1 Abstract

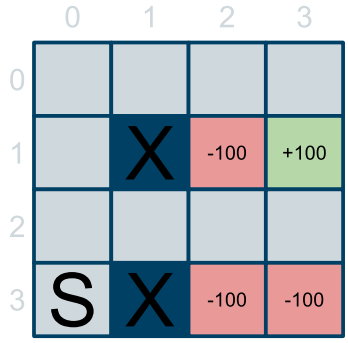
2 Markov Entscheidungs-Probleme

Markov Entscheidungs-Probleme (kurz **MDP** für *markov decision process*) sind ein Modell für Probleme, bei denen ein Agent versucht die größtmögliche Belohnung (*reward*) zu erzielen. Er bewegt sich dazu durch eine Menge von Zuständen (*states*) indem er aus einer Menge von Aktionen (*actions*) wählt. Welchen Zustand der Agent erreicht ist nicht deterministisch, die Wahrscheinlichkeiten sind jedoch nur von der gewählten Aktion und dem aktuellen Zustand abhängig. Die Belohnung, die der Agent für einen Übergang erhält, wird durch Ausgangs- und Endzustand des Übergangs so wie die gewählte Aktion bestimmt. Der Agent startet in einem Startzustand, es kann diverse Endzustände geben, nach deren Erreichen keine Aktionen mehr ausgeführt werden können.

Ziel ist es, eine Strategie (*policy*) zu finden, welche jedem Zustand die Aktion zuordnet, durch die der höchste Gesamtgewinn (also die höchste Summe über alle erzielten Gewinne) erwartet werden kann.

2.1 Ein Beispiel

In diesem Beispiel werden die Zustände durch die Felder eines 4×4-Grids visualisiert. Der Startzustand $(3, 0)$ ist durch ein großes S markiert, die Zustände mit einem X sind nicht erreichbar.



Die Menge der Aktionen besteht aus 4 Elementen, jedes symbolisiert einen Schritt in eine der vier Himmelsrichtungen. Beim Ausführen einer Aktion landet der Agent mit einer Chance von 0.8 ein Feld weiter in der gewählten Himmelsrichtung, so wie mit je einer Chance von 0.1 in einer der zwei orthogonalen Richtungen. Würde der Agent hierbei aif einem Feld landen, welches er nicht betreten kann (eines der mit X markierten, oder außerhalb des Grids), bleibt er auf seinem Feld.

Von den bunten Feldern aus führt jede Aktion mit einer Chance von 1 in den Endzustand. Der Agent erhält für diesen Übergang die auf dem Feld verzeichnete Belohnung (100 oder -100). Für jede andere Aktion wird eine Belohnung von -2 verbucht.

2.2 Notation

Zur Beschreibung eines MDPs werden einige Notationen benötigt, die hier angegeben werden. *(Zu einigen Notationen werden Beispiele zum oben genannten Beispiel angegeben.)* Zur Problemstellung selbst gehören:

- $S = \{s_1, \ldots, s_n\}$ ist die Menge der **Zustände**. Die Elemente der Menge können beliebig benannt werden.
- $A = \{a_1, \ldots, a_n\}$ ist die Menge der **Aktionen**. Auch diese Elemente können beliebig benannt werden.
- $T : S \times A \times S \to [0, 1]$ ist die **Übergangs-Funktion** (*transition function*). $T(s, a, s')$ beschreibt die Wahrscheinlichkeit, mit der der Agent von Zustand s in den Zustand s' wechselt, wenn er Aktion a ausführt. Eine allgemeinere, jedoch hier nicht verwendete Schreibweise wäre $T(s, a, s') = p(s'|s, a)$.

$T(s_{0,2}, a_E, s_{0,3}) = 0.8$	(Agent wählt Osten und kommt dort an)
$T(s_{0,2}, a_E, s_{0,2}) = 0.1$	(Agent wählt Osten, geht nach Norden, kein valider Zug)
$T(s_{0,2}, a_E, s_{1,2}) = 0.1$	(Agent wählt Osten, geht nach Süden)
$T(s_{0,2}, a_E, s_{0,1}) = 0.0$	(Agent wählt Osten, Bewegung nach Westen unmöglich)
$T(s_{1,2}, \_, s_{end}) = 1.0$	(jede Aktion von bunten Feldern führt zum Endzustand)

- $R : S \times A \times S \to \mathbb{R}$ ist die **Belohnungs-Funktion** (*reward function*). $R(s, a, s')$ ordnet dem Übergang von Zustand s nach Zustand s' mit Hilfe von Aktion a eine Belohung zu.

$R(s_{1,2}, \_, s_{end}) = -100$	(Belohnung von buntem Feld)
$R(s_{1,3}, \_, s_{end}) = +100$	(Belohnung von buntem Feld)
$R(s_{0,2}, a_N, s_{0,1}) = -2$	(jede Bewegung kostet 2 Belohung)

Zum Lösen der Probleme werden folgende Notationen verwendet:

- $\pi(s)$ ist eine **Policy**, die dem Zustand s die optimale Aktion zuordnet. Hierbei beschreibt π eine Policy im Allgemeinen, π^* die optimale Policy.
- $\gamma \in [0, 1]$ ist der sogenannte **Discount-Faktor**. Für den Agenten werden nach jeder Aktion alle noch erreichbaren Belohnungen mit γ multipliziert, damit er kürzere Wege, die zur selben Belohnung führen, bevorzugen wird. Mit $\gamma = 1$ ist für den Agenten ein langer Weg ebenso gut wie ein kurzer, solange er die selbe Belohnung erhält. Mit $\gamma = 0$ wird der Agent nicht mehr vorausschauend planen, da für ihn nur die nächste Belohnung Wert hat. Es gilt also, einen geeigneten Trade-Off zu finden.

2.3 V-Values

Um ein MDP zu lösen, lässt sich auf ein Bewertungsschema für Zustände zurückgreifen. Hierbei ergibt sich die Bewertung eines Zustandes aus der Gesamtbelohung, die der Agent zu erwarteten hat, wenn er eine Simulation in besagtem Zustand startet und von dort aus optimal handelt. Die Markov-Bedingung erlaubt uns in jedem Zustand anzunehmen, wir hätten die Simulation gerade erst gestartet, da es keine Faktoren gibt, die von bereits vergangenen Ereignissen abhängen.

Da die Bewertung jedes Zustandes angibt, welche Belohnung von ihm aus zu erwarten ist, lässt sich die Bewertung wie folgt rekursiv definieren:

$$V^*(s) = \max_{a \in A} \sum_{s' \in S} T\left(s, a, s'\right) \left[R\left(s, a, s'\right) + \gamma \cdot V^*\left(s'\right)\right]$$

In der eckigen Klammer befindet sich die unmittelbare Belohung, wenn ich von Zustand s über Aktion a in Zustand s' gelandet bin, addiert zur zu erwartenden Belohnung von Zustand s' aus (welche mit γ multipliziert wurde, da eine Aktion vorrüber ist).

2.4 Value Iteration

2.5 Policy Extraction

3 Reinforcement Learning

3.1 Verbindung zu MDPs

3.2 Verschiedene Ansätze

3.3 Q-Values/Q-Learning

3.4 Exploration

3.5 State Features

3.6 Deep Q-Learning

4 Fazit

5 Quellen