Logical Agents

Outline

I. Knowledge-based agents

II. The Wumpus world

III. Logic

* Figures/images are from the textbook site (or by the instructor).

I. Knowledge-Based Agents

- Problem solving agents do not know general facts.

  An 8-puzzle agent does not know that two tiles cannot occupy the same space.

- Their atomic representations are very limited.

  e.g., a list of all possible concrete states.

I. Knowledge-Based Agents

♣ Problem solving agents do not know general facts.

An 8-puzzle agent does not know that two tiles cannot occupy the same space.

♣ Their atomic representations are very limited.

e.g., a list of all possible concrete states.

♦ Intelligent agents need *knowledge about the world* in order to carry out reasoning for good decision making.

I. Knowledge-Based Agents

- Problem solving agents do not know general facts.

  An 8-puzzle agent does not know that two tiles cannot occupy the same space.

- Their atomic representations are very limited.

  e.g., a list of all possible concrete states.

- Intelligent agents need *knowledge about the world* in order to carry out reasoning for good decision making.

  - Represent states, actions, etc.
  - Incorporate new percepts.
  - Update internal representation of the world.
  - Deduce hidden properties of the world.
  - Deduce appropriate actions.

Knowledge Base

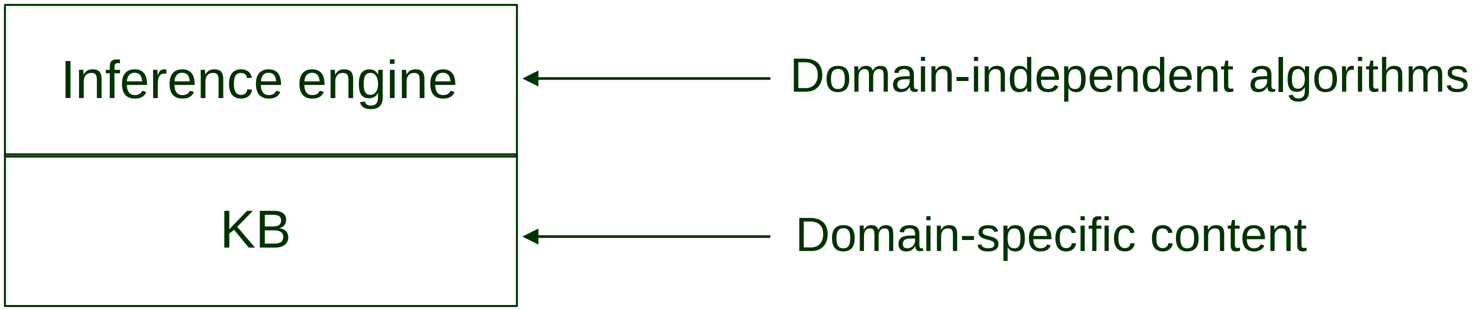
A *knowledge base (KB)* is a set of sentences that represent some assertion about the world.

An *axiom* is such a sentence that is taken to be true without being derived from other sentences.

Knowledge Base

A *knowledge base (KB)* is a set of sentences that represent some assertion about the world.

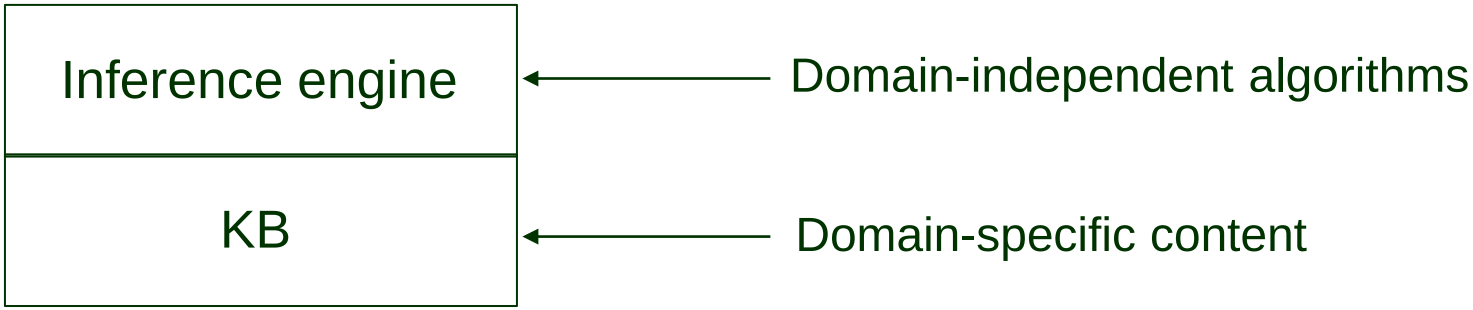
An *axiom* is such a sentence that is taken to be true without being derived from other sentences.



Knowledge Base

A *knowledge base (KB)* is a set of sentences that represent some assertion about the world.

An *axiom* is such a sentence that is taken to be true without being derived from other sentences.

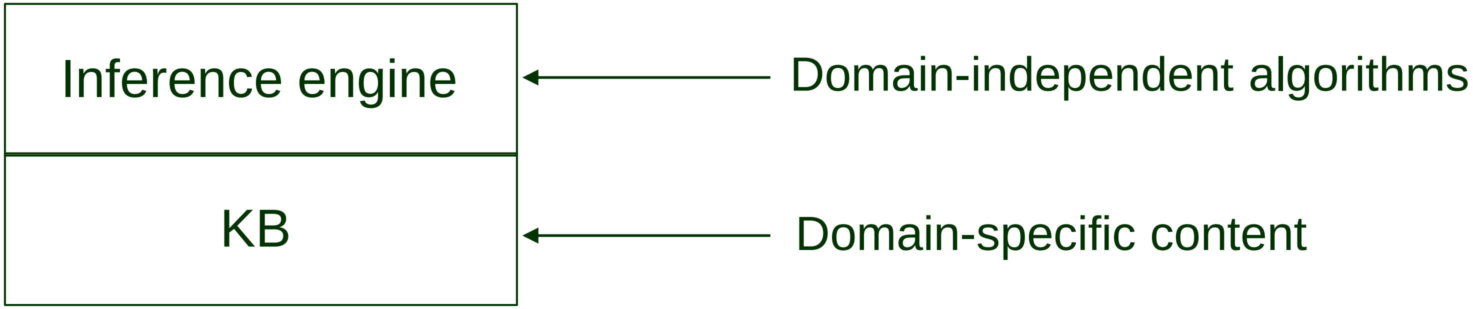


TELL: Add new sentences to the KB.

Knowledge Base

A *knowledge base (KB)* is a set of sentences that represent some assertion about the world.

An *axiom* is such a sentence that is taken to be true without being derived from other sentences.



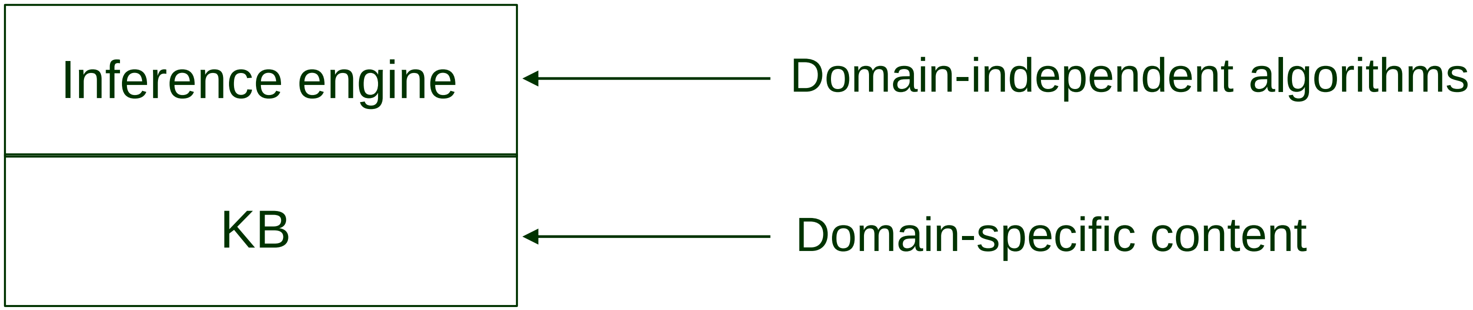
TELL: Add new sentences to the KB.

ASK: Query the KB.

Knowledge Base

A *knowledge base (KB)* is a set of sentences that represent some assertion about the world.

An *axiom* is such a sentence that is taken to be true without being derived from other sentences.

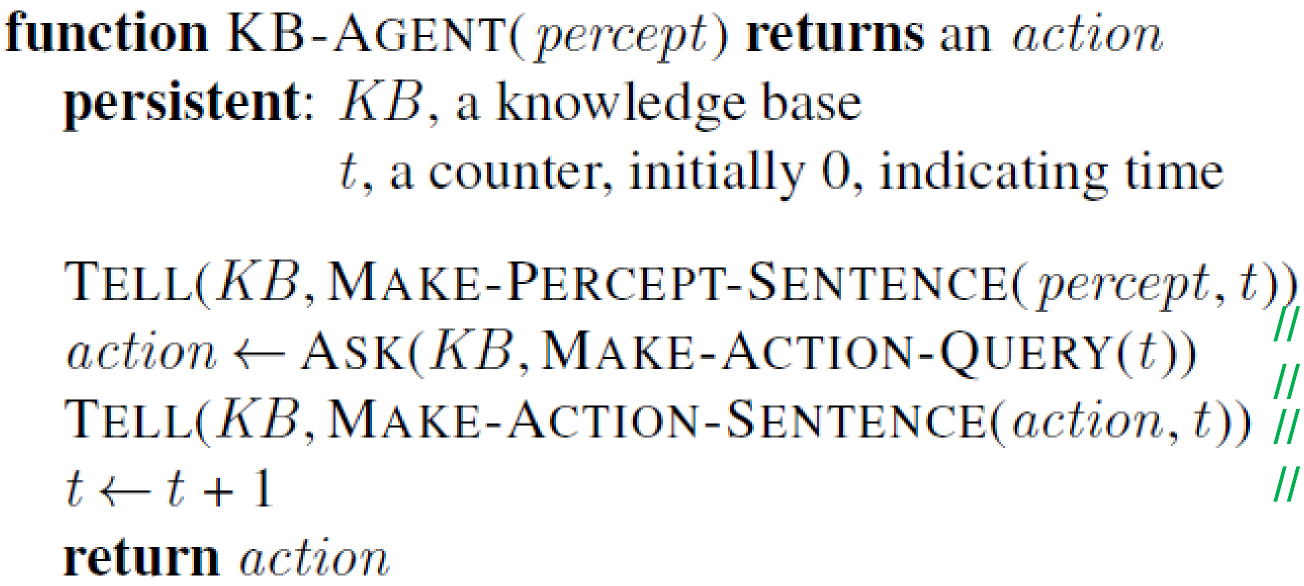


TELL: Add new sentences to the KB.

ASK: Query the KB.

Inference: Derive new sentences from old.

Generic Knowledge-Based Agent



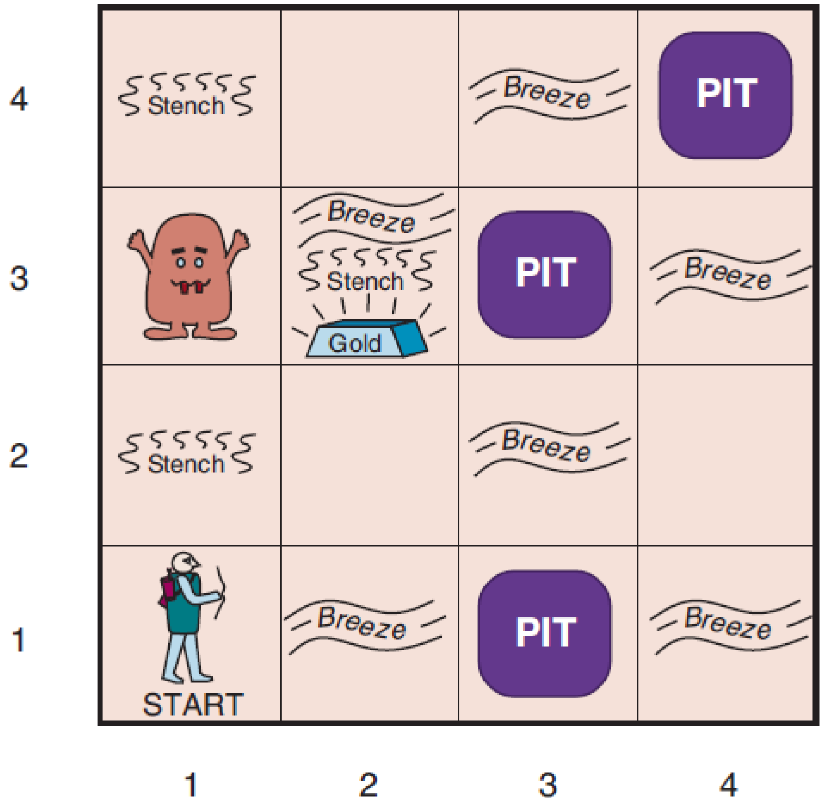
```
function KB-AGENT(percept) returns an action
    persistent: KB, a knowledge base
                t, a counter, initially 0, indicating time

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

// asks what action
// it should perform.
// tells what action
// was chosen

II. The Wumpus World

Cave consisting of connected rooms.

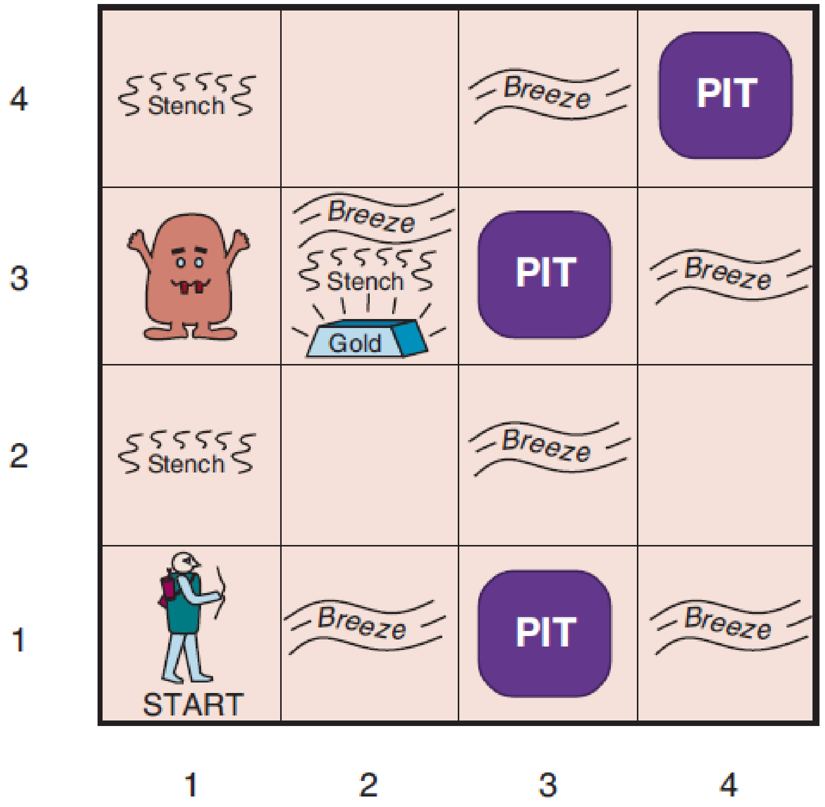
- Some rooms contain pits that will trap whoever enters them.
- The wumpus lurks in one room ready to eat whoever enters the room.
- The wumpus can be shot by the agent, who has only one arrow.
- A heap of gold is in a different room than where the wumpus lurks.



II. The Wumpus World

Cave consisting of connected rooms.

- Some rooms contain pits that will trap whoever enters them.
- The wumpus lurks in one room ready to eat whoever enters the room.
- The wumpus can be shot by the agent, who has only one arrow.
- A heap of gold is in a different room than where the wumpus lurks.

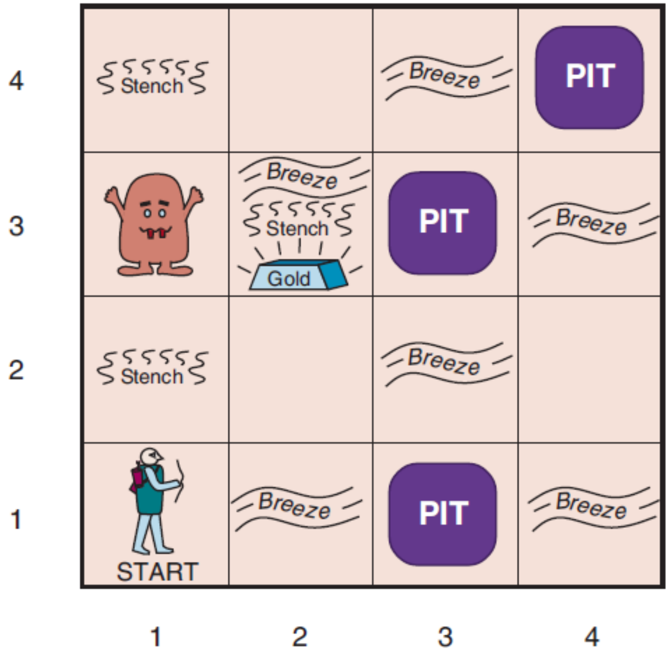


Goal: Find the gold and bring it back to the start without getting killed.

Task Environment

Performance measure

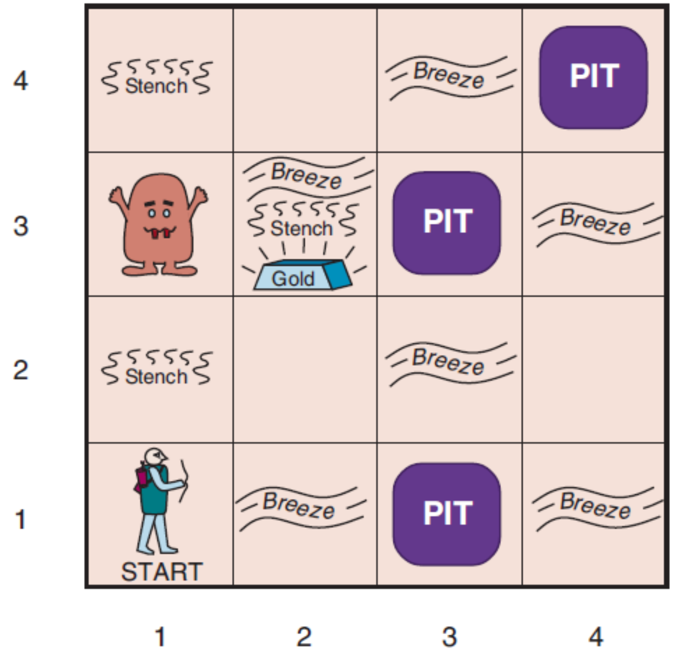
- +1000 (climbing out of the cave with the gold)
- –1000 (falling into a pit or being eaten by the wumpus)
- –1 (each action taken)
- –10 (using up the arrow)



Task Environment

Performance measure

- +1000 (climbing out of the cave with the gold)
- −1000 (falling into a pit or being eaten by the wumpus)
- −1 (each action taken)
- −10 (using up the arrow)



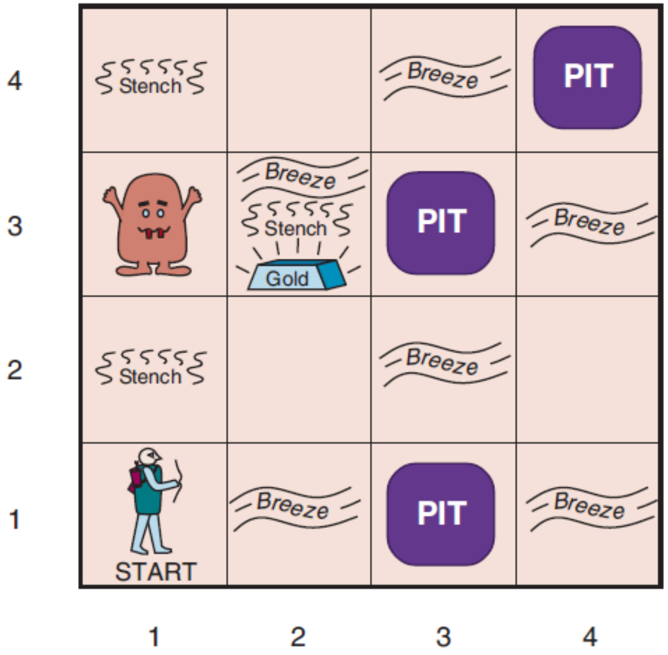
Environment

- 4×4 grid surrounded by walls
- $[1,1]$: the starting square for the agent, who faces east
- locations of the gold and the wumpus:
  - different from [1, 1]
  - otherwise randomly generated under uniform distribution
- 0.2 probability for a square other than $[1, 1]$ and without gold or wumpus to be pit

Actuators

Actuators:

1) *Forward*, *TurnLeft by* 90°, *TurnRight by* 90°
    - Death of the agent if it enters a square containing a pit or a live wumpus.
    - No movement if bumping into a wall.
2) *Grab*
    - Picks up the gold if it in the same square as the agent.
3) *Shoot*
    - Fire an arrow in the direction the agent is facing.
    - The arrow continues until hitting the wumpus (who gets killed consequently) or a wall.
4) *Climb*
    - Climb out of the cave if at $[1, 1]$.



Sensors

5 Sensors, each providing one bit of information:

1) *Stench*

- in the squares directly (not diagonally) adjacent to the wumpus

2) *Breeze*

- in the squares directly (not diagonally) adjacent to a pit

3) *Glitter*

- in the square where the gold is

4) *Bump*

- when the agent walks into a wal

5) *Scream*

- when the wumpus is killed

Sensors

5 Sensors, each providing one bit of information:

1) *Stench*

- in the squares directly (not diagonally) adjacent to the wumpus

2) *Breeze*

- in the squares directly (not diagonally) adjacent to a pit

3) *Glitter*

- in the square where the gold is

4) *Bump*

- when the agent walks into a wal

5) *Scream*

- when the wumpus is killed

Percepts in the form of a 5-vector:

e.g., [*Stench, Breeze, None, None, None*]

Characteristics of WW

- Deterministic, discrete, static, and single-agent

Characteristics of WW

- Deterministic, discrete, static, and single-agent

Outcome specified.

Characteristics of WW

- Deterministic, discrete, static, and single-agent

Outcome specified.

The wumpus does not move.

Characteristics of WW

- Deterministic, discrete, static, and single-agent

Outcome specified.

The wumpus does not move.

- Partially observable

Locations of the pits and the wumpus are unknown.

Characteristics of WW

- Deterministic, discrete, static, and single-agent

Outcome specified.

The wumpus does not move.

- Partially observable

Locations of the pits and the wumpus are unknown.

Challenge: The agent needs to use logical reasoning to overcome its initial lack of knowledge about the environment’s configuration.

Characteristics of WW

- Deterministic, discrete, static, and single-agent

Outcome specified.

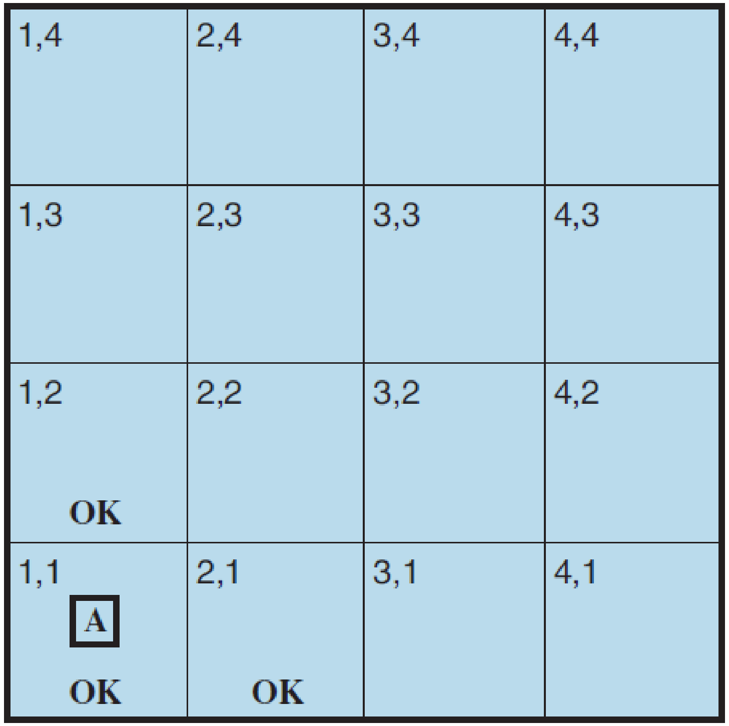
The wumpus does not move.

- Partially observable

Locations of the pits and the wumpus are unknown.

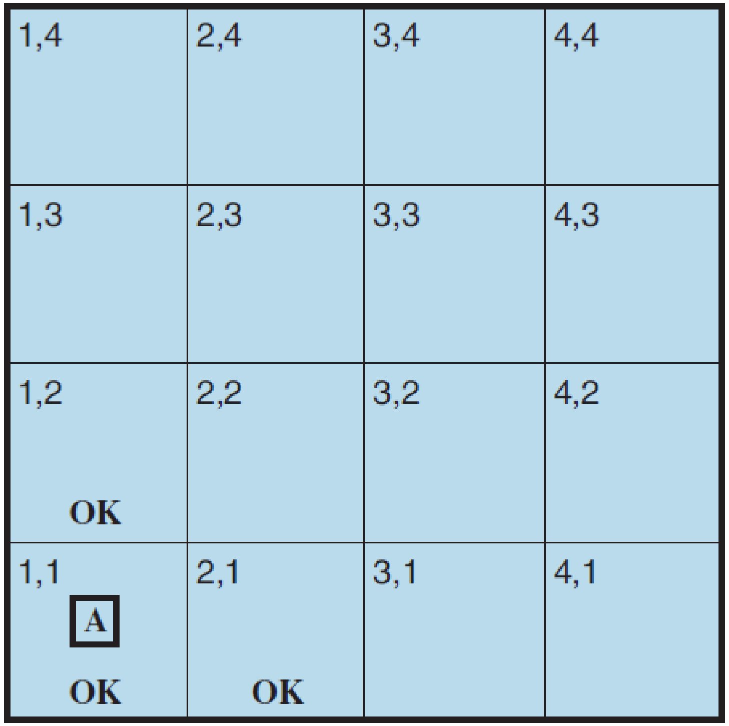
Challenge: The agent needs to use logical reasoning to overcome its initial lack of knowledge about the environment's configuration.

Solution by a Knowledge-Based Agent



A	= *Agent*
B	= *Breeze*
G	= *Glitter, Gold*
OK	= *Safe square*
P	= *Pit*
S	= *Stench*
V	= *Visited*
W	= *Wumpus*

Solution by a Knowledge-Based Agent

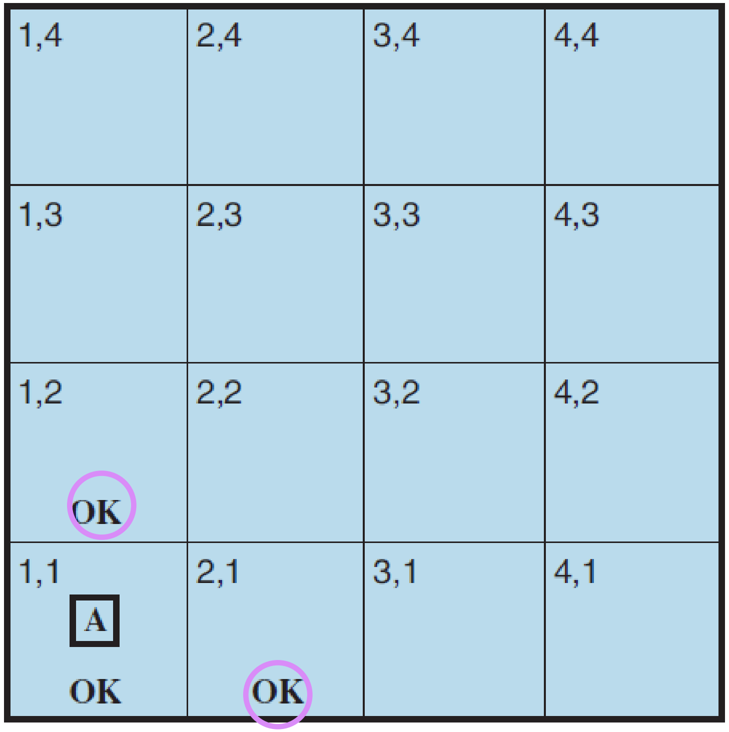


1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2	3,2	4,2
1,1 A OK	2,1 OK	3,1	4,1

A	= *Agent*
B	= *Breeze*
G	= *Glitter, Gold*
OK	= *Safe square*
P	= *Pit*
S	= *Stench*
V	= *Visited*
W	= *Wumpus*

Percept: [*None, None, None, None, None*]

Solution by a Knowledge-Based Agent



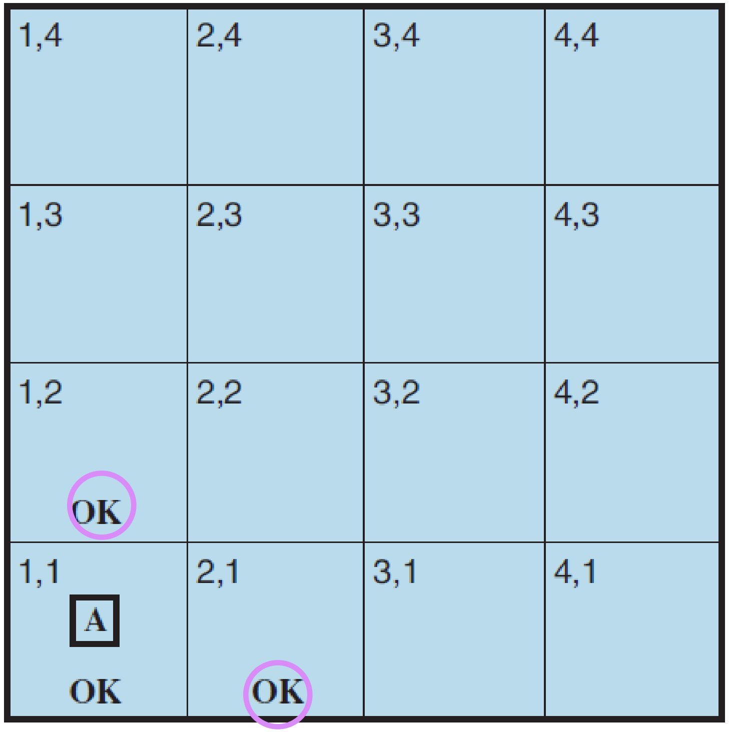
A	= *Agent*
B	= *Breeze*
G	= *Glitter, Gold*
OK	= *Safe square*
P	= *Pit*
S	= *Stench*
V	= *Visited*
W	= *Wumpus*

Percept: [*None, None, None, None, None*]

⇩

[1,2] and [2, 1] are free of dangers.

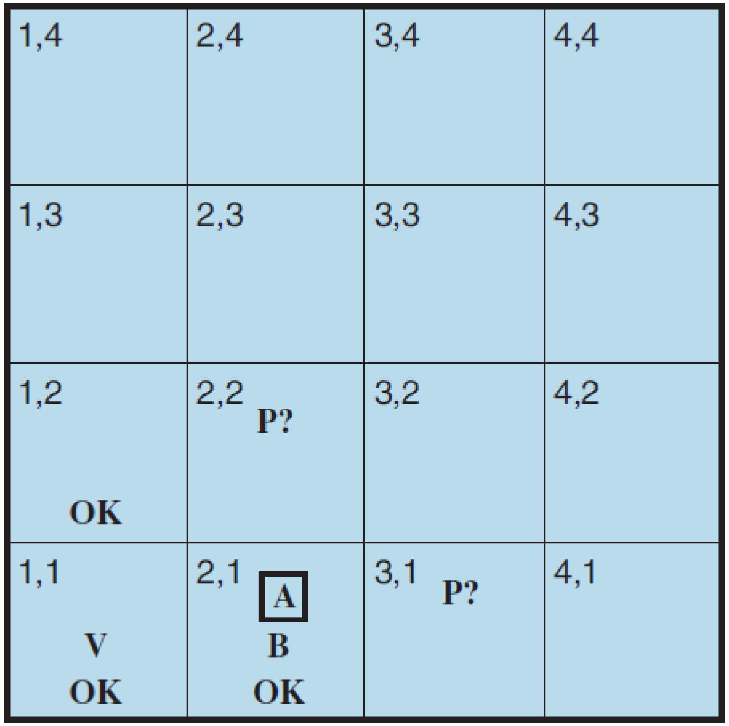
Solution by a Knowledge-Based Agent



1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2	3,2	4,2
1,1 A OK	2,1 OK	3,1	4,1

A	= *Agent*
B	= *Breeze*
G	= *Glitter, Gold*
OK	= *Safe square*
P	= *Pit*
S	= *Stench*
V	= *Visited*
W	= *Wumpus*

Forward ⇨



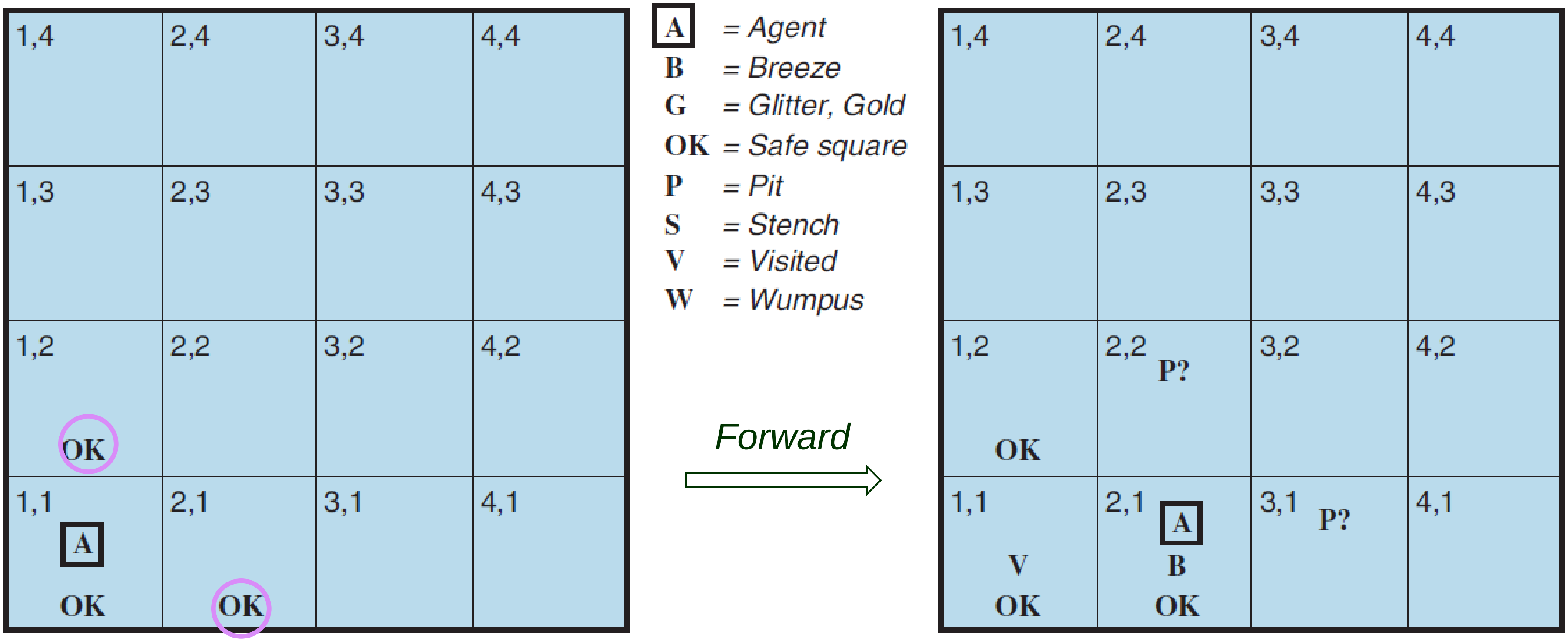
1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 A B OK	3,1 P?	4,1

Percept: [*None, None, None, None, None*]

⇩

[1,2] and [2, 1] are free of dangers.

Solution by a Knowledge-Based Agent



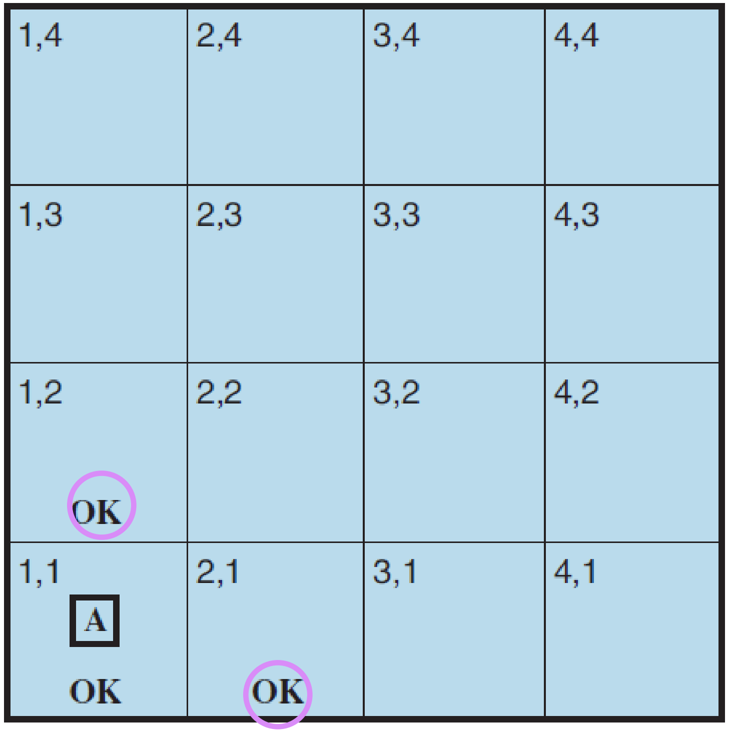
Percept: [*None, None, None, None, None*]

[*None, Breeze, None, None, None*]

⇩

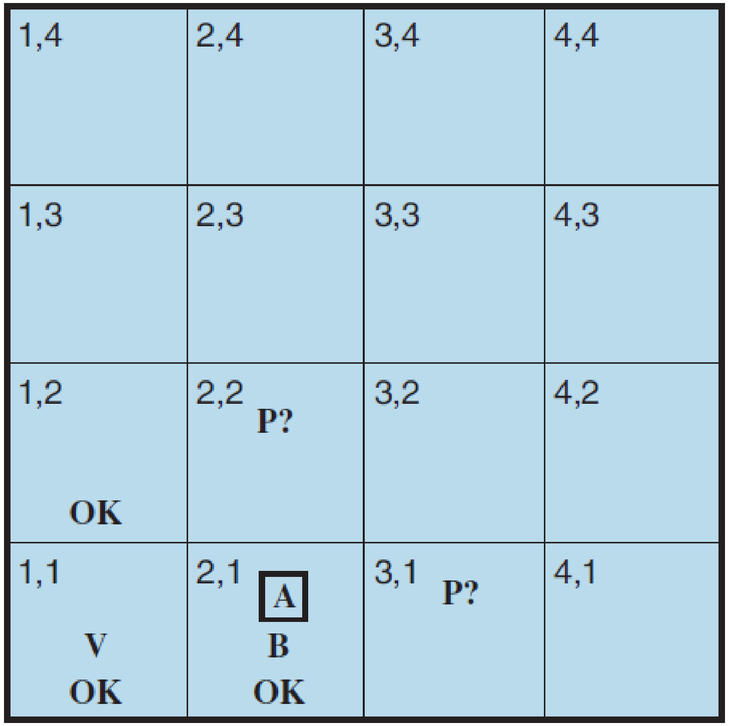
[1,2] and [2, 1] are free of dangers.

Solution by a Knowledge-Based Agent



A	= *Agent*
B	= *Breeze*
G	= *Glitter, Gold*
OK	= *Safe square*
P	= *Pit*
S	= *Stench*
V	= *Visited*
W	= *Wumpus*

Forward ⇒



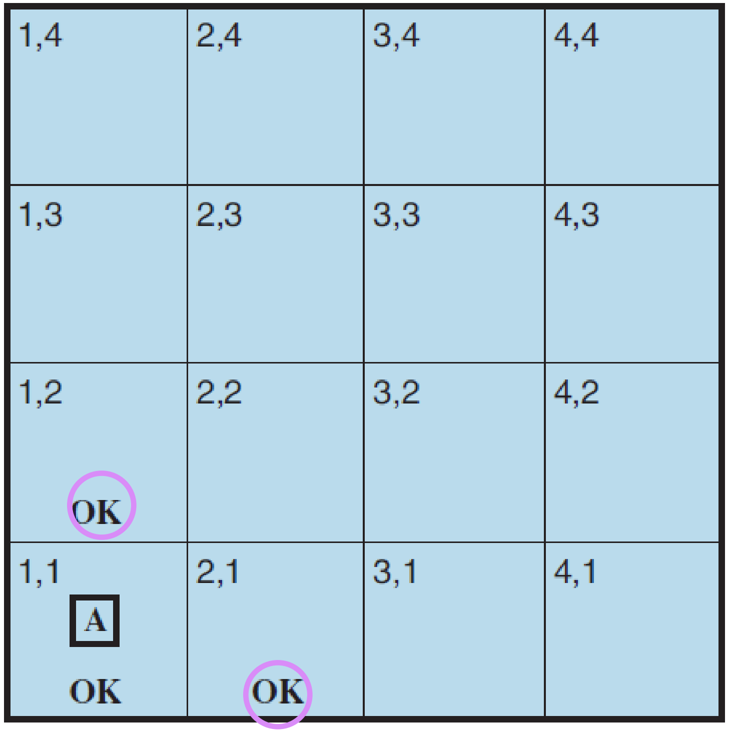
Percept: [*None, None, None, None, None*]

⇓

[1,2] and [2, 1] are free of dangers.

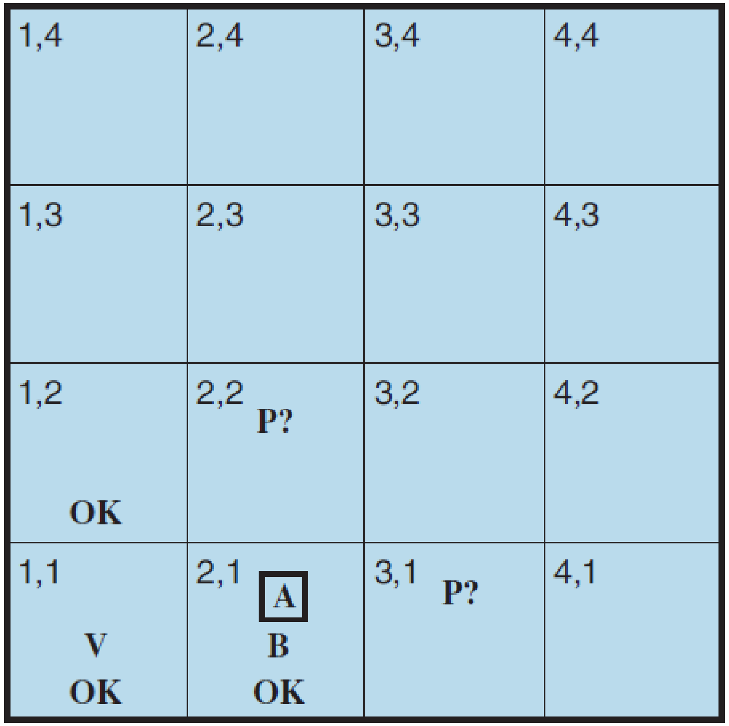
[*None, Breeze, None, None, None*]

Solution by a Knowledge-Based Agent



A	= *Agent*
B	= *Breeze*
G	= *Glitter, Gold*
OK	= *Safe square*
P	= *Pit*
S	= *Stench*
V	= *Visited*
W	= *Wumpus*

Forward



Percept: *[None, None, None, None, None]*

⇩

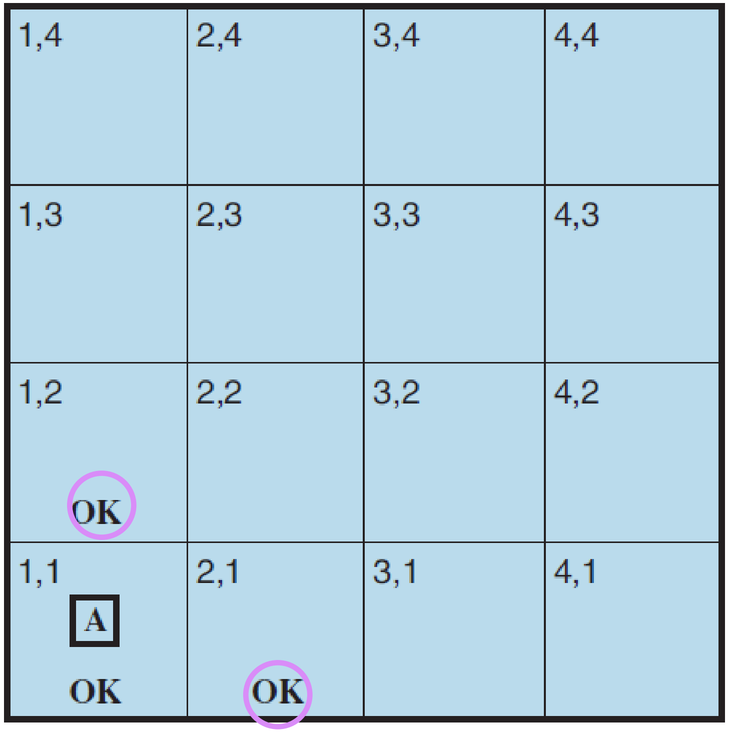
[1,2] and [2, 1] are free of dangers.

[None, Breeze, None, None, None]

⇩

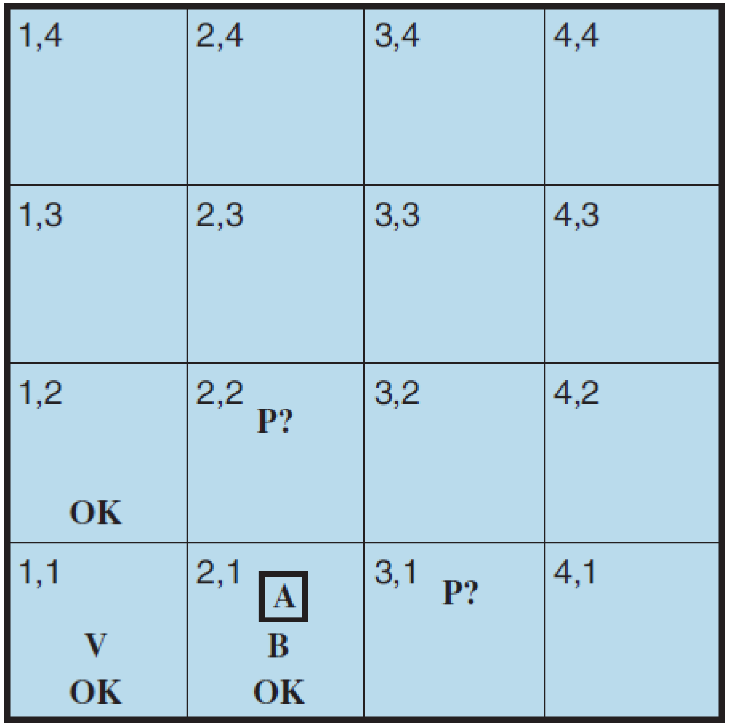
A pit in [1,1],,[2,2] , or [3, 1].

Solution by a Knowledge-Based Agent



A	= *Agent*
B	= *Breeze*
G	= *Glitter, Gold*
OK	= *Safe square*
P	= *Pit*
S	= *Stench*
V	= *Visited*
W	= *Wumpus*

Forward ⇒



Percept: [*None, None, None, None, None*]

⇩

[1,2] and [2, 1] are free of dangers.

[*None, Breeze, None, None, None*]

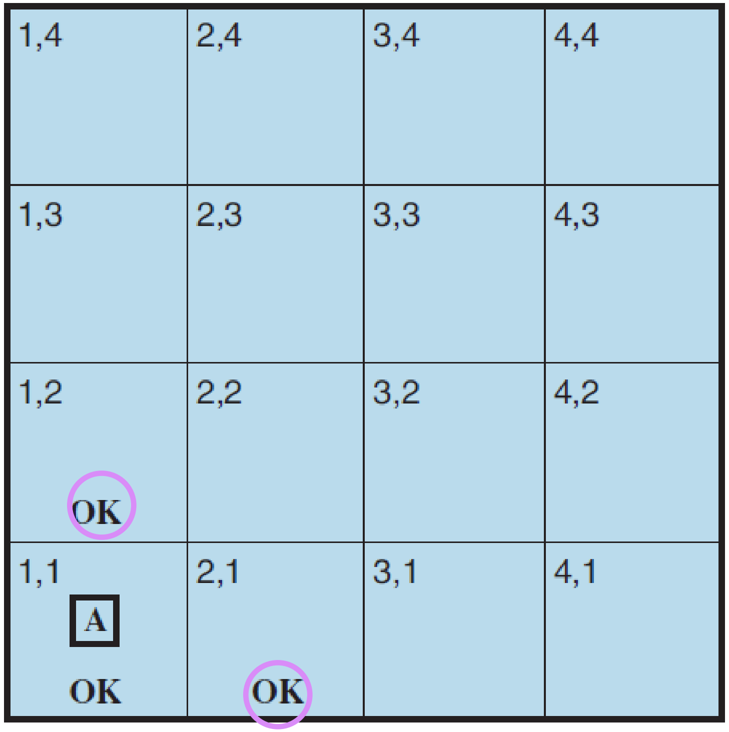
⇩

A pit in [1,1],,[2,2] , or [3, 1].

⇩ [1,1] has just been visited.

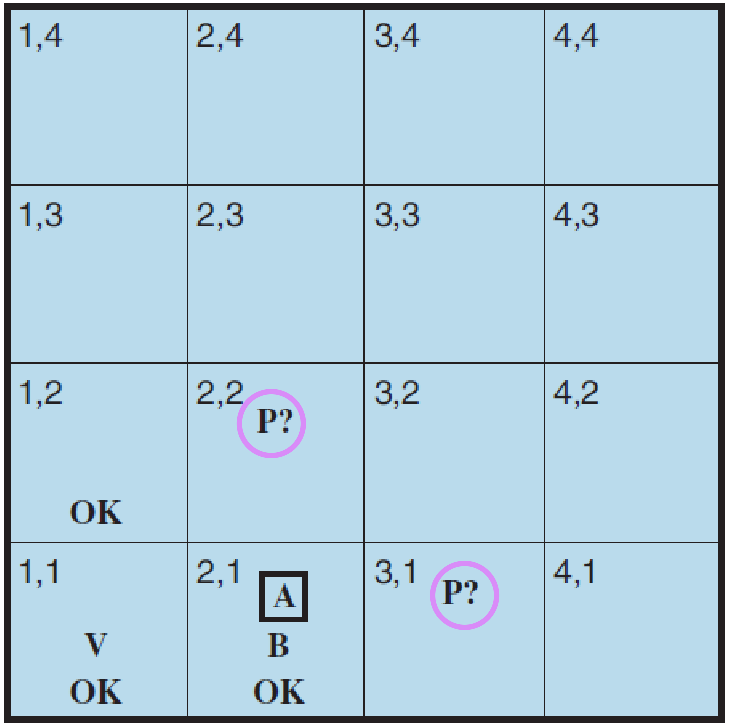
A pit in [2,2] or [3, 1].

Solution by a Knowledge-Based Agent



A	= *Agent*
B	= *Breeze*
G	= *Glitter, Gold*
OK	= *Safe square*
P	= *Pit*
S	= *Stench*
V	= *Visited*
W	= *Wumpus*

Forward ⇒



Percept: [*None, None, None, None, None*]

⇓

[1,2] and [2, 1] are free of dangers.

[*None, Breeze, None, None, None*]

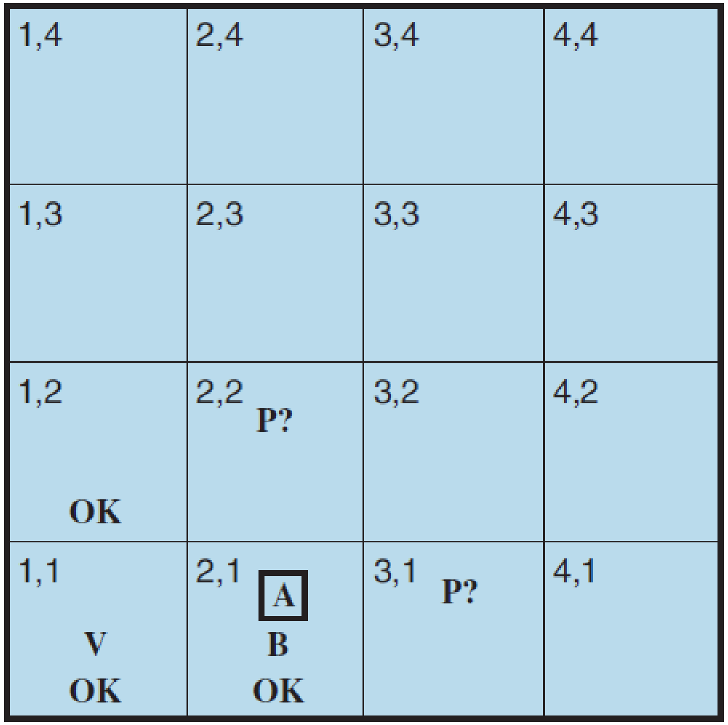
⇓

A pit in [1,1],,[2,2] , or [3, 1].

⇓ [1,1] has just been visited.

A pit in [2,2] or [3, 1].

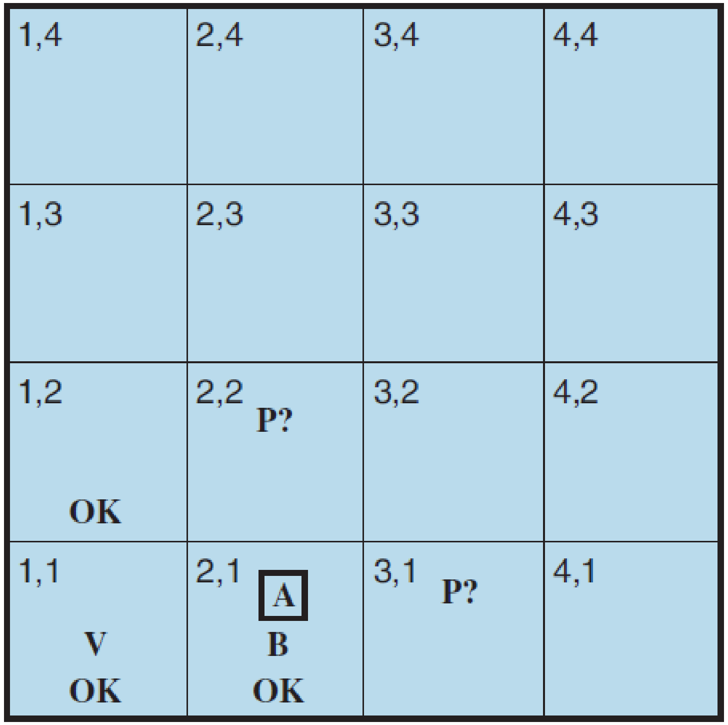
Next Step



1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 $\boxed{A}$ B OK	3,1 P?	4,1

Only one unexplored square [1,2] is OK.

Next Step

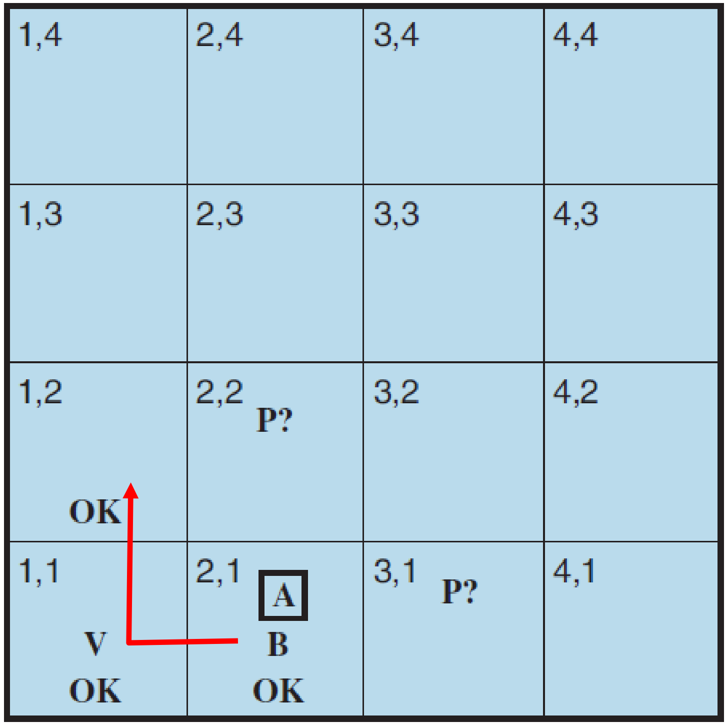


1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 $\boxed{A}$ B OK	3,1 P?	4,1

Only one unexplored square [1,2] is OK.

Be prudent: Turn around, go back to [1,1] and move onto [1, 2].

Next Step

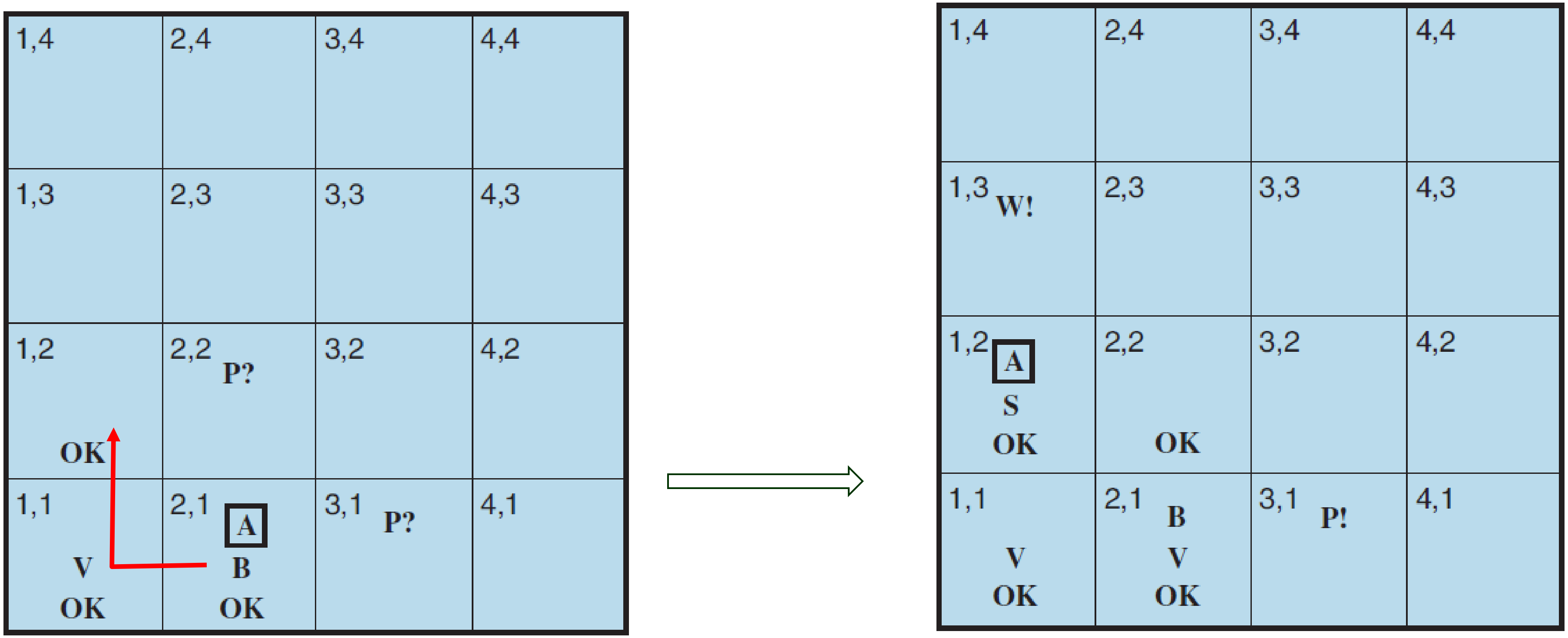


1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 A B OK	3,1 P?	4,1

Only one unexplored square [1,2] is OK.

Be prudent: Turn around, go back to [1,1]
and move onto [1, 2].

Next Step



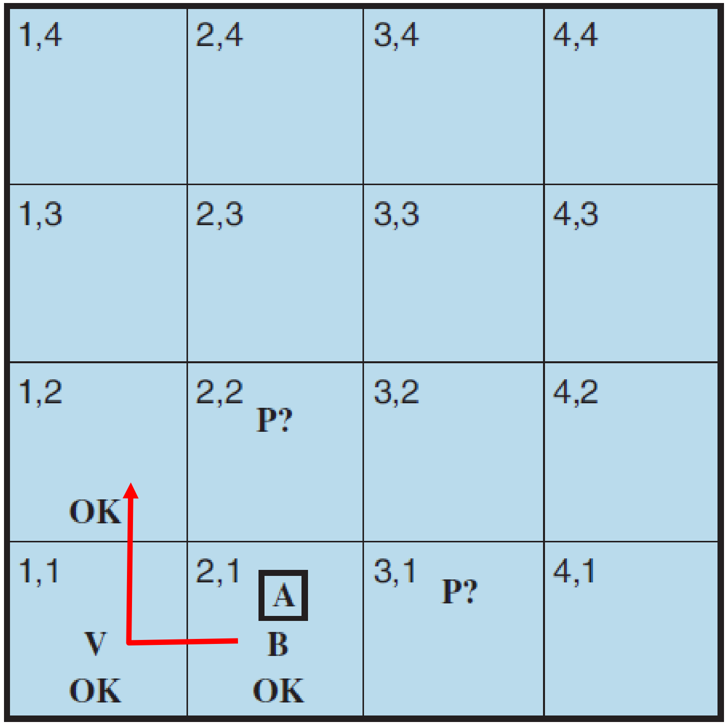
1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 A B OK	3,1 P?	4,1

1,4	2,4	3,4	4,4
1,3 W!	2,3	3,3	4,3
1,2 A S OK	2,2 OK	3,2	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1

Only one unexplored square [1,2] is OK.

Be prudent: Turn around, go back to [1,1]
and move onto [1, 2].

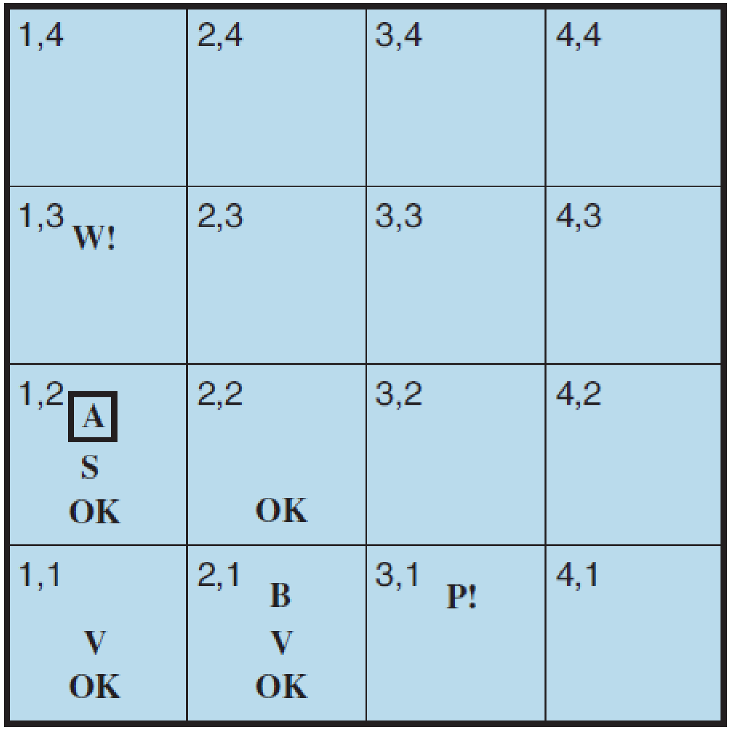
Next Step



1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 A B OK	3,1 P?	4,1

Only one unexplored square [1,2] is OK.

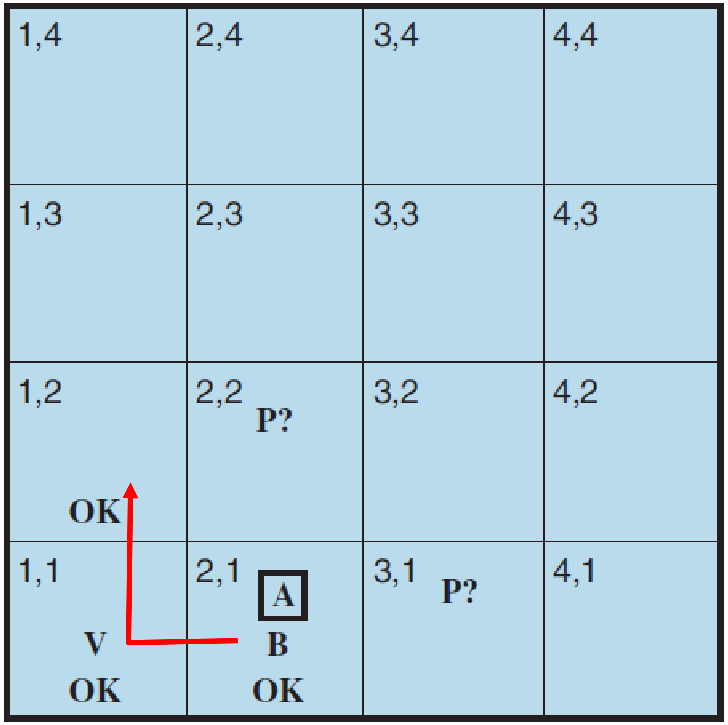
Be prudent: Turn around, go back to [1,1] and move onto [1, 2].



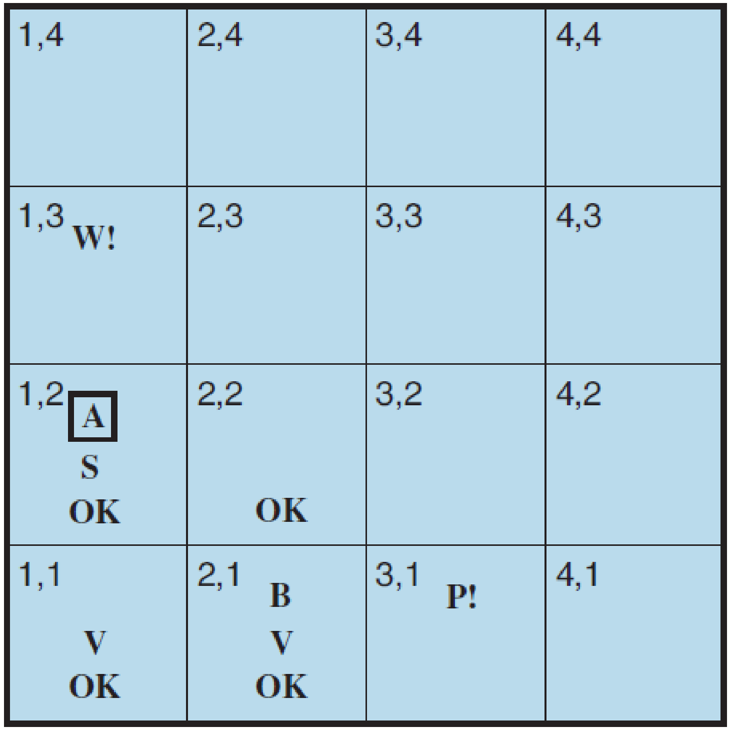
1,4	2,4	3,4	4,4
1,3 W!	2,3	3,3	4,3
1,2 A S OK	2,2 OK	3,2	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1

[Stench, None, None, None, None]

Next Step



1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 A B OK	3,1 P?	4,1



1,4	2,4	3,4	4,4
1,3 W!	2,3	3,3	4,3
1,2 A S OK	2,2 OK	3,2	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1

Only one unexplored square [1,2] is OK.

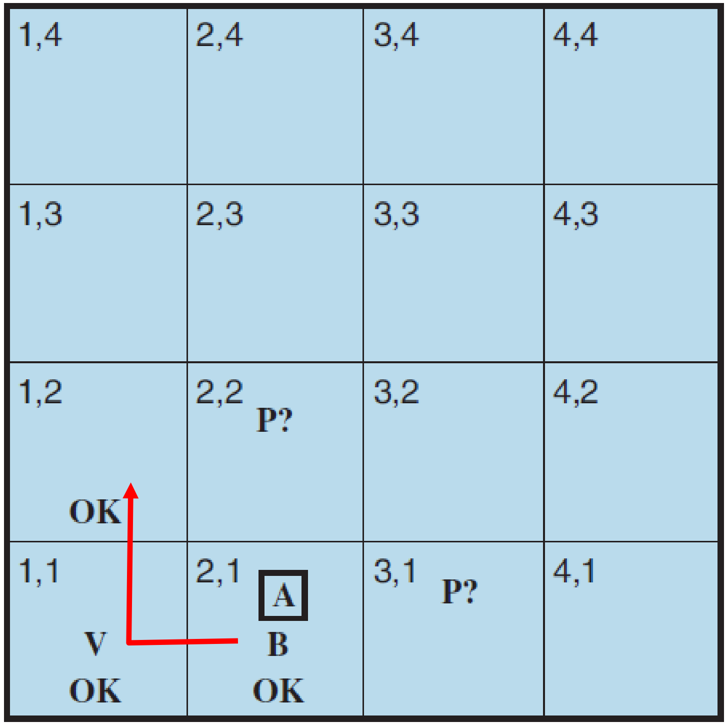
Be prudent: Turn around, go back to [1,1] and move onto [1, 2].

[Stench, None, None, None, None]

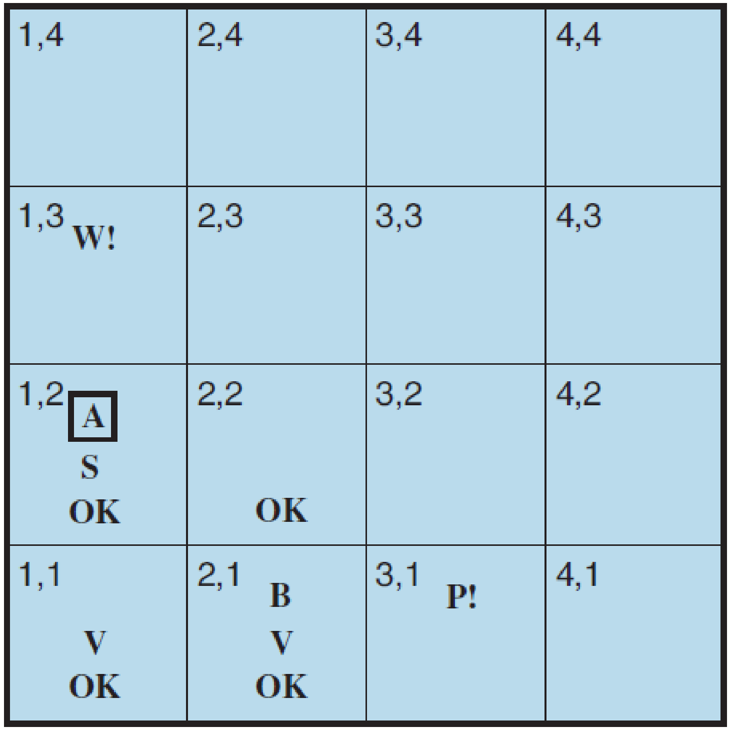
⇓

The wumpus is in [1,1] , [2,2] , or [1, 3].

Next Step



1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 A B OK	3,1 P?	4,1



1,4	2,4	3,4	4,4
1,3 W!	2,3	3,3	4,3
1,2 A S OK	2,2 OK	3,2	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1

Only one unexplored square [1,2] is OK.

Be prudent: Turn around, go back to [1,1] and move onto [1, 2].

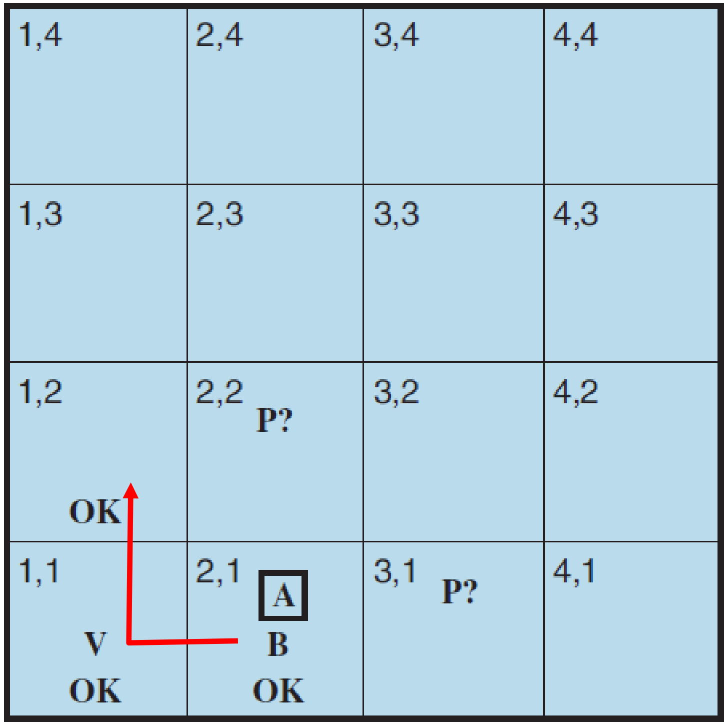
[Stench, None, None, None, None]

⇓

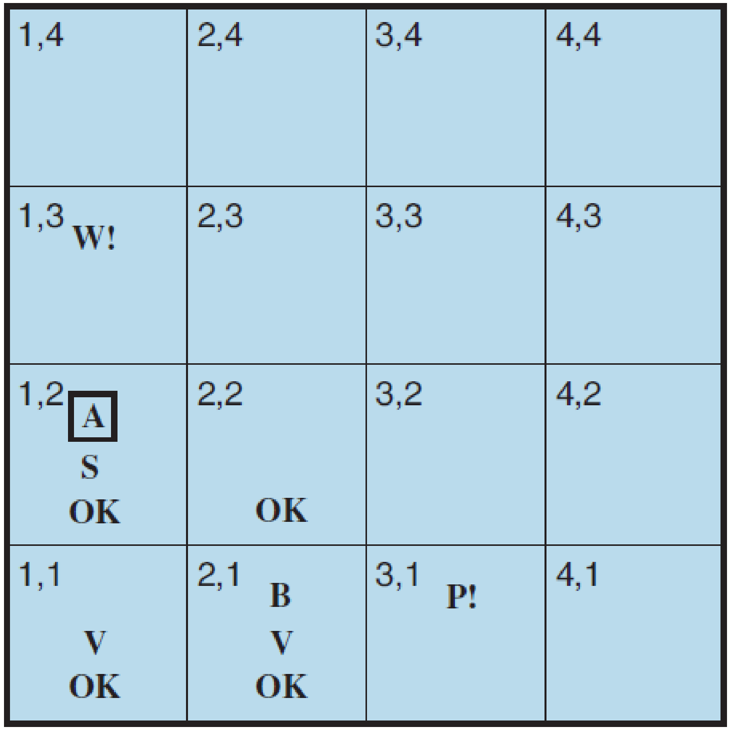
The wumpus is in [1,1] , [2,2] , or [1, 3].

[1,1] is OK
[2,2] is impossible because no stench was detected at [2, 1].

Next Step



1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 A B OK	3,1 P?	4,1



1,4	2,4	3,4	4,4
1,3 W!	2,3	3,3	4,3
1,2 A S OK	2,2 OK	3,2	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1

Only one unexplored square [1,2] is OK.

Be prudent: Turn around, go back to [1,1] and move onto [1, 2].

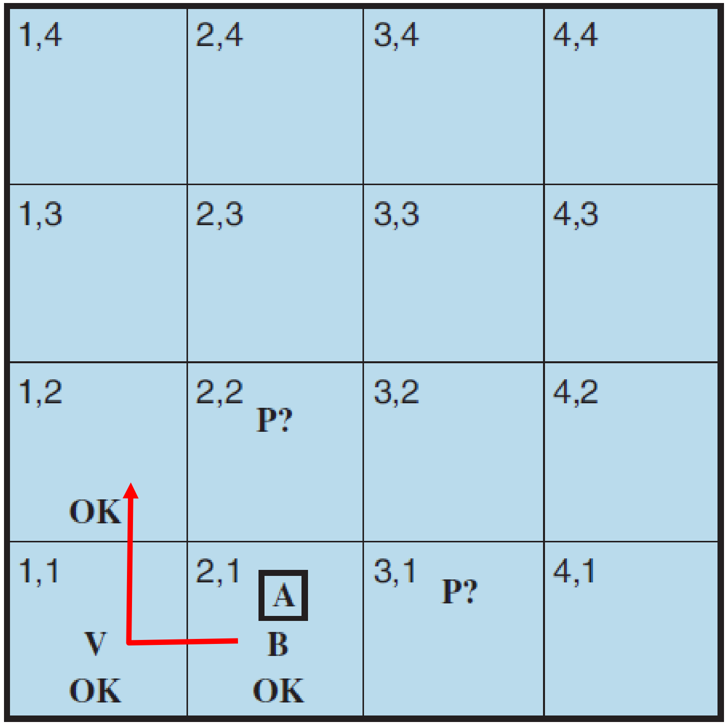
[Stench, None, None, None, None]

⇩

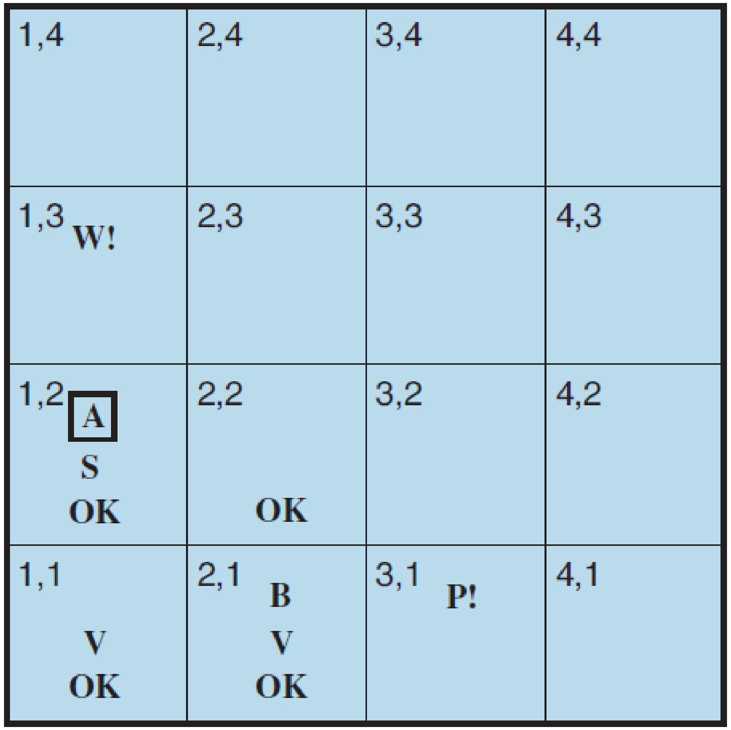
The wumpus is in [1,1] , [2,2] , or [1, 3].

[1,1] is OK
[2,2] is impossible because no stench was detected at [2, 1].

Next Step



1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 A B OK	3,1 P?	4,1



1,4	2,4	3,4	4,4
1,3 W!	2,3	3,3	4,3
1,2 A S OK	2,2 OK	3,2	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1

Only one unexplored square [1,2] is OK.

Be prudent: Turn around, go back to [1,1] and move onto [1, 2].

[Stench, None, None, None, None]

⇓

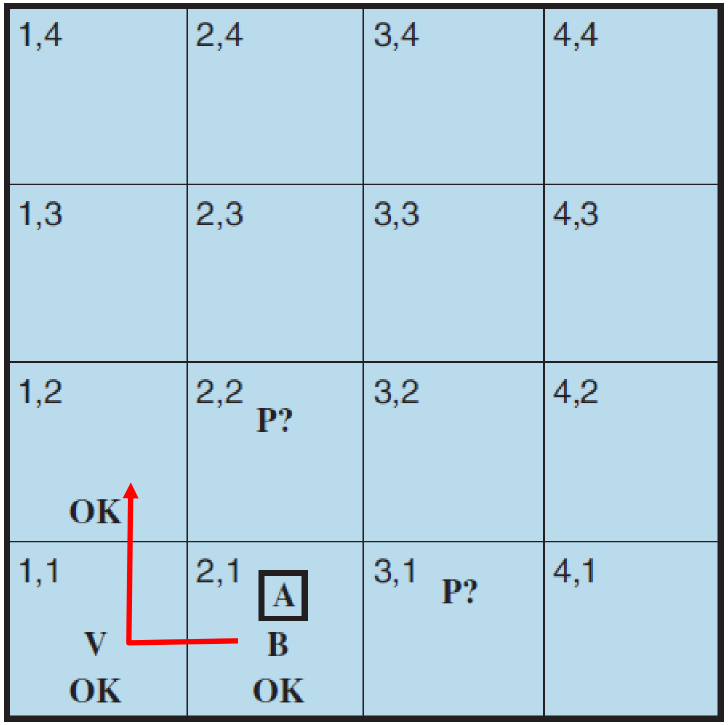
The wumpus is in [1,1] , [2,2] , or [1, 3].

⇓

[1,1] is OK
[2,2] is impossible because no stench was detected at [2, 1].

The wumpus is in [1, 3].

Next Step

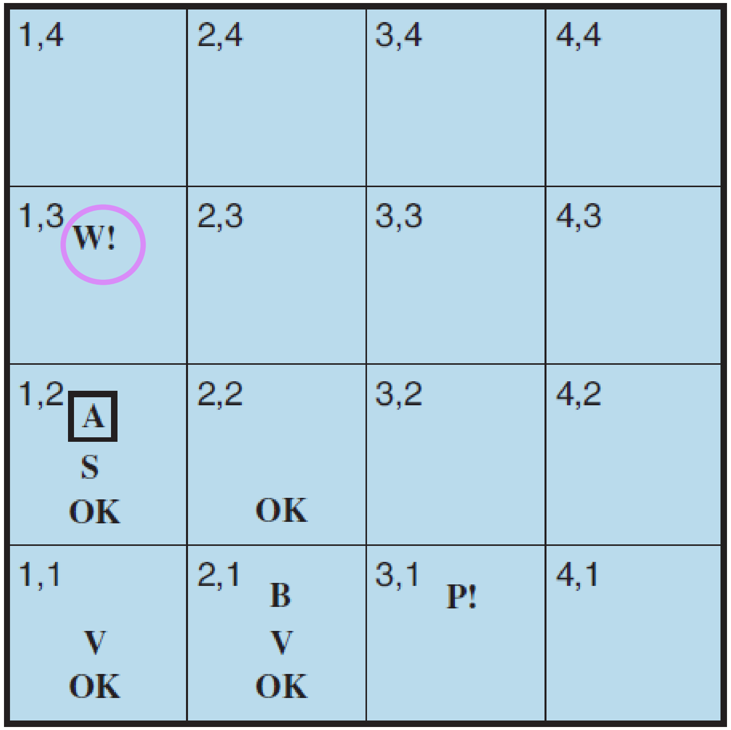


1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 A B OK	3,1 P?	4,1

Only one unexplored square [1,2] is OK.

Be prudent: Turn around, go back to [1,1] and move onto [1, 2].

⇒



1,4	2,4	3,4	4,4
1,3 W!	2,3	3,3	4,3
1,2 A S OK	2,2 OK	3,2	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1

[Stench, None, None, None, None]

⇓

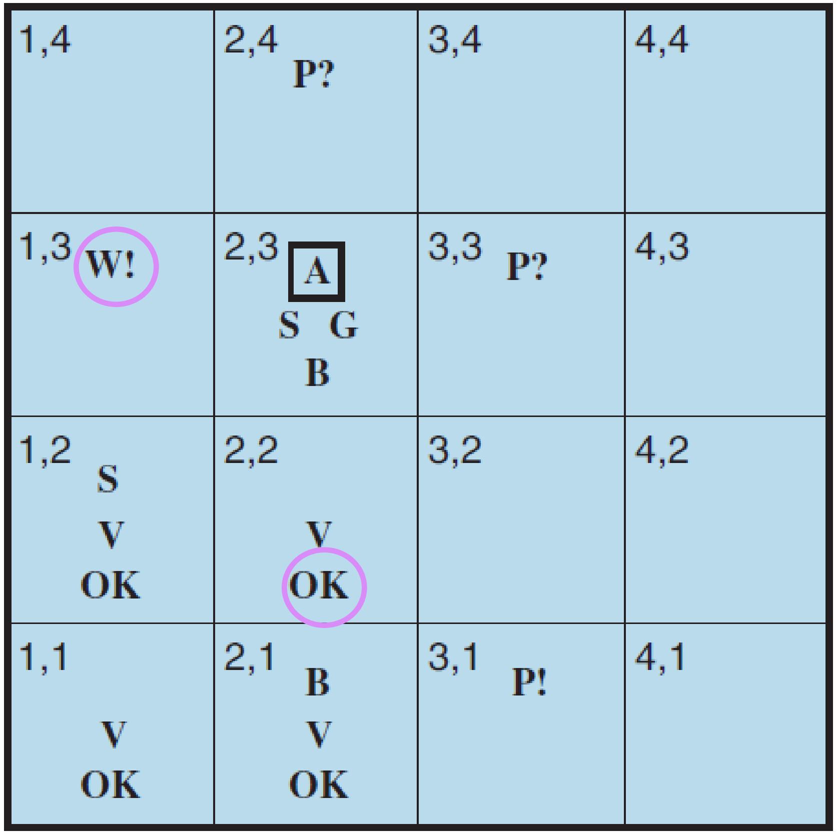
The wumpus is in [1,1] , [2,2] , or [1, 3].

⇓

[1,1] is OK
[2,2] is impossible because no stench was detected at [2, 1].

The wumpus is in [1, 3].

More Inference



[*Stench*, *None*, *None*, *None*, *None*]

The wumpus is in [1, 3].

More Inference

1,4	2,4 P?	3,4	4,4
1,3 W!	2,3 A S G B	3,3 P?	4,3
1,2 S V OK	2,2 V OK	3,2	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1

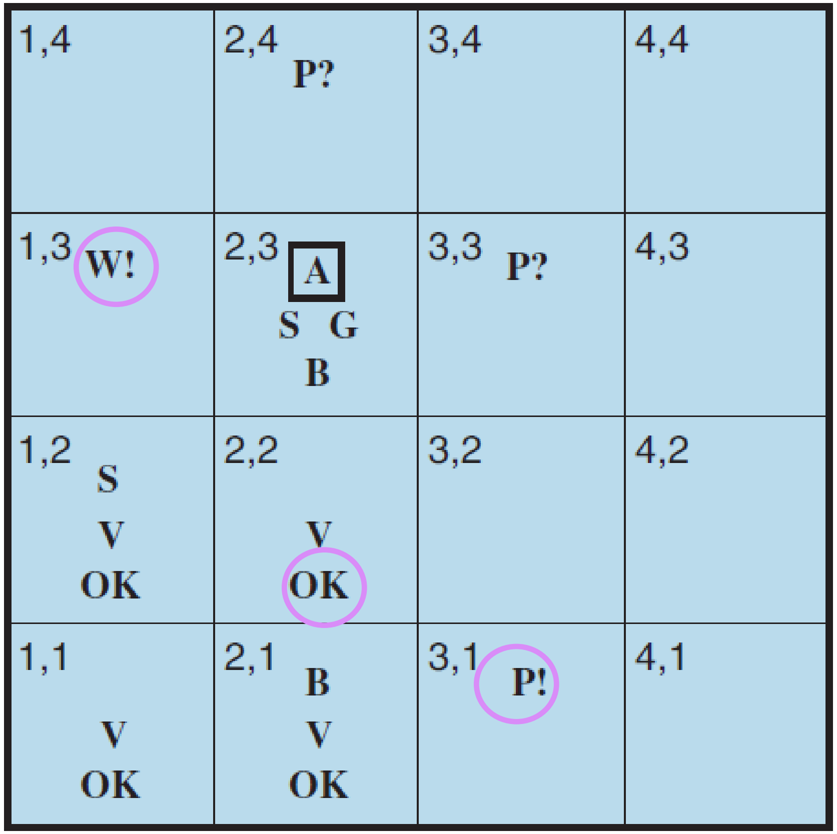
[*Stench, None, None, None, None*]

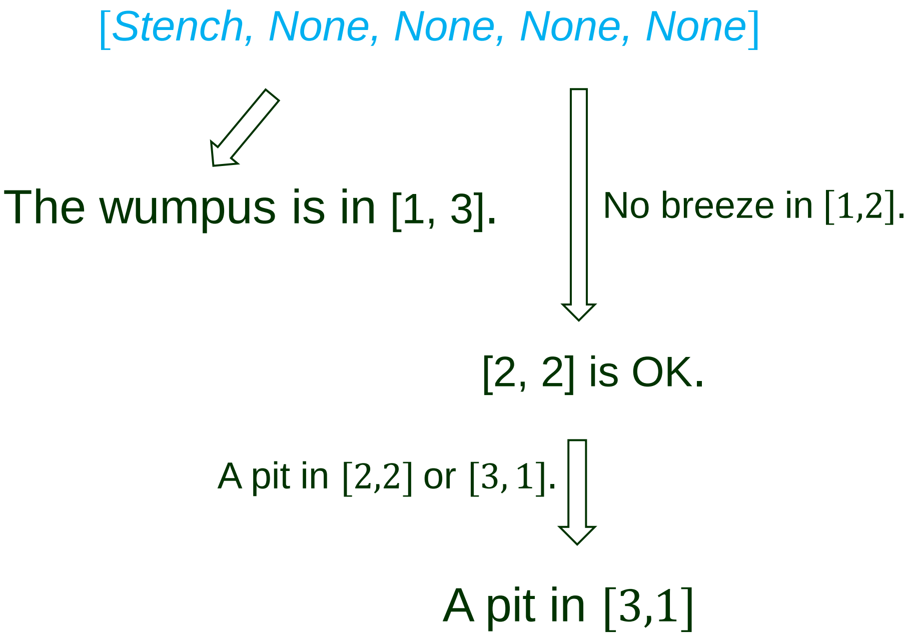
The wumpus is in [1, 3].

No breeze in [1,2].

[2, 2] is OK.

More Inference





[Stench, None, None, None, None]

The wumpus is in [1, 3].

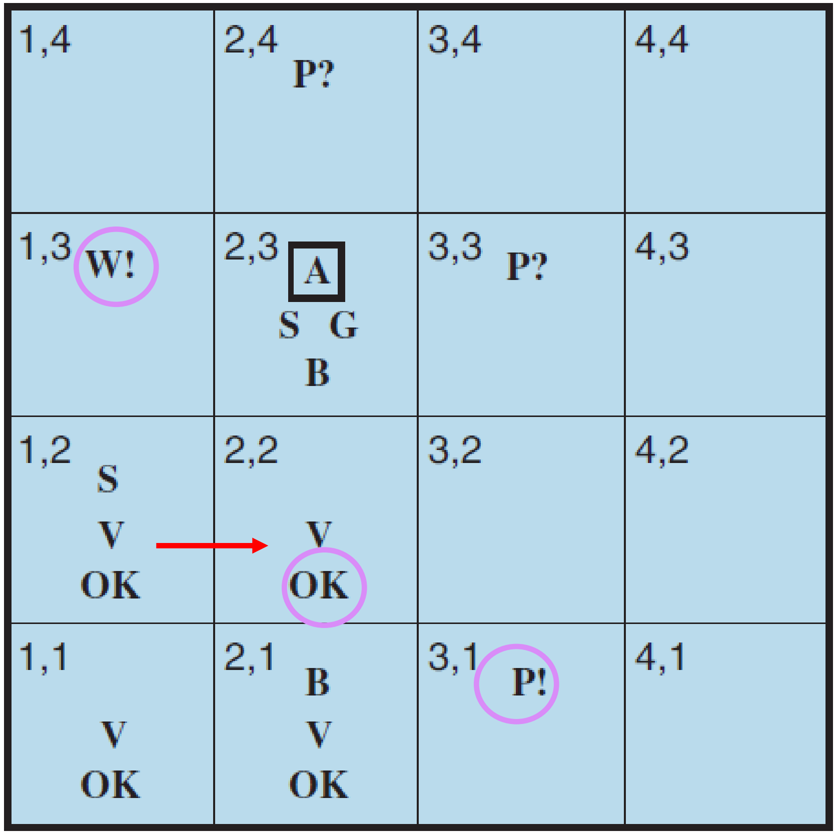
No breeze in [1,2].

[2, 2] is OK.

A pit in [2,2] or [3, 1].

A pit in [3,1]

More Inference



1,4	2,4 P?	3,4	4,4
1,3 W!	2,3 A S G B	3,3 P?	4,3
1,2 S V OK	2,2 V OK	3,2	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1

[*Stench, None, None, None, None*]

The wumpus is in [1, 3].

No breeze in [1,2].

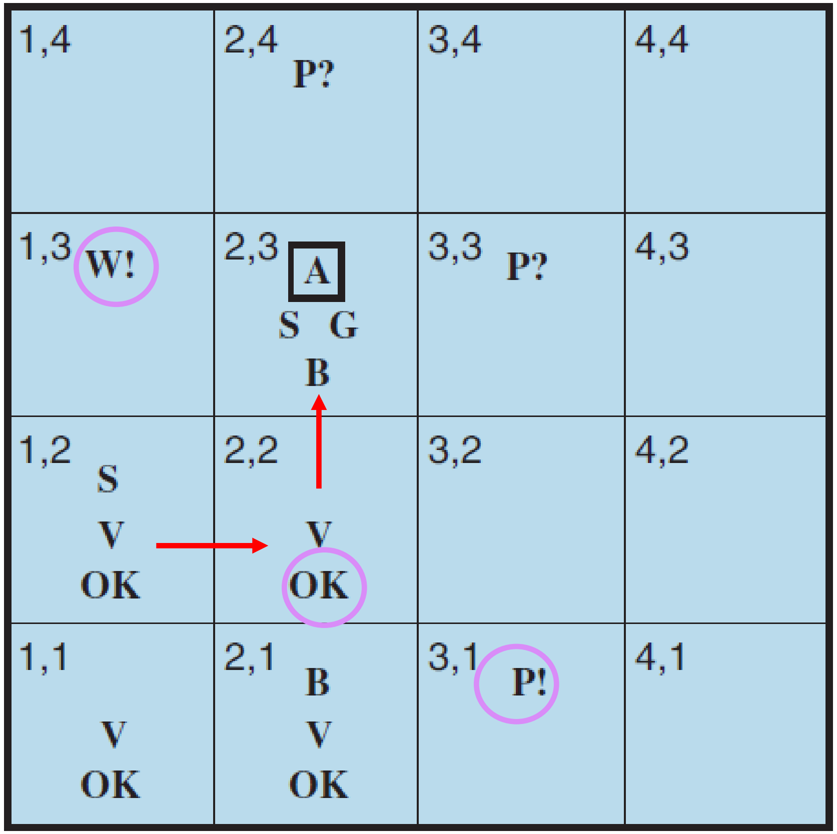
[2, 2] is OK.

A pit in [2,2] or [3, 1].

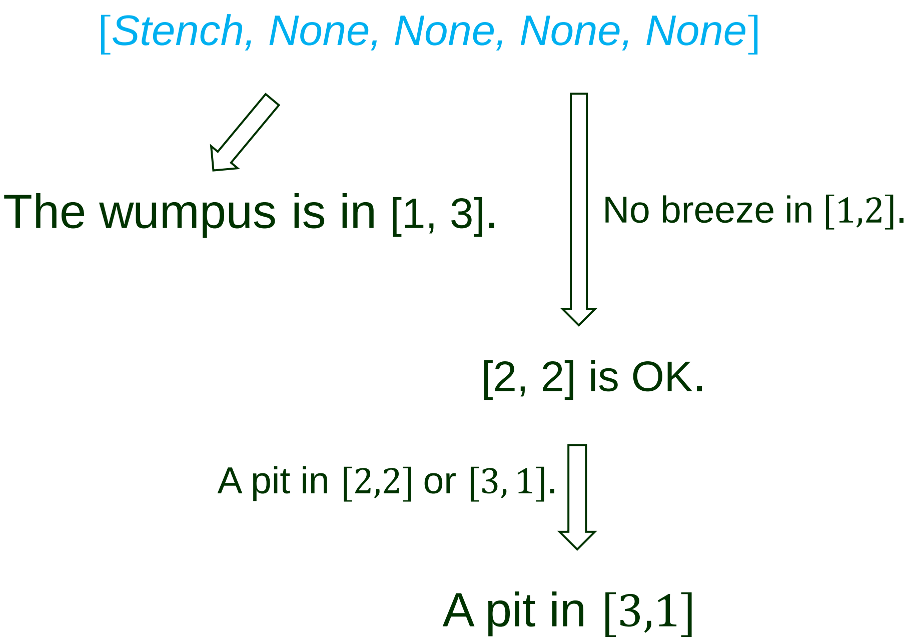
A pit in [3,1]

- Moves to (2, 2).

More Inference



1,4	2,4 P?	3,4	4,4
1,3 W!	2,3 A S G B	3,3 P?	4,3
1,2 S V OK	2,2 V OK	3,2	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1



[Stench, None, None, None, None]

The wumpus is in [1, 3].

No breeze in [1,2].

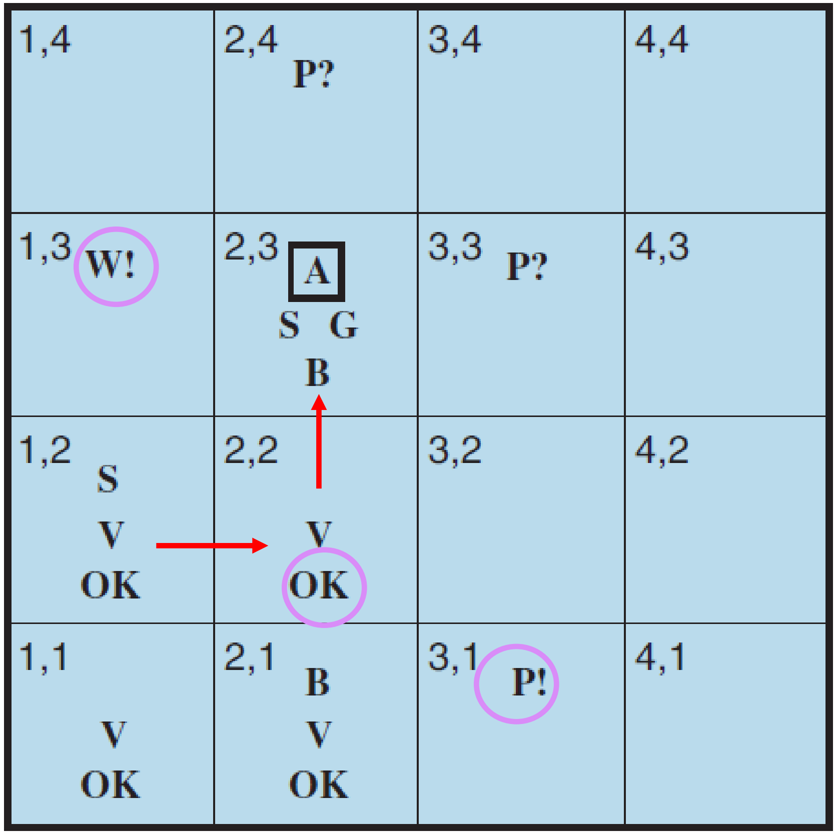
[2, 2] is OK.

A pit in [2,2] or [3, 1].

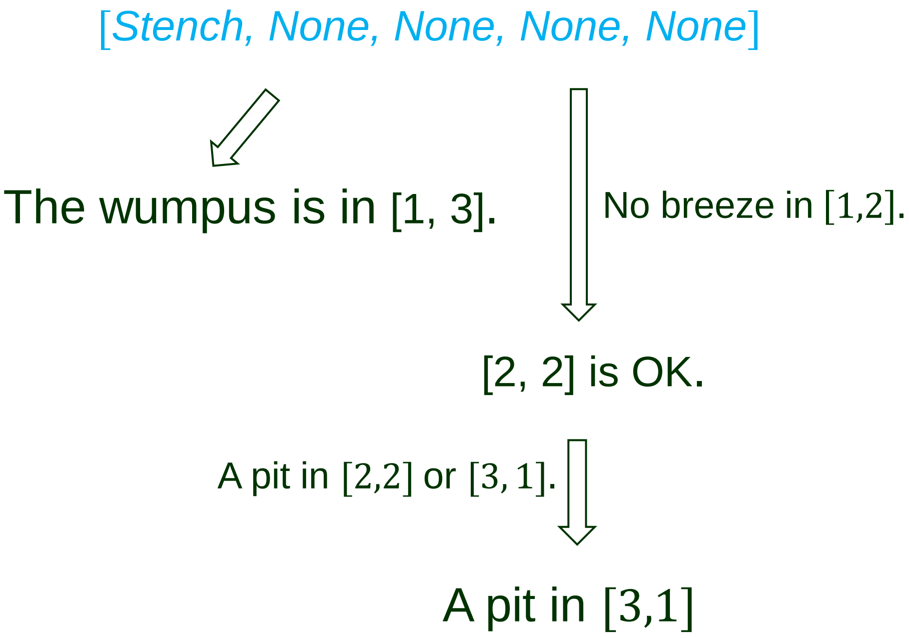
A pit in [3,1]

- Moves to (2, 2).
- Assume then it turns and moves to (2, 3) based on percept at (2,2).

More Inference



1,4	2,4 P?	3,4	4,4
1,3 W!	2,3 A S G B	3,3 P?	4,3
1,2 S V OK	2,2 V OK	3,2	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1



[Stench, None, None, None, None]

The wumpus is in [1, 3].

No breeze in [1,2].

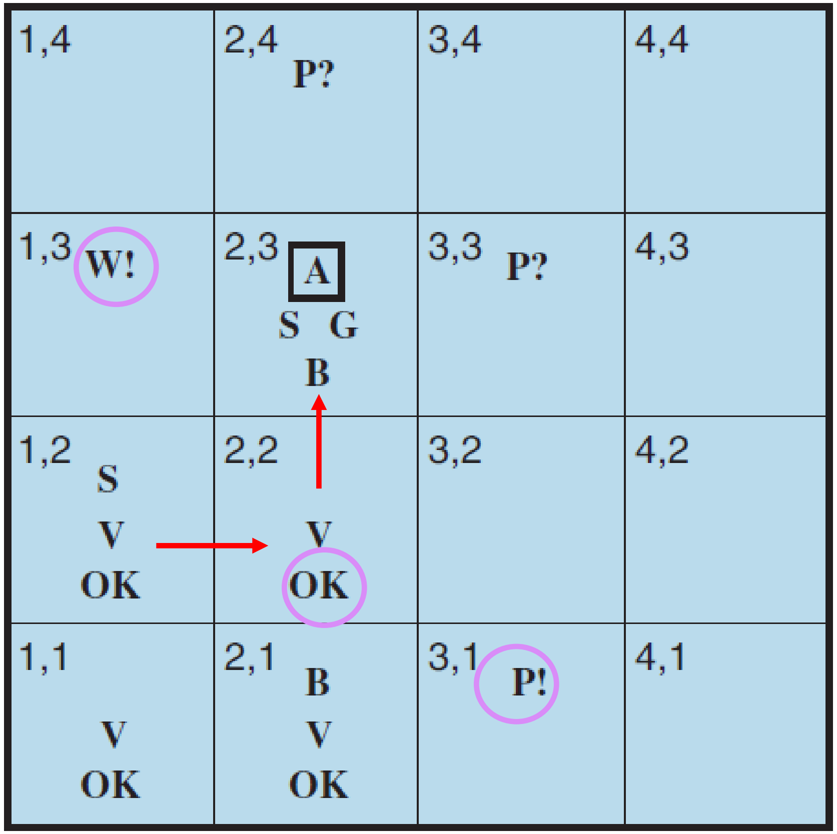
[2, 2] is OK.

A pit in [2,2] or [3, 1].

A pit in [3,1]

- Moves to (2, 2).
- Assume then it turns and moves to (2, 3) based on percept at (2,2).
- Grab the gold and return home.

More Inference



[Stench, None, None, None, None]

The wumpus is in [1, 3].

No breeze in [1,2].

[2, 2] is OK.

A pit in [2,2] or [3, 1].

A pit in [3,1]

A conclusion drawn is guaranteed if the available information is correct.

- Moves to (2, 2).
- Assume then it turns and moves to (2, 3) based on percept at (2,2).
- Grab the gold and return home.

III. Logic

- A systematic study of rules of inference.
- A formal language for representing information such that conclusions can be drawn.

III. Logic

- A systematic study of rules of inference.
- A formal language for representing information such that conclusions can be drawn.
  - *Syntax* – what expressions are legal (*well-formed* sentences)

III. Logic

- A systematic study of rules of inference.
- A formal language for representing information such that conclusions can be drawn.

  - ♦ *Syntax* – what expressions are legal (*well-formed* sentences)

    “$x + y = 4$” is a sentence but “$x4y +=$” is not.

III. Logic

- A systematic study of rules of inference.
- A formal language for representing information such that conclusions can be drawn.

  - ♦ *Syntax* – what expressions are legal (*well-formed* sentences)

    “$x + y = 4$” is a sentence but “$x4y +=$” is not.

  - ♦ *Semantics* – what the “meanings” of sentences are.

III. Logic

- A systematic study of rules of inference.
- A formal language for representing information such that conclusions can be drawn.

  - *Syntax* – what expressions are legal (*well-formed* sentences)

    “$x + y = 4$” is a sentence but “$x4y +=$” is not.

  - *Semantics* – what the “meanings” of sentences are.

    Truth of a sentence w.r.t. each possible world (*model*).

III. Logic

- A systematic study of rules of inference.
- A formal language for representing information such that conclusions can be drawn.

  ♦ *Syntax* – what expressions are legal (*well-formed* sentences)

  “$x + y = 4$” is a sentence but “$x4y +=$” is not.

  ♦ *Semantics* – what the “meanings” of sentences are.

  Truth of a sentence w.r.t. each possible world (*model*).

  “$x + y = 4$” is true in a world where x is 2 and y is 2,
  but false in a world where x is 1 and y is 1.

III. Logic

- A systematic study of rules of inference.
- A formal language for representing information such that conclusions can be drawn.

  - ♦ *Syntax* – what expressions are legal (*well-formed* sentences)

    "$x + y = 4$" is a sentence but "$x4y +=$" is not.

  - ♦ *Semantics* – what the "meanings" of sentences are.

    Truth of a sentence w.r.t. each possible world (*model*).

    "$x + y = 4$" is true in a world where x is 2 and y is 2,
    but false in a world where x is 1 and y is 1.

Every sentence must be either true or false in each possible world.

Model and Reasoning

Model m: assigns values to variables.

m *satisfies* a sentence α, or m is a model of α, if α is true in m.

Model and Reasoning

Model m: assigns values to variables.

m *satisfies* a sentence α, or m is a model of α, if α is true in m.

$M(\alpha)$: set of all models of α.

Model and Reasoning

Model m: assigns values to variables.

m *satisfies* a sentence α, or m is a model of α, if α is true in m.

$M(\alpha)$: set of all models of α.

Logical entailment

$$\underset{\text{“The sentence } \alpha \text{ \textit{entails} the sentence } \beta\text{.”}}{\alpha \models \beta}$$

Model and Reasoning

Model m: assigns values to variables.

m *satisfies* a sentence α, or m is a model of α, if α is true in m.

$M(\alpha)$: set of all models of α.

Logical entailment

$\alpha \models \beta$ if and only if every model of α is also a model of β.

"The sentence α *entails* the sentence β."

Model and Reasoning

Model m: assigns values to variables.

m *satisfies* a sentence α, or m is a model of α, if α is true in m.

$M(\alpha)$: set of all models of α.

Logical entailment

$\alpha \vDash \beta$ if and only if every model of α is also a model of β.

"The sentence α *entails* the sentence β."

Equivalently,

$\alpha \vDash \beta$ if and only if $M(\alpha) \subseteq M(\beta)$

Model and Reasoning

Model m: assigns values to variables.

m *satisfies* a sentence α, or m is a model of α, if α is true in m.

$M(\alpha)$: set of all models of α.

Logical entailment

$\alpha \vDash \beta$ if and only if every model of α is also a model of β.

"The sentence α *entails* the sentence β."

Equivalently,

$\alpha \vDash \beta$ if and only if $M(\alpha) \subseteq M(\beta)$

$\Downarrow$

α is a stronger assertion than β.

Model and Reasoning

Model m: assigns values to variables.

m *satisfies* a sentence α, or m is a model of α, if α is true in m.

$M(\alpha)$: set of all models of α.

Logical entailment

$\alpha \vDash \beta$ if and only if every model of α is also a model of β.

"The sentence α *entails* the sentence β."

Equivalently,

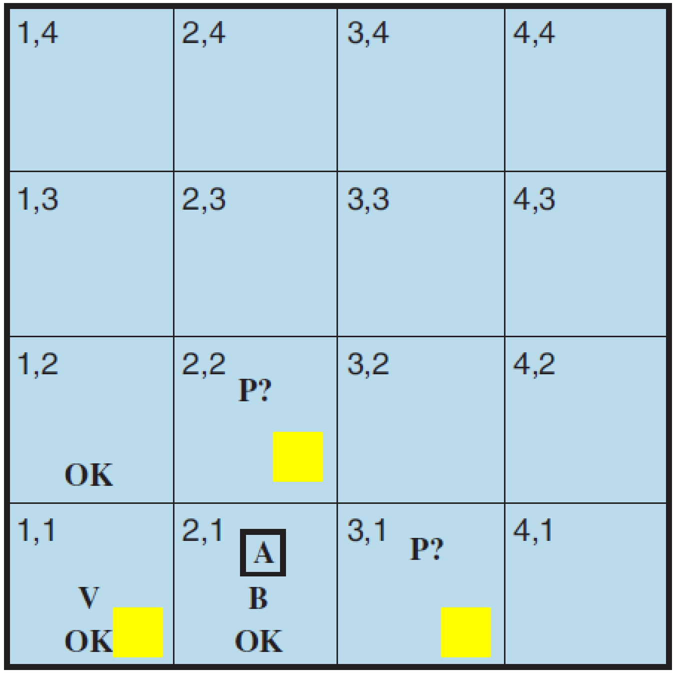
$\alpha \vDash \beta$ if and only if $M(\alpha) \subseteq M(\beta)$

$\Downarrow$

α is a stronger assertion than β.

Example $x = 0$ entails $xy = 0$.

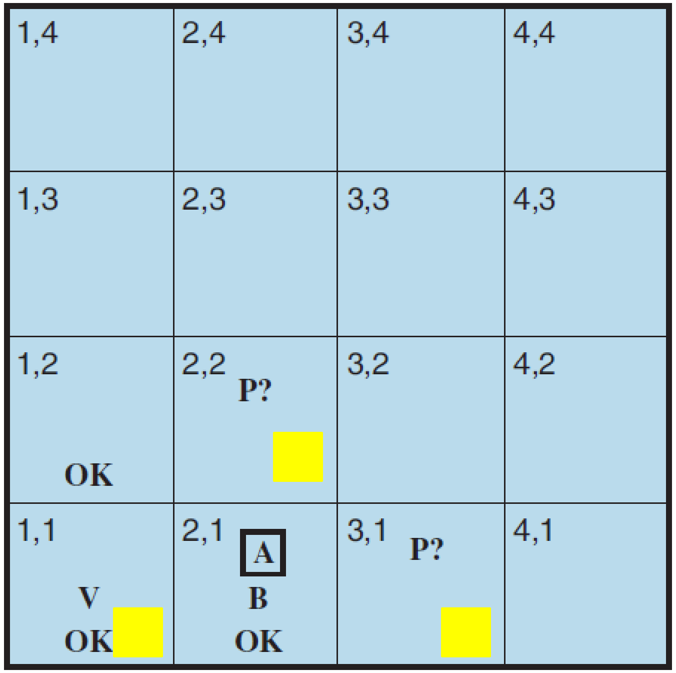
Back to the Wumpus World



Knowledge base (KB) includes

- All the rules.
- Percepts:
  [None, None, None, None, None] in [1,1]
  [None, Breeze, None, None, None] in [2,1]

Back to the Wumpus World



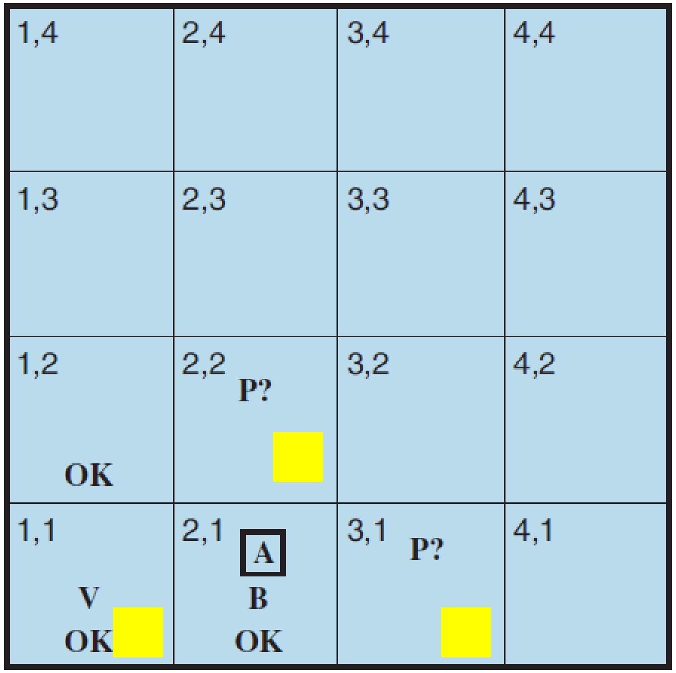
1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P? 🟨	3,2	4,2
1,1 V OK 🟨	2,1 $\boxed{A}$ B OK	3,1 P? 🟨	4,1

Knowledge base (KB) includes

- All the rules.
- Percepts:
  [None, None, None, None, None] in [1,1]
  [None, Breeze, None, None, None] in [2,1]

Q. Does the three squares adjacent to $[2,1]$ contain pits?

Back to the Wumpus World



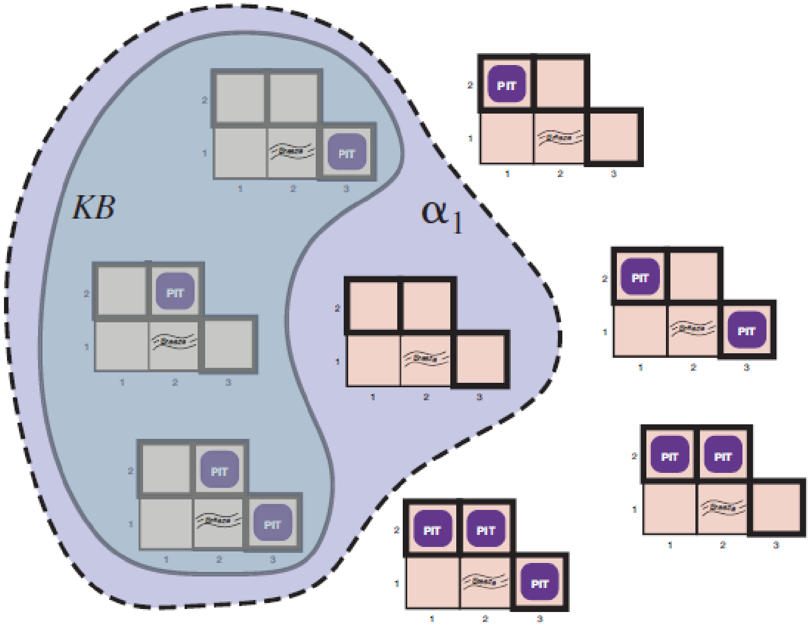
Knowledge base (KB) includes

- All the rules.
- Percepts:
  [*None, None, None, None, None*] in [1,1]
  [*None, Breeze, None, None, None*] in [2,1]

Q. Does the three squares adjacent to $[2,1]$ contain pits?

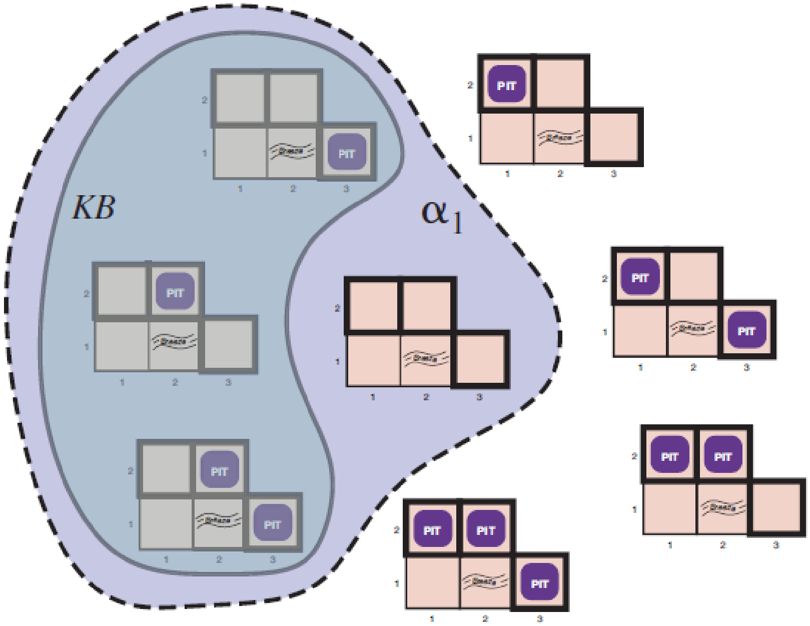
8 possibilities if ignoring the KB.

Which Neighbors Contain a pit?



8 possible models for the presence of pits in squares $[1,2]$, $[2,2]$, and $[3,1]$.

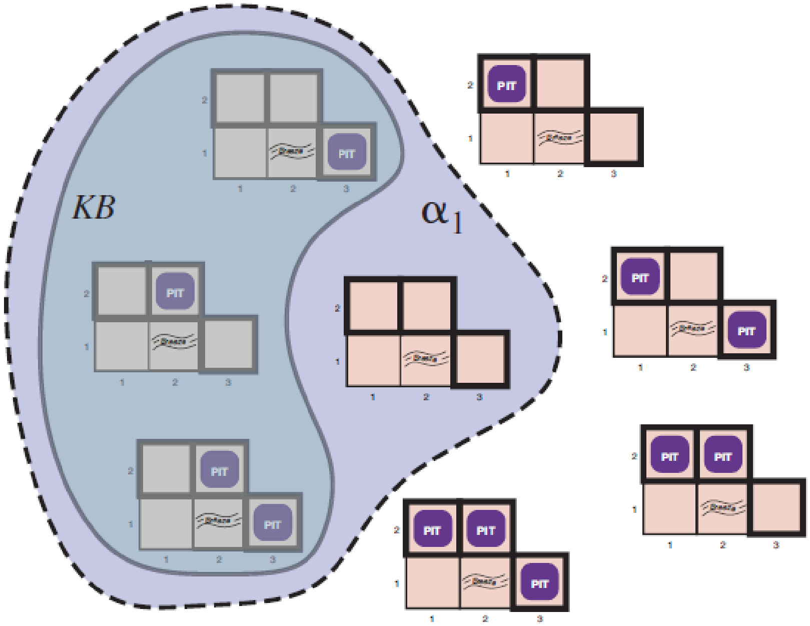
Which Neighbors Contain a pit?



8 possible models for the presence of pits in squares $[1,2]$, $[2,2]$, and $[3,1]$.

3 models in which the KB is true given the two percepts.

Which Neighbors Contain a pit?

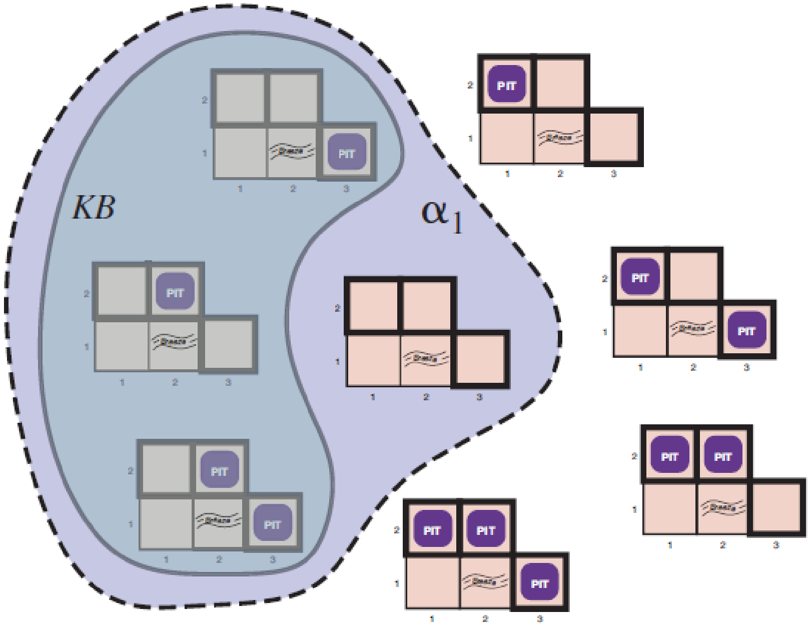


8 possible models for the presence of pits in squares $[1,2]$, $[2,2]$, and $[3,1]$.

3 models in which the KB is true given the two percepts.

α_1 =“There is no pit in $[1,2]$.”

Which Neighbors Contain a pit?



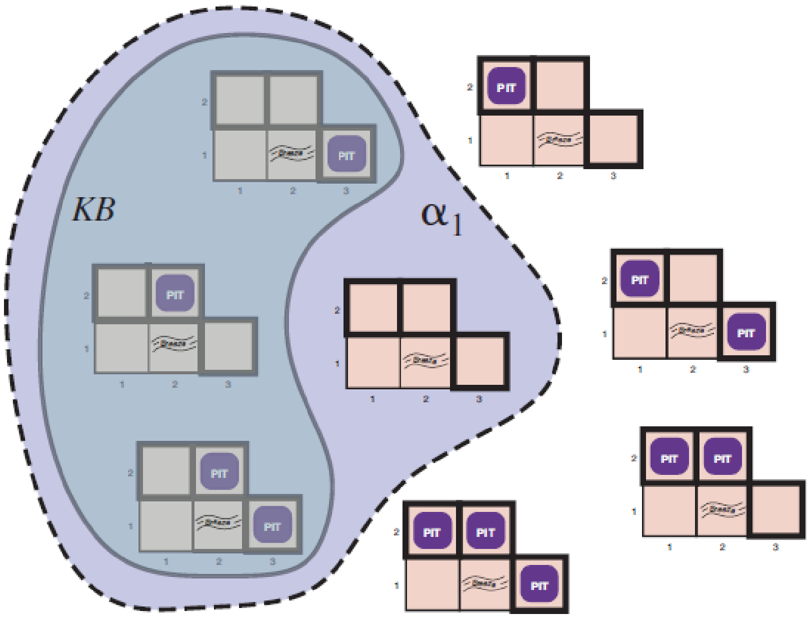
8 possible models for the presence of pits in squares $[1,2]$, $[2,2]$, and $[3,1]$.

3 models in which the KB is true given the two percepts.

α_1 =“There is no pit in $[1,2]$.”

True in 4 models.

Which Neighbors Contain a pit?



8 possible models for the presence of pits in squares $[1,2]$, $[2,2]$, and $[3,1]$.

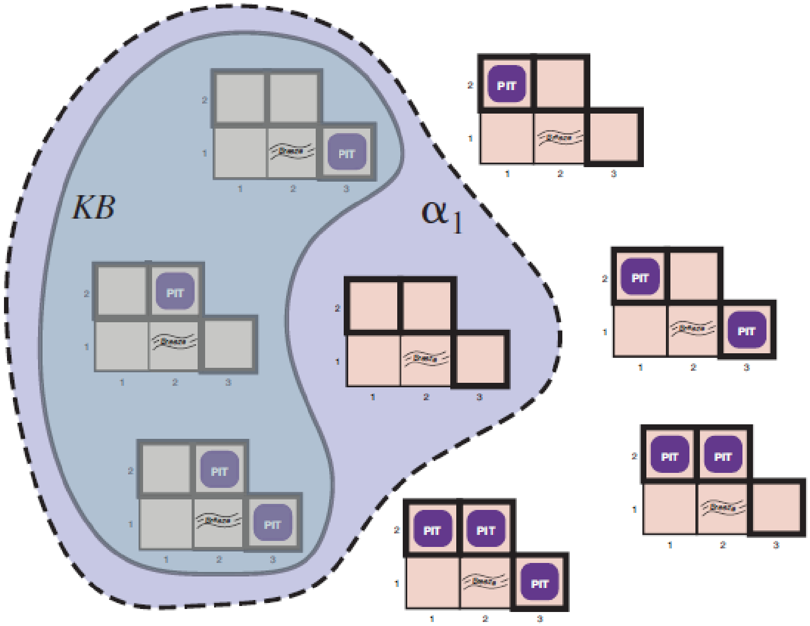
3 models in which the KB is true given the two percepts.

α_1 =“There is no pit in $[1,2]$.”

True in 4 models.

KB $\vDash \alpha_1$ since $M($*KB*$) \subseteq M(\alpha_1)$

Which Neighbors Contain a pit?



8 possible models for the presence of pits in squares $[1,2]$, $[2,2]$, and $[3,1]$.

3 models in which the KB is true given the two percepts.

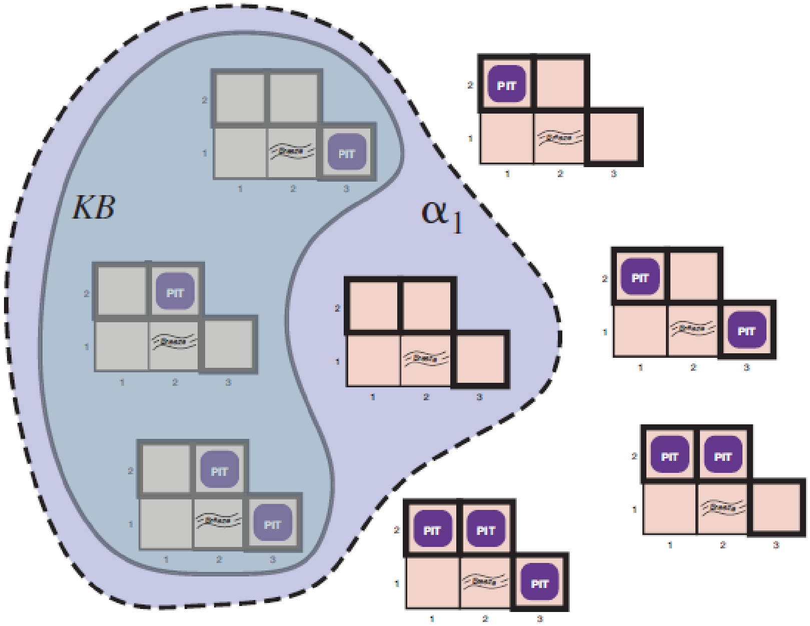
α_1 =“There is no pit in $[1,2]$.”

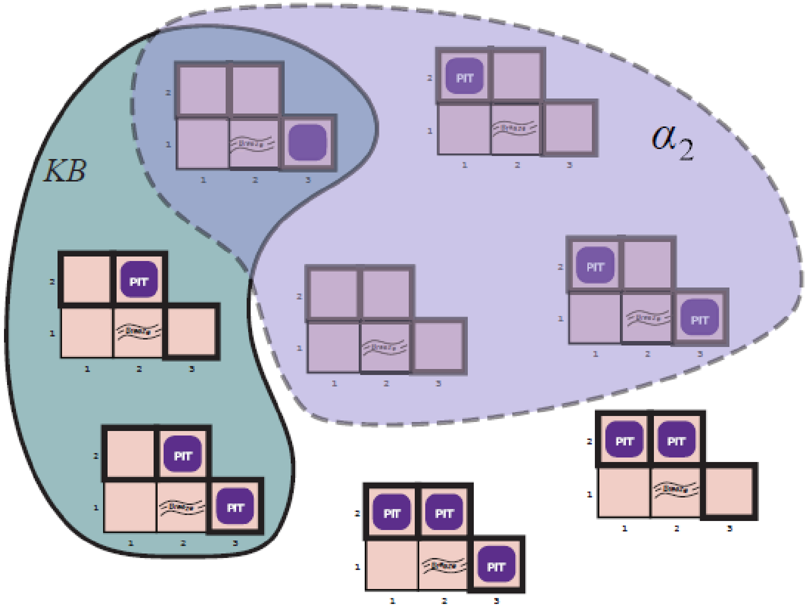
True in 4 models.

model checking

$KB \vDash \alpha_1$ since $\boxed{M(KB) \subseteq M(\alpha_1)}$

Which Neighbors Contain a pit?





8 possible models for the presence of pits in squares $[1,2]$, $[2,2]$, and $[3,1]$.

3 models in which the KB is true given the two percepts.

α_1 =“There is no pit in $[1,2]$.”

True in 4 models.

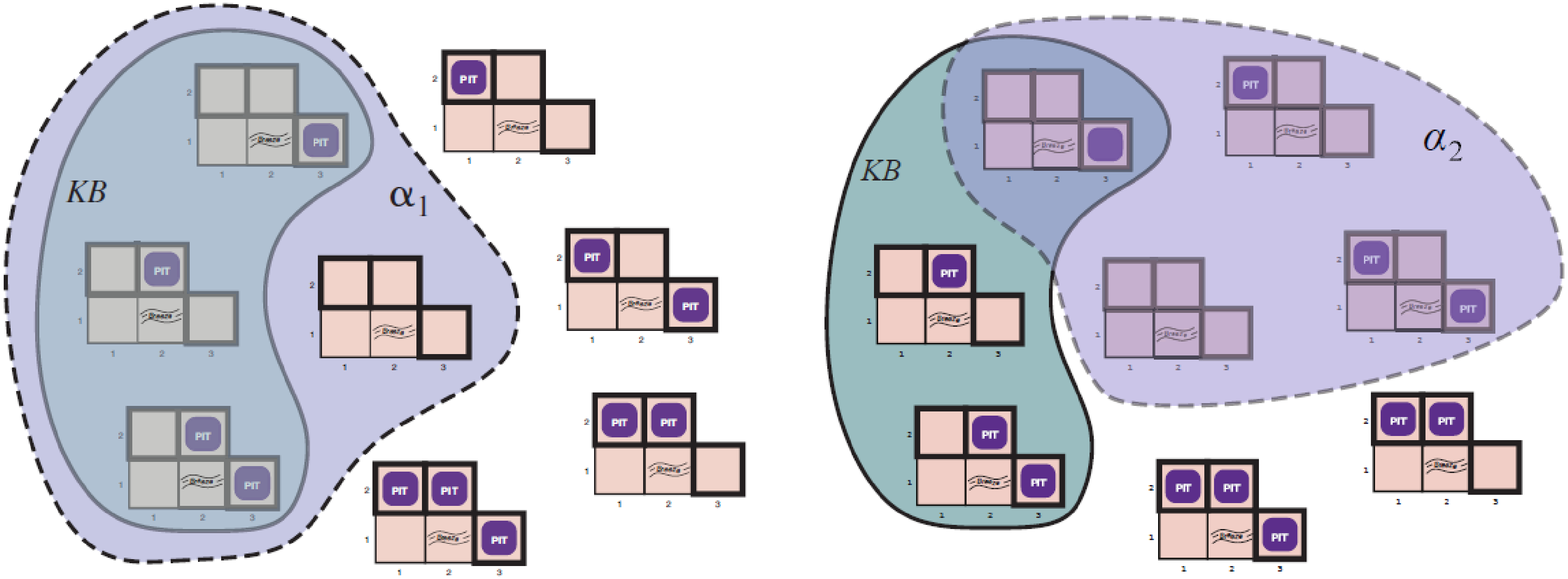
α_2 =“There is no pit in $[2,2]$.”

True in 4 models.

model checking

$KB \vDash \alpha_1$ since $\boxed{M(KB) \subseteq M(\alpha_1)}$

Which Neighbors Contain a pit?



8 possible models for the presence of pits in squares $[1,2]$, $[2,2]$, and $[3,1]$.

3 models in which the KB is true given the two percepts.

α_1 ="There is no pit in $[1,2]$."

True in 4 models.

α_2 ="There is no pit in $[2,2]$."

True in 4 models.

model checking

$KB \models \alpha_1$ since $\boxed{M(KB) \subseteq M(\alpha_1)}$

$KB \not\models \alpha_2$ since $M(KB) \not\subseteq M(\alpha_2)$

Derivation vs. Entailment

Inference is like finding a needle entailed by (known to be in) a haystack (KB).

Derivation vs. Entailment

Inference is like finding a needle entailed by (known to be in) a haystack (KB).

$$KB \vdash_i \alpha$$

Derivation vs. Entailment

Inference is like finding a needle entailed by (known to be in) a haystack (KB).

$KB \vdash_i \alpha$ “α is *derived* from KB by (the inference algorithm) i.”

Derivation vs. Entailment

Inference is like finding a needle entailed by (known to be in) a haystack (KB).

$KB \vdash_i \alpha$ “α is *derived* from KB by (the inference algorithm) i.”

“i *derives* α from KB.”

Derivation vs. Entailment

Inference is like finding a needle entailed by (known to be in) a haystack (KB).

$KB \vdash_i \alpha$ “α is *derived* from KB by (the inference algorithm) i.”

“i *derives* α from KB.”

- ♣ An inference algorithm i is *sound* or *truth-preserving* if it derives only entailed sentences, that is.

$KB \models \alpha$ whenever $KB \vdash_i \alpha$

Derivation vs. Entailment

Inference is like finding a needle entailed by (known to be in) a haystack (KB).

$KB \vdash_i \alpha$ "α is *derived* from KB by (the inference algorithm) i."

"i *derives* α from KB."

♣ An inference algorithm i is *sound* or *truth-preserving* if it derives only entailed sentences, that is.

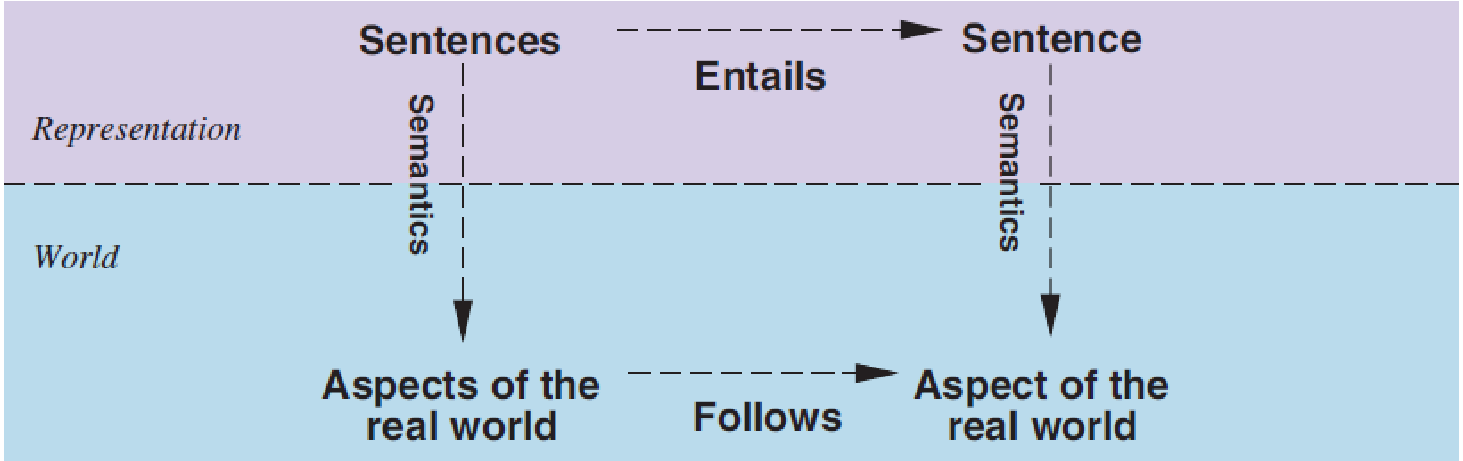
$KB \vDash \alpha$ whenever $KB \vdash_i \alpha$

♣ It is *complete* if it can derive any sentence that is entailed, that is,

$KB \vdash_i \alpha$ whenever $KB \vDash \alpha$

Logical Reasoning

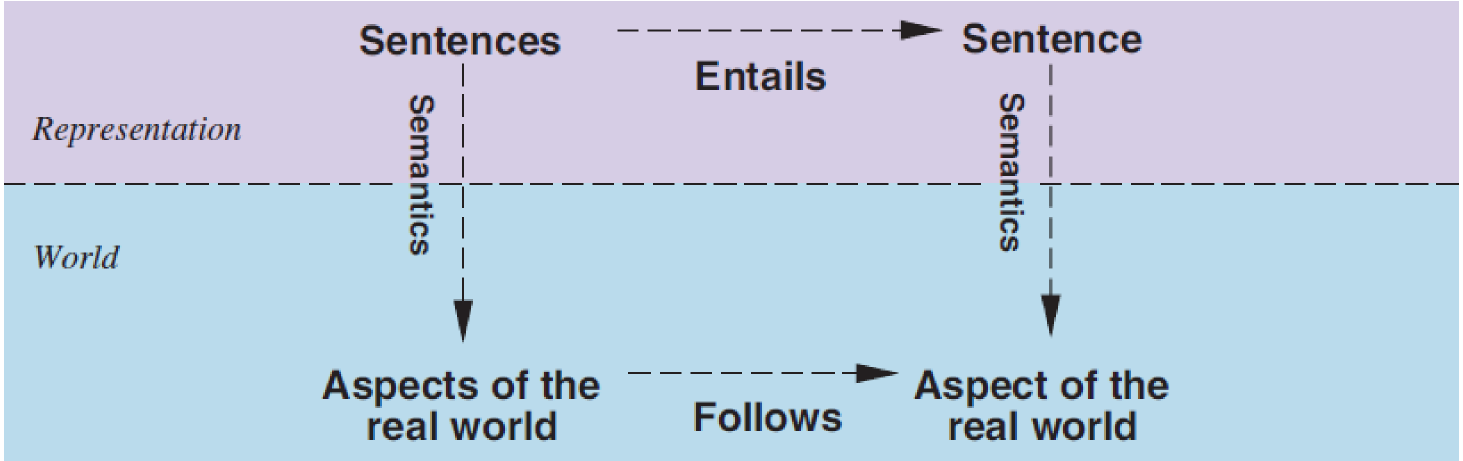
A process whose conclusions are guaranteed to be true in any world in which the premises (the *KB* in this case) are true.



Correspondence between world and representation

Logical Reasoning

A process whose conclusions are guaranteed to be true in any world in which the premises (the *KB* in this case) are true.



Correspondence between world and representation