Umjetna inteligencija

4. Igranje igara

prof. dr. sc. Bojana Dalbelo Bašić
izv. prof. dr. sc. Jan Šnajder

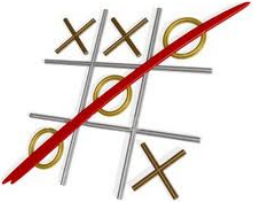
Sveučilište u Zagrebu
Fakultet elektrotehnike i računarstva

Ak. god. 2018./2019.

 v2.10

Igre

- Također problem pretraživanja prostora stanja, ali postoji **protivnik**
- U svakom stanju potrebno je donijeti **optimalnu odluku** o sljedećem potezu, tj. treba pronaći optimalnu **strategiju**
- Fokusiramo se na **determinističke** igre s **dva igrača**, **potpunom informacijom** i **sumom nula**



Formalizacija problema

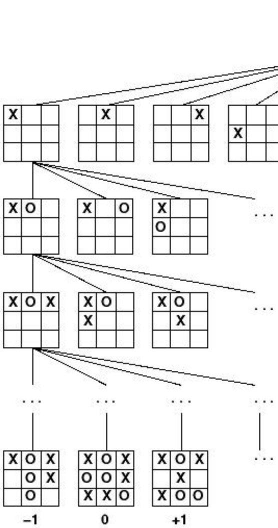
Problem pretraživanja sa sljedećim komponentama:

Igra

- **Početno stanje** igre s_0
- **Funkcija sljedbenika** $\mathrm{succ} : S \to \wp(S)$, koja definira valjane poteze igre (prijelaze između stanja)
- **Test na završno stanje** $\mathrm{terminal} : S \to \{\top, \bot\}$
- **Isplatna funkcija** $\mathrm{utility} : S \to \mathbb{R}$ koja pridjeljuje numeričku vrijednost koju kao nagradu dobiva igrač u završnom stanju igre
  Npr. u šahu: $\mathrm{utility}(s) \in \{+1, 0, -1\}$

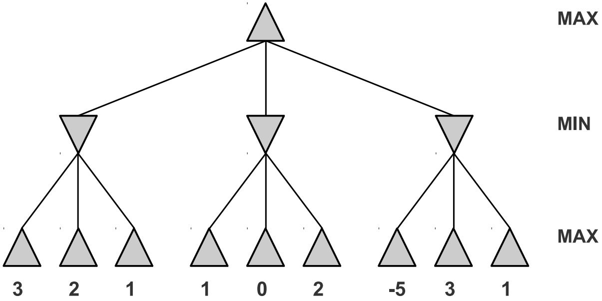
Početno stanje i funkcija sljedbenika definiraju **stablo igre**

Stablo igre



Metoda minimaks

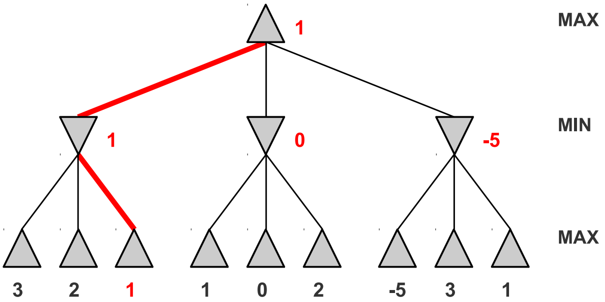
- Igrače ćemo nazvati MAX (računalo) i MIN (protivnik)
- Igrač MAX nastoji **maksimizirati** svoj dobitak, dok igrač MIN nastoji **minimizirati** dobitak igrača MAX
- Igrači igraju naizmjenično: čvorovi na parnoj udaljenosti neka su MAX, a čvorovi na neparnoj udaljenosti neka su MIN



- **Q**: Što je u ovom slučaju optimalna strategija igrača MAX?

Optimalna strategija

- **Optimalna strategija** igrača MAX je strategija koja mu donosi najveći dobitak, uz pretpostavku da igrač MIN koristi istu strategiju
- Svaki igrač koristi strategiju koja **minimizira maksimalan gubitak**

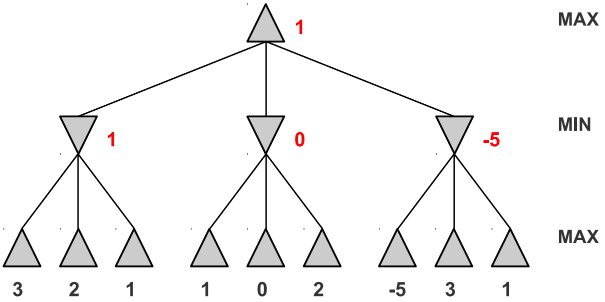


- Da bismo odredili **optimalnu strategiju** onog igrača koji je upravo na redu, trebamo izračunati **minimaks-vrijednost** korijenskog čvora

Minimaks-vrijednost

- Minimaks-vrijednost čvora s definirana je rekurzivno:

$$m(s) = \begin{cases} \text{utility}(s) & \text{ako terminal}(s) \\ \max_{t \in \text{succ}(s)} m(t) & \text{ako } s \text{ je MAX čvor} \\ \min_{t \in \text{succ}(s)} m(t) & \text{ako } s \text{ je MIN čvor} \end{cases}$$



$$m(s_0) = \max\big(\min(3,2,1), \min(1,0,2), \min(-5,3,1)\big) = 1$$

Algoritam minimaks

function $\mathrm{maxValue}(s)$
  if $\mathrm{terminal}(s)$ **then return** $\mathrm{utility}(s)$
  $m \leftarrow -\infty$
  for $t \in \mathrm{succ}(s)$ **do**
    $m \leftarrow \max(m, \mathrm{minValue}(t))$
  return m

function $\mathrm{minValue}(s)$
  if $\mathrm{terminal}(s)$ **then return** $\mathrm{utility}(s)$
  $m \leftarrow +\infty$
  for $t \in \mathrm{succ}(s)$ **do**
    $m \leftarrow \min(m, \mathrm{maxValue}(t))$
  return m

NB: Pretraživanje **u dubinu** ostvareno dvjema međusobno rekurzivnim funkcijama (alternacija između stanja MAX i MIN)

Algoritam minimaks – napomene

- U praksi je protivnikova strategija **nepoznata** (i vjerojatno različita of strategije igrača MAX), pa zbog toga nije moguće savršeno predvidjeti protivnikove poteze (inače bi igra ionako bila dosadna)
- Zbog toga, kako bi povukli optimalan potez, svaki puta kada su na redu igrači moraju **nanovo izračunati** svoju optimalnu strategiju, krenuvši od trenutne pozicije igre u korijenu stabla
- Minimax **pretražuje u dubinu**, pa je njegova prostorna složenost $\mathcal{O}(m)$, gdje je m dubina stabla pretraživanja
- Međutim, vremenska složenost je $\mathcal{O}(b^m)$, gdje je b faktor grananja igre. To je vrlo nezgodno!

Nesavršene odluke

- U stvarnosti nemamo vremena potpuno pretražiti stablo sve do završnih čvorova
- Moramo donositi **vremenski ograničene** i **nesavršene odluke**
- Pretraživanje treba **presjeći** na određenoj dubini d i napraviti **procjenu** vrijednosti isplatne funkcije uporabom **heurističke funkcije**
- Vrijednost $h(s)$ je procjena isplativosti stanja s za igrača MAX
- Npr. za šah: zbroj vrijednosti igračevih figura
- Heuristička funkcija često je definirana kao **težinska linearna kombinacija** više značajki:

$$h(s) = w_1 x_1(s) + w_2 x_2(s) + \cdots + w_n x_n(s)$$

- **NB:** Igrači tipično imaju različite heurističke funkcije (zato se i doimaju nepredvidivi)

Algoritam minimaks (2)

Algoritam minimaks s presjecanjem

function $\text{maxValue}(s, d)$
  if $\text{terminal}(s)$ **then return** $\text{utility}(s)$
  if $d = 0$ **then return** $h(s)$
  $m \leftarrow -\infty$
  for $t \in \text{succ}(s)$ **do**
    $m \leftarrow \max(m, \text{minValue}(t, d-1))$
  return m

function $\text{minValue}(s, d)$
  if $\text{terminal}(s)$ **then return** $\text{utility}(s)$
  if $d = 0$ **then return** $h(s)$
  $m \leftarrow +\infty$
  for $t \in \text{succ}(s)$ **do**
    $m \leftarrow \min(m, \text{maxValue}(t, d-1))$
  return m

Kviz: Minimaks-vrijednost

Pitanje 1

Stablo igre definirano je prijelazima $succ(A) = \{B, C, D\}$, $succ(B) = \{E, F\}$, $succ(C) = \{G, H\}$, $succ(D) = \{I, J\}$. Heurističke vrijednost listova jesu $h(E) = -1$, $h(F) = 3$, $h(G) = 2$, $h(H) = 4$, $h(I) = 5$, $h(J) = 1$. Što je minimaks-vrijednost čvora A, ako je to MIN čvor?

(A) -1

(B) 3

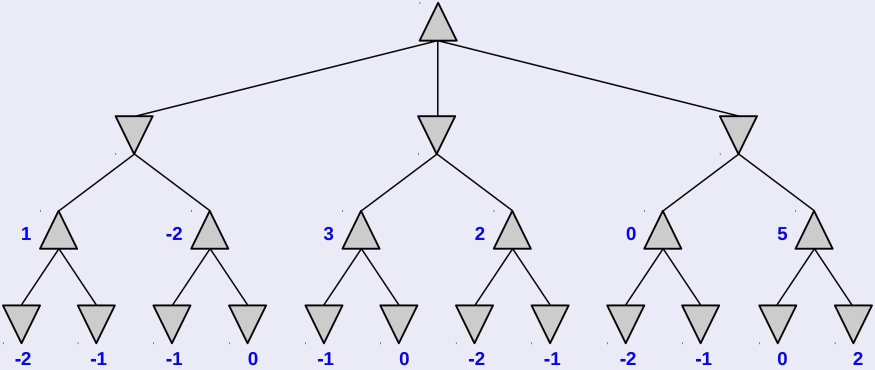
(C) 2

(D) 4

Kviz: minimaks strategija s heuristikom

Pitanje 2

Koje je krajnje stanje, ako igrači pretražuju dvije razine u dubinu?

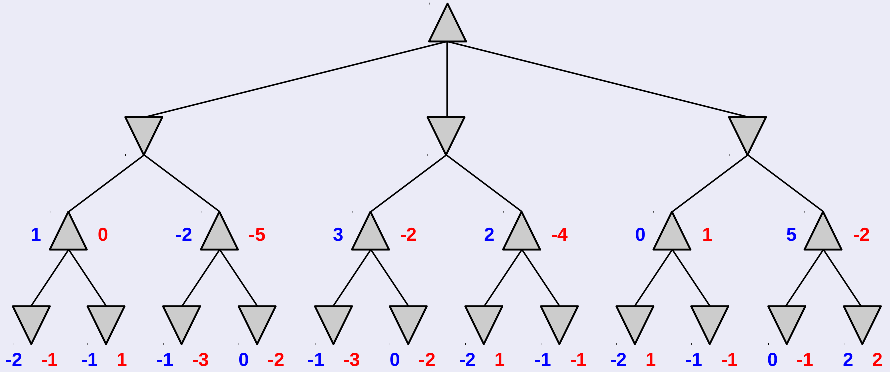


Q: Što ako pretražuju tri razine u dubinu?

Kviz: minimaks strategija s dvije heuristike

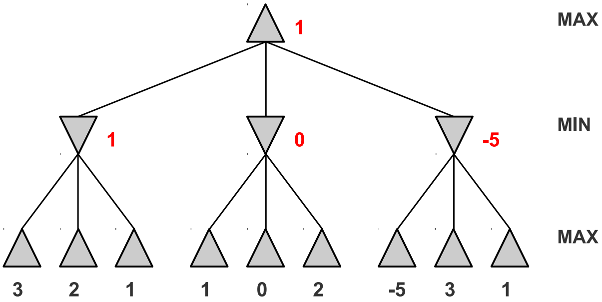
Pitanje 3

Koje je krajnje stanje igre, ako igrači pretražuju dvije razine u dubinu, ali da koriste različite heuristike, h_1 (plavo) i $-h_2$ (crveno)?



Podrezivanje alfa-beta

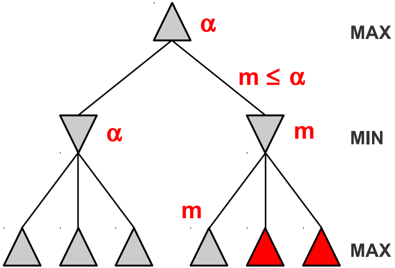
- Broj stanja igre raste eksponencijalno s brojem poteza
- Primjenom **podrezivanja** taj broj možemo međutim prepoloviti
- **Q:** Možemo li odrediti minimaks-vrijednost, a da ne obiđemo cijelo stablo igre? **A:** Da!



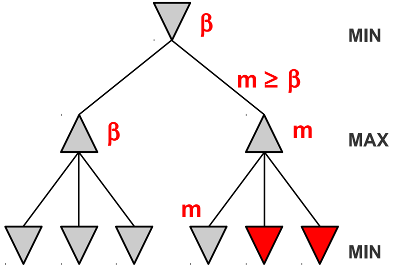
$$m(s_0) = \max\big(\min(3,2,1), \min(1, {\color{red}X}, {\color{red}X}), \min(-5, {\color{red}X}, {\color{red}X})\big) = {\color{red}1}$$

Podrezivanje alfa-beta

- Podrezujemo kad god se za neki čvor ustanovi da potezi **ni u kojem slučaju ne mogu biti povoljniji** od nekog već istraženog poteza
- Ako podrezujemo ispod MIN-čvora: **alfa-podrezivanje**
- Ako podrezujemo ispod MAX-čvora: **beta-podrezivanje**



α – najveća nađena MAX-vrijednost



β – najmanja nađena MIN-vrijednost

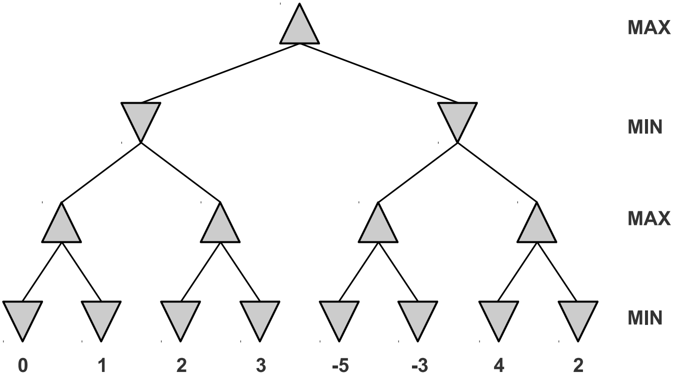
Algoritam minimaks (3)

Algoritam minimaks s podrezivanjem alfa-beta

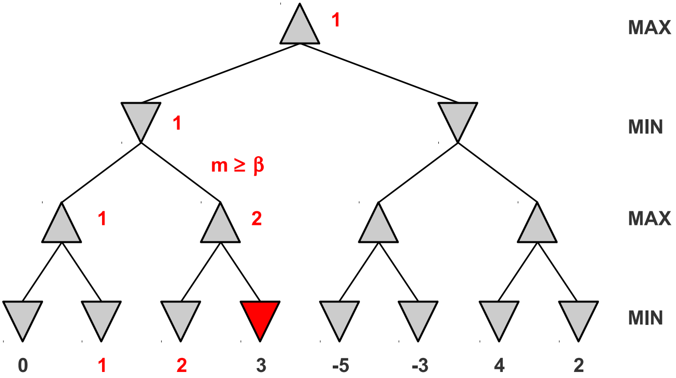
function $\mathrm{maxValue}(s, \alpha, \beta)$ -- *početno:* $\mathrm{maxValue}(\mathrm{s}_0, -\infty, +\infty)$
  if $\mathrm{terminal}(s)$ **then return** $\mathrm{utility}(s)$
  $m \leftarrow \alpha$
  for $t \in \mathrm{succ}(s)$ **do**
    $m \leftarrow \max(m, \mathrm{minValue}(t, m, \beta))$
    if $m \geq \beta$ **then return** β -- *beta-podrezivanje*
  return m

function $\mathrm{minValue}(s, \alpha, \beta)$
  if $\mathrm{terminal}(s)$ **then return** $\mathrm{utility}(s)$
  $m \leftarrow \beta$
  for $t \in \mathrm{succ}(s)$ **do**
    $m \leftarrow \min(m, \mathrm{maxValue}(t, \alpha, m))$
    if $m \leq \alpha$ **then return** α -- *alfa-podrezivanje*
  return m

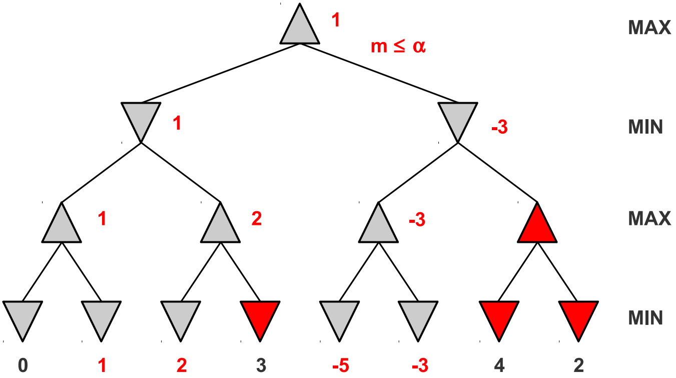
Podrezivanje alfa-beta – primjer (1)



Podrezivanje alfa-beta – primjer (2)



Podrezivanje alfa-beta – primjer (3)



Kviz: Algoritam minimaks

Pitanje 4

Cilj algoritma minimaks jest:

(A) minimizirati najveći mogući gubitak
(B) minimizirati najveći gubitak protivnika
(C) maksimizirati najmanji mogući dobitak
(D) maksimizirati najmanji dobitak protivnika
(E) smanjiti prostor pretraživanja
(F) podrezati stablo igre

Laboratorijski zadatak: Igra šibica

Napišite program za igru šibica na temelju **algoritma minimaks**.

U igri šibica sudjeluju dva igrača. Pred njima se nalazi n odvojenih hrpa šibica (u svakoj hrpi može se nalaziti različit broj šibica). Igrač koji je na potezu mora sa samo jedne hrpe maknuti najmanje jednu a najviše k šibica. Igra završava kada su sve šibice uklonjene, a gubi onaj igrač koji je morao odigrati zadnji potez.

Napravite minimalističko korisničko sučelje koje će prikazivati trenutno stanje igre te omogućiti čovjeku da igra protiv računala. Za svaki potez igrača program treba dojaviti je li taj potez odigran optimalno u smislu minimaks-odluke. Korisnik pri pokretanju programa zadaje parametre n (broj hrpa), k (najveći broj šibica koje je u jednom potezu moguće ukloniti) te početni broj šibica u svakoj od n hrpa.

Laboratorijski zadatak: Igra dame

Napišite program za igru Dame temeljen na **algoritmu minimaks** uz ograničeno pretraživanje i **podrezivanje alfa-beta**.

Definirajte barem dvije različite heurističke funkcije za vrednovanje isplativosti svake pozicije igre. Pri oblikovanju heurističke funkcije posebnu pažnju pokušajte posvetiti situacijama kada se u igri dođe do kraljeva. Pretraživanje ograničite, bilo brojem razmatranih pozicija, dubinom pretraživanja ili stvarnim vremenom.

Napravite minimalističko korisničko sučelje koje će omogućiti čovjeku da igra protiv računala te prikazivati trenutnu poziciju igre. Za svaki potez igrača program treba dojaviti je li taj potez odigran optimalno u smislu minimaks-odluke. Omogućite i da program igra protiv samoga sebe, pri čemu igrači mogu koristiti različite heurističke funkcije.

Sažetak

- Igranje igara je problem pretraživanje stanja kod kojega se izmjenjuju potezi dvaju protivnika
- **Algoritam minimaks** pronalazi optimalnu strategiju koja **minimizira maksimalan očekivani gubitak** koji bi mogao zadati protivnik
- U stvarnosti nije moguće potpuno pretražiti stablo igre, pa umjesto toga pretražujemo do određene dubine i zatim koristimo **heurističku funkciju** da bismo procijenili vrijednosti stanja igre
- Različiti igrači koriste različite heurističke funkcije. Igrač ne poznaje protivnikovu heuristiku
- **Podrezivanje alfa-beta** smanjuje broj čvorova koje treba ispitati
- Nismo razmotrili: igre s više igrača, igre s elementom slučajnosti

Sljedeća tema: Prikazivanje znanja