

# Belöningsbaserad Inläring

## Reinforcement Learning

- 1 Definition av problemet
  - Inlärningsituationen
  - Belöningens roll
  - Förenklade antaganden
  - Centrala begrepp
- 2 Känd omgivning
  - Bellmans ekvation
  - Lösningmetoder
- 3 Okänd omgivning
  - Monte-Carlo metoden
  - Temporal-Difference
  - Q-Learning
  - Sarsa-Learning
- 4 Förbättringar
  - Nyttan av att göra fel
  - Eligibility Trace

- 1 Definition av problemet
  - Inlärningsituationen
  - Belöningens roll
  - Förenklade antaganden
  - Centrala begrepp
- 2 Känd omgivning
  - Bellmans ekvation
  - Lösningmetoder
- 3 Okänd omgivning
  - Monte-Carlo metoden
  - Temporal-Difference
  - Q-Learning
  - Sarsa-Learning
- 4 Förbättringar
  - Nyttan av att göra fel
  - Eligibility Trace

# Belöningsbaserad inläring

## Reinforcement Learning

Inläring av ett beteende utan tillgång till facit.

# Belöningsbaserad inläring

## Reinforcement Learning

Inläring av ett beteende utan tillgång till facit.

- En **belöning** ger information om hur bra det går

# Belöningsbaserad inlärning

## Reinforcement Learning

Inlärning av ett beteende utan tillgång till facit.

- En **belöning** ger information om hur bra det går
- Belöningen kommer inte *samtidigt* som man gör något bra  
**Temporal credit assignment**

# Belöningsbaserad inläring

## Reinforcement Learning

Inläring av ett beteende utan tillgång till facit.

- En **belöning** ger information om hur bra det går
- Belöningen kommer inte *samtidigt* som man gör något bra  
**Temporal credit assignment**
- Belöningen anger inte *vad* som var bra  
**Structural credit assignment**

# Modell för inlärningsituationen

## Modell för inlärningsituationen

- En **agent** interagerar med sin **omgivning**

## Modell för inlärningsituationen

- En **agent** interagerar med sin **omgivning**
- Agenten utför **handlingar**

## Modell för inlärningsituationen

- En **agent** interagerar med sin **omgivning**
- Agenten utför **handlingar**
- Handlingarna påverkar omgivningens **tillstånd**

## Modell för inlärningsituationen

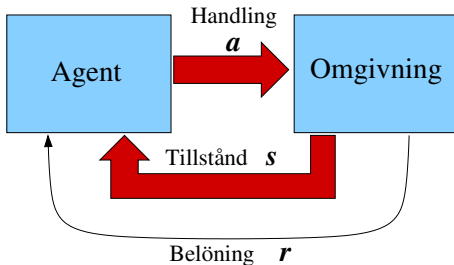
- En **agent** interagerar med sin **omgivning**
- Agenten utför **handlingar**
- Handlingarna påverkar omgivningens **tillstånd**
- Agenten observerar omgivningens tillstånd

## Modell för inlärningsituationen

- En **agent** interagerar med sin **omgivning**
- Agenten utför **handlingar**
- Handlingarna påverkar omgivningens **tillstånd**
- Agenten observerar omgivningens tillstånd
- Agenten får även en **belöning** från omgivningen

## Modell för inlärningssituationen

- En **agent** interagerar med sin **omgivning**
- Agenten utför **handlingar**
- Handlingarna påverkar omgivningens **tillstånd**
- Agenten observerar omgivningens tillstånd
- Agenten får även en **belöning** från omgivningen



## Uppgiften för agenten

Hitta ett beteende som maximerar den *totala* belöningen.

## Uppgiften för agenten

Hitta ett beteende som maximerar den *totala* belöningen.

Hur lång framtid ska vi ta hänsyn till?

## Uppgiften för agenten

Hitta ett beteende som maximerar den *totala* belöningen.

Hur lång framtid ska vi ta hänsyn till?

- Begränsad tidshorisont

$$\max \left[ \sum_{t=0}^h r_t \right]$$

## Uppgiften för agenten

Hitta ett beteende som maximerar den *totala* belöningen.

Hur lång framtid ska vi ta hänsyn till?

- Begränsad tidshorisont

$$\max \left[ \sum_{t=0}^h r_t \right]$$

- Oändlig tidshorisont

$$\max \left[ \sum_{t=0}^{\infty} \gamma^t r_t \right]$$

Kräver nedskrivning av framtida belöningar ( $0 < \gamma < 1$ )

# Belöningsfunktionen

Belöningsfunktionen styr vilken uppgift som ska lösas

# Belöningsfunktionen

Belöningsfunktionen styr vilken uppgift som ska lösas

- Spel (Schack, Backgammon)

# Belöningsfunktionen

Belöningsfunktionen styr vilken uppgift som ska lösas

- Spel (Schack, Backgammon)  
Belöning bara i slutet: +1 vid vinst, -1 vid förlust

# Belöningsfunktionen

Belöningsfunktionen styr vilken uppgift som ska lösas

- Spel (Schack, Backgammon)  
Belöning bara i slutet: +1 vid vinst, -1 vid förlust
- Undvika misstag (cykla, ramla, ...)

# Belöningsfunktionen

Belöningsfunktionen styr vilken uppgift som ska lösas

- Spel (Schack, Backgammon)  
Belöning bara i slutet: +1 vid vinst, -1 vid förlust
- Undvika misstag (cykla, ramla, ...)  
Belöning -1 i slutet (när man misslyckas)

# Belöningsfunktionen

Belöningsfunktionen styr vilken uppgift som ska lösas

- Spel (Schack, Backgammon)  
Belöning bara i slutet: +1 vid vinst, -1 vid förlust
- Undvika misstag (cykla, ramla, ...)  
Belöning -1 i slutet (när man misslyckas)
- Hitta kort/snabb/billig väg till målet

# Belöningsfunktionen

Belöningsfunktionen styr vilken uppgift som ska lösas

- Spel (Schack, Backgammon)  
Belöning bara i slutet: +1 vid vinst, -1 vid förlust
- Undvika misstag (cykla, ramla, ...)  
Belöning -1 i slutet (när man misslyckas)
- Hitta kort/snabb/billig väg till målet  
Belöning -1 hela tiden

# Förenklade antaganden

## Förenklade antaganden

- Diskret tid

## Förenklade antaganden

- Diskret tid
- Ändligt antal handlingar  $a_i$

$$a_i \in a_1, a_2, a_3, \dots, a_n$$

## Förenklade antaganden

- Diskret tid
- Ändligt antal handlingar  $a_i$

$$a_i \in a_1, a_2, a_3, \dots, a_n$$

- Ändligt antal tillstånd  $s_i$

$$s_i \in s_1, s_2, s_3, \dots, s_m$$

## Förenklade antaganden

- Diskret tid
- Ändligt antal handlingar  $a_i$

$$a_i \in a_1, a_2, a_3, \dots, a_n$$

- Ändligt antal tillstånd  $s_i$

$$s_i \in s_1, s_2, s_3, \dots, s_m$$

- Omgivningen är en konstant MDP  
(*Markov Decision Process*)

## Förenklade antaganden

- Diskret tid
- Ändligt antal handlingar  $a_i$

$$a_i \in a_1, a_2, a_3, \dots, a_n$$

- Ändligt antal tillstånd  $s_i$

$$s_i \in s_1, s_2, s_3, \dots, s_m$$

- Omgivningen är en konstant MDP  
(*Markov Decision Process*) Belöningen och nästa tillstånd beror bara på  $s$ ,  $a$  och slumpen

## Förenklade antaganden

- Diskret tid
- Ändligt antal handlingar  $a_i$

$$a_i \in a_1, a_2, a_3, \dots, a_n$$

- Ändligt antal tillstånd  $s_i$

$$s_i \in s_1, s_2, s_3, \dots, s_m$$

- Omgivningen är en konstant MDP  
(*Markov Decision Process*) Belöningen och nästa tillstånd beror bara på  $s$ ,  $a$  och slumpen
- Deterministisk eller icke-deterministisk omgivning

Definition av problemet

○○○○●○○

Känd omgivning

○○○

Okänd omgivning

○○○○○

Förbättringar

○○

Centrala begrepp

# Agentens interna representation

# Agentens interna representation

- *Policy*

Den handling agenten väljer i varje tillstånd

$$\pi(s) \mapsto a$$

# Agentens interna representation

- *Policy*

Den handling agenten väljer i varje tillstånd

$$\pi(s) \mapsto a$$

- *Värdefunktionen*

Förväntad framtida belöning från  $s$  när man följer policy  $\pi$

$$V^\pi(s) \mapsto \mathfrak{R}$$

## Klassiskt modellproblem: *Grid World*

## Klassiskt modellproblem: *Grid World*

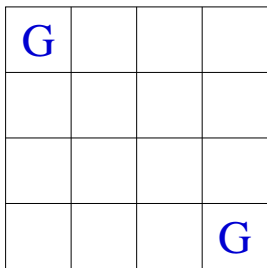
- Varje **tillstånd** representeras av en plats i ett rutnät

## Klassiskt modellproblem: *Grid World*

- Varje **tillstånd** representeras av en plats i ett rutnät
- Agenten **handlar** genom att gå till andra rutor

## Klassiskt modellproblem: *Grid World*

- Varje **tillstånd** representeras av en plats i ett rutnät
- Agenten **handlar** genom att gå till andra rutor



Trivial labyrint

## Klassiskt modellproblem: *Grid World*

- Varje **tillstånd** representeras av en plats i ett rutnät
- Agenten **handlar** genom att gå till andra rutor

G			
			G

Trivial labyrint

**Belöning:**  $-1$  i varje steg  
tills man når något av  
måltillstånden ( $G$ )

Värdet av ett tillstånd beror av aktuell policy.

Värdet av ett tillstånd beror av aktuell policy.

0	-1	-2	-3
-1	-2	-3	-2
-2	-3	-2	-1
-3	-2	-1	0

$V$  vid  
optimal policy

Värdet av ett tillstånd beror av aktuell policy.

0	-1	-2	-3
-1	-2	-3	-2
-2	-3	-2	-1
-3	-2	-1	0

$V$  vid  
optimal policy

0	-14	-20	-22
-14	-18	-22	-20
-20	-22	-18	-14
-22	-20	-14	0

$V$  vid  
slumpmässig policy

- 1 Definition av problemet
  - Inlärningssituationen
  - Belöningens roll
  - Förenklade antaganden
  - Centrala begrepp
- 2 Känd omgivning
  - Bellmans ekvation
  - Lösningmetoder
- 3 Okänd omgivning
  - Monte-Carlo metoden
  - Temporal-Difference
  - Q-Learning
  - Sarsa-Learning
- 4 Förbättringar
  - Nyttan av att göra fel
  - Eligibility Trace

## Modell av omgivningen

## Modell av omgivningen

- Var hamnar vi?

$$\delta(s, a) \mapsto s'$$

## Modell av omgivningen

- Var hamnar vi?

$$\delta(s, a) \mapsto s'$$

- Hur mycket belöning får vi?

$$r(s, a) \mapsto \mathfrak{R}$$

## Modell av omgivningen

- Var hamnar vi?

$$\delta(s, a) \mapsto s'$$

- Hur mycket belöning får vi?

$$r(s, a) \mapsto \mathcal{R}$$

Värdet av olika tillstånd hänger ihop

## Modell av omgivningen

- Var hamnar vi?

$$\delta(s, a) \mapsto s'$$

- Hur mycket belöning får vi?

$$r(s, a) \mapsto \mathfrak{R}$$

Värdet av olika tillstånd hänger ihop

*Bellmans ekvation:*

$$V^\pi(s) = r(s, \pi(s)) + \gamma \cdot V^\pi(\delta(s, \pi(s)))$$

Kan man lösa *Bellmans ekvation*?

$$V^\pi(s) = r(s, \pi(s)) + \gamma \cdot V^\pi(\delta(s, \pi(s)))$$

Kan man lösa *Bellmans ekvation*?

$$V^\pi(s) = r(s, \pi(s)) + \gamma \cdot V^\pi(\delta(s, \pi(s)))$$

- Direkt lösning (linjärt ekvationssystem)

Kan man lösa *Bellmans ekvation*?

$$V^\pi(s) = r(s, \pi(s)) + \gamma \cdot V^\pi(\delta(s, \pi(s)))$$

- Direkt lösning (linjärt ekvationssystem)
- Iterativt (*value iteration*)

$$V_{k+1}^\pi(s) \leftarrow r(s, \pi(s)) + \gamma \cdot V_k^\pi(\delta(s, \pi(s)))$$

Hur får man fram en **optimal policy**  $\pi^*$ ?

Hur får man fram en **optimal policy**  $\pi^*$ ?

Lätt om man visste den optimala värdefunktionen  $V^*$ :

$$\pi^*(s) = \underset{a}{\operatorname{argmax}} (r(s, a) + \gamma \cdot V^*(\delta(s, a)))$$

Hur får man fram en **optimal policy**  $\pi^*$ ?

Lätt om man visste den optimala värdefunktionen  $V^*$ :

$$\pi^*(s) = \operatorname{argmax}_a (r(s, a) + \gamma \cdot V^*(\delta(s, a)))$$

Optimala varianten av Bellmans ekvation

$$V^*(s) = \max_a (r(s, a) + \gamma \cdot V^*(\delta(s, a)))$$

Hur får man fram en **optimal policy**  $\pi^*$ ?

Lätt om man visste den optimala värdefunktionen  $V^*$ :

$$\pi^*(s) = \operatorname{argmax}_a (r(s, a) + \gamma \cdot V^*(\delta(s, a)))$$

Optimala varianten av Bellmans ekvation

$$V^*(s) = \max_a (r(s, a) + \gamma \cdot V^*(\delta(s, a)))$$

Svår att lösa

Hur får man fram en **optimal policy**  $\pi^*$ ?

Lätt om man visste den optimala värdefunktionen  $V^*$ :

$$\pi^*(s) = \operatorname{argmax}_a (r(s, a) + \gamma \cdot V^*(\delta(s, a)))$$

Optimala varianten av Bellmans ekvation

$$V^*(s) = \max_a (r(s, a) + \gamma \cdot V^*(\delta(s, a)))$$

Svår att lösa

Policy iteration:

Iterera policy och värdeberäkningarna växelvis

- 1 Definition av problemet
  - Inlärningssituationen
  - Belöningens roll
  - Förenklade antaganden
  - Centrala begrepp
- 2 Känd omgivning
  - Bellmans ekvation
  - Lösningmetoder
- 3 Okänd omgivning
  - Monte-Carlo metoden
  - Temporal-Difference
  - Q-Learning
  - Sarsa-Learning
- 4 Förbättringar
  - Nyttan av att göra fel
  - Eligibility Trace

Vanligen är  $r(s, a)$  och  $\delta(s, a)$  inte kända av agenten

Vanligen är  $r(s, a)$  och  $\delta(s, a)$  inte kända av agenten

$V^\pi$  måste skattas genom **erfarenhet**

Vanligen är  $r(s, a)$  och  $\delta(s, a)$  inte kända av agenten

$V^\pi$  måste skattas genom **erfarenhet**

*Monte-Carlo tekniken*

Vanligen är  $r(s, a)$  och  $\delta(s, a)$  inte kända av agenten

$V^\pi$  måste skattas genom **erfarenhet**

*Monte-Carlo tekniken*

- Starta från slumpmässig  $s$

Vanligen är  $r(s, a)$  och  $\delta(s, a)$  inte kända av agenten

$V^\pi$  måste skattas genom **erfarenhet**

### *Monte-Carlo tekniken*

- Starta från slumpmässig  $s$
- Följ  $\pi$ , lagra belöningar och  $s_t$

Vanligen är  $r(s, a)$  och  $\delta(s, a)$  inte kända av agenten

$V^\pi$  måste skattas genom **erfarenhet**

### *Monte-Carlo tekniken*

- Starta från slumpmässig  $s$
- Följ  $\pi$ , lagra belöningar och  $s_t$
- När man nått målet, uppdatera  $V^\pi(s)$ -skattningen för alla besökta tillstånd med den framtida belöning man verkligen fick

Vanligen är  $r(s, a)$  och  $\delta(s, a)$  inte kända av agenten

$V^\pi$  måste skattas genom **erfarenhet**

### *Monte-Carlo tekniken*

- Starta från slumpmässig  $s$
- Följ  $\pi$ , lagra belöningar och  $s_t$
- När man nått målet, uppdatera  $V^\pi(s)$ -skattningen för alla besökta tillstånd med den framtida belöning man verkligen fick

Mycket långsam konvergens

Definition av problemet  
○○○○○○○

Känd omgivning  
○○○

Okänd omgivning  
●○○○○

Förbättringar  
○○

Temporal-Difference

# Temporal Difference

# Temporal Difference

Idén bakom Temporal Difference:

Utnyttja att finns två skattningar för värdet av ett tillstånd:  
*före och efter*

# Temporal Difference

Idén bakom Temporal Difference:

Utnyttja att finns två skattningar för värdet av ett tillstånd:  
*före och efter*

- Vad man tror **innan** man handlat

$$V^\pi(s_t)$$

# Temporal Difference

## Idén bakom Temporal Difference:

Utnyttja att finns två skattningar för värdet av ett tillstånd:  
*före och efter*

- Vad man tror **innan** man handlat

$$V^\pi(s_t)$$

- Vad man tror **efter** man handlat

$$r_{t+1} + \gamma \cdot V^\pi(s_{t+1})$$

Viktig observation:

Den andra skattningen är bättre!

Viktig observation:

Den andra skattningen är bättre!

Uppdatera skattningen av värdet i riktning mot den bättre

## Viktig observation:

Den andra skattningen är bättre!

Uppdatera skattningen av värdet i riktning mot den bättre

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

## Viktig observation:

Den andra skattningen är bättre!

Uppdatera skattningen av värdet i riktning mot den bättre

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

Mått på **överraskningen** / **besvikelsen**

## Viktig observation:

Den andra skattningen är bättre!

Uppdatera skattningen av värdet i riktning mot den bättre

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

Mått på **överraskningen** / **besvikelsen**Lär sig *betydligt snabbare* än Monte-Carlo tekniken

Problem:

Även om man har skattat  $V$  bra kan man inte räkna ut  $\pi$   
eftersom agenten inte känner  $\delta$  och  $r$ !

Problem:

Även om man har skattat  $V$  bra kan man inte räkna ut  $\pi$  eftersom agenten inte känner  $\delta$  och  $r$ !

Trick:

Skatta  $Q(s, a)$  istället för  $V(s)$

Problem:

Även om man har skattat  $V$  bra kan man inte räkna ut  $\pi$  eftersom agenten inte känner  $\delta$  och  $r$ !

Trick:

Skatta  $Q(s, a)$  istället för  $V(s)$

$Q(s, a)$ : Förväntad total belöning när man gör  $a$  från  $s$ .

Problem:

Även om man har skattat  $V$  bra kan man inte räkna ut  $\pi$  eftersom agenten inte känner  $\delta$  och  $r$ !

Trick:

Skatta  $Q(s, a)$  istället för  $V(s)$

$Q(s, a)$ : Förväntad total belöning när man gör  $a$  från  $s$ .

$$\pi(s) = \operatorname{argmax}_a Q(s, a)$$

$$V^*(s) = \max_a Q^*(s, a)$$

Hur kan vi lära oss  $Q$ ?

Hur kan vi lära oss  $Q$ ?

Även  $Q$ -funktionen kan läras med Temporal-Difference

Hur kan vi lära oss  $Q$ ?

Även  $Q$ -funktionen kan läras med Temporal-Difference

$$Q(s, a) \leftarrow Q(s, a) + \eta \left[ r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right]$$

$s'$  är nästa tillstånd.

Hur kan vi lära oss  $Q$ ?

Även  $Q$ -funktionen kan läras med Temporal-Difference

$$Q(s, a) \leftarrow Q(s, a) + \eta \left[ r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right]$$

$s'$  är nästa tillstånd.

Litet problem: max-operationen kräver att man söker igenom alla tänkbara handlingar i nästa steg.

## SARSA-learning

## SARSA-learning

Nästan samma som Q-learning, men man låter **aktuell policy** bestämma  $a'$ :

$$Q(s, a) \leftarrow Q(s, a) + \eta [r + \gamma Q(s', a') - Q(s, a)]$$

## SARSA-learning

Nästan samma som Q-learning, men man låter **aktuell policy** bestämma  $a'$ :

$$Q(s, a) \leftarrow Q(s, a) + \eta [r + \gamma Q(s', a') - Q(s, a)]$$

Har fått sitt namn av att "erfarenhets-tuplerna" har formen

$$\langle s, a, r, s', a' \rangle$$

- 1 Definition av problemet
  - Inlärningssituationen
  - Belöningens roll
  - Förenklade antaganden
  - Centrala begrepp
- 2 Känd omgivning
  - Bellmans ekvation
  - Lösningmetoder
- 3 Okänd omgivning
  - Monte-Carlo metoden
  - Temporal-Difference
  - Q-Learning
  - Sarsa-Learning
- 4 Förbättringar
  - Nyttan av att göra fel
  - Eligibility Trace

Vad gör man när...

Vad gör man när...

- Omgivningen är inte fullt observerbar

Vad gör man när...

- Omgivningen är inte fullt observerbar
- Tillstånden är alltför många

Vad gör man när...

- Omgivningen är inte fullt observerbar
- Tillstånden är alltför många
- Tillstånden är inte diskreta

Vad gör man när...

- Omgivningen är inte fullt observerbar
- Tillstånden är alltför många
- Tillstånden är inte diskreta
- Agenten handlar i kontinuerlig tid

## Exploration–Exploitation dilemmat

Om man följer en policy baserad på aktuell skattning av  $Q$  konvergerar  $Q$  inte säkert mot  $Q^*$

## Exploration–Exploitation dilemmat

Om man följer en policy baserad på aktuell skattning av  $Q$  konvergerar  $Q$  inte säkert mot  $Q^*$

Enkel lösning:

Använd en policy som har viss sannolikhet att ”göra fel”

## Exploration–Exploitation dilemmat

Om man följer en policy baserad på aktuell skattning av  $Q$  konvergerar  $Q$  inte säkert mot  $Q^*$

Enkel lösning:

Använd en policy som har viss sannolikhet att ”göra fel”

- $\epsilon$ -greedy

Gör ibland (med sannolikheten  $\epsilon$ ) en slumpmässig handling istället för den som verkar bäst (giriga)

## Exploration–Exploitation dilemmat

Om man följer en policy baserad på aktuell skattning av  $Q$  konvergerar  $Q$  inte säkert mot  $Q^*$

Enkel lösning:

Använd en policy som har viss sannolikhet att ”göra fel”

- **$\epsilon$ -greedy**  
Gör ibland (med sannolikheten  $\epsilon$ ) en slumpmässig handling istället för den som verkar bäst (giriga)
- **Softmax**  
Vikta sannolikheten att göra olika handlingar med hur bra de verkar

## Ytterligare uppsnabbning

## Ytterligare uppsnabbning

Idé: TD-uppdateringarna kan utnyttjas till att förbättra skattningen även av tillstånd där vi varit tidigare.

## Ytterligare uppsnabbning

Idé: TD-uppdateringarna kan utnyttjas till att förbättra skattningen även av tillstånd där vi varit tidigare.

$$\forall s, a : Q(s, a) \leftarrow Q(s, a) + \eta [r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \cdot e$$

$e$  är ett kvardröjande spår (**eligibility trace**) som beskriver hur länge sedan man var i  $s$  och gjorde  $a$ .

## Ytterligare uppsnabbning

Idé: TD-uppdateringarna kan utnyttjas till att förbättra skattningen även av tillstånd där vi varit tidigare.

$$\forall s, a : Q(s, a) \leftarrow Q(s, a) + \eta [r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \cdot e$$

$e$  är ett kvardröjande spår (**eligibility trace**) som beskriver hur länge sedan man var i  $s$  och gjorde  $a$ .

Kallas ofta **TD( $\lambda$ )** där  $\lambda$  är tidskonstanten för avklingningen av spåret