Logički agenti

Poglavlje 7

Prema slajdovima Stuarta Russella (hvala)!

Pregled

◇ Agenti bazirani na znanju (engl. Knowledge–based agents)

◇ Wumpusov svijet

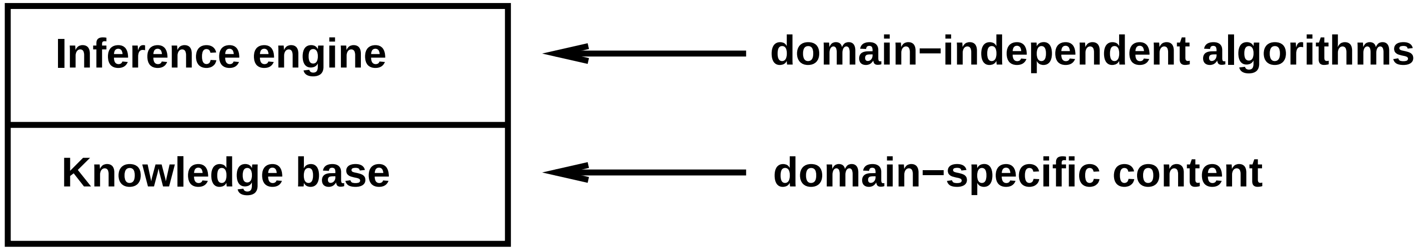
◇ Općenito o logici—modeli i relacija logičke posljedice

◇ Logika sudova (propozicijska, Boolova logika)

◇ Ekvivalencija, valjanost, ispunjivost

◇ Pravila zaključivanja i dokazivanje teorema
  - metoda rezolucije (razrješavanja) (engl. resolution)
  - ulančavanje unaprijed (engl. forward chaining)
  - ulančavanje unatrag (engl. backward chaining)

Baza znanja



Baza znanja = skup rečenica u **formalnom** jeziku

Deklarativni pristup izgradnji agenta (ili nekog drugog sustava):

KAŽI mu što treba znati (TELL)

Tada se agent može ZAPITATI (ASK) što činiti—odgovori bi trebali slijediti iz baze znanja

Agenti se mogu gledati kao nivo znanja

tj., **što znaju**, bez obzira kako se to implementira

Ili, na implementacijskoj razini,

strukture podataka u bazi znanja i algoritmi koji njima manipuliraju

Jednostavni agent baziran na znanju

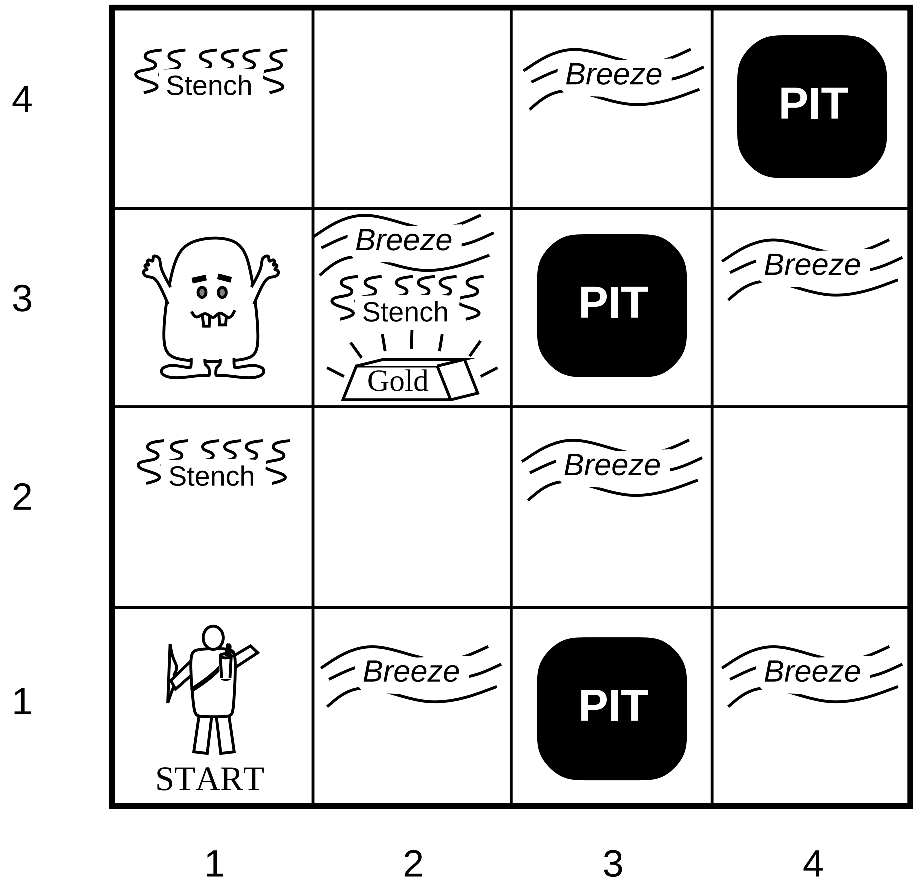
function KB-AGENT(*percept*) **returns** an *action*
  static: *KB*, a knowledge base
    t, a counter, initially 0, indicating time

  TELL(*KB*, MAKE-PERCEPT-SENTENCE(*percept*, *t*))
  action ← ASK(*KB*, MAKE-ACTION-QUERY(*t*))
  TELL(*KB*, MAKE-ACTION-SENTENCE(*action*, *t*))
  $t \leftarrow t + 1$
  return *action*

Agent mora moći:

- reprezentirati stanja, akcije i dr.
- uključivati nove percepcije
- ažurirati internu reprezentaciju svijeta
- zaključivati o skrivenim svojstvima svijeta
- zaključiti koje su odgovarajuće akcije

Wumpusov svijet — jedna mogućnost



Agent se nalazi na polju [1, 1]. Sva ostala polja su slučajna (W, P, G).
Polje je ponor (P) s vjerojatnošću 0.2

PEAS opis Wumpusovog svijeta

Mjere učinka:

zlato +1000, smrt –1000

–1 po koraku, –10 za upotrebu strelice

Okolina:

kvadrati susjedni Wumpusovom su smrdljivi

kvadrati susjedni ponoru su vjetroviti

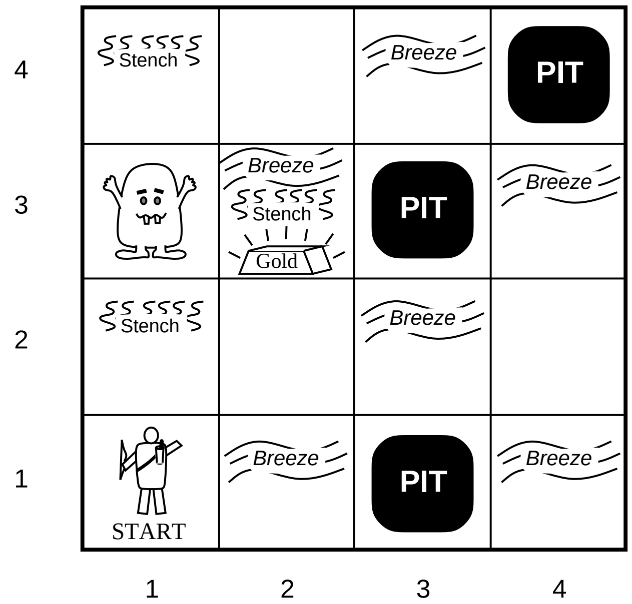
ako je zlato na kvadratu, on blista

pucanje ubija Wumpusa ako je okrenut licem

pucanje koristi samo jednu strelicu

uzimanje zlata—samo s istog kvadrata

ispuštanje zlata—samo na istom kvadratu



Dozovoljene radnje: okret lijevo/desno, naprijed, uzmi, ispusti, pucaj, izlazak iz pećine (samo s kvadrata = polja [1, 1])

Senzori za: vjetrovitost, blistanje, smrad + bum u zid + urlik Wumpusa

Karakterizacija Wumpusovog svijeta

Opažanje??

Karakterizacija Wumpusovog svijeta

Opažanje?? Nema—samo lokalna percepcija

Deterministički??

Karakterizacija Wumpusovog svijeta

Opažanje?? Nema—samo lokalna percepcija

Deterministički?? Da—ishodi su točno specificirani

Epizode??

Karakterizacija Wumpusovog svijeta

Opažanje?? Nema—samo lokalna percepcija

Deterministički?? Da—ishodi su točno specificirani

Epizode?? Ne—sekvencijalan na nivou akcije

Statičan??

Karakterizacija Wumpusovog svijeta

Opažanje?? Nema—samo lokalna percepcija

Deterministički?? Da—ishodi su točno specificirani

Epizode?? Ne—sekvencijalan na nivou akcije

Statičan?? Da—Wumpus i ponori se ne miču

Diskretan??

Karakterizacija Wumpusovog svijeta

Opažanje?? Nema—samo lokalna percepcija

Deterministički?? Da—ishodi su točno specificirani

Epizode?? Ne—sekvencijalan na nivou akcije

Statičan?? Da—Wumpus i ponori se ne miču

Diskretan?? Da

Samo jedan-agent??

Karakterizacija Wumpusovog svijeta

Opažanje?? Nema—samo lokalna percepcija

Deterministički?? Da—ishodi su točno specificirani

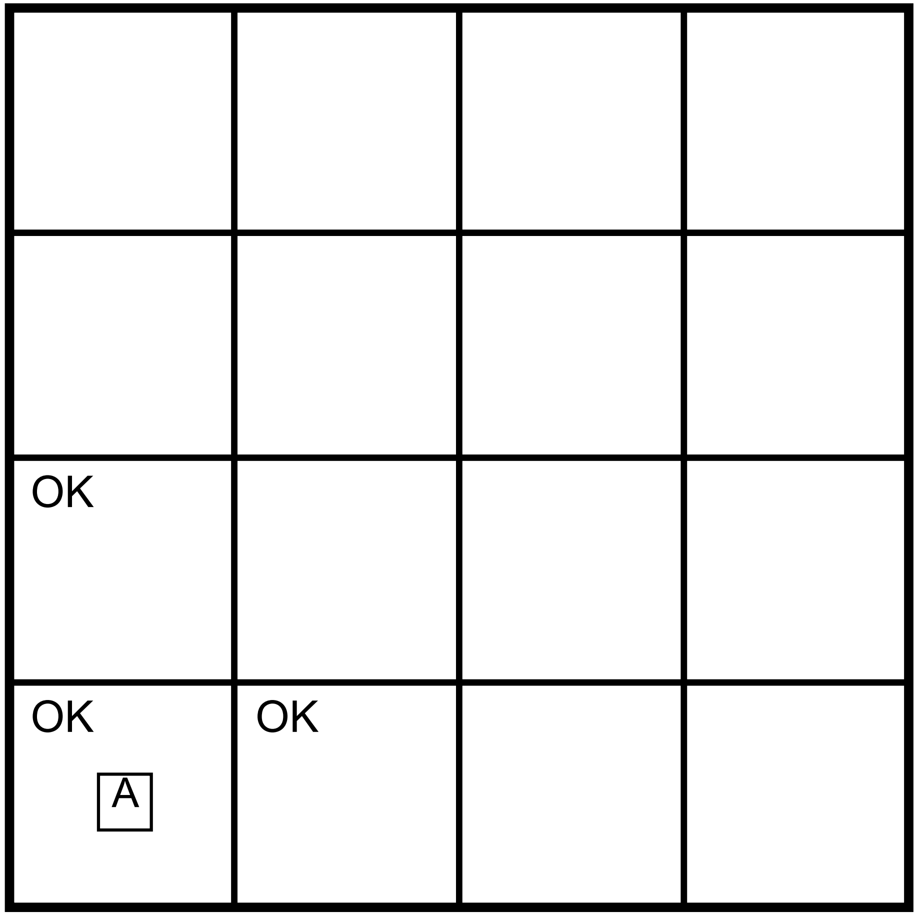
Epizode?? Ne—sekvencijalan na nivou akcije

Statičan?? Da—Wumpus i ponori se ne miču

Diskretan?? Da

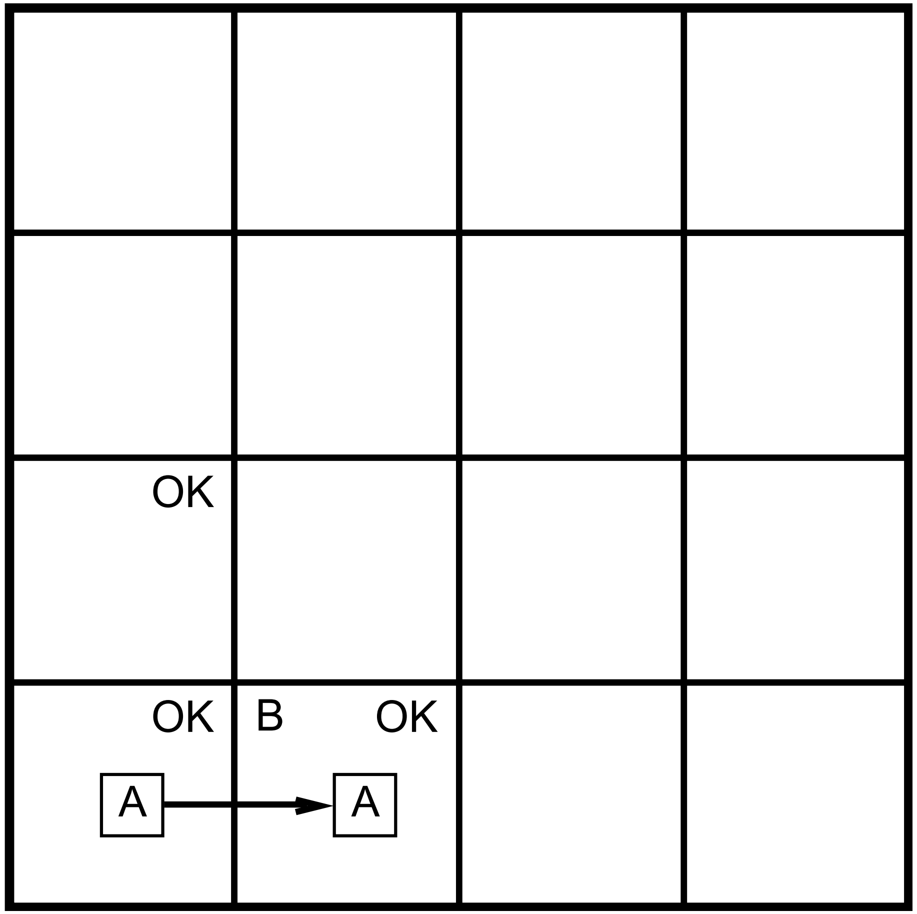
Samo jedan-agent?? Da—Wumpus je u biti fizička značajka
(ne djeluje samostalno, tj. on je kao druga vrsta ponora)

Istraživanje Wumpusovog svijeta



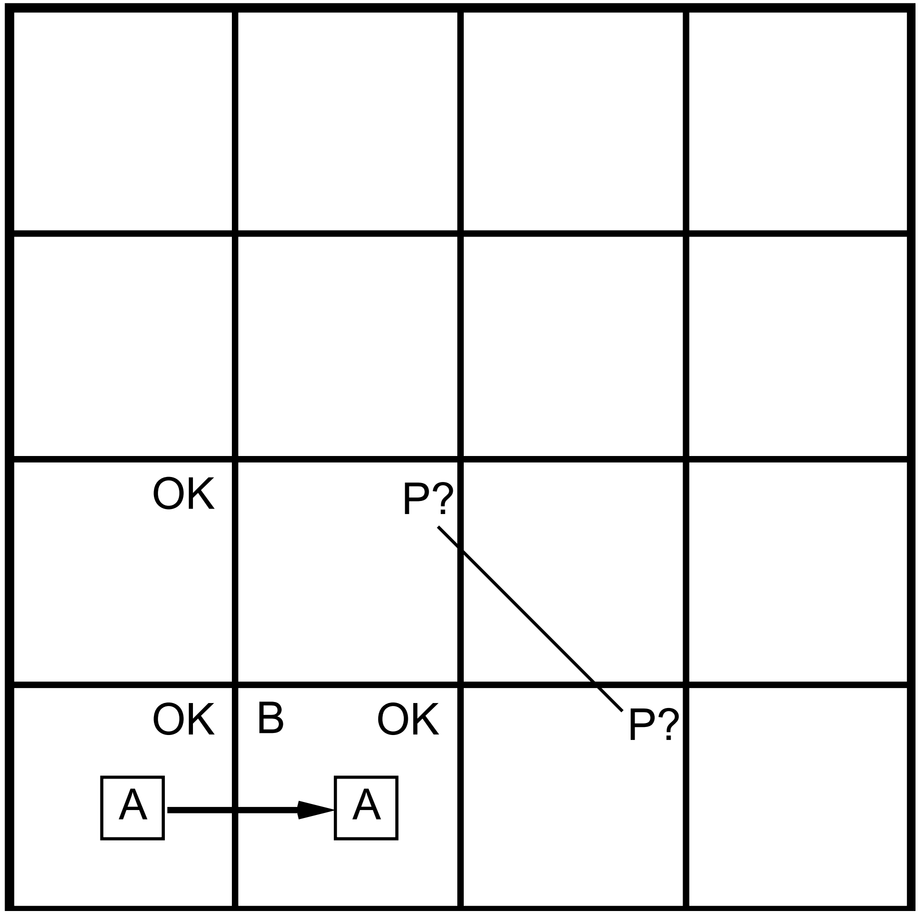
Nema vjetra na [1, 1] $\Longrightarrow$ polja [1, 2] i [2, 1] su OK
Idemo (na pr.) na [2, 1]

Istraživanje Wumpusovog svijeta



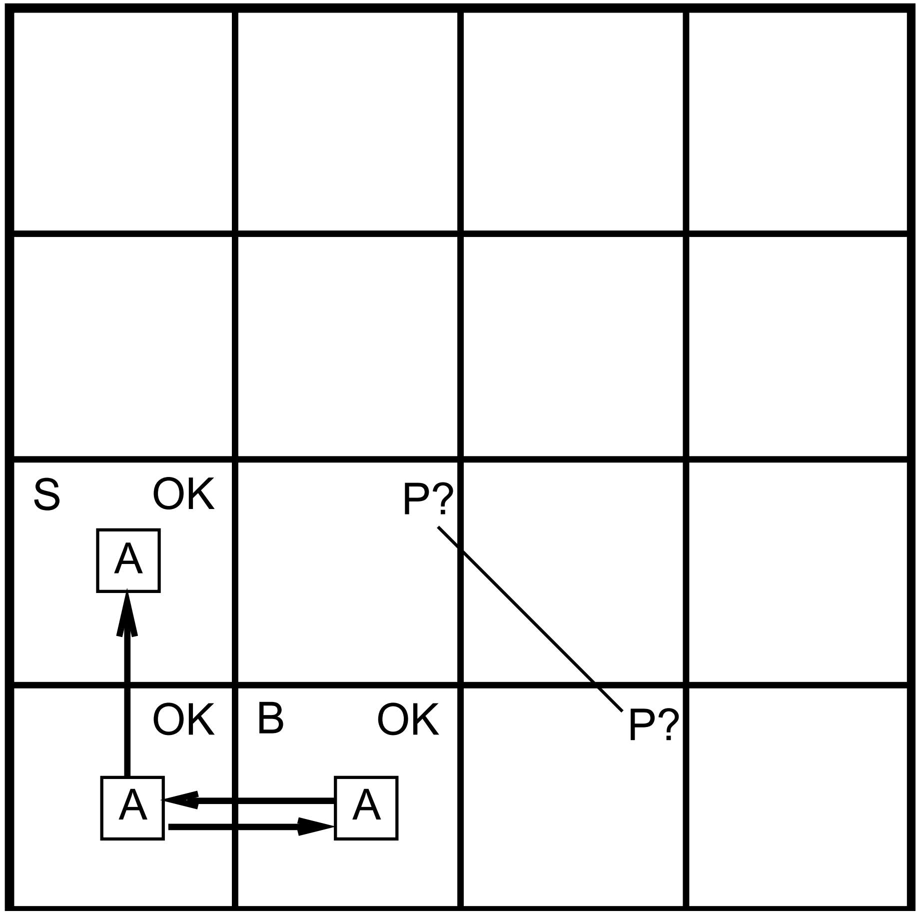
Vjetar na [2, 1] $\Longrightarrow$ . . .

Istraživanje Wumpusovog svijeta



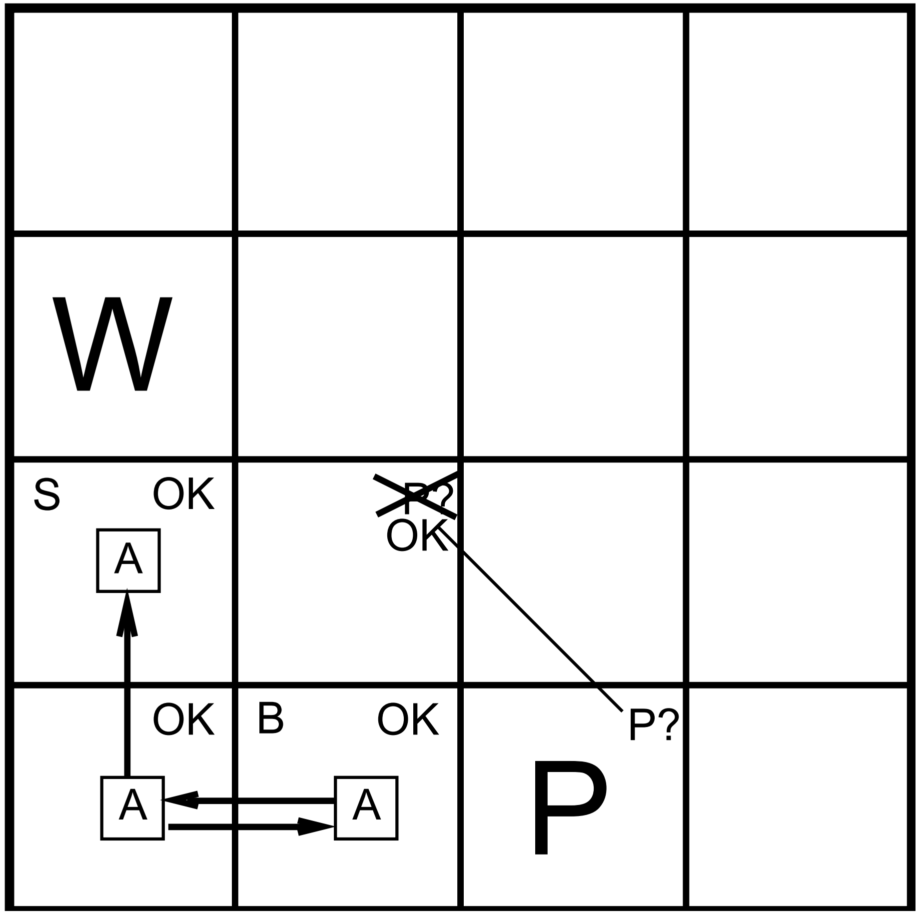
Vjetar na [2, 1] $\Longrightarrow$ na [3, 1] ili [2, 2] je ponor (ne mogu dalje/gore)
Vrati se na [1, 1] (dva okreta + naprijed), okret i idi na [1, 2]

Istraživanje Wumpusovog svijeta



Smrad na [1, 2] $\Longrightarrow$ . . .

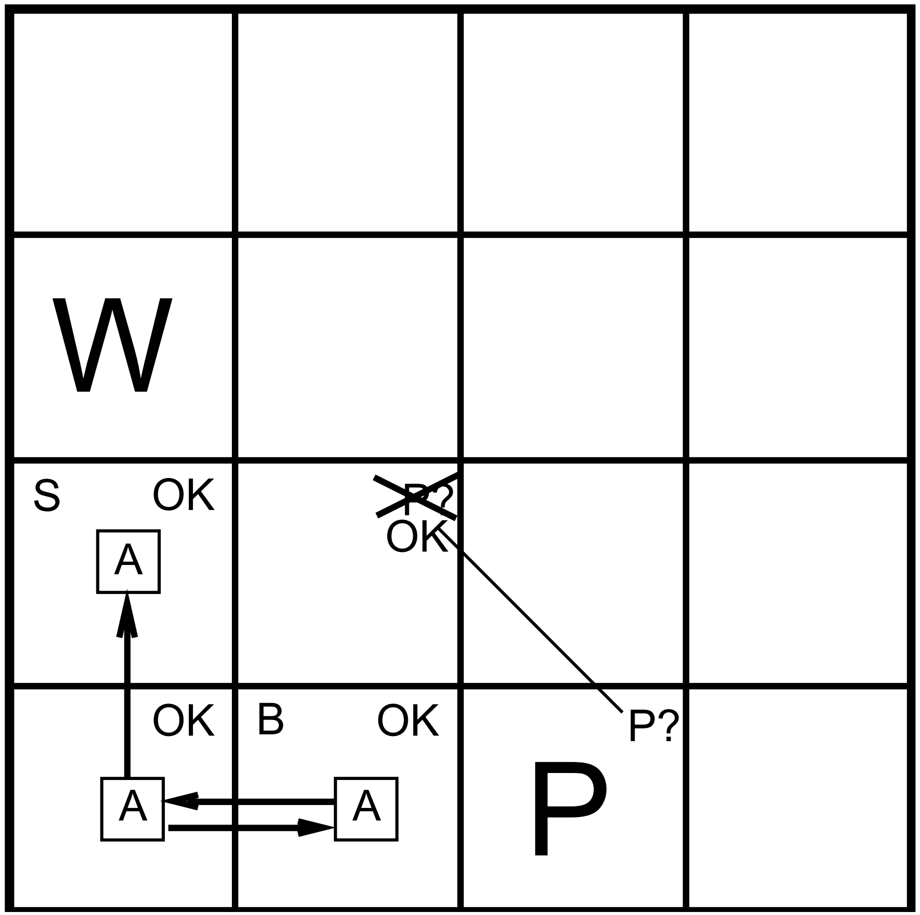
Istraživanje Wumpusovog svijeta



Smrad na [1, 2] $\Longrightarrow$ Wumpus je na [1, 3]

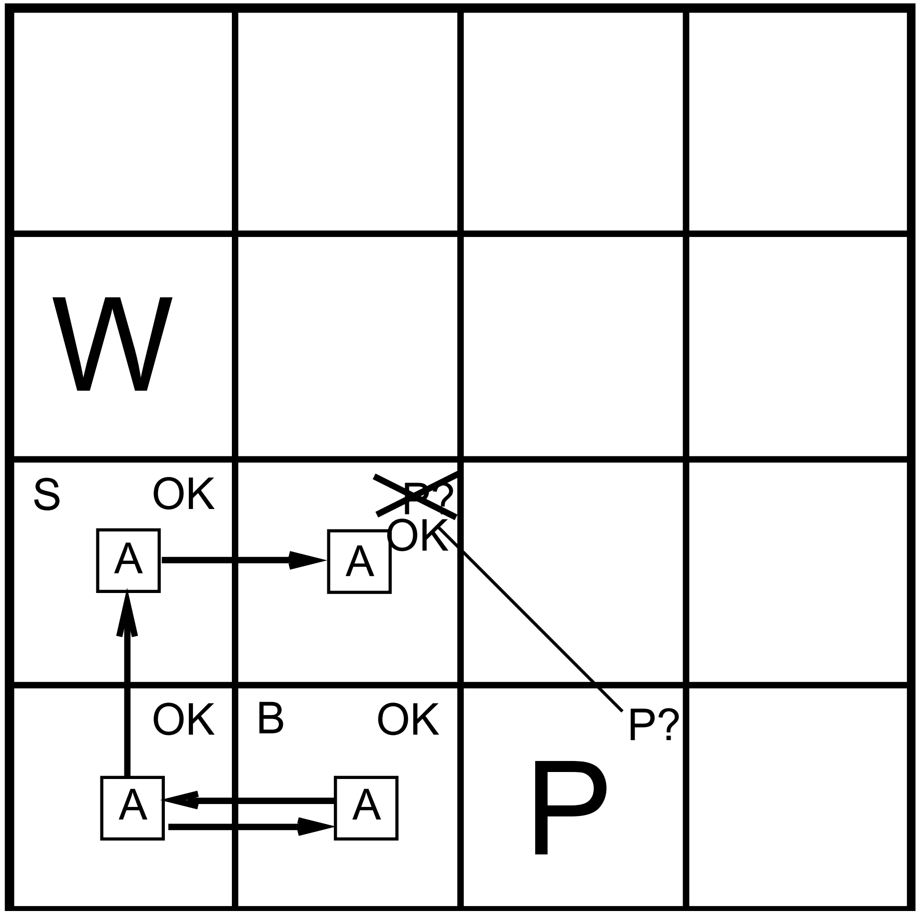
Ne može biti na [2, 2], jer bi smrdilo i na [2, 1], a ne smrdi tamo

Istraživanje Wumpusovog svijeta — na istom jestu



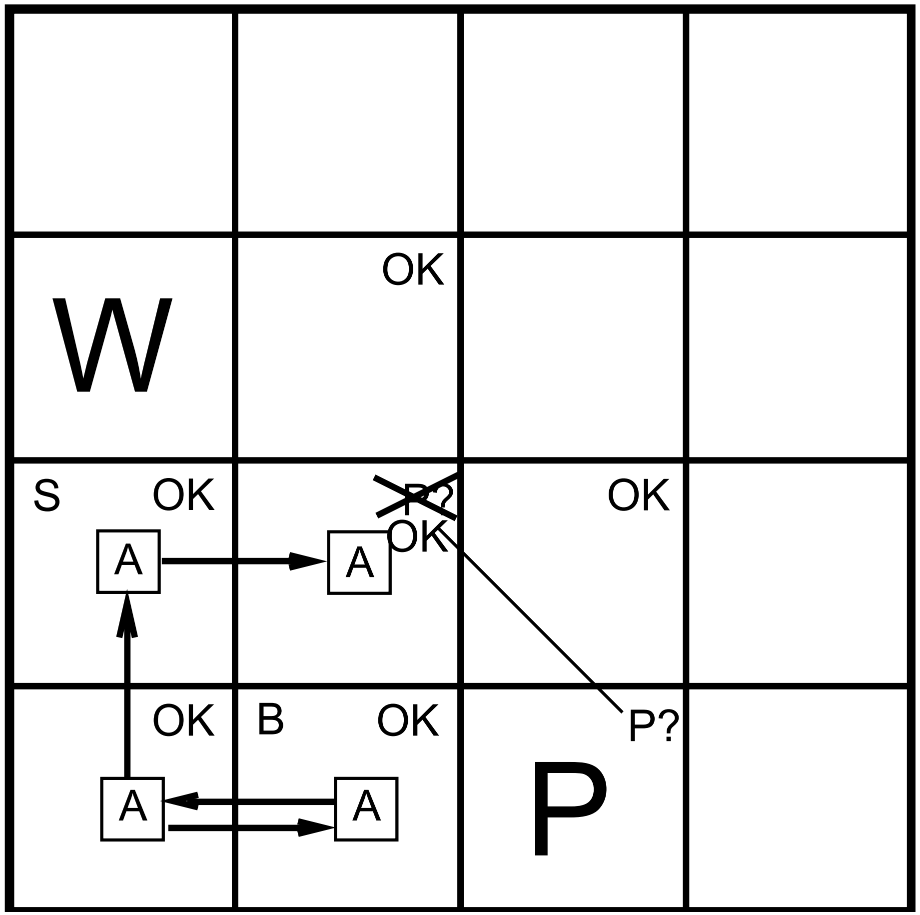
Nema vjetra na [1, 2] $\Longrightarrow$ Ponor nije na [2, 2], nego na [3, 1]
Dakle, [2, 2] je OK, idemo tamo (okret, naprijed)

Istraživanje Wumpusovog svijeta



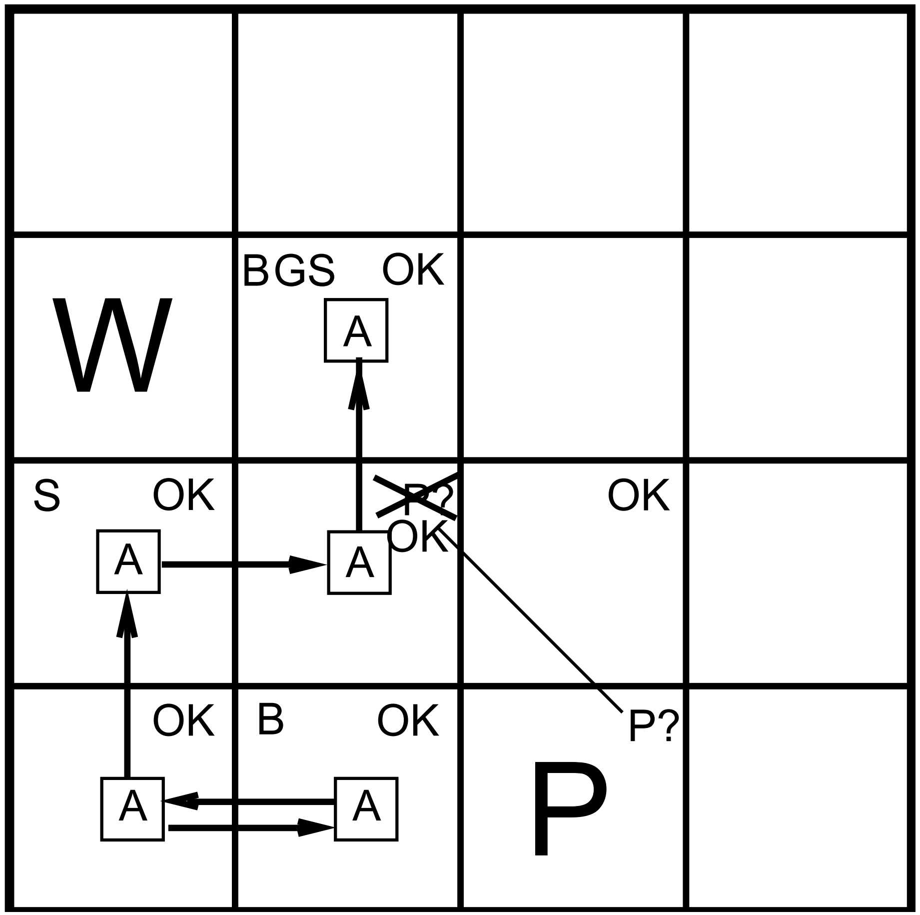
Nema ničega na [2, 2] — ni vjetra, ni smrada, ni blistanja $\Longrightarrow$

Istraživanje Wumpusovog svijeta



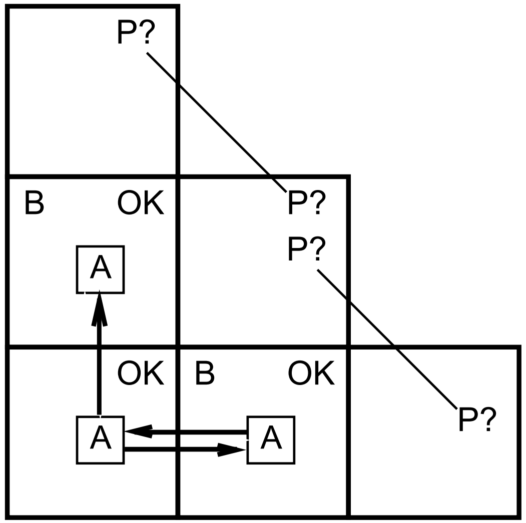
Nema ničega na [2, 2] $\Longrightarrow$ polja [2, 3] i [3, 2] su OK.
Idemo (na pr.) gore na [2, 3] (okret, naprijed)

Istraživanje Wumpusovog svijeta



Ima svega na [2, 3] — vjetar, smrad, blistanje $\Longrightarrow$
Pusti Wumpusa (strelica košta –10), uzmi zlato, idi natrag na [1, 1]!

Neke nezgodne točke

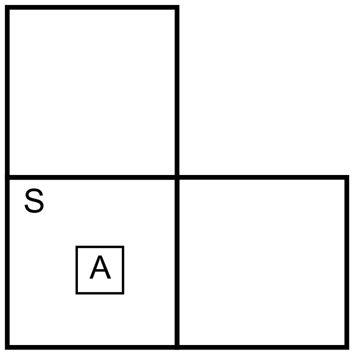


Vjetar na [1, 2] i [2, 1]

$\Rightarrow$ nema sigurne akcije

Uz pretpostavku da su ponori uniformno distribuirani,

[2, 2] ima ponor s vjerojatnošću 0.86, nasuprot 0.31



Smrad na [1, 1]

$\Rightarrow$ ne može se sigurno pomaknuti

Može uzeti strategiju prisile:

pucati ravno naprijed

Wumpus je bio tamo $\Rightarrow$ mrtav $\Rightarrow$ siguran

Wumpus nije bio tamo $\Rightarrow$ siguran

Općenito o logici

Logika je formalni jezik za reprezentiranje informacija
takvih da se mogu izvesti zaključci

Sintaksa definira rečenice u jeziku

Semantika definira "značenje" rečenica;
tj., definira istinitost rečenice u svijetu

Na primjer, jezik je aritmetika

$x + 2 \geq y$ je rečenica; $x2 + y >$ nije rečenica

$x + 2 \geq y$ je istinita ako i samo ako broj $x + 2$ nije manji od broja y

$x + 2 \geq y$ je istinita u svijetu u kojem vrijedi $x = 7,\ y = 1$
$x + 2 \geq y$ je lažna u svijetu u kojem vrijedi $x = 0,\ y = 6$

Relacija logičke posljedice

Logička posljedica znači da jedna stvar **logički slijedi iz** druge:

$$KB \models \alpha$$

Iz baze znanja KB logički slijedi α
ako i samo ako je
α istinita u svim svjetovima u kojima je KB istinita

Npr., iz baze znanja koja sadrži "Dinamo je pobijedio" i "Hajduk je pobijedio" logički slijedi "Dinamo je pobijedio ili Hajduk je pobijedio"

Npr., iz $x + y = 4$ logički slijedi $4 = x + y$

Logička posljedica je relacija među rečenicama (tj., **sintaksa**) koja je utemeljena na **semantici**

Napomena: mozak obrađuje **sintaksu** (neke vrste)

Modeli

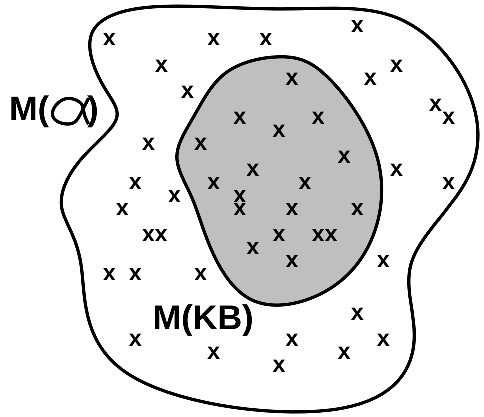
Logičari obično razmišljaju u terminima modela, tj. formalno strukturiranih svjetova u odnosu na koje možemo ustanoviti istinitost

Kažemo da je m model za rečenicu α ako je α istinita u m

$M(\alpha)$ je skup svih modela od α

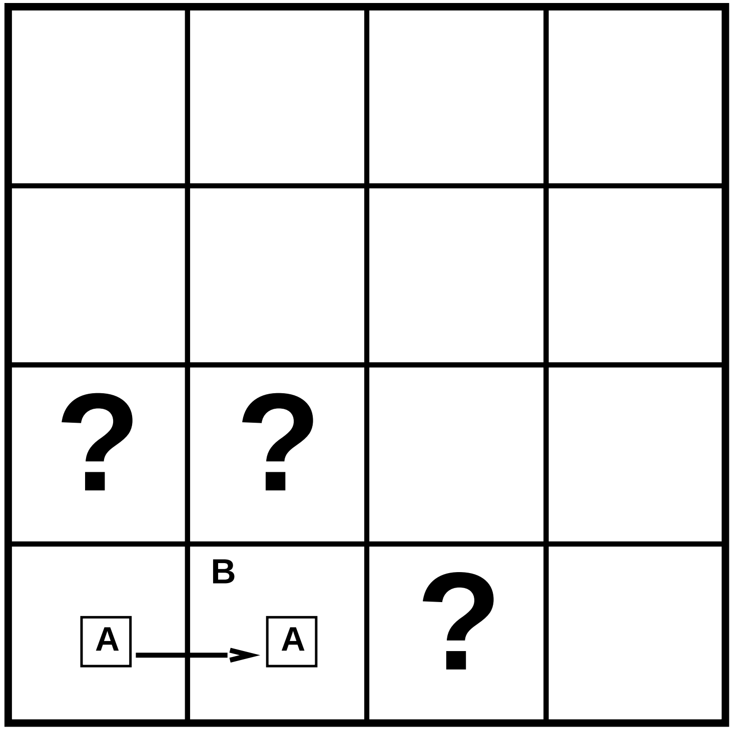
Tada, $KB \models \alpha$ ako i samo ako $M(KB) \subseteq M(\alpha)$

Na primjer,
$KB =$ Dinamo je pobijedio i Hajduk je pobijedio
$\alpha =$ Dinamo je pobijedio



Logička posljedica u Wumpusovom svijetu

Situacija nakon
- Nema vjetra (ničega) na [1, 1],
- idi desno na [2, 1],
- Vjetar na [2, 1]

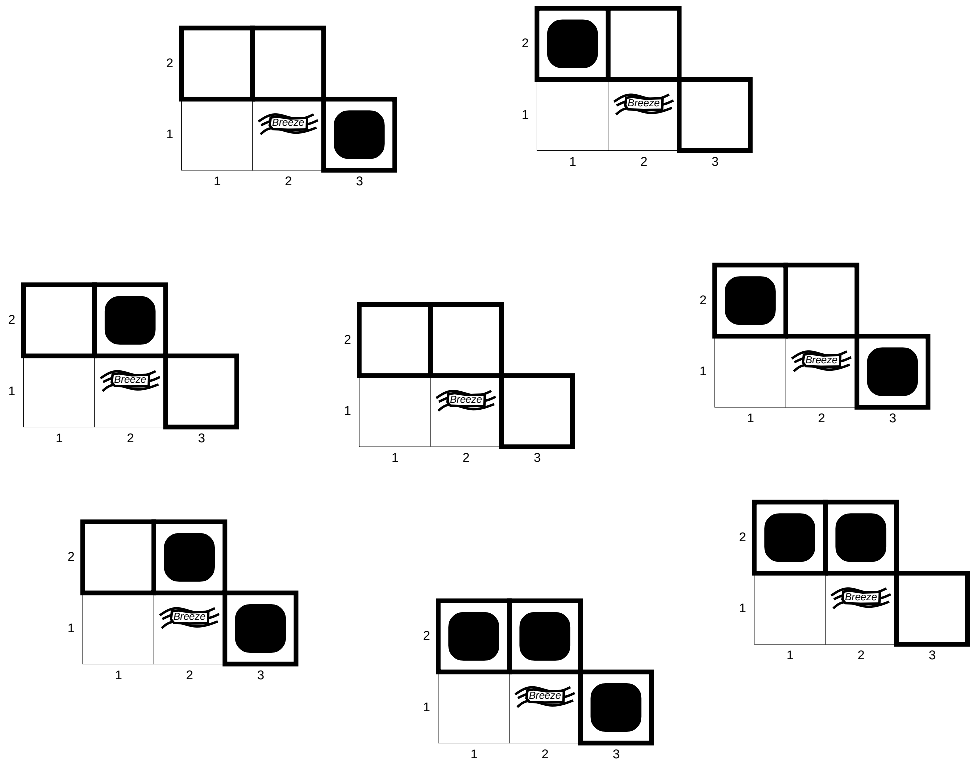


Razmotrimo moguće modele za tri "?"

Pretpostavimo samo ponore (P)

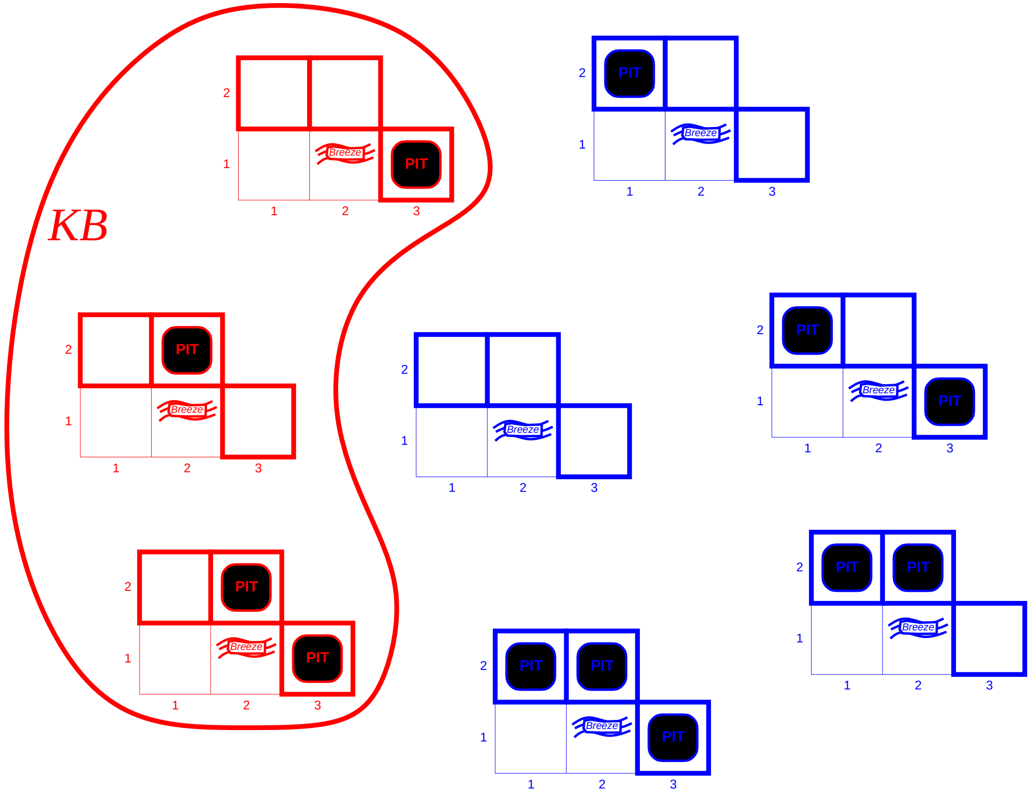
3 Booleovska izbora (ima P/nema P) $\Rightarrow$ 8 mogućih modela

Wumpusovi modeli



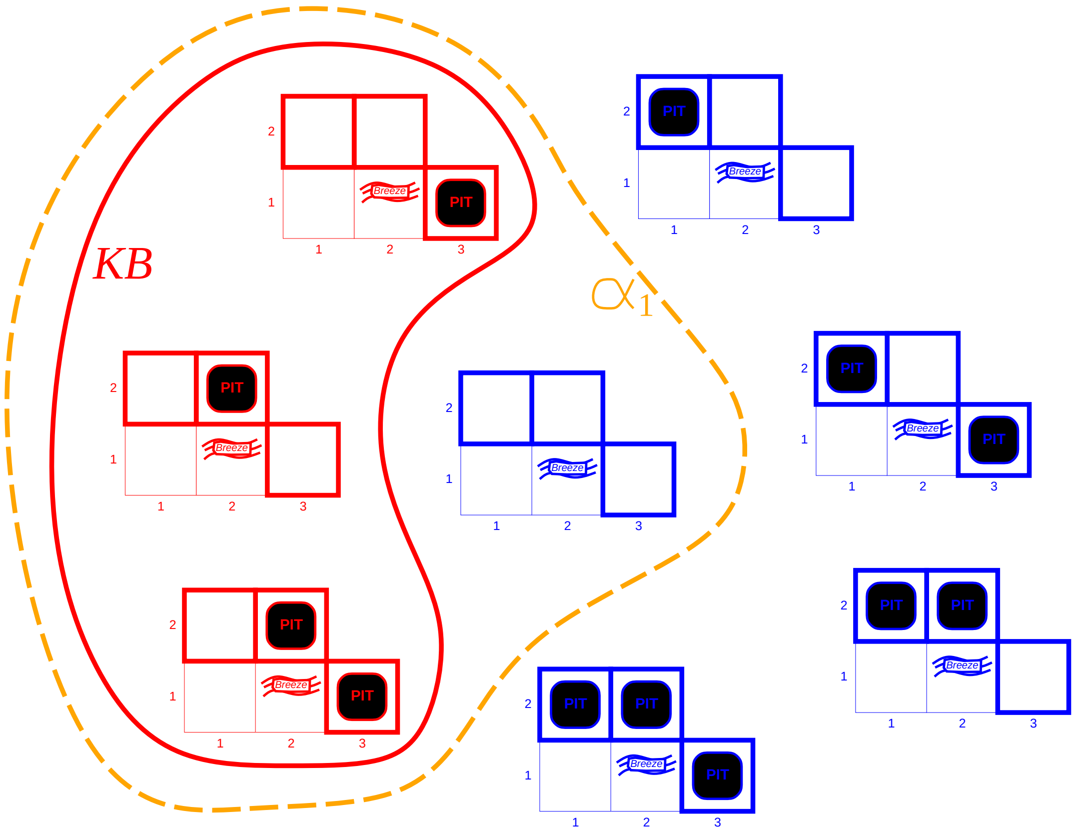
Imamo ovih 8 modela

Wumpusovi modeli



KB = pravila Wumpusovog svijeta + opažanja do tada

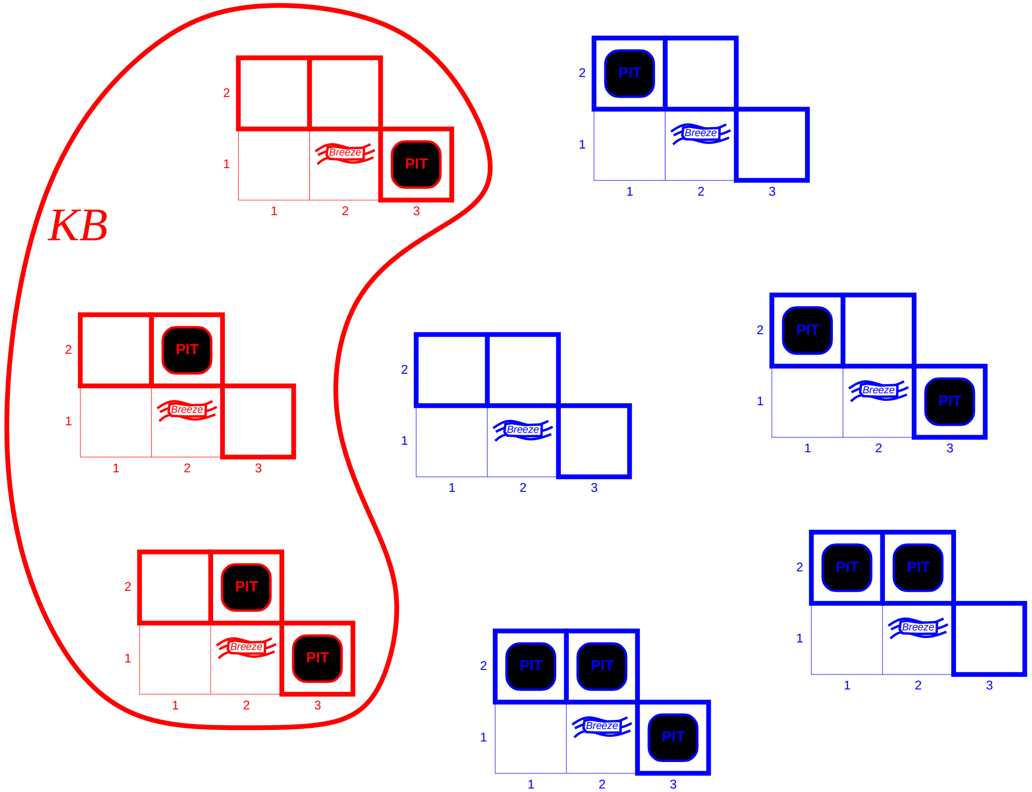
Wumpusovi modeli



KB = pravila Wumpusovog svijeta + opažanja do tada

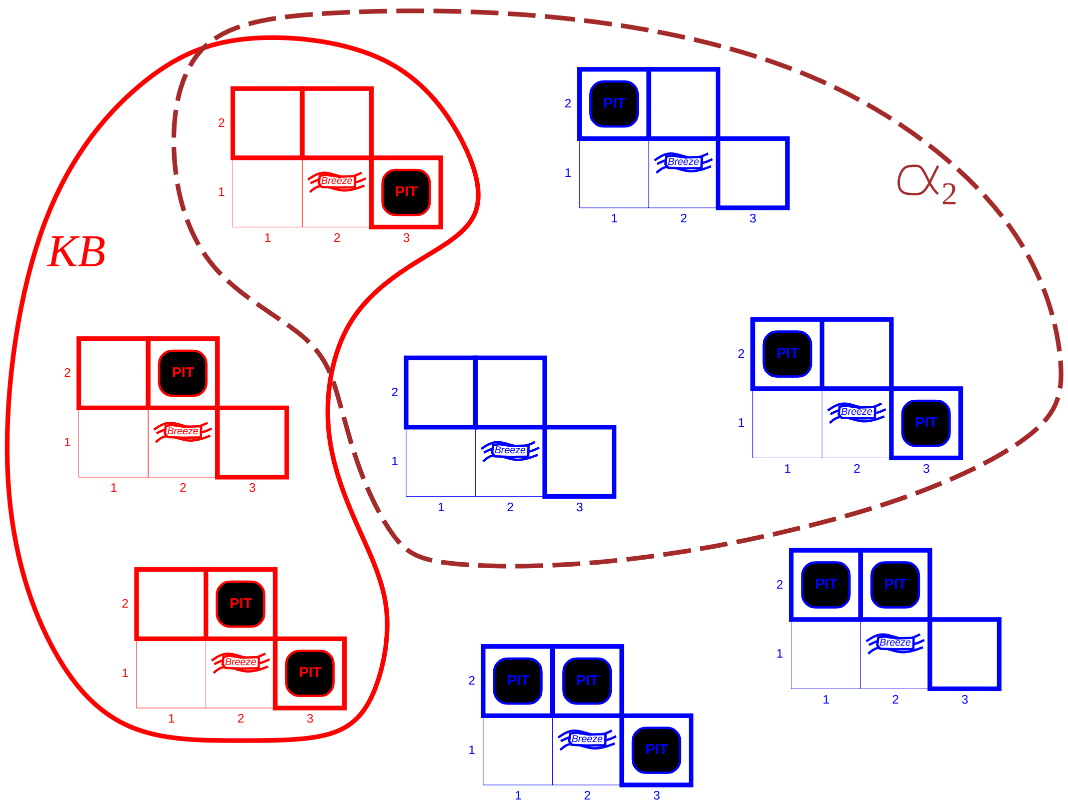
α_1 = "[1, 2] nema ponor", $KB \models \alpha_1$, dokazano provjerom modela

Wumpusovi modeli



KB = pravila Wumpusovog svijeta + opažanja do tada

Wumpusovi modeli



KB = pravila Wumpusovog svijeta + opažanja do tada

α_2 = “[2, 2] nema ponor”, $KB \not\models \alpha_2$

Može ga biti/ne mora ga biti (nema zaključka)!

Zaključivanje

$KB \vdash_i \alpha$ = rečenica α može biti izvedena iz KB sustavom pravila i

Posljedice od KB su "plast sijena"; α je "igla".
Logička posljedica = "igla u plastu"; zaključivanje = nalaženje "igle"

Ispravnost: i je ispravan (adekvatan = čuva istinitost) ako
kad god je $KB \vdash_i \alpha$, također vrijedi $KB \models \alpha$

Potpunost: i je potpun ako
kad god je $KB \models \alpha$, također vrijedi $KB \vdash_i \alpha$

"Pogled unaprijed": Definirat ćemo logiku (logiku prvog reda) koja je dovoljno izražajna da iskaže skoro sve što nas zanima, i za koju postoji ispravan i potpun sustav pravila zaključivanja.

Tj., pravila će odgovoriti na svako pitanje čiji odgovor slijedi iz onoga što znamo u KB.

Logika sudova: sintaksa

Logika sudova je najjednostavnija logika—ilustrira osnovne ideje

Propozicionalne varijable P_1, P_2 su rečenice

Ako je S rečenica, $\neg S$ je rečenica (negacija)

Ako su S_1 i S_2 rečenice, $S_1 \wedge S_2$ je rečenica (konjunkcija)

Ako su S_1 i S_2 rečenice, $S_1 \vee S_2$ je rečenica (disjunkcija)

Ako su S_1 i S_2 rečenice, $S_1 \Rightarrow S_2$ je rečenica (implikacija, kondicional)

Ako su S_1 i S_2 rečenice, $S_1 \Leftrightarrow S_2$ je rečenica (bikondicional)

Logika sudova: semantika

Svaki model određuje istinitost za svaku propozicionalnu varijablu

Na primjer:

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

(S tim varijablama može se automatski pobrojati 8 mogućih modela.)

Pravila za određivanje istinitosti obzirom na model m:

$\neg S$	je istinita akko	S	je lažna		
$S_1 \wedge S_2$	je istinita akko	S_1	je istinita **i**	S_2	je istinita
$S_1 \vee S_2$	je istinita akko	S_1	je istinita **ili**	S_2	je istinita
$S_1 \Rightarrow S_2$	je istinita akko	S_1	je lažna **ili**	S_2	je istinita
tj.,	je lažna akko	S_1	je istinita **i**	S_2	je lažna
$S_1 \Leftrightarrow S_2$	je istinita akko	$S_1 \Rightarrow S_2$	je istinita **i**	$S_2 \Rightarrow S_1$	je istinita

Jednostavnom rekurzivnom primjenom ovih pravila računa se istinitost bilo koje rečenice

Tablice istinitosti za veznike

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*

Rekurzivnom primjenom tablice računa se istinitost bilo koje rečenice

Na primjer: $\begin{array}{ccc} P_{1,2} & P_{2,2} & P_{3,1} \\ false & false & true \end{array}$

pa za rečenicu $\neg P_{1,2} \wedge (P_{2,2} \vee P_{3,1})$ dobivamo

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

Rečenice Wumpusovog svijeta — opis svijeta

KB za svijet Wumpusa = stvari koje su fiksne u svijetu
(varijabilnost/ovisnost tek u logici prvog reda)

Treba opisati stanje svakog polja (kvadrata) obzirom na osnovne stvari
P = ponor, B = vjetar (breeze), S = smrad (stench),
W = Wumpus, G = blistanje (glitter) za zlato

Neka je $P_{i,j}$ istinita ako postoji ponor na $[i, j]$.
Neka je $B_{i,j}$ istinita ako postoji vjetar na $[i, j]$.
Neka je $S_{i,j}$ istinita ako postoji smrad na $[i, j]$.
Neka je $W_{i,j}$ istinita ako postoji Wumpus na $[i, j]$.
Neka je $G_{i,j}$ istinita ako postoji blistanje na $[i, j]$.

U nastavku gledamo samo početno zaključivanje o ponorima (P) na osnovu opažanja vjetra (B).

Rečenice Wumpusovog svijeta

Neka je $P_{i,j}$ istinita ako postoji ponor na $[i, j]$.
Neka je $B_{i,j}$ istinita ako postoji vjetar na $[i, j]$.

Rečenice označavamo s R_k za kasniji poziv na nju

1. Početno stanje svijeta = nema ponora na polju [1, 1]:

$$R_1 : \qquad \neg P_{1,1}$$

Rečenice Wumpusovog svijeta

Neka je $P_{i,j}$ istinita ako postoji ponor na $[i, j]$.
Neka je $B_{i,j}$ istinita ako postoji vjetar na $[i, j]$.

Rečenice označavamo s R_k za kasniji poziv na nju

1. Početno stanje svijeta = nema ponora na polju [1, 1]:

$$R_1: \qquad \neg P_{1,1}$$

2. Ponori uzrokuju vjetar na susjednim poljima, tj.
   Na danom polju je vjetar ako i samo ako je ponor na susjednom polju

$$R_2: \qquad B_{1,1} \;\Leftrightarrow\; (P_{1,2} \vee P_{2,1})$$
$$R_3: \qquad B_{2,1} \;\Leftrightarrow\; (P_{1,1} \vee P_{2,2} \vee P_{3,1})$$

Rečenice Wumpusovog svijeta

Neka je $P_{i,j}$ istinita ako postoji ponor na $[i, j]$.
Neka je $B_{i,j}$ istinita ako postoji vjetar na $[i, j]$.

Rečenice označavamo s R_k za kasniji poziv na nju

1. Početno stanje svijeta = nema ponora na polju [1, 1]:

$$R_1 : \qquad \neg P_{1,1}$$

2. Ponori uzrokuju vjetar na susjednim poljima, tj.
   Na danom polju je vjetar ako i samo ako je ponor na susjednom polju

$$\begin{aligned} R_2 : &\qquad B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1}) \\ R_3 : &\qquad B_{2,1} \Leftrightarrow (P_{1,1} \lor P_{2,2} \lor P_{3,1}) \end{aligned}$$

3. Dodajmo još konkretna opažanja agenta za prva dva polja

$$\begin{aligned} R_4 : &\qquad \neg B_{1,1} \\ R_5 : &\qquad B_{2,1} \end{aligned}$$

Tablice istinitosti za zaključivanje

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	R_1	R_2	R_3	R_4	R_5	KB
false	*false*	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*false*	*false*
false	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*false*	*true*	*false*	*false*
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
false	*true*	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*false*	*true*	*true*	*false*
false	*true*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*false*	*true*	*false*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*true*	*false*	*false*	*true*	*false*	*false*	*true*	*true*	*false*
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
true	*true*	*true*	*true*	*true*	*true*	*true*	*false*	*true*	*true*	*false*	*true*	*false*

Pobrojimo sve retke = sve različite vrijednosti istinitosti za varijable,
ovdje je 128 redaka za 7 varijabli.

KB je istinita u nekom retku $\leftrightarrow$ $R_1, \ldots, R_5$ istinite = samo 3 retka

Provjerimo je li i α istinita na tim mjestima
$P_{1,2}$ nije = nema ponora na [1, 2], $P_{2,2}$ može/ne mora biti

Zaključivanje pobrojavanjem

Pobrojavanje prvo-u-dubinu (DFS) svih modela je ispravno i potpuno

```
function TT-Entails?(KB, α) returns true or false
    inputs: KB, the knowledge base, a sentence in propositional logic
            α, the query, a sentence in propositional logic

    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))
```

$O(2^n)$ za n varijabli; problem je **co-NP–potpun**

Logička ekvivalencija

Dvije rečenice su logički ekvivalentne akko su istinite na istim modelima:

$\alpha \equiv \beta$ ako i samo ako $\alpha \models \beta$ i $\beta \models \alpha$

Standardne logičke ekvivalencije = "zakoni":

$(\alpha \wedge \beta) \equiv (\beta \wedge \alpha)$	komutativnost $\wedge$
$(\alpha \vee \beta) \equiv (\beta \vee \alpha)$	komutativnost $\vee$
$((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma))$	asocijativnost $\wedge$
$((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma))$	asocijativnost $\vee$
$\neg(\neg\alpha) \equiv \alpha$	eliminacija dvostruke negacije
$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$	kontrapozicija
$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta)$	eliminacija implikacije
$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$	eliminacija bikondicionala
$\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$	De Morgan
$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta)$	De Morgan
$(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma))$	distributivnost $\wedge$ prema $\vee$
$(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$	distributivnost $\vee$ prema $\wedge$

Valjanost i ispunjivost

Rečenica je valjana (tautologija) ako je istinita na **svakom** modelu,
npr., $True$, $A \vee \neg A$, $A \Rightarrow A$, $(A \wedge (A \Rightarrow B)) \Rightarrow B$

Valjanost je vezana uz zaključivanje teoremom dedukcije:
$KB \models \alpha$ ako i samo ako je $(KB \Rightarrow \alpha)$ valjana

Rečenica je ispunjiva ako je istinita na **nekom** modelu (problem SAT)
npr., $A \vee B$, C

Rečenica je neispunjiva (antitautologija, kontradikcija) ako nije istinita **niti na jednom** modelu
npr., $A \wedge \neg A$ (zakon kontradikcije)

Ispunjivost je vezana uz zaključivanje na sljedeći način:
$KB \models \alpha$ ako i samo ako $(KB \wedge \neg \alpha)$ je neispunjiva
tj., dokaži α pomoću *svođenja na kontradikciju*

Metode dokazivanja

Metode dokazivanja dijele se (okvirno) u dvije vrste:

Primjena pravila zaključivanja

– Legitimno (tj. ispravno) stvaranje novih rečenica iz već postojećih

– Dokaz = niz primjena pravila zaključivanja

Pravila zaključivanja mogu se koristiti kao operatori u standardnim algoritmima pretrage

– Uobičajeno zahtijeva prevođenje rečenica u normalnu formu

Provjera modela (model checking)

pobrojavanje tablice istinitosti (uvijek eksponencijalno u n)

unaprijeđeni backtracking, tj. Davis–Putnam–Logemann–Loveland

heuristička pretraga prostora modela (ispravno, ali nepotpuno)

na pr., min-conflicts-like hill-climbing algoritmi

Rezolucija (razlučivanje/razrješavanje)

v. FER, UI–5, str. 42–61.

Primjer 2 = Diplomatski problem — treba str. 33–34.

Rezolucija (razlučivanje/razrješavanje)

Konjunktivna Normalna Forma (CNF—općenita)

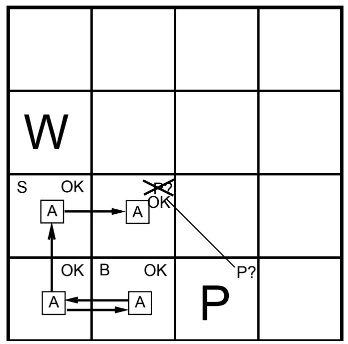
konjunkcija klauzula, **klazula = disjunkcija literala**

Na pr., $(A \lor \neg B) \land (B \lor \neg C \lor \neg D)$

Rezolucijsko pravilo zaključivanja (za CNF): potpuno za logiku sudova

$$\frac{\ell_1 \lor \cdots \lor \ell_i \lor \cdots \lor \ell_k, \qquad m_1 \lor \cdots \lor m_j \lor \cdots \lor m_n}{\ell_1 \lor \cdots \lor \ell_{i-1} \lor \ell_{i+1} \lor \cdots \lor \ell_k \lor m_1 \lor \cdots \lor m_{j-1} \lor m_{j+1} \lor \cdots \lor m_n}$$

gdje su ℓ_i i m_j međusobno komplementarni literali.



Na pr.,

$$\frac{P_{1,3} \lor P_{2,2}, \qquad \neg P_{2,2}}{P_{1,3}}$$

Rezolucija (opovrgavanjem) je ispravna i potpuna za logiku sudova

Pretvorba u CNF

$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$ $\quad =$ pravilo za vjetar na početku, na [1, 1]

1. Eliminiraj $\Leftrightarrow$, zamjenom $\alpha \Leftrightarrow \beta$ s $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$.

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

2. Eliminiraj $\Rightarrow$, zamjenom $\alpha \Rightarrow \beta$ s $\neg\alpha \vee \beta$.

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

3. Makni $\neg$ unutra do literala, po de Morganovim pravilima i eliminiraj dvostruke negacije:

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

4. Primijeni distributivnost ($\vee$ prema $\wedge$) i sredi:

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

Algoritam rezolucije (opovrgavanjem)

Dokaz kontradikcijom, tj., pokaži da je $KB \wedge \neg\alpha$ neispunjiva

function PL-RESOLUTION(KB, α) **returns** *true* or *false*
  inputs: KB, the knowledge base, a sentence in propositional logic
    α, the query, a sentence in propositional logic

  $clauses \leftarrow$ the set of clauses in the CNF representation of $KB \wedge \neg\alpha$
  $new \leftarrow \{\}$
  loop do
    for each C_i, C_j **in** $clauses$ **do**
      $resolvents \leftarrow$ PL-RESOLVE(C_i, C_j)
      if $resolvents$ contains the empty clause **then return** *true*
      $new \leftarrow new \cup resolvents$
    if $new \subseteq clauses$ **then return** *false*
    $clauses \leftarrow clauses \cup new$

Reprezentacija CNF = skup skupova, svaki "podskup" je klauzula

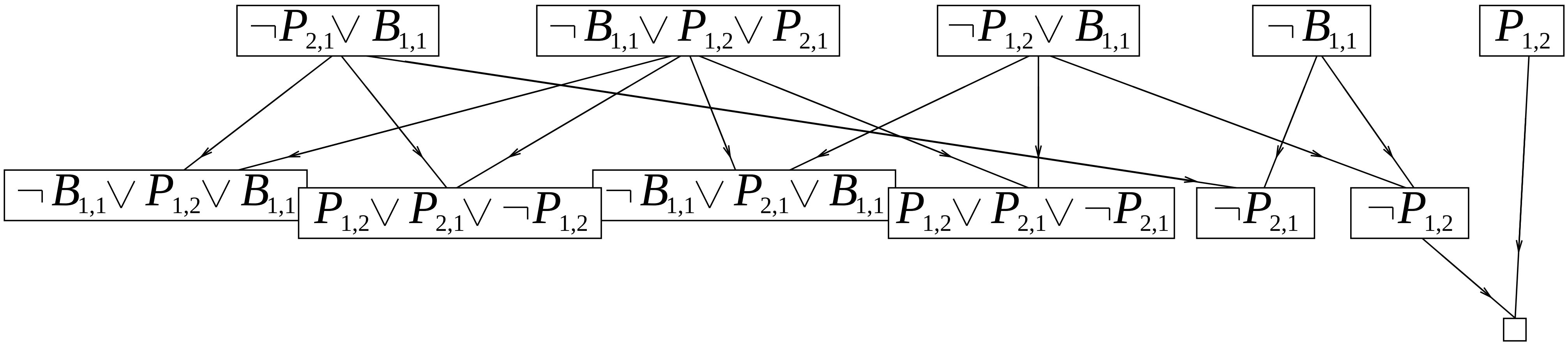
Primjer rezolucije

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

dodamo opažanje da nema vjetra na [1, 1]

$\alpha = \neg P_{1,2}$ = zaključak da nema ponora na [1, 2]

Uvrstimo CNF za lijevu stranu i “dodamo” $\neg\alpha$, tj. $P_{1,2}$



Spajamo parove iz prvog reda koji imaju komplementarne literale

eliminiramo ih pravilom rezolucije i pišemo rezolvente

(može ih biti više za jedan par klauzula — prve dvije, druge dvije)

Hornova forma CNF

Hornova forma (ograničena CNF)

KB = **konjunkcija Hornovih klauzula**

Hornova klauzula =

- ♢ propozicionalna varijabla; ili
- ♢ (konjunkcija varijabli) $\Rightarrow$ varijabla

Ekvivalentno: klauzula u kojoj je **najviše jedan** literal pozitivan.

Na pr., $C \wedge (B \Rightarrow A) \wedge (C \wedge D \Rightarrow B)$

Negacijom "premise" dobivamo klauzulu (disjunkciju) s negativnim literalima i dodamo zaključak (pozitivan ili ga nema = false)!

Podjela prema broju pozitivnih literala — ako je točno jedan:

— implikacija = točno jedan pozitivni literal (glava)

— ako nema premise = činjenica

Ako ih nema = nema zaključka = kao da je false = ciljna klauzula!

Ulančavanje unaprijed i unatrag

Modus Ponens (za Hornovu formu): potpun za Hornove baze znanja

$$\frac{\alpha_1, \ldots, \alpha_n, \qquad \alpha_1 \land \cdots \land \alpha_n \;\Rightarrow\; \beta}{\beta}$$

Može biti korišten s ulančavanjem unaprijed ili ulančavanjem unatrag.

Ti algoritmi su vrlo prirodni i izvršavaju se u **linearnom** vremenu u veličini KB.

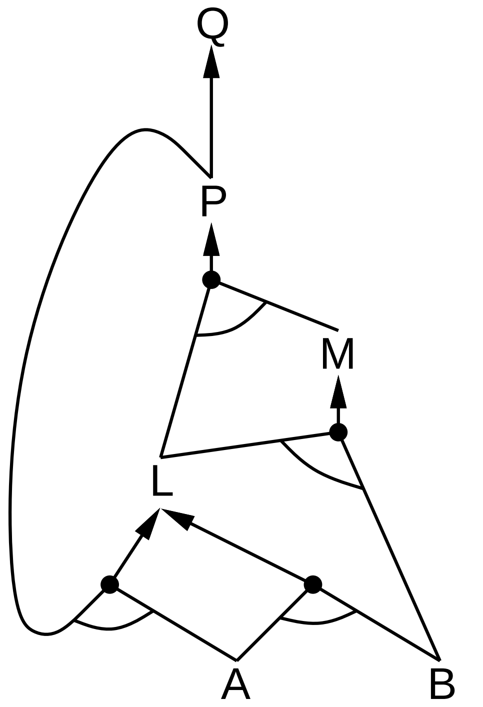
Reprezentacija klauzula u Hornovoj formi
— premisa je tijelo (engl. body), kad je ima
— zaključak je glava (engl. head), kad ga ima
(svagdje osim u ciljnoj klazuli)

Idealno za reprezentaciju listama (LISP)

Ulančavanje unaprijed (Forward chaining)

Ideja: uzmi bilo koju implikaciju čije premise su zadovoljene u KB, dodaj pripadni zaključak u KB, sve dok ne nađeš upit.

$$
\begin{aligned}
P &\Rightarrow Q \\
L \wedge M &\Rightarrow P \\
B \wedge L &\Rightarrow M \\
A \wedge P &\Rightarrow L \\
A \wedge B &\Rightarrow L \\
A & \\
B &
\end{aligned}
$$



Broj označava koliko premisa svake implikacije je još nepoznato.
Kad stigne na nula — dodamo zaključak (simbol) u KB

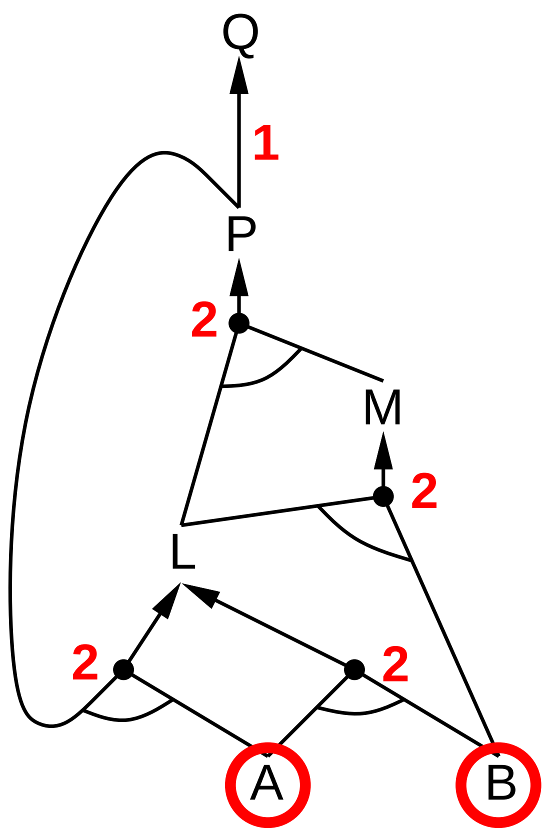
Ulančavanje unaprijed — algoritam

```
function PL-FC-Entails?(KB, q) returns true or false
    inputs: KB, the knowledge base, a set of propositional Horn clauses
            q, the query, a proposition symbol
    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 in KB

    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
```

Vodimo računa o već "zaključenim" simbolima

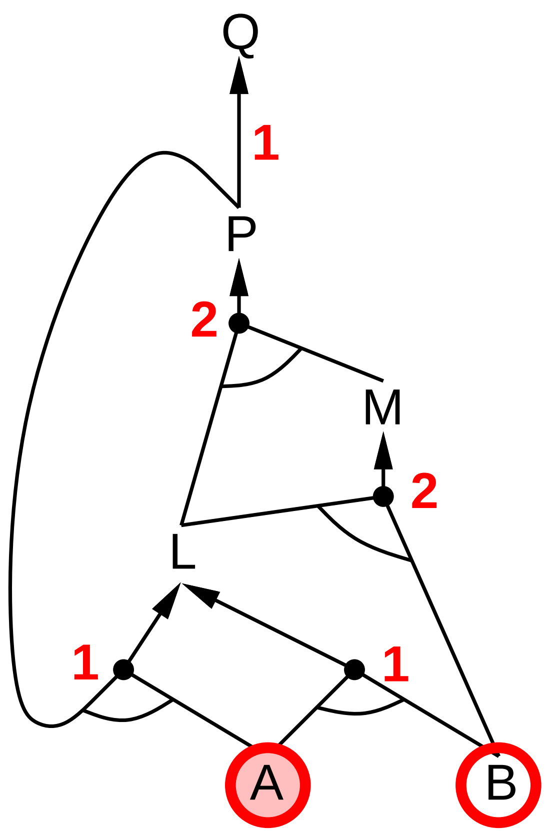
Ulančavanje unaprijed — primjer



Start: Upit = Q   samo A i B su u KB

Broj označava koliko premisa svake implikacije je još nepoznato.

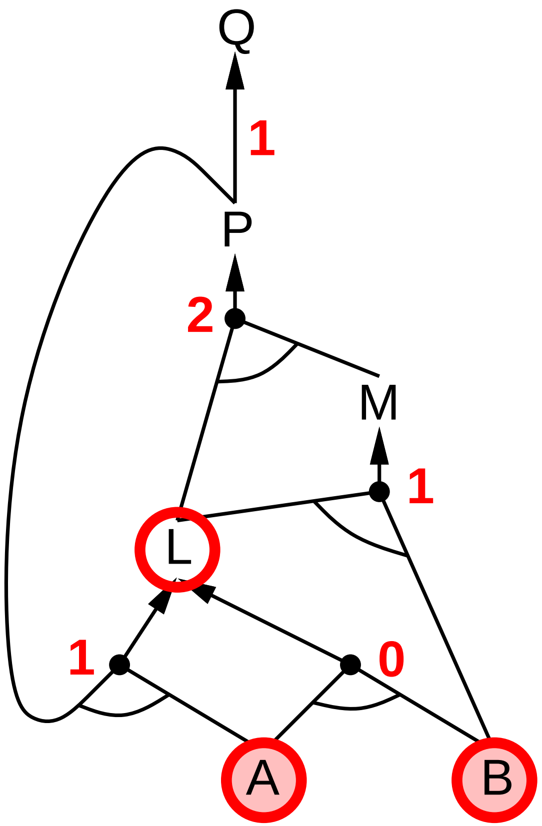
Ulančavanje unaprijed — primjer



A je u KB premisa za $A \wedge P \Rightarrow L$ i $A \wedge B \Rightarrow L$

Pripadni brojevi tih implikacija padaju za jedan — na 1 i 1

Ulančavanje unaprijed — primjer

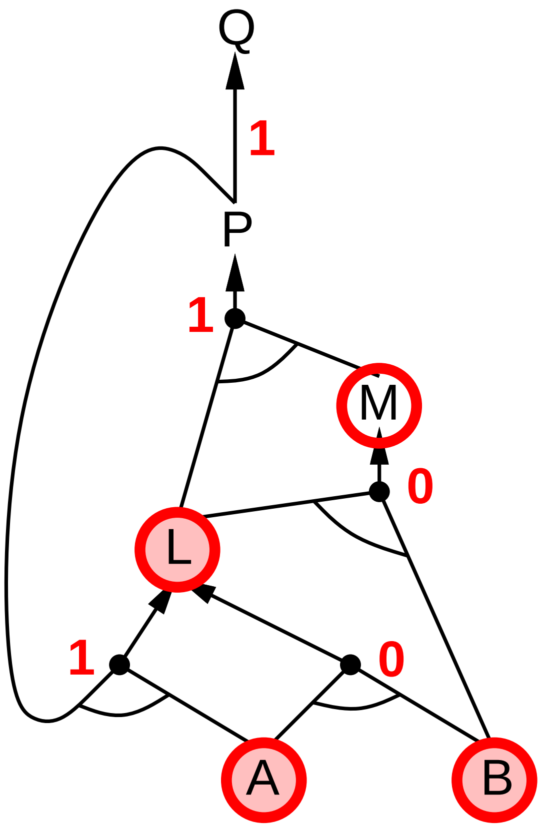


B je u KB premisa za $A \wedge B \Rightarrow L$ i $B \wedge L \Rightarrow M$

Pripadni brojevi tih implikacija padaju za jedan — na 0 i 1

Slijedi L je u KB

Ulančavanje unaprijed — primjer

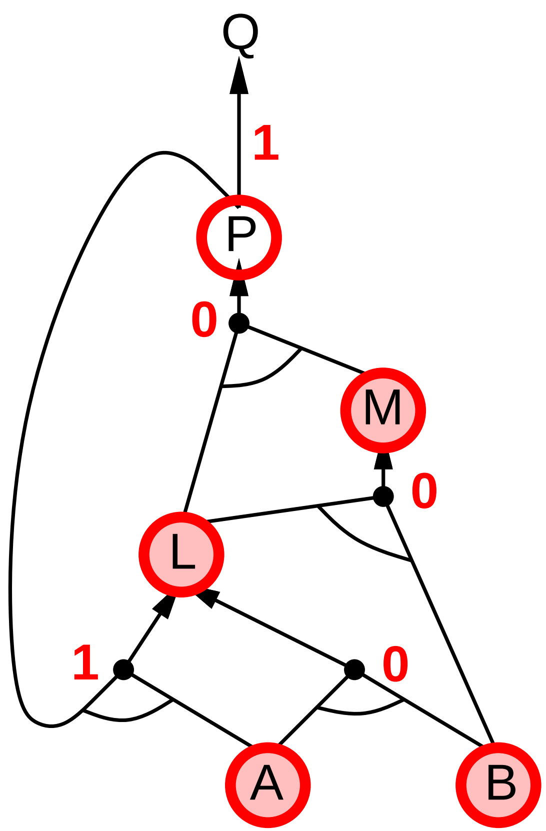


L je u KB premisa za $B \wedge L \Rightarrow M$ i $L \wedge M \Rightarrow P$

Pripadni brojevi tih implikacija padaju za jedan — na 0 i 1

Slijedi M je u KB

Ulančavanje unaprijed — primjer

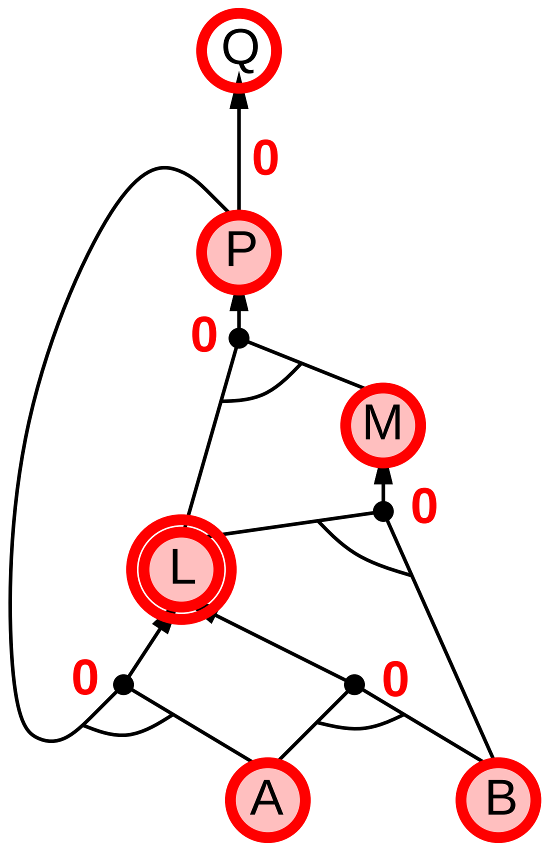


M je u KB premisa za $L \wedge M \Rightarrow P$

Pripadni broj te implikacije pada za jedan — na 0

Slijedi P je u KB

Ulančavanje unaprijed — primjer

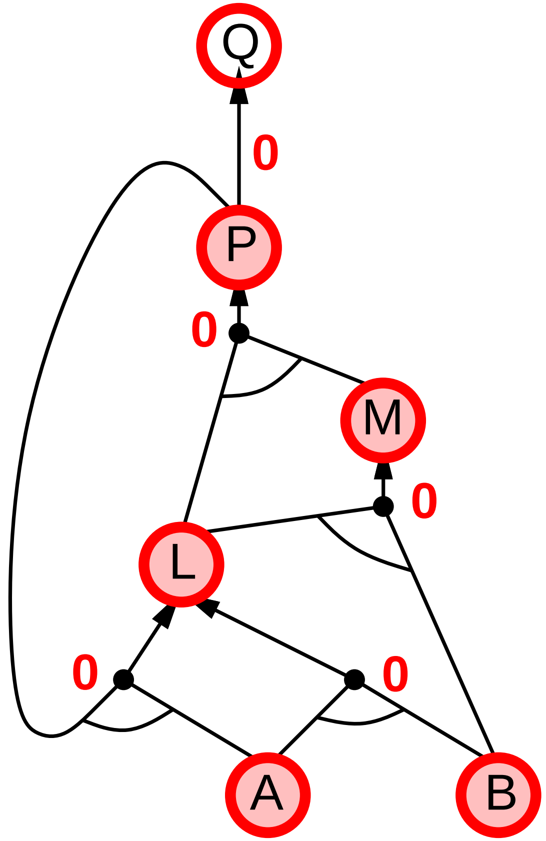


P je u KB premisa za $A \wedge P \Rightarrow L$ i $P \Rightarrow Q$

Pripadni brojevi tih implikacija padaju za jedan — na 0 i 0

Slijedi L je u KB i Q je u KB

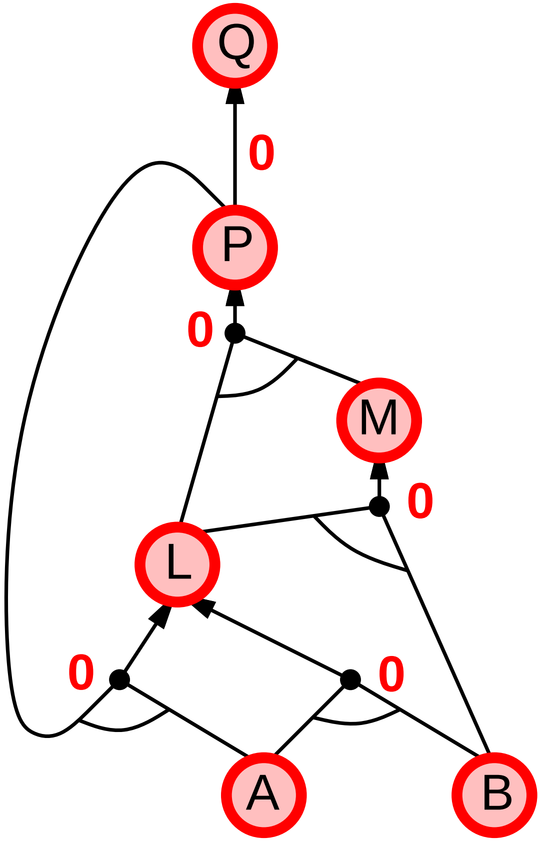
Ulančavanje unaprijed — primjer



L je već u KB, pa ga ne dodajemo ponovno

Implikacija $A \wedge P \Rightarrow L$ je "višak" u KB, irelevantna za zaključak (petlja u zaključku)

Ulančavanje unaprijed — primjer



Q je u KB — upit je istinit

Dokaz potpunosti — gruba skica

FC izvodi sve atomske rečenice koje su posljedica KB

1. FC stiže na fiksnu točku = nema izvoda novih atomskih rečenica
2. Pogledajmo završno stanje kao model m, i dodijelimo vrijednosti istina/laž svim simbolima
3. Svaka klauzula iz originalne KB je istinita u m
   Dokaz: (Suprotno) Neka je klauzula $a_1 \wedge \ldots \wedge a_k \Rightarrow b$ lažna u m
   Onda je $a_1 \wedge \ldots \wedge a_k$ istinita u m i b je lažna u m
   Stoga, algoritam sigurno nije stigao u fiksnu točku (nije gotov)!
4. Dakle, m je model za KB
5. Ako $KB \models q$, q je istina u svakom modelu od KB, uključujući m

Opća ideja: konstruiraj bilo koji model za KB koristeći ispravno zaključivanje, provjeri α

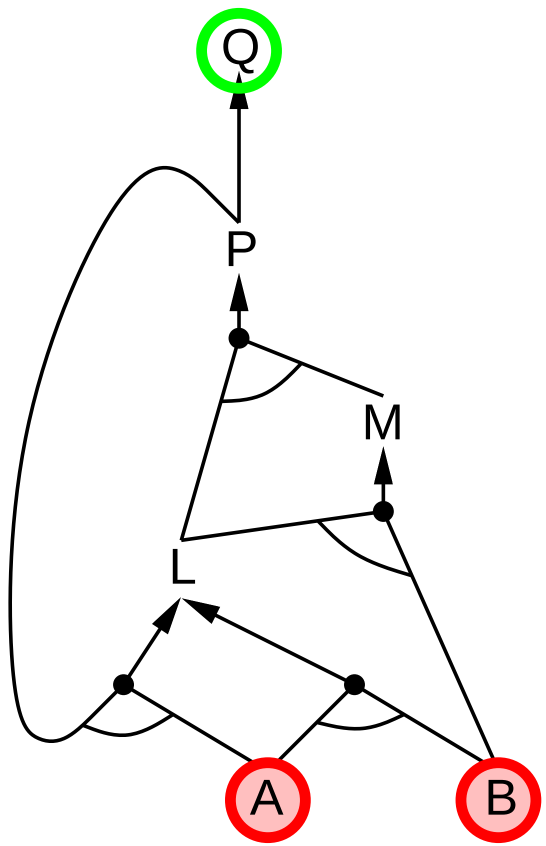
Ulančavanje unatrag (Backward chaining)

Ideja: idi unatrag od upita q:
da dokažemo q putem BC,
provjeri je li q već poznat (istina/laž), ili
dokaži putem BC sve premise neke implikacije iz koje slijedi q

Izjegni petlje: provjeri je li novi podcilj već na stogu ciljeva

Izbjegni ponavljanje posla: provjeri je li novi podcilj
1) već dokazan kao istinit, ili
2) već propao = opovrgnut (lažan)

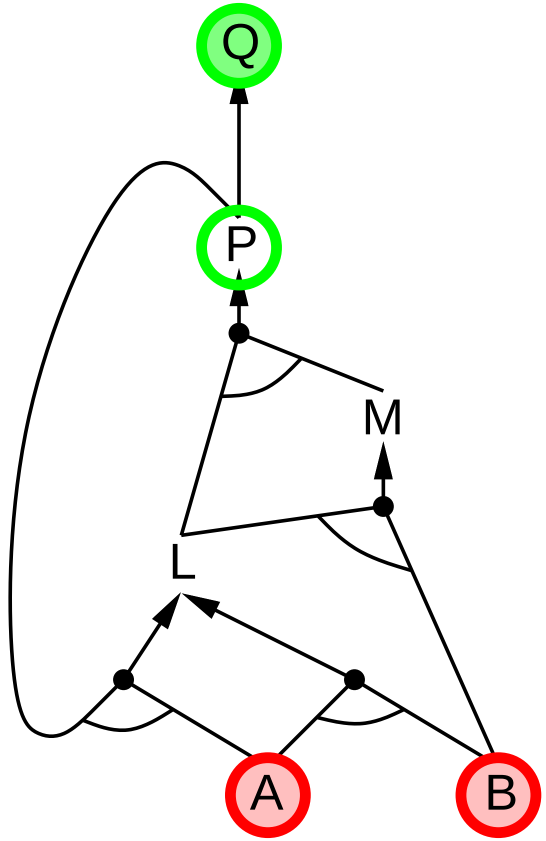
Ulančavanje unatrag — primjer



Start: Upit = Q   samo A i B su u KB, tj. već dokazani

Q nije dokazan, dodaj ga u stog = Q

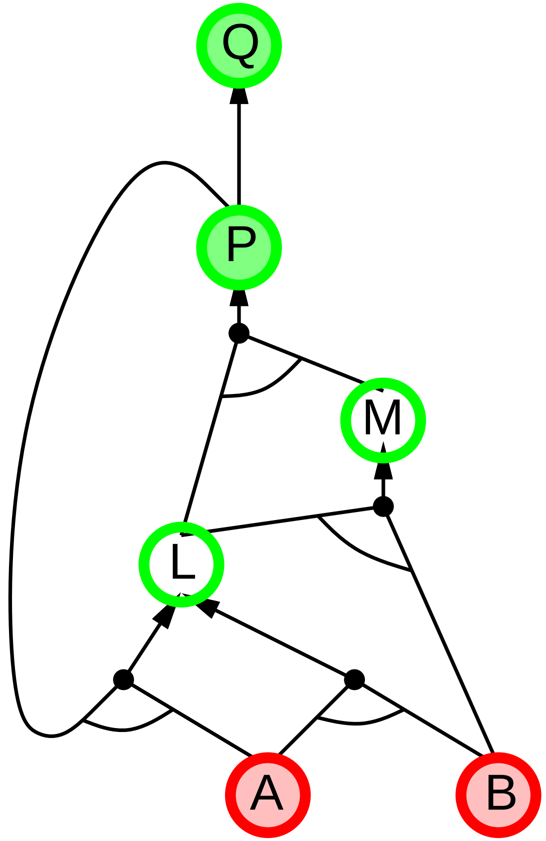
Ulančavanje unatrag — primjer



Q slijedi iz $P \Rightarrow Q$

P nije dokazan, dodaj ga u stog $= P, Q$

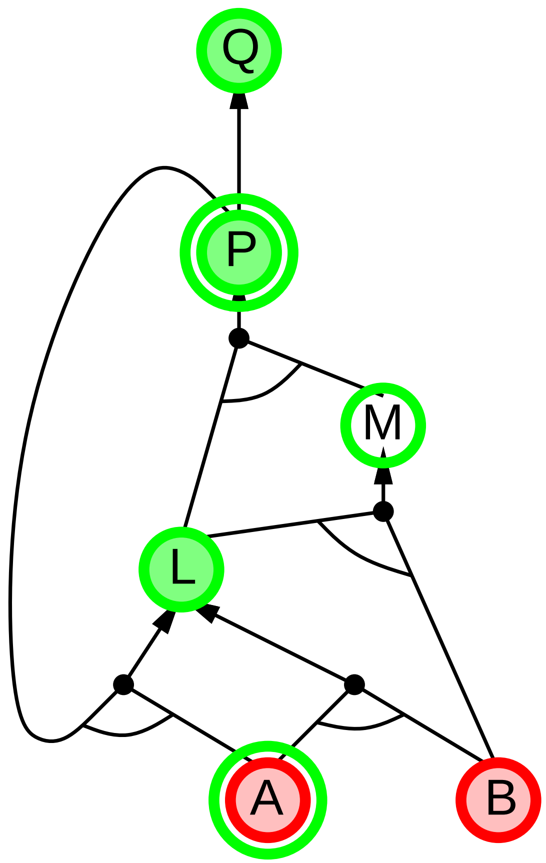
Ulančavanje unatrag — primjer



P slijedi iz $L \wedge M \Rightarrow P$

L i M nisu dokazani, dodaj ih u stog $= L, M, P, Q$

Ulančavanje unatrag — primjer

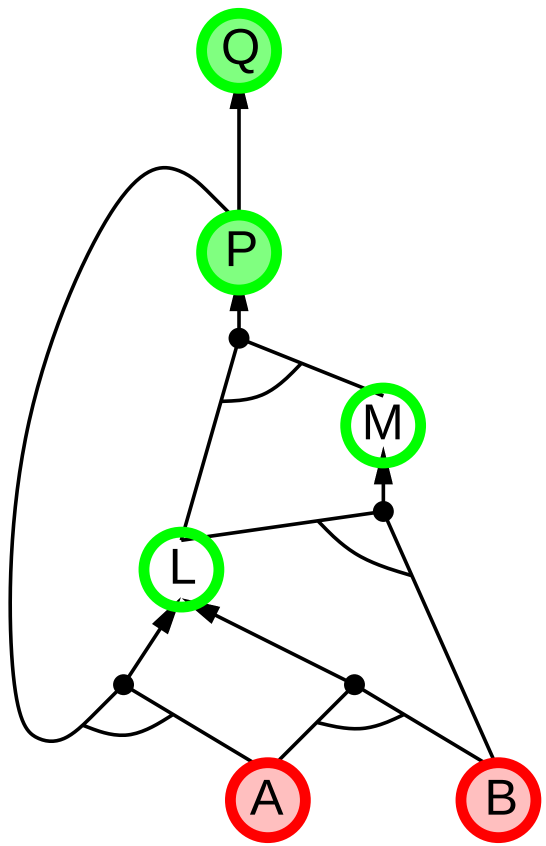


L slijedi iz $A \wedge P \Rightarrow L$

P nije dokazan, ali je već na stogu, stog $= L, M, P, Q$

A je već dokazan (istinita činjenica u KB), pa ne ide na stog

Ulančavanje unatrag — primjer

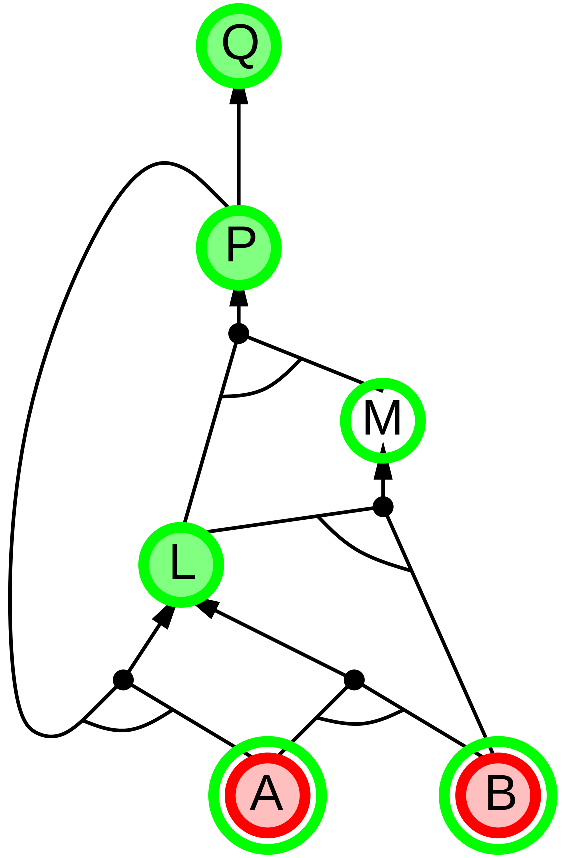


Dakle, situacija ostaje ista kao i prije (irelevantna implikacija)

stog = L, M, P, Q

Tražimo daljnje implikacije za podciljeve u KB

Ulančavanje unatrag — primjer

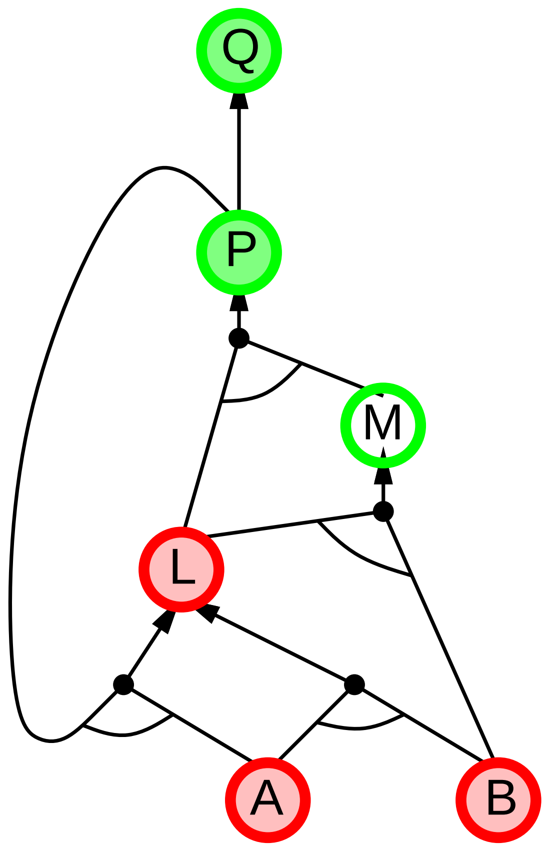


L slijedi iz $A \wedge B \Rightarrow L$

A i B su već dokazani (istinite činjenice u KB), pa ne idu na stog

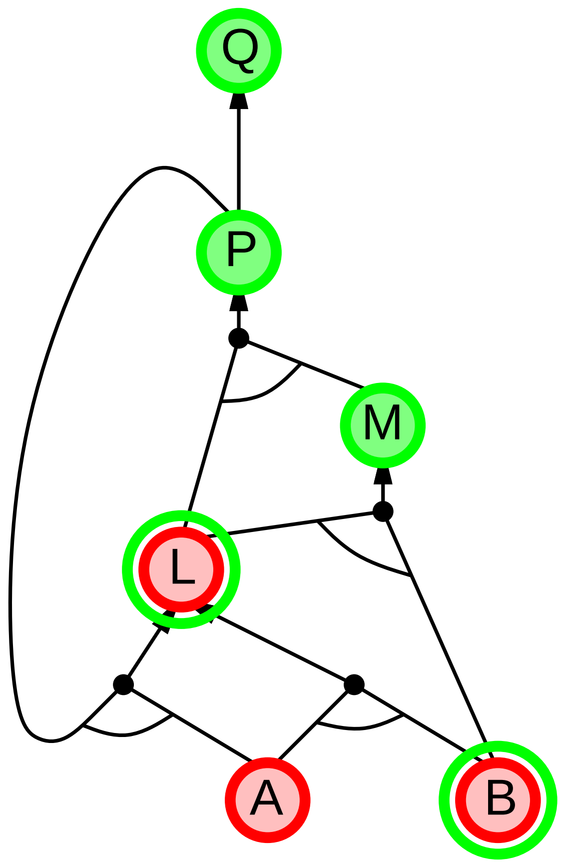
Zaključak: L je dokazan

Ulančavanje unatrag — primjer



L je dokazan, pa ga brišemo sa stoga, stog $= M, P, Q$

Ulančavanje unatrag — primjer

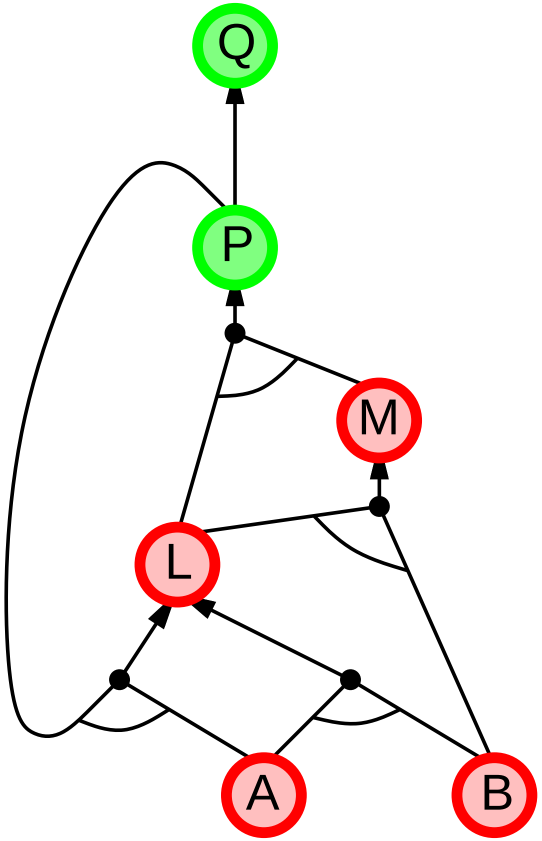


M slijedi iz $B \wedge L \Rightarrow M$

B i L su već dokazani

Zaključak: M je dokazan, pa ga brišemo sa stoga, stog $= P, Q$

Ulančavanje unatrag — primjer

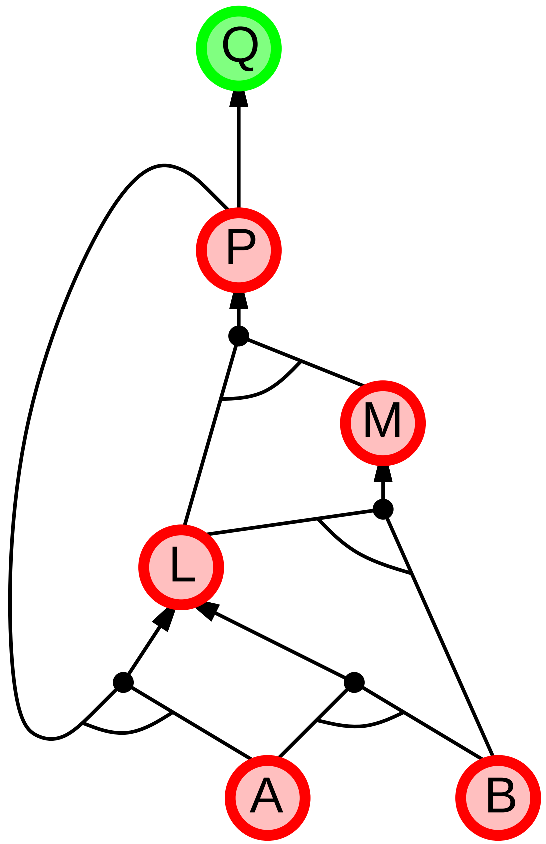


P slijedi iz $L \wedge M \Rightarrow P$

L i M su već dokazani

Zaključak: P je dokazan, pa ga brišemo sa stoga, stog $= Q$

Ulančavanje unatrag — primjer

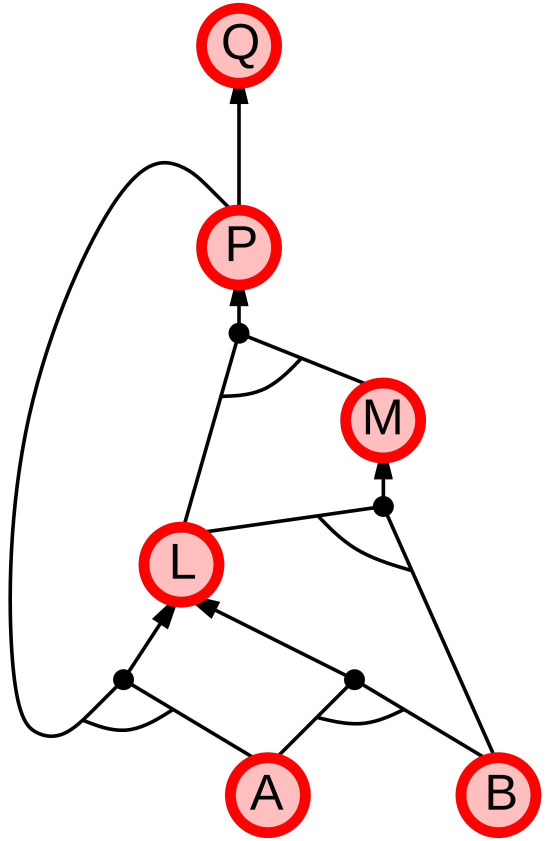


Q slijedi iz $P \Rightarrow Q$

P je već dokazan

Zaključak: Q je dokazan, pa ga brišemo sa stoga, stog $=$ prazan

Ulančavanje unatrag — primjer



stog = prazan = gotovo

Q je dokazan, dakle Q je u KB — upit je istinit

Ulančavanje unaprijed vs. unatrag

FC je vođeno podacima, tj. automatska, "nesvjesna" obrada,
na pr., za prepoznavanje objekata, rutinske odluke

Može raditi puno posla koji je irelevantan za cilj

BC je vođeno ciljem, primjereno za rješavanje problema,
na pr., "Gdje su moji ključevi?",
"Kako se upišem na doktorski studij?"

Složenost BC može biti **mnogo manja** od linearne u veličini KB.

Sažetak

Logički agenti koriste zaključivanje na temelju baze znanja (KB)
za izvođenje novih informacija i donošenje odluka

Osnovni koncepti logike:
- sintaksa: formalna struktura rečenica
- semantika: istinitost rečenica obzirom na modele
- posljedičnost: nužna istinitost rečenice za zadanu drugu
- zaključivanje: izvođenje rečenica iz drugih rečenica
- ispravnost: izvođenje daje samo posljedične rečenice
- potpunost: izvođenje daje sve posljedične rečenice

Wumpusov svijet treba mogućnost za prikaz djelomičnih i negiranih informacija, razmišljanje prema slučaju, itd.

Rezolucija (opovrgavanjem) je potpuna za logiku sudova
FC i BC — linearno vrijeme (u vel. KB), potpuni za Hornove klauzule

Zaključak i kamo dalje

Logika sudova (propozicionalna logika) nije dovoljno izražajna

Potrebna je "jača" logika za detaljniji prikaz stvari i odnosa među njima

tzv. logika prvog reda (FOL) = logika ili račun predikata