Logický agent, výroková logika

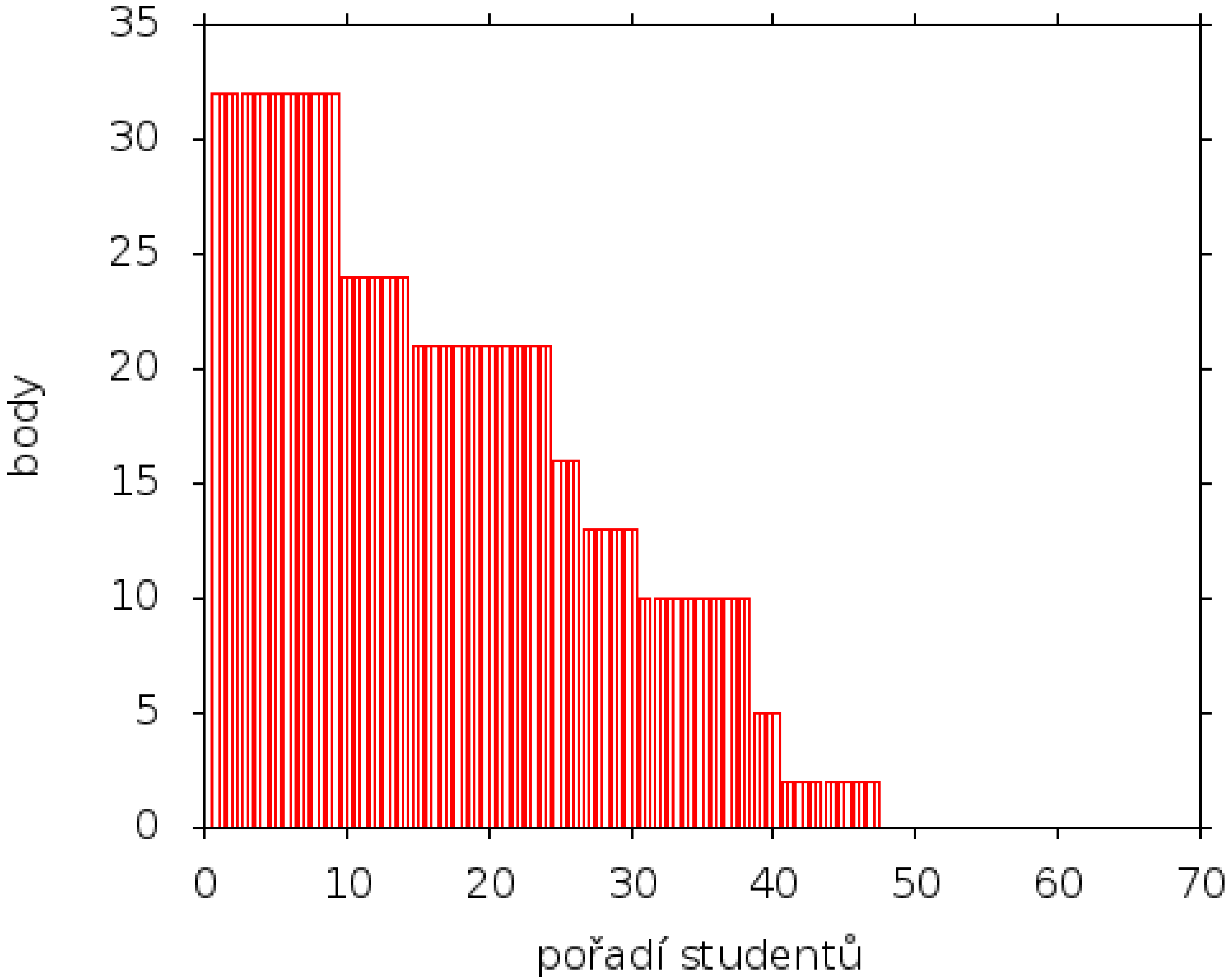
Aleš Horák

E-mail: `hales@fi.muni.cz`

`http://nlp.fi.muni.cz/uui/`

Obsah:

Statistické výsledky průběžné písemky



průběžná písemka PB016

60 studentů

Body	Počet studentů
32	8
24	5
21	10
16	2
13	4
10	8
5	2
2	7
0	14

Průměr: 12.9

Logický agent

logický agent = agent využívající znalosti (*knowledge-based agent*)

2 koncepty:
- – reprezentace znalostí (*knowledge representation*)
- – vyvozování znalostí (*knowledge reasoning*) $\longrightarrow$ inference

rozdíly od prohledávání stavového prostoru:

➜ znalost při prohledávání stavového prostoru $\longrightarrow$ jen zadané funkce (přechodová funkce, cílový test,...)

➜ znalosti logického agenta $\longrightarrow$ obecná forma umožňující kombinace těchto znalostí

obecné znalosti – důležité v částečně pozorovatelných prostředích (*partially observable* environments)

flexibilita logického agenta:
- ➜ schopnost řešit i nové úkoly
- ➜ možnost učení nových znalostí
- ➜ úprava stávajících znalostí podle stavu prostředí

Návrh logického agenta

agent musí umět:
- ➜ reprezentovat stavy, akce, ...
- ➜ zpracovat nové vstupy z prostředí
- ➜ aktualizovat svůj vnitřní popis světa
- ➜ odvodit skryté informace o stavu světa
- ➜ odvodit vlastní odpovídající akce

přístupy k tvorbě agenta (systému) – deklarativní $\times$ procedurální (kombinace obou)

návrh agenta $\rightarrow$ víc pohledů:

- ❑ **znalostní hledisko** – tvorba agenta $\rightarrow$ zadání znalostí pozadí, znalostí domény a cílového požadavku

  např. automatické taxi
  - znalost mapy, dopravních pravidel, ...
  - požadavek – dopravit zákazníka na FI MU Brno
- ❑ **implementační hledisko** – jaké datové struktury KB obsahuje + algoritmy, které s nimi manipulují

Komponenty agenta, Báze znalostí

komponenty logického agenta:

inferenční stroj (inference engine)	← algoritmy nezávislé na doméně
báze znalostí (knowledge base)	← “informace” o doméně

báze znalostí (KB) = množina vět (*tvrzení*) vyjádřených v jazyce reprezentace znalostí

obsah báze znalostí:

- ➜ na začátku – tzv. znalosti pozadí (*background knowledge*)
- ➜ průběžně doplňované znalosti → úkol **tell(+KB,+Sentence)**

akce logického agenta:

```
% kb_agent_action(+KB,+ATime,+Percept,-Action,-NewATime)
kb_agent_action(KB,ATime,Percept,Action,NewATime):-
    make_percept_sentence(Percept,ATime,Sentence),
    tell(KB,Sentence),                          % přidáme výsledky pozorování do KB
    make_action_query(ATime,Query),
    ask(KB,Query,Action),                       % zeptáme se na další postup
    make_action_sentence(Action,ATime,ASentence),
    tell(KB,ASentence),                         % přidáme informace o akci do KB
    NewATime is ATime + 1.
```

POPIS SVĚTA – PEAS

zadání světa rozumného agenta:

- **míra výkonnosti** (*Performance measure*)
  plus body za dosažené (mezi)cíle, pokuty za nežádoucí následky
- **prostředí** (*Environment*)
  objekty ve světě, se kterými agent musí počítat, a jejich vlastnosti
- **akční prvky** (*Actuators*)
  možné součásti činnosti agenta, jeho akce se skládají z použití těchto prvků
- **senzory** (*Sensors*)
  zpětné vazby akcí agenta, podle jejich výstupů se tvoří další akce

např. zmiňované automatické taxi:

míra výkonnosti	doprava na místo, vzdálenost, bezpečnost, bez přestupků, komfort, . . .
prostředí	ulice, křižovatky, účastníci provozu, chodci, počasí, . . .
akční prvky	řízení, plyn, brzda, houkačka, blinkry, komunikátory, . . .
senzory	kamera, tachometr, počítač kilometrů, senzory motoru, GPS, . . .

Wumpusova jeskyně

PEAS zadání Wumpusovy jeskyně:

- **P – míra výkonnosti**

  zlato +1000, smrt -1000, -1 za krok, -10 za užití šípu

- **E – prostředí**

  Místnosti vedle Wumpuse zapáchají

  V místnosti vedle jámy je vánek

  V místnosti je zlato ⇔ je v ní třpyt

  Výstřel zabije Wumpuse, pokud jsi obrácený k němu

  Výstřel vyčerpá jediný šíp, který máš

  Zvednutím vezmeš zlato ve stejné místnosti

  Položení odloží zlato v aktuální místnosti

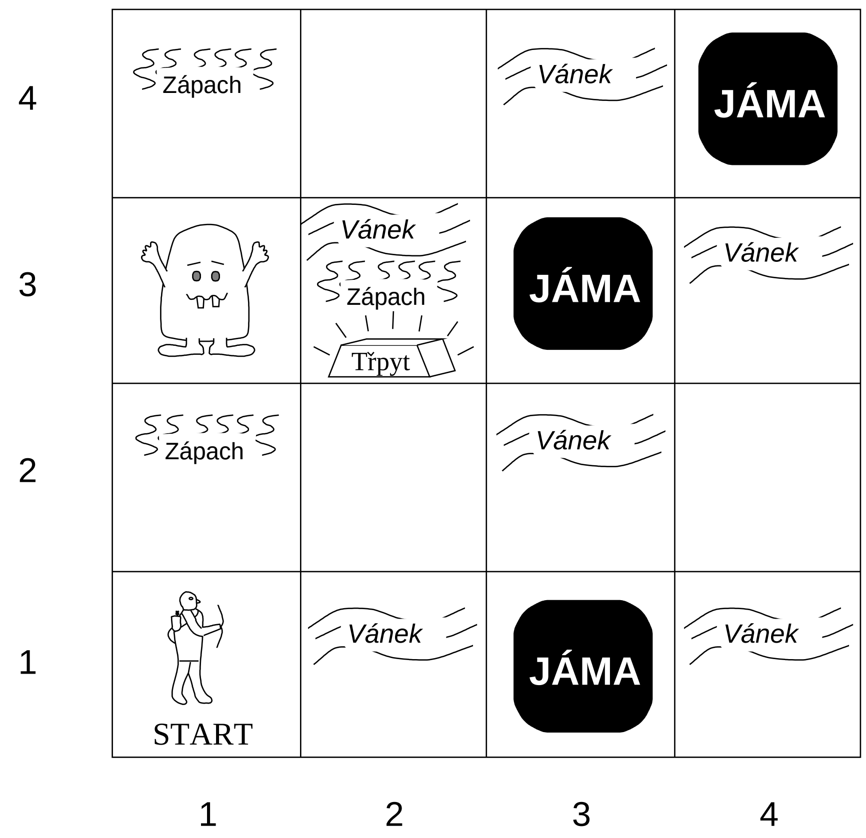
- **A – akční prvky**

  Otočení vlevo, Otočení vpravo, Krok dopředu,

  Zvednutí, Položení, Výstřel

- **S – senzory**

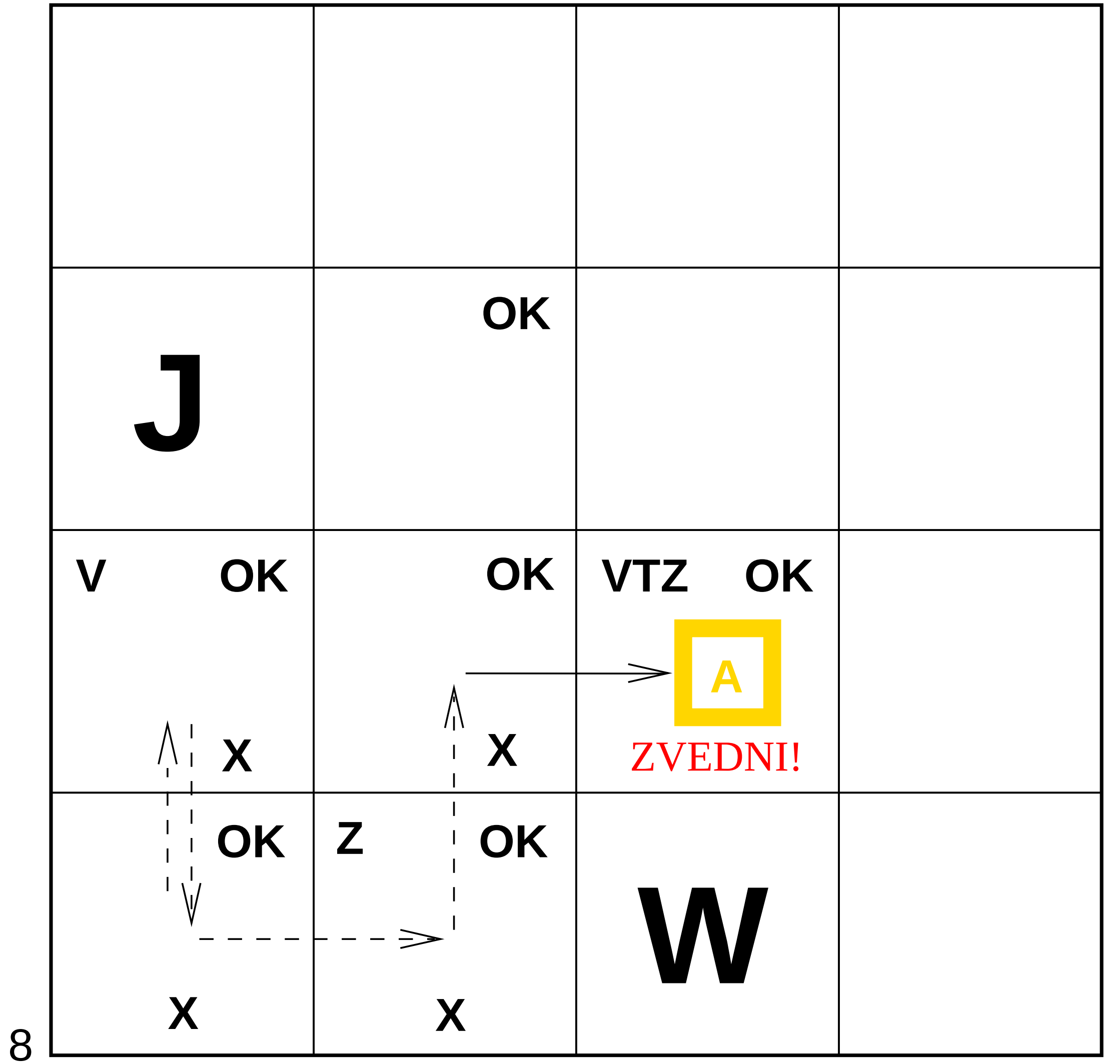
  Vánek, Třpyt, Zápach, Náraz do zdi, Chroptění Wumpuse



Vlastnosti problému Wumpusovy jeskyně

pozorovatelné	ne, jen lokální vnímání
deterministické	ano, přesně dané výsledky
episodické	ne, sekvenční na úrovni akcí
statické	ano, Wumpus a jámy se nehýbou
diskrétní	ano
více agentů	ne, Wumpus je spíše vlastnost prostředí

Průzkum Wumpusovy jeskyně



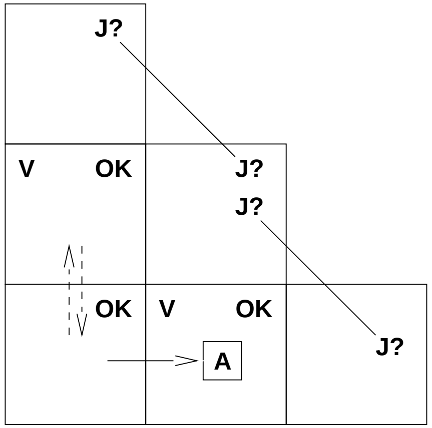
A	=	Agent
V	=	Vánek
T	=	Třpyt
OK	=	bezpečí
J	=	Jáma
Z	=	Zápach
X	=	navštíveno
W	=	Wumpus

Průzkum Wumpusovy jeskyně – problémy

základní vlastnost logického vyvozování:

> *Kdykoliv agent dospěje k závěru z daných informací $\longrightarrow$ tento závěr je zaručeně správný, pokud jsou správné dodané informace.*

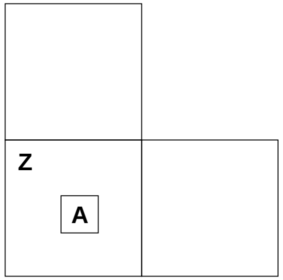
obtížné situace:



Vánek v $(1,2)$ i v $(2,1)$ $\Rightarrow$ žádná bezpečná akce

Při předpokladu uniformní distribuce děr

$\longrightarrow$ díra v $(2,2)$ má pravděpodobnost 0.86, na krajích 0.31



Zápach v $(1,1)$ $\Rightarrow$ nemůže se pohnout

je možné použít donucovací strategii (*strategy of coercion*):

1. Výstřel jedním ze směrů
2. byl tam Wumpus $\Rightarrow$ je mrtvý (poznám podle Chroptění) $\Rightarrow$ bezpečné
3. nebyl tam Wumpus (žádné Chroptění) $\Rightarrow$ bezpečný směr

Logika

Logika = *syntaxe* a *sémantika* formálního jazyka pro reprezentaci informací umožňující vyvozování závěrů

Syntaxe definuje všechny *dobře utvořené věty* jazyka

Sémantika definuje "*význam*" vět $\Rightarrow$ definuje pravdivost vět v jazyce (v závislosti na *možném světě*)

např. jazyk aritmetiky:

- ➜ $x + 2 \geq y$ je dobře utvořená věta; $x2 + y >$ není věta
- ➜ $x + 2 \geq y$ je pravda $\Leftrightarrow$ číslo $x + 2$ není menší než číslo y
- ➜ $x + 2 \geq y$ je pravda ve světě, kde $x = 7,\ y = 1$
- ➜ $x + 2 \geq y$ je nepravda ve světě, kde $x = 0,\ y = 6$

zápis na papíře v libovolné syntaxi $\rightarrow$ v KB se jedná o konfiguraci (částí) agenta

vlastní vyvozování $\rightarrow$ generování a manipulace s těmito konfiguracemi

Důsledek

Důsledek (vyplývání, *entailment*) – jedna věc logicky vyplývá z druhé (je jejím důsledkem):

$$KB \models \alpha$$

Z báze znalostí KB vyplývá věta α $\Leftrightarrow$ α je pravdivá ve *všech světech*, kde je KB pravdivá

např.:

- KB obsahuje věty – "Češi vyhráli"

  – "Slováci vyhráli"

  z KB pak vyplývá – "Buď Češi vyhráli nebo Slováci vyhráli"
- z $x + y = 4$ vyplývá $4 = x + y$

Důsledek je vztah mezi větami (*syntaxe*), který je založený na *sémantice*.

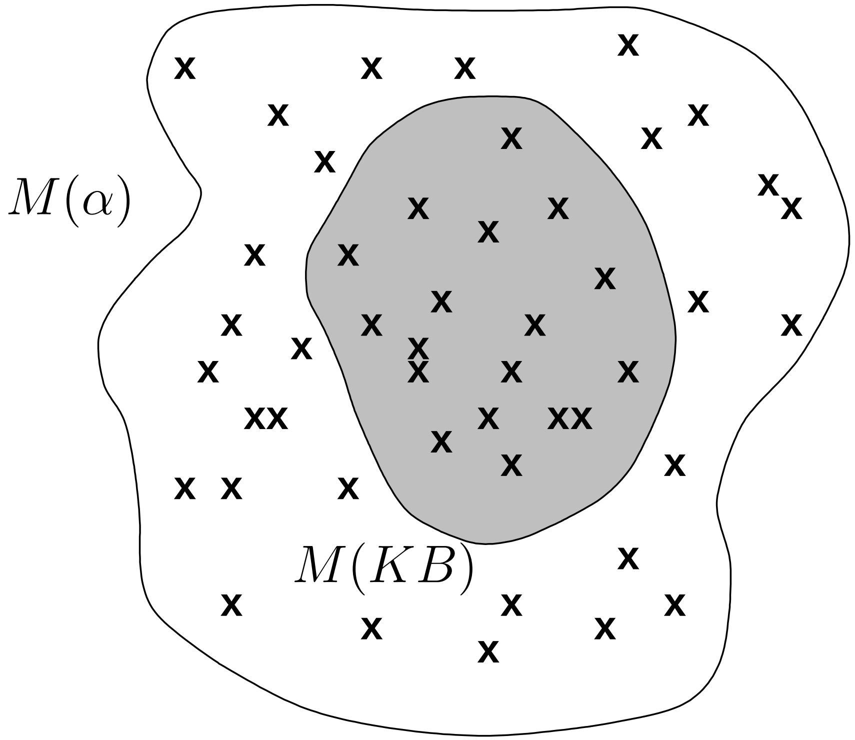
Model

možný svět = model . . . formálně strukturovaný (abstraktní) svět, umožňuje vyhodnocení pravdivosti

říkáme: m je model věty α $\Leftrightarrow$ α je pravdivá v m

$M(\alpha)$ . . . množina všech modelů věty α

$$KB \models \alpha \quad \Leftrightarrow \quad M(KB) \subseteq M(\alpha)$$



např.: $KB =$ "Češi vyhráli" $\wedge$ "Slováci vyhráli"

$\alpha =$ "Češi vyhráli"

INFERENCE

Vyvozování požadovaných důsledků – inference

$KB \vdash_i \alpha \ldots$ věta α může být vyvozena z KB pomocí (procedury) i (i odvodí α z KB)

všechny možné důsledky KB jsou "kupka sena"; α je jehla

vyplývání = jehla v kupce sena; inference = její nalezení

Bezespornost: i je bezesporná $\Leftrightarrow$ $\forall KB \vdash_i \alpha \Rightarrow KB \models \alpha$

Úplnost: i je úplná $\Leftrightarrow$ $\forall KB \models \alpha \Rightarrow KB \vdash_i \alpha$

Vztah k *reálnému světu*:

> Pokud je KB pravdivá v reálném světě $\Rightarrow \forall$ věta α vyvozená z KB pomocí bezesporné inference je také pravdivá ve skutečném světě

Jestliže máme sémantiku "pravdivou" v reálném světě $\rightarrow$ můžeme vyvozovat závěry o skutečném světě pomocí logiky

Výroková logika

Výroková logika – nejjednodušší logika, ilustruje základní myšlenky

- **výrokové symboly** $P_1, P_2, \ldots$ jsou věty
- **negace** – S je věta $\Rightarrow$ $\neg S$ je věta
- **konjunkce** – S_1 a S_2 jsou věty $\Rightarrow$ $S_1 \wedge S_2$ je věta
- **disjunkce** – S_1 a S_2 jsou věty $\Rightarrow$ $S_1 \vee S_2$ je věta
- **implikace** – S_1 a S_2 jsou věty $\Rightarrow$ $S_1 \Rightarrow S_2$ je věta
- **ekvivalence** – S_1 a S_2 jsou věty $\Rightarrow$ $S_1 \Leftrightarrow S_2$ je věta

Sémantika výrokové logiky

- každý model musí určit pravdivostní hodnoty výrokových symbolů
  např.: $m_1 = \{P_1 = false, P_2 = false, P_3 = true\}$
- pravidla pro vyhodnocení pravdivosti u složených výroků pro model m:

$\neg S$	je $true$	$\Leftrightarrow$	S	je $false$			
$S_1 \wedge S_2$	je $true$	$\Leftrightarrow$	S_1	je $true$	a	S_2	je $true$
$S_1 \vee S_2$	je $true$	$\Leftrightarrow$	S_1	je $true$	nebo	S_2	je $true$
$S_1 \Rightarrow S_2$	je $true$	$\Leftrightarrow$	S_1	je $false$	nebo	S_2	je $true$
tj.	je $false$	$\Leftrightarrow$	S_1	je $true$	a	S_2	je $false$
$S_1 \Leftrightarrow S_2$	je $true$	$\Leftrightarrow$	$S_1 \Rightarrow S_2$	je $true$	a	$S_2 \Rightarrow S_1$	je $true$

- rekurzivním procesem vyhodnotíme lib. větu:
  $\neg P_1 \wedge (P_2 \vee P_3)$ = $true \wedge (false \vee true) = true \wedge true = true$

pravdivostní tabulka:

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

Logická ekvivalence

Dva výroky jsou logicky ekvivalentní právě tehdy, když jsou pravdivé ve stejných modelech:

$$\alpha \equiv \beta \quad \Leftrightarrow \quad \alpha \models \beta \text{ a } \beta \models \alpha$$

$$\begin{array}{rcll}
(\alpha \wedge \beta) & \equiv & (\beta \wedge \alpha) & \text{komutativita } \wedge \\
(\alpha \vee \beta) & \equiv & (\beta \vee \alpha) & \text{komutativita } \vee \\
((\alpha \wedge \beta) \wedge \gamma) & \equiv & (\alpha \wedge (\beta \wedge \gamma)) & \text{asociativita } \wedge \\
((\alpha \vee \beta) \vee \gamma) & \equiv & (\alpha \vee (\beta \vee \gamma)) & \text{asociativita } \vee \\
\neg(\neg\alpha) & \equiv & \alpha & \text{eliminace dvojí negace} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\beta \Rightarrow \neg\alpha) & \text{kontrapozice} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\alpha \vee \beta) & \text{eliminace implikace} \\
(\alpha \Leftrightarrow \beta) & \equiv & ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) & \text{eliminace ekvivalence} \\
\neg(\alpha \wedge \beta) & \equiv & (\neg\alpha \vee \neg\beta) & \text{de Morgan} \\
\neg(\alpha \vee \beta) & \equiv & (\neg\alpha \wedge \neg\beta) & \text{de Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) & \equiv & ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) & \text{distributivita } \wedge \text{ nad } \vee \\
(\alpha \vee (\beta \wedge \gamma)) & \equiv & ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) & \text{distributivita } \vee \text{ nad } \wedge
\end{array}$$

Platnost a splnitelnost

- Výrok je platný $\Leftrightarrow$ je pravdivý ve všech modelech

  např.: $true, \quad A \vee \neg A, \quad A \Rightarrow A, \quad (A \wedge (A \Rightarrow B)) \Rightarrow B$

  Platnost je spojena s inferencí pomocí věty o dedukci:

  $KB \models \alpha \quad \Leftrightarrow \quad (KB \Rightarrow \alpha)$ je platný výrok

- Výrok je splnitelný $\Leftrightarrow$ je pravdivý v některých modelech

  např.: $A \vee B, \quad C$

  Výrok je nesplnitelný $\Leftrightarrow$ je nepravdivý ve všech modelech

  např.: $A \wedge \neg A$

  Splnitelnost je spojena s inferencí pomocí důkazu α sporem (*reductio ad absurdum*):

  $KB \models \alpha \quad \Leftrightarrow \quad (KB \wedge \neg\alpha)$ je nesplnitelný

Tvrzení pro Wumpusovu jeskyni

Definujeme výrokové symboly $J_{i,j}$ je pravda $\Leftrightarrow$ V $[i,j]$ je Jáma.
a $V_{i,j}$ je pravda $\Leftrightarrow$ V $[i,j]$ je Vánek.

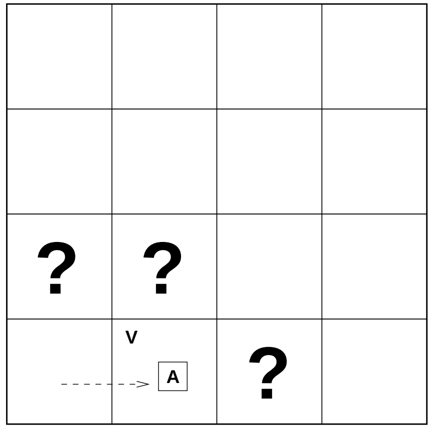
báze znalostí KB:

- pravidlo pro $[1,1]$: R_1: $\neg J_{1,1}$
- pozorování: R_2: $\neg V_{1,1}$
  R_3: $V_{2,1}$
- pravidla pro vztah Jámy a Vánku:
  "Jámy způsobují Vánek ve vedlejších místnostech"

$$R_4'\colon V_{1,1} \quad \Leftarrow \quad (J_{1,2} \vee J_{2,1})$$
$$R_5'\colon V_{2,1} \quad \Leftarrow \quad (J_{1,1} \vee J_{2,2} \vee J_{3,1})$$

"V poli je Vánek právě tehdy, když je ve vedlejším poli Jáma."

$$R_4\colon V_{1,1} \quad \Leftrightarrow \quad (J_{1,2} \vee J_{2,1})$$
$$R_5\colon V_{2,1} \quad \Leftrightarrow \quad (J_{1,1} \vee J_{2,2} \vee J_{3,1})$$



- $KB = R_1 \wedge R_2 \wedge R_3 \wedge R_4 \wedge R_5$

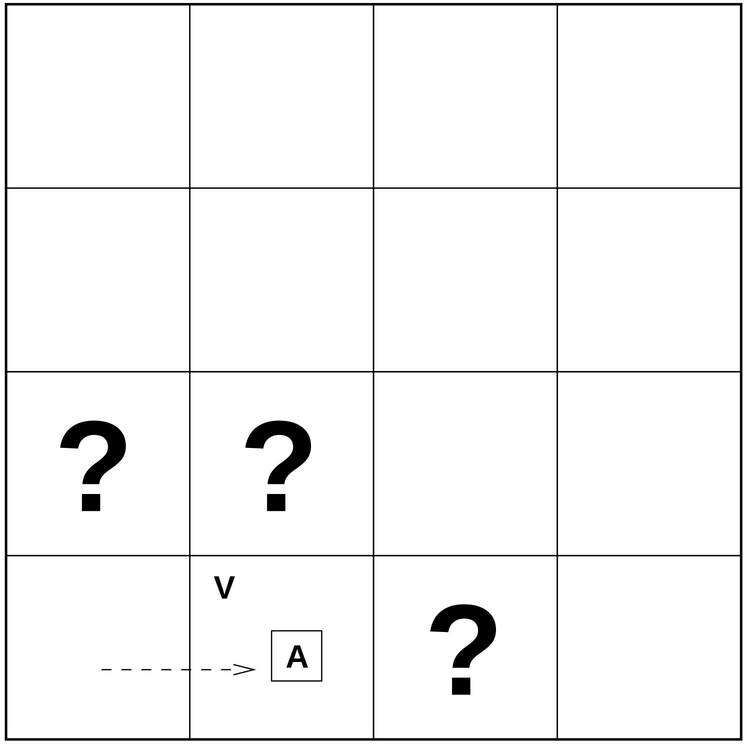
Vyplývání ve Wumpusově jeskyni

situace:

- v $[1, 1]$ nedetekováno nic
- krok doprava, v $[2, 1]$ Vánek

uvažujeme možné *modely* pro '**?**'

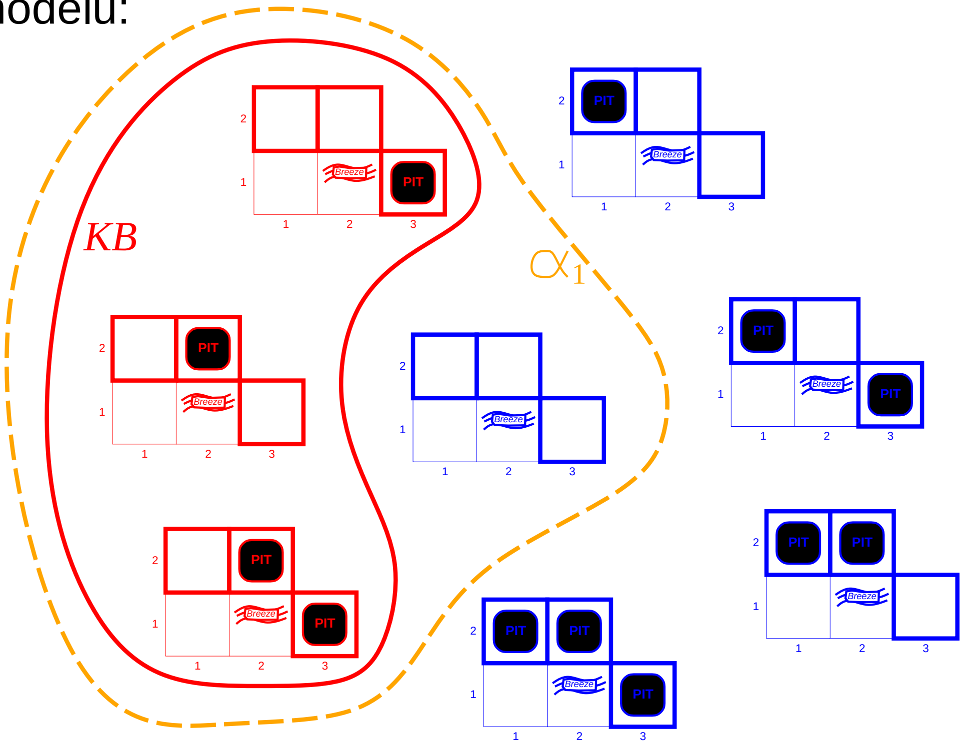
(budou nás zajímat jen Jámy)



3 pole s Booleovskými možnostmi $\{T, F\} \quad \Rightarrow \quad 2^3 = 8$ možných modelů

Modely ve Wumpusově jeskyni

uvažujeme všech 8 možných modelů:



$KB =$ pravidla Wumpusovy jeskyně $+$ pozorování

$\alpha_1 =$ "$[1, 2]$ je bezpečné pole" $\quad KB \models \alpha_1$

$\alpha_2 =$ "$[2, 2]$ je bezpečné pole" $\quad KB \not\models \alpha_2$

kontrola modelů $\rightarrow$ jednoduchý způsob logické inference

Pravdivostní tabulka pro inferenci

$V_{1,1}$	$V_{2,1}$	$J_{1,1}$	$J_{1,2}$	$J_{2,1}$	$J_{2,2}$	$J_{3,1}$	KB	α_1
$false$	$false$	$false$	$false$	$false$	$false$	$false$	$false$	$true$
$false$	$false$	$false$	$false$	$false$	$false$	$true$	$false$	$true$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$false$	$true$	$false$	$false$	$false$	$false$	$false$	$false$	$true$
$false$	$true$	$false$	$false$	$false$	$false$	$true$	$\underline{true}$	$\underline{true}$
$false$	$true$	$false$	$false$	$false$	$true$	$false$	$\underline{true}$	$\underline{true}$
$false$	$true$	$false$	$false$	$false$	$true$	$true$	$\underline{true}$	$\underline{true}$
$false$	$true$	$false$	$false$	$true$	$false$	$false$	$false$	$true$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$true$	$true$	$true$	$true$	$true$	$true$	$true$	$false$	$false$

$KB =$ pravidla Wumpusovy jeskyně $+$ pozorování

$\alpha_1 =$ “$[1, 2]$ je bezpečné pole”

Důkazové metody

- **kontrola modelů**
  - procházení pravdivostní tabulky (vždycky exponenciální v n)
  - vylepšené prohledávání s navracením (*improved backtracking*), např. Davis–Putnam–Logemann–Loveland
  - heuristické prohledávání prostoru modelů (bezesporné, ale neúplné)
- **aplikace inferenčních pravidel**
  - legitimní (bezesporné) generování nových výroků ze starých
  - důkaz = sekvence aplikací inferenčních pravidel

    je možné použít inferenční pravidla jako operátory ve standardních prohledávacích algoritmech
  - typicky vyžaduje překlad vět do normální formy

INFERENCE KONTROLOU MODELŮ

Kontrola všech modelů *do hloubky* je bezesporná a úplná (pro konečný počet výrokových symbolů)

```
% tt_entails (+KB,+Alpha)
tt_entails (KB,Alpha):-proposition_symbols(Symbols,[KB,Alpha]),
    tt_check_all (KB,Alpha,Symbols,[]).

% tt_check_all (+KB,+Alpha,+Symbols,+Model)
tt_check_all (KB,Alpha,[],Model):-pl_true(KB,Model),!,pl_true(Alpha,Model).
tt_check_all (KB,Alpha,[],Model):-!,fail.
tt_check_all (KB,Alpha,[P|Symbols],Model):-
    tt_check_all (KB,Alpha,Symbols,[P-true|Model]),
    tt_check_all (KB,Alpha,Symbols,[P-false|Model]).
```

$O(2^n)$ pro n symbolů, NP-úplný problém

Dopředné a zpětné řetězení

Hornovy klauzule: KB = konjunkce Hornových klauzulí

$$\text{Hornova klauzule} = \begin{cases} \text{výrokový symbol; nebo} \\ (\text{konjunkce symbolů}) \Rightarrow \text{symbol} \end{cases}$$

např.: $KB = C \wedge (B \Rightarrow A) \wedge (C \wedge D \Rightarrow B)$

pravidlo Modus Ponens – pro KB z Hornových klauzulí je úplné

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

pravidla pro logickou ekvivalenci se taky dají použít pro inferenci

inference Hornových klauzulí $\longrightarrow$ algoritmus dopředného nebo zpětného řetězení

oba tyto algoritmy jsou přirozené a mají lineární časovou složitost

Dopředné řetězení

Idea: aplikuj pravidlo, jehož premisy jsou splněné v KB

přidej jeho důsledek do KB

pokračuj do doby, než je nalezena odpověď

KB:

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

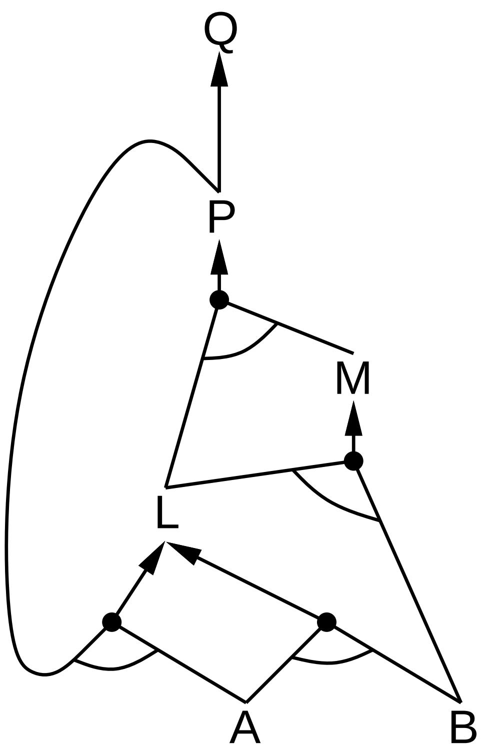
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B

AND-OR graf KB:



Dopředné řetězení – příklad

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

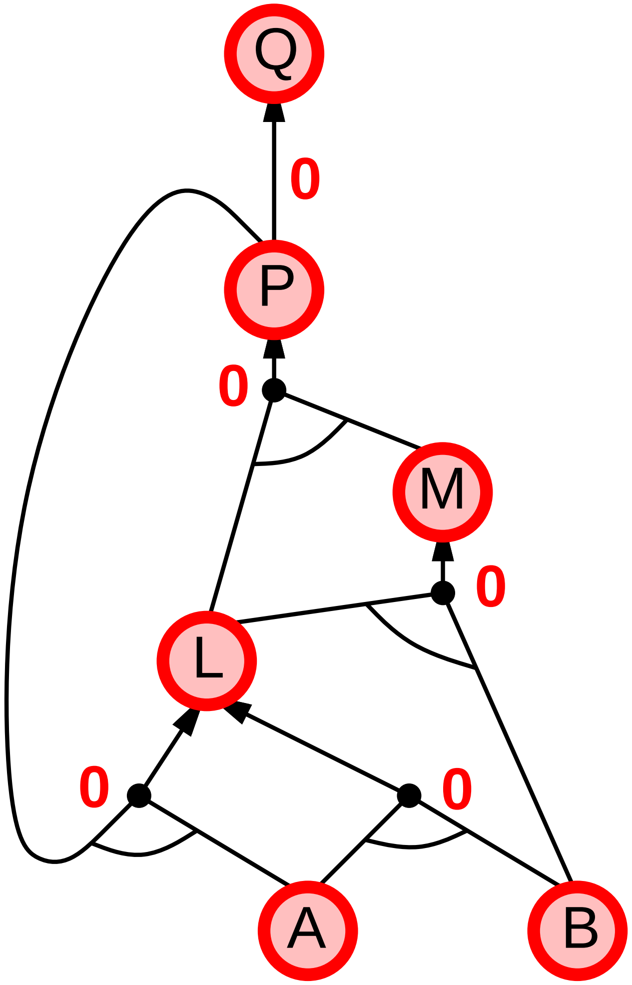
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Algoritmus dopředného řetězení

```
:- op( 800, fx, if ),
   op( 700, xfx, then),
   op( 300, xfy, or),
   op( 200, xfy, and).

forward :- new_derived_fact( P), !,                     % A new fact
    write( 'Derived: '), write( P), nl,
    assert( fact( P)),
    forward                                             % Continue
    ;
    write( 'No more facts').                            % All facts derived

new_derived_fact( Concl) :- if Cond then Concl,         % A rule
    \+ fact( Concl),                                    % Rule's conclusion not yet a fact
    composed_fact( Cond).                               % Condition true?

composed_fact( Cond) :- fact( Cond).                    % Simple fact
composed_fact( Cond1 and Cond2) :- composed_fact( Cond1),
    composed_fact( Cond2).                              % Both conjuncts true
composed_fact( Cond1 or Cond2) :- composed_fact( Cond1)
    ; composed_fact( Cond2).
```

Zpětné řetězení

Idea: pracuje zpětně od dotazu q

zkontroluj, jestli není q už známo

dokaž zpětným řetězením všechny premisy nějakého pravidla, které má q jako důsledek

kontrola cyklů – pro každý podcíl se nejprve podívej, jestli už nebyl řešen (tj. pamatuje si $true$ i $false$ výsledek)

Zpětné řetězení – příklad

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

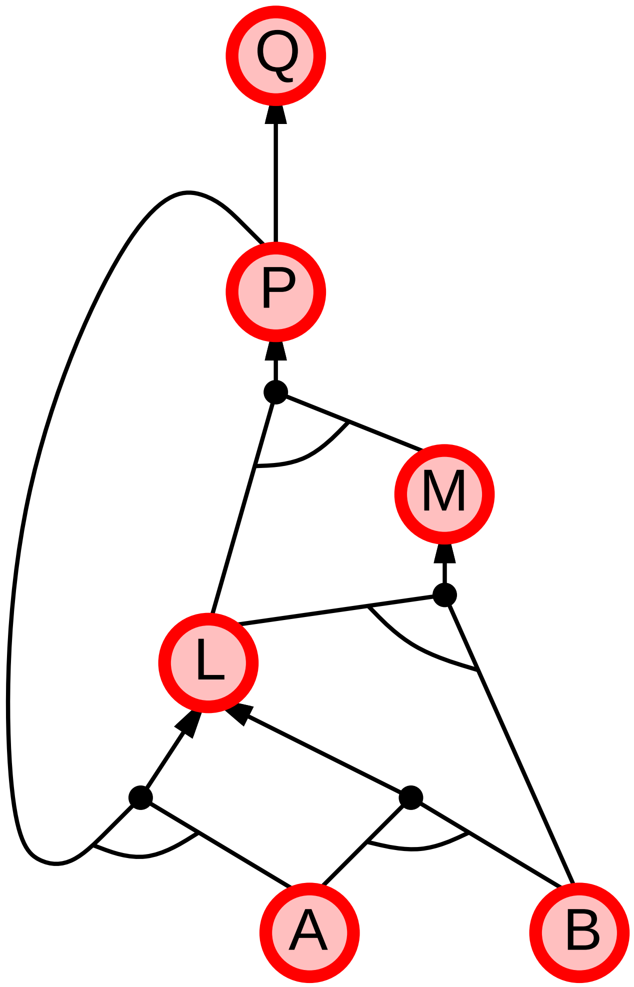
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Porovnání dopředného a zpětného řetězení

- **dopředné řetězení** je řízeno daty
  - automatické, nevědomé zpracování
  - např. rozpoznávání objektů, rutinní rozhodování
  - může udělat hodně nadbytečné práce bez vztahu k dotazu/cíli
- **zpětné řetězení** je řízeno dotazem
  - vhodné pro hledání odpovědí na konkrétní dotaz
  - např. “Kde jsou moje klíče?” “Jak se mám přihlásit na PGS?”
  - složitost zpětného řetězení může být mnohem menší než lineární vzhledem k velikosti KB

obecný inferenční algoritmus – rezoluce

zpracovává formule v konjunktivní normální formě (konjunkce disjunkcí literálů)

pro výrokovou logiku je rezoluce bezesporná a úplná