IT5005 Artificial Intelligence

Sirigina Rajendra Prasad
AY2025/2026: Semester 1

Logical Agents

Recap

Logical Connectives

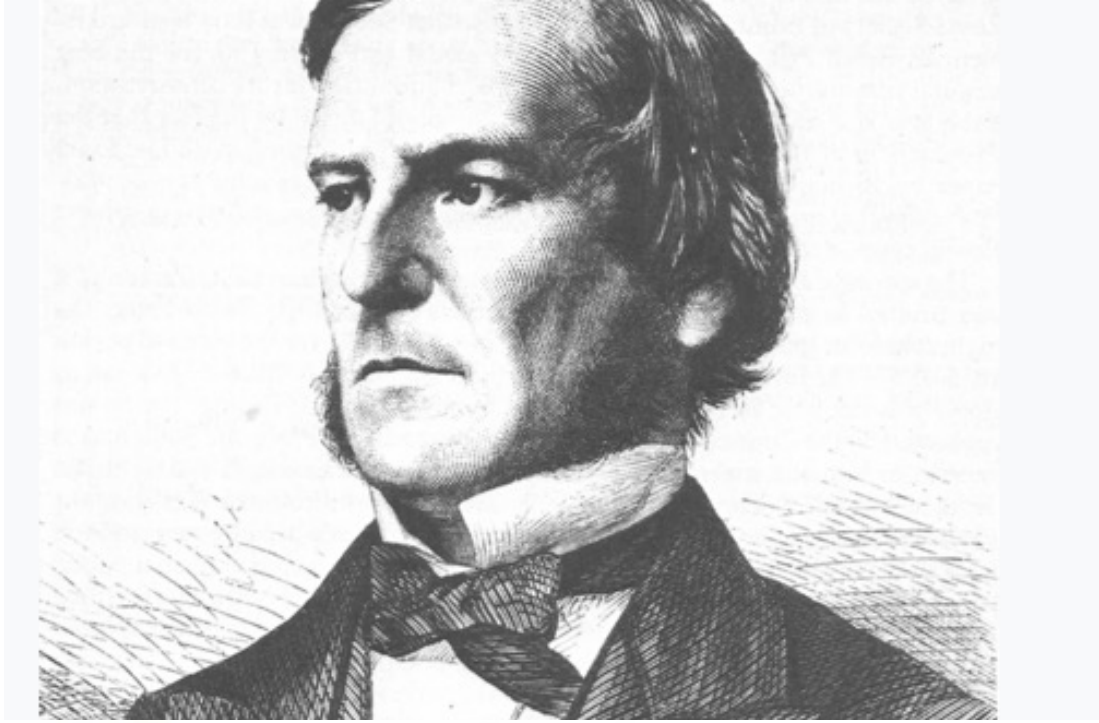
$P \Longrightarrow Q$ is true means
if P is true, then Q is true
if P is false, no claim on Q

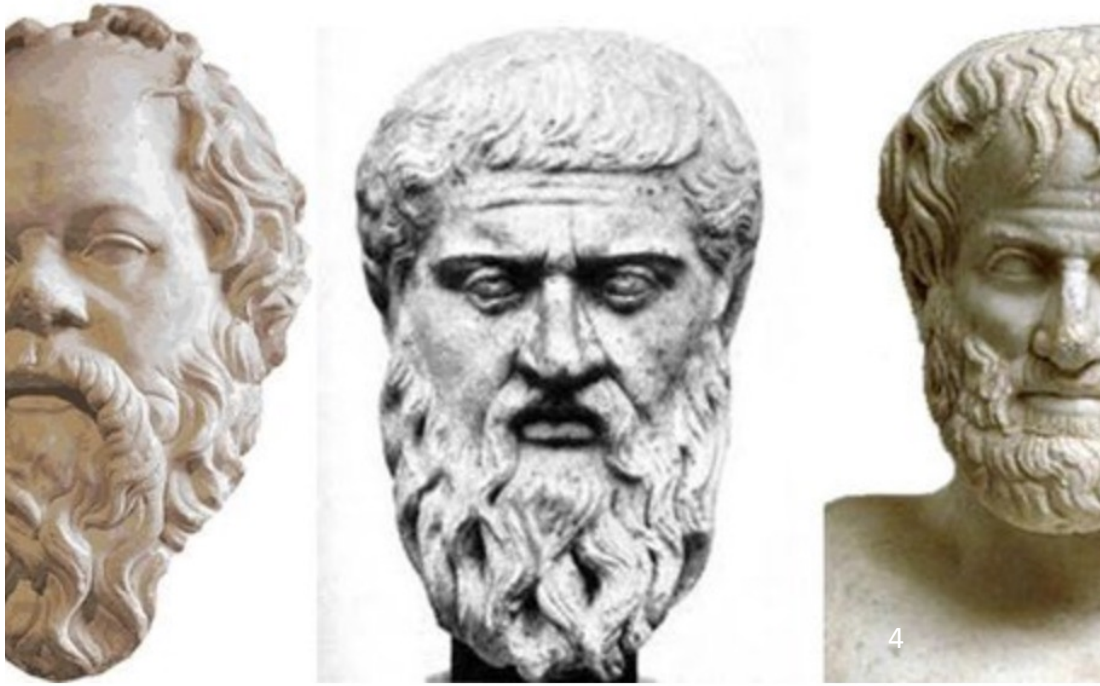
P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
$false$	$false$	$true$	$false$	$false$	$true$	$true$
$false$	$true$	$true$	$false$	$true$	$true$	$false$
$true$	$false$	$false$	$false$	$true$	$false$	$false$
$true$	$true$	$false$	$true$	$true$	$true$	$true$

$P \Leftrightarrow Q$ is true means
$P \Longrightarrow Q$ is true
$Q \Longrightarrow P$ is true

Logic: Historical View

- 500-300 BCE: Socrates, Plato, Aristotle…
  - Syllogisms
    - Modus Ponens, etc.
- Around 300 BCE: Stoic schools
  - Systematic Study of Inference Rules
  - Used deduction theorems to derive new rules
- 1500 AC to 1900 AC: Leibniz, Boole, Schroder…
  - Translation of logical inferences to purely mathematical processes
  - Enabled automated reasoning





https://www.greecehighdefinition.com/blog/2024/2/21/socrates-plato-and-aristotle-similarities-and-differences

https://en.wikipedia.org/wiki/George_Boole

Dictionary

Definitions from Oxford Languages · Learn more

syllogism

/ˈsɪləˌdʒɪz(ə)m/

noun

an instance of a form of reasoning in which a conclusion is drawn from two given or assumed propositions (premises); a common or middle term is present in the two premises but not in the conclusion, which may be invalid (e.g. *all dogs are animals; all animals have four legs; therefore all dogs have four legs*).

- deductive reasoning as distinct from induction.
  "this school of epistemology is highly advanced in syllogism and logical reasoning"

Syllogisms

Premise 1: All men are mortal

Premise 2: Socrates is a man

Conclusion: Socrates is mortal

Deductive Reasoning

Rules of Inference

- Modus Pollens
- Modus Tollens
- Elimination
- Transitivity
- Generalization
- Specialization

Considered True without the need for proof

Rules of Inference

Μην απαιτείτε απόδειξη

Modus Ponens:

Premise 1: If Statement 1 is True, then Statement 2 is True

Premise 2: Statement 1 is True

Conclusion: Statement 2 is True

Example:
Statement 1: It rains
Statement 2: It is cloudy

Modus Tollens:

Premise 1: If Statement 1 is True, then Statement 2 is True

Premise 2: Statement 2 is not True

Conclusion: Statement 1 is not True

The following statement is a tautology
If **Premise 1** and **Premise 2** are true, then **Conclusion** is true

Μην απαιτείτε απόδειξη

Rules of Inference

Elimination:

Premise 1: Statement 1 is True or Statement 2 is True

Premise 2: Statement 2 is not True

Conclusion: Statement 1 is True

Example:
Statement 1: Venue of IT5005 is LT15
Statement 2: Venue of IT5005 is LT14

Transitivity:

Premise 1: If Statement 1 is True, then Statement 2 is True

Premise 2: If Statement 2 is True, then Statement 3 is True

Conclusion: If Statement 1 is True, then Statement 3 is True

The following statement is a tautology
If **Premise 1** and **Premise 2** are true, then **Conclusion** is true

Domino effect

Rules of Inference

Μην απαιτείτε απόδειξη

Generalization:

Premise: Statement 1 is True

Conclusion: Statement 1 is True or Statement 2 is True

Example:
Statement 1: Venue of IT5005 is LT15
Statement 2: Venue of IT5005 is LT14

Specialization:

Premise: Statement 1 is True and Statement 2 is True

Conclusion: Statement 1 is True

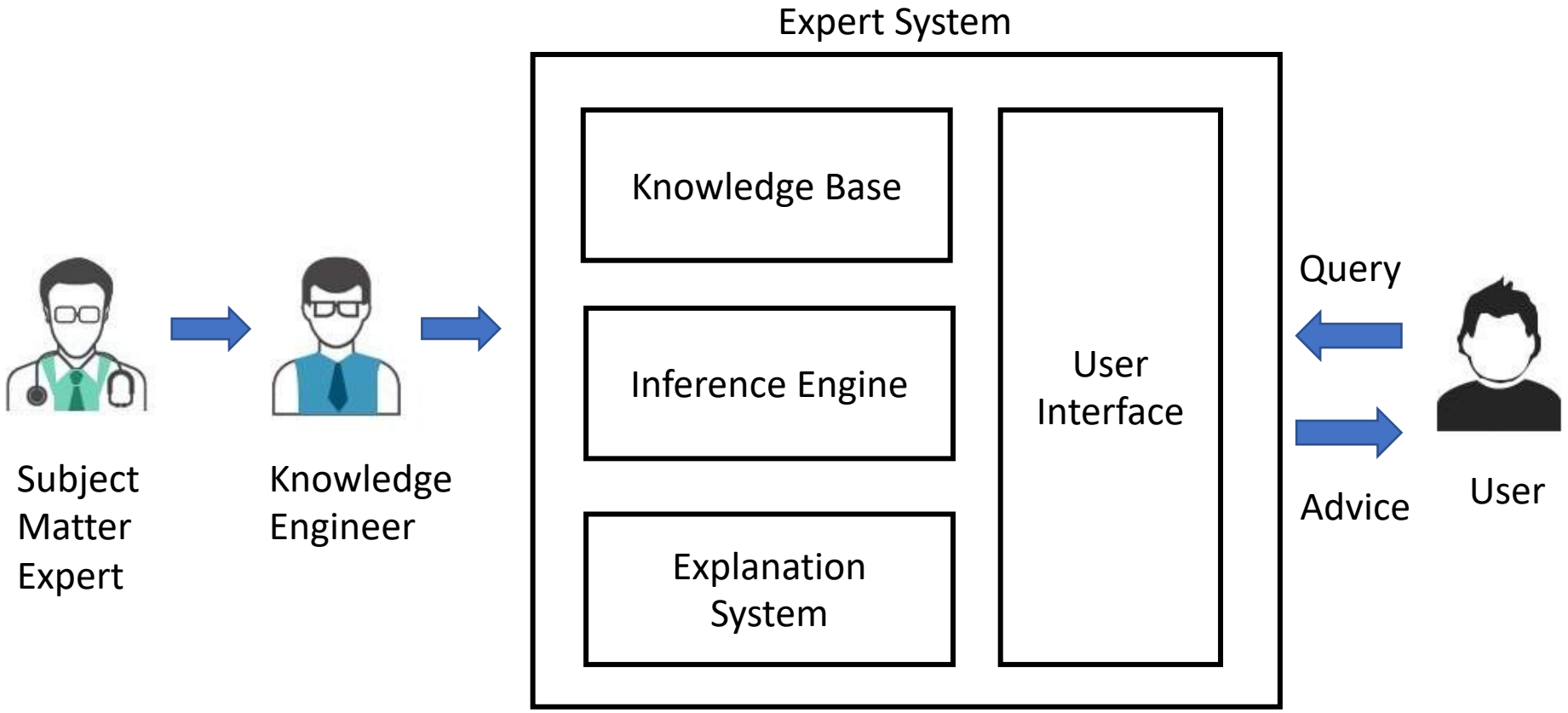
The following statement is a tautology
If **Premise** is true, then **Conclusion** is true

Example:
Statement 1: Venue of IT5005 is LT15
Statement 2: Venue of IT5005 is LT14

- You are about to leave for school in the morning and discover that you don't have your glasses. You know the following statements are true:
  a. If I was reading the newspaper in the kitchen, then my glasses are on the kitchen table.
  b. If my glasses are on the kitchen table, then I saw them at breakfast.
  c. I did not see my glasses at breakfast.
  d. I was reading the newspaper in the living room or I was reading the newspaper in the kitchen.
  e. If I was reading the newspaper in the living room then my glasses are on the coffee table.

Query: My glasses are on the coffee table: True/False?

Expert Systems



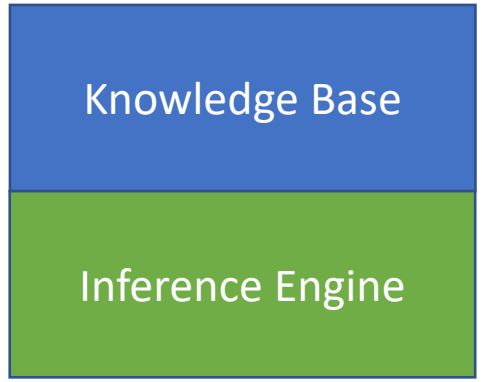
Eg: MYCIN

Agenda

- Representation of Knowledge
  - Propositional Forms
  - Conjunctive Normal Form (CNF)
  - Definite Clause
- Inferencing
  - Brute-force method
    - Model Checking
  - Proof-by-Contradiction
    - Resolution-refutation
  - Proof-by-Deduction
    - Forward Chaining
    - Backward Chaining

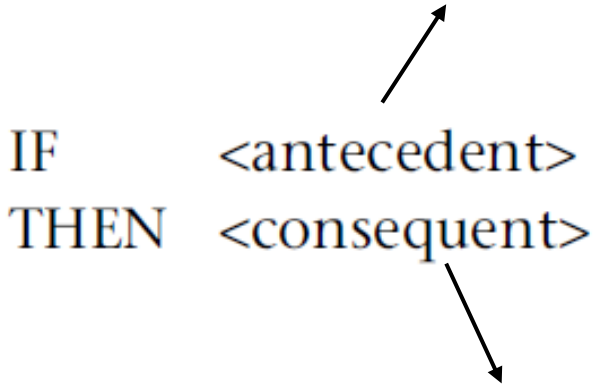
Knowledge Based (KB) Agents

- Knowledge Base
  - Contains domain-specific rules and facts
  - Set of sentences in a formal language
- Inference Engine
  - Reasoning mechanism
  - Domain-independent algorithms



Basic Syntax of a Rule

also known as Condition
or Premise
or Independent Variables



also known as Conclusion
or Action
or Result
or Outcome
or Dependent Variable

Rules with Multiple Antecedents

- Conjunction of antecedents

IF <antecedent 1>
AND <antecedent 2>
.
.
.
AND <antecedent *n*>
THEN <consequent>

Example:

IF the season is autumn
AND the sky is cloudy
AND the forecast is drizzle
THEN the advice is 'take an umbrella'

Rules with Multiple Antecedents

- Disjunction of antecedents

IF <antecedent 1>
OR <antecedent 2>
.
.
.
OR <antecedent *n*>
THEN <consequent>

Example:

if the environment is papers
or the environment is manuals
or the environment is documents
or the environment is textbooks
then the stimulus_situation is verbal

How to reason with rules?

- KB can have hundreds of rules

- Theory of logic provides tools to build KB and Reasoning

Propositional Logic

Propositional Logic

Syntax

- Sentences
- Connectives
- Operator Precedence

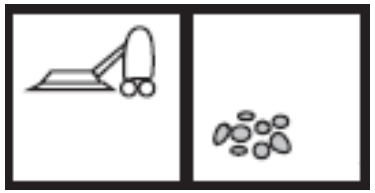
Semantics

- Possible Worlds or Models
- Truth Tables
- Logical Equivalences
- Validity
- Satisfiability
- Entailment, etc.

Propositional Logic

- Proposition
  - Statement with truth (true/false) value

- Eg:



$$Room\_A\_Clean = true$$
$$Room\_B\_Clean = false$$

Syntax and Semantics

- Syntax
  - Defines sentences in a language
- Semantics
  - Defines meaning to a sentence in the language

Propositional Logic: Syntax

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

$$
\text{OPERATOR PRECEDENCE} : \neg, \wedge, \vee, \Rightarrow, \Leftrightarrow
$$

Connectives

- Negation - NOT $(\neg)$:
  - Ex: $\neg P$
- Conjunction - AND: $(\wedge)$
  - Ex: $P \wedge Q$
    - P and Q are called Conjuncts
- Disjunction - OR: $(\vee)$
  - Ex: $P \vee Q$
    - P and Q are called Disjuncts
- Implication or Rule - IF-THEN Statement: $(\Longrightarrow)$
  - Ex: $P \Longrightarrow Q$
    - P is Premise or Antecedent
    - Q is Conclusion or Consequent
- Biconditional – If-And-Only-If - IFF: $(\Leftrightarrow)$
  - Ex: $P \Leftrightarrow Q$

Which are well-formed PL sentences?

- P
- $\neg P$
- $\neg\neg P$
- $P \wedge Q$
- $P \vee Q$
- $P \neg Q$
- $P + Q$

Semantics

- Truth Tables of Logical Connectives
- Models (Possible Worlds)
- Models of Sentences
- Satisfiable
- Logical Equivalence
- Validity and Tautology
- Contradiction
- Entailment

Truth Tables of Logical Connectives

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

$P \Longrightarrow Q$ is true means
if P is true, then Q is true
if P is false, no claim on Q

$P \Leftrightarrow Q$ is true means
$P \Longrightarrow Q$ is true
$Q \Longrightarrow P$ is true

Model

- Model (possible world)
  - An assignment of truth values to variables in a sentence
- Number of models: 2^n
  - n: number of propositional symbols

Possible worlds for P

P
$true$
$false$

Possible worlds for $\neg P$

$\neg P$
$false$
$true$

Possible worlds for $P \wedge Q$

P	Q
$true$	$true$
$false$	true
$true$	$false$
$false$	$false$

P, Q: Proposition Symbols

Model of a Sentence

m satisfies α

(or)

m is a model of α

(or)

Sentence α is true in model m

- $M(\alpha)$: Set of all models of sentence α

Model of a Sentence: Examples

P

P
$true$
$false$

$M(P) = \{P = true\}$

$\neg P$

P	$\neg P$
$true$	$false$
$false$	$true$

$M(\neg P) = \{P = false\}$

$P \wedge Q$

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

$M(P \wedge Q) = \{P = true, Q = true\}$

$M(P) = \{\{P = true, Q = true\}, \{P = true, Q = false\}\}$

$P \Rightarrow Q$

P	Q	$P \Rightarrow Q$
$true$	$true$	$true$
$false$	$true$	$true$
$true$	$false$	$false$
$false$	$false$	$true$

$M(P \Rightarrow Q) = \{\{P = true, Q = true\}, \{P = false, Q = true\}, \{P = false, Q = false\}\}$

Validity or Tautology

- A proposition is valid or a tautology if it is true in all models (possible worlds)

- Which of the following are tautologies?
  - $P \wedge Q$
  - $P \vee Q$
  - $P \Longrightarrow Q$
  - $P \wedge \neg P$
  - $P \vee \neg P$

Contradiction

- A proposition is a contradiction if it is false in all models
- Which of the following are contradictions?
  - $P \wedge Q$
  - $P \vee Q$
  - $P \Longrightarrow Q$
  - $P \wedge \neg P$
  - $P \vee \neg P$

Satisfiable

- A proposition is satisfiable if it is true in at least one model
- Which of the following are satisfiable?
  - $P \wedge Q$
  - $P \vee Q$
  - $P \wedge \neg P$
  - $P \vee \neg P$

Logical Equivalence ($\equiv$)

If $M(\alpha) = M(\beta)$, then $\alpha \equiv \beta$

(or)

If $\alpha \Leftrightarrow \beta$ is a tautology, then $\alpha \equiv \beta$

α, β: Propositional Sentences

Standard Logical Equivalences

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

Figure 7.11 Standard logical equivalences. The symbols α, β, and γ stand for arbitrary sentences of propositional logic.

Is $(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta)$?

α	β	$\neg\alpha$	$\alpha \Rightarrow \beta$	$\neg\alpha \vee \beta$
$true$	$true$	$false$	$true$	$true$
$true$	$false$	$false$	$false$	$false$
$false$	$true$	$true$	$true$	$true$
$false$	$false$	tue	$true$	$true$

Grammar for CNF

CNF sentence is formed by conjunction of clauses

A clause is formed by disjunction of literals

$$
\begin{aligned}
CNFSentence &\rightarrow Clause_1 \wedge \cdots \wedge Clause_n \\
Clause &\rightarrow Literal_1 \vee \cdots \vee Literal_m \\
Fact &\rightarrow Symbol \\
Literal &\rightarrow Symbol \mid \neg Symbol \\
Symbol &\rightarrow P \mid Q \mid R \mid \ldots \\
HornClauseForm &\rightarrow DefiniteClauseForm \mid GoalClauseForm \\
DefiniteClauseForm &\rightarrow Fact \mid (Symbol_1 \wedge \cdots \wedge Symbol_l) \Rightarrow Symbol \\
GoalClauseForm &\rightarrow (Symbol_1 \wedge \cdots \wedge Symbol_l) \Rightarrow False
\end{aligned}
$$

Figure 7.12 A grammar for conjunctive normal form, Horn clauses, and definite clauses. A CNF clause such as $\neg A \vee \neg B \vee C$ can be written in definite clause form as $A \wedge B \Rightarrow C$.

Grammar for CNF

- Symbols
  - Ex: P, Q
- Literals
  - Ex: P,$\neg P$, Q,$\neg Q$
- Facts
  - Ex: P

$$
\begin{aligned}
CNFSentence &\rightarrow Clause_1 \wedge \cdots \wedge Clause_n \\
Clause &\rightarrow Literal_1 \vee \cdots \vee Literal_m \\
Fact &\rightarrow Symbol \\
Literal &\rightarrow Symbol \mid \neg Symbol \\
Symbol &\rightarrow P \mid Q \mid R \mid \ldots
\end{aligned}
$$

Grammar for CNF

- Clauses
  - Disjunction of literals
  - Ex: $P \vee Q, P \vee \neg Q$
- Unit Clauses
  - single literals
  - Ex: $P, \neg Q$
- CNF
  - Ex: $(P \vee Q) \wedge (P \vee \neg Q)$
  - Conjunction of clauses

$$
\begin{aligned}
CNFSentence &\rightarrow Clause_1 \wedge \cdots \wedge Clause_n \\
Clause &\rightarrow Literal_1 \vee \cdots \vee Literal_m \\
Fact &\rightarrow Symbol \\
Literal &\rightarrow Symbol \mid \neg Symbol \\
Symbol &\rightarrow P \mid Q \mid R \mid \ldots
\end{aligned}
$$

Conversion to CNF

- Biconditional Elimination
  - Replace $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$

- Implication Elimination
  - Replace $\alpha \Rightarrow \beta$ with $\neg\alpha \vee \beta$

- Move $\neg$ inwards (should only appear before literals)

Conversion to CNF: Example

Formula	Remark
$B_{1,1} \Leftrightarrow P_{1,2} \vee P_{2,1}$	Given
$(B_{1,1} \Rightarrow P_{1,2} \vee P_{2,1}) \wedge (P_{1,2} \vee P_{2,1} \Rightarrow B_{1,1})$	Biconditional Elimination
$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$	Implication Elimination
$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1})$	Moving Negation Inside
$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$	Distributivity

Propositional Definite Clauses

$$
\begin{aligned}
CNFSentence &\rightarrow Clause_1 \wedge \cdots \wedge Clause_n \\
Clause &\rightarrow Literal_1 \vee \cdots \vee Literal_m \\
Fact &\rightarrow Symbol \\
Literal &\rightarrow Symbol \mid \neg Symbol \\
Symbol &\rightarrow P \mid Q \mid R \mid \ldots \\
HornClauseForm &\rightarrow DefiniteClauseForm \mid GoalClauseForm \\
DefiniteClauseForm &\rightarrow Fact \mid (Symbol_1 \wedge \cdots \wedge Symbol_l) \Rightarrow Symbol \\
GoalClauseForm &\rightarrow (Symbol_1 \wedge \cdots \wedge Symbol_l) \Rightarrow False
\end{aligned}
$$

Figure 7.12 A grammar for conjunctive normal form, Horn clauses, and definite clauses. A CNF clause such as $\neg A \vee \neg B \vee C$ can be written in definite clause form as $A \wedge B \Rightarrow C$.

Propositional Definite Clauses

- Definite Clauses
  - Must contain one positive literal

- Examples:
  - $(X \vee Y)$ is not a definite clause
    - Two positive literals
  - $(\neg X \vee Y)$ is a definite clause
    - One positive literal
  - $(\neg X \vee \neg Y)$ is not a definite clause
    - No positive literal
  - X is a definite clause
    - One positive literal

Definite Clauses

- Definite clauses can be written as implications
  - because $X \Rightarrow Y \equiv (\neg X \vee Y)$

- Examples:
  - $(\neg X \vee Y)$ can be written as $X \Rightarrow Y$
  - $(\neg X \vee \neg Y \vee Z)$ can be written as $X \wedge Y \Rightarrow Z$

Horn Clauses

- Allows at most one positive literal
  - Clauses without positive literals allowed

- Examples:
  - $(X \vee Y)$ is not a Horn clause
    - Two positive literals
  - $(\neg X \vee Y)$ is a Horn clause
    - One positive literals
  - $(\neg X \vee \neg Y)$ is a Horn clause (Not a definite clause)
    - Zero positive literals
  - X is a Horn clause
    - One positive literal

All definite clauses are Horn clauses, but not vice-versa

Fact and Goal Clauses

- Fact
  - Horn clause with single positive literal
  - Ex: X
    - $X \equiv (True \Rightarrow X)$

- Goal Clause
  - Horn clause with no positive literal
  - Ex: $(\neg X \vee \neg Y)$
    - $(\neg X \vee \neg Y) \equiv (X \wedge Y \Rightarrow False)$

dictionary.com says…

entailment

[en-**teyl**-m*uh*nt] Phonetic (Standard) IPA

noun

1 the act or fact of entailing, or involving by necessity or as a consequence:

> *The logical entailment of this approach is that the right way to design a curriculum is to make it free of bias.*

2 something involved as a necessary part or consequence of something:

> *Long hours of work are an entailment of the job.*

3 *Linguistics.* a relationship between two sentences such that if the first is true, the second must also be true, as in *Her son drives her to work every day* and *Her son knows how to drive* .

Entailment ($\models$)

$\alpha \models \beta$

(or)

$\alpha \Rightarrow \beta$ is a tautology

(or)

$M(\alpha) \subseteq M(\beta)$

(or)

$\alpha \wedge \neg\beta$ is unsatisfiable $(UNSAT)$

(or)

Sentence β logically follows sentence α

(or)

Sentence α entails sentence β

(or)

$IF\ \alpha$

$THEN\ \beta$

Question:
When are the following conditions violated?

$\alpha \Rightarrow \beta$ is a tautology

$\alpha \wedge \neg\beta$ is unsatisfiable

α	β	$\alpha \Rightarrow \beta$	$\alpha \wedge \neg\beta$
$true$	$true$	$true$	$false$
$true$	$false$	$false$	$true$
$false$	$true$	$true$	$false$
$false$	$false$	$true$	$false$

α, β: Propositional Sentences

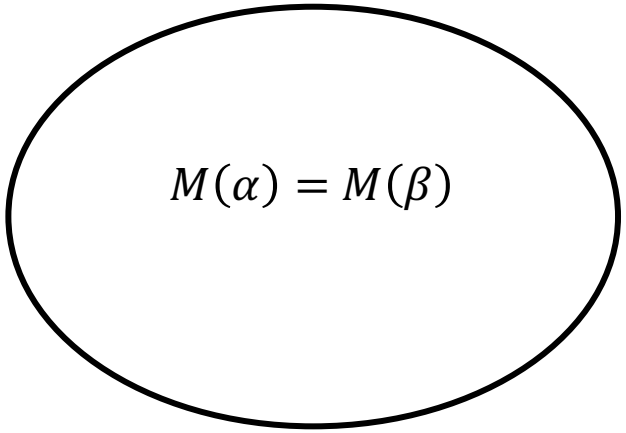
Rules of Inference and Entailment

- For a given inference rule, the following statement is a tautology
  - If the premises are true, then conclusion is true
- In other words, premises entail conclusion
- **Example**:
  - **Modus Ponens**:
    - **Premise 1**: If it rains, then it is cloudy
    - **Premise 2**: It rains
    - **Conclusion**: Therefore, it is cloudy

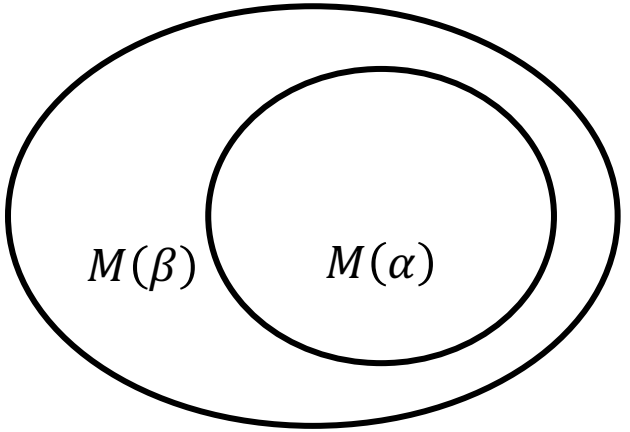
  The following statement is a tautology
  If **Premise 1** and **Premise 2** are true, then **Conclusion** is true

Equivalence vs Entailment

$\alpha \equiv \beta$



$\alpha \models \beta$



$M(\alpha) \subseteq M(\beta)$

Which of the following are true?

- $(P \wedge (P \Rightarrow Q)) \vDash Q$
- $(Q \wedge (P \Rightarrow Q)) \vDash P$

P	Q	$P \Rightarrow Q$	$P \wedge (P \Rightarrow Q)$	$Q \wedge (P \Rightarrow Q)$	$(P \wedge (P \Longrightarrow Q)) \Rightarrow Q$	$(Q \wedge (P \Longrightarrow Q)) \Rightarrow P$
$true$	$true$	$true$	$true$	$true$	$true$	$true$
$true$	$false$	$false$	$false$	$false$	$true$	$true$
$false$	$true$	$true$	$false$	$true$	$true$	$false$
$false$	$false$	$true$	$false$	$false$	$true$	$true$

Proof via tautology of $\alpha \Rightarrow \beta$

Representation and Reasoning using PL

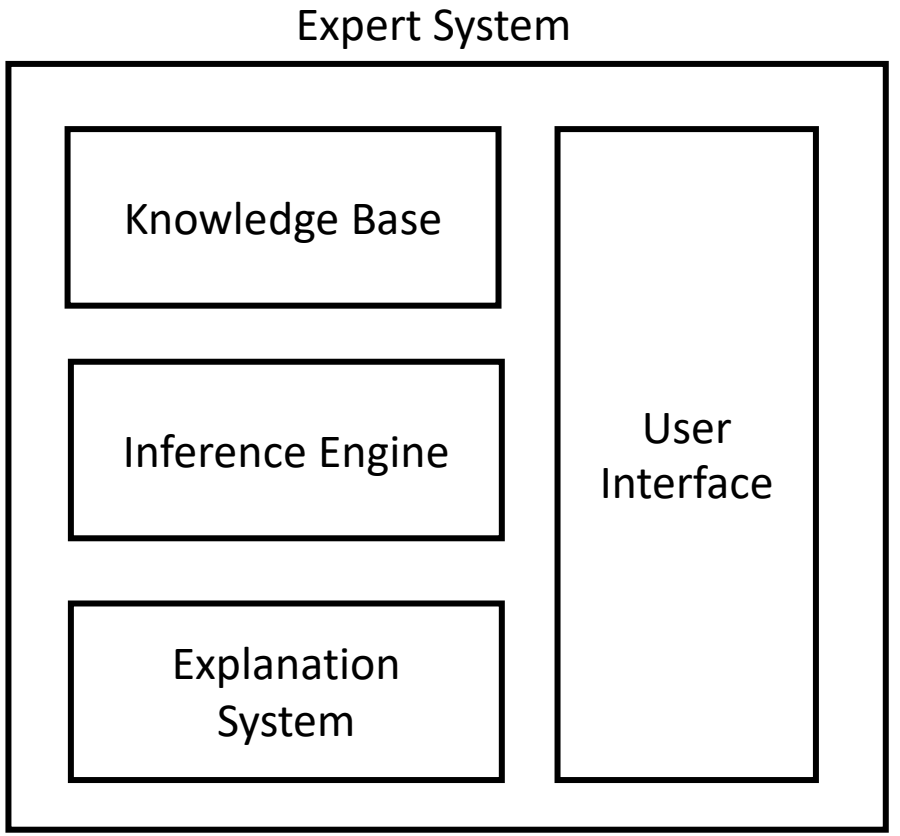
Representation using PL

- Rules are encoded into propositional sentences
- KB:
  - A set of sentences that are true
    (or)
  - A single sentence that asserts all sentences

$$KB = \{\alpha_1, \alpha_2, \ldots, \alpha_N\}$$ Set of rules

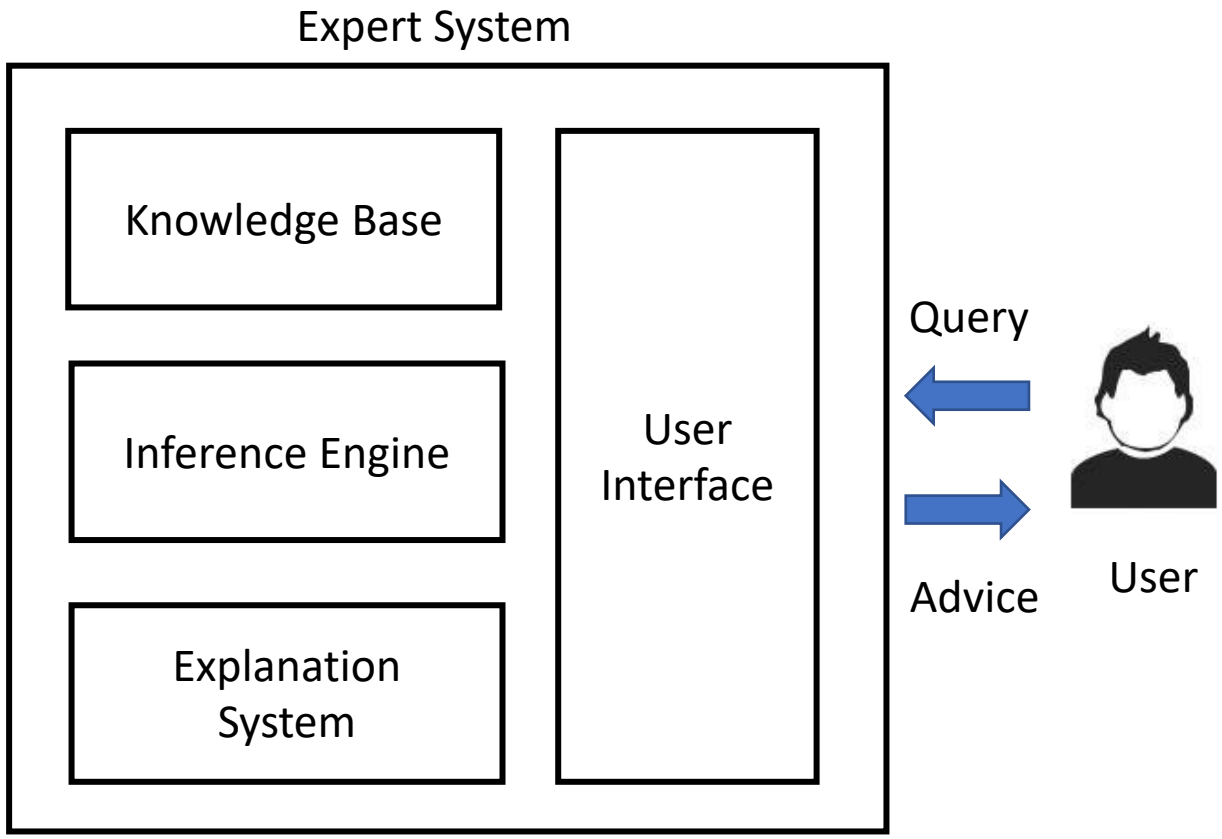
or

$$KB = \alpha_1 \wedge \alpha_2 \wedge \cdots \wedge \alpha_N$$ Conjunction of rules



Inferencing

- Inferencing in Logic:
  - Process of deriving new sentences that are true from the KB
- $KB \vdash_i \alpha$ means sentence α is derived from KB



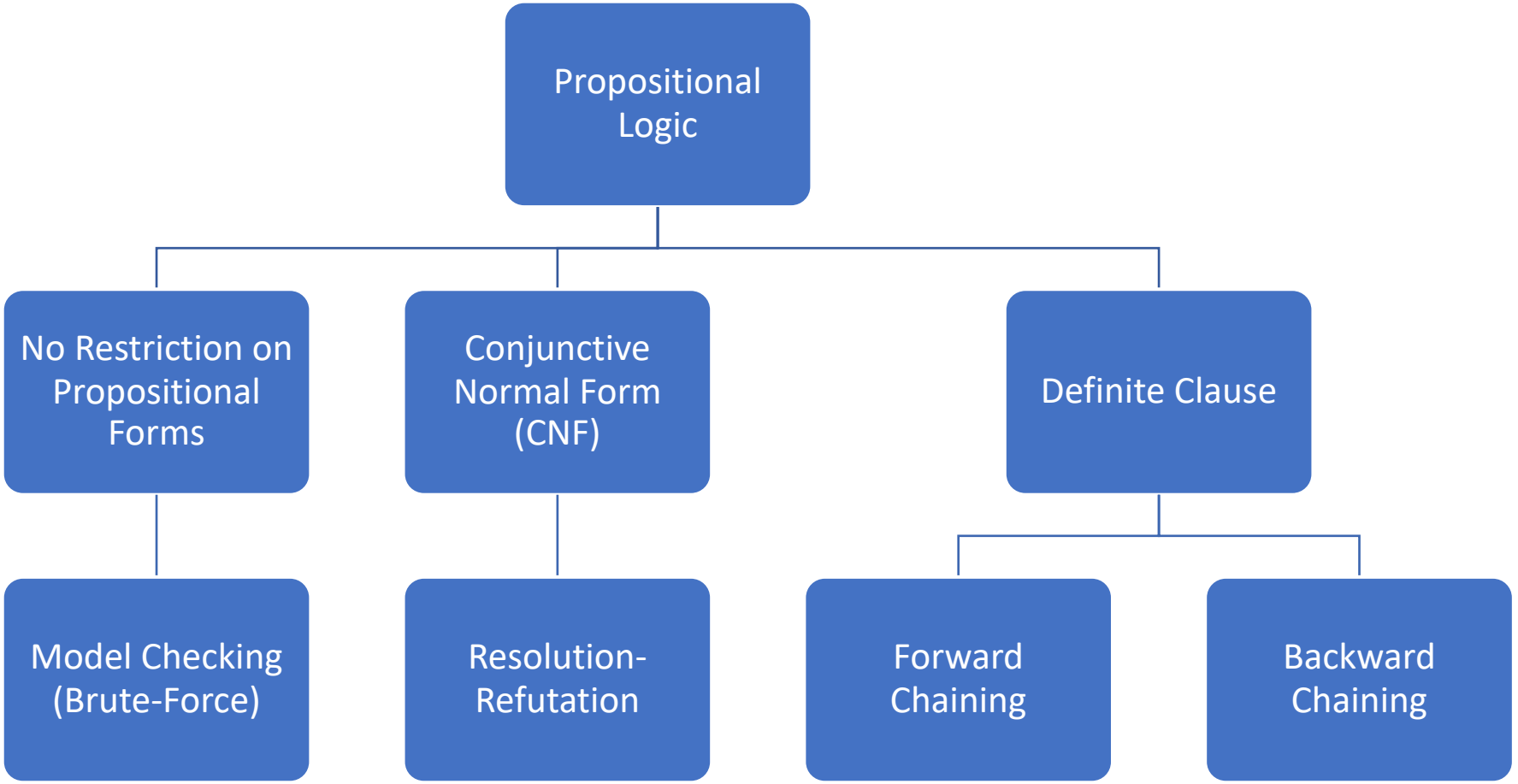
Convert user queries to propositional sentences and check whether the query sentence can be derived from KB

Soundness and Completeness

- Inferencing is sound if it always derives entailed sentences
  - If $KB \vdash_i \alpha,$ then $KB \models \alpha$
  - Doesn't make stuff up

- Inferencing is complete if it can derive all entailed sentences
  - If $KB \models \alpha,$ then $KB \vdash_i \alpha$
  - Derives all derivable sentences

Inferencing Methods



Model Checking

- Check whether $M(KB) \subseteq M(Query)$

where $KB = \alpha_1 \wedge \alpha_2 \wedge \cdots \wedge \alpha_N$

$Query$: Query Sentence

$M(KB)$: Set of models of KB

$M(Query)$: Set of models of $Query$

$$M(KB) = \bigcap_{i=1}^{N} M(\alpha_i)$$

$M(KB)$ indicate critical rows of truth table
(Refer Slide # 57 in Logic Jan2024.pdf)

Toy Example (Revisit)

- Given $KB = \{P, P \Longrightarrow Q\}$ and $Query = Q$.
- We know from Modus Ponens that the query Q is $True$
- We need to prove it through Model Checking algorithm

Toy Example (Revisit)

Refer Slide # 60 in Logic Jan2024.pdf

- Given $KB = \{P, P \Longrightarrow Q\}$ and $Query\ Q$, does KB entails $Query$?

P	Q	$P \Rightarrow Q$	KB	$Query$	
$true$	$true$	$true$	$true$	$true$	Critical row
$false$	$true$	$true$	$false$	$true$	
$true$	$false$	$false$	$false$	$false$	
$false$	$false$	$true$	$false$	$false$	

$M(P) = \{\{P = true, Q = true\}, \{P = true, Q = false\}\}$

$M(P \Rightarrow Q) = \{\{P = true, Q = true\}, \{P = false, Q = true\}, \{P = false, Q = false\}\}$

- Proof by Model Checking:
  - $M(KB) = M(P) \cap M(P \Longrightarrow Q)$

$$= \{P = true, Q = true\}$$

  - $M(Query) = \{\{P = true, Q = true\}, \{P = false, Q = true\}\}$

$M(KB) \subseteq M(Query)$

$\therefore KB \vDash Query$

Algorithm for Model Checking

Have you seen similar algorithm before?

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

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

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

Base case

For complete assignment check if KB is True, α is also True or not.

Selecting a variable for assignment

Assigning values to selected variable

Figure 7.10 A truth-table enumeration algorithm for deciding propositional entailment. (TT stands for truth table.) PL-TRUE? returns *true* if a sentence holds within a model. The variable *model* represents a partial model—an assignment to some of the symbols. The keyword **and** here is an infix function symbol in the pseudocode programming language, not an operator in proposition logic; it takes two arguments and returns *true* or *false*.

$PL-TRUE(KB, model)$ checks whether the KB is true for the given $model$

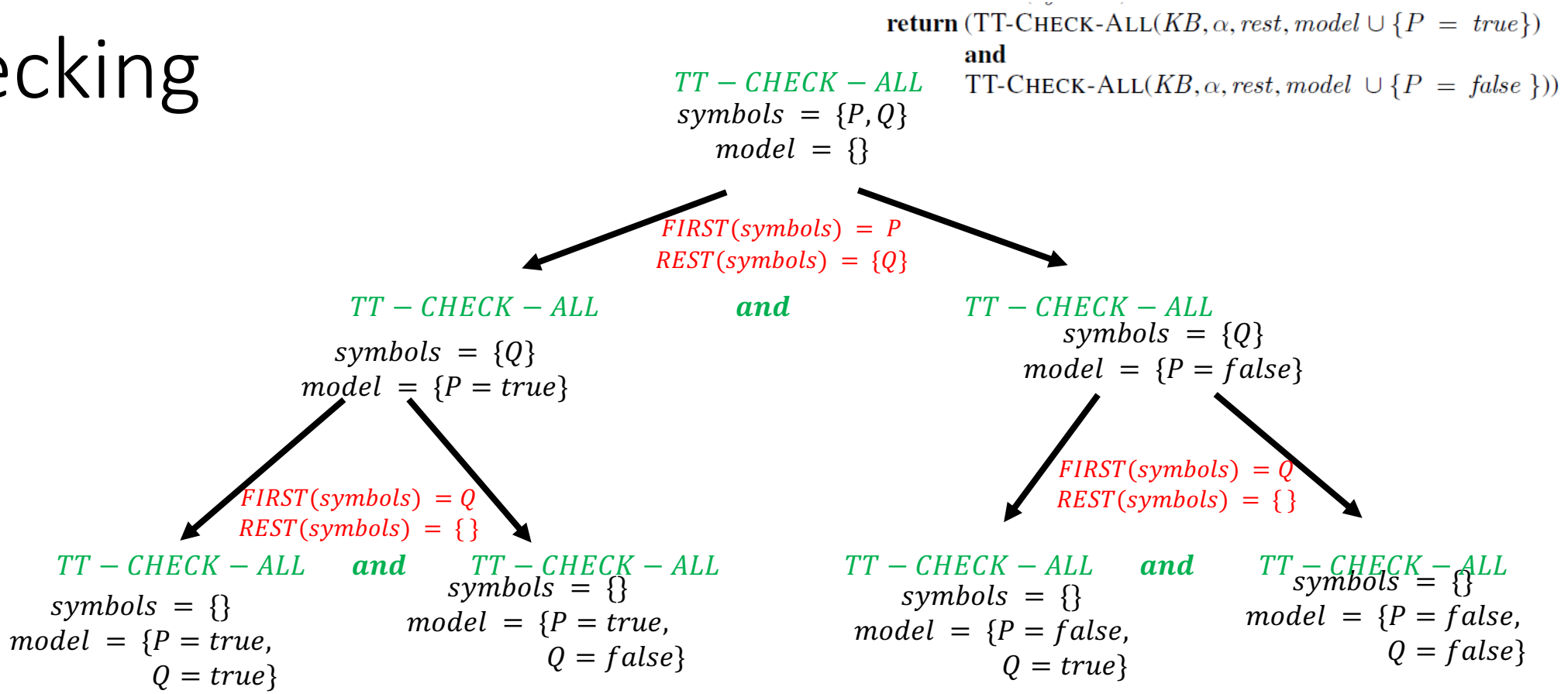
$PL-TRUE(\alpha, model)$ checks whether the α is true for the $model$

$EMPTY?(symbols)$ checks whether all variables are assigned a value

$TT-CHECK-ALL$ is a recursive function

Model Checking

- $KB = \{P, P \Longrightarrow Q\}$
- $\alpha = \{Q\}$



Base case

$PL-TRUE(KB, model)$?	*true*	$false$	$false$	$false$
$PL-TRUE(\alpha, model)$?	*true*	$false$	$true$	$false$
$TT-CHECK-ALL$	*true*	$true$	$true$	$true$

if EMPTY?($symbols$) **then**
  if PL-TRUE?(KB, $model$) **then return** PL-TRUE?(α, $model$)
  else return $true$ // when KB is false, always return true

Model Checking: Another Example

- Given $KB = \{P \Longrightarrow Q\}$ and $Query$= Q,

  does KB entails $Query$?

- We know that this query evaluates to $False$ because we don't know whether P is $True/False$.

- So we cannot use Modus Ponens to evaluate the query

Model Checking: Another Example

- Given $KB = \{P \Longrightarrow Q\}$ and $Query = Q$, does KB entails $Query$?

P	Q	$P \Rightarrow Q$	KB	$Query$
$true$	$true$	$true$	$true$	$true$
$false$	$true$	$true$	$true$	$true$
$true$	$false$	$false$	$false$	$false$
$false$	$false$	$true$	$true$	$false$

Critical rows

$$M(P \Rightarrow Q) = \{\{P = true, Q = true\}, \{P = false, Q = true\}, \{P = false, Q = false\}\}$$

- Proof by Model Checking:
  - $M(KB) = M(P \Longrightarrow Q)$

$$= \{\{P = true, Q = true\}, \{P = false, Q = true\}, \{P = false, Q = false\}\}$$

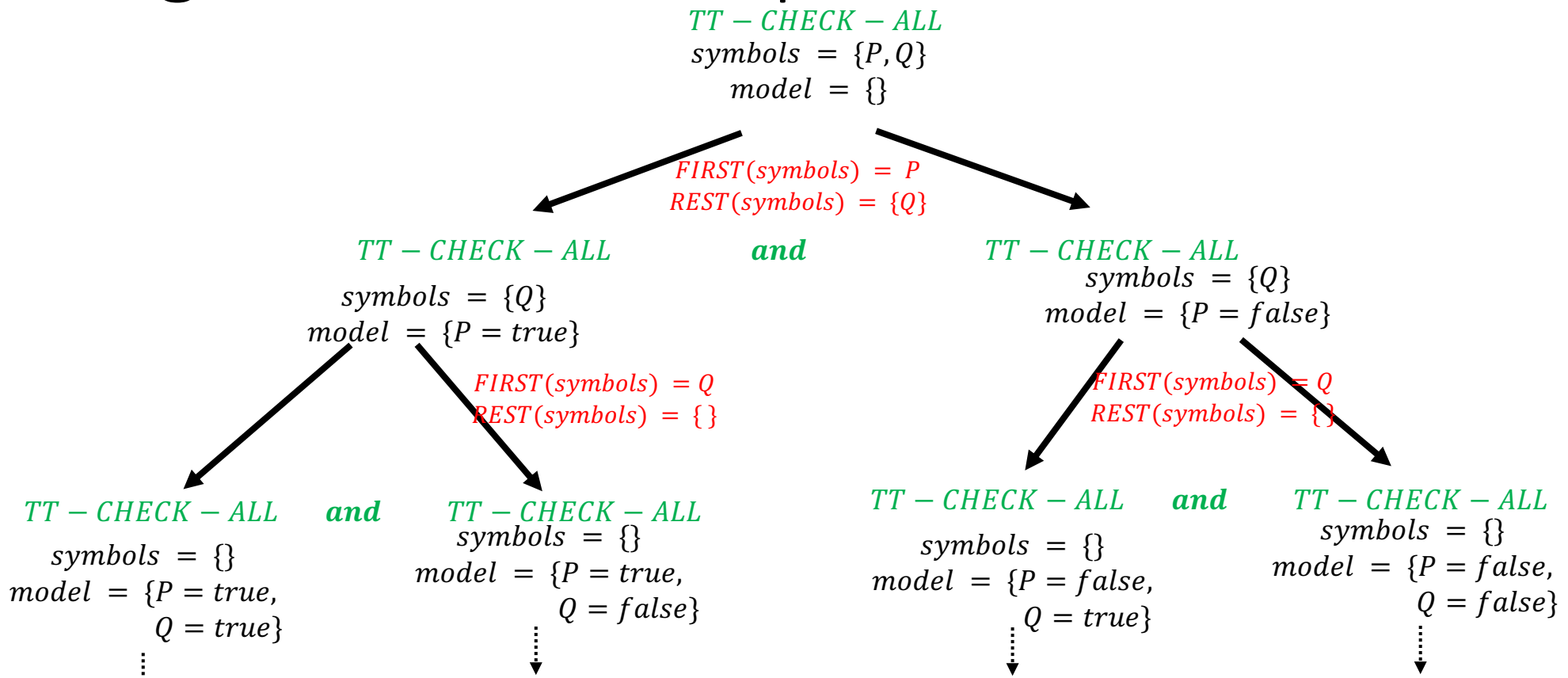
  - $M(Query) = \{\{P = true, Q = true\}, \{P = false, Q = true\}\}$

$$M(KB) \nsubseteq M(Query)$$

$$\therefore KB \nvdash Query$$

Model Checking: Another Example

- $KB = \{P \Longrightarrow Q\}$
- $\alpha = \{Q\}$



Base case

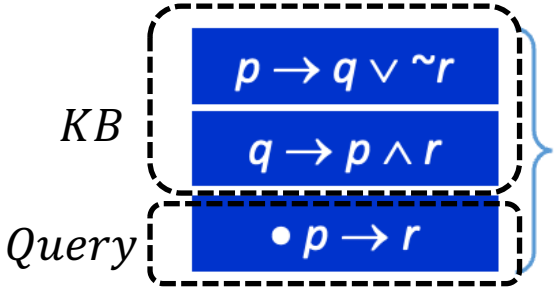
$PL - TRUE(KB, model)$?	$true$	$false$	$true$	$true$
$PL - TRUE(\alpha, model)$?	$true$	$false$	$true$	$false$
$TT - CHECK - ALL$	$true$	$true$	$true$	$false$

Model Checking: Homework (1)

- $KB = \{Q, P \implies Q\}, Query = \{P\}$

Model Checking: Homework (2)

Refer Slide # 58 in Logic Jan2024.pdf



Invalid argument

p	q	r	~r	q ∨ ~r	p ∧ r	p → q ∨ ~r	q → p ∧ r	p → r
						premises		*conclusion*
T	T	T	F	T	T	T	T	T
T	T	F	T	T	F	T	F	
T	F	T	F	F	T	F	T	
T	F	F	T	T	F	T	T	F
F	T	T	F	T	F	T	F	
F	T	F	T	T	F	T	F	
F	F	T	F	F	F	T	T	T
F	F	F	T	T	F	T	T	T

Critical rows

$M(KB)$ indicate critical rows of truth table

We check whether $Query$ is true in the critical row are not?

CSP Vs Model Checking

- CSP
  - Objective is to find a complete and consistent assignment
  - Stop the search as soon as an assignment that satisfies the constrains is found
- Model Checking
  - Objective is to check when KB is $True$, whether the $Query$ is also $True$ or not.
    - KB and and $Query$ are two constraints and consistent assignment means the constraints are evaluated to $True$
  - Need to verify it for all assignments

Performance of Model Checking

- Sound
  - Provides only correct inferences
- Complete
  - Because exhaustive search
- Number of models are exponential in number of variables (2^n)

Inferencing Methods

- Model Checking
  - Exhaustive truth table enumeration
- Proof by Deduction
  - Modus Ponens, AND-Elimination rules, Inferencing Rules
  - Forward Chaining and Backward Chaining
- Proof by Contradiction
  - Resolution Theorem

Proof by Deduction

- Inference Rules
  - Modus Ponens

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

  - AND-Elimination

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

  - Rules derived from logical equivalences

$$\frac{\alpha \Leftrightarrow \beta}{(\alpha \Longrightarrow \beta) \wedge (\beta \Longrightarrow \alpha)}$$

$$\frac{(\alpha \Longrightarrow \beta) \wedge (\beta \Longrightarrow \alpha)}{\alpha \Leftrightarrow \beta}$$

Proof by Deduction: Toy Example

Only for your reading

- Given a KB and query α, does KB entails query α ?

- Example:
  - KB: $\{P, P \Rightarrow Q\} = P \wedge (P \Rightarrow Q)$
  - Query: Q

  - Answer: $True$
    - From Modus Ponens

Proof by Deduction: Example

- Example:
  - KB: $(R_1 \wedge R_2 \wedge R_3 \wedge R_4 \wedge R_5)$
    - R_1: $\neg P_{11}$
    - R_2: $B_{1,1} \Leftrightarrow P_{1,2} \vee P_{2,1}$
    - R_3: $B_{2,1} \Leftrightarrow P_{1,1} \vee P_{2,2} \vee P_{3,1}$
    - R_4: $\neg B_{1,1}$
    - R_5: $B_{2,1}$
  - Query:
    - $\neg P_{2,1}$

Proof by Deduction: Example

- KB: $(R_1 \wedge R_2 \wedge R_3 \wedge R_4 \wedge R_5)$
  - R_1: $\neg P_{11}$
  - R_2: $B_{1,1} \Leftrightarrow P_{1,2} \vee P_{2,1}$
  - R_3: $B_{2,1} \Leftrightarrow P_{1,1} \vee P_{2,2} \vee P_{3,1}$
  - R_4: $\neg B_{1,1}$
  - R_5: $B_{2,1}$

$Query$: $\neg P_{2,1}$

Answer: $True$

- Apply biconditional elimination to R_2

  R_6: $\left(B_{1,1} \Rightarrow \left(P_{1,2} \vee P_{2,1}\right)\right) \wedge \left(\left(P_{1,2} \vee P_{2,1}\right) \Rightarrow B_{1,1}\right)$
- Apply AND-Elimination to R_6

  R_7: $\left(P_{1,2} \vee P_{2,1}\right) \Rightarrow B_{1,1}$
- Logical Equivalence for Contra Positive

  R_8: $\neg B_{1,1} \Rightarrow \neg\left(P_{1,2} \vee P_{2,1}\right)$
- Apply Modus Ponens

  R_9: $\neg\left(P_{1,2} \vee P_{2,1}\right)$
- Apply DeMorgan's Rule

  R_{10}: $\neg P_{1,2} \wedge \neg P_{2,1}$
- Apply AND-Elimination to R_{10}

  R_{11}: $\neg P_{2,1}$

How to convert this to an algorithm?

Refer Fig. 7.11 for Logical Equivalences

Procedure for Search

- **INITIAL STATE**: Start with initial KB and Inference Rules
- **ACTIONS**: Set of actions consists of all inference rules applied to all sentences that match top half of inference rules
- **RESULT**: Result of action is to add the sentence in bottom half of inference rule to KB
- **GOAL**: State with KB that contains the sentence that we are trying to prove

Inferencing Methods

- Model Checking
  - Exhaustive truth table enumeration
- Proof by Deduction
  - Modus Ponens, AND-Elimination rules, Inferencing Rules
  - Forward Chaining and Backward Chaining
- Proof by Contradiction
  - Resolution Theorem

Proof by Contradiction

- Uses a single rule called Resolution
  - Easy to automate the reasoning (theorem proving)

- But it requires sentences in CNF
  - CNF: Conjunctive Normal Form

Inferencing by Resolution

- Converts inferencing problem into SAT (satisfiability) problem
  - Clauses as constraints

- Relies on two tools
  - Refutation
  - Resolution

Resolution Rule

$$
\begin{aligned}
CNFSentence &\rightarrow Clause_1 \wedge \cdots \wedge Clause_n \\
Clause &\rightarrow Literal_1 \vee \cdots \vee Literal_m \\
Fact &\rightarrow Symbol \\
Literal &\rightarrow Symbol \mid \neg Symbol \\
Symbol &\rightarrow P \mid Q \mid R \mid \ldots
\end{aligned}
$$

Single Inferencing Rule:

$$\{Literal_1 \vee Clause_1, \neg Literal_1 \vee Clause_2\} \vDash (Clause_1 \vee Clause_2)$$

or

$$\big((Literal_1 \vee Clause_1) \wedge (\neg Literal_1 \vee Clause_2)\big) \vDash (Clause_1 \vee Clause_2)$$

or

$$\frac{(Literal_1 \vee Clause_1) \wedge (\neg Literal_1 \vee Clause_2)}{(Clause_1 \vee Clause_2)}$$

Resolution Rule: Example

- Given either X or Y are true.
- If we know $\neg X$ is true, then we can resolve Y as true
- Formally:
  - $\{X \vee Y, \neg X\} \vDash Y$

    or
  - $\{(X \vee Y) \wedge (\neg X)\} \vDash Y$

    or
  - Resolvent of $(X \vee Y)$ and $(\neg X)$ is Y

    or

$$\frac{(X \vee Y) \wedge (\neg X)}{Y}$$

Resolution Rule: Examples

- $((X \vee Y) \wedge (\neg X \vee Y)) \vDash Y$
  - Resolvent of $(X \vee Y)$ and $(\neg X \vee Y)$ is $(Y \vee Y) \equiv Y$

$$\frac{(X \vee Y) \wedge (\neg X \vee Y)}{Y}$$

$(Y \vee Y) \equiv Y$
Removal of multiple copies is known as factoring

- $((X \vee Y \vee Z) \wedge (X \vee \neg Y \vee Z)) \vDash (X \vee Z)$
  - Resolvent of $(X \vee Y \vee Z)$ and $(X \vee \neg Y \vee Z)$ is $(X \vee Z)$

$$\frac{(X \vee Y \vee Z) \wedge (X \vee \neg Y \vee Z)}{(X \vee Z)}$$

- Resolvent of X and $\neg X$ is an empty clause

$$\frac{X \wedge \neg X}{\square}$$

$\square$ indicates empty clause

Refutation

Proof-by-Contradiction

$KB \vDash \alpha$ if and only if $KB \wedge \neg\alpha$ is a contradiction

or

$KB \vDash \alpha$ if and only if $KB \wedge \neg\alpha \vDash false$

$KB \implies \alpha$ is a Tautology

or

$\neg KB \vee \alpha$ is a Tautology

or

$KB \wedge \neg\alpha$ is a Contradiction

or

$KB \wedge \neg\alpha$ is UNSAT

Refutation: Example

$$KB = P$$
$$Query: \alpha = P$$

$$\frac{KB \wedge \neg\alpha}{\square}$$

Refers to empty clause

Resolution Algorithm

function PL-RESOLUTION(KB, α) **returns** *true* or *false*
  inputs: KB, the knowledge base, a sentence in propositional logic
    α, the query, a sentence in propositional logic

  clauses $\leftarrow$ the set of clauses in the CNF representation of $KB \wedge \neg\alpha$
  new $\leftarrow \{\}$
  while *true* **do**
    for each pair of clauses C_i, C_j **in** *clauses* **do**
      resolvents $\leftarrow$ PL-RESOLVE(C_i, C_j)
      if *resolvents* contains the empty clause **then return** *true* Terminate if contradiction is detected
      new $\leftarrow$ *new* $\cup$ *resolvents*
    if *new* $\subseteq$ *clauses* **then return** *false* If new clauses are not created, no contradiction is reached
    clauses $\leftarrow$ *clauses* $\cup$ *new* If new clauses are created, add them to set of clauses

Figure 7.13 A simple resolution algorithm for propositional logic. PL-RESOLVE returns the set of all possible clauses obtained by resolving its two inputs.

Resolution Algorithm: Example

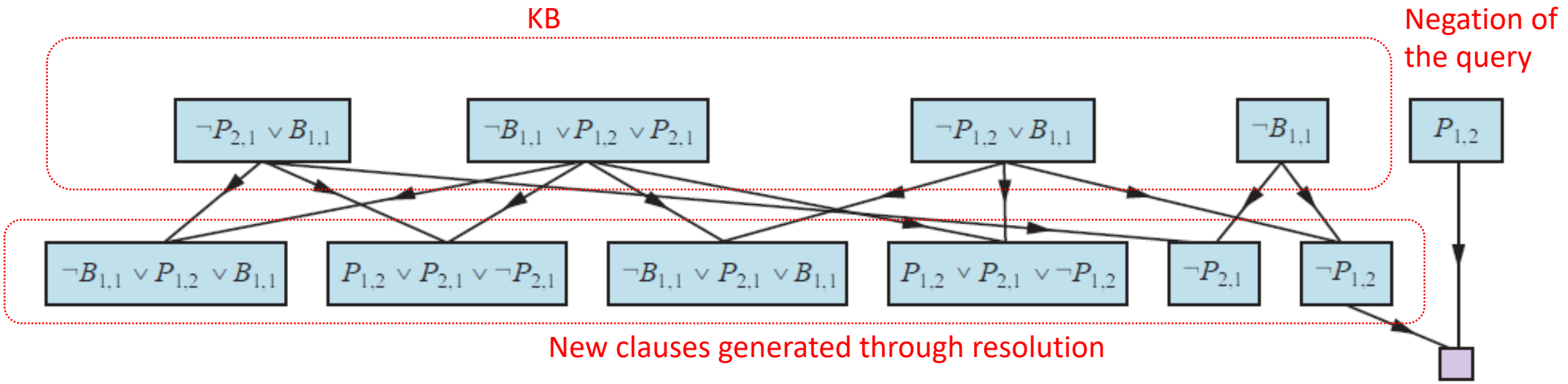


Figure 7.14 Partial application of PL-RESOLUTION to a simple inference in the wumpus world to prove the query $\neg P_{1,2}$. Each of the leftmost four clauses in the top row is paired with each of the other three, and the resolution rule is applied to yield the clauses on the bottom row. We see that the third and fourth clauses on the top row combine to yield the clause $\neg P_{1,2}$, which is then resolved with $P_{1,2}$ to yield the empty clause, meaning that the query is proven.

Horn Clauses

- Closed under resolution
  - Resolvent of two Horn clauses is a Horn clause

- Example:
  - $((\neg X \vee \neg Y \vee \neg Z) \wedge (\neg X \vee \neg Y \vee Z)) \models (\neg X \vee \neg Y)$

Inferencing with Horn Clauses

- If KB contains only Horn clauses, inferencing via Modus ponens is complete

- Modus Ponen's Rule

$$\frac{X_1, X_2, \ldots, X_{n,}\ \left(X_1 \wedge X_2 \wedge \ldots \wedge X_{n,}\right) \Rightarrow Y}{Y}$$

Inferencing with Horn Clauses: Algorithms

- Forward Chaining
  - Data-driven reasoning

- Backward Chaining
  - Goal-driven reasoning

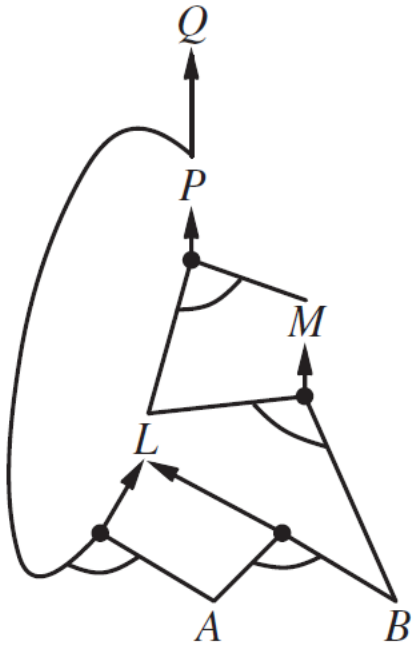
Forward Chaining: Example

KB:

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

Query: Q

AND-OR Graph or Hyper Graph



Intuition:

Identify rules whose premises are all true in KB; add its conclusion to KB
Repeat the above step until query is added to KB

Forward Chaining

function PL-FC-ENTAILS?(KB, q) **returns** *true* or *false*

inputs: KB, the knowledge base, a set of propositional definite clauses
q, the query, a proposition symbol

count ← a table, where *count*[c] is initially the number of symbols in clause c's premise

inferred ← a table, where *inferred*[s] is initially *false* for all symbols

queue ← a queue of symbols, initially symbols known to be true in KB

while *queue* is not empty **do**
  p ← POP(*queue*)
  if $p = q$ **then return** *true*
  if *inferred*[p] = *false* **then**
    inferred[p] ← *true*
    for each clause c in KB where p is in c.PREMISE **do**
      decrement *count*[c]
      if *count*[c] = 0 **then** add c.CONCLUSION to *queue*
return *false*

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B

queue = $[A, B]$

s	*inferred*
A	False
B	False
L	False
M	False
P	False
Q	False

c	*count*
1	1
2	2
3	2
4	2
5	2
6	0
7	0

Forward Chaining: Example

KB:

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

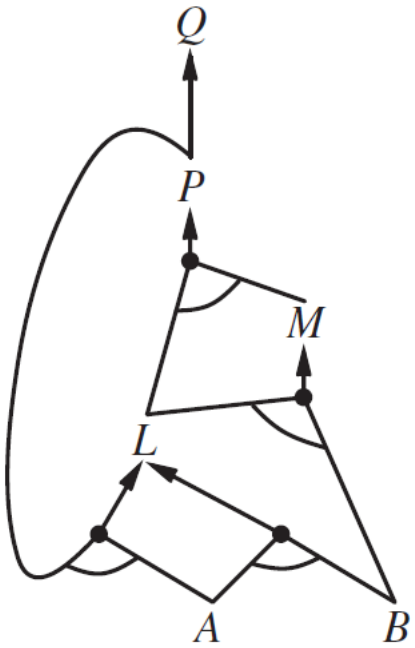
$A \wedge B \Rightarrow L$

A

B

Query: Q

AND-OR Graph or Hyper Graph



c	count
1	1
2	2
3	2
4	2
5	2
6	0
7	0

s	$inferred$
A	False
B	False
L	False
M	False
P	False
Q	False

queue = $[A, B]$

Intuition:

Identify rules whose premises are all true in KB; add its conclusion to KB

Repeat the above step until query is added to KB

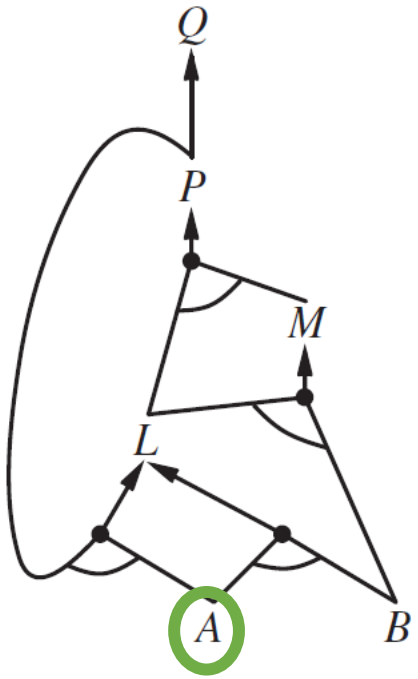
Forward Chaining: Example

KB:

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

Query: Q

AND-OR Graph



Intuition:

Identify rules whose premises are all true in KB; add its conclusion to KB
Repeat the above step until query is added to KB

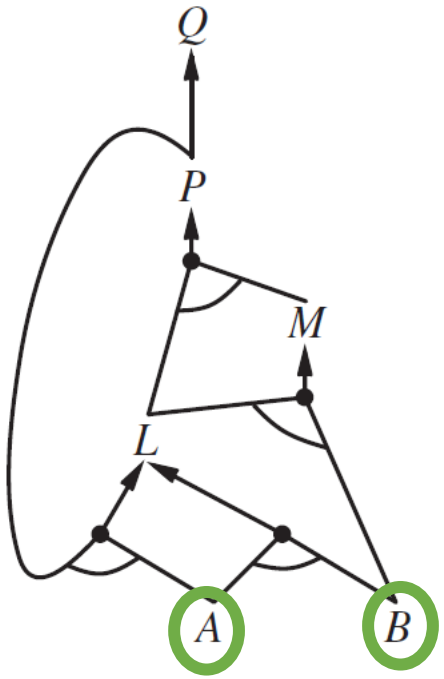
Forward Chaining: Example

KB:

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

Query: Q

AND-OR Graph



Intuition:

Identify rules whose premises are all true in KB; add its conclusion to KB
Repeat the above step until query is added to KB

Forward Chaining: Example

KB:

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

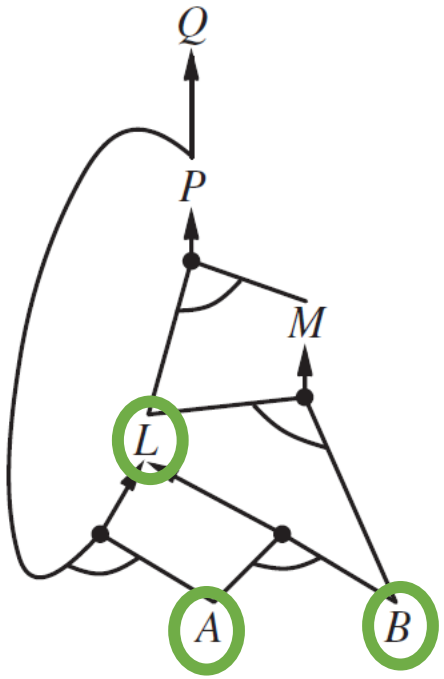
A

B

L

Query: Q

AND-OR Graph



Add L to KB

Intuition:

Identify rules whose premises are all true in KB; add its conclusion to KB

Repeat the above step until query is added to KB

Forward Chaining: Example

KB:

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

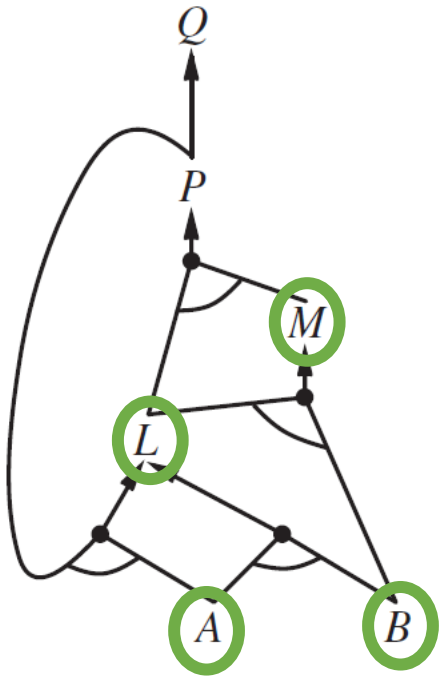
B

L

M

Query: Q

AND-OR Graph



Add M to KB

Intuition:

Identify rules whose premises are all true in KB; add its conclusion to KB

Repeat the above step until query is added to KB

Forward Chaining: Example

KB:

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

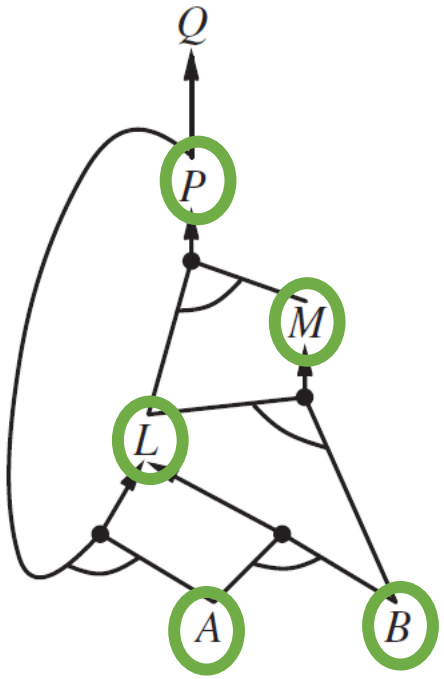
B

L

M

P

AND-OR Graph



Add P to KB

Query: Q

Intuition:

Identify rules whose premises are all true in KB; add its conclusion to KB

Repeat the above step until query is added to KB

Forward Chaining: Example

KB:

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

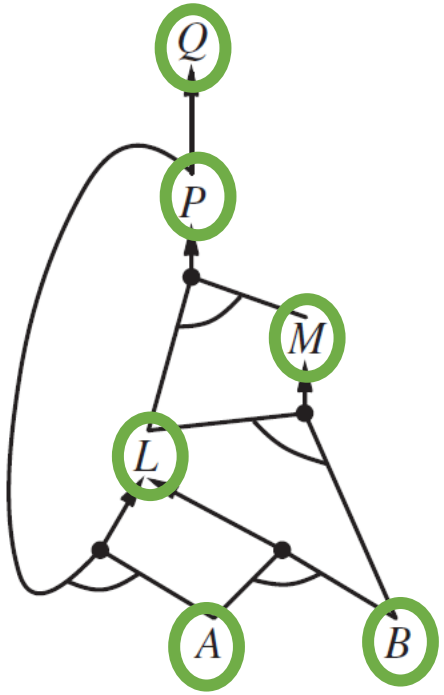
B

L

M

P

AND-OR Graph



Add P to KB

Query: Q

Intuition:

Identify rules whose premises are all true in KB; add its conclusion to KB

Repeat the above step until query is added to KB

Backward Chaining: Intuition

- Start with query (α)
- Find the implications whose conclusion is α
- Recursively prove the antecedents of each implication
- If at least one of the antecedents cannot be proved to be true, return False

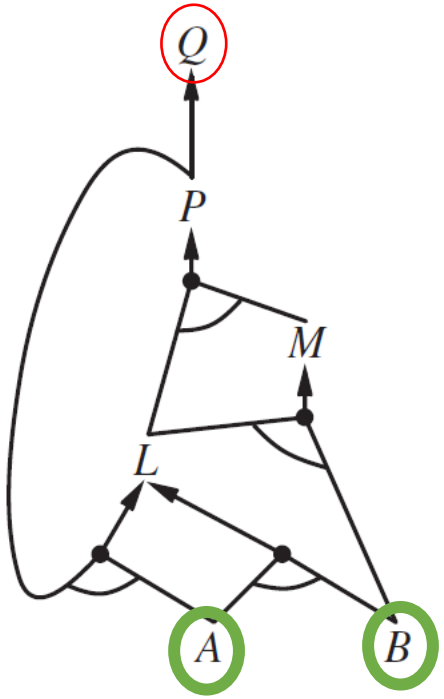
Backward Chaining: Example

KB:

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

Query: Q

AND-OR Graph



Start with Query Q

Intuition:

Prove antecedents of the query

Backward Chaining (Psuedocode)

function $PL - BC - ENTAILS?\,(KB, q)$ returns true or false
inputs: KB, the knowledge base, a set of propositional definite clauses
q: the query, a propositional symbol

If q matches a fact, **then return** true
If there is no clause with a consequent that matches q, **then return** false

for each clause c in KB where q is in $c.CONCLUSION$ **do**
$count$ = Number of symbols in $c.PREMISE$
for all symbols p in $c.PREMISE$ do
if $PL - BC - ENTAILS?\,(KB, p)$
if p not in KB **then** add p to KB
$count$ = $count$ - 1
else break
if $count$ == 0, **then return** true
return false

KB:

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

Query: Q

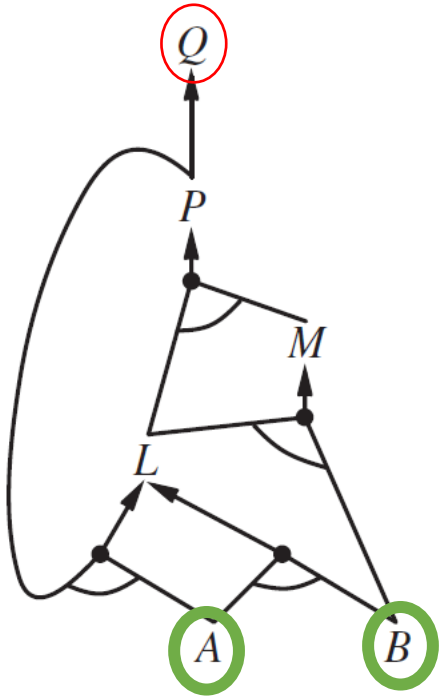
Backward Chaining: Example

KB:

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

Query: Q

AND-OR Graph



Start with Query Q

Intuition:

Prove antecedents of the query

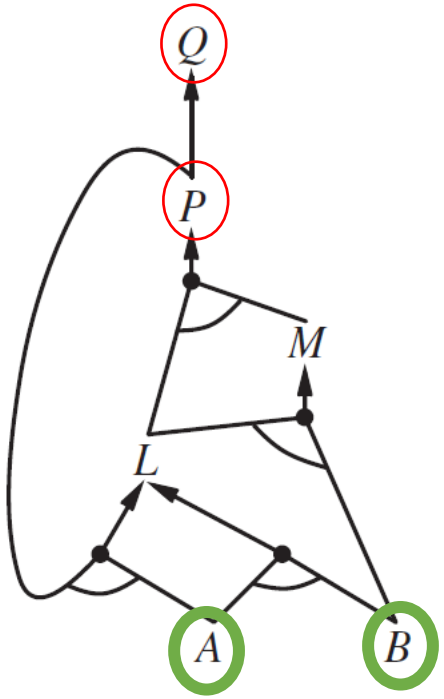
Backward Chaining: Example

KB:

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

Query: Q

AND-OR Graph



Start with Query Q
Is antecedent of Q, i.e., P, true?

Intuition:

Prove antecedents of the query

Backward Chaining: Example

KB:

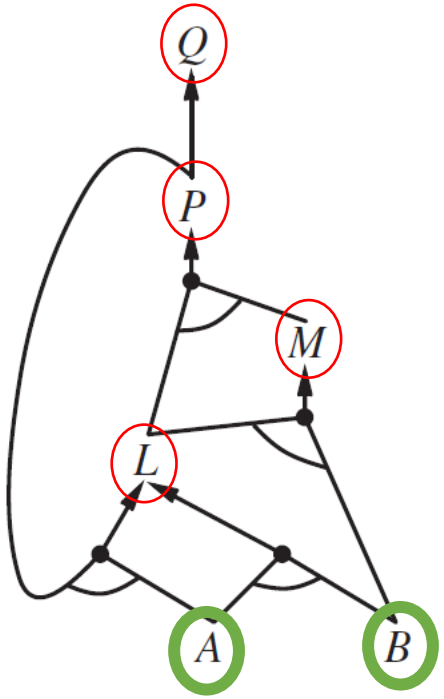
$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$

Query: Q

Intuition:

Prove antecedents of the query

AND-OR Graph



Start with Query Q
Is antecedent of Q, i.e., P, true?
Are antecedents of P true?

Backward Chaining: Example

KB:

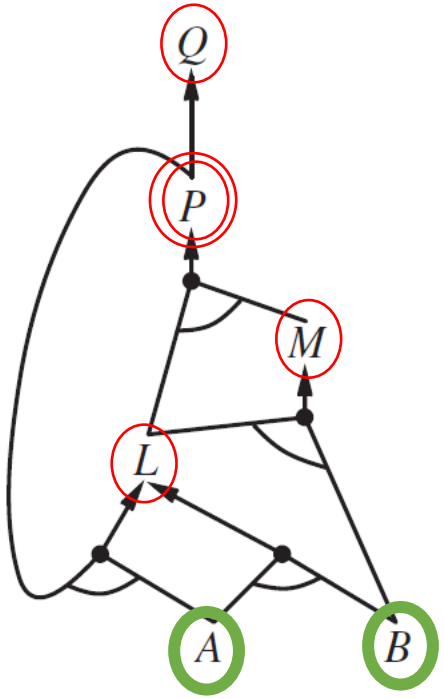
$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$

Query: Q

Intuition:

Prove antecedents of the query

AND-OR Graph



Start with Query Q
Is antecedent of Q, i.e., P, true?
Are antecedents of P true?
Are antecedents of L true?
Yes

Backward Chaining: Example

KB:

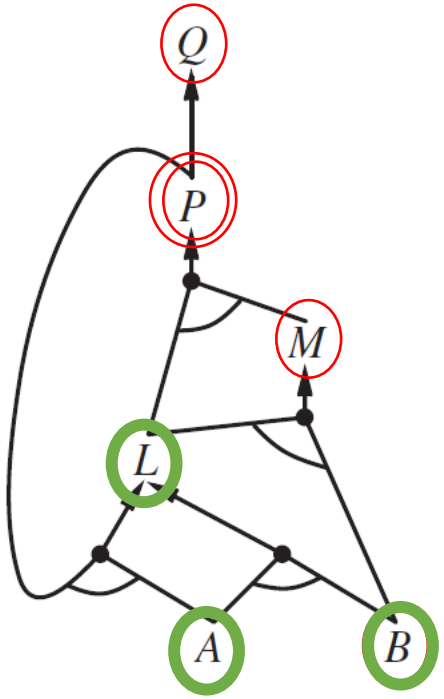
$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$

Query: Q

Intuition:

Prove antecedents of the query

AND-OR Graph



Start with Query Q
Is antecedent of Q, i.e., P, true?
Are antecedents of P true?
Are antecedents of L true?
Yes
Are antecedents of M true?
Yes

Backward Chaining: Example

KB:

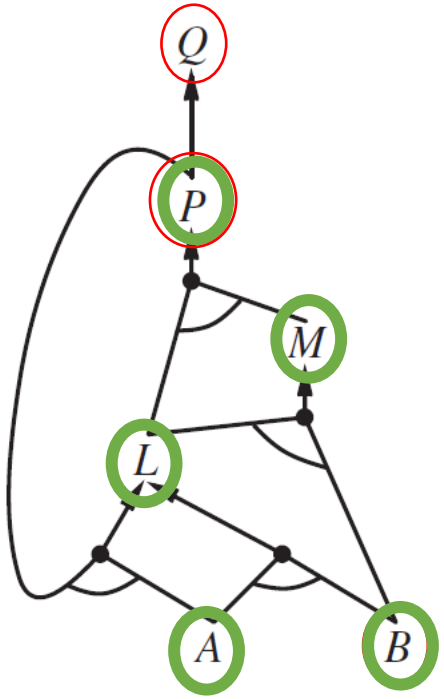
$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$

Query: Q

Intuition:

Prove antecedents of the query

AND-OR Graph



Start with Query Q
Is antecedent of Q, i.e., P, true?
Are antecedents of P true?
Yes
Are antecedents of L true?
Yes
Are antecedents of M true?
Yes

Backward Chaining: Example

KB:

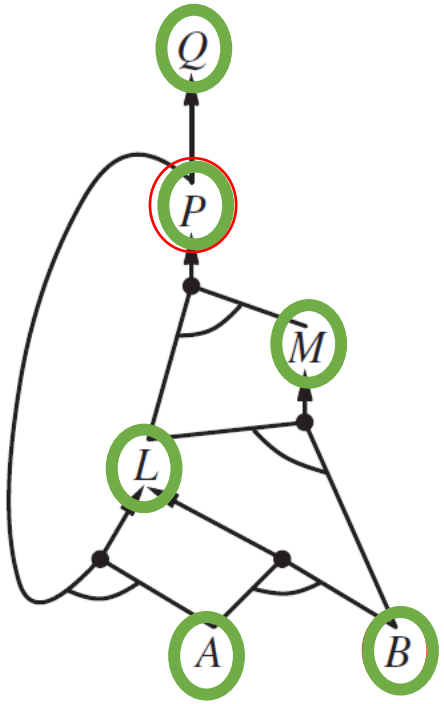
$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$

Query: Q

Intuition:

Prove antecedents of the query

AND-OR Graph



Start with Query Q
Is antecedent of Q, i.e., P, true?
true
Are antecedents of P true?
Yes
Are antecedents of L true?
Yes
Are antecedents of M true?
Yes

Issues with Backward Chaining

- Naïve implementation of BC Cannot handle cycles

- Example:
  - KB:
    - $P \Rightarrow Q$,
    - $Q \Rightarrow R$,
    - $R \Rightarrow P$
  - Query: Q

- Homework:
  - How to eliminate cycles?

Forward Vs Backward Chaining

Forward Chaining

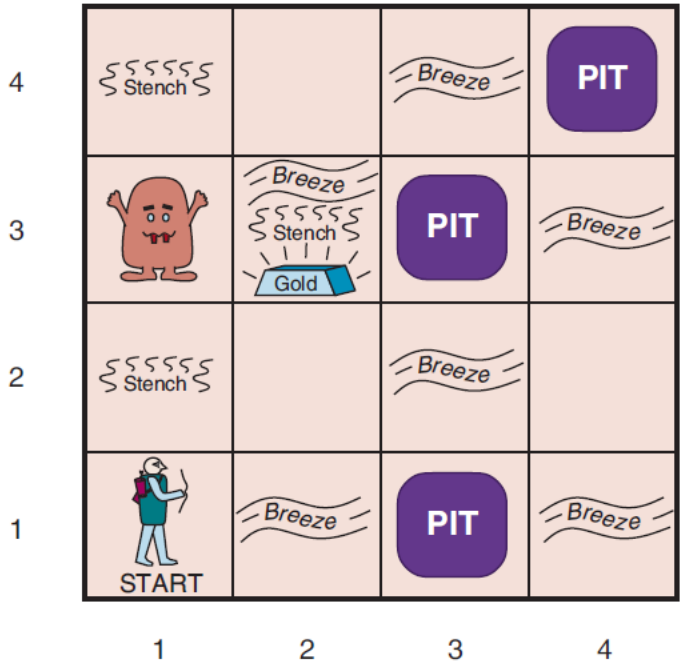
- Data-driven
- Derives many irrelevant facts
- Complexity is linear in size of KB
- Applications:
  - Analysis and Interpretation
    - Eg: DENDRAL – An expert system for determining the molecular structure based on the mass spectral data

Backward Chaining

- Goal-driven
- Lower complexity compared to FC
- Applications:
  - Diagnosis
    - Eg: MYCIN - A medical expert system for diagnosing infectious blood diseases

Issues with Propositional Definite Clauses

- Cannot model uncertainty
  - Breeze implies pit in neighbours
    - ~~$B_{i,j} \Leftrightarrow P_{i-1,j} \vee P_{i+1,j} \vee P_{i,j-1} \vee P_{i,j+1}$~~
  - Pit implies breeze in all neighbours
    - $P_{i,j} \Longrightarrow B_{i-1,j} \wedge B_{i+1,j} \wedge B_{i,j-1} \wedge B_{i,j+1}$
  - Stench implies Wumpus in neighbour
    - ~~$S_{i,j} \Leftrightarrow W_{i-1,j} \vee W_{i+1,j} \vee W_{i,j-1} \vee W_{i,j+1}$~~
  - Wumpus implies stench in all neighbours
    - $W_{i,j} \Longrightarrow S_{i-1,j} \wedge S_{i+1,j} \wedge S_{i,j-1} \wedge S_{i,j+1}$



Summary

- Representation
  - Syntax and Semantics
- Inferencing
  - Model Checking
    - Exhaustive search through truth table enumeration
  - Resolution Method
    - Relies on CNF
  - Proof by Deduction
    - Efficient inferencing with Modus Ponens
    - Restricted form of Propositional Logic (only Horn clauses)