Optimierendes Lernen -
Adaptive Verfahren für dynamische Optimierungsprobleme

VE 9: Lernen einer Q-Funktion - *Q-learning*

Prof. Dr. Martin Riedmiller
Fachbereich Mathematik/ Informatik
Institute of Cognitive Science
Universität Osnabrück

Martin.Riedmiller@uos.de

VE 9: Lernen einer Q-Funktion - *Q-learning*

Ziele:

- Strategie-Verbesserung ohne Modell

Gliederung

- Idee der Q-Funktion
- Q-learning - Algorithmus

Ziel: Modellfreies Lernen

- Typische Lernsituation: Steure Prozess, für den kein explizites Modell vorhanden ist; nur Beobachtung der Simulation oder des realen Prozesses
- Value Iteration/ Policy Iteration benötigt Modell an zwei Stellen:
  - Schätzen des Kostenwerts:

$$J_{k+1}(i) = \min_{u \in U(i)} \sum_{j=0}^{n} p_{ij}(u)(c(i,u) + J_k(j))$$

  - Bestimmung der Strategie:

$$\pi(i) = \arg\min_{u \in U(i)} \sum_{j=0}^{n} p_{ij}(u)(c(i,u) + J_k(j))$$

Die Q-Funktion

Idee (Watkins, Diss, 1989:) In einem Zustand i wird für *jede Aktion* $u \in U(i)$ *explizit* die erwarteten Pfadkosten bei Anwendung einer bestimmten Strategie π berechnet. Dieser Wert wird als $Qmu(i,u)$ bezeichnet:

$$Q_\pi(i,u) := E[c(i,u) + J_\pi(f(i,u,w))]$$

bzw.

$$Q_\pi(i,u) = \sum_{j=0}^{n} p_{ij}(u)(c(i,u) + J_\pi(j))$$

$Q_\pi(i,u)$ sind die Kosten, die entstehen wenn mann in Zustand i Aktion u anwendet und sich von da an gemäss π verhält.

Bestimmung einer 'greedy' Strategie

Sei Q die zu J korrespondierende Q-Funktion

$$
\begin{aligned}
Q(i,u) &= E[c(i,u) + J(f(i,u,w)) \\
&= \sum_{j=0}^{n} p_{ij}(u)(c(i,u) + J(j))
\end{aligned}
\tag{1}
$$

und für die Strategie π gelte: $Q(i,\pi(i)) = \min_{u \in U} Q(i,u)$, dann ist π greedy bzgl. J (sieht man durch direktes Einsetzen).
Mit der Q-Funktion kann man eine greedy Strategie *ohne Modell* finden:

$$\pi(i) = \arg\min Q(i,u)$$

aber: Zur Bestimmung der Q-Funktion wird noch ein Modell gebraucht?!

⇒Lernen der Q-Funktion z.B. durch Monte-Carlo Methoden (s. VE 6)

Die optimale Q-funktion

Für die optimale Q-funktion Q^* muss die folgende Form der Bellman-Gleichung erfüllt sein:

$$
\begin{aligned}
Q^*(i,u) &= E[c(i,u) + J^*(f(i,u,w)) \\
&= \sum_{j=0}^{n} p_{ij}(u)(c(i,u) + J^*(j)) \\
&= \sum_{j=0}^{n} p_{ij}(u)(c(i,u) + \min_{u \in U(i)} Q^*(i,u))
\end{aligned}
$$

mit $J^*(i) = \min_{u \in U(i)} Q^*(i,u)$

Die Q-Funktion kann durch einen Iterations - Algorithmus (Value Iteration) bestimmt werden

Bestimmung der optimalen Q-Funktion

Ausgangspunkt: 'normales' Value Iteration:

$$Q_{k+1}(i,u) := E[c(i,u) + J_k(f(i,u,w))]$$

Idee: Bestimme den Erwartungswert durch Sampling (durchführen von Simulationen des stochastischen Übergangs (vgl. UE6).

1. Ansatz: Mittelung über N Samples von konkreten Übergängen:

$$Q_{k+1}(i,u) := \frac{1}{N}\sum_{t=1}^{N} c(i,u) + J_k(j)$$

mit j der sich bei Sample t einstellende Folgezustand und $J_k(j) = \min_{u \in U(i)} Q_k(j,u)$

2. Ansatz Q-LEARNING: Rekursive Anpassung mit stochastischer Approximation:

$$Q_{k+1}(i,u) := (1-\gamma)\, Q_k(i,u) + \gamma\, (c(i,u) + \min_{u' \in U(j)} Q_k(j,u'))$$

wobei der Folgezustand j und die Kosten $c(i,u)$ in der Simulation bestimmt werden.

- Die Schrittweite γ muss wieder so gewählt werden, dass sie den Bedingungen der stochastischen Approximation genügt, also z.B. wähle γ umgekehrt proportional zur Anzahl der Ausführungen von Zustands-Aktionspaar (i,u).
- *Q-learning* ist eine Variante des optimistischen $TD(0)$-Algorithmus für den modellfreien Fall (d.h. Simulationsbasiertes inkrementelles Ausführen der Value Iteration Vorschrift). Es existieren Varianten von Q-learning, die die Idee des $TD(\lambda)$-Verfahrens aufgreifen: $Q(\lambda)$.
- Weitere Variante: SARSA-Algorithmus

  $$Q_{k+1}(i,u) := (1-\gamma)\, Q_k(i,u) + \gamma\, (c(i,u) + Q_k(j,u'))$$

  mit u' die gemäss der aktuellen Strategie ausgewählte Aktion

Konvergenz Q-learning

- Voraussetzungen: Jedes Paar (i, u) wird unendlich oft besucht. γ ist vernünftig gewählt.
- entweder alle Strategien sind erfüllend
- oder es gibt eine erfüllende Strategie und alle nicht erfüllenden Strategien erzeugen unendliche Kosten
- Q-LEARNING funktioniert auch für diskontierte Probleme mit der offensichtlichen Anpassung:

$$Q(i,u) := (1-\gamma)\, Q(i,u) + \gamma\, (c(i,u) + \alpha \min_{u' \in U(j)} Q^*(j,u'))$$

Zusammenfassung

- Q-Funktion beschreibt Pfadkosten für Zustands-Aktions-paare
- Q-Learning: stochastische Approximation der optimalen Pfadkosten $Q^*(i, u)$ -
  dafür wird kein Modell benötigt
- optimale Strategie durch greedy-Auswertung der optimalen Q-Funktion

  $$\pi^*(i) = \arg\min_{u \in U} Q^*(i,u)$$

  dafür wird kein Modell benötigt
- Konvergenzvoraussetzung: Alle Zustands-Aktionspaare werden unendlich oft angepasst $\Rightarrow$praktisch: Exploration

Algorithmus

REPEAT
  starte in Zustand s_0; $t = 0$
  REPEAT {
    Wähle Aktion $u_t := \arg\min_{u \in U} Q_k(s_t, u)$
    oder u_t gemäss Explorationsstrategie beliebig
    Wende u_t auf System an: $s_{t+1} = f(s_t, u_t, w_t)$
    Lernschritt:
    $Q_{k+1}(s_t, u_t) := (1-\gamma) Q_k(s_t, u_t) + \gamma(c(s_t, u_t) + J_k(s_{t+1}))$
    mit $J_k(s_{t+1}) := \min_{u \in U} Q_k(s_{t+1}, u)$
  UNTIL Terminalzustand erreicht
UNTIL Strategie optimal