Künstliche Intelligenz

Markov-Entscheidungsprozesse

Dr.-Ing. Stefan Lüdtke

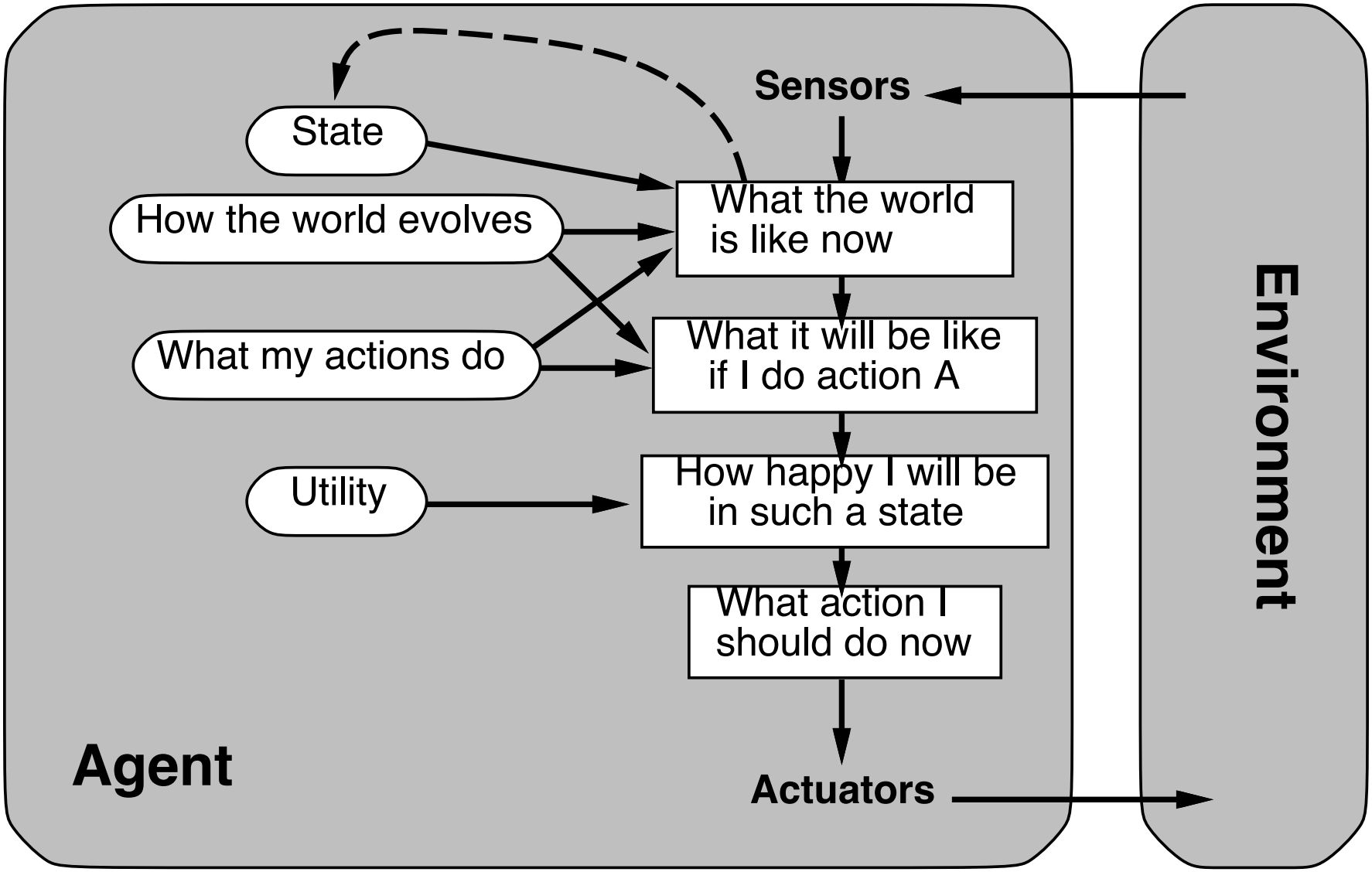
Universität Leipzig

Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI)

Motivation

- Bisher: Zielbasierte Agenten ($\rightarrow$ Suche, Planung), Agenten in nicht vollständig beobachtbaren Umgebungen ($\rightarrow$ Wahrscheinlichkeitsrechnung), lernende Agenten ($\rightarrow$ Machine Learning)
- Jetzt: Nutzenbasierte Agenten

Nutzenbasierte Agenten



Nutzenbasierte Agenten

Reward

- Am Anfang der Lehrveranstaltung nur sehr allgemeine Beschreibung von nutzenbasierten Agenten, jetzt mehr Details
- Ein nutzenbasierter Agent hat eine *Reward-Funktion* $R : (S \times A) \to \mathbb{R}$, die beschreibt, wie "gut" es für den Agenten ist, in Zustand s die Aktion a zu wählen
- Reward-Signal ist extern bestimmt: Der Agent kann seine eigene Reward-Funktion nicht umdefinieren
- z.B. für zielbasierte Agenten: Reward $+1$ bei Zielerreichung, 0 sonst.
- oder z.B. für Spiele: $+1$ wenn gewonnen, -1 wenn verloren, 0 sonst (wenn Spiel noch nicht vorbei)

Nutzenbasierte Agenten

Dynamik

- Annahme: Nächster Zustand der Welt s_{t+1} hängt nur von aktuellem Zustand der Welt s_t und Aktion des Agenten a_t ab
  - Kann als Wahrscheinlichkeitsverteilung $P(S_{t+1} \mid S_t, A_t)$ beschrieben werden
- Annahme: Der Agent kennt zu jedem Zeitpunkt den aktuellen Zustand
  - Es gibt auch Methoden um mit nicht-beobachtbarem Zustand umzugehen (Partially Observable Markov Decision Processes), aber die behandeln wir hier nicht (Bei Interesse: Vorlesung AI7 im Master)

Nutzenbasierte Agenten

- Die *Utility* des Agenten ist die Summe aller Rewards (erstmal, wir sehen gleich, dass wir die Definition etwas anpassen müssen)
- Der Agent versucht, Aktionen so zu wählen, dass der Erwartungswert der Utility maximiert wird
- Nennen eine Funktion $\pi : S \to A$ *Policy*
  - Beschreibt die Aktionsauswahl des Agenten
  - Optimale Policy π^*: Maximiert erwartete Utility

Nutzenbasierte Agenten

Discount Factor

- Oft möchte man eine Verhalten erreichen, bei dem der Agent aktuelle Rewards höher gewichtet als Rewards weit in der Zukunft (Warum ist das sinnvoll?)
- Führen *Discount Factor* $0 < \gamma \leq 1$ ein, und definieren Utility als:

$$U([(s_1, a_1), (s_1, a_1), \dots]) = R(s_1, a_1) + \gamma R(s_2, a_2) + \gamma^2 R(s_3, a_3) + \dots \quad (1)$$

- Auch hilfreich für Lösungsalgorithmen, da unendlich große Utility vermieden wird
- Im Folgenden: Alles noch mal formal...

Markov Decision Processes

An MDP is a 5-tuple (S, A, P, R, γ) with:

- set S of states.
- set A of actions.
- $P(S_{t+1} \mid S_t, A_t)$ specifies the dynamics.
- $R(S_t, A_t, S_{t+1})$ specifies the reward at time t.
  - $R(s, a, s')$ is the expected reward received when the agent is in state s, does action a and ends up in state s'.
  - Usually we use $R(s, a) = \sum_{s'} P(s' \mid s, a) R(s, a, s')$.
- γ is discount factor.

- I.e., an MDP is a DN with a certain internal structure

T Do we need to know $P(S_0)$ – the initial distribution?

Policies

- A *stationary policy* is a function:

$$\pi : S \to A$$

  Given a state s, $\pi(s)$ specifies what action the agent who is following π will do.
- An *optimal policy* is one with maximum expected discounted reward.
- For a fully-observable MDP with stationary dynamics and rewards with infinite or indefinite horizon, there is always an optimal stationary policy.

Example 1: Exercise or not to exercise?

Each week *Sam* has to decide whether to exercise or not:

- 2 States: $S = \{fit, unfit\}$
- 2 Actions: $A = \{exercise, relax\}$
- Dynamics $P(S_{t+1} \mid S_t, A_t)$:

S_t	A_t	$P(fit \mid State, Action)$
fit	exercise	0.99
fit	relax	0.7
unfit	exercise	0.2
unfit	relax	0.0

- Reward $R(S_t, A_t, S_{t+1})$ (here independent of S_{t+1}):

S_t	A_t	S_{t+1}	R
fit	exercise	*	8
fit	relax	*	10
unfit	exercise	*	0
unfit	relax	*	5

Example: to exercise or not?

Each week *Sam* has to decide whether to exercise or not:

- States: $\{fit, unfit\}$
- Actions: $\{exercise, relax\}$

T How many stationary policies are there?
Let s be the number of states, and a be the number of actions, then there are s^a possible policies.

T What are they?

S	π_1	π_2	π_3	π_4
f	e	e	r	r
u	e	r	e	r

Value of a Policy

Given a policy π:

- $V^\pi(s)$ is the expected discounted reward value of following policy π in state s.
- $Q^\pi(s, a)$ is the total expected discounted reward value of doing a in state s, then following policy π.
- V^π and Q^π can be defined mutually recursively:

$$V^\pi(s) = Q^\pi(s, \pi(s))$$

$$Q^\pi(s, a) = \sum_{s'} P(s' \mid a, s) \left(R(s, a, s') + \gamma \cdot V^\pi(s') \right)$$

Value of the Optimal Policy

Let π^* be the optimal policy.

- $V^*(s)$ is the expected discounted reward value of following policy π^* in state s.
- $Q^*(s, a)$ is the total expected discounted reward value of doing a in state s, then following policy π^*.
- V^* and Q^* can be defined mutually recursively:

$$V^*(s) = \max_a Q^*(s, a)$$

$$Q^*(s, a) = \sum_{s'} P(s' \mid a, s) \left(R(s, a, s') + \gamma \cdot V^*(s') \right)$$

$$\pi^*(s) = argmax_a Q^*(s, a)$$

Agenda

Markov Decision Processes

Solution Techniques

- Value Iteration
- Asynchronous Value Iteration
- Policy Iteration

Examples

Agenda

- Markov Decision Processes
- Solution Techniques
  - Value Iteration
  - Asynchronous Value Iteration
  - Policy Iteration
- Examples

Value Iteration

- Let V_k and Q_k be k-step lookahead value and Q functions.
- Idea: Given an estimate of the k-step lookahead value function, determine the $k+1$ step lookahead value function.
- Set V_0 arbitrarily.
- Compute Q_{i+1}, V_{i+1} from V_i.
- This converges exponentially fast (in k) to the optimal value function.

  The error reduces proportionally to $\frac{\gamma^k}{1-\gamma}$

Value Iteration

- Set $V^{(0)}$ arbitrarily.
- Compute $V^{(i+1)}$ from $V^{(i)}$:

$$V^{(i+1)}(s) = \sum_{s'} \left(P(s' \mid \pi^{(i)}(s), s) \left(R(s, \pi^{(i)}(s), s') + \gamma V^{(i)}(s') \right) \right)$$

or alternatively, if $R(s, a, s') = R(s, a)$

$$= R(s, \pi^{(i)}(s)) + \gamma \sum_{s'} P(s' \mid \pi^{(i)}(s), s) V^{(i)}(s')$$

with $\pi^{(i)}$ being the optimal policy wrt. $V^{(i)}$

- This iteration converges to the optimal value function V^*:

$$\lim_{i \to \infty} V^{(i)} = V^*$$

Example 1: Exercise or not to exercise?

Each week *Sam* has to decide whether to exercise or not:

- 2 States: $S = \{fit, unfit\}$
- 2 Actions: $A = \{exercise, relax\}$
- Dynamics $P(S_{t+1} \mid S_t, A_t)$:

S_t	A_t	$P(fit \mid State, Action)$
fit	exercise	0.99
fit	relax	0.7
unfit	exercise	0.2
unfit	relax	0.0

- Reward $R(S_t, A_t, S_{t+1})$ (here independent of S_{t+1}):

S_t	A_t	S_{t+1}	R
fit	exercise	*	8
fit	relax	*	10
unfit	exercise	*	0
unfit	relax	*	5

Example 1: Exercise or not to exercise? – It depends!

S_t	A_t	$P(fit \mid S_t, A_t)$	R
f	e	0.99	8
f	r	0.7	10
u	e	0.2	0
u	r	0.0	5

$$V^{\pi}(s) = Q^{\pi}(s, \pi(s))$$

$$Q^{\pi}(s, a) = \sum_{s'} P(s' \mid a, s) \left(R(s, a, s') + \gamma \cdot V^{\pi}(s') \right)$$

Iter.	V(fit)	V(unfit)
0	0.000	0.00000
1	10.000	5.00000
2	17.650	9.50000
3	23.812	13.55000
4	29.338	17.19500
5	34.295	20.47550
...	...	...
9	49.515	30.62898
10	52.394	32.56608
...	...	...
49	77.151	49.71368
50	77.189	49.74231

Example 1: Exercise or not to exercise? – It depends!

S_t	A_t	$P(fit \mid S_t, A_t)$	R
f	e	0.99	8
f	r	0.7	10
u	e	0.2	0
u	r	0.0	5

$$Q_i(s,a) = \sum_{s'} P(s' \mid a, s) \left(r(s,a,s') + \gamma \cdot V_i(s') \right)$$

$$\gamma = 0.9$$

$$\pi_i(s) = argmax_a Q_i(s,a)$$

$$V_{i+1}(s) = \max_a Q_i(s,a) = Q_i(s, \pi_i(s))$$

It.	$V(f)$	$V(u)$	$Q(f,e)$	$Q(f,r)$	$Q(u,e)$	$Q(u,r)$	$\pi(f)$	$\pi(u)$
0	0.00	0.00	8.00	10.00	0.00	5.00	r	r
1	10.00	5.00	16.95	17.65	5.40	9.50	r	r
2	17.65	9.50	23.81	23.68	10.01	13.55	e	r
3	23.81	13.55	29.33	28.66	14.04	17.19	e	r
4	29.33	17.19	34.29	33.12	17.66	20.47	e	r
5	34.29	20.47	38.74	37.13	20.91	23.42	e	r
				...				
9	49.51	30.62	52.39	49.46	30.96	32.56	e	r
10	52.39	32.56	54.97	51.80	32.87	34.30	e	r
				...				
49	77.15	49.71	77.18	72.02	49.68	49.74	e	r
50	77.18	49.74	77.22	72.06	49.70	49.76	e	r

Example 1: Exercise or not to exercise? – It depends!

- But the resulting policy does also depend on the discount factor γ for $i = 10$:

γ	$V^i(f)$	$V^i(u)$	$Q^i(f,e)$	$Q^i(f,r)$	$Q^i(u,e)$	$Q^i(u,r)$	$\pi^i(f)$	$\pi^i(u)$
0.2	12.0	6.2	10.4	12.0	1.4	6.2	r	r
0.5	17.6	9.9	16.8	17.6	5.7	9.9	r	r
0.9	52.3	32.5	54.9	51.8	32.8	34.3	e	r
0.95	64.7	40.1	69.3	64.5	42.8	43.1	e	r
0.99	77.4	48.1	84.3	77.9	53.4	52.7	e	e

T What does this mean?
Discuss the different results!

Zusammenfassung

- Ein MDP ist über das Transitionsmodell und die Reward-Funktion spezifiziert
- Die Utility eines MDP-Agenten ist der summierte discounted Reward
- Eine Lösung eines MDP ist eine Policy, die jedem Zustand eine Entscheidung zuweist. Die optimale Policy maximiert die erwartete Utility
- Die Utility eines Zustands ist die erwartete Utility der Zustandssequenz, wenn die optimale Policy von diesem Zustand ausgehend ausgeführt wird
- *Value Iteration* löst ein MDP, indem iterativ die Gleichungen gelöst werden, die die Utility eines Zustands mit der Utility der Nachbarn verbinden