Formal Methods in Software Engineering

Propositional Logic — Spring 2025

Konstantin Chukharev

§1 Propositional Logic

Motivation

- Boolean functions are at the core of logic-based reasoning.
- A Boolean function $F(X_1, ..., X_n)$ describes the output of a system based on its inputs.
- Boolean gates (AND, OR, NOT) form the building blocks of digital circuits.
- Propositional logic formalizes reasoning about Boolean functions and circuits.
- **Applications**:
  - Digital circuit design.
  - Verification and synthesis of hardware and software.
  - Expressing logical constraints in AI and optimization problems.
  - Automated reasoning and theorem proving.

Boolean Circuits and Propositional Logic

Boolean circuit is a directed acyclic graph (DAG) of Boolean gates.

- Inputs: Propositional variables.
- Outputs: Logical expressions describing the circuit's behavior.

"Can the output of a circuit ever be true?"

- Propositional logic provides a formal framework to answer such questions.

Real-world examples:

- Error detection circuits.
- Arithmetic logic units (ALUs) in processors.
- Routing logic in network devices.

What is Logic?

A formal logic is defined by its **syntax** and **semantics**.

□ Syntax

- An **alphabet** Σ is a set of symbols.
- A finite sequence of symbols (from Σ) is called an **expression** or **string** (over Σ).
- A set of rules defines the **well-formed** expressions.

□ Semantics

- Gives meaning to (well-formed) expressions.

Syntax of Propositional Logic

□ Alphabet

1. Logical connectives: $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$.
2. Propositional variables: $A_1, A_2, \ldots, A_n$.
3. Parentheses for grouping: (,).

□ Well-Formed Formulas (WFFs)

1. A single propositional symbol (e.g. A) is a WFF.
2. If α and β are WFFs, so are: $\neg\alpha,\ (\alpha \wedge \beta),\ (\alpha \vee \beta),\ (\alpha \rightarrow \beta),\ (\alpha \leftrightarrow \beta)$.
3. No other expressions are WFFs.

□ Conventions

- Large variety of propositional variables: $A, B, C, \ldots, p, q, r, \ldots$.
- Outer parentheses can be omitted: $A \wedge B$ instead of $(A \wedge B)$.
- Operator precedence: $\neg > \wedge > \vee > \rightarrow > \leftrightarrow$.

Semantics of Propositional Logic

- Each propositional variable is assigned a truth value: T (true) or F (false).
- More formally, *interpretation* $\nu : V \to \{0, 1\}$ assigns truth values to all variables (atoms).
- Truth values of complex formulas are computed (evaluated) recursively:
  1. $[\![p]\!] \triangleq \nu(p)$, where $p \in V$ is a propositional variable
  2. $[\![\neg\alpha]\!] \triangleq 1 - [\![\alpha]\!]$
  3. $[\![\alpha \land \beta]\!] \triangleq \min([\![\alpha]\!], [\![\beta]\!])$
  4. $[\![\alpha \lor \beta]\!] \triangleq \max([\![\alpha]\!], [\![\beta]\!])$
  5. $[\![\alpha \to \beta]\!] \triangleq [\![\alpha]\!] \leq [\![\beta]\!]$
  6. $[\![\alpha \leftrightarrow \beta]\!] \triangleq [\![\alpha]\!] = [\![\beta]\!]$

Truth Tables

α	β	γ	$\alpha \land (\beta \lor \neg\gamma)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

§2 Semantical Aspects

Validity, Satisfiability, Entailment

□ Validity

- α is a **tautology** if α is true under all truth assignments.
  Formally, α is **valid**, denoted "$\models \alpha$", iff $\nu(\alpha) = 1$ for all interpretations $\nu \in \{0,1\}^V$.
- α is a **contradiction** if α is false under all truth assignments.
  Formally, α is **unsatisfiable** if $\nu(\alpha) = 0$ for all interpretations $\nu \in \{0,1\}^V$.

□ Satisfiability

- α is **satisfiable** (**consistent**) if there exists an interpretation $\nu \in \{0,1\}^V$ where $\nu(\alpha) = 1$.
  When α is satisfiable by ν, denoted $\nu \models \alpha$, this interpretation is called a **model** of α.
- α is **falsifiable** (**invalid**) if there exists an interpretation $\nu \in \{0,1\}^V$ where $\nu(\alpha) = 0$.

□ Entailment

- Let Γ be a set of WFFs. Then Γ **tautologically implies** (**semantically entails**) α, denoted $\Gamma \models \alpha$, if every truth assignment that satisfies all formulas in Γ also satisfies α.
- Formally, $\Gamma \models \alpha$ iff for all interpretations $\nu \in \{0,1\}^V$ and formulas $\beta \in \Gamma$, if $\nu \models \beta$, then $\nu \models \alpha$.
- Note: $\alpha \models \beta$, where α and β are WFFs, is just a shorthand for $\{\alpha\} \models \beta$.

Examples

- $A \vee B \wedge (\neg A \wedge \neg B)$ is satisfiable, but not valid.
- $A \vee B \wedge (\neg A \wedge \neg B) \wedge (A \leftrightarrow B)$ is unsatisfiable.
- $\{A \to B, A\} \vDash B$
- $\{A, \neg A\} \vDash A \wedge \neg A$
- $\neg(A \wedge B)$ is tautologically equivalent to $\neg A \vee \neg B$.

Duality of SAT vs VALID

- **SAT**: Given a formula α, determine if it is satisfiable.
$$\exists \nu . [\![\alpha]\!]$$
- **VALID**: Given a formula α, determine if it is valid.
$$\forall \nu . [\![\alpha]\!]$$
- **Duality**: α is valid iff $\neg \alpha$ is unsatisfiable.
- Note: SAT is NP, but VALID is co-NP.

Solving SAT using Truth Tables

Algorithm for satisfiability:

To check whether α is satisfiable, construct a truth table for α. If there is a row where α evaluates to true, then α is satisfiable. Otherwise, α is unsatisfiable.

Algorithm for semantical entailment (tautological implication):

The check whether $\{\alpha_1, \ldots, \alpha_k\} \vDash \beta$, check the satisfiability of $(\alpha_1 \land \ldots \land \alpha_k) \land (\neg\beta)$. If it is unsatisfiable, then $\{\alpha_1, \ldots, \alpha_k\} \vDash \beta$. Otherwise, $\{\alpha_1, \ldots, \alpha_k\} \nvDash \beta$.

Logical Laws and Tautologies

- **Associative** and **Commutative** laws for $\wedge$, $\vee$, $\leftrightarrow$:
  - $A \circ (B \circ C) \equiv (A \circ B) \circ C$
  - $A \circ B \equiv B \circ A$
- **Distributive laws**:
  - $A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$
  - $A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$
- **Negation**:
  - $\neg\neg A \equiv A$
- **De Morgan's laws**:
  - $\neg(A \wedge B) \equiv \neg A \vee \neg B$
  - $\neg(A \vee B) \equiv \neg A \wedge \neg B$

Logical Laws and Tautologies [2]

- **Implication**:
  - $(A \to B) \equiv (\neg A \lor B)$
- **Contraposition**:
  - $(A \to B) \equiv (\neg B \to \neg A)$
- **Law of Excluded Middle**:
  - $(A \lor \neg A) \equiv \top$
- **Contradiction**:
  - $(A \land \neg A) \equiv \bot$
- **Exportation**:
  - $((A \land B) \to C) \equiv (A \to (B \to C))$

Example Problems

- Given $\alpha = (A \lor B) \land \neg A$, determine:
  1. Is α consistent (satisfiable)?
  2. Is α valid (a tautology)?
  3. Compute the truth table for α.

Completeness of Connectives

- All Boolean functions can be expressed using $\{\neg, \land, \lor\}$ (so called *"standard Boolean basis"*).
- Even smaller sets are sufficient:
  - $\{\neg, \land\}$ — AIG (And-Inverter Graph), see also: AIGER format.
  - $\{\neg, \lor\}$
  - $\{\overline{\land}\}$ — NAND
  - $\{\overline{\lor}\}$ — NOR

Incompleteness of Connectives

To prove that a set of connectives is incomplete, we find a property that is true for all WFFs expressed using those connectives, but that is not true for some Boolean function.

Example: $\{\land, \to\}$ is not complete.

Proof. Let α be a WFF which uses only these connectives. Let ν be an interpretation such that $\nu(A_i) = 1$ for all propositional variables A_i. Next, we prove by induction that $[\![\alpha]\!] = 1$.

- Base case:
  - $[\![A_i]\!] = \nu(A_i) = 1$
- Inductive step:
  - $[\![\beta \land \gamma]\!] = \min([\![\beta]\!], [\![\gamma]\!]) = 1$
  - $[\![\beta \to \gamma]\!] = \max(1 - [\![\beta]\!], [\![\gamma]\!]) = 1$

Thus, $[\![\alpha]\!] = 1$ for all WFFs α built from $\{\land, \to\}$. However, $[\![\neg A_1]\!] = 0$, so there is no such formula α tautologically equivalent to $\neg A_1$.

Compactness

Recall:

- A WFF α is **satisfiable** if there exists an interpretation ν such that $\nu \vDash \alpha$.
- Hereinafter, let Γ denote a *finite* set of WFFs, and Σ denote a *possibly infinite* set of WFFs.
- A set of WFFs Σ is **satisfiable** if there exists an interpretation ν that satisfies all formulas in Σ.
- A set of WFFs Σ is **finitely satisfiable** if every finite subset of Σ is satisfiable.

Theorem 1: Compactness Theorem.

A set of WFFs Σ is satisfiable iff it is finitely satisfiable.

Proof.

(=>) Suppose Σ is satisfiable, i.e. there exists an interpretation ν that satisfies all formulas in Σ.

This direction is trivial: any subset of a satisfiable set is clearly satisfiable.

- For each finite subset $\Sigma' \subseteq \Sigma$, ν also satisfies all formulas in Σ'.
- Thus, every finite subset of Σ is satisfiable.

Compactness [2]

(<=) Suppose Σ is finitely satisfiable, i.e. every finite subset of Σ is satisfiable.

Construct a *maximal* finitely satisfiable set Δ as follows:

- Let $\alpha_1, ..., \alpha_n, ...$ be a fixed enumeration of all WFFs.
  - *This is possible since the set of all sequences of a countable set is countable.*
- Then, let:

$$\Delta_0 = \Sigma,$$

$$\Delta_{n+1} = \begin{cases} \Delta_n \cup \{\alpha_{n+1}\} & \text{if this is finitely satisfiable,} \\ \Delta_n \cup \{\neg\alpha_{n+1}\} & \text{otherwise.} \end{cases}$$

  - *Note that each Δ_n is finitely satisfiable by construction.*
- Let $\Delta = \bigcup_{n \in \mathbb{N}} \Delta_n$. Note:
  1. $\Sigma \subseteq \Delta$
  2. $\alpha \in \Delta$ or $\neg\alpha \in \Delta$ for any WFF α
  3. Δ is finitely satisfiable by construction.

Compactness [3]

Now we need to show that Δ is satisfiable (and thus $\Sigma \subseteq \Delta$ is also satisfiable).

Define an interpretation ν as follows: for each propositional variable p, let $\nu(p) = 1$ iff $p \in \Delta$.

We claim that ν satisfies a WFF α iff $\alpha \in \Delta$. The proof is by induction on well-formed formulas.

- Base case:
  - Suppose $\alpha \equiv p$ for some propositional variable p.
  - By definition, $[\![p]\!] = \nu(p) = 1$.
- Inductive step:
  - *(Note: here, we consider only one case for brevity)*
  - Suppose $\alpha \equiv \beta \land \gamma$.
  - $[\![\alpha]\!] = 1$ iff both $[\![\beta]\!] = 1$ and $[\![\gamma]\!] = 1$ **iff** both $\beta \in \Delta$ and $\gamma \in \Delta$.
  - Now, if both β and γ are in Δ, then since $\{\beta, \gamma, \neg\alpha\}$ is not satisfiable, we must have $\alpha \in \Delta$.
  - Similarly, if either β or γ is not in Δ, then its negation must be in Δ, and thus $\alpha \notin \Delta$.

Compactness [4]

Corollary 1.

If $\Sigma \vDash \alpha$, then there is a finite $\Sigma_0 \subseteq \Sigma$ such that $\Sigma_0 \vDash \alpha$.

Proof.

Suppose that $\Sigma_0 \nvDash \alpha$ for every finite $\Sigma_0 \subseteq \Sigma$.

Then, $\Sigma_0 \cup \{\neg\alpha\}$ is satisfiable for every finite $\Sigma_0 \subseteq \Sigma$.

Then, by the compactness theorem, $\Sigma \cup \{\neg\alpha\}$ is satisfiable, which contradicts the assumption that $\Sigma \vDash \alpha$.

Normal Forms

- **Conjunctive Normal Form (CNF)**:
  - A formula is in CNF if it is a conjunction of *clauses* (disjunctions of literals).
  - Example: $(A \lor B) \land (\neg A \lor C) \land (B \lor \neg C)$ — CNF with 3 clauses.
- **Disjunctive Normal Form (DNF)**:
  - A formula is in DNF if it is a disjunction of *cubes* (conjunctions of literals).
  - Example: $(\neg A \land B) \lor (B \land C) \lor (\neg A \land B \land \neg C)$ — DNF with 3 cubes.
- **Algebraic Normal Form (ANF)**:
  - A formula is in ANF if it is a sum of *products* of variables (or a constant 1).
  - Example: $B \oplus AB \oplus ABC$ — ANF with 3 terms.

§3 Proof Systems

Natural Deduction

- **Natural deduction** is a proof system for propositional logic.
- **Axioms**:
  - **No axioms**.
- **Rules**:
  - **Introduction**: $\land$-introduction, $\lor$-introduction, $\to$-introduction, $\neg$-introduction.
  - **Elimination**: $\land$-elimination, $\lor$-elimination, $\to$-elimination, $\neg$-elimination.
  - **Reduction ad Absurdum**
  - **Law of Excluded Middle** (note: forbidden in *intuitionistic* logic)
- **Proofs** are constructed by applying rules to assumptions and previously derived formulas.

$$\frac{\Gamma_1 \vdash (\textit{premise 1}) \qquad \Gamma_2 \vdash (\textit{premise 2}) \qquad \dots}{\Gamma\ (\textit{assumptions}) \vdash (\textit{conclusion})}\ \text{rule name}$$

Inference Rules

$$\frac{}{\Gamma \vdash \varphi \vee \neg\varphi}\ \text{law of excluded middle}$$

$$\frac{}{\Gamma, \varphi \vdash \varphi}\ \text{assumption}$$

$$\frac{\Gamma \vdash \alpha \qquad \Gamma \vdash \neg\alpha}{\Gamma \vdash \beta}\ \text{reduction ad absurdum}$$

$$\frac{\Gamma \vdash \alpha \wedge \beta}{\Gamma \vdash \alpha}\ \wedge\text{-elimination}$$

$$\frac{\Gamma \vdash \alpha \wedge \beta}{\Gamma \vdash \beta}\ \wedge\text{-elimination}$$

$$\frac{\Gamma \vdash \alpha \qquad \Gamma \vdash \beta}{\Gamma \vdash \alpha \wedge \beta}\ \wedge\text{-introduction}$$

$$\frac{\Gamma \vdash \alpha_1 \vee \alpha_2 \qquad \Gamma, \alpha_1 \vdash \beta \qquad \Gamma, \alpha_2 \vdash \beta}{\Gamma \vdash \beta}\ \vee\text{-elim}$$

$$\frac{\Gamma \vdash \alpha}{\Gamma \vdash \alpha \vee \beta}\ \vee\text{-intro}$$

$$\frac{\Gamma \vdash \beta}{\Gamma \vdash \alpha \vee \beta}\ \vee\text{-intro}$$

$$\frac{\Gamma \vdash \alpha \qquad \Gamma \vdash \alpha \to \beta}{\Gamma \vdash \beta}\ \to\text{-elimination}$$

$$\frac{\Gamma, \alpha \vdash \beta}{\Gamma \vdash \alpha \to \beta}\ \to\text{-introduction}$$

Soundness and Completeness

- A formal system is **sound** if every provable formula is true in all models.
  - **Weak soundness**: "every provable formula is a tautology".

    If $\vdash \alpha$, then $\vDash \alpha$.

  - **Strong soundness**: "every derivable (from Γ) formula is a logical consequence (of Γ)".

    If $\Gamma \vdash \alpha$, then $\Gamma \vDash \alpha$.

- A formal system is **complete** if every formula true in all models is provable.
  - **Weak completeness**: "every tautology is provable".

    If $\vDash \alpha$, then $\vdash \alpha$.

  - **Strong completeness**: "every logical consequence (of Γ) is derivable (from Γ)".

    If $\Gamma \vDash \alpha$, then $\Gamma \vdash \alpha$.

§4 Theory of Computation

Computability

Definition 1: Church–Turing thesis.

Computable functions are exactly the functions that can be calculated using a mechanical (that is, automatic) calculation device given unlimited amounts of time and storage space.

"Every model of computation that has ever been imagined can compute *only* computable functions, and *all* computable functions can be computed by any of several *models of computation* that are apparently very different, such as Turing machines, register machines, lambda calculus and general recursive functions."

For example, a partial function $f : \mathbb{N}^k \hookrightarrow \mathbb{N}$ is computable ("can be calculated") if there exists a computer program with the following properties:

- If $f(x)$ is defined, then the program terminates on the input x with the value $f(x)$ stored in the computer memory.
- If $f(x)$ is undefined, then the program never terminates on the input x.

Definition 2.

An **effective procedure** is a finite, deterministic, mechanical algorithm that guarantees to terminate and produce the correct answer in a finite number of steps.

Decidability

Definition 3: Decidable set.

Given a universal set $\mathcal{U}$, a set $S \subseteq \mathcal{U}$ is **decidable** (or **computable**) if there exists a computable function $f : \mathcal{U} \longrightarrow \{0, 1\}$ such that $f(x) = 1$ iff $x \in S$.

Examples:

- The set W of all WFFs is decidable.
  - *We can check if a given string is well-formed by recursively verifying the syntax rules.*
- For a given finite set Σ of WFFs, the set $\{\alpha \mid \Sigma \vDash \alpha\}$ of all tautological consequences of Σ is decidable.
  - *We can decide $\Sigma \vDash \alpha$ using a truth table algorithm by enumerating all possible interpretations (at most $2^{|\Sigma|}$) and check if each satisfies all formulas in Σ.*
- The set of all tautologies is decidable.
  - *It is the set of all tautological consequences of the empty set.*

TODO: undecidable sets (existence proof)

Semi-decidability

Suppose we want to determine $\Gamma \vDash \alpha$ where Γ is infinite. In general, it is undecidable.

However, it is possible to obtain a weaker result.

Definition 4: Semi-decidable set.

A set S is **computably enumerable** if there is an *enumeration procedure* which lists, in some order, every member of S: $s_1, s_2, s_3 \ldots$

Equivalently, a set S is **semi-decidable** if there is an algorithm such that the set of inputs for which the algorithm halts is exactly S.

Note that if S is infinite, the enumeration procedure will *never* finish, but every member of S will be listed *eventually*, after some finite amount of time.

Some properties:

- Decidable sets are closed under union, intersection, Cartesian product, and complement.
- Semi-decidable sets are closed under union, intersection, and Cartesian product.

Semi-decidability [2]

Theorem 2.

A set S is computably enumerable iff it is semi-decidable.

(here, we assume that S is a set of WFFs)

(=> *proof of "only if" part)*

If S is computably enumerable, we can check if $\alpha \in S$ by enumerating all members of S and checking if α is among them. If it is, we answer "yes"; otherwise, we continue enumerating. Thus, if $\alpha \in S$, the procedure produces "yes". If $\alpha \notin S$, the procedure runs forever.

(<= *proof of "if" part)*

On the other hand, suppose we have a procedure P which, given α, terminates and produces "yes" iff $\alpha \in S$. To show that S is computably enumerable, we can proceed as follows.

1. Construct a systematic enumeration of **all** expressions (for example, by listing all strings over the alphabet in length-lexicographical order): $\beta_1, \beta_2, \beta_3, \ldots$
2. Break the procedure P into a finite number of "steps" (for example, by program instructions).
3. Run the procedure on each expression in turn, for an increasing number of steps (see dovetailing):

Semi-decidability [3]

- Run P on β_1 for 1 step.
- Run P on β_1 for 2 steps, then on β_2 for 2 steps.
- …
- Run P on each of $\beta_1, \ldots, \beta_n$ for n steps each.
- …

4. If P produces "yes" for some β_i, output (yield) β_i and continue enumerating.

This procedure will eventually list all members of S.

Semi-decidability [4]

Theorem 3.

A set is decidable iff both it and its complement are semi-decidable.

Proof. Alternate between running the procedure for the set and the procedure for its completement. One of them will eventually produce “yes”.

Semi-decidability [5]

Theorem 4.

If Σ is an effectively enumerable set of WFFs, then the set of tautological consequences of Σ is effectively enumerable.

Proof. Consider an enumeration of the elements of Σ: $\sigma_1, \sigma_2, \sigma_3, \ldots$

By the compactness theorem, $\Sigma \vDash \alpha$ iff $\{\sigma_1, \ldots, \sigma_n\} \vDash \alpha$ for some n.

Hence, it is sufficient to successively test:

- $\emptyset \vDash \alpha$
- $\{\sigma_1\} \vDash \alpha$
- $\{\sigma_1, \sigma_2\} \vDash \alpha$
- …

If any of these tests succeeds (each is decidable), then $\Sigma \vDash \alpha$.

Complexity

TODO

TODO

- [x] Natural deduction
- [ ] Soundnsess and completeness of propositional logic
- [x] Compactness
- [x] Computability
- [x] Decidability
- [ ] Undecidable sets
- [x] Semi-decidability
- [ ] Complexity
- [ ] Normal forms
- [ ] Canonical normal forms
- [ ] Equisatisfiability, Tseitin transformation, Example
- [ ] DIMACS format
- [ ] SAT
- [ ] Cook theorem

Summary

- Propositional logic provides a foundation for reasoning about Boolean functions.
- Key concepts: Syntax, semantics, WFFs, truth tables, and logical laws.
- Next steps: SAT solvers and their role in automated reasoning.