Logical Agents

Chastine Fatichah
Departemen Teknik Informatika
Mei 2023

Capaian Pembelajaran Matakuliah

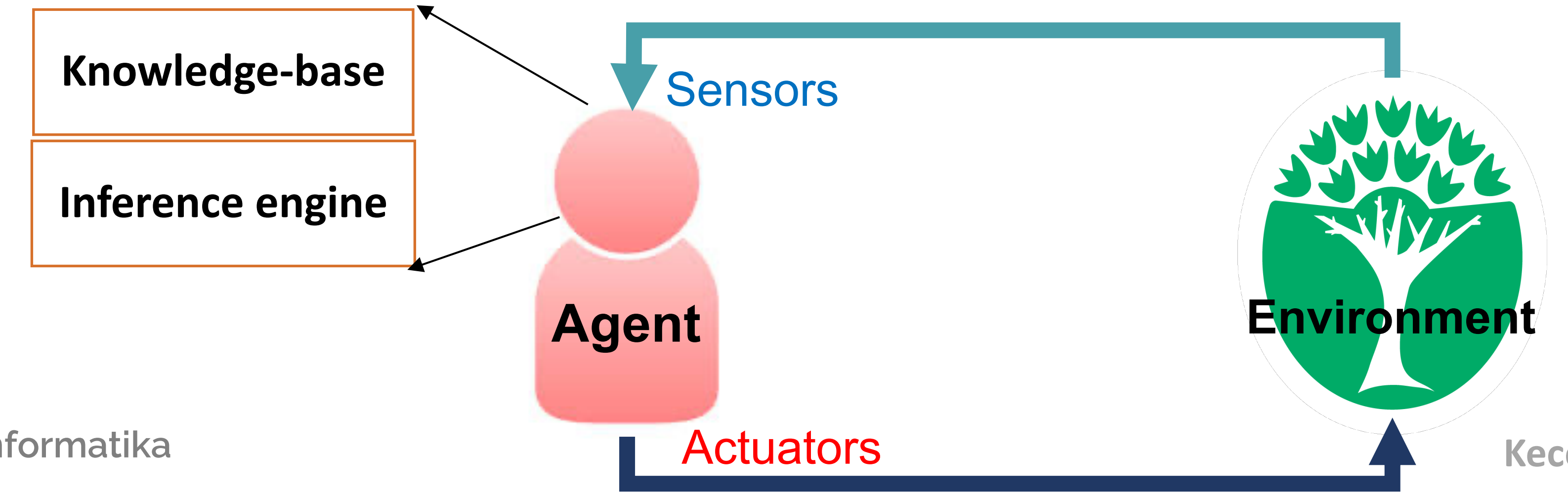
Mahasiswa mampu menjelaskan, merancang, dan menerapkan *knowledge-based intelligent agent* dengan merepresentasikan *knowledge base* menjadi *propositional logic* atau *first order logic* serta memanfaatkan algoritma *resolution, forward* dan *backward chaining* untuk melakukan proses inferensi

Pokok Bahasan

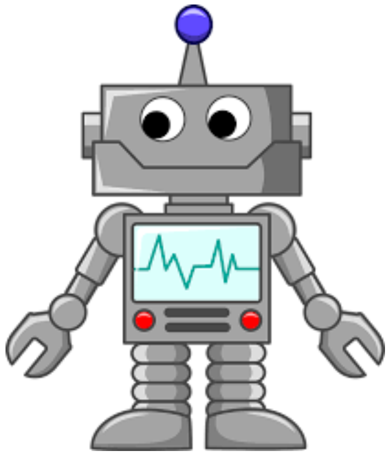
- *Knowledge-based agent*
- *Wumpus World*
- *Propositional logic*
- *Equivalence, validity, satisfiability*
- *Inference rules* dan metode pembuktian
  - *forward chaining*
  - *backward chaining*
  - *Resolution*

Knowledge-based Agent

- *Agent* dapat membentuk representasi, melakukan inferensi untuk menghasilkan representasi baru, dan menggunakan representasi baru untuk melakukan aksi
  - Pada lingkungan *partially observable*, *knowledge-based agent* dapat megkombinasikan pengetahuan umum dengan kondisi saat ini untuk melakukan inferensi kemudian memilih aksi yang akan dilakukan
  - Contoh: Dokter menggunakan pengetahuan dan pengalaman untuk mendiagnosis penyakit dan memberikan tindakan pengobatan yang tepat



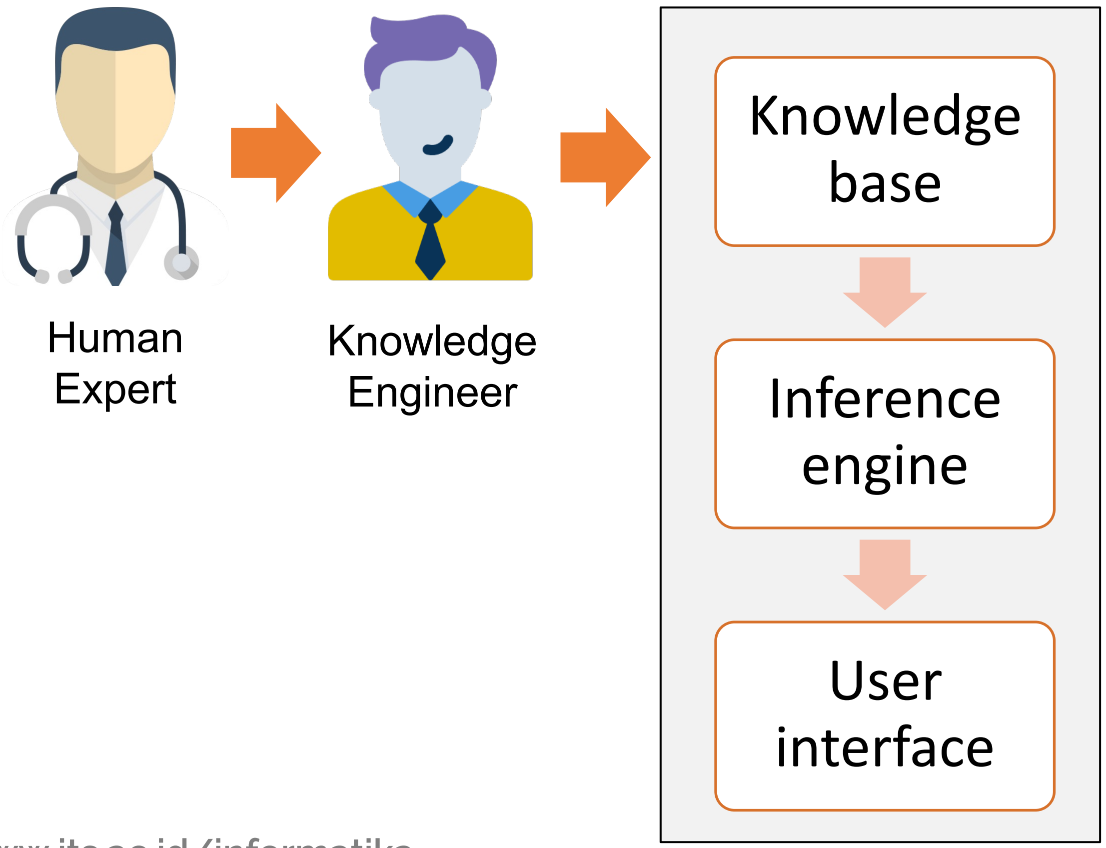
Knowledge-based Agent



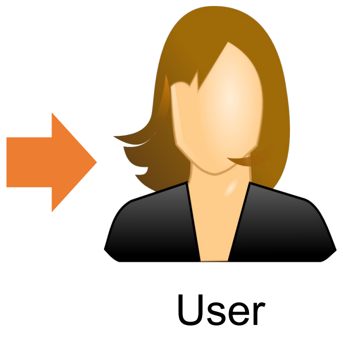
- Agent harus dapat:
  - Merepresentasikan *state*, *action*, dan sebagainya
  - Menerima informasi baru
  - Mengupdate representasi
  - Menyimpulkan pengetahuan lain yang tidak eksplisit (*hidden property*)
  - Menyimpulkan aksi apa yang perlu diambil

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

    TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
    action ← ASK(KB, MAKE-ACTION-QUERY(t))
    TELL(KB, MAKE-ACTION-SENTENCE(action, t))
    t ← t + 1
    return action
```

Knowledge-based Agent pada Sistem Pakar (*Expert System*)



- *Knowledge base*
  Berisi *domain-specific* dan *high-quality knowledge*
- Inference engine
  Prosedur dan *rules* untuk inferensi
- User interface
  Menyediakan interaksi ke user (*Natural language processing*)



Knowledge-based Models

- Data, Informasi, Pengetahuan
  - Data: Fakta yang belum diproses
  - Informasi: Data yang sudah diproses, memberikan makna
  - Pengetahuan: Sekumpulan informasi yang digunakan untuk inferensi

- Pengetahuan dapat direpresentasikan sebagai struktur simbol dan dapat diperoleh melalui *reasoning/inference*
  - *Red:* merepresentasikan warnah merah
  - *My_car:* merepresentasikan mobil saya
  - *Red(my_car):* merepresentasikan warna dari mobil saya adalah merah

Knowledge-based Models

- Inferensi (inference): merupakan proses derivasi pengetahuan baru dari pengetahuan yang sudah ada
- Kumpulan pengetahuan yang sudah ada disebut *knowledge-base* (**KB**):
  - Kumpulan kalimat
  - Kalimat merepresentasikan kebenaran pada *real world* dalam *knowledge representation language*
- Contoh:
  - Rino is an elephant
  - All elephants are in grey color
  - Elephants are mammal
  - **Inferensi**:
    - Rino is a mammal
  - **Pertanyaan**: Is Rino in red color?
    - Answer: No

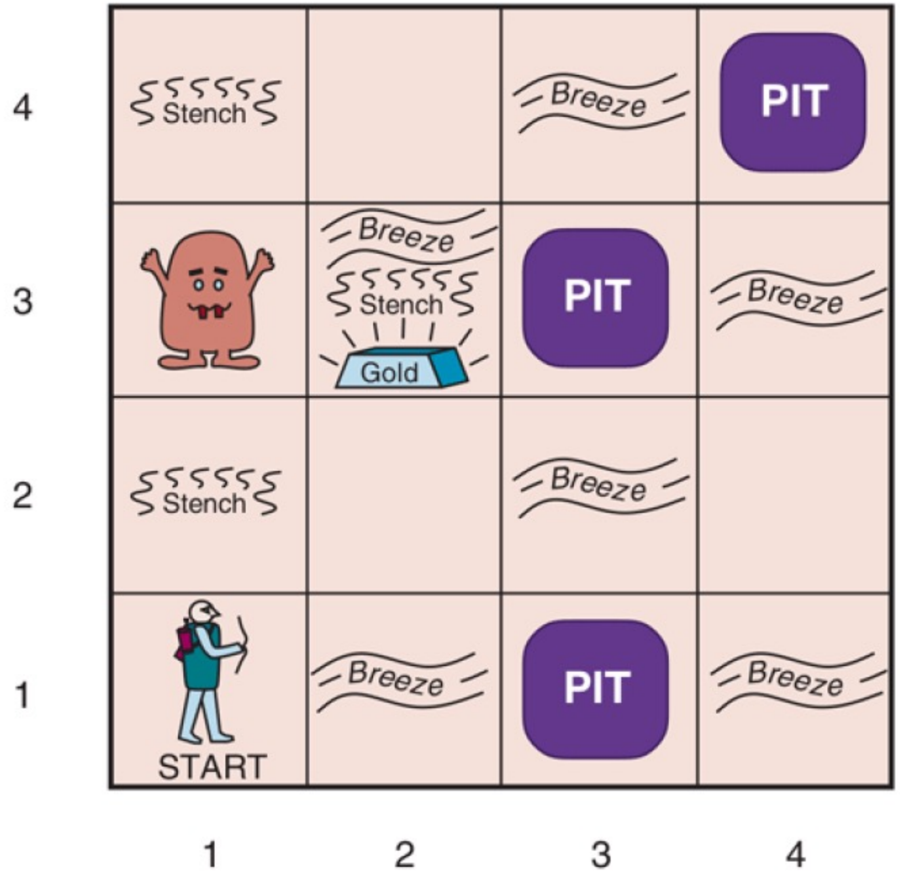
Knowledge Representation Language

- *Knowledge representation language* seharusnya bisa:
  - ***Merepresentasikan*** pengetahuan yang cukup
  - Menggunakan ***syntax*** yang jelas dan ***semantic*** yang benar
  - ***Reasoning*** untuk mengambil kesimpulan baru

- *Knowledge representation language* menggunakan:
  - *Propositional Logic*
  - *Predicate Logic / First Order Logic*

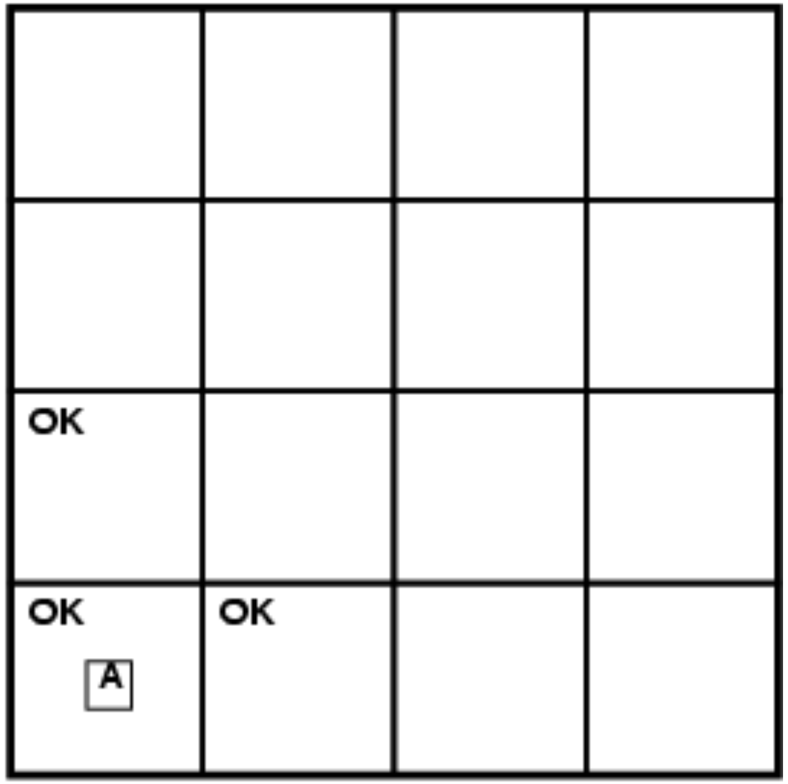
Contoh: *Wumpus World*

- Performance measure
  - emas +1000, mati -1000
  - gerak -1 , panah -10
- Environment
  - Squares adjacent to wumpus are smelly
  - Squares adjacent to pit are breezy
  - Glitter iff gold is in the same square
  - Shooting kills wumpus if you are facing it
  - Shooting uses up the only arrow
  - Grabbing picks up gold if in same square
  - Releasing drops the gold in same square
- Sensors: Stench, Breeze, Glitter, Bump, Scream
- Actuators: Left turn, Right turn, Forward, Grab, Release, Shoot



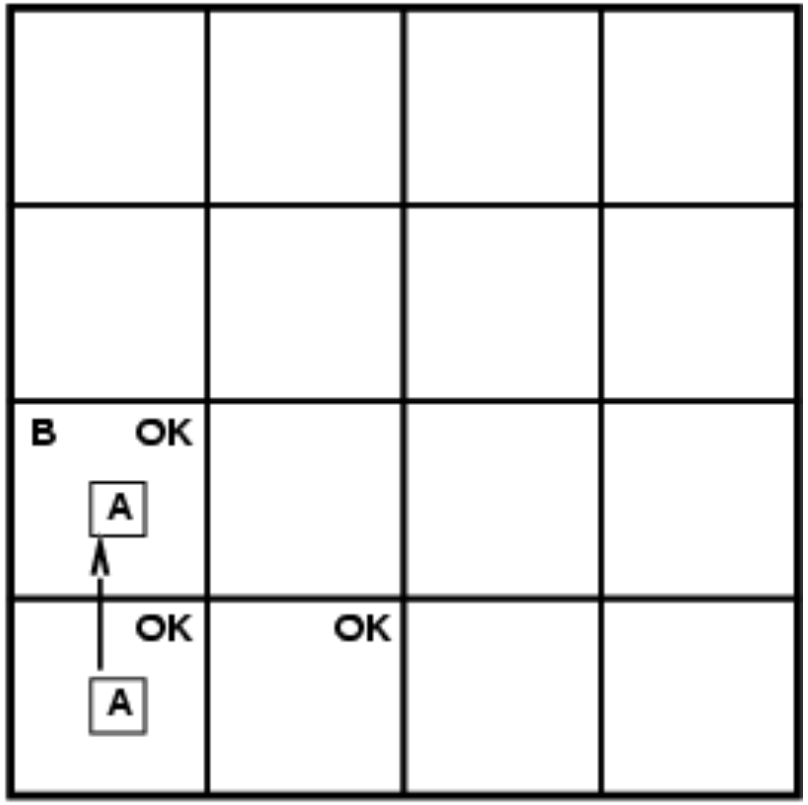
Sumber: S. Russel, P. Norving, Artificial Intteligencen: A Modern Approach

Contoh: *Wumpus World*



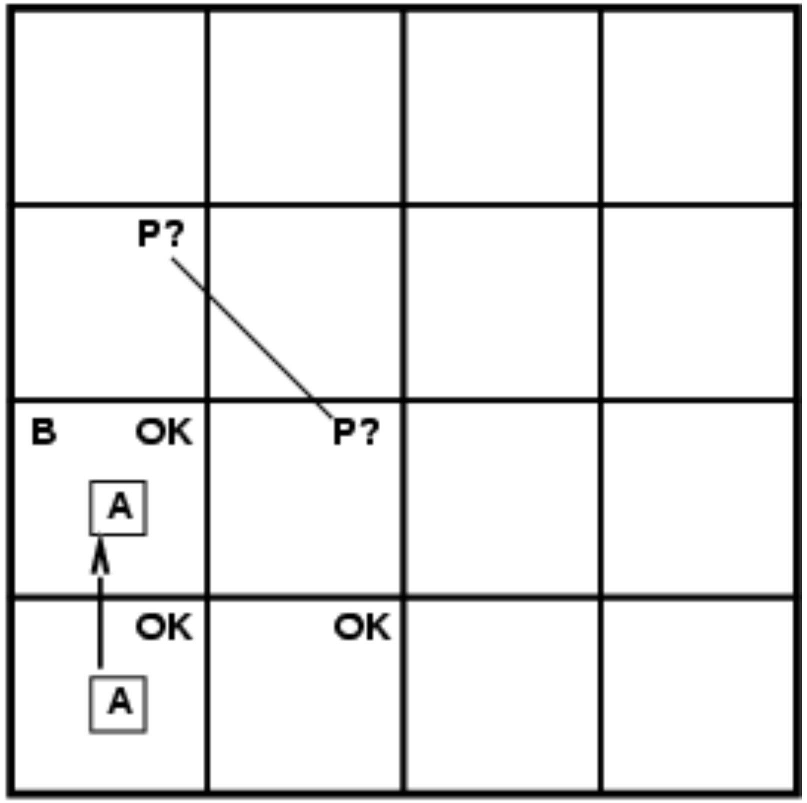
Sumber: S. Russel, P. Norving, Artificial Intteligencen: A Modern Approach

Contoh: *Wumpus World*



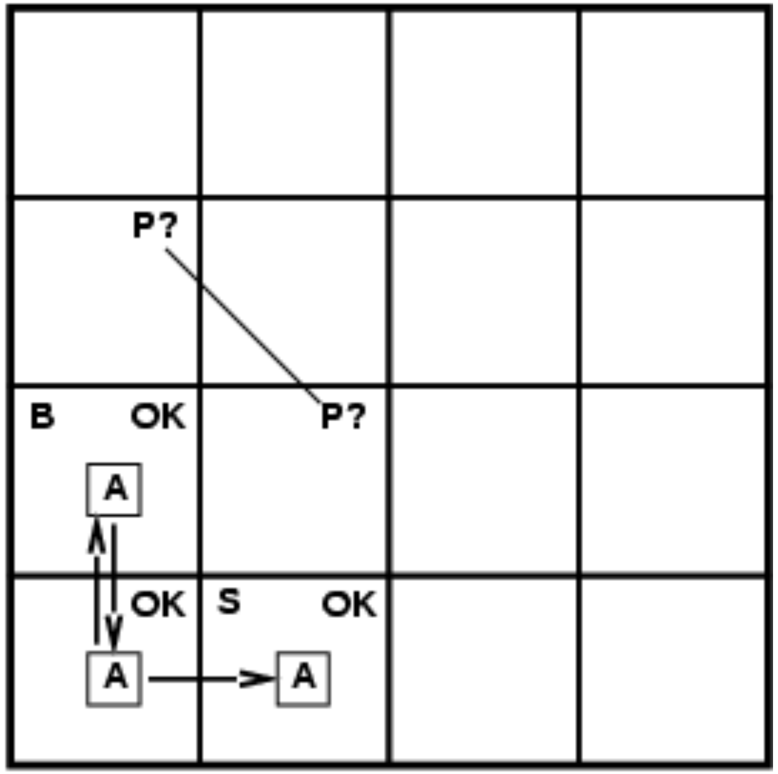
Sumber: S. Russel, P. Norving, Artificial Inttelligencen: A Modern Approach

Contoh: *Wumpus World*



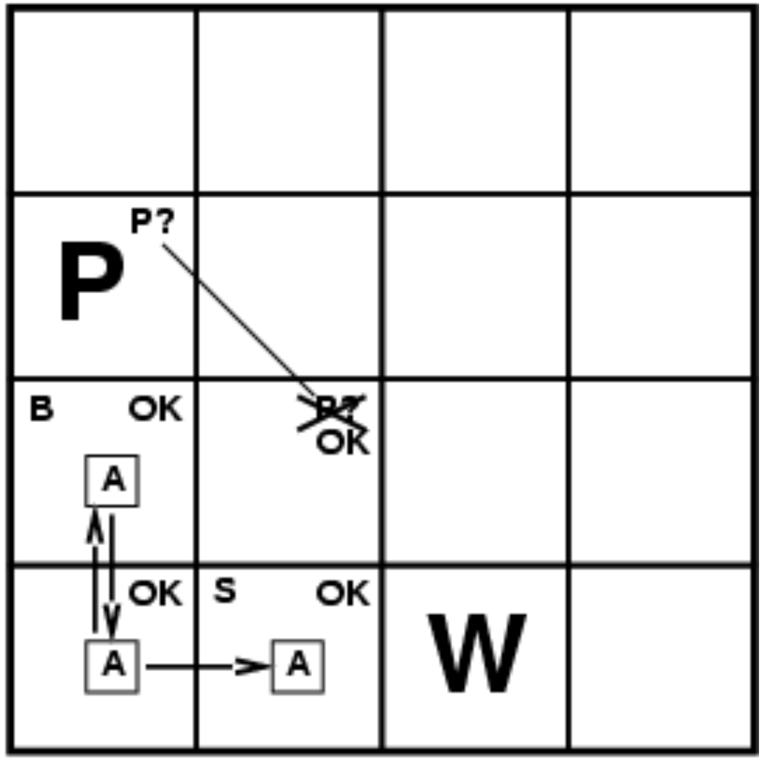
Sumber: S. Russel, P. Norving, Artificial Intelligencen: A Modern Approach

Contoh: *Wumpus World*



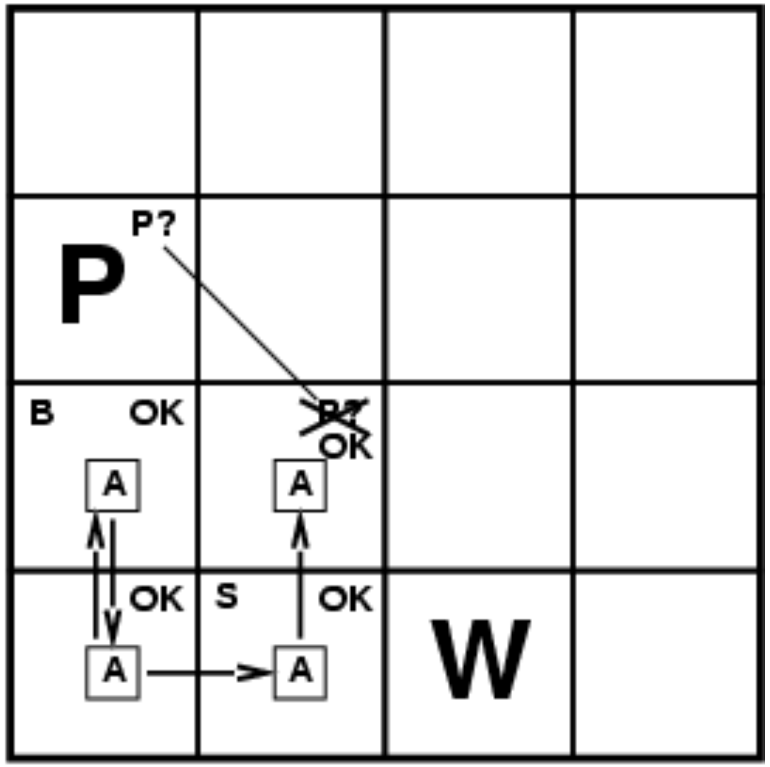
Sumber: S. Russel, P. Norving, Artificial Intteligencen: A Modern Approach

Contoh: *Wumpus World*



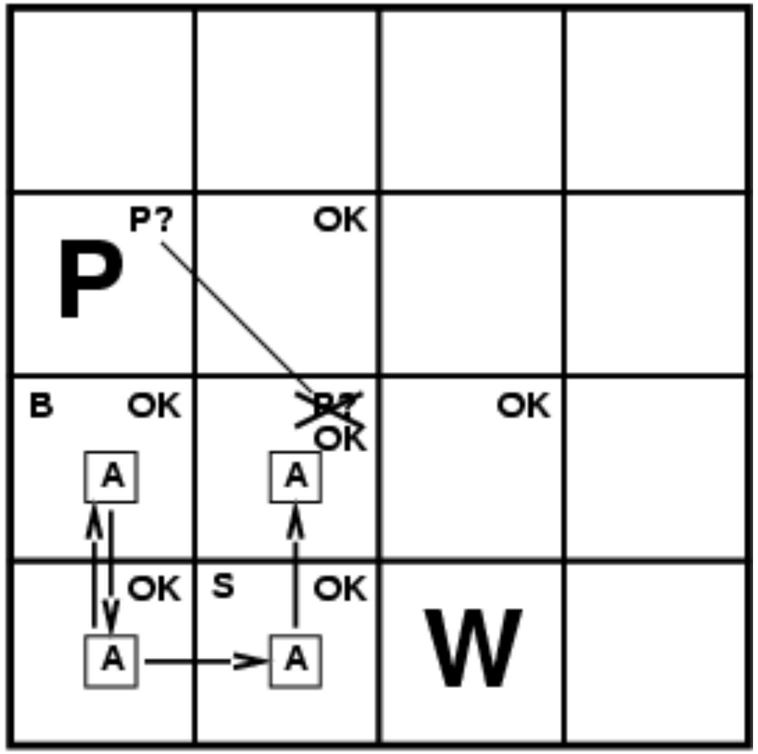
Sumber: S. Russel, P. Norving, Artificial Intelligencen: A Modern Approach

Contoh: *Wumpus World*



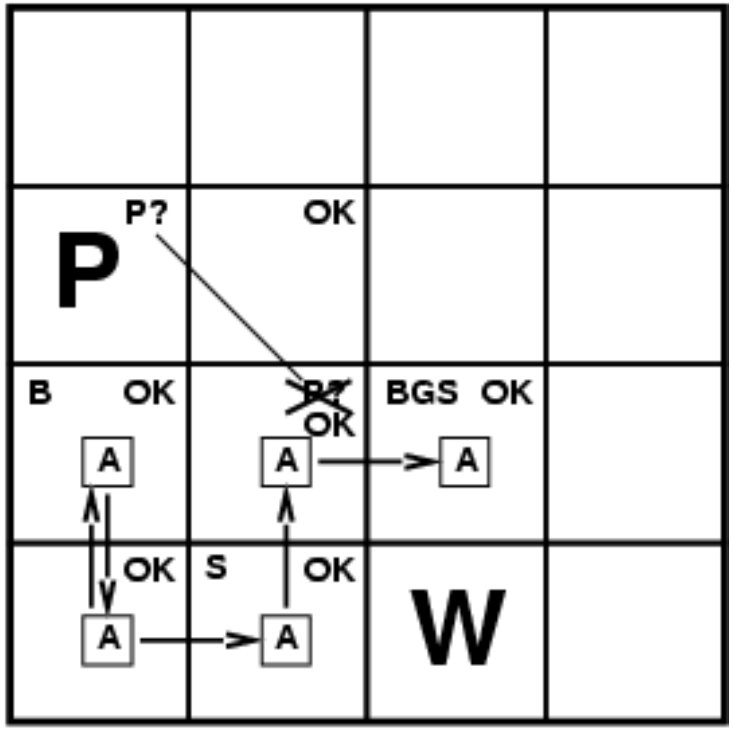
Sumber: S. Russel, P. Norving, Artificial Intteligencen: A Modern Approach

Contoh: *Wumpus World*



Sumber: S. Russel, P. Norving, Artificial Intteligencen: A Modern Approach

Contoh: *Wumpus World*



Sumber: S. Russel, P. Norving, Artificial Intteligencen: A Modern Approach

Logic

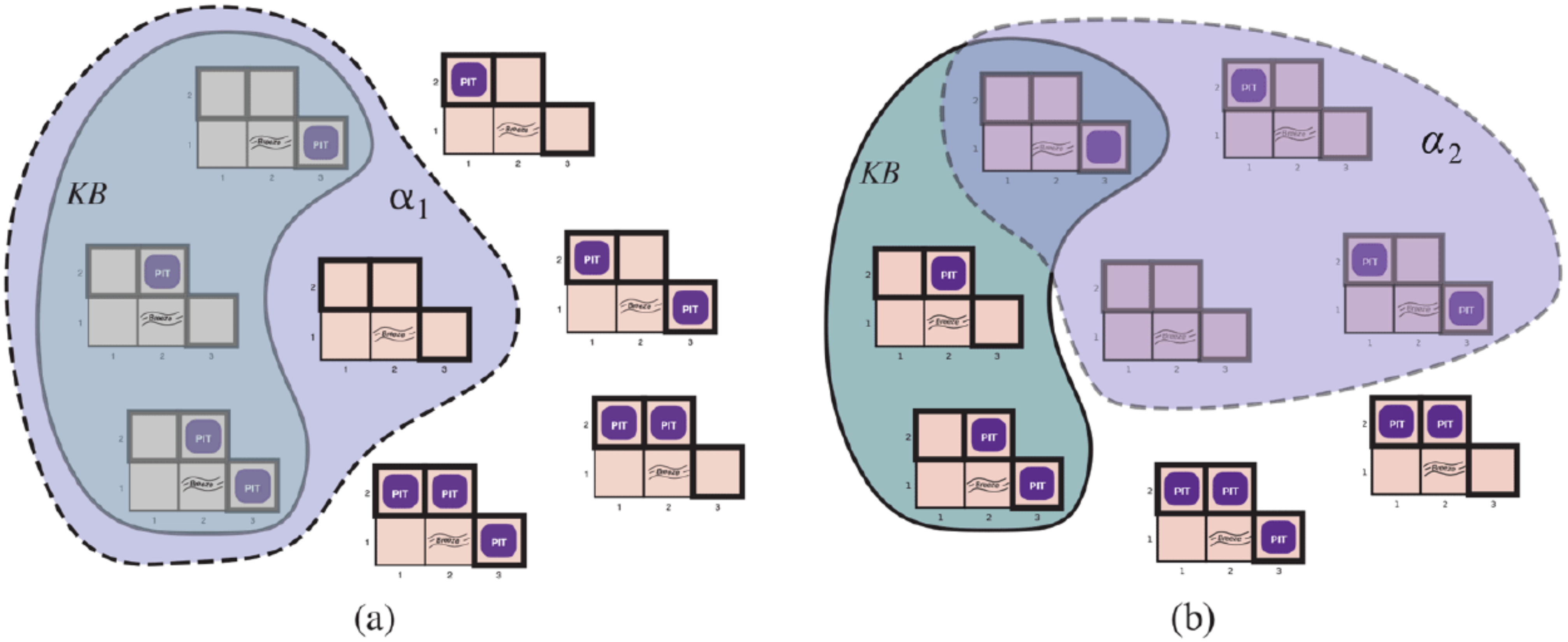
- *Logic* adalah bahasa formal untuk merepresentasikan informasi sedemikian hingga kesimpulan dapat dibuat
- *Syntax* → sturktur kalimat pada bahasa
- *Semantics* → arti kalimat, misalnya mendefinisikan kebenaran sebuah kalimat
- Contoh kalimat aritmatika:
  - *$x+2 \geq y$ is a sentence; $x2+y >$ is not a sentence*
  - *$x+2 \geq y$ is true iff the number $x+2$ is no less than the number y*
  - *$x+2 \geq y$ is true in a world where $x = 7$, $y = 1$*
  - *$x+2 \geq y$ is false in a world where $x = 0$, $y = 6$*

Entailment

- ***Entailment***: $\alpha \models \beta$ ("α entails β" or "β follows from α") iff in every world where α is true, β is also true
  - Misalnya, α-worlds adalah subset dari β-worlds [***models***(α) $\subseteq$ ***models***(β)]

- Contoh, $\alpha_2 \models \alpha_1$
  (α_2 is $\neg Q \wedge R \wedge S \wedge W$
  α_1 is $\neg Q$)

Entailment



Possible models for the presence of pits in squares [1,2], [2,2], and [3,1]. The KB corresponding to the observations of nothing in [1,1] and a breeze in [2,1] is shown by the solid line. (a) Dotted line shows models of α_1 (no pit in [1,2]). (b) Dotted line shows models of α_2 (no pit in [2,2]).

Sumber: S. Russel, P. Norving, Artificial Inttelligencen: A Modern Approach

Entailment

$$KB \models$$

- *Knowledge base KB* entails kalimat α jika dan hanya jika α adalah true pada semua kasus dimana *KB* bernilai true, misalnya
  - KB *"the Giants won"* dan *"the Reds won"* entails *"Either the Giants won or the Reds won"*
  - x+y = 4 entails 4 = x+y
- *Entailment* adalah sebuah hubungan antar kalimat (*syntax*) yang didasarkan pada *semantics*
- *Inferensi:*
  - $KB \vdash_i \alpha$ = kalimat α dapat diderivasi dari *KB* dengan prosedur *i*
  - *Soundness*: jika $KB \vdash_i \alpha$ benar, bernilai juga benar pada $KB \models \alpha$
  - *Completeness*: jika $KB \models \alpha$ benar, bernilai juga benar pada $KB \vdash_i \alpha$

Proofs

Metode pembuktian entailment antara α dan β

- Method 1: ***model-checking***
  - For every possible world, if α is true make sure that is β true too
  - OK for propositional logic (finitely many worlds); not easy for first-order logic
- Method 2: ***theorem-proving***
  - Search for a sequence of proof steps (applications of ***inference rules***) leading from α to β
  - E.g., from $P \wedge (P \Rightarrow Q)$, infer Q by ***Modus Ponens***

Propositional Logic: Syntax

- *Propositional logic* adalah logika paling sederhana – menggambarkan ide dasar
- Simbol proposisi S_1, S_2 adalah sebuah kalimat
  - If S is a sentence, $\neg S$ is a sentence (*negation*)
  - If S_1 and S_2 are sentences, $S_1 \wedge S_2$ is a sentence (*conjunction*)
  - If S_1 and S_2 are sentences, $S_1 \vee S_2$ is a sentence (*disjunction*)
  - If S_1 and S_2 are sentences, $S_1 \Rightarrow S_2$ is a sentence (*implication*)
  - If S_1 and S_2 are sentences, $S_1 \Leftrightarrow S_2$ is a sentence (*biconditional*)

Formal Grammar For Propositional Logic

$$
\begin{array}{rcl}
Sentence & \rightarrow & AtomicSentence \mid ComplexSentence \\
AtomicSentence & \rightarrow & True \mid False \mid P \mid Q \mid R \mid \ldots \\
ComplexSentence & \rightarrow & (\, Sentence \,) \\
& \mid & \neg\, Sentence \\
& \mid & Sentence \wedge Sentence \\
& \mid & Sentence \vee Sentence \\
& \mid & Sentence \;\Rightarrow\; Sentence \\
& \mid & Sentence \;\Leftrightarrow\; Sentence
\end{array}
$$

Propositional Logic: Semantics

- Tiap model mempunyai nilai true/false untuk setiap proposisi

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

- Rule untuk mengevaluasi nilai kebenaran dengan model *m*:

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

- Proses rekursif sederhana dalam mengevaluasi sebuah kalimat, misalnya

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

Logical Equivalent

- Dua kalimat adalah *logically equivalent* jika bernilai true pada model yang sama: $\alpha \equiv \beta$ iff $\alpha \models \beta$ and $\beta \models \alpha$

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

Validity dan *Satisfiability*

- Sebuah kalimat adalah valid jika bernilai true pada semua model,
  - Misal: *True*, $A \vee \neg A$, $A \Rightarrow A$, $(A \wedge (A \Rightarrow B)) \Rightarrow B$
- Validity dihubungkan ke inference melalui Deduction Theorem:
  - $KB \models \alpha$ if and only if $(KB \Rightarrow \alpha)$ is valid
- Sebuah kalimat adalah *satisfiable* jika bernilai true pada beberapa model
  - Misal: $A \vee B$, C
- Sebuah kalimat adalah unsatisfiable jika bernilai salah pada semua model
  - Misal: $A \wedge \neg A$
- Satisfiability dihubungkan ke inference melalui:
  - $KB \models \alpha$ if and only if $(KB \wedge \neg \alpha)$ is unsatisfiable

Conjunctive Normal Form (CNF)

- Setiap kalimat dapat diekspresikan sebagai sebuah ***conjunction*** *of* ***clauses***
- Setiap *clause* adalah sebuah ***disjunction*** *of* ***literals***
- Setiap *literal* merupakan sebuah simbol atau negasi simbol

$$
\begin{array}{rcl}
\mathit{CNFSentence} & \rightarrow & \mathit{Clause}_1 \land \cdots \land \mathit{Clause}_n \\
\mathit{Clause} & \rightarrow & \mathit{Literal}_1 \lor \cdots \lor \mathit{Literal}_m \\
\mathit{Fact} & \rightarrow & \mathit{Symbol} \\
\mathit{Literal} & \rightarrow & \mathit{Symbol} \mid \neg \mathit{Symbol} \\
\mathit{Symbol} & \rightarrow & P \mid Q \mid R \mid \ldots \\
\mathit{HornClauseForm} & \rightarrow & \mathit{DefiniteClauseForm} \mid \mathit{GoalClauseForm} \\
\mathit{DefiniteClauseForm} & \rightarrow & \mathit{Fact} \mid (\mathit{Symbol}_1 \land \cdots \land \mathit{Symbol}_l) \Rightarrow \mathit{Symbol} \\
\mathit{GoalClauseForm} & \rightarrow & (\mathit{Symbol}_1 \land \cdots \land \mathit{Symbol}_l) \Rightarrow \mathit{False}
\end{array}
$$

Konversi ke bentuk CNF

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

1. Eliminate $\Leftrightarrow$, replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$.
   $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$

2. Eliminate $\Rightarrow$, replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \vee \beta$.
   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$

3. Move $\neg$ inwards using de Morgan's rules and double-negation:
   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1})$

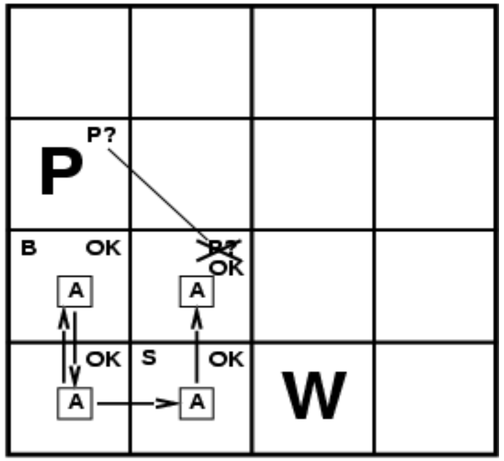
4. Apply distributive law ($\wedge$ over $\vee$) and flatten:
   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

Proof By Resolution

- *Resolution inference rule* (untuk CNF):

$$\frac{\ell_1 \vee \ldots \vee \ell_k, \qquad m_1 \vee \ldots \vee m_n}{\ell_1 \vee \ldots \vee \ell_{i-1} \vee \ell_{i+1} \vee \ldots \vee \ell_k \vee m_1 \vee \ldots \vee m_{j-1} \vee m_{j+1} \vee \ldots \vee m_n}$$

- Dimana ℓ_i dan m_j adalah *complementary literals*
- Misal: $\dfrac{P_{1,3} \vee P_{2,2}, \quad \neg P_{2,2}}{P_{1,3}}$



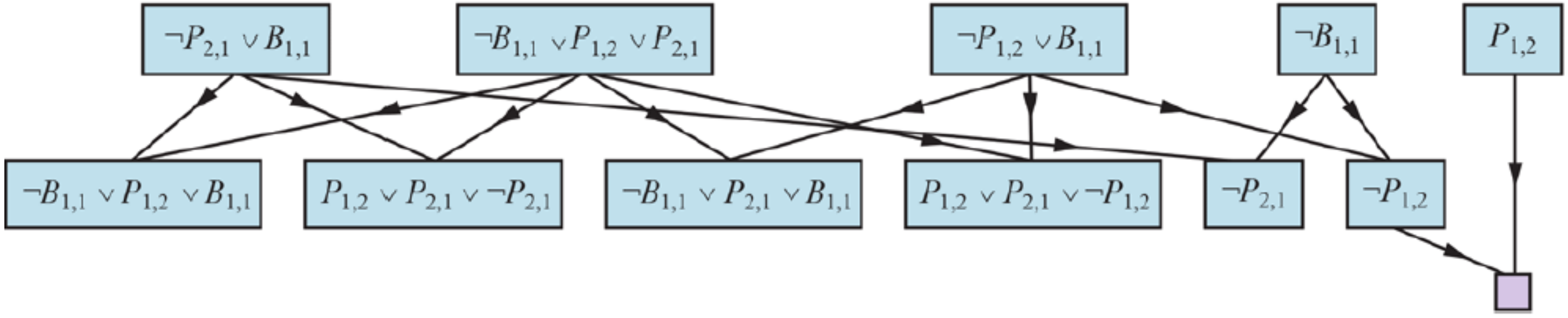
Algoritma *Resolution*

```
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 ← the set of clauses in the CNF representation of KB ∧ ¬α
  new ← { }
  while true do
     for each pair of clauses C_i, C_j in clauses do
        resolvents ← PL-RESOLVE(C_i, C_j)
        if resolvents contains the empty clause then return true
        new ← new ∪ resolvents
     if new ⊆ clauses then return false
     clauses ← clauses ∪ new
```

Contoh: *Resolution*

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

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



Horn Clauses

- *Horn Form*:

KB = conjunction of Horn clauses

- *Horn clause = proposition symbol* atau

(conjunction of symbols) $\Rightarrow$ *symbol*

Contoh: $C \wedge (B \Rightarrow A) \wedge (C \wedge D \Rightarrow B)$

- Modus Ponens (untuk *Horn Form*):

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

- Inferensi dengan *Horn clause* bisa menggunakan algoritma *forward chaining* atau *backward chaining*

Forward Chaining

- *Forward chaining* mengaplikasikan Modus Ponens untuk generate fakta baru:
  - *Given* $X_1 \wedge X_2 \wedge \ldots X_n \Rightarrow Y$ *and* $X_1, X_2, \ldots, X_n$
  - *Infer* Y
- *Forward chaining* terus mengaplikasikan aturan ini, menambah fakta baru, sampai tidak ada lagi yang ditambah
- Perlu *KB* untuk yang hanya berisi *definite clauses (horn clauses)*:
  - (Conjunction of symbols) $\Rightarrow$ symbol; atau
  - *Single symbol* (X is equivalent to True $\Rightarrow X$)

Forward Chaining

- Ide: sembarang rule yang memiliki premise yang *satisfy* dalam *KB,*
  - Tambahkan kesimpulan ke *KB,* sampai query ditemukan

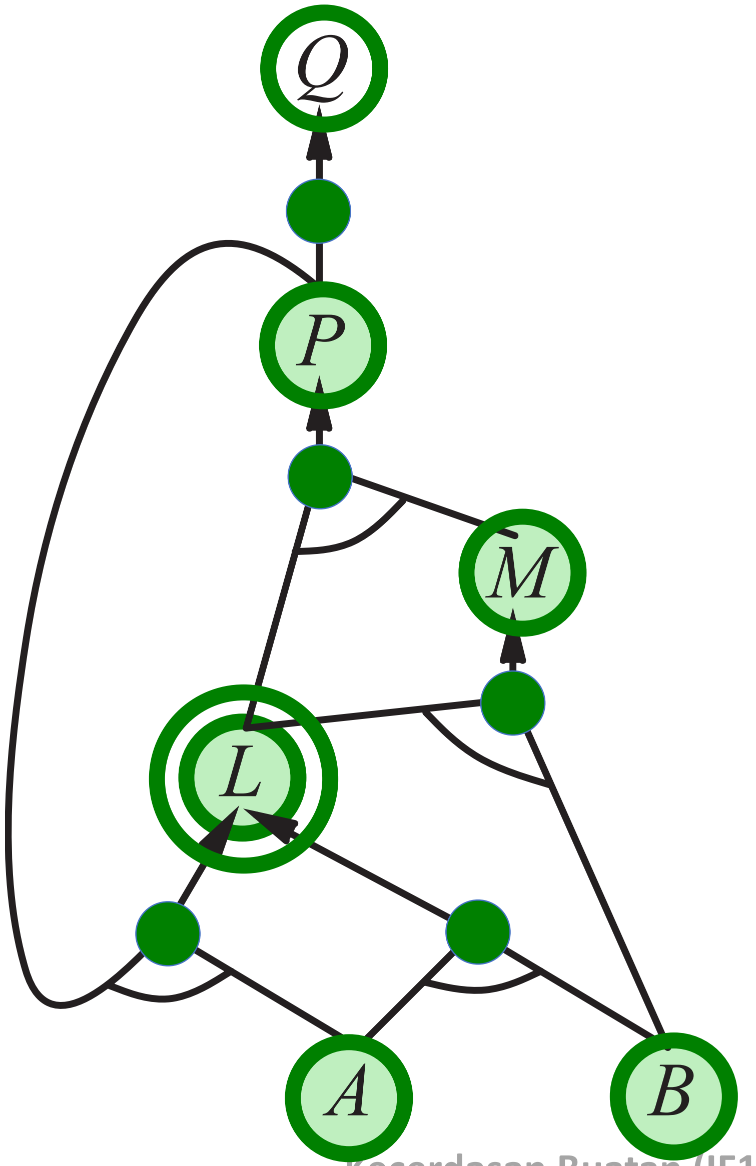
```
function PL-FC-ENTAILS?(KB, q) returns true or false
  local variables: count, a table, indexed by clause, initially the number of premises
                   inferred, a table, indexed by symbol, each entry initially false
                   agenda, a list of symbols, initially the symbols known to be true

  while agenda is not empty do
      p ← POP(agenda)
      unless inferred[p] do
          inferred[p] ← true
          for each Horn clause c in whose premise p appears do
              decrement count[c]
              if count[c] = 0 then do
                  if HEAD[c] = q then return true
                  PUSH(HEAD[c], agenda)
  return false
```

Contoh *Forward Chaining: Proving Q*

CLAUSES	COUNT	INFERRED
$P \Rightarrow Q$	~~1~~ 0	A ~~false~~ true
$L \wedge M \Rightarrow P$	~~2~~ ~~1~~ 0	B ~~false~~ true
$B \wedge L \Rightarrow M$	~~2~~ ~~1~~ 0	L ~~false~~ true
$A \wedge P \Rightarrow L$	~~2~~ ~~1~~ 0	M ~~false~~ true
$A \wedge B \Rightarrow L$	~~2~~ ~~1~~ 0	P ~~false~~ true
A	0	Q ~~false~~ true
B	0	

AGENDA

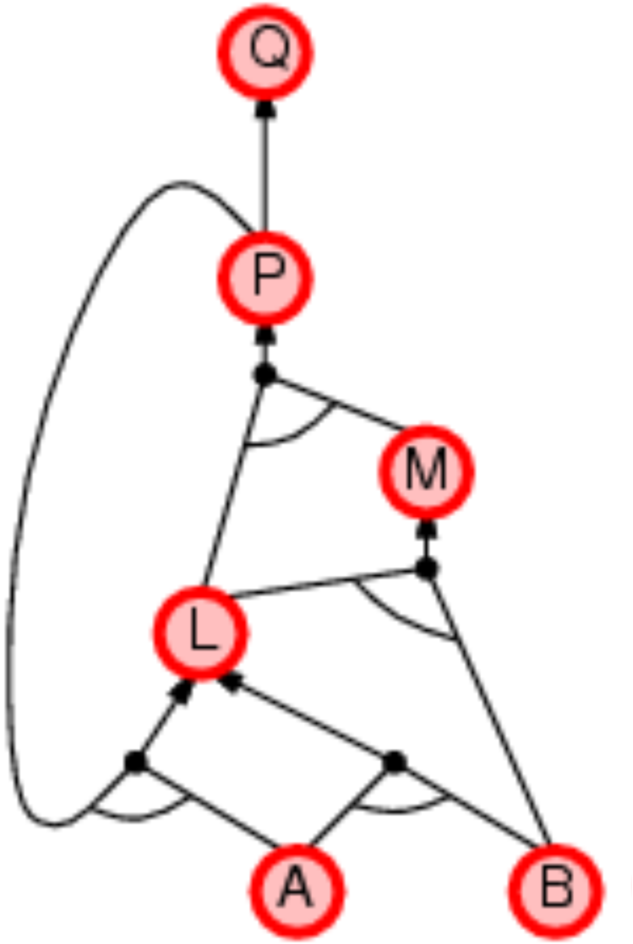
~~A~~ ~~B~~ ~~L~~ ~~M~~ ~~P~~ ~~L~~ ~~Q~~



Backward Chaining

- Ide: bekerja *backward* dari query *q*:
  - membuktikan *q* dengan *BC*,
  - cek jika *q* sudah diketahui, atau
  - buktikan dengan *BC* semua premise pada beberapa rule yang menyimpulkan *q*
- *Avoid loops*: chek jika *new subgoal* sudah ada
- *Avoid repeated work*: check if *new subgoal*
  - telah terbukti benar, atau
  - telah gagal

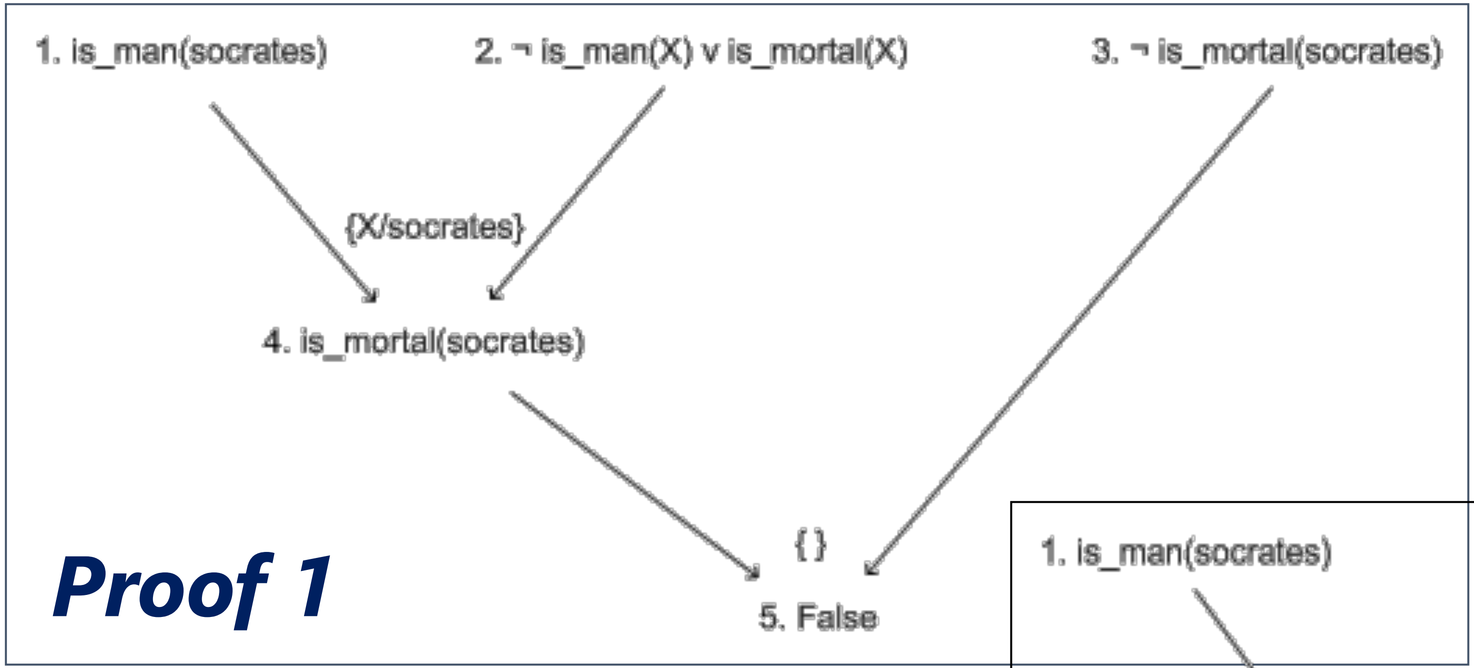
Contoh *Backward Chaining*

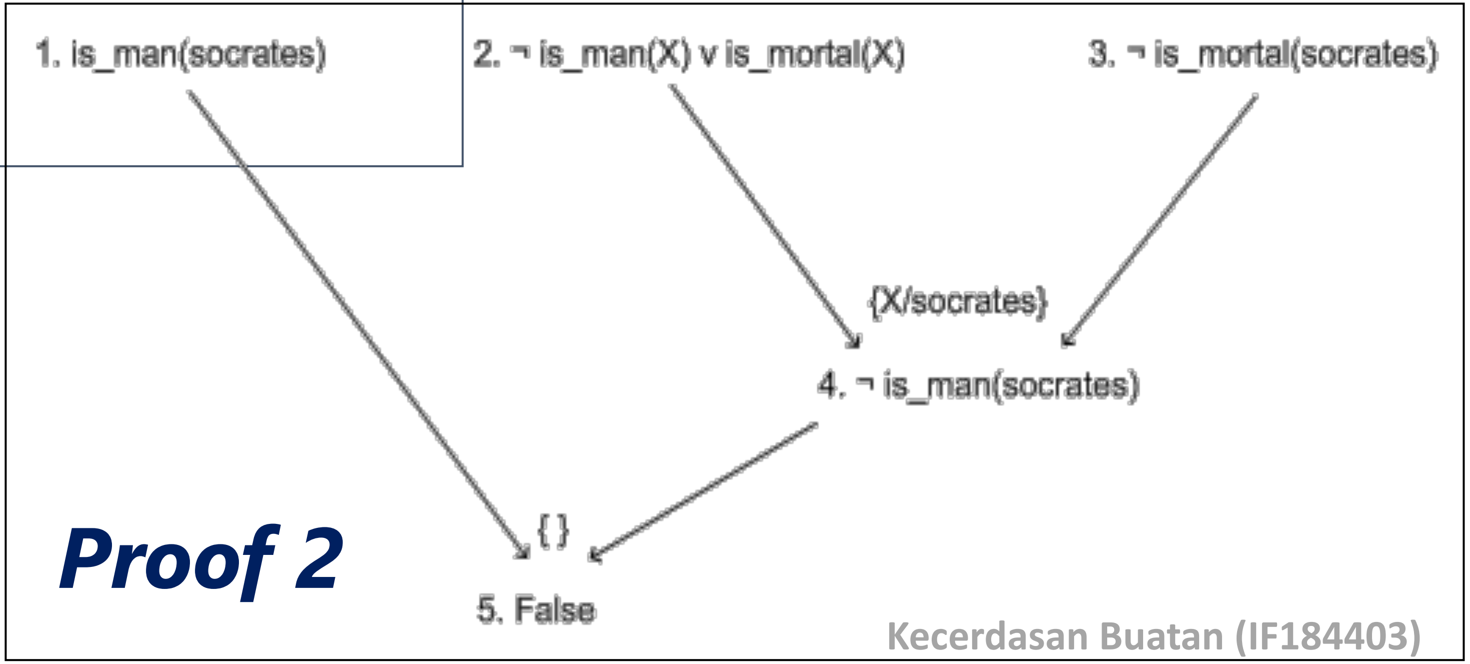


Contoh: *Aristotle*

- "Socrates is a man and all men are mortal therefore Socrates is mortal"
- *First Order Logic* (FOL)
  - *is_man(socrates)*
  - $\forall x \; is\_man(x) \Rightarrow is\_mortal(x)$ (Dirubah ke CNF)
- Initial state for Resolution
  - 1) is_man(socrates)
  - 2) ¬is_man(X) ∨ is_mortal(X)
  - 3) ¬is_mortal(socrates) (negation of theorem)

Resolution Proof Tree





- TERIMA KASIH -