B1.1 Logic

Martin Bays

Oxford, MT 2023

PART I: Propositional Calculus

1. The language of propositional calculus

... is a very coarse language with limited expressive power;

... allows you to break a complicated sentence down into its subclauses, but not any further;

... will be refined in PART II *Predicate Calculus*, the true language of 1st order logic;

... is nevertheless well suited for entering formal logic.

1.1 Propositional variables

The propositional calculus implements logic of the following kind:

- 1. Socrates is alive or Socrates is dead.
  2. Socrates is not alive.
  Therefore: Socrates is dead.

- 1. If Socrates is a vampire and vampires are immortal, then Socrates is not dead.
  2. Socrates is dead.
  Therefore: Either Socrates is not a vampire, or vampires are not immortal.

We use *propositional variables* to denote propositions - e.g. p_0 for "Socrates is a vampire".

A *proposition* is something which can be true or false.

1.2 The alphabet of propositional calculus

The alphabet of the propositional language $\mathcal{L}_{\text{prop}}$ consists of the following symbols:

the propositional variables $p_0, p_1, \ldots, p_n, \ldots$

negation $\neg$ - the unary connective *not*

four binary connectives $\rightarrow$, $\wedge$, $\vee$, $\leftrightarrow$
implies, and, or and *if and only if* respectively

two punctuation marks (and)
left parenthesis and *right parenthesis*.

Note that these are *abstract symbols*.
Note also that we use $\rightarrow$, and not $\Rightarrow$.

1.3 Strings

- A **string (of $\mathcal{L}_{\text{prop}}$)** is any finite sequence of symbols from the alphabet of $\mathcal{L}_{\text{prop}}$.

- **Examples**

$$\begin{array}{rl} \text{(i)} & \to p_{17}() \\ \text{(ii)} & ((p_0 \wedge p_1) \to \neg p_2) \\ \text{(iii)} &))\neg)p_{32} \end{array}$$

- The **length** of a string is the number of symbols in it.
  So the strings in the examples have length $4, 10, 5$ respectively.
  (A propositional variable has length 1.)

- We now single out from all strings those which make grammatical sense (*formulas*).

1.4 Formulas

The notion of a **formula of** $\mathcal{L}_{\text{prop}}$ is defined (*recursively*) by the following rules:

I. Every propositional variable is a formula.

II. If the string A is a formula then so is $\neg A$.

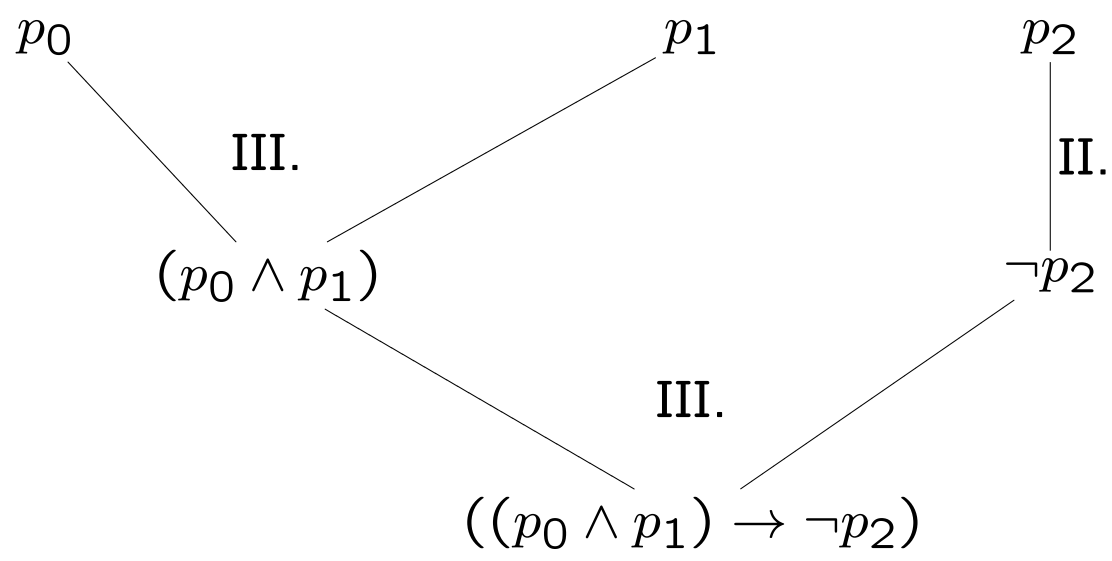
III. If the strings A and B are both formulas then so are the strings

$$\begin{array}{cl} (A \rightarrow B) & \text{read } A \textit{ implies } B \\ (A \wedge B) & \text{read } A \textit{ and } B \\ (A \vee B) & \text{read } A \textit{ or } B \\ (A \leftrightarrow B) & \text{read } A \textit{ if and only if } B. \end{array}$$

IV. Nothing else is a formula,
i.e. a string ϕ is a formula if and only if ϕ can be obtained from propositional variables by finitely many applications of the *formation rules* II. and III.

Examples

- The string $((p_0 \wedge p_1) \to \neg p_2)$ is a formula (Example (ii) in 1.3).
  Proof:



$\square$

- Parentheses are important, e.g. $(p_0 \wedge (p_1 \to \neg p_2))$ is a different formula and $p_0 \wedge (p_1 \to \neg p_2)$ is not a formula at all.

Examples

- The strings $\rightarrow p_{17}()$ and $))\neg)p_{32}$ from Example (i) and (iii) in 1.3 are not formulas.

  Indeed, if ϕ is a formula, then ϕ arises from one of I., II, or III., and so one of the following must hold:

  1. ϕ is a propositional variable.

  2. The first symbol of ϕ is $\neg$.

  3. The first symbol of ϕ is (.

The unique readability theorem

A formula can be constructed in only one way:

For each formula ϕ **exactly one** *of the following holds*

(a) ϕ is p_i for some unique $i \in \mathbb{N}$;

(b) ϕ is $\neg\psi$ for some **unique** formula ψ;

(c) ϕ is $(\psi \star \chi)$ for some **unique** pair of formulas ψ, χ and a **unique** binary connective $\star \in \{\rightarrow, \wedge, \vee, \leftrightarrow\}$.

Proof: Problem sheet 1.