# CSCM603130: Sistem Cerdas
# Logical Agents & Propositional Logic Inference

Fariz Darari, Aruni Yasmin Azizah

Fakultas Ilmu Komputer
Universitas Indonesia

2019/2020 • Semester Ganjil

# Outline

1. Knowledge-based agent
   - Contoh: Wumpus World
   - Knowledge representation language
   - Logika: representasi dan penalaran
2. KRR dalam Propositional Logic
   - Propositional logic sebagai KRL
   - Inference Procedure
   - Satisfiability testing: DPLL
   - Satisfiability testing: Local Search

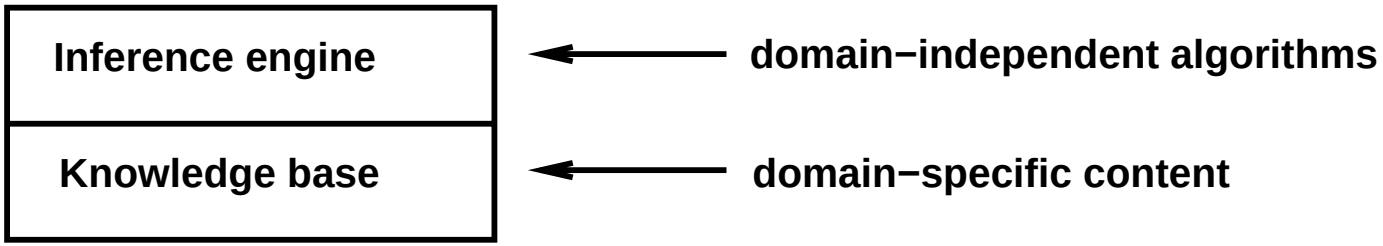
# Outline



# Pentingnya pengetahuan, representasi, dan penalaran

- Problem solving agent: memilih solusi di antara kemungkinan yang ada.
  - Apa yang ia "ketahui" tentang dunia sifatnya implisit (mis: dalam RESULT function) dan tidak berkembang.
- Knowledge-based agent: lebih "pintar".
  - Ia "mengetahui" hal-hal tentang dunia (direpresentasikan secara eksplisit) ...
  - ... dan dapat melakukan penalaran (reasoning) mengenai:
    - Hal-hal yang tidak diketahui sebelumnya (imperfect/partial information)
    - Tindakan yang paling baik untuk diambil

# Komponen Knowledge-based Agent




- Knowledge Base (KB): apa yang "diketahui" oleh si agent
  - Himpunan representasi fakta yang diketahui tentang lingkungannya
  - Tiap fakta disebut sentence.
  - Dinyatakan dalam bahasa formal $\rightarrow$ bisa diolah
- Inference Engine:
  - Menentukan fakta baru yang dapat diturunkan dari pengetahuan yang sudah ada dalam KB.
  - Terlibat dalam dua operasi dasar: TELL dan ASK

# Knowledge-based agent program

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

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

**Figure 7.1** A generic knowledge-based agent. Given a percept, the agent adds the percept to its knowledge base, asks the knowledge base for the best action, and tells the knowledge base that it has in fact taken that action.

- Apa yang dilakukan oleh agent program?
  - beritahu KB informasi baru berdasarkan persepsi $\rightarrow$ (TELL).
  - bertanya tindakan yang sebaiknya dipilih untuk dilakukan berdasarkan KB $\rightarrow$ (ASK).
  - beritahu KB tindakan yang dipilih $\rightarrow$ (TELL).

# Analisis vs. Implementasi

- Agent dapat dipandang dari knowledge level: yang diperlukan adalah spesifikasi apa yang diketahui oleh agent dan goalnya. Mis: agent mengetahui bahwa gedung B menghubungkan gedung A dan gedung C, dan si agent memiliki goal untuk pergi ke gedung A dari gedung C.
- Analisis pada level ini terlepas dari implementation level: bagaimana informasi yang diketahuinya diimplementasikan?
  - Tabel posisi koordinat gedung-gedung
  - Gambar diagram peta Fasilkom (bitmap, dll.)
  - ...

# Pendekatan deklaratif vs. prosedural

- Pendekatan deklaratif: programmer memberitahu (TELL) agent informasi tentang environment.
- Kalau informasi kurang, agent bisa melengkapinya sendiri.
- Bandingkan dengan pendekatan prosedural: programmer secara eksplisit memrogram agent untuk bertindak.
- Ini adalah salah satu isu dalam knowledge representation: bagaimana representasi yang tepat?
  - Expressiveness vs. Tractability

**Knowledge is power**

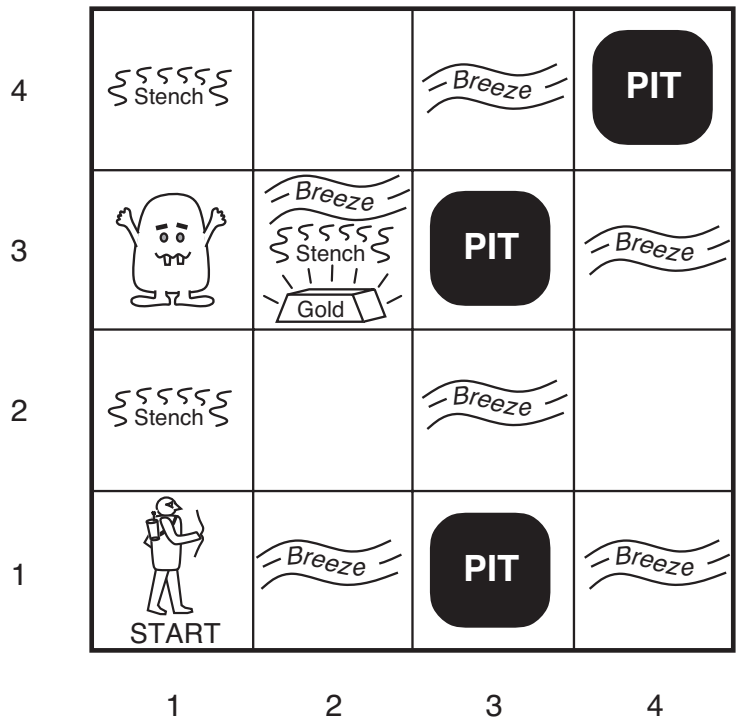
Representation + Reasoning = Intelligence!

# Outline



# Task Environment Wumpus World (PEAS)

- Performance measure: panjat keluar membawa emas +1000, mati -1000, gerak -1, panah -10
- Environment: Matriks 4x4. Lokasi initial: [1,1]. Ada emas, wumpus dan pit yang lokasinya dipilih secara acak di luar lokasi initial [1,1]
- Actuator: *maju*, belok *kiri* 90°, *kanan* 90°, *tembak* panah (hanya 1!), *ambil* emas, *panjat* keluar dari gua melalui [1,1].
- Sensor
  Stench: ruang di samping Wumpus berbau busuk
  Breeze: ruang di samping lubang jebakan ada hembusan angin
  Glitter: ruang di mana ada emas ada kilauan/sinar
  Bump: menabrak dinding ruang
  Scream: teriakan wumpus yang terbunuh



# Dimensi Environment dari Wumpus World

- Fully/partially observable?
- Deterministic/stochastic?
- Episodic/sequential?
- Static/dynamic?
- Discrete/continuous?
- Single/multi-agent?

# Dimensi Environment dari Wumpus World

- Fully/partially observable? partially; hanya bisa persepsi lokal
- Deterministic/stochastic?
- Episodic/sequential?
- Static/dynamic?
- Discrete/continuous?
- Single/multi-agent?

# Dimensi Environment dari Wumpus World

- Fully/partially observable? partially; hanya bisa persepsi lokal
- Deterministic/stochastic? deterministic; hasil tindakan jelas & pasti
- Episodic/sequential?

- Static/dynamic?
- Discrete/continuous?
- Single/multi-agent?

# Dimensi Environment dari Wumpus World

- Fully/partially observable? partially; hanya bisa persepsi lokal
- Deterministic/stochastic? deterministic; hasil tindakan jelas & pasti
- Episodic/sequential? sequential; reward dicapai setelah serangkaian action
- Static/dynamic?
- Discrete/continuous?
- Single/multi-agent?

# Dimensi Environment dari Wumpus World

- Fully/partially observable? partially; hanya bisa persepsi lokal
- Deterministic/stochastic? deterministic; hasil tindakan jelas & pasti
- Episodic/sequential? sequential; reward dicapai setelah serangkaian action
- Static/dynamic? static; emas, wumpus, pit tidak bergerak
- Discrete/continuous?
- Single/multi-agent?

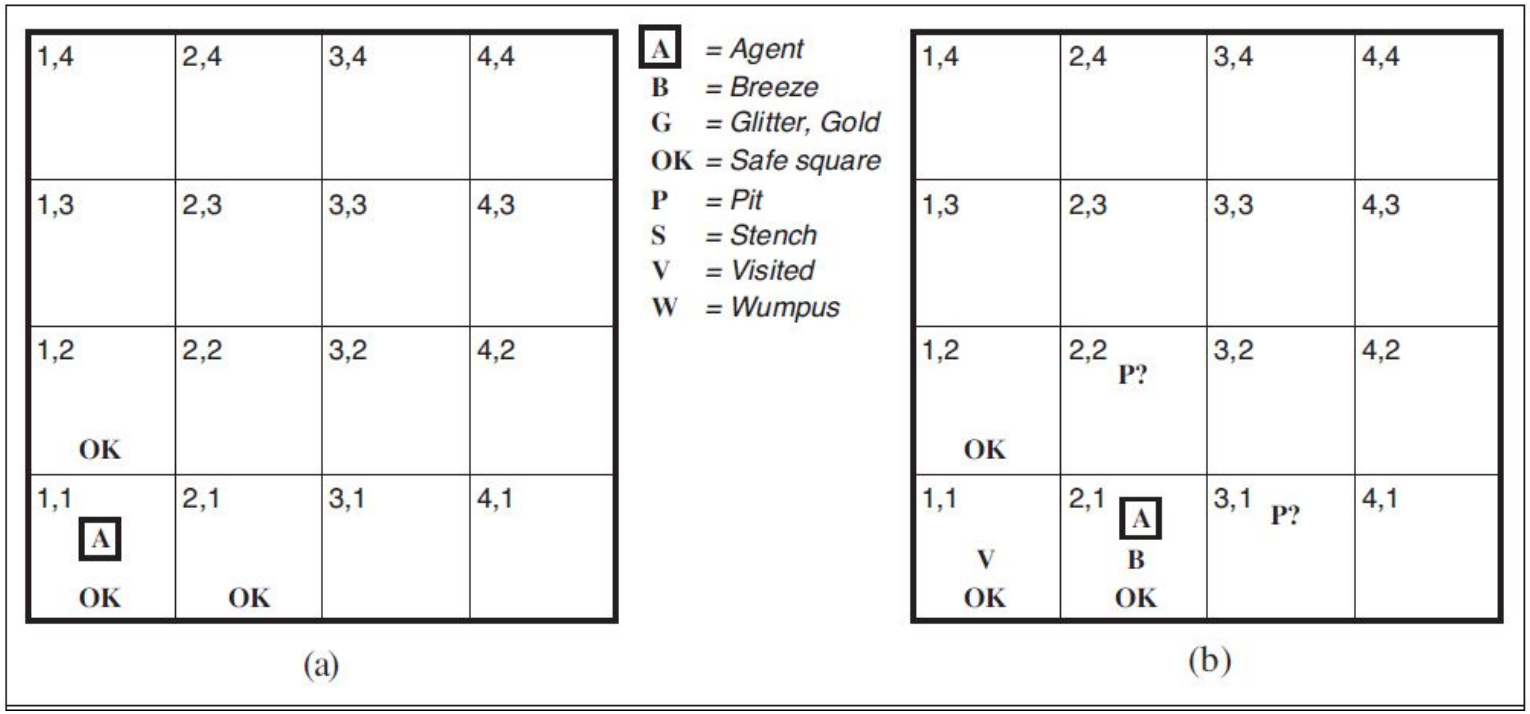
# Dimensi Environment dari Wumpus World

- Fully/partially observable? partially; hanya bisa persepsi lokal
- Deterministic/stochastic? deterministic; hasil tindakan jelas & pasti
- Episodic/sequential? sequential; reward dicapai setelah serangkaian action
- Static/dynamic? static; emas, wumpus, pit tidak bergerak
- Discrete/continuous? discrete
- Single/multi-agent?

# Dimensi Environment dari Wumpus World

- Fully/partially observable? partially; hanya bisa persepsi lokal
- Deterministic/stochastic? deterministic; hasil tindakan jelas & pasti
- Episodic/sequential? sequential; reward dicapai setelah serangkaian action
- Static/dynamic? static; emas, wumpus, pit tidak bergerak
- Discrete/continuous? discrete
- Single/multi-agent? single

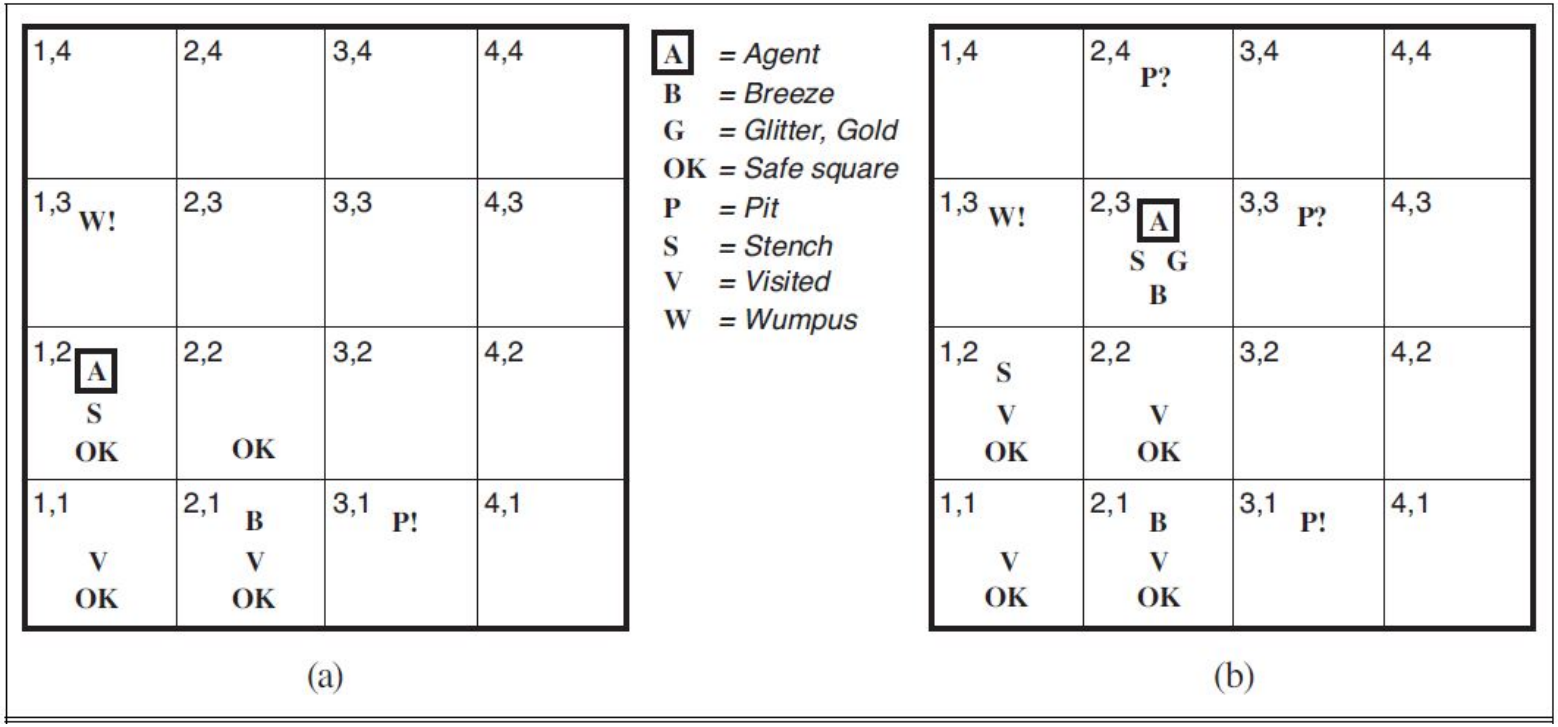
# Menjelajahi Wumpus World




(a) (b)

- Agent (A) berada di [1,1]; kamar [1,1] aman (OK).
- Percept [St,Br,Gl,Bu,Sc] di [1,1]: [*None*, *None*, *None*, *None*, *None*].
- Setelah bergerak satu langkah ke [2,1], menerima percept [*None*, *Breeze*, *None*, *None*, *None*].
- Hanya aman untuk kembali ke [1,1] dan bergerak ke [1,2].

# Menjelajahi Wumpus World




- Di kamar [1,2] menerima percept [*Stench*, *None*, *None*, *None*, *None*]. Lakukan penalaran untuk memperoleh kesimpulan gambar (a)!
- Singkat cerita, setelah langkah kelima, di kamar [2,3] menerima percept [*Stench*, *Breeze*, *Glitter*, *None*, *None*]: ambil emas dan balik untuk keluar dari gua.

# Outline



# Knowledge representation language

- Knowledge representation language (KRL): bahasa yang digunakan untuk menyatakan fakta tentang "dunia".
- Syntax: aturan yang mendefinisikan kalimat (sentence) yang sah dalam bahasa.
- Semantics: aturan yang mendefinisikan arti sebuah kalimat: kebenaran kalimat dalam suatu "dunia" tertentu (possible world).

# Contoh KRL: bahasa aritmetika

- Syntax:
  - $x + 2 \geq y$ adalah kalimat sah.
  - $x2 + y \geq$ bukan kalimat sah.
- Semantics: $x + 2 \geq y$ benar jhj bilangan $x + 2$ tidak lebih kecil dari bilangan $y$:
  - $x + 2 \geq y$ benar dalam "dunia" di mana $x = 7, y = 1$
  - $x + 2 \geq y$ salah dalam "dunia" di mana $x = 0, y = 6$

# Outline

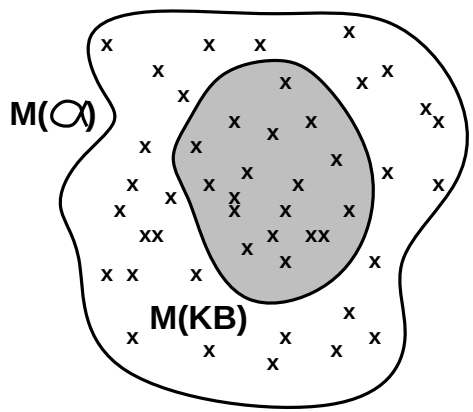


# Logika sebagai KRL

- Logika: bahasa formal untuk merepresentasikan fakta sehingga kesimpulan (fakta baru, jawaban) dapat ditarik.
- Ada banyak metode inference yang diketahui untuk penalaran.
- Kita bisa membangun agent Wumpus World dengan logika.

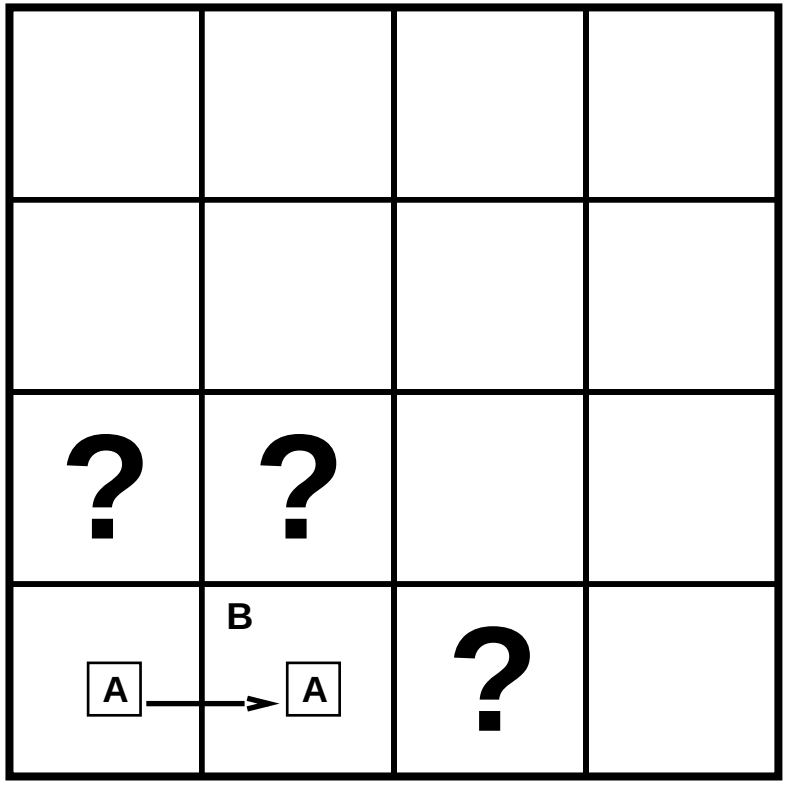
# Model dan Entailment

- Model: sebuah "dunia" di mana kebenaran suatu sentence bisa diuji: $m$ adalah model $\alpha$ jika $\alpha$ true di dalam $m$.
- $M(\alpha)$ adalah himpunan semua model dari $\alpha$
- Entailment berarti sesuatu fakta bisa disimpulkan dari (kumpulan) fakta lain.
- $KB \models \alpha$: KB entails sentence $\alpha$ jhj $\alpha$ true dalam setiap model di mana $KB$ true.
- $KB \models \alpha$ jhj $M(KB) \subseteq M(\alpha)$



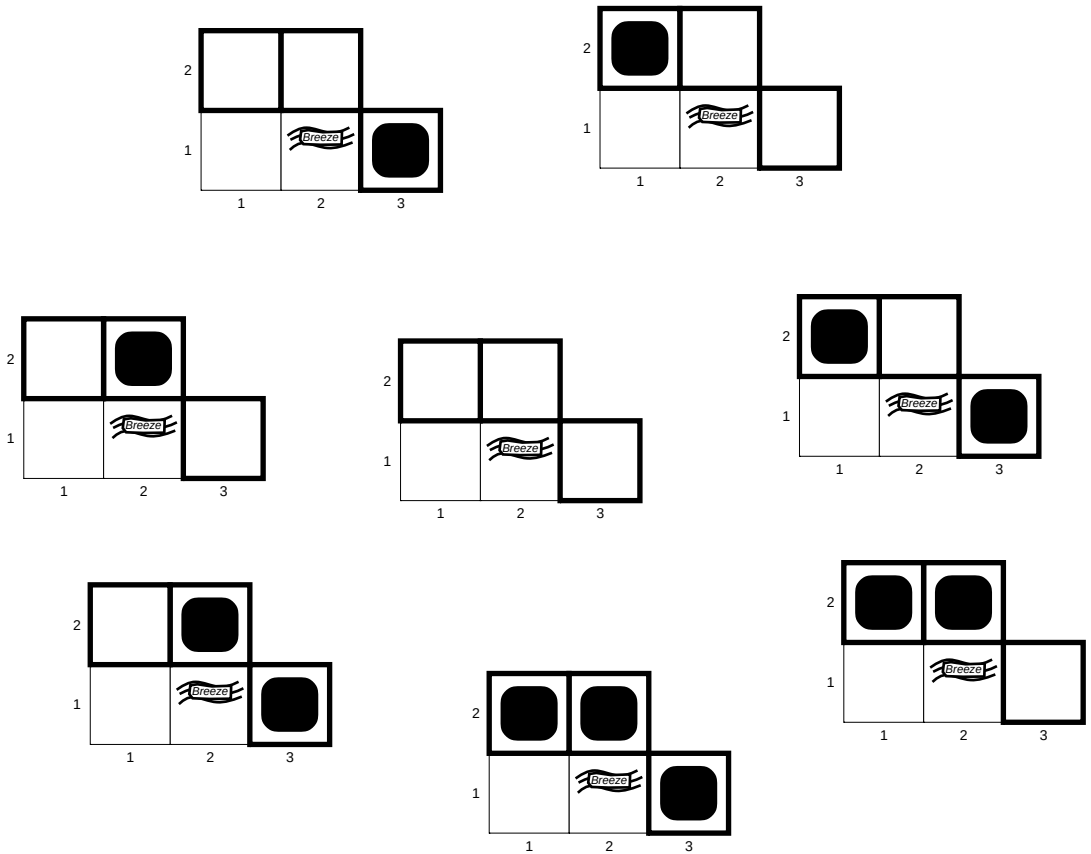
# Entailment dalam Wumpus World

Dengan persepsi None di [1,1] dan Breeze di [2,1]:

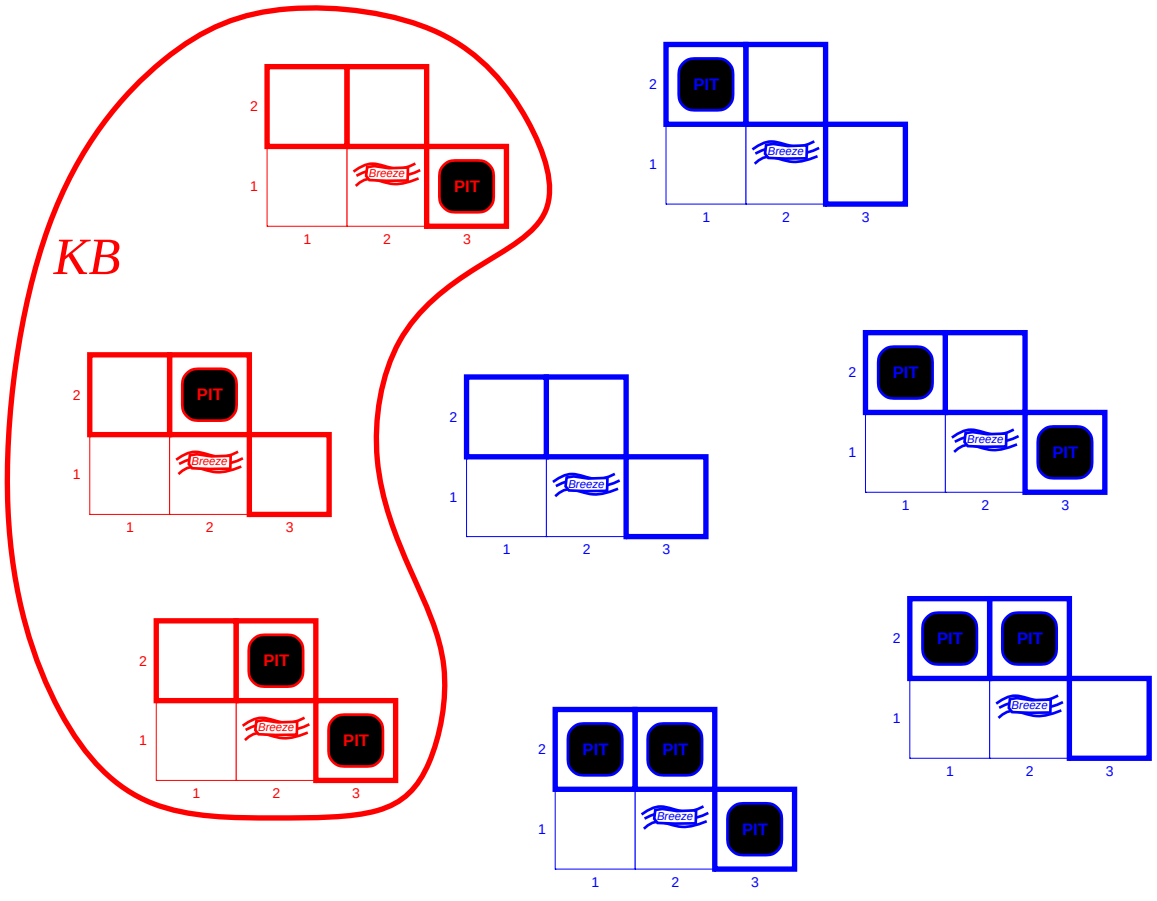



Apakah ada lubang jebakan di [1,2],[2,2],[3,1]? 3 pilihan boolean $\rightarrow$ 8 kemungkinan model.

# Kemungkinan model keberadaan lubang jebakan



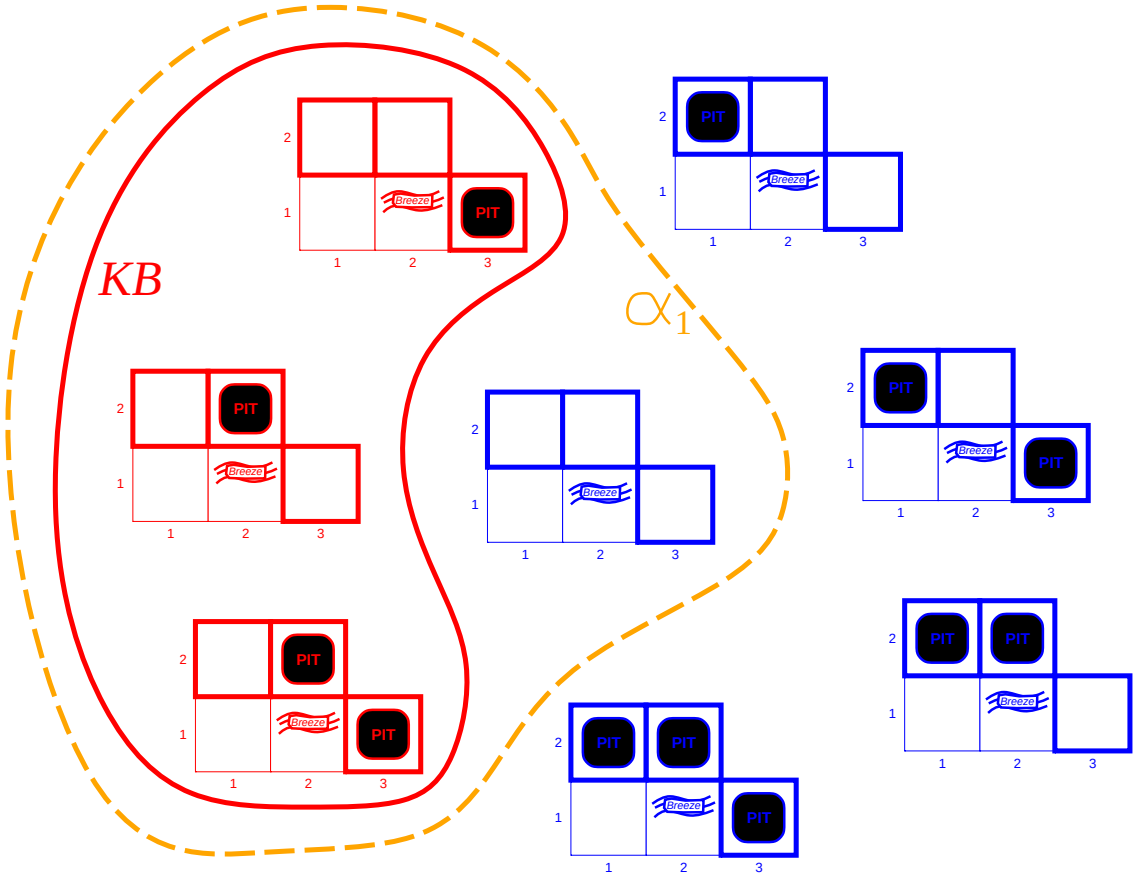
# Entailment dalam Wumpus World




$KB$ = pengamatan (percept) + aturan main Wumpus World

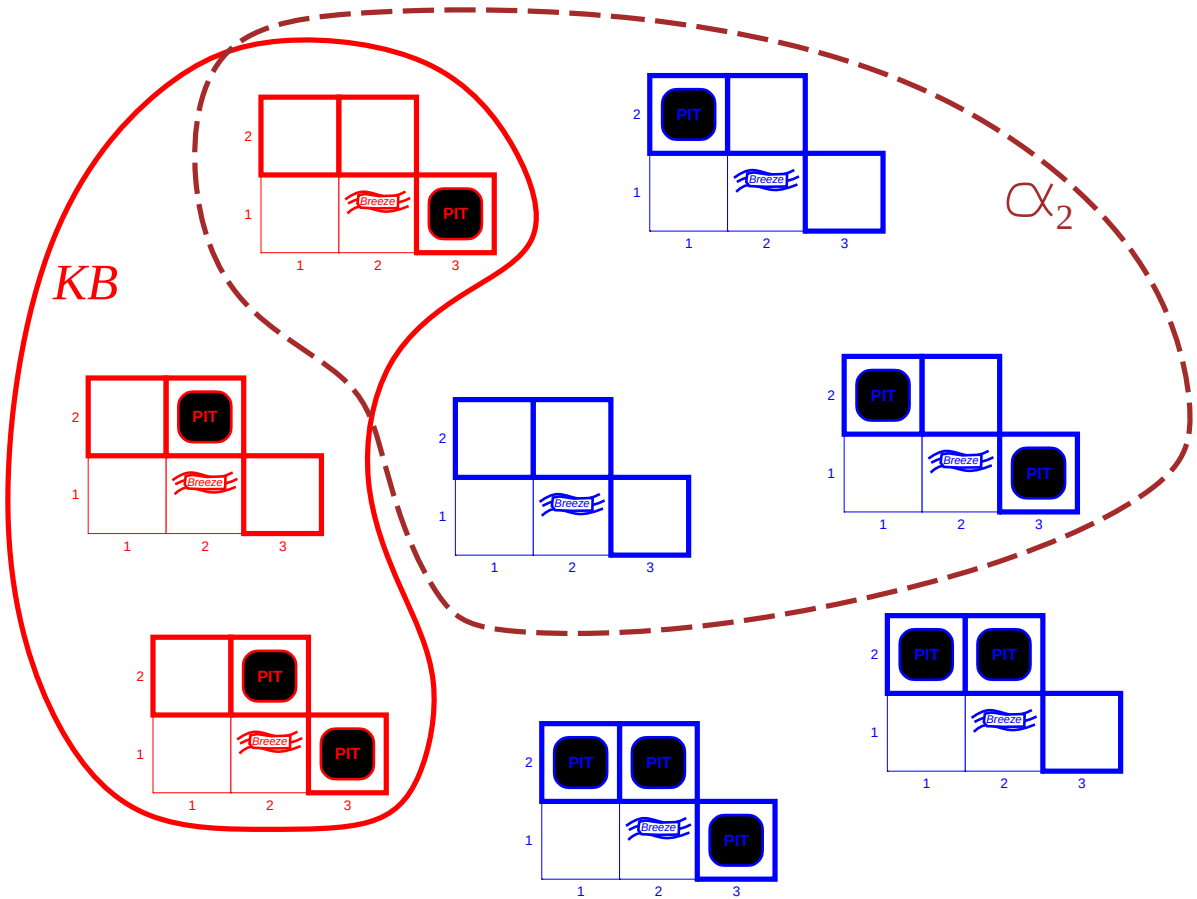
KB: None di [1,1] dan breeze di [2,1] + aturan main Wumpus World

# Model Checking




$\alpha_1$ = "Tidak ada lubang di kamar [1,2]", $KB \models \alpha_1$, dibuktikan dengan model checking: periksa semua kemungkinan model $M(KB)$ dan $M(\alpha_1)$

# Model Checking




$\alpha_2$ = "Tidak ada lubang di kamar [2,2]", $KB \not\models \alpha_2$

$\alpha_3$ = "Tidak ada lubang di kamar [3,1]", apakah $KB \models \alpha_3$?

# Inference Procedure

- Inference procedure bukan dilakukan pada level semantics, melainkan representasi fakta dalam KRL si agent (syntax).
- $KB \vdash_i \alpha$: sentence $\alpha$ bisa diturunkan dari $KB$ oleh prosedur $i$
- Soundness: $i$ dikatakan *sound*: untuk semua $\alpha$, jika $KB \vdash_i \alpha$, $KB \models \alpha$ benar
- Completeness: $i$ dikatakan *complete*: untuk semua $\alpha$, jika $KB \models \alpha$, $KB \vdash_i \alpha$ benar

# Outline



# Outline



# Propositional logic

- Propositional logic adalah logic yang paling sederhana
- Sebuah *sentence* dibentuk dari propositional symbol $P_1, P_2$, dst.

$$
\begin{aligned}
\mathit{Sentence} &\rightarrow \mathit{AtomicSentence} \mid \mathit{ComplexSentence} \\
\mathit{AtomicSentence} &\rightarrow \mathit{True} \mid \mathit{False} \mid P \mid Q \mid R \mid \ldots \\
\mathit{ComplexSentence} &\rightarrow (\ \mathit{Sentence}\ ) \mid [\ \mathit{Sentence}\ ] \\
&\mid \neg\ \mathit{Sentence} \\
&\mid \mathit{Sentence} \wedge \mathit{Sentence} \\
&\mid \mathit{Sentence} \vee \mathit{Sentence} \\
&\mid \mathit{Sentence} \Rightarrow \mathit{Sentence} \\
&\mid \mathit{Sentence} \Leftrightarrow \mathit{Sentence} \\
\text{OPERATOR PRECEDENCE} &: \neg, \wedge, \vee, \Rightarrow, \Leftrightarrow
\end{aligned}
$$

# *Semantics* dari propositional logic

- Sebuah kemungkinan model memberi menilai *true*/*false* terhadap setiap *proposition*, mis:

| $P_{1,2}$ | $P_{2,2}$ | $P_{3,1}$ |
|---|---|---|
| *false* | *false* | *true* |

(Dari 3 *propositional symbol* dapat dijabarkan 8 kemungkinan model)

**Aturan menentukan kebenaran sebuah kalimat untuk model *m*:**

| | | | | | |
|---|---|---|---|---|---|
| $\neg S$ | true iff | $S$ | false | | dalam model $m$ |
| $S_1 \wedge S_2$ | true iff | $S_1$ | true and | $S_2$ true | dalam model $m$ |
| $S_1 \vee S_2$ | true iff | $S_1$ | true or | $S_2$ true | dalam model $m$ |
| $S_1 \Rightarrow S_2$ | true iff | $S_1$ | false or | $S_2$ true | dalam model $m$ |
| dkl. | false iff | $S_1$ | true and | $S_2$ false | dalam model $m$ |
| $S_1 \Leftrightarrow S_2$ | true iff | $S_1 \Rightarrow S_2$ | true and | $S_2 \Rightarrow S_1$ true | dalam model $m$ |

- Kebenaran sebuah kalimat logika proposisi dapat dievaluasi secara rekursif.

# Wumpus World: Contoh Representasi KB & Entailment

- Propositional Symbols:
  - $P_{i,j}$ = *true* kalau ada lubang jebakan (*pit*) di $[i, j]$.
  - $B_{i,j}$ = *true* kalau ada hembusan angin (*breeze*) di $[i, j]$.
- Aturan main: *kamar di samping lubang jebakan ada hembusan angin* dan *tidak ada lubang jebakan di [1,1]*
  - $R_1 : \neg P_{1,1}$
  - $R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
  - $R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$
- Hasil pengamatan (*percept*):
  - $R_4 : \neg B_{1,1}$
  - $R_5 : B_{2,1}$
- $KB$ = aturan main Wumpus World + percept
  - $KB = R_1 \wedge R_2 \wedge R_3 \wedge R_4 \wedge R_5$
- KB $\overset{?}{\models} \neg P_{1,2}$

# Outline



# Jenis-jenis Inference Procedure

Secara umum, ada 2 jenis:

- Model checking
  - Penjabaran *truth table* (eksponensial dalam $n$)
  - Backtracking lebih efisien, mis: algoritma DPLL
  - Local search dalam *model space* (*sound* tapi *incomplete*), mis: min-conflicts hill-climbing
- Theorem proving
  - Hasilkan kalimat baru yang sah (*sound*) dari yang lama
  - Bukti (proof): serangkaian penerapan *inference rule*
    Inference rule sebagai action $\rightarrow$ algoritma search.
  - Biasanya, kalimat harus diterjemahkan ke dalam sebuah normal form

# Wumpus World: Inference dengan truth-table

Kita dapat membuktikan apakah $KB \models \alpha_1$ menggunakan *truth table*, dimana $\alpha_1 : \neg P_{1,2}$. Ini adalah pendekatan model checking.

| $B_{1,1}$ | $B_{2,1}$ | $P_{1,1}$ | $P_{1,2}$ | $P_{2,1}$ | $P_{2,2}$ | $P_{3,1}$ | $KB$ | $\alpha_1$ |
|---|---|---|---|---|---|---|---|---|
| false | false | false | false | false | false | false | false | true |
| false | false | false | false | false | false | true | false | true |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| false | true | false | false | false | false | false | false | true |
| false | true | false | false | false | false | true | true | true |
| false | true | false | false | false | true | false | true | true |
| false | true | false | false | false | true | true | true | true |
| false | true | false | false | true | false | false | false | true |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| true | true | true | true | true | true | true | false | false |

# Inference Procedure dengan truth-table

**function** TT-ENTAILS?($KB, \alpha$) **returns** *true* or *false*
  **inputs**: $KB$, the knowledge base, a sentence in propositional logic
    $\alpha$, the query, a sentence in propositional logic

  $symbols \leftarrow$ a list of the proposition symbols in $KB$ and $\alpha$
  **return** TT-CHECK-ALL($KB, \alpha, symbols, \{\}$)

---

**function** TT-CHECK-ALL($KB, \alpha, symbols, model$) **returns** *true* or *false*
  **if** EMPTY?($symbols$) **then**
    **if** PL-TRUE?($KB, model$) **then return** PL-TRUE?($\alpha, model$)
    **else return** *true* // *when KB is false, always return true*
  **else do**
    $P \leftarrow$ FIRST($symbols$)
    $rest \leftarrow$ REST($symbols$)
    **return** (TT-CHECK-ALL($KB, \alpha, rest, model \cup \{P = true\}$)
      **and**
      TT-CHECK-ALL($KB, \alpha, rest, model \cup \{P = false\}$))

**Figure 7.8** A truth-table enumeration algorithm for deciding propositional entailment. (TT stands for truth table.) PL-TRUE? returns *true* if a sentence holds within a model. The variable *model* represents a partial model—an assignment to some of the symbols. The keyword "**and**" is used here as a logical operation on its two arguments, returning *true* or *false*.

# Inference Procedure dengan Theorem-Proving

## *Logical Equivalence*

- *Dua kalimat logically equivalent jhj mereka benar dalam model yang sama:* $\alpha \equiv \beta$ *jhj* $\alpha \models \beta$ *dan* $\beta \models \alpha$

| | | | |
|---|---|---|---|
| $(\alpha \wedge \beta)$ | $\equiv$ | $(\beta \wedge \alpha)$ | commutativity of $\wedge$ |
| $(\alpha \vee \beta)$ | $\equiv$ | $(\beta \vee \alpha)$ | commutativity of $\vee$ |
| $((\alpha \wedge \beta) \wedge \gamma)$ | $\equiv$ | $(\alpha \wedge (\beta \wedge \gamma))$ | associativity of $\wedge$ |
| $((\alpha \vee \beta) \vee \gamma)$ | $\equiv$ | $(\alpha \vee (\beta \vee \gamma))$ | associativity of $\vee$ |
| $\neg(\neg\alpha)$ | $\equiv$ | $\alpha$ | double-negation elimination |
| $(\alpha \Rightarrow \beta)$ | $\equiv$ | $(\neg\beta \Rightarrow \neg\alpha)$ | contraposition |
| $(\alpha \Rightarrow \beta)$ | $\equiv$ | $(\neg\alpha \vee \beta)$ | implication elimination |
| $(\alpha \Leftrightarrow \beta)$ | $\equiv$ | $((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$ | biconditional elimination |
| $\neg(\alpha \wedge \beta)$ | $\equiv$ | $(\neg\alpha \vee \neg\beta)$ | de Morgan |
| $\neg(\alpha \vee \beta)$ | $\equiv$ | $(\neg\alpha \wedge \neg\beta)$ | de Morgan |
| $(\alpha \wedge (\beta \vee \gamma))$ | $\equiv$ | $((\alpha \wedge \beta) \vee (\alpha \wedge \gamma))$ | distributivity of $\wedge$ over $\vee$ |
| $(\alpha \vee (\beta \wedge \gamma))$ | $\equiv$ | $((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$ | distributivity of $\vee$ over $\wedge$ |

# Inference Procedure dengan Theorem-Proving

## *Validity*

- Sebuah kalimat valid jika ia *true* dalam semua model

**Deduction Theorem**

$KB \models \alpha$ jika dan hanya jika $(KB \Rightarrow \alpha)$ valid

## *Satisfiability*

- Sebuah kalimat satisfiable jika ada model di mana ia *true*
- Sebuah kalimat unsatisfiable jika tidak ada model di mana ia *true*

**Reductio ad absurdum (proof by contradiction)**

$KB \models \alpha$ jika dan hanya jika $(KB \wedge \neg\alpha)$ unsatisfiable

# Rules of Inference

- Sebuah inference rule adalah pola *syntax* yang dapat menurunkan sebuah kalimat baru yang sah (*sound*).
- Rule yang paling terkenal adalah modus ponens:

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

- Contoh rule lain: and elimination:

$$\frac{\alpha \wedge \beta}{\alpha} \text{ dan } \frac{\alpha \wedge \beta}{\beta}$$

- Semua logical equivalence juga bisa dipakai sebagai inference rule.

Untuk membuktikan $KB \models \alpha$, kita bisa mengaplikasikan serangkaian inference rule mulai dari $KB$ dengan hasil akhir $\alpha$.

Jika kita gunakan semua *inference rule* sebagai action $\rightarrow$ algoritma search!

# Wumpus World: Inference dengan Theorem-Proving

| | | |
|---|---|---|
| $R_1$ : | $\neg P_{1,1}$ | $KB$ |
| $R_2$ : | $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$ | $KB$ |
| $R_3$ : | $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$ | $KB$ |
| $R_4$ : | $\neg B_{1,1}$ | $KB$ |
| $R_5$ : | $B_{2,1}$ | $KB$ |

# Wumpus World: Inference dengan Theorem-Proving

| | | |
|---|---|---|
| $R_1$ : | $\neg P_{1,1}$ | $KB$ |
| $R_2$ : | $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$ | $KB$ |
| $R_3$ : | $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$ | $KB$ |
| $R_4$ : | $\neg B_{1,1}$ | $KB$ |
| $R_5$ : | $B_{2,1}$ | $KB$ |
| $R_6$ : | $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ | $\Leftrightarrow$-elim.: $R_2$ |

# Wumpus World: Inference dengan Theorem-Proving

| | | |
|---|---|---|
| $R_1$ : | $\neg P_{1,1}$ | $KB$ |
| $R_2$ : | $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$ | $KB$ |
| $R_3$ : | $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$ | $KB$ |
| $R_4$ : | $\neg B_{1,1}$ | $KB$ |
| $R_5$ : | $B_{2,1}$ | $KB$ |
| $R_6$ : | $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ | $\Leftrightarrow$-elim.: $R_2$ |
| $R_7$ : | $(P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1}$ | $\wedge$-elim.: $R_6$ |

# Wumpus World: Inference dengan Theorem-Proving

| | | |
|---|---|---|
| $R_1$ : | $\neg P_{1,1}$ | $KB$ |
| $R_2$ : | $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$ | $KB$ |
| $R_3$ : | $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$ | $KB$ |
| $R_4$ : | $\neg B_{1,1}$ | $KB$ |
| $R_5$ : | $B_{2,1}$ | $KB$ |
| $R_6$ : | $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ | $\Leftrightarrow$-elim.: $R_2$ |
| $R_7$ : | $(P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1}$ | $\wedge$-elim.: $R_6$ |
| $R_8$ : | $\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1})$ | contrapos.:$R_7$ |

# Wumpus World: Inference dengan Theorem-Proving

| | | |
|---|---|---|
| $R_1$ : | $\neg P_{1,1}$ | $KB$ |
| $R_2$ : | $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$ | $KB$ |
| $R_3$ : | $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$ | $KB$ |
| $R_4$ : | $\neg B_{1,1}$ | $KB$ |
| $R_5$ : | $B_{2,1}$ | $KB$ |
| $R_6$ : | $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ | $\Leftrightarrow$-elim.: $R_2$ |
| $R_7$ : | $(P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1}$ | $\wedge$-elim.: $R_6$ |
| $R_8$ : | $\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1})$ | contrapos.:$R_7$ |
| $R_9$ : | $\neg(P_{1,2} \vee P_{2,1})$ | m. ponens: $R_8$ & $R_4$ |

# Wumpus World: Inference dengan Theorem-Proving

| | | |
|---|---|---|
| $R_1$ : | $\neg P_{1,1}$ | $KB$ |
| $R_2$ : | $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$ | $KB$ |
| $R_3$ : | $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$ | $KB$ |
| $R_4$ : | $\neg B_{1,1}$ | $KB$ |
| $R_5$ : | $B_{2,1}$ | $KB$ |
| $R_6$ : | $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ | $\Leftrightarrow$-elim.: $R_2$ |
| $R_7$ : | $(P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1}$ | $\wedge$-elim.: $R_6$ |
| $R_8$ : | $\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1})$ | contrapos.:$R_7$ |
| $R_9$ : | $\neg(P_{1,2} \vee P_{2,1})$ | m. ponens: $R_8$ & $R_4$ |
| $R_{10}$ : | $\neg P_{1,2} \wedge \neg P_{2,1}$ | de Morgan: $R_9$ |

# Wumpus World: Inference dengan Theorem-Proving

| | | |
|---|---|---|
| $R_1$ : | $\neg P_{1,1}$ | $KB$ |
| $R_2$ : | $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$ | $KB$ |
| $R_3$ : | $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$ | $KB$ |
| $R_4$ : | $\neg B_{1,1}$ | $KB$ |
| $R_5$ : | $B_{2,1}$ | $KB$ |
| $R_6$ : | $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ | $\Leftrightarrow$-elim.: $R_2$ |
| $R_7$ : | $(P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1}$ | $\wedge$-elim.: $R_6$ |
| $R_8$ : | $\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1})$ | contrapos.:$R_7$ |
| $R_9$ : | $\neg(P_{1,2} \vee P_{2,1})$ | m. ponens: $R_8$ & $R_4$ |
| $R_{10}$ : | $\neg P_{1,2} \wedge \neg P_{2,1}$ | de Morgan: $R_9$ |
| $R_{11}$ : | $\neg P_{1,2}$ | $\wedge$-elim.: $R_{10}$ |

# Pembuktian dengan *resolution*

- Conjunctive Normal Form (CNF): *conjunction* of *clauses* (*disjunction* of *literals*), mis: $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$
- Resolution inference rule:

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

  di mana $\ell_i$ dan $m_j$ adalah complementary literal (mis: $P$ dan $\neg P$).
- Contoh pada Wumpus World:
  - Saat berada di [1,2], tidak dirasakan hembusan angin; pada KB ditambahkan $\neg B_{1,2}$ dan $B_{1,2} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{1,3})$
  - Latihan: Seperti langkah pada slides sebelumnya, buktikan $\neg P_{2,2}$ dan $\neg P_{1,3}$ berlaku!
  - Dari $R_3$ dan $R_5$ diperoleh: $P_{1,1} \vee P_{2,2} \vee P_{3,1}$.
  - Resolusi:

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

- Resolution adalah inference procedure yang *sound* dan *complete*!

# Algoritma *resolution*

## Reductio ad absurdum (proof by contradiction)

$KB \models \alpha$ jika dan hanya jika $(KB \wedge \neg\alpha)$ unsatisfiable

- Terjemahkan $(KB \wedge \neg\alpha)$ ke dalam CNF.
- Terapkan algoritma resolution hingga mencapai empty clause. Mengapa?

## Algoritma Resolution

**function** PL-RESOLUTION($KB, \alpha$) **returns** *true* or *false*

*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*

# Wumpus World: Inference dengan Resolution

**Latihan: $KB \overset{?}{\models} \alpha$**

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

# Inference pada Horn Clauses dan Definite Clauses

- Horn Clause: clause yang terdiri dari paling banyak 1 literal positif.
  - Definite clause: clause yang terdiri dari tepat 1 literal positif
- Manakah yang merupakan Horn clause?
  - $\neg P \vee Q$
  - $\neg L \vee \neg M \vee P$
  - $\neg B \vee P \vee Q$
  - $A$
  - $\neg Q \vee \neg R$
- Resolusi pada dua Horn clause juga menghasilkan Horn clause
- KB yang terdiri dari definite clause = KB yang terdiri dari implikasi: $Body \Rightarrow Head$.
  - Body: conjunction literal positif; Head: sebuah literal positif
- Inference procedure pada definite clause: algoritma forward chaining dan backward chaining → pembahasan pada FOL.

# Outline



# Satisfiability testing sebagai CSP

- Satisfiability testing untuk $KB \models \alpha$: CSP
  - Variable: propositional symbols dari $KB$ dan $\alpha$
  - Domain: {true,false}
  - Constraint: $KB \wedge \neg\alpha$ mengembalikan false
- Secara teoritis: depth-first search semua kemungkinan model = truth table!
- Algoritma backtracking untuk CSP bisa dipakai.
- Ada heuristic yang bisa digunakan untuk pruning

# Algoritma Davis-Putnam-Logemann-Loveland

- Algoritma DPLL adalah backtracking + heuristic
- Sentence harus dalam bentuk CNF (conjunctive normal form)
- Memanfaatkan tiga heuristic
  - Early termination
  - Pure symbol
  - Unit clause

# Heuristic: early termination

- DPLL mendeteksi sentence true atau false sebelum model lengkap (semua symbol telah di-assign).
- Sebuah clause bernilai *true* jika *ada* literal yang *true*.
  Mis.: Jika $A = true, (A \vee B) \wedge (A \vee C)$ pasti *true* juga
- Sebuah sentence bernilai *false* jika *ada* clause yang *false*.

# Heuristic: pure symbol

- Pure symbol adalah symbol yang muncul dengan "tanda" yang sama dalam semua clause.
- Mis.: $(A \vee \neg B) \wedge (\neg B \vee \neg C) \wedge (C \vee A)$
  $A$ adalah pure symbol karena selalu muncul "positif"
  $B$ adalah pure symbol karena selalu muncul "negatif"
  $C$ bukan pure symbol
- Jika *pure symbol* diberi nilai shg. *literal*-nya *true*, *clause* di mana ia muncul pasti *true*.
- Dalam menentukan apakah symbol itu *pure* atau tidak, abaikan *clause* yang sudah diketahui *true*.
  Mis.: jika $B = \mathit{false}$, maka pada *sentence* di atas $(\neg B \vee \neg C)$ pasti *true*, sehingga $C$ pada *sentence* di atas menjadi *pure*.

# Heuristic: unit clause

- Unit clause adalah *clause* dengan hanya satu *literal*, atau *clause* dengan $n > 1$ literal di mana semua $(n - 1)$ literal telah bernilai *false*.
- Mis.: Jika $C = true$, $(B \vee \neg C)$ adalah *unit clause*. Agar true, $B$ harus di-*assign* nilai *true*.
- Heuristic ini diterapkan terlebih dahulu sebelum assignment (*branching*) pada symbol lain dilakukan.
- Meng-*assign* nilai symbol di dalam *unit clause* dapat berakibat timbul *unit clause* lain $\rightarrow$ unit propagation.
  - Mirip forward chaining pada definite clause.

# Algoritma DPLL

## Algoritma DPLL

**function** DPLL-SATISFIABLE?($s$) **returns** *true* or *false*
**inputs**: $s$, a sentence in propositional logic

*clauses* $\leftarrow$ the set of clauses in the CNF representation of $s$
*symbols* $\leftarrow$ a list of the proposition symbols in $s$
**return** DPLL(*clauses*, *symbols*, [ ])

---

**function** DPLL(*clauses*, *symbols*, *model*) **returns** *true* or *false*

**if** every clause in *clauses* is true in *model* **then return** *true*
**if** some clause in *clauses* is false in *model* **then return** *false*
$P$, *value* $\leftarrow$ FIND-PURE-SYMBOL(*symbols*, *clauses*, *model*)
**if** $P$ is non-null **then return** DPLL(*clauses*, *symbols*–$P$, [$P =$ *value*|*model*])
$P$, *value* $\leftarrow$ FIND-UNIT-CLAUSE(*clauses*, *model*)
**if** $P$ is non-null **then return** DPLL(*clauses*, *symbols*–$P$, [$P =$ *value*|*model*])
$P \leftarrow$ FIRST(*symbols*); *rest* $\leftarrow$ REST(*symbols*)
**return** DPLL(*clauses*, *rest*, [$P =$ *true*|*model*]) **or** DPLL(*clauses*, *rest*, [$P =$ *false*|*model*])

# Outline



# Pendekatan local search

- Goal dari *satisfiability*: mencari *assignment* sehingga setiap *clause* bernilai *true*.
- Ide baru: berikan *complete assignment* secara acak, lalu coba ganti nilai kebenaran simbol proposisi pada clause yang belum terpenuhi (*unsatisfied clause*).
- Dua cara memilih simbol proposisi yang diganti nilai kebenarannya:
  - *min-conflicts*: meminimalkan jumlah *unsatisfied clauses*.
  - *random-walk*: memilih simbol secara acak.

# Algoritma local search untuk satisfiability solving

## Algoritma WalkSAT

**function** WalkSAT(*clauses*, *p*, *max-flips*) **returns** a satisfying model or *failure*
  **inputs**: *clauses*, a set of clauses in propositional logic
    *p*, the probability of choosing to do a "random walk" move, typically around 0.5
    *max-flips*, number of flips allowed before giving up

  *model* ← a random assignment of *true*/*false* to the symbols in *clauses*
  **for** $i = 1$ **to** *max-flips* **do**
    **if** *model* satisfies *clauses* **then return** *model*
    *clause* ← a randomly selected clause from *clauses* that is false in *model*
    **with probability** *p* flip the value in *model* of a randomly selected symbol from *clause*
    **else** flip whichever symbol in *clause* maximizes the number of satisfied clauses
  **return** *failure*

- Dalam praktek, WalkSAT bisa lebih cepat dari DPLL!
- Namun, WalkSAT tidak cocok untuk mendeteksi unsatisfiability.