Universität Bielefeld

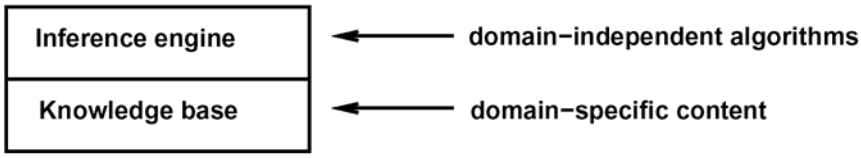
Logical Agents

*(Chapter 7 (Russel & Norvig, 2004))

Outline

- Knowledge-based agents
- Wumpus world
- Logic in general - models and entailment
- Propositional (Boolean) logic
- Equivalence, validity, satisfiability
- Inference rules and theorem proving
  - forward chaining
  - backward chaining
  - resolution

Knowledge bases



- Knowledge base = set of **sentences** in a **formal** language
- **Declarative** approach to building an agent (or other system):
  - `Tell` it what it needs to know
- Then it can `Ask` itself what to do - answers should follow from the KB
- Agents can be viewed at the **knowledge level**
  i.e., what they know, regardless of how implemented
- Or at the **implementation level**
  - i.e., data structures in KB and algorithms that manipulate them

A simple knowledge-based agent

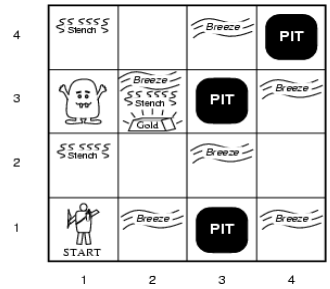
```
function KB-AGENT(percept) returns an action
   static: KB, a knowledge base
           t, a counter, initially 0, indicating time

   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
```

- The agent must be able to:
  - Represent states, actions, etc.
  - Incorporate new percepts
  - Update internal representations of the world
  - Deduce hidden properties of the world
  - Deduce appropriate actions

Wumpus World PEAS description

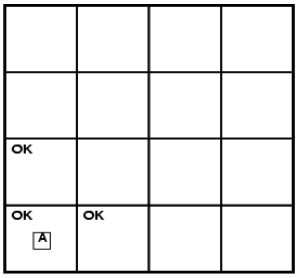
- **Performance measure**
  - gold +1000, death -1000
  - -1 per step, -10 for using the arrow
- **Environment**
  - Squares adjacent to wumpus are smelly
  - Squares adjacent to pit are breezy
  - Glitter iff gold is in the same square
  - Shooting kills wumpus if you are facing it
  - Shooting uses 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



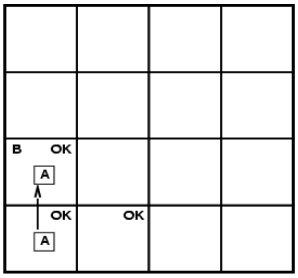
Wumpus world characterization

- Fully Observable No – only **local** perception
- Deterministic Yes – outcomes exactly specified
- Episodic No – sequential at the level of actions
- Static Yes – Wumpus and Pits do not move
- Discrete Yes
- Single-agent? Yes – Wumpus is essentially a natural feature

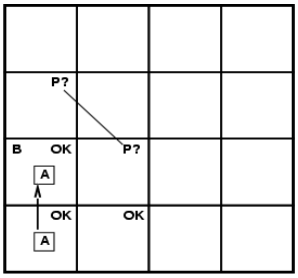
Exploring a wumpus world



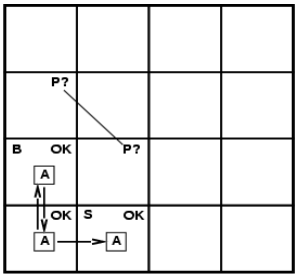
Exploring a wumpus world



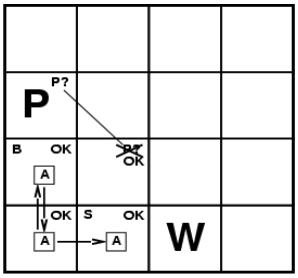
Exploring a wumpus world



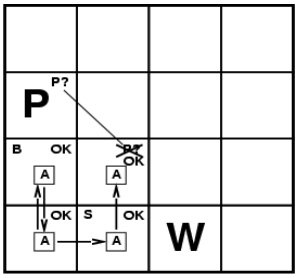
Exploring a wumpus world



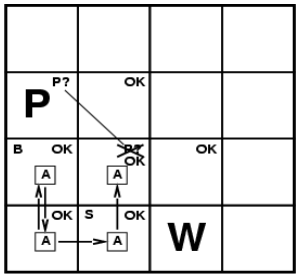
Exploring a wumpus world



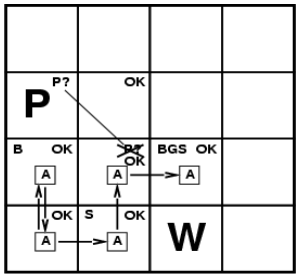
Exploring a wumpus world



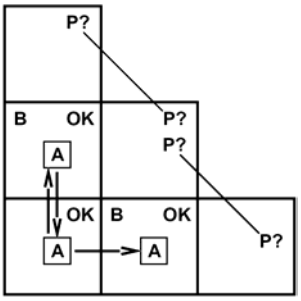
Exploring a wumpus world



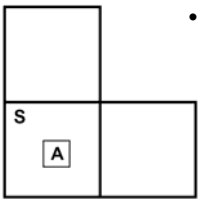
Exploring a wumpus world



Exploring a wumpus world



- Breeze in (2,1) and (1,2):
  - no safe actions
- Assuming evenly distribution of pits:
  - (2,2) has pit w/ prob 0.86, vs. 0.31



- Smell in (1,1)
  - cannot move, can use a strategy of coercion:
    - shoot straight ahead
    - if wumpus was there then
      - wumpus dead
      - if wumpus dead then safe
    - else safe

Logic in general

- **Logics** are formal languages for representing information such that conclusions can be drawn
- **Syntax** defines the sentences in the language
- **Semantics** define the "meaning" of sentences;
  - i.e., define **truth** of a sentence in a world

- E.g., the language of arithmetic
  - $x+2 \geq y$ 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** means that one thing logically follows from another:

$$KB \models \alpha$$

- Knowledge base *KB* entails sentence α if and only if α is true in all worlds where *KB* is true
  - E.g., the KB containing "the Giants won" and "the Reds won" entails "Either the Giants won or the Reds won"
  - E.g., x+y = 4 entails 4 = x+y
  - Entailment is a relationship between sentences (i.e., syntax) that is based on semantics

Models

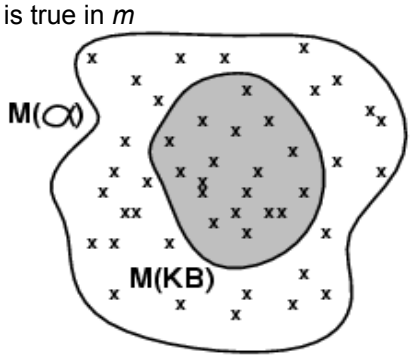
- Logicians typically think in terms of **models**, which are formally structured worlds with respect to which truth can be evaluated
- We say *m* **is a model of** a sentence α if α is true in *m*
- *M(α)* is the set of all models of α
- Then the **formal definition of KB** $\models$ **α** is:

  KB $\models$ α holds if and only if α is true in every model in which KB is true

  or

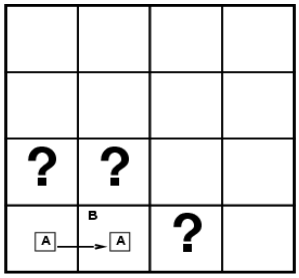
  iff $M(KB) \subseteq M(\alpha)$

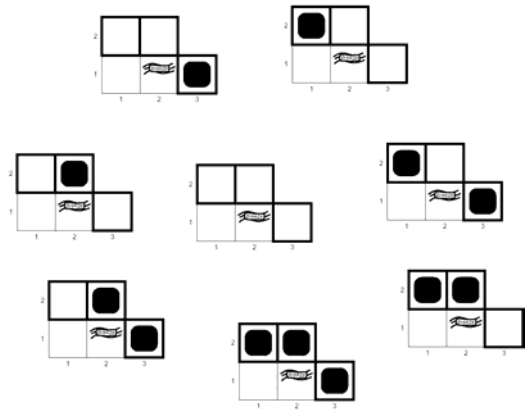
  - E.g. *KB* = Giants won and Reds won α = Giants won



Entailment in the wumpus world

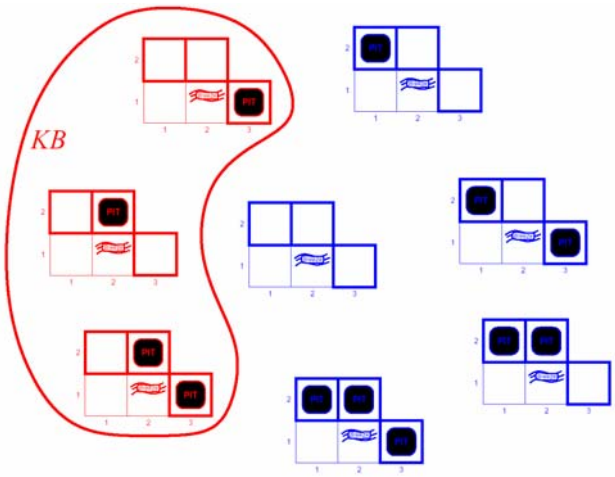
- Situation after detecting nothing in [1,1], moving right, breeze in [2,1]:





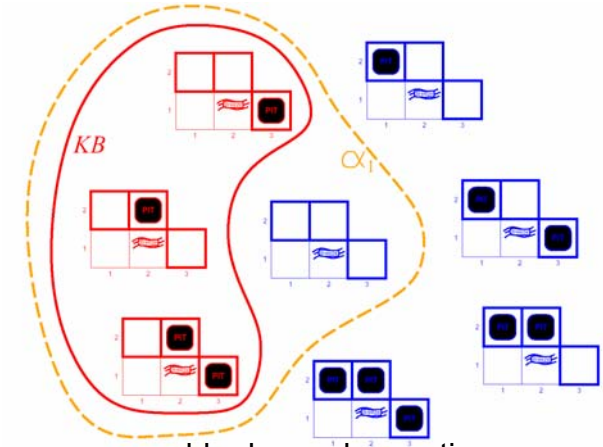
- Consider possible models for wumpus *KB* assuming only pits
- 3 Boolean choices ⇒ 8 possible models

Wumpus models



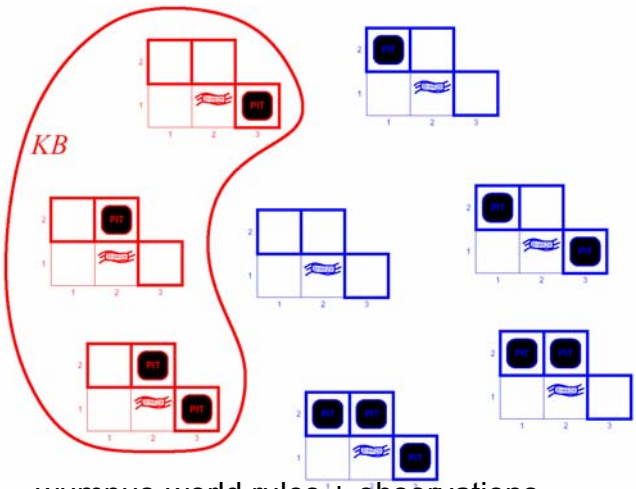
- *KB* = wumpus-world rules + observations

Wumpus models



- *KB* = wumpus-world rules + observations
- α_1 = "[1,2] is safe", *KB* $\models \alpha_1$, proved by **model checking**

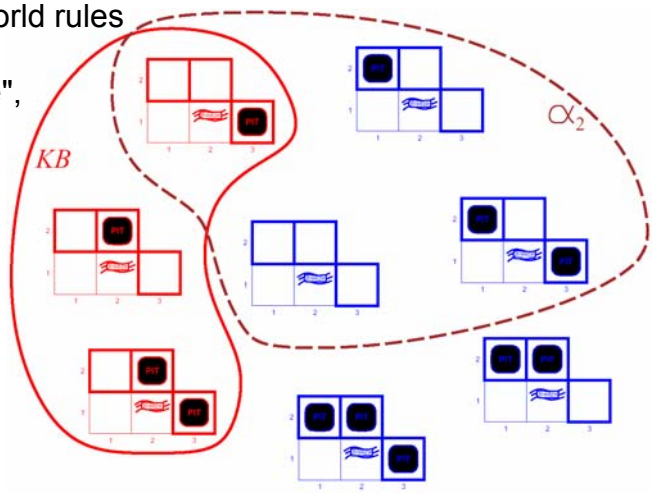
Wumpus models



- *KB* = wumpus-world rules + observations

Wumpus models

- *KB* = wumpus-world rules + observations
- α_2 = "[2,2] is safe", $KB \not\models \alpha_2$



Inference

- $KB \vdash_i \alpha$ = sentence α can be derived from *KB* by procedure *i*
- **Soundness**: *i* is sound if whenever $KB \vdash_i \alpha$, it is also true that $KB \models \alpha$
- **Completeness**: *i* is complete if whenever $KB \models \alpha$, it is also true that $KB \vdash_i \alpha$
- Preview: we will define a logic (first-order logic) which is expressive enough to say almost anything of interest, and for which there exists a sound and complete inference procedure.
- That is, the procedure will answer any question whose answer follows from what is known by the *KB*.

Propositional logic: Syntax

- Propositional logic is the simplest logic – illustrates basic ideas
- The proposition symbols P_1, P_2 etc are sentences
  - 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**)

Propositional logic: Semantics

Each model specifies true/false for each proposition symbol, e.g.:

$$m_1 = \{P_{1,2} = true, P_{2,2} = true, P_{3,1} = false\}$$

Using the 3 symbols, 8 possible models, can be enumerated automatically.

Rules for evaluating truth with respect to a 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
i.e.,	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

Simple recursive process evaluates an arbitrary sentence, e.g.,

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

Truth tables for connectives

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*

Wumpus world sentences

- Let $P_{i,j}$ be true if there is a pit in [i, j].
- Let $B_{i,j}$ be true if there is a breeze in [i, j].
- For all wumpus worlds:
  - No pit in $P_{1,1}$
  - "Pits cause breezes in adjacent squares“ or
    “A square is breezy if and only if there is an adjacent pit”
- For the specific world:
  - No breeze in $B_{1,1}$
  - Breeze in $B_{2,1}$

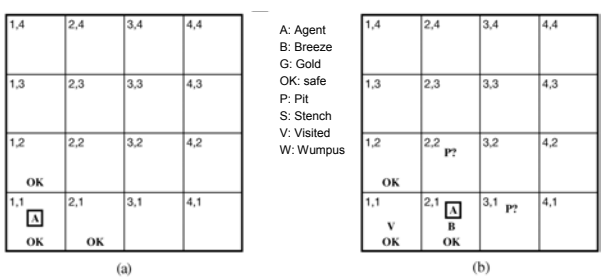
$$R_1 : \neg P_{1,1}$$

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

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

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

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



Truth tables for inference

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	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*

$KB : R_1, R_2, R_3, R_4, R_5$ implies $R_1 \wedge R_2 \wedge R_3 \wedge R_4 \wedge R_5$

- Enumerate rows (different assignments to symbols), if KB is true in row, check that α is too.

Inference by enumeration

- Depth-first enumeration of all models is sound and complete

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

   symbols ← a list of the proposition symbols in KB and α
   return TT-CHECK-ALL(KB, α, symbols, [])
```

```
function TT-CHECK-ALL(KB, α, symbols, model) returns true or false
   if EMPTY?(symbols) then
      if PL-TRUE?(KB, model) then return PL-TRUE?(α, model)
      else return true
   else do
      P ← FIRST(symbols); rest ← REST(symbols)
      return TT-CHECK-ALL(KB, α, rest, EXTEND(P, true, model)) and
             TT-CHECK-ALL(KB, α, rest, EXTEND(P, false, model))
```

- For *n* symbols, time complexity is $O(2^n)$, space complexity is $O(n)$

Logical equivalence

- Two sentences are **logically equivalent** iff true in same models: $\alpha \equiv \beta$ iff $\alpha \models \beta$ and $\beta \models \alpha$

$(\alpha \wedge \beta)$	$\equiv$	$(\beta \wedge \alpha)$	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 $\vee$
$\neg(\neg\alpha)$	$\equiv$	α	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 $\wedge$

Validity and satisfiability

A sentence is **valid** if it is true in all models,
e.g., *True*, $A \vee \neg A$, $A \Rightarrow A$, $(A \wedge (A \Rightarrow B)) \Rightarrow B$

Validity is connected to inference via the **Deduction Theorem**:
$KB \models \alpha$ if and only if $(KB \Rightarrow \alpha)$ is valid

A sentence is **satisfiable** if it is true in **some** model
e.g., $A \vee B$, C

A sentence is **unsatisfiable** if it is true in **no** models
e.g., $A \wedge \neg A$

Satisfiability is connected to inference via the following:
$KB \models \alpha$ if and only if $(KB \wedge \neg\alpha)$ is unsatisfiable

Proof methods

- Proof methods divide into (roughly) two kinds:

  1. **Model checking**
     - truth table enumeration (always exponential in n)
     - improved backtracking, e.g., Davis--Putnam-Logemann-Loveland (DPLL)
     - heuristic search in model space (sound but incomplete)
     - e.g., min-conflicts-like hill-climbing algorithms

  2. **Application of inference rules**
     - Legitimate (sound) generation of new sentences from old
     - **Proof** = a sequence of inference rule applications
       Can use inference rules as operators in a standard search algorithm
     - Typically require transformation of sentences into a normal form

Inference in PL

- Inference rules
  - Modus Ponens: $\frac{\alpha \Rightarrow \beta, \alpha}{\beta}$
  - and-elimination: $\frac{\alpha \wedge \beta}{\alpha}$
  - all logical equivalences cam be used as inference rules, e.g.,
  - $\frac{\alpha \Leftrightarrow \beta}{(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)}$ and $\frac{(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)}{\alpha \Leftrightarrow \beta}$
  - but not all are bidirectional (see Modus Ponens)

Inference in PL

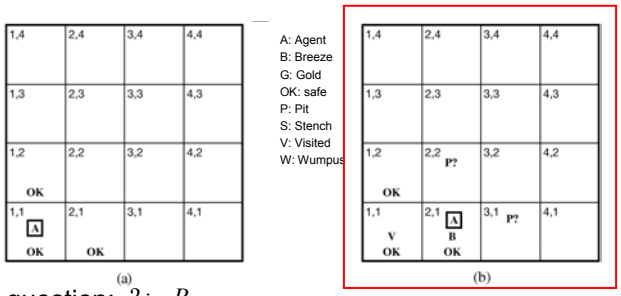
$R_1 : \neg P_{1,1}$

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

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

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

$R_5 : B_{2,1}$



- question: $? : \neg P_{1,2}$
- biconditional elimination of R_2: $R_6 : (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$
- and-elimination: $R_7 : ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$
- contraposition: $R_8 : (\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1}))$
- Modus Ponens (with perception R_4): $R_9 : \neg(P_{1,2} \vee P_{2,1})$
- de Morgan $R_{10} : \neg P_{1,2} \wedge \neg P_{2,1}$

➢ No pit in (1,2) and (2,1)