PROPOSITIONAL LOGIC

- Proposition - A declarative statement that is either true or false (but not both).

- Symbols in propositional logic:

  Proposition symbols **TRUE, FALSE**, p, q, r, ...
  Connectives $\neg, \wedge, \vee, \rightarrow, \longleftrightarrow$.

- Atom - a proposition symbol
  Literal - an atom p or its negation ¬p. An atom p is a positive literal and ¬p is a negative literal.

- **Definition. (Well-Formed Formulas (WFF) = sentences).**

  The well-formed formulas (or formulas for short), are defined inductively as follows:

  (1) An atom is a formula.
  (2) If G is a formula, then ¬G is a formula.
  (3) If G and H are formulas, then $(G \wedge H)$, $(G \vee H)$, $(G \rightarrow H)$ and $(G \longleftrightarrow H)$ are formulas.
  (4) All formulas are generated by applying the above rules.

- A propositional theory Δ - a finite set of propositional formulas.

- Herbrand Base of Δ - the (finite) set of propositions (atoms) occurring in Δ, denoted as HB(Δ).

- Truth value of a formula ϕ in terms of the truth values of atoms occurring in ϕ.

  Let p and q be two propositions. The truth values of the formulas $\neg p$, $p \wedge q$, $p \vee q$, $p \rightarrow q$ and $p \longleftrightarrow q$ in terms of the truth values of p and q are given by the following table:

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \rightarrow q$	$p \longleftrightarrow q$
T	**T**	**F**	**T**	**T**	**T**	**T**
T	**F**	**F**	**F**	**T**	**F**	**F**
F	**T**	**T**	**F**	**T**	**T**	**F**
F	**F**	**T**	**F**	**F**	**T**	**T**

- Interpretation - an assignment which assigns either **T** or **F** to each atom in HB(Δ). Equivalently, an interpretation I for a propositional theory Δ is a subset of HB(Δ) such that atoms in I are assigned **T** and those not in I are assigned **F**.

- Model of Δ - an interpretation M is a model of Δ if for each formula $\phi \in \Delta$, the truth value of ϕ under M is **T**. If the truth value of ϕ under I is **T**, then we say ϕ is *satisfied* by I. Otherwise, we say ϕ is *falsified* by I.

- **Example 1. (propositional theory, interpretation and model).**
  Consider the set of formulas $\Delta = \{p \wedge q, r \vee s, \neg a \vee b\}$. Clearly Δ is a propositional theory. Consider the following interpretations $I_1 = \{p, r, b\}$, $I_2 = \{p, q\}$ and $I_3 = \{p, q, s\}$. We can verify that I_1,

I_2 are not models of Δ and I_3 is a model from the following truth table:

Inter.	a	b	p	q	r	s	$p \wedge q$	$r \vee s$	$\neg a \vee b$
I_1	**F**	**T**	**T**	**F**	**T**	**F**	**F**	**T**	**T**
I_2	**F**	**F**	**T**	**T**	**F**	**F**	**T**	**F**	**T**
I_3	**F**	**F**	**T**	**T**	**F**	**T**	**T**	**T**	**T**

- Valid formula - A formula ϕ is *valid* if it is true under all interpretations.
  Unsatisfiable formula - A formula ϕ is *unsatisfiable* if it is false under all interpretations, i.e., it has no models.
  Satisfiable formula - A formula ϕ is satisfiable if and only if ϕ has a model, i.e., if and only if is NOT unsatisfiable.

- Equivalent formulas - Two formulas ϕ and ψ are equivalent if they have the same models. In other words, ϕ and ψ are equivalent if they have the same truth value under every interpretation for ϕ and ψ.

For example, the formulas $p \rightarrow q$ and $\neg p \vee q$ are equivalent. The formulas $p \rightarrow q$ and $\neg q \rightarrow \neg p$ are also equivalent.

Laws (Equivalent formulas) which can be used to perform formula transformation.

(1)	$\phi \longleftrightarrow \psi = (\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi)$		
(2)	$\phi \rightarrow \psi = \neg\phi \vee \psi$		
(3a)	$\phi \vee \psi = \psi \vee \phi$	(3b)	$\phi \wedge \psi = \psi \wedge \phi$
(4a)	$\phi \vee (\psi \vee \gamma) = (\phi \vee \psi) \vee \gamma$	(4b)	$\phi \wedge (\psi \wedge \gamma) = (\phi \wedge \psi) \wedge \gamma$
(5a)	$\phi \vee (\psi \wedge \gamma) = (\phi \vee \psi) \wedge (\phi \vee \gamma)$	(5b)	$\phi \wedge (\psi \vee \gamma) = (\phi \wedge \psi) \vee (\phi \wedge \gamma)$
(6a)	$\phi \vee \text{false} = \phi$	(6b)	$\phi \wedge \text{true} = \phi$
(7a)	$\phi \vee \text{true} = \text{true}$	(7b)	$\phi \wedge \text{false} = \text{false}$
(8a)	$\phi \vee \neg\phi = \text{true}$	(8b)	$\phi \wedge \neg\phi = \text{false}$
(9)	$\neg(\neg\phi) = \phi$		
(10a)	$\neg(\phi \vee \psi) = \neg\phi \wedge \neg\psi$	(10b)	$\neg(\phi \wedge \psi) = \neg\phi \vee \neg\psi$

- Clause - a disjunction of literals of the form $L_1 \vee L_2 \vee ... \vee L_m$.

  Theorem. Each formula ϕ can be equivalently transformed to a formula ϕ' such that ϕ' is of the form $C_1 \wedge C_2 \wedge ... \wedge C_n$ where each C_j is a clause.

  Such a form ϕ' is called a *conjunctive normal form* of ϕ.

Conjunctive-Normal-Form Algorithm (outline).

Input:

A formula ϕ.

Output:

A formula $\phi' = \phi$ such that ϕ' is in conjunctive normal form.

(1) Use laws $\phi \longleftrightarrow \psi = (\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi)$ and $\phi \rightarrow \psi = \neg\phi \vee \psi$ to eliminate connectives "$\longleftrightarrow$" and "$\rightarrow$".

(2) Repeatedly apply the law $\neg(\neg\phi) = \phi$ to bring the negation sign "$\neg$" immediately before atom.

(3) Repeatedly apply distributive law $\phi \vee (\psi \wedge \gamma) = (\phi \vee \psi) \wedge (\phi \vee \gamma)$ and other laws to obtain a conjunctive normal form.

For example, the formula $\phi = (p \longleftrightarrow q) \vee \neg(r \vee s)$ can be transformed to the formula $\phi' = (\neg p \vee q \vee \neg r) \wedge (\neg p \vee q \vee \neg s) \wedge (p \vee \neg q \vee \neg r) \wedge (p \vee \neg q \vee \neg s)$.

LOGICAL ENTAILMENT (also called LOGICAL CONSEQUENCE)

Definition (Logical Entailment). Let $\Delta = \{\phi_1, \phi_2, ..., \phi_n\}$ be a set of formulas and ϕ be a formula. We say $\phi_1 \wedge \phi_2 \wedge ... \wedge \phi_n$ logically entails ϕ, if and only if any model of $\phi_1 \wedge \phi_2 \wedge ... \wedge \phi_n$ is a model of ϕ. When $\phi_1 \wedge \phi_2 \wedge ... \wedge \phi_n$ logically entails ϕ, we also say ϕ is a logical consequence of $\phi_1, \phi_2, ...,$ and ϕ_n (or ϕ logically follows from $\phi_1, \phi_2, ..., \phi_n$).

Example. Consider formulas $\{p \vee q \vee r, p \vee \neg r\}$. The formula $p \vee q$ is a logical consequence of $p \vee q \vee r$ and $p \vee \neg r$.

Theorem. A formula ϕ is a logical consequence of formulas $\phi_1, \phi_2, ..., \phi_n$ if and only if the formula $((\phi_1 \wedge \phi_2 \wedge ... \wedge \phi_n) \rightarrow \phi)$ is valid.

Theorem. A formula ϕ is a logical consequence of formulas $\phi_1, \phi_2, ..., \phi_n$ if and only if the formula $\phi_1 \wedge \phi_2 \wedge ... \wedge \phi_n \wedge \neg\phi$ is unsatisfiable.

The above two theorems are very important because they tell us that the problem of showing ϕ being a logical consequence of a set of formulas can be reduced to the problem of showing a related formula to be unsatisfiable. The latter problem can be solved efficiently using *resolution* which we will describe shortly.

THE RESOLUTION PRINCIPLE

We assume from now on that each propositional formula ϕ is represented in conjunctive normal form and thus we can equivalently represent ϕ as $\{C_1, C_2, ..., C_n\}$ where each C_j is a clause and $\phi = C_1 \wedge C_2 \wedge ... \wedge C_n$.

- Complementary literals - an atom p and its negation ¬p are called complementary literals.

Definition.(resolvent). Let C_1 and C_2 be two clauses such that $C_1 = C'_1 \vee p$ and $C_2 = C'_2 \vee \neg p$. The clause $C = C'_1 \vee C'_2$ is called the *resolvent* of C_1 and C_2, denoted as $C = \text{res}(C_1, C_2)$. Here the atom p is called **the resolving literal**.

For example, let $C_1 = a \vee \neg b \vee d$ and $C_2 = q \vee \neg r \vee \neg d$. Then we have $C = \text{res}(C_1, C_2) = a \vee \neg b \vee q \vee \neg r$.

Theorem. Let $C = \text{res}(C_1, C_2)$ be the resolvent of clauses C_1 and C_2. Then C is a logical consequence of C_1 and C_2.

Definition. (resolution derivation). Let S be a set of clauses. A *resolution derivation* of a clause C from S is a sequence $\sigma = (C_1, C_2, ... C_k)$ of clauses such that

(1) Each C_l, either $C_l \in S$ or $C_l = \text{res}(C_i, C_j)$ for $i, j < l$.

(2) $C_k = C$.

A resolution derivation of the empty clause □ from S is called a *refutation*.

Theorem. If a clause C has a resolution derivation from a set S of clauses, then C is a logical consequence of S.

Theorem. (Soundness of the resolution principle). Let S be a set of clauses. If there is a resolution derivation of the empty clause □ from S, then S is unsatisfiable.

Theorem. (Completeness of the resolution principle). Let S be a set of clauses. If S is unsatisfiable, then there is a resolution derivation of the empty clause □ from S.

From the above theorems and the theorems about logical consequence, we can easily see the equivalence of the following statements: (assume $S = \{C_1, C_2, ..., C_n\}$ is a set of clauses and G is a formula)

1. G is a logical consequence of S;
2. the formula $(C_1 \wedge C_2 \wedge ... \wedge C_n \wedge \neg G)$ is unsatisfiable;
3. the set of clauses $S \cup \{C_{n+1}, C_{n+2}, ..., C_{n+k}\}$ is unsatisfiable, where $C_{n+1} \wedge C_{n+2} \wedge ... \wedge C_{n+k} = \neg G$.
4. there is a resolution derivation of the empty clause □ from $S \cup \{C_{n+1}, C_{n+2}, ..., C_{n+k}\}$.

LOGICAL CONSEQUENCE ALGORITHM

Input:

A set S of clauses and a goal formula G.

Output:

a yes/no answer according to whether G is a logical consequence of S or not.

(1) Negate the goal G to get ¬G. Then transform ¬G to a set of clauses S′.

(2) If there is a resolution derivation of the empty clause □ from S ∪ S′, then answer "yes" and terminate. Otherwise answer "no" and terminate.

Example. Let S = {p ∨ q, ¬p ∨ ¬q, ¬p ∨ r, ¬q ∨ s, p ∨ ¬w, q ∨ u} and let G = (r ∨ s) ∧ (u ∨ ¬w). We want to show that G is a logical consequence of S.

We first transform ¬G into clausal form: ¬G = ¬[(r ∨ s) ∧ (u ∨ ¬w)] = (¬(r ∨ s) ∨ ¬(u ∨ ¬w)) = ((¬r ∧ ¬s) ∨ (¬u ∧ w)) = (¬r ∨ ¬u) ∧ (¬r ∨ w) ∧ (¬s ∨ ¬u) ∧ (¬s ∨ w). Thus S′ = {¬r ∨ ¬u, ¬r ∨ w, ¬s ∨ ¬u, ¬s ∨ w}.

We then search for a resolution derivation of the empty clause □ from S ∪ S′. One such derivation is given below.

¬r ∨ w   ¬q ∨ s   p∨ q   ¬p ∨ r   ¬r ∨ ¬u   ¬s ∨ ¬u   ¬p ∨ ¬q   p ∨ ¬w

q ∨ r   ¬q ∨ ¬w   q ∨ u

¬s ∨ w   r ∨ s   ¬w ∨ u

s ∨ w   s ∨ ¬u

w   ¬u

¬w

□

Exercises.

1. Let Δ = {(p ∨ ¬r) → q, (a ↔ b) → c}. Convert Δ into an equivalent set of clauses.

2. Let S = {p ∨ q, p ∨ ¬q, ¬p∨ q, ¬p∨ ¬q}. Indicate whether S is consistent or not. Support your conclusion by 2 ways: (i). Indicate whether S has a model; (ii). Indicate whether there is a resolution

derivation of the empty clause □ from S.

3. Let $S = \{a \vee \neg b \vee c,\ d \vee b,\ \neg a \vee d\}$. Show that the clause $c \vee d$ is a logical consequence of S by resolution.

References

1. Chang & Lee, Symbolic Logic and Mechanical Theorem Proving, Academic Press, 1973.