Logical inference problem

**Logical inference problem:**

- **Given:**
  - a knowledge base KB (a set of sentences) and
  - a sentence $\alpha$ (called a **theorem**),
- **How is the logical inference problem defined?**
Logical inference problem

Logical inference problem:
• Given:
  – a knowledge base KB (a set of sentences) and
  – a sentence $\alpha$ (called a theorem),
• Does a KB semantically entail $\alpha$? $KB \models \alpha$
  In other words: In all interpretations in which sentences in the KB are true, is also $\alpha$ true?

Approaches:
• Truth-table approach
• Inference rules
• Conversion to SAT
  – Resolution refutation
Inference problem and satisfiability

How is the logical inference problem related to the satisfiability problem?

\[ KB \models \alpha \quad \text{if and only if} \quad (KB \land \lnot \alpha) \text{ is unsatisfiable} \]
Universal inference rule: Resolution rule

Sometimes inference rules can be combined into a single rule

**Resolution rule**

- sound inference rule that works for KB in CNF
- It is complete for **propositional logic (refutation complete)**

\[
\begin{align*}
A \lor B, & \quad \neg A \lor C \\
\hline 
B \lor C
\end{align*}
\]

**Resolution algorithm**

**Algorithm:**

- Convert KB to the CNF form;
- Apply iteratively the resolution rule starting from \( KB, \neg \alpha \) (in the CNF form)
- Stop when:
  - Contradiction (empty clause) is reached:
    - \( A, \neg A \rightarrow Q \)
    - proves entailment.
  - No more new sentences can be derived
    - disproves it.
Example. Resolution.

**KB**: \((P \land Q) \land (P \Rightarrow R) \land [(Q \land R) \Rightarrow S]\)  \hspace{1cm} \text{Theorem: } S

**Step 1. convert KB to CNF:**
- \(P \land Q \rightarrow P \land Q\)
- \(P \Rightarrow R \rightarrow (\neg P \lor R)\)
- \((Q \land R) \Rightarrow S \rightarrow (\neg Q \lor \neg R \lor S)\)

**KB**: \(P \land Q \land (\neg P \lor R) \land (\neg Q \lor \neg R \lor S)\)

**Step 2. Negate the theorem to prove it via refutation**

\(S \rightarrow \neg S\)

**Step 3. Run resolution on the set of clauses**

\(P \land Q \land (\neg P \lor R) \land (\neg Q \lor \neg R \lor S) \land \neg S\)
Example. Resolution.

KB: \((P \land Q) \land (P \Rightarrow R) \land [(Q \land R) \Rightarrow S]\)  \[\text{Theorem: } S\]

\[
\begin{array}{c}
P \quad Q \\
\begin{array}{c}
(\neg P \lor R) \\
(\neg Q \lor \neg R \lor S) \\
\neg S \\
\end{array}
\end{array}
\]

\[
\begin{array}{c}
R \\
\quad \\
(\neg R \lor S)
\end{array}
\]
Example. Resolution.

KB: \((P \land Q) \land (P \Rightarrow R) \land [(Q \land R) \Rightarrow S]\)  Theorem: \(S\)

\[
\begin{align*}
P &\quad Q & \quad \neg P \lor R &\quad \neg Q \lor \neg R \lor S &\quad \neg S \\
R &\quad & \quad \neg R \lor S &\quad & \quad \neg S \\
\end{align*}
\]

Contradiction \(\{\}\)

Proved: \(S\)
KB in restricted forms

If the sentences in the KB are restricted to some special forms some of the sound inference rules may become complete

Example:

- **Horn form (Horn normal form)**

\[(A \lor \neg B) \land (\neg A \lor \neg C \lor D)\]

Can be written also as:

\[(B \Rightarrow A) \land ((A \land C) \Rightarrow D)\]

Resolution (or modus ponens) are sound and complete for inferences on propositional symbols in the HNF

---

KB in Horn form

- **Horn form**: a clause with **at most one positive literal**

\[(A \lor \neg B) \land (\neg A \lor \neg C \lor D)\]

- **Note**: Not all sentences in propositional logic can be converted into the Horn form

- **KB in Horn normal form**:
  - Two types of propositional statements:
    - **Rules**
      \[\neg(B_1 \lor B_2 \lor \ldots \lor B_k \lor A)\]
      \[\neg((B_1 \land B_2 \land \ldots \land B_k) \lor A)\]
      \[(B_1 \land B_2 \land \ldots \land B_k \Rightarrow A)\]

- **Propositional symbols**: facts \[B\]
KB in Horn form

• Application of the resolution rule:
  – Infers new facts from previous facts
    \[
    \frac{(A \lor \neg B), B}{A}, \quad \frac{(A \lor \neg B), (B \lor \neg C)}{(A \lor C)}
    \]
  – Resolution is sound and complete for inferences on propositional symbols for KB in the Horn normal form (clausal form)

• Similarly, modus ponens is sound and complete when the HNF is written in the implicational form

Complexity of inferences for KBs in HNF

Question: How efficient are the inferences in the HNF?
• If we consider only inferences on propositional symbols
• procedures linear in the size of the KB in the HNF exist.

Terminology:
• Size of a clause: the number of literals it contains.
• Size of the KB in the HNF: the sum of the sizes of its elements.

Example:
\[
A, B, (A \land B \Rightarrow C), (C \Rightarrow D), (C \Rightarrow E), (E \land F \Rightarrow G)
\]
or
\[
A, B, (\neg A \lor \neg B \lor C), (\neg C \lor D), (\neg C \lor E), (\neg E \lor \neg F \lor G)
\]
The size is: 12
Complexity of inferences for KBs in HNF

How to do the inference? If the HNF (is in the clausal form) we can apply resolution.

$A, B, (\neg A \lor \neg B \lor C), (\neg C \lor D), (\neg C \lor E), (\neg E \lor \neg F \lor G)$

Features:
- Every resolution is a **positive unit resolution**; that is, a resolution in which one clause is a positive unit clause (i.e., a proposition symbol).

$A, B, (\neg A \lor \neg B \lor C), (\neg C \lor D), (\neg C \lor E), (\neg E \lor \neg F \lor G)$
Complexity of inferences for KBs in HNF

Features:
- At each resolution, the input clause which is not a unit clause is a logical consequence of the result of the resolution. (Thus, the input clause may be deleted upon completion of the resolution operation.)

\[ A, B, (\neg A \lor \neg B \lor C), (\neg C \lor D), (\neg C \lor E), (\neg E \lor \neg F \lor G) \]
Complexity of inferences for KBs in HNF

Features:
• Following this deletion, the size of the KB (the sum of the lengths of the remaining clauses) is one less than it was before the operation.

\[ A, B, (\neg A \lor B \lor C), (\neg C \lor D), (\neg C \lor E), (\neg E \lor \neg F \lor G) \]

Complexity of inferences for KBs in HNF

Features:
• If \( n \) is the size of the KB, then at most \( n \) positive unit resolutions may be performed on it.

\[ A, B, (\neg A \lor B \lor C), (\neg C \lor D), (\neg C \lor E), (\neg E \lor \neg F \lor G) \]
Complexity of inferences for KBs in HNF

A linear time resolution algorithm:
• The number of positive unit resolutions is limited to the size of the formula (n)
• But to assure overall linear time we need to access each proposition in a constant time:
  • Data structures indexed by proposition names may be accessed in constant time. (This is possible if the proposition names are number in a range (e.g., 1..n), so that array lookup is the access operation.
  • If propositions are accessed by name, then a symbol table is necessary, and the algorithm will run in time $O(n \cdot \log(n))$.

Forward and backward chaining

Two inference procedures based on **modus ponens** for Horn KBs:
• **Forward chaining**
  **Idea:** Whenever the premises of a rule are satisfied, infer the conclusion. Continue with rules that became satisfied.

• **Backward chaining (goal reduction)**
  **Idea:** To prove the fact that appears in the conclusion of a rule prove the premises of the rule. Continue recursively.

Both procedures are **complete for KBs in the Horn form!!!**
Forward chaining example

- **Forward chaining**
  
  **Idea:** Whenever the premises of a rule are satisfied, infer the conclusion. Continue with rules that became satisfied.

Assume the KB with the following rules and facts:

KB:
- R1: $A \land B \Rightarrow C$
- R2: $C \land D \Rightarrow E$
- R3: $C \land F \Rightarrow G$

  | F1 | A |
  | F2 | B |
  | F3 | D |

Theorem: $E$ ?

---

Forward chaining example

**Theorem:** $E$

KB:
- R1: $A \land B \Rightarrow C$
- R2: $C \land D \Rightarrow E$
- R3: $C \land F \Rightarrow G$

  | F1 | A |
  | F2 | B |
  | F3 | D |

---
Forward chaining example

Theorem: \( E \)

KB:

R1: \( A \land B \Rightarrow C \)
R2: \( C \land D \Rightarrow E \)
R3: \( C \land F \Rightarrow G \)

F1: \( A \)
F2: \( B \)
F3: \( D \)

\textbf{Rule R1 is satisfied.}
F4: \( C \)

\textbf{Rule R2 is satisfied.}
F5: \( E \)
Forward chaining

- Efficient implementation: linear in the size of the KB
- Example:

\[
\begin{align*}
P & \Rightarrow Q \\
L \land M & \Rightarrow P \\
B \land L & \Rightarrow M \\
A \land P & \Rightarrow L \\
A \land B & \Rightarrow L \\
A & \\
B
\end{align*}
\]

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
Forward chaining

**

\[ P \Rightarrow Q \]
\[ L \land M \Rightarrow P \]
\[ B \land L \Rightarrow M \]
\[ A \land P \Rightarrow L \]
\[ A \land B \Rightarrow L \]
\[ A \]
\[ B \]

---

Forward chaining

\[ P \Rightarrow Q \]
\[ L \land M \Rightarrow P \]
\[ B \land L \Rightarrow M \]
\[ A \land P \Rightarrow L \]
\[ A \land B \Rightarrow L \]
\[ A \]
\[ B \]
Forward chaining

- $P \Rightarrow Q$
- $L \land M \Rightarrow P$
- $B \land L \Rightarrow M$
- $A \land P \Rightarrow L$
- $A \land B \Rightarrow L$
- $A$
- $B$

Inferred

Add to agenda
Forward chaining

- $P \Rightarrow Q$
- $L \land M \Rightarrow P$
- $B \land L \Rightarrow M$
- $A \land P \Rightarrow L$
- $A \land B \Rightarrow L$
- $A$
- $B$
**Forward chaining**

- $P \Rightarrow Q$
- $L \land M \Rightarrow P$
- $B \land L \Rightarrow M$
- $A \land P \Rightarrow L$
- $A \land B \Rightarrow L$
- $A$
- $B$

**Backward chaining example**

KB:  
R1:  $A \land B \Rightarrow C$
R2:  $C \land D \Rightarrow E$
R3:  $C \land F \Rightarrow G$
F1:  $A$
F2:  $B$
F3:  $D$

- Backward chaining is more focused:
  - tries to prove the theorem only
Backward chaining example

• Backward chaining is more focused:
  – tries to prove the theorem only

KB:
R1: \( A \land B \Rightarrow C \)
R2: \( C \land D \Rightarrow E \)
R3: \( C \land F \Rightarrow G \)
F1: \( A \)
F2: \( B \)
F3: \( D \)

Backward chaining

\[ P \Rightarrow Q \]
\[ L \land M \Rightarrow P \]
\[ B \land L \Rightarrow M \]
\[ A \land P \Rightarrow L \]
\[ A \land B \Rightarrow L \]
\[ A \]
\[ B \]
Backward chaining

- $P \Rightarrow Q$
- $L \land M \Rightarrow P$
- $B \land L \Rightarrow M$
- $A \land P \Rightarrow L$
- $A \land B \Rightarrow L$
- $A$
- $B$

Backward chaining

- $P \Rightarrow Q$
- $L \land M \Rightarrow P$
- $B \land L \Rightarrow M$
- $A \land P \Rightarrow L$
- $A \land B \Rightarrow L$
- $A$
- $B$
Backward chaining

\[ P \Rightarrow Q \quad \text{←} \]
\[ L \land M \Rightarrow P \quad \text{←} \]
\[ B \land L \Rightarrow M \quad \text{←} \]
\[ A \land P \Rightarrow L \quad \text{←} \]
\[ A \land B \Rightarrow L \quad \text{←} \]
\[ A \quad \text{←} \]
\[ B \quad \text{←} \]

Backward chaining

\[ P \Rightarrow Q \quad \text{←} \]
\[ L \land M \Rightarrow P \quad \text{←} \]
\[ B \land L \Rightarrow M \quad \text{←} \]
\[ A \land P \Rightarrow L \quad \text{←} \]
\[ A \land B \Rightarrow L \quad \text{←} \]
\[ A \quad \text{←} \]
\[ B \quad \text{←} \]
Backward chaining

\[ P \Rightarrow Q \quad \begin{array}{l} L \land M \Rightarrow P \\ B \land L \Rightarrow M \\ A \land P \Rightarrow L \\ A \land B \Rightarrow L \end{array} \]

\[ A \quad B \]

CS 2740 Knowledge Representation  M. Hauskrecht
Backward chaining

\[
\begin{align*}
P & \Rightarrow Q \\
L \land M & \Rightarrow P \\
B \land L & \Rightarrow M \\
A \land P & \Rightarrow L \\
A \land B & \Rightarrow L \\
A & \\
B &
\end{align*}
\]
Backward chaining

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

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
KB agents based on propositional logic

- Propositional logic allows us to build knowledge-based agents capable of answering queries about the world by inferring new facts from the known ones
- **Example:** an agent for diagnosis of a bacterial disease

**Facts:**
- The stain of the organism is gram-positive
- The growth conformation of the organism is chains

**Rules:**
- (If) The stain of the organism is gram-positive \( \land \) The morphology of the organism is coccus \( \land \) The growth conformation of the organism is chains
- (Then) \( \Rightarrow \) The identity of the organism is streptococcus

First order logic
Limitations of propositional logic

The world we want to represent and reason about consists of a number of objects with variety of properties and relations among them.

Propositional logic:
- Represents statements about the world without reflecting this structure and without modeling these entities explicitly.

Consequence:
- Some knowledge is hard or impossible to encode in the propositional logic.
- Two cases that are hard to represent:
  - Statements about similar objects, relations
  - Statements referring to groups of objects.

To derive:
- Statements about similar objects and relations need to be enumerated.

Example: Seniority of people domain

Assume we have: John is older than Mary
               Mary is older than Paul

To derive John is older than Paul we need:

John is older than Mary \∧ Mary is older than Paul
⇒ John is older than Paul

Assume we add another fact: Jane is older than Mary
To derive Jane is older than Paul we need:

Jane is older than Mary \∧ Mary is older than Paul
⇒ Jane is older than Paul

What is the problem?
Limitations of propositional logic

• Statements about similar objects and relations needs to be enumerated

• Example: Seniority of people domain

Assume we have:  John is older than Mary
                   Mary is older than Paul

To derive  John is older than Paul  we need:
                   John is older than Mary  ∧  Mary is older than Paul
                   ⇒  John is older than Paul

Assume we add another fact: Jane is older than Mary

To derive  Jane is older than Paul  we need:
                   Jane is older than Mary  ∧  Mary is older than Paul
                   ⇒  Jane is older than Paul

Problem: KB grows large
Limitations of propositional logic

• **Statements about similar objects and relations needs to be enumerated**

• **Example:** Seniority of people domain
  
  For inferences we need:
  
  \[ \text{John is older than Mary} \land \text{Mary is older than Paul} \Rightarrow \text{John is older than Paul} \]
  
  \[ \text{Jane is older than Mary} \land \text{Mary is older than Paul} \Rightarrow \text{Jane is older than Paul} \]

• **Problem:** if we have many people and facts about their seniority we need represent many rules like this to allow inferences

• **Possible solution:** introduce variables

  \[ \text{Pers}_A \text{ is older than } \text{Pers}_B \land \text{Pers}_B \text{ is older than } \text{Pers}_C \Rightarrow \text{Pers}_A \text{ is older than } \text{Pers}_C \]

---

Limitations of propositional logic

• **Statements referring to groups of objects require exhaustive enumeration of objects**

• **Example:**
  
  Assume we want to express \( \text{Every student likes vacation} \)
  
  Doing this in propositional logic would require to include statements about every student

  \[ \text{John likes vacation} \land \cdot \cdot \cdot \land \text{Ann likes vacation} \land \cdot \cdot \cdot \]

• **Solution:** Allow quantification in statements
First-order logic (FOL)

• More expressive than propositional logic

• **Eliminates deficiencies of PL by:**
  – Representing objects, their properties, relations and statements about them;
  – Introducing variables that refer to an arbitrary objects and can be substituted by a specific object
  – Introducing quantifiers allowing us to make statements over groups objects without the need to represent each of them separately