# Software Analysis

## - A Journey from Concretization to Abstraction -

**Ji, Yong-Hyeon**

A document presented for
the Software Analysis

Department of Information Security, Cryptology, and Mathematics
College of Science and Technology
Kookmin University

June 28, 2024

# Contents

# Symbols

In this paper, symbols are defined as follows.

| | |
|---|---|
| $I \models P$ | $I$ satisfies $P$ |
| $I \not\models P$ | $I$ does not satisfies $P$ |

# Chapter 1

# Introduction

## 1.1 intro

# Chapter 2

# Propositional Logic I

## 2.1 Syntax and Semantic

### 2.1.1 Syntax

- **Atom**: basic elements
  - truth symbols $\bot$("false") and $\top$("true")
  - propositional variables $P, Q, R, \ldots$
- **Literal**: an atom $\alpha$ or its negation $\neg\alpha$.
- **Formula**: a literal or the application of a logical connective (boolean connectives) to formulas

$$
\begin{array}{rcll}
F & \rightarrow & \bot & \\
& | & \top & \\
& | & P & \\
& | & \neg F & \text{negation ("not")} \\
& | & F_1 \wedge F_2 & \text{conjunction ("and")} \\
& | & F_1 \vee F_2 & \text{disjunction ("or")} \\
& | & F_1 \rightarrow F_2 & \text{implication ("implies")} \\
& | & F_1 \leftrightarrow F_2 & \text{iff ("if and only if")}
\end{array}
$$

- Formula $G$ is a **subformula** of formula $F$ if it occurs syntactically within $G$.

$$
\begin{aligned}
\mathsf{sub}(\bot) &= \{\bot\} \\
\mathsf{sub}(\top) &= \{\top\} \\
\mathsf{sub}(P) &= \{P\} \\
\mathsf{sub}(\neg F) &= \{\neg F\} \cup \mathsf{sub}(F) \\
\mathsf{sub}(F_1 \wedge F_2) &= \{F_1 \wedge F_2\} \cup \mathsf{sub}(F_1) \cup \mathsf{sub}(F_2) \qquad \vdots
\end{aligned}
$$

- Consider $F : (P \wedge Q) \rightarrow (P \vee \neg Q)$. Then

$$\mathsf{sub}(F) = \{F, P \wedge Q, P \vee \neg Q, P, Q, \neg Q\}.$$

- The strict subformulas of a formula are all its subformulas except itself.

- To minimally use parentheses, we define the relative precedence of the logical connectives from highest to lowest as follows:

$$\neg \quad \wedge \quad \vee \quad \rightarrow \quad \leftrightarrow$$

- (Currying) Additionally, $\rightarrow$ and $\leftrightarrow$ associate to the right, e.g.,

$$P \rightarrow Q \rightarrow R \iff P \rightarrow (Q \rightarrow R).$$

- Example:
  - $(P \wedge Q) \rightarrow (P \vee \neg Q) \iff P \wedge Q \rightarrow P \vee \neg Q$
  - $(P_1 \wedge ((\neg P_2) \wedge \top)) \vee ((\neg P_1) \wedge P_2) \iff P_1 \wedge P_2 \top \vee \neg P_1 \wedge P_2$

### 2.1.2 Semantics

- The semantics of a logic provides its meaning. The meaning of a PL formula is either true or false.
- The semantics of a formula is defined with an **interpretation** (or assignment) that assigns truth values to propositional variables.
- For example, $F : P \wedge Q \rightarrow P \vee \wedge Q$ evaluates to true under the interpretation $I : \{P \mapsto \text{true}, Q \mapsto \text{false}\}$ :

| $P$ | $Q$ | $\neg Q$ | $P \wedge Q$ | $P \vee \neg Q$ | $F$ |
|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1 | 1 |

- The tabular notation is unsuitable for predicate logic. Instead, we define the semantics inductively.

### 2.1.3 Inductive Definition of Semantics

In an inductive definition, the meaning of basic elements is defined first. The meaning of complex elements is defined in terms of subcomponents.

- We write $I \models F$ if $F$ evaluates to **true** under $I$.
- We write $I \not\models F$ if $F$ evaluates to **false** under $I$.

Note that

$$\begin{array}{ll}
I \models \top, \quad I \not\models \bot, & \\
I \models P & \text{iff } I[P] = \text{true} \\
I \models \neg F & \text{iff } I[P] = \text{false} \\
I \models \neg F & \text{iff } I \not\models F \\
I \models F_1 \wedge F_2 & \text{iff } I \models F_1 \text{ and } I \models F_2 \\
I \models F_1 \vee F_2 & \text{iff } I \models F_1 \text{ or } I \models F_2 \\
I \models F_1 \rightarrow F_2 & \text{iff } I \not\models F_1 \text{ or } I \models F_2 \\
I \models F_1 \leftrightarrow F_2 & \text{iff } (I \models F_1 \text{ and } I \models F_2) \text{ or } (I \not\models F_1 \text{ and } I \not\models F_2)
\end{array}$$

**Example 2.1.1.** Consider the formula

$$F : P \land Q \to P \lor \neg Q$$

and the interpretation

$$I : \{P \mapsto \text{true}, Q \mapsto \text{false}\} .$$

The truth value of $F$ is computed as follows:

| | | |
|---|---|---|
| 1. | $I \models P$ | since $I[P] = \text{true}$ |
| 2. | $I \not\models Q$ | since $I[P] = \text{true}$ |
| 3. | $I \models \neg Q$ | by 2 and semantics of $\neg$ |
| 4. | $I \not\models P \land Q$ | by 2 and semantics of $\land$ |
| 5. | $I \models P \lor \neg Q$ | by 1 and semantics of $\lor$ |
| 6. | $I \models F$ | by 4 and semantics of $\to$ |

## 2.2 Satisfiability and Validity

- A formula $F$ is **satisfiable** iff there exists an interpretation $I$ such that $I \models F$.
- A formula $F$ is **valid** iff for all interpretations $I$, $I \models F$.
- Satisfiability and validity are dual:

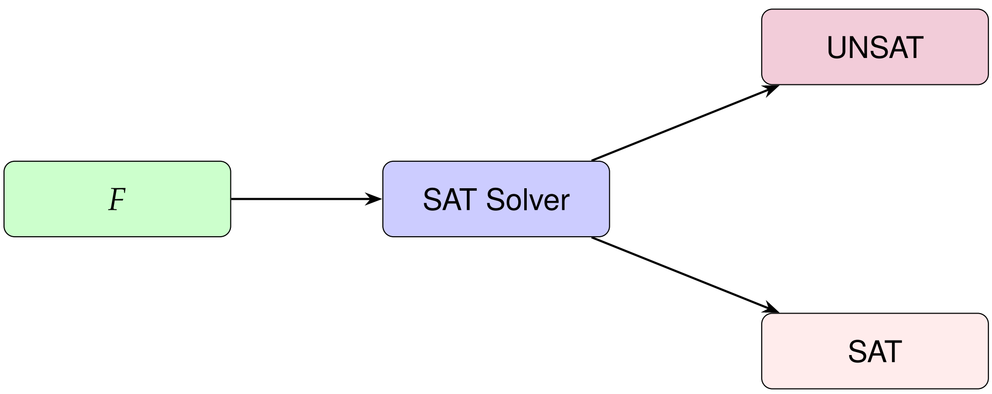
$$F \text{ is valid} \iff \neg F \text{ is unsatisfiable}$$

*Proof.* content... □

- We can check satisfiability by deciding validity, and vice versa.

$$\begin{tikzcd}
& \text{VALID} \arrow[dr, "\neg"] & \\
\text{SAT} \arrow[ur, "\neg"] \arrow[rr, "\neg"] & & \text{UNSAT} \\
& \text{INVALID} \arrow[ul, "\neg"] \arrow[ur, "\neg"'] &
\end{tikzcd}$$

Figure 2.1: Relation of SAT, UNSAT, VALID and INVALID



### 2.2.1 Deciding Validity and Satisfiability

Two approaches to show $F$ is valid:

- **Truth Table Method** performs exhaustive search: e.g.,

$$F : P \land Q \rightarrow P \lor \neg Q.$$

| $P$ | $Q$ | $P \land Q$ | $\neg Q$ | $P \lor \neg Q$ | $F$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 | 1 |

  Non-applicable to logic with infinite domain (e.g., first-order logic).

- **Semantic Argument Method** uses deduction:
  - Assume $F$ is valid: $I \not\models F$ for some $I$ (falsifying interpretation).
  - Apply deduction rules (proof rules) to derive a contradiction.
  - If every branch of the proof derives a contradiction, then $F$ is valid.
  - If some branch of the proof never derives a contradiction, then $F$ is invalid.

### 2.2.2 Deduction Rules for Propositional Logic

**Negation Elimination**

$$\frac{I \models \neg F}{I \not\models F}$$

This rule shows how to derive $F$ from $\neg F$. It eliminates the negation by asserting that if $\neg F$ holds, then $F$ cannot hold.

**Conjunction Elimination**[a]

$$\frac{I \models F \land G}{I \models F, I \models G}$$

This rule breaks down a conjunction into its individual components, showing that if $F \land G$ is true, then both $F$ and $G$ are true.

**Disjunction Elimination**[b]

$$\frac{I \models F \lor G}{I \models F \mid I \models G}$$

This rule asserts that if $F \lor G$ is true in an interpretation, then either $F$ is true, or $G$ is true, or both are true. This is also sometimes referred to as the rule of cases.

**Implication Elimination**[c]

$$\frac{I \models F \rightarrow G}{I \not\models F \mid I \models G}$$

This rule states that if $F \rightarrow G$ is true in an interpretation, then either $F$ is false or $G$ is true. This corresponds to the definition of material implication in classical logic.

**Biconditional Elimination**[d]

$$\frac{I \models F \leftrightarrow G}{I \models F \land G \mid I \models \neg F \land \neg G}$$

This rule states that if $F \leftrightarrow G$ is true in an interpretation, then either both $F$ and $G$ are true, or both $F$ and $G$ are false. This corresponds to the definition of a biconditional statement in classical logic.

[a] And-Elimination
[b] Or-Elimination
[c] Material Implication
[d] If and Only If Elimination

**Negation Introduction**

$$\frac{I \not\models \neg F}{I \models F}$$

This rule introduces $F$ from the fact that $\neg F$ does not hold. It asserts that if the negation of $F$ does not hold, then $F$ must hold.

**De Morgan's Law for Conjunction**[a]

$$\frac{I \not\models F \land G}{I \not\models F \mid I \not\models G}$$

This rule states that if $F \land G$ is not true, then at least one of $F$ or $G$ is not true. This is an application of De Morgan's laws.

**De Morgan's Law for Disjunction**[b]

$$\frac{I \not\models F \lor G}{I \not\models F, I \not\models G}$$

This rule states that if $F \lor G$ is not true, then both $F$ is not true and $G$ is not true. This is directly derived from De Morgan's laws in classical logic.

**Denial of Implication**[c]

$$\frac{I \not\models F \rightarrow G}{I \models F, I \not\models G}$$

This rule states that if $F \rightarrow G$ is not true in an interpretation, then $F$ must be true and $G$ must be false. This is the contrapositive form of material implication.

**Negation of Biconditional**[d]

$$\frac{I \not\models F \leftrightarrow G}{I \models F \land \neg G \mid I \models \neg F \land G}$$

This rule states that if $F \leftrightarrow G$ is not true in an interpretation, then $F$ and $G$ have opposite truth values, i.e., either $F$ is true and $G$ is false, or $F$ is false and $G$ is true. This aligns with the concept of exclusive or (XOR).

[a] Contrapositive of Conjunction Introduction
[b] Contrapositive of Disjunction Introduction
[c] Contrapositive of Implication
[d] Exclusive Or Elimination

**Contradiction Introduction** (Proof by Contradiction)

$$\frac{I \models F \quad I \not\models F}{I \models \bot}$$

**Example 2.2.1.** To prove that the formula

$$F : P \wedge Q \rightarrow P \vee \neg Q$$

is valid, assume that it is invalid and derive a contradiction:

| | | |
|---|---|---|
| 1. | $I \not\models P\wedge \rightarrow P \vee \neg Q$ | assumption |
| 2. | $I \models P \wedge Q$ | by 1 and semantics of $\rightarrow$ |
| 3. | $I \not\models P \vee \neg Q$ | by 1 and semantics of $\rightarrow$ |
| 4. | $I \models P$ | by 2 and semantics of $\wedge$ |
| 5. | $I \neg \models P$ | by 3 and semantics of $\vee$ |
| 6. | $I \models \bot$ | 4 and 5 are contradictory |

**Example 2.2.2.** To prove that the formula

$$F : (P \rightarrow Q) \wedge (Q \rightarrow R) \rightarrow (P \rightarrow R)$$

is valid, assume that it is invalid and derive a contradiction:

### 2.2.3 Proof Tree

### 2.2.4 Derived Rules

# Chapter 3

# Propositional Logic II

## 3.1 Equivalence and Implication, and Equisatisfiability

## 3.2 Substitution

## 3.3 Normal Forms: NNF, DNF, CNF

### 3.3.1 Negation Normal Form (NNF)

### 3.3.2 Disjunctive Normal Form (DNF)

### 3.3.3 Conjunctive Normal Form (CNF)

## 3.4 Decision Procedures for Satisfiability