11 Wegplanung mit Reinforcement Learning

11.1 Bezeichnungen – Hidden Markov Prozess

Einführung

- gelernt werden **Folgen von Zustands – Aktions – Paaren**
- zu einem gegebenen Zustand (Umweltsituation) soll eine passende (möglichst die beste) Aktion gewählt werden
- **nicht** durch Beispiele (Lehrer), sondern durch **Versuch und Irrtum**
- viele Versuche sind notwendig
- man erhält **nicht** zu jeder gewählten Aktion einen Feedback (gut/schlecht), sondern eher **selten**
- oft erst bei Erreichen eines **Zielzustands (Belohnung, Bestrafung)**

Beispiel - Schach

- Schachspiel mit einem Meister, der keine Kommentare gibt
- Zustände – Brettsituationen
- Aktionen – zulässige Züge
- eine Belohnung bekommt man nur am Ende einer Partie (Gewinn, Verlust, Remis)
- während der Partie gibt es keine Hinweise oder Belohnungen
- durch Spielen vieler Partien müssen gute Züge gelernt werden

Beispiel – Futtersuchender Agent

- Aktionen – Vorwärts-/Rückwärtsbewegung und Rechts-/Linksdrehung
- Belohnung – Futtereinheit
- vorher hat er keinen Hinweis darauf, wie er sich bewegen soll
- der Agent kann das Futter auch nicht sehen

Literatur

http://www-anw.cs.umass.edu/rlr/

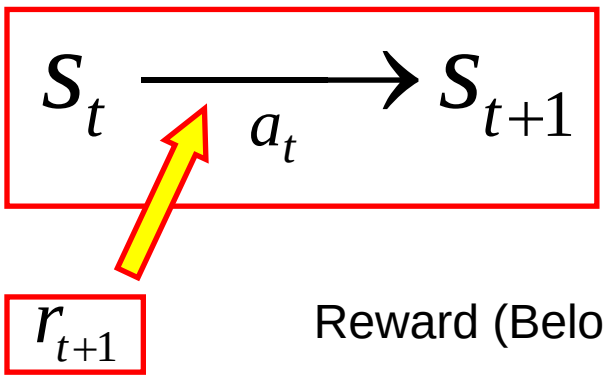
A.Barto; R.Sutton: Reinforcement Learning, 1998

Bezeichnungen

S Menge der Zustände

$A(s)$ Menge der möglichen Aktionen im Zustand $s \in S$

$A = \bigcup_{s \in S} A(s)$ Menge aller Aktionen

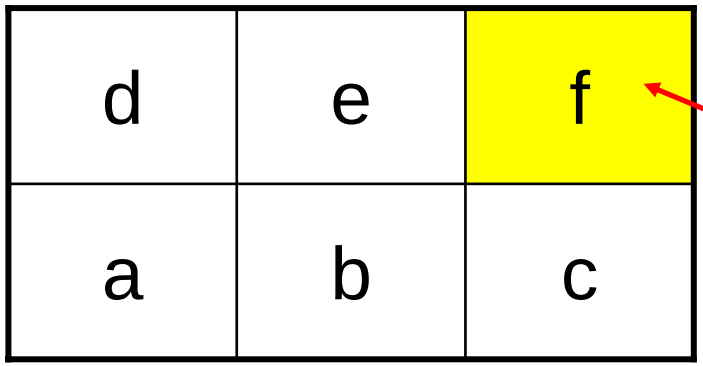


$s_t, s_{t+1} \in S$ $\quad a_t \in A(s_t)$

$t = 0,1,2,\ldots$

Reward (Belohnung) beim Übergang $s_t \longrightarrow s_{t+1}$

Beispiel – Labyrinth 1



$$S = \{a, b, c, d, e, f\}$$

$$A = \{\uparrow, \downarrow, \rightarrow, \leftarrow\}$$

$$A(a) = \{\uparrow, \rightarrow\}$$

$$A(b) = \{\uparrow, \rightarrow, \leftarrow\}$$

$$A(c) = \{\uparrow, \leftarrow\}$$

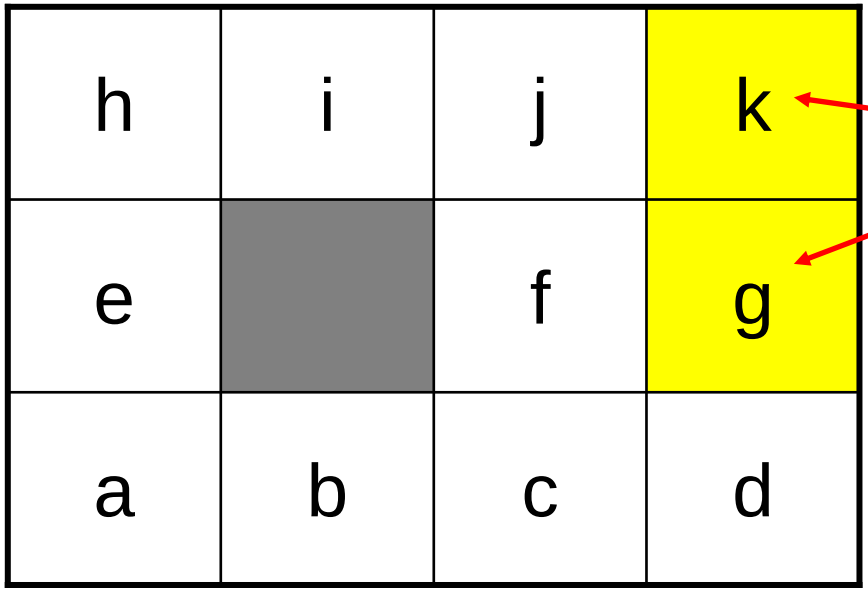
$$A(d) = \{\downarrow, \rightarrow\}$$

$$A(e) = \{\downarrow, \rightarrow, \leftarrow\}$$

$r = 100$ ⇒ $c \rightarrow f$, $e \rightarrow f$

$r = -1$ ⇒ sonst

Beispiel – Labyrinth 2



Ziele

$A = \{\uparrow, \downarrow, \rightarrow, \leftarrow\}$

$r = 100 \Rightarrow j \rightarrow k$

$r = -100 \Rightarrow f \rightarrow g \quad d \rightarrow g$

$r = 0 \Rightarrow$ sonst $(r = -20)$

Markov Prozess

$$s \xrightarrow[a]{} s'$$

Übergangswahrscheinlichkeiten $$P^a_{ss'} = P(s_{t+1} = s' \mid s_t = s, a_t = a)$$

Erwartungswert eines Rewards $$R^a_{ss'} = E(r_{t+1} \mid s_t = s, s_{t+1} = s', a_t = a)$$

Policy

$$\pi : S \times A \rightarrow [0,1]$$

$\pi(s,a)$ ist dabei die **Wahrscheinlichkeit** für die Ausführung der Aktion

$a \in A(s)$ in $s \in S$

Im einfachsten Fall sind nur die Werte 0 und 1 zulässig:

$\pi : S \rightarrow A$ Zu jedem Zustand $s \in S$

wird eine als nächstes auszuführende Aktion $a \in A(s)$ vorgeschlagen

Lernziel

Finden einer **optimalen** (muss noch genau definiert werden) **Policy**.

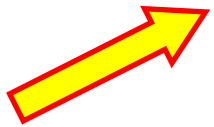
$$\pi^* : S \rightarrow A$$

hat immer diese Form

Beispiel – Labyrinth 1

d	e	f
a	b	c

eine optimale Policy:



$$\pi_1(a) = \uparrow$$

$$\pi_1(a,\uparrow) = 1 \qquad \pi_1(a,\rightarrow) = 0$$

$$\pi_1(b) = \rightarrow$$

$$\pi_1(c) = \uparrow$$

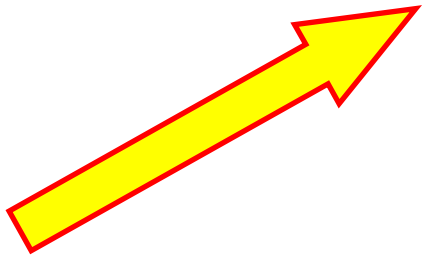
$$\pi_1(d) = \rightarrow$$

$$\pi_1(e) = \rightarrow$$

Beispiel – Labyrinth 1

d	e	f
a	b	c

zufällige Policy



$$\pi_2(a,\uparrow) = \pi_2(a,\rightarrow) = \frac{1}{2}$$

$$\pi_2(b,\uparrow) = \pi_2(b,\rightarrow) = \pi_2(b,\leftarrow) = \frac{1}{3}$$

$$\pi_2(c,\uparrow) = \pi_2(c,\leftarrow) = \frac{1}{2}$$

$$\pi_2(d,\downarrow) = \pi_2(d,\rightarrow) = \frac{1}{2}$$

$$\pi_2(e,\downarrow) = \pi_2(e,\rightarrow) = \pi_2(e,\leftarrow) = \frac{1}{3}$$

11.2 Wegplanung

Wegplanung mit Reinforcement Learning

$$s_t \rightarrow x_t \qquad a_t \rightarrow u_t$$

$$P^u_{xx'} = P(x_{t+1} = x' \,|\, x_t = x, u_t = u) \qquad \pi : x_t \rightarrow u_t$$

$$x_t \longrightarrow x_{t+1}$$

$$r_{t+1} = 100, \text{ falls } x_{t+1} \text{ Ziel}$$

$$r_{t+1} = -1, \text{ sonst}$$

11.3 Bellmanngleichungen

Erwarteter Return

$$R_t = \sum_{k=0}^{\infty} \gamma^k \cdot r_{t+k+1}$$

$0 \leq \gamma \leq 1$ ⟶ **Discountrate**

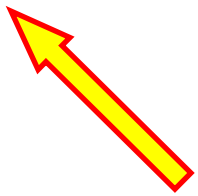
Je kleiner γ ist, desto mehr konzentriert sich der Agent auf kurzfristige Aktionen und je größer γ wird, desto vorausschauender verhält sich der Agent.

Zustandswertefunktion

$$V^{\pi}(s) = E(R_t \mid s_t = s)$$

Erwartungswert des Returns R_t, wenn der Agent in s startet und sich danach gemäß π verhält

$$R_t = \sum_{k=0}^{\infty} \gamma^k \cdot r_{t+k+1} \qquad 0 \leq \gamma \leq 1$$



erwarteter Return

Bellmanngleichungen

$$V^{\pi}(s) = \sum_{a \in A(s)} \pi(s,a) \sum_{s' \in S} P^{a}_{ss'} \cdot [R^{a}_{ss'} + \gamma \cdot V^{\pi}(s')]$$

deterministisch: $s' = s(a)$ $\qquad R^{a}_{ss'} = r(s,a)$

$$V^{\pi}(s) = \sum_{a \in A(s)} \pi(s,a) \cdot [r(s,a) + \gamma \cdot V^{\pi}(s(a))]$$

d	e	f
a	b	c

$$V^{\pi}(a) = \frac{1}{2}\gamma(V^{\pi}(b) + V^{\pi}(d))$$

$$V^{\pi}(e) = \frac{1}{3}\gamma(V^{\pi}(d) + V^{\pi}(b)) + \frac{1}{3}(100 + \gamma \cdot V^{\pi}(f))$$

Beispiel – Labyrinth 1

π - zufällig

$\gamma = 1$

100	100	0
100	100	100

$\gamma = 0.9$

52	66	0
49	57	76

Beispiel – Labyrinth 2

5.3	14.4	26.7	
-2.5		-36.6	
-11	-21.9	-37.6	-66.9

π - zufällig

$r = 0 \quad \gamma = 0.9$

Beispiel – Labyrinth 2

-128	-94	-37	
-145		-96	
-151	-145	-128	-117

π - zufällig

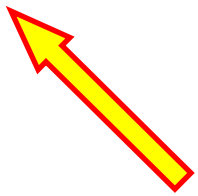
$r = -20 \quad \gamma = 0.9$

Aktionswertefunktion

$$Q^{\pi}(s,a) = E(R_t \mid s_t = s, a_t = a)$$

Erwartungswert des Returns R_t, wenn der Agent in s startet, die Aktion $a \in A(s)$ ausführt und sich danach gemäß π verhält

$$R_t = \sum_{k=0}^{\infty} \gamma^k \cdot r_{t+k+1} \qquad 0 \leq \gamma \leq 1$$



erwarteter Return

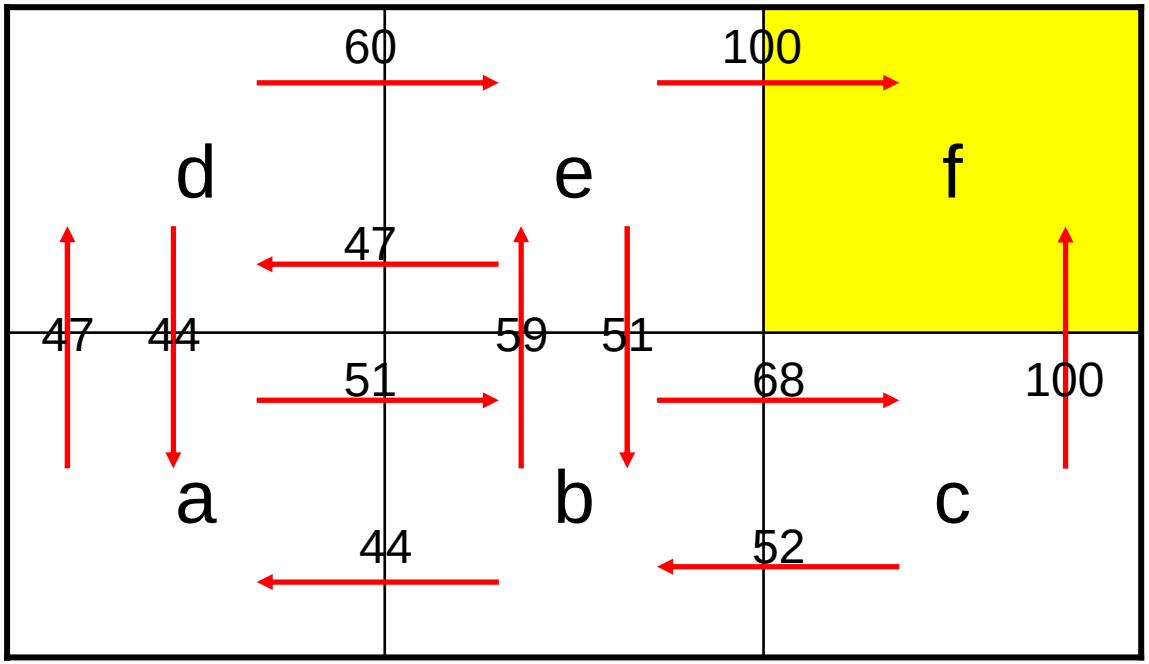
Bellmanngleichungen

$$Q^{\pi}(s,a) = \sum_{s' \in S} P^{a}_{ss'} \cdot [R^{a}_{ss'} + \gamma \cdot \sum_{a' \in A(s')} \pi(s',a') \cdot Q^{\pi}(s',a')]$$

deterministisch: $s' = s(a)$ $\qquad R^{a}_{ss'} = r(s,a)$

$$Q^{\pi}(s,a) = r(s,a) + \gamma \cdot \sum_{a' \in A(s(a))} \pi(s(a),a') \cdot Q^{\pi}(s(a),a')$$

Beispiel – Labyrinth 1



$\gamma = 0.9$

Optimale Policy

$$\pi \geq \pi' \Leftrightarrow V^{\pi}(s) \geq V^{\pi'}(s) \quad \forall s \in S$$

(Halbordnung)

$$\pi^* \geq \pi \qquad \forall \pi$$

Optimale Policy

$$V^*(s) := V^{\pi^*}(s) = \max_{\pi} V^{\pi}(s)$$

$$Q^*(s,a) := Q^{\pi^*}(s,a) = \max_{\pi} Q^{\pi}(s,a)$$

Optimale Werte

Bellmann - Optimaltätsgleichungen

$$V^*(s) = \max_{a \in A(s)} \sum_{s' \in S} P^a_{ss'} \cdot [R^a_{ss'} + \gamma \cdot V^*(s')]$$

$$Q^*(s,a) = \sum_{s' \in S} P^a_{ss'} \cdot [R^a_{ss'} + \gamma \cdot \max_{a' \in A(s')} Q^*(s',a')]$$

d	e	f
a	b	c

$$V^*(s) = \max_{a \in A(s)} \{r(s,a) + \gamma \cdot V^*(s(a))\}$$

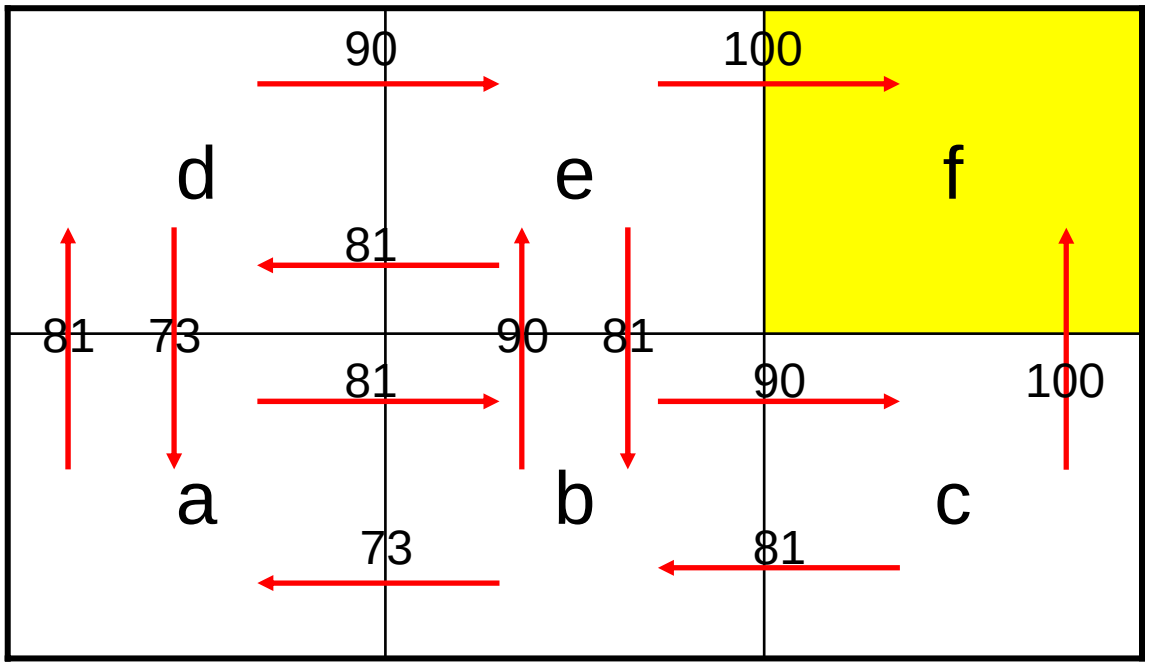
$$V^*(a) = \max\{\gamma \cdot V^*(b), \gamma \cdot V^*(d)\}$$

Beispiel – Labyrinth 1

$\gamma = 0.9$

90	100	f
81	90	100

Beispiel – Labyrinth 1



$\gamma = 0.9$

11.4 Lösungsmöglichkeiten

Lösungsmöglichkeiten

- Dynamische Programmierung
  - mathematisch gut untersucht
  - benötigen aber ein vollständiges Modell der Umgebung
- Monte – Carlo – Methoden
  - sind nicht auf ein Modell der Umgebung angewiesen
  - aber nicht für eine schrittweise, inkrementelle Berechnung geeignet, erst nach einer gesamten Episode können Iterationen durchgeführt werden
- Temporal – Difference - Methoden
  - benötigen kein Modell der Umgebung
  - vollständig inkrementel
  - aber schwer mathematisch zu analysieren

2 Lösungsansätze für RL

- TD – Lernen (Temporales Differenz Lernen)
- Q - Lernen

TD - Lernen

Wir berechnen die **Zustandswertefunktion**:

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

Die reellen Zahlen $V^{\pi}(s)$

teilen uns mit, wie **vorteilhaft das Erreichen eines Zustandes** $s \in S$ ist

TD - Lernen

Wir setzen $V^{\pi}(s) = 0$ s Zielzustand

Für alle anderen Zustände wird $V^{\pi}(s)$ iterativ berechnet.

TD - Lernen

Initialisierung: $V^{\pi}(s) = 0, \ \forall s \in S$

Iteration:

Beim Übergang $s_t \rightarrow s_{t+1}, \ (t = 0,1,\ldots)$ wird der Wert $V(s_t)$ neu berechnet:

$$V^{\pi}(s_t) \leftarrow V^{\pi}(s_t) + \alpha[r_{t+1} + \gamma \cdot V^{\pi}(s_{t+1}) - V^{\pi}(s_t)]$$

$\alpha > 0$	Lernparameter
r_{t+1}	Reward
$0 \leq \gamma \leq 1$	Discountrate

Beispiel – Labyrinth 1

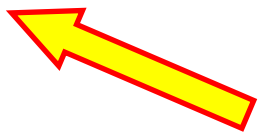
d	e	f
a	b	c

$\alpha = 0.9 \qquad \gamma = 1$

Wir erzeugen einige Episoden mit der zufälligen Policy.

0	0	0
0	0	0

Für jeweils einen Zustandsübergang berechnen wir die Zustandswerte iterativ.

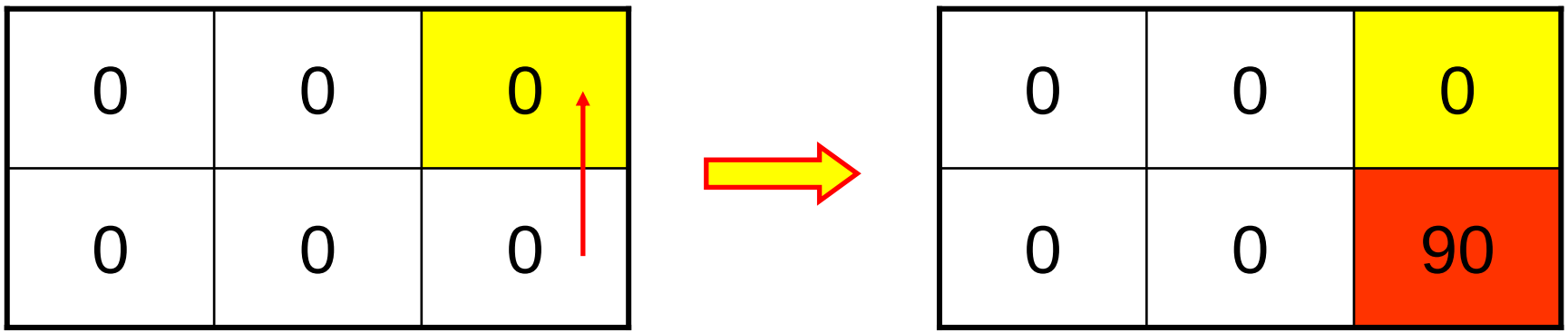


Initialisierung

Beispiel – Labyrinth 1

$$c \rightarrow f$$

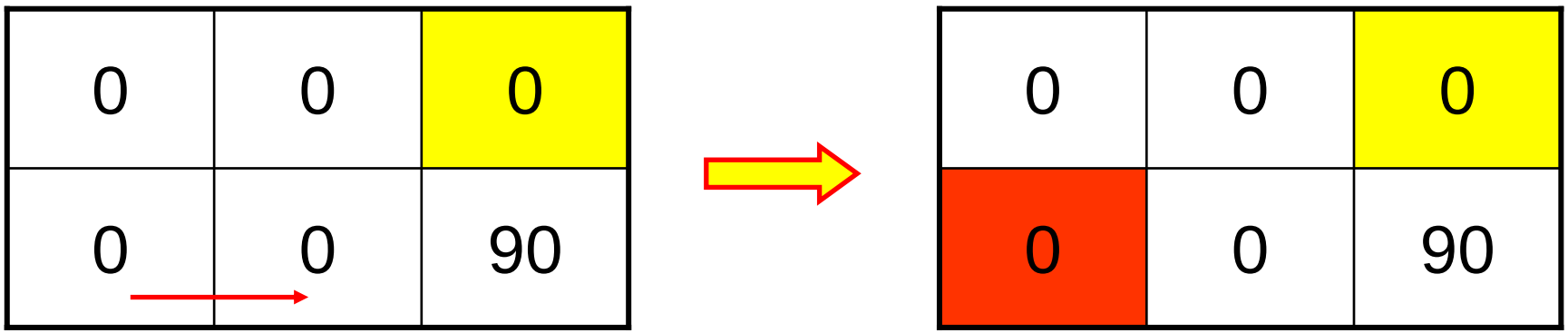
$$V(c) \leftarrow 0 + 0.9[100 + 0 - 0] = 90$$



Beispiel – Labyrinth 1

$$a \rightarrow b$$

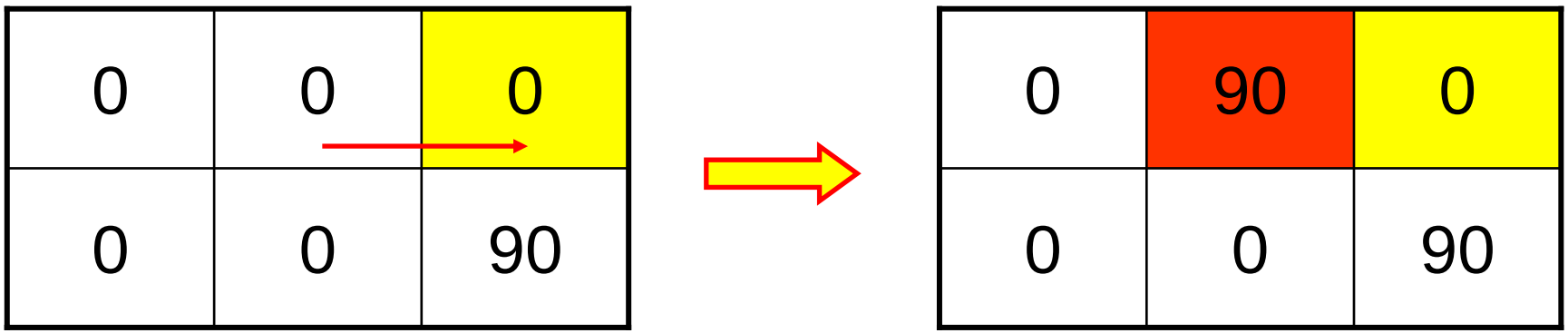
$$V(a) \leftarrow 0 + 0.9[0 + 0 - 0] = 0$$



Beispiel – Labyrinth 1

$$e \rightarrow f$$

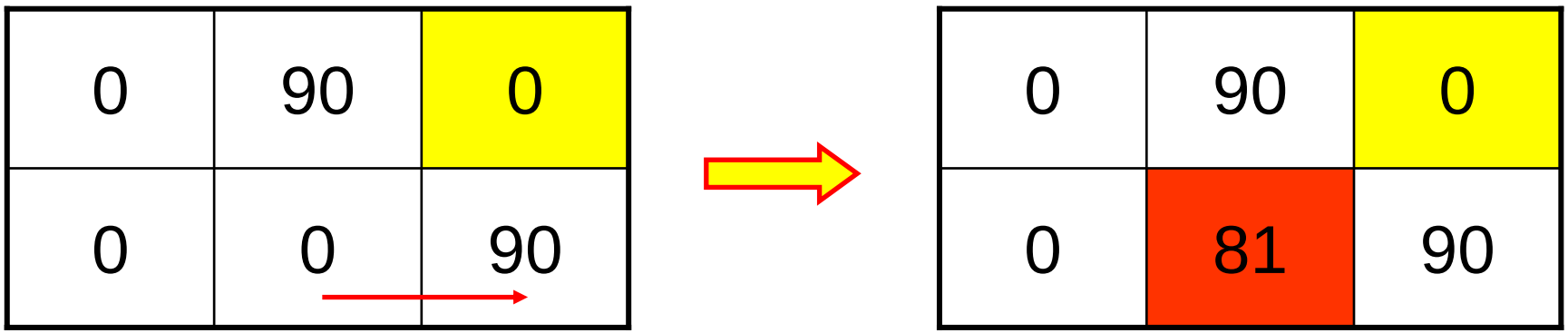
$$V(e) \leftarrow 0 + 0.9[100 + 0 - 0] = 90$$



Beispiel – Labyrinth 1

$$b \rightarrow c$$

$$V(b) \leftarrow 0 + 0.9[0 + 90 - 0] = 81$$



Beispiel – Labyrinth 1

$$d \rightarrow e$$

$$V(d) \leftarrow 0 + 0.9[0 + 90 - 0] = 81$$

0	90	0
0	81	90

81	90	0
0	81	90

Beispiel – Labyrinth 1

$$a \rightarrow b$$

$$V(a) \leftarrow 0 + 0.9[0 + 81 - 0] \approx 73$$

81	90	0
0	81	90

81	90	0
73	81	90

Beispiel – Labyrinth 1

$$c \rightarrow f$$

$$V(c) \leftarrow 90 + 0.9[100 + 0 - 90] = 99$$

81	90	0
73	81	90

81	90	0
73	81	99

Beispiel – Labyrinth 1

$$e \rightarrow f$$

$$V(e) \leftarrow 90 + 0.9[100 + 0 - 90] = 99$$

81	90	0
73	81	99

81	99	0
73	81	99

Beispiel – Labyrinth 1

$$c \rightarrow b$$

$$V(c) \leftarrow 99 + 0.9[0 + 81 - 99] \approx 83$$

81	99	0
73	81	99

81	99	0
73	81	83

Beispiel – Labyrinth 1

Im Grenzfall erhält man:

$\gamma = 1$

100	100	0
100	100	100

$\gamma = 0.9$

52	66	0
49	57	76

Beispiel – Labyrinth 2

5.3	14.4	26.7	
-2.5		-36.6	
-11	-21.9	-37.6	-66.9

$r = 0 \quad \gamma = 0.9$

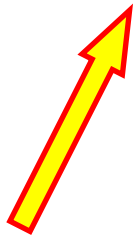
Beispiel – Labyrinth 2

-128	-94	-37	
-145		-96	
-151	-145	-128	-117

$r = -20 \quad \gamma = 0.9$

Policy - Iteration

$$\pi_0 \rightarrow V^{\pi_0} \rightarrow \pi_1 \rightarrow V^{\pi_1} \rightarrow \cdots \rightarrow \pi^* \rightarrow V^*$$



Policy-Verbesserung

$$\pi'(s) := \underset{a \in A(s)}{\operatorname{argmax}} Q^{\pi}(s,a)$$

Q - Lernen

Wir berechnen die optimale **Aktionswertefunktion**:

$$Q : S \times A \rightarrow R$$

Die reellen Zahlen $Q(s, a)$

teilen uns mit, wie **vorteilhaft eine Aktion** $a \in A(s)$ ist

Q - Lernen

Wir setzen $Q(s,a) = 0$ s Zielzustand

Für alle anderen Zustände wird $Q(s,a)$ iterativ berechnet.

Q - Lernen

Initialisierung: $Q(s,a) = 0, \ \forall s \in S, \ \forall \mathrm{a} \in \mathrm{A(s)}$

Iteration:

Beim Übergang $s_t \xrightarrow{a_t} s_{t+1}, \ (t = 0,1,\ldots)$ wird der Wert $Q(s_t, a_t)$ neu berechnet:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_{a' \in A(s_{t+1})} Q(s_{t+1}, a') - Q(s_t, a_t)]$$

$\alpha > 0$	Lernparameter
r_{t+1}	Reward
$0 \le \gamma \le 1$	Discountrate

Beispiel – Labyrinth 1

d	e	f
a	b	c

$\alpha = 0.9 \qquad \gamma = 1$

Wir erzeugen einige Episoden mit der zufälligen Policy.

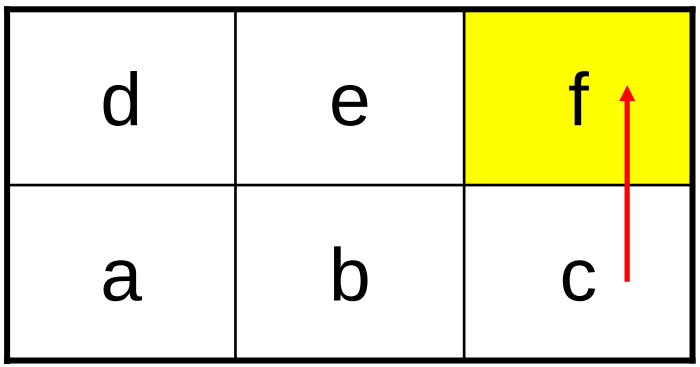
Für jeweils einen Zustandsübergang berechnen wir die Zustandswerte iterativ.

Initialisierung $Q(s,a) = 0$

Beispiel – Labyrinth 1

$$c \rightarrow f$$

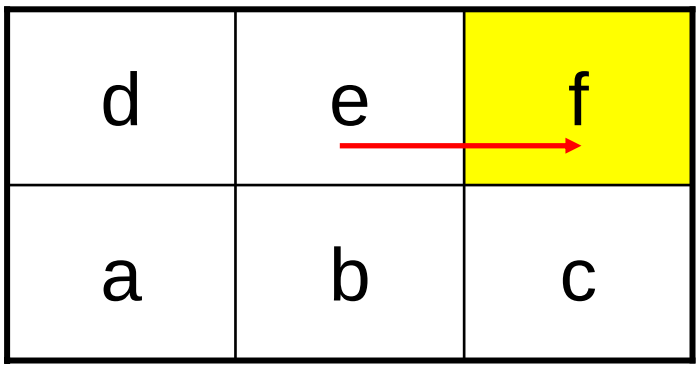
$$Q(c, \uparrow) \leftarrow 0 + 0.9[100 + 0 - 0] = 90$$



Beispiel – Labyrinth 1

$$e \rightarrow f$$

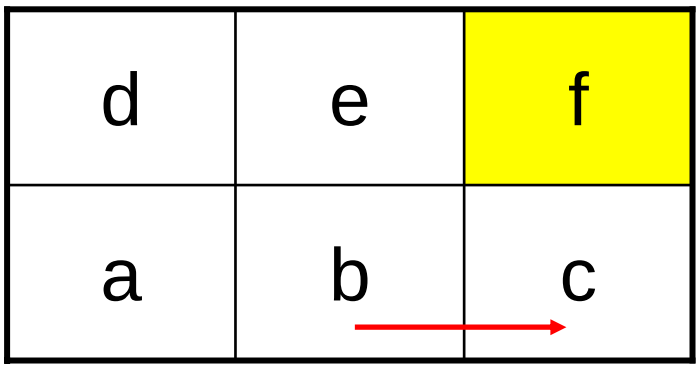
$$Q(e, \rightarrow) \leftarrow 0 + 0.9[100 + 0 - 0] = 90$$



Beispiel – Labyrinth 1

$$b \rightarrow c$$

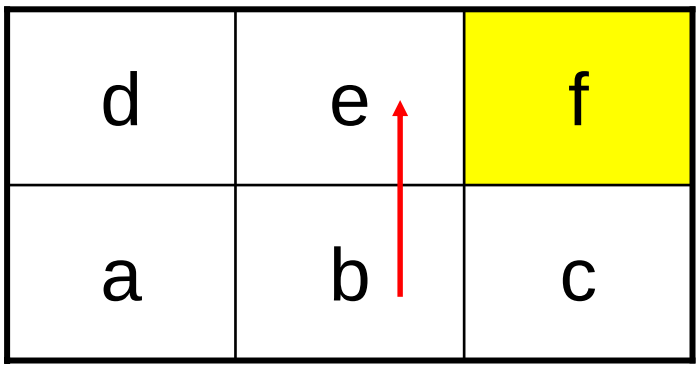
$$Q(b, \rightarrow) \leftarrow 0 + 0.9[0 + 90 - 0] = 81$$



Beispiel – Labyrinth 1

$$b \rightarrow e$$

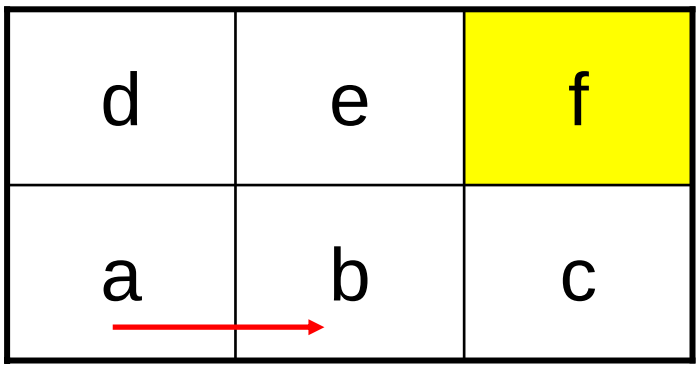
$$Q(b, \uparrow) \leftarrow 0 + 0.9[0 + 90 - 0] = 81$$



Beispiel – Labyrinth 1

$$a \rightarrow b$$

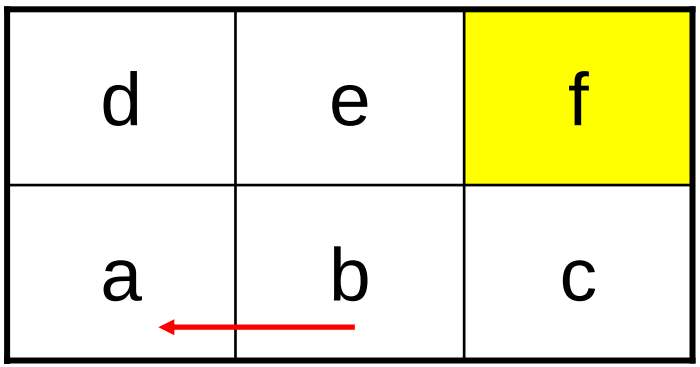
$$Q(a, \rightarrow) \leftarrow 0 + 0.9[0 + 81 - 0] \approx 73$$



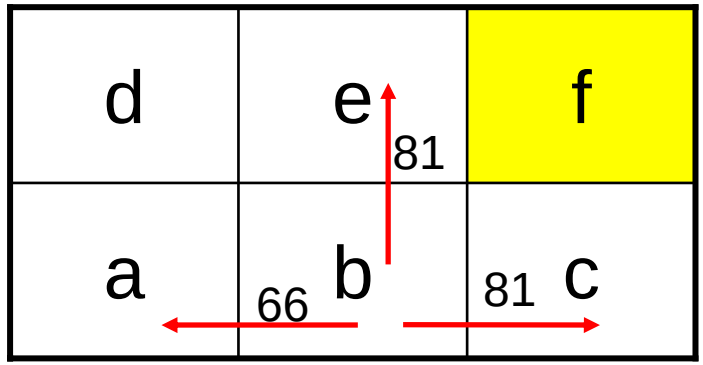
Beispiel – Labyrinth 1

$$b \rightarrow a$$

$$Q(b, \leftarrow) \leftarrow 0 + 0.9[0 + 73 - 0] \approx 66$$



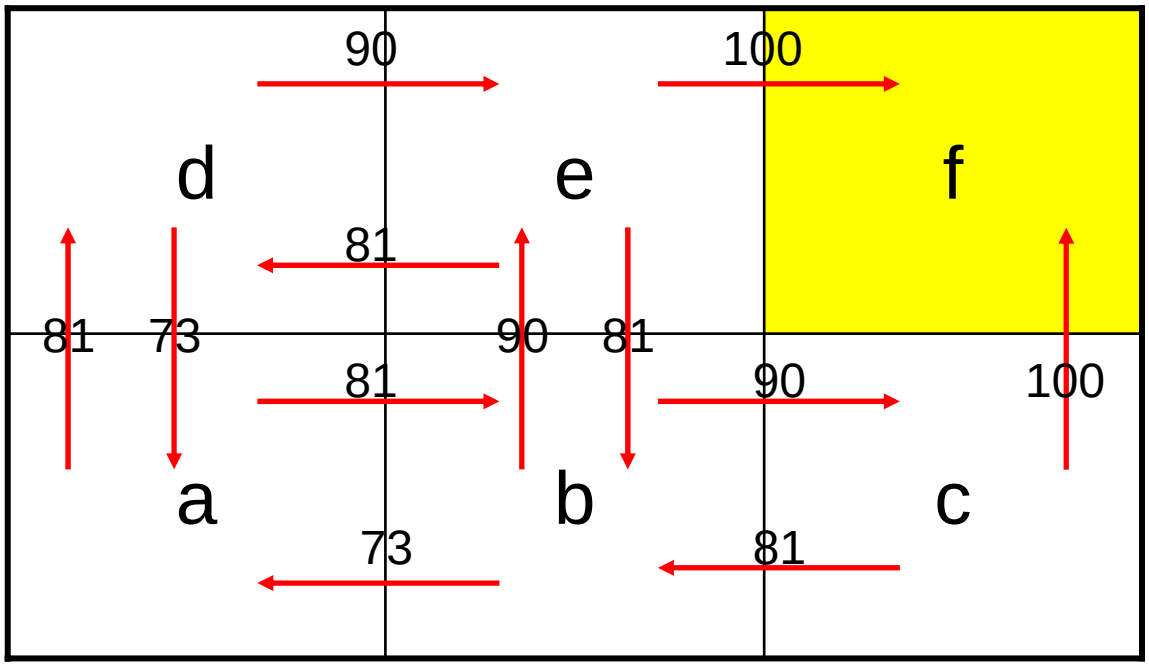
Beispiel – Labyrinth 1



Q - Lernen

Die Werte $Q(s, a)$ konvergieren unter bestimmten Bedingungen gegen die optimalen Aktionswerte $Q*(s, a)$.

Beispiel – Labyrinth 1



$\gamma = 0.9$

Weitere Möglichkeiten

$TD(\lambda)$

Funktionsapproximation (z.B. Neuronale Netze)

11.5 Partially Observable MDP

Partially Observable MDP (POMDP)

- In der realen Welt ist der aktuelle Zustand selten bekannt
- Umgebung ist nur partiell beobachtbar
- Es lassen sich lediglich durch Beobachtungen Wahrscheinlichkeitsaussagen über Zustände machen (Glaubenszustand)
- Dies führt zu POMDPs

Glaubenszustand

b - Glaubenszustand

Wahrscheinlichkeitsverteilung über alle möglichen Zustände

$b(x)$ - Wahrscheinlichkeit für den Zustand x,
falls der Roboter den Glaubenszustand b hat

$x \rightarrow x'$ mit Aktion u und Beobachtung z

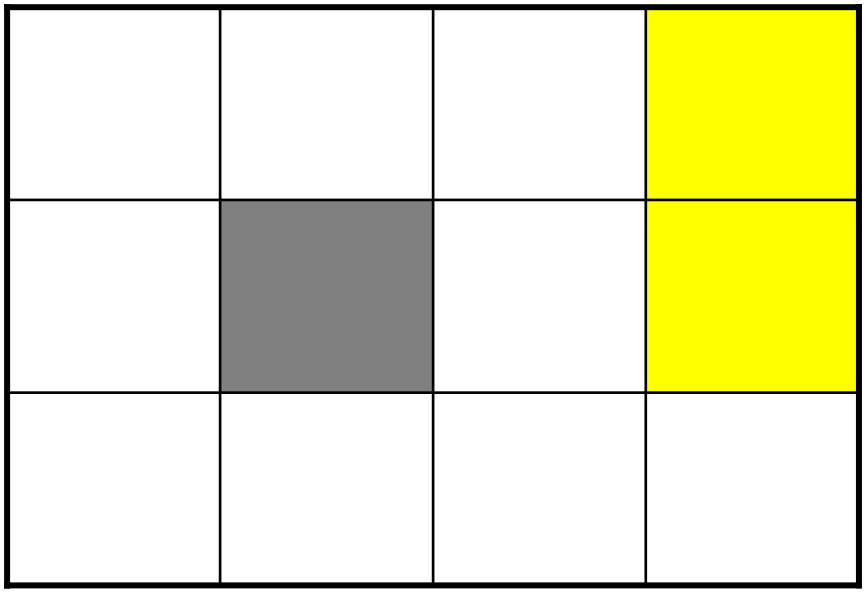
$$b'(x') = \eta \cdot p(z \mid x') \cdot \sum_{x} p(x' \mid x, u) \cdot b(x)$$

$$b' = \text{FORWARD}(b, u, z)$$

$$\pi : b \rightarrow u$$

Taktik hängt von b ab

Beispiel



$$b = \left(\frac{1}{9}, \frac{1}{9}, \frac{1}{9}, \frac{1}{9}, \frac{1}{9}, \frac{1}{9}, \frac{1}{9}, \frac{1}{9}, \frac{1}{9}, 0, 0 \right)$$

Entscheidungszyklus eines Roboters

$$b \rightarrow u = \pi(b)$$

Beobachtung z

$$b' = \mathrm{FORWARD}(b, u, z)$$

Markov Modell für den Glaubenzustandsraum

$$b \xrightarrow{u} b'$$

$\rho(b,u)$

$\rho(b,u)$ - Reward

$$\tau(b,u,b') = p(b'|b,u)$$

Übergangswahrscheinlichkeit

Markov Modell für den Glaubenzustandsraum

- Jetzt beobachtbar im Raum der Glaubenzustände b
- Die Lösung eines partiell beobachtbaren Problems im Raum der Zustände x kann auf die Lösung eines beobachtbaren Problems für den zugehörigen Glaubenszustandsraum reduziert werden
- Aber jetzt ist der Zustandsraum kontinuierlich (und in der Regel hochdimensional)

Übergänge von Glaubenszuständen

$$p(z \mid u,b) = \sum_{x'} p(z \mid u,x',b) \cdot p(x' \mid u,b)$$

$$= \sum_{x'} p(z \mid x') \cdot p(x' \mid u,b)$$

$$= \sum_{x'} p(z \mid x') \cdot \sum_{x} p(x' \mid u,x) \cdot b(x)$$

$$\tau(b,u,b') = p(b' \mid u,b) = \sum_{z} p(b' \mid z,u,b) \cdot p(z \mid u,b)$$

$$= \sum_{z} p(b' \mid z,u,b) \cdot \sum_{x'} p(z \mid x') \cdot \sum_{x} p(x' \mid u,x) \cdot b(x)$$

$$p(b' \mid z,u,b) = \begin{cases} 1 & b' = \text{FORWARD}(b,u,z) \\ 0 & \text{sonst} \end{cases}$$

Belohnung

$$b \xrightarrow{u} b'$$

$\rho(b,u)$

$$\rho(b,u) = \sum_x b(x) \cdot r(x,u)$$

11.6 Beispiel

Beispiel – Zustände – Aktionen – Sensoren

Zustände: x_1, x_2, ziel

Aktionen: u_1, u_2, u_3

$x_1 \xrightarrow{u_1,u_2} \text{ziel}$ $\quad x_2 \xrightarrow{u_1,u_2} \text{ziel}$ $\quad x_1 \xrightarrow{u_3} x_2$ $\quad x_2 \xrightarrow{u_3} x_1$

Sensoren: z_1, z_2

Beispiel – Reward

$$r(x_1, u_1) = -100$$

$$r(x_1, u_2) = 100$$

$$r(x_2, u_1) = 100$$

$$r(x_2, u_2) = -50$$

$$r(x_1, u_3) = -1$$

$$r(x_2, u_3) = -1$$

Beispiel – Übergangswahrscheinlichkeiten

$$p(x_1'|x_1,u_3)=0.2$$

$$p(x_1'|x_2,u_3)=0.8$$

$$p(x_2'|x_1,u_3)=0.8$$

$$p(x_2'|x_2,u_3)=0.2$$

Beispiel – Sensoren

$$p(z_1 \mid x_1) = 0.7 \qquad p(z_2 \mid x_1) = 0.3$$

$$p(z_1 \mid x_2) = 0.3 \qquad p(z_2 \mid x_2) = 0.7$$

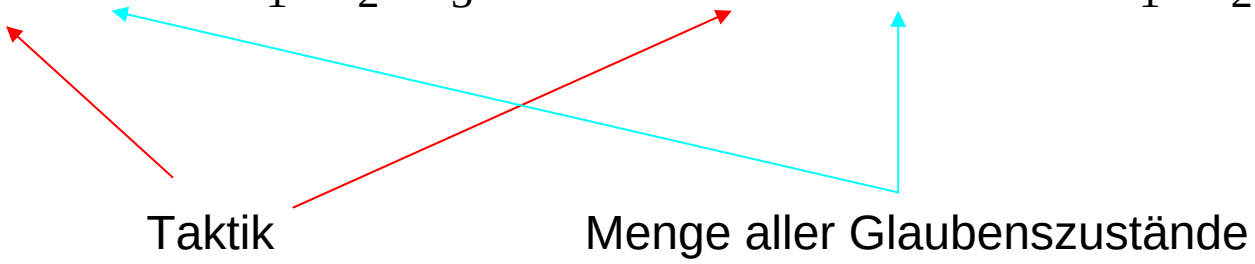
Beispiel – Glaubenszustand

$$b = (p_1, p_2)$$

$$p_1 + p_2 = 1$$

$$p_1 = b(x_1) \qquad p_2 = b(x_2)$$

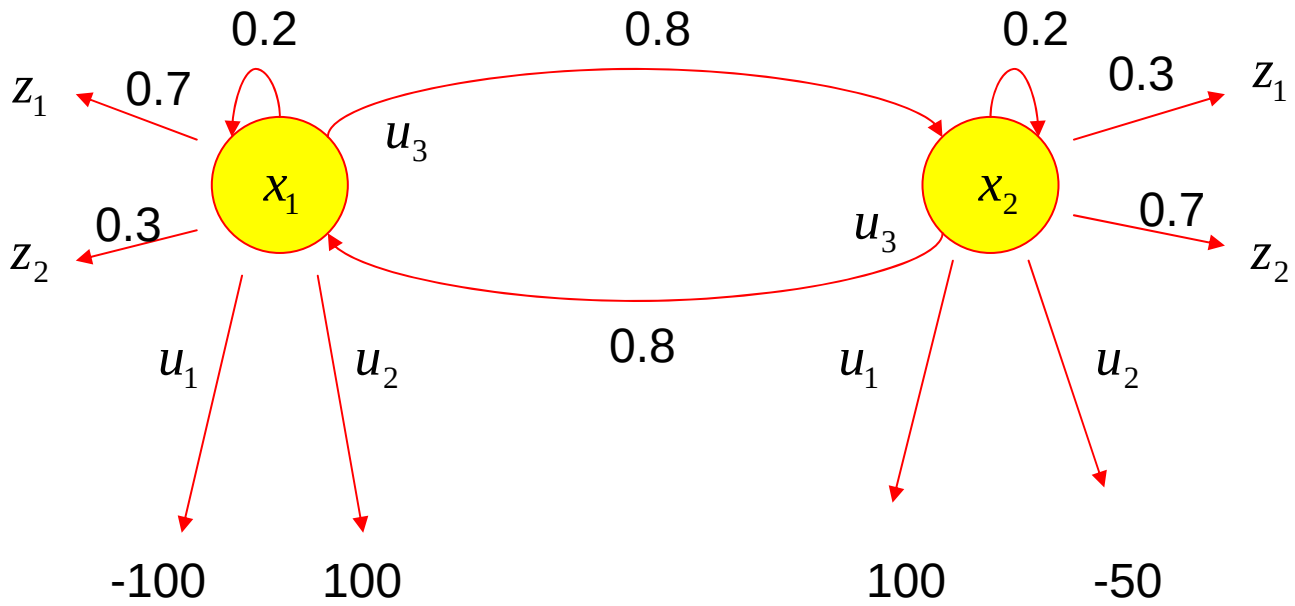
$$\pi : B \rightarrow \{u_1, u_2, u_3\} \qquad \pi : [0,1] \rightarrow \{u_1, u_2, u_3\}$$



Taktik

Menge aller Glaubenszustände

Beispiel


0.2
0.8
0.2
z_1
0.7
u_3
0.3
z_1
x_1
x_2
0.3
u_3
0.7
z_2
z_2
u_1
u_2
0.8
u_1
u_2
-100
100
100
-50

Reward für Übergänge im Glaubenszustandsraum

$$\rho(b,u) = \sum_x b(x) \cdot r(x,u)$$

$$\rho(b,u) = p_1 \cdot r(x_1,u) + p_2 \cdot r(x_2,u)$$

$$\rho(b,u_1) = p_1 \cdot (-100) + p_2 \cdot 100 = -100\,p_1 + 100(1 - p_1)$$

$$\rho(b,u_2) = p_1 \cdot 100 - p_2 \cdot 50 = 100\,p_1 - 50(1 - p_1)$$

$$\rho(b,u_3) = p_1 \cdot (-1) + p_2 \cdot (-1) = -1$$

Eine Aktion wählen – Horizont 1

$$V_1^*(b) = \max_u \rho(b,u) = \max\begin{Bmatrix} -100\,p_1 + 100(1-p_1) \\ 100\,p_1 - 50(1-p_1) \\ -1 \end{Bmatrix}$$

$$= \max\begin{Bmatrix} -100\,p_1 + 100(1-p_1) \\ 100\,p_1 - 50(1-p_1) \end{Bmatrix}$$

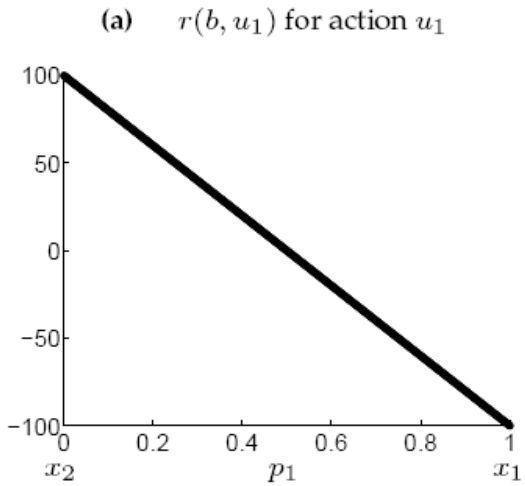
$$-100\,p_1 + 100(1-p_1) = 100\,p_1 - 50(1-p_1) \Rightarrow p_1 = \frac{3}{7}$$

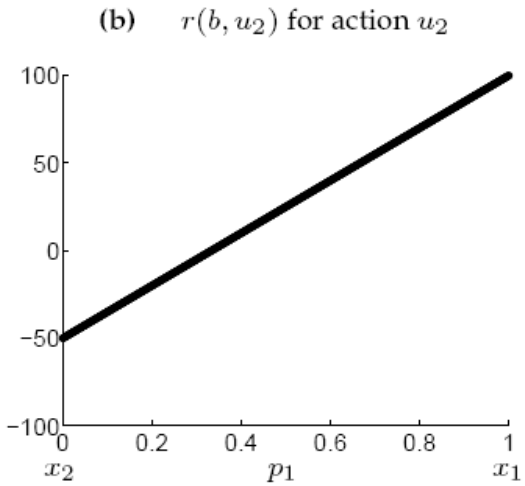
$$\pi_1^*(b) = \operatorname*{argmax}_u \rho(b,u) = \begin{cases} u_1 \text{ falls } p_1 \le \frac{3}{7} \\ u_2 \text{ falls } p_1 > \frac{3}{7} \end{cases}$$

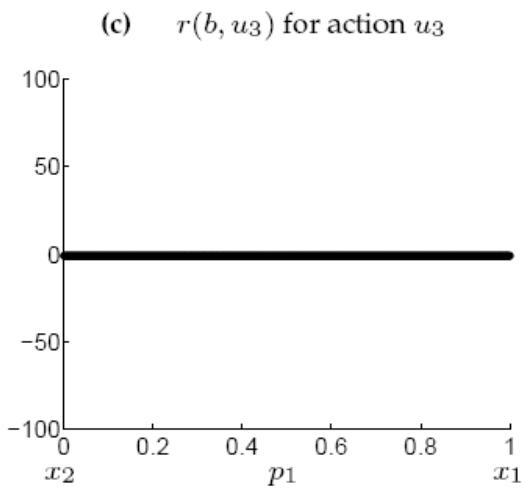
$$\rho(b,u_1) = \frac{100}{7} \approx 14.3,$$

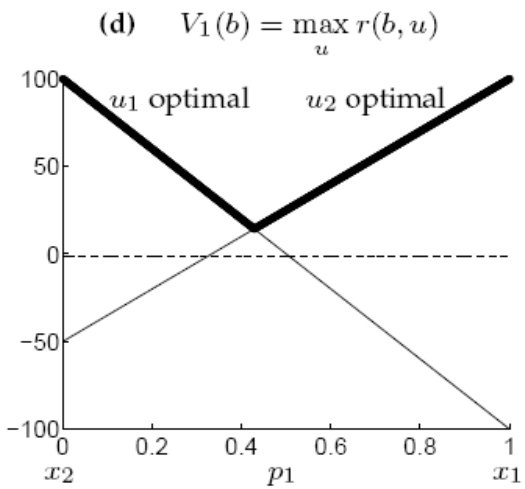
$$b = \left(\frac{3}{7}, \frac{4}{7}\right)$$

3 Aktionen – Horizont 1









Sensoren

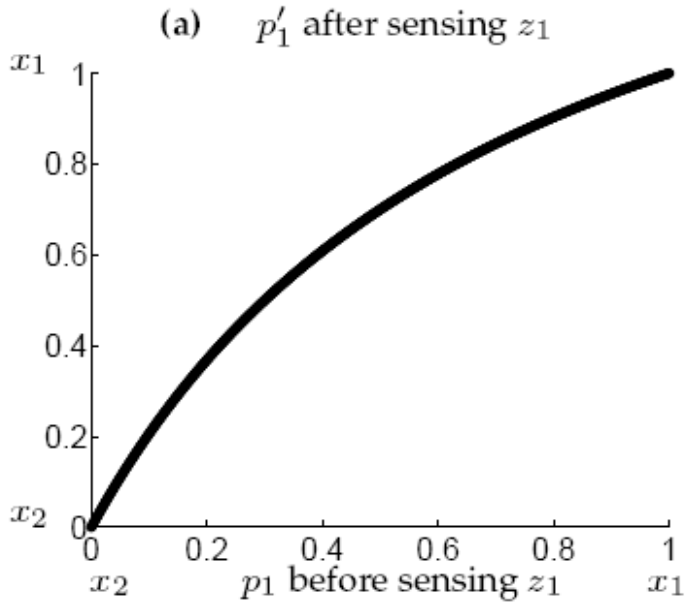
$$p'_1 = p(x_1 \mid z_1) = \frac{p(z_1 \mid x_1) \cdot p(x_1)}{p(z_1)} = \frac{0.7 \cdot p_1}{p(z_1)}$$

$$p'_2 = p(x_2 \mid z_1) = \frac{p(z_1 \mid x_2) \cdot p(x_2)}{p(z_1)} = \frac{0.3 \cdot (1 - p_1)}{p(z_1)}$$

$$p(z_1) = 0.7 \cdot p_1 + 0.3 \cdot (1 - p_1) = 0.4 \cdot p_1 + 0.3$$

$$p'_1 = \frac{0.7 \cdot p_1}{0.4 \cdot p_1 + 0.3} \qquad p'_2 = \frac{0.3 \cdot (1 - p_1)}{0.4 \cdot p_1 + 0.3}$$

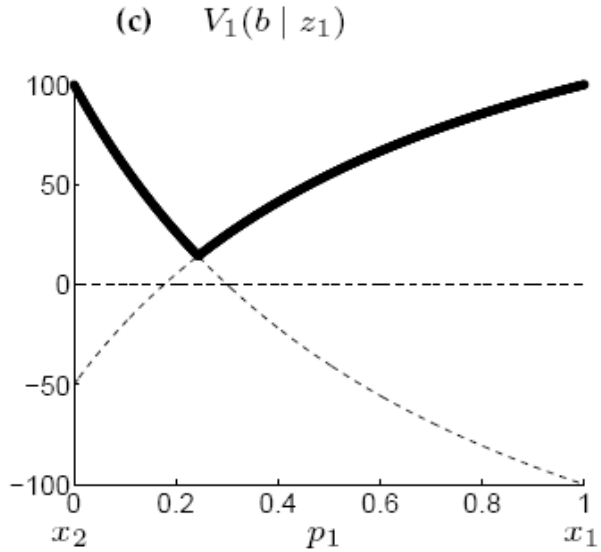
Sensoren

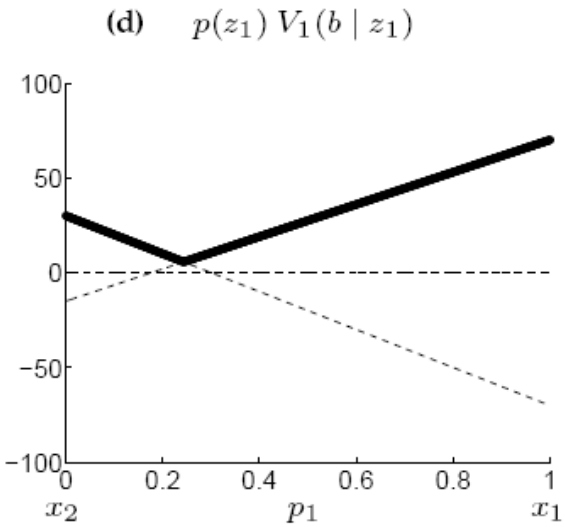

(a) p_1' after sensing z_1
x_1
1
0.8
0.6
0.4
0.2
x_2
0
0
0.2
0.4
0.6
0.8
1
x_2
p_1 before sensing z_1
x_1

Sensoren

$$V_1^*(b \mid z_1) = \max\left\{\begin{array}{l} -100 \cdot \dfrac{0.7 \cdot p_1}{p(z_1)} + 100 \cdot \dfrac{0.3 \cdot (1-p_1)}{p(z_1)} \\ 100 \cdot \dfrac{0.7 \cdot p_1}{p(z_1)} - 50 \cdot \dfrac{0.3 \cdot (1-p_1)}{p(z_1)} \end{array}\right\}$$

$$= \frac{1}{p(z_1)} \max\left\{\begin{array}{l} -70 \cdot p_1 + 30 \cdot (1-p_1) \\ 70 \cdot p_1 - 15 \cdot (1-p_1) \end{array}\right\}$$





$$-70\,p_1 + 30(1-p_1) = 70\,p_1 - 15(1-p_1) \Rightarrow p_1 = \frac{9}{37} \approx 0.25$$

Sensoren

$$p''_1 = p(x_1 \mid z_2) = \frac{p(z_2 \mid x_1) \cdot p(x_1)}{p(z_2)} = \frac{0.3 \cdot p_1}{p(z_2)}$$

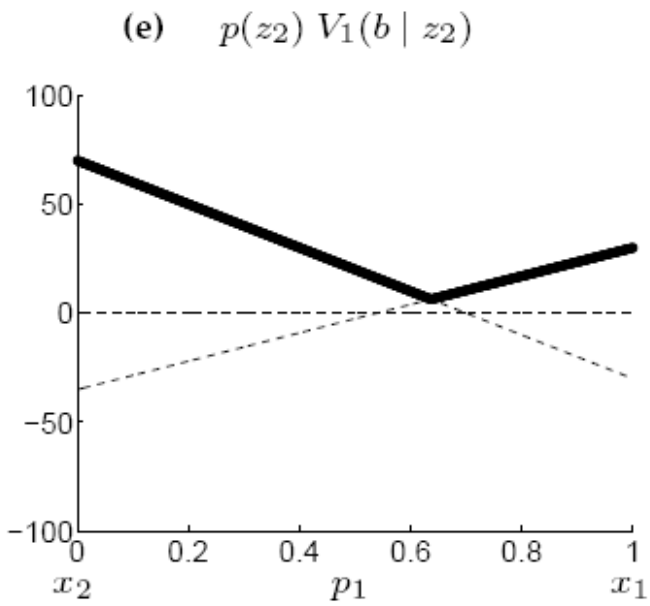
$$p''_2 = p(x_2 \mid z_2) = \frac{p(z_2 \mid x_2) \cdot p(x_2)}{p(z_2)} = \frac{0.7 \cdot (1 - p_1)}{p(z_2)}$$

$$p(z_2) = 0.3 \cdot p_1 + 0.7 \cdot (1 - p_1) = -0.4 \cdot p_1 + 0.7$$

$$p''_1 = \frac{0.3 \cdot p_1}{-0.4 \cdot p_1 + 0.7} \qquad p''_2 = \frac{0.7 \cdot (1 - p_1)}{-0.4 \cdot p_1 + 0.7}$$

Sensoren

$$p(z_2)\cdot V_1^*(b\mid z_2)=\max\left\{\begin{matrix}-30\,p_1+70(1-p_1)\\ 30\,p_1-35(1-p_1)\end{matrix}\right\}$$



Sensoren

$$\tilde{V}_1(b) = E_z[V_1^*(b \mid z)] = \sum_{i=1}^{2} p(z_i) \cdot V_1^*(b \mid z_i)$$

$$\tilde{V}_1(b) = \max\begin{Bmatrix} -70\,\mathrm{p}_1 + 30(1-p_1) \\ 70\,\mathrm{p}_1 - 15(1-p_1) \end{Bmatrix} + \max\begin{Bmatrix} -30\,\mathrm{p}_1 + 70(1-p_1) \\ 30\,\mathrm{p}_1 - 35(1-p_1) \end{Bmatrix}$$

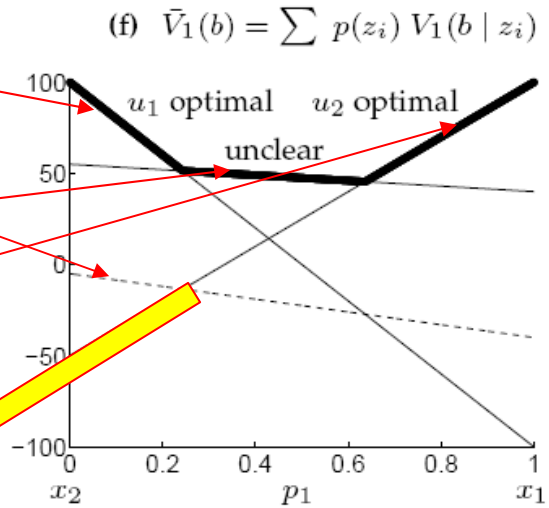
$$\tilde{V}_1(b) = \max\begin{Bmatrix} -70\,p_1 + 30(1-p_1) - 30\,p_1 + 70(1-p_1) \\ -70\,p_1 + 30(1-p_1) + 30\,p_1 + 35(1-p_1) \\ 70\,p_1 - 15(1-p_1) - 30\,p_1 + 70(1-p_1) \\ 70\,p_1 - 15(1-p_1) + 30\,p_1 - 35(1-p_1) \end{Bmatrix}$$

Sensoren

$$\tilde{V}_1(b) = \max \begin{Bmatrix} -70\,p_1 + 30(1-p_1) - 30\,p_1 + 70(1-p_1) \\ -70\,p_1 + 30(1-p_1) + 30\,p_1 + 35(1-p_1) \\ 70\,p_1 - 15(1-p_1) - 30\,p_1 + 70(1-p_1) \\ 70\,p_1 - 15(1-p_1) + 30\,p_1 - 35(1-p_1) \end{Bmatrix}$$

$$\tilde{V}_1(b) = \max \begin{Bmatrix} -100\,p_1 + 100(1-p_1) \\ -40\,p_1 - 5(1-p_1) \\ 40\,p_1 + 55(1-p_1) \\ 100\,p_1 - 50(1-p_1) \end{Bmatrix}$$

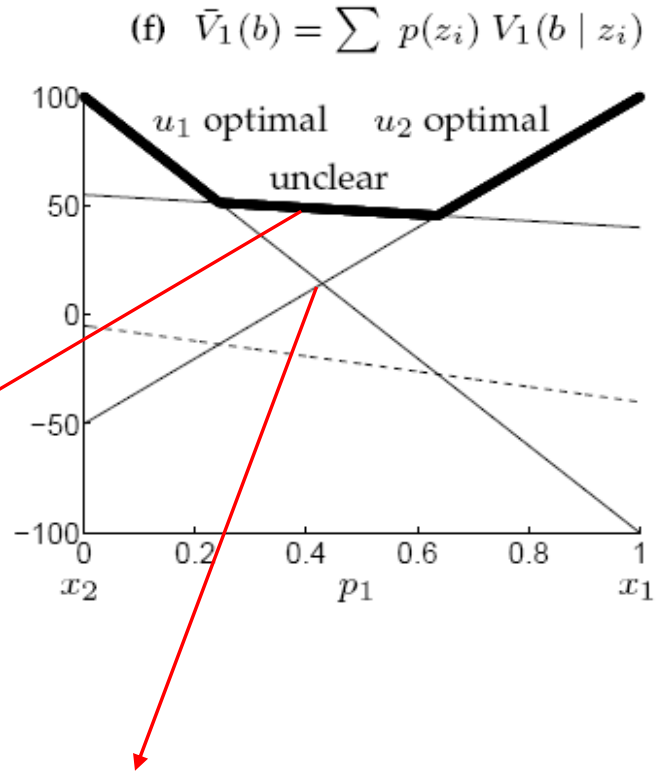
wie ohne Sensoren



nicht erforderlich

Sensoren

$$\tilde{V}_1(b) = \max\begin{Bmatrix} -100\,p_1 + 100(1-p_1) \\ 40\,p_1 + 55(1-p_1) \\ 100\,p_1 - 50(1-p_1) \end{Bmatrix}$$



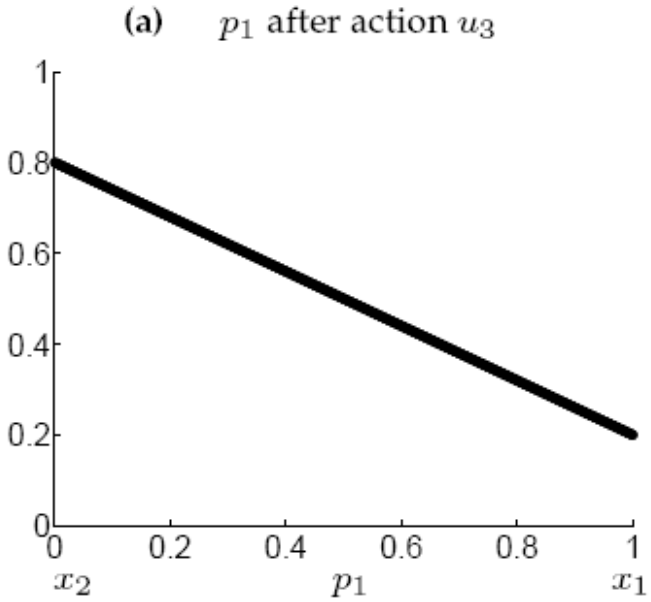
Sensorbeobachtung wichtig

Wert ohne Sensoren

Übergangsaktion

$$p'_1 = \sum_{i=1}^{2} p(x'_1 \mid x_i, u_3) \cdot p_i = 0.2 \cdot p_1 + 0.8(1 - p_1) = 0.8 - 0.6 \cdot p_1$$

$$p'_2 = \sum_{i=1}^{2} p(x'_2 \mid x_i, u_3) \cdot p_i = 0.8 \cdot p_1 + 0.2(1 - p_1) = 0.2 + 0.6 \cdot p_1$$



Übergangsaktion und Sensoren

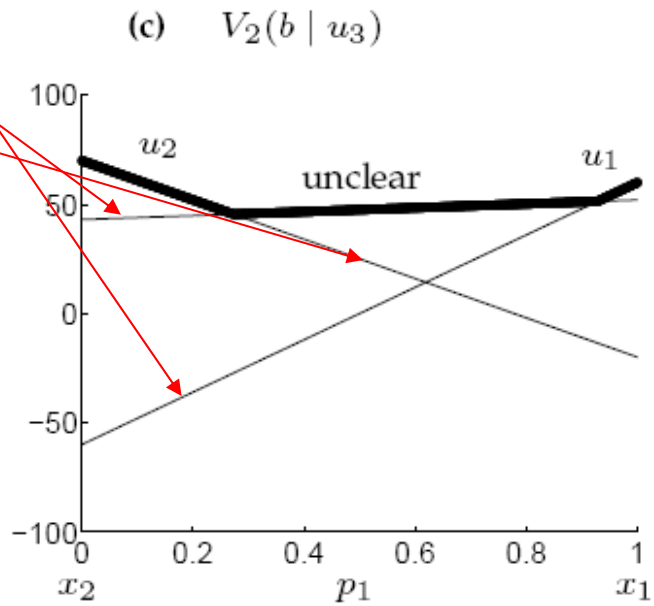
$$\tilde{V}_1(b) = \max\begin{Bmatrix} -100\,p_1 + 100(1-p_1) \\ 40\,p_1 + 55(1-p_1) \\ 100\,p_1 - 50(1-p_1) \end{Bmatrix}$$

$$\tilde{V}_1(b \mid u_3) = \max\begin{Bmatrix} -100(0.8-0.6\,p_1) + 100(0.2+0.6\,p_1) \\ 40(0.8-0.6\,p_1) + 55(0.2+0.6\,p_1) \\ 100(0.8-0.6\,p_1) - 50(0.2+0.6\,p_1) \end{Bmatrix} = V_2^*(b \mid u_3)$$

$$V_2^*(b \mid u_3) = \max\begin{Bmatrix} 60\,p_1 - 60(1-p_1) \\ 52\,p_1 + 43(1-p_1) \\ -20\,p_1 + 70(1-p_1) \end{Bmatrix}$$

Übergangsaktion und Sensoren

$$V_2^*(b \mid u_3) = \max\begin{Bmatrix} 60\,p_1 - 60(1-p_1) \\ 52\,p_1 + 43(1-p_1) \\ -20\,p_1 + 70(1-p_1) \end{Bmatrix}$$



Horizont 2

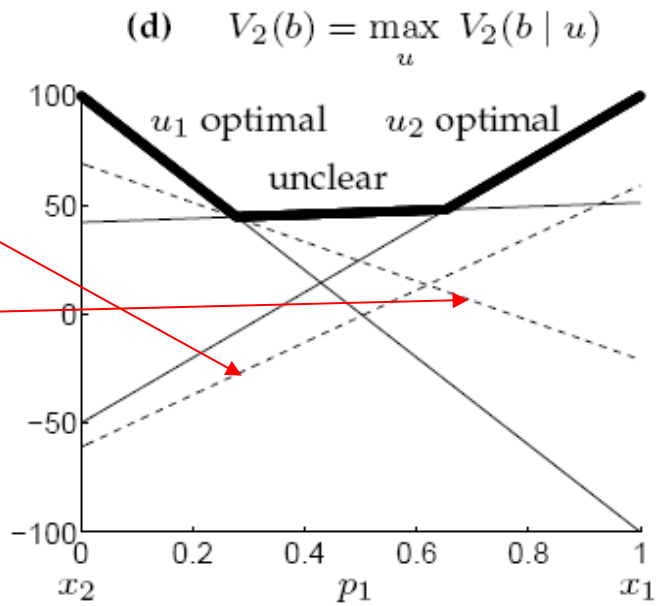
$$V_2^*(b \mid u_3) = \max \begin{Bmatrix} 60p_1 - 60(1-p_1) \\ 52p_1 + 43(1-p_1) \\ -20p_1 + 70(1-p_1) \end{Bmatrix}$$

$$60p_1 - 60(1-p_1) - 1 = 59p_1 - 61(1-p_1)$$

$$V_2^*(b) = \max \begin{Bmatrix} -100p_1 + 100(1-p_1) \\ 100p_1 - 50(1-p_1) \\ 59p_1 - 61(1-p_1) \\ 51p_1 + 42(1-p_1) \\ -21p_1 + 69(1-p_1) \end{Bmatrix}$$

Horizont 2

$$V_2^*(b) = \max \begin{cases} -100\,p_1 + 100(1-p_1) \\ 100\,p_1 - 50(1-p_1) \\ 59\,p_1 - 61(1-p_1) \\ 51\,p_1 + 42(1-p_1) \\ -21\,p_1 + 69(1-p_1) \end{cases}$$



$$V_2^*(b) = \max \begin{cases} -100\,p_1 + 100(1-p_1) \\ 100\,p_1 - 50(1-p_1) \\ 51\,p_1 + 42(1-p_1) \end{cases}$$

Value function (T=10, T=20)

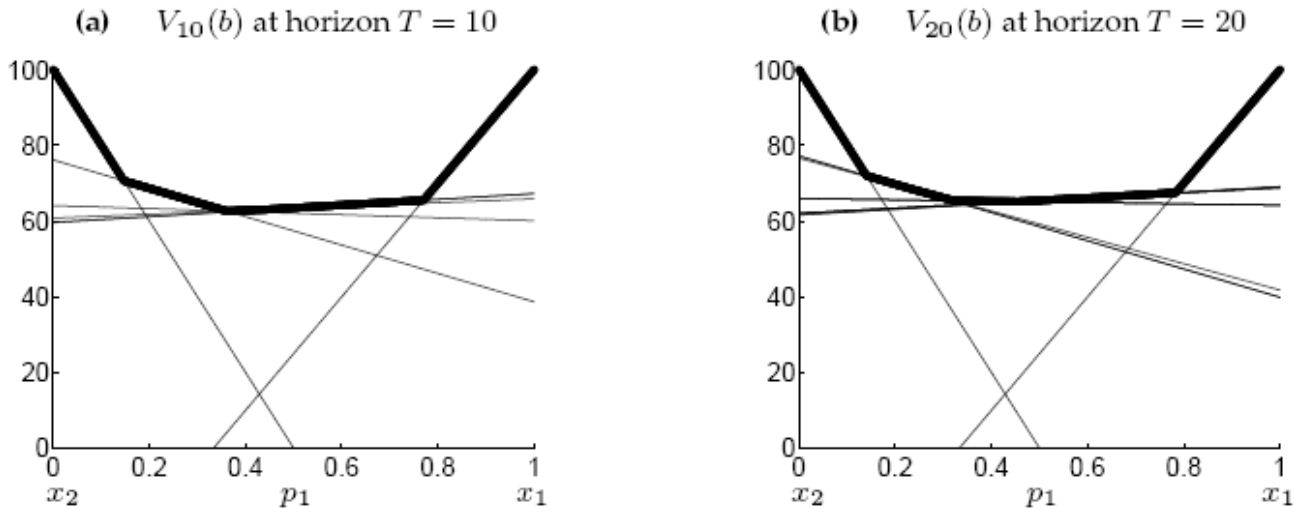


Figure 15.5 The value function V for horizons $T = 10$ and $T = 20$. Note that the vertical axis in these plots differs in scale from previous depictions of value functions.

POMDP – Algorithmus

$$V_{T-1}^{*}(b) \Rightarrow V_{T}^{*}(b)$$

Eingabe: Horizont T

Ausgabe: $\left(v_1^k, \cdots v_N^k\right) \quad k = 1,2,\cdots$

$$V_T^*(b) = \max_k \sum_{i=1}^{N} v_i^k \cdot p_i$$