Propositional and Predicate Logic - II

Petr Gregor

KTIML MFF UK

ZS 2014/2015

Language

Propositional logic is a *"logic of propositional connectives"*. We start from a (nonempty) set $\mathbb{P}$ of *propositional letters* (*variables*), e.g.

$$\mathbb{P} = \{p, p_1, p_2, \dots, q, q_1, q_2, \dots\}$$

We usually assume that $\mathbb{P}$ is at most countable.

The *language* of propositional logic (over $\mathbb{P}$) consists of symbols

- propositional letters from $\mathbb{P}$
- propositional connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$
- parentheses (,) , [,] , { , } , ...

Thus the language is given by the set $\mathbb{P}$. We say that connectives and parentheses are *symbols of logic*.

We also use symbols for constants $\top$ (true), $\bot$ (false) which are introduced as shortcuts for $p \vee \neg p$, resp. $p \wedge \neg p$ where p is any fixed variable from $\mathbb{P}$.

Formula

Propositional formulae (*propositions*) (over $\mathbb{P}$) are given inductively by

(i) every propositional letter from $\mathbb{P}$ is a proposition,

(ii) if φ, ψ are propositions, then also

$$(\neg\varphi)\ ,(\varphi \wedge \psi)\ ,(\varphi \vee \psi)\ ,(\varphi \rightarrow \psi)\ ,(\varphi \leftrightarrow \psi)$$

are propositions,

(iii) every proposition is formed by a finite number of steps (i), (ii).

- Thus propositions are (well-formed) finite sequences of symbols from the given language (strings).
- A proposition that is a part of another proposition φ as a substring is called a *subformula* (*subproposition*) of φ.
- The set of all propositions over $\mathbb{P}$ is denoted by $\mathrm{VF}_{\mathbb{P}}$.
- The set of all letters (variables) that occur in φ is denoted by $\mathrm{var}(\varphi)$.

Conventions

After introducing (standard) *priorities* for connectives we are allowed in a concise form to omit parentheses that are around a subformula formed by a connective of a higher priority.

(1) $\rightarrow$, $\leftrightarrow$

(2) $\wedge$, $\vee$

(3) $\neg$

The outer parentheses can be omitted as well, e.g.

$$(((\neg p) \wedge q) \rightarrow (\neg(p \vee (\neg q)))) \quad \text{is shortly} \quad \neg p \wedge q \rightarrow \neg(p \vee \neg q)$$

Note If we do not respect the priorities, we can obtain an ambiguous form or even a concise form of a non-equivalent proposition.

Further possibilities to omit parentheses follow from semantical properties of connectives (associativity of $\vee$, $\wedge$).

Formation trees

A *formation tree* is a finite ordered tree whose nodes are labeled with propositions according to the following rules

- leaves (and only leaves) are labeled with propositional letters,
- if a node has label $(\neg\varphi)$, then it has a single son labeled with φ,
- if a node has label $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \rightarrow \psi)$, or $(\varphi \leftrightarrow \psi)$, then it has two sons, the left son labeled with φ, and the right son labeled with ψ.

A *formation tree of a proposition* φ is a formation tree with the root labeled with φ.

Proposition *Every proposition is associated with a unique formation tree.*

Proof By induction on the number of nested parentheses. □

Note Such proofs are called proofs by the structure of the formula or by the depth of the formation tree.

Semantics

- We consider only two-valued logic.
- Propositional letters represent (atomic) statements whose 'meaning' is given by an assignment of *truth values* 0 *(false)* or 1 *(false)*.
- Semantics of propositional connectives is given by their *truth tables*.

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \rightarrow q$	$p \leftrightarrow q$
0	0	1	0	0	1	1
0	1	1	0	1	1	0
1	0	0	0	1	0	0
1	1	0	1	1	1	1

  This determines the truth value of every proposition based on the values assigned to its propositional letters.
- Thus we may assign *"truth tables"* also to all propositions. We say that propositions represent Boolean functions (up to the order of variables).
- A *Boolean function* is an n-ary operation on $2 = \{0, 1\}$.

Truth valuations

- A *truth assignment* is a function $v\colon \mathbb{P} \to \{0,1\}$, i.e. $v \in {}^{\mathbb{P}}2$.
- A *truth value* $\overline{v}(\varphi)$ of a proposition φ for a truth assignment v is given by

$$
\begin{array}{rclcrcl}
\overline{v}(p) &=& v(p) \ \text{ if } \ p \in \mathbb{P} & \qquad & \overline{v}(\neg\varphi) &=& -_1(\overline{v}(\varphi)) \\
\overline{v}(\varphi \wedge \psi) &=& \wedge_1(\overline{v}(\varphi), \overline{v}(\psi)) & & \overline{v}(\varphi \vee \psi) &=& \vee_1(\overline{v}(\varphi), \overline{v}(\psi)) \\
\overline{v}(\varphi \to \psi) &=& \to_1(\overline{v}(\varphi), \overline{v}(\psi)) & & \overline{v}(\varphi \leftrightarrow \psi) &=& \leftrightarrow_1(\overline{v}(\varphi), \overline{v}(\psi))
\end{array}
$$

where $-_1$, $\wedge_1$, $\vee_1$, $\to_1$, $\leftrightarrow_1$ are the Boolean functions given by the tables.

Proposition *The truth value of a proposition φ depends only on the truth assignment of $\mathrm{var}(\varphi)$.*

Proof Easily by induction on the structure of the formula. $\square$

Note Since the function $\overline{v}\colon \mathrm{VF}_{\mathbb{P}} \to 2$ is a unique extension of the function v, we can (unambiguously) write v instead of $\overline{v}$.

Semantic notions

A proposition φ over $\mathbb{P}$ is

- *is true in* (*satisfied by*) *an assignment* $v \in {}^{\mathbb{P}}2$, if $\overline{v}(\varphi) = 1$. Then v is a *satisfying assignment* for φ, denoted by $v \models \varphi$.
- *valid* (*a tautology*), if $\overline{v}(\varphi) = 1$ for every $v \in {}^{\mathbb{P}}2$, i.e. φ is satisfied by every assignment, denoted by $\models \varphi$.
- *unsatisfiable* (*a contradiction*), if $\overline{v}(\varphi) = 0$ for every $v \in {}^{\mathbb{P}}2$, i.e. $\neg\varphi$ is valid.
- *independent* (*a contingency*), if $\overline{v_1}(\varphi) = 0$ and $\overline{v_2}(\varphi) = 1$ for some $v_1, v_2 \in {}^{\mathbb{P}}2$, i.e. φ is neither a tautology nor a contradiction.
- *satisfiable*, if $\overline{v}(\varphi) = 1$ for some $v \in {}^{\mathbb{P}}2$, i.e. φ is not a contradiction.

Propositions φ and ψ are (logically) *equivalent*, denoted by $\varphi \sim \psi$, if $\overline{v}(\varphi) = \overline{v}(\psi)$ for every $v \in {}^{\mathbb{P}}2$, i.e. the proposition $\varphi \leftrightarrow \psi$ is valid.

Models

We reformulate these semantic notions in the terminology of models.

A *model of a language* $\mathbb{P}$ is a truth assignment of $\mathbb{P}$. The class of all models of $\mathbb{P}$ is denoted by $M(\mathbb{P})$, so $M(\mathbb{P}) = {}^{\mathbb{P}}2$. A proposition φ over $\mathbb{P}$ is

- *true in a model* $v \in M(\mathbb{P})$, if $\overline{v}(\varphi) = 1$. Then v is a *model of* φ, denoted by $v \models \varphi$ and $M^{\mathbb{P}}(\varphi) = \{v \in M(\mathbb{P}) \mid v \models \varphi\}$ is the *class of all models* of φ.
- *valid* (*a tautology*) if it is true in every model of the language, denoted by $\models \varphi$.
- *unsatisfiable* (*a contradiction*) if it does not have a model.
- *independent* (*a contingency*) if it is true in some model and false in other.
- *satisfiable* if it has a model.

Propositions φ and ψ are (logically) *equivalent*, denoted by $\varphi \sim \psi$, if they have same models.

Adequacy

The language of propositional logic has *basic* connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$.
In general, we can introduce n-ary connective for any Boolean function, e.g.

$p \downarrow q$	*"neither p nor q"*	(NOR, Peirce arrow)
$p \uparrow q$	*"not both p and q"*	(NAND, Sheffer stroke)

A set of connectives is *adequate* if they can express any Boolean function by some (well) formed proposition from them.

Proposition $\{\neg\,, \wedge\,, \vee\}$ *is adequate.*

Proof Any $f\colon {}^n2 \to 2$ is expressed by the proposition $\bigvee_{v \in f^{-1}[1]} \bigwedge_{i=0}^{n-1} p_i^{v(i)}$ where $p_i^{v(i)}$ stands for the proposition p_i if $v(i) = 1$; and for $\neg p_i$ if $v(i) = 0$. For $f^{-1}[1] = \emptyset$ we take the proposition $\bot$. $\square$

Proposition $\{\neg\,, \rightarrow\}$ *is adequate.*

Proof $(p \wedge q) \sim \neg(p \rightarrow \neg q)$, $(p \vee q) \sim (\neg p \rightarrow q)$. $\square$

CNF and DNF

- A *literal* is a propositional letter or its negation. For a propositional letter p let p^0 denote the literal $\neg p$ and let p^1 denote the literal p. For a literal l let $\bar{l}$ denote the *complementary* literal of l.
- A *clause* is a disjunction of literals, by the empty clause we mean $\perp$.
- A proposition is in *conjunctive normal form* (*CNF*) if it is a conjunction of clauses. By the empty proposition in CNF we mean $\top$.
- An *elementary conjunction* is a conjunction of literals, by the empty conjunction we mean $\top$.
- A proposition is in *disjunctive normal form* (*DNF*) if it is a disjunction of elementary conjunctions. By the empty proposition in DNF we mean $\perp$.

Note A clause or an elementary conjunction is both in CNF and DNF.

Observation *A proposition in CNF is valid if and only if each of its clauses contains a pair of complementary literals. A proposition in DNF is satisfiable if and only if at least one of its elementary conjunctions does not contain a pair of complementary literals.*

Transformations by tables

Proposition *Let $K \subseteq {}^{\mathbb{P}}2$ where $\mathbb{P}$ is finite. Denote $\overline{K} = {}^{\mathbb{P}}2 \setminus K$. Then*

$$M^{\mathbb{P}}\Big(\bigvee_{v\in K}\bigwedge_{p\in\mathbb{P}} p^{v(p)}\Big) = K = M^{\mathbb{P}}\Big(\bigwedge_{v\in\overline{K}}\bigvee_{p\in\mathbb{P}} \overline{p^{v(p)}}\Big)$$

Proof The first equality follows from $\overline{w}(\bigwedge_{p\in\mathbb{P}} p^{v(p)}) = 1$ whenever $w = v$, for every $w \in {}^{\mathbb{P}}2$. Similarly, the second one follows from $\overline{w}(\bigvee_{p\in\mathbb{P}} \overline{p^{v(p)}}) = 1$ whenever $w \neq v$. $\square$

For example, $K = \{(1,0,0),(1,1,0),(0,1,0),(1,1,1)\}$ can be modeled by

$$(p \wedge \neg q \wedge \neg r) \vee (p \wedge q \wedge \neg r) \vee (\neg p \wedge q \wedge \neg r) \vee (p \wedge q \wedge r) \sim$$
$$(p \vee q \vee r) \wedge (p \vee q \vee \neg r) \wedge (p \vee \neg q \vee \neg r) \wedge (\neg p \vee q \vee \neg r)$$

Corollary *Every proposition has CNF and DNF equivalents.*

Proof The value of a proposition φ depends only on the assignment of $\mathrm{var}(\varphi)$ which is finite. Hence we can apply the above proposition for $K = M^{\mathbb{P}}(\varphi)$ and $\mathbb{P} = \mathrm{var}(\varphi)$. $\square$

Transformations by rules

Proposition *Let φ' be the proposition obtained from φ by replacing some occurrences of a subformula ψ with ψ'. If $\psi \sim \psi'$, then $\varphi \sim \varphi'$.*

Proof Easily by induction on the structure of the formula. □

(1) $(\varphi \rightarrow \psi) \sim (\neg\varphi \vee \psi), \quad (\varphi \leftrightarrow \psi) \sim ((\neg\varphi \vee \psi) \wedge (\neg\psi \vee \varphi))$

(2) $\neg\neg\varphi \sim \varphi, \quad \neg(\varphi \wedge \psi) \sim (\neg\varphi \vee \neg\psi), \quad \neg(\varphi \vee \psi) \sim (\neg\varphi \wedge \neg\psi)$

(3) $(\varphi \vee (\psi \wedge \chi)) \sim ((\psi \wedge \chi) \vee \varphi) \sim ((\varphi \vee \psi) \wedge (\varphi \vee \chi))$

(3)' $(\varphi \wedge (\psi \vee \chi)) \sim ((\psi \vee \chi) \wedge \varphi) \sim ((\varphi \wedge \psi) \vee (\varphi \wedge \chi))$

Proposition *Every proposition can be transformed into CNF / DNF applying the transformation rules $(1), (2), (3)/(3)'$.*

Proof Easily by induction on the structure of the formula. □

Proposition *Assume that φ contains only $\neg$, $\wedge$, $\vee$ and φ^* is obtained from φ by interchanging $\wedge$ and $\vee$, and by complementing all literals. Then* $\neg\varphi \sim \varphi^*$.

Proof Easily by induction on the structure of the formula. □

2-SAT

- A proposition in CNF is in *k-CNF* if every its clause has at most k literals.
- *k-SAT* is the following problem (for fixed $k > 0$)

  INSTANCE: *A proposition φ in k-CNF.*

  QUESTION: *Is φ satisfiable?*

Although for $k = 3$ it is an NP-complete problem, we show that 2-SAT can be solved in *linear* time (with respect to the length of φ).

We neglect implementation details (computational model, representation in memory) and we use the following fact, see [ADS I].

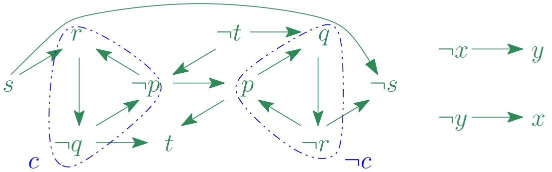
Proposition *A partition of a directed graph (V, E) to strongly connected components can be found in time $\mathcal{O}(|V| + |E|)$.*

- A directed graph G is *strongly connected* if for every two vertices u and v there are directed paths in G both from u to v and from v to u.
- A strongly connected *component* of a graph G is a maximal strongly connected subgraph of G.

Implication graphs

An *implication graph* of a proposition φ in 2-CNF is a directed graph G_φ s.t.

- vertices are all the propositional letters in φ and their negations,
- a clause $l_1 \vee l_2$ in φ is represented by a pair of edges $\overline{l_1} \to l_2$, $\overline{l_2} \to l_1$,
- a clause l_1 in φ is represented by an edge $\overline{l_1} \to l_1$.



$$p \wedge (\neg p \vee q) \wedge (\neg q \vee \neg r) \wedge (p \vee r) \wedge (r \vee \neg s) \wedge (\neg p \vee t) \wedge (q \vee t) \wedge \neg s \wedge (x \vee y)$$

Proposition *φ is satisfiable if and only if no strongly connected component of G_φ contains a pair of complementary literals.*

Proof Every satisfying assignment assigns the same value to all the literals in a same component. Thus the implication from left to right holds (necessity).

Satisfying assignment

For the implication from right to left (sufficiency), let G^*_φ be the graph obtained from G_φ by contracting strongly connected components to single vertices.

Observation G^*_φ *is acyclic, and therefore has a topological ordering* $<$.

- A directed graph is *acyclic* if it is has no directed *cycles*.
- A linear ordering $<$ of vertices of a directed graph is *topological* if $p < q$ for every edge from p to q.

Now for every unassigned component in an increasing order by $<$, assign 0 to all its literals and 1 to all literals in the complementary component.

It remains to show that such assignment v satisfies φ. If not, then G^*_φ contains edges $p \to q$ and $\overline{q} \to \overline{p}$ with $v(p) = 1$ and $v(q) = 0$. But this contradicts the order of assigning values to components since $p < q$ and $\overline{q} < \overline{p}$. □

Corollary 2-*SAT can be solved in a linear time.*