Propositional Logic

Dr Paolo Guagliardo

THE UNIVERSITY of EDINBURGH
informatics

Fall 2020 (v20.1.0)

Logic in general

Logics are formal languages for

- representing **what we know** about the world
- **reasoning** about this knowledge (draw conclusions from it)

Two components:

Syntax defines the sentences in the language

Semantics defines the **meaning** of the sentences

Used in many areas of Computer Science:

- Artificial Intelligence
- Semantic Web
- Software & Hardware verification
- **Databases**
- ... many many others

Many families of logics

There is not **one** logic, but many families of logics:

- Modal logics (epistemic, temporal, spatial, ...)
- Description logics
- Non-monotonic logics
- Intuitionistic logic
- ... many many others

We will study two **classical** logics:

- propositional logic
- first-order logic

Propositional logic: Building blocks

Propositions

Atomic statements that cannot be further decomposed

For example:

- "It is raining"
- "The cat is on the table"
- "The sky is blue"

Usually denoted with uppercase letters: P, Q, ...

Also called **propositional variables**

Logical connectives

- Conjunction: $\wedge$ (**and**)
- Negation: $\neg$ (**not**)

Propositional logic: Syntax

Let **Prop** be a countable set of propositional variables

The language $\mathcal{L}$ of propositional logic over **Prop**
is defined inductively as follows:

1. Every $P \in$ **Prop** is in $\mathcal{L}$ (**atomic formulae** or **atoms**)
2. If ϕ and ψ are in $\mathcal{L}$, then $\phi \wedge \psi$ is also in $\mathcal{L}$
3. If ϕ is in $\mathcal{L}$, then also $\neg\phi$ is in $\mathcal{L}$
4. Nothing else is in $\mathcal{L}$

In other words, $\mathcal{L}$ is generated by the following grammar:

$$\phi := P \mid \phi \wedge \phi \mid \neg\phi \qquad \text{(with } P \in \textbf{Prop}\text{)}$$

Propositional logic: Semantics

Intuition

- Atomic statements can be either **true** ($\mathbf{t}$) or **false** ($\mathbf{f}$)
- The truth value of a formula
  is determined by the truth values of its atoms

Formally

A **truth-value assignment** is a function $\alpha\colon \textbf{Prop} \to \{\mathbf{t}, \mathbf{f}\}$

Then, α **satisfies** a formula ϕ (written $\alpha \models \phi$)
inductively as follows:

$$\begin{array}{lcl} \alpha \models P & \iff & \alpha(P) = \mathbf{t} \\ \alpha \models \neg\phi & \iff & \alpha \not\models \phi \\ \alpha \models \phi \wedge \psi & \iff & \alpha \models \phi \text{ and } \alpha \models \psi \end{array}$$

Truth tables

Reflect the semantics of the connectives

$\wedge$	t	f
t	t	f
f	f	f

	$\neg$
t	f
f	t

Given α and ϕ, allow us to determine whether $\alpha \models \phi$:

1. Replace each propositional variable P in ϕ by $\alpha(P)$
2. Propagate **t** and **f** according to the truth tables

Example

Given the formula

$$\phi = \neg(\neg A \wedge B) \wedge \neg(C \wedge \neg D)$$

and the assignment

$$\alpha = \{ \quad A \mapsto \mathbf{t}\,, \quad B \mapsto \mathbf{f}\,, \quad C \mapsto \mathbf{f}\,, \quad D \mapsto \mathbf{t} \quad \}$$

does α satisfy ϕ?

BLACKBOARD-TIME!

Given α and ϕ, checking whether $\alpha \models \phi$
can be done in **polynomial time** in the size of ϕ

Satisfiability and Validity

A formula ϕ is

- **satisfiable** if there is some α that satisfies ϕ
- **unsatisfiable** if ϕ is not satisfiable
- **falsifiable** if there is some α that does not satisfy ϕ
- **valid** if every α satisfies ϕ
  (in this case ϕ is called a **tautology**)

Consequences

- ϕ is a tautology iff $\neg\phi$ is unsatisfiable
- ϕ is unsatisfiable iff $\neg\phi$ is a tautology

Equivalence

Two formulas are **logically equivalent** (written $\phi \equiv \psi$) if for all α

$$\alpha \models \phi \quad \Longleftrightarrow \quad \alpha \models \phi$$

(i.e., there is no assignment that satisfies one formula but not the other)

Intuition: Equivalent formulae have the **same meaning**
even though they may **differ syntactically**

(they say the same thing in different ways)

Propositional logic: Extended language

Syntax

New connectives:

- Disjunction $\vee$ (**or**)
- Implication $\rightarrow$ (**if-then**)
- Double implication $\leftrightarrow$ (**if and only if**)

New grammar:

$$\phi := P \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \phi \rightarrow \phi \mid \phi \leftrightarrow \phi$$

Semantics

$\alpha \models \phi \vee \psi$	iff	$\alpha \models \phi$ or $\alpha \models \psi$
$\alpha \models \phi \rightarrow \psi$	iff	if $\alpha \models \phi$, then $\alpha \models \psi$
$\alpha \models \phi \leftrightarrow \psi$	iff	$(\alpha \models \phi$ if and only if $\alpha \models \psi)$

Expressive power

Every formula in the extended language
can be **equivalently expressed** using only $\wedge$ and $\neg$

- $\phi \vee \psi \equiv \neg(\neg\phi \wedge \neg\psi)$
- $\phi \rightarrow \psi \equiv \neg\phi \vee \psi$
- $\phi \leftrightarrow \psi \equiv (\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi)$

The new connectives do not add expressive power to the language

- they are just **syntactic sugar**
- but useful to write formulae more succinctly

Truth tables of the new connectives

Disjuction

$\vee$	t	f
t	t	t
f	t	f

Implication

$\to$	t	f
t	t	f
f	t	t

Double Implication

$\leftrightarrow$	t	f
t	t	f
f	f	t

All of the above can be derived from the truth tables for $\wedge$ and $\neg$

Equivalences

Commutativity

$$\phi \vee \psi \equiv \psi \vee \phi$$
$$\phi \wedge \psi \equiv \psi \wedge \phi$$

Associativity

$$(\phi \vee \psi) \vee \chi \equiv \phi \vee (\psi \vee \chi)$$
$$(\phi \wedge \psi) \wedge \chi \equiv \phi \wedge (\psi \wedge \chi)$$

Distributivity

$$\phi \wedge (\psi \vee \chi) \equiv (\phi \wedge \psi) \vee (\phi \wedge \chi)$$
$$\phi \vee (\psi \wedge \chi) \equiv (\phi \vee \psi) \wedge (\phi \vee \chi)$$

Idempotence

$$\phi \vee \phi \equiv \phi$$
$$\phi \wedge \phi \equiv \phi$$

Absorption

$$\phi \vee (\phi \wedge \psi) \equiv \phi$$
$$\phi \wedge (\phi \vee \psi) \equiv \phi$$

Equivalences (continued)

Double Negation	$\neg\neg\phi \equiv \phi$
De Morgan	$\neg(\phi \vee \psi) \equiv \neg\phi \wedge \neg\psi$
	$\neg(\phi \wedge \psi) \equiv \neg\phi \vee \neg\psi$
Implication	$\phi \rightarrow \psi \equiv \neg\phi \vee \psi$

Entailment

We extend the satisfaction relationship $\models$ to **sets Σ of formulae**:

$$\alpha \models \Sigma \quad \Longleftrightarrow \quad \alpha \models \phi \text{ for all } \phi \in \Sigma \quad \Longleftrightarrow \quad \alpha \models \bigwedge_{\phi \in \Sigma} \phi$$

Then, we say that Σ **entails** a formula ϕ if for all α

$$\alpha \models \phi \quad \text{whenever} \quad \alpha \models \Sigma$$

(i.e., every assignment that satisfies Σ also satisfies ϕ)

Properties of Entailment

Deduction Theorem

$$\Sigma \cup \{\phi\} \models \psi \quad \text{iff} \quad \Sigma \models \phi \rightarrow \psi$$

Contraposition Theorem

$$\Sigma \cup \{\phi\} \models \neg\psi \quad \text{iff} \quad \Sigma \cup \{\psi\} \models \neg\phi$$

Contradiction Theorem

$$\Sigma \cup \{\phi\} \text{ is unsatisfiable} \quad \text{iff} \quad \Sigma \models \neg\phi$$

Decision problems

A **decision problem**

- takes (one or more) inputs
- answers a Yes / No question

Example

SAT: Given a propositional formula ϕ, is ϕ satisfiable?
(i.e., can we find an assignment α such that $\alpha \models \phi$?)

A **decision procedure** is an algorithm that

- always terminates
- solves a decision problem

A decision problem is **decidable** if there is decision procedure for it

Solving SAT

Satisfiability in propositional logic is a **decidable** problem

Naive algorithm

1. Enumerate all possible assignments
   (there are 2^n where n is the number of atoms in the formula)
2. For each assignment check whether the formula is satisfied:
   2.1 if it is, stop and answer YES
   2.2 otherwise, continue to next assignment
3. If there are no more assignment, stop and answer NO

SAT is an **NP-complete** problem:

- a (candidate) solution can be verified in polynomial time
- no known efficient (polynomial) way to locate a solution
  (in the worst case, it requires **exponential** time)

Reduction to satisfiability

Validity, equivalence, and entailment
can all be reduced to checking satisfiability:

- ϕ is valid iff $\neg\phi$ is not satisfiable
- $\phi \models \psi$ iff $\phi \rightarrow \psi$ is valid (deduction theorem)
- $\phi \equiv \psi$ iff $\phi \leftrightarrow \psi$ is valid

A decision procedure for satisfiability is all we need

Validity, equivalence and entailment are decidable problems
(in particular, coNP-complete) in propositional logic

Some additional terminology

Atom	atomic formula
Literal	atom (**positive literal**) or its negation (**negative literal**)
Clause	disjunction of literals
Term	conjunction of literals

Normal forms

Formulae expressed in a standard syntactic form

Conjunctive Normal Form (CNF)

Conjunction of clauses: $\bigwedge_{i=1}^{n}\left(\bigvee_{j=1}^{m} L_{i,j}\right)$

Example: $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$

Disjunctive Normal Form (DNF)

Disjunction of terms: $\bigvee_{i=1}^{n}\left(\bigwedge_{j=1}^{m} L_{i,j}\right)$

Example: $(A \wedge B) \vee (A \wedge \neg C) \vee (A \wedge \neg D) \vee (\neg B \wedge \neg C) \vee (\neg B \wedge \neg D)$

Negation Normal Form (NNF)

Only $\wedge, \vee, \neg$ and negation can appear only in front of atoms
(in particular, every formula in CNF or DNF is also in NNF)

Example: $\big(A \wedge (B \vee \neg C \vee \neg D)\big) \vee \big(\neg B \wedge (B \vee \neg C \vee \neg D)\big)$

Converting to NNF, CNF and DNF

For every formula
there exist **equivalent** formulae in CNF, DNF and NNF

To convert into NNF

1. Replace $\{\rightarrow, \leftrightarrow\}$ by $\{\wedge, \vee, \neg\}$
2. Apply De Morgan's laws

To convert into CNF or DNF

1. Convert into NNF
2. Apply the distributivity laws

Why normal forms?

Some approaches to inference use syntactic operations on formulae, often expressed in standardized forms

CNF tells us something as to whether a formula is **valid**:

- If all clauses contain complementary literals, then the formula is a tautology
- Otherwise, the formula is falsifiable

DNF tells us something as to whether a formula is **satisfiable**:

- If all terms contain complementary literals, then the formula is unsatisfiable
- Otherwise, the formula is satisfiable

DNF and Satisfiability

Whether a formula in DNF is satisfiable
can be decided in **linear time** in the size of the formula

Wait! Didn't we say that SAT is NP-complete? **Yes**

So what's the catch with DNF?

The transformation into DNF is **expensive** (in time/space)

- The size of the formula may blow-up exponentially!