The logical approach

- Define a **formal language**: logical & non-logical symbols, syntax rules
- Provide language with **compositional semantics**:
  - Fix **universe** of discourse
  - Specify how the non-logical symbols can be **interpreted**: interpretation
  - Rules how to combine interpretation of single symbols
  - Satisfying interpretation = **model**
  - Semantics often entails concept of **logical implication** / entailment
- Specify a **calculus** that allows to **derive** new formulae from old ones – according to the entailment relation

Motivation: Deductive Agent

<table>
<thead>
<tr>
<th>function <code>action</code> in (∆ ∈ D) out (α ∈ Ac)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: function <code>action</code> in (∆ ∈ D) out (α ∈ Ac)</td>
</tr>
<tr>
<td>2: for all α ∈ Ac do</td>
</tr>
<tr>
<td>3: if ∆ ⊨ p Do(α) then</td>
</tr>
<tr>
<td>4: return α</td>
</tr>
<tr>
<td>5: end if</td>
</tr>
<tr>
<td>6: end for</td>
</tr>
<tr>
<td>7: for all α ∈ Ac do</td>
</tr>
<tr>
<td>8: if ∆ ⊨ p ¬ Do(α) then</td>
</tr>
<tr>
<td>9: return α</td>
</tr>
<tr>
<td>10: end if</td>
</tr>
<tr>
<td>11: end for</td>
</tr>
<tr>
<td>12: return <code>null</code></td>
</tr>
</tbody>
</table>

- ∆: Set of formulae written in some logic.
- ⊨: Relation that holds between ∆s and formulae that can be derived from ∆.
Propositional logic: main ideas

- Non-logical symbols: propositional variables or atoms
  - representing propositions which cannot be decomposed
  - which can be true or false (for example: “Snow is white”, “It rains”)
- Logical symbols: propositional connectives such as:
  - and ($\land$), or ($\lor$), and not ($\neg$)
- Formulae: built out of atoms and connectives
- Universe of discourse: truth values

Syntax

Countable alphabet $\Sigma$ of propositional variables: $a, b, c, \ldots$

Propositional formulae are built according to the following rule:

$$\varphi ::= a \quad \text{atomic formula}$$
$$\perp \quad \text{falsity}$$
$$\top \quad \text{truth}$$
$$\neg \varphi' \quad \text{negation}$$
$$(\varphi' \land \varphi'') \quad \text{conjunction}$$
$$(\varphi' \lor \varphi'') \quad \text{disjunction}$$
$$(\varphi' \rightarrow \varphi'') \quad \text{implication}$$
$$(\varphi' \leftrightarrow \varphi'') \quad \text{equivalence}$$

Parentheses can be omitted if no ambiguity arises.
Operator precedence: $\neg > \land > \lor > \rightarrow = \leftrightarrow$.

2 Syntax

Language and meta-language

- $(a \lor b)$ is an expression of the language of propositional logic.
- $\varphi ::= a \ldots \mid (\varphi' \leftrightarrow \varphi'')$ is a statement about how expressions in the language of propositional logic can be formed. It is stated using meta-language.
- In order to describe how expressions (in this case formulae) can be formed, we use meta-language.
- When we describe how to interpret formulae, we use meta-language expressions.
3 Semantics

Semantics: idea

- Atomic propositions can be true (1, T) or false (0, F).
- Provided the truth values of the atoms have been fixed (truth assignment or interpretation), the truth value of a formula can be computed from the truth values of the atoms and the connectives.
- Example:

  
  \[(a \lor b) \land c\]

  is true if and only if \(c\) is true and, additionally, \(a\) or \(b\) is true.

Logical implication can then be defined as follows:

- \(\varphi\) is implied by a set of formulae \(\Theta\) iff \(\varphi\) is true for all truth assignments (world states) that make all formulae in \(\Theta\) true.

Example

Given \(\mathcal{I} : a \mapsto T,\ b \mapsto F,\ c \mapsto F,\ d \mapsto T,\)

Is \(((a \lor b) \leftrightarrow (c \lor d)) \land (\neg(a \land c) \lor (c \land \neg d))\) true or false?

\[(\neg(a \land c) \lor (c \land \neg d))\)

\[(\neg(a \land c) \lor (c \land \neg d))\)

\[(\neg(a \land c) \lor (c \land \neg d))\)

\[(\neg(a \land c) \lor (c \land \neg d))\)

\[(\neg(a \land c) \lor (c \land \neg d))\)

\[(\neg(a \land c) \lor (c \land \neg d))\)
4 Terminology

An interpretation \( \mathcal{I} \) is a model of \( \varphi \) iff \( \mathcal{I} \models \varphi \).
A formula \( \varphi \) is
- satisfiable if there is an \( \mathcal{I} \) such that \( \mathcal{I} \models \varphi \);
- unsatisfiable, otherwise; and
- valid if \( \mathcal{I} \models \varphi \) for each \( \mathcal{I} \) (or tautology);
- falsifiable, otherwise.

Formulae \( \varphi \) and \( \psi \) are logically equivalent (symb. \( \varphi \equiv \psi \)) if for all interpretations \( \mathcal{I} \),
\[ \mathcal{I} \models \varphi \text{ iff } \mathcal{I} \models \psi. \]

Examples

Satisfiable, unsatisfiable, falsifiable, valid?

\((a \lor b \lor \neg c) \land (\neg a \lor \neg b \lor d) \land (\neg a \lor b \lor \neg d)\)

~ satisfiable: \( a \mapsto \text{T}, b \mapsto \text{F}, d \mapsto \text{F}, \ldots \)
~ falsifiable: \( a \mapsto \text{F}, b \mapsto \text{F}, c \mapsto \text{T}, \ldots \)

\((\neg a \rightarrow \neg b) \rightarrow (b \rightarrow a)\)

~ satisfiable: \( a \mapsto \text{T}, b \mapsto \text{T} \)
~ valid: Consider all interpretations or argue about falsifying ones.

Equivalence? \( \neg(a \lor b) \equiv \neg a \land \neg b \)

~ Of course, equivalent (de Morgan).

Some obvious consequences

Proposition
\( \varphi \) is valid iff \( \neg \varphi \) is unsatisfiable.
\( \varphi \) is satisfiable iff \( \neg \varphi \) is falsifiable.

Proposition
\( \varphi \equiv \psi \) iff \( \varphi \leftrightarrow \psi \) is valid.

Theorem
If \( \varphi \equiv \psi \), and \( \chi' \) results from substituting \( \varphi \) by \( \psi \) in \( \chi \), then \( \chi' \equiv \chi \).
Some equivalences

- Simplifications: $\varphi \rightarrow \psi \equiv \neg \varphi \lor \psi$  
  $\varphi \leftrightarrow \psi \equiv (\varphi \rightarrow \psi) \land (\psi \rightarrow \varphi)$
- Idempotency: $\varphi \lor \varphi \equiv \varphi$  
  $\varphi \land \varphi \equiv \varphi$
- Commutativity: $\varphi \lor \psi \equiv \psi \lor \varphi$  
  $\varphi \land \psi \equiv \psi \land \varphi$
- Associativity:
  - Distributivity: $\varphi \land (\psi \lor \chi) \equiv (\varphi \land \psi) \lor (\varphi \land \chi)$
  - $\varphi \lor (\psi \land \chi) \equiv (\varphi \lor \psi) \land (\varphi \lor \chi)$
- Absorption: $\varphi \lor (\varphi \land \psi) \equiv \varphi$  
  $\varphi \land (\varphi \lor \psi) \equiv \varphi$
- Double negation: $\neg \neg \varphi \equiv \varphi$
- De Morgan:
  - Truth: $\neg (\varphi \lor \psi) \equiv \neg \varphi \land \neg \psi$  
  $\varphi \lor \top \equiv \top$  
  $\varphi \land \top \equiv \varphi$
  - Falsity: $\neg \top \equiv \bot$  
  $\varphi \lor \bot \equiv \bot$
- Taut./contrad.: $\varphi \lor \neg \varphi \equiv \top$  
  $\varphi \land \neg \varphi \equiv \bot$

How many different formulae are there …

… for a given finite alphabet $\Sigma$?

- Infinitely many: $a, a \lor a, a \land a, a \lor a, \ldots$
- How many different logically distinguishable (not equivalent) formulae?
  - A formula can be characterized by its set of models (if two formulae are not logically equivalent, then their sets of models differ).
  - For $\Sigma$ with $n = |\Sigma|$, there are $2^n$ different interpretations.
  - There are $2^{2^n}$ different sets of interpretations.
  - There are $2^{2^n}$ (logical) equivalence classes of formulae.

Logical implication

- $\varphi$ is logically implied by $\Theta$ (symbolically $\Theta \models \varphi$) iff $\varphi$ is true in all models of $\Theta$:
  - $\Theta \models \varphi$ iff $\mathcal{I} \models \varphi$ for all $\mathcal{I}$ such that $\mathcal{I} \models \Theta$
  - Deduction theorem: $\Theta \cup \{ \varphi \} \models \psi$ iff $\models \varphi \rightarrow \psi$

Deciding entailment

- We want to decide $\Theta \models \varphi$.
  - Use deduction theorem and reduce to validity:
    - $\Theta \models \varphi$ iff $\land \Theta \rightarrow \varphi$ is valid.
  - Now negate and test for unsatisfiability using tableaux.
  - Different approach: Try to derive $\varphi$ from $\Theta$ – find a proof of $\varphi$ from $\Theta$.
  - Use inference rules to derive new formulae from $\Theta$.
  - Continue to deduce new formulae until $\varphi$ can be deduced.
Propositional Tableaux

- **Goal**: Prove the unsatisfiability of a formula by trying to construct a model.
- **General principle**: Break each formula into its components up to the simplest one, where contradiction is easy to spot.
- Tableaux algorithm for propositional logic always terminates, and is sound and complete.

Propositional Tableaux

- A tableaux is a tree. Each branch of that tree corresponds to one attempt to find a model for the input formula.
- Initial Tableaux consists of the node: \( \Theta \land \neg \varphi \)
  - \( \Theta \models \varphi \) iff \( \Theta \land \varphi \) is valid iff \( \neg (\Theta \land \varphi) \) is unsatisfiable
  - The tableaux can be incrementally extended by applying rules:
    - **And-Rule**: If \( \varphi \land \psi \) is in a branch, then add \( \varphi \) and \( \psi \) to it.
    - **Or-Rule**: If \( \varphi \lor \psi \) is in a branch, then add \( \varphi \) to it, add a new branch, and add \( \psi \) to it.
    - **Implication**: If \( \varphi \to \psi \) is in a branch, then add \( \neg \varphi \) to it, add a new branch, and add \( \psi \) to it.

Propositional Tableaux: Closed Tableaux

- A branch is saturated if no more rule can be applied.
- A branch is closed if it contains formulae \( \varphi \) and \( \neg \varphi \).
- A tableaux is closed if all branches are closed.
- If the tableaux is closed, this means no model for the input formula could be found, hence, its negation is valid.

- **NotNot**: If \( \neg \neg \varphi \) is in a branch, then add \( \varphi \) to it.
- **NotAnd**: If \( \neg (\varphi \land \psi) \) is in a branch, then add \( \neg \varphi \) to it, add a new branch, and add \( \neg \psi \) to it.
- **NotOr**: If \( \neg (\varphi \lor \psi) \) is in a branch, then add \( \neg \varphi \) and \( \neg \psi \) to it.
- **NotImplication**: If \( \neg (\varphi \to \psi) \) is in a branch, then add \( \varphi \) and \( \neg \psi \) to that branch.