in stochastischen Umgebungen

Die Lernrate α ist ein wichtiger Einflussfaktor für die Konvergenz der Q -Funktion (George und Powell, 2006; Powell, 2007). Während große Lernraten ein schnelles Adaptieren an den TD-Fehler Δ ermöglichen, kann solch eine Belegung bei stochastischen Rewards zur Folge haben, dass der Q -Wert um den wahren Q^* -Wert oszilliert, anstatt zu konvergieren. Versucht man dieses Problem mit niedrigen Lernraten zu lösen, so hat dies zur Folge, dass der Lernprozess insgesamt langsamer von staten geht, dafür aber im Limit die Q -Funktion genauer approximiert wird.

Bildhaft kann man dies durch folgendes Minimalbeispiel demonstrieren, in welchem gezeigt wird, welche Auswirkung das Wählen der Lernrate

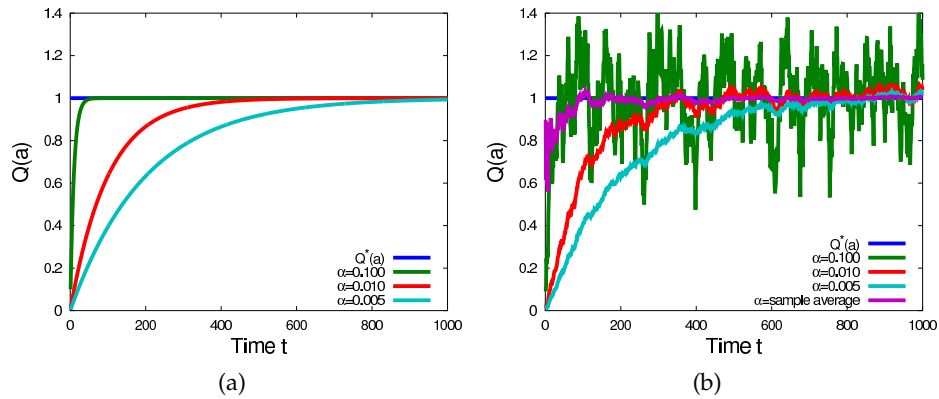


Abbildung 4.2: Vergleich des Lernverhaltens in Abhängigkeit unterschiedlicher Lernraten: (a) für deterministische Rewards und (b) für stochastische Rewards.

α auf den geschätzten Q -Wert während des Lernens besitzt. Das Ziel ist, den wahren Q -Wert einer einzelnen Aktion a , anhand einer Sequenz von mehreren Rewards, zu erlernen. Für Q -learning bzw. Sarsa wird keine Diskontierung benötigt ($\gamma = 0$), da kein Weitblick über zukünftige Zustände stattfinden muss. Hierdurch entsteht die vereinfachte Lernregel zur inkrementellen Berechnung des Aktionswertes (Robbins und Monro, 1951):

$$Q^n(a) \leftarrow Q(a) + \alpha(r_{t+1} - Q(a)) \quad . \quad (4.10)$$

Im Folgenden wird diese Lernregel mit deterministischen und stochastischen Rewards untersucht. Der Reward ist im deterministischen Fall nach jeder Aktion durch $r_{t+1} = Q^*(a)$ definiert, also spricht der wahre Q -Wert der gewählten Aktion a . Im stochastischen Fall wird hingegen der Reward normalverteilt, mit Mittelwert $Q^*(a)$ und Standardabweichung 0.8, bestimmt:

$$r_{t+1} \sim \mathcal{N}(Q^*(a), 0.8) \quad . \quad (4.11)$$

Als Ergebnis erhält man das in Abbildung 4.2 dargestellte Lernverhalten für konstante Lernraten, wobei $Q^*(a)$ konstant durch den Wert 1 belegt wurde.

Abbildung 4.2(a) zeigt, dass für deterministische Rewards $Q(a) \rightarrow Q^*(a)$ konvergiert, wobei die Konvergenzgeschwindigkeit von der gewählten Lernrate α abhängig ist. Im Fall von stochastischen Rewards geht die Schätzung $Q(a)$ zwar in Richtung $Q^*(a)$, konvergiert jedoch nicht, sondern os-

zilliert um $Q^*(a)$. Je höher hierbei die Lernrate gewählt wird, desto größer wird die Amplitude in Abhängigkeit der Stochastik.

Möchte man auch eine Konvergenz für stochastische Rewards erzielen, so ist dies über das sogenannte *Incremental-Sample-Average*-Verfahren möglich (Sutton und Barto, 1998). Dieses setzt voraus, dass Reward-Schwankungen stationär sind, also ein zeitlich unabhängiger, konstanter Mittelwert existiert. Mittels einer aktionsabhängigen Lernrate $\alpha(a)$ wird inkrementell über alle bislang erhaltenen Rewards für Aktion a gemittelt, wodurch das Rauschen geglättet wird:

$$\alpha(a) = \frac{1}{1 + k(a)} . \quad (4.12)$$

Der Aktionszähler $k(a)$, welcher zu Beginn des Lernprozesses für alle Aktionen mit $k(a) = 0$ initialisiert wird, gibt an, wie oft Aktion a bereits gewählt wurde. Wie Abbildung 4.2(b) zu entnehmen ist, stellt dieses Verfahren eine gute Möglichkeit dar, eine Konvergenz ebenfalls für stationäre Reward-Verteilung zu erzielen. Für nichtstationäre Reward-Verteilungen kann hingegen oftmals nur eine experimentell bestimmte, konstante Belegung von α gewählt werden.

Um theoretisch die Konvergenz garantieren zu können, gibt es zwei notwendige Bedingungen, damit die *stochastische Approximation* $Q(a)$ mit Wahrscheinlichkeit 1 konvergiert (Robbins und Monro, 1951; Sutton und Barto, 1998; Watkins und Dayan, 1992):

$$\sum_{k=1}^{\infty} \alpha_k(a) = \infty \quad \text{sowie} \quad \sum_{k=1}^{\infty} \alpha_k^2(a) < \infty . \quad (4.13)$$

Die erste Bedingung steht dafür, dass Lernschritte groß genug sein müssen, um eventuelle Startbedingungen oder zufällige Schwankungen zu überwinden. Die zweite Bedingung hingegen dafür, dass Lernschritte über die Zeit kleiner werden müssen, um schlussendlich auch die Konvergenz zu erreichen. Beide Bedingungen sind für das Incremental-Sample-Average-Verfahren erfüllt, jedoch nicht für konstante Belegungen der Lernrate α (Verletzung der zweiten Bedingung). Zusammengefasst bedeutet dies für VDBE-Strategien, dass Stochastik nicht unbedingt ein Problem darstellt, solange die Lernrate α entsprechend adaptiert wird. Für weiterführende Arbeiten bezüglich der Steuerung von α , sei jedoch an dieser Stelle auf die Arbeiten von George und Powell (2006) sowie Powell (2007) verwiesen.

4.5 Das n -armige Banditenproblem

Ein typisches Lernproblem zur Untersuchung von Exploration und Exploitation bietet das n -armige Banditenproblem (Robbins, 1952; Sutton und Barto, 1998; Tokic, 2010). Dieses modelliert eine Verallgemeinerung eines einarmigen Glücksspielautomaten, in Analogie zum einführenden Bettler-Beispiel aus Kapitel 3. Der Agent kann zu jedem Zeitschritt eine von n möglichen Aktionen wählen (einer der n Arme), $a \in \{a_1, \dots, a_n\}$, und erhält einen stochastischen Reward in Abhängigkeit einer aktionsabhängigen Wahrscheinlichkeitsverteilung³. Alle Reward-Verteilungen sind dem Agenten gegenüber unbekannt. Diese müssen folglich geschätzt werden, um eine zielgerichtete Aktionswahl zu ermöglichen. Dabei muss der Agent zu jedem Zeitpunkt entscheiden, ob er basierend auf den bislang erlernten Mittelwert-Schätzungen: (1) eine Exploitationsaktion wählt, die einen hohen Reward einbringt, oder (2) eine Explorationsaktion wählt, welche die Schätzung über die assoziierte Reward-Verteilung verbessert. Real-Welt-Anwendungen, die dem Modell eines n -armigen Banditen ähnlich sind, finden sich in folgenden Beispielen wieder:

- Adaptives Routing in Netzwerken, mit dem Ziel der Wartezeitverkürzung (Awerbuch und Kleinberg, 2004).
- Behandlungsdiagnose bei klinischen Untersuchungen, um Patientensterben zu minimieren (Hardwick und Stout, 1991).
- Das wirtschaftliche Problem der Auswahl eines Lieferanten, auf Basis von unvollständigen Informationen (Azoulay-Schwartz u. a., 2004).

4.5.1 Untersuchung

Das Experiment basiert auf der Testumgebung eines 10-armigen Banditen ($n = |\mathcal{A}| = 10$), welcher von Sutton und Barto (1998) sowie Tokic (2010) verwendet wurde. Zum Erlernen der Q -Werte wird Gleichung (4.10) verwendet⁴, wobei die aktionsabhängigen Lernraten $\alpha(a)$ durch das Incremental-Sample-Average-Verfahren aus Gleichung (4.12) adaptiert werden. Die stationären Reward-Verteilungen besitzen den Mittelwert $Q^*(a)$ mit Standardabweichung 1. Zu Beginn einer Lernepisode werden die Mittelwerte

³Aufgrund dass es bei diesem Lernproblem nur einen Zustand gibt, wird zur vereinfachten Schreibweise die Notation über Zustände weggelassen.

⁴Entspricht Q -learning mit Diskontierungsfaktor $\gamma = 0$.

zufällig, auf Basis einer Normalverteilung mit Mittelwert 0 und Standardabweichung 1, initialisiert:

$$\forall a : Q^*(a) \sim \mathcal{N}(0, 1) . \quad (4.14)$$

Alle Schätzungen, über die Mittelwerte der Reward-Verteilungen, werden zu Beginn einer Lernepisode mit $Q(a) = 0$ initialisiert. Während einer Lernepisode kann der Agent seine Aktionsauswahl innerhalb von $T = 1000$ Schritten verbessern, um möglichst viel Reward zu erhalten. Schlussendlich werden die Ergebnisse über 2000 zufällig erzeugte Banditen gemittelt.

Die Explorationsrate der ε -Greedy-Strategie wird für verschiedene Werte im Intervall $\varepsilon \in [0, 1]$ untersucht. Die Temperatur der Softmax-Strategie hingegen für verschiedene Werte im Intervall $\tau \in [0.04, 25]$, da innerhalb diesem gleich gute sowie gleich schlechte Ergebnisse erzielt werden können, wie unter Verwendung einer ε -Greedy-Strategie. Das Intervall des Parameters τ wird dabei ebenso für die inverse Sensibilität σ der VDBE-Strategie verwendet. Die Lernrate der zustandsabhängigen Explorationsrate (VDBE) ist konstant mit dem Wert $\delta = \frac{1}{|\mathcal{A}(s)|} = \frac{1}{10} = 0.1$ belegt, was der inversen Anzahl an Aktionen entspricht.

4.5.2 Ergebnisse

Die Ergebnisse des Experimentes sind in Abbildung 4.3 und Tabelle 4.1 aufgezeigt (Tokic, 2010). Der Vergleich von ε -Greedy und Softmax zeigt auf, dass beide Strategien im untersuchten Parameterintervall gleich gut sowie gleich schlecht abschneiden können. Die Kurven mit niedrigem Reward erklären sich dadurch, dass eine konstant hohe Explorationsrate zu konstant hoher Exploration führt, was analog für hohe Temperaturen τ bei Softmax gilt. Die Ergebnisse entsprechen in solch einem Fall denen einer *Random-Exploration*, bei welcher ein mittlerer Reward in Höhe von 0 erhalten wird. Dies resultiert aus dem Mittelwert der Normalverteilung, anhand welcher die Mittelwerte der n Reward-Verteilungen initialisiert wurden.

Den Abbildungen ist ebenso zu entnehmen, dass hohe Parameterbelegungen bei ε -Greedy und Softmax den Reward konvergieren lassen, jedoch diesen nicht weiter verbessern. Hingegen verbessern niedrige Parameterbelegungen das Ergebnis in der Zukunft (wenn auch sehr langsam), sobald die Anzahl von Zeitschritten gegen unendlich geht und stets ein paar Explorationsaktionen ausgeführt werden. Falls keine expliziten Explorationsaktionen ausgeführt werden (Greedy-Strategie), konvergiert die Werte-

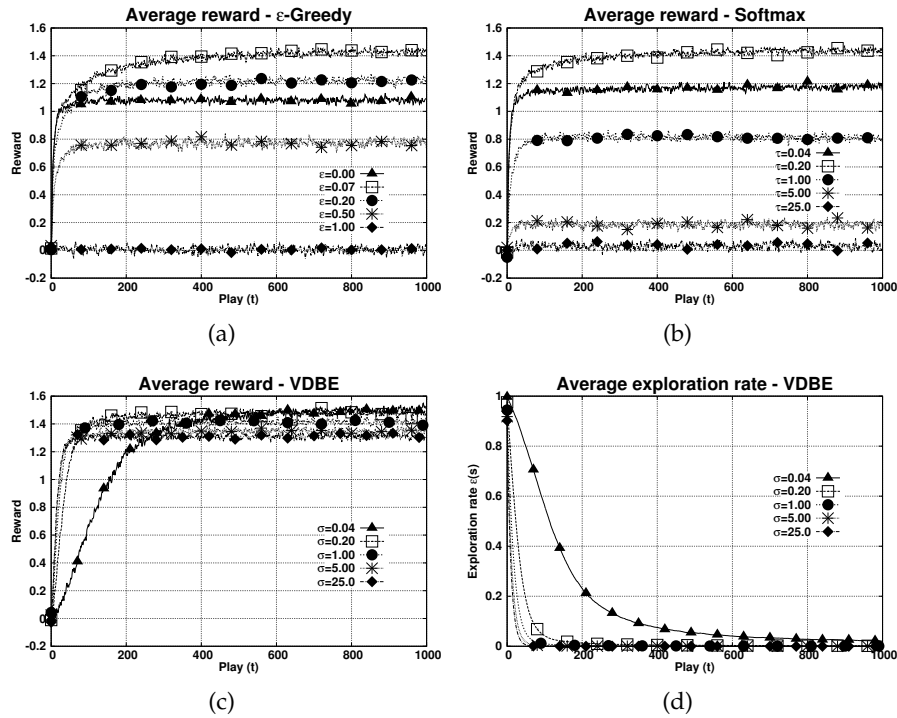


Abbildung 4.3: Ergebnisse des n -armiges Banditen-Problems (Tokic, 2010). Vergleich des mittleren Rewards pro Aktion für (a) ε -Greedy, (b) Softmax und (c) VDBE. Abbildung (d) zeigt für VDBE die Explorationsrate $\varepsilon(s)$, in Abhängigkeit der inversen Sensibilität σ .

Strategie	\bar{r}	r_{1000}^{\min}	r_{1000}^{\max}
ε -Greedy	1.35 ($\varepsilon = 0.07$)	0.00 ($\varepsilon = 1.00$)	1.43 ($\varepsilon = 0.07$)
Softmax	1.38 ($\tau = 0.20$)	0.00 ($\tau = 25.0$)	1.44 ($\tau = 0.20$)
VDBE	1.42 ($\sigma = 0.33$)	1.30 ($\sigma = 25.0$)	1.50 ($\sigma = 0.04$)

Tabelle 4.1: Ergebnisse des n -armiges Banditen-Problems (Tokic, 2010). Vergleich des Rewards pro Zeitschritt: mit optimaler Parameterbelegung (\bar{r}), sowie minimaler und maximaler Reward zum Zeitpunkt $t = 1000$ (r_{1000}^{\min} bzw. r_{1000}^{\max}). Zahlen in Klammern geben die Parameterbelegung an, durch welche das Ergebnis erzielt wird.

funktion schnell in einem lokalen Minimum, wie der Kurve von $\varepsilon = 0$ zu entnehmen ist.

Der Vorteil einer adaptiven Explorationssteuerung ist den Abbildungen der VDBE-Strategie zu entnehmen. Verglichen mit dem identischen Para-

meterintervall von Softmax, ist der Dynamikumfang der Ergebnisse zum Zeitpunkt $t = 1000$ signifikant kleiner, was auf eine Robustheit des Parameters σ hindeutet. Darüber hinaus befinden sich die Ergebnisse im oberen Rewardbereich, wobei zum Zeitpunkt $t = 1000$ die Belegung $\sigma = 0.04$ den meisten Reward einbringt. Abbildung 4.3(d) ist außerdem zu entnehmen, dass die Explorationsrate für $t \rightarrow \infty$ gegen $\varepsilon(s) \rightarrow 0$ konvergiert, was dazu führt, dass die Strategie Greedy wird. Die Ursache hierfür ist, dass die Q -Funktion aufgrund des Incremental-Sample-Average-Verfahrens konvergiert.

4.6 Die VDBE-Softmax-Strategie

Die vorgestellte VDBE-Strategie konnte erfolgreich am n -armigen Banditen-Problem eingesetzt werden (Tokic, 2010) sowie in einem Framework für Self-Aware Computing (Hoffmann u. a., 2011). Bei dieser Strategie gibt es jedoch zwei Aspekte, die verbessert werden können:

1. Explorationsaktionen werden gleichverteilt über alle möglichen Aktionen in Zustand s gewählt (zurückzuführen auf die ε -Greedy-Strategie). Dies kann zu einem vergleichsweise niedrigen Reward führen, falls nur sehr wenige Aktionen mit hohem Reward existieren. Obwohl dieses Wissen in Form der bislang geschätzten Q -Werte vorliegt, würde dieses bei einer gleichverteilten Aktionsauswahl nicht mitberücksichtigt werden.
2. Für stochastische Rewards oder Folgezustände, können bei einer konstanten Wahl der Lernrate α , Fluktuationen in der Q -Funktion entstehen. Dies gilt ebenso für on-policy Lernverfahren, wie beispielsweise Sarsa, falls die Aktionsauswahl einer hohen Stochastik unterliegt. Als Folge dessen konvergiert die Aktivierungsfunktion $f(s, a, \sigma)$ nicht gegen 0, sondern erzeugt eine Explorationsrate von $\varepsilon(s) > 0$.

Der erste Aspekt kann dadurch entschärft werden, indem die Wahl von Explorationsaktionen nicht mehr gleichverteilt über alle möglichen Aktionen stattfindet, sondern gewichtet in Abhängigkeit der Q -Werte. Hierbei sollen Aktionen mit hohem Q -Wert bevorzugt gewählt werden, hingegen Aktionen mit niedrigem Q -Wert gemieden werden. Ein Verfahren das diese Idee umsetzt wurde von Wiering (1999) unter dem Namen *Max-Boltzmann*-

Exploration (MBE) vorgeschlagen, welches für die Auswahl von Explorationsaktionen die Softmax-Strategie verwendet:

$$\pi(s, a) = \begin{cases} \text{Softmax-Strategie nach Gleichung (3.5),} & \text{falls } \xi \leq \varepsilon \\ \text{Greedy-Strategie nach Gleichung (3.2),} & \text{ansonsten} \end{cases} \quad (4.15)$$

Hierbei ist ξ eine gleichverteilte Zufallszahl aus dem Intervall $[0, 1]$. Obwohl die *Max-Boltzmann-Exploration-Strategie* zwei zu konfigurierende Parameter benötigt (ε sowie τ), werden die Vorteile beider Verfahren miteinander kombiniert (Wiering, 1999).

Die Idee von MBE wird nun dazu verwendet, die VDBE-Strategie zur sogenannten VDBE-Softmax-Strategie zu erweitern (Tokic und Palm, 2011). Im Unterschied zu MBE, wird die zustandsabhängige Explorationsrate $\varepsilon(s)$ nach VDBE adaptiert, hingegen für $\xi < \varepsilon(s)$ Explorationsaktionen nach Softmax gewählt. Konvergiert die Wertefunktion, so werden analog zu VDBE ausschließlich gierig Aktionen mit dem höchsten Q -Wert gewählt, während hingegen bei Unsicherheiten die Aktionswahl durch Softmax stattfindet.

Zusätzlich wird vorgeschlagen die Q -Werte in ein festes Intervall zu normalisieren, z. B. $[-1, 1]$, wodurch der Temperaturparameter von Softmax intuitiver gewählt werden kann, z. B. $\tau = 1$, (Tokic und Palm, 2011). Somit erhält man mit VDBE-Softmax eine dynamische Strategie, die im Falle von voller Exploration, was fälschlicherweise durch Rauschen in der Wertefunktion verursacht werden kann, die Softmax-Strategie verwendet, um das Auswählen von gelernten schlechten Aktionen zu unterdrücken. Im Gegensatz dazu werden gierig Aktionen auf Basis der Wertefunktion gewählt, falls diese konvergiert. In den folgenden Experimenten wird gezeigt, dass durch diese Vorgehensweise das Wählen von schlechten Explorationsaktionen erfolgreich unterdrückt werden kann.

4.7 Das Bandit-World-Problem

Im Folgenden wird das Erlernen einer Strategie für das Bandit-World-Problem untersucht, welches in Abbildung 4.4 dargestellt ist (Tokic und Palm, 2011). Hierbei handelt es sich um ein vollständig beobachtbares Lernproblem, bestehend aus mehreren Zuständen, wobei das Reward-Modell stochastische und deterministische Komponenten besitzt.

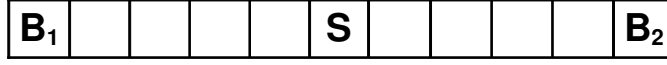


Abbildung 4.4: Bandit-World-Problem: Skizze nach Tokic und Palm (2011).

4.7.1 Untersuchung

In der Bandit-World existieren zwei Bandit-Zustände, \mathbf{B}_1 und \mathbf{B}_2 , in denen der Agent einen mehrarmigen Banditen betätigen kann. Hierbei gibt es einen guten sowie einen schlechten Banditen, wobei die Reward-Mittelwerte durch $Q^*(\mathbf{B}_1) = \{-1.0, 0.0, 1.0\}$ und $Q^*(\mathbf{B}_2) = \{2.0, 3.0, 4.0\}$ definiert sind. In einem Bandit-Zustand ist zusätzlich eine vierte Aktion erlaubt (a_{left} oder a_{right}), durch welche der Bandit verlassen werden kann und zu einem deterministischen Reward in Höhe von $r_{t+1} = -1$ führt. Dieser wird ebenso für getätigte Aktionen in allen anderen Zuständen erhalten, die keine Banditen-Zustände sind, $\mathcal{S} \setminus \{\mathbf{B}_1, \mathbf{B}_2\}$. Der Reward für getätigte Banditen-Aktionen ist normalverteilt mit Standardabweichung 1, in Abhängigkeit des betätigten Armes a_{lever} in Zustand s_{bandit} :

$$r_{t+1}(s_{\text{bandit}}, a_{\text{lever}}) \sim \mathcal{N}(Q^*(s_{\text{bandit}}, a_{\text{lever}}), 1) . \quad (4.16)$$

Gemessen wird der Return über 500 Lernepisoden, wobei die Ergebnisse über 500 Experimente gemittelt werden. Jede Episode beginnt in der Mitte der Bandit-World in Zustand \mathbf{S} , welche spätestens nach $T = 100$ Aktionen terminiert. Die untersuchten Parameterintervalle sind: ε -Greedy mit $\varepsilon \in [0, 1]$, Softmax mit $\tau \in [0.01, 100.0]$, VDBE und VDBE-Softmax mit $\sigma \in [0.01, 100.0]$ sowie $\delta(s) = 1/\mathcal{A}(s)$. Für VDBE-Softmax werden alle Q -Werte eines Zustandes in das Intervall $[-1, 1]$ normalisiert und der Parameter τ mit dem Wert $\tau = 1$ belegt. Aufgrund von stochastischen Rewards wird eine konstante Lernrate in Höhe von $\alpha = 0.1$ verwendet.

4.7.2 Ergebnisse

Die Ergebnisse des Experimentes sind in Abbildung A.4 und Tabelle 4.2 dargestellt. Für ε -Greedy und Softmax spielt die Verwendung von off- und on-policy Lernverfahren offenbar keine Rolle. Für beide Strategien erhält man im Worst Case (volle Exploration) einen Return in Höhe von $R_{\min} \approx -50$. Im Gegensatz dazu konvergieren beide VDBE-Strategien mit Q -learning zu einem signifikant höheren Return. Für Sarsa in Kombination mit VDBE erkennt man, dass Fluktuationen in der Wertefunktion, auf-

grund von stochastischen Rewards, ebenso zur Divergenz führen können, falls die inverse Sensibilität σ zu niedrig gewählt wird. Im Gegensatz dazu erkennt man die Robustheit der VDBE-Softmax-Strategie. Für beide Lernverfahren ist ein signifikant höher Worst-Case-Return zu entnehmen, mit fast identischen Ergebnissen der evaluierten Explorationsparameter.

Return in Episode 100				
	Q -learning	(Opt. %)	Sarsa	(Opt. %)
Greedy	315,81	85,42	315,18	85,27
Best Case ε -Greedy	341,05	91,30	335,18	89,93
Best Case Softmax	345,61	92,36	351,85	93,82
Best Case VDBE	364,79	96,83	353,51	94,20
Best Case VDBE-Softmax	371,82	98,47	371,36	98,37
Worst Case ε -Greedy	-44,91	1,33	-48,38	0,52
Worst Case Softmax	-39,82	2,52	-36,95	3,19
Worst Case VDBE	89,73	32,72	-30,43	4,71
Worst Case VDBE-Softmax	261,45	72,74	252,25	70,60

Return in Episode 500				
	Q -learning	(Opt. %)	Sarsa	(Opt. %)
Greedy	315,79	85,41	315,10	85,25
Best Case ε -Greedy	375,02	99,22	374,72	99,15
Best Case Softmax	346,89	92,66	351,82	93,81
Best Case VDBE	377,12	99,71	378,37	100,00
Best Case VDBE-Softmax	373,68	98,91	372,85	98,71
Worst Case ε -Greedy	-48,07	0,59	-50,62	0,00
Worst Case Softmax	-37,58	3,04	-41,82	2,05
Worst Case VDBE	103,30	35,88	-31,64	4,43
Worst Case VDBE-Softmax	262,99	73,10	256,06	71,49

Tabelle 4.2: Bandit-World-Problem: Return in den Lernepisoden 100 und 500 über 2000 Experimente gemittelt. Prozentangaben beziehen sich auf den am höchsten und am niedrigsten (gemittelt) gemessenen Return, $R_{\max} \approx 378.37$ und $R_{\min} \approx -50.62$.

4.8 Diskussion

In diesem Kapitel wurden zwei neue Explorationsstrategien, für modellfreies Reinforcement Learning mit diskreten Aktionen, vorgestellt und analysiert. Die VDBE-Strategie erweitert die ε -Greedy-Strategie dahingehend, dass die Explorationsrate ε nicht mehr vom Experimentator festgelegt werden muss, sondern beim Aktualisieren der Wertefunktion, auf Basis des

Lernfortschritts, adaptiert wird. Große Wertunterschiede führen dazu, dass sich der Agent erkundend verhält, was dadurch erreicht wird, dass die zustandsabhängige Explorationsrate gegen $\varepsilon(s) \rightarrow 1$ strebt. Hingegen führen kleine Wertunterschiede dazu, dass der Agent das bereits erlernte Wissen über die Umgebung ausnutzt, da die Explorationsrate gegen $\varepsilon \rightarrow 0$ strebt (Tokic, 2010). Auf Basis eines natürlichen Lernsignals, dem TD-Fehler, erweitert VDBE das Modell von Kobayashi u. a. (2009) in zwei Punkten:

1. Es wird eine lokale, zustandsabhängige Explorationsrate $\varepsilon(s)$ verwendet, was durch die angenommene Markov-Eigenschaft einer Umgebung begründet wird. Hierdurch ist es möglich in Bereichen des Zustandsraumes explorativ zu sein, in denen sich die Dynamik der Umgebung geändert hat bzw. die Prädiktionen noch unzureichend die Observationen widerspiegeln.
2. Es wird ebenso auch die Lernrate α in Betracht gezogen, da der TD-Fehler bei stochastischen Rewards nicht konvergiert.

In der Untersuchung stellt sich für die VDBE-Strategie heraus, dass Fluktuationen in der Wertefunktion ebenfalls den Return beeinträchtigen. Diese können zum einen dadurch entstehen, dass die Umgebung ein nichtdeterministisches Verhalten aufweist, z. B. durch stochastische Rewards oder stochastische Folgezustände. Zum anderen kann eine stochastische Aktionswahl bei on-policy Lernverfahren, wie z. B. Sarsa, ebenso Fluktuationen hervorrufen. Um diesen Aspekt zu verbessern wurde die VDBE-Strategie zur VDBE-Softmax-Strategie erweitert, bei welcher Explorationsaktionen nicht mehr gleichverteilt gewählt werden, sondern gewichtet auf Basis der Q -Werte (Tokic und Palm, 2011; Tokic u. a., 2012).

Kapitel 5

Problemdimensionen & Experimente

Im Folgenden werden die in dieser Arbeit vorgestellten VDBE-Strategien an weiteren Lernproblemen evaluiert und ein Überblick über Problemdimensionen gegeben. Hierfür werden bekannte Lernprobleme aus der RL-Literatur verwendet, ebenso aber auch selbst entwickelte, zweckdienliche Lernprobleme.

5.1 Lernprobleme im Reinforcement Learning

In den letzten Jahrzehnten wurde Reinforcement Learning erfolgreich zum Lösen unterschiedlicher Problemstellungen eingesetzt, wodurch das entstandene Anwendungsspektrum mittlerweile recht umfangreich ist. Daher ist es sinnvoll, Lernprobleme aufgrund der Eigenschaft des Rewards, der Größe des Zustandsraums sowie des Verhaltens der Umgebung zu kategorisieren. Grob zusammengefasst kristallisieren sich signifikante Lernproblemdimensionen heraus¹, die im Folgenden kurz erläutert werden:

Zeitverzögerte oder unmittelbare Rewards

Der Reward des Agenten kann unmittelbar oder zeitverzögert erhalten werden. Bei letzterem entsteht das Temporal Credit-Assignment Problem (Sutton, 1984), da nicht nur der Reward der zuletzt getätigten Aktion ziel führend war, sondern ebenso viele zuvor ausgeführte Aktionen. Die Ursache dieses Problems ist oftmals darauf zurückzuführen, dass es für viele

¹vgl. u. a. (Russell und Norvig, 2009, S. 42ff.)

Lernprobleme schlichtweg schwierig ist, unmittelbare Rewards exakt zu definieren, wie z. B. beim Mountain-Car-Problem (Moore, 1990) oder bei Brettspielen (Faußer und Schwenker, 2010; Tesauro, 2002; Thrun, 1995). Allgemein gilt hierbei: je später der Reward vom Agenten empfangen wird, desto länger ist die Lernzeit des Agenten, falls die in Kapitel 2.7 beschriebenen Lernverfahren zum Einsatz kommen. Eine Erweiterung dieser Verfahren bieten sogenannte *Eligibility-Traces* (Singh und Sutton, 1996; Sutton und Barto, 1998), welche beim Update der Wertefunktion den unmittelbaren Reward r_{t+1} , gewichtet auf zuvor ausgeführten Aktionen, zurückpropagieren, jedoch hierfür das separate Speichern der Aktionshistorie benötigen.

Deterministische oder stochastische Rewards

Der Reward eines Agenten kann deterministisch oder stochastisch sein. Insbesondere bei Robotikanwendungen sind Sensorwerte oftmals mit Rauschen behaftet, wodurch ein eigentlich deterministischer Reward eventuell stochastisch wahrgenommen wird. Dies kann zu Fluktuationen in der Wertefunktion und folglich zu suboptimalem Verhalten führen. Für stationäre Reward-Verteilungen kann dieser Effekt dadurch verringert werden, indem z. B. das in Kapitel 4.4.3 beschriebene Incremental-Sample-Average-Verfahren zur Steuerung der Lernrate α verwendet wird. Ein bekanntes Lernproblem ist hierbei das n -armige Banditen-Problem (Robbins, 1952; Sutton und Barto, 1998; Tokic, 2010), welches bereits in Kapitel 4.5 untersucht wurde.

Stationarität der Umgebung

Zusätzlich ist die Eigenschaft der Stationarität des Reward-Signals und der Umgebungsdynamik wichtig (Choi u. a., 1999; Främling, 2005). Lernprobleme aus der Realwelt sind typischerweise nichtstationär, da Objekte und Zielzustände erscheinen, sich bewegen und ebenso wieder verschwinden können. Der Lernerfolg hängt für derartige Umgebungen signifikant von der gewählten Lernrate α ab, da über diese gesteuert wird, mit welchem Gewicht die observierten Ereignisse in die Wertefunktion einfließen. Im Allgemeinen gilt, dass mit hohen Lernraten schneller auf nichtstationäre Prozesse reagiert werden kann, jedoch dafür evtl. Sensorrauschen verstärkt mit in die Wertefunktion einfließt, was Fluktuationen hervorrufen kann. Für nichtstationäre Rewards können daher Meta-Lernverfahren zum Einsatz kommen (Kobayashi u. a., 2009; Schweighofer und Doya, 2003) oder

niedrige, konstante Lernraten α verwendet werden. Im Folgenden wird dieser Effekt am Dynamic-Cliff-Problem genauer untersucht.

Partielle Beobachtbarkeit von Zustandsinformationen

Eine weitere Dimension ist der Beobachtungsgrad des Zustandssignals s (Russell und Norvig, 2009, S. 658ff.). Partiell beobachtbare Lernprobleme sind oftmals in der Realwelt anzutreffen, beispielsweise im Robotik-Bereich, bei welchen aufgrund der Sensorik nur Teile des Zustandssignals s wahrgenommen werden können. Beispielhaft sei in diesem Zusammenhang die RoboCup MiddleSize-League zu erwähnen, wo Roboter über eine omnidirektionale Kamera die Spielfeldlinien und Farben der Tore wahrnehmen können, die jedoch oftmals von anderen Mit- und Gegenspielern verdeckt sind. Dies hat zur Folge, dass relevante Informationen zur genauen Roboterlokalisierung fehlen können und daher z.B. über Kalman-Filter geschätzt werden müssen.

Kontinuierliche oder diskrete Zustandsdimensionen

Die Form einer Zustandsdimension ist ebenso entscheidend für die Komplexität eines Lernproblems, da diese entweder diskretisiert oder kontinuierlich sein kann (Sutton, 1996). Bei letzterer geht oftmals die Konvergenz-Garantie von Lernverfahren wie Q -learning verloren, da die Annahme, dass alle Zustände unendlich oft durchlaufen werden, nicht erfüllt werden kann. Zur Vereinfachung kann eine Diskretisierung kontinuierlicher Zustandsdimensionen Abhilfe schaffen (Tokic u. a., 2010a). Bei dieser Vorgehensweise entsteht jedoch die Problematik, dass die Umgebung partiell-beobachtbar wird, da der Agent nicht auf Informationen *zwischen* zwei diskreten Zuständen zugreifen kann. Im Folgenden wird dieses Phänomen anhand des diskreten Mountain-Car-Problems sowie an einem Laufroboter, der selbstständig das Vorwärtsbewegen erlernt, untersucht.

Deterministische oder stochastische Folgezustände

Die Zustandsübergangsfunktion $\mathcal{P}_{ss'}^a$ kann stochastisch oder deterministisch sein. Bei deterministischen Übergangsfunktionen befindet sich der Agent nach Ausführung von Aktion a in Zustand s , deterministisch in einem Folgezustand s' . Bei stochastischen Übergangsfunktionen unterliegt s' hingegen einer Verteilung von möglichen Folgezuständen, die der Agent im modellfreien Temporal-Difference Learning nicht kennt. Ein bekanntes Lernproblem aus diesem Bereich ist das *Windy-Gridworld*-Problem von Sut-

ton und Barto (1998), bei welchem der Folgezustand nicht nur durch die Aktionswahl des Agenten beeinflusst wird, sondern zusätzlich auch von einer Zufallsvariablen, dem *stochastischen Wind*.

Episodische oder kontinuierliche Lernprobleme

Ein Lernproblem kann episodisch oder kontinuierlich sein (Sutton und Barto, 1998). Bei episodischen Problemen existiert oftmals eine Menge von Zielzuständen ($\mathcal{S}^+ \setminus \mathcal{S}$), in welchen eine Lernepisode terminiert. Das Episodenende kann jedoch auch zu einem festen Zeitpunkt T erfolgen, auch wenn kein Zielzustand erreicht wird. Im Gegensatz dazu endet bei einem kontinuierlichen Lernproblem die Episode nicht durch das Lernproblem selbst, sondern meist nur durch externe Intervention eines Bedieners. Dies ist beispielsweise bei Temperaturreglern der Fall, bei welchen ein Regelkreis das gewünschte Temperaturlevel ansteuert. Ebenso ist der Laufroboter aus Kapitel 5.4 ein kontinuierliches Lernproblem, bei welchem eine zyklische Strategie π erlernt wird, die es dem Roboter ermöglicht sich vorwärts zu bewegen (Tokic und Bou Ammar, 2012; Tokic u. a., 2009, 2010a).

5.2 Klassifikation von Lernproblemen

Wie man sieht existieren unterschiedliche Problemdimensionen, die allesamt wichtige Aspekte zum möglichst optimalen Lösen einer gegebenen Problemstellung sind. Die wichtigsten Dimensionen sind hierbei:

- Größe des Zustandsraumes (klein oder groß)
- Reward (fast unmittelbar, stark zeitverzögert, deterministisch oder stochastisch)
- Beobachtbarkeit (vollständig oder partiell)
- Art der Zustandsübergangsfunktion (deterministisch oder stochastisch)

In Tabelle 5.1 sind einige erfolgreich gelöste Lernprobleme aus der RL-Literatur aufgezeigt und nach Ihren Problemdimensionen klassifiziert. Auf den ersten Blick fällt auf, dass bei Realwelt-Problemen der Zustandsraum typischerweise *groß* und *partiell beobachtbar* ist; letzteres oftmals aufgrund von sensorischer Wahrnehmung des Zustandssignals. Hingegen ist bei Spielen der Zustandsraum überwiegend *vollständig beobachtbar*, es sei denn

es existieren z. B. verdeckte gegnerische Karten, wie bei „Siedler von Catan“. Bei Realwelt-Problemen wird der Reward hingegen oftmals *unmittelbar* erhalten, da dieser über Sensoren gemessen werden kann, wie beispielsweise beim Helikopterfliegen (Ng u. a., 2004). Bei Brettspielen ist dieser jedoch typischerweise *deterministisch* und *zeitverzögert*, da es oftmals schwierig ist, unmittelbare Rewards ohne Expertenwissen zu definieren.

5.3 Das Cliff-Walking-Problem

Das dargestellte *Cliff-Walking-Problem* in Abbildung 5.1 ist ein episodisches Lernproblem, welches durch eine zweidimensionale Gridworld modelliert wird (Sutton und Barto, 1998, S. 149f.). Das Ziel des Agenten ist es, ausgehend vom unteren linken Startzustand *S* den unteren rechten Zielzustand *G* zu erreichen. Eine Klippe befindet sich direkt zwischen dem Start- und dem Zielzustand, mit der Folge, dass der kürzeste Weg zum Ziel direkt an der Klippe entlangführt. Für getätigte Aktionen, die nicht zur Überschreitung der Klippe (*The Cliff*) führen, erhält der Agent einen Reward in Höhe von $r_{t+1} = -1$, sozusagen als Wegekosten. Überschreitet der Agent jedoch die Klippe, wird diese Aktion mit einem negativen Reward in Höhe von $r_{t+1} = -100$ bestärkt und der Agent zusätzlich in den Startzustand *S* zurückversetzt. Falls der Agent hingegen den Zielzustand *G* erreicht, terminiert die aktuelle Lernepisode.

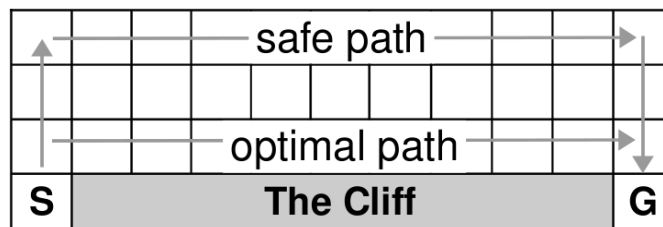


Abbildung 5.1: Cliff-Walking-Problem: Der *safe path* wird von Sarsa mit stochastischen Strategien gelernt, hingegen der (gefährlichere) *optimal path* von *Q*-learning. Zustand *S* markiert den Startzustand, hingegen *G* den Zielzustand. Skizze nach (Sutton und Barto, 1998).

Sutton und Barto illustrieren am Cliff-Walking-Problem das unterschiedliche Lernverhalten von TD-Lernverfahren bei Verwendung von stochastischen Strategien. Da Sarsa im TD-Fehler den *Q*-Wert des Zustands-Aktions-Paars (s', a') verwendet, d. h. den erwarteten Reward der tatsächlich getätigten Aktion im Folgezustand, wird hierdurch die Stochastik einer Strategie mitgelernt. *Q*-learning verwendet hingegen die Aktion mit dem höchsten *Q*-Wert des Folgezustandes, $\arg \max_{b \in \mathcal{A}(s')} Q(s', b)$, was demnach

Lernproblem	Größe		Reward			Beob.		Überg.		
	klein	groß	unmittelbar	zeitverzögert	deterministisch	stochastisch	vollständig	partiell	deterministisch	stochastisch
Realwelt-Probleme										
Aktien-Prognosesysteme (Lee, 2001)		X		X		X		X		X
Autofahren (Riedmiller u. a., 2007)		X	X		X		X		X	X
Helikopterfliegen (Ng u. a., 2004)		X	X		X		X		X	X
Laufroboter (Kimura u. a., 1997; Tokic u. a., 2009)	X		X			X		X		X
Lernen von Motorprimitiven (Kober u. a., 2010)		X		X		X		X		X
Slot-Car Racing (Kietzmann und Riedmiller, 2009)		X	X		X		X		X	X
Von menschlichem Reward lernen (Thomaz und Breazeal, 2008)		X		X		X	X		X	
Spiele										
Backgammon (Tesauro, 2002)		X		X	X		X		X	
Blackjack (Kakvi, 2009; Pérez-Uribe und Sanchez, 1998)	X		X		X		X			X
Dame (Faußer und Schwenker, 2010)		X		X	X		X		X	
<i>n</i> -armiges Banditenproblem (Robbins, 1952; Tokic, 2010)	X		X			X	X		X	
Othello (Eck und Wezel, 2008)		X		X	X		X		X	
Pig (Neller u. a., 2006)		X	X		X		X			X
Roulette (Hasselt, 2010)	X		X		X		X			X
Schach (Thrun, 1995)		X		X	X		X		X	
Siedler von Catan (Pfeiffer, 2004)		X		X	X			X		X
Vier Gewinnt (Faußer und Schwenker, 2008)		X		X	X		X		X	
Sonstige (simulierte) Lernprobleme										
Cliff-Walking-Problem (Sutton und Barto, 1998)	X		X		X		X		X	
Simulierte Gasturbinensteuerung (Hans u. a., 2008)		X	X		X		X		X	
Inverses-Pendel / Cart-Pole (Flentge, 2005)		X		X	X		X		X	
Mountain-Car (Sutton, 1996) sowie in Kapitel 5.5		X		X	X		X		X	
RoboCup Simulation (Gabel u. a., 2009; Stone u. a., 2005)		X		X		X		X		X

Tabelle 5.1: Einordnung der Problemdimensionen einzelner Lernprobleme.

keine Kosten mit einbezieht, die durch eine stochastische Aktionswahl im Folgezustand entstehen können. Am Cliff-Walking-Problem wird dieser

Unterschied dadurch erkennbar, dass Sarsa bei hoch stochastischen Strategien den *safe path* erlernt, welcher weiter entfernt von der Klippe entlangführt. Dieser Pfad führt zwar zu mehr Wegekosten, jedoch nicht zu den Zuständen direkt an der Klippe, in welchen durch Explorationsaktionen hohe Kosten ($r_{t+1} = -100$) entstehen können. Q -learning hingegen lernt den kürzeren *optimal path* mit den geringsten Wegekosten. Bei stochastischen Strategien fällt hierdurch der Agent öfters die Klippe herunter, was Sutton und Barto (1998, S. 150) für ε -Greedy mit Explorationsrate $\varepsilon = 0.1$ veranschaulichen.

5.3.1 Experimentelle Untersuchung

Der Versuchsaufbau des Cliff-Walking-Experimentes besteht aus einer 12x4 Gridworld, wie in Abbildung 5.2 dargestellt. Die untere linke Ecke gibt den Startzustand S an, die untere rechte Ecke den Zielzustand G (analog zu Abbildung 5.1). Aufgrund der Zustandsraumgröße ist der kumulierte Reward für das Folgen des optimalen Pfades $R_{\text{optimal}} = -13$, hingegen beim sicheren Pfad $R_{\text{safe}} = -17$. Beim Überschreiten der Klippe erhält der Agent den stark negativen Reward in Höhe von $r_{t+1} = -100$. Rote Wände (rote Rechtecke) innerhalb eines Zustandes sind für den Agenten nicht passierbar. Das Überschreiten von Wänden außerhalb des aktuellen Zustandes, beispielsweise die Klippe von $B2$ nach $B3$, ist jedoch möglich (Einbahnstraßenprinzip).

In jeder Lernepisode stehen dem Agenten maximal $T = 100$ Schritte zur Verfügung, um Zielzustand G zu erreichen. Wird dieser erreicht oder das Zeitlimit überschritten, terminiert die aktuelle Episode und der Agent wird wieder zurück in den Startzustand S versetzt. Während eines Experimentes kann der Agent seine Aktionsauswahl innerhalb von 500 Lernepisoden verbessern. Die untersuchten Parameterintervalle sind: ε -Greedy mit $\varepsilon \in [0, 1]$, Softmax mit $\tau \in [0.04, 100.0]$, VDBE und VDBE-Softmax mit $\sigma \in [0.04, 100.0]$ sowie $\delta(s) = 1/\mathcal{A}(s)$. Für VDBE-Softmax wurden alle Werte eines Zustandes in das Intervall $[-1, 1]$ normalisiert und der Parameter τ mit dem Wert $\tau = 1$ belegt.

5.3.2 Ergebnisse

Die Ergebnisse des Cliff-Walking-Experimentes sind in den Abbildungen A.1 und A.2 sowie in Tabelle 5.2 dargestellt (Tokic und Palm, 2011). Die abgebildeten Werte sind aufgrund der Stochastik über 1000 Experimente gemittelt. Den Ergebnissen ist zu entnehmen, dass alle untersuch-

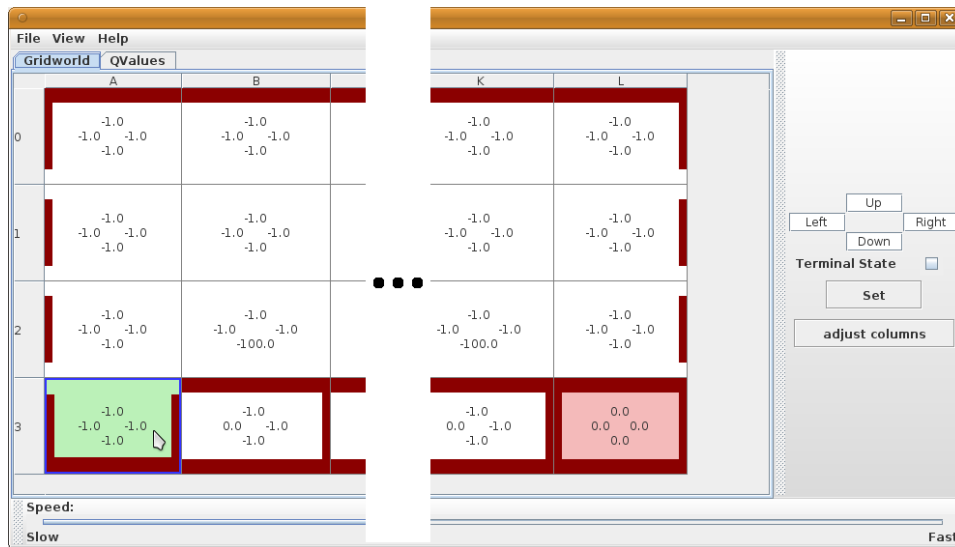


Abbildung 5.2: Cliff-Walking-Problem: Modellerte Umgebung im Gridworld-Editor der Teaching-Box (Ertel u. a., 2009; Tokic u. a., 2010a).

ten Strategien mit einer „Greedy“ Parameterbelegung optimale Ergebnisse erzielen. Dabei ist unerheblich ob Q -learning oder Sarsa verwendet wird, da beide Lernverfahren sich bei einer Greedy-Strategie identisch verhalten. Die entsprechenden Parameterbelegungen sind hierbei: $\varepsilon = 0$ bei ε -Greedy, niedrige Belegungen der Temperatur τ bei Softmax sowie hohe inverse Sensibilitäten σ bei VDBE-Strategien.

ε -Greedy und Softmax

Bei hoch stochastischer Aktionsauswahl ist den Ergebnissen zu entnehmen, dass Sarsa in Kombination mit ε -Greedy und Softmax einen höheren Reward gegenüber Q -learning aufzeigt, was die Ergebnisse von Sutton und Barto bestätigt. Die Ursache für dieses Verhalten ist darauf zurückzuführen, dass Sarsa ein *on-policy* Verfahren ist, welches eventuelle Kosten für Aktionen im Folgezustand, verursacht durch die Stochastik der Strategie im Folgezustand, mitberücksichtigt. Folglich lernt Sarsa den *sicheren Pfad*, wodurch der Agent seltener über die Klippe fällt und dementsprechend seltener den negativen Reward von $r_{t+1} = -100$ erhält. Die Lernrate α besitzt für beide Explorationsverfahren in Kombination mit Q -learning keinen signifikanten Einfluss, was unter anderem auf die deterministischen Rewards zurückzuführen ist. Im Limit konvergieren die Ergebnisse für $t \rightarrow \infty$ zu den selben Werten, unabhängig von der gewählten Lernrate α .

Return in Episode 100				
	Q-learn.	(Opt. %)	Sarsa	(Opt. %)
Greedy	-13,00	100,00	-13,00	100,00
Best Case ε -Greedy	-13,00	100,00	-13,00	100,00
Best Case Softmax	-13,06	99,99	-13,04	99,99
Best Case VDBE	-13,00	100,00	-15,33	99,51
Best Case VDBE-SM.	-13,00	100,00	-14,60	99,66
Worst Case ε -Greedy	-491,27	0,00	-483,14	1,70
Worst Case Softmax	-301,93	39,59	-224,03	55,88
Worst Case VDBE	-13,31	99,93	-489,12	0,45
Worst Case VDBE-SM.	-13,00	100,00	-212,97	58,19

Return in Episode 500				
	Q-learn.	(Opt. %)	Sarsa	(Opt. %)
Greedy	-13,00	100,00	-13,00	100,00
Best Case ε -Greedy	-13,00	100,00	-13,00	100,00
Best Case Softmax	-13,00	100,00	-13,00	100,00
Best Case VDBE	-13,00	100,00	-15,31	99,52
Best Case VDBE-SM.	-13,00	100,00	-14,60	99,67
Worst Case ε -Greedy	-490,73	0,11	-486,36	1,03
Worst Case Softmax	-291,59	41,75	-181,68	64,73
Worst Case VDBE	-13,00	100,00	-486,97	0,90
Worst Case VDBE-SM.	-13,00	100,00	-17,05	99,15

Tabelle 5.2: Cliff-Walking-Problem: Return in den Lernepisoden 100 und 500 über 1000 Experimente gemittelt. Prozentangaben beziehen sich auf den am höchsten und am niedrigsten gemessenen Return, $R_{\max} \approx -13$ und $R_{\min} \approx -491.27$.

Hingegen besitzt die Lernrate bei Sarsa durchaus Einfluss, wie den Ergebnissen von ε -Greedy (z. B. für $\varepsilon = 0.5$) zu entnehmen ist. Die Begründung hierfür ist, dass hohe Lernraten zu umso stärkeren Fluktuationen in den Q -Werten führen, desto höher der Stochastik-Anteil einer Strategie ist.

VDBE und VDBE-Softmax

Im Vergleich dazu ist den Ergebnissen der VDBE-Strategie zu entnehmen, dass für beliebige Parameterbelegungen von σ der Reward bei Q -learning stets zu den minimalen Kosten konvergiert (*optimal path*). Bei Sarsa hingegen ist erkenntlich, dass der Reward divergieren kann. Die guten Ergebnisse für Q -learning sind darauf zurückzuführen, dass VDBE bestrebt ist *Greedy* zu werden, was bei allen untersuchten Explorationsstrategien zu

optimalen Ergebnissen führt. Bei Sarsa kann die Konvergenz zu optimalen Kosten nur für bestimmte Parameterbereiche erreicht werden, die ihrerseits signifikant von der gewählten Lernrate α abhängen (z. B. für $\sigma = 1.0$). Dieses Verhalten ist darauf zurückzuführen, dass hohe Lernraten bei Sarsa zu großen Wertunterschieden führen und folglich auch zu einer erhöhten Stochastik der VDBE-Strategie, wie bereits in Abschnitt 4.4.2 erläutert. Den Ergebnissen von Q -learning kann hingegen entnommen werden, dass beliebige Belegungen des Parameters σ stets zu optimalen Ergebnissen konvergieren, wobei niedrige Belegungen lediglich die Konvergenzgeschwindigkeit verlangsamen. Analog gilt dies ebenso für die Lernrate α .

Den Ergebnissen von VDBE-Softmax kann man hingegen entnehmen, dass stets optimale Ergebnisse erzielt werden, sowohl für Q -learning als auch für Sarsa. Im Gegensatz zu VDBE sind auch die Ergebnisse für VDBE-Softmax unter Verwendung des Lernverfahrens Sarsa stabil. Analog zur VDBE-Strategie besitzen die Parameter σ und α lediglich Einfluss auf die Konvergenzgeschwindigkeit, wobei niedrige Belegungen diese verlangsamen.

5.3.3 Diskussion

Zusammengefasst bleibt festzuhalten, dass am Cliff-Walking-Problem die optimale Strategie *Greedy* ist und eine Lernrate von $\alpha = 1$ zu optimalen Ergebnissen führt. Selbstverständlich kann diese Belegung nicht für alle Lernprobleme verallgemeinert werden, wie bereits am n -armigen Banditen-Problem gezeigt wurde, da eine *Greedy*-Strategie oftmals nur zu suboptimalen Ergebnissen führt. Darüber hinaus wurde gezeigt, dass durch den Einsatz der VDBE-Softmax-Strategie das Lernen robuster gegenüber Fluktuationen in der Wertefunktion wird. Am Cliff-Walking-Problem wurden diese künstlich durch den Einsatz des on-policy Verfahrens Sarsa, in Kombination mit hohen Lernraten α , erzeugt. In der Praxis können Fluktuationen jedoch auch andere Quellen haben, z. B. das Rauschen von Sensoren an Robotern.

5.4 Der Laufroboter

Je mehr die Komplexität eines Roboters oder die seiner Umgebung zunimmt, desto schwieriger wird es für einen Ingenieur *optimales Verhalten* von Hand zu programmieren. Die Konsequenz dessen ist, dass Roboter oftmals nur mit suboptimalem Verhalten ausgestattet werden können. Aufgrund dessen ist es von besonderem Interesse, Roboterverhalten erlernen

zu lassen, anstatt dieses von Hand zu programmieren (Peters und Schaal, 2008; Tokic u. a., 2009).

Der im Folgenden vorgestellte Laufroboter erlernt sein Verhalten vollkommen selbstständig. Das Ziel ist es, eine Strategie zu erlernen, welche die Fortbewegungsgeschwindigkeit auf beliebigen Untergründen maximiert. Interessanterweise variiert die optimale Strategie in Abhängigkeit der Oberflächenbeschaffenheit. Für einen Ingenieur wäre die Modellierung eines entsprechend adaptiven Verhaltens eine schwer zu lösende Aufgabe, da auf unterschiedlichen Untergründen unterschiedliche Trajektorien den Reward maximieren. Anstelle dessen kann es die Aufgabe des Ingenieurs sein, ein entsprechendes RL-Lernverfahren zu implementieren, die Sensorik und Aktorik des Roboters damit zu verbinden, wodurch es dem Roboter ermöglicht wird, die Strategie zur Vorwärtsbewegung anhand sensorischer Interaktion zu erlernen.

5.4.1 Die Architektur

Ursprünglich wurde die Architektur des Laufroboters von Kimura u. a. (1997) vorgestellt (vgl. Abbildung 5.3). Das Modell ist sehr einfach gehalten und besteht im Wesentlichen aus dem Körper des Roboters mit zwei Gelenken, die einen *Krabbelarm* zur Fortbewegung bilden. Für das Lernen von Vorwärtsbewegungen wird der Zustandsraum des Roboters, der aus den beiden Gelenk-Positionen des Krabbelarmes besteht, diskretisiert. Damit sich der Roboter vorwärts bewegen kann, muss dessen Arm zyklisch mehrere aufeinanderfolgende Bewegungen tätigen (vgl. Tabelle 5.3 sowie Abbildung 5.4). Der Reward r_{t+1} ist die gemessene Distanz, die der Roboter nach dem Tätigen einer Aktion zurückgelegt hat. Dieser ist demnach positiv für Vorwärtsbewegungen, hingegen negativ für Rückwärtsbewegungen.

Der Nachbau des Laufroboters ist in Abbildung 5.3(b) dargestellt (Tokic u. a., 2009, 2010a). Die Gelenke des Krabbelarms sind durch Dynamixel AX-12 Aktuatoren realisiert, die an einer Mikrocontroller-Platine angeschlossen sind. Die Platine mit einem ATmega32-Mikrocontroller befindet sich auf der Oberseite des Roboters, womit die Sensorik und Aktorik des Roboters verbunden ist. Dabei ist es möglich Aktionskommandos über eine Funkschnittstelle an den Roboter zu senden sowie aktuelle Daten des Lernverfahrens zur Laufzeit abzufragen (Wertefunktionstabelle und Rewardtabelle etc.). Im Unterschied zum Modell von Kimura u. a. (1997) werden im Nachbau Räder an der Unterseite des Roboters verwendet, die an einer

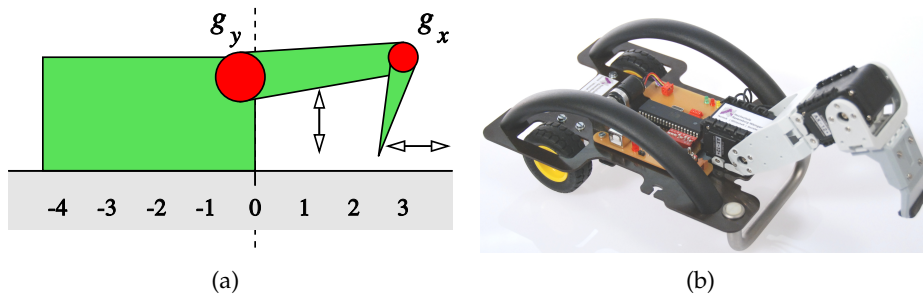


Abbildung 5.3: Laufroboter: (a) zeigt das Modell nach Kimura u. a. (1997) sowie (b) den entwickelten Hardware-Roboter (Tokic und Bou Ammar, 2012; Tokic u. a., 2009, 2010a).

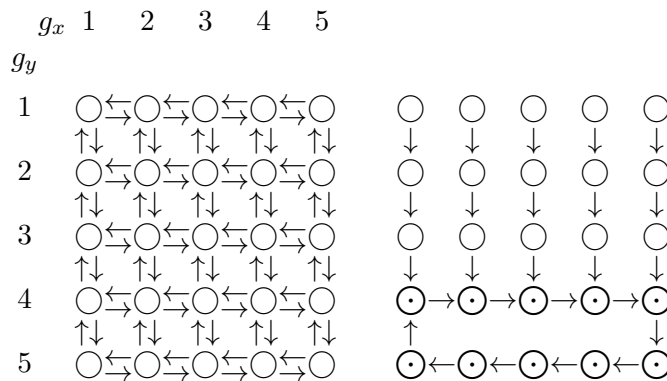


Abbildung 5.4: Laufroboter: Das 5×5 Zustandsraum-Modell (links) sowie eine zyklische Strategie zur Vorwärtsbewegung (rechts). Zustände innerhalb des Zyklus sind mit \odot markiert.

durchgängigen Achse befestigt sind. Diese haben zur Aufgabe die Reibung zwischen dem Roboter und dem Untergrund zu verringern, wodurch sich der Roboter leichter vorwärts bzw. rückwärts bewegen kann.

Das Reward-Signal für den Lernprozess liefert ein optischer Inkremental-Encoder², dessen Drehscheibe über einen Zahnriemen mit der Radachse verbunden ist. Zwei Signalleitungen des Encoders sind mit dem externen Interrupt des Mikrocontrollers verbunden, der nur während der Ausführung einer Roboteraktion aktiviert ist. Während einer Aktion werden die einzelnen Encoderimpulse innerhalb eines Zeitfensters von 500 Millisekunden aufsummiert und anschließend als Reward-Signal r_{t+1} an das Lernverfahren geliefert. Der Rewardbereich befindet sich zwischen $r \in$

²Ein ME16 der Firma RWP Ruhlatec.

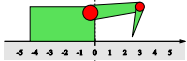



robot	time t	state		reward	action a_t
		g_y	g_x	x	
	0	up	left	0	right
	1	up	right	0	down
	2	down	right	0	left
	3	down	left	1	up

Tabelle 5.3: Laufroboter: Vier Schritte einer einfachen zyklischen Strategie zur Vorwärtsbewegung (Tokic u. a., 2009).

$[-30, 23]$, wobei ein vollständiges Reward-Modell in Abbildung 5.5(a) dargestellt ist.

5.4.2 Partielle Beobachtbarkeit

Die Umgebung des Laufroboters ist partiell beobachtbar, was dazu führen kann, dass nicht die tatsächlich optimale Strategie erlernt wird. Diese Einschränkung ist darauf zurückzuführen, dass ein eigentlich kontinuierlicher Zustandsraum, aus didaktischen Gründen, diskretisiert wurde, wodurch nicht auf Informationen zwischen den Zuständen zugegriffen werden kann. Es werden jedes der beiden Gelenke in fünf equidistant verteilte Zustände unterteilt, was eine Zustandsraumgröße von insgesamt 25 Zuständen ergibt. In den Zustandsübergängen besitzt das Reward-Signal eine nichtlineare Eigenschaft, wie man Abbildung 5.5(a) entnehmen kann³. Ebenso wurde der Aktionsraum diskretisiert, wobei nur die folgenden vier Aktionen möglich sind:

$$\mathcal{A} = \{\text{HOCH, RUNTER, LINKS, RECHTS}\} \quad .$$

Die Diskretisierung des Zustands- sowie Aktionsraumes erfolgt aus didaktischen Gründen, da der Laufroboter als Demonstrator für Reinforcement Learning in der Lehre verwendet wird (Montresor u. a., 2011; Tokic und

³Zum Beispiel beim Betrachten der Aktion RECHTS in Zeile 3 ($17 \rightarrow 17 \rightarrow 13 \rightarrow 12$) oder Zeile 4 ($14 \rightarrow 16 \rightarrow 9 \rightarrow 10$).

Bou Ammar, 2012; Tokic u. a., 2009, 2010a). Ein weiterer Vorteil der Diskretisierung ist, dass der Zustandsraum erheblich verkleinert wird, wodurch das Speichern der Wertefunktionstabelle in den internen Speicher des ATmega32-Mikrocontrollers ermöglicht wird, welcher eine Größe von 2 kB besitzt. Der Roboter entwickelt nach dem Einschalten eine *gute Strategie* zur Vorwärtsbewegung gewöhnlich binnen 15-20 Sekunden unter Verwendung des Wert-Iteration-Verfahrens (Tokic u. a., 2009). Die Fortbewegungsstrategie wird beim Ausschalten des Roboters nicht gespeichert, sondern nach jedem Anschalten von Grund auf neu gelernt.

5.4.3 Experimentelle Untersuchung

Das Lernproblem beim Laufroboter ist es, eine optimale Strategie zur Vorwärtsbewegung zu finden, welche die Anzahl positiver Enkodersignale maximiert. Ein Modell der Umgebung, wie sie der Lernalgorithmus des Roboters sieht, ist in Abbildung 5.5(a) dargestellt. Das abgebildete Reward-Modell stammt von einer Interaktion des Laufroboters auf einem Linoleumboden, welches über eine Funkschnittstelle von der Teaching-Box-Software aufgezeichnet wurde (Ertel u. a., 2009; Tokic u. a., 2010a).

Bei Befolgung des optimalen Zyklus erhält der Roboter einen Reward von durchschnittlich $\bar{r}_{t+1} = \frac{17+20-6-8}{4} = 5.75$. Falls eine Lernepisode nach $T = 200$ Schritten künstlich terminiert, ergibt sich hieraus ein maximal zu erreichender Return in Höhe von $R_{\text{optimal}} = 1150$,

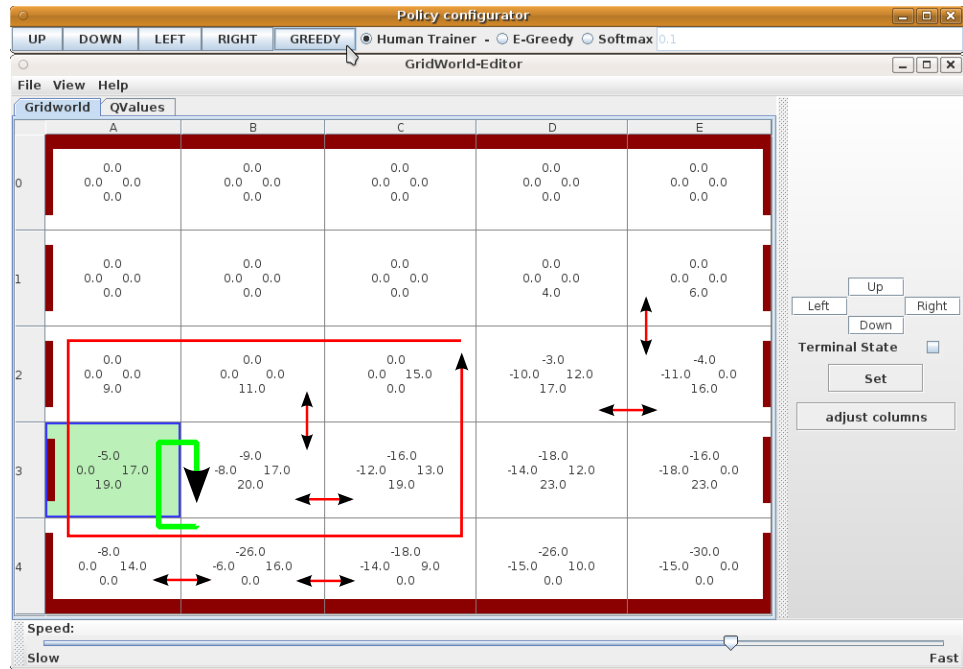
$$R_{\text{optimal}} = 200 \frac{\text{Schritte}}{\text{Episode}} * \bar{r}_{t+1} \frac{\text{Reward}}{\text{Schritt}} = 1150 \frac{\text{Reward}}{\text{Episode}} .$$

Bei einer Random-Strategie ($\varepsilon = 1$) bleibt der Roboter auf dem Boden stehen, da die Summe aller Rewards in Abbildung 5.5(a)

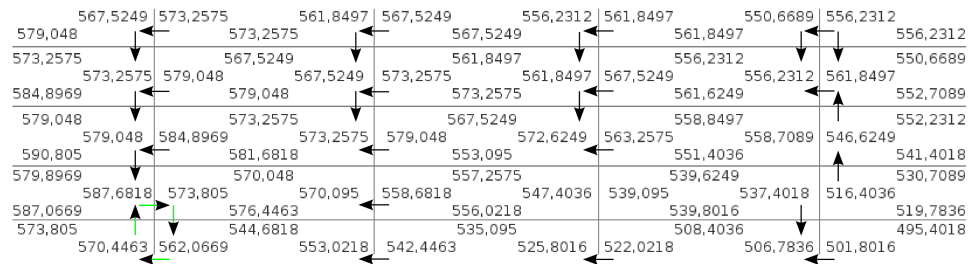
$$\bar{r}_{t+1}^{\text{random}} = \sum \mathcal{R}_{s,s'}^a = 0$$

ergibt. In der Praxis sowie in den folgenden simulierten Untersuchungen ist dieser Effekt beobachtbar.

Für den Laufroboter existieren viele suboptimale Strategien, die Zyklen im Zustandsraum darstellen. Welchen Zyklus der Roboter lernt, hängt stark von der Explorationsstrategie und dessen Parameterbelegung ab. Im Folgenden werden nun verschiedene Explorationsstrategien in der Simulati-



(a)



(b)

Abbildung 5.5: Laufroboter: Der Gridworld-Editor der Teaching-Box (Tokic u. a., 2010a): (a) zeigt die aufgezeichneten Rewards von einem Linoleumboden und (b) die dazugehörige Wertefunktion, welche durch Wert-Iteration mit $\gamma = 0.99$ erlernt wurde. Rote Pfeile in (a) kennzeichnen eine suboptimale Strategie, hingegen grüne und schwarze Pfeile in (a)+(b) die tatsächlich optimale Strategie. Der grüne Zyklus in (b) besitzt einen mittleren Reward pro Aktion in Höhe von $\bar{r}_{t+1} = \frac{17+20-6-8}{4} = 5.75$.

on untersucht, um herauszufinden, welche Fortbewegungsgeschwindigkeit vom Laufroboter durchschnittlich erreicht wird. Diese wird als der kumulierte Reward einer Lernepisode gemessen, wobei jede Episode nach $T = 200$ Aktionen künstlich terminiert. Am Anfang einer Episode ist der Roboterarm auf eine zufällige Position gesetzt, um ein Festhängen in suboptimalen Strategien zu überwinden. Um Rauschen im Reward-Sensor zu simulieren wird der Reward einer Aktion normalverteilt an den Agenten geliefert. Hierzu wird der observierte Reward (vgl. Abbildung 5.5) als Mittelwert für ein stochastisches Reward-Signal verwendet, auf welches ein Rauschen mit Standardabweichung $\sigma = 1.0$ addiert wird:

$$r_{t+1} \sim \mathcal{N}(\mathcal{R}_{ss'}^a, 1.0) \ .$$

Der Return wird für verschiedene Belegungen des Explorationsparameters über jeweils 500 Episoden gemessen und die erhaltenen Ergebnisse über 2000 Experimente gemittelt. Die untersuchten Parameterintervalle sind: Explorationsrate $\varepsilon \in [0.0, 1.0]$ für ε -Greedy; Temperatur $\tau \in [0.01, 100.0]$ für Softmax; inverse Sensibilität $\sigma \in [0.01, 100.0]$ und Lernrate $\delta(s) = 1/|\mathcal{A}(s)|$ für VDBE sowie VDBE-Softmax. Für VDBE-Softmax werden die Q -Werte in das Intervall $[-1, 1]$ normalisiert. Für Q -learning und Sarsa wird der Diskontierungsfaktor $\gamma = 0.95$ und die Lernrate $\alpha = 1.0$ verwendet.

5.4.4 Ergebnisse

Die Ergebnisse sind in Abbildung A.3 und Tabelle 5.4 dargestellt. Im Gegensatz zum Cliff-Walking-Problem erhält man optimale Ergebnisse nicht anhand einer Greedy-Strategie ($\varepsilon = 0$), sondern durch unterschiedliche Parameterbelegungen der untersuchten Explorationsstrategien. Die Ursache hierfür ist, dass in der Umgebung des Laufroboters viele Möglichkeiten für suboptimale Strategien existieren, in welchen sich dieser beim Folgen einer Greedy-Strategie verfangen kann. Aufgrund dessen wird mit einer Greedy-Strategie nach 500 Lernepisoden ein durchschnittlicher Return in Höhe von $R_{\text{greedy}} = 290$ erreicht, was ca. 25 % des höchstmöglichen Returns einer optimalen Strategie entspricht ($R_{\text{optimal}} = 1150$).

ε -Greedy und Softmax

Mit Q -learning erhält man bei Verwendung einer ε -Greedy-Strategie mit niedrigen Explorationsraten ($\varepsilon \gtrsim 0$) nahezu optimale Ergebnisse. Das beste Ergebnis wird mit einer Explorationsrate von $\varepsilon = 0.01$ erzielt. Der Re-

Return in Episode 100				
	Q-learn.	(Opt. %)	Sarsa	(Opt. %)
Greedy	278,11	25,62	280,97	25,88
Best Case ε -Greedy	1010,51	92,31	544,90	49,91
Best Case Softmax	475,40	43,58	1007,63	92,05
Best Case VDBE	1040,05	95,00	505,75	46,35
Best Case VDBE-SM.	1047,98	95,72	571,87	52,37
Worst Case ε -Greedy	-0,07	0,29	-0,58	0,24
Worst Case Softmax	43,04	4,21	43,81	4,28
Worst Case VDBE	5,11	0,76	1,24	0,41
Worst Case VDBE-SM.	741,41	67,81	300,08	27,62

Return in Episode 500				
	Q-learn.	(Opt. %)	Sarsa	(Opt. %)
Greedy	288,01	26,52	290,01	26,70
Best Case ε -Greedy	1078,61	98,51	783,83	71,67
Best Case Softmax	531,61	48,70	1006,86	91,98
Best Case VDBE	1094,94	100,00	778,73	71,21
Best Case VDBE-SM.	1067,99	97,55	695,23	63,60
Worst Case ε -Greedy	-1,15	0,19	-1,67	0,14
Worst Case Softmax	42,49	4,16	43,60	4,26
Worst Case VDBE	5,09	0,76	-3,21	0,00
Worst Case VDBE-SM.	741,13	67,78	304,10	27,98

Tabelle 5.4: Laufroboter: Return in den Lernepisoden 100 und 500 über 2000 Experimente gemittelt. Prozentangaben beziehen sich auf den am höchsten und am niedrigsten (gemittelt) gemessenen Return, $R_{\max} \approx 1094.94$ und $R_{\min} \approx -3.21$.

turn konvergiert in Lernepisode 500 zu $R_{\max}^{\text{egreedy}} \approx 1078$, was ca. 93 % von R_{optimal} entspricht. Für Sarsa führt diese Belegung ebenfalls zu den besten Ergebnissen, erreicht jedoch mit $R_{\max}^{\text{egreedy-sarsa}} \approx 783$ nur ca. 68 % von R_{optimal} . Darüber hinaus ist den Ergebnissen zu entnehmen, dass im untersuchten Parameterintervall beide Verfahren sehr gut aber auch sehr schlecht abschneiden können. Bei ε -Greedy liegt das Optimum im Bereich $\varepsilon \in [0.01, 0.10]$, bei Softmax hingegen bei $\tau \approx 10$.

VDBE und VDBE-Softmax

Den Ergebnissen der VDBE-Strategie in Kombination mit Q -learning ist zu entnehmen, dass nach 500 Lernepisoden die Höhe des Returns mindestens

dem einer Greedy-Strategie entspricht. Die besten Ergebnisse werden mit inversen Sensibilitäten von $\sigma \gtrsim 10$ erreicht. Je kleiner man jedoch σ wählt, desto mehr nähern sich die Ergebnisse denen einer Random-Strategie an, was aufgrund von Wertschwankungen durch die stochastischen Rewards verursacht wird. Die Ergebnisse für VDBE in Kombination mit Sarsa zeigen jedoch, dass für selbige Explorationsparameter schlechtere Ergebnisse erzielt werden. Dieser Effekt erklärt sich dadurch, dass *sichere Pfade* am Laufroboter dazu neigen, in suboptimalen Trajektorien zu enden, in welchen sich der Arm z. B. in der Luft nur hin- und herschwingt. Die Ergebnisse von VDBE-Softmax zeigen für Q -learning stabile Ergebnisse, hingegen jedoch nicht für Sarsa, aufgrund der hohen Lernrate bei stochastischen Rewards. Diese Faktoren haben eine nicht konvergente Wertefunktion zur Folge, wodurch eine hohe Explorationsrate entsteht.

5.5 Das diskrete Mountain-Car-Problem

Beim Mountain-Car-Problem handelt es sich um ein Lernproblem mit kontinuierlichen Zustandsdimensionen (Moore, 1990), bei welchem ein Auto einen Zielzustand auf einem Berg erreichen soll (vgl. Abbildung 5.6). Zu Beginn einer Lernepisode steht das Auto im Tal zwischen zwei Bergen und kann nicht aus eigener Kraft hinauffahren, da die Gravitationskraft stärker ist als die der Motorkraft. Um trotzdem das Ziel auf dem rechten Berg erreichen zu können, muss das Auto im Tal durch Hin- und Herschaukeln zusätzliche Energie sammeln, wodurch die Gravitation dennoch überwunden werden kann.

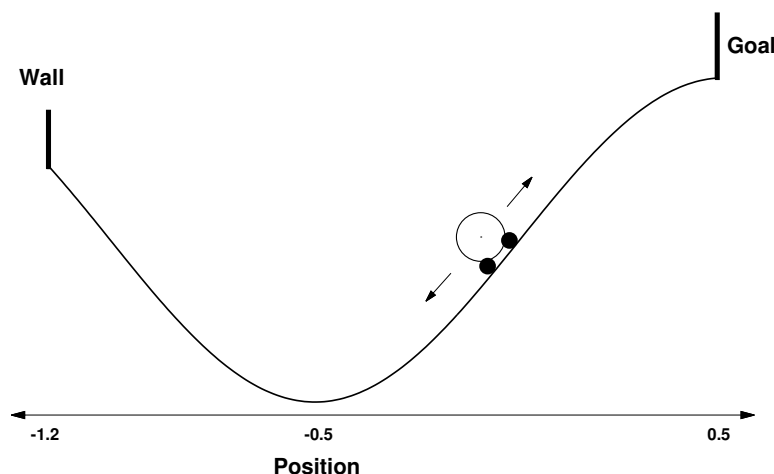


Abbildung 5.6: Mountain-Car-Problem: Skizze nach (Sutton und Barto, 1998).

Beim Mountain-Car-Problem ist der Zustand des Agenten beschrieben durch zwei Komponenten: (1) der Position x des Autos sowie (2) der momentanen Geschwindigkeit \dot{x} . In der Simulation wird die Dynamik der Umgebung durch Differential-Gleichungen modelliert, die abhängig von der gewählten Aktion sind:

$$\begin{aligned} x_{t+1} &= \text{bound}[x_t + \dot{x}_{t+1}] \\ \dot{x}_{t+1} &= \text{bound}[\dot{x}_t + 0.001a_t - 0.0025 \cos(3x_t)] \quad , \end{aligned} \quad (5.1)$$

wobei *bound* erzwingt, dass $-1.2 \leq x_{t+1} \leq 0.5$ sowie $-0.07 \leq \dot{x}_{t+1} \leq 0.07$. Für $x_{t+1} \leq -1.2$ wird $\dot{x}_{t+1} = 0$ gesetzt, was bedeutet, dass beim Erreichen der oberen linken Ecke die Geschwindigkeit auf 0 gesetzt wird. Um die Dynamik des Autos zu beeinflussen, stehen zu jedem Zeitschritt eine von drei möglichen Aktionen zur Verfügung:

$$a_t = \begin{cases} +1 & \text{Vorwärtsbeschleunigen, oder} \\ -1 & \text{Rückwärtsbeschleunigen, oder} \\ 0 & \text{Gleiten} \quad . \end{cases}$$

Der Reward für das Ausführen einer Aktion wird hierbei definiert durch:

$$r_{t+1} = \begin{cases} 0 & \text{falls } x_{t+1} \geq 0.5 \text{ (Zielzustand erreicht),} \\ -1 & \text{ansonsten (Wegekosten)} \quad . \end{cases}$$

Jede Lernepisode startet in Zustand $s_{\text{tal}} = \{x = -0.5, \dot{x} = 0\}$, was dem tiefsten Punkt im Tal entspricht. Das Ende einer Episode ist dann erreicht, wenn entweder der Zielzustand auf dem rechten Berg erreicht ($x_{t+1} \geq 0.5$) oder eine maximale Anzahl von Schritten getätigt wurde. Die Länge einer optimalen Aktionssequenz, ausgehend von Zustand s_{tal} , ist hierbei ca. 125 Schritte.

5.5.1 Partielle Beobachtbarkeit

Wie man Gleichung (5.1) entnehmen kann, ist Mountain-Car ein Lernproblem mit zwei kontinuierlichen Zustandsdimensionen. Daraus folgt, dass es prinzipiell unendlich viele Zustände gibt, für welche die Q -Funktion approximiert werden muss. Um dieses Problem zu lösen, werden oftmals Funktionsapproximatoren zum Erlernen der Q -Funktion verwendet, wie

z. B. CMACs (Singh und Sutton, 1996; Sutton, 1996), RBF-Netze (Kretschmar und Anderson, 1997), selbstorganisierende Karten (Flentge, 2005) oder KD-Bäume (Vollbrecht, 2000, 2003). Bei der im Folgenden verwendeten Implementierung wird der Zustandsraum partiell beobachtbar, da die kontinuierlichen Zustandsdimensionen diskretisiert, von einer Tabelle approximiert werden (Bagnell u. a., 2001). In den diskreten Koordinaten kennt der Agent seine tatsächliche Position nicht exakt, sondern nur die Zelle, in der er sich momentan befindet. Der tatsächliche Zustand wird jedoch im Hintergrund weiterhin im Kontinuum, auf Basis von Gleichung (5.1), berechnet. Demnach kommt es vor, dass zum Verlassen eines Zustandes mehrmals dieselbe Aktion ausgeführt werden muss, um in den gewünschten Folgezustand zu gelangen. Als Folge können Fluktuationen in der Wertefunktion entstehen, welche z. B. negativ die VDBE-Strategie beeinflussen können.

5.5.2 Experimentelle Untersuchung

Die kontinuierlichen Zustandsdimensionen werden in Anlehnung an (Tokic und Palm, 2012a,b) in jeweils 100 equidistante Zustände diskretisiert, woraus sich eine Zustandsraumgröße von insgesamt 10000 Zuständen ergibt. Alle Zustands-Aktions-Werte werden zu Beginn des Experimentes mit $Q(s, a) = -200$ initialisiert. Der kumulierte Reward wird für verschiedene Belegungen des Explorationsparameters über jeweils 10000 Lernepisoden gemessen und die erhaltenen Ergebnisse über 2000 Experimente gemittelt. Jede Lernepisode beginnt in Zustand $s_{\text{tal}} = \{x = -0.5, \dot{x} = 0\}$. Die untersuchten Intervalle des Explorationsparameters sind: Explorationsrate $\epsilon \in [0.0, 1.0]$ für ϵ -Greedy; Temperatur $\tau \in [0.001, 1000.0]$ für Softmax; inverse Sensibilität $\sigma \in [0.001, 1000.0]$ sowie Lernrate $\delta(s) = 1/|\mathcal{A}(s)|$ für VDBE und VDBE-Softmax. Für VDBE-Softmax werden die Q -Werte in das Intervall $[-1, 1]$ normalisiert. Der Diskontierungsfaktor wird für Sarsa und Q -learning mit dem Wert $\gamma = 1.0$ belegt, hingegen die Lernrate mit $\alpha = 0.7$.

5.5.3 Ergebnisse

Die Ergebnisse des Mountain-Car-Experimentes sind in Abbildung A.5 und Tabelle 6.4 dargestellt. Analog zum Cliff-Walking-Problem stellt sich bei Mountain-Car ebenso heraus, dass die beste Strategie eine Greedy-Strategie ist. Diese konvergiert zu einem Return in Höhe von $R_{\text{greedy}} \approx -125$, eine Random-Strategie hingegen zu $R_{\text{random}} \approx -9400$. Bemerkenswerterweise kann Softmax in Kombination mit Sarsa sicherere Strategien erlernen, was dazu führt, dass der Return in Episode 20000 im Worst Ca-

se mit $R_{\text{softmax}}^{20000} \approx -3761$ besser ist. Deutlich besser sehen die Ergebnisse für die VDBE-Softmax-Strategie aus, bei welcher die Ergebnisse im Worst Case zu $R_{\text{vdbSoftmax}}^{20000} \approx -320$ konvergieren.

Return für Q -learning				
Strategie / Return	R_{1000}^{\max}	R_{1000}^{\min}	R_{20000}^{\max}	R_{20000}^{\min}
ε -Greedy	-256,16	-9166,06	-125,98	-9419,31
Softmax	-254,69	-9174,15	-127,00	-9278,68
VDBE	-259,12	-9135,70	-134,20	-9093,03
VDBE-Softmax	-259,96	-437,68	-129,17	-322,05
Return für Sarsa				
Strategie / Return	R_{1000}^{\max}	R_{1000}^{\min}	R_{20000}^{\max}	R_{20000}^{\min}
ε -Greedy	-258,88	-9135,14	-126,51	-9083,35
Softmax	-256,11	-8911,23	-126,39	-3761,51
VDBE	-263,86	-9256,20	-202,92	-9288,13
VDBE-Softmax	-255,20	-416,11	-141,94	-319,14

Tabelle 5.5: Mountain-Car-Problem: Return in den Episoden 1000 und 20000 über 2000 Experimente gemittelt.

5.6 Diskussion

In diesem Kapitel wurden Experimente an Lernproblemen mit unterschiedlichen Lernproblemdimensionen durchgeführt. Hierfür wurden zunächst mögliche Problemdimensionen diskutiert und erfolgreich gelöste Lernprobleme aus der Literatur kategorisiert. In den untersuchten Lernproblemen stellt sich heraus, dass für $t \rightarrow \infty$ der Reward einer VDBE-Strategie in Kombination mit Q -learning mindestens dem Reward einer Greedy-Strategie entsprechen kann. Hierfür ist jedoch als Bedingung notwendig, dass es sich beim Lernverfahren der Wertefunktion um ein off-policy-Verfahren handelt und die Rewards der Umgebung deterministisch sind, da ansonsten eine unerwünschte Explorationsrate von $\varepsilon > 0$ entstehen kann.

Ferner wurde gezeigt, dass mit VDBE-Strategien eine nahezu optimale Strategie erlernt werden kann. Dabei stellt sich heraus, dass es wichtig ist, die inverse Sensibilität σ entsprechend niedrig zu wählen, um mit ausreichender Empfindlichkeit das Verhältnis zwischen Exploration und Exploitation zu steuern. Außerdem wird deutlich, dass die Verwendung eines off-policy-Verfahrens bei VDBE-Strategien signifikante Reward-Verbesserungen einbringt. Die Ergebnisse zeigen in diesem Sinne

die Robustheit des Explorationsparameters auf (Tokic u. a., [2012](#)), welcher für bestimmte Parameterbereiche von ε -Greedy und Softmax einen deutlich niedrigeren Worst-Case-Return zur Folge haben kann.

Kapitel 6

Parameteradaption mit stochastischen Neuronen

Bislang wurde untersucht, wie die Explorationsrate der ε -Greedy-Strategie auf Basis des Lernfortschritts adaptiert werden kann. Die hierbei entwickelten VDBE-Strategien besitzen jedoch ebenso einen Explorationsparameter, der beim Lernen über Erfolg oder Misserfolg entscheidet. Dabei stellte sich in den durchgeführten Experimenten heraus, dass dieser Meta-Parameter für eine VDBE-Softmax-Strategie ein wenig empfindliches Verhalten gegenüber Fehlbelegungen aufzeigt, im Gegensatz zu Strategien wie ε -Greedy und Softmax. Ein weiterer Nachteil von VDBE-Strategien ist jedoch, dass für jeden Zustand eine lokale Explorationsrate $\varepsilon(s)$ approximiert werden muss. Insbesondere stellt dies für große Zustandsräume ein Problem dar, analog zur Approximation der Wertefunktion. Genau aus diesem Grund werden Strategien wie ε -Greedy und Softmax, die keinen zusätzlichen Speicher benötigen, bevorzugt in der Praxis eingesetzt. Sie erzielen dabei oftmals mit einer ad-hoc gewählten Explorationsrate von z. B. $\varepsilon = 0.1$ gute Ergebnisse, wofür es jedoch keine Garantie gibt. Von daher stellt sich natürlich die Frage, ob es nicht möglich ist, das Explorationsverhalten ebenso global zu adaptieren? Tatsächlich ist dies möglich, wie im Folgenden gezeigt wird. Als Weiterentwicklung der VDBE-Strategien wird hierbei nicht mehr die Konvergenz der Wertefunktion betrachtet, sondern die Qualität ρ von Strategie π in Abhängigkeit des momentanen Explorationsparameters. Hierbei wird das Ziel verfolgt, ρ über ein Gradientenverfahren zu maximieren.

Im Folgenden werden Verfahren auf Basis von stochastischen Neuronen beschrieben, welche den Explorationsparameter entweder lokal oder global steuern. Das Ziel ist es, den Explorationsparameter einer *Basisstrategie* (ε -Greedy, Softmax, MBE oder VDBE-Softmax) zu adaptieren. Hierbei ist es möglich Zeitpunkte zu erkennen, zu welchen es sinnvoll ist zu explorieren oder das bislang erlernte Wissen auszunutzen. Das vorgeschlagene Verfahren wird in zwei Experimenten evaluiert.

6.1 Das Lernproblem

Aus Sicht des Reinforcement Learning muss eine kontinuierliche Aktion a_e gefunden werden, die nicht direkt vom Agenten in seiner Umgebung ausgeführt wird, sondern der Belegung des kontinuierlichen Explorationsparameters entspricht (z. B. für ε -Greedy: $a_e \equiv \varepsilon$). Die Belohnung ρ , welche der Agent für die Wahl von a_e erhält, kann bei episodischen Lernproblemen als der Return R einer Lernepisode gemessen werden:

$$\rho = E\{r_1 + r_2 + \dots + r_T | \pi(a_e, \cdot, \cdot)\} = R . \quad (6.1)$$

Für das Erlernen von kontinuierliche Aktionen werden in der RL-Literatur die sogenannten *Strategie-Gradientenverfahren* bzw. *Aktor-Kritiker-Ansätze* (Flentge, 2005; Röttger, 2009; Sutton u. a., 2000; Williams, 1992) verwendet. Das Lernen von Zustands-Aktions-Werten $Q(s, a)$ ist für kontinuierliche Aktionen nicht möglich, da praktisch unendlich viele Aktionswerte pro Zustand gelernt werden müssten. Demnach kann die Strategie auch nicht mehr von einer Wertefunktion abgeleitet werden, sondern wird stattdessen explizit durch einen Parametervektor θ repräsentiert, z. B. durch die Gewichte eines neuronalen Netzwerkes. Hierbei ist ρ ein Maß über die Qualität von Strategie π , z. B. der durchschnittliche Reward pro Aktion oder der kumulierte Reward einer Lernepisode. Kontinuierliche Aktionen werden gelernt, indem der Parametervektor θ durch ein Gradientenverfahren dahingehend angepasst wird ρ in Zukunft zu verbessern:

$$\theta_{t+1} \approx \theta_t + \alpha \nabla_{\theta} \rho , \quad (6.2)$$

wobei α eine positiv zu wählende Lernrate ist.

Eine Möglichkeit zum Lernen von kontinuierlichen Aktionen bieten die von Williams (1992) entwickelten REINFORCE¹-Algorithmen, welche hierfür ein berechenbares Modell eines stochastischen Neurons verwenden. Es wird das Ziel verfolgt die Belohnung ρ zu optimieren, ohne den Gradienten $\nabla_{\theta}\rho$ explizit berechnen zu müssen. REINFORCE-Algorithmen verwenden nur den unmittelbaren Reward r_{t+1} , können jedoch auch in Kombination mit einer *Aktor-Kritiker*-Architektur auf Lernprobleme mit verzögertem Reward angewandt werden (Flentge, 2005). Hierbei repräsentiert der *Aktor* die Strategie π des Agenten, hingegen lernt der *Kritiker* eine Zustandswertefunktion, durch welche die Strategie des *Aktors* bewertet wird. Für die folgende Untersuchung ist jedoch die Betrachtung des unmittelbaren Rewards ausreichend, da der Return einer Lernepisode als unmittelbarer Reward angesehen wird, für welchen die gewählte Aktion, der Explorationsparameter einer Lernepisode, verantwortlich ist.

Im Folgenden wird nun gezeigt, wie der Explorationsparameter a_e für episodische Lernprobleme global auf Basis des kumulierten Rewards adaptiert werden kann (Tokic und Palm, 2012a). Dieses Vorgehen ist für Lernprobleme von Vorteil, die aus einer kleinen Menge von Startzuständen bestehen. Anschließend wird eine lokale Strategie abgeleitet, bei welcher für jeden Zustand $s \in \mathcal{S}$ ein lokaler Explorationsparameter $a_e(s)$ gelernt wird (Tokic und Palm, 2012b). Hierdurch ist es möglich nur in Zuständen explorativ zu sein, in welchen tatsächlich Verbesserungspotential besteht. Die episodische Strategie ist jedoch aufgrund ihrer Speicher- sowie Recheneffizienz von besonderem Interesse, insbesondere für episodische Lernprobleme mit einer kleinen Menge von Startzuständen; oder sogar nur einem, wie bei vielen Brettspielen.

6.2 Globale Adaption

Eine interessante Anwendung der REINFORCE-Algorithmen sind Lernverfahren, welche die zu tätige Aktion anhand einer stochastischen Einheit, einem sogenannten *stochastischen Neuron* (vgl. Abbildung 6.1), bestimmen (Williams, 1992). Grundlegend wird hierfür eine differenzierbare Normalverteilung verwendet, anhand welcher die auszuführende Aktion a_e bestimmt wird (die neuronale Aktivierungsfunktion):

$$a_e \sim \mathcal{N}(\mu, \sigma) \quad , \quad (6.3)$$

¹REINFORCE ist ein Akronym der Lernregel: **RE**ward **I**ncrement = **N**onnegative **F**actor \times **O**ffset **R**einforcement \times **C**haracteristic **E**ligibility.

deren Wahrscheinlichkeitsdichte $g(a_e, \mu, \sigma)$ vom Mittelwert μ sowie die Standardabweichung σ abhängig ist:

$$g(a_e, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(a_e - \mu)^2 / 2\sigma^2} . \quad (6.4)$$

Zur besseren Lesbarkeit wird im Folgenden die Notation über Zustände bzw. Eingabevektoren weggelassen. Prinzipiell ist es aber möglich mehrere solcher Einheiten in einem Netzwerk miteinander zu verbinden, mit Kompatibilität zum Backpropagation-Lernverfahren (Williams, 1992).

Der Parametervektor einer Normalverteilung, dessen Komponenten per REINFORCE adaptiert werden sollen, setzt sich aus dem Mittelwert μ sowie der Standardabweichung σ zusammen:

$$\theta = \begin{pmatrix} \mu \\ \sigma \end{pmatrix} . \quad (6.5)$$

Die Ausführung von Aktion a_e führt zu einer unmittelbaren Belohnung $\rho \in \mathbb{R}$, auf deren Basis die i -te Vektorkomponente wie folgt adaptiert wird:

$$\Delta\theta_i = \alpha_i(\rho - \bar{\rho}_i)e_i , \quad (6.6)$$

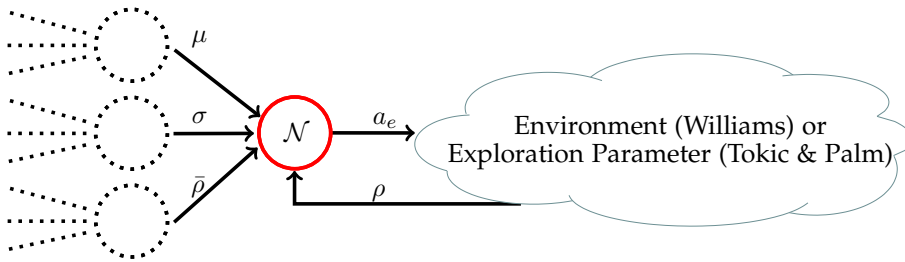


Abbildung 6.1: Modell eines stochastischen REINFORCE-Neurons im Verbund eines neuronalen Netzwerkes. Die Eingabe des Neurons kann anhand von vorgeschalteten Neuronen (mit evtl. andersartiger Aktivierungsfunktion) bestimmt werden, oder durch tabellarische Speicherung. Die Ausgabe (Aktivierung) des Neurons wird anhand einer Normalverteilung bestimmt, $a_e \sim \mathcal{N}(\mu, \sigma)$, und als kontinuierliche Aktion in der Umgebung (Williams, 1992) oder als Explorationsparameter (Tokic und Palm, 2012a,b) ausgeführt. Die Umgebung liefert die Bestärkung in Form von ρ zurück, wodurch die Parameter μ und σ , auf Basis von $\bar{\rho}$ und ρ , nach REINFORCE adaptiert werden.

wobei α_i eine positiv zu wählende Lernrate ist (der *Nonnegative Factor*). Die *Reinforcement Baseline* $\bar{\rho}_i$ ist ein Vergleichswert, welcher beispielsweise der mittleren Belohnung von bislang gewählten Aktionen im selbigen Zustand entspricht. Die *Characteristic Eligibility* e_i gibt hingegen an, wie stark die Vektorkomponente θ_i Einfluss auf ρ besitzt, was über die erste partielle Ableitung näherungsweise geschätzt werden kann:

$$e_i = \frac{\partial \ln g(a_e, \mu, \sigma)}{\partial \theta_i} . \quad (6.7)$$

Diese sind für eine Normalverteilung:

$$\frac{\partial \ln g(a_e, \mu, \sigma)}{\partial \mu} = \frac{a_e - \mu}{\sigma^2} \quad (6.8)$$

$$\frac{\partial \ln g(a_e, \mu, \sigma)}{\partial \sigma} = \frac{(a_e - \mu)^2 - \sigma^2}{\sigma^3} , \quad (6.9)$$

woraus sich in Kombination mit Gleichung (6.6) ein Verfahren zur Anpassung von μ und σ ableiten lässt:

$$\Delta\mu = \alpha_\mu (\rho - \bar{\rho}_\mu) \frac{a_e - \mu}{\sigma^2} \quad (6.10)$$

$$\Delta\sigma = \alpha_\sigma (\rho - \bar{\rho}_\sigma) \frac{(a_e - \mu)^2 - \sigma^2}{\sigma^3} . \quad (6.11)$$

Dem Vorschlag von Williams folgend kann $\alpha_\mu = \alpha_\sigma = \alpha\sigma^2$ verwendet werden. Die Baseline $\bar{\rho}_\mu = \bar{\rho}_\sigma = \bar{\rho}$ wird analog zu einem Reinforcement-Comparison-Schema (Kapitel 3.4) adaptiert:

$$\bar{\rho} = \bar{\rho} + \alpha_\rho (\rho - \bar{\rho}) . \quad (6.12)$$

Zusammengefasst führt dies zu einer Vereinfachung von Gleichungen (6.10) und (6.11):

$$\Delta\mu = \alpha (\rho - \bar{\rho}) (a_e - \mu) \quad (6.13)$$

$$\Delta\sigma = \alpha (\rho - \bar{\rho}) \frac{(a_e - \mu)^2 - \sigma^2}{\sigma} . \quad (6.14)$$

Analytisch betrachtet wird in Gleichung (6.13) der Mittelwert μ mehr in Richtung der getätigten Aktion a_e verschoben, falls der Return einer Episode höher ist als die bisherige Baseline $\rho > \bar{\rho}$ (vgl. Abbildung 6.2). Im Gegensatz dazu wird μ in die entgegengesetzte Richtung verschoben, falls ρ niedriger ist als durch $\bar{\rho}$ erwartet. Die Standardabweichung σ wird in Gleichung (6.14) entsprechend angepasst.

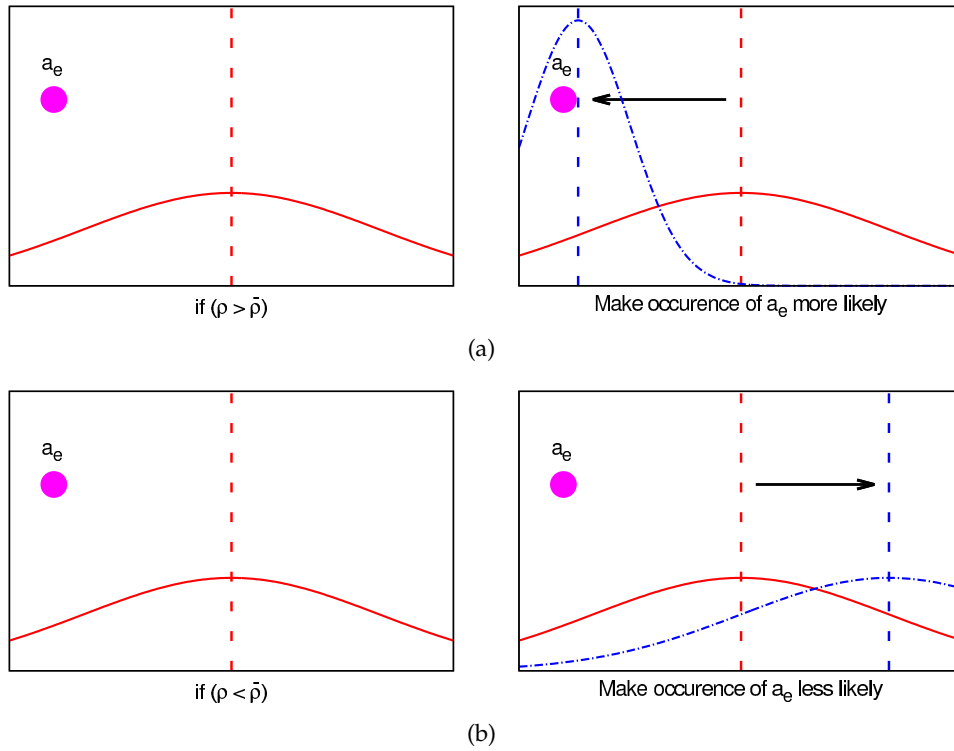


Abbildung 6.2: Beispiel einer REINFORCE-Adaptierung: (a) Anpassung der Normalverteilung in Richtung von a_e , falls $\rho > \bar{\rho}$. (b) Anpassung der Normalverteilung in entgegengesetzter Richtung, falls $\rho < \bar{\rho}$.

chung (6.14) derartig adaptiert, dass die Auswahlwahrscheinlichkeit für a_e steigt, falls ρ höher ist als durch $\bar{\rho}$ erwartet. Analog wird die Auswahlwahrscheinlichkeit für a_e reduziert, falls $\rho < \bar{\rho}$. In diesem Sinne ist die Standardabweichung sozusagen für die Exploration im Raum von a_e zuständig. Schlussendlich gilt für alle Algorithmen in Form von Gleichung (6.6):

Satz (REINFORCE). Für jeden REINFORCE-Algorithmus ist das Skalarprodukt $E\{\Delta\theta|\theta\} \cdot \nabla_{\theta} E\{\rho|\theta\} \geq 0$.

Falls $\alpha_i > 0$ gilt für alle i : $E\{\Delta\theta|\theta\} \cdot \nabla_{\theta} E\{\rho|\theta\} = 0 \implies \nabla_{\theta} E\{\rho|\theta\} = 0$.

Ist $\alpha_i = \alpha$ konstant, so gilt: $E\{\Delta\theta|\theta\} = \alpha \nabla_{\theta} E\{\rho|\theta\}$

Beweis: siehe (Williams, 1992). □

Dies bedeutet, dass der durchschnittliche Anpassungsvektor $\Delta\theta$ in Richtung des Gradienten der Performanz zeigt, wodurch diese anhand der Gewichtsänderungen durchschnittlich verbessert wird. Es bleibt zu erwähnen, dass es sich bei REINFORCE um ein Gradientenverfahren handelt, bei

welchem die Performanz auch in einem lokalen Maximum konvergieren kann.

Eine effiziente Arbeitsweise der REINFORCE-Strategie hängt im Wesentlichen von zwei Faktoren ab. Um die Suche nach sinnvollen Parametern einzuschränken, müssen der Explorationsparameter a_e , der Mittelwert μ sowie die Standardabweichung σ durch Minimal- und Maximalwerte begrenzt werden (siehe Tabelle 6.1). Darüber hinaus muss für Lernprobleme mit mehreren Startzuständen für jeden auftretenden Startzustand die Parameter μ , σ und $\bar{\rho}$ separat gelernt werden ($\mu \rightarrow \mu(s)$, $\sigma \rightarrow \sigma(s)$ und $\bar{\rho} \rightarrow \bar{\rho}(s)$), da Wegekosten ρ ungleich beeinflussen können. Oftmals haben Lernprobleme nur einen Startzustand, wie etwa Othello, Dame oder die untersuchten Lernprobleme im Folgenden. In solch einem Fall ist es möglich, die Parameter einer REINFORCE-Strategie global zu approximieren.

	ε -Greedy $\pi(\varepsilon, \cdot, \cdot)$	MBE $\pi(\varepsilon, \cdot, \cdot)$	Softmax $\pi(\tau, \cdot, \cdot)$	VDBE-Softmax $\pi(\sigma_{\text{vdb}}, \cdot, \cdot)$
$a_e^{\min}; \mu_{\min}$	0.0	0.0	0.001	0.001
$a_e^{\max}; \mu_{\max}$	1.0	1.0	1000.0	1000.0
μ_{init}	0.0	0.0	0.001	1000.0
σ_{\min}	0.001	0.001	0.1	0.1
σ_{\max}	5.0	5.0	5000.0	5000.0
σ_{init}	0.001	0.001	0.1	0.1

Tabelle 6.1: Verwendete Minimal-, Maximal- und Initialwerte für REINFORCE-Strategien.

6.3 Lokale Adaption

Die Ergebnisse der globalen Adaption können einfach erweitert werden, um ebenso eine lokale Explorationsstrategie zu erhalten (Tokic und Palm, 2012b). Hierdurch ist es möglich sich nur in Zuständen explorativ zu verhalten, in denen Verbesserungspotential existiert. Generell werden alle relevanten Parameter lokal einem Zustand zugeordnet ($\mu \rightarrow \mu(s)$, $\sigma \rightarrow \sigma(s)$ und $\bar{\rho} \rightarrow \bar{\rho}(s)$) und müssen entsprechend approximiert werden. Der Mittelwert sowie die Standardabweichung werden nach jeder vom Agenten getätigten Aktion durch den verbesserten Q -Wert bestärkt.

Vor dem Wählen der vom Agenten zu tätigenen Aktion a_t in Zustand s_t wird der Explorationsparameter a_e durch die zustandsabhängige Normalverteilung bestimmt:

$$a_e \sim \mathcal{N}(\mu(s_t), \sigma(s_t)) \quad . \quad (6.15)$$

Durch diesen bestimmt die Basisstrategie die vom Agenten zu tätige Aktion a_t , nach deren Ausführung der unmittelbare Reward r_{t+1} von der Umgebung erhalten wird. Für REINFORCE kann nun der Nutzen des Explorationsparameters aus r_{t+1} bestimmt werden, was jedoch für Lernprobleme mit verzögertem Reward unzureichend ist. Aus diesem Grunde sollte auch der geschätzte zukünftige Reward mitberücksichtigt werden, welcher aus der parallel zu erlernenden Wertefunktion abgeleitet werden kann (Flentge, 2005):

$$\begin{aligned} \rho &= \gamma \arg \max_{b \in \mathcal{A}(s_{t+1})} Q(s_{t+1}, b) + r_{t+1} \\ &= \gamma V(s_{t+1}) + r_{t+1} \quad . \end{aligned}$$

Bei Verwendung von Sarsa oder Q -learning kann hierfür auch die Schätzung der getätigten Aktion, nach Anwendung der entsprechenden Lernregel, verwendet werden:

$$\rho = Q^n(s_t, a_t) \quad (6.16)$$

Nun werden die Parameter der Normalverteilung in Zustand s_t analog zu den Gleichungen (6.13)+(6.14) angepasst:

$$\Delta\mu(s_t) = \alpha(\rho - \bar{\rho}(s_t))(a_e - \mu(s_t)) \quad (6.17)$$

$$\Delta\sigma(s_t) = \alpha(\rho - \bar{\rho}(s_t)) \frac{(a_e - \mu(s_t))^2 - \sigma(s_t)^2}{\sigma(s_t)} \quad , \quad (6.18)$$

sowie schlussendlich auch die lokale Baseline analog zu Gleichung (6.12):

$$\bar{\rho}(s_t) = \bar{\rho}(s_t) + \alpha_\rho(\rho - \bar{\rho}(s_t)) \quad . \quad (6.19)$$

6.4 Das Dynamic-Cliff-Problem

Das *Dynamic-Cliff-Problem* ist eine Erweiterung des Cliff-Walking-Problems, um Lernverhalten in nichtstationären Umgebungen zu untersuchen (Tokic und Palm, 2012a,b; Tokic u. a., 2012). Die Umgebung ist in drei zeitliche Phasen (a)-(c) unterteilt, mit jeweils unterschiedlicher Klippenlänge in Abhängigkeit der aktuellen Lernepisode (vgl. Abbildung 6.3).

Das Ziel des Agenten ist es, ausgehend vom Startzustand S , einen der beiden Zielzustände G_1 oder G_2 zu erreichen, wobei G_2 nur in Phase (c) aktiv ist. Fällt der Agent während der Suche nach dem Zielzustand die Klippe herunter, so erhält er einen hoch negativen Reward in Höhe von $r_{t+1} = -100$ und wird zusätzlich, analog zum Cliff-Walking-Problem, in den Startzustand S zurückversetzt. Für jede getätigte Aktion erhält der Agent einen Reward in Höhe von $r_{t+1} = -1$, was in der Summe den Wegekosten entspricht. Aktionen, die zur Erreichung des Zielzustandes führen, werden mit einem höheren Reward belohnt (vgl. Tabelle 6.2). Eine Lernepisode terminiert sobald ein Zielzustand erreicht wird oder die maximale Anzahl von Schritten pro Episode $T_{\max} = 200$ getätigt wurde.

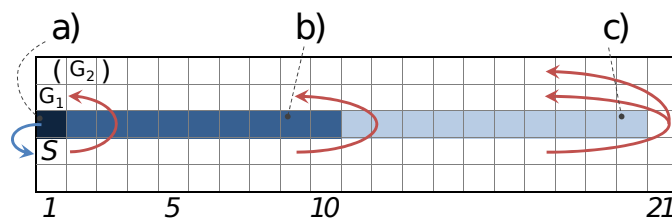


Abbildung 6.3: Dynamic-Cliff-Problem: Schematische Darstellung.

	Episode 0 ... 199	Episode 200 ... 999	Episode 1000 ... 2999
r_{G1}	3	21	41
r_{G2}	n/a	n/a	500

Tabelle 6.2: Dynamic-Cliff-Problem: Rewards für die Erreichung eines Zielzustandes.

6.4.1 Experimentelle Untersuchung

In Phase (a) ist nur ein Klippenelement vorhanden, welches sich zwischen dem Startzustand S und dem Zielzustand G_1 befindet. Der kürzeste Pfad zum Ziel ist, von S ausgehend einen Schritt nach rechts, zwei Schritte nach oben sowie einen Schritt nach links zu tätigen. Nach 200 Lernepisoden verändert sich die Welt, wie in Phase (b) skizziert. Die Klippe ist nun länger (von $x = 1$ bis $x = 10$) und schneidet den kürzesten Pfad aus Phase (a). Nach weiteren 800 Lernepisoden ändert sich die Welt, wie in Phase (c) skizziert. Gleichzeitig wird der zweite Zielzustand G_2 aktiv, dessen Erreichung einen viel höheren Reward einbringt als Zielzustand G_1 . Wie die folgenden Ergebnisse aufzeigen, finden Greedy-Strategien in Phase (c) relativ selten den Zielzustand G_2 , da dieser in den Phasen (a)-(b) nicht aktiv war. Wäh-

rend des Experimentes wird eine Lernrate in Höhe von $\alpha = 0.2$ verwendet sowie ein Diskontierungsfaktor in Höhe von $\gamma = 1.0$. Für REINFORCE adaptierte Basisstrategien befinden sich die Parameter a_e , σ und μ innerhalb der angegebenen Grenzen aus Tabelle 6.1.

6.4.2 Ergebnisse

Die gemittelten Reward-Ergebnisse über 500 Durchläufe sind in den Abbildungen 6.4 und A.7(a) sowie in Tabelle 6.3 dargestellt. Hingegen ist die Ausgabe des stochastischen Neurons in Abbildung 6.5 dargestellt, welche das adaptive Explorationsverhalten veranschaulicht. Hieraus ist zu entnehmen, dass im Falle von Umgebungsänderungen die Explorationsrate ansteigt (Episode 200 und 1000), wodurch ermöglicht wird das bisher erlernte Verhalten auf die veränderte Umgebung neu anzupassen.

Die Ergebnisse der globalen Adaption zeigen auf, dass VDBE-Softmax den Return durch REINFORCE maximiert, dafür jedoch das separate Speichern von $\varepsilon(s)$ benötigt. Von den verbleibenden drei Basisstrategien hat sich MBE als am performantesten erwiesen. Für ε -Greedy und Softmax ist hingegen eine Tendenz zu negativem Reward in den ersten Lernepisoden wahrnehmbar, welche jedoch mit lokaler Adaption deutlich verbessert werden kann. Ebenso zeigt Sarsa für alle vier Strategien bessere Ergebnisse bei globaler Adaption, da das Agentenverhalten eher dazu tendiert den entfernten Pfad zur Klippe zu lernen, wodurch in Phase (c) der Zustand G_2 häufiger erreicht wird. Im Gegensatz dazu wird das lokale Verfahren zu schnell Greedy, wodurch der Agent den zweiten Zielzustand in Phase (c) seltener entdeckt. Zusammengefasst sind in Episode 3000 alle REINFORCE-Strategien deutlich besser als die einer Greedy-Strategie, welche stets zu einem Return in Höhe von $R_{\text{greedy}} = 0$ konvergiert. Im Gegensatz dazu konvergiert eine Random-Strategie gegen $R_{\text{random}} \approx -2750$.

6.5 Das Mountain-Car-Problem mit zwei Zielzuständen

Das *Mountain-Car-Problem mit zwei Zielzuständen* (Tokic und Palm, 2012a,b) ist eine Erweiterung des in Kapitel 5.5 vorgestellten Mountain-Car-Problems, da in der ursprünglichen Problembeschreibung eine Greedy-Strategie zu optimalen Ergebnissen führt, falls stets im Tal gestartet wird. Möchte man unterschiedliches Lernverhalten in Abhängigkeit von Explo-

Return bei globaler Parameteradaption

B.-Strategie / Episode	Q-learning			Sarsa		
	200	1000	3000	200	1000	3000
ϵ -Greedy	-0,01	-91,13	175,37	-21,82	-67,85	287,01
Softmax	-0,03	-66,09	28,17	-25,51	-7,22	271,79
MBE	-0,47	-0,10	242,09	-0,90	-5,52	344,75
VDBE-Softmax	0,00	-0,91	396,31	-0,81	-5,50	486,02

Return bei lokaler Parameteradaption

B.-Strategie / Episode	Q-learning			Sarsa		
	200	1000	3000	200	1000	3000
ϵ -Greedy	-0,82	-2,56	341,12	-1,35	-7,59	201,52
Softmax	-0,02	-0,06	78,12	-0,26	-2,51	91,29
MBE	-2,20	-2,95	317,25	-0,75	-4,95	193,37
VDBE-Softmax	0	-0,19	41,73	-0,19	-3,27	104,49

Tabelle 6.3: Dynamic-Cliff-Problem: Return in den Lernepisoden 200, 1000 und 3000 über 500 Experimente gemittelt.

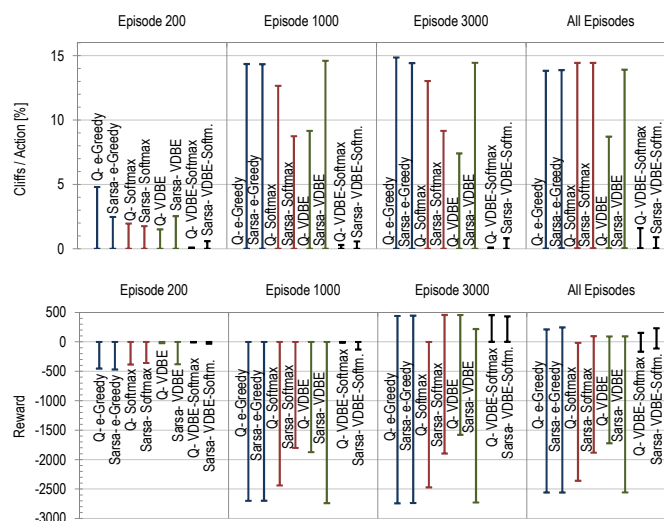


Abbildung 6.4: Dynamic-Cliff-Problem: Reward- sowie Klippenabsturz-Bandbreite in Abhängigkeit des untersuchten Parameterbereichs (Tokic u. a., 2012), für konstante Belegungen des Explorationsparameters (ohne REINFORCE).

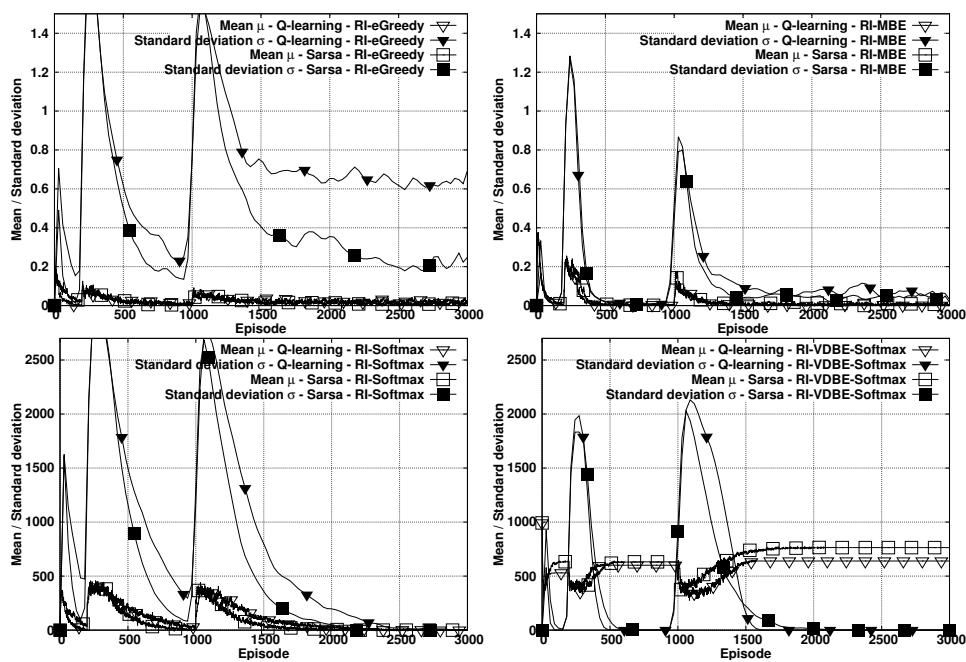


Abbildung 6.5: Dynamic-Cliff-Problem: Entwicklung des Mittelwerts und der Standardabweichung für durch REINFORCE adaptierte Basisstrategien. Man beachte die Dynamik der Exploration zum Zeitpunkt von Nichtstationaritäten.

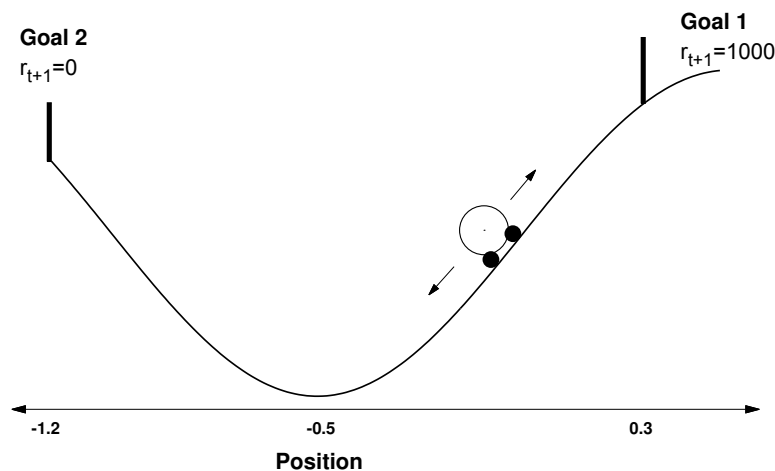


Abbildung 6.6: Mountain-Car-Problem mit zwei Zielzuständen: Skizze.

ration und Exploitation erhalten, so kann man das Lernproblem durch drei einfache Modifikationen wie folgt erweitern:

1. Die Positionsgrenzen werden auf $[-1.2, 0.3]$ begrenzt.
2. Man verwendet zwei Zielzustände, wie in Abbildung 6.6 dargestellt, wobei die Erreichung des Ziels auf dem linken Berg ($x = -1.2$) mit $r_{t+1} = 0$ belohnt wird. Hingegen wird die Erreichung des Zielzustandes auf dem rechten Berg ($x = 0.3$) mit $r_{t+1} = 1000$ bestärkt, was folglich das höherwertige Ziel ist.
3. Die Menge von möglichen Aktionen wird auf insgesamt sieben Aktionen erweitert:

$$\mathcal{A}(s) = \{-1, -0.66, -0.33, 0, 0.33, 0.66, 1\} \quad . \quad (6.20)$$

6.5.1 Experimentelle Untersuchung

In jeder Lernepisode startet das Auto am tiefsten Punkt im Tal in Zustand $s_{\text{tal}} = \{x = -0.5, \dot{x} = 0\}$. Gemessen wird das Lernverhalten über 20000 Episoden, in welchen der Agent maximal $T_{\text{max}} = 10000$ Aktionen pro Episode tätigen kann, bevor sie künstlich terminiert. Gemittelt werden die Ergebnisse über 200 Durchläufe, wobei zu Beginn eines Experimentes alle Zustands-Aktions-Werte mit $Q(s, a) = -200$ initialisiert werden. Als Lernrate wird für REINFORCE $\alpha = 0.7$ verwendet, ebenso auch für Q -learning

Q-learning mit REINFORCE				
Strategie / Return	R_{1000}^{Local}	R_{1000}^{Global}	R_{20000}^{Local}	$R_{20000}^{\text{Global}}$
ε -Greedy	-251,52	-1926,50	632,83	-658,00
MBE	-164,63	-241,02	670,75	548,31
Softmax	-220,79	-1514,62	682,73	-1049,32
VDDBE-Softmax	-107,60	-204,68	433,87	413,61
Sarsa mit REINFORCE				
Strategie / Return	R_{1000}^{Local}	R_{1000}^{Global}	R_{20000}^{Local}	$R_{20000}^{\text{Global}}$
ε -Greedy	-265,69	-1396,63	-193,41	-657,74
MBE	-140,58	-259,10	-57,21	409,62
Softmax	-204,06	-933,03	-127,62	-133,50
VDDBE-Softmax	-142,04	-229,05	-103,48	-171,62
Q-learning ohne REINFORCE				
Strategie / Return	R_{1000}^{max}	R_{1000}^{min}	R_{20000}^{max}	R_{20000}^{min}
ε -Greedy	-140,01	-5120,02	794,06	-5603,16
MBE	-129,56	-563,88	799,98	-1245,07
Softmax	-96,03	-5424,20	178,89	-5346,07
VDDBE-Softmax	-168,39	-587,68	810,05	-1292,00
Sarsa ohne REINFORCE				
Strategie / Return	R_{1000}^{max}	R_{1000}^{min}	R_{20000}^{max}	R_{20000}^{min}
ε -Greedy	-121,78	-4983,91	352,80	-5791,45
MBE	-108,31	-410,64	262,44	-309,16
Softmax	-120,98	-4820,37	745,43	-3009,02
VDDBE-Softmax	-139,66	-471,08	301,31	-312,57

Tabelle 6.4: Mountain-Car-Problem mit zwei Zielzuständen: Return in den Episoden 1000 und 20000 über 200 Experimente gemittelt.

bzw. Sarsa. Bei durch REINFORCE adaptierten Basisstrategien befinden sich die Parameter a_e , σ und μ innerhalb der angegebenen Grenzen laut Tabelle 6.1.

6.5.2 Ergebnisse

Die Reward-Ergebnisse sind in Abbildung A.7(b) und Tabelle 6.4 dargestellt. Der Vergleich ohne REINFORCE-basierter Parameteradaption zeigt auf, dass eine Greedy-Strategie zu einem durchschnittlichen Return in Höhe von $R_{\text{greedy}} \approx 114$ konvergiert. Verglichen hierzu konvergiert eine

Random-Strategie zu $R_{\text{random}} \approx -5800$. Ebenso kann man erkennen, dass entgegen den Ergebnissen vom ursprünglichen Mountain-Car-Problem (vgl. Abschnitt 5.5) eine Greedy-Strategie nicht mehr zu optimalen Ergebnissen führt, sondern diese von der Steuerung zwischen Exploration und Exploitation signifikant abhängig sind. Für ε -Greedy und MBE werden hierfür niedrige Belegungen von $\varepsilon \gtrsim 0$ benötigt, hingegen für VDBE-Softmax hohe inverse Sensibilitäten von $\sigma \approx 10$. Bemerkenswerterweise ist es für Softmax schwierig geeignete Temperaturen zu finden. Die besten Ergebnisse werden für $\tau = 100$ mit Sarsa erzielt, was erneut darauf hindeutet, wie schwierig es ist geeignete Temperaturen τ für ein beliebiges Lernproblem zu finden.

Die besten Ergebnisse mit REINFORCE basierter Parameteradaption werden für Q -learning in Kombination mit ε -Greedy, MBE und Softmax erzielt. Für Sarsa hingegen ist lediglich die globale Adaption von MBE von Vorteil. Das Lernverhalten von VDBE-Softmax liefert nur in den ersten Episoden gute Ergebnisse, welche in denen von MBE ähnlich sind. Auf lange Sicht jedoch erhält man für MBE die besseren Ergebnisse, was darauf zurückzuführen ist, dass der Explorationsparameter von VDBE-Softmax über eine zusätzliche Ebene adaptiert wird, obwohl beide Verfahren auf der grundlegend gleichen Aktionsauswahlstrategie basieren. REINFORCE steuert hierbei die inverse Sensibilität σ , welche von VDBE-Softmax dazu verwendet wird die zustandsabhängige Explorationsrate $\varepsilon(s)$ auf Basis des TD-Fehlers zu adaptieren. Zu Beginn eines Experimentes ist oftmals eine Schwankung im Reward wahrnehmbar, die darauf zurückzuführen ist, dass zunächst ein Pfad zum linken, schlechteren Ziel gelernt wird, welcher anschließend jedoch zum rechten Ziel umgelernt wird. Interessanterweise sind derartige Schwankungen für lokale REINFORCE-Strategien mit Sarsa nicht wahrnehmbar. Dieses Verhalten verdeutlicht, dass Sarsa eher dazu geneigt ist sichere Strategien zu lernen, welche in diesem Fall eher zum linken Ziel führen und geringere Wegekosten haben. Für die Erreichung des rechten Zieles sind diese nämlich zunächst höher, da der Startzustand weiter entfernt ist.

6.6 Diskussion

In diesem Kapitel wurde der Einsatz von Gradientenverfahren zur Steuerung eines Explorationsparameters untersucht. Die Basis ist der REINFORCE-Algorithmus von Williams (1992), um einen kontinuierlichen

Explorationsparameter anhand eines stochastischen Neurons zu repräsentieren und zu adaptieren.

Im Kern wurden zwei Variationen untersucht. Zum Einen ist dies ein speichereffizientes, globales Verfahren, welches für Lernprobleme mit einer geringen Menge an Startzuständen eingesetzt werden kann, da Explorationsdaten nur für Startzustände gespeichert werden (Tokic und Palm, 2012a). Zum Anderen wurde ein lokales Verfahren untersucht (Tokic und Palm, 2012b), welches für jeden Zustand einen separaten Explorationsparameter auf Basis der zugrunde liegenden Wertefunktion lernt, analog zu den VDBE-Strategien.

Die durchgeführten Experimente in zwei unterschiedlichen Problemklassen verdeutlichen die Funktionsweise der vorgestellten Verfahren. Hierbei wurde anhand des *Dynamic-Cliff-Problems* gezeigt, wie das Explorationsverhalten eines Lernagenten dynamisch adaptiert wird, insbesondere in nichtstationären Prozessen. Ferner wurde im *Mountain-Car-Problem mit zwei Zielzuständen* gezeigt, dass das Verfahren auch für Reinforcement Learning mit kontinuierlichen Zustandsdimensionen verwendet werden kann. Über beide Experimente betrachtet stellt sich die Adaption des Explorationsparameters der MBE-Strategie als am effizientesten heraus. Die Adaption des Explorationsparameters von VDBE-Softmax kann Vorteile bringen, was am Dynamic-Cliff-Problem gezeigt wurde. Diese Aussage ist jedoch nicht verallgemeinerbar, wie die Ergebnisse des *Mountain-Car-Problems mit zwei Zielzuständen* aufzeigen.

Kapitel 7

Psychologische und neurobiologische Aspekte

Da der Kern dieser Arbeit ein technischer Beitrag ist, soll am Ende nun die Analogie zur Neurobiologie und Verhaltenspsychologie hergestellt werden. Hierfür möchte ich zunächst die psychologischen Aspekte von Reinforcement Learning beleuchten, welche auf den Behaviorismus zurückführen. Anschließend werden die neurobiologischen Aspekte übersichtsartig zusammengefasst, welche in enger Verbindung zu anderen Lernparadigmen, wie Supervised Learning und Unsupervised Learning, stehen. Abschließend werden die erarbeiteten, technischen Beiträge dieser Arbeit versucht in die Neurobiologie einzuordnen¹.

7.1 Psychologische Aspekte

Verhaltenspsychologen führten seit Beginn des 20. Jahrhunderts Experimente über das Lernen bei Tieren (Pavlov, 1927; Thorndike, 1911) und Menschen (Skinner, 1953; Watson und Rayner, 1920) durch, bis in die 1970er Jahre hinein, in dem wissenschaftstheoretisch begründeten Paradigma des *Behaviorismus*. Bei diesen Experimenten und Modellbildungen war unter anderem keine Introspektion (Selbstbeobachtung) zur Messung des Lernfortschritts zugelassen. Hingegen waren lediglich äußere, objektive Methoden zulässig; wie zum Beispiel das Messen des Lernfortschritts durch die zeitliche Dauer, die vom Probanden benötigt wird, um eine bestimmte Aufgabe zu erfüllen. Typischerweise verkürzt sich die Dauer, je mehr

¹Dieses Kapitel basiert auf (Tokic, 2013).

Trainings-Durchläufe getätigt werden. Zwei Paradigmen sind hierbei das *Klassische* sowie das *Operante Konditionieren*, welche in engem Zusammenhang zu den berechenbaren Modellen von Reinforcement Learning stehen (Niv, 2009; Niv u. a., 2006).

7.1.1 Klassisches Konditionieren

Iwan P. Pavlov entdeckte am Anfang des 20. Jahrhunderts das Prinzip der *Klassischen Konditionierung* (Pavlov, 1927), was direkt aus seinen 1904 mit dem Nobelpreis gewürdigten Studien, über neurophysiologische Untersuchungen zur Rolle des Speichels und der Magensekretionen bei der Verdauung, hervorging. Am Beispiel von Hunden konnte Pavlov zeigen, dass eine unkonditionierte Reaktion, die durch einen unkonditionierten Stimulus hervorgerufen wird, durch einen neutralen Stimulus konditioniert werden kann, welcher bislang noch nicht mit der gewünschten Reaktion assoziiert ist.

Pavlov zeigte dies, indem er einem Hund Futter präsentierte (der unkonditionierte Stimulus), worauf der Speichel im Mund des Hundes zu fließen begann (die unkonditionierte Reaktion). Durch entsprechendes Training kann der Speichelfluss bei Präsentation eines bislang vom Hund nicht mit Futter assoziierten, neutralen Stimulus gezielt hervorgerufen werden. Pavlov präsentierte hierfür zusätzlich zum Futter einen Glockenton als zweiten (zu konditionierenden) Stimulus, der entweder gleichzeitig oder in zeitlich naher Abfolge mit dem Futter ertönen muss. Wenn die Kombination aus diesem und dem unkonditionierten Stimulus nun in ausreichendem Maße dem Hund präsentiert wurde, reagierte dieser mit Speichelfluss auch bei alleinigem Ertönen des Glockentons, also auch ohne einer Präsentation des Futters. Der Hund hatte folglich gelernt den Glockenton mit einer Futtergabe zu assoziieren, was bedeutet, dass der bislang neutrale Stimulus zu einem konditionierten Stimulus wurde und die Futtergabe infolgedessen zu einer konditionierten Reaktion.

Rescorla und Wagner (1972) stellten für dieses Verhalten ein einfaches, berechenbares Modell vor, wodurch viele Effekte der Klassischen Konditionierung mathematisch erklärbar wurden. Dieses ist das bis heute einflussreichste Modell für Lernen bei Menschen und Tieren (Niv, 2009), welches u. a. aufzeigt, dass Lernen nur dann stattfindet, wenn auftretende Ereignisse nicht den prädizierten Erwartungen entsprechen. Wenn beispielsweise in einem Experiment zwei zu konditionierende Stimuli CS_1 und CS_2 (z. B. Licht und Glockenton) gemeinsam mit einem unkonditionierten aber re-

aktionsauslösenden Stimulus US (wie z. B. Futter) präsentiert werden, so wird die Assoziation $V(CS_i)$ für jeden zu konditionierenden Stimulus wie folgt bestärkt:

$$V_{\text{new}}(CS_i) = V_{\text{old}}(CS_i) + \eta \left[\lambda_{\text{US}} - \sum_i V_{\text{old}}(CS_i) \right] . \quad (7.1)$$

Wie man dem Rescorla-Wagner-Modell entnehmen kann, findet Lernen genau dann statt, wenn sich der Wert des prädizierten Verhaltens $\sum_i V_{\text{old}}(CS_i)$, von der tatsächlichen Beobachtung λ_{US} unterscheidet. Der Parameter η ist dabei eine Lernrate, welche von den Eigenschaften aller präsentierten Stimuli abhängig sein kann.

7.1.2 Operantes Konditionieren

Thorndike (1911) studierte ebenfalls das Lernverhalten von Tieren, jedoch in anderer Art und Weise als Pavlov dies tat. Unter anderem wurde das Lernverhalten von hungrigen Katzen untersucht, wofür ein spezieller Käfig, die sogenannte *puzzle box*, entwickelt wurde. Die Aufgabe des Tieres bestand darin eine sichtbare Futterquelle vor dem Käfig zu erreichen. Am Käfig wurden über Seile verbunden mehrere Mechaniken verbaut, wobei das Betätigen in richtiger Reihenfolge die Tür öffnete und infolgedessen dem Tier ermöglichte, das Futter zu erlangen. Thorndike beobachtete, wie die Katze versuchte, dem Käfig zu entkommen, und hierfür wild, mehr oder weniger versehentlich, die Seile betätigte, bis sich schließlich die Tür öffnete. Das Experiment wurde mehrmals mit derselben Katze wiederholt, wobei sich beobachten ließ, dass immer schneller die richtige Reihenfolge wiedergewählt wurde. Anfangs benötigte eine der Katzen hierfür ca. 160 Sekunden, nach 24 Durchläufen jedoch nur noch ca. 7 Sekunden.

Als Erkenntnis leitete Thorndike ab, dass dem Tier nicht nur die angeborene Intelligenz zur Verfügung steht, sondern auch bestimmte Verhaltensweisen, durch Versuch und Irrtum, hinzugelernt werden können. Aus diesen und weiteren Beobachtungen an Hühnern und Hunden, ließen sich folgende zwei Gesetzmäßigkeiten zusammenfassen (Thorndike, 1911, S. 244):

„The Law of Effect is that: Of several responses made to the same situation, those which are accompanied or closely followed by satisfaction to the animal will, other things being equal, be more firmly connected with the situation, so that, when it recurs, they

will be more likely to recur; those which are accompanied or closely followed by discomfort to the animal will, other things being equal, have their connections with that situation weakened, so that, when it recurs, they will be less likely to occur. The greater the satisfaction or discomfort, the greater the strengthening or weakening of the bond.

The Law of Exercise is that: *Any response to a situation will, other things being equal, be more strongly connected with the situation in proportion to the number of times it has been connected with that situation and to the average vigor and duration of the connections. “*

Das erste Gesetz sagt aus, dass eine Reaktion auf einen Stimulus umso mehr bestärkt wird, je größer die Zufriedenheit für das Tier ist. Auf der anderen Seite führen unzufriedene oder gar aversive Auswirkungen dazu, dass die Reaktion auf den Stimulus seltener erfolgt. Je größer hierbei die (Un)Zufriedenheit der Auswirkung ist, desto mehr wird die getätigte Reaktion auf den Stimulus bestärkt. Das zweite Gesetz sagt aus, dass eine entsprechende Reaktion umso häufiger auf einen Stimulus erfolgt, je öfter die selbige Auswirkung mit der Stimulus-Reaktion verbunden ist.

Ähnliche Experimente, jedoch an Menschen, wurden später u. a. von Watson und Rayner (1920) (anhand des berühmten Little-Albert-Experimentes) und Skinner (1953) durchgeführt. Schlussendlich entstand hieraus das Konzept der *Operanten Konditionierung*.

7.1.3 Fazit

Zusammenfassend betrachtet sind beide Arten der Konditionierung sehr ähnlich, da Stimulus-Reaktions-Assoziationen konditioniert werden. Genauer betrachtet unterscheiden sie sich jedoch signifikant in einer wichtigen Annahme. Bei der Klassischen Konditionierung wird der unkonditionierte und neutrale Stimulus von außen (vom Trainer) vorgegeben, mit dem Ziel, eine Assoziation zwischen der gewünschten Reaktion und der alleinigen Gegebenheit des neutralen Stimulus zu erzeugen. Hingegen wird beim Operanten Konditionieren lediglich der unkonditionierte Stimulus präsentiert und vom Individuum eine passende Reaktion erlernt bzw. verbessert. In einfachen Worten ausgedrückt wird also beim Klassischen Konditionieren gelernt das Verhalten der Umgebung zu prädictieren; hingegen bei der Operanten Konditionierung das Verhalten an sich erlernt, um die Dynamik der Umgebung zielführend auszunutzen. Aus diesem Grund

passt das behavioristische Konzept der Operanten Konditionierung besser zum Modell des Reinforcement Learnings (genauer gesagt zum modellfreien Temporal-Difference Learning).

7.2 Neurobiologische Aspekte

Nach dem derzeitigen Kenntnisstand gibt es Belege dafür, dass die Lernparadigmen *Supervised Learning*, *Unsupervised Learning* und *Reinforcement Learning* (vgl. Abb. 7.1) in unterschiedlichen Hirnarealen zum Einsatz kommen (Doya, 1999, 2000). Ebenso entdeckten Wissenschaftler, dass die jeweils beteiligten Areale unterschiedlicher Lernparadigmen miteinander kommunizieren (Bostan u. a., 2010; Doya, 1999; Houk und Wise, 1995). Diesbezüglich fasst Doya (1999), basierend auf vielen anatomischen, physiologischen und theoretischen Belegen, den Kenntnisstand der Hirntheorie wie folgt zusammen:

- **Unsupervised Learning in der Großhirnrinde:** Präzise Repräsentation des sensorischen Zustands, des Kontexts und der Aktion. Finden einer passenden modularen Architektur für die gegebene Aufgabe.
- **Reinforcement Learning in den Basalganglien:** Evaluierung der aktuellen Situation durch Reward-Prognosen. Selektierung von passenden Aktionen, durch Evaluation von Aktionskandidaten.
- **Supervised Learning im Kleinhirn:** Lernen von internen Modellen des Körpers und der Umgebung. Replizierung von beliebigem Input-Output-Mapping, das an anderer Stelle im Gehirn erlernt wurde.

Die Funktionsweise in den jeweiligen Hirnarealen wird nun übersichtartig dargestellt, mit genauem Einblick in die zu RL gehörige Steuerung von Exploration und Exploitation.

7.2.1 Unsupervised Learning in der Großhirnrinde

Als ein Unsupervised-Learning-System wird die Großhirnrinde (*engl. cerebral cortex*) verstanden, in welcher die Aktivität eines Neurons auf den wahrgenommenen Signalen vom Zelleingang basiert. Der Neocortex ist hierbei der stammesgeschichtlich jüngste Teil der Großhirnrinde, welcher bei Menschen ca. 90% dieser umfasst. Organisiert in sechs Ebenen, verarbeitet der Neocortex unter anderem sensorische, motorische und kontextuelle Informationen, wobei massiv rückgekoppelte Verbindungen dazu dienen,

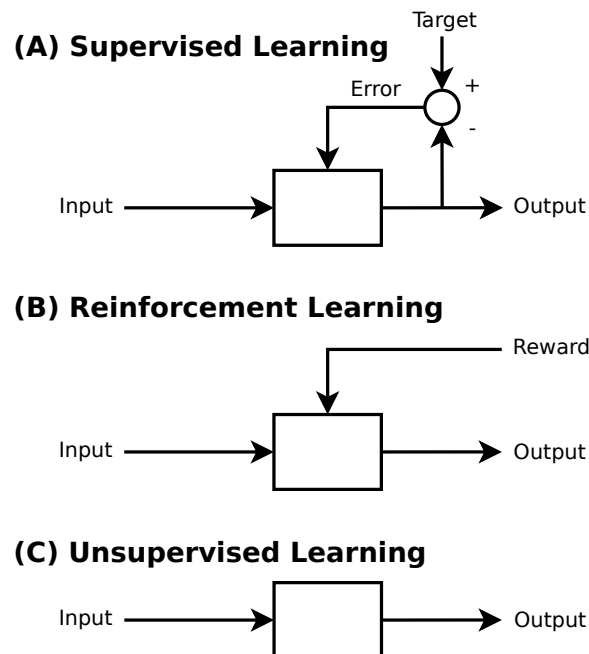


Abbildung 7.1: Drei grundlegende Lernparadigmen: (A) Supervised Learning durch Fehlerpropagierung; (B) Reinforcement Learning durch ein Reward-Signal; (C) Unsupervised Learning durch Statistiken basierend auf dem Input-Signal.

zeitlich kodierte Informationen erlernen zu können. Dabei deckten Untersuchungen an Katzen auf, dass das Antwortverhalten von Neuronen im visuellen Cortex nicht durch ein Fehler- oder Reward-Signal beeinflusst wird, sondern durch sensorische Erfahrungen aus der Vergangenheit (Blakemore und Cooper, 1970; Hirsch und Spinelli, 1970). Ebenso wurde beobachtet, dass die Plastizität von kortikalen Synapsen analog zur Hebb-schen Lernregel erfolgt (Artola u. a., 1990; Tsumoto und Suda, 1979): Die Verbindung zweier miteinander verbundenen Neuronen A und B wird verstärkt, falls A und B gleichzeitig aktiv sind; hingegen abgeschwächt, falls dies nicht der Fall ist (Hebb, 1949). Diese Beobachtungen deuten somit darauf hin, dass das Lernen in der Großhirnrinde nach dem Unsupervised-Learning-Paradigma erfolgt, da wahrgenommene Muster am Zelleingang, die gleichzeitige Aktivität eines nachgeschalteten Neurons beeinflussen (Doya, 1999).

Eine wichtige Aufgabe der Großhirnrinde ist die Aktivierung einzelner Neuronengruppen zur Beschreibung der aktuellen Situation, was auf Basis einer Art Sensor-Fusion, z. B. durch ein Clustering mehrerer Sensor-

Modalitäten, erreicht werden kann. Bildhaft kann man sich dies so vorstellen, dass es für eine auszuführende Motorik irrelevant ist *wie* ein Befehl ankommt; z. B. das Arbeitszimmer zu verlassen, um zu einem Meeting zu gehen. Der Befehl könnte akustisch wahrgenommen werden, wenn ein Kollege auf dem Gang »Meeting« ruft; ebenso aber auch visuell, wenn ein Pop-Up-Fenster des elektronischen Terminkalenders auf dem Bildschirm erscheint. Die Aktivierung der entsprechenden Neuronengruppen zum Durchführen der passenden Motorik (Verlassen des Arbeitszimmers), ist unabhängig von der Sensor-Modalität, mittels welcher die Aufforderung zum Teammeeting zu gehen wahrgenommen wurde.

7.2.2 Reinforcement Learning in den Basalganglien

In physiologischen Experimenten an Primaten wurde beobachtet, dass Aktivitäten von dopaminergen Neuronen im Mittelhirn ein Signal übertragen, welches Reward frühzeitig prädiziert (Schultz u. a., 1993; Schultz, 1998; Schultz u. a., 1997). Derartige Neuronen projizieren den Neurotransmitter Dopamin in die Basalganglien (*engl. basal ganglia*), welche unter anderem an der motorischen Aktionswahl stark beteiligt zu sein scheinen (Vitay u. a., 2009). Dopamin dient in diesem Zusammenhang als Reward zur Bestärkung von Aktionen, was darauf schließen lässt, dass in diesem Areal das Lernparadigma von Reinforcement Learning vorzufinden ist (Doya, 2007). Interessanterweise wurde beobachtet, dass Aktivitäten von dopaminergen Neuronen nicht nur auf den unmittelbaren Reward zurückzuführen sind, sondern ebenfalls zukünftigen Reward prädizieren. Das erstaunliche hieran ist, dass diese Erkenntnis die Theorie von berechenbaren RL-Modellen widerspiegelt, in welchen der *Temporal-Difference Error* die selbige Funktion besitzt (Barto, 1995; Sutton, 1988; Sutton und Barto, 1998). Dieser ist das Basiselement für das Bestärken von Aktionen, da er den Unterschied zwischen dem prädizierten Reward und der tatsächlichen Observation angibt. Wie bereits anhand des Rescorla-Wagner-Modells gezeigt wurde (vgl. Abschnitt 7.1.1), findet Lernen hierbei nur statt, wenn sich Prädiktionen und Observationen unterscheiden.

In der Hirntheorie geht man davon aus, dass die aktuelle Situation evaluiert wird, indem Reward-Prognosen für mögliche Aktionen erstellt werden (Doya, 1999). Auf Basis dieser wird eine passende, motorische Aktion getätigt. Erhalten daraufhin die beteiligten Neuronen Dopamin, so steigt die Wahrscheinlichkeit, in zukünftigen, ähnlichen Situationen die selbige Aktion erneut zu wählen (McClure u. a., 2003).

7.2.3 Supervised Learning im Kleinhirn

Das Kleinhirn (*engl. cerebellum*) scheint laut einer Hypothese von Albus (1971) und Marr (1969) ein Supervised-Learning-System zu sein. In einem derartigen wird nicht von einem Reward-Signal gelernt, sondern von einem Fehlersignal, über zu erlernende Input-Output-Muster; beispielsweise Modelle des Körpers oder der Umgebung (Doya, 1999).

Experimente über die Stabilisierung der visuellen Wahrnehmung zeigen, dass die dauerhafte Abschwächung der Signalübertragung (Langzeit-Depression) einer Purkinjezellsynapse im Kleinhirn, von Signalen einer Kletterfaser abhängig ist (Ito u. a., 1982). Die Kletterfasern besitzen ihren Ursprung im Olivenkomplex, wobei eine Purkinjezelle mit je einer der Kletterfasern verbunden ist. Eine Purkinjezelle besitzt jedoch weitere, geschätzte 200.000 Eingänge vom Parallelfaser-System, wodurch der rückgekoppelte Eingang der Kletterfaser als Fehlersignal interpretiert wird.

7.2.4 Steuerung von Exploration/Exploitation

Cohen u. a. (2007) berichten, dass die Erforschung von Exploration und Exploitation in der Neurobiologie noch sehr jung ist und demnach als hoch spekulativ einzustufen gilt. Diese Beobachtung spiegelt dabei teilweise die eigentliche Problemstellung wider, nämlich dass es keinen allgemeingültigen, optimalen Mechanismus für Exploration und Exploitation gibt, auch wenn die Ziele der gegebenen Aufgabe klar definiert sind. In der Neurobiologie gibt es bereits einen Konsens darüber, dass für die Steuerung zwischen beiden Modi das neuromodulatorische System stark involviert zu sein scheint (Cohen u. a., 2007; Doya, 2008). Dieses ist dafür zuständig Neurotransmitter zu modulieren, welche beim Lernen dazu verwendet werden, die Verbindungen beteiligter Neuronen abzuschwächen oder zu verstärken.

In Experimenten an Menschen fanden Daw u. a. (2006) mittels funktionaler Magnetresonanztomographie heraus, dass Entscheidungen über Exploration und Exploitation zu einer erhöhten Aktivität im präfrontalen Cortex (PFC) führen. Hierbei wurde die Gehirnaktivität von Probanden beim Spielen eines mehrarmigen Banditen gemessen, wobei sich herausstellte, dass Explorationsentscheidungen zu erhöhter Aktivität im frontopolareren Bereich des PFC führen, hingegen Exploitationsentscheidungen zu erhöhter Aktivität im ventromedialen PFC. Für die Steuerung zwischen beiden Modi wurde der Neurotransmitter *Norepinephrin* identifiziert (Aston-Jones

und Cohen, 2005). Dessen Ausschüttung wurde im *locus coeruleus* beobachtet, von wo aus der Transmitter in mehrere Hirnareale projiziert wird. Die Steuerung der Ausschüttung scheint in Abhängigkeit von Unsicherheit und erwarteten Reward zu erfolgen: „ongoing assessments of both uncertainty and utility are likely to be important in regulating this balance“ (Aston-Jones und Cohen, 2005, S. 939). Allerdings können auch andere Transmitter eine wichtige Rolle spielen, wie beispielsweise Dopamin, was ein Modell von McClure u. a. (2006) suggeriert.

7.3 Einordnung der technischen Beiträge in die Neurobiologie

Die vorgestellten Explorationsstrategien wurden anhand der modellfreien Lernverfahren *Q*-learning (Watkins, 1989) und Sarsa (Rummery und Niranjan, 1994) untersucht, mit welchen intelligentes Verhalten anhand sensorischer Interaktionen erlernt werden kann. Die Auswahl der Lernverfahren war dadurch motiviert, Kompatibilität der entwickelten Explorationsstrategien zu weitläufig genutzten Lernverfahren zu demonstrieren, welche als plausible, berechenbare Modelle für das Lernen in natürlichen Organismen gelten (Dayan, 2009; Niv, 2009; Niv u. a., 2006; Watkins, 1989). Da jedoch *Q*-learning und Sarsa keine integrative Steuerung von Exploration und Exploitation besitzen, muss diese über eine separate Explorationsstrategie realisiert werden, welche die Reward-Schätzungen des entsprechenden Lernverfahrens verwendet.

Die Separation beider „Lernmodule“ spiegelt in diesem Sinne auch das Lernen im hirntheorischen Kontext wieder, in welchem für beide Aufgaben unterschiedliche Hirnareale zugeordnet werden. Wie unter 7.2 dargestellt wurde, ist das Lernparadigma *Reinforcement Learning* in den Basalganglien beobachtet (Doya, 1999, 2007) worden. Aktionen werden in diesem Bereich anhand des Neurotransmitters Dopamin bestärkt, welcher als TD-Fehler interpretiert wird (Doya, 2008) und durch dopaminerge Neuronen im Mittelhirn entsteht (Schultz u. a., 1993; Schultz, 1998; Schultz u. a., 1997). Hingegen wurde die Steuerung von Exploration und Exploitation im präfrontalen Kortex beobachtet (Aston-Jones und Cohen, 2005; Daw u. a., 2006; McClure u. a., 2006), welche auf den Neurotransmittern Dopamin und Norepinephrin basiert. Dabei ist bemerkenswert, dass die motorische Steuerung in einem Nachbarbereich des präfrontalen Kortex stattfindet, dem *motorischen Kortex*. Scheinbar hat es sich in der Evolution als Vorteil erwiesen

die Steuerung von Exploration und Exploitation in direkter Nachbarschaft zur motorischen Steuerung anzusiedeln.

7.3.1 VDBE-Strategien

Die in dieser Arbeit entwickelten VDBE-Strategien ordnen sich dahingehend in das biologische Vorbild ein, dass u. a. der TD-Fehler (Dopamin aus dem Mittelhirn) zur Steuerung von Exploration und Exploitation verwendet wird, hingegen keine zusätzlichen Hilfsmittel, wie z. B. Aktionszähler (Auer u. a., 2002; Kocsis und Szepesvári, 2006; Thrun, 1992). Die Modulation des Explorationsverhaltens kann man als Norepinephrin-Ausschüttung ansehen, um die in den Basalganglien erstellten Reward-Prognosen entsprechend zu modulieren. Hierfür ist zwangsläufig Wissen aus der Vergangenheit notwendig, um die Ausschüttung von Norepinephrin vor der tatsächlichen Aktionswahl zu gewährleisten. Die Speicherung der dazugehörigen Informationen findet sich bei VDBE-Strategien anhand der Explorationsrate $\varepsilon(s)$ wider. Eine zustandsabhängige Steuerung erscheint aufgrund dessen vorteilhaft, da im Gegensatz zu einer globalen, konstanten Explorationsrate, man nicht in allen besuchten Zuständen einer Lernepisode mit gleichem Maße explorativ ist. Bislang erlerntes Wissen wird also in bekannten Zuständen, in denen die Dynamik der Umgebung bereits bekannt ist, ausgenutzt.

Verglichen mit der Studie von Daw u. a. (2006) passt von beiden VDBE-Strategien VDBE-Softmax am ehesten zum biologischen Vorbild. Diese Analogie kann man damit begründen, da Explorationsaktionen wertebasiert durch Softmax gewählt werden, was hingegen gleichverteilt bei der VDBE-Strategie erfolgt. Für die Steuerung der Lernrate δ erscheint es plausibel, dass diese in derselben Weise gesteuert werden könnte, wie die Lernrate α . Laut Doya (2002) wurde für diese der Neurotransmitter *Acetylcholine* identifiziert, der evtl. auch zur Steuerung von δ zur Anwendung kommen könnte. Ungeklärt bleibt jedoch in diesem Modell die Steuerung der inversen Sensibilität σ , wofür in der vorliegenden Arbeit eine REINFORCE-Steuerung in Betracht gezogen wurde.

Da beim Adaptieren des Explorationsparameters jedoch nicht nur der TD-Fehler verwendet wird, sondern ebenso die Lernrate α , scheint dies auf ein analoges Zusammenspiel im Gehirn hinzudeuten. In diesem Kontext stellen McClure u. a. (2006) ein entsprechendes Modell vor, welches Exploration und Exploitation durch ein Zusammenspiel der Neurotransmitter Dopamin und Norepinephrin steuert. Zusätzlich müsste demnach geschluss-

folgt werden, dass auch Acetylcholine an diesem Prozess beteiligt ist, da dieser Neurotransmitter den Lernraten zugeordnet wird (Doya, 2002).

7.3.2 REINFORCE-Strategien

Die entwickelten REINFORCE-Strategien ordnen sich in den Kontext der Hirntheorie dahingehend ein, dass sie in der schrittweisen Variante ebenfalls lokale Informationen verwenden. Hierdurch ist es möglich, exploratives Verhalten nur Bereichen des Zustandsraumes hervorzurufen, in denen Verbesserungen des Rewards zu erwarten sind. Im Unterschied zu VDBE-Strategien liegt der Vorteil jedoch darin, dass nicht nur eine zustandsabhängige Explorationsrate $\varepsilon(s)$ adaptiert wird, sondern der Explorationsparameter einer beliebigen Explorationsstrategie. Hierdurch kann z. B. direkt der Temperatur-Parameter τ von Softmax gesteuert werden oder die inverse Sensibilität σ von VDBE-Strategien. Da Explorationsstrategien typischerweise auf Schätzungen über den Rewards basieren, kann die Adaption eines Explorationsparameters (analog zu den VDBE-Strategien) als Regulierung der Ausschüttung von Norepinephrin angesehen werden. Im Gegensatz zu VDBE-Strategien wird jedoch beim Adaptieren nicht der TD-Fehler verwendet, sondern die auf dessen Basis verbesserte Schätzung $Q^n(s, a)$ nach Anwendung eines TD-Lernverfahrens. Diesbezüglich stellte sich in den Voruntersuchungen heraus (die nicht in dieser Arbeit dargestellt wurden), dass die Verwendung der verbesserten Schätzung $Q^n(s, a)$ zu besseren Ergebnissen führt als der TD-Fehler. Dies stellt somit eine technische Optimierung dar.

Zur Repräsentation der Verteilung des Explorationsparameters verwenden REINFORCE-Verfahren lokale Skalare (Mittelwert und Standardabweichung), die anhand von Informationen aus der Gegenwart (Schätzung $Q^n(s, a)$) und aus der Vergangenheit (Baseline $\bar{\rho}(s)$ etc.) adaptiert werden. Alle Lernkomponenten als eine Einheit betrachtet, wurden bereits von Williams (1992) als ein *stochastisches Neuron* bezeichnet.

7.3.3 Interaktion mit dem Kleinhirn

Als ungeklärte Frage verbleibt bislang, *wo* im Gehirn die zustandsabhängigen Daten ($\varepsilon(s)$, $\mu(s)$ etc.) der VDBE- und REINFORCE-Strategien verwaltet werden. Im Kontext des berechenbaren Reinforcement Learning werden Zustands- bzw. Zustands-Aktions-Werte typischerweise anhand von Tabellen oder Funktionsapproximatoren repräsentiert, was dem Lernparadigma des Supervised Learning entspricht. Für dieses ist aus neurobiolo-

gischer Sicht das Kleinhirn beteiligt (Doya, 2000). Mit Hinblick auf Doya's Theorie erscheint es daher plausibel, dass ein Modell der zustandsabhängigen Explorationsdaten ebenfalls im Kleinhirn erlernt wird, da man diesem die Fähigkeit zuschreibt beliebiges Input-Output-Mapping zu replizieren, das an anderer Stelle im Gehirn erlernt wurde. Tatsächlich wurde im Gehirn eine Zwei-Wege-Kommunikation zwischen den Basalganglien und dem Kleinhirn entdeckt, wie Bostan u. a. (2010) berichten. Diese Entdeckung lässt es demnach plausibel erscheinen, dass Reward-Prognosen und Explorationsdaten möglicherweise im Kleinhirn verwaltet werden, welche die Basalganglien sozusagen *abrufen* und wieder *abspeichern*.

7.4 Diskussion

In diesem Kapitel wurde gezeigt, was unter *Lernen* in der Neurobiologie verstanden wird, welche Lernparadigmen hierfür in einzelnen Hirnarealen vorzufinden sind und wie sich die technischen Beiträge dieser Arbeit in den Kenntnisstand der Neurobiologie einordnen. Ebenso wurde das Verwandtschaftsverhältnis zwischen dem Paradigma Reinforcement Learning und der Verhaltenspsychologie dargestellt, welches auf den Behaviorismus zurückführt. Die Neurobiologie zeigt hierbei, dass Reinforcement Learning nicht ausschließlich für Lernen im Gehirn verantwortlich ist, sondern ebenso Unsupervised Learning und Supervised Learning zum Einsatz kommen. Ein Zusammenspiel deren berechenbarer Modelle ist in der Praxis von großem Interesse (Distler, 2012; Handrich u. a., 2011; Sasakawa u. a., 2008), was ebenso im Gehirn beobachtet wurde (Bostan u. a., 2010; Doya, 1999; Houk und Wise, 1995). Insbesondere für den Robotik-Bereich bin ich der Überzeugung, dass das biologische Vorbild sehr nützlich ist, um universelle Software für das Trainieren von Robotern entwickeln zu können (Ertel u. a., 2009).

Abschließend möchte ich noch anmerken, dass in der Natur verschiedene Quellen von Reward verfolgt werden, die ein Organismus optimieren möchte; z. B. Futter und Wasser, die das Überleben sichern, jedoch unabhängige Reward-Quellen sind. Eine weitere wichtige Quelle ist die Neugier (*engl. curiosity*), welche z. B. uns Wissenschaftler antreibt neue Erkenntnisse zu gewinnen. Der Reward für Neugier kommt nicht von der externen Umgebung, sondern entstammt vom Inneren eines Organismus, weswegen er auch als *intrinsischer Reward* bezeichnet wird: „*Children do not play for a reward-praise, money, or food. They play because they like it.*“ (Wardle, 1987, S. 28). Die Entwicklung von berechenbaren Modellen ist ebenfalls

Gegenstand aktueller Forschungen und wird bereits für das Erlernen von Roboter-Skills eingesetzt (Ngo u. a., [2012](#); Pape u. a., [2012](#)).

Kapitel 8

Schlussbetrachtung und Ausblick

In der vorliegenden Arbeit wurden neue Verfahren zur Steuerung von Exploration und Exploitation für modellfreies Reinforcement Learning in diskreten Aktionsräumen entwickelt. Im untersuchten Framework besitzt ein Lernagent kein Vorwissen über die Umgebung, sondern lernt sein Verhalten ausschließlich durch sensomotorische Interaktion mit dieser, weshalb die Steuerung von Exploration und Exploitation einen signifikanten Einfluss auf die langfristige Maximierung des Rewards besitzt. Zum Einen müssen genügend Explorationsaktionen zur Umgebungserkundung getätigt werden, wodurch ein Agent erfährt, welche Aktionen zielführend sind. Zum Anderen dürfen jedoch nicht übermäßig viele Explorationsaktionen getätigt werden, um negativen Reward möglichst zu vermeiden. Der Nachteil von Verfahren wie ϵ -Greedy und Softmax ist, dass ein Explorationsparameter gefunden werden muss (typischerweise durch trial-and-error), dessen optimale Belegung von der Dynamik des Lernproblems abhängig ist. Aufgrund dessen stellt die Variation des Explorationsparameters während des Lernprozesses eine große Herausforderung dar. Existieren sogar nichtstationäre Prozesse, erfordert dies für den Explorationsparameter eine dynamische Adaption, um bei wahrgenommenen Änderungen in der Umgebung diese aktiv neu zu erkunden und lokalen Minima zu entweichen. Im Vergleich zum Lernen bei Menschen und Tieren stellt man schnell fest, dass auch hier solch ein dynamischer Explorationsprozess existiert. Der Vorteil von ϵ -Greedy und Softmax ist wiederum, dass diese Explorationsstrategien von Haus aus keine Speicherung von Explorationsdaten be-

nötigen, sondern direkt auf dem aktuellen Stand der Wertefunktion arbeiten.

8.1 Beiträge zur Zielsetzung dieser Arbeit

In dieser Arbeit wurde der Frage nachgegangen, wie eine Explorationsstrategie auf Basis des TD-Fehlers gestaltet werden muss, um die in der Einleitung genannten Forschungsfragen *Nichtstationarität*, *Stochastik*, *Robustheit*, *Partielle Beobachtbarkeit* und *Neurobiologische Plausibilität* zu adressieren. Hierdurch unterscheidet sich diese Vorgehensweise von anderen Verfahren, indem beispielsweise keine Aktionszähler (Auer, 2002; Kocsis und Szepesvári, 2006; Thrun, 1992) oder nicht nur der unmittelbare Reward (Dayan, 1991; Schweighofer und Doya, 2003; Williams, 1992) verwendet werden. Stattdessen wird der Explorationsprozess vom TD-Fehler angetrieben, welcher einem natürlichen Lernsignal entspricht, welches in der Neurobiologie mit Dopamin assoziiert wird (Schultz u. a., 1993; Schultz, 1998). Im Rahmen dieser Arbeit wurden folgende Ergebnisse erzielt:

- **Nichtstationaritäten**

Die VDBE-Strategien können Nichtstationaritäten (Umgebungsänderungen) in Form von Unterschieden zwischen der Prädiktion und Observation von Reward detektieren. Ein großer Betrag des Wertunterschieds zeigt auf, dass entweder der Agent zu selten gewisse Aktionen in Zustand s gewählt hat, er daher noch zu wenig über deren langfristige Auswirkung weiß, oder nichtstationäre Prozesse in der Umgebung stattfanden. Die vorgestellten REINFORCE-Strategien detektieren letztere anhand des kumulierten Rewards einer Lernepisode (globale Adaption) oder des Q -Werts (lokale Adaption) der getätigten Aktion in Zustand s_t . Um diesen Aspekt zu adressieren wurden in den experimentellen Untersuchungen dieser Arbeit die vorgestellten Strategien anhand des Dynamic-Cliff-Problems evaluiert. Abbildung 6.4 zeigt hierfür die Performanz von VDBE-Strategien verglichen mit ε -Greedy und Softmax, Abbildung 6.5 die dynamische Adaption des Explorationsparameters für REINFORCE-Strategien. Den Ergebnissen ist zu entnehmen, dass die vorgestellten Verfahren das Problem von Nichtstationaritäten adäquat adressieren (Tokic und Palm, 2012a,b; Tokic u. a., 2012).

- **Stochastische Rewards**

Um die Anforderung an *stochastische Rewards* entsprechend zu adres-

sieren, hat es sich für VDBE-Strategien als Vorteil erwiesen nicht nur den TD-Fehler in Betracht zu ziehen, sondern ebenso auch die Lernrate α des Lernverfahrens der Wertefunktion. Letztere repräsentiert das erlernte Wissen über zukünftige Rewards. Das resultierende Produkt aus Lernrate und TD-Fehler wird als Wertunterschied bezeichnet, welcher durch Anwendung eines Lernverfahrens zum Neuschätzen der Wertefunktion, als Differenz zwischen der alten und neuen Reward-Prädiktion, entsteht. Durch Berücksichtigung von α kann ebenso der Mittelwert von stochastischen Rewards, anhand stochastischer Approximation (Robbins und Monro, 1951), geschätzt werden. Um eine Konvergenz zu erzielen existieren gewisse Anforderungen an die Lernrate α , damit der TD-Fehler für stochastische Rewards gegen Null geht und demnach ebenso die Explorationsrate $\varepsilon(s) \rightarrow 0$. Im Rahmen dieser Arbeit wurde dies insbesondere am n -armigen Banditenproblem in Kapitel 4.5 sowie am Bandit-World-Problem in Kapitel 4.7 untersucht (Tokic, 2010; Tokic und Palm, 2011).

- **Robustheit**

Die Anforderung *Robustheit* wird adressiert, indem die Wahl von Explorationsaktionen wertebasiert gesteuert wird, d. h. das Auswählen von schlechten Aktionen gezielt abgeschwächt wird. Die Robustheit des Explorationsparameters wird benötigt, da die VDBE-Strategie über einen Parameter zur Skalierung des TD-Fehlers verfügt (inverse Empfindlichkeit), der in Abhängigkeit der minimalen und maximalen Reward-Höhe gewählt werden muss. Sollte die Empfindlichkeit während des Lernens zu hoch eingestellt sein, werden Aktionen nicht wie bei einer ε -Greedy-Strategie mit selbiger Wahrscheinlichkeit gewählt, sondern wertebasiert in Abhängigkeit der Softmax-Strategie. Diese Vorgehensweise wurde von Wiering (1999) vorgeschlagen, und in der VDBE-Softmax-Strategie in vereinfachter Form angewandt (Tokic und Palm, 2011; Tokic u. a., 2012). Insbesondere sind an dieser Stelle die Ergebnisse des Cliff-Walking-Problems in Kapitel 5.3, des Mountain-Car-Problems in Kapitel 5.5 sowie des Dynamic-Cliff-Problems in Kapitel 6.4 erwähnenswert.

Die durchgeführten Experimente zeigen auch eine Robustheit für die Wahl der Basispolitik bei REINFORCE-Strategien. Bei der Verwendung von Q -learning als Lernverfahren für die Wertefunktion wurde in der lokalen Variante eine geringere Varianz der Ergebnisse beobachtet, wobei interessanterweise die MBE- und ε -Greedy-Strategien in beiden Experimenten die besten Ergebnisse lieferten (vgl. Abbil-

dung A.7). Ob diese Beobachtung auch auf andere Lernprobleme verallgemeinert werden kann, müssen weitere Untersuchungen feststellen.

- **Partielle Beobachtbarkeit**

Durch die Robustheit von VDBE-Softmax wird auch indirekt die *partielle Beobachtbarkeit* einer Umgebung adressiert. Da hierbei einem Agenten Wissen über die exakte Zustandskoordinate fehlt, führt dies bei modellfreien Lernverfahren oftmals zu Fluktuationen in der Wertefunktion und dadurch auch zu einem eventuell ungewollten Anstieg der Explorationsrate. Fluktuationen entstehen dadurch, dass modellfreie Lernverfahren die Transitionswahrscheinlichkeiten $\mathcal{P}_{ss'}^a$ für mögliche Folgezustände nicht explizit lernen, sondern lediglich den Q -Wert der getätigten Aktion bezüglich der Observation anpassen. Die *Robustheit* des Verfahrens stellt hierfür sicher, dass das Wählen von gelernten schlechten Aktionen wertebasiert unterdrückt wird. In den Experimenten dieser Arbeit wurde dieser Effekt künstlich am Mountain-Car-Problem in Kapitel 5.5+6.5 erzeugt, indem kontinuierliche Zustandskoordinaten diskretisiert wurden. Das Laufroboter-Problem aus Kapitel 5.4 ist ebenso partiell beobachtbar, was daraus resultiert, dass ebenfalls die Zustandskoordinaten diskretisiert werden sowie die Oberflächenbeschaffenheit nicht explizit im Zustandssignal abgebildet ist. Letzteres wurde in den Untersuchungen anhand eines stochastischen Reward-Signals modelliert.

- **Neurobiologische Plausibilität**

Der Forschungsfrage, ob die in dieser Arbeit vorgestellten Verfahren auch Rückschlüsse für die Neurobiologie zulassen, wurde in Kapitel 7 nachgegangen und spekulativ versucht zu beantworten. Die Forschungen von Schultz u. a. zeigen, dass dopaminerge Neuronen im Mittelhirn ein dem TD-Fehler ähnliches Signal übertragen, welches zukünftigen Reward prädiziert (Schultz u. a., 1993; Schultz, 1998; Schultz u. a., 1997). Die These, dass Dopamin ebenso die Steuerung von Exploration und Exploitation unterstützt, deuteten McClure u. a. (2006) in einem berechenbaren Modell an und wird demnach von den VDBE-Strategien unterstützt. Im Rahmen dieser Arbeit wurde als komplementäre Komponente die Lernrate α identifiziert, da die Untersuchungen in Kapitel 4.4.3 zeigen, dass nur die Betrachtung des TD-Fehlers ungeeignet für das Lernen mit stochastischen Rewards ist. Bei letzteren stimmt die Prädiktion sehr selten mit der Observation überein, weswegen ein Zusammenspiel beider Komponenten im

Gehirn als möglich erscheint. Doya zeigt in diesem Zusammenhang auf, dass Lernraten durch den Neurotransmitter Acetylcholine moduliert werden (Doya, 2002), was für VDBE-Strategien auf ein Zusammenspiel zwischen Dopamin und Acetylcholine hindeutet.

8.2 Limitierungen

In den durchgeführten Untersuchungen dieser Arbeit stellten sich folgende Limitierungen heraus, die es für zukünftige Folgearbeiten in Angriff zu nehmen gilt:

- **Reward-Stochastik bei globaler REINFORCE-Steuerung**

Obwohl die globale REINFORCE-Strategie sehr speichereffizient ist, hat sie zum Nachteil, dass variable Pfadlängen zu ungewollter Exploration führen können. Die Begründung hierfür ist, dass der Return einer Lernepisode zum Bestärken verwendet wird, welcher bei variablen Pfadlängen eine hohe Varianz besitzen kann. Bei der lokalen Explorationssteuerung (VDBE und REINFORCE) wird hingegen der Q -Wert bzw. dessen Differenz beim Update verwendet, wodurch die Stochastik der Pfade in die Wertefunktion einfließt. Für derartige Lernprobleme sollte daher bevorzugt modellbasiertes Lernen verwendet werden, bei welchem ebenso die Transitionswahrscheinlichkeiten $\mathcal{P}_{ss'}^a$ für mögliche Folgezustände mitgelernt werden. Folgearbeiten sollten daher diesen Aspekt genauer untersuchen.

- **Nichtstationäre Prozesse außerhalb des konvergierten Pfades**

Ein Problem, welches Ishii u. a. (2002) ebenso feststellen, ist das Wahrnehmen von Umgebungsänderungen außerhalb des konvergierten Pfades. Verdeutlicht wurde dies am Dynamic-Cliff-Problem, bei welchem ein für VDBE zu hoch eingestelltes σ zur Greedy-Strategie führt und hierdurch Umgebungsänderungen in Phase (c) nicht wahrgenommen werden. Zu erklären ist dieses Phänomen dadurch, dass dem Agenten Informationen in der Wahrnehmung fehlen, die auf eine veränderte Umgebung hindeuten, jedoch im TD-Fehler des konvergierten Pfades nicht enthalten sind. Strategien, die wenigstens von Zeit zu Zeit ein wenig explorativ sind, "stolpern" zufällig über den zweiten Zielzustand und erlernen hierdurch dessen Existenz. Das Problem ist jedoch, dass diese dauerhaft explorativen Strategien auch von Zeit zu Zeit die Klippe hinabstürzen, was bei einer sehr langen

Klippe dazu führen kann, dass der Agent unter Verwendung von Q -learning unter Umständen nie am Ziel ankommt.

8.3 Ausblick

Die vorliegende Arbeit eröffnet neue Wege für zukünftige Forschungsarbeiten, die ich abschließend noch aufzeigen möchte. Neben der Lösung von den oben genannten Limitierungen, ergeben sich darüber hinaus folgende interessante Forschungsmöglichkeiten:

- **Zeitverzögerte Rewards – n -step Return**

In der vorliegenden Arbeit wurde lediglich das sogenannte TD(0)-Lernen betrachtet. Bei diesem wird nach jeder getätigten Aktion der dazugehörige Erwartungswert, auf Basis des unmittelbar erhaltenen Rewards, neu geschätzt. Demzufolge werden bei zeitverzögerten Rewards viele Lernepisoden benötigt, um den Reward auf die anfänglichen Aktionen einer Lernepisode zurückzupropagieren. Eine Möglichkeit dieses Problem zu entschärfen bieten sogenannte *Eligibility Traces*, die erhaltene Rewards in abgeschwächter Form zurückpropagieren und dadurch weniger Lernepisoden benötigen (Singh und Sutton, 1996). In den Folgeuntersuchungen wäre daher ein möglicher Ansatz, Adaptierungen des Explorationsparameters ebenso für durch Eligibility Traces hervorgerufene Neuschätzungen durchzuführen.

- **Adaption der Lernrate**

Ein weiterer Parameter mit erheblichem Einfluss auf die Lernzeit ist die Lernrate α . Eine interessante Erweiterungsmöglichkeit wäre zu untersuchen, ob diese parallel zur Explorationsrate $\varepsilon(s)$ adaptiert werden kann, in Analogie zu (Schweighofer und Doya, 2003) und (Kobayashi u. a., 2009). Bei stochastischen Rewards werden meistens eher niedrige Lernraten benötigt, um Fluktuationen zu vermeiden; hingegen bei deterministischen Rewards eher höhere Lernraten. Durch die Adaption dieses Parameters könnte somit eine weitere Last vom Experimentator genommen werden. Ein möglicher Ansatz hierfür wäre eine zweidimensionale Normalverteilung durch REINFORCE zu adaptieren, deren Parameter die Lernrate sowie den Explorationsparameter repräsentieren.

- **Funktionsapproximation**

Da VDBE und REINFORCE eine lokale Speicherung der Explorati-

onsdaten benötigen, wäre eine folgerichtige Untersuchung, Explorationsdaten anhand eines Funktionsapproximators (z. B. neuronale Netzwerke) zu repräsentieren. Ebenso gilt dies natürlich auch für die Q -Funktion (Eck und Wezel, 2008; Riedmiller, 2005; Riedmiller u. a., 2007). Im Prinzip sollen hierbei die Q -Werte nicht mehr tabellarisch gespeichert werden, mit dem Ziel eine Speicherplatz- sowie Lernzeitreduktion zu erzielen, die auf Generalisierung in Nachbarschaften bzw. ähnlichen Zuständen basiert.

- **Berücksichtigung innerer Ressourcen / Neugier**

Eine weitere Komponente zur Steuerung von Exploration und Exploitation, die ebenso in der Natur vorkommt, ist die Berücksichtigung von inneren Ressourcen, wie beispielsweise Futter, Wasser oder Energie. Wenn z. B. ein Tier aus eigenem Antrieb heraus von Zeit zu Zeit nach neuen Futter- oder Wasserquellen Ausschau hält, führt dies zu einem Wissenszuwachs, der in der Not das Überleben sichern kann. Hierarchisch gesteuert wird bei niedrigem Ressourcenstand (z. B. Hunger oder Durst) hingegen versucht bekannte Quellen auszubeuten, wofür das Wissen über diese notwendig ist. Ein hierfür in Frage kommender Antrieb zur hierarchischen Steuerung ist die *Neugier* (eine intrinsische Motivation), auf deren Basis zusätzliches Verhalten bei ausreichendem Ressourcenstand hinzugelernt werden kann, ohne dass es unmittelbar zu einem Nutzen führt. Zu einem späteren Zeitpunkt kann der Nutzen jedoch sehr wohl von Relevanz sein, wenn in gewissen Situationen die zu benötigende Lernzeit fatale Folgen hätte. In den vorgestellten Explorationsmodellen dieser Arbeit ist dieser Aspekt noch unberücksichtigt und somit eine interessante Fragestellung für verhaltenspsychologisch motivierte Folgearbeiten. In der Literatur ist er bereits Gegenstand aktueller Forschungen, insbesondere aus den Arbeitsgruppen um Andrew Barto (Simsek und Barto, 2006), Pierre-Yves Oudeyer (Kaplan und Oudeyer, 2007; Oudeyer und Kaplan, 2007; Oudeyer u. a., 2007) und Jürgen Schmidhuber (Ngo u. a., 2012; Pape u. a., 2012).

Literaturverzeichnis

- Ainslie, G. W. (1974). "Impulse control in pigeons". In: *Journal of the Experimental Analysis of Behavior* 21.3, S. 485–489.
- Albus, J. S. (1971). "A theory of cerebellar function". In: *Mathematical Biosciences* 10.1–2, S. 25–61.
- Artola, A., S. Bröcher und W. Singer (1990). "Different voltage-dependent thresholds for inducing long-term depression and long-term potentiation in slices of rat visual cortex". In: *Nature* 347.6288, S. 69–72.
- Aston-Jones, G. und J. D. Cohen (2005). "An integrative theory of locus coeruleus-norepinephrine function: adaptive gain and optimal performance". In: *Annual Review of Neuroscience* 28, S. 403–450.
- Auer, P. (2002). "Using confidence bounds for exploitation-exploration trade-offs". In: *Journal of Machine Learning Research* 3, S. 397–422.
- Auer, P., N. Cesa-Bianchi und P. Fischer (2002). "Finite-time Analysis of the Multiarmed Bandit Problem". In: *Machine Learning* 47, S. 235–256.
- Awerbuch, B. und R. D. Kleinberg (2004). "Adaptive routing with end-to-end feedback: distributed learning and geometric approaches". In: *Proceedings of the 36th annual ACM Symposium on Theory of Computing*. Chicago, IL, USA: ACM, S. 45–53.
- Azoulay-Schwartz, R., S. Kraus und J. Wilkenfeld (2004). "Exploitation vs. exploration: choosing a supplier in an environment of incomplete information". In: *Decision Support Systems* 38.1, S. 1–18.
- Bagnell, J. A., A. Y. Ng und J. G. Schneider (2001). *Solving Uncertain Markov Decision Processes*. Technical Report Paper 85. Pittsburgh, PA, USA: Robotics Institute, Carnegie Mellon University.
- Barto, A. G. (1995). "Adaptive critics and the basal ganglia". In: *Models of Information Processing in the Basal Ganglia*. Hrsg. von J. C. Houk, J. L. Davis und D. G. Beiser. Cambridge, MA: MIT Press, S. 215–232.
- Bellman, R. E. (1957). *Dynamic Programming*. Princeton, NJ: Princeton University Press.

- Bertsekas, D. P. (1987). *Dynamic Programming: Deterministic and Stochastic Models*. Prentice Hall. ISBN: 0132215810.
- Blakemore, C. und G. F. Cooper (1970). "Development of the Brain depends on the Visual Environment". In: *Nature* 228.5270, S. 477–478.
- Bostan, A. C., R. P. Dum und P. L. Strick (2010). "The Basal Ganglia Communicate with the Cerebellum". In: *Proceedings of the National Academy of Sciences* 107.18, S. 8452–8456.
- Boyan, J. A. und A. W. Moore (1995). "Generalization in Reinforcement Learning: Safely Approximating the Value Function." In: *Advances in Neural Information Processing Systems* 7. Bd. 7. MIT Press, S. 369–376.
- Bridle, J. S. (1990). "Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters". In: *Advances in neural information processing systems* 2. Morgan Kaufmann Publishers Inc., S. 211–217. ISBN: 1-55860-100-7.
- Choi, S. P. M., D.-Y. Yeung und N. L. Zhang (1999). "An Environment Model for Nonstationary Reinforcement Learning". In: *Advances in Neural Information Processing Systems* 12. Hrsg. von S. A. Solla, T. K. Leen und K.-R. Müller. The MIT Press, S. 987–993.
- Chung, S. (1965). "Effects of delayed reinforcement in a concurrent situation". In: *Journal of the Experimental Analysis of Behavior* 8.6, S. 439–444.
- Chung, S. und R. J. Herrnstein (1967). "Choice and delay of reinforcement". In: *Journal of the Experimental Analysis of Behavior* 10.1, S. 67–74.
- Cohen, J. D., S. M. McClure und A. J. Yu (2007). "Should I stay or should I go? How the human brain manages the trade-off between exploitation and exploration". In: *Philosophical Transactions of the Royal Society B: Biological Sciences* 362.1481, S. 933–942.
- Daw, N. D., J. P. O'Doherty, P. Dayan, B. Seymour und R. J. Dolan (2006). "Cortical substrates for exploratory decisions in humans". In: *Nature* 441.7095, S. 876–879.
- Dayan, P. (1991). "Reinforcement Comparison". In: *Proceedings of the 1990 Summer School*. San Mateo, CA: Morgan Kaufmann, S. 45–51.
- (2009). "Prospective and Retrospective Temporal Difference Learning". In: *Network* 20.1, S. 32–46.
- Distler, M. (2012). *Können Lernalgorithmen interagieren wie im Gehirn?* Bachelor-Thesis. Technische Universität Darmstadt: Fachgebiet für Intelligente Autonome Systeme.
- Doya, K. (1999). "What are the computations of the cerebellum, the basal ganglia and the cerebral cortex?" In: *Neural Networks* 12.7–8, S. 961–974.

- Doya, K. (2000). "Complementary roles of basal ganglia and cerebellum in learning and motor control". In: *Current Opinion in Neurobiology* 10.6, S. 732–739.
- (2002). "Metalearning and neuromodulation". In: *Neural Networks* 15.4–6, S. 495–506.
- (2007). "Reinforcement learning: Computational theory and biological mechanisms". In: *HFSP Journal* 1.1, S. 30–40.
- (2008). "Modulators of Decision Making". In: *Nature Neuroscience* 11.4, S. 410–416.
- Eck, N. J. van und M. van Wezel (2008). "Application of reinforcement learning to the game of Othello". In: *Computers and Operations Research* 35, S. 1999–2017.
- Ertel, W., M. Schneider, R. Cubek und M. Tokic (2009). "The Teaching-Box: A universal robot learning framework". In: *Proceedings of the 14th International Conference on Advanced Robotics ICAR'09*. S. 1–6.
- Ertel, P., M. Tokic, R. Cubek, H. Voos und D. Söffker (2012). "Towards Learning of Safety Knowledge from Human Demonstrations". In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2012)*. Vilamoura, Algarve, Portugal: IEEE Press, S. 5394–5399.
- Faußer, S. und F. Schwenker (2008). "Neural Approximation of Monte Carlo Policy Evaluation Deployed in Connect Four". In: *Proceedings of the 3rd IAPR Workshop on Artificial Neural Networks in Pattern Recognition*. ANNPR '08. Springer Berlin / Heidelberg, S. 90–100. ISBN: 978-3-540-69938-5.
- (2010). "Learning a Strategy with Neural Approximated Temporal-Difference Methods in English Draughts". In: *Proceedings of the 20th International Conference on Pattern Recognition*. ICPR'10. Istanbul, Turkey: IEEE Computer Society, S. 2925–2928.
- Flentge, F. (2005). "Aktionenlernen mit Selbstorganisierenden Karten und Reinforcement Learning". Diss. University Mainz.
- Främling, K. (2005). "Adaptive robot learning in a non-stationary environment". In: *Proceedings of the 13th European Symposium on Artificial Neural Networks (ESANN'05)*, S. 381–386.
- Gabel, T., M. Riedmiller und F. Trost (2009). "A Case Study on Improving Defense Behavior in Soccer Simulation 2D: The NeuroHassle Approach". In: *RoboCup 2008: Robot Soccer World Cup XII*. Hrsg. von L. Iocchi, H. Matsubara, A. Weitzenfeld und C. Zhou. Bd. 5399. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, S. 61–72.

- Gama, J., R. Camacho, P. Brazdil, A. Jorge und L. Torgo, Hrsg. (2005). *Machine Learning: ECML 2005, 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005, Proceedings*. Bd. 3720. Lecture Notes in Computer Science. Springer Berlin / Heidelberg. ISBN: 3-540-29243-8.
- George, A. P. und W. B. Powell (2006). "Adaptive stepsizes for recursive estimation with applications in approximate dynamic programming". In: *Machine Learning* 65.1, S. 167–198. ISSN: 0885-6125.
- Gordon, G. (1995). "Stable Function Approximation in Dynamic Programming". In: *Proceedings of International Conference on Machine Learning ICML '95*.
- Green, L., A. F. Fry und J. Myerson (1994). "Discounting of Delayed Rewards: A Life-Span Comparison". In: *Psychological Science* 5.1, S. 33–36.
- Groß, A., J. Friedland und F. Schwenker (2008). "Learning to play Tetris applying reinforcement learning methods". In: *Proceedings of ESANN'08, 16th European Symposium on Artificial Neural Networks*, S. 131–136.
- Grześ, M. und D. Kudenko (2010). "Online learning of shaping rewards in reinforcement learning". In: *Neural Networks* 23.4, S. 541–550.
- Gullapalli, V. (1990). "A stochastic reinforcement learning algorithm for learning real-valued functions". In: *Neural Networks* 3.6, S. 671–692.
- Handrich, S., A. Herzog, A. Wolf und C. S. Herrmann (2011). "Combining Supervised, Unsupervised, and Reinforcement Learning in a Network of Spiking Neurons". In: *Advances in Cognitive Neurodynamics (II)*. Hrsg. von R. Wang und F. Gu. Springer Netherlands, S. 163–176.
- Hans, A. (2007). "Untersuchungen zur sicheren Exploration in Reinforcement Learning". Diploma Thesis. Ilmenau, Germany: TU Ilmenau.
- Hans, A., D. Schneegaß, A. M. Schäfer und S. Udluft (2008). "Safe exploration for reinforcement learning". In: *Proceedings of the 16th European Symposium on Artificial Neural Networks ESANN'08*, S. 143–148.
- Hardwick, J. P. und Q. F. Stout (1991). "Bandit strategies for ethical sequential allocation". In: *Computing Science and Statistics* 23, S. 421–424.
- Hasselt, H. v. (2010). "Double Q-learning". In: *Advances in Neural Information Processing Systems* 23. Hrsg. von J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. Zemel und A. Culotta, S. 2613–2621.
- Hebb, D. O (1949). *The organization of behavior: A neuropsychological theory*. John Wiley & Sons.
- Heidrich-Meisner, V. (2009). "Interview with Richard S. Sutton". In: *Künstliche Intelligenz* 3, S. 41–43.

- Hirsch, H. V. B. und D. N. Spinelli (1970). "Visual Experience Modifies Distribution of Horizontally and Vertically Oriented Receptive Fields in Cats". In: *Science* 168.3933, S. 869–871.
- Hoffmann, H., M. Maggio, M. D. Santambrogio, A. Leva und A. Agarwal (2011). *SEEC: A General and Extensible Framework for Self-Aware Computing*. Technical Report MIT-CSAIL-TR-2011-046. Cambridge, MA: Massachusetts Institute of Technology.
- Houk, J. C. und S. P. Wise (1995). "Distributed modular architectures linking basal ganglia, cerebellum, and cerebral cortex: their role in planning and controlling action". In: *Cerebral Cortex* 5.2, S. 95–110.
- Ishii, S., W. Yoshida und J. Yoshimoto (2002). "Control of exploitation-exploration meta-parameter in reinforcement learning". In: *Neural Networks* 15.4–6, S. 665–687.
- Ito, M., M. Sakurai und P. Tongroach (Jan. 1982). "Climbing Fibre Induced Depression of Both Mossy Fibre Responsiveness and Glutamate Sensitivity of Cerebellar Purkinje Cells". In: *The Journal of Physiology* 324.1, S. 113–134.
- Kakvi, S. A. (2009). "Reinforcement Learning for Blackjack". In: *Proceedings of the 8th International Conference on Entertainment Computing*. Hrsg. von S. Natkin und J. Dupire. Bd. 5709. LNCS. Paris, France, S. 300–301. ISBN: 3-642-04051-9.
- Kaplan, F. und P.-Y. Oudeyer (2007). "In search of the neural circuits of intrinsic motivation". In: *Frontiers in Neuroscience* 1 (1), S. 225–236.
- Kietzmann, T. C. und M. Riedmiller (2009). "The Neuro Slot Car Racer: Reinforcement Learning in a Real World Setting". In: *4th International Conference on Machine Learning and Applications (ICMLA'09)*. Los Alamitos, CA, USA: IEEE Computer Society, S. 311–316.
- Kimura, H., K. Miyazaki und S. Kobayashi (1997). "Reinforcement Learning in POMDPs with Function Approximation". In: *Proceedings of the 14th International Conference on Machine Learning (ICML'97)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., S. 152–160.
- Kirkpatrick, S., C. D. Gelatt und M. P. Vecchi (1983). "Optimization by Simulated Annealing". In: *Science* 220.4598, S. 671–680.
- Kobayashi, K., H. Mizoue, T. Kuremoto und M. Obayashi (2009). "A Meta-learning Method Based on Temporal Difference Error". In: *Neural Information Processing (ICONIP 2009)*. Hrsg. von C. S. Leung, M. Lee und J. H. Chan. Lecture Notes in Computer Science 5863. Springer Berlin / Heidelberg, S. 530–537.
- Kober, J., B. Mohler und J. Peters (2010). "Imitation and Reinforcement Learning for Motor Primitives with Perceptual Coupling". In: *From*

- Motor Learning to Interaction Learning in Robots*. Hrsg. von O. Sigaud und J. Peters. Bd. 264. Berlin, Heidelberg: Springer Berlin / Heidelberg, S. 209–225.
- Kocsis, L. und C. Szepesvári (2006). “Bandit Based Monte-Carlo Planning”. In: *Machine Learning: ECML 2006*. Hrsg. von J. Fürnkranz, T. Scheffer und M. Spiliopoulou. Bd. 4212. LNCS. Berlin, Heidelberg: Springer Berlin / Heidelberg, S. 282–293.
- Kretchmar, R. M. und C. W. Anderson (1997). “Comparison of CMACs and Radial Basis Functions for Local Function Approximators in Reinforcement Learning”. In: *Proceedings of the International Conference on Neural Networks*. Houston, TX, S. 834–837.
- Lee, J. W. (2001). “Stock price prediction using reinforcement learning”. In: *ISIE 2001. 2001 IEEE International Symposium on Industrial Electronics Proceedings*. Pusan, South Korea, S. 690–695.
- Littman, M. L., T. L. Dean und L. P. Kaelbling (1995). “On the Complexity of Solving Markov Decision Problems”. In: *Proceedings of the 11th International Conference on Uncertainty in Artificial Intelligence '1995*, S. 394–402.
- Marr, D. (1969). “A theory of cerebellar cortex”. In: *The Journal of Physiology* 202.2, S. 437–470.1.
- McClure, S. M., N. D. Daw und P. R. Montague (2003). “A computational substrate for incentive salience”. In: *Trends in Neurosciences* 26.8, S. 423–428.
- McClure, S. M., M. S. Gilzenrat und J. D. Cohen (2006). “An exploration-exploitation model based on norepinephrine and dopamine activity”. In: *Advances in Neural Information Processing Systems 18*. Hrsg. von Y. Weiss, B. Schölkopf und J. Platt. MIT Press, Cambridge, MA, S. 867–874.
- Mitchell, T. (1997). *Machine Learning*. McGraw Hill.
- Montresor, S., J. Kay, M. Tokic und J. Summerton (2011). “Work In Progress: Programming in a Confined Space - A Case Study in Porting Modern Robot Software to an Antique Platform”. In: *Proceedings of the 41st ASEE/IEEE Frontiers in Education Conference*. Rapid City, SD, USA: IEEE Press, T3H-1–T3H-3.
- Moore, A. W. (1990). *Efficient memory-based learning for robot control*. Technical Report UCAM-CL-TR-209. Cambridge, UK: University of Cambridge, Computer Laboratory.
- Morris, G., A. Nevet, D. Arkadir, E. Vaadia und H. Bergman (2006). “Mid-brain dopamine neurons encode decisions for future action”. In: *Nature Neuroscience* 9.8, S. 1057–1063. (Besucht am 22. 10. 2012).

- Neller, T. W., C. G. M. Presser, I. Russell und Z. Markov (2006). "Pedagogical possibilities for the dice game pig". In: *Journal of Computing Sciences in Colleges* 21.6, S. 149–161. ISSN: 1937-4771.
- Ng, A., A. Coates, M. Diel, V. Ganapathi, J. Schulte, B. Tse, E. Berger und E. Liang (2004). "Autonomous Inverted Helicopter Flight via Reinforcement Learning". In: *Proceedings of the International Symposium on Experimental Robotics*, S. 363–372.
- Ngo, H., M. Luciw, A. Förster und J. Schmidhuber (2012). "Learning Skills from Play: Artificial Curiosity on a Katana Robot Arm". In: *Proceedings of the International Joint Conference of Neural Networks (IJCNN 2012)*. Brisbane, Australia, S. 1–8.
- Nissen, S. (2007). "Large Scale Reinforcement Learning using Q-SARSA(λ) and Cascading Neural Networks". Master Thesis. Denmark: University of Copenhagen.
- Niv, Y. (2009). "Reinforcement learning in the brain". In: *Journal of Mathematical Psychology* 53.3, S. 139–154. ISSN: 0022-2496.
- Niv, Y., N. D. Daw und P. Dayan (2006). "Choice values". In: *Nature Neuroscience* 9.8, S. 987–988.
- Oudeyer, P.-Y. und F. Kaplan (2007). "What is intrinsic motivation? A typology of computational approaches". In: *Frontiers in Neurorobotics* 1 (6), S. 1–14.
- Oudeyer, P.-Y., F. Kaplan und V. Hafner (2007). "Intrinsic Motivation Systems for Autonomous Mental Development". In: *IEEE Transactions on Evolutionary Computation* 11.2, S. 265–286.
- Pape, L., C. M. Oddo, M. Controzzi, A. Förster und J. Schmidhuber (2012). "Learning tactile skills through curious exploration". In: *Frontiers in Neurorobotics* 6, S. 1–16.
- Pavlov, I. P. (1927). *Conditioned Reflexes - An Investigation of The Physiological Activity of the Cerebral Cortex*. Translated and edited by G. V. Anrep. London: Oxford University Press.
- Pérez-Urbe, A. und E. Sanchez (1998). "Blackjack as a test bed for learning strategies in neural networks". In: *Proceedings of the IEEE International Joint Conference on Neural Networks Proceedings*. Anchorage, AK, USA, S. 2022–2027.
- Peters, J. und S. Schaal (2008). "Reinforcement learning of motor skills with policy gradients". In: *Neural Networks* 21.4, S. 682–697.
- Pfeiffer, M. (2004). "Reinforcement Learning of Strategies for Settlers of Catan". In: *Proceedings of the International Conference on Computer Games: Artificial Intelligence, Design and Education*.

- Powell, W. B. (2007). *Approximate Dynamic Programming: Solving the Curses of Dimensionality* (Wiley Series in Probability and Statistics). Wiley-Interscience. ISBN: 0470171553.
- Precup, D. und R. S. Sutton (1997). "Exponentiated Gradient Methods for Reinforcement Learning". In: *Proceedings of the Fourteenth International Conference on Machine Learning*. Morgan Kaufmann Publishers Inc., S. 272–277.
- Precup, D., R. S. Sutton und S. Dasgupta (2001). "Off-Policy Temporal Difference Learning with Function Approximation". In: *Proceedings of the 18th International Conference on Machine Learning*. Morgan Kaufmann Publishers Inc., S. 417–424.
- Puterman, M. L. und M. C. Shin (1978). "Modified Policy Iteration Algorithms for Discounted Markov Decision Problems". In: *Management Science* 24.11, S. 1127–1137.
- Rescorla, R. und A. Wagner (1972). "A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement and nonreinforcement". In: *Classical conditioning II: Current research and theory*. Hrsg. von A. Black und W. Prokasy. New York: Appleton-Century-Crofts, S. 64–99.
- Riedmiller, M. (2005). "Neural Fitted Q Iteration – First Experiences with a Data Efficient Neural Reinforcement Learning Method". In: *Machine Learning: ECML 2005*. Bd. 3720. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, S. 317–328.
- Riedmiller, M., M. Montemerlo und H. Dahlkamp (2007). "Learning to Drive a Real Car in 20 Minutes". In: *Proceedings of the FBIT 2007 conference*. Jeju, Korea: Springer Berlin / Heidelberg.
- Robbins, H. (1952). "Some Aspects of the Sequential Design of Experiments". In: *Bulletin of the American Mathematical Society* 58, S. 527–535.
- Robbins, H. und S. Monro (1951). "A Stochastic Approximation Method". In: *The Annals of Mathematical Statistics* 22.3, S. 400–407. ISSN: 00034851.
- Roesch, M. R., D. J. Calu und G. Schoenbaum (2007). "Dopamine neurons encode the better option in rats deciding between differently delayed or sized rewards". In: *Nature Neuroscience* 10.12, S. 1615–1624.
- Röttger, M. C. (2009). "Reinforcement Learning für kontinuierliche Zustands- und Aktionsräume unter Berücksichtigung der wissenschaftlichen Informationsverarbeitung". Diss. Freiburg, Deutschland: University Freiburg.
- Rummery, G. A. und M. Niranjan (1994). *On-Line Q-Learning Using Connectionist Systems*. Technical Report CUED/F-INFENG/TR 166. Cambridge University.

-
- Russell, S. und P. Norvig (2009). *Artificial Intelligence: A Modern Approach*. 3. Aufl. Prentice Hall.
- Sasakawa, T., J. Hu und K. Hirasawa (2008). "A brainlike learning system with supervised, unsupervised, and reinforcement learning". In: *Electrical Engineering in Japan* 162.1, S. 32–39.
- Schultz, W., P. Apicella und T. Ljungberg (1993). "Responses of Monkey Dopamine Neurons to Reward and Conditioned Stimuli During Successive Steps of Learning a Delayed Response Task". In: *The Journal of Neuroscience* 13.3, S. 900–913.
- Schultz, W. (1998). "Predictive Reward Signal of Dopamine Neurons". In: *Journal of Neurophysiology* 80.1, S. 1–27.
- Schultz, W., P. Dayan und P. R. Montague (1997). "A Neural Substrate of Prediction and Reward". In: *Science* 275.5306, S. 1593–1599.
- Schweighofer, N. und K. Doya (2003). "Meta-learning in Reinforcement Learning". In: *Neural Networks* 16.1, S. 5–9.
- Simsek, O. und A. G. Barto (2006). "An intrinsic reward mechanism for efficient exploration". In: *Proceedings of the 23rd International Conference on Machine Learning*, S. 833–840.
- Singh, S. und R. S. Sutton (1996). "Reinforcement Learning with Replacing Eligibility Traces". In: *Machine Learning* 22, S. 123–158.
- Skinner, B. F. (1953). *Science And Human Behavior*. New York: Macmillan.
- Stone, P., R. S. Sutton und G. Kuhlmann (2005). "Reinforcement Learning for RoboCup Soccer Keepaway". In: *Adaptive Behavior* 13.3, S. 165–188.
- Sutton, R. S. (1984). "Temporal credit assignment in reinforcement learning". Diss. University of Massachusetts Amherst.
- (1988). "Learning to predict by the methods of temporal differences". In: *Machine Learning* 3.1, S. 9–44.
- (1990). "Integrated architectures for learning, planning, and reacting based on approximating dynamic programming". In: *Proceedings of the Seventh International Conference on Machine Learning*. Morgan Kaufmann, S. 216–224.
- (1996). "Generalization in Reinforcement Learning: Successful Examples Using Sparse Coarse Coding". In: *Advances in Neural Information Processing Systems* 8. Bd. 8. MIT Press, S. 1038–1044.
- Sutton, R. S. und A. G. Barto (1998). *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.
- Sutton, R. S., D. A. McAllester, S. P. Singh und Y. Mansour (2000). "Policy Gradient Methods for Reinforcement Learning with Function Approximation". In: *Advances in Neural Information Processing Systems 12 (NIPS)*

- 1999). Hrsg. von S. A. Solla, T. K. Leen und K. Müller. MIT Press, S. 1057–1063.
- Tesauro, G. (2002). “Programming backgammon using self-teaching neural nets”. In: *Artificial Intelligence* 134.1-2, S. 181–199.
- Thathachar, M. und P. Sastry (1985). “A new approach to the design of reinforcement schemes for learning automata”. In: *IEEE Transactions on Systems, Man and Cybernetics* 15, S. 168–175.
- Thomaz, A. L. und C. Breazeal (2008). “Teachable robots: Understanding human teaching behavior to build more effective robot learners”. In: *Artificial Intelligence* 172.6–7, S. 716–737.
- Thorndike, E. L. (1911). *Animal Intelligence*. The Macmillan company, New York.
- Thrun, S. (1995). “Learning To Play the Game of Chess”. In: *Advances in Neural Information Processing Systems* 7. Hrsg. von G. Tesauro, D. Touretzky und T. Leen. MIT Press, S. 1069–1076.
- Thrun, S. B. (1992). *Efficient Exploration In Reinforcement Learning*. Techn. Ber. Carnegie Mellon University.
- Thrun, S. B. und K. Möller (1992). “Active Exploration in Dynamic Environments”. In: *Advances in Neural Information Processing Systems* 4. San Mateo, CA: Morgan Kaufmann.
- Timmer, S. und M. Riedmiller (2007). “Fitted Q Iteration with CMACs”. In: *Proceedings of the International Symposium on Approximate Dynamic Programming and Reinforcement Learning (ADPRL)*. Honolulu, USA.
- Tokic, M. (2010). “Adaptive e-Greedy Exploration in Reinforcement Learning Based on Value Differences”. In: *KI 2010: Advances in Artificial Intelligence*. Hrsg. von R. Dillmann, J. Beyerer, U. Hanebeck und T. Schultz. Bd. 6359. Lecture Notes in Artificial Intelligence. Springer Berlin / Heidelberg, S. 203–210.
- (2013). “Reinforcement Learning: Psychologische und neurobiologische Aspekte”. In: *Künstliche Intelligenz* 27.3, S. 213–219.
- Tokic, M. und H. Bou Ammar (2012). “Teaching Reinforcement Learning using a Physical Robot”. In: *Proceedings of the Workshop on Teaching Machine Learning at the 29th International Conference on Machine Learning*. Edinburgh, UK, S. 1–4.
- Tokic, M. und G. Palm (2011). “Value-Difference Based Exploration: Adaptive Control between Epsilon-Greedy and Softmax”. In: *KI 2011: Advances in Artificial Intelligence*. Hrsg. von J. Bach und S. Edelkamp. Bd. 7006. Lecture Notes in Artificial Intelligence. Springer Berlin / Heidelberg, S. 335–346.

- Tokic, M. und G. Palm (2012a). "Adaptive Exploration Using Stochastic Neurons". In: *Artificial Neural Networks and Machine Learning – ICANN 2012*. Hrsg. von A. Villa, W. Duch, P. Érdi, F. Masulli und G. Palm. Bd. 7553. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, S. 42–49.
- (2012b). "Gradient Algorithms for Exploration/Exploitation Trade-Offs: Global and Local Variants". In: *Artificial Neural Networks in Pattern Recognition*. Hrsg. von N. Mana, F. Schwenker und E. Trentin. Bd. 7477. Lecture Notes in Artificial Intelligence. Springer Berlin / Heidelberg, S. 60–71.
- Tokic, M., W. Ertel, H. Radtke, J. Akmal und W. Krökel (2006). "Reinforcement Learning on a Simple Real Walking Robot". In: *Proceedings of the 29th Annual German Conference on Artificial Intelligence*. Bremen, Germany, S. 1–3.
- Tokic, M., J. Fessler und W. Ertel (2009). "The Crawler, A Class Room Demonstrator for Reinforcement Learning". In: *Proceedings of the 22th International Florida Artificial Intelligence Research Society Conference FLAIRS'09*. Hrsg. von C. Lane und H. Guesgen. Menlo Park, California, USA: AAAI Press, S. 160–165.
- Tokic, M., A. Usadel, J. Fessler und W. Ertel (2010a). "On an educational approach to behavior learning for robots". In: *Proceedings of the 1st International Conference on Robotics in Education*. Bratislava, Slovak Republic: Slovak University of Technology in Bratislava, S. 171–176.
- (2010b). "On an educational approach to behavior learning for robots". In: *AT&P Journal Plus* 2010.2, S. 103–108. ISSN: 1336-5010.
- Tokic, M., P. Ertle, G. Palm, D. Söffker und H. Voos (2012). "Robust Exploration/Exploitation Trade-Offs in Safety-Critical Applications". In: *Proceedings of the 8th International Symposium on Fault Detection, Supervision and Safety of Technical Processes*. Mexico City, Mexico: IFAC, S. 660–665.
- Tsumoto, T. und K. Suda (1979). "Cross-depression: an electrophysiological manifestation of binocular competition in the developing visual cortex". In: *Brain Research* 168.1, S. 190–194.
- Varges, S., G. Riccardi, S. Quarteroni und A. Ivanov (2009). "The Exploration/Exploitation Trade-off in Reinforcement Learning for Dialogue Management". In: *Proceedings of the IEEE Workshop on Automatic Speech Recognition & Understanding ASRU'2009*. Merano: IEEE Press, S. 479–484.
- Vermorel, J. und M. Mohri (2005). "Multi-armed Bandit Algorithms and Empirical Evaluation". In: *Machine Learning: ECML 2005*. Hrsg. von J.

- Gama, R. Camacho, P. Brazdil, A. Jorge und L. Torgo. Bd. 3720. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, S. 437–448. ISBN: 3-540-29243-8.
- Vitay, J., J. Fix, F. Beuth, H. Schroll und F. Hamker (2009). "Biological Models of Reinforcement Learning". In: *Künstliche Intelligenz* 03/2009, S. 12–18.
- Vollbrecht, H. (2000). "Hierarchic function approximation in kd-Q-learning". In: *Proceedings of the Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies (KES'2000)*. Brighton, UK, S. 466–469.
- (2003). "Hierarchical Reinforcement Learning in Continuous State Spaces". Diss. Ulm, Germany: University of Ulm.
- Wardle, F. (1987). "Getting Back to the Basics of Children's Play". In: *Child Care Information Exchange* 57, S. 27–30.
- Watkins, C. (1989). "Learning from Delayed Rewards". Diss. Cambridge, England: University of Cambridge.
- Watkins, C. und P. Dayan (1992). "Technical Note: Q-Learning". In: *Machine Learning* 8.3, S. 279–292.
- Watson, J. B. und R. Rayner (1920). "Conditioned emotional reactions". In: *Journal of Experimental Psychology* 3.1, S. 1–14.
- Whitehead, S. D. (1991a). *A Study of Cooperative Mechanisms for Faster Reinforcement Learning*. Technical Report 365. Rochester, NY: University of Rochester, Computer Science Department.
- (1991b). "Complexity and Cooperation in Q-Learning". In: *Proceedings of the Eight International Workshop on Machine Learning*, S. 363–367.
- Wiering, M. (1999). "Explorations in Efficient Reinforcement Learning". Diss. Amsterdam: University of Amsterdam.
- Williams, R. J. (1992). "Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning". In: *Machine Learning* 8, S. 229–256.

Anhang A

Plots

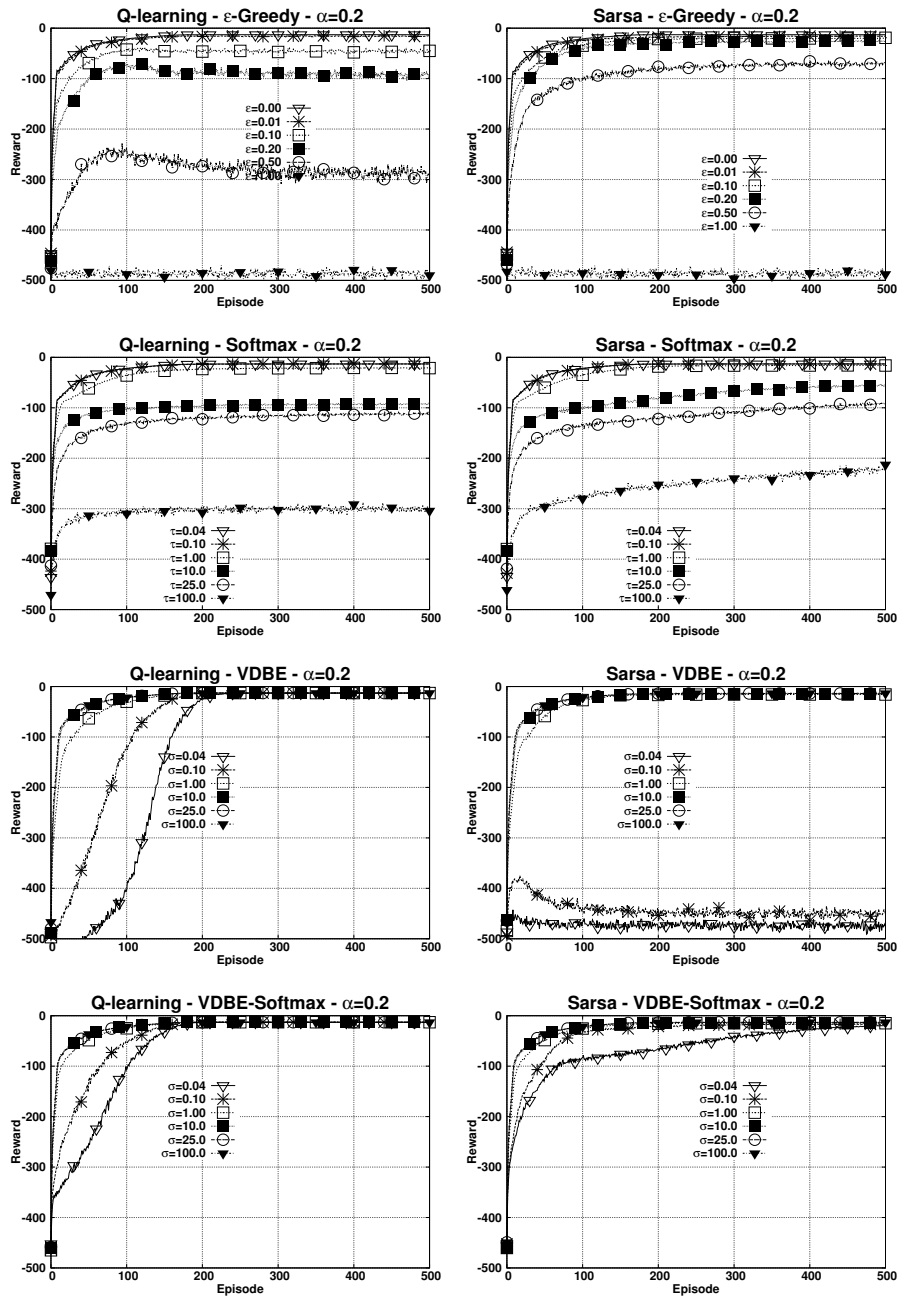


Abbildung A.1: Cliff-Walking-Problem: Return für Lernrate $\alpha = 0.2$ über 1000 Experimente gemittelt.

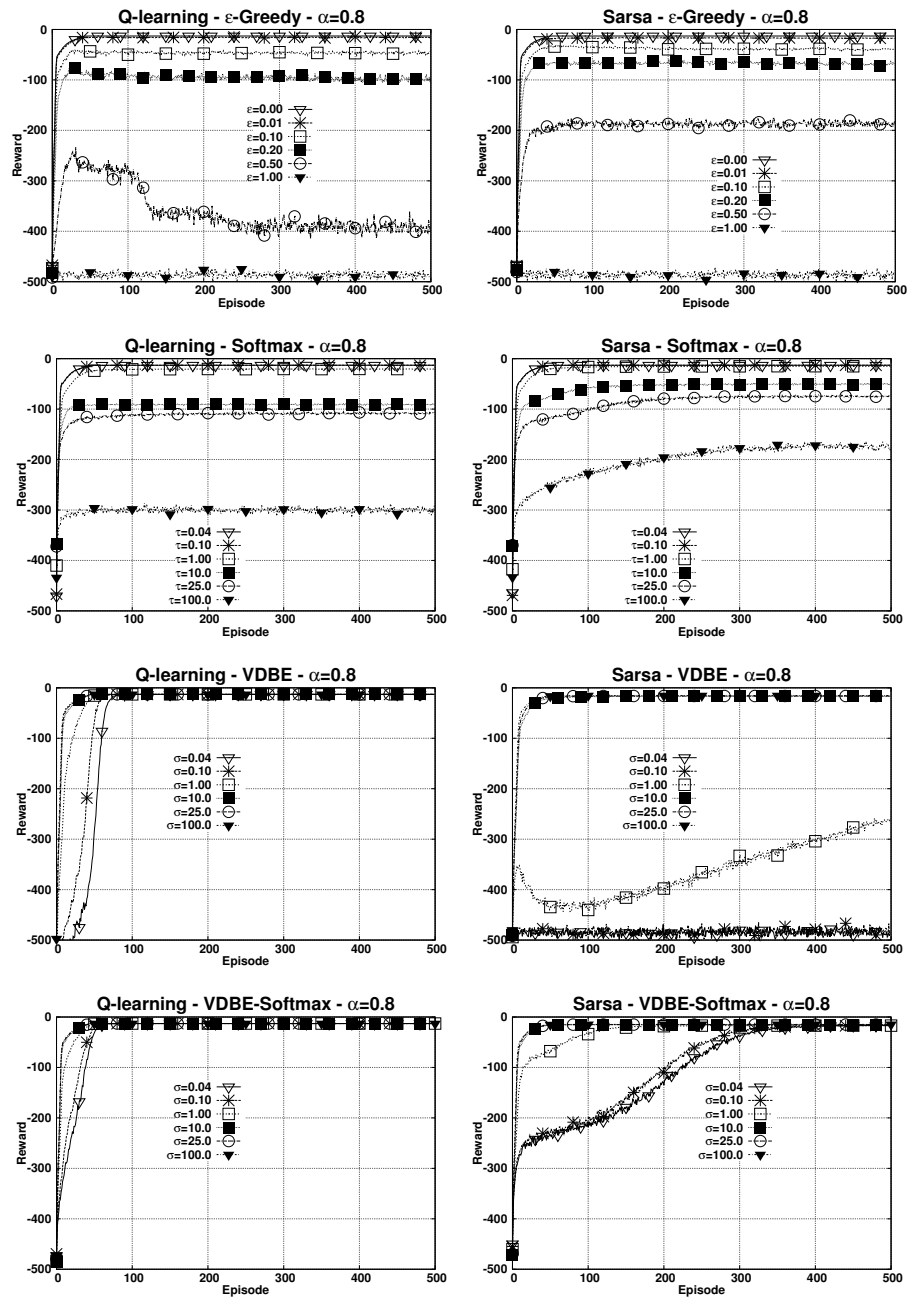


Abbildung A.2: Cliff-Walking-Problem: Return für Lernrate $\alpha = 0.8$ über 1000 Experimente gemittelt.

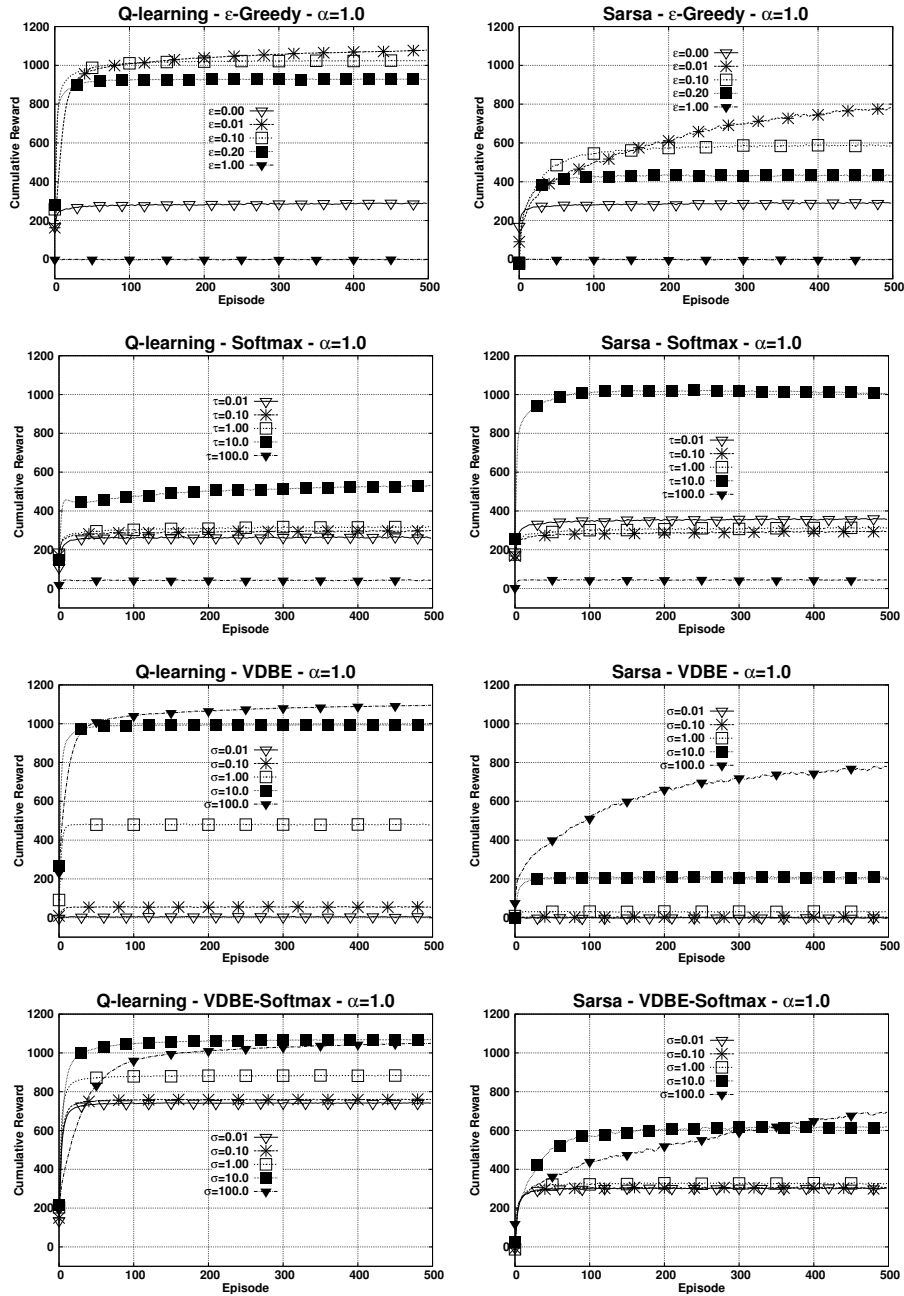


Abbildung A.3: Laufroboter: Return über 2000 Experimente gemittelt.

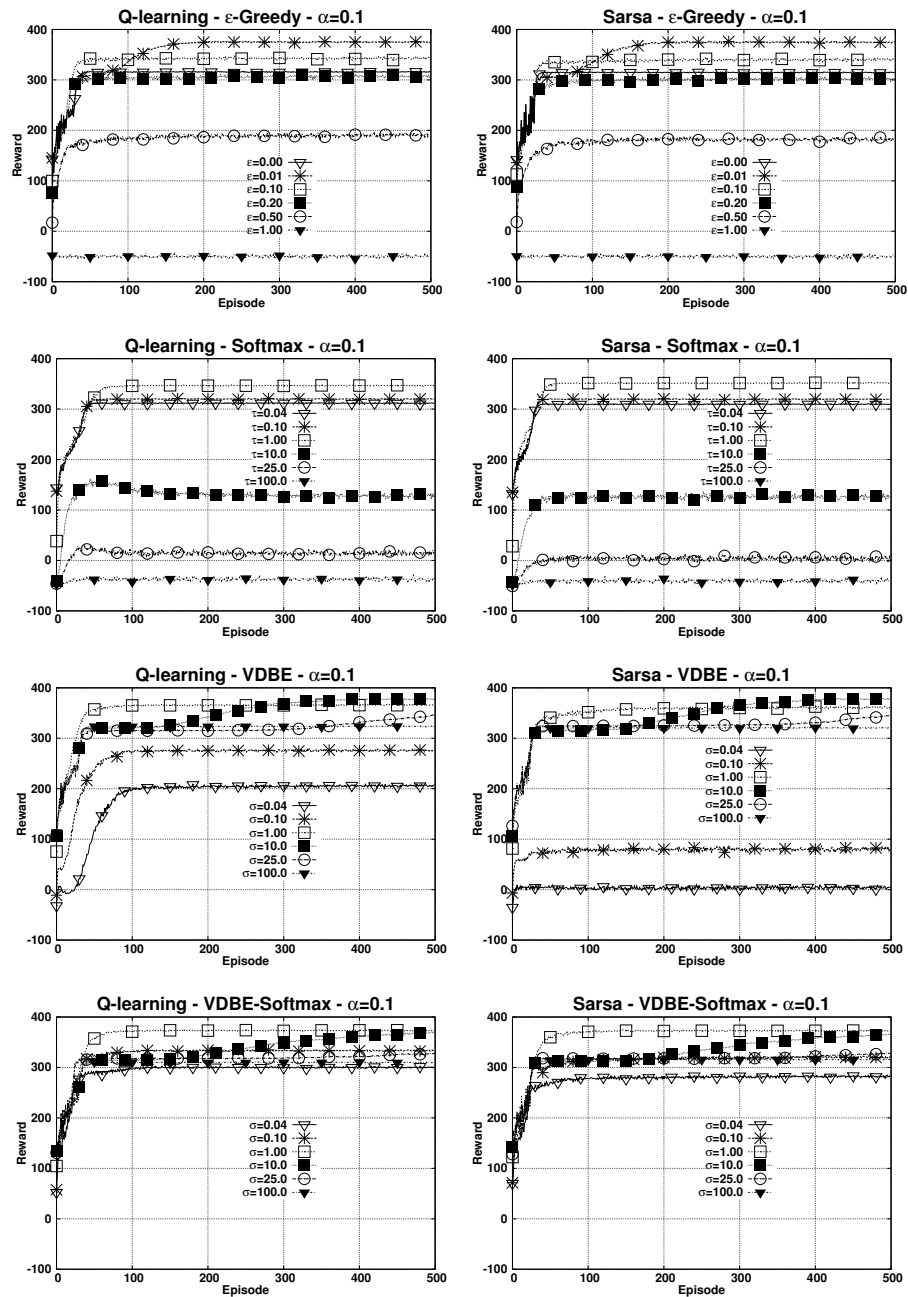


Abbildung A.4: Bandit-World-Problem: Return über 500 Experimente gemittelt.

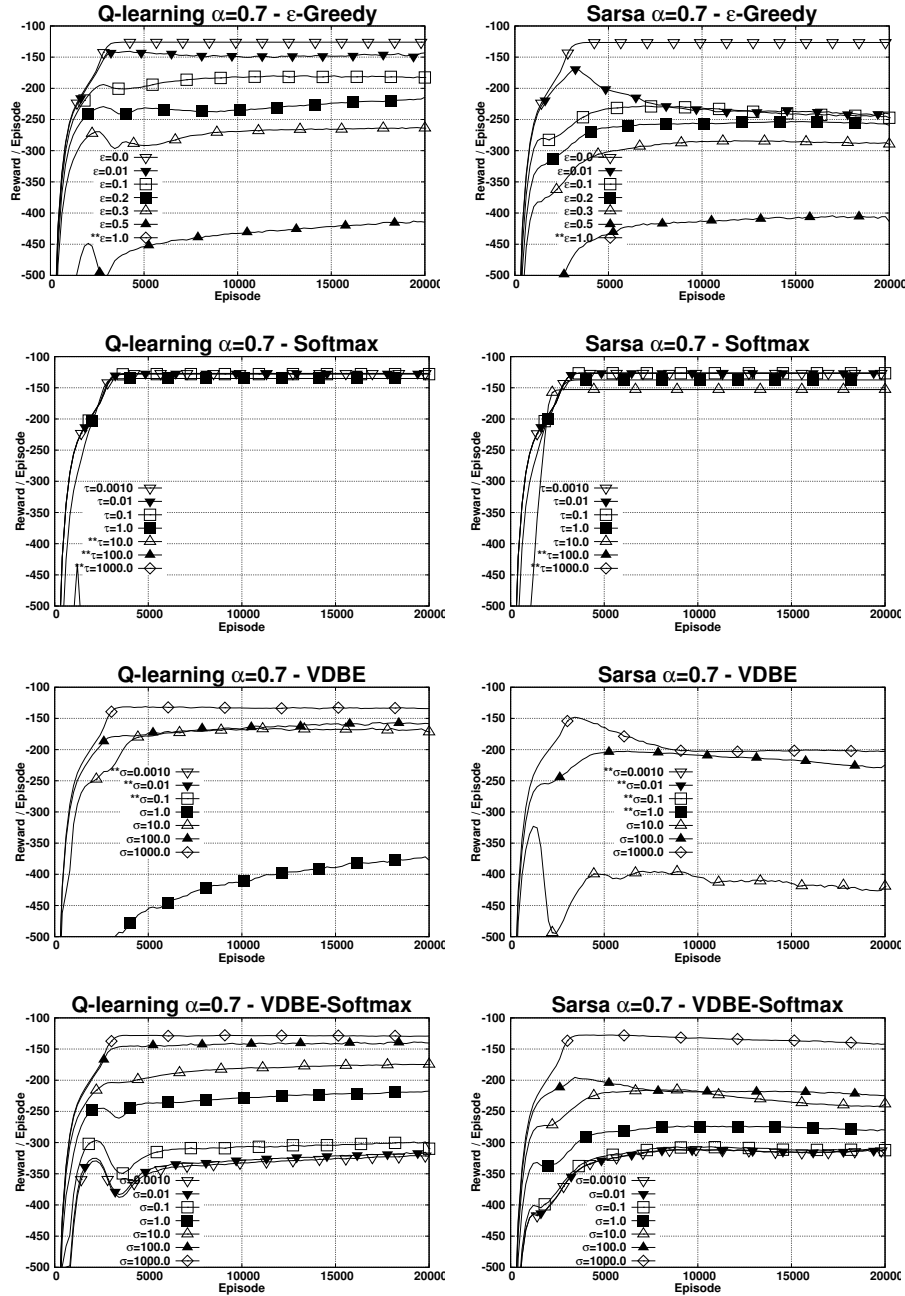


Abbildung A.5: Mountain-Car-Problem: Return über 2000 Experimente gemittelt (und geglättet). Aufgrund von schlechten Ergebnissen liegen Kurven mit ** gekennzeichneten Parametern außerhalb des Darstellungsbereichs.

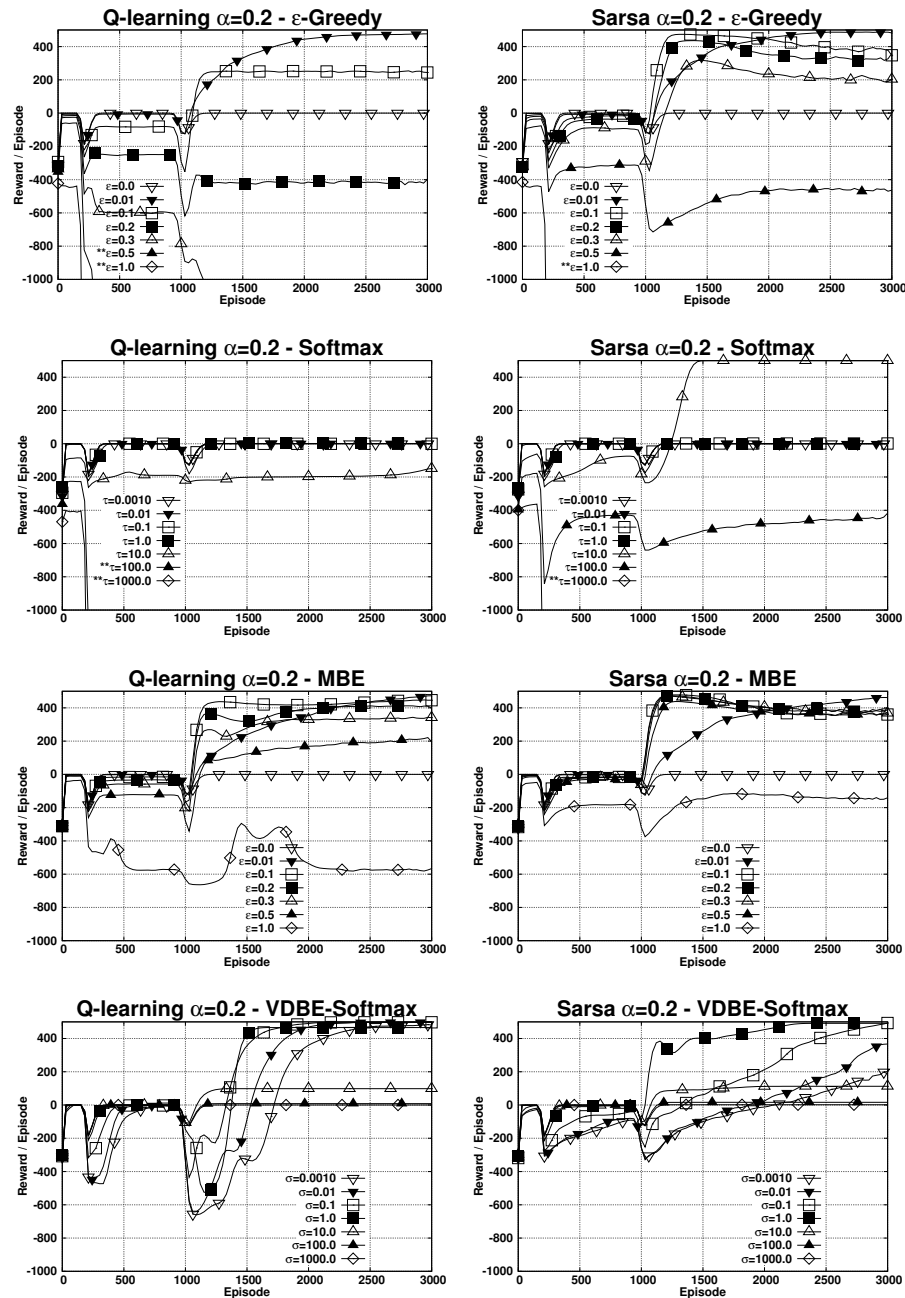
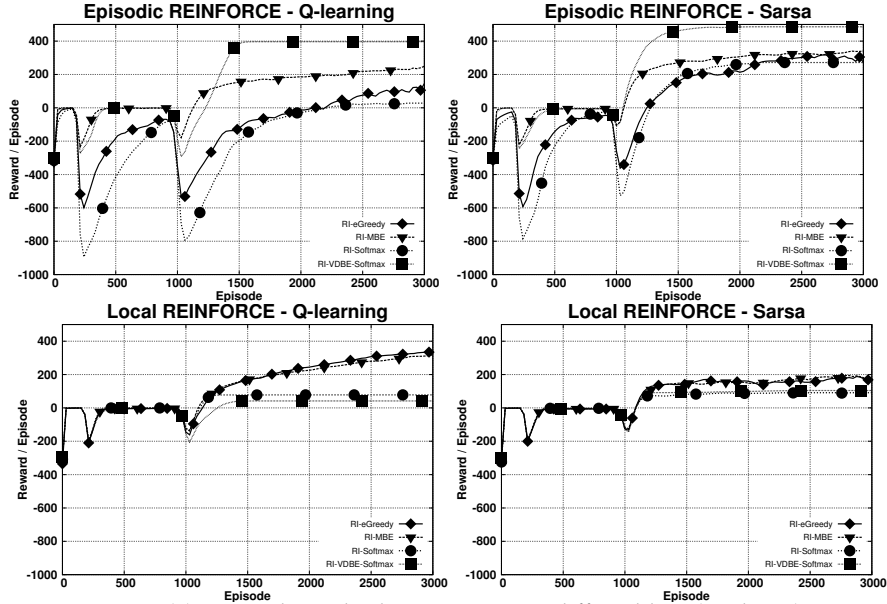
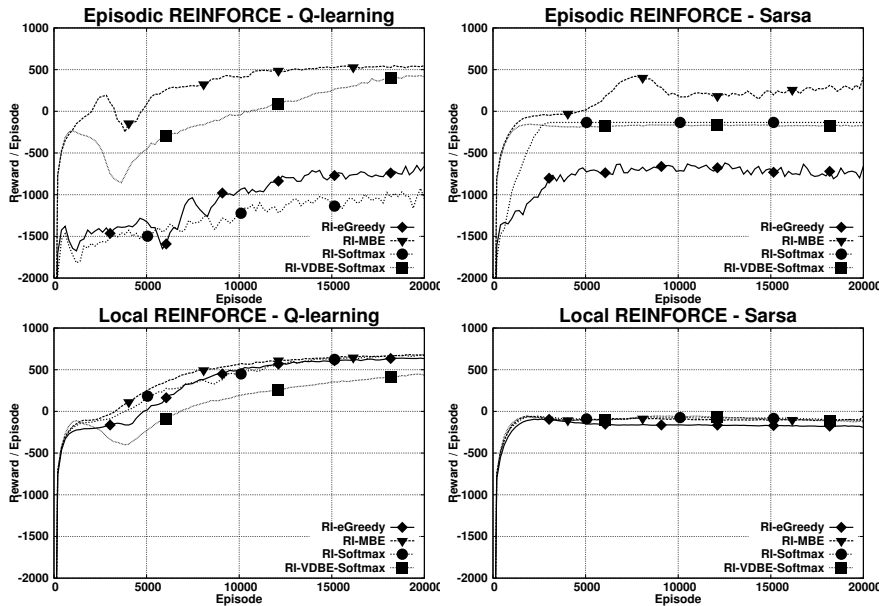


Abbildung A.6: Dynamic-Cliff-Problem: Return der Basisstrategien über 500 Experimente gemittelt. Aufgrund von schlechten Ergebnissen liegen Kurven mit ** gekennzeichneten Parametern außerhalb des Darstellungsbereichs.



(a) Reward-Vergleich am Dynamic-Cliff-Problem (geglättet).



(b) Reward-Vergleich am Mountain-Car-Problem mit zwei Zielzuständen (geglättet).

Abbildung A.7: Dynamic-Cliff- sowie Mountain-Car-Problem mit zwei Zielzuständen: Return für REINFORCE-adaptierte Basisstrategien.

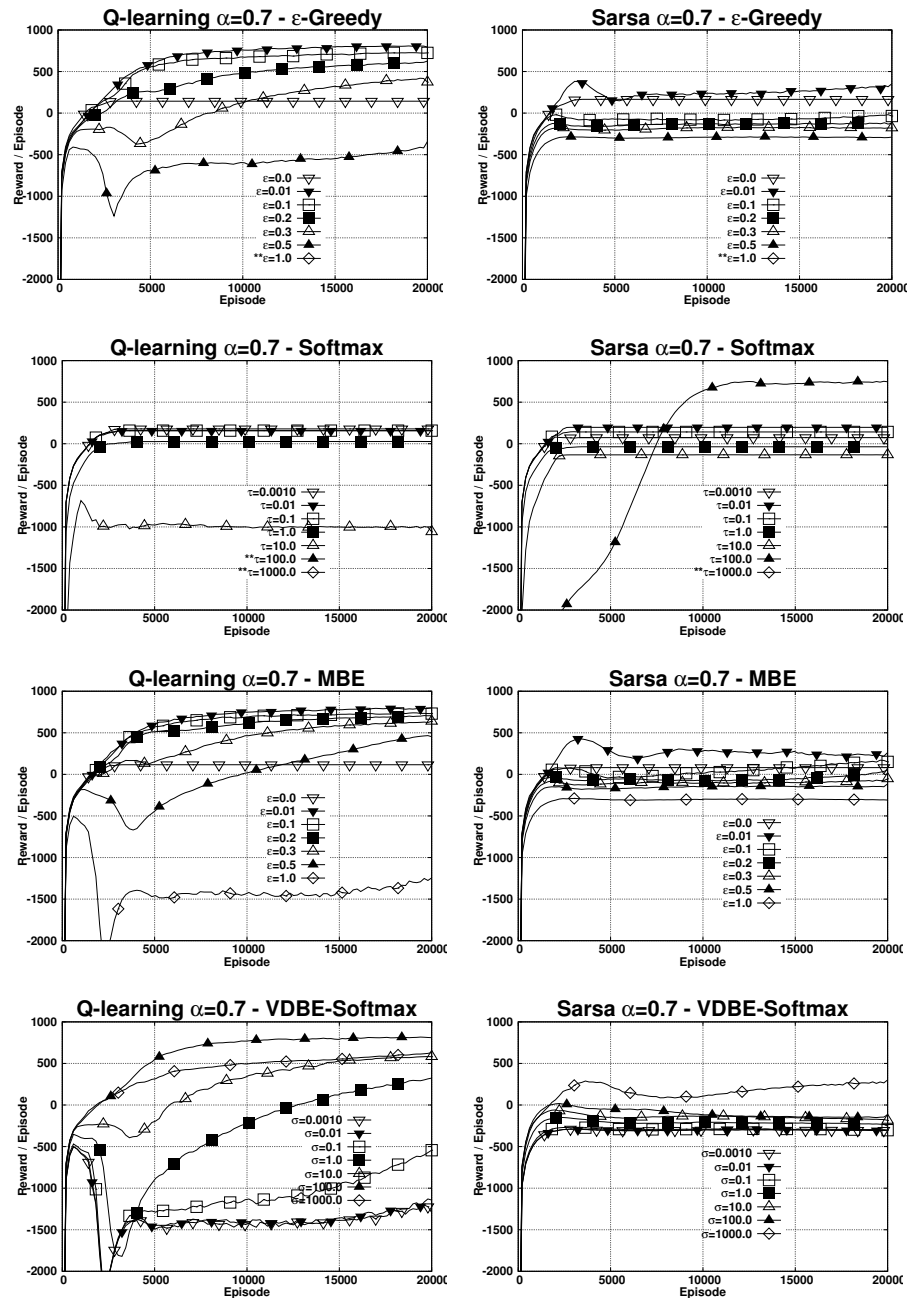


Abbildung A.8: Mountain-Car-Problem mit zwei Zielzuständen: Return der Basisstrategien über 200 Experimente gemittelt. Aufgrund von schlechten Ergebnissen liegen Kurven mit ** gekennzeichneten Parametern außerhalb des Darstellungsbereichs.

Anhang B

Lebenslauf

Michel Tokic

■ Persönliche Daten

Geburtsdatum 08. Dezember 1980

Nationalität Deutsch

■ Arbeitserfahrung

Aktuelle Stellen

seit 09.2006 **Wissenschaftlicher Mitarbeiter**, *Hochschule Ravensburg-Weingarten*, Weingarten.

seit 09.2009 **Lehrbeauftragter**, *Hochschule Ravensburg-Weingarten, Fakultät für Elektrotechnik und Informatik*, Weingarten.

Vorlesung: Angewandte Neuroinformatik (seit WS13/14) / Systemadministration (WS09/10 bis SS13)

Frühere Stellen

- 01.2002 **Freiberufler**, *Tettnang*, Server- und Netzwerkadministration.
- 12.2010 u.A. für <http://www.inkidu.de>, sowie mehrere Unternehmen in den Regionen Bodenseekreis und Oberschwaben.
- 09.2004 **Praktikant**, *ND SatCom AG, Abteilung: Embedded Systems*, Immenstaad.
- 02.2005 Evaluation von Echtzeit-Anwendungen unter VxWorks und Embedded Linux.
- 10.2002 **Praktikant**, *Miroglio S.p.A.*, Alba / Italien.
- 01.2003 Entwicklung einer Netzwerk-Software zur Überwachung von Router und Switches.
- 09.2002 **Stipendiat**, *InWEnt (Internationale Weiterbildung und Entwicklung, gemeinnützige GmbH, Köln)*, Florenz und Alba / Italien.
- 01.2003 4-wöchige Sprachschule für Italienisch in Florenz, 8-wöchiges Praktikum bei Miroglio S.p.A. in Alba.

■ Ausbildung

2008–2013 **Dr. rer. nat.**, *Universität Ulm*.

Titel: Reinforcement Learning mit adaptiver Steuerung von Exploration und Exploitation; Erstgutachter: Prof. Dr. G. Palm (Universität Ulm / Institut für Neuroinformatik); Zweitgutachter: Prof. Dr. Martin Riedmiller (Universität Freiburg / Machine Learning Lab)

2006–2008 **M.Sc.**, *Fakultät Elektrotechnik und Informatik*, Hochschule Ravensburg-Weingarten, Studiengang: Informatik.

Thema der Masterarbeit: Reinforcement Learning an Robotern mit neuronalen Netzen

2002–2006 **Dipl.-Inform.(FH)**, *Fakultät Elektrotechnik und Informatik*, Studiengang: *Angewandte Informatik*, Hochschule Ravensburg-Weingarten.

Thema der Diplomarbeit: Entwicklung eines lernenden Laufroboters

- 2000–2002 **staatl. gepr. Techniker für Medien- und Informationssysteme**, Elektronikschule Tettnang.
- 1997–1999 **staatl. gepr. Assistent für Kommunikations- und Informationstechnik**, Elektronikschule Tettnang.

■ Auszeichnungen

- 2012 **Nominiert (1 von 4) für den „New Technology Foundation Award for Entertainment Robots and Systems“**, Für unser Paper „Towards learning of safety knowledge from human demonstrations“ auf der IROS 2012 (1801 Submissions, 812 akzeptierte Paper)., Vilamoura/Portugal.
- 2007 **LISTA-Innovationspreis (3. Rang mit 5000,- SFR) der LISTA AG Erlen/Schweiz**, Für meine Diplomarbeit: Entwicklung eines lernenden Laufroboters.
- 2006 **Bester Absolvent des Studiengangs Angewandte Informatik**, Hochschule Ravensburg-Weingarten, Weingarten.

■ Mitgliedschaften

- IEEE Technical Committee on Robot Learning
- Natur- und Bewegungskindergarten Tettnang e.V., Mitglied des Vereinsvorstands (Kassierer)

■ Akademischer Service

- *Reviewer*, 41st ASEE/IEEE Frontiers in Education Conference (FIE'11)
- *Reviewer*, 5th INNS IAPR TC3 GIRPR International Workshop on Artificial Neural Networks in Pattern Recognition (ANNPR'12)
- *Guest Reviewer*, Pattern Recognition Letters, Special Issue on Partially Supervised Learning, 2012/2013, Elsevier

Anhang C

Veröffentlichungen

Teile dieser Dissertation wurden bereits in folgenden Fachartikeln veröffentlicht:

- Tokic, M., W. Ertel, H. Radtke, J. Akmal und W. Krökel (2006). "Reinforcement Learning on a Simple Real Walking Robot". In: *Proceedings of the 29th Annual German Conference on Artificial Intelligence*. Bremen, Germany, S. 1–3.
- Ertel, W., M. Schneider, R. Cubek und M. Tokic (2009). "The Teaching-Box: A universal robot learning framework". In: *Proceedings of the 14th International Conference on Advanced Robotics ICAR'09*. S. 1–6.
- Tokic, M., J. Fessler und W. Ertel (2009). "The Crawler, A Class Room Demonstrator for Reinforcement Learning". In: *Proceedings of the 22th International Florida Artificial Intelligence Research Society Conference FLAIRS'09*. Hrsg. von C. Lane und H. Guesgen. Menlo Park, California, USA: AAAI Press, S. 160–165.
- Tokic, M. (2010). "Adaptive e-Greedy Exploration in Reinforcement Learning Based on Value Differences". In: *KI 2010: Advances in Artificial Intelligence*. Hrsg. von R. Dillmann, J. Beyerer, U. Hanebeck und T. Schultz. Bd. 6359. Lecture Notes in Artificial Intelligence. Springer Berlin / Heidelberg, S. 203–210.
- Tokic, M., A. Usadel, J. Fessler und W. Ertel (2010a). "On an educational approach to behavior learning for robots". In: *Proceedings of the*

1st International Conference on Robotics in Education. Bratislava, Slovak Republic: Slovak University of Technology in Bratislava, S. 171–176.

- Tokic, M., A. Usadel, J. Fessler und W. Ertel (2010b). “On an educational approach to behavior learning for robots”. In: *AT&P Journal Plus* 2010.2, S. 103–108. ISSN: 1336-5010.
- Tokic, M. und G. Palm (2011). “Value-Difference Based Exploration: Adaptive Control between Epsilon-Greedy and Softmax”. In: *KI 2011: Advances in Artificial Intelligence*. Hrsg. von J. Bach und S. Edelkamp. Bd. 7006. Lecture Notes in Artificial Intelligence. Springer Berlin / Heidelberg, S. 335–346.
- Montresor, S., J. Kay, M. Tokic und J. Summerton (2011). “Work In Progress: Programming in a Confined Space - A Case Study in Porting Modern Robot Software to an Antique Platform”. In: *Proceedings of the 41st ASEE/IEEE Frontiers in Education Conference*. Rapid City, SD, USA: IEEE Press, T3H-1–T3H-3.
- Tokic, M. und H. Bou Ammar (2012). “Teaching Reinforcement Learning using a Physical Robot”. In: *Proceedings of the Workshop on Teaching Machine Learning at the 29th International Conference on Machine Learning*. Edinburgh, UK, S. 1–4.
- Tokic, M. und G. Palm (2012b). “Gradient Algorithms for Exploration/Exploitation Trade-Offs: Global and Local Variants”. In: *Artificial Neural Networks in Pattern Recognition*. Hrsg. von N. Mana, F. Schwenker und E. Trentin. Bd. 7477. Lecture Notes in Artificial Intelligence. Springer Berlin / Heidelberg, S. 60–71.
- Tokic, M. und G. Palm (2012a). “Adaptive Exploration Using Stochastic Neurons”. In: *Artificial Neural Networks and Machine Learning – ICANN 2012*. Hrsg. von A. Villa, W. Duch, P. Érdi, F. Masulli und G. Palm. Bd. 7553. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, S. 42–49.
- Tokic, M., P. Ertle, G. Palm, D. Söffker und H. Voos (2012). “Robust Exploration/Exploitation Trade-Offs in Safety-Critical Applications”. In: *Proceedings of the 8th International Symposium on Fault Detection, Supervision and Safety of Technical Processes*. Mexico City, Mexico: IFAC, S. 660–665.
- Ertle, P., M. Tokic, R. Cubek, H. Voos und D. Söffker (2012). “Towards Learning of Safety Knowledge from Human Demonstrations”. In:

Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2012). Vilamoura, Algarve, Portugal: IEEE Press, S. 5394–5399.

- Tokic, M. (2013). “Reinforcement Learning: Psychologische und neurobiologische Aspekte”. In: *Künstliche Intelligenz* 27.3, S. 213–219.