Logical Agents

Fundamentals of Artificial Intelligence

Slides are mostly adapted from AIMA and MIT Open Courseware

knowledge-based agents

agents that reason by operating on internal representations of knowledge

If it didn't rain, Harry visited Hagrid today.
Harry visited Hagrid or Dumbledore today, but not both.
Harry visited Dumbledore today.

It rained today.
Harry did not visit Hagrid today.

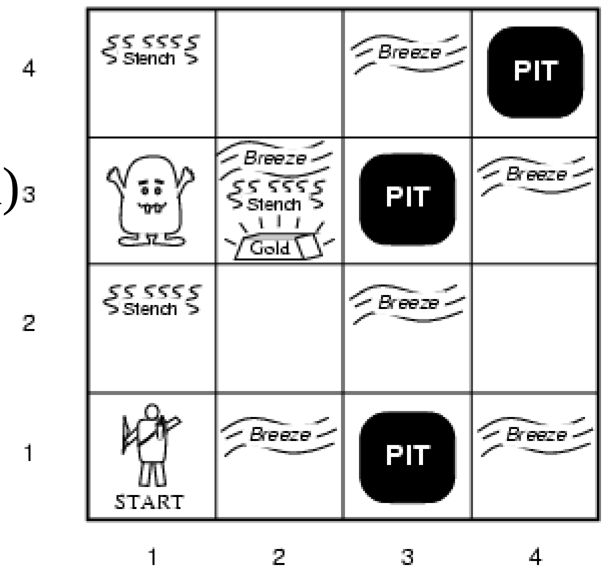
Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Introduction

- The **representation of knowledge** and the **reasoning processes** that bring knowledge to life are central to entire field of artificial intelligence
- Knowledge and reasoning are important to artificial agents because they enable successful behaviors that would be very hard to achieve otherwise (no piece in chess can be on two different squares at the same time)
- Knowledge and reasoning also play a crucial role in dealing with partially observable environments (inferring hidden states in diagnosing diseases, natural language understanding)
- Knowledge also allows flexibility.

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 (stench)
  - 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



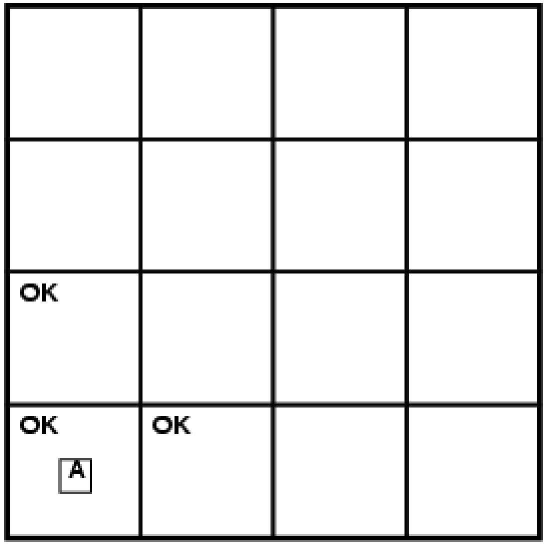
- Sensors: Stench, Breeze, Glitter, Bump, Scream
- Actuators: Left turn, Right turn, Forward, Grab, Release, Shoot

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

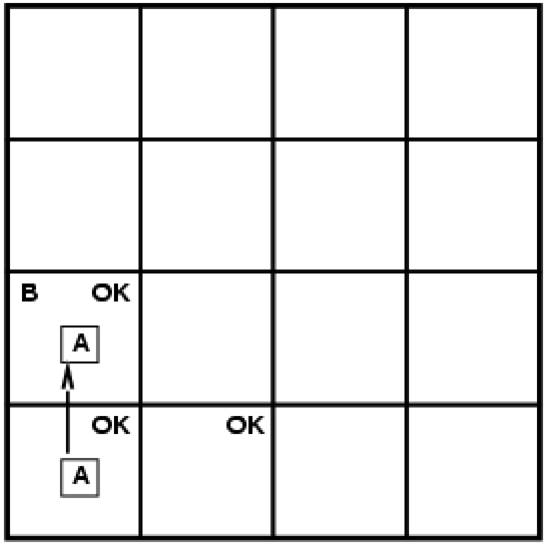
[1,1] is OK
Because
Haven’t fallen into a pit
Haven’t been eaten by a Wumpus



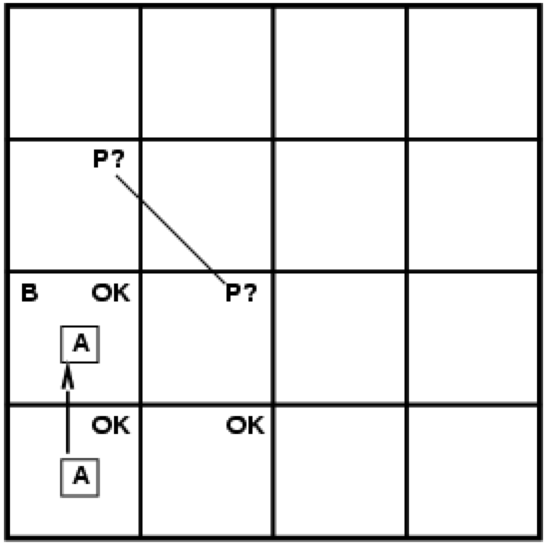
[1,2] and [2,1] are OK
Because
No stench
No breeze

Exploring a wumpus world

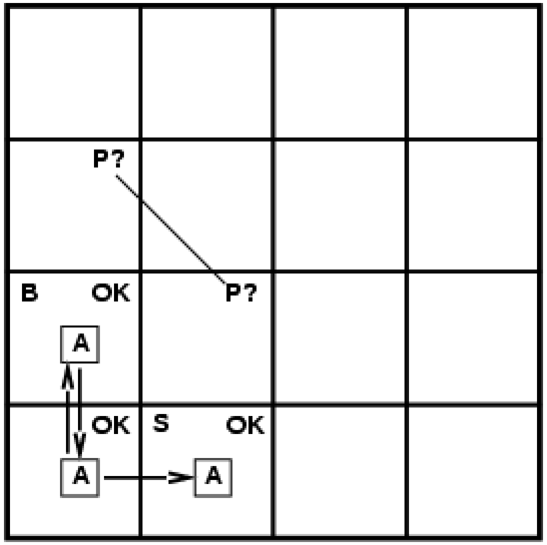
We move to [1,2] and
Feel a Breeze



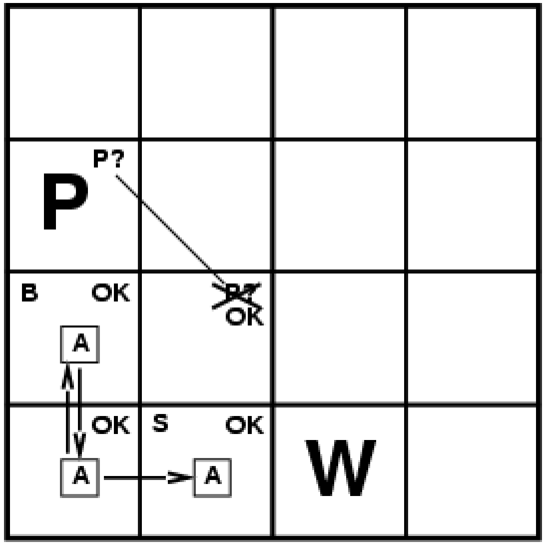
Exploring a wumpus world



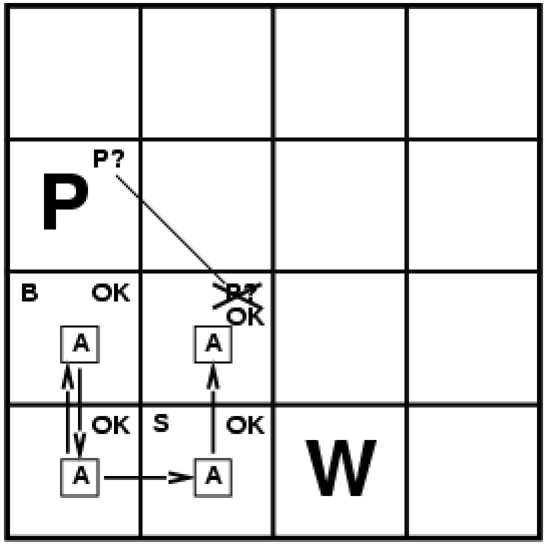
Exploring a wumpus world



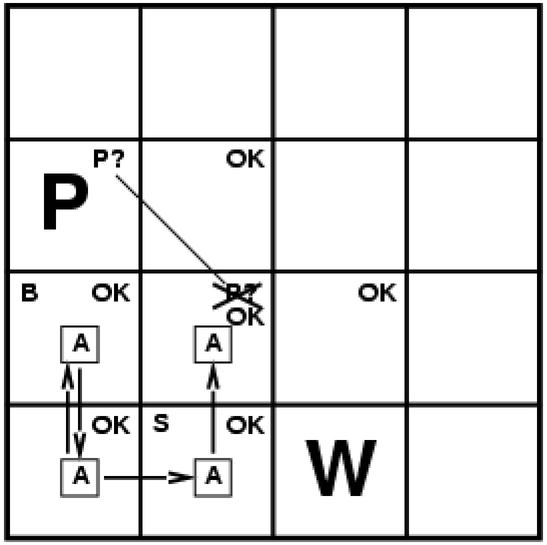
Exploring a wumpus world



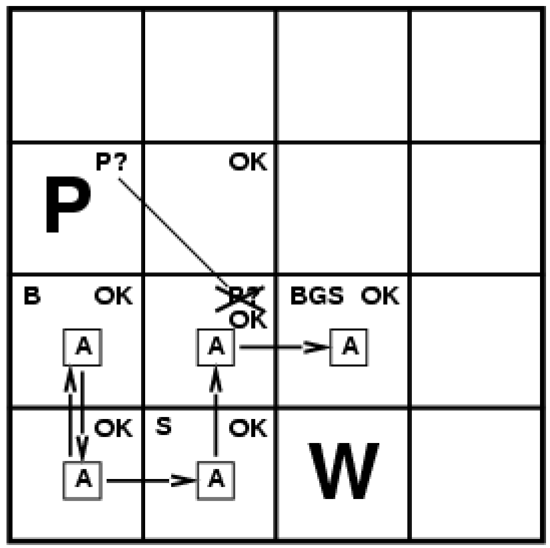
Exploring a wumpus world



Exploring a wumpus world



Exploring a wumpus world



Logic in general

- Logics are formal languages for representing information such that conclusions can be drawn

- Sentence: an assertion about the world in a knowledge representation language

- 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 $x = 0$, $y = 6$

Propositional Logic

Propiositions: Statements about the world

Propositional Symbols: represent a fact about the world

P , Q , R

It is raining

Harry wisited Hagrid today, etc

Logical Connectives

¬ not

∧ and

∨ or

→ implication

↔ biconditional

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Propositional Logic

Not (¬)

P	$\neg P$
false	true
true	false

P : it is raining

~P : It is not raining

And (∧)

P	Q	$P \wedge Q$
false	false	false
false	true	false
true	false	false
true	true	true

Both P and Q are true

Or (∨)

P	Q	$P \vee Q$
false	false	false
false	true	true
true	false	true
true	true	true

Either of P or Q or both are true

Slide credit : HarvardX CS50AICS50's Introduction to
David J. Malan and Brian Yu

Propositional Logic

Implication (→)

P	Q	$P \to Q$
false	false	true
false	true	true
true	false	false
true	true	true

If P is true Q is also true
If it is raining, I will be indoors

If it is raining but I am not indoors,
then my original statement is not true
If P is false, then the statement does not
make any claim
If it is not raining, I am not making any
claim about whether I will be indoors or not

Biconditional (↔)

P	Q	$P \leftrightarrow Q$
false	false	true
false	true	false
true	false	false
true	true	true

I will be indoors if and only if it is raining

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python,
David J. Malan and Brian Yu

Model

assignment of a truth value to every propositional symbol (a "possible world")

P: It is raining.

Q: It is a Tuesday.

{*P* = false, *Q* = true}

For n number of symbols there are 2^n possible models

Knowledge Base and Entailment

knowledge base
a set of sentences known by a knowledge-based agent

This information is used to come up with conclusions

Entailment
$\alpha \vDash \beta$ In every model in which sentence α is true, sentence β is also true.

If it a Tuesday in March then it entails that it is March

Inference

the process of deriving new sentences from old ones

If it didn't rain, Harry visited Hagrid today.
Harry visited Hagrid or Dumbledore today, but not both.
Harry visited Dumbledore today.

Harry did not visit Hagrid today.
It rained today.

Inference

P: It is a Tuesday.
Q: It is raining.
R: Harry will go for a run.

KB:
$(P \land \neg Q) \rightarrow R$
P
$\neg Q$

Inference:
R

Entailment

- Entailment means that one thing follows from another:

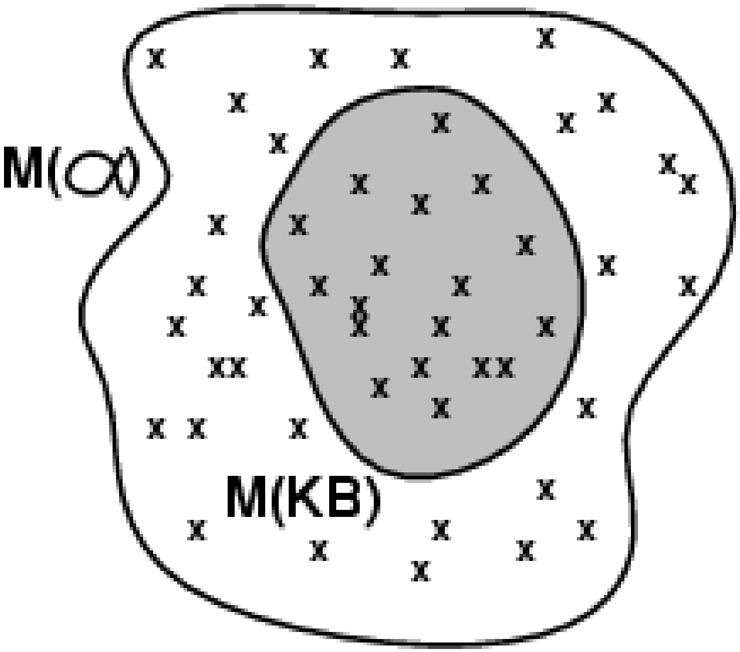
$$KB \models \alpha$$

- Knowledge base *KB* entails sentence α if and only if α is true in all worlds where *KB* is true
  - If α true then *KB* must also be true
  - 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 KB $\models$ α iff *M(KB)* $\subseteq$ *M*(α)
  - E.g. *KB* = It is a Tuesday in March

    α = It is March



Possible world – model

m is a model of α – the sentence α is true in model m

Model checking

To determine if KB $\vDash \alpha$:

- Enumerate all possible models.
- If in every model where KB is true, α is true, then KB entails α.
- Otherwise, KB does not entail α.

P: It is a Tuesday. Q: It is raining. R: Harry will go for a run.

KB: $(P \land \neg Q) \rightarrow R$ P $\neg Q$

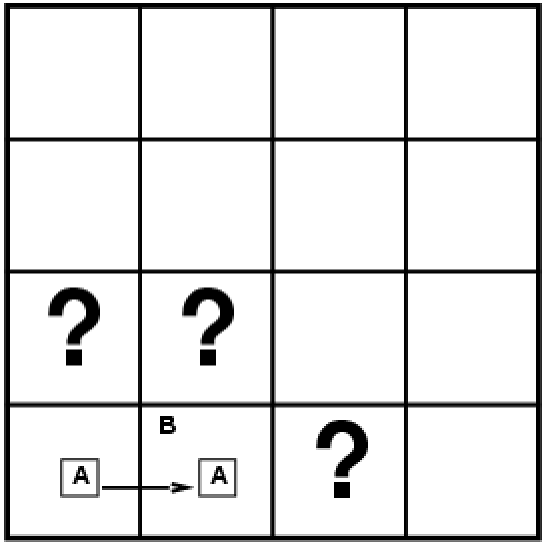
Query: R

P	Q	R	KB
false	false	false	false
false	false	true	false
false	true	false	false
false	true	true	false
true	false	false	false
true	false	true	true
true	true	false	false
true	true	true	false

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

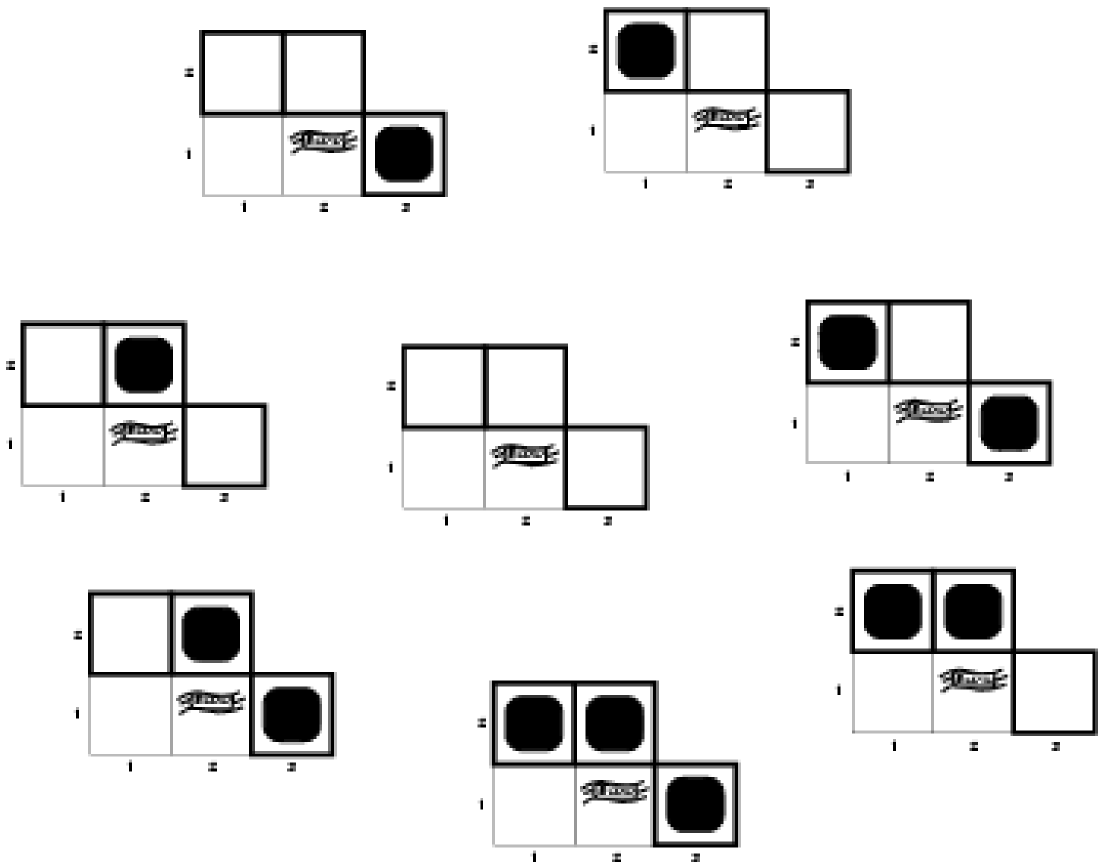
Entailment in the wumpus world

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



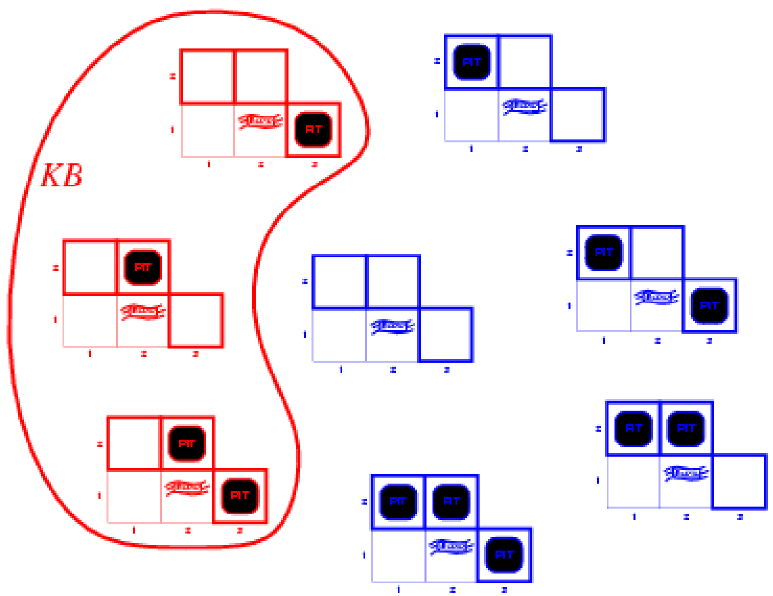
Wumpus models

3 Boolean choices ⇒ 8 possible models
for the adjacent squares [1,2], [2,2] and [3,1] to contain pits



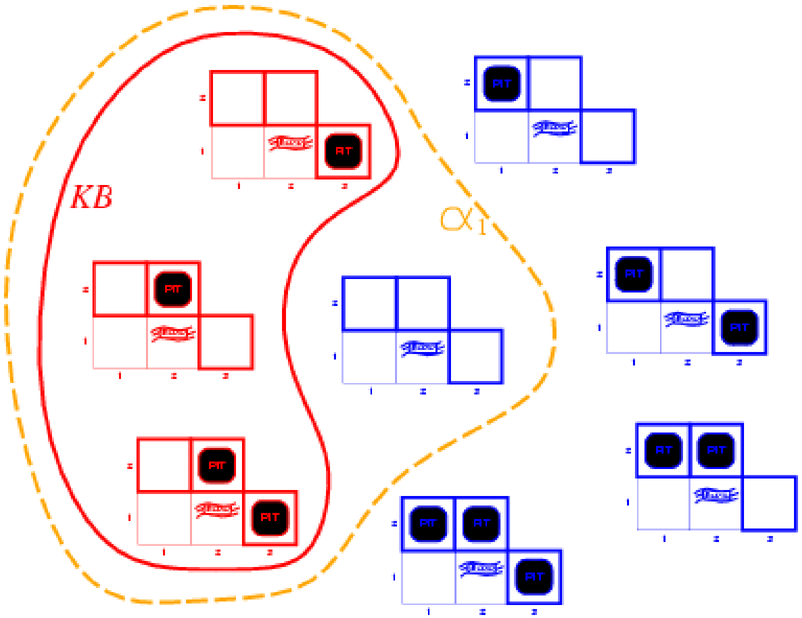
Wumpus models

Consider possible models for *KB* assuming only pits



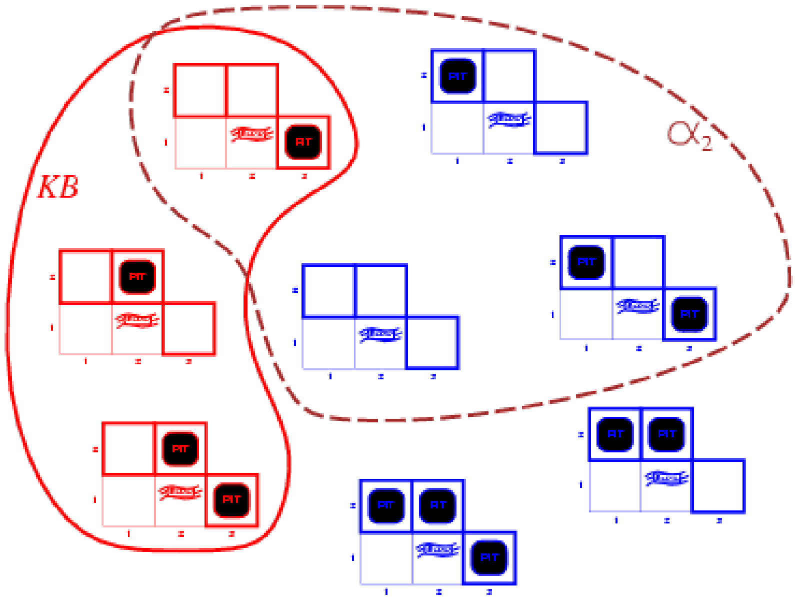
- *KB* = wumpus-world rules + observations
- KB is false in any model in which [1,2] contains a pit, because there is no breeze in [1,1]

Wumpus models



- Consider α_1 = "[1,2] is safe" = "There is no pit in [1,2]"
- *In every model KB is true* α_1 *is also true*
- *KB* $\models \alpha_1$, proved by model checking
- We can conclude that there is no pit in [1,2]

Wumpus models



- Consider α_2 = “[2,2] is safe” = “There is no pit in [2,2]”
- *In some models in which KB is true* α_2 *is false*
- $KB \not\models \alpha_2$
- We cannot conclude that there is no pit in [2,2]

Wumpus world sentences

Let $P_{i,j}$ be true if there is a pit in [i, j].

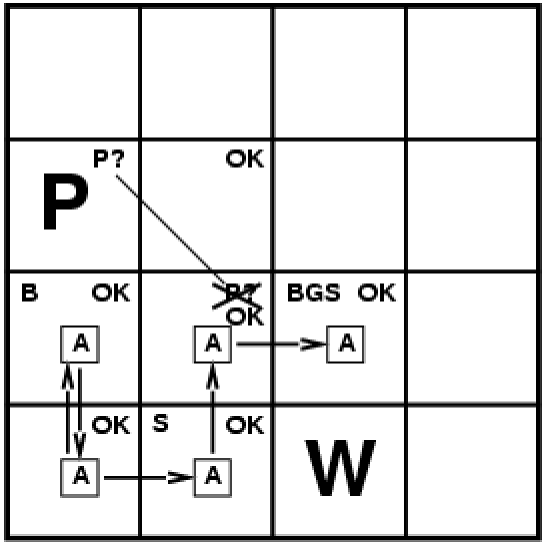
Let $B_{i,j}$ be true if there is a breeze in [i, j].

Knowledge base includes:

R1: $\neg P_{1,1}$ No pit in [1,1]

R2: $\neg B_{1,1}$ No breeze in [1.1]

R3: $B_{2,1}$ Breeze in [2,1]



- "Pits cause breezes in adjacent squares"

R4: $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

R5: $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

$$KB = R1 \wedge R2 \wedge R3 \wedge R4 \wedge R5$$

Truth tables for inference

- Decide whether $KB \models \alpha$
- First method: enumerate the models and check that α is true in every model in which KB is true
- $B_{1,1}$ $B_{2,1}$, $P_{1,1}$, $P_{1,2}$, $P_{2,1}$, $P_{2,2}$, $P_{3,1}$
- 7 symbols : $2^7 = 128$ possible models

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

R1: $\neg P_{1,1}$

R2: $\neg B_{1,1}$

R3: $B_{2,1}$

R4: $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

R5: $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

KB = R1 ∧ R2 ∧ R3 ∧ R4 ∧ R5

$\alpha 1 = \neg P_{1,2}$

$\alpha 2 = P_{2,2}$

α1 is true in all models that KB is true

α2 is true only in two models that KB is true, but false in the other one

Example Problem

Imagine we knew that:

- If today is sunny, then Tomas will be happy ($S \rightarrow H$)
- If Tomas is happy, the lecture will be good ($H \rightarrow G$)
- Today is sunny (S)

Should we conclude that the lecture will be good?

Checking Interpretations

- Start by figuring out what set of interpretations make our original sentences true.
- Then, if G is true in all those interpretations, it must be OK to conclude it from the sentences we started out with (our knowledge base).

- ***In a universe with only three variables, there are 8 possible interpretations in total.***

S	H	G
t	t	t
t	t	f
t	f	t
t	f	f
f	t	t
f	t	f
f	f	t
f	f	f

Checking Interpretations

- Only one of these interpretations makes all the sentences in our knowledge base true:
- S = true, H = true, G = true.

S	H	G	S→ H	H→G	S
t	t	t	t	t	t
t	t	f	t	f	t
t	f	t	f	t	t
t	f	f	f	t	t
f	t	t	t	t	f
f	t	f	t	f	f
f	f	t	t	t	f
f	f	f	t	t	f

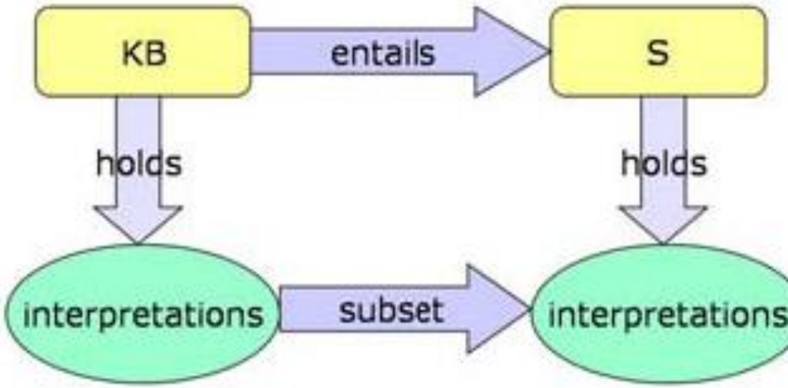
Checking Interpretations

- it's easy enough to check that G is true in that interpretation, so it seems like it must be reasonable to draw the conclusion that the lecture will be good.

S	H	G	S →H	H→G	S	G
t	t	t	t	t	t	t
t	t	f	t	f	t	f
t	f	t	f	t	t	t
t	f	f	f	t	t	f
f	t	t	t	t	f	t
f	t	f	t	f	f	f
f	f	t	t	t	f	t
f	f	f	t	t	f	f

Computing entailment

A knowledge base (KB) *entails* a sentence S iff every interpretation that makes KB true also makes S true



- enumerate all interpretations
- select those in which all elements of KB are true
- check to see if S is true in all of those interpretations

Model checking

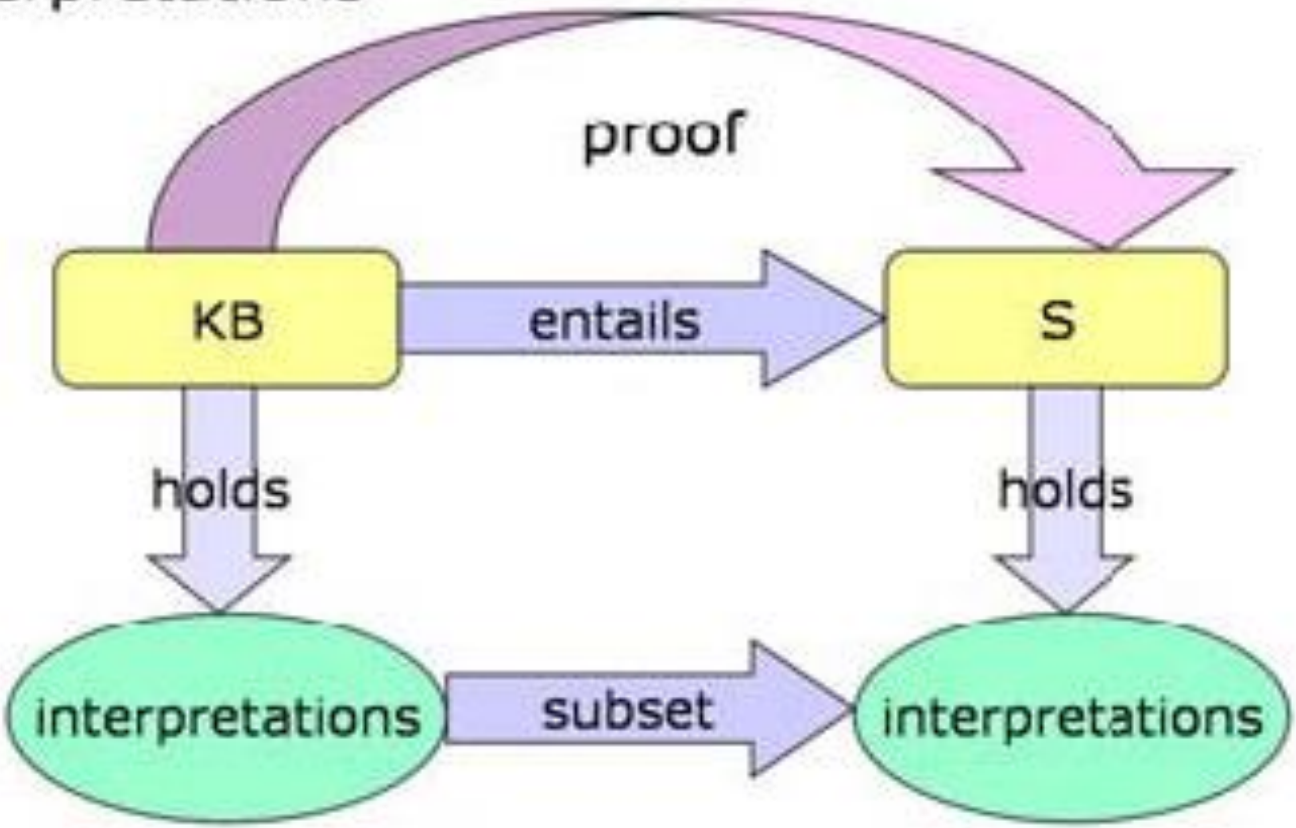
Truth table enumeration

For n symbols the time complexity is O(2^n)

Need a smarter way to do inference

Entailment and Proof

A proof is a way to test whether a KB entails a sentence, without enumerating all possible interpretations



Proof

- Proof is a sequence of sentences
- First ones are premises (KB)
- Then, you can write down on the next line the result of applying an inference rule to previous lines
- When S is on a line, you have proved S from KB

- If inference rules are *sound*, then any S you can prove from KB is entailed by KB

- If inference rules are *complete*, then any S that is entailed by KB can be proved from KB

Rules of inference

- $KB \vdash_i \alpha$ = sentence α can be derived from *KB* by a procedure *i* (an inference algorithm)

- If we generalize it to any two sentences α and β
- $\alpha \vdash \beta$ means that β is derived from α by an inference
- The alternative notation:

$$\frac{\alpha}{\beta}$$

premise

conclusion

Inference Rules

Modus Ponens

If it is raining, then Harry is inside.

It is raining.

Harry is inside.

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

Inference Rules

And Elimination

Harry is friends with Ron and Hermione.

Harry is friends with Hermione.

$$\frac{\alpha \land \beta}{\alpha}$$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Inference Rules

Double Negation Elimination

It is not true that Harry did not pass the test.

Harry passed the test.

$$\frac{\neg(\neg\alpha)}{\alpha}$$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python,
David J. Malan and Brian Yu

Inference Rules

Implication Elimination

If it is raining, then Harry is inside.

It is not raining or Harry is inside.

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

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Inference Rules

Biconditional Elimination

It is raining if and only if Harry is inside.

If it is raining, then Harry is inside,
and if Harry is inside, then it is raining.

$$\frac{\alpha \leftrightarrow \beta}{(\alpha \rightarrow \beta) \land (\beta \rightarrow \alpha)}$$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python,
David J. Malan and Brian Yu

Inference Rules

De Morgan's Law

It is not true that both
Harry and Ron passed the test.

Harry did not pass the test
or Ron did not pass the test.

$$\frac{\neg(\alpha \land \beta)}{\neg\alpha \lor \neg\beta}$$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Distributive Property

$$\frac{(\alpha \land (\beta \lor \gamma))}{(\alpha \land \beta) \lor (\alpha \land \gamma)}$$

$$\frac{(\alpha \lor (\beta \land \gamma))}{(\alpha \lor \beta) \land (\alpha \lor \gamma)}$$

Slide credit : HarvardX CS50AI CS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Search Problem

- initial state
- actions
- transition model
- goal test
- path cost function

Theorem Proving

- initial state: starting knowledge base
- actions: inference rules
- transition model: new knowledge base after inference
- goal test: check statement we're trying to prove
- path cost function: number of steps in proof

Example

Prove S

Step	Formula	Derivation
1	$P \wedge Q$	Given
2	$P \rightarrow R$	Given
3	$(Q \wedge R) \rightarrow S$	Given

Some inference rules:

Modus ponens:

$$\frac{\alpha \rightarrow \beta \quad \alpha}{\beta}$$

Modus tolens:

$$\frac{\alpha \rightarrow \beta \quad \neg \beta}{\neg \alpha}$$

And-introduction:

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

And-elimination:

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

Example

Prove S

Step	Formula	Derivation
1	$P \wedge Q$	Given
2	$P \rightarrow R$	Given
3	$(Q \wedge R) \rightarrow S$	Given
4	P	1 And-Elim

Some inference rules:

$$\frac{\alpha \rightarrow \beta \quad \alpha}{\beta}$$

Modus ponens

$$\frac{\alpha \rightarrow \beta \quad \neg \beta}{\neg \alpha}$$

Modus tolens

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

And-introduction

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

And-elimination

Example

Prove S

Step	Formula	Derivation
1	$P \wedge Q$	Given
2	$P \rightarrow R$	Given
3	$(Q \wedge R) \rightarrow S$	Given
4	P	1 And-Elim
5	R	4,2 Modus Ponens

Some inference rules:

$$\frac{\alpha \rightarrow \beta \quad \alpha}{\beta}$$

Modus ponens

$$\frac{\alpha \rightarrow \beta \quad \neg \beta}{\neg \alpha}$$

Modus tolens

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

And-introduction

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

And-elimination

Example

Prove S

Step	Formula	Derivation
1	$P \wedge Q$	Given
2	$P \rightarrow R$	Given
3	$(Q \wedge R) \rightarrow S$	Given
4	P	1 And-Elim
5	R	4,2 Modus Ponens
6	Q	1 And-Elim

Some inference rules:

$$\frac{\alpha \rightarrow \beta \quad \alpha}{\beta}$$

Modus ponens

$$\frac{\alpha \rightarrow \beta \quad \neg \beta}{\neg \alpha}$$

Modus tolens

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

And-introduction

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

And-elimination

Example

Some inference rules:

Prove S

Step	Formula	Derivation
1	$P \wedge Q$	Given
2	$P \rightarrow R$	Given
3	$(Q \wedge R) \rightarrow S$	Given
4	P	1 And-Elim
5	R	4,2 Modus Ponens
6	Q	1 And-Elim
7	$Q \wedge R$	5,6 And-Intro

$$\frac{\alpha \rightarrow \beta \quad \alpha}{\beta}$$

Modus ponens

$$\frac{\alpha \rightarrow \beta \quad \neg \beta}{\neg \alpha}$$

Modus tolens

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

And-introduction

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

And-elimination

Example

Prove S

Step	Formula	Derivation
1	$P \wedge Q$	Given
2	$P \rightarrow R$	Given
3	$(Q \wedge R) \rightarrow S$	Given
4	P	1 And-Elim
5	R	4,2 Modus Ponens
6	Q	1 And-Elim
7	$Q \wedge R$	5,6 And-Intro
8	S	7,3 Modus Ponens

Some inference rules:

$$\frac{\alpha \rightarrow \beta \quad \alpha}{\beta}$$

Modus ponens

$$\frac{\alpha \rightarrow \beta \quad \neg \beta}{\neg \alpha}$$

Modus tolens

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

And-introduction

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

And-elimination

Logical equivalence

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

Example from Wumpus World

R1: $\neg P_{1,1}$

R2: $\neg B_{1,1}$

R3: $B_{2,1}$

R4: $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

R5: $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

KB = R1 $\wedge$ R2 $\wedge$ R3 $\wedge$ R4 $\wedge$ R5

Prove $\alpha 1 = \neg P_{1,2}$

Example from Wumpus World

R1: $\neg P_{1,1}$

R2: $\neg B_{1,1}$

R3: $B_{2,1}$

R4: $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

R5: $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

R6 : $B_{1,1} \Leftrightarrow (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ Biconditional elimination

Example from Wumpus World

R1: $\neg P_{1,1}$

R2: $\neg B_{1,1}$

R3: $B_{2,1}$

R4: $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

R5: $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

R6 : $B_{1,1} \Leftrightarrow (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ Biconditional elimination

R7 : $((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ And Elimination

Example from Wumpus World

R1: $\neg P_{1,1}$

R2: $\neg B_{1,1}$

R3: $B_{2,1}$

R4: $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

R5: $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

R6 : $B_{1,1} \Leftrightarrow (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ Biconditional elimination

R7 : $((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ And Elimination

R8: $(\neg B_{1,1} \Rightarrow \neg (P_{1,2} \vee P_{2,1}))$ Equivalence for contrapositives

Example from Wumpus World

R1: $\neg P_{1,1}$

R2: $\neg B_{1,1}$

R3: $B_{2,1}$

R4: $B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$

R5: $B_{2,1} \Leftrightarrow (P_{1,1} \lor P_{2,2} \lor P_{3,1})$

R6 : $B_{1,1} \Leftrightarrow (B_{1,1} \Rightarrow (P_{1,2} \lor P_{2,1})) \land ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$ Biconditional elimination

R7 : $((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$ And Elimination

R8: $(\neg B_{1,1} \Rightarrow \neg (P_{1,2} \lor P_{2,1}))$ Equivalence for contrapositives

R9: $\neg (P_{1,2} \lor P_{2,1})$ Modus Ponens with R2 and R8

Example from Wumpus World

R1: $\neg P_{1,1}$

R2: $\neg B_{1,1}$

R3: $B_{2,1}$

R4: $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

R5: $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

R6 : $B_{1,1} \Leftrightarrow (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ Biconditional elimination

R7 : $((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ And Elimination

R8: $(\neg B_{1,1} \Rightarrow \neg (P_{1,2} \vee P_{2,1}))$ Equivalence for contrapositives

R9: $\neg (P_{1,2} \vee P_{2,1})$ Modus Ponens with R2 and R8

R10: $\neg P_{1,2} \wedge \neg P_{2,1}$ De Morgan's Rule

Example from Wumpus World

R1: $\neg P_{1,1}$

R2: $\neg B_{1,1}$

R3: $B_{2,1}$

R4: $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

R5: $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

R6 : $B_{1,1} \Leftrightarrow (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ Biconditional elimination

R7 : $((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ And Elimination

R8: $(\neg B_{1,1} \Rightarrow \neg (P_{1,2} \vee P_{2,1}))$ Equivalence for contrapositives

R9: $\neg (P_{1,2} \vee P_{2,1})$ Modus Ponens with R2 and R8

R10: $\neg P_{1,2} \wedge \neg P_{2,1}$ De Morgan's Rule

R11: $\neg P_{1,2}$ And Elimination

Monotonicity

- The set of entailed sentences can only increase as information is added to the knowledge base

- If
- KB $\models$ α
- Then
- KB $\wedge$ β $\models$ α

Inference

- $KB \vdash_i \alpha$ = sentence α can be derived from *KB* by a procedure *i* (an inference algorithm)
- Soundness: *i* is sound if whenever $KB \vdash_i \alpha$, it is also true that $KB \models \alpha$
- An inference algorithm that derives only entailed sentences is sound or truth preserving (model checking is a sound procedure)
  - If the system proves that something is true, then it is really true. The system doesn't derive contradictions
- Completeness: *i* is complete if whenever $KB \models \alpha$, it is also true that $KB \vdash_i \alpha$
- An inference algorithm is complete if it can derive any sentence that is entailed
  - If something is really true, it can be proven using the system. The system can be used to derive all the true statements one by one
- If KB is true in the real world then any sentence α derived from KB by a sound inference procedure is also true in real world
  - The conclusions of the reasoning process are guaranteed to be true in any world in which the premises are true

Inference

All consequences of a KB is a haystack
α is a needle
Entailment
The needle being in the haystack
Inference
Finding the needle

- An unsound inference procedure essentially makes things up as it goes along – it announces the discovery of non-existent needles

- For completeness, a systematic examination can always decide whether the needle is in the haystack which is finite

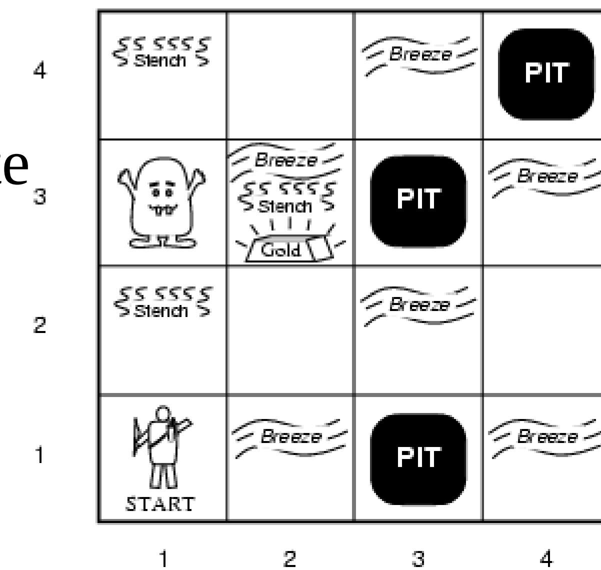
Cartoon illustrations by John P. Weiss

Semantics and Inference

- Interpretation: establishing a correspondence between sentences and facts

- Compositional: the meaning of a sentence is a function of the meaning of its parts

- A sentence is TRUE under a particular inte f the affairs it represents is the case



- $S_{1,2}$ would be true
  - under the interpretation

    "there is a stench in [1,2]"
  - on this Wumpus world

Validity and satisfiability

A sentence is valid if it is true in all models,

e.g. “There is a stench at [1,1] or there is not a stench at [1,1]”

Valid sentences are also called as tautologies

Every valid sentence is equivalent to *True*

A sentence is satisfiable if it is true in some model

e.g. “There is a Wumpus at [1,2]”

If a sentence is true in a model m, then we say m satisfies the sentence, or a model of the sentence

A sentence is unsatisfiable if it is true in no models

e.g. “There is a wall in front of me and there is no wall in front of me”

Truth tables for connectives

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

Most sentences are sometimes true:

e.g. $P \wedge Q$

Some sentences are always true (valid)

e.g. $\neg P \vee P$

Some sentences are never true (unsatisfiable)

e.g. $\neg P \wedge P$

Validity and Inference

Validity and inference cont.

P	H	$P \vee H$	$(P \vee H) \wedge \neg H$	$((P \vee H) \wedge \neg H) \Rightarrow P$
False	*False*	*False*	*False*	*True*
False	*True*	*True*	*False*	*True*
True	*False*	*True*	*True*	*True*
True	*True*	*True*	*False*	*True*

Figure 6.10 Truth table showing validity *of* a complex sentence

Russell Stuart, Norvig Peter, Artificial Intelligence: A Modern Approach, 1995

Logical equivalence

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

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

Examples

Sentence	Valid?	Interpretation that make sentence's truth value = **f**
smoke → smoke	valid	
smoke ∨ ¬smoke	valid	
smoke → fire	satisfiable, not valid	smoke = **t**, fire = **f**
(s → f) → (¬ s → ¬ f)	satisfiable, not valid	s = **f**, f = **t** s → f = **t**, ¬ s → ¬ f = **f**
contrapositive (s → f) → (¬ f → ¬ s)	valid	
b ∨ d ∨ (b → d)	valid	
b ∨ d ∨ ¬ b ∨ d	valid	

Satisfiability

- Related to constraint satisfaction
- Given a sentence S, try to find an interpretation i where S is true
- Analogous to finding an assignment of values to variables such that the constraint hold
- Example problem: scheduling nurses in a hospital
  - Propositional variables represent for example that Nurse1 is working on Tuesday at 2
  - Constraints on the schedule are represented using logical expressions over the variables
- Brute force method: enumerate all interpretations and check

Validity and Inference

premises $\Rightarrow$ conclusion

Validity and inference cont.

P	H	$P \vee H$	$(P \vee H) \wedge \neg H$	$((P \vee H) \wedge \neg H) \Rightarrow P$
False	*False*	*False*	*False*	*True*
False	*True*	*True*	*False*	*True*
True	*False*	*True*	*True*	*True*
True	*True*	*True*	*False*	*True*

Figure 6.10 Truth table showing validity *of* a complex sentence

Russell Stuart, Norvig Peter, Artificial Intelligence: A Modern Approach, 1995

Recall

- Logical Inference creates new sentences that logically follow from a set of sentences in KB

- An inference rule is sound if every sentence X it produces when operating on a KB logically follows from the KB

- An inference rule is complete if it can produce every expression that logically follows from (is entailed by) the KB

Resolution

(Ron is in the Great Hall) v (Hermione is in the library)

Ron is not in the Great Hall

Hermione is in the library

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

$$\frac{P \lor Q \qquad \neg P}{Q}$$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

$$P \lor Q_1 \lor Q_2 \lor \ldots \lor Q_n$$

$$\neg P$$

$$Q_1 \lor Q_2 \lor \ldots \lor Q_n$$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

(Ron is in the Great Hall) v (Hermione is in the library)

(Ron is not in the Great Hall) v (Harry is sleeping)

(Hermione is in the library) v (Harry is sleeping)

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

$$\frac{P \vee Q \qquad \neg P \vee R}{Q \vee R}$$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

$$P \lor Q_1 \lor Q_2 \lor \ldots \lor Q_n$$

$$\neg P \lor R_1 \lor R_2 \lor \ldots \lor R_m$$

$$\overline{}$$

$$Q_1 \lor Q_2 \lor \ldots \lor Q_n \lor R_1 \lor R_2 \lor \ldots \lor R_m$$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

clause

a disjunction of literals

e.g. $P \vee Q \vee R$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

conjunctive normal form

logical sentence that is a conjunction of clauses

e.g. $(A \lor B \lor C) \land (D \lor \neg E) \land (F \lor G)$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

Conversion to CNF

- Eliminate biconditionals
  - turn $(\alpha \leftrightarrow \beta)$ into $(\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$
- Eliminate implications
  - turn $(\alpha \rightarrow \beta)$ into $\neg\alpha \vee \beta$
- Move $\neg$ inwards using De Morgan's Laws
  - e.g. turn $\neg(\alpha \wedge \beta)$ into $\neg\alpha \vee \neg\beta$
- Use distributive law to distribute $\vee$ wherever possible

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

Conversion to CNF

$(P \lor Q) \rightarrow R$

$\neg(P \lor Q) \lor R$ — eliminate implication

$(\neg P \land \neg Q) \lor R$ — De Morgan's Law

$(\neg P \lor R) \land (\neg Q \lor R)$ — distributive law

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

$$P \vee Q$$

$$\neg P \vee R$$

$$(Q \vee R)$$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

$$P \lor Q \lor S$$

$$\neg P \lor R \lor S$$

$$(Q \lor S \lor R \lor S)$$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

$$P \lor Q \lor S$$

$$\neg P \lor R \lor S$$

$$(Q \lor R \lor S)$$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

- Resolution is a sound and complete inference procedure

$$\frac{\alpha \vee \beta, \neg \beta \vee \gamma}{\alpha \vee \gamma}$$

- If β is True, since we know that $\neg \beta \vee \gamma$ holds, it must be the case that γ is true
- If β is false, then since we know that $\alpha \vee \beta$ holds, it must be the case that α is true
- So either α or γ is true

Resolution

Resolution

$$\frac{\alpha \vee \beta, \neg \beta \vee \gamma}{\alpha \vee \gamma}$$

or equivalently

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

$$\frac{\alpha \vee \beta, \beta \Rightarrow \gamma}{\alpha \vee \gamma}$$

Example:

α: "The weather is dry"

β: "The weather is rainy"

γ: "I carry an umbrella"

Soundness of the resolution inference rules

- An inference rule is sound if the conclusion is true in all cases where the premises are true

Rules of inference for propositional logic

α	β	γ	$\alpha \vee \beta$	$\neg\beta \vee \gamma$	$\alpha \vee \gamma$
False	False	False	False	True	False
False	False	True	False	True	True
False	True	False	True	False	False
False	True	True	True	True	True
True	False	False	True	True	True
True	False	True	True	True	True
True	True	False	True	False	True
True	True	True	True	True	True

Figure 6.14 A truth table demonstrating the soundness of the resolution inference rule. We have underlined the rows where both premises are true.

Russell Stuart, Norvig Peter, Artificial Intelligence: A Modern Approach, 1995

Generalized resolution rule

- $P_1 \vee P_2 \vee \ldots \vee P_n$
- $\neg P_1 \vee Q_2 \vee \ldots \vee Q_m$
- Resolvent: $P_2 \vee \ldots \vee P_n \vee Q_2 \vee \ldots \vee Q_m$

- Where P1 and ¬ P1 are complementary literals

Proof with resolution

Given the following hypotheses:

1. If it rains, Joe brings his umbrella
   (r -> u)

2. If Joe has an umbrella, he doesn't get wet
   (u -> NOT w)

3. If it doesn't rain, Joe doesn't get wet
   (NOT r -> NOT w)

prove that Joes doesn't get wet (NOT w)

Proof with resolution

We first convert each hypothesis into disjunctions

1. r -> u

   (NOT r OR u)

2. u -> NOT w

   (NOT u OR NOT w)

3. NOT r -> NOT w

   (r OR NOT w)

Proof with resolution

We then use resolution on the hypotheses to derive the conclusion (NOT w):

1. NOT r OR u Premise
2. NOT u OR NOT w Premise
3. r OR NOT w Premise
4. NOT r OR NOT w L1, L2, resolution
5. NOT w OR NOT w L3, L4, resolution
6. NOT w L5, idempotence

Resolution covers many cases

- Modes Ponens
  - from P and $P \rightarrow Q$ derive Q
  - from P and $\neg P \vee Q$ derive Q
- Chaining
  - from $P \rightarrow Q$ and $Q \rightarrow R$ derive $P \rightarrow R$
  - from $(\neg P \vee Q)$ and $(\neg Q \vee R)$ derive $\neg P \vee R$
- Contradiction detection
  - from P and $\neg P$ derive false
  - from P and $\neg P$ derive the empty clause (=false)

Rewrite

$P \equiv P \vee \text{False}$

$\neg P \equiv \text{False} \vee \neg P$

Apply resolution (P and ¬ P are complementary literals)

$P \vee \text{False}$

$\text{False} \vee \neg P$

$\text{False} \vee \text{False}$

$\text{False} \vee \text{False} \equiv \text{False}$

Properties of the resolution rule:

- Sound
- Complete (yields to a complete inference algorithm).

The resolution rule forms the basis for a family of complete inference algorithms.

Resolution rule is used to either confirm or refute a sentence but it cannot be used to enumerate true sentences.

Resolution

$$P$$

$$\neg P$$

$$(\;)$$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

Inference by Resolution

- To determine if $\mathrm{KB} \vDash \alpha$:
  - Check if $(\mathrm{KB} \land \neg\alpha)$ is a contradiction?
    - If so, then $\mathrm{KB} \vDash \alpha$.
    - Otherwise, no entailment.

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

Inference by Resolution

- To determine if $\mathrm{KB} \vDash \alpha$:
  - Convert $(\mathrm{KB} \wedge \neg\alpha)$ to Conjunctive Normal Form.
  - Keep checking to see if we can use resolution to produce a new clause.
    - If ever we produce the **empty** clause (equivalent to False), we have a contradiction, and $\mathrm{KB} \vDash \alpha$.
    - Otherwise, if we can't add new clauses, no entailment.

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

Inference by Resolution

Does $(A \lor B) \land (\neg B \lor C) \land (\neg C)$ entail A?

$$(A \lor B) \land (\neg B \lor C) \land (\neg C) \land (\neg A)$$

$(A \lor B)$ $(\neg B \lor C)$ $(\neg C)$ $(\neg A)$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

Inference by Resolution

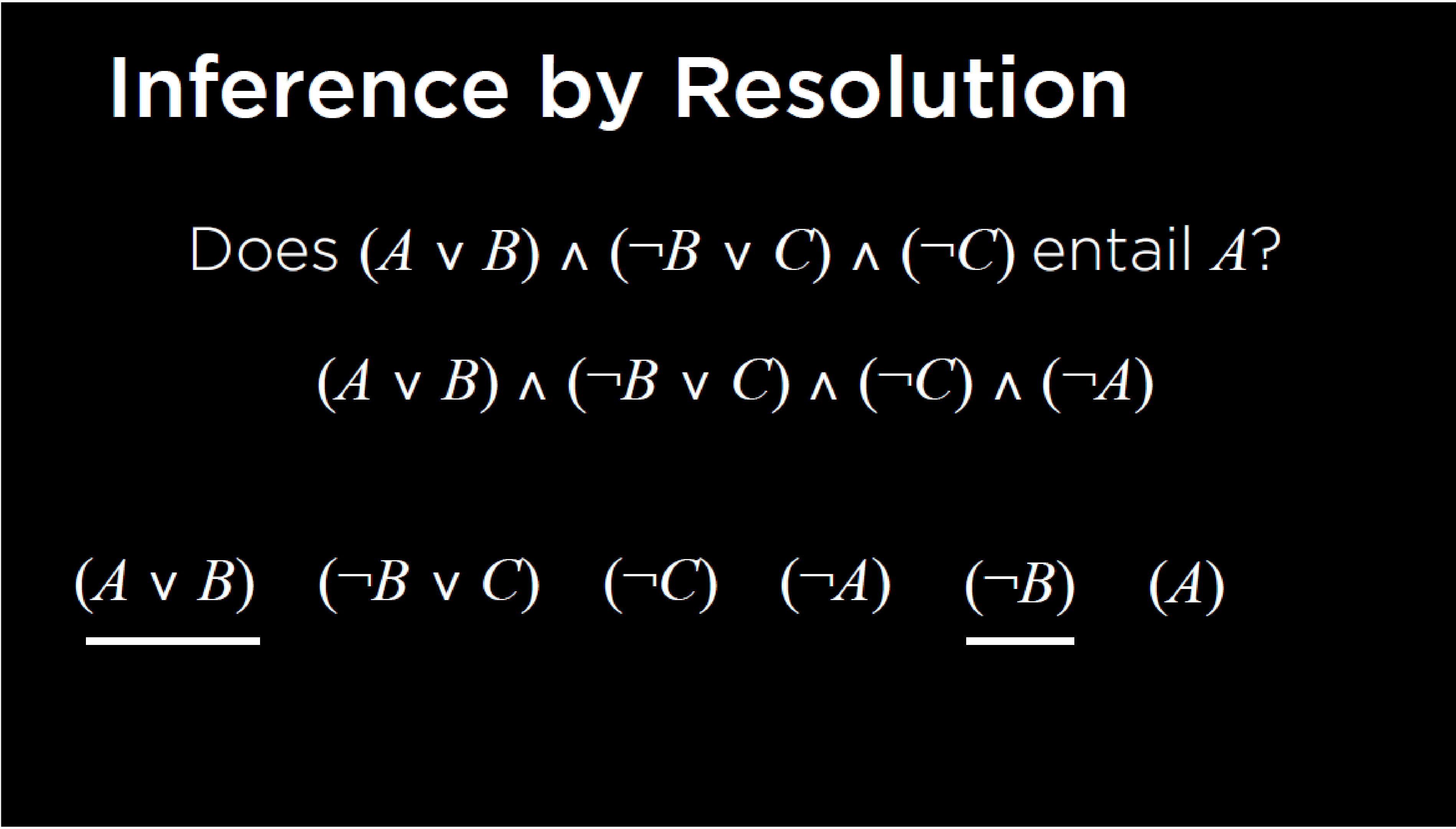
Does $(A \vee B) \wedge (\neg B \vee C) \wedge (\neg C)$ entail A?

$$(A \vee B) \wedge (\neg B \vee C) \wedge (\neg C) \wedge (\neg A)$$

$(A \vee B)$ $\quad$ $\underline{(\neg B \vee C)}$ $\quad$ $\underline{(\neg C)}$ $\quad$ $(\neg A)$ $\quad$ $(\neg B)$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution



Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Resolution

Inference by Resolution

Does $(A \lor B) \land (\neg B \lor C) \land (\neg C)$ entail A?

$$(A \lor B) \land (\neg B \lor C) \land (\neg C) \land (\neg A)$$

$(A \lor B)$ $(\neg B \lor C)$ $(\neg C)$ $\underline{(\neg A)}$ $(\neg B)$ $\underline{(A)}$ $(\,)$

Slide credit : HarvardX CS50AICS50's Introduction to Artificial Intelligence with Python, David J. Malan and Brian Yu

Recall

A sentence is valid if it is true in all models,

$KB \models \alpha$ if and only if ($KB \Rightarrow \alpha$) is valid

A sentence is unsatisfiable if it is true in no models

α is valid iff $\neg\alpha$ is unsatisfiable,

($KB \Rightarrow \alpha$) is equivalent to ($\neg$KB $\vee$ α)

$\neg$ ($KB \Rightarrow \alpha$) is equivalent to (KB $\wedge \neg\alpha$)

$KB \models \alpha$ if and only if ($KB \wedge \neg\alpha$) is unsatisfiable

Inference procedures based on resolution work by using the principle of proof by contradiction:

To show that $KB \models \alpha$ we show that $(KB \wedge \neg\alpha)$ is unsatisfiable

Proof by contradiction using resolution

1. NOT r OR u Premise
2. NOT u OR NOT w Premise
3. r OR NOT w Premise
4. w Negation of conclusion
5. NOT r OR NOT w L1, L2, resolution
6. NOT w OR NOT w L3, L5, resolution
7. NOT w L6, idempotence
8. FALSE L4, L7, resolution

Resolution can be applied only to disjunctions of literals. How can it lead to a complete inference procedure for all propositional logic?

Turns out any knowledge base can be expressed as a conjunction of disjunctions (conjunctive normal form, CNF).

E.g., $(A \lor \neg B) \land (B \lor \neg C \lor \neg D)$

Conjunctive Normal form

- Conjunctive normal form (CNF) formulas:

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

  - $(A \vee B \vee \neg C)$ is a clause, which is a disjunction of literals
  - A, B, and $\neg$ C are literals, each of which is a variable or the negation of a variable.
  - Each clause is a requirement that must be satisfied and can be satisfied in multiple ways
  - Every sentence in propositional logic can be written in CNF

Converting to CNF

1. Eliminate arrows using definitions
2. Drive in negations using De Morgan's Laws

$$\neg(\phi \vee \varphi) \equiv \neg\phi \wedge \neg\varphi$$
$$\neg(\phi \wedge \varphi) \equiv \neg\phi \vee \neg\varphi$$

3. Distribute **or** over **and**

$$A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$$

4. Every sentence can be converted to CNF, but it may grow exponentially in size

CNF Conversion Example

$$(A \vee B) \rightarrow (C \rightarrow D)$$

1. Eliminate arrows

$$\neg(A \vee B) \vee (\neg C \vee D)$$

2. Drive in negations

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

3. Distribute

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

Resolution

- Resolution rule:

$$\frac{\alpha \vee \beta \qquad \neg\beta \vee \gamma}{\alpha \vee \gamma}$$

- Resolution refutation:
  - Convert all sentences to CNF
  - Negate the desired conclusion (converted to CNF)
  - Apply resolution rule until either
    - Derive false (a contradiction)
    - Can't apply any more
- Resolution refutation is sound and complete
  - If we derive a contradiction, then the conclusion follows from the axioms
  - If we can't apply any more, then the conclusion cannot be proved from the axioms.

Example

Prove R

1	P v Q
2	P → R
3	Q → R

Step	Formula	Derivation
1	P v Q	Given
2	¬ P v R	Given
3	¬ Q v R	Given
4	¬ R	Negated conclusion

Example

Prove R

1	P ∨ Q
2	P → R
3	Q → R

Step	Formula	Derivation
1	P ∨ Q	Given
2	¬ P ∨ R	Given
3	¬ Q ∨ R	Given
4	¬ R	Negated conclusion
5	Q ∨ R	1,2
6	¬ P	2,4
7	¬ Q	3,4
8	R	5,7
9	•	4,8

Example

Prove R

1	P v Q
2	P → R
3	Q → R

false v R

¬ R v false

false v false

Step	Formula	Derivation
1	P v Q	Given
2	¬ P v R	Given
3	¬ Q v R	Given
4	¬ R	Negated conclusion
5	Q v R	1,2
6	¬ P	2,4
7	¬ Q	3,4
8	R	5,7
9	•	4,8

The power of false

Prove Z

1	P
2	¬ P

Step	Formula	Derivation
1	P	Given
2	¬ P	Given
3	¬ Z	Negated conclusion
4	•	1,2

Note that $(P \wedge \neg P) \rightarrow Z$ is valid

Any conclusion follows from a contradiction – and so strict logic systems are very brittle.

Example

Prove R

1	$(P \to Q) \to Q$
2	$(P \to P) \to R$
3	$(R \to S) \to \neg(S \to Q)$

Convert to CNF

- $\neg(\neg P \lor Q) \lor Q$
- $(P \land \neg Q) \lor Q$
- $(P \lor Q) \land (\neg Q \lor Q)$
- $(P \lor Q)$

- $\neg(\neg P \lor P) \lor R$
- $(P \land \neg P) \lor R$
- $(P \lor R) \land (\neg P \lor R)$

- $\neg(\neg R \lor S) \lor \neg(\neg S \lor Q)$
- $(R \land \neg S) \lor (S \land \neg Q)$
- $(R \lor S) \land (\neg S \lor S) \land (R \lor \neg Q) \land (\neg S \lor \neg Q)$
- $(R \lor S) \land (R \lor \neg Q) \land (\neg S \lor \neg Q)$

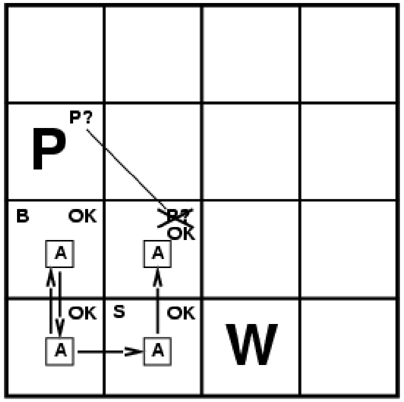
Example

Prove R

1	(P → Q) → Q
2	(P → P) → R
3	(R → S) → ¬ (S → Q)

1	P v Q	
2	P v R	
3	¬ P v R	
4	R v S	
5	R v ¬ Q	
6	¬ S v ¬ Q	
7	¬ R	Neg
8	S	4,7
9	¬ Q	6,8
10	P	1,9
11	R	3,10
12	•	7,11

Conversion to CNF



Givem

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

$\neg B_{1,1}$

Prove: $P_{1,2}$

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

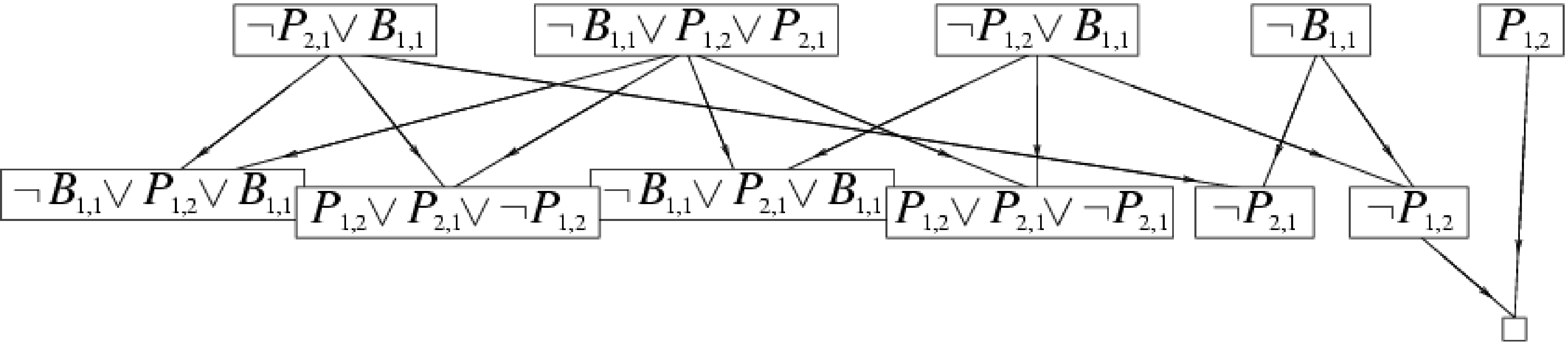
2. Eliminate $\Rightarrow$, replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \lor \beta$.
   $(\neg B_{1,1} \lor P_{1,2} \lor P_{2,1}) \land (\neg(P_{1,2} \lor P_{2,1}) \lor B_{1,1})$

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

4. Apply distributivity law ($\land$ over $\lor$) and flatten:
   $(\neg B_{1,1} \lor P_{1,2} \lor P_{2,1}) \land (\neg P_{1,2} \lor B_{1,1}) \land (\neg P_{2,1} \lor B_{1,1})$

Resolution example

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



Resolution algorithm

- Proof by contradiction, i.e., show $KB \wedge \neg \alpha$ unsatisfiable

function PL-RESOLUTION(KB, α) **returns** *true* or *false*

    clauses ← the set of clauses in the CNF representation of KB ∧ ¬α
    new ← { }
    loop do
        for each 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

Horn Clauses

- Horn Form (restricted)
  KB = conjunction of Horn clauses
  Horn clause =
  proposition symbol (conjunction of symbols) $\Rightarrow$ symbol

  $P1 \wedge P2 \wedge \ldots\ldots \wedge Pn \Rightarrow Q$
  where Pi and Q are nonnegated atoms
  – E.g., $C \wedge (B \Rightarrow A) \wedge (C \wedge D \Rightarrow B)$

- When Q is False we get a sentence that is equivalent to
  $\neg P1 \vee \neg P2 \vee \ldots\ldots \vee \neg Pn$
- When n=1 and p1 is True we get True $\Rightarrow$ Q
  which is equivalent to Q

Forward and backward chaining

- Modus Ponens (for Horn Form): complete for Horn KBs

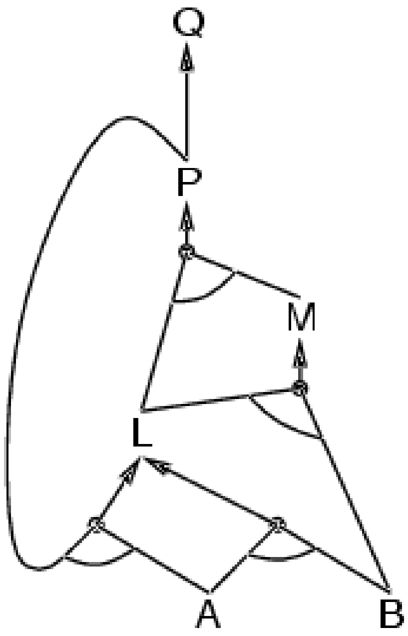
$$\frac{\alpha_1, \ldots, \alpha_n, \qquad \alpha_1 \wedge \ldots \wedge \alpha_n \Rightarrow \beta}{\beta}$$

- Can be used with forward chaining or backward chaining.
- These algorithms are very natural and run in linear time

Forward chaining

- Idea: fire any rule whose premises are satisfied in the *KB*,
  – add its conclusion to the *KB*, until query is found

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Forward chaining algorithm

```
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
```

- Fo

Forward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

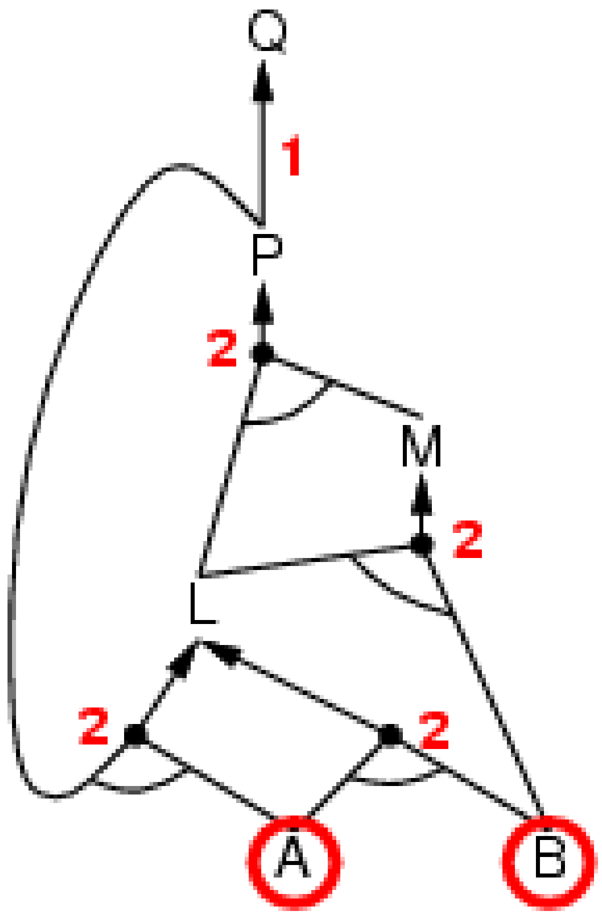
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

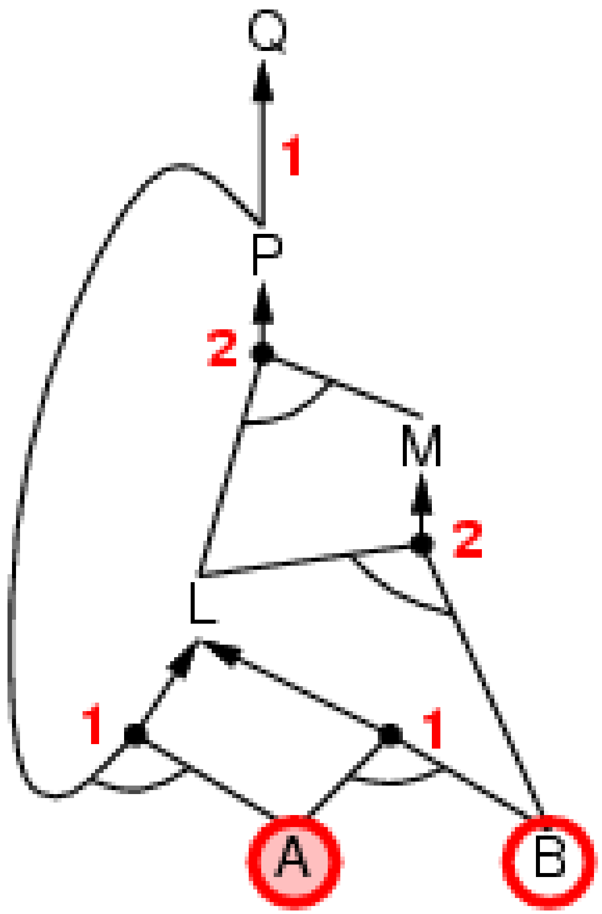
A

B



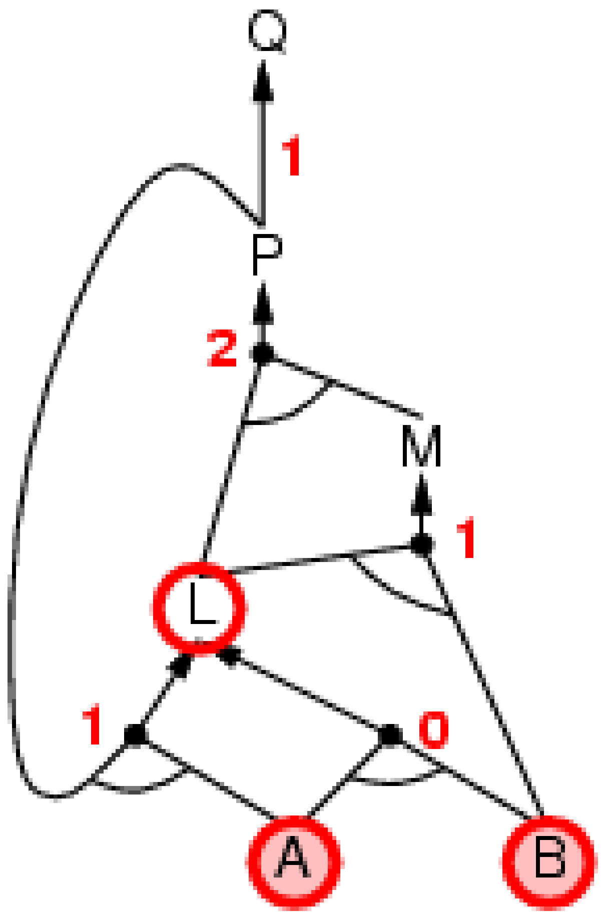
Forward chaining example

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Forward chaining example

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B



Forward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

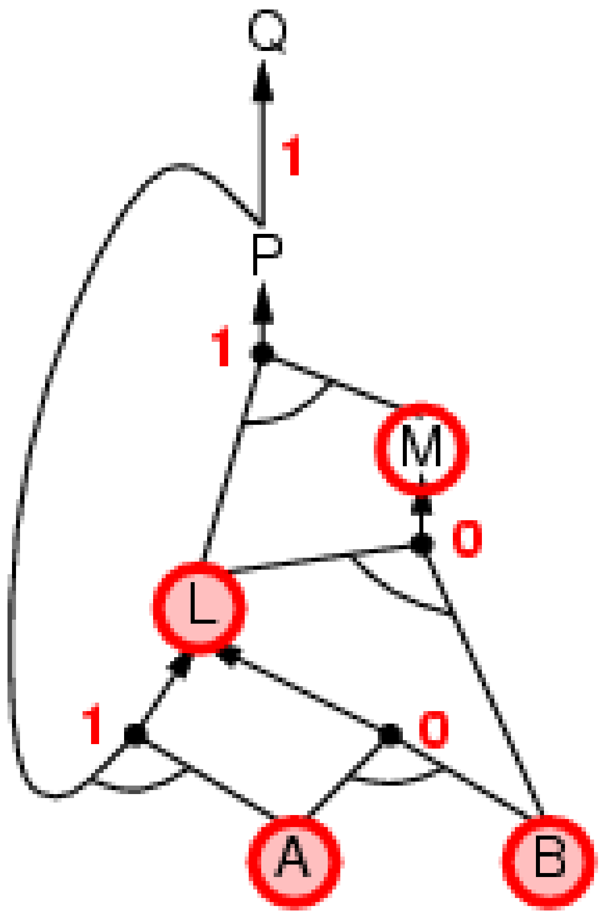
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Forward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

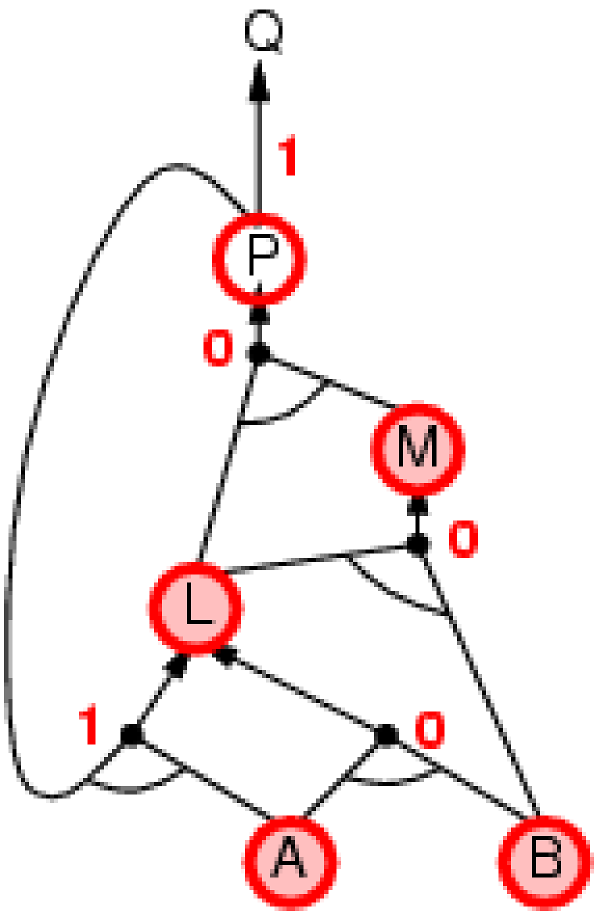
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

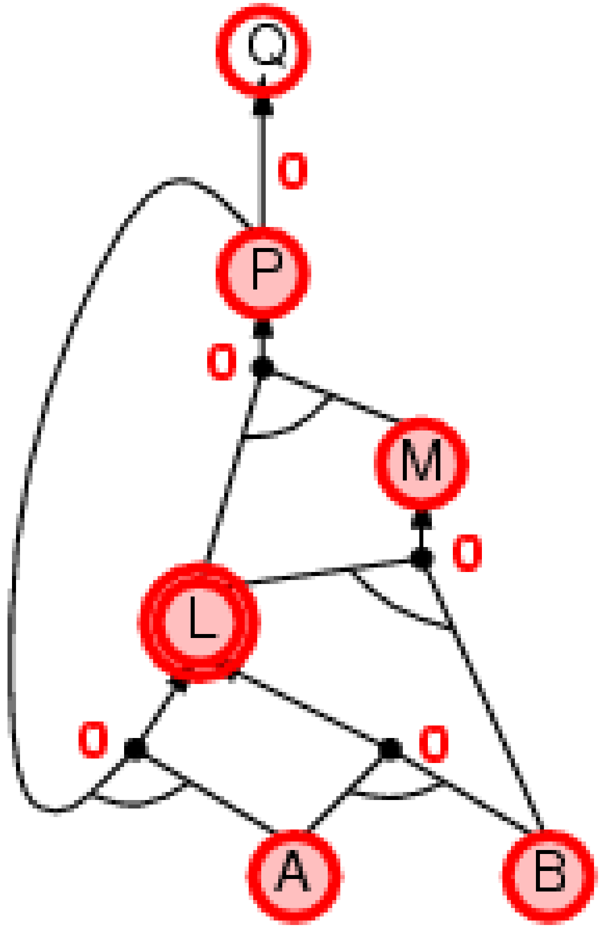
A

B



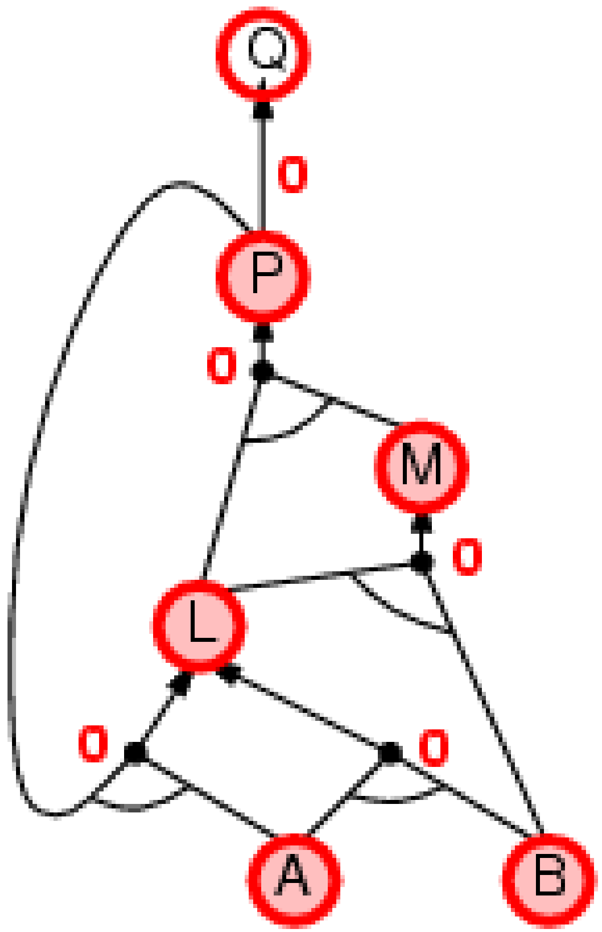
Forward chaining example

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B



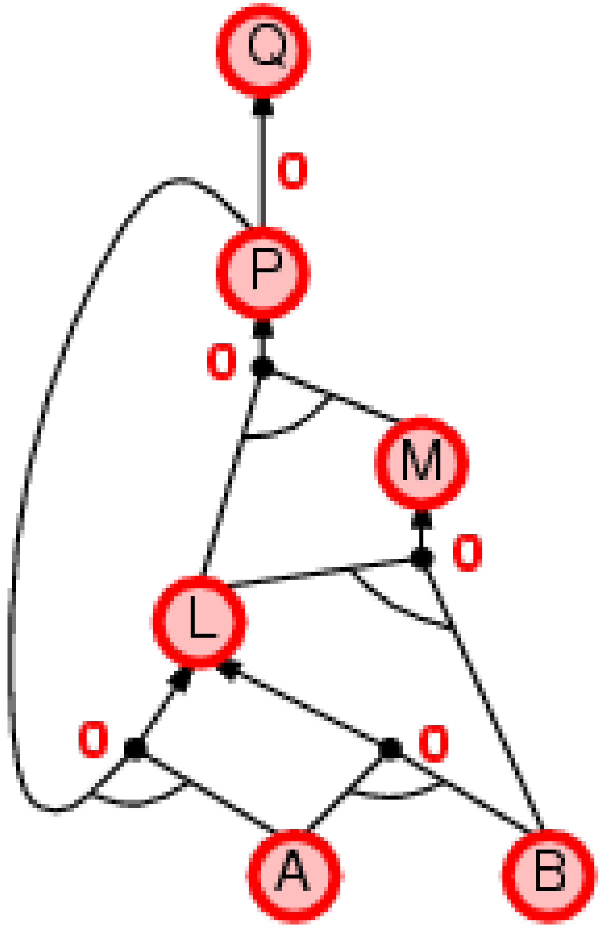
Forward chaining example

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Forward chaining example

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Proof of completeness

- FC derives every atomic sentence that is entailed by *KB*
  1. FC reaches a fixed point where no new atomic sentences are derived
  2. Consider the final state as a model *m*, assigning true/false to symbols
  3. Every clause in the original *KB* is true in *m*

     $a_1 \wedge \ldots \wedge a_k \Rightarrow b$
  4. Hence *m* is a model of *KB*
  5. If $KB \models q$, *q* is true in every model of *KB*, including *m*

Backward chaining

Idea: work backwards from the query q:

to prove q by BC,

check if q is known already, or

prove by BC all premises of some rule concluding q

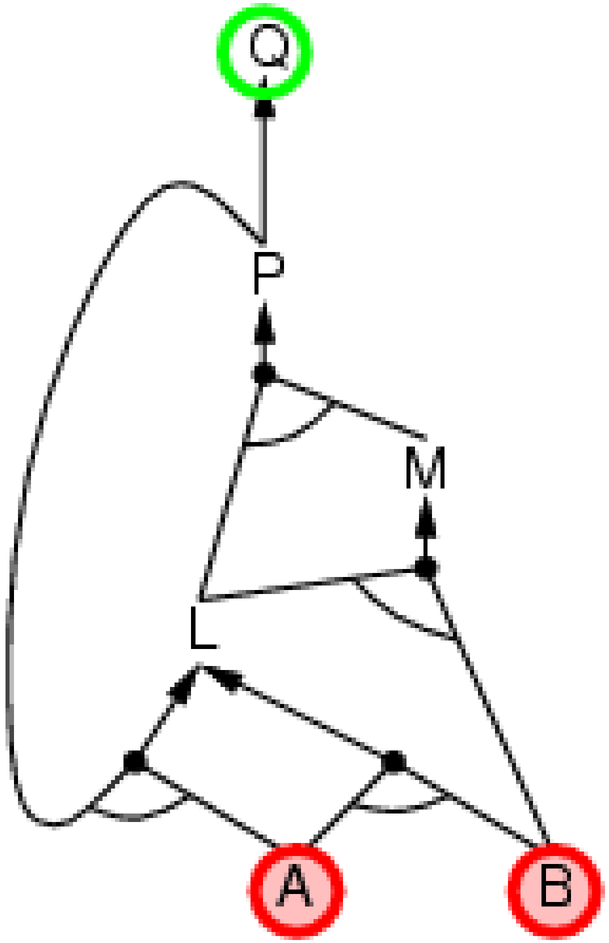
Avoid loops: check if new subgoal is already on the goal stack

Avoid repeated work: check if new subgoal

1. has already been proved true, or
2. has already failed

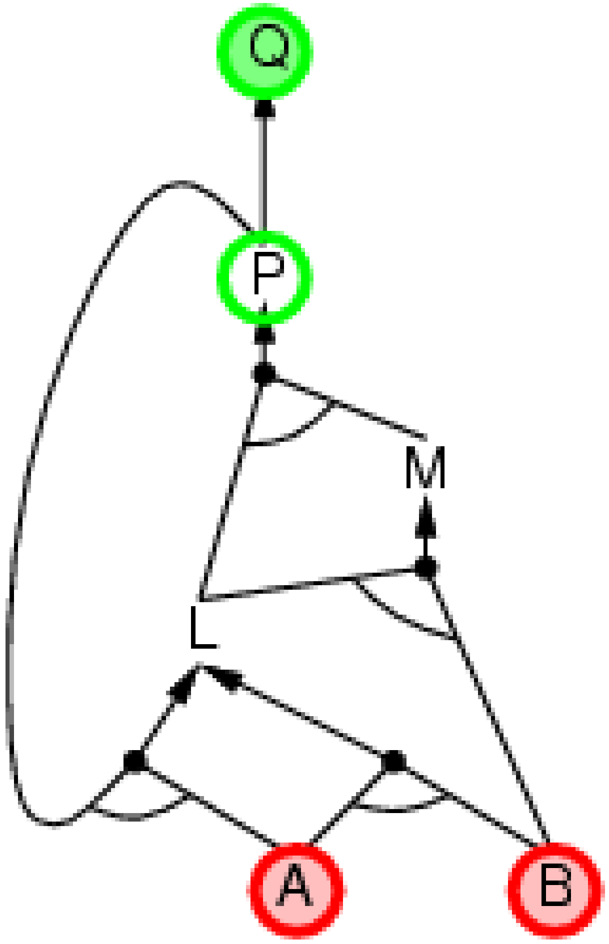
Backward chaining example

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B



Backward chaining example

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B



Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

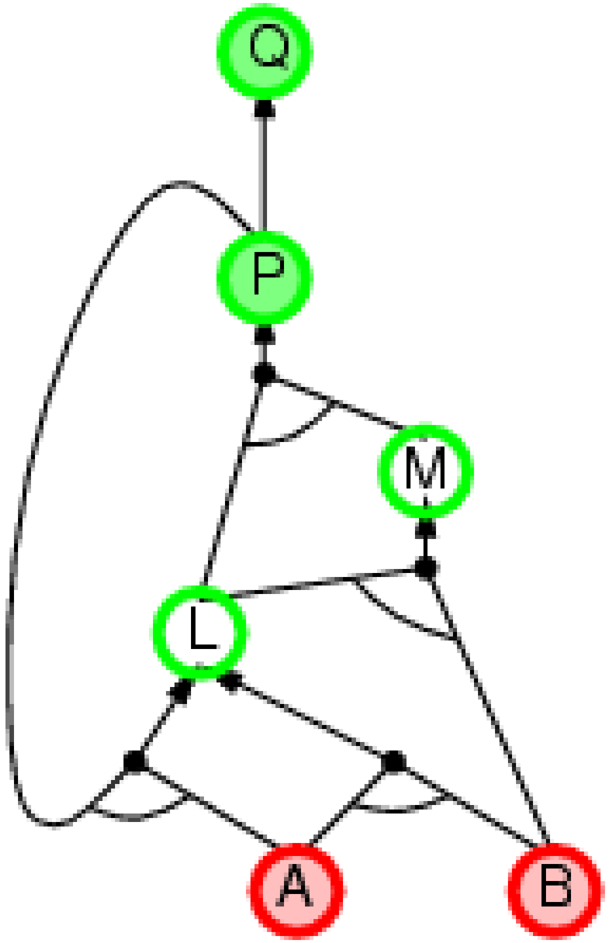
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

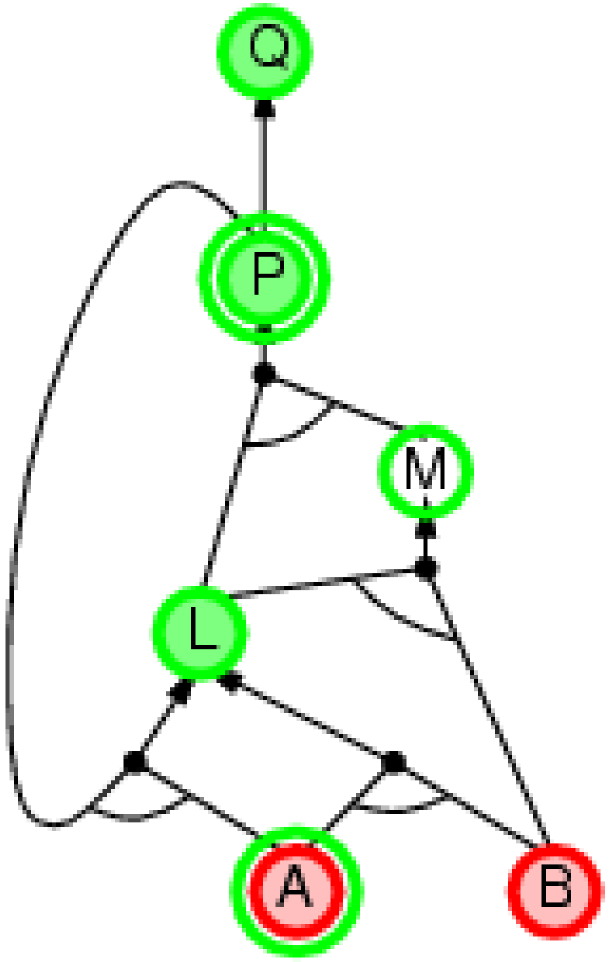
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

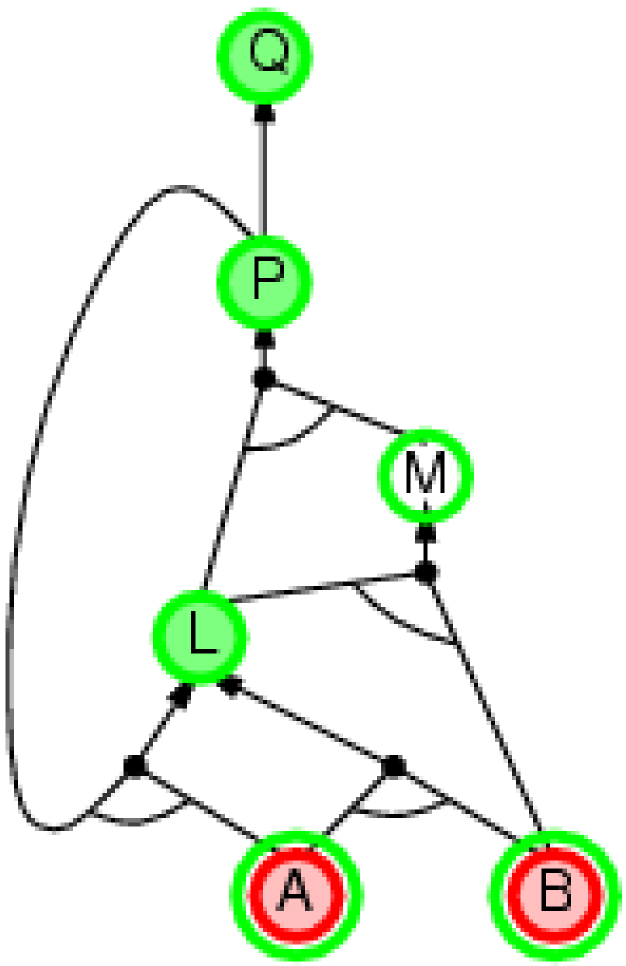
A

B



Backward chaining example

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B



Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

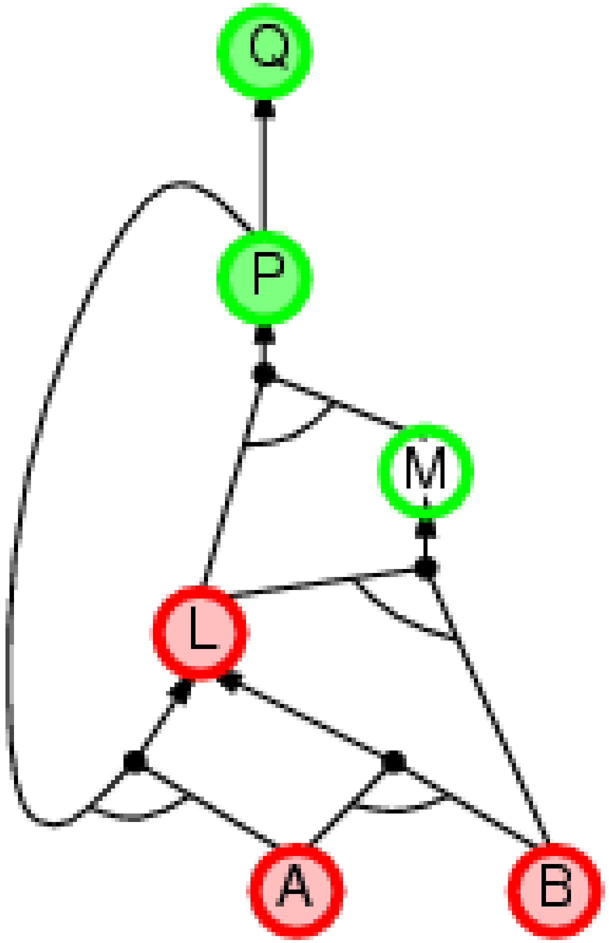
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

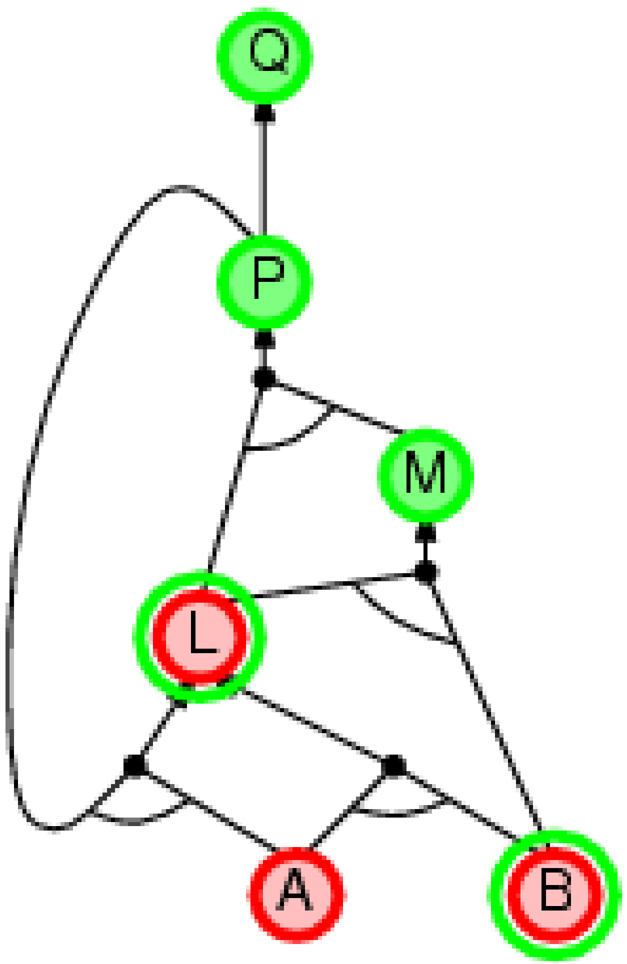
A

B



Backward chaining example

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

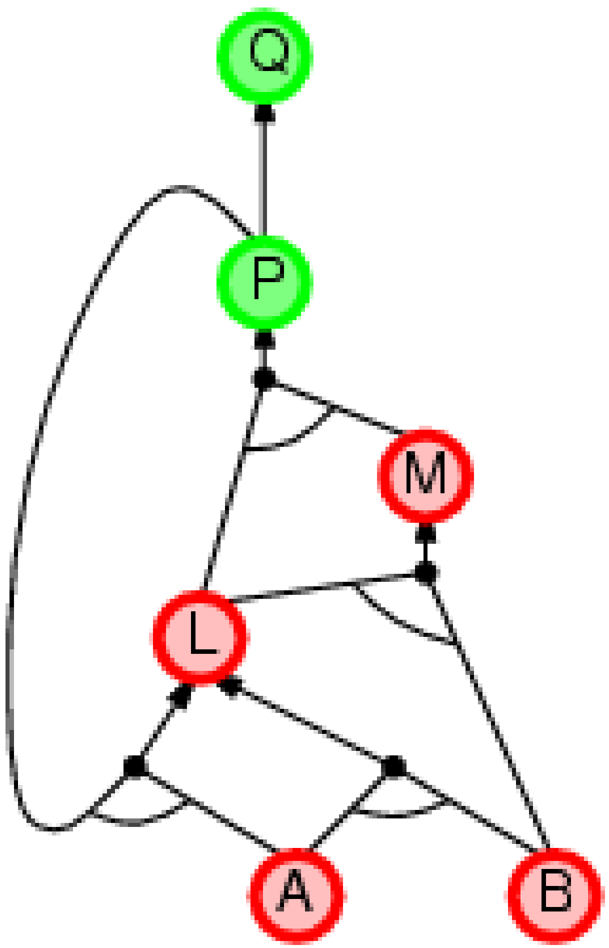
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

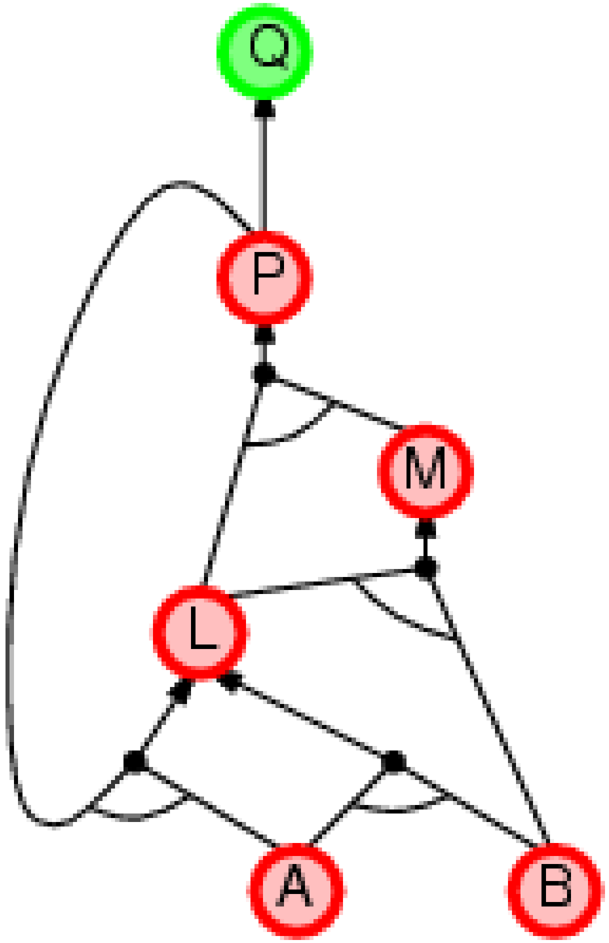
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

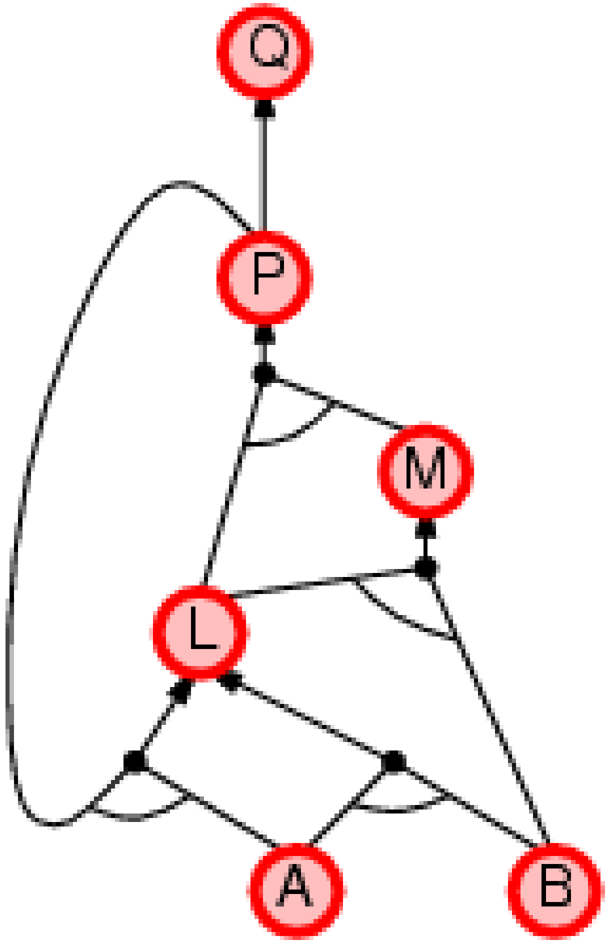
A

B



Backward chaining example

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B



Forward vs. backward chaining

- FC is data-driven, automatic, unconscious processing,
  – e.g., object recognition, routine decisions
- May do lots of work that is irrelevant to the goal
- BC is goal-driven, appropriate for problem-solving,
  – e.g., Where are my keys? How do I get into a PhD program?
- Complexity of BC can be much less than linear in size of KB

Summary

- Logical agents apply inference to a knowledge base to derive new information and make decisions
- Basic concepts of logic:
  - syntax: formal structure of sentences
  - semantics: truth of sentences wrt models
  - entailment: necessary truth of one sentence given another
  - inference: deriving sentences from other sentences
  - soundness: derivations produce only entailed sentences
  - completeness: derivations can produce all entailed sentences
- Wumpus world requires the ability to represent partial and negated information, reason by cases, etc.
- Resolution is complete for propositional logic
  Forward, backward chaining are linear-time, complete for Horn clauses
- Propositional logic lacks expressive power

Proof methods

- 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

- 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

Resolution as a complete inference

- To prove KB $\models$ α, assume KB $\wedge \neg$ α and derive a contradiction
- Rewrite KB $\wedge \neg$ α as a conjunction of *clauses*, or disjunctions of *literals*
  - *Conjunctive normal form* (CNF)
- Keep applying resolution to clauses that contain complementary literals and adding resulting clauses to the list
  - If there are no new clauses to be added, then KB does not entail α
  - If two clauses resolve to form an empty clause, we have a contradiction and KB $\models$ α

The process: 1. convert KB $\wedge \neg$α to CNF

2. resolution rule is applied to the resulting clauses.