propositions and connectives

... 

two-valued logic — every sentence is either true or false

some sentences are minimal — no proper part which is also a sentence

others — can be taken apart into smaller parts

we can build larger sentences from smaller ones by using connectives

propositional logic

... 

connectives — each has one or more meanings in natural language — need for precise, formal language

If I open the window then 1+3=4.

John is working or he is at home.

Euclid was a Greek or a mathematician.

propositional logic

... 

cannot do ...

if it is raining then everyone will have their umbrellas up

John is a person

John does not have his umbrella up

(it is not raining)

no way to substitute the name of the individual John into the general rule about everyone

propositional logic

... 

we will look at three different versions

- Hilbert system (the view that logic is about proofs)
- Gentzen system and tableau (or Beth) system (the view that is about consistency and entailment)
propositional logic

... two stages of formalization

- present a formal language
- specify a procedure for obtaining valid/true propositions

Hilbert system

**PROP\_H**

the syntax is extremely simple

- a set of names for prepositions
- single logical operator
- a single rule for combining simple expressions via logical operator (using brackets to avoid ambiguity)

propositional logic Hilbert system

**PROP\_H**

is a system which is tailored for talking about what can and what cannot be proved within the language, rather than for actually saying things and exploring entailments

Hilbert system

language ...

- **propositions names**: \( p_0, q_0, r_0, \ldots, p_1, p_2, \ldots \)
- **a name for “false”**: \( \bot \)

propositional logic Hilbert system

language ...

- **the connective**: \( \rightarrow \) (intended to be read “implies”)

  **single combining rule**: if \( A \) and \( B \) are expressions then so is \( A \rightarrow B \) (\( A, B \) stand for arbitrary expressions of the language)

Hilbert system - semantics

is given by explaining the meaning of the basic formulae - the proposition letters and \( \bot \)

is given in terms of two objects \( T \) and \( F \)

is given by providing a function \( V \) called **valuation** (it defines meaning of expressions)
propositional logic  
Hilbert system - semantics

\[ V(P) \] – maps \( P \) (basic proposition letters and \( \bot \)) to \{T, F\}

\(|expr|\) denotes the meaning assigned to \( expr \)

so, how to find out the value of complex expressions (built from simple ones using \( \rightarrow \), i.e., \(|A \rightarrow B|\) ...?

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A \rightarrow B</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
</tbody>
</table>

answer – truth tables

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A \rightarrow B</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
</tbody>
</table>

da truth table of complex expressions

\[ A \rightarrow (B \rightarrow A) \]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A \rightarrow B</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
</tbody>
</table>

negation - \( \neg \)

<table>
<thead>
<tr>
<th>A</th>
<th>\bot</th>
<th>A \rightarrow \bot</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
</tbody>
</table>

disjunction \( \lor \)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>\neg A</th>
<th>\neg A \rightarrow B</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
</tbody>
</table>

conjunction \( \land \)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>\neg A</th>
<th>\neg B</th>
<th>\neg A \lor \neg B</th>
<th>\neg (\neg A \lor \neg B)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
</tbody>
</table>
propositional logic
Hilbert system – inference rules

only one inference rule

for any expressions A and B:
from A and A → B we can infer B

it is known as MODUS PONENS (MP)

+ three axioms (expressions that are accepted as being true under any evaluation)

PH1: A → (B → A)
PH2: [A → (B → C)] → [(A → B) → (A → C)]
PH3: ¬ (¬ A) → A

proof of A → A

1 [A → ((A → A) → A)] → [(A → (A → A)) → (A → A)] (PH2)
2 [A → ((A → A) → A)]
3 [A → (A → A)] → (A → A) (MP on 1, 2)
4 (A → (A → A)) (PH1)
5 (A → A) (MP on 3, 4)

proof of r from p
hypotheses: p → q, q → r

1 p (hypothesis)
2 p → q (hypothesis)
3 q (from 1 and 2 by MP)
4 q → r (hypothesis)
5 r (from 3 and 4 by MP)
propositional logic
Hilbert system – inference rules

deduction theorem

if \( A_1, \ldots, A_n \vdash B \) then \( A_1, \ldots, A_{n-1} \vdash A_n \rightarrow B \)

proof of \( A \rightarrow \bot \)

1. \( \bot \rightarrow \bot \) (MP)
2. \( A \rightarrow \bot \rightarrow \bot \) (DT)
3. \( \bot \rightarrow ((A \rightarrow \bot) \rightarrow \bot) \) (DT)
4. \( \bot \rightarrow \neg \neg A \) (A \( \rightarrow \bot \equiv \neg \neg A \))

proof of \( \bot \rightarrow A \)

1. \( \bot \rightarrow A \) (DT)
2. \( A \rightarrow \bot \) (DT)
3. \( (A \rightarrow \bot) \rightarrow \bot \) (DT)
4. \( \bot \rightarrow \bot \) (DT)

propositional logic
Hilbert system – soundness
definition

if \( A_1, \ldots, A_n \vdash A \) then \( \| A \| = T \) for any valuation function \( V \) for which all the \( |A_i| \) are \( T \)

propositional logic
Hilbert system – completeness

would like to know that

if some conclusion is always true whenever every member of a given set of hypotheses is true,

then there is a proof of the conclusion from the hypotheses
propositional logic
Hilbert system – completeness definition

if $A$ is valid then $\vdash A$

propositional logic
Hilbert system – consistency definition

there is no proof of $\bot$

propositional logic
Hilbert system – decidability definition

there is a mechanical procedure which can decide for any $A$ whether or not $\vdash A$