# Mathematical Logic and Proofs

## 1 Mathematical Logic

Mathematical logic is a precise mathematical language used in much of mathematics. Its purpose is to eliminate ambiguities that arise in communication in natural languages. Its applicability is limited in scope but where it can be applied, it brings in greater precision.

1. Propositional Logic
2. Predicate Logic

## 2 Propositional Logic

> A **proposition** is a statement that is either true or false. It cannot be **neither** nor **both**.

While the truth or falsity of certain propositions can be debated outside the framework of logic, the field of logic only enables conclusions once the valuations of propositions is known.

**Truth tables**: A tabular way to list standard (and non-standard) boolean functions.

**Syntax**: It is the structure by which formulae are constrructed.

**Semantics:** It is the interpretation of a formula under different truth values of its propositions. In simple terms, the truth table.

Two formulae in propositional logic are said to be **semantically equivalent** if they both evaluate to the same truth values under each possible assignment.

**Standard connectives/ operators in boolean logic:** And ($\wedge$), Or ($\vee$), Not ($\neg$), implication ($\Rightarrow$), equivalence ($\Leftrightarrow$), exclusive-or ($\oplus$).

**axioms. inference rules, theorems, proofs**. Building proofs, contradictions and tautologies. Fallicious proofs.

**Converse, contrapositive and inverse**.
Applications of propositional logic:

1. Queries
2. Puzzles
3. System design
4. Hardware and algorithm design/implementation

Standard equivalences, non-standard formulae and the number of formulae over a fixed number of propositions. Standard forms like conjunctive normal form (CNF) and disjunctive normal form (DNF). De Morgan's Laws.

## 3 Predicate Logic

Here we have the standard existential ($\exists$) and universal ($\forall$) quantifiers and several variants. There is an analogue to De Morgan's Laws here also. Nested quantifiers and applications in the context of strategy games.

## 4 Proof techniques

1. Direct proofs.
2. Proof by contraposition.
3. Proof by contradiction
4. Exhaustive proofs for finite settings
5. Proof by cases using finite quotienting
6. Without loss of generality: eliminating equivalent cases or cases considered trivial or already handled
7. Proof by construction
8. Non-constructive existence proof
9. Uniqueness proofs
10. The concept of counter-examples.

# 5 syntax

**Syntax** in the context of mathematical logic is rules used to describe all legal ways to construct a formula in the logic.

The number of formulae is unlimited, but the rules are a limited finite set. Thus they are powerful rules that can be applied to generate all possible formulae. The set of rules for a given logic is not unique. Some are not minimal and can be reduced. Greater variety of operators make it easier to use, but smaller number of operators make circuit building and proving thorems about the logic, much easier.

A set of rules is deemed adequate to construct all formulae in the logic, if every distinct semantic formula can be represented syntactically in at least one way using the set of rules.

**Inductive rules construction**:

1. The constant boolean values $T$ and $F$ (often represented by the numerals 1 and 0 respectively).
2. The atomic propositions $p_1, p_2, \ldots$
3. Formulae constructed using one of the following operations
   (a) If $\phi_1$ is a formula, then so is $\neg\phi_1$
   (b) If $\phi_1$ and $\phi_2$ are formulae, then so is $\phi_1 \vee \phi_2$
   (c) If $\phi_1$ and $\phi_2$ are formulae, then so is $\phi_1 \wedge \phi_2$
   (d) If $\phi_1$ and $\phi_2$ are formulae, then so is $\phi_1 \Rightarrow \phi_2$
   (e) If $\phi_1$ and $\phi_2$ are formulae, then so is $\phi_1 \Leftrightarrow \phi_2$

One could use more (like NAND, NOR, XOR etc.) or fewer operations to write all posible formulae in propositional logic. A minimal set of operators could be ($\vee$,$\neg$) or others.

This process of building up formulae using a few basic cases together with rules to form the non-basic cases gives rise to a technique of proof of results that is very similar to the well-known mathematical induction: it is called **structural induction**- or induction based on the structure of a formula. There are other settings where structural induction is present, in discrete mathematics.

Semantics is basically given by truth tables.

An **assignment** for a propositional logic formula is giving specific truth values (either true or false) for each atomic proposition occurring in the formula.

Table 1: NOT

| $\neg$ | $p_1$ | $\neg p_1$ |
|---|---|---|
| $Ass_1$ | $T$ | $F$ |
| $Ass_2$ | $F$ | $T$ |

Table 2: OR

| $\vee$ | $p_1$ | $p_2$ | $p_1 \vee p_2$ |
|---|---|---|---|
| $Ass_1$ | $T$ | $T$ | $T$ |
| $Ass_2$ | $T$ | $F$ | $T$ |
| $Ass_3$ | $F$ | $T$ | $T$ |
| $Ass_4$ | $F$ | $F$ | $F$ |

A formula is said to be **satisfiable** if it has at least one assignment under which it evaluates to true.

A formula is said to be a **tautology** or a **validity** if it evaliuates to true under every assignment.

A formula is said to be a **contradiction** if it evaluates to false under every possible assignment.

The number of assignments for a boolean formula with $k$ propositional variables is $2^k$. The number of semantically distinct propositional logic formula involving $k$ variables is $2^{2^k}$.

Converting any truth table to a formula involving only $\wedge$, $\vee$ and $\neg$:

1. Only consider asignment rows which evaluate to $T$ (or 1)
2. For each such row create an $\vee$ of all the propositional variables that are set to true, as they are and all those variables set to false in the negation form.
3. Take the $\wedge$ of the expressions obtained in step 2.

Table 3: AND

| $\wedge$ | $p_1$ | $p_2$ | $p_1 \wedge p_2$ |
|---|---|---|---|
| $Ass_1$ | $T$ | $T$ | $T$ |
| $Ass_2$ | $T$ | $F$ | $F$ |
| $Ass_3$ | $F$ | $T$ | $F$ |
| $Ass_4$ | $F$ | $F$ | $F$ |

Table 4: IMPLIES

| $\Rightarrow$ | $p_1$ | $p_2$ | $p_1 \Rightarrow p_2$ |
|---|---|---|---|
| $Ass_1$ | $T$ | $T$ | $T$ |
| $Ass_2$ | $T$ | $F$ | $F$ |
| $Ass_3$ | $F$ | $T$ | $T$ |
| $Ass_4$ | $F$ | $F$ | $T$ |

Table 5: EQUIVALENT

| $\Leftrightarrow$ | $p_1$ | $p_2$ | $p_1 \Leftrightarrow p_2$ |
|---|---|---|---|
| $Ass_1$ | $T$ | $T$ | $T$ |
| $Ass_2$ | $T$ | $F$ | $F$ |
| $Ass_3$ | $F$ | $T$ | $F$ |
| $Ass_4$ | $F$ | $F$ | $T$ |

A formula obtained in this way is said to be in **conjunctive normal form** or CNF. Similarly it is possible to obtain an identical formula in **disjunctive normal form** or DNF by focussing only on rows which evaluate to 0 and modifying the steps in the above procedure, accordingly.

A list of well known logical equivalence statements (by name):

- Identity laws
- Domination laws
- Idempotent laws
- Double negation laws
- Commutative laws
- Associative laws
- Distributive laws
- De Morgan's laws
- Negation laws
- Laws for implication
- Laws for equivalence