Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Mesterséges Intelligencia és Rendszertervezés Tanszék

Mesterséges intelligencia előadássorozat

Az előadás diái az AIMA könyvre épülve (http://aima.cs.berkeley.edu) készültek a University of California, Berkeley mesterséges intelligencia kurzusának anyagainak felhasználásával (http://ai.berkeley.edu).

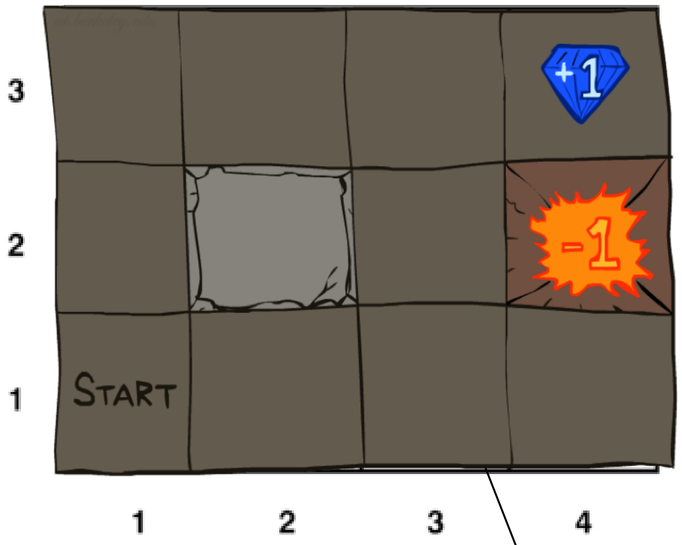
These slides are based on the AIMA book (http://aima.cs.berkeley.edu) and were adapted from the AI course material of University of California, Berkeley (http://ai.berkeley.edu).

MŰEGYETEM 1782

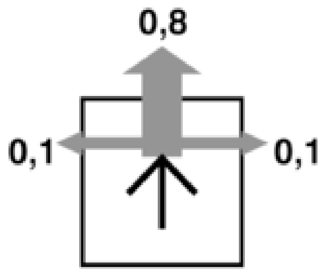
Markov döntési folyamat

Előadó: Dr. Hullám Gábor

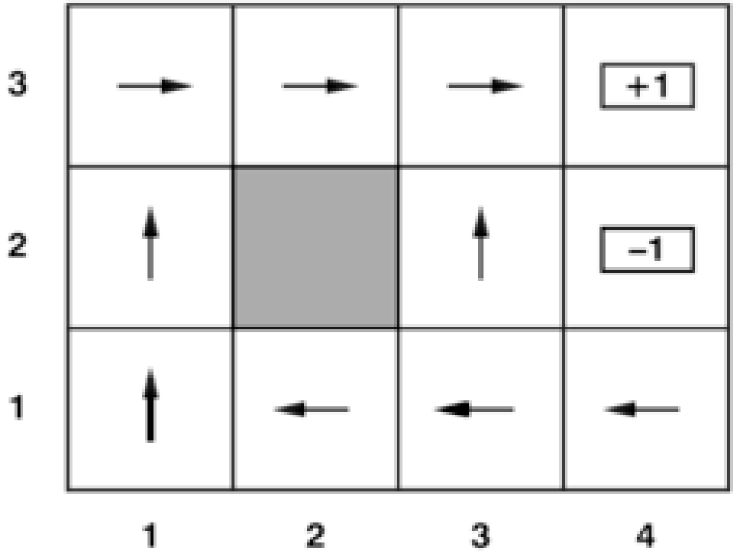
Szekvenciális döntési probléma



(a)



- 0.04



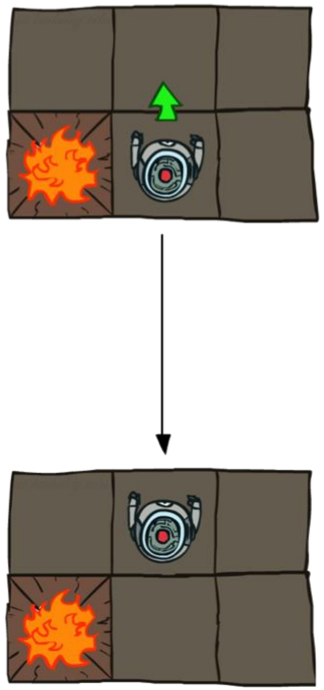
(b)

Fix út: fel, fel, jobbra, jobbra, jobbra?

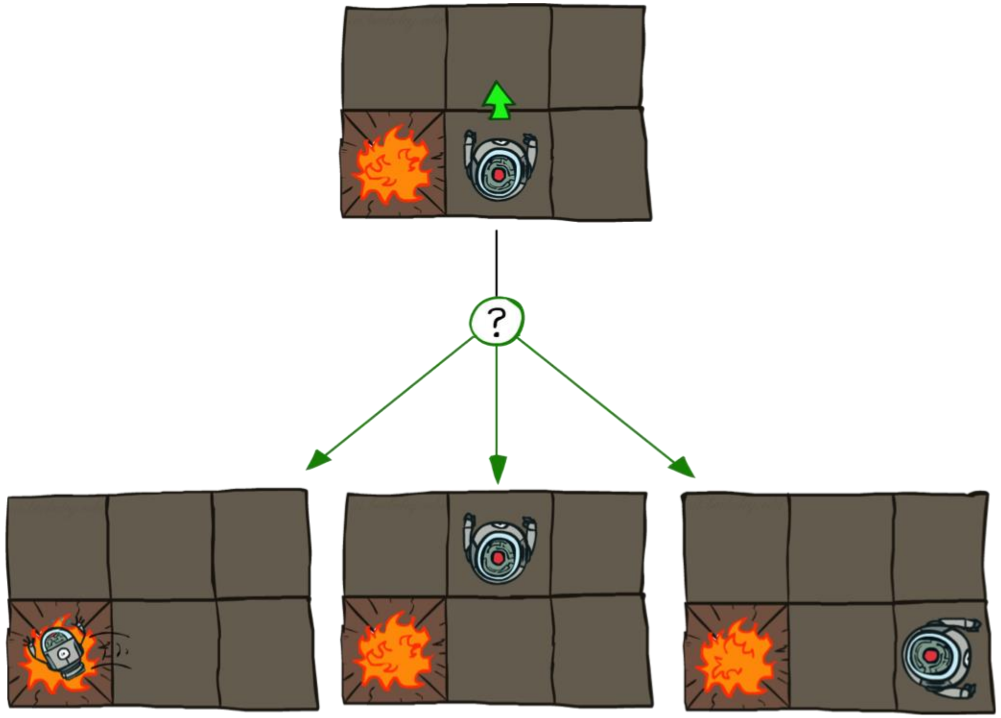
(optimális) Eljárásmód: $\pi(s) = a$

Grid World Cselekvések

Determinisztikus Grid World

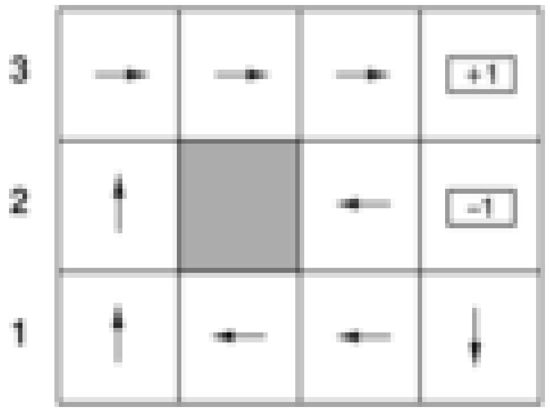


Sztochasztikus Grid World

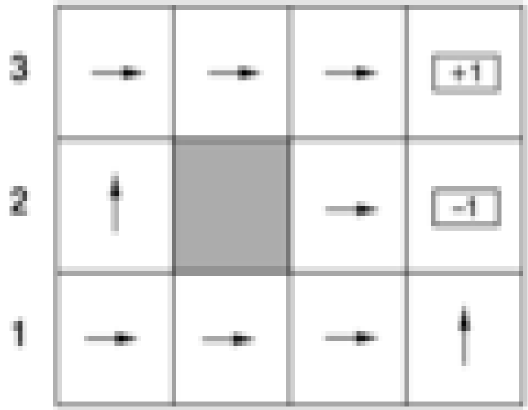


Jutalom és viselkedés

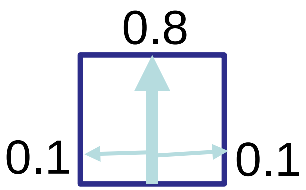
– 0,0221 < R(s) < 0



Az „élet” csak kevéssé bánatos, ne legyen kockázat!

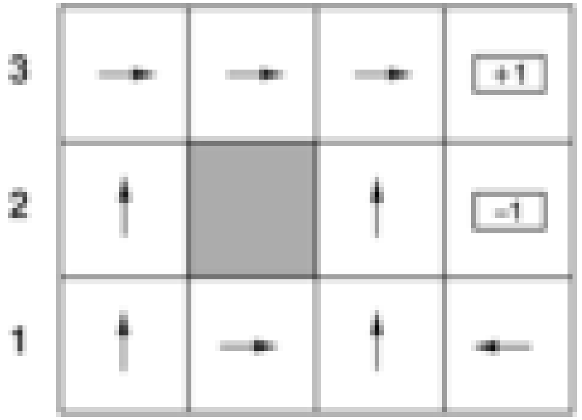


Az „élet” elviselhetetlen, kilépés a problémából.



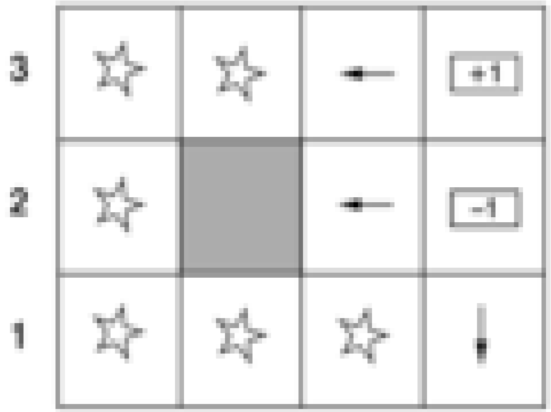
R(s) ≤ –1,6284

– 0,4278 ≤ R(s) ≤ – 0,0850



Az „élet” kellemetlen; +1 állapot, –1 kockázattal

R(s) > 0



Az „élet” kifejezetten élvezhető, az ágens benn akar maradni

Szekvenciális döntési probléma

Optimalis szekvenciális döntési probléma

	optimális eljárásmód
végtelen horizont	stacionárius
véges horizont	nem-stacionárius

Többattribútumú hasznosságelmélet:
ágens preferenciái az állapotsorozatok között stacionáriusok

additív jutalmak $U_h([s_0, s_1, s_2, \ldots]) = R(s_0) + R(s_1) + R(s_2) + \ldots$

leszámított jutalmak

$$U_h([s_0, s_1, s_2, \ldots]) = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \ldots$$

Leszámítolás

- Észszerű a jutalmak maximalizálására törekedni
- Észszerű egy jelenlegi jutalmat preferálni egy jövőbeli jutalom helyett
- Egy megoldás: a jutalmak értékei exponenciálisan csökkennek

1

Hasznosság most

γ

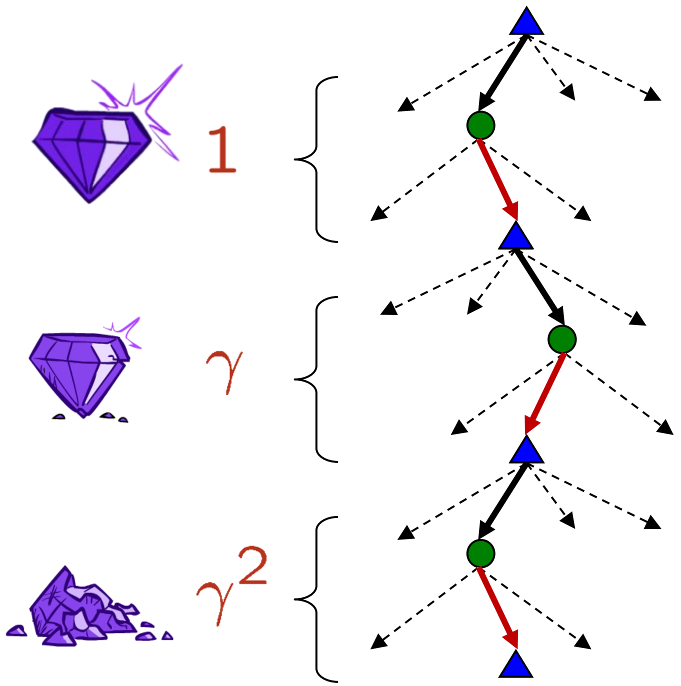
Hasznosság a következő lépésben

γ^2

Hasznosság 2 lépés múlva

Leszámítolás

- Észszerű a jutalmak maximalizálására törekedni
- Észszerű egy jelenlegi jutalmat preferálni egy jövőbeli jutalom helyett
- Egy megoldás: a jutalmak értékei exponenciálisan csökkennek



Szekvenciális jutalmak hasznossága

Leszámítolt jutalmak, egy végtelen sorozat hasznossága

$$U_h([s_0, s_1, s_2, \ldots]) = \Sigma_{t=0\ldots\infty} \gamma^t R(s_t) =< \Sigma_{t=0\ldots\infty} \gamma^t R_{\max} = R_{\max} / (1-\gamma)$$

Alternatív esetek:

1.) Ha van végállapot, és ha garantált, hogy az ágens végül bele kerül, akkor nincs szükség végtelen sorozatok összehasonlítására.
Egy eljárásmód, ami garantáltan végállapotba juttat, véges eljárásmód, $\gamma = 1$

2.) Időegységenkénti átlagjutalom

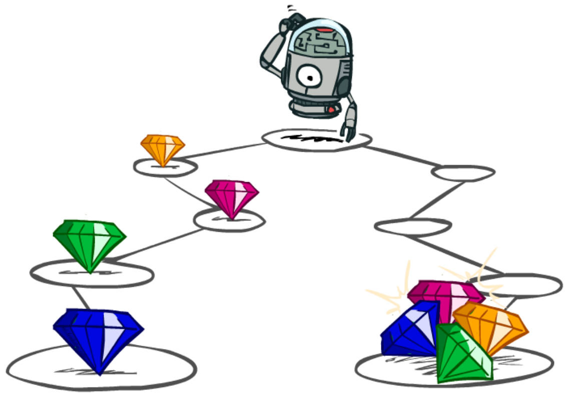
Eljárásmód (policy)

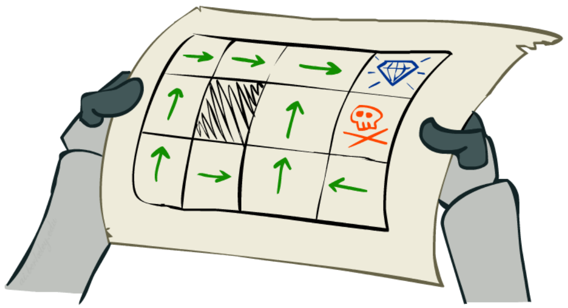
Leszámítolt jutalmak, egy végtelen sorozat hasznossága

$$U_h([s_0, s_1, s_2, \ldots]) = \Sigma_{t=0\ldots\infty} \gamma^t R(s_t) =< \Sigma_{t=0\ldots\infty} \gamma^t R_{\max} = R_{\max} / (1-\gamma)$$

Optimális eljárásmód

$$\pi^* = \arg\max_{\pi} E[\Sigma_{t=0\ldots\infty} \gamma^t R(s_t) \mid \pi]$$





Optimális eljárásmód meghatározása - Értékiteráció

Egy **állapot hasznossága –** a belőle kiinduló állapotsorozatok várható hasznossága

Az állapotsorozatok függnek a végrehajtott eljárásmódtól, így elsőként egy adott π eljárásmódra definiáljuk a hasznosságot:

$$U^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \gamma^{t} R(s_t) \,|\, \pi, s_0 = s\right]$$

Optimális eljárásmód

$$\pi^{*}(s) = \arg\max_{a} \sum_{s'} T(s,a,s') U(s')$$

Optimális eljárásmód meghatározása - Értékiteráció

Az **állapot hasznossága** - az állapotban tartózkodás közvetlen jutalmának és a következő állapot várható leszámítolt hasznosságának az összege, feltéve, hogy az ágens az optimális cselekvést választja

$$U(s) = R(s) + \gamma \max_a \sum_{s'} T(s,a,s')U(s')$$

Bellman- egyensúlyi **egyenlet**

Rácsvilág (1,1) állapot hasznossága

Legyen $\gamma = 1$ és a nem végállapotoknál R(s) = – 0,04

Nézzük meg a 4 × 3-as világ Bellman-egyenleteinek egyikét.

Az (1, 1) állapothoz tartozó egyenlet:

U(1, 1) = – 0,04 +

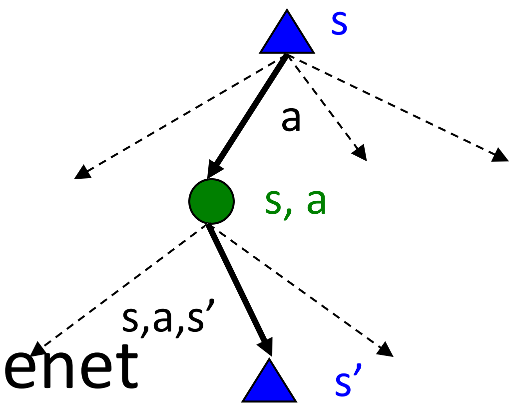
γ **max**{0,8 U(1, 2) + 0,1 U(2, 1) + 0,1 U(1, 1) (Fel),
0,8 U(2, 1) + 0,1 U(1, 2) + 0,1 U(1, 1) (Jobbra),
0,9 U(1, 1) + 0,1 U(1, 2) (Balra),
0,9 U(1, 1) + 0,1 U(2,1)} (Le)

Sajnos nemlineáris!

3	0,812	0,868	0,918	+ 1
2	0,762		0,660	–1
1	0,705	0,655	0,611	0,388
	1	2	3	4

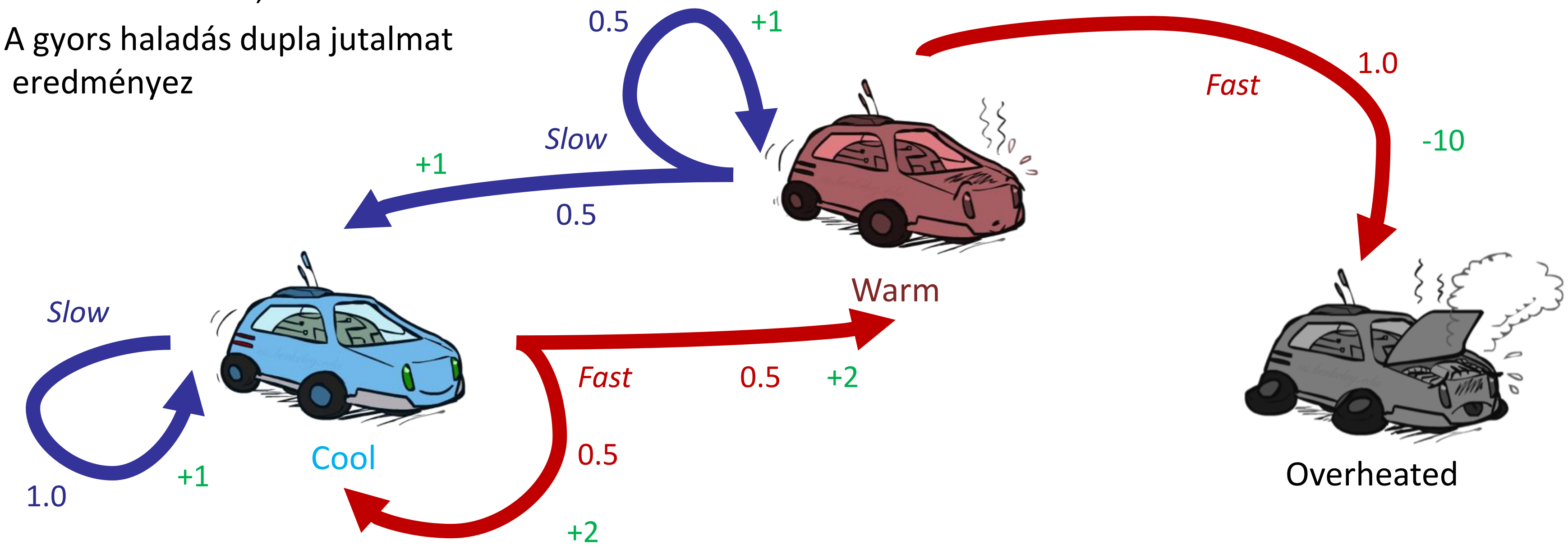
Markov döntési folyamat - definíció

- Markov döntési folyamat:
  - S: állapotok halmaza
  - s_0 : Kezdőállapot
  - A: cselekvések halmaza
  - P(s’|s,a) (or T(s,a,s’)): állapotátmenet
  - R(s,a,s’): jutalom (reward)
  - γ: leszámítolási tényező (discount)

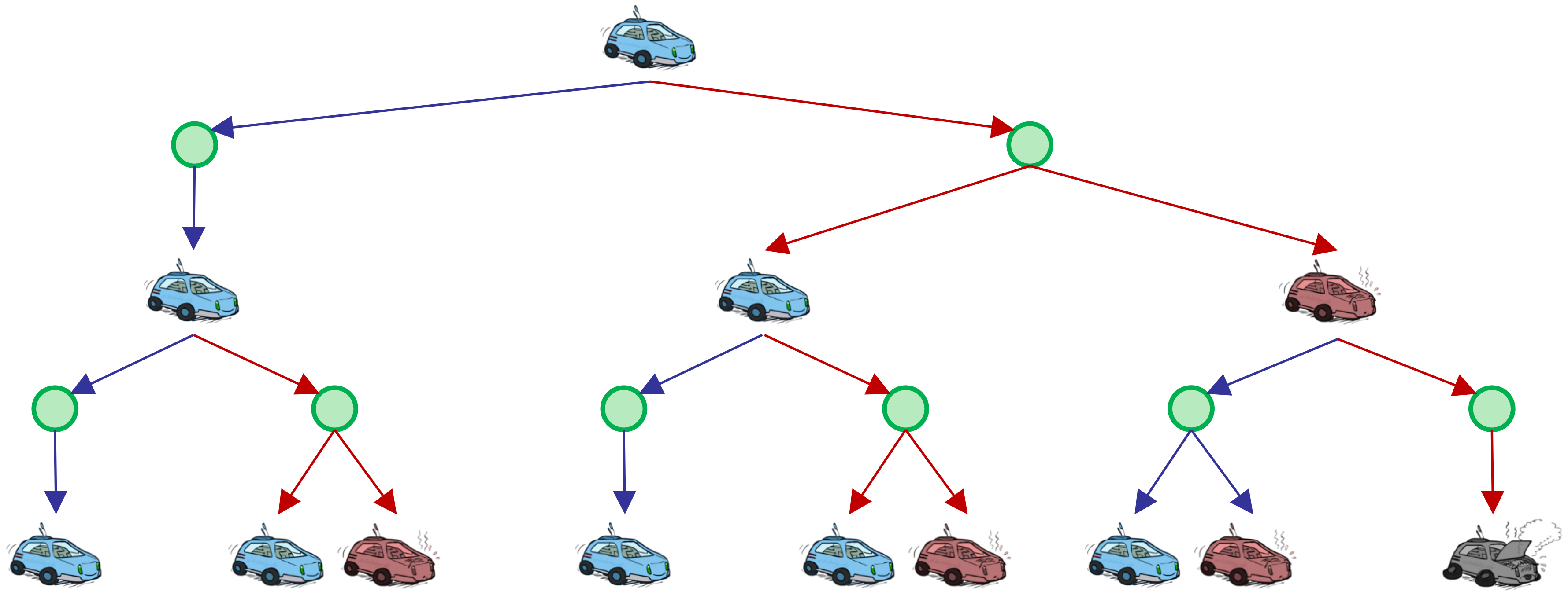


Példa: Versenyautó

- Egy robot (versenyautó) messzire szeretne utazni, gyorsan.
- 3 állapot: Cool, Warm, Overheated
- 2 cselekvés: *Slow*, *Fast*
- A gyors haladás dupla jutalmat eredményez

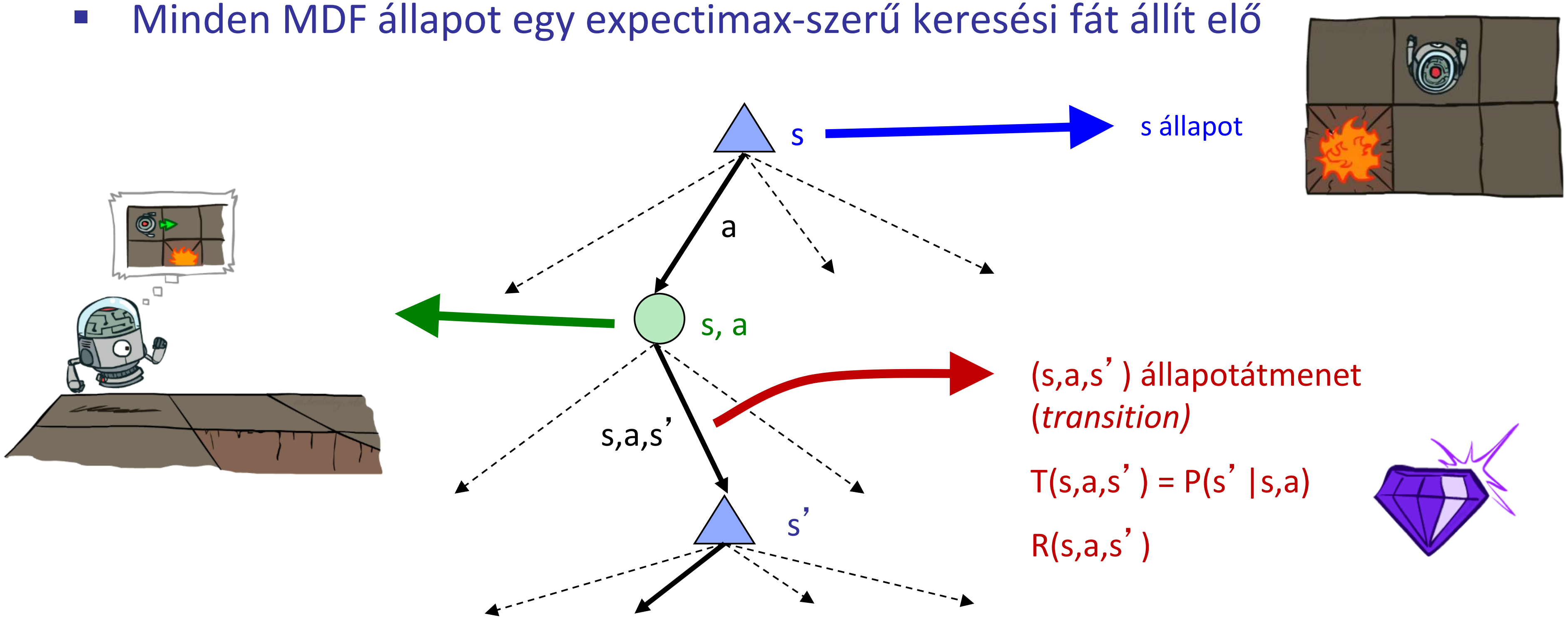


Versenyautó játékfa (keresési fa)



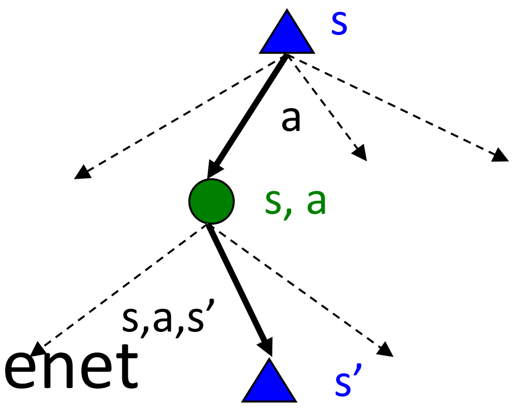
MDF keresési fák

- MDF: Markov döntési folyamat – MDP: Markov decision process
- Minden MDF állapot egy expectimax-szerű keresési fát állít elő



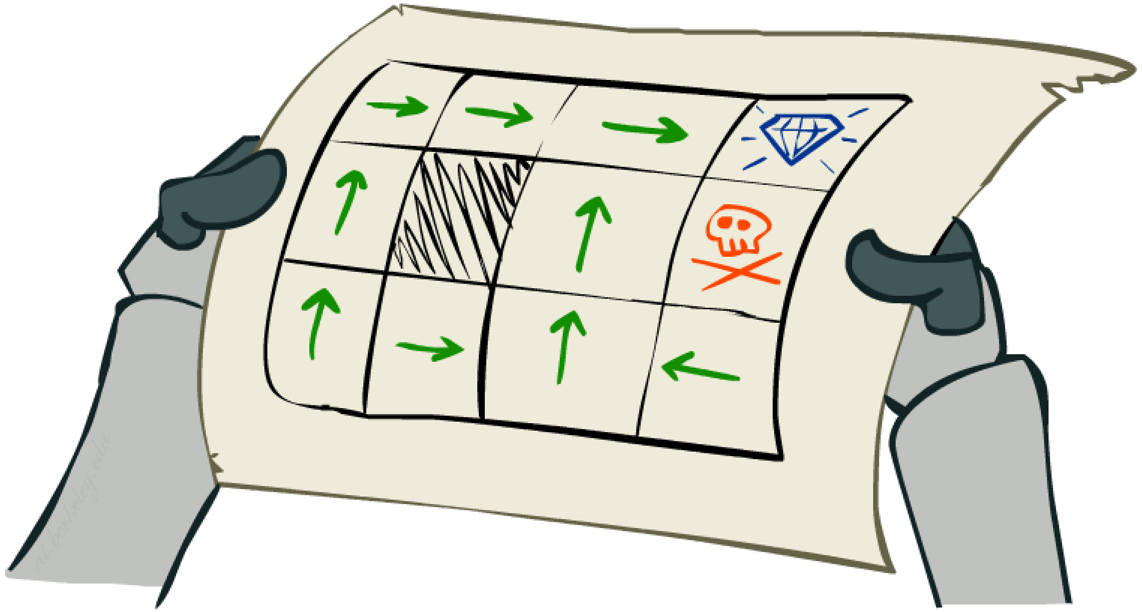
Markov döntési folyamat - definíció

- Markov döntési folyamat:
  - S: állapotok halmaza
  - s_0 : Kezdőállapot
  - A: cselekvések halmaza
  - P(s'|s,a) (or T(s,a,s')): állapotátmenet
  - R(s,a,s'): jutalom (reward)
  - γ: leszámítolási tényező (discount)



- MDF mennyiségek:
  - Eljárás = Egy cselekvés választása minden állapotban
  - Hasznosság = leszámítolt jutalmak összege

Markov döntési folyamatok megoldása



Optimális „mértékek”

- Az s állapot hasznossága:

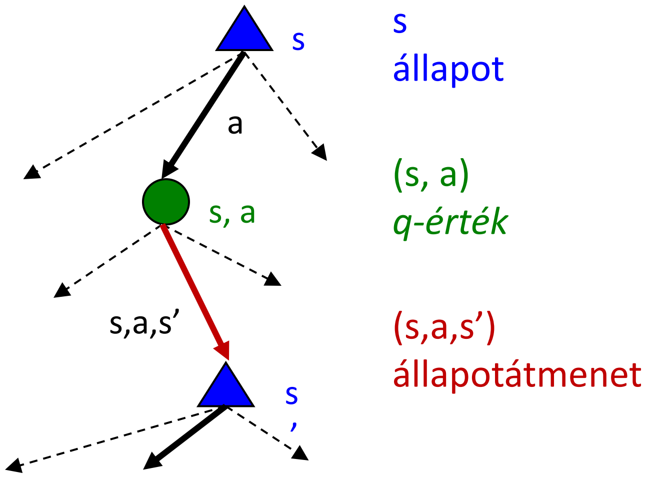
  $U^*(s)$ = várható hasznosság „s” állapotban kezdve, optimális cselekvést végrehajtva

- Egy q(s,a) cselekvésérték (q-érték) hasznossága:

  $Q^*(s,a)$ = várható hasznosság „a” optimális cselekvés után „s” állapotból kezdve

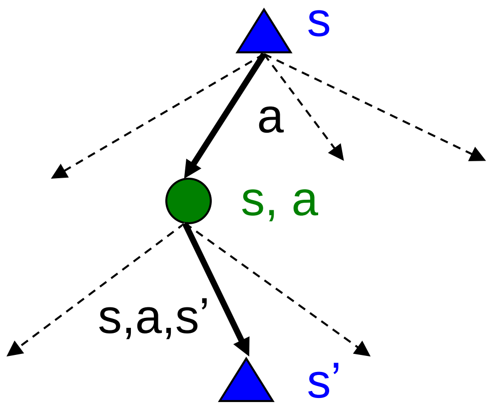
- Az optimális eljárásmód:

  $\pi^*(s)$ = optimális cselekvés „s” állapotból



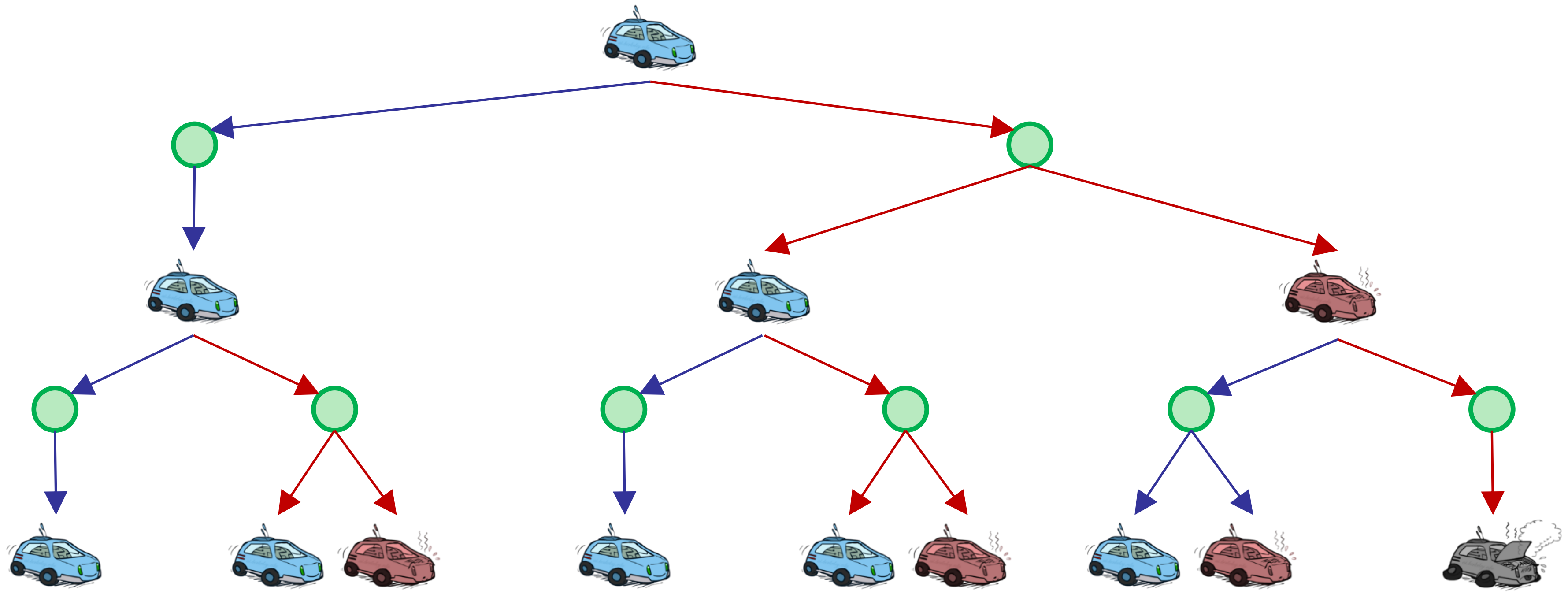
Állapotok hasznossága

- Alapvető művelet: egy állapot (expectimax) értékének kiszámítása
  - Várható hasznosság optimális cselekvés mellett
  - A (leszámítolt) jutalmak átlagos összege
  - Az expectimax éppen ezt számította ki!

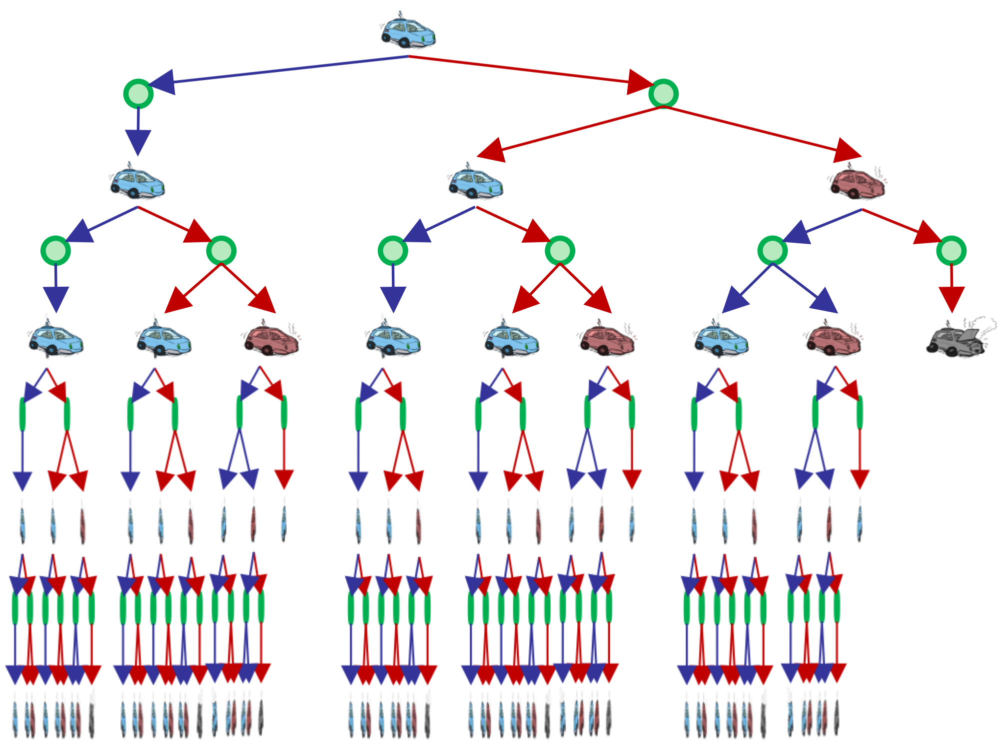


- A hasznosság meghatározása rekurzív:
  - $U^*(s) = \max_a Q^*(s,a)$
  - $Q^*(s,a) = \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma\, U^*(s')]$
  - $U^*(s) = \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma\, U^*(s')]$

Versenyautó játékfa

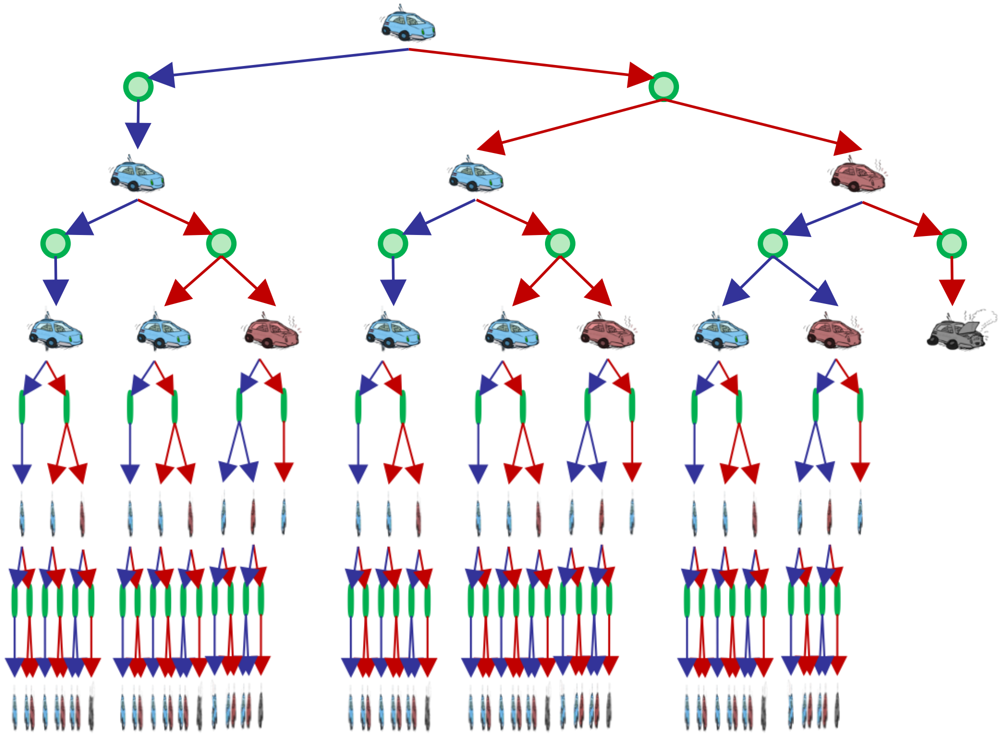


Versenyautó játékfa



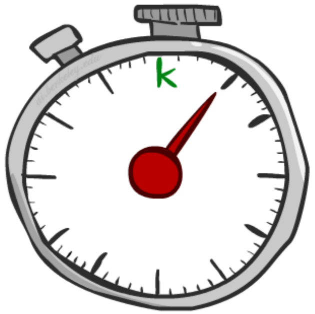
Versenyautó játékfa

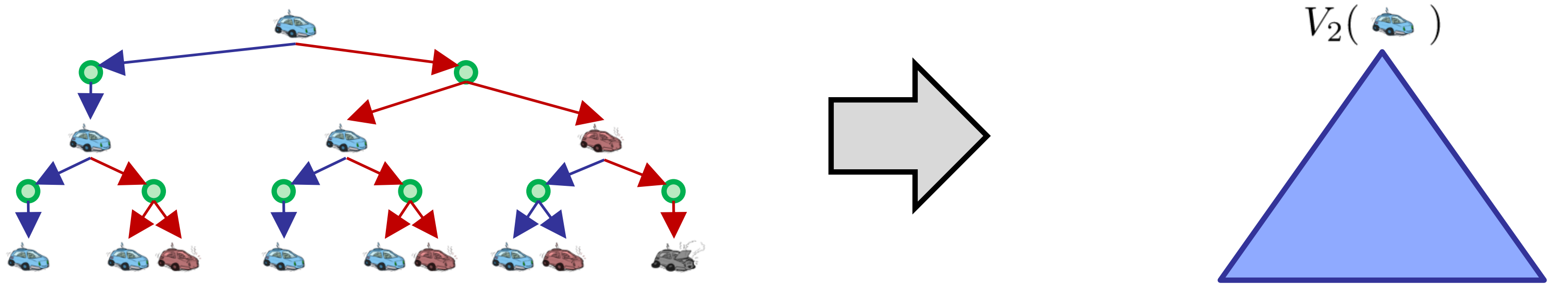
- Túl sok munkát végzünk az expectimax-szal!
- Probléma: Állapotok ismétlődnek
  - Ötlet: Csak egyszer számítsuk ki a szükséges mennyiségeket
- Probléma: A fa örökké tart
  - Ötlet: Mélységkorlátos számítás, de növekvő mélységgel, amíg a változás kicsi nem lesz.
  - Megjegyzés: a fa mély részei végül nem számítanak, ha γ < 1.



Időben korlátozott hasznosság

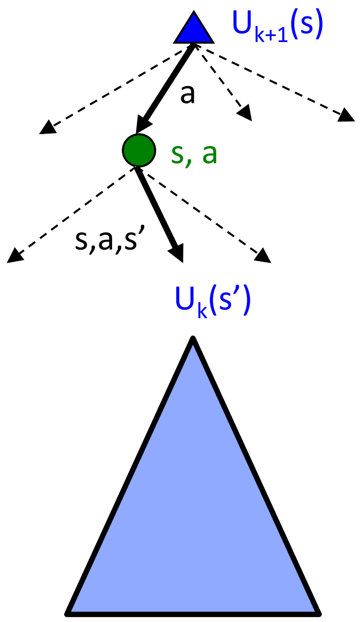
- Kulcsgondolat: időben korlátozott hasznosságok
- Legyen $U_k(s)$ az „s” optimális hasznossága, ha a játék k időlépésen belül ér véget.
  - Ekvivalens módon, ez az, amit egy k-mélységű expectimax adna az s állapotból indítva





Értékiteráció

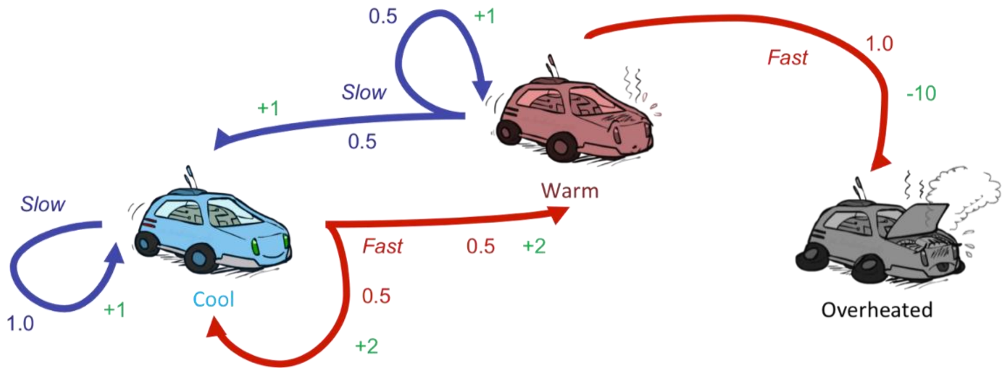
- $U_0(s)$ = 0-val kezdünk: ha nincs hátralévő időlépés, a várható jutalom összege nulla.
- Adott $U_k(s)$ értékek vektora, végezzünk minden állapotból egy-egy expectimax lépést:
  - $U_{k+1}(s) \leftarrow \max_{a} \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma \; U_k(s')]$



- Ismételjük egészen konvergenciáig
- Az egyes iterációk bonyolultsága: $O(S^2A)$
- Tétel: egyetlen optimális értékhez konvergál.
  - Alapötlet: a közelítések az optimális értékek felé finomodnak.

Példa: értékiteráció

V_2	3.5	2.5	0
V_1	2	1	0
V_0	0	0	0



$$U_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma\ U_k(s')]$$

Leszámítolási tényező=1

Értékiteráció - összefoglalás

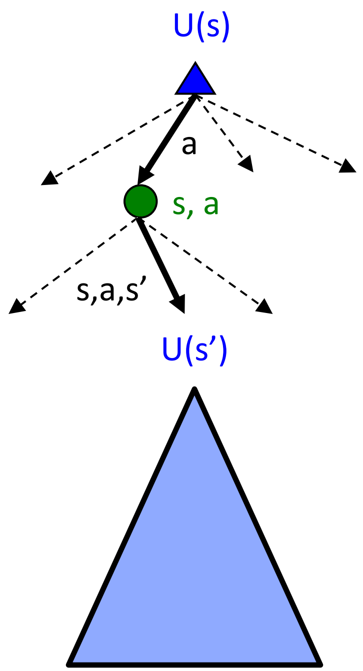
- A Bellman-egyenlet leírja az optimális hasznosságokat:

$$U^*(s) = \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma \ U^*(s')]$$

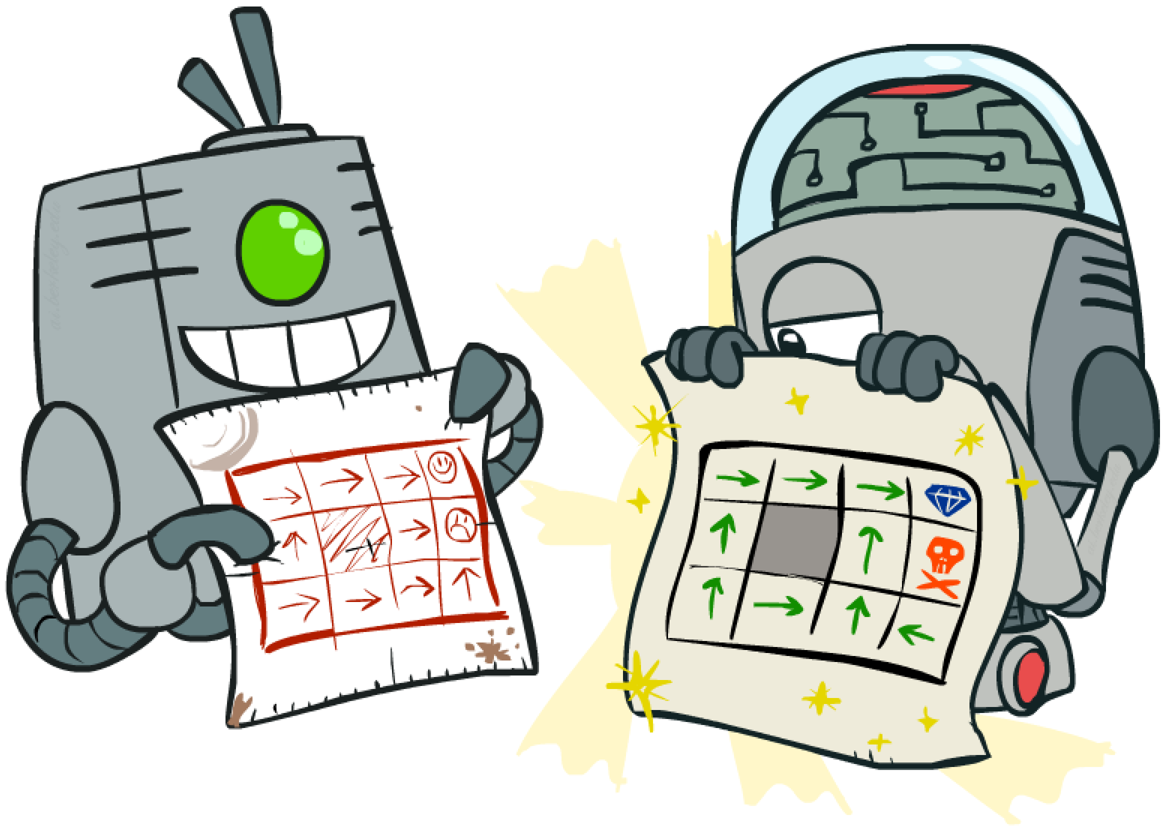
- Az értékiteráció elvégzi a számítást:

$$U_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma \ U_k(s')]$$

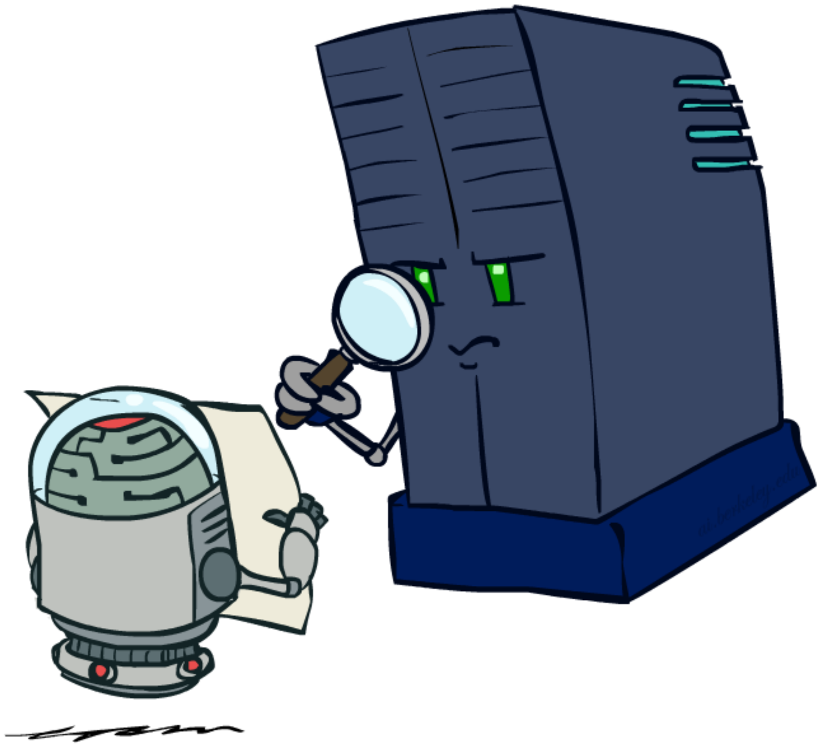
- Az értékiteráció csak egy fixpontos megoldási módszer
  - bár a Uk vektorok időben korlátozott értékekként is értelmezhetők.



Eljárásmód módszerek

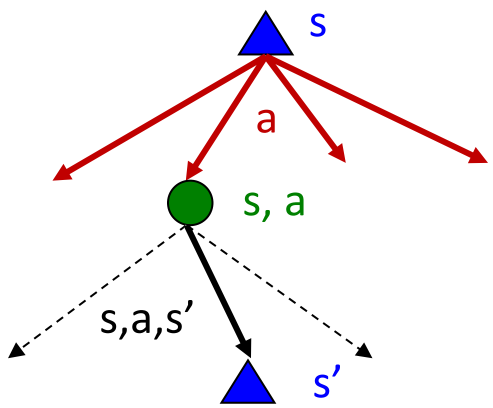


Eljárásmód kiértékelés

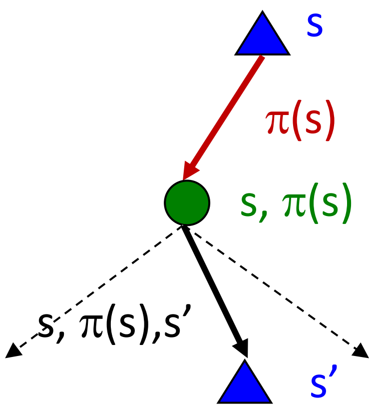


Rögzített eljárásmód

Az optimális cselekvés választása



π eljárás szerinti cselekvés

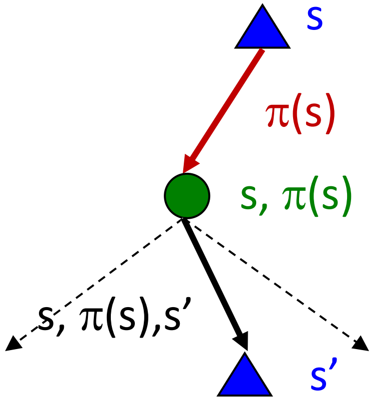


- Expectimax fák az összes akció felett képzik a maximumot az optimális hasznosságok kiszámításához
- Ha rögzítenénk néhány eljárásmódot π(s), akkor a fa egyszerűbb lenne - állapotonként csak egy akció lenne....
  - bár a fa értéke attól függene, hogy melyik eljárásmódot rögzítettük.

Hasznosságok rögzített eljárásmód esetén

- Egy másik alapvető művelet: kiszámítjuk egy „s” állapot hasznosságát egy rögzített (általában nem optimális) eljárásmód mellett.
- Határozzuk meg az s állapot hasznosságát egy rögzített π eljárásmód mellett:

  $U^{\pi}(s)$ = várható összes leszámítolt jutalom „s”-től kezdődően π-t követve

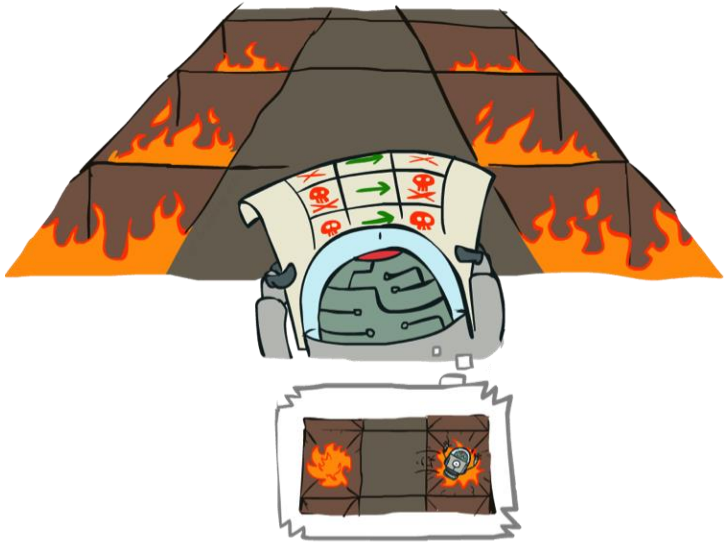


- Rekurzív kapcsolat (egylépéses előretekintés / Bellman-egyenlet):

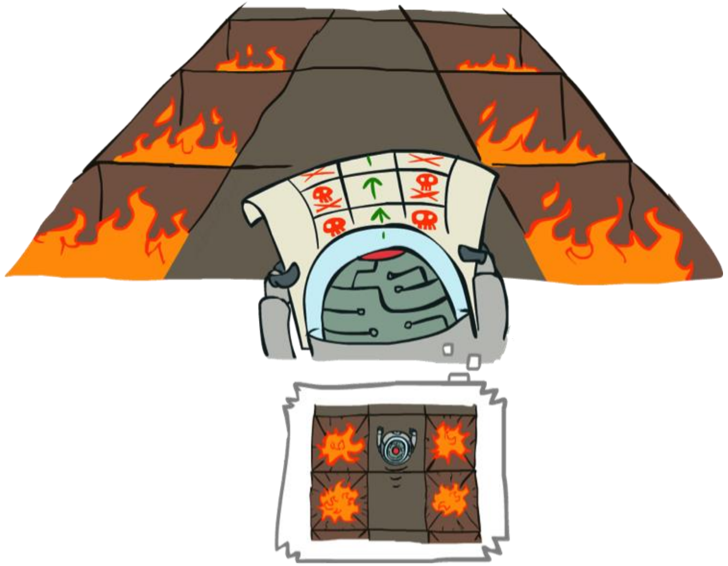
$$U^{\pi}(s) = \sum_{s'} T(s,\pi(s),s')[R(s,\pi(s),s') + \gamma\, U^{\pi}(s')]$$

Példa: Eljárásmód kiértékelés

Always Go Right



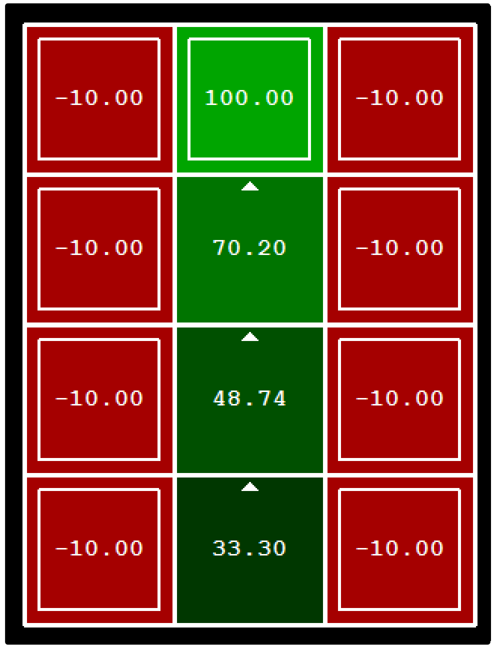
Always Go Forward



Példa: Eljárásmód kiértékelés

Always Go Right

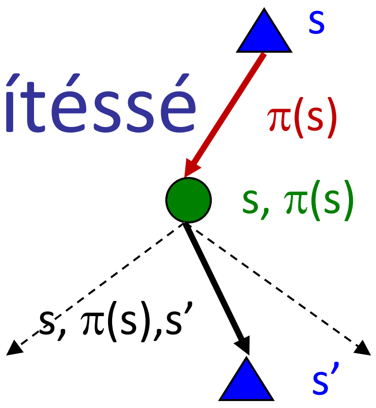
Always Go Forward



Eljárásmód kiértékelés

- Hogyan számoljuk ki a U-kat egy rögzített π eljáráshoz?

- 1. ötlet: A rekurzív Bellman-egyenleteket frissítéssé alakítjuk (mint az értékiterációnál)



- Hatékonyság: $O(S^2)$ per iteráció

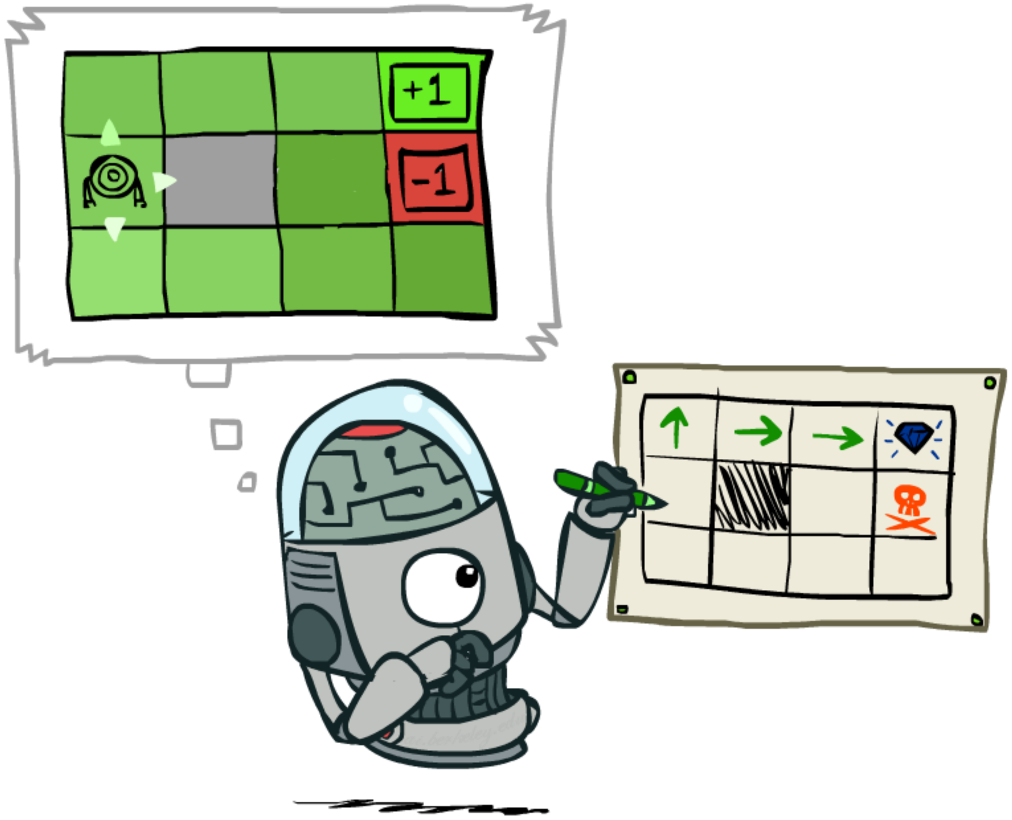
- 2. ötlet: A maximumok nélkül a Bellman-egyenletek lineáris rendszert alkotnak

$$U_0{}^{\pi}(s) = 0$$

  - Megoldás Matlab vagy más lineáris egyenletrendszer megoldó (solver) segítségével

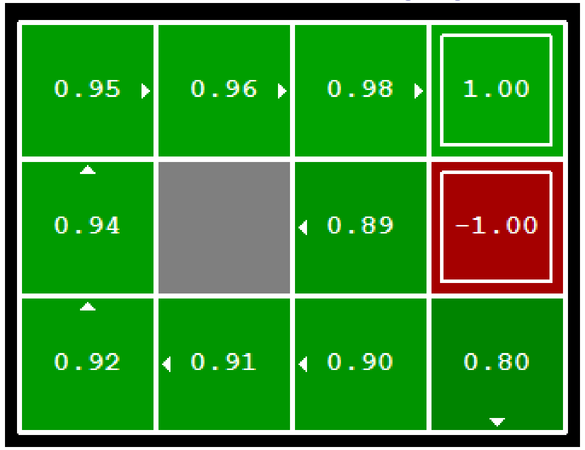
$$U_{k+1}{}^{\pi}(s) \leftarrow \sum_{s'} T(s,\pi(s),s')[R(s,\pi(s),s') + \gamma\, U_k{}^{\pi}(s')]$$

Eljárásmód kinyerése



Cselekvések számítása hasznosságokból

- Képzeljük el, hogy megvannak az optimális értékek U*(s)
- Hogyan kell cselekednünk?
  - Ez nem egyértelmű!



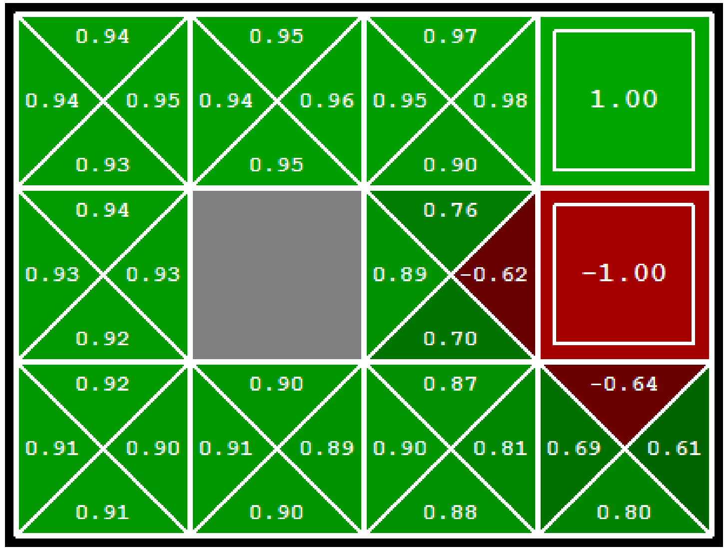
- Egy mini-expectimaxot kell végeznünk (egy lépésnyit)

$$\pi^*(s) = \underset{a}{\operatorname{argmax}} \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma\ U^*(s')]$$

- Ezt nevezzük eljárásmód **kinyerésnek**, mivel az hasznosságok által implikált eljárásmódot kapjuk meg

Cselekvések számítása Q-értékekből

- Képzeljük el, hogy megvannak az optimális q-értékek:

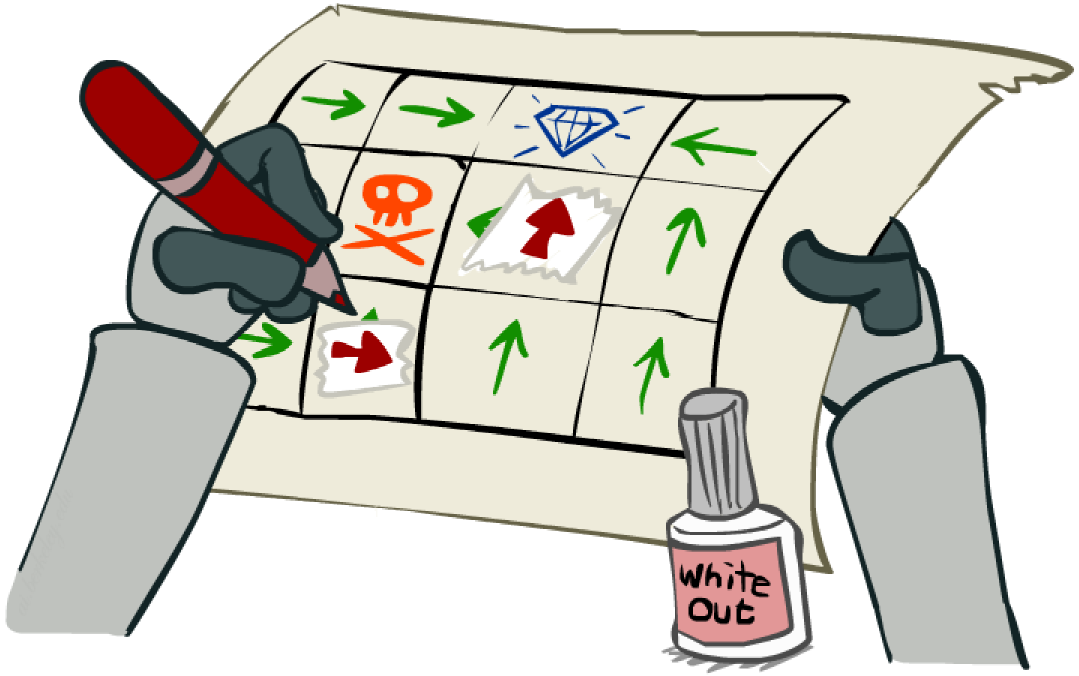


- Hogyan kell cselekednünk?
  - Teljesen triviális a döntés!

$$\pi^*(s) = \underset{a}{\operatorname{argmax}}\, Q^*(s,a)$$

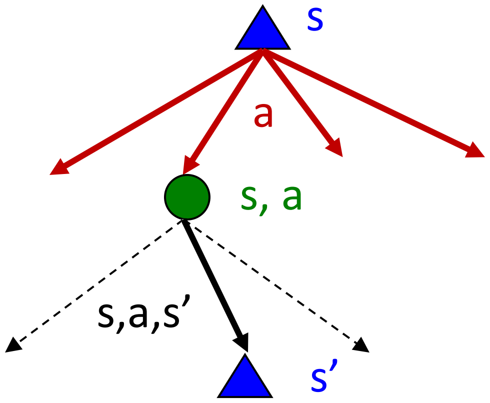
- Fontos tanulság: a q-értékekből könnyebb cselekvéseket választani, mint a hasznosságokból

Eljárásmód-iteráció



Értékiteráció problémái

- Az értékiteráció a Bellman-frissítéseket ismétli
- 1. probléma: lassú, $O(S^2A)$ iterációnként



- 2. probléma: Az egyes állapotok "max" értéke ritkán változik.
- 3. probléma: Az eljárásmód gyakran jóval az értékek előtt konvergál

Eljárásmód-iteráció

- Az optimális értékek alternatív megközelítése:
  - 1. Lépés: Eljárás kiértékelése: számítsuk ki a hasznosságokat valamilyen rögzített eljárásmódra (nem optimális hasznosságok!) a konvergenciáig.
  - 2. lépés: Eljárás javítása: az eljárás frissítése egylépéses előretekintéssel, az így kapott konvergált (de nem optimális!) hasznosságokkal mint jövőbeli értékekkel.
  - A lépések ismétlése az eljárásmód konvergenciájáig

- Ez az eljárásmód-iteráció
  - Még mindig optimális
  - Bizonyos feltételek mellett (sokkal) gyorsabban konvergálhat.

Eljárásmód-iteráció

- Kiértékelés: Rögzített eljárásmód mellett határozzuk meg a hasznosságokat az eljárásmód kiértékelésével
  - Ismétlés, amíg az értékek nem konvergálnak:

$$U_{k+1}{}^{\pi_i}(s) \leftarrow \sum_{s'} T(s,\pi(s),s')[R(s,\pi(s),s') + \gamma\, U_k{}^{\pi_i}(s')]$$

- Javítás: Rögzített hasznosságok mellett jobb eljárásmód kialakítása eljárásmód kinyeréssel
  - Egylépéses előretekintés:

$$\pi_{i+1}(s) = \underset{a}{\operatorname{argmax}} \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma\, U^{\pi_i}(s')]$$

Összehasonlítás

- Mind az értékiteráció, mind az eljárásmód-iteráció ugyanazt a dolgot számítja ki : az összes optimális értéket.
- Érték-iterációban:
  - Minden iteráció frissíti a hasznosságokat és (implicit módon) az eljárásmódot is.
  - Nem vizsgáljuk az eljárásmódot, de a cselekvések maximális értékének figyelembe vétele implicit módon újraszámítja azt.
- A eljárásmód-iterációban:
  - Több iterációt végzünk, amelyek a hasznosságokat fix eljárásmóddal frissítik (minden egyes iteráció gyors, mert csak egy akciót veszünk figyelembe, nem az összeset).
  - Az eljárásmód kiértékelése után új eljárásmódot választunk (lassú, mint egy érték iterációs lépés).
  - Az új politika jobb lesz (vagy végeztünk).

  - Mindkettő dinamikus programozást igényel az MDF-ek megoldására