From syllogism to common sense: a tour through the logical landscape

Propositional logic

Mehul Bhatt Oliver Kutz *Thomas Schneider*

24 November 2011

Propositional logic (PL) . . .

- allows to analyse connections of given sentences A, B, such as

  A and B, A or B, not A, if A then B;

  but only for certain meanings of these connections.

- does not allow to analyse connections of temporal or modal nature:

  first A, then B,
  here A, there B,
  it is necessarily true that A

- is based on a beautiful mathematical theory that explains principles relevant for many other logics

Literature

Contents is taken from Chapter 1 of

W. Rautenberg:
A Concise Introduction to Mathematical Logic, Springer, 2010.

- This issue at Universitext: DOI 10.1007/978-1-4419-1221-3_1
- German version of 2008: DOI 10.1007/978-3-8348-9530-1
- Chapter 1 available in StudIP under “Dateien”

Plan for today and the next 1–2 weeks . . .

1. Boolean functions and formulas
2. Semantic equivalence and normal forms
3. Tautologies and logical consequence
4. A calculus of natural deduction
5. Application of the compactness theorem
6. Hilbert calculi

And now ...

1. Boolean functions and formulas
2. Semantic equivalence and normal forms
3. Tautologies and logical consequence
4. A calculus of natural deduction
5. Application of the compactness theorem
6. Hilbert calculi

Principles of two-valued logics

- Principle of bivalence:
  there are only two truth values – true and false
  - no third (fourth, ...) truth value
  - no degrees of truth
  - interpretation of true and false is irrelevant
    $\rightsquigarrow$ denote them with $1, 0$ or $\top, \bot$ or $\mathtt{t}, \mathtt{f}$
- Principle of extensionality:
  truth value of a sentence depends only
  on *truth values* of its parts, not on their *meaning*
- Classical modal, temporal, description, first-order logic and other logics build on these principles.
- Of course, principles are an idealisation!
  (If that doesn't suffice, change your logic.)

Joining two sentences

- Let A, B be sentences. Then the following are also sentences:

$A \wedge B$	conjunction *A and B* true if both A, B are true, and false otherwise.
$A \vee B$	(inclusive) disjunction *A or B* true if $\geqslant 1$ of A, B is true, and false otherwise.

- $\wedge, \vee$ are Boolean connectives
- $\wedge$ corresponds to a binary function $f : \{0,1\}^2 \rightarrow \{0,1\}$,
  given by its value matrix $\begin{pmatrix} 1 \wedge 1 & 1 \wedge 0 \\ 0 \wedge 1 & 0 \wedge 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$
- Analogously: $\vee$ corresponds to a binary function given by

$$\begin{pmatrix} 1 \vee 1 & 1 \vee 0 \\ 0 \vee 1 & 0 \vee 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

Let's generalise: joining n sentences

- A function $f : \{0,1\}^n \rightarrow \{0,1\}$ is called n-ary Boolean function or truth function.
- $\mathcal{B}_n$ = set of all n-ary Boolean functions

Questions to you

- How many unary (binary) Boolean functions are there?
- What is the cardinality of $\mathcal{B}_n$?

Prominent members:

- constants $0, 1 \in \mathcal{B}_0$
- negation $\neg \in \mathcal{B}_1$ defined by $\neg 1 = 0$ and $\neg 0 = 1$
- conjunction and disjunction from $\mathcal{B}_2$

Common binary connections in English and in logic

(We'll now use Boolean connectives/functions interchangeably.)

compound sentence	symbol	truth table
conjunction *A and B; A as well as B*	$\land$, &	1 0 0 0
disjunction *A or B*	$\lor$, $\bigvee$	1 1 1 0
implication *if A then B; B provided A*	$\rightarrow$, $\Rightarrow$	1 0 1 1
equivalence *A if and only if B; A iff B*	$\leftrightarrow$, $\Leftrightarrow$	1 0 0 1
exclusive disjunction *either A or B but not both*	$+$	0 1 1 0
nihilation *neither A nor B*	$\downarrow$	0 0 0 1
incompatibility *not at once A and B*	$\uparrow$	0 1 1 1

From W. Rautenberg: A Concise Introduction to Mathematical Logic, Springer, 2010.

Logical equivalence

- Two sentences are logically equivalent
  if their corresponding truth tables are identical.
- Example: *A provided B* $\equiv$ *A or not B* (Check for yourself!)
  (Converse implication $A \leftarrow B$)

$\rightsquigarrow$ Only few of the 16 binary Boolean functions require notation

- Example 2: *if A and B then C* $\equiv$ *if B then C provided A*

Goal

Recognise and systematically describe logical equiv. of sentences.

Use a formal language.
(Think of arithmetical formulas built from basic symbols.)

Syntax of propositional logic

- Basic symbols
  - propositional variables PV $= \{p, q, r, \dots\}$
  - logical connectives $\land, \lor, \neg, \dots$
  - parentheses (,) as a technical aid
- Formulas (Intuitive, recursive definition)
  1. $p, q, r, \dots$ are formulas (atomic formulas).
  2. If α, β are formulas, then so are $(\alpha \land \beta)$, $(\alpha \lor \beta)$, and $\neg\alpha$.
     (compound formulas)
- Examples:
  - $(p \land (q \lor \neg p))$ is a formula
  - $(p \land (q \lor \neg p)$ and $p\, q \land$ are not

Formulas, precise set-theoretic definition

... more useful for proving general theorems

Definition

Set $\mathcal{F}$ of all formulas is the smallest (i.e., the intersection)
of all sets S of strings built from the basic symbols,
with the properties

(f1) $p, q, \dots \in S$

(f2) if $\alpha, \beta \in S$, then $(\alpha \land \beta)$, $(\alpha \lor \beta)$, $\neg\alpha \in S$

Signatures

- Formulas are also called Boolean formulas: they are obtained using the Boolean signature $\{\wedge, \vee, \neg\}$
- Need further connectives? Extend your signature!
- However, $(\alpha \to \beta)$ and $(\alpha \leftrightarrow \beta)$ are just abbreviations:

$$(\alpha \to \beta) = (\neg\alpha \vee \beta)$$
$$(\alpha \leftrightarrow \beta) = ((\alpha \to \beta) \wedge (\beta \to \alpha))$$

  (Check their truth tables.)

- Extend signature by symbols that are always true (false): verum (falsum) $\top$ ($\bot$)
  - they are either additional atomic formulas
  - or abbreviations $\bot = (p \wedge \neg p)$, $\top = \neg\bot$

Parenthesis economy

Conventions similar to those in writing arithmetical terms

- Outermost parentheses of a formula may be omitted (if any).
  - string $(p \vee q) \wedge \neg p$ denotes formula $((p \vee q) \wedge \neg p)$
- Binding preference of binary connectives: $\neg, \wedge, \vee, \to, \leftrightarrow$, with $\neg$ binding most strongly
  - $p \vee q \wedge \neg p$ denotes $p \vee (q \wedge \neg p)$
- $\to$ is right-associative
  - $p \to q \to p$ denotes $p \to (q \to p)$
- all other binary connectives are left-associative
  - $p \wedge q \wedge \neg p$ denotes $(p \wedge q) \wedge \neg p$

The principle of formula induction

- Previous properties rely on intuitively clear facts, e.g.: identical number of left and right parentheses in a formula
- Such facts are usually proven via induction on the construction of a formula.
- Illustration of such an inductive proof with the above example:
  - use $\mathcal{E}\varphi$ to say that property $\mathcal{E}$ holds for string φ
  - E.g.: $\mathcal{E}\varphi \mathrel{\hat{=}}$ "φ is a formula that has equally many '(' and ')'"
  - $\mathcal{E}$ is trivially valid for atomic formulas
  - if $\mathcal{E}\alpha$ and $\mathcal{E}\beta$, then also $\mathcal{E}(\alpha \wedge \beta)$, $\mathcal{E}(\alpha \vee \beta)$, and $\mathcal{E}\neg\alpha$
  - Hence, $\mathcal{E}$ is valid for all propositional formulas

The principle of formula induction

Theorem

Let $\mathcal{E}$ be a property of strings that satisfies the conditions

(o) $\mathcal{E}\pi$ for all atomic formulas π,

(s) For all fmas α, β: if $\mathcal{E}\alpha$ and $\mathcal{E}\beta$, then $\mathcal{E}(\alpha\wedge\beta)$, $\mathcal{E}(\alpha\vee\beta)$, $\mathcal{E}\neg\alpha$.

Then $\mathcal{E}\varphi$ holds for all formulas φ.

Proof: easy given our precise definition of formulas on Slide 12:

- Take the set S of all formulas with property $\mathcal{E}$.
- Thanks to (o) and (s), S has properties (f1) and (f2).
- Since $\mathcal{F}$ is the smallest such set, $\mathcal{F} \subseteq S$.

$\Rightarrow$ $\mathcal{E}$ applies to all formulas $\varphi \in \mathcal{F}$. $\square$

(In the presence of other operators, Cond. (s) has to be extended.)

The unique formula reconstruction property

- Every compound fma. is of the form $\neg\alpha$ or $(\alpha \wedge \beta)$ or $(\alpha \vee \beta)$, for suitable $\alpha, \beta \in \mathcal{F}$.
- Intuitively clear and easily proven by induction.
- More interestingly, this decomposition is unique!
  E.g., $(\alpha \wedge \beta)$ cannot at the same time be, say, $(\alpha' \vee \beta')$

Theorem

Each compound formula $\varphi \in \mathcal{F}$
is of exactly one of the forms $\neg\alpha$ or $(\alpha \wedge \beta)$ or $(\alpha \vee \beta)$,
for some uniquely determined formulas $\alpha, \beta \in \mathcal{F}$.

- Is not obvious. Proof: exercise
- Does *not* rely on parentheses:
  e.g., Polish notation $\wedge\alpha\beta$, $\vee\alpha\beta$, $\neg\alpha$

Subformulas

- Subformulas of φ are all substrings of φ that are again fmas.
- Defined recursively on the construction of formulas
- The set of all subformulas of a fma. φ, written $\mathsf{sf}\,\varphi$, is defined as:

$$\begin{aligned} \mathsf{sf}\,\pi &= \{\pi\} \quad \text{for atomic formulas } \pi \\ \mathsf{sf}\,\neg\alpha &= \mathsf{sf}\,\alpha \cup \{\neg\alpha\} \\ \mathsf{sf}(\alpha \wedge \beta) &= \mathsf{sf}\,\alpha \cup \mathsf{sf}\,\beta \cup \{(\alpha \wedge \beta)\} \\ \mathsf{sf}(\alpha \vee \beta) &= \mathsf{sf}\,\alpha \cup \mathsf{sf}\,\beta \cup \{(\alpha \vee \beta)\} \end{aligned}$$

$\Rightarrow$ $\varphi \in \mathsf{sf}\,\varphi$

The rank of a formula

- Length of φ doesn't always provide a useful measure for the complexity of φ
- Alternative measure: rank of φ, written $\mathsf{rk}\,\varphi$, determines highest number of nested connectives in φ
- Defined recursively on the construction of formulas

$$\begin{aligned} \mathsf{rk}\,\pi &= 0 \quad \text{for atomic formulas } \pi \\ \mathsf{rk}\,\neg\alpha &= \mathsf{rk}\,\alpha + 1 \\ \mathsf{rk}(\alpha \wedge \beta) &= \max\{\mathsf{rk}\,\alpha, \mathsf{rk}\,\beta\} + 1 \\ \mathsf{rk}(\alpha \vee \beta) &= \max\{\mathsf{rk}\,\alpha, \mathsf{rk}\,\beta\} + 1 \end{aligned}$$

- (View φ as a tree $\rightsquigarrow$ rank $\triangleq$ depth of the tree)

Recursive definitions and inductive proofs

- Principle of defining a function f recursively on the construction of formulas
  relies on the unique formula reconstruction property.
- From now on, we'll say: f is defined by recursion on φ
- Similarly: property $\mathcal{E}$ is proven by induction on φ

Semantics of propositional logic

- Remember: Principle of extensionality – truth value of a sentence depends only on *truth values* of its parts, not on their *meaning*

$\leadsto$ assign truth value to every propositional variable in φ and use them to calculate the truth value of φ

$\leadsto$ every formula in n propositional variables describes an n-ary Boolean function

- (Analogy: evaluation of arithmetical terms over real numbers)

Semantics of propositional logic

- Propositional valuation is a mapping $w : \mathrm{PV} \to \{0,1\}$
- Can be understood as a mapping from atomic fmas to $\{0,1\}$.
- Every valuation w is extended to a mapping $w : \mathcal{F} \to \{0,1\}$:

$$
\begin{aligned}
w(\alpha \wedge \beta) &= w(\alpha) \wedge w(\beta) \\
w(\alpha \vee \beta) &= w(\alpha) \vee w(\beta) \\
w\neg\alpha &= \neg w\alpha
\end{aligned}
$$

Operators on the left-hand side: Boolean connectives
Operators on the right-hand side: Boolean functions!

- Value of φ under $w : \mathrm{PV} \to \{0,1\}$: value $w\varphi$ under the extension of w to $\mathcal{F}$

Semantics under extended signature

- If logical signature contains more connectives, e.g., $\rightarrow$, then the definition of extension must contain additional cases, e.g., $w(\alpha \rightarrow \beta) = w\alpha \rightarrow w\beta$.
- For $\rightarrow$, this is actually not necessary: remember, $(\alpha \rightarrow \beta)$ is an abbreviation of $(\neg\alpha \vee \beta)$

$\Rightarrow$ $w(\alpha \rightarrow \beta) = w(\neg\alpha \vee \beta) = w\neg\alpha \vee w\beta = \neg w\alpha \vee w\beta = w\alpha \rightarrow w\beta$

- Similarly, $w\top = 1$, $w\bot = 0$ (Check for yourself)

Formulas represent Boolean functions

- Let $\mathcal{F}_n$ be the set of all formulas in which at most the variables $p_1, \dots, p_n$ occur.
- Then the truth value $w\alpha$ depends only on $wp_1, \dots, wp_n$:

Theorem

For all $n \geqslant 0$, all $\alpha \in \mathcal{F}_n$, all valuations w, w':

$$\text{if } wp_i = w'p_i \text{ for all } i = 1, \dots, n, \quad \text{then } w\alpha = w'\alpha$$

(Proof via induction on $\varphi \in \mathcal{F}_n$.)

- Now we can define: $\alpha \in \mathcal{F}_n$ represents the function $f \in \mathcal{B}_n$ if, for all valuations w, it holds that $w\alpha = f(wp_1, \dots, wp_n)$
- Example: both $p_1 \wedge p_2$ and $\neg(\neg p_1 \vee \neg p_2)$ represent the $\wedge$-function