Components of Propositional Logic

- Logic constants: True (1), and False (0)
- Propositional variables: \( X = \{p_1, q_2, \ldots\} \), a set of Boolean variables
- Logical connectives: \( F = \{\neg, \land, \lor, \Rightarrow, \Leftrightarrow, \ldots\} \)
- Logical sentences: \( L(X, F) \), expressions built from \( X \) and \( F \)
- Logical interpretations: a mapping \( \theta \) from \( X \) to \( \{0, 1\} \)
- Logical evaluations: a process of applying a mapping \( \theta \) to sentences in \( L(X, F) \) (to obtain a value in \( \{0, 1\} \))

Properties of Logical Connectives

- \( \land \) and \( \lor \) are commutative
  \[ \varphi_1 \land \varphi_2 \equiv \varphi_2 \land \varphi_1 \]
  \[ \varphi_1 \lor \varphi_2 \equiv \varphi_2 \lor \varphi_1 \]
- \( \land \) and \( \lor \) are associative
  \[ \varphi_1 \land (\varphi_2 \land \varphi_3) \equiv (\varphi_1 \land \varphi_2) \land \varphi_3 \]
  \[ \varphi_1 \lor (\varphi_2 \lor \varphi_3) \equiv (\varphi_1 \lor \varphi_2) \lor \varphi_3 \]
- \( \land \) and \( \lor \) are mutually distributive
  \[ \varphi_1 \land (\varphi_2 \lor \varphi_3) \equiv (\varphi_1 \land \varphi_2) \lor (\varphi_1 \land \varphi_3) \]
  \[ \varphi_1 \lor (\varphi_2 \land \varphi_3) \equiv (\varphi_1 \lor \varphi_2) \land (\varphi_1 \lor \varphi_3) \]
- \( \land \) and \( \lor \) are related by \( \neg \) (DeMorgan’s Laws)
  \[ \neg(\varphi_1 \land \varphi_2) \equiv \neg \varphi_1 \lor \neg \varphi_2 \]
  \[ \neg(\varphi_1 \lor \varphi_2) \equiv \neg \varphi_1 \land \neg \varphi_2 \]

Propositional Variables

- Also called Boolean variables, 0-1 variables.
- Every statement can be represented by a propositional variable:
  \[ p_1 = \text{“It is sunny today”} \]
  \[ p_2 = \text{“Tom went to school yesterday”} \]
  \[ p_3 = \text{“} f(x) = 0 \text{ has two solutions”} \]
  \[ p_4 = \text{“point A is on line } BC \text{”} \]
  \[ p_5 = \text{“place a queen at position (1, 2) on a 8 by 8 chessboard”} \]
- ...
Properties of Logical Connectives

\(\land, \Rightarrow, \text{ and } \Leftrightarrow\) are actually redundant:

\[
\begin{align*}
\varphi_1 \land \varphi_2 & \equiv \neg (\neg \varphi_1 \lor \neg \varphi_2) \\
\varphi_1 \Rightarrow \varphi_2 & \equiv \neg \varphi_1 \lor \varphi_2 \\
\varphi_1 \Leftrightarrow \varphi_2 & \equiv (\varphi_1 \Rightarrow \varphi_2) \land (\varphi_2 \Rightarrow \varphi_1)
\end{align*}
\]

We keep them all mainly for convenience.

Exercise Use the truth tables to prove all the logical equivalences seen so far.

Existence of CNF

Every propositional sentence can be transformed into an equivalent CNF.

\[
\begin{align*}
\varphi_1 \Leftrightarrow \varphi_2 & \equiv (\varphi_1 \Rightarrow \varphi_2) \land (\varphi_2 \Rightarrow \varphi_1) \\
\varphi_1 \Rightarrow \varphi_2 & \equiv \neg \varphi_1 \lor \varphi_2 \\
\neg (\varphi_1 \land \varphi_2) & \equiv \neg \varphi_1 \lor \neg \varphi_2 \\
\neg (\varphi_1 \lor \varphi_2) & \equiv \neg \varphi_1 \land \neg \varphi_2
\end{align*}
\]

Conjunctive Normal Form (CNF)

A propositional sentence \(\varphi\) is said to be in Conjunctive Normal Form if \(\varphi\) is True, False, or a conjunction of \(\alpha_i\)'s:

\[
\alpha_1 \land \alpha_2 \land \cdots \land \alpha_n,
\]

where each \(\alpha\) is a clause (a disjunction of \(\beta_j\)'s):

\[
\alpha = (\beta_1 \lor \beta_2 \lor \cdots \lor \beta_m),
\]

where each \(\beta\) is called a literal, which is either a variable or the negation of a variable: \(\beta = p\) or \(\beta = \neg (p)\).

\(-p, p \lor q, p \land (q \lor \neg r)\) are CNF.

\(p \lor (q \land \neg r)\) are not CNF.

Every CNF is an NNF but the opposite does not hold.
Interpretations and Models

- A (partial) interpretation is a mapping from variables to \{0, 1\}.
- An complete interpretation is if it maps every variables to \{0, 1\}.
- An interpretation \(\theta\) can be extended to be a mapping from propositional sentences to \{0, 1\} if it obeys the following rules:
  - \(\theta(\neg p) = 1\) if and only if \(\theta(p) = 0\);
  - if \(\theta(p) = 1\) or \(\theta(q) = 1\) then \(\theta(p \lor q) = 1\);
  - if \(\theta(p) = 1\) and \(\theta(q) = 1\) then \(\theta(p \land q) = 1\); ...

Computing this interpretation is called evaluation.

- For any propositional sentence \(\Phi\), if there exists an interpretation \(\theta\) such that \(\theta(\Phi) = 1\), then \(\theta\) is a model of \(\Phi\) and \(\Phi\) is said to be satisfiable.
- If every complete interpretation is a model for \(\Phi\), then \(\Phi\) is said to be valid.

Example: The 8-Queen Problem

- Variables: \(q_{i,j}\), \(1 \leq i, j \leq 8\). \(q_{i,j}\) is true iff a queen is placed in the square at row \(i\) and column \(j\).
- Clauses:
  - Every row has a queen: For each \(1 \leq i \leq 8\),
    \[q_{i,1} \lor q_{i,2} \lor q_{i,3} \lor q_{i,4} \lor q_{i,5} \lor q_{i,6} \lor q_{i,7} \lor q_{i,8}\]
  - Two queens from the same row cannot see each other:
    \[\neg q_{1,1} \lor \neg q_{1,2}, \ldots\]
  - Two queens from the same column cannot see each other:
    \[\neg q_{1,j} \lor \neg q_{2,j}, \ldots\]
  - Two queens on the same diagonal cannot see each other:
    \[\neg q_{1,1} \lor \neg q_{2,2}, \ldots\]

Inference Systems

- In practice, an inference system \(I\) for PL is a procedure that given a set \(\Gamma = \{\alpha_1, \ldots, \alpha_m\}\) of sentences and a sentence \(\varphi\), may reply "yes", "no", or run forever.
- If \(I\) replies "yes" on input \((\Gamma, \varphi)\), we say that \(\Gamma\) derives \(\varphi\) in \(I\), Or, \(I\) derives \(\varphi\) from \(\Gamma\), or, \(\varphi\) is deduced (derived) from \(\Gamma\) in \(I\). and write
  \[\Gamma \vdash_I \varphi\]
- Intuitively, \(I\) should be such that it replies "yes" on input \((\Gamma, \varphi)\) only if \(\varphi\) is in fact deduced from \(\Gamma\) by \(I\).

Validity vs. Satisfiability

A sentence is

- satisfiable if it is true in some interpretation,
- valid if it is true in every possible interpretation.

Reasoning Tools for Propositional Logic:

- A tool to prove a sentence is valid needs only to return yes or no.
- A tool to prove a sentence is satisfiable often returns an interpretation under which the sentence is true.
- A sentence \(\phi\) is valid if and only if \(\neg \phi\) is unsatisfiable.
All These Fancy Symbols!

In other words,

- \( \Rightarrow \) is a formal symbol of the logic, which is used by the inference system.
- \( \models \) is a shorthand for the entailment of formal sentences that we use to talk about the meaning of formal sentences.
- \( \vdash_I \) is a shorthand for the inference procedure that we use to talk about the output of the inference system \( I \).

The formal symbol \( \Rightarrow \) and the shorthands \( \models, \vdash_I \) are related.

Soundness and Completeness

An inference system \( I \) is **sound** if it derives only sentences that logically follow from a given set of sentences:

\[
\text{if } \Gamma \vdash_I \varphi \text{ then } \Gamma \models \varphi.
\]

An inference system \( I \) is **complete** if it derives all sentences that logically follow from a given set of sentences:

\[
\text{if } \Gamma \models \varphi \text{ then } \Gamma \vdash_I \varphi.
\]

or equivalently,

\[
\text{if } \Gamma \not\vdash_I \varphi \text{ then } \Gamma \not\models \varphi.
\]

All These Fancy Symbols!

The sentence \( \varphi_1 \Rightarrow \varphi_2 \) is valid (always true) if and only if \( \varphi_1 \models \varphi_2 \).

Example: \( A \Rightarrow (A \lor B) \) is valid and \( A \models (A \lor B) \)

<table>
<thead>
<tr>
<th>( A )</th>
<th>( B )</th>
<th>( A \lor B )</th>
<th>( A \Rightarrow (A \lor B) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. False</td>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>2. False</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>3. True</td>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>4. True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
</tbody>
</table>
Inference in Propositional Logic

There are two (equivalent) types of inference systems of Propositional Logic:

- one based on truth tables ($TT$)
- one based on derivation rules ($R$)

Truth Tables. The inference system $TT$ is specified as follows:

\[
\{\alpha_1, \ldots, \alpha_m\} \vdash_{TT} \varphi \quad \text{iff} \quad \text{all the values in the truth table of } \ (\alpha_1 \land \cdots \land \alpha_m) \Rightarrow \varphi \text{ are True}. 
\]

Rule-Based Inference System

Inference Rules

- **And-Introduction**
  \[
  \alpha, \beta \quad \frac{}{\alpha \land \beta}
  \]

- **And-Elimination**
  \[
  \alpha \land \beta \quad \frac{\alpha \land \beta}{\alpha} \quad \frac{\alpha \land \beta}{\beta}
  \]

- **Or-Introduction**
  \[
  \alpha \quad \frac{\alpha}{\alpha \lor \beta} \quad \frac{\alpha}{\beta \lor \alpha}
  \]
Rule-Based Inference Systems

Inference Rules (cont’d.)

- **Double-Negation-Elimination**
  \[ \neg\neg\alpha \Rightarrow \alpha \]

- **False-Introduction**
  \[ \alpha \land \neg\alpha \Rightarrow \text{False} \]

- **False-Elimination**
  \[ \text{False} \Rightarrow \beta \]

Rule-Based Inference System

Inference Rules (cont’)

- **Implication-Elimination (aka Modus Ponens)**
  \[ \alpha \Rightarrow \beta \]

  \[ \alpha \]
  \[ \beta \]

- **Unit Resolution**
  \[ \alpha \lor \beta \]
  \[ \neg\beta \]

  \[ \alpha \]

- **Resolution**
  \[ \alpha \lor \beta \]
  \[ \neg\beta \lor \gamma \]

  \[ \alpha \lor \gamma \]
  or, equivalently,

  \[ \neg\alpha \Rightarrow \beta, \beta \Rightarrow \gamma \]
  \[ \neg\alpha \Rightarrow \gamma \]

Inference by Proof

We can represent a proof more visually as a proof tree:

*Example:*

\[
\begin{align*}
(P \lor H) \land \neg H &\Rightarrow (P \lor H) \land \neg H \\
P \lor H &\Rightarrow \neg H \\
P &
\end{align*}
\]

Inference by Proof

We say there is a proof of \( \varphi \) from \( \Gamma \) in \( \mathcal{R} \) if we can derive \( \varphi \) by applying the rules of \( \mathcal{R} \) repeatedly to \( \Gamma \) and its derived sentences.

*Example: a proof of \( P \) from \( \{ (P \lor H) \land \neg H \} \)*

1. \( (P \lor H) \land \neg H \) by assumption
2. \( P \lor H \) by \( \land \)-elimination applied to (1)
3. \( \neg H \) by \( \land \)-elimination applied to (1)
4. \( P \) by unit resolution applied to (2),(3)
Soundness of Rule-Based Inferences

\[ \mathcal{R} \text{ is sound if all of its rules are sound.} \]

Example: the Resolution rule

\[
\frac{\alpha \lor \beta, \neg \beta \lor \gamma}{\alpha \lor \gamma}
\]

<table>
<thead>
<tr>
<th>(\alpha)</th>
<th>(\beta)</th>
<th>(\gamma)</th>
<th>(\neg \beta)</th>
<th>(\alpha \lor \beta)</th>
<th>(\neg \beta \lor \gamma)</th>
<th>(\alpha \lor \gamma)</th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</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>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
<td>False</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>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
</tbody>
</table>

All the interpretations that make both \(\alpha \lor \beta\) and \(\neg \beta \lor \gamma\) true (ie, 4,5,6,8) make \(\alpha \lor \gamma\) also true.

Rule-Based Inference System

More formally, there is a proof of \(\varphi\) from \(\Gamma\) in \(\mathcal{R}\) if

1. \(\varphi \in \Gamma\) or,
2. there is a rule in \(\mathcal{R}\) that applies to \(\Gamma\) and produces \(\varphi\) or,
3. there is a proof of each \(\varphi_1, \ldots, \varphi_m\) from \(\Gamma\) in \(\mathcal{R}\) and a rule that applies to \(\{\varphi_1, \ldots, \varphi_m\}\) and produces \(\varphi\).

Then, the inference system \(\mathcal{R}\) is specified as follows:

\[ \Gamma \vdash_{\mathcal{R}} \varphi \text{ iff there is a proof of } \varphi \text{ from } \Gamma \text{ in } \mathcal{R} \]

One Rule Suffices!

- Assumptions:
  - \(\beta, \neg \beta\) is a special case of \(\alpha \lor \beta, \neg \beta\) when \(\alpha\) is False.
  - \(\alpha \lor \beta, \neg \beta\) is a special case of \(\alpha \lor \beta, \neg \beta \lor \gamma\) when \(\gamma\) is False.
  - To prove that \(\Gamma \Rightarrow \varphi\) is valid,
    - Convert \(\Gamma \land \neg(\varphi)\) into a set of clauses.
    - Repeatedly apply the Resolution Rule on the clauses.
    - If False is derived out, then \((\Gamma \Rightarrow \varphi)\) is valid.

The Rules of \(\mathcal{R}\)

\[
\begin{array}{cccc}
\alpha & \beta & \alpha \land \beta & \alpha \\
\hline
\alpha \lor \beta & \alpha \lor \gamma & \beta \lor \alpha \\
\hline
\alpha \land \beta & \alpha \land \alpha & \alpha \lor \gamma \\
\hline
\alpha \Rightarrow \beta & \alpha \lor \beta & \neg \beta & \alpha \lor \gamma \\
\hline
\neg \neg \alpha & \alpha \land \neg \alpha & \text{False} & \beta
\end{array}
\]
### Proof by Contradiction

\[ \Gamma \land \neg \varphi \vdash_R \text{False} \]

implies \( (\Gamma \land \neg \varphi) \Rightarrow \text{False} \) is valid.

implies \( \neg (\Gamma \land \neg \varphi) \lor \text{False} \) is valid.

implies \( \neg \Gamma \lor \varphi \) is valid.

implies \( \Gamma \Rightarrow \varphi \) is valid.

### Resolution Proof

**Example:** A proof of \( P \) from \( \{(P \lor H) \land \neg H\} \)

- Convert \( \{(P \lor H) \land \neg H\} \land \neg P \) into a set of clauses (CNF):
  
  \[ \{(1) \ (P \lor H), \ (2) \ \neg H, \ (3) \ \neg P\} \]

- Apply the Resolution Rule:
  1. \( (1) \ (P \lor H) \) and \( (2) \ \neg H \) produces \( (4) \ P \)
  2. \( (3) \ \neg P \) and \( (4) \ P \) produces \text{False}

- Since \text{False} is produced, \( ((P \lor H) \land \neg H) \Rightarrow P \) is valid.

### How to get CNF from \( (p \iff q) \iff r \)

\( (p \iff q) \iff r \equiv ((p \iff q) \Rightarrow r) \land (r \Rightarrow (p \iff q)) \)

\( (p \iff q) \Rightarrow r \equiv \neg (p \iff q) \lor r \equiv (\neg p \iff q) \lor r \equiv ((\neg p \Rightarrow q) \land (q \Rightarrow \neg p)) \lor r \equiv ((p \lor q) \land (\neg q \lor \neg p)) \lor r \equiv (p \lor q \lor r) \land (\neg q \lor \neg p \lor r) \)

\( r \Rightarrow (p \iff q) \equiv \neg r \lor (p \iff q) \equiv \neg r \lor ((p \Rightarrow q) \land (q \Rightarrow p)) \equiv \neg r \lor ((\neg p \lor q) \land (\neg q \lor p)) \equiv (\neg p \lor q \lor \neg r) \land (\neg q \lor \neg p \lor r) \)

### An Example

**Theorem:** If \( (p \iff q) \iff r \) and \( p \iff r \), then \( q \).

- Obtain clauses from the premises and the negation of the conclusion.
  - From \( (p \iff q) \iff r \), we obtain:
    1. \( p \lor q \lor r \), \( 2) \ \neg p \lor \neg q \lor r \), \( 3) \ p \lor \neg q \lor \neg r \), \( 4) \ \neg p \lor q \lor \neg r \).
  - From \( p \iff r \) we obtain:
    5. \( \neg p \lor r \), \( 6) \ p \lor \neg r \).
  - From \( \neg q \), we obtain:
    7. \( \neg q \).

- Apply Resolution to the clauses:
  - From (7), (1): \( 8) \ p \lor r \)
  - From (7), (4): \( 9) \ \neg p \lor \neg r \)
  - From (5), (8): \( 10) \ r \)
  - From (10), (6): \( 11) \ p \)
  - From (10), (9): \( 12) \ \neg p \)
  - From (11), (12): \( 13) \ \Box \)
Proof System for Satisfiability

- The Davis-Putnam-Logemann-Loveland (DPLL) method takes a set of input clauses and converts them into an equivalent set of unit clauses if the set is satisfiable.

- Inference Rules:
  \[
  \begin{align*}
  S \cup \{\alpha \lor \beta, \neg \beta\} & \quad \Rightarrow \quad S \cup \{\alpha, \neg \beta\} \\
  S \cup \{\alpha \lor \beta, \beta\} & \quad \Rightarrow \quad S \cup \{\beta\} \\
  S \quad & \quad \Rightarrow \quad S \cup \{\alpha\} \text{ or } S \cup \{\neg \alpha\}
  \end{align*}
  \]

  : Unit Resolution
  : Unit Subsumption
  : Case Splitting

Some Resolution Strategies

- **Unit resolution**: Unit resolution only.
- **Input resolution**: One of the two clauses must be an input clause.
- **Set of support**: One of the two clauses must be from a designed set called *set of support*. New resolvent are added into the *set of support*.
- **Linear resolution** The latest resolvent must be used in the current resolution.

**Note**: The first 3 strategies above are incomplete.
The Unit resolution strategy is equivalent to the Input resolution strategy: a proof in one strategy can be converted into a proof in the other strategy.