Propositional logic

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

Knowledge representation
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 use the representational language to infer new facts from known ones or answer a variety of KB queries. Inferences typically require search.
  - Domain independent

---

**Example: MYCIN**

- MYCIN: an expert system for diagnosis of bacterial infections
- **Knowledge base** represents
  - Facts about a specific patient case
  - Rules describing relations between entities in the bacterial infection domain

```
If
  1. The stain of the organism is gram-positive, and
  2. The morphology of the organism is coccus, and
  3. The growth conformation of the organism is chains
Then
  the identity of the organism is streptococcus
```

- **Inference engine:**
  - manipulates the facts and known relations to answer diagnostic queries (consistent with findings and rules)
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 semantical 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.

\[
V: \text{sentence} \times \text{interpretation} \rightarrow \{True, False\}
\]
Example of logic

Language of numerical constraints:
• A sentence:
  \[ x + 3 \leq z \]
  \( x, z \) - variable symbols (primitives in the language)

• An interpretation:
  I: \( x = 5, z = 2 \)
  Variables mapped to specific real numbers

• Valuation (meaning) function \( V \):
  \[ V( x + 3 \leq z, I) \text{ is } \text{False} \text{ for } I: x = 5, z = 2 \]
  \[ \text{is } \text{True} \text{ for } I: x = 5, z = 10 \]

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

- **Proposition:** a statement that is either true or false
- Examples of propositions:
  - *Pitt is located in the Oakland section of Pittsburgh.*
  - *France is in Europe.*
  - *It rains outside.*
  - *2 is a prime number and 6 is a prime number.*
  - *How are you?* Not a proposition.

Propositional logic. Syntax

- **Formally propositional logic P:**
  - Is defined by **Syntax + interpretation + semantics of P**

  **Syntax:**
  - **Symbols (alphabet) in P:**
  - **Constants:** *True, False*
  - **Propositional symbols**
    - Examples:
      - *P*
      - *Pitt is located in the Oakland section of Pittsburgh.*
      - *It rains outside,* etc.
  - **A set of connectives:**
    - ¬, ∧, ∨, →, ↔
Propositional logic. Syntax

Sentences in the propositional logic:

• **Atomic sentences:**
  – Constructed from constants and propositional symbols
  – True, False are (atomic) sentences
  – $P \cdot Q$ 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
    \[ \neg A \quad (A \land B) \quad (A \lor B) \quad (A \Rightarrow B) \quad (A \Leftrightarrow B) \]
    or \[ (A \lor B) \land (A \lor \neg B) \]
    are sentences

Propositional logic. Semantics.

The semantic gives the meaning to sentences.

the semantics in the propositional logic 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
• a statement about the world that is either true or false

Examples:
– Pitt is located in the Oakland section of Pittsburgh
– It rains outside
– Light in the room is on

• 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

Semantic: propositional symbols

The meaning (value) of the propositional symbol for a specific interpretation is given by its interpretation

I: Light in the room is on -> True, It rains outside -> False

\[ V(\text{Light in the room is on}, I) = \text{True} \]
\[ V(\text{It rains outside}, I) = \text{False} \]

I’: Light in the room is on -> False, It rains outside -> False

\[ V(\text{Light in the room is on}, I’) = \text{False} \]
Semantics: constants

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

  \[
  V(\text{True}, I) = \text{True} \\
  V(\text{False}, I) = \text{False}
  \]

  For any interpretation \( I \)

Semantics: composite sentences.

- **The meaning (truth value) of complex propositional sentences.**
  - Determined using the standard rules of logic:

<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>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
<td>False</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>True</td>
<td>False</td>
<td>True</td>
</tr>
</tbody>
</table>
Translation

Assume the following sentences:
• It is not sunny this afternoon and it is colder than yesterday.
• We will go swimming only if it is sunny.
• If we do not go swimming then we will take a canoe trip.
• If we take a canoe trip, then we will be home by sunset.

Denote:
• \( p = \) It is sunny this afternoon
• \( q = \) it is colder than yesterday
• \( r = \) We will go swimming
• \( s = \) we will take a canoe trip
• \( t = \) We will be home by sunset
Translation

Assume the following sentences:
• It is not sunny this afternoon and it is colder than yesterday. \( \neg p \land q \)
• We will go swimming only if it is sunny. \( r \rightarrow p \)
• If we do not go swimming then we will take a canoe trip. \( \neg r \rightarrow s \)
• If we take a canoe trip, then we will be home by sunset.

Denote:
• \( p \) = It is sunny this afternoon
• \( q \) = it is colder than yesterday
• \( r \) = We will go swimming
• \( s \) = we will take a canoe trip
• \( t \) = We will be home by sunset
Translation

Assume the following sentences:

• It is not sunny this afternoon and it is colder than yesterday. \( \neg p \land q \)
• We will go swimming only if it is sunny. \( r \rightarrow p \)
• If we do not go swimming then we will take a canoe trip. \( \neg r \rightarrow s \)
• If we take a canoe trip, then we will be home by sunset. \( s \rightarrow t \)

Denote:

• \( p = \) It is sunny this afternoon
• \( q = \) it is colder than yesterday
• \( r = \) We will go swimming
• \( s = \) we will take a canoe trip
• \( t = \) We will be home by sunset

Contradiction and Tautology

Some composite sentences may always (under any interpretation) evaluate to a single truth value:

• \textbf{Contradiction} (always \textit{False})
  \[ P \land \neg P \]

• \textbf{Tautology} (always \textit{True})
  \[ P \lor \neg P \]

\[
\begin{align*}
\neg (P \lor Q) & \iff (\neg P \land \neg Q) \\
\neg (P \land Q) & \iff (\neg P \lor \neg Q)
\end{align*}
\]
DeMorgan’s Laws
Model, validity and satisfiability

• 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 \lor Q)</th>
<th>((P \lor Q) \land \neg Q)</th>
<th>(((P \lor Q) \land \neg Q) \Rightarrow 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>

Model, validity and satisfiability

• 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 \lor Q)</th>
<th>((P \lor Q) \land \neg Q)</th>
<th>(((P \lor Q) \land \neg Q) \Rightarrow 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>
Model, validity and satisfiability

- 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>Satisfiable sentence</th>
<th>Valid sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P \lor Q )</td>
<td>( (P \lor Q) \land \neg \neg Q )</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>
Entailment

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

```
Sentences --------> Entails --------> Sentence
                  
Facts --------> Follows --------> Fact
```

- 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.
- 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 table:
- enumerates 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>False</td>
<td>True</td>
<td>False</td>
</tr>
</tbody>
</table>

\[ \alpha \]
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 table:**
- enumerates truth values of sentences for all possible interpretations (assignments of True/False to propositional symbols)

**Example:**

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

CS 2710 Foundations of AI
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></th>
<th></th>
<th></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>False</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>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>False</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>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
</tbody>
</table>

CS 2710 Foundations of AI
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>True</td>
<td>False</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>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>
<td>False</td>
</tr>
</tbody>
</table>

• The truth-table approach is sound and complete for the propositional logic!!
Limitations of the truth table approach.

\[ KB \models \alpha \ ? \]

What is the computational complexity of the truth table approach?

- \( \alpha = |KB| \)

Exponential in the number of the proposition symbols

\[ 2^n \] Rows in the table has to be filled
**Limitations of the truth table approach.**

$$KB \models \alpha$$

**What is the computational complexity of the truth table approach?**

Exponential in the number of the proposition symbols

$$2^n$$ Rows in the table has to be filled

**But typically only for a small subset of rows the KB is true**

---

**Limitations of the truth table 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
Limitation of the truth table 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 only on a small subset interpretations

**How to make the process more efficient?**

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 only on a smaller subset

**How to make the process more efficient?**

**Solution:** check only entries for which KB is *True.*
This is the idea behind the inference rules approach

**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

\[
\frac{A \Rightarrow B, \quad A}{B}
\]

– 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.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A ⇒ B</th>
</tr>
</thead>
<tbody>
<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

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

• And-introduction

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

• Or-introduction

\[
\frac{A_i}{A_1 \lor A_2 \lor \ldots \lor A_i \lor A_n}
\]
## Inference rules for logic

<table>
<thead>
<tr>
<th>Inference Rule</th>
<th>Logical Formulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elimination of double negation</td>
<td>( \neg \neg A \vdash A )</td>
</tr>
<tr>
<td>Unit resolution</td>
<td>( A \lor B, \neg A \vdash B )</td>
</tr>
<tr>
<td>Resolution</td>
<td>( A \lor B, \neg B \lor C \vdash A \lor C )</td>
</tr>
</tbody>
</table>

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

### Example. Inference rules approach.

**KB:** \( P \land Q \quad P \Rightarrow R \quad (Q \land R) \Rightarrow S \)  

**Theorem:** \( S \)

1. \( P \land Q \)
2. \( P \Rightarrow R \)
3. \( (Q \land R) \Rightarrow S \)
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. \( P \) \hspace{1cm} \text{From 1 and And-elim}
   \[
   \frac{A_1 \land A_2 \land \ldots \land A_n}{A_i}
   \]
5. \( R \) \hspace{1cm} \text{From 2, 4 and Modus ponens}
   \[
   \frac{A \Rightarrow B, \quad A}{B}
   \]
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. \( Q \)

\[
\begin{array}{c}
A_1 \land A_2 \land \ldots \land A_n \\
A_i
\end{array}
\]

From 1 and And-elim

CS 2710 Foundations of AI

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. \( Q \)
7. \( (Q \land R) \)

\[
\begin{array}{c}
A_1, A_2, \ldots, A_n \\
A_1 \land A_2 \land \ldots \land A_n
\end{array}
\]

From 5,6 and And-introduction
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. \( (Q \land R) \)
8. \( S \)

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?*

\[
\begin{align*}
P \Rightarrow Q \\
R \Rightarrow S \\
P \\
R \\
\ldots
\end{align*}
\]

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 rules to be can be applied next

*Looks familiar?*

\[
\begin{align*}
P \Rightarrow Q, & P \\
Q \\
R \Rightarrow S, & R \\
S \\
\ldots
\end{align*}
\]

*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 \( \Rightarrow, \iff \)

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

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: \textit{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. Why?**
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)

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

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

---

CS 2710 Foundations of AI
Universal rule: Resolution.

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: \( KB \land \neg \alpha \)
  - Proves the entailment \( KB \models \alpha \)

Resolution algorithm

Algorithm:
- Convert KB to the CNF form;
- Apply iteratively the resolution rule starting from \( KB \land \neg \alpha \) (in CNF form)
- Stop when:
  - Contradiction (empty clause) is reached:
    - \( A \land \neg A \rightarrow \bot \)
    - 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:**
- \(P \land Q \quad \rightarrow \quad P \land Q\)
- \(P \Rightarrow R \quad \rightarrow \quad (\neg P \lor R)\)
- \((Q \land R) \Rightarrow S \quad \rightarrow \quad (\neg Q \lor \neg R \lor S)\)

**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**

\(S \quad \rightarrow \quad \neg S\)

**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.

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

\[
P Q \quad (\neg P \lor R) \quad (\neg Q \lor \neg R \lor S) \quad \neg S
\]

\[
R
\]

\[
R \quad (\neg R \lor S)
\]
Example. Resolution.

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

Theorem: \(S\)

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

Contradiction \(\{\}\)  

Proved: \(S\)