Moderne Methoden der KI

Prof. Dr. sc. Hans-Dieter Burkhard
Vorlesung Sommer-Semester 2007

Einführung
Agenten
MDP

(1) EINFÜHRUNG

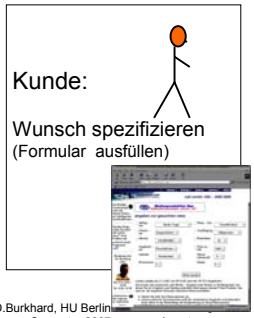
Was ist ein Agent?
Agenten-Orientierte Techniken/Programme
Multi-Agenten-Systeme (MAS)
Verteilte KI (VKI; Distributed AI -- DAI)
Modellierung
Markov-Entscheidungsprozesse

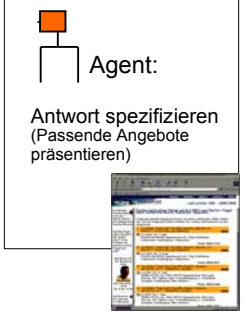
Beispiele

- Reise-Beratung
- Klinik-Informations-System
- Workflow
- RoboCup

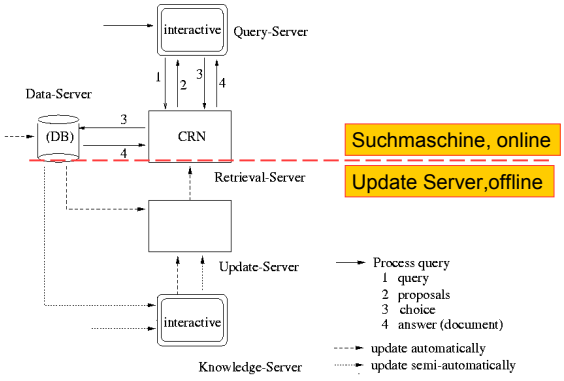
Reiseberatung: einfache Variante

Suchmaschine
"Stimulus-Response"





Reiseberatung: einfache Variante



Suchmaschine, online

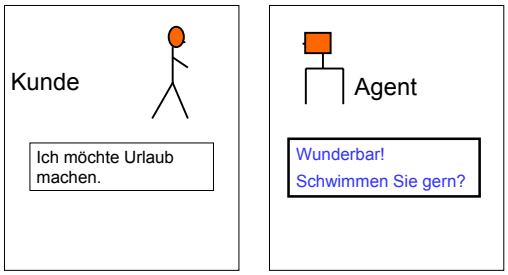
Update Server,offline

Reiseberatung: einfache Variante

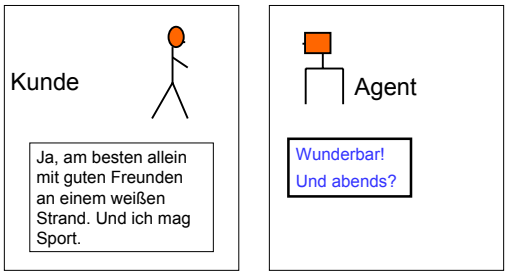
- Web-Basierte Technologie für Kommunikation
- Aufbereitung von Anfragen
- Suchmaschine
- Komfortable Präsentation

- Kontinuierliche automatische Aktualisierung,
- Hintergrund-Wissen für Anfragebearbeitung/Zugriff

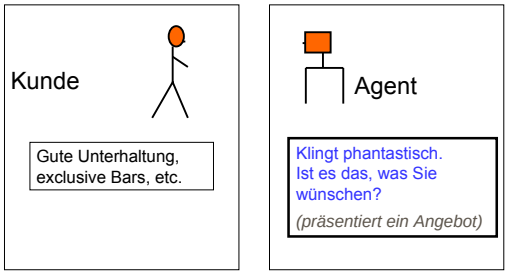
Reiseberatung: komplexer



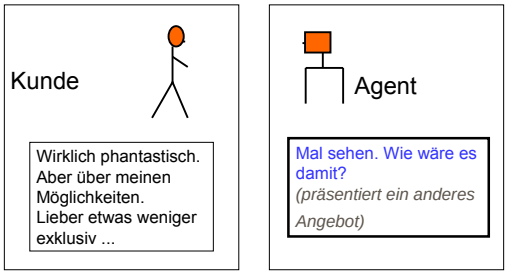
Reiseberatung: komplexer



Reiseberatung: komplexer



Reiseberatung: komplexer



Modellierung menschlichen Handelns

Reiseberatung: komplexer

- Dialogführung: Interaktive Beratung
  - –Einschätzung des Gesprächs-Standes
  - –Gezielte Fragen
  - –Wissen über üblichen Gesprächsverlauf
  - –Wissen über aktuellen Gesprächsverlauf
  - –Verhandeln mit Kunden
- Kooperation mit anderen Agenten (Hotel, Flug, ...)
- Individuelle Konfiguration von Angeboten
- Visualisierung
- Charakter, Emotionen, soziales Verhalten, ...

Reiseberatung: komplexer

Agent benötigt “dynamisches” Wissen zum Dialog:

Historie des Dialogs

(Hypothetisches) Modell des Kunden mit dessen
- Wünschen, Absichten
- Fähigkeiten
- Ansichten

(Flexibler) Plan für
- Erkundung der Wünsche, Absichten des Kunden
- Verkauf profitabler Angebote

*Modellierung von Ansichten, Wünschen, Absichten, Entscheidung/Plan bei **Kunde** und **Agent***

Klinik-Informations-Systeme

Elektronische Patientenakte:
- Administration, Abrechnung
- Krankengeschichte, Komplikationen, Allergien
- Befunde (Labor, Röntgen, Ultraschall, EEG, ...)
- Therapieverlauf
- Ambulante Behandlung …

Zugriff für externe Behandlung
Datenschutz
Datensicherheit

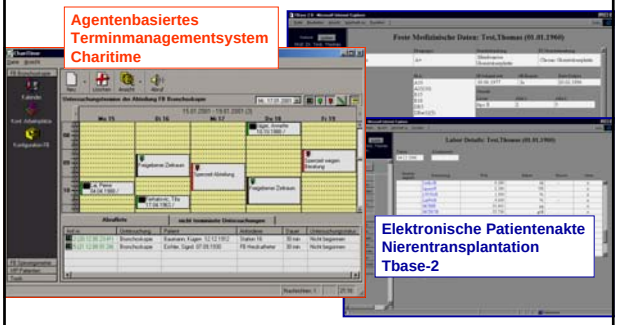
Klinik-Informations-Systeme

Klinikmanagement
- Bestellsystem
- Versorgung
- Wartung
- Bestellung
- Abrechnung
- Personalmanagement
- Dienstplanung …

Klinik-Informations-Systeme

Forschung
- Bereitstellung von Daten
- Dokumentation
- Testserien …
- …

Klinik-Informations-Systeme



Klinik-Informations-Systeme

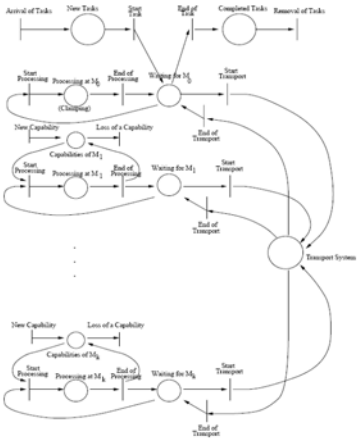
Agenten für
- Patienten
- Klinik-Personal
- Abteilungen
- Rollen/Aufgaben
- …

Workflow-Management (flexible Fertigung)

„Maintenance-Problem“:
Dauerhafte Sicherung von Eigenschaften
(vs. „Achievement Problem“: Erreichen eines Ziels)

Scheduling:
- Fortschreibung/aktualisierung von Produktionsplänen

Automatisierung:
- Produktionsüberwachung



Petri-Netz für Werkstück-Agent

Workflow-Management (flexible Fertigung)

Realisierungsvarianten:

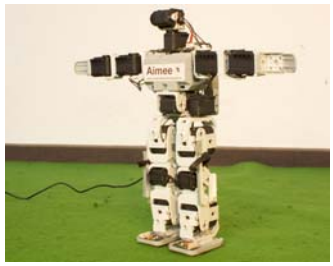
- Monolithisches System (Komplexer „Agent“)
- Verteiltes System („Multi-Agenten-System“)
  - Agenten für Maschinen
  - Agenten für Werkstücke
  - Agenten für Produkte
  - Agenten für Produktionsbereiche …

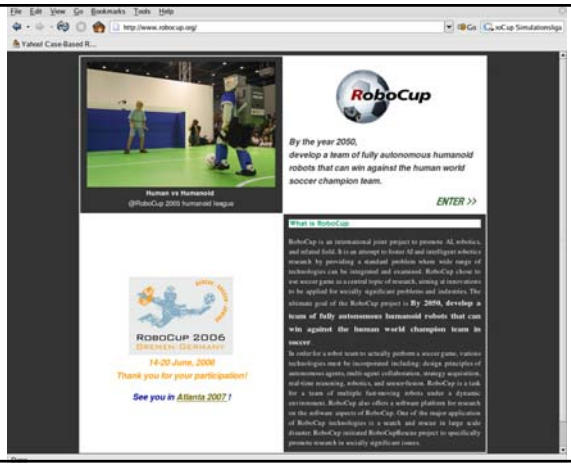
RoboCup

Diplomarbeit Daniel Hein

http://www.robocup.de/AT-Humboldt/simloid-evo.shtml?de

RoboCup





RoboCup

10 Jahre RoboCup: 1997 - 2006

Melbourne 2000

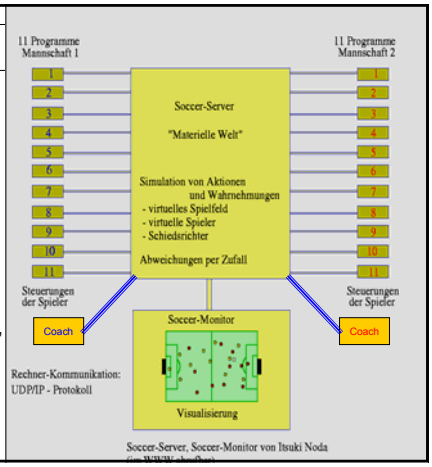
Bremen 2006

RoboCup

Simulation

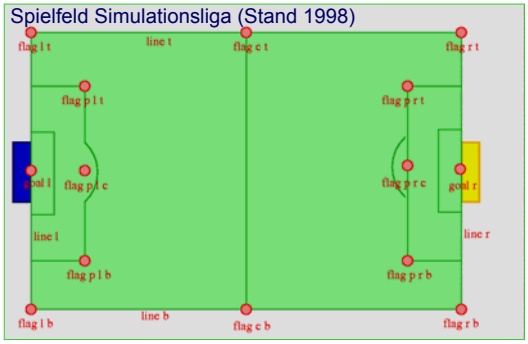
Aktionen
kick,dash,turn,
turn-neck, catch,
say, ..

Sensorik
see, sense-body,
hear, ...



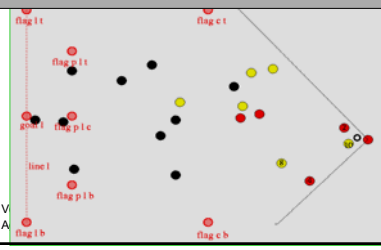
RoboCup

Spielfeld Simulationsliga (Stand 1998)



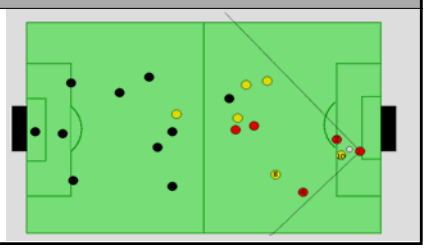
1. Informationen des RoboCup–Servers

```
Receive:
(see 271 ((goal l) 100.5 0) ((flag c t) 61.6 36) ((flag c b) 56.8 -33)
         ((flag l t) 107.8 19) ((flag l b) 104.6 -17)
         ((flag p l t) 87.4 14) ((flag p l c) 83.9 1)
         ((flag p l b) 85.6 -12) ((flag p r c) 12.8 17 -0 -0)
         ((ball) 2 5 -0.92 29.2) ((player) 99.5 2) ((player) 90 1)
         ((player) 81.5 -14) ((player) 81.5 14) ((player) 54.6 -3)
         ((player GELB) 54.6 9) ((player GELB) 40.4 5)
         ((player GELB 8) 33.1 -29 0 -0) ((player GELB) 36.6 37)
         ((player GELB 10) 4.1 -1 0 0) ((player GELB) 40.4 28)
         ((player ROT 2) 7.4 27 -0 -0) ((player ROT 4) 24.5 -42 -0 0.1)
         ((player) 44.7 13) ((player ROT) 40.4 -3) ((player ROT) 36.6 1)
         ((player) 66.7 20) ((player) 54.6 -17) ((player) 73.7 9)
         ((player) 60.3 -7) ((line l) 100.5 88))
```



1. Informationen des RoboCup–Servers

```
Receive:
(see 271 ((goal l) 100.5 0) ((flag c t) 61.6 36) ((flag c b) 56.8 -33)
         ((flag l t) 107.8 19) ((flag l b) 104.6 -17)
         ((flag p l t) 87.4 14) ((flag p l c) 83.9 1)
         ((flag p l b) 85.6 -12) ((flag p r c) 12.8 17 -0 -0)
         ((ball) 2 5 -0.92 29.2) ((player) 99.5 2) ((player) 90 1)
         ((player) 81.5 -14) ((player) 81.5 14) ((player) 54.6 -3)
         ((player GELB) 54.6 9) ((player GELB) 40.4 5)
         ((player GELB 8) 33.1 -29 0 -0) ((player GELB) 36.6 37)
         ((player GELB 10) 4.1 -1 0 0) ((player GELB) 40.4 28)
         ((player ROT 2) 7.4 27 -0 -0) ((player ROT 4) 24.5 -42 -0 0.1)
         ((player) 44.7 13) ((player ROT) 40.4 -3) ((player ROT) 36.6 1)
         ((player) 66.7 20) ((player) 54.6 -17) ((player) 73.7 9)
         ((player) 60.3 -7) ((line l) 100.5 88))
```

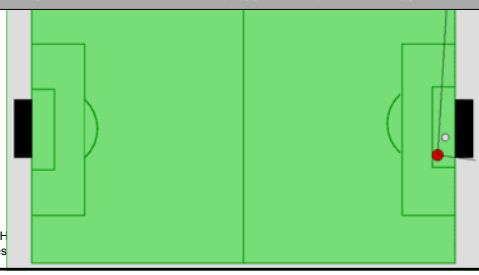


2. Aktionen eines Spielers

```
Send: (turn 129.675491)
Send: (dash 100.000000)
Send: (kick 100.000000, -129.675491)
```

3. Neue Informationen nach Drehung und Schuß des Spielers

```
Receive:
(see 274 ((goal r) 5.9 -1) ((flag r t) 38.5 -41)
         ((ball) 2.7 -27 0.972 11.9) ((line r) 5.8 -48))
```



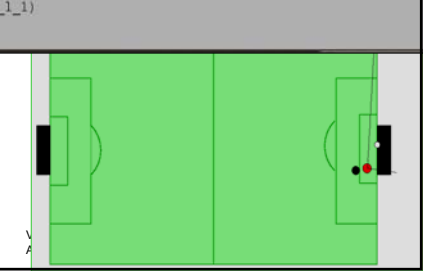
4. Weitere Aktionen des Spielers

```
Send: (dash 100.000000)
Send: (dash 100.000000)
Send: (dash 100.000000)
```

5. Neue Informationen nach Sprint des Spielers

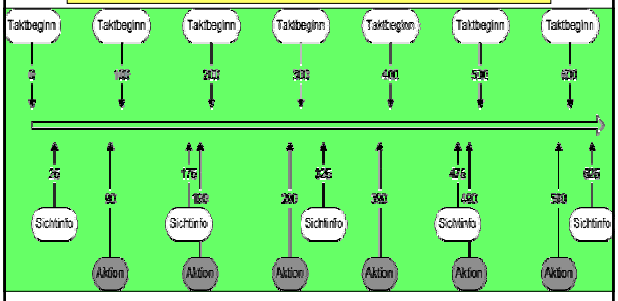
```
Receive:
(see 277 ((goal r) 3.4 0) ((flag r t) 36.2 -44) ((ball) 3.7 -14 0.518 3.7)
         ((Player) 2.5 -134) ((line r) 3.4 -48))
Receive:
(hear 278 referee goal_l_1)
```

--- TOR!!! ---



RoboCup

Sichtinformation wahlweise alle 75,150 oder 300 ms in entsprechend unterschiedlicher Qualität



RoboCup

Varianten:

Roboter = Agent

Roboter als Agenten-System
Einzelne Moduln als Agenten („Society of Mind“)

Neue Technologie?

The Guardian 12.5.1992:

Agent-based computing (ABC) is likely to be the next significant breakthrough in software development. ...

ABC has the characteristics of a typical new paradigm in software:

Many people will say that it is unnecessary,

others will say that it will never work,

others will say that it is simply a new name for something that they have been doing all along.

Neue Technologie?

Man kann auch alles mit TURING-Maschinen beschreiben.

Neue Technologie?

Anwendungen (z.B. im Internet) stellen neue Anforderungen

- Autonome Programme: Programme (re-)agieren selbständig.
- Interaktion zwischen Programmen.
- Keine zentrale Steuerung.

„Agenten“
als Metapher für selbständiges Handeln zur Erfüllung von Aufträgen.

Offene Systeme

Definition von Hewitt:

- kontinuierliche Bereitschaft
- Erweiterbarkeit
- dezentrale Steuerung
- asynchrone Arbeitsweise
- inkonsistente Informationen
- „Armlängen-Reichweite“

Verteilte KI (VK)

Distributed AI (DAI)

Zusammenarbeit intelligenter Systeme (Agenten)

- Kooperation
- Koordination
- Rollen
- Verhandlungen
- …

Sozionik

Soziologie + Informatik (speziell Verteilte KI)

1. Anwendung sozialer Begriffe in der Informatik
2. Simulation sozialer Systeme
3. Untersuchung hybrider Systeme (Mensch, Maschine)
   Akteure: Menschen + Maschinen (Agenten, Roboter)

- Technische Agenten als menschlichen Akteuren vergleichbare Kooperationspartner
- Verteilte Handlungsträgerschaft
- Anpassung von Strukturen und Kooperationsformen

Verteiltes Problemlösen vs. MAS

Verteiltes Problemlösen:
Erledigung von (gemeinsamen) Aufgaben durch kooperierende Programme

Gesichtspunkte:
- Aufgabenverteilung
- Kooperation

Multi-Agenten-Systeme (MAS):
Erledigung individueller Aufgaben in einer gemeinsamen Umgebung

Gesichtspunkte:
- Koordination

Was ist ein Agent?

Mögliche Definition:
Ein Software-Agent ist ein längerfristig arbeitendes Programm, dessen Arbeit sinnvoll als eigenständiges Erledigen von Aufträgen oder Verfolgen von Zielen in Interaktion mit einer Umgebung beschrieben werden kann.

Autonomie!

Weitere Attribute: intelligent, rational, … (später mehr dazu)

Roboter: Agent in realer Umwelt mit Sensoren und Aktoren.

Was ist ein Agent?

Zuschreibungsproblem:

Ab welcher Komplexität ist es sinnvoll, von Agenten mit entsprechenden Attributen (Autonomie, Wissen, Absichten, Ziele, Entscheidung/Wille, Bewusstsein,…) zu reden?

Dabei auch Skalierungsproblem
(Thermostat vs. Intelligentes Haus)

Was ist ein Agent?

To ascribe certain *beliefs, free will, intentions, consciousness, abilities or wants* to a machine or computer program is
- legitimate when such an ascription expresses the same information about the machine that it expresses about a person. It is
- useful when the ascription helps us understand the structure of the machine, its past or future behaviour, or how to repair or improve it. It is
- perhaps never logically required even for humans, but expressing reasonably briefly what is actually known about the state of the machine in a particular situation may require mental qualities or qualities isomorphic to them. Theories of belief, knowledge and wanting can be constructed for machines in a simpler setting than for humans, and later applied to humans. Ascription of mental qualities is
- most straightforward for machines of known structure such as thermostats and computer operating systems, but it is
- most useful when applied to entities whose structure is very incompletely known.

J. Mc. Carthy: "Ascribing mental qualities to machines". Technical Report Memo 326, Stanford AI Lab, 1979. Zitiert nach Y. Shoham: "Agent-oriented Programming". Artificial Intelligence(60),1993,51--92.

Agent und Umwelt

Interaktion des Agenten mit der Umwelt:

1. Aufnahme von Informationen („Eingabe“)
2. Beeinflussung der Umwelt („Ausgabe“)

Interne Verarbeitung im Agenten:

1. Auswertung der aufgenommenen Information
2. Entscheidungsprozesse (Auswahl von Aktionen)
3. Aktionen ausführen

Verarbeitungszyklus: sense-think-act

Umwelteigenschaften

Beobachtbarkeit
- –vollständig vs. partiell
- –korrekt vs. unsicher („Rauschen“)

Bestimmtheit
- –determiniert vs. nicht-determ./stochastisch

Wiederholbarkeit
- –episodisch (wiederholbar) vs. fortlaufend verändert

Dynamik
- –Dynamisch (schnell veränderlich) vs. statisch

Skalierung
- –Diskret vs. kontinuierlich

Modelle für Interaktion Umwelt/Agent

Terminologie für

- Zeit
- Agenten, Akteure (= Menschen, Agenten)
- Aktionen (der Akteure), Ereignisse (allgemein)
- Taktiken der Akteure (Abfolge von Aktionen)
- Wahrnehmungen der Akteure
- Zustandsbeschreibung („statische“ Beschreibung)
- Zustandsübergänge („dynamische Beschreibung“)
- Werten für die Aktionen der Akteure

Modelle für Interaktion Umwelt/Agent

Vereinfachende Annahmen notwendig, z.B.

- Zeit: diskrete Zeitpunkte t („Takte“, i.a. linear geordnet)
- Menge A von Akteuren/Agenten a
- Mengen U von Aktionen u , bzw. E von Ereignissen e
- Menge S von Zustände s („statische Beschreibung“)
- Werten/Kosten r („rewards“, $r \in \mathcal{R}$)
- Wahrnehmung

Zeit

Für Modellierung geeignet festlegen
- Kontinuierlich
- Diskret
- Nebenläufig
- Parallel

usw.

Synchronisationsprobleme
- zwischen Agenten
- zwischen internen/externen Aktionen eines Agenten

Wahrnehmung

Mögliche Annahmen:
- Agent kann Zustand unmittelbar beobachten
- Agent hat internes Weltmodell, das mittels Beobachtungen aktualisiert wird

Probleme:
- Korrektheit der Annahmen (belief vs. knowledge)
- Vollständigkeit der Beobachtungen/Annahmen

Modelle für Interaktion Umwelt/Agent

Dynamische Beschreibung in diskreter Welt

- Zusammenhänge zwischen Zuständen und Ereignissen
- Wie verändert sich die Welt durch Ereignisse

Agent beeinflusst die Welt durch Aktionen
- Strategie π legt Aktion fest
  (z.B. abhängig vom aktuellen Zustand/Ziel)

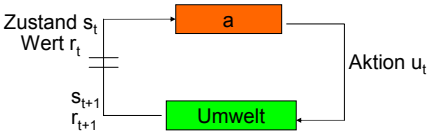
Modelle für Interaktion Umwelt/Agent

Ablaufbeschreibung für diskrete lineare Zeit z.B.

$r_0 s_0 \ -e_0\rightarrow\ r_1 s_1 \ -e_1\rightarrow\ r_2 s_2 \ -e_2\rightarrow\ ....$

oder mit Aktionen u_t eines Agenten a

$r_0 s_0 \ -u_0\rightarrow\ r_1 s_1 \ -u_1\rightarrow\ r_2 s_2 \ -u_2\rightarrow\ ....$



Modelle für Interaktion Umwelt/Agent

Genauere Modellierung mit
- Zuständen s^{ai} von Agenten a_i
- Aktionen u^{ai} von Agenten a_i
- Werten r^{ai} für Agenten a_i
- Zustand s^{Umwelt} der Umwelt
- Ereignissen e der Umwelt

Modelle für Interaktion Umwelt/Agent

Ereignisbezogene Modellierung, z.B.
- Ereignisse in einem Zeitabschnitt zwischen t und t+1

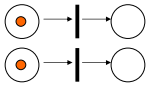
oder
- Berücksichtigung von Nebenläufigkeit
  (ohne genaue Zeitzuordnung)

Zustandsbasierte Modellierung
Zustände als Mengen von Fakten/Relationen (Datenbank)
z.B. modale Logik

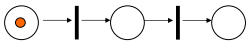
Nebenläufigkeit

Parallelität: zur gleichen Zeit

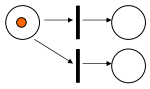
Nebenläufigkeit: zeitlich unabhängig



Kausalität: zeitliche Anordnung



Konflikt: keine Gleichzeitigkeit



Nebenläufige Systeme

Beschreibungsmöglichkeiten:
- Beschriftete Halbordnungen (“Traces”)
- “nondeterministic interleaving”:
  Mögliche sequentielle Abläufe

  Resultat: Verhalten des Systems als Sprache L über Menge der Ereignisse

  $L \subseteq E^*$

  Konvention:
  Betrachten alle Anfangsstücke, d.h.
  L ist Prefix-abgeschlossen: $pq \in L \Rightarrow p \in L$

Definition Deadlock

Eigenschaften von L (bzw. System) u.a.:

Verklemmung (Deadlock): $\exists p (p \in L \wedge \neg \exists e (e \in E \wedge pe \in L))$

Verklemmungsfrei: $\forall p(p \in L \Rightarrow \exists e (e \in E \wedge pe \in L))$

Definition Fairness

L ist fair bzgl. $e \in E$, falls

$\forall w (w \in Adh(L)$

$\Rightarrow \exists^{\infty} p (p \in Pref(w) \wedge pe \in L) \Rightarrow$ e unendlich oft in w

mit $Adh(L) := \{ w \in E^{\infty} \mid \forall p \in Pref(w) : p \in L \}$ „Adhärenz"

$Pref(w) := \{ p \in E^{*} \mid \exists q \in E^{\infty}: pq = w \}$ „endliche Präfixe"

L ist **fair**, falls für alle $e \in E$ gilt: L ist fair bzgl. e

Nebenläufige Systeme

Agent a im Gesamtsystem mit Aktionen $U \subseteq E$

System-Verhalten: $L \subseteq E^*$

Agenten-Verhalten: $L_a \subseteq U^*$

definierbar mittels löschendem Homomorphismus

$h_a(e) :=$ if $e \in U$ then e else leeres_Wort

Analog für weitere Agenten (Multi-Agenten-System).

Nebenläufige Systeme

Verklemmungsfreiheit des Gesamtsystems definieren mit L

lokale Verklemmungsfreiheit des Agenten definieren mit L_a

Fairness des Gesamtsystems definieren mit Adh(L)

lokale Fairness des Agenten definieren mit $Adh(L_a)$

Nebenläufige Systeme

Es gibt verklemmungsfreie Systeme, bei denen die Agenten aber nicht lokal verklemmungsfrei sind.

z.B. $L = Pref(ax^{\infty})$ $U_1 = \{a\}$, $U_2 = \{x\}$

Es gibt Systeme, die nicht verklemmungsfrei sind,
bei denen aber die Agenten lokal verklemmungsfrei sind.

z.B. $L = \{a\} \cup Pref(xa^{\infty}) \cup Pref(x^{\infty})$

$U_1 = \{a\}$, $U_2 = \{x\}$

Nebenläufige Systeme

Es gibt faire Systeme, bei denen die Agenten aber nicht lokal fair sind.

z.B. $L=Pref (\{ (ax)^n xy \mid n \in N \} \cup \{ (xa)^n ab \mid n \in N \})$

$U_1 = \{a,b\}$, $U_2 = \{x,y\}$

Es gibt Systeme, die nicht fair sind,
bei denen aber die Agenten lokal fair sind.

z.B. $L= \{a,x\}^*$

$U_1 = \{a\}$, $UA_2 = \{x\}$

Zustandsbeschreibungen

Zustand s der Welt insgesamt
Zustände s_a der Agenten a

Unterschied:
Beschreibung aus Sicht eines Beobachters (Programmierers)
Beschreibung aus Sicht eines Agenten

z.B.
als Menge von Fakten/Relationen (Datenbank) zur Zeit t

Zustandsbeschreibungen

Komplexitätsproblem: Nicht alles beschreibbar

Vereinfachende Annahmen
vgl. CWA, nichtmonotone Logiken (EKI)

Auswirkungen auf Beschreibung von Übergängen
Markov-Bedingung (später mehr)

Zustandsbeschreibungen

Zustände der Agenten: nur zugängliche Fakten

• Annahmen über Zustand der Umwelt/Agenten
$bel_a(bel_b(2+2=5))$
• Aktuelle Aufgaben/Ziele
$goal_a\ (later(bel_b(2+2=4))\ )$

davon Aktionen ableiten:
IF $bel_a(bel_b(2+2=5))$ AND $goal_a\ (later(bel_b(2+2=4))\ )$
THEN do(a, teach b)

Modale Logik

Formalismus zur logischen Beschreibung von
• Möglichkeiten (Modale Logik)
• Annahmen/Wissen (Epistemische Logik)
• Verpflichtungen (Deontische Logik)
• Zeitlichen Entwicklungen (Temporale Logik)

(später mehr dazu)

Werte (rewards), analog: Kosten

Für einen Agenten zur Zeit t: r_t

Kumulativer Wert für $r_0s_0\ -u_0\rightarrow\ r_1s_1\ -u_1\rightarrow\ r_2s_2\ -u_2\rightarrow\ \ldots.$

Erwarteter Wert ab Zeit t : $R_t := \sum_{i=0\ldots\infty}\ \gamma^i\ r_{t+1+i}$
mit Discount-Faktor $0 < \gamma <=1$

Falls r_t beschränkt: R_t endlich

Strategie π sollen R_t maximieren (bzw. Kosten: minimieren)

Werte: Zielzustand erreichen

$r_0s_0\ -u_0\rightarrow\ r_1s_1\ -u_1\rightarrow\ r_2s_2\ -u_2\rightarrow\ \ldots.$
r_i =1, falls s_i Zielzustand, sonst 0

Bsp.: Labyrinth
Falls Zielzustand erreicht: Aktionen beenden
bzw. Zustand unverändert (rewards dabei 0)

Strategie π soll Zielzustand erreichen
(möglichst schnell: γ <1)

Werte (rewards): Stab (pole balancing)

Aktionen: Bewegung
Zustand: Ort, Winkel des Stabs
Geschwindigkeit bzgl. Ort und Winkel

Rewards: +1 für nicht umgefallen (ohne discount: Zeit)

Alternativ: -1 für umgefallen (mit discount), 0 sonst

Werte (rewards) in nichtdeterminist. Welt

Aktionen des Agenten können zu unterschiedlichen Zuständen führen (Schach, Fußball, …)

Mögliche Abläufe $r_0 s_0 \ -u_0\rightarrow\ r_1 s_1 \ -u_1\rightarrow\ r_2 s_2 \ -u_2\rightarrow\ \ldots.$ betrachten und z.B. (?)

- Maximal/minimal erreichbares $R_t := \sum_{i=0\ldots\infty} \gamma^i r_{t+1+i}$ („optimistisch“ / „pessimistisch“)
- Erwartungswert für R_t

Aber: Agent kann Aktionen u_i (Strategie π) an erreichte Zustände anpassen,

Nutzen/Utility

Aktionen u führen von Zustand s zu Zuständen s‘
mit Wahrscheinlichkeiten P(s,u,s‘)

Welche Aktion soll ein rationaler Agent wählen?

Kriterium zunächst nur: Wert r(s‘) des erreichten Zustandes
(später: Erwartungswert R_t in Abhängigkeit von π)

Erwartungswert: $N(s,u) := \sum_{s' \in S} P(s,u,s') r(s')$

Wahl der Aktion u mit maximalem Nutzen N(s,u)

Nutzen/Utility

Frage: Gibt es evtl. andere Kriterien?

Betrachten

„Lotterie“ $[p_1:r_1, p_2:r_2, \ldots, p_n:r_n]$ mit $\sum p_i = 1$
mit Wahrscheinlichkeit p_i tritt Ergebnis r_i ein

Nutzen/Utility

Beispiel (Russell/Norvig)

Lotterie	A	80% : 4000
	B	100%: 3000
	C	20% : 4000
	D	25%: 3000

Welche Lotterie sollte ein rationaler Agent wählen?

Weitere Faktoren:
Erwartete Veränderung statt absoluter Werte der Lotterie

Nutzen/Utility

Betrachten nur Präferenzen für Ergebnisse:

Höhere Präferenz für A als für B : $A \succ B$
Indifferenz zwischen A und B : $A \sim B$
Höhere Präferenz oder Indifferenz: $A \succeq B$

Nutzen/Utility

Axiome, d.h. erwünschte Eigenschaften:

Linearität $A \succ B \vee B \succ A \vee A \sim B$

Transitivität $(A \succ B \wedge B \succ C) \rightarrow A \succ C$

Steitgkeit $(A \succ B \wedge B \succ C) \rightarrow \exists p\,[p:A, 1\text{-}p:C] \sim B$

Einsetzbarkeit: Indifferenz bleibt bei Erweiterung erhalten

$A \sim B \rightarrow [p:A, 1\text{-}p:C] \sim [p:B, 1\text{-}p:C]$

Monotonie: Präferenz bleibt bei erhöhter Wahrsch. erhalten

$A \succ B \rightarrow (p \geq q \leftrightarrow [p:A, 1\text{-}p:B] \succ [q:A, 1\text{-}q:B])$

Dekomponierbarkeit: Reduktion im Sinne Wahrsch.-Theorie

$[p:A, 1\text{-}p: [q:B, 1\text{-}q:C] \sim [p:A, (1\text{-}p)q:B, (1\text{-}p)(1\text{-}q):C]$

Nutzen/Utility

Wenn die Präferenz-Relation $\succ$ die Axiome erfüllt, dann gilt:

Es gibt eine reell-wertige Nutzensfunktion N mit

$N(A) > N(B)$ gdw. $A \succ B$

$N(A) = N(B)$ gdw. $A \sim B$

Der Erwartungswert ist eine Nutzensfunktion für Lotterien:

$N([p_1:r_1, p_2:r_2, \ldots, p_n:r_n]) = \sum p_i r_i$

Rationaler Agent maximiert Nutzensfunktion N

Mögliche Probleme:

- Maß für Werte r
- Abhängigkeit vom Ausgangszustand
- Mehrere Kriterien

Nützlichkeit bei mehreren Kriterien

Unterschiedliche Bezugspunkte (Aspekte/Kriterien) für Nützlichkeit

z.B. Kosten, Sicherheit, Umwelt, Aussehen, Akzeptanz, ...

Problematisch wegen

- Trade-offs (z.B. Kosten vs. Sicherheit)
- Abhängigkeiten (z.B. Kosten und Akzeptanz)

Beschreibung durch die Werte $x^{(i)}$ für Attribute $A^{(i)}$:

$[x^{(1)}, \ldots, x^{(m)}]$

Nützlichkeit bei mehreren Kriterien

Präferenzen auf Grundlage der Präferenzen für die Attribute

Strikte Dominanz:

$[x^{(1)}, \ldots, x^{(m)}] \succ [y^{(1)}, \ldots, y^{(m)}]$, falls $x^{(i)} \succ y^{(i)}$ für alle i

Berechnung einer Gesamtwerts z.B. als gewichtete Summe der Werte (oder auch andere Funktion) :

$$\sum w_i x_i$$

mit geeigneten Gewichten w_i als „Wichtigkeiten“.

Markov-Bedingung

Abfolge

$r_0 s_0 \xrightarrow{-u_0} r_1 s_1 \xrightarrow{-u_1} r_2 s_2 \xrightarrow{-u_2} \ldots. \xrightarrow{u_{t-1}} r_t s_t \xrightarrow{-u_t}$

i.a. Zustand/Wert $r_{t+1} s_{t+1}$ zur Zeit t+1
abhängig von gesamter Vergangenheit:

$$P(s_{t+1} = s', r_{t+1} = r \mid s_0 u_0 r_1 s_1 u_1 r_2 s_2 u_2 \ldots u_{t-1} r_t s_t u_t)$$

Markov-Bedingung:

$$= P(s_{t+1} = s', r_{t+1} = r \mid s_t u_t)$$

d.h. nur aus letztem Zustand/letzter Aktion berechenbar

Markov-Bedingung

Dynamik des Systems für **deterministisch** beschreibbare Welt mit Markov-Bedingung:

$$f: S \times U \rightarrow S \times \mathcal{R}$$

$$[s_{t+1}, r_{t+1}] = f(s_t, u_t)$$

kann man aufspalten in :

Übergangsfunktion $f_s: S \times U \rightarrow S$

Rewardfunktion $f_r: S \times U \rightarrow \mathcal{R}$

$$s_{t+1} = f_s(s_t, u_t)$$

$$r_{t+1} = f_r(s_t, u_t)$$

Markov-Bedingung

Dynamik des Systems für **stochastisch** beschriebene Welt mit Markov-Bedingung :

F: S x U → Verteilungen über S x $\mathcal{R}$

$P(s' = s_{t+1}, r = r_{t+1} \mid s_t u_t = su) = F(s',u)(s,r)$

daraus abgeleitet

Übergangswahrscheinlichkeit $\mathcal{P}_{sus'} := P(s' = s_{t+1} \mid s_t u_t = su)$

Reward-Wahrscheinlichkeit $\mathcal{P}_{sus'r} := P(r = r_{t+1} \mid s_t u_t s_{t+1} = sus')$

Erwartungswert für Reward $r_{sus'} := E(r_{t+1} \mid s_t u_t s_{t+1} = sus')$

Markov-Bedingung

Wann gilt Markov-Bedingung?

Eigenschaften der Welt?

Eigenschaften des Modells?

Markov-Entscheidungsprozess

Markov Decision Process (MDP) ist gegeben durch

- Diskrete lineare Zeitskala t=0,1,2,…
- (Endliche) Menge S von Zuständen s,
- Agent kann s vollständig und zuverlässig beobachten
- (Endliche) Menge U von Aktionen u des Agenten
- Dynamik beschrieben mittels
  - Übergangswahrscheinlichkeiten $\mathcal{P}_{sus'}$
  - Erwartungswerten $r_{sus'} \in \mathcal{R}$

Der Agent soll sich zu jedem Zeitpunkt t für eine Aktion entscheiden.

Markov-Entscheidungsprozess

Mögliche Erweiterungen:

Partielle Beobachtbarkeit der Zustände:
POMDP = partially observable MDP
mit beliefs des Agenten oder Wahrsch. modellieren

Vom Agenten nicht beeinflussbare Ereignisse:
Mittels Übergangswahrscheinlichkeiten modellieren

Nicht-stationäre Prozesse: Funktionen zeitabhängig

Strategie (policy)

Deterministische MDP mit deterministischer Strategie:

π: S → U

Agent entscheidet sich für die Aktion $u=\pi(s)$ in Abhängigkeit vom aktuellen Zustand s

Damit ergibt sich die Abfolge

$r_0 s_0 \; -u_0\rightarrow \; r_1 s_1 \; -u_1\rightarrow \; r_2 s_2 \; -u_2\rightarrow \; \ldots. \; u_{t-1}\rightarrow \; r_t s_t \; -u_t\rightarrow \; \ldots$

auf der Basis von

$u_t = \pi(s_t)$

$r_{t+1} = f_r(s_t,u_t)$

$s_{t+1} = f_s(s_t,u_t)$

Wertfunktion (value function) V

Bei gegebener Strategie π ist für jeden Zustand s der Wert bestimmt:

$V^{\pi}: S \rightarrow \mathcal{R}$

$V^{\pi}(s)$ ist der Wert, der bei Start in s und Arbeit gemäß π insgesamt erreicht wird:

$V^{\pi}(s_t) := R_t = \sum_{i=0\ldots\infty} \gamma^i \, r_{t+1+i}$

Strategie (policy)

Für stochastische MDP:

π: S x U → [0,1]

Agent entscheidet sich mit Wahrscheinlichkeit $\pi(s,u)$ für die Aktion u in Abhängigkeit vom aktuellen Zustand s

Daraus ergeben sich Wahrscheinlichkeiten für die Abfolgen

$r_0 s_0 \ -u_0\rightarrow\ r_1 s_1 \ -u_1\rightarrow\ r_2 s_2 \ -u_2\rightarrow\ \ldots.\ u_{t-1}\rightarrow\ r_t s_t \ -u_t\rightarrow$

auf der Basis von

$\pi(s_i,u_i)$, $\mathcal{P}_{sus'}$, $\mathcal{P}_{sus'r}$

Wertfunktion (value function) V

Bei gegebener Strategie π ist für jeden Zustand s der Erwartungswert bestimmt:

V^π: S → $\mathcal{R}$

$V^\pi(s)$ ist der Erwartungswert, der bei Start in s und Arbeit gemäß π insgesamt erreicht wird:

$V^\pi(s) := E(R_t \mid s = s_t, \pi) = E(\sum_{i=0\ldots\infty} \gamma^i r_{t+1+i} \mid s = s_t, \pi)$

Bellmann-Gleichung

$V^\pi(s) := E(R_t \mid s = s_t, \pi)$

$= E(\sum_{i=0\ldots\infty} \gamma^i r_{t+1+i} \mid s = s_t, \pi)$

$= E(r_{t+1} + \gamma \sum_{i=0\ldots\infty} \gamma^i r_{t+2+i} \mid s = s_t, \pi)$

$= \sum_u \pi(s,u) \sum_{s'} \mathcal{P}_{sus'} (r_{sus'} + \gamma E(\sum_{i=0\ldots\infty} \gamma^i r_{t+2+i} \mid s' = s_{t+1}, \pi))$

$= \sum_u \pi(s,u) \sum_{s'} \mathcal{P}_{sus'} (r_{sus'} + \gamma V^\pi(s'))$

Iterative Beziehungen zwischen den Werten $V^\pi(s)$

Bei n = card(S) gibt es n lineare Gleichnungen für n Werte.

Nach bekannten Verfahren lösbar.

Optimale Strategie π^*

π besser oder gleichwertig bzgl. π' ,
falls $V^\pi(s) \geq V^{\pi'}(s)$ für alle $s \in S$

Es gibt maximale Strategien (muss nicht eindeutig sein).

Maximale Strategie: π^*

Wertfunktion für maximale Strategie: V^*

$V^*(s) = \max_\pi V^\pi(s)$

Optimale Strategie π^*

Bellmann-Gleichung für π^*

$V^{\pi^*}(s) = \max_u \sum_{s'} \mathcal{P}_{sus'} (r_{sus'} + \gamma V^{\pi^*}(s'))$

n := card(S)

n nicht-lineare Gleichungen für n Werte $V^{\pi^*}(s)$

Lösung mit Methoden der dynamischen Programmierung

Voraussetzung: Modell ist bekannt

Spezialfälle: Suchverfahren (vgl. EKI)