Propositional logic: Inference

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Administration

• PS–3 due today (before the class)
  – Report
  – Programs through ftp

• PS–4 is out
  – on the course web page
  – due next week on Tuesday, October 1, 2002
    • Report
    • Programs
Knowledge-based agent

- **Knowledge base (KB):**
  - A set of sentences that describe facts about the world in some formal (representational) language
  - Domain specific
- **Inference engine:**
  - A set of procedures that work upon the representational language and can infer new facts or answer KB queries
  - Domain independent

Knowledge representation

- The **objective of knowledge representation** is to express the knowledge about the world in a computer-tractable form

- Key aspects of knowledge representation languages:
  - **Syntax:** describes how sentences are formed in the language
  - **Semantics:** describes the meaning of sentences, what is it the sentence refers to in the real world
  - **Computational aspect:** describes how sentences and objects are manipulated in concordance with semantic conventions

Many KB systems rely on some variant of logic
Logic

A formal language for expressing knowledge and ways of reasoning.

Logic is defined by:

• **A set of sentences**
  – A sentence is constructed from a set of primitives according to syntax rules.

• **A set of interpretations**
  – An interpretation gives a semantic to primitives. It associates primitives with values.

• **The valuation (meaning) function** $V$
  – Assigns a value (typically the truth value) to a given sentence under some interpretation

  $$V : \text{sentence} \times \text{interpretation} \rightarrow \{\text{True}, \text{False}\}$$

Types of logic

• Different types of logics possible:
  – **Propositional logic**
  – First-order logic
  – Temporal logic
  – Numerical constraints logic
  – Map-coloring logic

  In the following:
  • **Propositional logic.**
    – Formal language for making logical inferences
    – Foundations of **propositional logic**: George Boole (1854)
Propositional logic. Syntax

- Propositional logic P:
  - defines a language for symbolic reasoning

First step: define Syntax + interpretation + semantics of P

Syntax:
- Symbols (alphabet) in P:
  - Constants: True, False
  - A set of propositional variables (propositional symbols):
    Examples: \( P, Q, R, \ldots \) or statements like:
    - Light in the room is on,
    - It rains outside, etc.
  - A set of connectives:
    \( \neg, \wedge, \vee, \Rightarrow, \Leftrightarrow \)

Sentences in the propositional logic:
- Atomic sentences:
  - Constructed from constants and propositional symbols
  - True, False are (atomic) sentences
  - \( P, Q \) or Light in the room is on, It rains outside are (atomic) sentences
- Composite sentences:
  - Constructed from valid sentences via connectives
  - If \( A, B \) are sentences then
    \[ \neg A \quad (A \wedge B) \quad (A \vee B) \quad (A \Rightarrow B) \quad (A \Leftrightarrow B) \]
    or
    \[ (A \vee B) \wedge (A \vee \neg B) \]
    are sentences
Propositional logic. Semantics.

The semantic gives the meaning to sentences.

In the propositional logic the semantics is defined by:

1. **Interpretation of propositional symbols and constants**
   - Semantics of atomic sentences

2. **Through the meaning of connectives**
   - Meaning (semantics) of composite sentences

Semantic: propositional symbols

A **propositional symbol** (an atomic sentence) can stand for an arbitrary fact (statement) about the world

Examples: “Light in the room is on”,

“It rains outside”, etc.

- An **interpretation** maps symbols to one of the two values: **True** (T), or **False** (F), depending on whether the symbol is satisfied in the world
  - I: **Light in the room is on** -> **True**, **It rains outside** -> **False**
  - I’: **Light in the room is on** -> **False**, **It rains outside** -> **False**

- The **meaning (value)** of the propositional symbol for a specific interpretation is given by its interpretation
  \[
  V(\text{Light in the room is on, } I) = \text{True}\\
  V(\text{Light in the room is on, } I’) = \text{False}
  \]
Semantics: constants

- **The meaning (truth) of constants:**
  - True and False constants are always (under any interpretation) assigned the corresponding *True, False* value.

\[
\begin{align*}
V(True, I) &= True \\
V(False, I) &= False
\end{align*}
\]

For any interpretation \( I \)

Semantics: composite sentences.

- **The meaning (truth value) of complex propositional sentences.**
  - Determined using the “standard” rules for combining logical sentences:

<table>
<thead>
<tr>
<th>( P )</th>
<th>( Q )</th>
<th>( \neg P )</th>
<th>( P \land Q )</th>
<th>( P \lor Q )</th>
<th>( P \Rightarrow Q )</th>
<th>( P \Leftrightarrow Q )</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td></td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td></td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td></td>
</tr>
</tbody>
</table>
Some definitions

- A **model (in logic)**: An interpretation is a model for a set of sentences if it assigns true to each sentence in the set.
- A sentence is **satisfiable** if it has a model;
  - There is at least one interpretation under which the sentence can evaluate to True.
- A sentence is **valid** if it is *True* in all interpretations
  - i.e., if its negation is **not satisfiable** (leads to contradiction)

<table>
<thead>
<tr>
<th>P</th>
<th>Q</th>
<th>P ∨ Q</th>
<th>(P ∨ Q) ∧ ¬Q</th>
<th>((P ∨ Q) ∧ ¬Q) ⇒ P</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
</tbody>
</table>

Entailment

- **Entailment** reflects the relation of one fact in the world following from the others

- Entailment $KB \models \alpha$
- Knowledge base KB entails sentence $\alpha$ if and only if $\alpha$ is true in all worlds where KB is true
Sound and complete inference.

Inference is a process by which conclusions are reached.

Our goal:
• We want to implement the inference process on a computer!!

Assume an inference procedure $i$ that
• derives a sentence $\alpha$ from the KB: $KB \vdash_i \alpha$

Properties of the inference procedure in terms of entailment
• Soundness: An inference procedure is sound
  
  If $KB \vdash_i \alpha$ then it is true that $KB \models \alpha$

• Completeness: An inference procedure is complete
  
  If $KB \models \alpha$ then it is true that $KB \vdash_i \alpha$

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?

Question: Is there a procedure (program) that can decide this problem in a finite number of steps?

Answer: Yes. Logical inference problem for the propositional logic is decidable.
Solving logical inference problem

In the following:

**How to design the procedure that answers:**

\[ KB \models \alpha \]?

**Three approaches:**

- Truth-table approach
- Inference rules
- Conversion to the inverse SAT problem
  - Resolution-refutation

---

**Truth-table approach**

**Problem:** \( KB \models \alpha \)?

- We need to check all possible interpretations for which the KB is true (models of KB) whether \( \alpha \) is true for each of them

**Truth tables:**

- Enumerate truth values of sentences for all possible interpretations (assignments of True/False values to propositional symbols)

**Example:**

<table>
<thead>
<tr>
<th>( P )</th>
<th>( Q )</th>
<th>( P \lor Q )</th>
<th>( P \iff Q )</th>
<th>( (P \lor \neg Q) \land Q )</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>
**Truth-table approach**

**Problem:** \( KB \models \alpha \) ?

- We need to check all possible interpretations for which the KB is true (models of KB) whether \( \alpha \) is true for each of them

**Truth tables:**

- enumerate truth values of sentences for all possible interpretations (assignments of True/False to propositional symbols)

**Example:**

<table>
<thead>
<tr>
<th>( P )</th>
<th>( Q )</th>
<th>( P \lor Q )</th>
<th>( P \iff Q )</th>
<th>( (P \lor Q) \land Q )</th>
<th>( \alpha )</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>

**Truth-table approach**

A two steps procedure:

1. Generate table for all possible interpretations
2. Check whether the sentence \( \alpha \) evaluates to true whenever \( KB \) evaluates to true

**Example:** \( KB = (A \lor C) \land (B \lor \neg C) \) \( \alpha = (A \lor B) \)

<table>
<thead>
<tr>
<th>( A )</th>
<th>( B )</th>
<th>( C )</th>
<th>( A \lor C )</th>
<th>( (B \lor \neg C) )</th>
<th>( KB )</th>
<th>( \alpha )</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>
Truth-table approach

A two steps procedure:
1. Generate table for all possible interpretations
2. Check whether the sentence $\alpha$ evaluates to true whenever $KB$ evaluates to true

Example: $KB = (A \lor C) \land (B \lor \neg C) \quad \alpha = (A \lor B)$

<table>
<thead>
<tr>
<th>$A$</th>
<th>$B$</th>
<th>$C$</th>
<th>$A \lor C$</th>
<th>$(B \lor \neg C)$</th>
<th>$KB$</th>
<th>$\alpha$</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
</tbody>
</table>

$KB$ entails $\alpha$

- The truth-table approach is sound and complete for the propositional logic!!
Inference rules approach.

\[ KB \models \alpha ? \]

**Problem with the truth table approach:**
- the truth table is **exponential** in the number of propositional symbols (we checked all assignments)
- KB is true on only a smaller subset

**Idea:** Can we check only entries for which KB is True?

**Solution:** apply inference rules to sentences in the KB

**Inference rules:**
- Represent sound inference patterns repeated in inferences
- Can be used to generate new (sound) sentences from the existing ones

---

**Inference rules for logic**

- **Modus ponens**

\[
\frac{A \Rightarrow B, \quad A}{B} \quad \text{premise} \quad \text{conclusion}
\]

- If both sentences in the premise are true then conclusion is true.
- The modus ponens inference rule is **sound**.
  - We can prove this through the truth table.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( A )</td>
<td>( B )</td>
<td>( A \Rightarrow B )</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
</tbody>
</table>
Inference rules for logic

• And-elimination

\[
\frac{A_1 \land A_2 \land A_n}{A_i}
\]

• And-introduction

\[
\frac{A_1, A_2, A_n}{A_1 \land A_2 \land A_n}
\]

• Or-introduction

\[
\frac{A_i}{A_1 \lor A_2 \lor \ldots A_i \lor A_n}
\]

All of the above inference rules are sound. We can prove this through the truth table, similarly to the modus ponens case.

Elimination of double negation

\[
\frac{\neg\neg A}{A}
\]

Unit resolution

\[
\frac{A \lor B, \neg A}{B}
\]

Resolution

\[
\frac{A \lor B, \neg B \lor C}{A \lor C}
\]

A special case of

CS 1571 Intro to AI
Example. Inference rules approach.

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

1. \( P \land Q \)
2. \( P \Rightarrow R \)
3. \( (Q \land R) \Rightarrow S \)

4. \( P \quad \text{From 1 and And-elim} \)
   \[
   \frac{A_1 \land A_2 \land \ldots \land A_n}{A_i}
   \]
Example. Inference rules approach.

**KB:** \( P \land Q \quad P \Rightarrow R \quad (Q \land R) \Rightarrow S \quad \text{Theorem: } S \)

1. \( P \land Q \)
2. \( P \Rightarrow R \)
3. \( (Q \land R) \Rightarrow S \)
4. \( P \)
5. \( R \)  
6. \( Q \)  

From 1 and And-elim

\[
\frac{A_1 \land A_2 \land \ldots \land A_n}{A_i}
\]

---

CS 1571 Intro to AI
Example. Inference rules approach.

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

1. \( P \land Q \)
2. \( P \Rightarrow R \)
3. \( (Q \land R) \Rightarrow S \)
4. \( P \)
5. \( R \)
6. \( Q \)
7. \( (Q \land R) \) \hspace{1cm} **From 5,6 and And-introduction**
   \[
   \frac{A_1, A_2, \ldots, A_n}{A_1 \land A_2 \land \ldots \land A_n}
   \]

---

Example. Inference rules approach.

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

1. \( P \land Q \)
2. \( P \Rightarrow R \)
3. \( (Q \land R) \Rightarrow S \)
4. \( P \)
5. \( R \)
6. \( Q \) \hspace{1cm} \[ A \Rightarrow B, \quad A \quad \quad \frac{A \Rightarrow B, \quad A}{B} \]
7. \( (Q \land R) \)
8. \( S \) \hspace{1cm} **From 7,3 and Modus ponens**

---

Proved: \( S \)
Example. Inference rules approach.

**KB:** $P \land Q \quad P \implies R \quad (Q \land R) \implies S \quad \textbf{Theorem:} S$

1. $P \land Q$
2. $P \implies R$
3. $(Q \land R) \implies S$
4. $P$ \quad From 1 and And-elim
5. $R$ \quad From 2,4 and Modus ponens
6. $Q$ \quad From 1 and And-elim
7. $(Q \land R)$ \quad From 5,6 and And-introduction
8. $S$ \quad From 7,3 and Modus ponens

**Proved:** $S$

---

Inference rules

- To show that theorem $\alpha$ holds for a KB
  - we may need to apply a number of sound inference rules

**Problem:** many possible inference rules to be applied next

**Looks familiar?**

```
\begin{array}{c}
P \implies Q, \quad P \\
R \implies S \\
P \\
R \\
\vdots
\end{array}
```

```
\begin{array}{c}
P \implies Q, \quad P \\
Q \\
R \implies S, \quad R \\
S
\end{array}
```
Logic inferences and search

- To show that theorem $\alpha$ holds for a KB
  - we may need to apply a number of sound inference rules

**Problem:** many possible inference rules to be applied next

**Looks familiar?**

This is an instance of a search problem:

Truth table method (from the search perspective):
- blind enumeration and checking

Logic inferences and search

**Inference rule method as a search problem:**
- **State:** a set of sentences that are known to be true
- **Initial state:** a set of sentences in the KB
- **Operators:** applications of inference rules
  - Allow us to add new sound sentences to old ones
- **Goal state:** a theorem $\alpha$ is derived from KB

**Logic inference:**
- **Proof:** A sequence of sentences that are immediate consequences of applied inference rules
- **Theorem proving:** process of finding a proof of theorem
Normal forms

Sentences in the propositional logic can be transformed into one of the normal forms. This can simplify the inferences.

Normal forms used:

Conjunctive normal form (CNF)
- conjunction of clauses (clauses include disjunctions of literals)
  \[(A \lor B) \land (\neg A \lor \neg C \lor D)\]

Disjunctive normal form (DNF)
- Disjunction of terms (terms include conjunction of literals)
  \[(A \land \neg B) \lor (\neg A \land C) \lor (C \land \neg D)\]

Conversion to a CNF

**Assume:** \(\neg(A \Rightarrow B) \lor (C \Rightarrow A)\)
1. Eliminate \(\Rightarrow, \Leftrightarrow\)
   \[\neg(\neg A \lor B) \lor (\neg C \lor A)\]
2. Reduce the scope of signs through DeMorgan Laws and double negation
   \[(A \land \neg B) \lor (\neg C \lor A)\]
3. Convert to CNF using the associative and distributive laws
   \[(A \lor \neg C \lor A) \land (\neg B \lor \neg C \lor A)\]
   and
   \[(A \lor \neg C) \land (\neg B \lor \neg C \lor A)\]
Satisfiability (SAT) problem

Determine whether a sentence in the conjunctive normal form (CNF) is satisfiable (i.e. can evaluate to true)

\[(P \lor Q \lor \neg R) \land (\neg P \lor \neg R \lor S) \land (\neg P \lor Q \lor \neg T)\ldots\]

It is an instance of a constraint satisfaction problem:

- **Variables:**
  - Propositional symbols \((P, R, T, S)\)
  - Values: *True, False*

- **Constraints:**
  - Every conjunct must evaluate to true, at least one of the literals must evaluate to true

- **All techniques developed for CSPs can be applied to solve the logical inference problem ! !

---

Relationship between inference problem and satisfiability

**Inference problem:**

- we want to show that the sentence \(\alpha\) is entailed by KB

**Satisfiability:**

- The sentence is satisfiable if there is some assignment (interpretation) under which the sentence evaluates to true

**Connection:**

\[KB \models \alpha \quad \text{if and only if} \quad (KB \land \neg \alpha) \text{ is unsatisfiable}\]

**Consequences:**

- inference problem is NP-complete
- programs for solving the SAT problem can be used to solve the inference problem
Universal inference rule: Resolution rule

Sometimes inference rules can be combined into a single rule

Resolution rule

• sound inference rule that works for CNF
• It is complete for propositional logic (refutation complete)

\[
\frac{A \lor B, \lnot A \lor C}{B \lor C}
\]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>( A \lor B )</th>
<th>( \lnot B \lor C )</th>
<th>( A \lor C )</th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
</tbody>
</table>

Initial obstacle:

• Repeated application of the resolution rule to a KB in CNF may fail to derive new valid sentences

Example:

We know: \((A \land B)\) We want to show: \((A \lor B)\)

Resolution rule fails to derive it (incomplete ??)

A trick to make things work:

• proof by contradiction
  – Disproving: \(KB, \lnot \alpha\)
  – Proves the entailment \(KB \models \alpha\)
Resolution algorithm

Algorithm:
• Convert KB to the CNF form;
• Apply iteratively the resolution rule starting from \( KB, \neg \alpha \) (in 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: \((P \land Q) \land (P \Rightarrow R) \land [(Q \land R) \Rightarrow S]\)  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: \( P \quad Q \quad (\neg P \lor R) \quad (\neg Q \lor \neg R \lor S) \)

Step 2. Negate the theorem to prove it via refutation
\( S \rightarrow \neg S \)

Step 3. Run resolution on the set of clauses
\( P \quad Q \quad (\neg P \lor R) \quad (\neg Q \lor \neg R \lor S) \quad \neg S \)
Example. Resolution.

\begin{align*}
\text{KB:} & \quad (P \land Q) \land (P \Rightarrow R) \land [(Q \land R) \Rightarrow S] \\
\text{Theorem:} & \quad S
\end{align*}

\begin{align*}
P & \quad Q \\
& \quad (\neg P \lor R) \\
& \quad (\neg Q \lor \neg R \lor S) \\
R & \quad (\neg R \lor S) \\
S & \quad \text{Contradiction} \quad \Rightarrow \quad \emptyset \\
\text{Proved:} & \quad S
\end{align*}

Horn clauses

A special type of clause with at most one positive literal

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

Can be written also as: \((B \Rightarrow A) \land ((A \land C) \Rightarrow D)\)

- Two types of propositional statements:
  - Implications: called rules \((B \Rightarrow A)\)
  - Propositional symbols: facts \(B\)

Modus ponens:

- is the “universal “(complete) rule for the sentences in the Horn form

\[
\begin{array}{c}
A \Rightarrow B, \quad A \\
\hline
B
\end{array} \quad \quad \quad \quad \begin{array}{c}
A_1 \land A_2 \land \ldots \land A_k \Rightarrow B, \quad A_1, A_2, \ldots A_k \\
\hline
B
\end{array}
\]
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 |

Rule R1 is satisfied.

| F4 | C |

Rule R2 is satisfied.

| F5 | E |

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