POLITECHNIKA WARSZAWSKA
WYDZIAŁ MECHANICZNY ENERGETYKI I LOTNICTWA

WPROWADZENIE DO SZTUCZNEJ INTELIGENCJI

NS 586

Dr inż. Franciszek Dul

Poziomy sztucznej inteligencji

Sztuczna świadomość ?

Uczenie się

Planowanie i podejmowanie decyzji

Uwzględnianie niepewności

Wnioskowanie logiczne

Rozwiązywanie problemów

✓

7. AGENT LOGICZNY

Agent logiczny

Pokażemy agentów którzy posiadają wiedzę o świecie, potrafią wyciągać z niej nowe wnioski na temat środowiska, a także umieją wykorzystać tę wiedzę do osiągnięcia swoich celów.

Inteligencja agenta celowego...

„Inteligencja” programu *Deep Blue* zafascynowała samego Kasparowa, który stwierdził po przegranym meczu

> *„...miałem wrażenie, że przeciwko mnie gra nowa jakość...”*

Firma IBM była jednak dużo bardziej sceptyczna:

> „... Deep Blue jest znacznie głupszy niż najgłupszy nawet człowiek...”

W istocie - Deep Blue wykorzystywał algorytmy oparte na dość prostych regułach poszukiwań.

Agent celowy nie grzeszy zbytnio inteligencją: program Deep Blue nie wie np., że żadna figura nie może być jednocześnie na dwóch polach, etc...

Bardziej inteligentny agent powinien posiadać wiedzę i wyciągać wnioski – taki jest **agent logiczny**.

7.1. Agent logiczny

Różnice w poziomie wiedzy agentów:

- Agent z refleksem może znaleźć rozwiązanie (np. drogę z Arad do Bukaresztu) jedynie przez łut szczęścia.
- Program szachowy potrafi wyznaczyć dopuszczalny ruch króla ale nie wie np., że dwa pionki nie mogą stać jednocześnie na tym samym polu szachownicy.
- Agent logiczny (z wiedzą) potrafi łączyć wiedzę ogólną z aktualnymi obserwacjami w celu wnioskowania o ukrytych cechach stanu aktualnego przed podjęciem działania.

Ta cecha agenta logicznego – myślenie racjonalne - ma zasadnicze znaczenie w przypadku gdy środowisko jest częściowo obserwowalne.

Myślenie racjonalne agenta logicznego jest oparte na ***wiedzy*** - zbiorze faktów i relacji pomiędzy nimi.

Możliwości agenta logicznego

Cechy agenta logicznego:

- Posiada reprezentację stanów i działań.
- Ma możliwość dołączania nowych obserwacji.
- Aktualizuje wewnętrzną reprezentację świata.
- **Wnioskuje o ukrytych własnościach świata.**
- Wnioskuje jakie działania są logicznie uzasadnione.

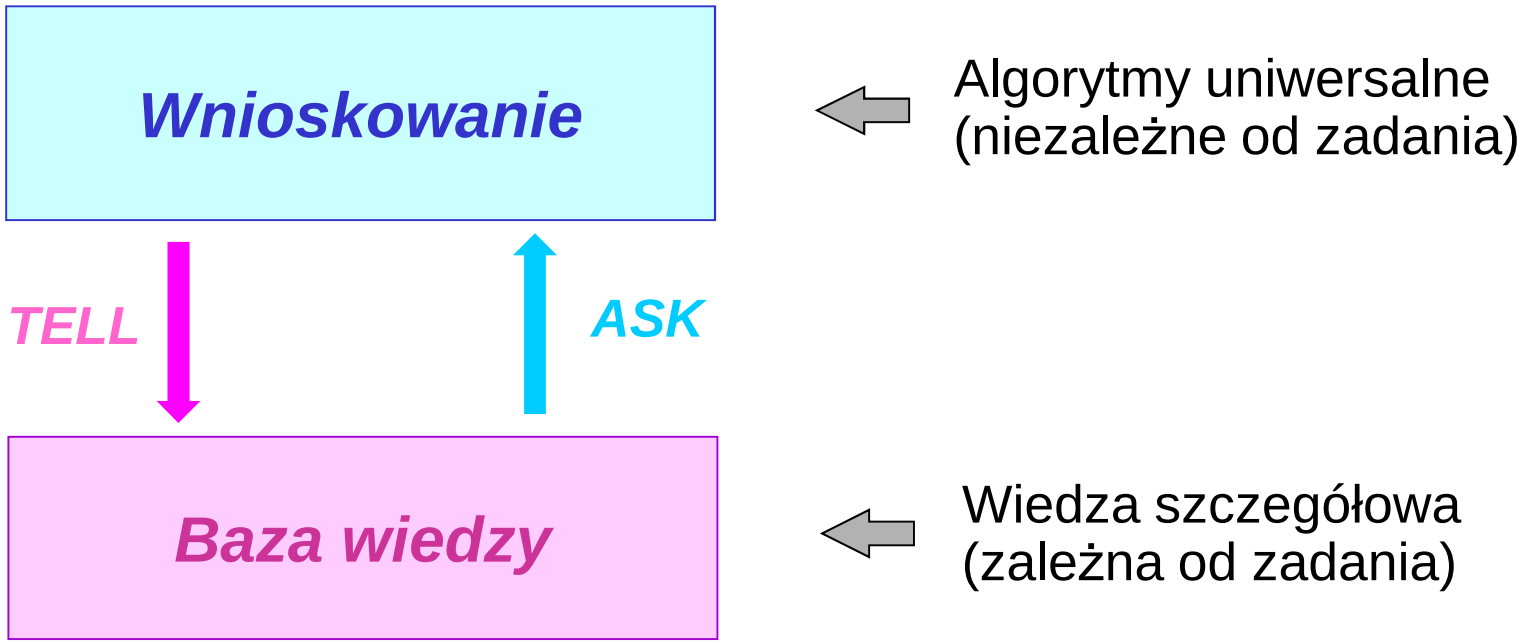
Agent logiczny ma więc dużo większe możliwości niż agent refleksowy lub rozwiązujący problemy.

Podejście logiczne dominowało zdecydowanie w badaniach i zastosowaniach AI w latach sześćdziesiątych i siedemdziesiątych XX wieku.

Możliwości agenta logicznego

Najważniejszym polem zastosowań agenta logicznego są bazy wiedzy i oparte na nich systemy ekspertowe.

Baza wiedzy (KB) jest to zbiór zdań zapisanych w języku formalnym opisujących cechy świata.

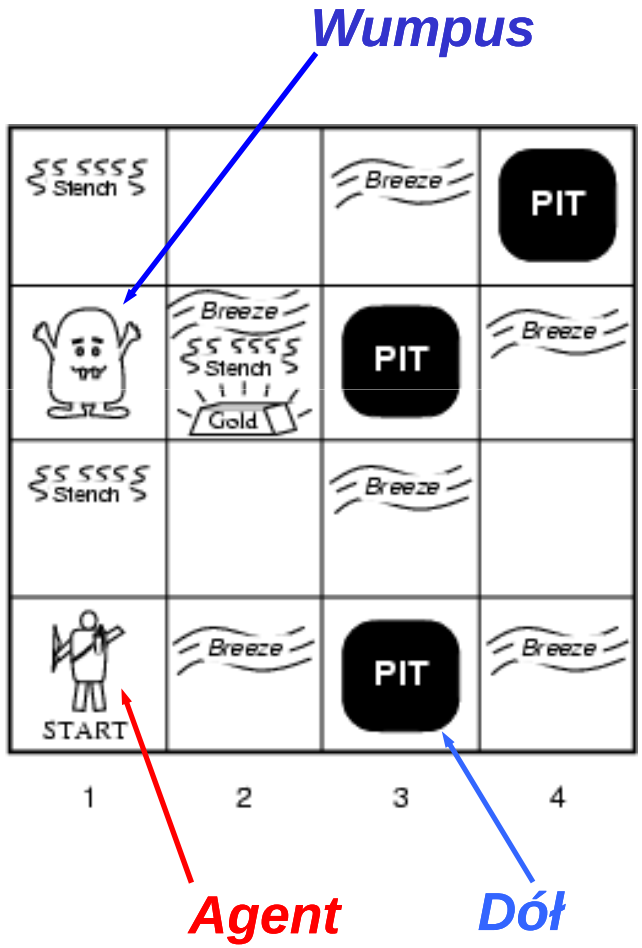


Przykład agenta logicznego: Świat Wumpusa

Agent logiczny w kultowej grze “Hunt the Wumpus” (ośmiobitowy komputerek Atari, lata 80. XX w.)

Struktura PEAS:

- Miara jakości (P)
  - złoto +*1000*, śmierć *-1000,*
  - krok *-1,* użycie strzały *-10*
- Środowisko (E)
  - pola przyległe do Wumpusa cuchną,
  - w polach przyległych do dołu wieje wiatr,
  - pola ze złotem błyszczą,
  - strzał zabija Wumpusa jeżeli agent jest skierowany w jego stronę,
  - agent ma tylko jedną strzałę,
  - można podnieść złoto gdy jest się w tym samym polu,
  - można położyć złoto w polu aktualnym.
- Działania (A)
  - w lewo, w prawo, naprzód, podnieś, upuść, strzał.
- Czujniki (S)
  - [*odór, wiatr, blask, zderzenie, krzyk*].



Charakterystyka świata Wumpusa

- Obserwowalność lokalna

  Możliwe są tylko obserwacje lokalne.
- Determinzm

  Wyniki działań są ściśle określone.
- Sekwencyjność

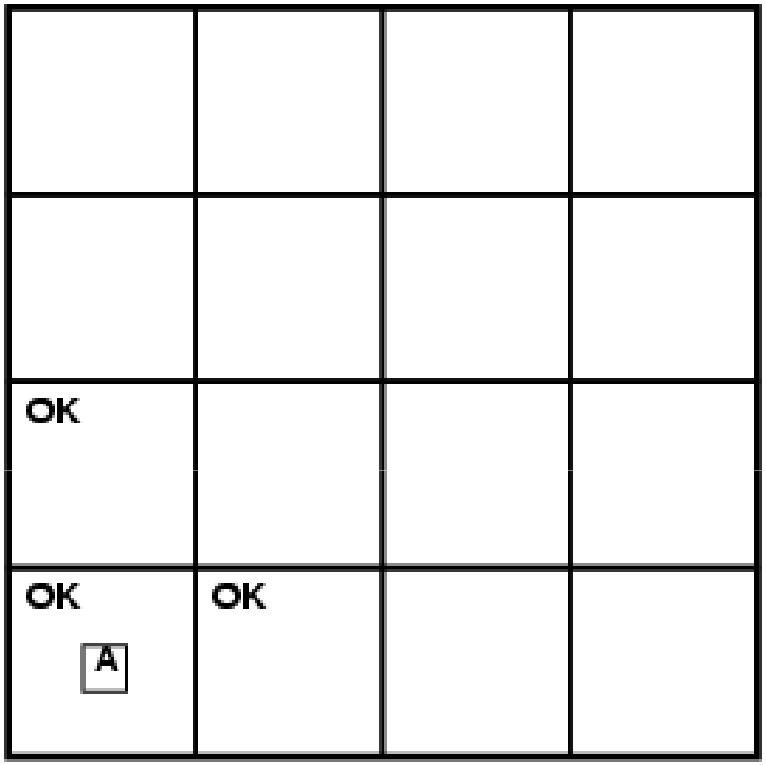
  Działania wykonywane są kolejno.
- Statyczność

  Wumpus i doły są nieruchome.
- Dyskretność

  Działania są wykonywane kolejno.
- Pojedyńczy agent

  Wumpus jest elementem środowiska, a nie przeciwnikiem.

Eksploracja logiczna świata Wumpusa



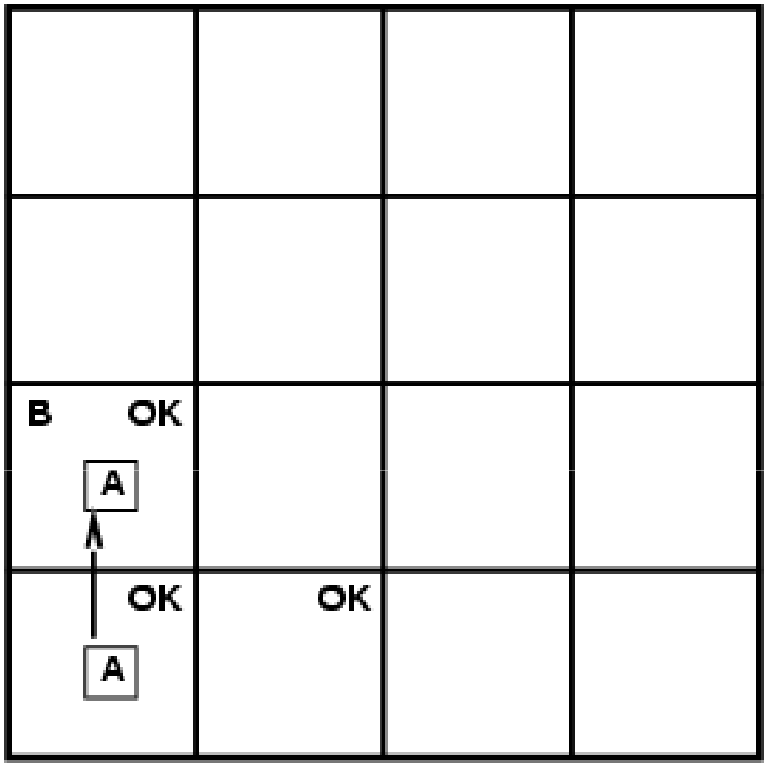
Baza danych zawiera początkowo tylko reguły środowiska.

Pozycja startowa agenta - (1,1)

Pierwsza obserwacja - [none, none, none, none, none]

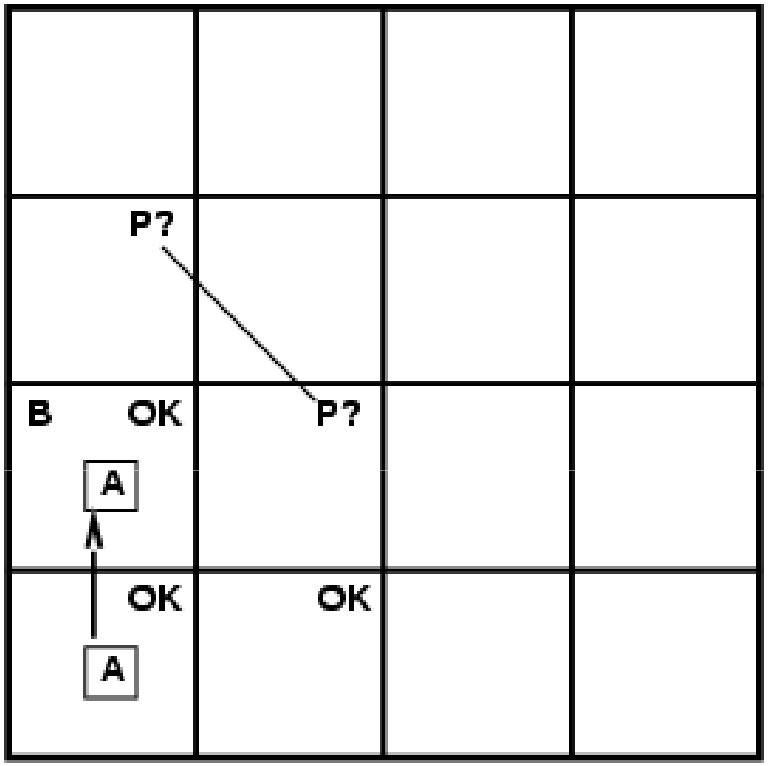
Należy przesunąć się do bezpiecznego pola np., (1,2)

Eksploracja świata Wumpusa



(1,2) wietrzyk, obserwacja = [none, breeze, none, none, none] oznacza, że w polu (2,2) lub (1,3) musi być dół

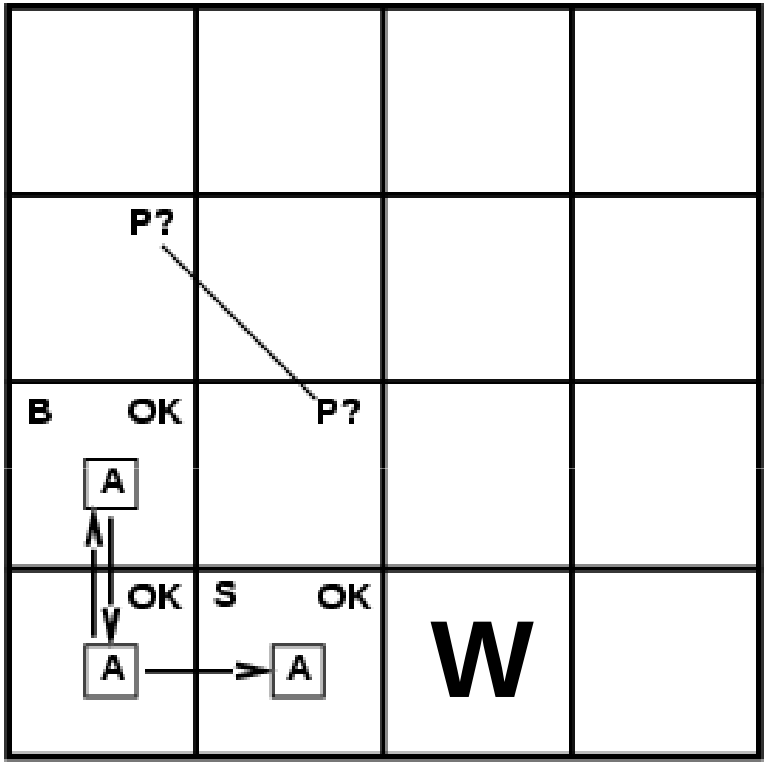
Eksploracja świata Wumpusa



(1,2) wietrzyk, obserwacja = [none, breeze, none, none, none] oznacza, że w polu (2,2) lub (1,3) musi być dół

Należy zatem wrócić do (1,1) i wypróbować następne bezpieczne pole.

Eksploracja świata Wumpusa



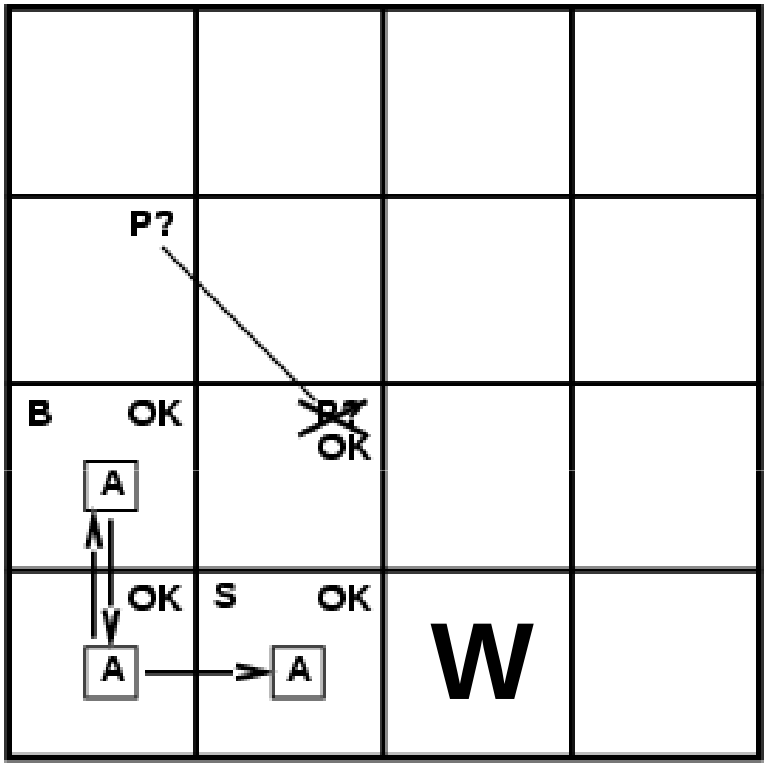
(2,1) odór, obserwacja = [stench, none, none, none, none] oznacza, że Wumpus musi być w polu (2,2) lub (3,1) i na pewno nie ma go w polu (1,1)

Wumpusa nie ma w polu (2,2), gdyż wtedy odór byłby wyczuwalny w polu (1.2)

Wumpus jest zatem w polu (3,1)

Ponieważ w polu (2,1) nie ma wiatru, to w (2,2) nie ma dołu, jest ono bezpieczne.

Eksploracja świata Wumpusa



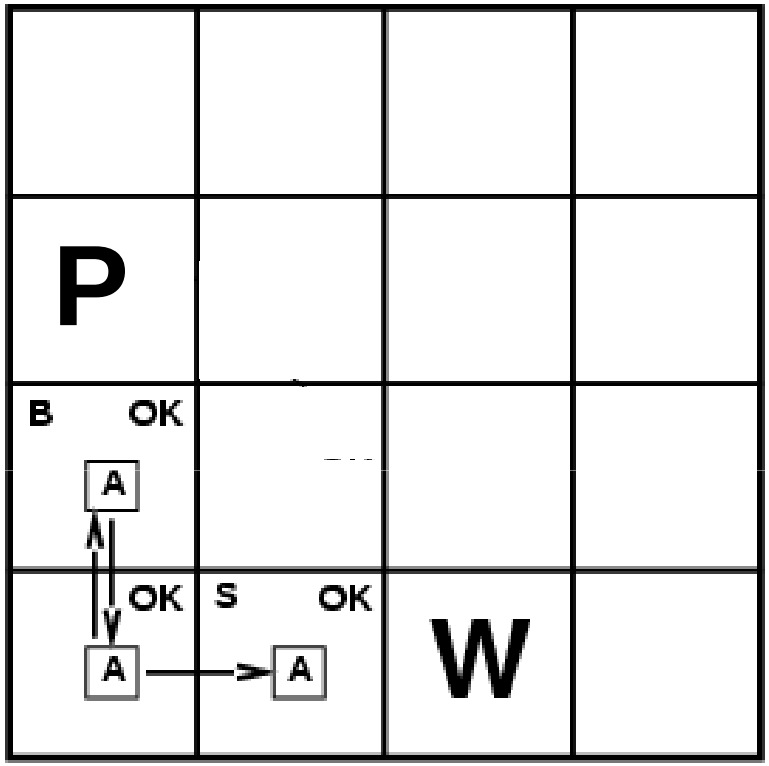
(2,1) odór, obserwacja = [stench, none, none, none, none] oznacza, że Wumpus musi być w polu (2,2) lub (3,1) i na pewno nie ma go w polu (1,1)

Wumpusa nie ma w polu (2,2), gdyż wtedy odór byłby wyczuwalny w polu (1.2)

Wumpus jest zatem w polu (3,1)

Ponieważ w polu (2,1) nie ma wiatru, to w (2,2) nie ma dołu, jest ono bezpieczne.

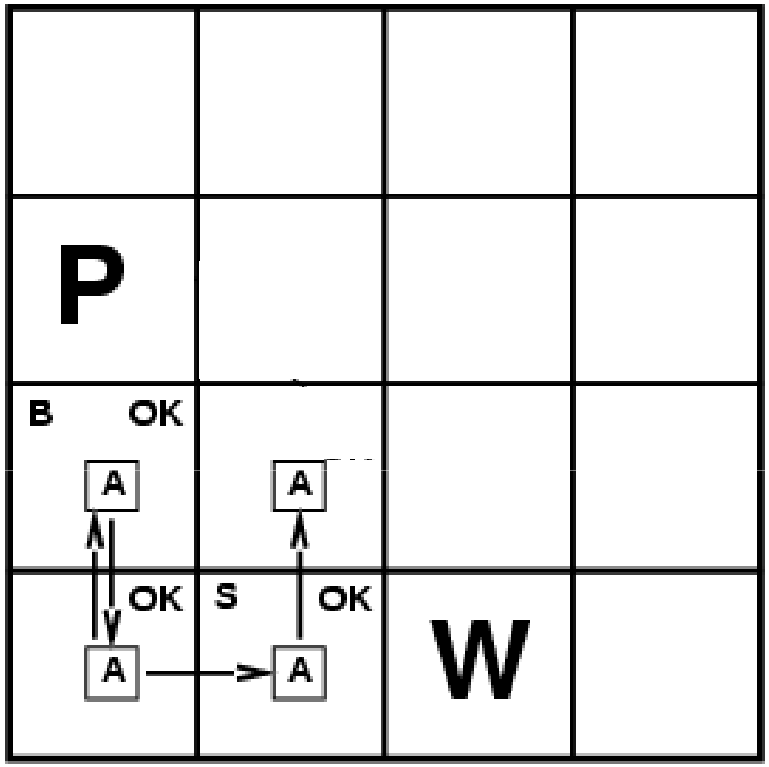
Eksploracja świata Wumpusa



Ponieważ w polu (1,2) jest wiatr, to dół musi być w polu (1,3)

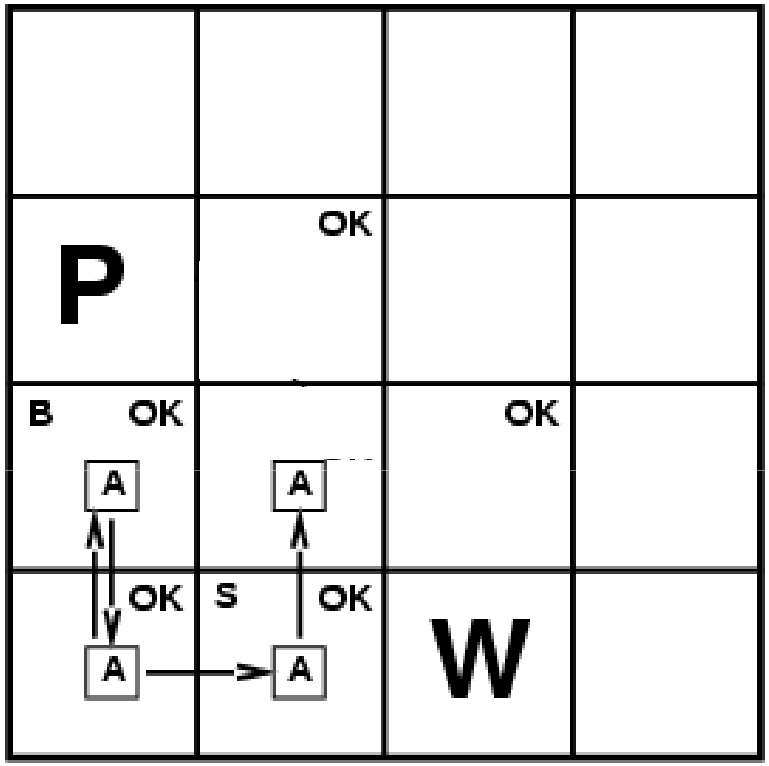
Można zatem przejść do bezpiecznego pola (2,2)

Eksploracja świata Wumpusa



W polu (2,2) obserwacja = [none, none, none, none, none] co oznacza, że pola (2,3) i (3,2) są bezpieczne

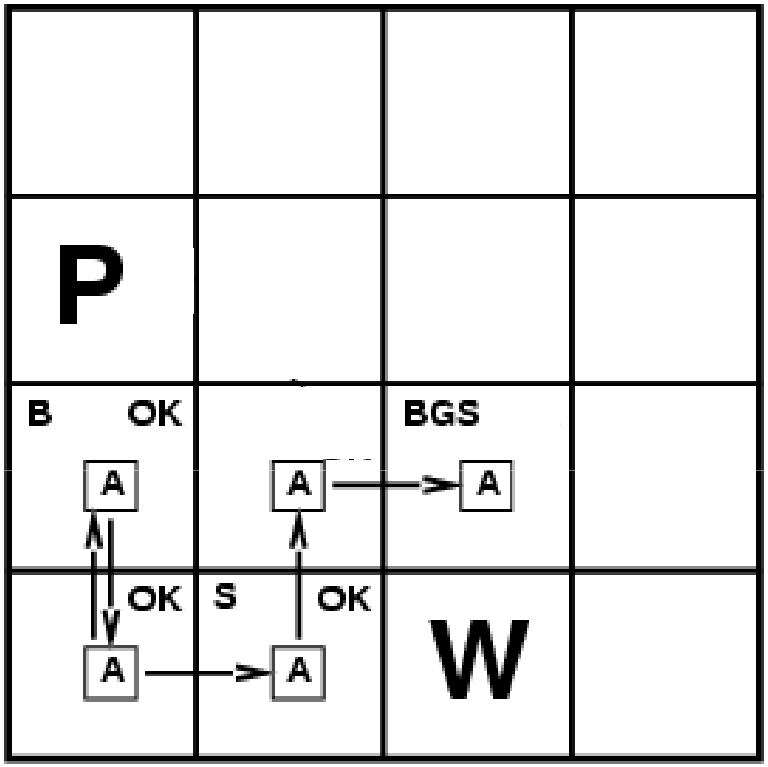
Eksploracja świata Wumpusa



W polu (2,2) obserwacja = [none, none, none, none, none] co oznacza, że pola (2,3) i (3,2) są bezpieczne

Można zatem przejść do bezpiecznego pola (3,2)

Eksploracja świata Wumpusa



W polu (3,2) obserwacja = [stench, breeze, glitter, none, none] co oznacza, że:

- można podnieść złoto które leży w polu (3,2),
- w polu (3,3) lub (4,2) jest dół,

itd...

7.3. Logika - modele i interpretacja

Czym jest logika?

- **Logika** jest formalnym językiem reprezentowania informacji umożliwiającym wyciąganie wniosków.
- **Zdanie logiczne** jest to wypowiedź w pewnym języku, która stwierdza określony stan rzeczy. Można mu przypisać **wartość logiczną**.
- **Składnia** (syntaktyka) określa sposób budowania zdań w języku logiki.
- **Semantyka** określa znaczenie zdań.

Świat lub **model** w logice to wybrany zbiór wartościowań wszystkich możliwych zdań logicznych.

Przykłady zdań logicznych w języku arytmetyki

- $x+2 \geq y$ jest zdaniem;
- $x+y > \{\}$ nie jest zdaniem;
- $x+2 \geq y$ jest prawdą w świecie w którym $x = 7, y = 1$
  ale jest fałszem w świecie w którym $x = 0, y = 6;$

Relacja pociągania/implikacji (*entailment*)

Relacja pociągania

$$KB \models \alpha$$

oznacza, że jedna rzecz wynika z drugiej.

Relacja $KB \models \alpha$ oznacza, że baza wiedzy *KB* implikuje zdanie α wtedy i tylko wtedy, gdy zdanie α jest prawdziwe w każdym świecie, w którym prawdziwa jest baza wiedzy *KB*.

Przykład:

Jeżeli baza wiedzy *KB* zawiera zdania:

„Wygrał X” oraz *„Wygrał Y”*

to baza *KB* pociąga m.in. zdania

„Wygrał X lub wygrał Y” , *„Wygrał X i wygrał Y”*

Pociąganie jest relacją pomiędzy zdaniami (tj. składnią) opartą na znaczeniu (semantyce).

UWAGA! Pociąganie jest czymś więcej niż implikacja $\Rightarrow$

Relację pociągania można również interpretować jako ***niejawne zawieranie logiczne***.

- Niech baza wiedzy *KB* zawiera pewną liczbę zdań, np.:

  „Wygrał X” ,

  „Wygrał Y”

- Korzystając z semantyki, tj. opierając się na ***znaczeniu*** tych zdań można zbudować (wyprowadzić) inne zdania, które nie należą ***jawnie*** do bazy *KB,* ale ***logicznie z nich wynikają***, *np.:*

  „Wygrał X lub wygrał Y”,

  „Wygrał X i wygrał Y”,

  „Nie przegrał X i wygrał Y”, ...

Relację pociągania można zatem rozumieć tak:

Zdania należące do bazy *KB* mają semantycznie dużo szersze znaczenie niż to, które wyrażają jawnie.

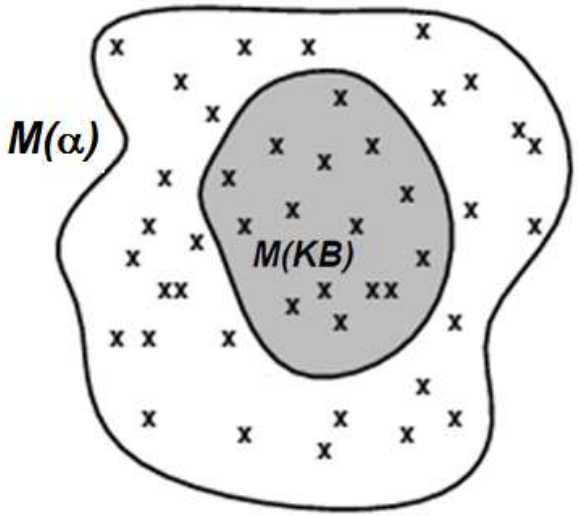
Pociąganie stanowi zatem podstawę wnioskowania logicznego opartego na bazach wiedzy.

Modele logiczne

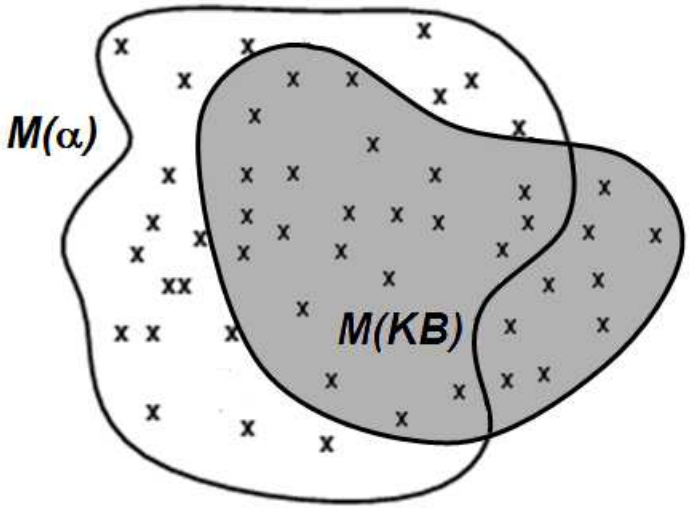
Modele logiczne są formalnymi reprezentacjami światów, wobec których można orzekać prawdziwość wyrażeń.

- Mówimy, że ***m*** jest modelem zdania α jeżeli α jest prawdziwe w ***m***.
- *M*(α) jest zbiorem wszystkich modeli dla α.
- Zatem *KB* |= α wtedy i tylko wtedy gdy *M(KB)* $\subseteq$ *M*(α).

KB |= α



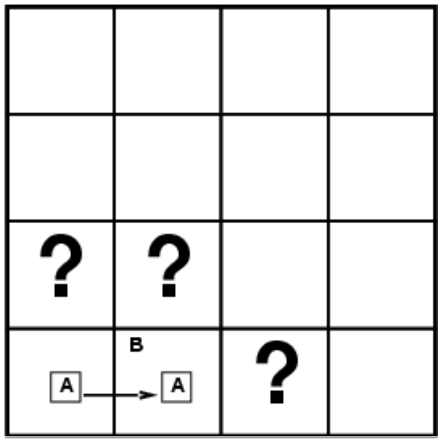
KB |≠ α



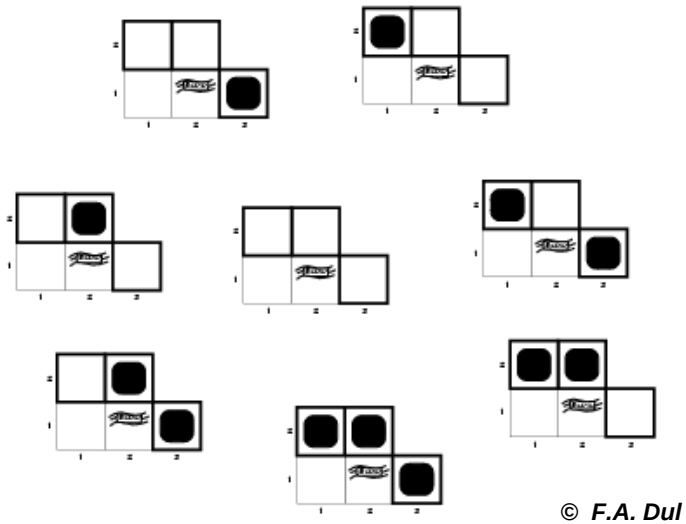
Pociąganie w świecie Wumpusa

Przykład zastosowania logiki do analizy świata Wumpusa.

- Stan świata:
  - po stwierdzeniu, że niczego nie ma w polu (1,1),
  - po ruchu w prawo,
  - po wykryciu wiatru w polu (2,1).

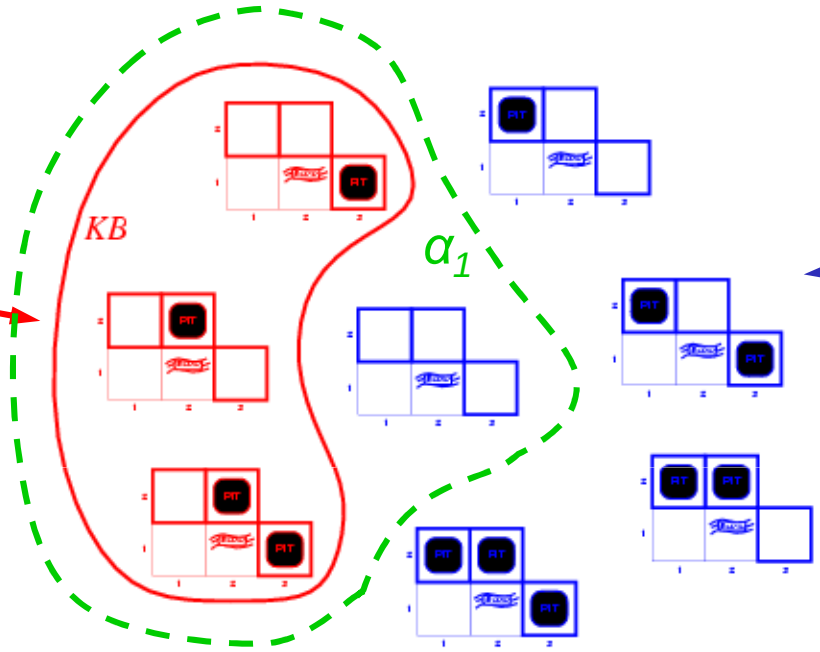


- Rozważmy możliwe modele bazy wiedzy KB zakładając, że w polach ***?*** mogą być tylko doły (gdyż nie czuć odoru).
- Są trzy pola ***?,*** każdemu można przypisać wartość logiczną 0 (nie ma dołu) lub 1 (jest dół).
- Istnieje osiem modeli dla stanu aktualnego opisujących wszystkie możliwe położenia dołów w polach ***?*** (niezależnie od poczynionych obserwacji).



Pociąganie/implikacja w świecie Wumpusa

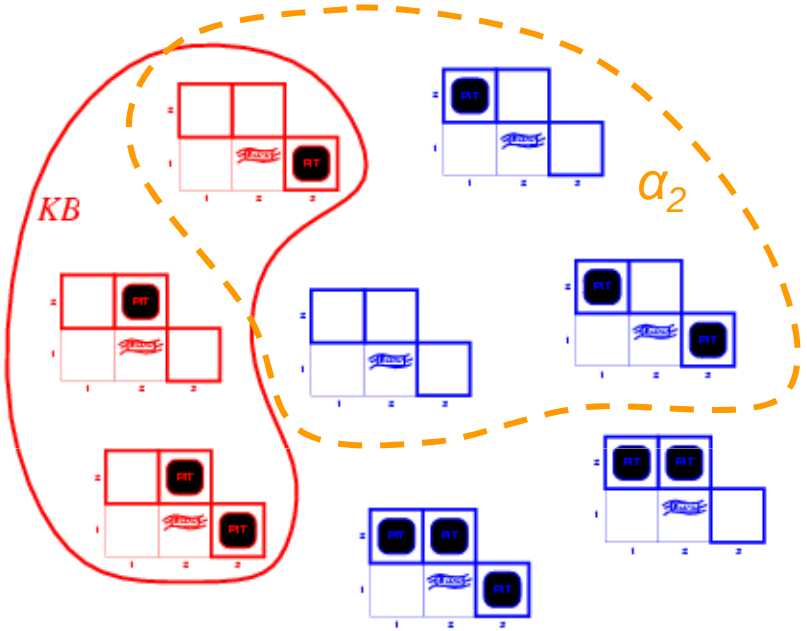
Te stany należą do KB gdyż w polu (1,1) nie ma wiatru, zaś w polu (2,1) jest wiatr



Te stany nie należą do KB gdyż w polu (1,1) nie ma wiatru

- *KB* = zasady świata Wumpusa + obserwacje
- α_1 = ”Pole (1,2) jest bezpieczne”,
- Pociąganie *KB* |= α_1 dowodzi się poprzez badanie modelu.
- $M(KB) \subseteq M(\alpha_1)$, zatem *KB* |= α_1.

Pociąganie/implikacja w świecie Wumpusa



- *KB* = zasady świata Wumpusa + obserwacje
- α_2 = ”(2,2) jest bezpieczne”,
- $M(KB) \not\subseteq M(\alpha_2)$, zatem $KB \not\models \alpha_2$

Przykład Wumpusa pokazuje, że sprawdzenie wszystkich przypadków pozwala dowieść prawdziwości zdania α.

Algorytmy wnioskowania logicznego w bazach wiedzy

Zapis $KB \vdash_i \alpha$ oznacza, że zdanie α może być wywiedzione z bazy *KB* za pomocą *algorytmu wnioskowania i.*

Jeżeli algorytm wnioskowania wywodzi z *KB* ***tylko zdania prawdziwe***, to jest on ***poprawny*** *(soundness)* lub *zachowujący prawdę (truth preserving)*

> Algorytm *i* jest poprawny jeżeli $KB \vdash_i \alpha$ zapewnia, że $KB \models \alpha$ jest prawdziwe.

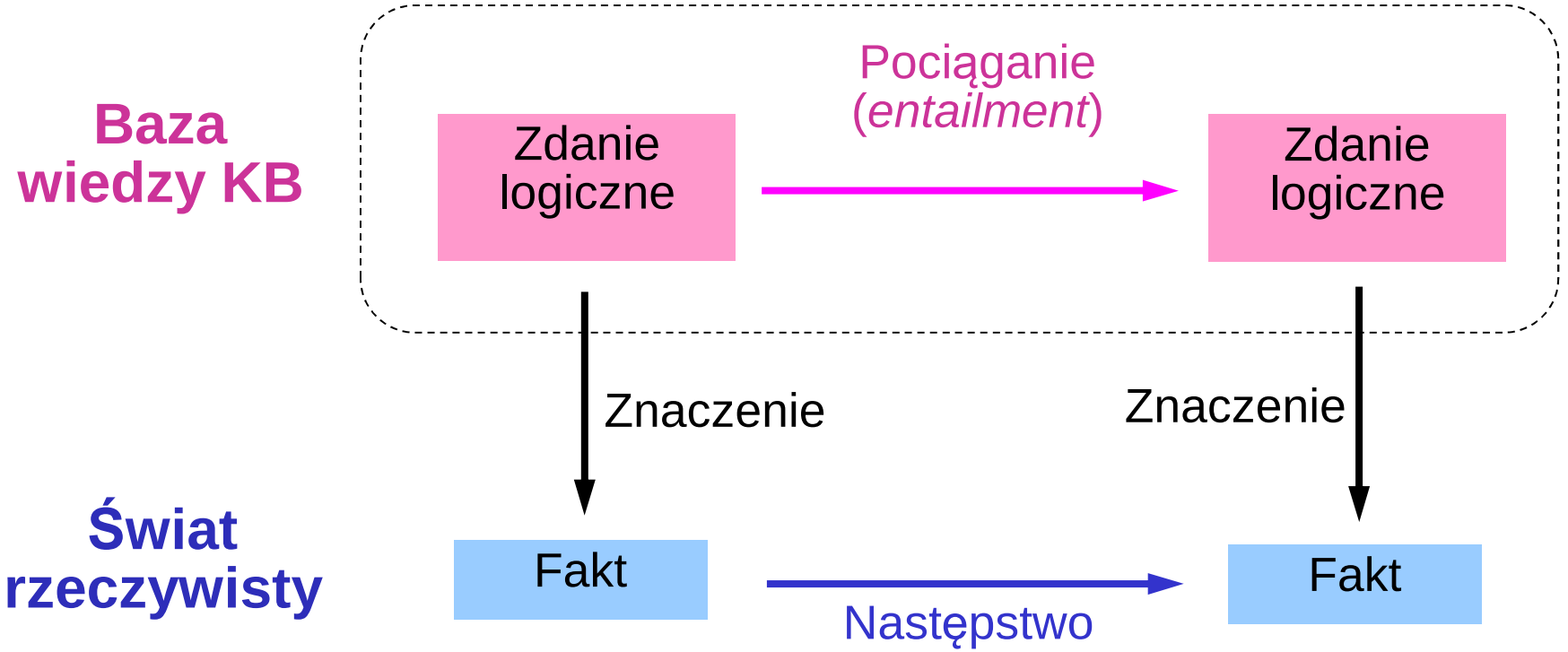
Algorytm jest ***kompletny (zupełny)*** gdy umożliwia wyprowadzenie ***każdego*** zdania, które wynika z *KB*.

> Algorytm *i* jest kompletny jeżeli $KB \models \alpha$ zapewnia, że $KB \vdash_i \alpha$ jest prawdziwe.

Algorytm poprawny i kompletny pozwala odpowiedzieć na każde pytanie na podstawie wiedzy zgromadzonej w bazie *KB*.

Baza danych a świat rzeczywisty

Jeżeli baza wiedzy *KB* zawiera zdania które są prawdziwe w świecie rzeczywistym, to ***każde*** zdanie wywnioskowane z bazy *KB* za pomocą algorytmu poprawnego i zupełnego jest również prawdziwe w świecie rzeczywistym.



Istota agenta logicznego

Baza wiedzy agenta zawiera pewną liczbę asercji (stwierdzeń uważanych za prawdziwe) o świecie.

Liczba wszystkich zdań wynikających z bazy wiedzy jest o wiele większa niż liczba asercji w bazie.

Mechanizm wnioskowania pozwala uzyskać wiedzę ukrytą (niejawną) zawartą w bazie wiedzy.

Agent logiczny ma więc tę przewagę nad agentem celowym, że dysponuje dużo większymi zasobami informacji.

Informacja w bazie wiedzy jest ponadto „spakowana” w postaci asercji.

Agent logiczny jest więc zdecydowanie bardziej inteligentny od agenta celowego.

Jak działa agent logiczny?

Uzyskanie takiej wiedzy ukrytej zawartej w bazie wiedzy wymaga wyciągania wniosków.

Aby wykorzystać posiadaną wiedzę agent logiczny musi zatem efektywnie wnioskować.

Wnioskowanie będzie efektywne, gdy można je zautomatyzować.

Umożliwia to logika formalna.

7.4. Rachunek zdań

Rachunek zdań jest najprostszą logiką, która pozwala zapisać podstawowe informacje o świecie.

Składnia (syntaktyka)

Wyrażenia proste, niepodzielne (*atomic sentences*): B, W_{12} ,...

Niech S, S_1 oraz S_2 będą wyrażeniami prostymi. Za pomocą **spójników** (*connectives*) tworzy się wyrażenia złożone:

- $\neg S$ (negacja)
- $S_1 \wedge S_2$ (koniunkcja)
- $S_1 \vee S_2$ (alternatywa)*)
- $S_1 \Rightarrow S_2$ (implikacja)
- $S_1 \Leftrightarrow S_2$ (równoważność)

Literały (*literals)*: wyrażenia proste lub ich negacje: B, $\neg C$...

Wyrażenia złożone (klauzule, *clauses*): wyrażenia proste połączone spójnikami np. $A \vee B \wedge C$.

---

*) UWAGA! W języku polskim znaczenia pojęć *alternatywa* i *dysjunkcja* są odwrócone: $\vee$ alternatywa – *disjunction*, dysjunkcja ≡ *NotAND*.

Semantyka (znaczenie)

Semantyka polega na określeniu wartości logicznej zdań.

Każdy model określa wartość logiczną zdań: prawdę lub fałsz.

Przykład Wumpusa: P_{12} = *fałsz*, P_{22} = *prawda*, P_{31} = *fałsz.*

Zasady określania wartości logicznej zdań względem modelu *m*

- $\neg S$ jest prawdziwe wtedy i tylko wtedy, gdy S jest fałszywe;
- $S_1 \wedge S_2$ jest prawdziwe wtedy i tylko wtedy, gdy S_1 jest prawdziwe **i** S_2 jest prawdziwe;
- $S_1 \vee S_2$ jest prawdziwe wtedy i tylko wtedy, gdy S_1 jest prawdziwe **lub** S_2 jest prawdziwe;
- $S_1 \Rightarrow S_2$ jest prawdziwe wtedy i tylko wtedy, gdy S_1 jest fałszywe **lub** S_2 jest prawdziwe;

  Innymi słowy: $S_1 \Rightarrow S_2$ jest fałszywe wtedy i tylko wtedy, gdy S_1 jest prawdziwe **i** S_2 jest fałszywe;
- $S_1 \Leftrightarrow S_2$ jest prawdziwe wtedy i tylko wtedy, gdy $S_1 \Rightarrow S_2$ jest prawdziwe **i** $S_2 \Rightarrow S_1$ jest prawdziwe;

Tabela wartości logicznych operatorów

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

Za pomocą powyższych reguł można rekurencyjnie wyznaczyć wartość logiczną dowolnego zdania złożonego, np.

P_{12} = *fałsz*, P_{22} = *prawda*, P_{31} = *fałsz.*

$$\begin{aligned} \neg P_{12} \wedge (P_{22} \vee P_{31}) &= \textit{prawda} \wedge (\textit{prawda} \vee \textit{fałsz}) = \\ &= \textit{prawda} \wedge \textit{prawda} = \\ &= \textit{prawda} \end{aligned}$$

UWAGA! Wartość logiczna może kłócić się z potocznym rozumieniem sensu zdania, np. zdanie

„Jeżeli Ziemia jest płaska, to jestem człowiekiem”

jest prawdziwe mimo, że nie ma sensu.

Baza wiedzy w świecie Wumpusa

Niech $P_{i,j}$ będzie prawdziwe, jeżeli w polu (i,j) jest dół (P).
Niech $B_{i,j}$ będzie prawdziwe, jeżeli w polu (i,j) jest wiatr (B).
Baza wiedzy KB zawiera następujące zdania:

- W polu (1,1) nie ma dołu

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

- W polu jest wiatr, jeżeli dół jest w polu przyległym:

$$R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1}),$$

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

- Obserwacje agenta w polach (1,1) oraz (2,1):

$$R_4 : \neg B_{1,1}$$

$$R_5 : \; B_{2,1}$$

Baza wiedzy może być wyrażona jako zdanie złożone

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

Należy wywnioskować, czy zdanie $\alpha = P_{2,2}$ wynika z bazy KB,

$$KB \models \alpha ?$$

Wnioskowanie przez wyliczenie

Wnioskowanie może być przeprowadzone poprzez wyliczenie, czyli sprawdzenie, czy zdanie $\alpha = P_{2,2}$ jest prawdziwe dla wszystkich modeli dla których baza KB jest prawdziwa.

Tabela wartości logicznych dla zdań tworzących bazę KB

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	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*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
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*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
true	*true*	*true*	*true*	*true*	*true*	*true*	*false*	*true*	*true*	*false*	*true*	*false*

Baza *KB* jest prawdziwa dla trzech modeli, dla których

$$R_1 \wedge R_2 \wedge R_3 \wedge R_4 \wedge R_5 = True.$$

Zdanie $\alpha = P_{2,2}$ nie wynika zatem z bazy *KB*, gdyż dla jednego z modeli dla którego baza jest prawdziwa jest ono fałszywe.

Równoważność logiczna

Dwa zdania logiczne są logicznie równoważne wtedy i tylko wtedy, gdy są prawdziwe dla tych samych modeli,

$$\alpha \equiv ß \text{ wtedy i tylko wtedy, gdy } \alpha \models ß \text{ i } ß \models \alpha$$

$(\alpha \wedge \beta) \equiv (\beta \wedge \alpha)$	przemienność $\wedge$
$(\alpha \vee \beta) \equiv (\beta \vee \alpha)$	przemienność $\vee$
$((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma))$	łączność $\wedge$
$((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma))$	łączność $\vee$
$\neg(\neg\alpha) \equiv \alpha$	eliminacja podwójnej negacji
$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$	kontrapozycja
$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta)$	eliminacja implikacji
$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$	eliminacja równoważności
$\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$	prawo de Morgana
$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta)$	prawo de Morgana
$(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma))$	rozdzielność $\wedge$ względem $\vee$
$(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$	rozdzielność $\vee$ względem $\wedge$

Zasadność i spełnialność. Dowodzenie

Zdanie logiczne jest **zasadne** (*validity*), jeżeli jest prawdziwe dla *wszystkich* modeli,

np.: $True, \quad A \vee \neg A, \quad A \Rightarrow A, \quad (A \wedge (A \Rightarrow B)) \Rightarrow B$

Dowodzenie pociągania poprzez **dedukcję**

$$(KB \models \alpha) \Leftrightarrow (KB \Rightarrow \alpha) \quad \text{jest } \textit{zasadne}$$

Zdanie logiczne jest **spełnialne** (*satisfability*), jeżeli jest prawdziwe dla *pewnych* modeli, (np. $A \vee B$), zaś niespełnialne, jeżeli nie jest prawdziwe dla *żadnego* modelu (np. $A \wedge \neg A$).

Dowodzenie pociągania poprzez **sprowadzanie do niedorzeczności** (łac.*"reductio ad absurdum"*).

$$(KB \models \alpha) \Leftrightarrow (KB \wedge \neg \alpha) \quad \text{jest } \textit{niespełnialne}$$

Dedukcja i sprowadzanie do niedorzeczności stanowią podstawowe metody dowodzenia twierdzeń.

7.5. Metody dowodzenia w rachunku zdań

Metody dowodzenia można podzielić na dwie grupy:

- **Zastosowanie reguł dowodzenia**:
  - Poprawne wyprowadzanie nowych zdań ze starych.
  - Dowód jako ciąg reguł dowodzenia.

    Użycie reguł logiki w algorytmach przeszukiwania.
  - Dowodzenie wymaga zazwyczaj przekształcenia zdań logicznych do tzw. *postaci normalnej*.

- **Sprawdzanie modelu (tautologia)**:
  - Sporządzenie tablicy prawdziwości (koszt wykładniczy),
  - Zastosowanie algorytmu przeszukiwania,
  - Przeszukiwania heurystyczne w przestrzeni modeli, np.: minimalnych konfliktów, najszybszego wzrostu.

Metody te są poprawne, lecz nie są zupełne.

Reguły dowodzenia (*inference rules*)

Reguły odrywania ***Modus Ponens*** i ***Modus Tollens***

$$\frac{\alpha \Rightarrow \beta, \quad \alpha}{\beta} \qquad \frac{\alpha \Rightarrow \beta, \ \neg\beta}{\neg\alpha}$$

(przecinek oznacza $\wedge$)

Eliminacja koniunkcji $\wedge$

$$\frac{\alpha \wedge \beta}{\alpha} \qquad \frac{\alpha \wedge \beta}{\beta}$$

Eliminacja implikacji $\Rightarrow$

$$\frac{\alpha \Rightarrow \beta}{\neg\alpha \vee \beta}$$

Eliminacja równoważności $\Leftrightarrow$

$$\frac{\alpha \Leftrightarrow \beta}{(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)} \qquad \frac{(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)}{\alpha \Leftrightarrow \beta}$$

Eliminacja (*skracanie*, *resolution*)

Postać normalna koniunktywna (*Conjunctive Normal Form*, *CNF*) jest to *koniunkcja alternatyw wyrażeń prostych*.

Przykład: $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$

Wyrażenia komplementarne – jedno jest negacją drugiego:

$$A \equiv \neg B \qquad (A \wedge \neg B \equiv True)$$

Zasada eliminacji wyrażeń komplementarnych

Jeżeli zdania P_i oraz Q_j w postaci CNF są ***wyrażeniami komplementarnymi,***

$$P_i \wedge \neg Q_j \equiv True$$

to można je wyeliminować tworząc alternatywę zdań,

$$\frac{P_1 \vee \ldots \vee \mathbf{P_i} \vee \ldots \vee P_k, \qquad (\text{domyślnie } \wedge) \qquad Q_1 \vee \ldots \vee \mathbf{Q_j} \vee \ldots \vee Q_n}{P_1 \vee \ldots \vee P_{i-1} \vee P_{i+1} \vee \ldots \vee P_k \vee Q_1 \vee \ldots \vee Q_{j-1} \vee Q_{j+1} \vee \ldots \vee Q_m}$$

W rachunku zdań eliminacja wyrażeń komplementarnych jest poprawna i zupełna.

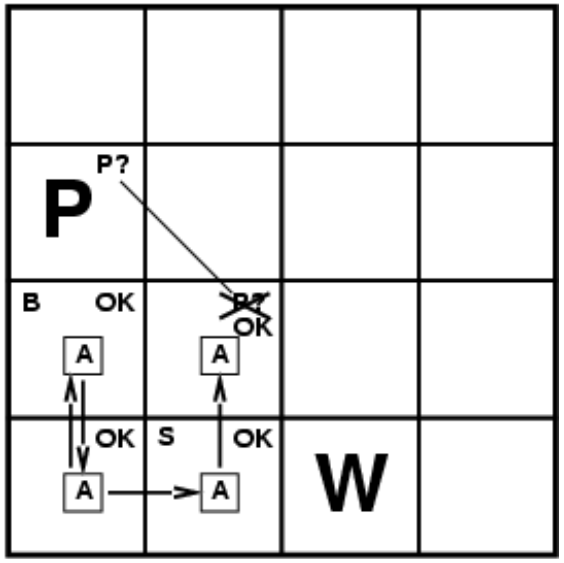
Eliminacja

Przykład:

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

$P_{2,2} \wedge \neg(\neg P_{2,2}) \equiv True$

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



Znaczenie: Jeżeli dół jest w polu (1,3) lub w polu (2,2) oraz nie ma dołu w polu (2,2), to dół jest w polu (1,3).

Dowód pociągania zdania α przez bazę *KB* przeprowadza się przez sprowadzenie do niedorzeczności (zaprzeczenie), tj. pokazanie, że $(KB \wedge \neg \alpha)$ jest niespełnialne.

W tym celu należy z $(KB \wedge \neg \alpha)$ wyprowadzić zdanie puste { }.

Sprowadzanie do postaci normalnej CNF

Przykład: sprowadzenie do postaci CNF wyrażenia

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

1. Eliminacja równoważności

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

2. Eliminacja implikacji

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

3. Eliminacja negacji wyrażenia za pomocą prawa de Morgana

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

4. Zastosowanie rozdzielności alternatywy względem koniunkcji

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

Wyrażenie w postaci CNF jest na ogół zupełnie niezrozumiałe, ale za to nadaje się do automatycznego dowodzenia.

Przykład dowodzenia

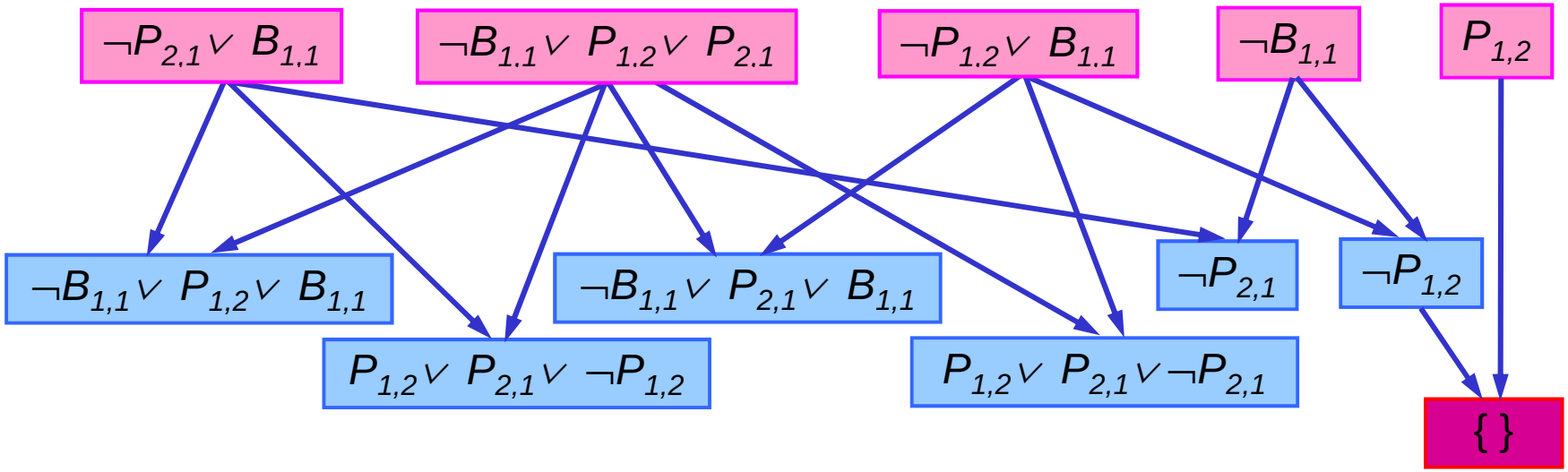
Baza wiedzy świata Wumpusa $(B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1}$

Baza wiedzy w postaci CNF

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

Zdanie dowodzone $\alpha = \neg P_{1,2}$ więc do bazy dołączamy $\neg(\neg P_{1,2})$

Dowodzenie zdania α przez zaprzeczenie metodą eliminacji



Puste zdanie $P_{1,2} \wedge \neg P_{1,2}$ dowodzi pociągania $KB \models \alpha$

„Przy okazji” dowiedliśmy (trochę niepotrzebnie) wielu innych zdań logicznych.

Dowodzenie w przód i wstecz

Zdanie Horna jest alternatywą literałów z których co najwyżej jeden jest pozytywny (np. $\neg A \vee \neg B \vee \neg C \vee D$).

Znaczenie zdań Horna wynika z faktu, że są one równoważne implikacjom - naturalnej postaci opisu świata:

$$(\neg A \vee \neg B \vee \neg C \vee D) \equiv \neg(A \wedge B \wedge C) \vee D \equiv (A \wedge B \wedge C) \Rightarrow D$$

Baza wiedzy ma najczęściej postać koniunkcji zdań Horna

$$KB = H_1 \wedge H_2 \wedge \ldots \wedge H_n$$

Reguła ***Modus Ponens*** dla baz wiedzy w postaci Horna

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

Do wnioskowania z baz wiedzy w formie Horna mogą być użyte algorytmy wyszukiwania w przód i wstecz.

Algorytmy te dobrze „pasują” do zadań wnioskowania, a ich koszt zależy co najwyżej liniowo od rozmiaru bazy *KB*.

Dowodzenie w przód (*forward chaining*)

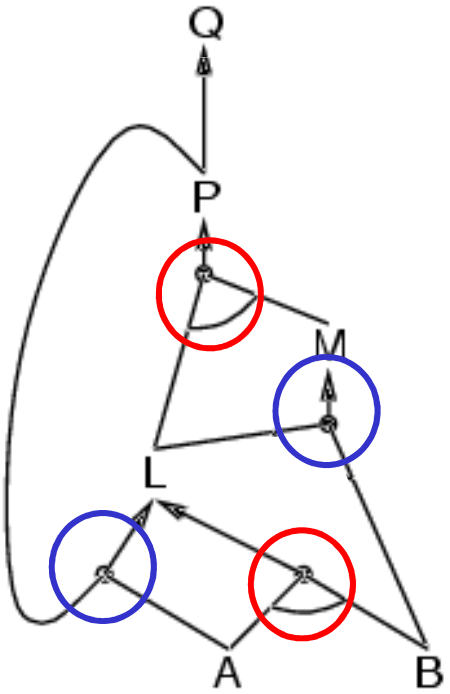
Idea:

- zastosować dowolną regułę której przesłanki są spełnione w bazie *KB* a uzyskany wniosek dodać do bazy *KB,*
- powtarzać tę procedurę do chwili, gdy zapytanie *Q* zostanie znalezione (i ew. dodane do *KB)*.

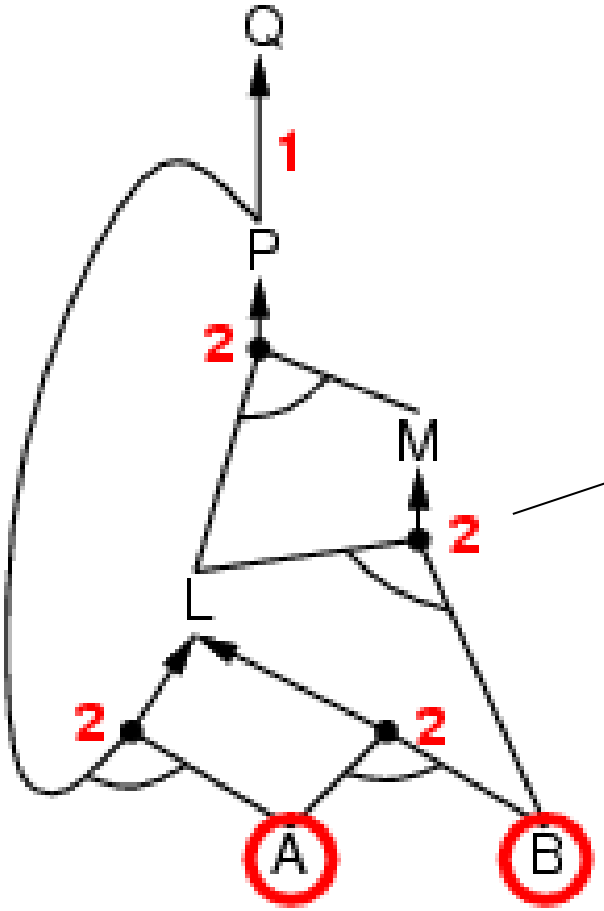
Algorytm dowodzenia w przód jest poprawny i zupełny w bazie wiedzy w postaci Horna.

Graf wnioskowania logicznego (*AND-OR graph*):

- koniunkcja - krawędzie zaznaczone łukiem, ***wszystkie*** przesłanki muszą być dowiedzione;
- alternatywa - krawędzie bez łuku, ***dowolna*** przesłanka musi być dowiedziona;



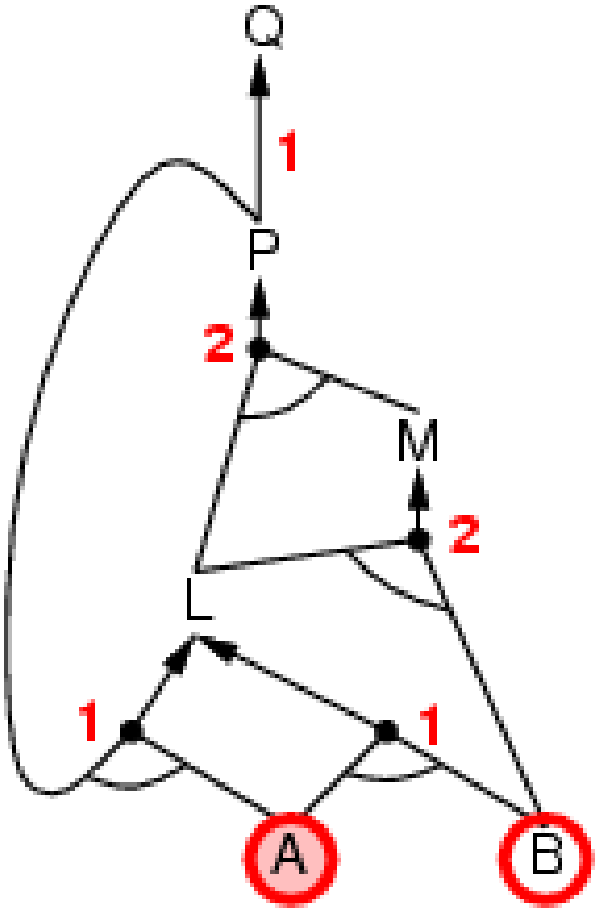
Dowodzenie w przód - przykład



liczba koniecznych dowodów w węźle

Dowodzenie w przód - przykład

Zaczynamy od dowiedzionych zdań *A, B*...



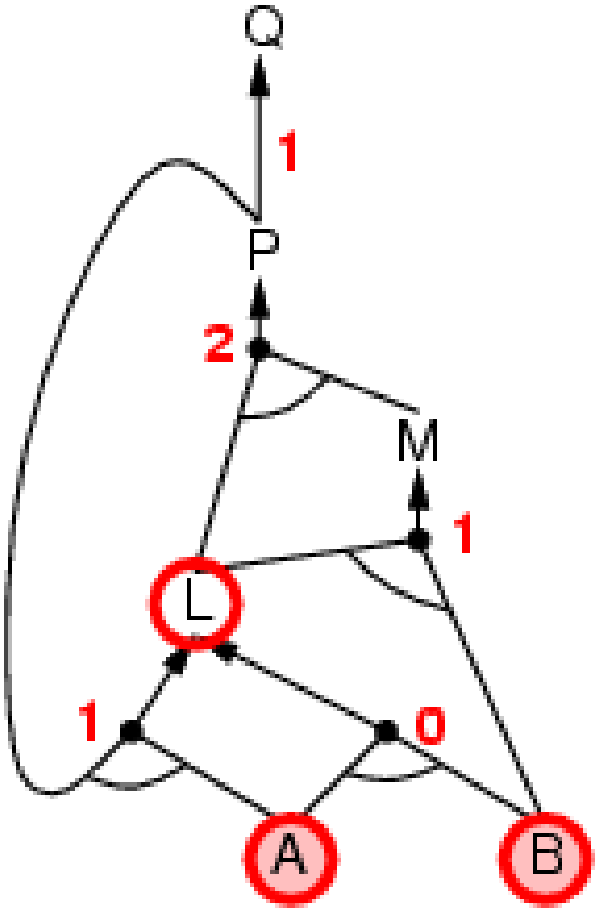
A, *B*

Dowodzenie w przód - przykład

... dowodzimy kolejne zdania...

$A \wedge B \Rightarrow L$

$A,\ B$



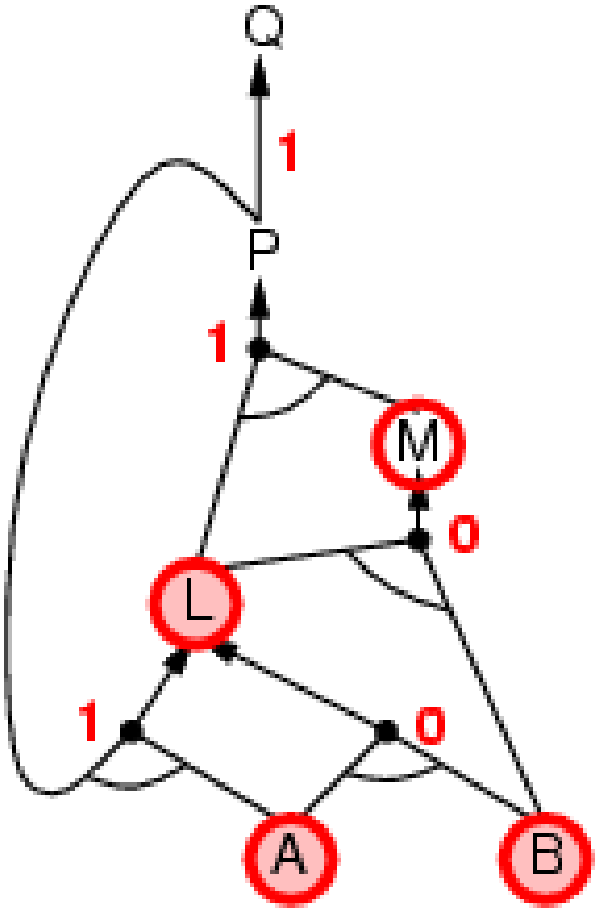
Dowodzenie w przód - przykład

... dowodzimy kolejne zdania...

$L \wedge B \Rightarrow M$

$A \wedge B \Rightarrow L$

$A,\ B$



Dowodzenie w przód - przykład

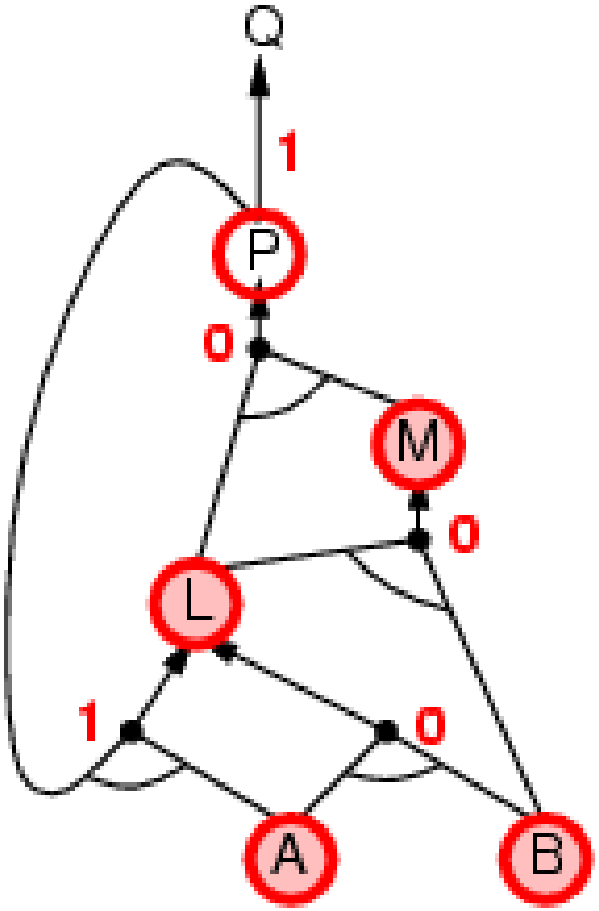
... dowodzimy kolejne zdania...

$L \wedge M \Rightarrow P$

$L \wedge B \Rightarrow M$

$A \wedge B \Rightarrow L$

$A,\ B$



Dowodzenie w przód - przykład

... dowodzimy kolejne zdania...

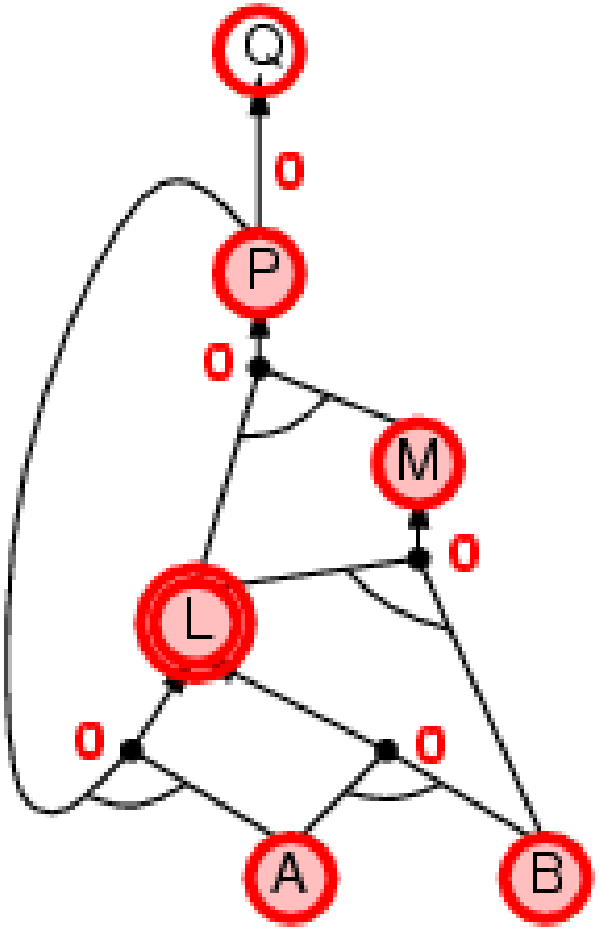
$P \wedge A \Rightarrow L$

$L \wedge M \Rightarrow P$

$L \wedge B \Rightarrow M$

$A \wedge B \Rightarrow L$

$A,\ B$



Dowodzenie w przód - przykład

... aż do udowodnienia zapytania *Q ...*

$P \Rightarrow Q$

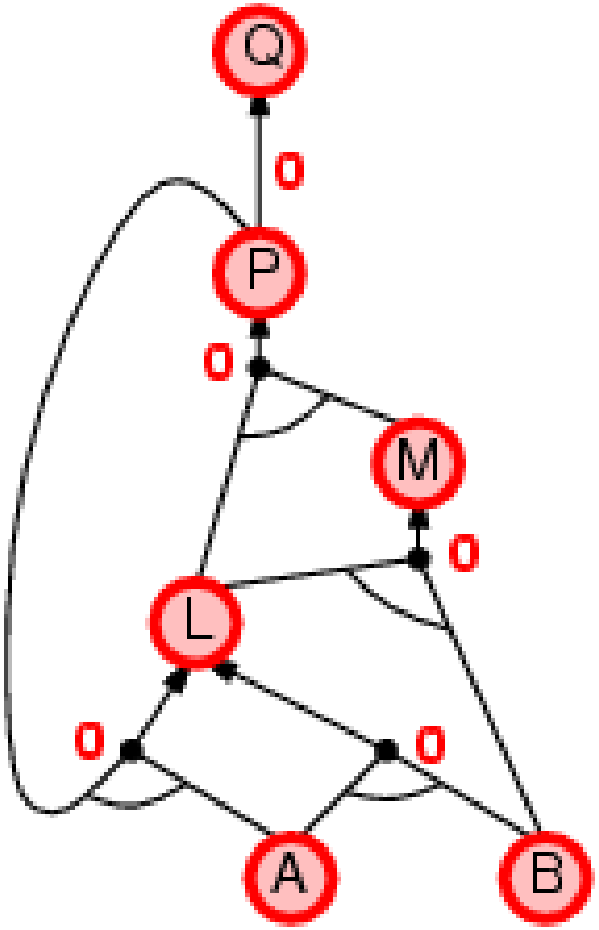
$P \wedge A \Rightarrow L$

$L \wedge M \Rightarrow P$

$L \wedge B \Rightarrow M$

$A \wedge B \Rightarrow L$

$A,\ B$



... lub niemożności dalszego dowodzenia.

Dowodzenie wstecz *(backward chaining)*

Idea:

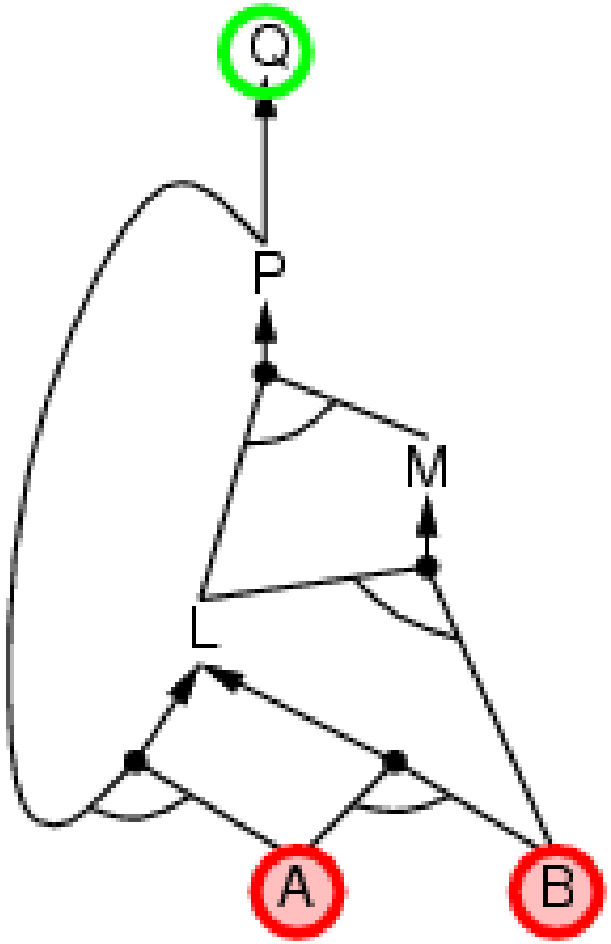
- sprawdzić, czy baza *KB* zawiera odpowiedź *Q*,
- jeżeli nie, to dowieść poprzez dowodzenie wstecz wszystkich przesłanek reguły prowadzącej do pytania *Q*,
- powtarzać procedurę aż do osiągnięcia zdań już dowiedzionych.

Należy przy tym unikać:

- zapętlania się algorytmu sprawdzając, czy kolejna przesłanka nie była udowodniona wcześniej;
- powtarzania pracy poprzez sprawdzanie:
  - czy kolejna przesłanka nie została już dowiedziona,
  - czy kolejna przesłanka nie jest fałszywa.

Dowodzenie wstecz - przykład

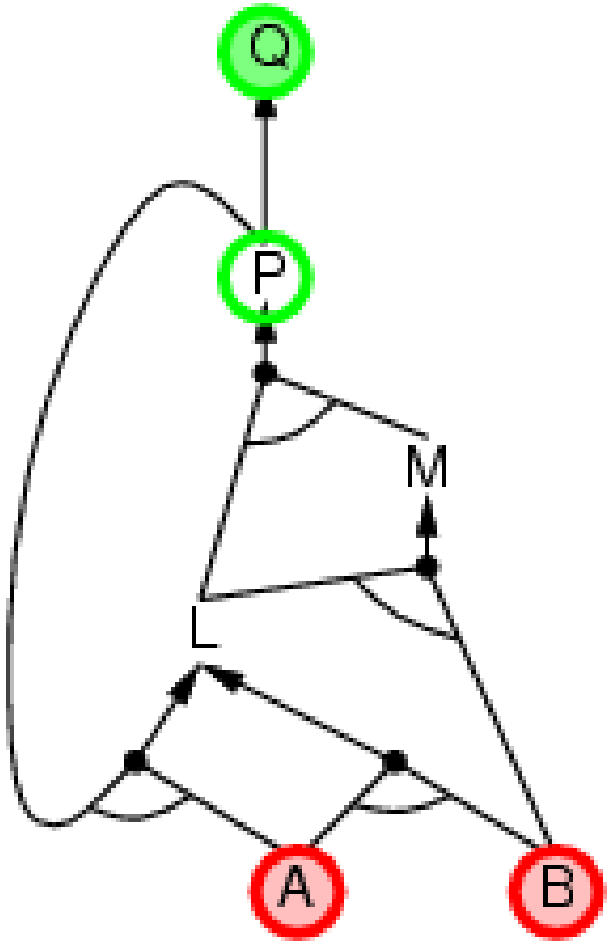
Zaczynamy od zapytania *Q* ...



Dowodzenie wstecz - przykład

... dowodzimy wstecznie kolejne przesłanki...

$P \Rightarrow Q$

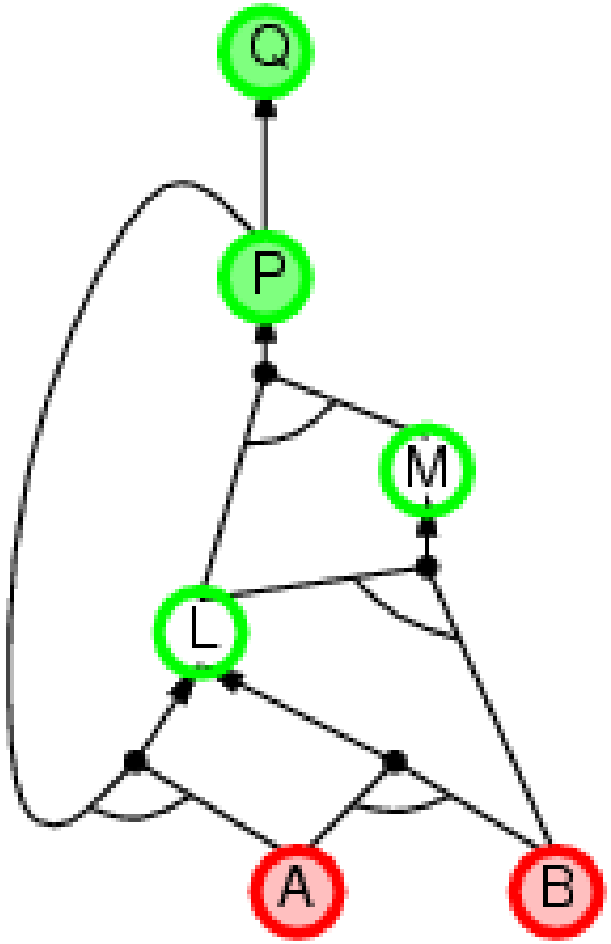


Dowodzenie wstecz - przykład

... dowodzimy wstecznie kolejne przesłanki...

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$



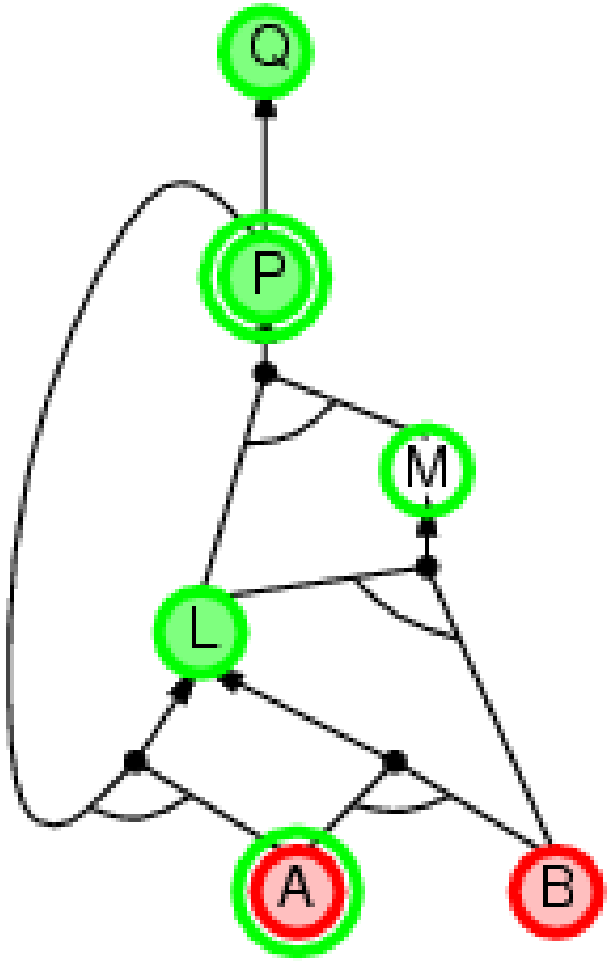
Dowodzenie wstecz - przykład

... dowodzimy wstecznie kolejne przesłanki...

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$A \wedge P \Rightarrow L$



Dowodzenie wstecz - przykład

... dowodzimy wstecznie kolejne przesłanki...

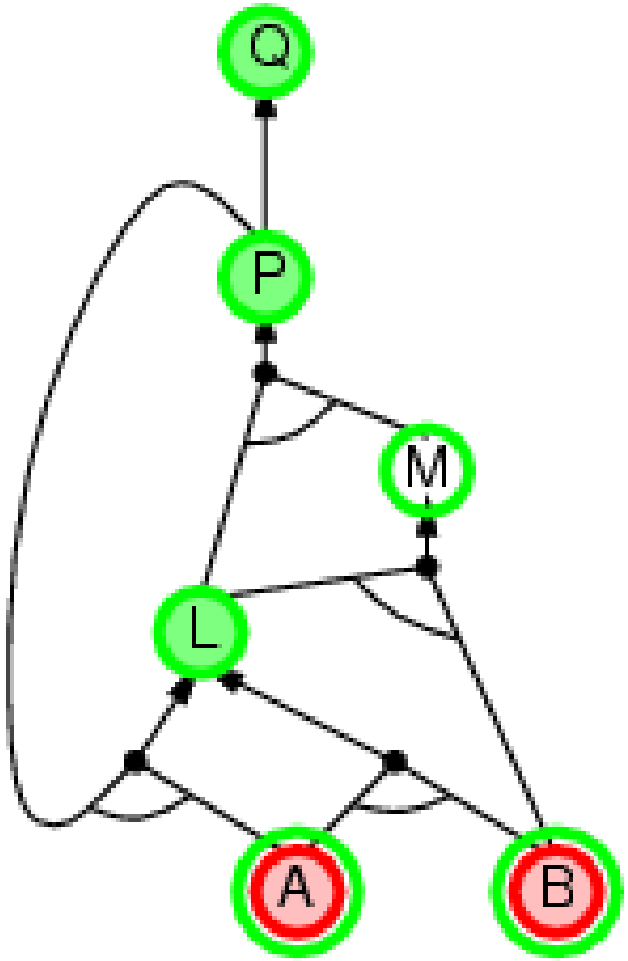
$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

$A,\ B$



Porównanie dowodzenia w przód i wstecz

- Dowodzenie w przód jest sterowane danymi.

  Nadaje się zwłaszcza do zadań „automatycznych” np. rozpoznawania obiektów, podejmowania decyzji.

  Dowodzenie w przód może wykonać wiele zbędnych działań, przez co jego koszt nie jest optymalny.

- Dowodzenie wstecz jest sterowane celem.

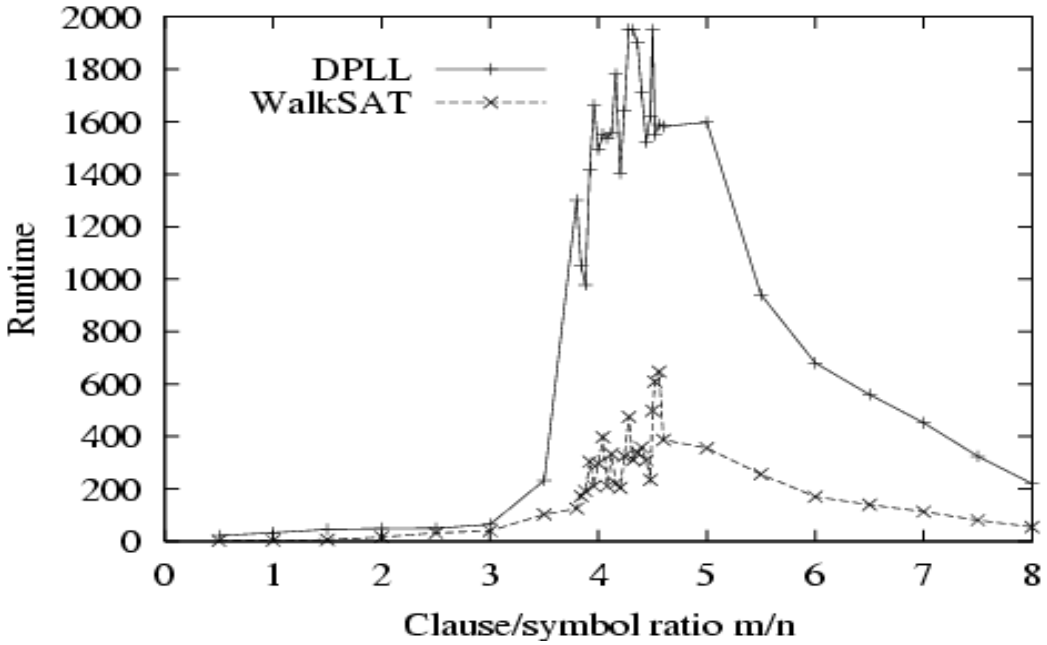
  Nadaje się zwłaszcza do rozwiązywania problemów, np. odpowiedzi na konkretne pytania, np. „Czy Wumpus jest w polu (3,2)?”.

  Koszt dowodzenia wstecz może być dużo niższy niż liniowy (względem rozmiaru bazy KB).

7.6. Efektywne algorytmy wnioskowania

Istnieją dwie grupy algorytmów wnioskowania w rachunku zdań:

- algorytmy poprawne i zupełne poszukiwania wstecz, np. DPLL (Davis, Putnam, Logemann, Loveland)
- algorytmy poszukiwania lokalnego (poprawne, ale niezupełne), np. WALKSAT.



7.7. Agenci wykorzystujący rachunek zdań

Agent logiczny w świecie Wumpusa oparty na rachunku zdań.

Baza wiedzy *KB*

$\neg P_{1,1}$	(nie ma dołu w polu 1,1)
$\neg W_{1,1}$	(nie ma Wumpusa w polu 1,1)
$B_{x,y} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$	(warunek wystąpienia wiatru)
$S_{x,y} \Leftrightarrow (W_{x,y+1} \vee W_{x,y-1} \vee W_{x+1,y} \vee W_{x-1,y})$	(warunek wystąpienia odoru)
$W_{11} \vee W_{12} \vee \ldots \vee W_{4,4}$	(istnieje co najmniej jeden Wumpus)
$\neg W_{11} \vee \neg W_{12}$ $(W_{11} \Rightarrow \neg W_{12})$	(co najwyżej jeden Wumpus w polach 1,1 i 1,2)
$\neg W_{11} \vee \neg W_{13}$	(co najwyżej jeden Wumpus w polach 1,1 i 1,3)
...	...
$\neg W_{34} \vee \neg W_{44}$ **120**	(co najwyżej jeden Wumpus w polach 3,4 i 4,4)

Baza wiedzy zawiera 64 różne predykaty i 155 zdań.
Fakt „*Jest tylko jeden Wumpus*” opisuje aż 120 zdań!

Pole (i,j) jest na pewno bezpieczne jeżeli baza *KB* pociąga zdanie $(\neg P_{i,j} \wedge \neg W_{i,j})$.

Pole (i,j) jest być może bezpieczne jeżeli baza *KB* nie pociąga zdania $(P_{i,j} \vee W_{i,j})$.

Mała ekspresywność rachunku zdań

Baza danych świata Wumpusa musi zawierać zdania opisujące „fizykę” każdego pola z osobna.

Z tego powodu baza wiedzy *KB* musi być duża.

Byłoby prościej pamiętać dwa zdania opisujące reguły pojawiania się wiatru lub odoru, słuszne dla ***wszystkich*** pól.

Nie jest to jednak możliwe w rachunku zdań.

Poza tym w bazie *KB* nie ma informacji o położeniu agenta:

- dla każdej chwili *t* i każdego pola *(x,y)*

$$L^{t}_{x,y} \wedge ZwróconyWPrawo^{t} \wedge DoPrzodu^{t} \Rightarrow L^{t}_{x+1,y}$$

Problemem w rachunku zdań jest gwałtowny wzrost liczby klauzul potrzebnych do opisu położenia agenta.

Rachunek zdań jest ***zbyt mało ekspresyjny*** aby zwięźle opisać nawet świat Wumpusa, nie mówiąc już o świecie realnym.

Do opisu świata rzeczywistego potrzebna jest zatem logika bardziej ekspresyjna - tzw. ***logika pierwszego rzędu***.

Podsumowanie

- Agenci logiczni stosują wnioskowanie do bazy wiedzy aby uzyskać nowe informacje i podjąć decyzje.
- Podstawą wnioskowania jest relacja pociągania między zdaniami logicznymi.
- Wnioskowanie umożliwia tworzenie nowych zdań na podstawie już istniejących.
- W rachunku zdań rozstrzygalność zdań jest zupełna.
- Algorytmy poszukiwania w przód i wstecz są bardzo efektywne dla baz wiedzy Horna - czas obliczeń zależy co najwyżej liniowo od wymiaru bazy.
- Rachunek zdań jest jednak mało ekspresywny.