

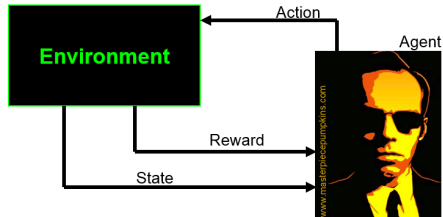
Vorlesung

Grundlagen adaptiver Wissenssysteme

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

Vorlesungseinheit 2

Optimierendes Lernen und dynamisches Programmieren



Ziel der Vorlesung

Einführung des Lernproblem-Typs

Optimierendes Lernen – Reinforcement Learning

Einführung in die mathematischen Grundlagen

für ein selbständig lernendes System

Optimierendes Lernen und dynamisches Programmieren

Überblick

Lernziele

Motivation, Definition und Abgrenzung

1. Beispiele

Optimierendes Lernen und dynamisches Programmieren

Überblick

Lernziele

Motivation, Definition und Abgrenzung

1. Beispiele
2. Lösungsansätze der künstlichen Intelligenz

Optimierendes Lernen und dynamisches Programmieren

Überblick

Lernziele

Motivation, Definition und Abgrenzung

1. Beispiele
2. Lösungsansätze der künstlichen Intelligenz
3. Maschinelles Lernen

Optimierendes Lernen und dynamisches Programmieren

Überblick

Lernziele

Motivation, Definition und Abgrenzung

1. Beispiele
2. Lösungsansätze der künstlichen Intelligenz
3. Maschinelles Lernen
4. Optimierendes Lernen

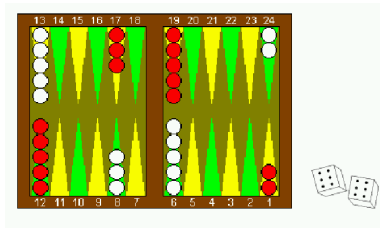
Optimierendes Lernen und dynamisches Programmieren

Überblick

1. Beispiele
2. Lösungsansätze der künstlichen Intelligenz
3. Maschinelles Lernen
4. Optimierendes Lernen

Beispiel: Backgammon

Kann ein Programm selbständig **Backgammon** lernen?

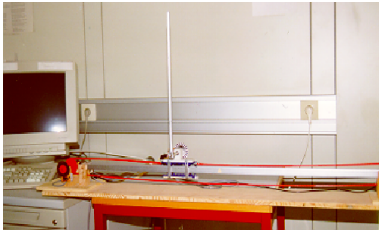


Lernen aus Erfolg (Sieg) und
Misserfolg (Niederlage)

Neuro-Backgammon:
Weltmeisterniveau [Tesauro,
1992]

Beispiel: Stabbalancierer (Regelungstechnik)

Kann ein Programm selbständig **balancieren** lernen?



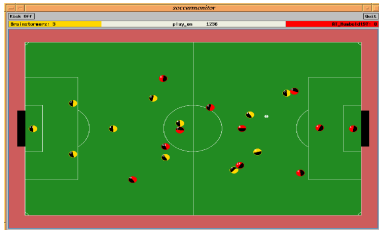
Lernen aus Erfolg und Misserfolg

Neuronale RL-Regler:

Störungen, Ungenauigkeiten,
unbekanntes Systemverhalten,
Nichtlinearitäten ,... [Riedmiller
et.al.]

Beispiel: Roboterfußball

Können Programme selbständig lernen zu kooperieren?



Lernen aus Erfolg und Misserfolg

Kooperative RL-Agenten:
Komplexität, verteilte Intelligenz,
... [Gabel et.al.]

Beispiel: Autonome (z.B. humanoide) Roboter

Aufgabe: Bewegungssteuerung ähnlich wie beim Menschen (Gehen, Laufen, Fussballspielen, Radfahren, Skifahren, ...)

Eingabe: Bild der Videokamera

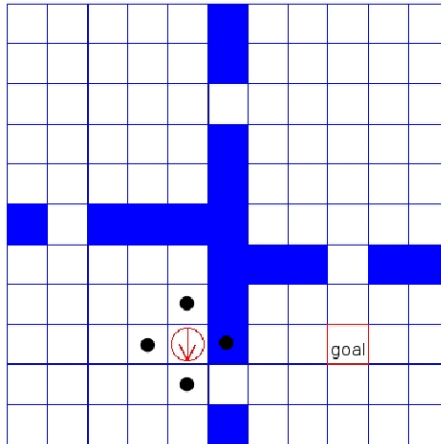
Ausgabe: Steuersignale an die Gelenke

Probleme:

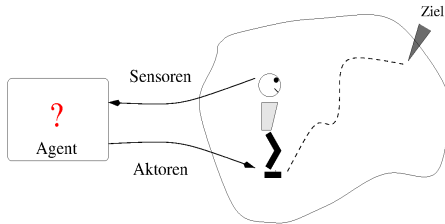
- sehr komplex
- Konsequenzen von Aktionen schwer einschätzbar
- Störungen / Rauschen



Beispiel: Labyrinth (engl. Maze)



Das "Agenten-Konzept"



[Russell and Norvig, AIAMA, page 33] "An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors."

konzeptionelle Vereinheitlichung und Vereinfachung

→ Der Begriff des **Agenten** dient als Abstraktion für das lernfähige System und wird daher im Rahmen dieser Vorlesung regelmäßig benutzt.

Optimierendes Lernen und dynamisches Programmieren

Überblick

1. Beispiele
2. Lösungsansätze der künstlichen Intelligenz
3. Maschinelles Lernen
4. Optimierendes Lernen

Teilgebiete der "Künstlichen Intelligenz" (KI)

Das Forschungsgebiet der KI umfasst eine größere Anzahl von Teilgebieten, die unterschiedliche (Teil-)Probleme intelligenten Verhaltens adressieren und dafür entsprechend unterschiedliche Lösungsansätze anbieten. Beispiele:

Teilgebiete der "Künstlichen Intelligenz" (KI)

Das Forschungsgebiet der KI umfasst eine größere Anzahl von Teilgebieten, die unterschiedliche (Teil-)Probleme intelligenten Verhaltens adressieren und dafür entsprechend unterschiedliche Lösungsansätze anbieten. Beispiele:

- Planen / Suchen (z.B. A^* , Backtracking)
- Deduktion (z.B. Logische Programmiersprachen, Prädikatenlogik)
- Expertensysteme (z.B. Regelwissen generiert von Experten)
- Fuzzy-Regelsysteme (Unscharfes Schliessen)
- Genetische Algorithmen (Evolvieren von Lösungen)
- Maschinelles Lernen (z.B. Reinforcement Lernen)

→ mehr dazu in der VL "Artificial Intelligence"

Optimierendes Lernen und dynamisches Programmieren

Überblick

1. Beispiele
2. Lösungsansätze der künstlichen Intelligenz
- 3. Maschinelles Lernen**
4. Optimierendes Lernen

Typen des Lernens (bei Menschen)

- Lernen von einem Lehrer
- Strukturieren von Objekten
- Lernen aus Erfahrung

Wiederholung: Typen des Maschinellen Lernens (ML)

- **Lernen mit Lehrer:** Überwachtes Lernen (Supervised Learning)
 - Beispiele von Eingabe / (Ziel-)Ausgabe
 - Zielsetzung: Generalisierung (i.A. nicht nur Auswendiglernen)

Wiederholung: Typen des Maschinellen Lernens (ML)

- **Lernen mit Lehrer:** Überwachtes Lernen (Supervised Learning)
 - Beispiele von Eingabe / (Ziel-)Ausgabe
 - Zielsetzung: Generalisierung (i.A. nicht nur Auswendiglernen)
- **Strukturierung / Erfassung von Zusammenhängen:**
Unüberwachtes Lernen (Unsupervised learning, Clustering)
 - Zielsetzung: Zusammenfassen von ähnlichen Datenpunkten', z.B. zur Vorverarbeitung.

Wiederholung: Typen des Maschinellen Lernens (ML)

- **Lernen mit Lehrer:** Überwachtes Lernen (Supervised Learning)
 - Beispiele von Eingabe / (Ziel-)Ausgabe
 - Zielsetzung: Generalisierung (i.A. nicht nur Auswendiglernen)
- **Strukturierung / Erfassung von Zusammenhängen:** Unüberwachtes Lernen (Unsupervised learning, Clustering)
 - Zielsetzung: Zusammenfassen von ähnlichen Datenpunkten', z.B. zur Vorverarbeitung.
- **Lernen durch Belohnen / Bestrafen:** Optimierendes Lernen (Reinforcement Learning)
 - Lernvorgabe: Spezifikation des zu erreichenden Ziels (bzw. zu vermeidender Ereignisse).

Maschinelles Lernen: Vorgehensweise

Wie gehe ich vor, wenn ich eine konkrete Aufgabenstellung mit Methoden des maschinellen Lernens lösen soll?

Maschinelles Lernen: Vorgehensweise

Wie gehe ich vor, wenn ich eine konkrete Aufgabenstellung mit Methoden des maschinellen Lernens lösen soll?

1. Typ des Lernproblems identifizieren
 - Was ist gegeben, was ist gesucht?
2. Repräsentation des gelernten Lösungswissens festlegen
 - Tabelle, Regeln, lineare Abbildung, neuronales Netz, ...
3. Lösungsverfahren auswählen
 - beobachtete Daten \mapsto Problemlösung
 - z.B. (heuristische) Suche, Gradientenabstieg, Optimierungsverfahren, ...

Maschinelles Lernen: Vorgehensweise

Wie gehe ich vor, wenn ich eine konkrete Aufgabenstellung mit Methoden des maschinellen Lernens lösen soll?

1. Typ des Lernproblems identifizieren
 - Was ist gegeben, was ist gesucht?
2. Repräsentation des gelernten Lösungswissens festlegen
 - Tabelle, Regeln, lineare Abbildung, neuronales Netz, ...
3. Lösungsverfahren auswählen
 - beobachtete Daten \mapsto Problemlösung
 - z.B. (heuristische) Suche, Gradientenabstieg, Optimierungsverfahren, ...

Also nicht: "Für dieses Problem brauche ich ein neuronales Netz."

Optimierendes Lernen und dynamisches Programmieren

Überblick

1. Beispiele
2. Lösungsansätze der künstlichen Intelligenz
3. Maschinelles Lernen
4. Optimierendes Lernen

Optimierendes Lernen

Schwerpunkt der nächsten Wochen

Reinforcement Learning:

- Terminologie: auf deutsch zumeist optimierendes Lernen, manchmal auch verstärkendes Lernen, bestärkendes Lernen oder autonomes Lernen
- es wird keine Information über eine Lösungsstrategie vorausgesetzt

Optimierendes Lernen

Schwerpunkt der nächsten Wochen

Reinforcement Learning:

- Terminologie: auf deutsch zumeist optimierendes Lernen, manchmal auch verstärkendes Lernen, bestärkendes Lernen oder autonomes Lernen
- es wird keine Information über eine Lösungsstrategie vorausgesetzt
- selbständiges Erlernen einer Lösungsstrategie durch geschicktes Ausprobieren von Lösungen ("Trial and Error")
- größte Herausforderung an ein lernendes System
- Repräsentation des Lösungswissens mithilfe eines Funktionsapproximators (z.B. Tabellen, lineare Modelle, neuronale Netze usw.)

RL am Beispiel autonomer Roboter



schlecht: Beschädigung
(z.B. durch Sturz o.ä.)

gut: Aufgabe erfolgreich erledigt

besser: Aufgabe schnell / mit
Einsatz von wenig Energie / unter
Verwendung gleichförmiger
Bewegungen (o.ä.) erledigt
⇒ Optimierung!

Optimierendes Lernen – Reinforcement Learning (RL)

Diverse alternative Bezeichnungen sind geläufig:

- Lernen aus Bewertungen, ver-/bestärkendes Lernen
- selbständiges / autonomes Lernen, (selbst-)adaptives Lernen
- Neuro Dynamic Programming

Optimierendes Lernen – Reinforcement Learning (RL)

Diverse alternative Bezeichnungen sind geläufig:

- Lernen aus Bewertungen, ver-/bestärkendes Lernen
- selbständiges / autonomes Lernen, (selbst-)adaptives Lernen
- Neuro Dynamic Programming

Kerneigenschaften:

- Definiert einen **Lerntypus** und nicht eine Methode!
 - zentrales Merkmal: bewertendes Trainingssignal
 - z.B. “gut” / “schlecht”
- RL mit unmittelbarer Bewertung: Bandit-Problem / Automatentheorie
 - Entscheidung \mapsto Bewertung
 - z.B. Parameter für einen Wurf in einen Basketballkorb
- zeitliche Verzögerung (delayed RL)
 - Entscheidung, Entscheidung, ..., Entscheidung
→ Bewertung
 - wesentlich schwerer; interessant, da vielseitiger einsetzbar

Bewertungen (1)

in Form von Belohnungen oder Kosten

- Eine Belohnung r_t ist ein skalaras Bewertungssignal.
- zeigt an, wie gut/schlecht der Agent sich im Zeitschritt t verhalten hat
- Ziel des Agenten ist es, über die Zeit so viele positive Bewertungen wie möglich aufzusammeln.

Bewertungen (1)

in Form von Belohnungen oder Kosten

- Eine Belohnung r_t ist ein skalarcs Bewertungssignal.
- zeigt an, wie gut/schlecht der Agent sich im Zeitschritt t verhalten hat
- Ziel des Agenten ist es, über die Zeit so viele positive Bewertungen wie möglich aufzusammeln.

Belohnungshypothese

Alle Ziele lassen sich beschreiben durch die Maximierung der erwarteten kumulierten Belohnungen.

- Alle Methoden des optimierenden Lernens basieren auf dieser Hypothese.

Bewertungen (2)

in Form von Belohnungen oder Kosten

Terminologie

- In der Literatur wird wechselseitig von Belohnungen, Bestrafungen bzw. Kosten (im Sinne negativer Belohnungen) sowie von Bewertungen (im neutralen Sinne) gesprochen.
- Alle drei Sprechweisen sind äquivalent.

Zeitlich verzögertes RL

Bewertungen (Belohnungen / Bestrafungen) werden dem Agenten erst mit zeitlicher Verzögerung mitgeteilt.

- Entscheidung, Entscheidung, . . . , Entscheidung → Bewertung
- Beispiel: Robotik, Regelungstechnik, Spiele (Schach, Backgammon)

Zeitlich verzögertes RL

Bewertungen (Belohnungen / Bestrafungen) werden dem Agenten erst mit zeitlicher Verzögerung mitgeteilt.

- Entscheidung, Entscheidung, . . . , Entscheidung → Bewertung
- Beispiel: Robotik, Regelungstechnik, Spiele (Schach, Backgammon)
- Grundproblem: **Temporal Credit Assignment**
 - Welche der einzelnen Entscheidungen trug wie stark dazu bei, die Belohnung zu erhalten.
- Basisarchitektur: Actor-Critic-System
 - zwei Hauptkomponenten: eine Aktionsauswahlkomponente (Actor) und eine Bewertungskomponente (Critic) sind im lernenden Agenten verankert

Mehrstufige Entscheidungsprobleme (1)

Mehrstufige (=sequentielle) Entscheidungsprobleme

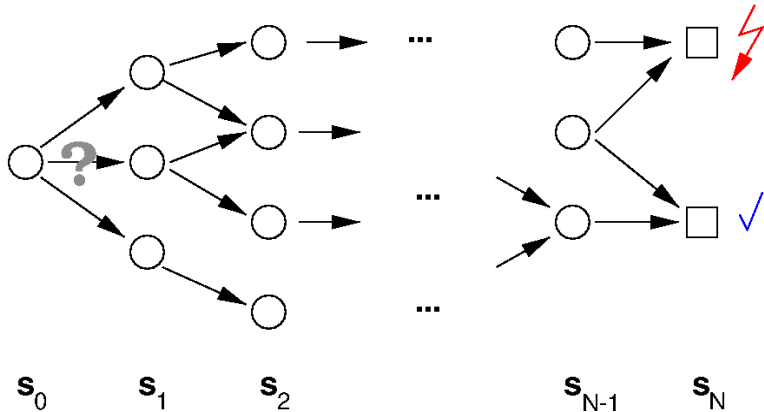
- Ziel: Aktionen auswählen, die die aufsummierten zukünftigen Belohnungen maximieren
- jede Aktion kann Konsequenzen erst in der entfernten Zukunft entfalten
- möglicherweise ist es besser, die unmittelbare Belohnung zugunsten einer (höheren) späteren Belohnung aufzugeben
- Können Sie Beispiele nennen?

Mehrstufige Entscheidungsprobleme (1)

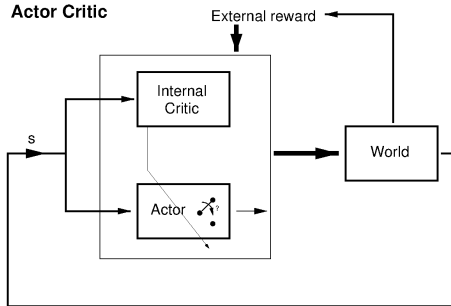
Mehrstufige (=sequentielle) Entscheidungsprobleme

- Ziel: Aktionen auswählen, die die aufsummierten zukünftigen Belohnungen maximieren
- jede Aktion kann Konsequenzen erst in der entfernten Zukunft entfalten
- möglicherweise ist es besser, die unmittelbare Belohnung zugunsten einer (höheren) späteren Belohnung aufzugeben
- Können Sie Beispiele nennen?
 - finanzielle Investition (zahlt sich evtl. erst in Monaten aus)
 - Tank eines Helikopters befüllen (kann einen Absturz in einigen Stunden verhindern)
 - gegnerische Züge blockieren (kann Gewinnchancen in der Zukunft erhöhen)

Mehrstufige Entscheidungsprobleme (2)



Actor-Critic System [Barto, Sutton, 1983]



Actor: Aktionsauswahlkomponente

- in Situation s wähle Aktion a (aka Strategie $\pi : S \rightarrow A$)

Critic: Bewertungskomponente

- “Verteilung” des externen (Gesamt-)Bewertungssignals auf einzelne Aktionen
- Fähigkeit zur Beurteilung, wie gut welche Aktion in welcher

Historie (1)

- 1959 Samuel's Checker-Player (Dame): TD-Verfahren
- 1968 Michie und Chambers: Boxes
- 1983 Barto, Sutton's AHC/ACE, 1987 Sutton's TD(λ)
- Anfang der 90-er: Zusammenhang von **Dynamischen Programmieren (DP)** und Reinforcement Learning
 - Werbos, Sutton, Barto, Watkins, Singh, Bertsekas

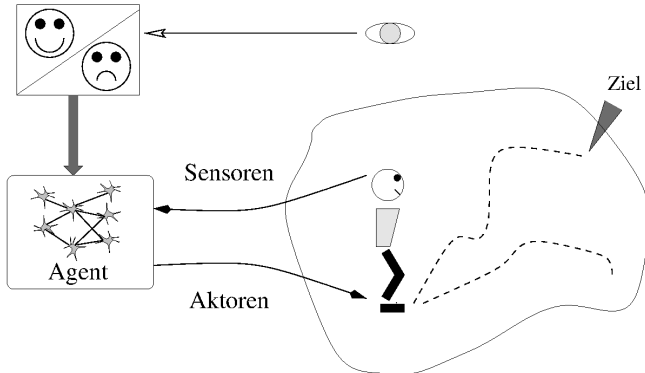
Historie (2)

- dynamisches Programmieren (DP, Dynamic Programming)
 - geht zurück auf Bellman (späte 1950er)
 - klassisches Optimierungsverfahren
 - zu hoher Aufwand bei größeren Aufgabenstellungen
 - Vorteil: saubere mathematische Formulierung, Konvergenzaussagen
- 2000 Policy-Gradient-Verfahren (Sutton, Peters, ...)
- 2005 Fitted Q (Batch DP Verfahren) (Ernst, Riedmiller, ...)
- 2010 Deep Reinforcement Learning (Lange, Mnih, ...)
- seither viele Beispiele von erfolgreichen zumindest praxisnahen Anwendungen

Weitere ausgewählte Beispiele

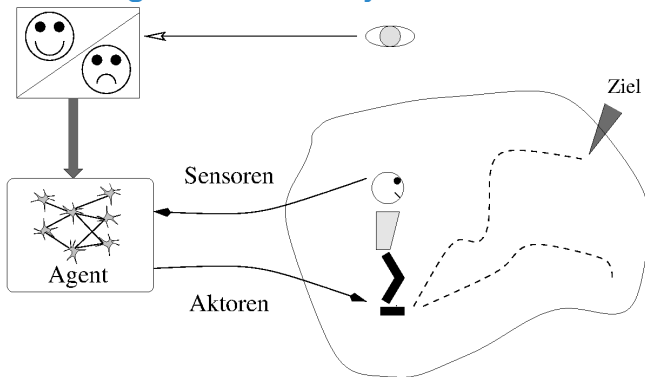
Gebiet	Eingabe Ausgabe (Aktionen)	Ziel	Beispiel
Spiele	Brettsituation gültiger Zug	Gewinnen	Backgammon, Schach
Robotik RT	Sensorwerte Stellgröße	Sollwert	Pendel, Roboterfußball
Reihenfolge- planung	Zustand Kandidat	Gewinn	Fertigungsstrasse Mobilfunknetz
Benchmark	Zustand Richtung	Zielposition	Labyrinth

Ziel: Selbständig lernendes System



Nennen Sie Stichworte zu diesem Bild!

Ziel: Selbständig lernendes System



Nennen Sie Stichworte zu diesem Bild! Interagieren mit der Umgebung, Belohnung oder Bestrafung durch die Umwelt, Wahrnehmung, Aktionsausführung, Lernen was gutes oder gar optimales Verhalten ist, ...

Vorgehensweise – Grobskizze

Mit welcher Schrittfolge kann eine Problemstellung als RL-Problem angegangen und gelöst werden?

Vorgehensweise – Grobskizze

Mit welcher Schrittfolge kann eine Problemstellung als RL-Problem angegangen und gelöst werden?

1. Formulierung des Lernproblems als Optimierungsaufgabe
 - Erfordert eine passende Formalisierung!
2. Lösung durch Lernen basierend auf Optimierungsverfahren (Algorithmen) des dynamischen Programmierens

Vorgehensweise – Grobskizze

Mit welcher Schrittfolge kann eine Problemstellung als RL-Problem angegangen und gelöst werden?

1. Formulierung des Lernproblems als Optimierungsaufgabe
 - Erfordert eine passende Formalisierung!
2. Lösung durch Lernen basierend auf Optimierungsverfahren (Algorithmen) des dynamischen Programmierens

Herausforderungen:

- sehr großer Zustandsraum (vor allem bei praktischen Problemstellungen)
- Prozessverhalten (Systemdynamik des konkreten Anwendungsproblems) sind unbekannt
- Einsatz approximativer Verfahren ist unumgänglich (z.B. Verwendung neuronaler Netze)

VL-Gliederung und Ausblick

Teil 1: Wissen, Lernen, Adaptivität (abgeschlossen)

Teil 2-3: Einführung in das optimierende Lernen

Teil 4-6: Dynamisches Programmieren

Markov'sche Entscheidungsprobleme, Rückwärts DP, Wertiteration, Strategieiteration, etc.



Teil 7-9: Approximatives optimierendes Lernen

Monte Carlo Methoden, stochastische Approximation, $TD(\lambda)$, Q-learning, etc.

ab Teil 10: Fortgeschrittene Aspekte des optimierenden Lernens

Funktionsapproximation, Policy-Gradient-Methoden, etc.

Literatureempfehlungen

-  A. Barto and R. Sutton. *Reinforcement Learning: An Introduction (second edition)*. Adaptive Computation and Machine Learning Series, 2018.
-  L. Graesser and K. Loon. *Foundations of Deep Reinforcement Learning: Theory and Practice in Python*. Addison-Wesley Data & Analytics, 2019



RL-FAQ (2004) <http://www.incompleteideas.net/RL-FAQ.html>

”
The only stupid question is the one
you were afraid to ask but never did.
Richard Sutton