Vorlesung
Grundlagen adaptiver Wissenssysteme

Prof. Dr. Thomas Gabel
Frankfurt University of Applied Sciences
Faculty of Computer Science and Engineering
tgabel@fb2.fra-uas.de

Vorlesungseinheit 6

Das Strategieiterationsverfahren

Das Strategieiterationsverfahren

Lernziele

- Wie wird eine Strategie bewertet.
- Wie wird eine Strategie gierig ausgewertet.
- Kennenlernen des Strategieiterationsverfahren: Policy Iteration

Das Strategieiterationsverfahren

Überblick

1. Motivation

Das Strategieiterationsverfahren

Überblick

1. Motivation

2. Strategiebewertung

Das Strategieiterationsverfahren

Überblick

1. Motivation

2. Strategiebewertung

3. Strategieverbesserung

Das Strategieiterationsverfahren

Überblick

1. Motivation
2. Strategiebewertung
3. Strategieverbesserung
4. Strategieiterationsverfahren

Das Strategieiterationsverfahren

Überblick

1. Motivation
2. Strategiebewertung
3. Strategieverbesserung
4. Strategieiterationsverfahren
5. Zusammenfassung dynamisches Programmieren

Das Strategieiterationsverfahren

Überblick

1. Motivation
2. Strategiebewertung
3. Strategieverbesserung
4. Strategieiterationsverfahren
5. Zusammenfassung dynamisches Programmieren

Strategieiteration (Policy Iteration)

Motivation

- Value Iteration benötigt im Allgemeinen unendlich viele Iterationen
- Aber: Es gibt nur endlich viele Policies!
  $\Rightarrow$ Durchführung einer Iteration über die Menge der möglichen Strategien

Policy Iteration: überblick

Kernidee:

- Starte mit einer beliebigen erfüllenden (proper) Strategie π_0.
- Ermittele die dazugehörige Bewertungsfunktion V_{π_0}
  - Strategiebewertung (Policy Evaluation)

Policy Iteration: überblick

Kernidee:

- Starte mit einer beliebigen erfüllenden (proper) Strategie π_0.
- Ermittele die dazugehörige Bewertungsfunktion V_{π_0}
  - Strategiebewertung (Policy Evaluation)
- Werte diese Bewertungsfunktion “gierig” (greedy) aus.
  $\Rightarrow$ Das ist die neue Strategie.
  - Strategieverbesserung (Policy Improvement)
- Wiederhole dies, bis die optimale Strategie gefunden ist.

Das Strategieiterationsverfahren

Überblick

Strategiebewertung (1)

Policy Evaluation

Kernidee:

- Problemstellung: bewerte eine gegebene Strategie π
- Lösung: iterative Anwendung der Bellman-Gleichung
- eine Folge von Wertfunktionen $V_0 \to V_1 \to V_2 \to \cdots \to V^\pi$ ergibt sich

Strategiebewertung (1)

Policy Evaluation

Kernidee:

- Problemstellung: bewerte eine gegebene Strategie π
- Lösung: iterative Anwendung der Bellman-Gleichung
- eine Folge von Wertfunktionen $V_0 \rightarrow V_1 \rightarrow V_2 \rightarrow \cdots \rightarrow V^\pi$ ergibt sich
- in jedem Schritt k
  - synchrone Aktualisierungen der Werte $V_k(i)$
  - für alle Zustände i
  - berechne $V_{k+1}(i)$ aus $V_k(j)$
  - wobei j für Nachfolgezustände von i stehen
- aus der Literatur bekannt: Konvergenz gegen V^π tritt ein

Strategiebewertung (2)

Policy Evaluation

Algorithmus: Iterative Strategiebewertung

- Eingabe: eine zu bewertende Strategie π
- Wähle V_0 beliebig.
- Setze Zähler $k = 0$.

Strategiebewertung (2)

Policy Evaluation

Algorithmus: Iterative Strategiebewertung

- Eingabe: eine zu bewertende Strategie π
- Wähle V_0 beliebig.
- Setze Zähler $k = 0$.
- REPEAT

  $k := k + 1$

  FORALL $i \in S$

Strategiebewertung (2)

Policy Evaluation

Algorithmus: Iterative Strategiebewertung

- Eingabe: eine zu bewertende Strategie π
- Wähle V_0 beliebig.
- Setze Zähler $k = 0$.
- REPEAT
  $k := k + 1$
  FORALL $i \in S$
  aktualisiere $V_k(i)$ auf Basis von $V_{k-1}(j)$ via

$$V_k(i) = \sum_{j=0}^{n} p_{ij}(\pi(i))\,(c(i, \pi(i)) + \gamma V_{k-1}(j))$$

UNTIL Konvergenz

Das Strategieiterationsverfahren

Überblick

1. Motivation
2. Strategiebewertung
3. Strategieverbesserung
4. Strategieiterationsverfahren
5. Zusammenfassung dynamisches Programmieren

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Strategieverbesserung (1)

Policy Improvement

Kernidee:

- Eine Strategie π ist gegeben, eine zugehörige Wertfunktionen V^π kann mit Strategiebewertung ermittelt werden.
- Gewinne nun aus π und V^π eine neue Strategie π'.
  - ... die natürlich besser als π sein soll!
- Vorgehensweise: Sogenannte "gierige Auswertung".

Strategieverbesserung (2)

Policy Improvement

Satz: Strategieverbesserung mittels gieriger Auswertung

Gegeben sei eine Strategie $\pi : S \to A$ und die zugehörige Wertfunktion $V^\pi : S \to \mathbb{R}$. Dann lässt sich eine verbesserte "gierige" Strategie π' wie folgt definieren:

$$\begin{aligned} \pi' &: S \to A \\ \text{i} &\mapsto \arg\min_{a \in A} \sum_{j=1}^{n} p_{ij}(a)\,(c(i,a) + \gamma V^\pi(j)) \end{aligned}$$

Strategieverbesserung (2)

Policy Improvement

Satz: Strategieverbesserung mittels gieriger Auswertung

Gegeben sei eine Strategie $\pi : S \to A$ und die zugehörige Wertfunktion $V^\pi : S \to \mathbb{R}$. Dann lässt sich eine verbesserte "gierige" Strategie π' wie folgt definieren:

$$\begin{aligned} \pi' &: S \to A \\ &\ \ \mathrm{i} \mapsto \arg\min_{a \in A} \textstyle\sum_{j=1}^{n} p_{ij}(a)\left(c(i,a) + \gamma V^\pi(j)\right) \end{aligned}$$

Bemerkung:

- π' verbessert die Werte für jeden Zustand, denn für alle i gilt

$$V^\pi(i) \geq V^{\pi'}(i) = \min_{a \in A} \sum_{j=1}^{n} p_{ij}(a)\left(c(i,a) + \gamma V^\pi(j)\right)$$

- π' stellt also eine Verbesserung gegenüber π dar.

Beispiel (1)

4x4-Gitterwelt

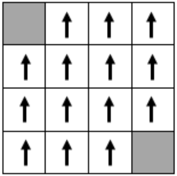
- diskontierter ($\gamma = 0.5$) MDP mit 16 Zuständen
- deterministische Zuständsübergänge
- zwei Terminalzustände ($i = 0$ und $i = 15$), in denen keine weiteren Kosten anfallen (und die nicht mehr verlassen werden)
- 4 Aktionen: $A = \{hoch, runter, links, rechts\}$
- Kosten für jede Aktion (jeden Schritt): 1
  - Aktionen, die den Agenten gegen die Wand bewegen lassen, verändern seinen Zustand nicht

0	1	2	3
4	5	6	7
8	9	10	11
12	13	15	16

Aktionen

Beispiel (2)

Strategie „immer hoch"



V_k für Strategie „immer hoch"

γ=0.5

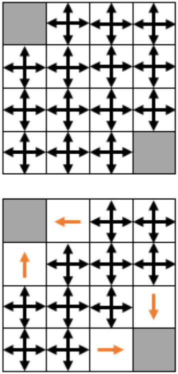
k=0

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

k=1

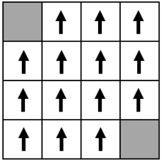
0	1	1	1
1	1	1	1
1	1	1	1
1	1	1	0

Gierige Strategie bezogen auf V_k



Beispiel (3)

Strategie „immer hoch“



V_k für Strategie „immer hoch“

$\gamma=0.5$

k=2

0	1.5	1.5	1.5
1	1.5	1.5	1.5
1.5	1.5	1.5	1.5
1.5	1.5	1.5	0

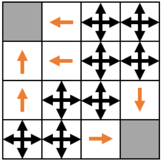
k=3

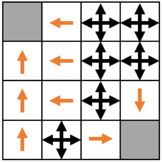
0	1.75	1.75	1.75
1	1.75	1.75	1.75
1.5	1.75	1.75	1.75
1.75	1.75	1.75	0

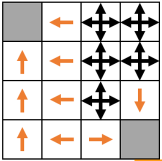
k=10

0	1.998	1.998	1.998
1	1.998	1.998	1.998
1.5	1.998	1.998	1.998
1.75	1.998	1.998	0

Gierige Strategie bezogen auf V_k







Das Strategieiterationsverfahren

Überblick

Strategieiteration (1)

Policy Iteration

Kernidee:

- Strategieiteration = Wiederholung von (Strategiebewertung + Strategieverbesserung)
- Policy Iteration = Repeated (Policy Evaluation + Policy Improvement)

Strategieiteration (1)

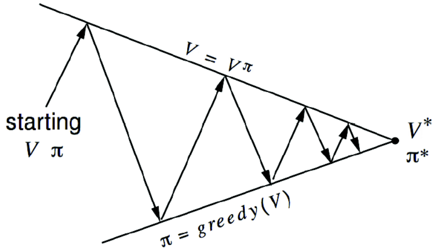
Policy Iteration

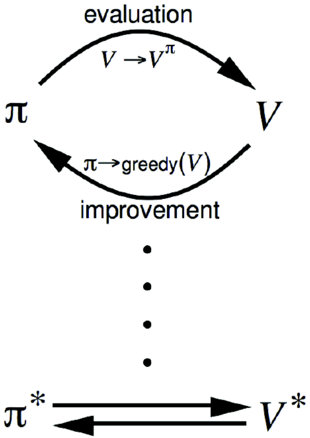
Kernidee:

- Strategieiteration = Wiederholung von (Strategiebewertung + Strategieverbesserung)
- Policy Iteration = Repeated (Policy Evaluation + Policy Improvement)
- Wenn sich keine Änderung beim Schritt von π zu π' mehr ergeben, so ist das Verfahren beendet.
- In diesem Fall gilt dann
  - $V^{\pi}(i) = V^{*}(i)$ für alle $i \in S$.
  - Es wurde also die optimale Strategie ermittelt.

Strategieiteration (Policy Iteration) (2)

Visualisierung:





- Policy Evaluation: Ermittele V^π
  - mittels iterativer Strategiebewertung
- Policy Improvement: Erzeuge π' (mit π' “besser als” π)
  - mittels “gieriger” Strategieverbesserung

Strategieiteration (3)

Policy Iteration

Strategieiteration (Policy Iteration)

- Gegeben ist initiale Strategie $\pi = \pi_0$.
- Setze Zähler $z = 0$.
- REPEAT

  berechne V^{π_z} mittels iterativer Strategiebewertung

Strategieiteration (3)

Policy Iteration

Strategieiteration (Policy Iteration)

- Gegeben ist initiale Strategie $\pi = \pi_0$.
- Setze Zähler $z = 0$.
- REPEAT
  berechne V^{π_z} mittels iterativer Strategiebewertung
  ermittele π_{z+1} aus V^{π_z} und π_z
  mittels gieriger Strategieverbesserung
  $z := z + 1$
  UNTIL $\pi_z = \pi_{z-1}$

Konvergenzaussagen

Satz: Konvergenz des Strategieiterationsverfahren

Wenn wir mit einer erfüllenden Strategie π_0 starten, generiert Policy Iteration eine Sequenz von sich monoton verbessernden, erfüllenden Strategien, d.h. $V_{\pi_{z+1}}(i) \leq V_{\pi_z}(i)$, und terminiert schließlich mit einer optimalen Strategie.

Bemerkungen

- Die Strategie verbessert sich in jedem Schritt.
- Wenn sie sich nicht mehr verbessert ($\pi_z = \pi_{z-1}$), ist die optimale Strategie π^* erreicht.
- Für die praktische Anwendung sind die folgenden Überlegungen relevant:

Bemerkungen

- Die Strategie verbessert sich in jedem Schritt.
- Wenn sie sich nicht mehr verbessert ($\pi_z = \pi_{z-1}$), ist die optimale Strategie π^* erreicht.
- Für die praktische Anwendung sind die folgenden Überlegungen relevant:
  - Die Wertfunktion V^{π_z} wird durch iterative Strategiebewertung gefunden. Dies kann unendlich viele Iterationen erfordern.
  - In praktischen Anwendungen ergibt sich die Frage, wie groß man den Zähler k (also die Anzahl der Iterationen im Rahmen der Strategiebewertung) werden lässt.
  - $\Rightarrow$ Rechenzeit vs. Genauigkeit!
  - MODIFZIERTES POLICY ITERATION: Policy Iteration mit begrenzter Anzahl von Evaluierungsschritten (also nicht $k \to \infty$).

Das Strategieiterationsverfahren

Überblick

1. Motivation
2. Strategiebewertung
3. Strategieverbesserung
4. Strategieiterationsverfahren
5. Zusammenfassung dynamisches Programmieren

Zusammenfassung

Problemlösen mit Methoden des dynamischen Programmierens

Mit 3 Schritten zum Ziel:

1. Entscheidungsproblem
2. Formulierung als MDP
   - Festlegung der Zustände / Aktionen
     - Wahl des Abstraktionsniveaus
   - Festlegung des Problemtyps
     - endlicher Horizont / unendlicher Horizont
     - SKP, Diskontierung
   - Festlegung der Kostenfunktion
3. Lösen des MDPs

Klassische DP-Verfahren

Grundprinzip

- Suche (bestimme) optimale Pfadkosten $V^*(i)$.
- Durch die optimalen Pfadkosten ist auch optimale Strategie definiert.
- Für $V^*(i)$ muss das Optimalitätsprinzip erfüllt sein:

$$V^*(i) = \min_{a \in A(i)} \sum_{j=1}^{n} p_{ij}(a)\,(c(i,a) + \gamma V^*(j)) \qquad i = 1 \ldots n$$

Verfahren zur Berechnung von V^*

- Wertiterationsverfahren (Value Iteration)

$$V_k^*(i) = \min_{a \in A(i)} \sum_{j=1}^{n} p_{ij}(a)\,(c(i,a) + \gamma V^*(j)) \qquad i = 1 \dots n$$

- Strategieiterationsverfahren (Policy Iteration)
  - Wähle π_z
  - Berechne V^{π_z} (lineares Gleichungssystem oder iterativ)
  - Wähle π_{z+1} “gierig” bezüglich V^{π_z}

Klassische DP-Verfahren

Voraussetzungen:

- Übergangswahrscheinlichkeiten (*Modell*) bekannt
- endlich viele Zustände (Iteration)
- endliche viele Aktionen (Minimumsuche)

Ergebnis:

- Verfahren findet $V^*(\cdot)$ und damit optimale Strategie für alle Zustände