Logički Agenti

U okviru lekcije

- Agenti bazirani na znanju
- Svet ljudoždera
- Logika generalno - modeli i korišćenje
- Boolova logika
- Ekvivalencija, validacija, satisfakcija
- Nasleđivanjei dokazivanje teorema
  - forward ulančavanje
  - povratno ulančavanje
  - rezolucija

Osnova znanja

- Osnova znanja = skup iskaza u formalnom jeziku
- Deklarativni pristup realizacije agenta (ili drugog sistema):
  - `Reci mu šta treba da zna`
- Tada neka `Pita` sebe šta da radi – odgovori iz baze znanja
- Agenti s emogu videti pomoću nivoa znanja
  na primer, šta znaju, nezaavisno kako je implementirano
- Ili na implementacionom niovu
  - na primer, strukture podataka u bazi znanjai algoritmi koji se primenjuju

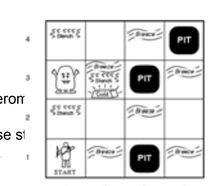
Agent baziran na znanju

```
function KB-AGENT(percept) returns an action
    static: KB, a knowledge base
            t, a counter, initially 0, indicating time
    TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
    action ← ASK(KB, MAKE-ACTION-QUERY(t))
    TELL(KB, MAKE-ACTION-SENTENCE(action, t))
    t ← t + 1
    return action
```

- Agent bi trebalo da bude u mogućnosti da
  - Prikazuje stanja, akcije, ...
  - Uključuje nove opažaje
  - Dopunjuje trenutni prikaz sveta
  - Otkriva skrivena značenja u svetu
  - Otkriva odgovarajuće akcije

Svet ljudoždera

- Performanse
  - zlato +1000, smrt -1000
  - -1 po koraku, -10 za korišćenje strele
- Okruženje
  - U kvadratima koji se graniče za ljudožderom prisustvo
  - U kvadratima do soba sa rupom oseća se st
  - Strela ubija ljudoždera ako je ispred nas
  - Postoji samo jedna strela
  - Moguće je uzeti zlato ako se nalazi u sobi
  - Moguće je ostaviti zlato

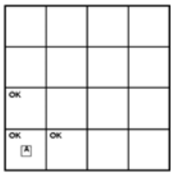


- Senzori: Zaudaranje, Strujanje, Bljesak, Udar, Vrisak
- Oruđa: Okret ulevo, Okret udesno, Napred, Uzeti, Ostaviti, Pucati

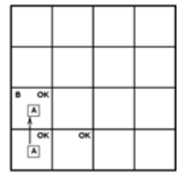
Karakteristike

- Potpuno definisan Ne – samo lokalni opažaji
- Deterministički Da – egzaktno definisani ciljevi
- Epizode Ne – sekvencijalan na nivou akcije
- Statički Da – Ljudožder i rupe se ne pomeraju
- Diskretan Da
- Pojedinačni agent? Da – Ljudožder je u suštini karakteristika okruženja

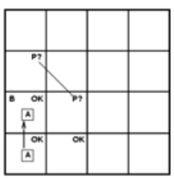
Istraživanje sveta



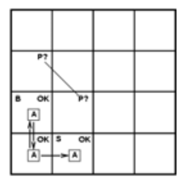
Istraživanje sveta



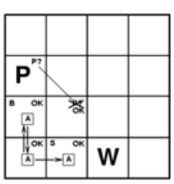
Istraživanje sveta



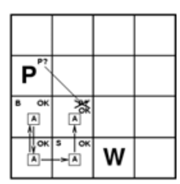
Istraživanje sveta



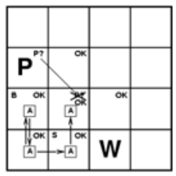
Istraživanje sveta



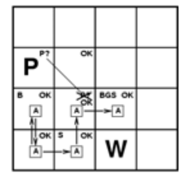
Istraživanje sveta



Istraživanje sveta



Istraživanje sveta



Logika

- Logika je formalni jezik za predstavljanje informacija na osnovu kojih se dobiti određeni zaključci
- Sintaksa definiše izraze u okviru jezika
- Semantika definiše “značenje" izraza
  - na priemr, definiše istinitost iskaza u svetu

- Aritmetika:
  - $x+2 \geq y$ je iskaz; x2+y > {} i nije
  - $x+2 \geq y$ je true ako x+2 nije manje od y
  - $x+2 \geq y$ je true u svetu gde je x = 7, y = 1
  - $x+2 \geq y$ je false u svetu gde je x = 0, y = 6

Nasleđivanje

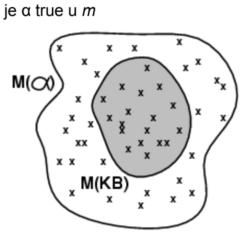
- Nasleđivanje definiše dajedna stvar sledi iz druge:

$$KB \models \alpha$$

- Baza znanja *KB* nasleđuje iskaz α ako i samo ako je α true tamo gde je i *KB*
  - x+y = 4 nasleđuje 4 = x+y
  - Nasleđivanje je relacija između iskaza (sintakse) koji su bazirani na semantici

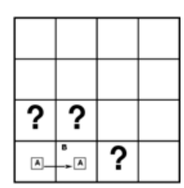
Modeli

- Logičko razmišljanje je obično pomoću modela, koji su formalno strukturirani podsvetovi
- Kaže se da *m* je model iskaza α ako je α true u *m*
- *M(α)* je skup svih modela za α
- Tada je $KB \models \alpha$ ako i samo ako je *M(KB)* $\subseteq$ *M*(α)



Nasleđivanje i ljudožder

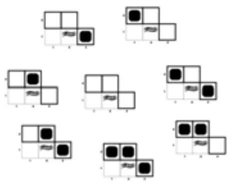
Situacija nakon nikakvog opžaja u [1,1], pomeranje desno, strujanje u [2,1]



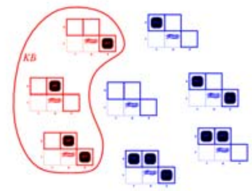
Neka mogući moduli KB razmatraju samo rupe

3 Boolean izbora ⇒ 8 mogućih modela

Mogući modeli

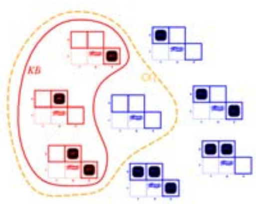


Mogući modelu



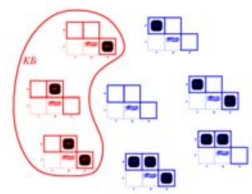
- *KB* = pravila + opažanje

Mogući modeli



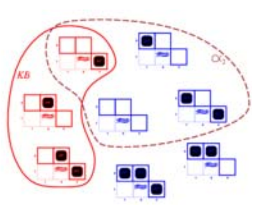
- *KB* = pravila + opažanja
- α_1 = "[1,2] je siguran", $KB \models \alpha_1$, dokazan pomoću provere modela

Mogući modeli



- *KB* = pravila + opažanja

Mogući modeli



- α_2 = "[2,2] je sigurno", $KB \not\models \alpha_2$

Zaključivanje

- $KB \vdash_i \alpha$ = iskaz α se može dobiti iz *KB* pomoću procedure *i*
- *i* je kompletna ako uvek kada je $KB \models \alpha$, tada je true i $KB \vdash_i \alpha$
- Definisana je logika (prvog reda) koja je dovoljno opisna da prikaže skoro sve što je od interesa
- Procedura bi trebala da odgovori na svako pitanje čiji se odgovor nalazi u *KB*.

Osnovna logika: sintaksa

- Prikaz osnovnih ideja
- Simboli S_1, S_2 ... su iskazi
  - Ako je S iskaz, $\neg S$ je iskaz (negacija)
  - Ako su S_1 i S_2 iskazi, $S_1 \wedge S_2$ je iskaz (konjunkcija)
  - Ako su S_1 i S_2 iskazi, $S_1 \vee S_2$ je iskaz (disjunkcija)
  - Ako su S_1 i S_2 iskazi, $S_1 \Rightarrow S_2$ je iskaz (implikacija)
  - Ako su S_1 i S_2 iskazi, $S_1 \Leftrightarrow S_2$ je iskaz (bikondicional)

Osnovna logika: Semantika

Svaki model specificira true/false za svaki simbol

$P_{1,2}$	$P_{2,2}$	$P_{3,1}$
false	true	false

Za date simbole 8 mogućih modela

Pravila za model *m*:

$\neg S$	je true iff	S je false	
$S_1 \wedge S_2$	je true iff	S_1 je true and	S_2 je true
$S_1 \vee S_2$	je true iff	S_1 je true or	S_2 je true
$S_1 \Rightarrow S_2$	je true iff	S_1 je false or	S_2 je true
na primer,	je false iff	S_1 je true and	S_2 je false
$S_1 \Leftrightarrow S_2$	je true iff	$S_1 \Rightarrow S_2$ is true and	$S_2 \Rightarrow S_1$ je true

$\neg P_{1,2} \wedge (P_{2,2} \vee P_{3,1}) = true \wedge (true \vee false) = true \wedge true = true$

Tabela istinitosti

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
$false$	$false$	$true$	$false$	$false$	$true$	$true$
$false$	$true$	$true$	$false$	$true$	$true$	$false$
$true$	$false$	$false$	$false$	$true$	$false$	$false$
$true$	$true$	$false$	$true$	$true$	$true$	$true$

Iskazi u primeru

Neka je $P_{i,j}$ true ako postoji rupa u [i, j].

Neka je $B_{i,j}$ true ako postoji strujanje u [i, j].

$\neg P_{1,1}$

$\neg B_{1,1}$

$B_{2,1}$

- "Rupe prouzrokuju strujanje u susednim kvadratima"

$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

$B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

Tabela istinitosti

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	KB	α_1
$false$	$false$	$false$	$false$	$false$	$false$	$false$	$false$	$true$
$false$	$false$	$false$	$false$	$false$	$false$	$true$	$false$	$true$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$false$	$true$	$false$	$false$	$false$	$false$	$false$	$false$	$true$
$false$	$true$	$false$	$false$	$false$	$false$	$true$	$\underline{true}$	$\underline{true}$
$false$	$true$	$false$	$false$	$false$	$true$	$false$	$\underline{true}$	$\underline{true}$
$false$	$true$	$false$	$false$	$false$	$true$	$true$	$\underline{true}$	$\underline{true}$
$false$	$true$	$false$	$false$	$true$	$false$	$false$	$false$	$true$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$true$	$true$	$true$	$true$	$true$	$true$	$true$	$false$	$false$

Nasleđivanje

```
function TT-ENTAILS?(KB, α) returns true or false
   symbols ← a list of the proposition symbols in KB and α
   return TT-CHECK-ALL(KB, α, symbols, [])

function TT-CHECK-ALL(KB, α, symbols, model) returns true or false
   if EMPTY?(symbols) then
      if PL-TRUE?(KB, model) then return PL-TRUE?(α, model)
      else return true
   else do
      P ← FIRST(symbols); rest ← REST(symbols)
      return TT-CHECK-ALL(KB, α, rest, EXTEND(P, true, model)) and
                TT-CHECK-ALL(KB, α, rest, EXTEND(P, false, model))
```

- Za *n* simbola, vremenska kompleksnost je $O(2^n)$, prostorna $O(n)$

Logička ekvivalencija

- Dva iskaza su logički ekvivalentna ako i samo ako je true u istim modelima: $\alpha \equiv \beta$ iff $\alpha \models \beta$ i $\beta \models \alpha$

$$
\begin{array}{rcll}
(\alpha \wedge \beta) & \equiv & (\beta \wedge \alpha) & \text{commutativity of } \wedge \\
(\alpha \vee \beta) & \equiv & (\beta \vee \alpha) & \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) & \equiv & (\alpha \wedge (\beta \wedge \gamma)) & \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) & \equiv & (\alpha \vee (\beta \vee \gamma)) & \text{associativity of } \vee \\
\neg(\neg \alpha) & \equiv & \alpha & \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg \beta \Rightarrow \neg \alpha) & \text{contraposition} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg \alpha \vee \beta) & \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) & \equiv & ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) & \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) & \equiv & (\neg \alpha \vee \neg \beta) & \text{de Morgan} \\
\neg(\alpha \vee \beta) & \equiv & (\neg \alpha \wedge \neg \beta) & \text{de Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) & \equiv & ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) & \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) & \equiv & ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) & \text{distributivity of } \vee \text{ over } \wedge
\end{array}
$$

Validacija i satisfakcija

Iskaz je validan ako je true u svim modelima,

, *True*, $A \vee \neg A$, $A \Rightarrow A$, $(A \wedge (A \Rightarrow B)) \Rightarrow B$

Dedukciona teorema:

$KB \models \alpha$ ako i samo ako je $(KB \Rightarrow \alpha)$ validno

Iskaz je zadovoljen ako je true u nekim modelima

$A \vee B$, C

Iskaz je nemoguć ako je true u nijednom modelu

$A \wedge \neg A$

$KB \models \alpha$ ako i samo ako je $(KB \wedge \neg \alpha)$ nemoguće

Metodi dokazivanja

- Metodi dokazivanja:
  - Pravila nasleđivanja
    - Dobijanje novih iskaza od starih
    - Dokaz = niz nalsеđivanja pravila
      Mogu se koristiti pravila nasleđivanja kao operatori kod standardnih algoritama pretraživanja
    - Obično je potrebno trasnformisati iskaze u normalnu formu
  - Provera modela
    - tabela istinitosti (eksponencijalno po *n*)
    - povratno traganje, Davis--Putnam-Logemann-Loveland (DPLL)
    - heurističko pretraživanje u prostoru stanja (moguće nekompletno)

Rezolucija

Conjunctive Normal Form (CNF)

conjunction of disjunctions of literals

clauses

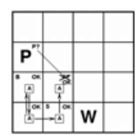
E.g., $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$

- Resolution

$$
\frac{l_i \vee \ldots \vee l_k, \qquad m_1 \vee \ldots \vee m_n}{l_i \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_k \vee m_1 \vee \ldots \vee m_{j-1} \vee m_{j+1} \vee \ldots \vee m_n}
$$

gde su l_i su m_j komplementarni literali

$$
\frac{P_{1,3} \vee P_{2,2}, \quad \neg P_{2,2}}{P_{1,3}}
$$



Rezolucija

$$
\begin{array}{r}
\neg(l_i \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_k) \Rightarrow l_i \\
\neg m_j \Rightarrow (m_1 \vee \ldots \vee m_{j-1} \vee m_{j+1} \vee \ldots \vee m_n) \\
\neg(l_i \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_k) \Rightarrow (m_1 \vee \ldots \vee m_{j-1} \vee m_{j+1} \vee \ldots \vee m_n) \\
\hline
\end{array}
$$

Konverzija u KNF

$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})\beta$

1. Eliminisati $\Leftrightarrow$, zameniti $\alpha \Leftrightarrow \beta$ sa $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$.
   $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$
2. Eliminisati $\Rightarrow$, zameniti $\alpha \Rightarrow \beta$ sa $\neg \alpha \vee \beta$.
   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$
3. Pomeriti $\neg$ unutar zagrade pomoću de Morganovih pravila
   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \vee \neg P_{2,1}) \vee B_{1,1})$
4. Primeniti distributivnost
   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

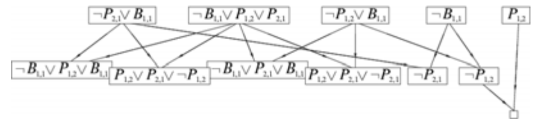
Algoritam rezolucije

- Dokazivanje negacijom, pokazati $KB \wedge \neg \alpha$ je nemoguće

```
function PL-RESOLUTION(KB, α) returns true or false
  clauses ← the set of clauses in the CNF representation of KB ∧ ¬α
  new ← { }
  loop do
     for each Ci, Cj in clauses do
        resolvents ← PL-RESOLVE(Ci, Cj)
        if resolvents contains the empty clause then return true
        new ← new ∪ resolvents
     if new ⊆ clauses then return false
     clauses ← clauses ∪ new
```

Primer

- $KB = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1}$ $\alpha = \neg P_{1,2}$



Unapred i povratno ulančavanje

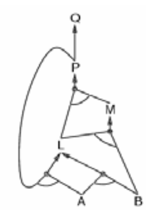
- Horn Form
  KB = conjunction od Horn klauzula
  - Horn klauzula =
    - simbol; ili
    - (konjukcija simbola) ⇒ simbol
  - $C \wedge (B \Rightarrow A) \wedge (C \wedge D \Rightarrow B)$
- Modus Ponens $\dfrac{\alpha_1, \ldots, \alpha_n, \qquad \alpha_1 \wedge \ldots \wedge \alpha_n \Rightarrow \beta}{\beta}$

- Ulančavanje
- Prirodno rešenje i linearno

Ulančavanje unapred

- Ideja: okidati ono praviločiji su uslovi zadovoljeni u *KB*
  - dodati zaključak u *KB*, do cilja

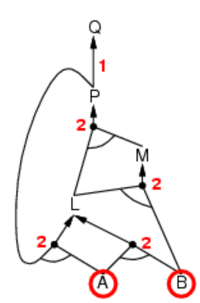
$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B



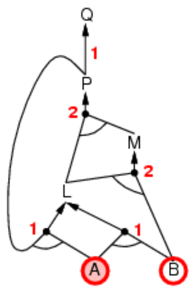
Algoritam za ulančavanje unapred

```
function PL-FC-ENTAILS?(KB, q) returns true or false
  local variables: count, a table, indexed by clause, initially the number of premises
                   inferred, a table, indexed by symbol, each entry initially false
                   agenda, a list of symbols, initially the symbols known to be true
  while agenda is not empty do
     p ← POP(agenda)
     unless inferred[p] do
        inferred[p] ← true
        for each Horn clause c in whose premise p appears do
           decrement count[c]
           if count[c] = 0 then do
              if HEAD[c] = q then return true
              PUSH(HEAD[c], agenda)
  return false
```

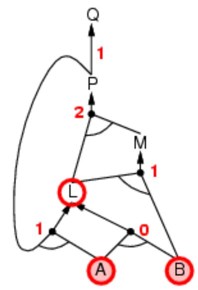
Ulančavanje unapred



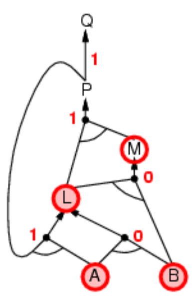
Ulančavanje unapred



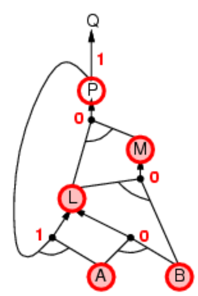
Ulančavanje unapred



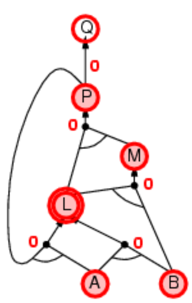
Ulančavanje unapred



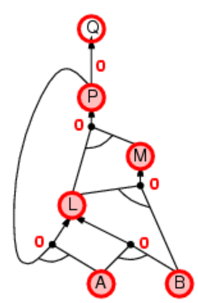
Ulančavanje unapred



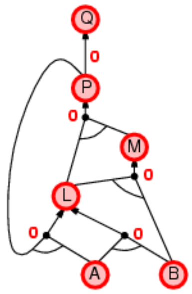
Ulančavanje unapred



Ulančavanje unapred



Ulančavanje unapred



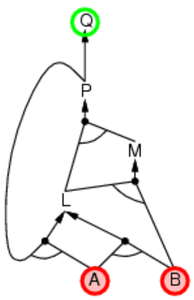
Povratno ulančavanje

Ideja: krenuti unazad od cilja
- da bi se dokazalo *q* ,
  - proveriti da li je *q* poznato, ili
  - dokazati pomoću povratnog ulančavanja sve uslove u nekom pravilu u čijem je zaključku *q*

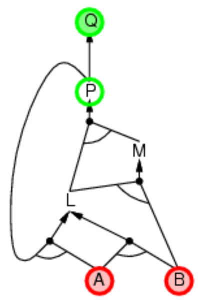
Izbeći petlje: proveriti da li je novi podcilj već na stku sa ciljevima

Izbeći ponavljanje: proveriti da li novi podcilj
1. je već dokazano true, ili
2. je već dokazano false

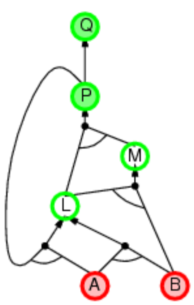
Povratno ulančavanje



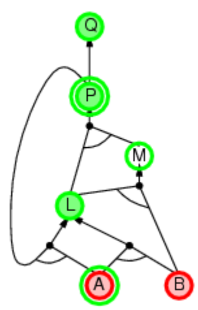
Povratno ulančavanje



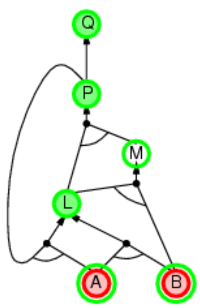
Povratno ulančavanje



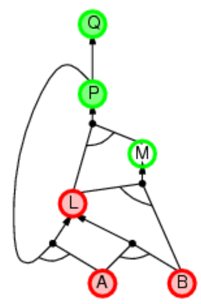
Povratno ulančavanje



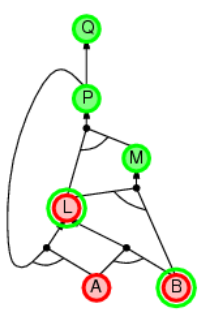
Povratno ulančavanje



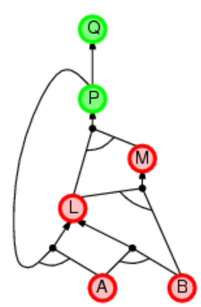
Povratno ulančavanje



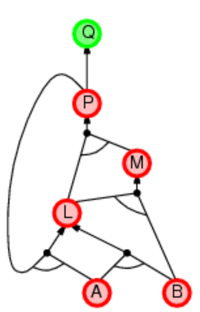
Povratno ulančavanje



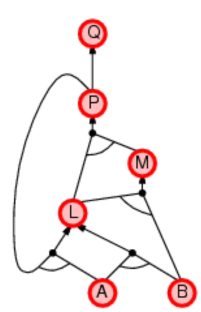
Povratno ulančavanje



Povratno ulančavanje



Povratno ulančavanje



Karakteristike ulančavanje

- Unapred je data-driven, automatskic, nesvestan proces,
  - na primer, prepoznavanje objekata, odluke o putanjama
- Možda puno rada oko nevažnih stvari za dostizanje cilja
- Povratno je goal-driven, usredsređen na rešavanje problema
- Kompleksnost povratnog može biti manja od linerane veličine KB

Svet ljudoždera

Korišćenjem logike

$\neg P_{1,1}$
$\neg W_{1,1}$
$B_{x,y} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$
$S_{x,y} \Leftrightarrow (W_{x,y+1} \vee W_{x,y-1} \vee W_{x+1,y} \vee W_{x-1,y})$
$W_{1,1} \vee W_{1,2} \vee \ldots \vee W_{4,4}$
$\neg W_{1,1} \vee \neg W_{1,2}$
$\neg W_{1,1} \vee \neg W_{1,3}$
…

$\Rightarrow$ 64 različitih simbola, 155 iskaza

```
function PL-WUMPUS-AGENT(percept) returns an action
   inputs: percept, a list, [stench, breeze, glitter]
   static: KB, initially containing the "physics" of the wumpus world
           x, y, orientation, the agent's position (init. [1,1]) and orient. (init. right)
           visited, an array indicating which squares have been visited, initially false
           action, the agent's most recent action, initially null
           plan, an action sequence, initially empty

   update x, y, orientation, visited based on action
   if stench then TELL(KB, S_x,y) else TELL(KB, ¬ S_x,y)
   if breeze then TELL(KB, B_x,y) else TELL(KB, ¬ B_x,y)
   if glitter then action ← grab
   else if plan is nonempty then action ← POP(plan)
   else if for some fringe square [i,j], ASK(KB, (¬ P_i,j ∧ ¬ W_i,j)) is true or
           for some fringe square [i,j], ASK(KB, (P_i,j ∨ W_i,j)) is false then do
        plan ← A*-GRAPH-SEARCH(ROUTE-PB([x,y], orientation, [i,j], visited))
        action ← POP(plan)
   else action ← a randomly chosen move
   return action
```

Ograničenost

- KB sadrži iskaze za svaki pojedinačni kvadrat
- Za svako *t* i svaku loakciju [*x,y*],

$L^{t}_{x,y} \wedge FacingRight^{t} \wedge Forward^{t} \Rightarrow L^{t}_{x+1,y}$

- Ubrzano povećavanje uslova