

Artificial Intelligence

Logic and Inferences

**Readings: Chapter 7 of Russell &
Norvig.**

Components of Propositional Logic

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

Propositional Variables

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

Properties of Logical Connectives

- \wedge and \vee are *commutative*

$$\varphi_1 \wedge \varphi_2 \equiv \varphi_2 \wedge \varphi_1$$

$$\varphi_1 \vee \varphi_2 \equiv \varphi_2 \vee \varphi_1$$

- \wedge and \vee are *associative*

$$\varphi_1 \wedge (\varphi_2 \wedge \varphi_3) \equiv (\varphi_1 \wedge \varphi_2) \wedge \varphi_3$$

$$\varphi_1 \vee (\varphi_2 \vee \varphi_3) \equiv (\varphi_1 \vee \varphi_2) \vee \varphi_3$$

- \wedge and \vee are *mutually distributive*

$$\varphi_1 \wedge (\varphi_2 \vee \varphi_3) \equiv (\varphi_1 \wedge \varphi_2) \vee (\varphi_1 \wedge \varphi_3)$$

$$\varphi_1 \vee (\varphi_2 \wedge \varphi_3) \equiv (\varphi_1 \vee \varphi_2) \wedge (\varphi_1 \vee \varphi_3)$$

- \wedge and \vee are related by \neg (DeMorgan's Laws)

$$\neg(\varphi_1 \wedge \varphi_2) \equiv \neg\varphi_1 \vee \neg\varphi_2$$

$$\neg(\varphi_1 \vee \varphi_2) \equiv \neg\varphi_1 \wedge \neg\varphi_2$$

Properties of Logical Connectives

\wedge , \Rightarrow , and \Leftrightarrow are actually redundant:

$$\varphi_1 \wedge \varphi_2 \equiv \neg(\neg\varphi_1 \vee \neg\varphi_2)$$

$$\varphi_1 \Rightarrow \varphi_2 \equiv \neg\varphi_1 \vee \varphi_2$$

$$\varphi_1 \Leftrightarrow \varphi_2 \equiv (\varphi_1 \Rightarrow \varphi_2) \wedge (\varphi_2 \Rightarrow \varphi_1)$$

We keep them all mainly for convenience.

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

Negation Normal Form (NNF)

- A propositional sentence is said to be in *Negation Normal Form* if it contains no connectives other than \vee , \wedge , \neg , and if $\neg(\alpha)$ appears in the sentence, then α must be a propositional variable.
 - $\neg p$, $p \vee q$, $p \vee (q \wedge \neg r)$ are NNF.
 - $p \Leftrightarrow q$, $\neg(p \vee q)$ are not NNF.
- Every propositional sentence can be transformed into an equivalent NNF.

$$\varphi_1 \Leftrightarrow \varphi_2 \quad \equiv \quad (\varphi_1 \Rightarrow \varphi_2) \wedge (\varphi_2 \Rightarrow \varphi_1)$$

$$\varphi_1 \Rightarrow \varphi_2 \quad \equiv \quad \neg\varphi_1 \vee \varphi_2$$

$$\neg(\varphi_1 \wedge \varphi_2) \quad \equiv \quad \neg\varphi_1 \vee \neg\varphi_2$$

$$\neg(\varphi_1 \vee \varphi_2) \quad \equiv \quad \neg\varphi_1 \wedge \neg\varphi_2$$

Conjunctive Normal Form (CNF)

- A propositional sentence ϕ is said to be in *Conjunctive Normal Form* if ϕ is True, False, or a conjunction of α_i 's:

$$\alpha_1 \wedge \alpha_2 \wedge \cdots \wedge \alpha_n,$$

where each α is a *clause* (a disjunction of β_j 's):

$$\alpha = (\beta_1 \vee \beta_2 \vee \cdots \vee \beta_m),$$

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

- $\neg p, p \vee q, p \wedge (q \vee \neg r)$ are CNF.
- $p \vee (q \wedge \neg r)$ are not CNF.
- Every CNF is an NNF but the opposite does not hold.

Existence of CNF

Every propositional sentence can be transformed into an equivalent CNF. That is, from NNF, using

$$(\varphi_1 \wedge \varphi_2) \vee \varphi_3 \equiv (\varphi_1 \vee \varphi_3) \wedge (\varphi_2 \vee \varphi_3)$$

Example: The 8-Queen Problem

● **Variables:** $q_{i,j}$, $1 \leq i, j \leq 8$. q_{ij} is true iff a queen is placed in the square at row i and column j .

● **Clauses:**

● Every row has a queen: For each $1 \leq i \leq 8$,

$$q_{i,1} \vee q_{i,2} \vee q_{i,3} \vee q_{i,4} \vee q_{i,5} \vee q_{i,6} \vee q_{i,7} \vee q_{i,8}$$

● Two queens from the same row cannot see each other:

$$\neg q_{i,1} \vee \neg q_{i,2}, \dots$$

● Two queens from the same column cannot see each other:

$$\neg q_{1,j} \vee \neg q_{2,j}, \dots$$

● Two queens on the same diagonal cannot see each other:

$$\neg q_{1,1} \vee \neg q_{2,2}, \dots$$

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* θ 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 $\theta(p \vee q) = 1$;
 - if $\theta(p) = 1$ and $\theta(q) = 1$ then $\theta(p \wedge q) = 1$; ...

Computing this interpretation is called *evaluation*.

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

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 ϕ is valid if and only if $\neg\phi$ is unsatisfiable.

Inference Systems

- In practice, an inference system \mathcal{I} for PL is a procedure that given a set $\Gamma = \{\alpha_1, \dots, \alpha_m\}$ of sentences and a sentence φ , may reply “yes”, “no”, or run forever.
- If \mathcal{I} replies “yes” on input (Γ, φ) , we say that Γ *derives* φ *in* \mathcal{I} , Or, \mathcal{I} derives φ from Γ , or, φ is deduced (derived) from Γ in \mathcal{I} . and write

$$\Gamma \vdash_{\mathcal{I}} \varphi$$

- Intuitively, \mathcal{I} should be such that it replies “yes” on input (Γ, φ) only if φ is in fact deduced from Γ by \mathcal{I} .

All These Fancy Symbols!

- $A \wedge B \Rightarrow C$

is a sentence (an expression built from variables and logical connectives) where \Rightarrow is a logical connective.

- $A \wedge B \models C$

is a mathematical abbreviation standing for the statement: “every interpretation that makes $A \wedge B$ true, makes C also true.” We say that the sentence $A \wedge B$ *entails* C .

- $A \wedge B \vdash_{\mathcal{I}} C$

is a mathematical abbreviation standing for the statement: “ \mathcal{I} returns yes on input $(A \wedge B, C)$ ” [C is deduced from $A \wedge B$ in \mathcal{I}].

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.
- $\vdash_{\mathcal{I}}$ is a shorthand for the inference procedure that we use to talk about the output of the inference system \mathcal{I} .

The formal symbol \Rightarrow and the shorthands \models , $\vdash_{\mathcal{I}}$ are related.

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: $A \Rightarrow (A \vee B)$ is valid and $A \models (A \vee B)$

	A	B	$A \vee B$	$A \Rightarrow (A \vee B)$
1.	<i>False</i>	<i>False</i>	<i>False</i>	<i>True</i>
2.	<i>False</i>	<i>True</i>	<i>True</i>	<i>True</i>
3.	<u><i>True</i></u>	<i>False</i>	<i>True</i>	<i>True</i>
4.	<u><i>True</i></u>	<i>True</i>	<i>True</i>	<i>True</i>

Soundness and Completeness

- An inference system \mathcal{I} is *sound* if it derives *only* sentences that logically follow from a given set of sentences:

$$\text{if } \Gamma \vdash_{\mathcal{I}} \varphi \text{ then } \Gamma \models \varphi.$$

- An inference system \mathcal{I} is *complete* if it derives *all* sentences that logically follow from a given set of sentences:

$$\text{if } \Gamma \models \varphi \text{ then } \Gamma \vdash_{\mathcal{I}} \varphi.$$

or equivalently,

$$\text{if } \Gamma \not\vdash_{\mathcal{I}} \varphi \text{ then } \Gamma \not\models \varphi.$$

Inference in Propositional Logic

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

- one based on truth tables (\mathcal{TT})
- one based on derivation rules (\mathcal{R})

Truth Tables The inference system \mathcal{TT} is specified as follows:

$\{\alpha_1, \dots, \alpha_m\} \vdash_{\mathcal{TT}} \varphi$ *iff* *all the values in the truth table of*
 $(\alpha_1 \wedge \dots \wedge \alpha_m) \Rightarrow \varphi$ *are True.*

Inference by Truth Tables

- The truth-tables-based inference system is sound:

$\alpha_1, \dots, \alpha_m \vdash_{TT} \varphi$ implies truth table of $(\alpha_1 \wedge \dots \wedge \alpha_m) \Rightarrow \varphi$ all true

implies $(\alpha_1 \wedge \dots \wedge \alpha_m) \Rightarrow \varphi$ is valid

implies $(\alpha_1 \wedge \dots \wedge \alpha_m) \models \varphi$

implies $\alpha_1, \dots, \alpha_m \models \varphi$

- It is also complete (exercise: prove it).
- Its time complexity is $O(2^n)$ where n is the number of propositional variables in $\alpha_1, \dots, \alpha_m, \varphi$.
- We cannot hope to do better because a related, simpler problem (determining the satisfiability of a sentence) is NP-complete.
- However, the really hard cases of propositional inference when we need $O(2^n)$ time are somewhat rare.

Rule-Based Inference System

- An inference system in Propositional Logic can also be specified as a set \mathcal{R} of inference (or derivation) rules.
- Each rule is actually a *pattern* premises/conclusion.
- A rule *applies* to Γ and *derives* φ if
 - some of the sentences in Γ match with the premises of the rule and
 - φ matches with the conclusion.
- A rule is **sound** if the set of its premises entails its conclusion.

Rule-Based Inference System

Inference Rules

● And-Introduction

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

● And-Elimination

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

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

● Or-Introduction

$$\frac{\alpha}{\alpha \vee \beta}$$

$$\frac{\alpha}{\beta \vee \alpha}$$

Rule-Based Inference System

Inference Rules (cont')

- **Implication-Elimination (aka Modus Ponens)**

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

- **Unit Resolution**

$$\frac{\alpha \vee \beta \quad \neg \beta}{\alpha}$$

- **Resolution**

$$\frac{\alpha \vee \beta \quad \neg \beta \vee \gamma}{\alpha \vee \gamma}$$

or, equivalently,

$$\frac{\neg \alpha \Rightarrow \beta, \quad \beta \Rightarrow \gamma}{\neg \alpha \Rightarrow \gamma}$$

Rule-Based Inference Systems

Inference Rules (cont'd.)

● **Double-Negation-Elimination**

$$\frac{\neg\neg\alpha}{\alpha}$$

● **False-Introduction**

$$\frac{\alpha \wedge \neg\alpha}{\text{False}}$$

● **False-Elimination**

$$\frac{\text{False}}{\beta}$$

Inference by Proof

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

Example: a proof of P from $\{(P \vee H) \wedge \neg H\}$

1. $(P \vee H) \wedge \neg H$ by assumption
2. $P \vee H$ by \wedge -elimination applied to (1)
3. $\neg H$ by \wedge -elimination applied to (1)
4. P by unit resolution applied to (2),(3)

Inference by Proof

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

Example:

$$\frac{\frac{(P \vee H) \wedge \neg H}{P \vee H} \quad \frac{(P \vee H) \wedge \neg H}{\neg H}}{P}$$

Rule-Based Inference System

More formally, there is a proof of φ from Γ in \mathcal{R} if

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

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

$\Gamma \vdash_{\mathcal{R}} \varphi$ *iff there is a proof of φ from Γ in \mathcal{R}*

Soundness of Rule-Based Inferences

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

Example: the Resolution rule
$$\frac{\alpha \vee \beta, \quad \neg\beta \vee \gamma}{\alpha \vee \gamma}$$

	α	β	γ	$\neg\beta$	$\alpha \vee \beta$	$\neg\beta \vee \gamma$	$\alpha \vee \gamma$
1.	<i>False</i>	<i>False</i>	<i>False</i>	<i>True</i>	<i>False</i>	<i>True</i>	<i>False</i>
2.	<i>False</i>	<i>False</i>	<i>True</i>	<i>True</i>	<i>False</i>	<i>True</i>	<i>True</i>
3.	<i>False</i>	<i>True</i>	<i>False</i>	<i>False</i>	<i>True</i>	<i>False</i>	<i>False</i>
4.	<i>False</i>	<i>True</i>	<i>True</i>	<i>False</i>	<u><i>True</i></u>	<u><i>True</i></u>	<i>True</i>
5.	<i>True</i>	<i>False</i>	<i>False</i>	<i>True</i>	<u><i>True</i></u>	<u><i>True</i></u>	<i>True</i>
6.	<i>True</i>	<i>False</i>	<i>True</i>	<i>True</i>	<u><i>True</i></u>	<u><i>True</i></u>	<i>True</i>
7.	<i>True</i>	<i>True</i>	<i>False</i>	<i>False</i>	<i>True</i>	<i>False</i>	<i>True</i>
8.	<i>True</i>	<i>True</i>	<i>True</i>	<i>False</i>	<u><i>True</i></u>	<u><i>True</i></u>	<i>True</i>

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

The Rules of \mathcal{R}

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

$$\frac{\alpha}{\alpha \vee \beta}$$

$$\frac{\alpha}{\beta \vee \alpha}$$

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

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

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

$$\frac{\alpha \vee \beta \quad \neg \beta}{\alpha}$$

$$\frac{\alpha \vee \beta \quad \neg \beta \vee \gamma}{\alpha \vee \gamma}$$

$$\frac{\neg \neg \alpha}{\alpha}$$

$$\frac{\alpha \wedge \neg \alpha}{\mathbf{False}}$$

$$\frac{\mathbf{False}}{\beta}$$

One Rule Suffices!

- Assumptions:

- $\frac{\beta \quad \neg\beta}{\text{False}}$ is a special case of $\frac{\alpha \vee \beta \quad \neg\beta}{\alpha}$ when α is *False*.

- $\frac{\alpha \vee \beta \quad \neg\beta}{\alpha}$ is a special case of $\frac{\alpha \vee \beta \quad \neg\beta \vee \gamma}{\alpha \vee \gamma}$ when γ is *False*.

- To prove that $\Gamma \Rightarrow \varphi$ is valid,

- Convert $\Gamma \wedge \neg(\varphi)$ into a set of clauses.

- Repeatedly apply the **Resolution Rule** on the clauses.

- If *False* is derived out, then $(\Gamma \Rightarrow \varphi)$ is valid.

Resolution Proof

Example: A proof of P from $\{(P \vee H) \wedge \neg H\}$

- Convert $\{(P \vee H) \wedge \neg H\} \wedge \neg P$ into a set of clauses (CNF):

$$\{(1) (P \vee H), \quad (2) \neg H, \quad (3) \neg P\}$$

- Apply the Resolution Rule:
 1. (1) $(P \vee H)$ and (2) $\neg H$ produces (4) P
 2. (3) $\neg P$ and (4) P produces *False*
- Since *False* is produced, so $((P \vee H) \wedge \neg H) \Rightarrow P$ is valid.

Proof by Contradiction

$\Gamma \wedge \neg\varphi \vdash_{\mathcal{R}} \textit{False}$

implies $(\Gamma \wedge \neg\varphi) \Rightarrow \textit{False}$ is valid.

implies $\neg(\Gamma \wedge \neg\varphi) \vee \textit{False}$ is valid.

implies $\neg\Gamma \vee \varphi$ is valid.

implies $\Gamma \Rightarrow \varphi$ is valid.

An Example

Theorem: If $(p \Leftrightarrow q) \Leftrightarrow r$ and $p \Leftrightarrow r$, then q .

● Obtain clauses from the premises and the negation of the conclusion.

● From $(p \Leftrightarrow q) \Leftrightarrow r$, we obtain:

(1) $p \vee q \vee r$, (2) $\neg p \vee \neg q \vee r$, (3) $p \vee \neg q \vee \neg r$, (4) $\neg p \vee q \vee \neg r$.

● From $p \Leftrightarrow r$ we obtain:

(5) $\neg p \vee r$, (6) $p \vee \neg r$

● From $\neg q$, we obtain:

(7) $\neg q$.

● Apply Resolution to the clauses:

From (7), (1): (8) $p \vee r$

From (7), (4): (9) $\neg p \vee \neg r$

From (5), (8): (10) r

From (10), (6): (11) p

From (10), (9): (12) $\neg p$

From (11), (12): (13) \square

How to get CNF from $(p \Leftrightarrow q) \Leftrightarrow r$

$$(p \Leftrightarrow q) \Leftrightarrow r \equiv ((p \Leftrightarrow q) \Rightarrow r) \wedge (r \Rightarrow (p \Leftrightarrow q))$$

$$\begin{aligned}(p \Leftrightarrow q) \Rightarrow r &\equiv \neg(p \Leftrightarrow q) \vee r \\ &\equiv (\neg p \Leftrightarrow q) \vee r \\ &\equiv ((\neg p \Rightarrow q) \wedge (q \Rightarrow \neg p)) \vee r \\ &\equiv ((p \vee q) \wedge (\neg q \vee \neg p)) \vee r \\ &\equiv (p \vee q \vee r) \wedge (\neg q \vee \neg p \vee r)\end{aligned}$$

$$\begin{aligned}r \Rightarrow (p \Leftrightarrow q) &\equiv \neg r \vee (p \Leftrightarrow q) \\ &\equiv \neg r \vee ((p \Rightarrow q) \wedge (q \Rightarrow p)) \\ &\equiv \neg r \vee ((\neg p \vee q) \wedge (\neg q \vee p)) \\ &\equiv (\neg p \vee q \vee \neg r) \wedge (\neg q \vee p \vee \neg r)\end{aligned}$$

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.

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:

$$\frac{S \cup \{\alpha \vee \beta, \neg\beta\}}{S \cup \{\alpha, \neg\beta\}}$$

: Unit Resolution

$$\frac{S \cup \{\alpha \vee \beta, \beta\}}{S \cup \{\beta\}}$$

: Unit Subsumption

$$\frac{S}{S \cup \{\alpha\} \quad \text{or} \quad S \cup \{\neg\alpha\}}$$

: Case Splitting