Logic

Chapter 1

Propositional Logic

1.1 The Language of Propositional Logic

Definition 1.1. An **alphabet** for propositional logic is a pair $\mathcal{A} = (\mathcal{V}, \mathcal{C})$, where each component is as follows.

- $\mathcal{V}$ is a countably infinite set of **propositional variables**.
- $\mathcal{C}$ is a finite set of **connectives** with

  $$\mathcal{C} = \bigcup_{i \geq 0} \mathcal{C}_i,$$

  where $\mathcal{C}_i$ is the set of connectives of arity i.

Remark. In the default setting, we usually let

$$\begin{aligned} \mathcal{C}_0 &= \{\bot, \top\} \\ \mathcal{C}_1 &= \{\neg\} \\ \mathcal{C}_2 &= \{\wedge, \vee, \rightarrow, \leftrightarrow\} \end{aligned}$$

and $\mathcal{C}_j = \varnothing$ for $j \geq 3$.

Definition 1.2. The language $\mathcal{L}$ of **formulas** over alphabet $\mathcal{A} = (\mathcal{V}, \mathcal{C})$ is the minimal set that satisfies the following statements.

- Each propositional variable in $\mathcal{V}$ is a formula.
- If $\star$ is a connective in $\mathcal{C}_k$ and $\alpha_1, \alpha_2, \ldots, \alpha_k$ are formulas, then $\star\alpha_1\alpha_2\cdots\alpha_k$ is a formula.

1.2 Truth Assignment

Definition 1.3. A **truth assignment** is a function $\tau : \mathcal{V} \to \{0,1\}$. It can be extended to $\bar{\tau} : \mathcal{L} \to \{0,1\}$ by assigning each connective with arity k to a boolean function from $\{0,1\}^k$ to $\{0,1\}$.

Remark. By convention, we use the truth table as follows.

$\bar{\tau}(\bot)$	$\bar{\tau}(\top)$
0	1

$\bar{\tau}(\alpha)$	$\bar{\tau}(\neg\alpha)$
0	1
1	0

$\bar{\tau}(\alpha)$	$\bar{\tau}(\beta)$	$\bar{\tau}(\alpha \wedge \beta)$	$\bar{\tau}(\alpha \vee \beta)$	$\bar{\tau}(\alpha \to \beta)$	$\bar{\tau}(\alpha \leftrightarrow \beta)$
0	0	0	0	1	1
0	1	0	1	1	0
1	0	0	1	0	0
1	1	1	1	1	1

Table 1.1: Truth Table

Definition 1.4. We say that a truth assignment τ **satisfies** a formula α if $\bar{\tau}(\alpha) = 1$. Also, we say that τ satisfies a set Σ of formulas if it satisfies each formula in Σ.

Definition 1.5. Let Σ be a set of formulas and let α be a formula. We say that Σ **tautologically implies** α, denoted by $\Sigma \models \alpha$, if every truth assignment satisfying Σ also satisfies α.

1.3 Proof System

Definition 1.6. The collection Λ of **axioms** consists of the formulas listed below, where α, β, γ are formulas.

(A1) $\alpha \to (\beta \to \alpha)$.

(A2) $(\alpha \to (\beta \to \gamma)) \to ((\alpha \to \beta) \to (\alpha \to \gamma))$.

(A3) $(\neg\beta \to \neg\alpha) \to (\alpha \to \beta)$.

Definition 1.7. A **proof** of a formula α from a collection Γ of formulas is a sequence of formulas

$$(\alpha_1, \alpha_2, \ldots, \alpha_n)$$

satisfying the following properties.

(a) $\alpha_n = \alpha$.

(b) For $k \in \{1, 2, \ldots, n\}$, either $\alpha_k \in \Lambda \cup \Gamma$ or there exist $i, j \in \{1, 2, \ldots, k-1\}$ with $\alpha_j = \alpha_i \to \alpha_k$.

If there exists a proof of φ from Γ, we write $\Gamma \vdash \varphi$. If $\varnothing \vdash \varphi$, we write $\vdash \varphi$ for short.

Theorem 1.8. For any formula α, we have $\vdash \alpha \to \alpha$.

Proof. We have a proof of $\alpha \to \alpha$ as follows.

1. $(\alpha \to ((\alpha \to \alpha) \to \alpha)) \to ((\alpha \to (\alpha \to \alpha)) \to (\alpha \to \alpha))$. (A2)
2. $\alpha \to ((\alpha \to \alpha) \to \alpha)$. (A1)
3. $(\alpha \to (\alpha \to \alpha)) \to (\alpha \to \alpha)$. (1, 2)
4. $\alpha \to (\alpha \to \alpha)$. (A1)
5. $\alpha \to \alpha$. (3, 4)

Thus, we can conclude that $\vdash \alpha \to \alpha$. □

Proposition 1.9. Let Γ and Δ be sets of formulas and let α be a formula. If $\Gamma \vdash \alpha$ and $\Gamma \subseteq \Delta$, then $\Delta \vdash \alpha$.

Proof. To be completed. □

Theorem 1.10 (Deduction Theorem). Let Γ be a set of formulas and let α and β be formulas. If $\Gamma \cup \{\alpha\} \vdash \beta$, then $\Gamma \vdash \alpha \to \beta$.

Proof. If $\beta \in \Lambda \cup \Gamma$, then we have $\Gamma \vdash \alpha \to \beta_k$ since $\vdash \beta_k \to (\alpha \to \beta_k)$. Furthermore, if $\beta = \alpha$, then we also have $\Gamma \vdash \alpha \to \beta$ since $\vdash \beta \to \beta$ by Theorem 1.8. Thus, one only needs to consider the case that $\beta \notin \Lambda \cup \Gamma \cup \{\alpha\}$.

Suppose that $(\beta_1, \beta_2, \ldots, \beta_n)$ is a proof of β from $\Gamma \cup \{\alpha\}$. For $1 \leq k \leq n$, we prove that $\Gamma \vdash \alpha \to \beta_k$ by induction on k. The induction basis holds for $k = 1$ since $\beta_1 \in \Lambda \cup \Gamma \cup \{\alpha\}$. For the induction step, let $k \geq 2$ and assume that $\Gamma \vdash \alpha \to \beta_\ell$ for $1 \leq \ell < k$. We have proved for the case that $\beta \in \Lambda \cup \Gamma \cup \{\alpha\}$, and thus we assume without loss of generality that there exist $1 \leq i < k$ and $1 \leq j < k$ such that $\beta_j = \beta_i \to \beta_k$. Note that $\Gamma \vdash \alpha \to \beta_i$ and $\Gamma \vdash \alpha \to (\beta_i \to \beta_k)$ hold by induction hypothesis. Therefore, since

$$\vdash (\alpha \to (\beta_i \to \beta_k)) \to ((\alpha \to \beta_i) \to (\alpha \to \beta_k)),$$

we can conclude that $\Gamma \vdash \alpha \to \beta_k$, which completes the proof. $\square$

Theorem 1.11. For any formula α and β, we have $\vdash \neg\alpha \to (\alpha \to \beta)$.

Proof. We have a proof of $\neg\alpha \to (\alpha \to \beta)$ as follows.

1. $((\neg\beta \to \neg\alpha) \to (\alpha \to \beta)) \to (\neg\alpha \to ((\neg\beta \to \neg\alpha) \to (\alpha \to \beta)))$. (A1)
2. $(\neg\beta \to \neg\alpha) \to (\alpha \to \beta)$. (A3)
3. $\neg\alpha \to ((\neg\beta \to \neg\alpha) \to (\alpha \to \beta))$. (1, 2)
4. $(\neg\alpha \to ((\neg\beta \to \neg\alpha) \to (\alpha \to \beta))) \to ((\neg\alpha \to (\neg\beta \to \neg\alpha)) \to (\neg\alpha \to (\alpha \to \beta)))$. (A2)
5. $(\neg\alpha \to (\neg\beta \to \neg\alpha)) \to (\neg\alpha \to (\alpha \to \beta))$. (3, 4)
6. $\neg\alpha \to (\neg\beta \to \neg\alpha)$. (A1)
7. $\neg\alpha \to (\alpha \to \beta)$. (5, 6)

Thus, we can conclude that $\vdash \neg\alpha \to (\alpha \to \beta)$. $\square$