THE LANGUAGE OF FIRST-ORDER LOGIC AND KNOWLEDGE REPRESENTATION

In the preceding example, we used the language of standard First-Order Logic (FOL) for the formal representation of knowledge and reasoning.

Now: Take a closer look at the language of FOL

Remember:
Operations can be defined formally (**proof theory**)
and receive semantic properties (**truth theory**): $\boxed{\vdash \; \equiv \; \models}$

Contents:

- A brief summary of the classical introduction to propositional and quantificational ("predicate") logic,
- Another example,
- Reasoning problems, decidability, and expressive power,
- Extensions motivated by requirements for knowledge representation and automated reasoning.

Remark: For a pragmatically based, constructive foundation of logic by means of dialogue games
⇒ *Excursus: "A Constructive Introduction to First Order Logic"*

Calculi of Formal Logic

(Summary of the Classical Approach)

Distinction: **Provability** and **Truth**

Formal theory $\mathcal{T}$:

1. Symbols, expressions (sequences of symbols)
2. Well formed formulas (wff) $\subset$ expressions
3. Axioms $\subset$ wffs
4. Inference rules (relations on wffs)

Important concepts:

- *Schema*: statement form;
  Instance: wff obtained by substitution.
  Infinite axiom systems via axiom schemata, metavariables
- P is *deducible* from Σ in $\mathcal{T}$: $\Sigma \vdash_{\mathcal{T}} P$;
  Concepts: *derivation, proof*
  P theorem: $\vdash_{\mathcal{T}} P$, $\quad P$ provable in $\mathcal{T}$
- *Interpretations*: meaning for each symbol s.t. any wff can be understood as a true or false statement in the interpretation.
- *Model*: interpretation for a set of wffs s.t. every wff is true in interpretation.
  Model for a theory $\mathcal{T} \rightleftharpoons$ model for the set of theorems of $\mathcal{T}$

- *Completeness* of $\mathcal{T}$: Every sentence that is true in all interpretations is provable in $\mathcal{T}$ (truth $\rightarrow$ provability)
- *Soundness* of $\mathcal{T}$: Every provable sentence is true in all interpretations (provability $\rightarrow$ truth)
- *Decidability* of $\mathcal{T}$: Effective procedure exists that will determine provability for any sentence
- *Consistency* of $\mathcal{T}$: contains no wff s.t. wff and its negation are provable

Fundamentals of Propositional Logic: Syntax

$\Rightarrow$ Logic course (e.g. R. Davis: Truth, Deduction and Computation, 1989)

Syntax:

- Symbols
  - Proposition letters (variables), subscripted
  - Connectives: $\rightarrow$, $\neg$
  - Auxiliary symbols: (,)
- Well-formed formulas (wffs)
  - Prime (atomic) formulas: proposition letters
  - Compound formulas: If P, Q are wffs, so $\neg P$ and $P \rightarrow Q$

- Further propositional connectives (redundant):
  - $A \wedge B \rightleftharpoons \neg(A \rightarrow \neg B)$
  - $A \vee B \rightleftharpoons \neg A \rightarrow B$
  - $A \leftrightarrow B \rightleftharpoons (A \rightarrow B) \wedge (B \rightarrow A)$

Fundamentals of Propositional Logic: Semantics

Semantics: Truth

Truth tables

A	$\neg A$
$\top$	$\bot$
$\bot$	$\top$

A	B	$A \rightarrow B$
$\top$	$\top$	$\top$
$\top$	$\bot$	$\bot$
$\bot$	$\top$	$\top$
$\bot$	$\bot$	$\top$

A	B	$A \vee B$
$\top$	$\top$	$\top$
$\top$	$\bot$	$\top$
$\bot$	$\top$	$\top$
$\bot$	$\bot$	$\bot$

A	B	$A \wedge B$
$\top$	$\top$	$\top$
$\top$	$\bot$	$\bot$
$\bot$	$\top$	$\bot$
$\bot$	$\bot$	$\bot$

A	B	$A \leftrightarrow B$
$\top$	$\top$	$\top$
$\top$	$\bot$	$\bot$
$\bot$	$\top$	$\bot$
$\bot$	$\bot$	$\top$

Fundamentals of Propositional Logic: Semantics (2)

- Truth tables: all interpretations
- Connectives as logical functions
- P *logically valid*: true in all interpretations
  $\models P$ (tautology)
- *Satisfiability*: wff is true in (at least) one interpretation
- *Unsatisfiability* of P: $\neg P$ is logically valid (a contradiction)

Propositional Logic: Deduction

Proof theory: Deduction

Axiomatisation in Hilbert style:

Axioms of $\mathcal{L}$

$\mathcal{L}1$ $(A \to (B \to A))$
$\mathcal{L}2$ $((A \to (B \to C)) \to ((A \to B) \to (B \to C)))$
$\mathcal{L}3$ $(((\neg B) \to (\neg A)) \to (((\neg B) \to A) \to B))$

Inference rule: Modus Ponens

$MP \;\; A, A \to B \vdash B$

Propositional Logic: Proof Example

Lemma: $A \to A$

Proof:

1. $(A \to ((A \to A) \to A)) \to ((A \to (A \to A)) \to (A \to A))$
   an instance of axiom schema $\mathcal{L}2$, with A for A, $(A \to A)$ for B, A for C
2. $A \to ((A \to A) \to A)$
   axiom schema $\mathcal{L}1$ with A for A, $(A \to A)$ for B
3. $(A \to (A \to A)) \to (A \to A)$ by MP on 1 and 2
4. $A \to (A \to A)$ axiom schema $\mathcal{L}1$, A for A, A for B
5. $A \to A$ by MP on 3 and 4

Propositional Logic: Metatheory

- Deduction theorem:
  Let Σ set of wffs, A, B wffs, and $\Sigma, A \vdash B$, then $\Sigma \vdash A \to B$.
  For $\Sigma = \emptyset$: If $A \vdash B$ then $\vdash A \to B$.
- Completeness of $\mathcal{L}$: If $\models P$ then $\vdash P$
- Soundness of $\mathcal{L}$: If $\vdash P$ then $\models P$
- Hence, truth $\equiv$ deduction, i.e. $\vdash P \Leftrightarrow \models P$
- Consistency
- Decidability

Seven Derived Inference Rules for Propositional Logic Natural Deduction Style

- Modus Ponens or subjunction elimination

$$\frac{\alpha \rightarrow \beta, \quad \alpha}{\beta}$$

- And elimination

$$\frac{\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n}{\alpha_i}$$

- And introduction

$$\frac{\alpha_1, \alpha_2, \ldots, \alpha_n}{\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n}$$

- Or introduction

$$\frac{\alpha_i}{\alpha_1 \vee \alpha_2 \vee \ldots \vee \alpha_n}$$

- Elimination of double negation

$$\frac{\neg\neg\alpha}{\alpha}$$

- Unit RESOLUTION

$$\frac{\alpha \vee \beta, \quad \neg\beta}{\alpha}$$

- RESOLUTION

$$\frac{\alpha \vee \beta, \quad \neg\beta \vee \gamma}{\alpha \vee \gamma} \quad \text{or equivalently} \quad \frac{\neg\alpha \rightarrow \beta, \quad \beta \rightarrow \gamma}{\neg\alpha \rightarrow \gamma}$$

Complementary literal elimination (ground resolution) provides the computational basis for propositional logic.

Soundness of the Resolution Inference Rule: Truth Table

α	β	γ	$\alpha \vee \beta$	$\neg\beta \vee \gamma$	$\alpha \vee \gamma$
False	*False*	*False*	*False*	*True*	*False*
False	*False*	*True*	*False*	*True*	*True*
False	*True*	*False*	*True*	*False*	*False*
False	*True*	*True*	*True*	*True*	*True*
True	*False*	*False*	*True*	*True*	*True*
True	*False*	*True*	*True*	*True*	*True*
True	*True*	*False*	*True*	*False*	*True*
True	*True*	*True*	*True*	*True*	*True*

Fundamentals of Quantificational Logic ("Predicate Calculus")

Investigation of truth and falseness of composed formulas whose atomic parts are no longer certain sentences, but *sentence forms (parameterized sentences)*.

Introduction of *variables* and *instantiation (substitution)*;
Sentence forms determine relations expressed by *predicates* over their arguments.

First order theories:
Restriction of arguments to terms which can be constructed from constants, variables and functional expressions and quantification over (simple) variables.

Fundamentals of Quantificational Logic: Syntax

Symbols of first order language:

- Connectives: $\neg, \rightarrow, \bigwedge$
- Auxiliary symbols: (,), ","
- Variables: $x, x_1, x_2, \ldots$
- Constant symbols: $a, a_1, a_2, \ldots$
- Function symbols: $f_k^n \quad (n, k \in \mathbf{N})$
- Predicate symbols: $A_k^n \quad (n, k \in \mathbf{N})$

Well-formed formulas:

- term ::= variable | constant | function_letter "(" termlist ")"
- termlist ::= term | term "," termlist
- prime_formula ::= predicate_letter "(" termlist ")"
- wff ::= prime_formula | "(¬" wff ")"
  "(" wff "→" wff ")" | "((⋀" variable ")" wff ")"

Existential quantifier: $((\bigvee x)A) \rightleftharpoons \neg((\bigwedge x)\neg A)$

Scope of a quantifier: wff to which it applies

Bound and **free** variables

Fundamentals of Quantificational Logic: Semantics

Interpretation:

1. Domain $\mathcal{D} \neq \emptyset$
2. Assignment to each n-ary predicate symbol A_k^n of an n-place relation in $\mathcal{D}$
3. Assignment to each n-ary function symbol f_k^n of an n-place operation closed over $\mathcal{D}$: $\mathcal{D}^n \longrightarrow \mathcal{D}$
4. Assignment to each individual constant a_i of some fixed element of $\mathcal{D}$

Closed wff $\rightleftharpoons$ wff containing no free variables

Fundamentals of Quantificational Logic: Semantics (2)

- Satisfiability in an interpretation, defined recursively;
  e.g. A sequence of elements s of $\mathcal{D}$ satisfies $\neg A$ if and only if s does not satisfy A.
  Note: Circularity on metalevel!
- Truth: A wff is true (for a given interpretation) iff every sequence in the set of sequences Σ satisfies A. A is false iff no sequence in Σ satisfies A
- Logical validity of a wff A iff A is true for every interpretation.
- Satisfiability of a wff A iff there is an interpretation for which A is satisfied by at least one sequence in Σ.
- A is contradictory (unsatisfiable) iff $\neg A$ is logically valid (i.e., iff A is false for every interpretation).

- A logically implies B iff, in every interpretation, any sequence satisfying A also satisfies B.
  Logical consequence of a set of sequences . . .
- Any sentence of a formal language that is an instance of a logically valid wff is called logically true, and an instance of a contradictory wff is said to be logically false.

Fundamentals of Quantificational Logic: Deduction

Proof theory

Axiomatisation in Hilbert style:
logical ($\mathcal{PL}$) and nonlogical (theory specific) axioms

Axioms of $\mathcal{PL}$

$\mathcal{PL}1 \ldots \mathcal{PL}3 = \mathcal{L}1 \ldots \mathcal{L}3$
$\mathcal{PL}4 \quad \bigwedge_x A(x) \rightarrow A(t);$ term t free for x in $A(x)$
$\mathcal{PL}5 \quad \bigwedge_x (A \rightarrow B) \rightarrow (A \rightarrow \bigwedge_x B);$
A is a wff in which x does not occur free.

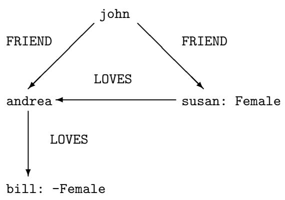
Inference rules:
$MP \quad A, A \rightarrow B \vdash B$ (Modus Ponens)
$Gen \quad A \vdash \bigwedge_x A$

Fundamentals of Quantificational Logic: Metatheory

- Consistency
- Deduction Theorem
- Soundness
- Completeness
- $\vdash S \leftrightarrow\ \models S$ (proof theory vs. model theory)
- UNDECIDABILITY (semi-decidability)

Theories and Models in FOL: An Example

$\Gamma =$ FRIEND(john,susan) $\wedge$ FRIEND(john,andrea) $\wedge$ LOVES(susan,andrea) $\wedge$ LOVES(andrea,bill) $\wedge$ Female(susan) $\wedge$ $\neg$Female(bill)



Example (cont.)

Does John have a female friend loving a male (i.e. not female) person?

$\Gamma \models \bigvee_{X,Y}.$ FRIEND(john,X) $\wedge$ Female(X) $\wedge$ LOVES(X,Y) $\wedge\neg$Female(Y).

Answer: **YES**

$\Gamma \not\models$ Female(andrea)
$\Gamma \not\models \neg$Female(andrea)

Unique minimal model:

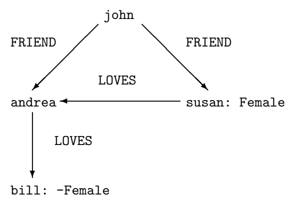
$\Delta^{\mathcal{I}} = \{$john,andrea,bill$\}$
Female$^{\mathcal{I}} = \{$susan$\}$

Two paths:

1. FRIEND(john,susan), Female(susan), LOVES(susan,andrea), $\neg$Female(andrea)
2. FRIEND(john,andrea), Female(andrea), LOVES(andrea,bill), $\neg$Female(bill)

Example (cont.)

$\Gamma_1 =$
FRIEND(john,susan) $\wedge$ FRIEND(john,andrea) $\wedge$
LOVES(susan,andrea) $\wedge$ LOVES(andrea,bill) $\wedge$
Female(susan) $\wedge$
Male(bill) $\wedge \bigwedge_X.$ Male(X) $\leftrightarrow \neg$Female(X)



Male$\doteq\neg$Female

Example (cont.)

Does John have a female friend loving a male person?

$\Gamma_1 \models \bigvee_{X,Y}.\ \texttt{FRIEND(john,}X\texttt{)} \wedge \texttt{Female(}X\texttt{)} \wedge \texttt{LOVES(}X,Y\texttt{)}$
$\quad \wedge\ \texttt{Male(}Y\texttt{)}.$
$\Gamma_1 \not\models \texttt{Female(andrea)}$ $\Gamma_1 \not\models \neg\texttt{Female(andrea)}$
$\Gamma_1 \not\models \texttt{Male(andrea)}$
$\Gamma_1 \not\models \neg\texttt{Male(andrea)}$

Four minimal models, a unique one does not exist:

$\Delta_1^{\mathcal{I}} = \{\texttt{john,susan,andrea,bill}\}$
$\texttt{Female}_1^{\mathcal{I}} = \{\texttt{susan,andrea}\}$
$\texttt{Male}_1^{\mathcal{I}} = \{\texttt{bill,john}\}$

$\Delta_1^{\mathcal{I}} = \{\texttt{john,susan,andrea,bill}\}$
$\texttt{Female}_1^{\mathcal{I}} = \{\texttt{susan,andrea,john}\}$
$\texttt{Male}_1^{\mathcal{I}} = \{\texttt{bill}\}$

$\Delta_2^{\mathcal{I}} = \{\texttt{john,susan,andrea,bill}\}$
$\texttt{Female}_2^{\mathcal{I}} = \{\texttt{susan}\}$
$\texttt{Male}_2^{\mathcal{I}} = \{\texttt{bill,andrea,john}\}$

$\Delta_2^{\mathcal{I}} = \{\texttt{john,susan,andrea,bill}\}$
$\texttt{Female}_2^{\mathcal{I}} = \{\texttt{susan,john}\}$
$\texttt{Male}_2^{\mathcal{I}} = \{\texttt{bill,andrea}\}$

Model Checking

Verify that a given interpretation $\mathcal{I}$ is a model for a closed formula ϕ:
$\models_{\mathcal{I}} \phi$

An interpretation is also called a *relational structure*

Example:

$\Delta = \{a, b\}$
$P(a)$
$Q(b)$

is a model of the formula

$\bigvee_y (P(y) \wedge \neg Q(y)) \wedge \bigwedge_z (P(z) \vee Q(z))$

Further Reasoning Problems: Subsumption and Instance Checking

Subsumption

- $\phi \sqsupseteq \psi$ "ϕ subsumes ψ",
  ϕ and ψ are predicate symbols of same arity
- $\models \bigwedge_{\hat{x}} (\psi(\hat{x}) \rightarrow \phi(\hat{x}))$
- The subsumption relation is a *partial ordering* relation (transitive, reflexive, antisymmetric) in the space of predicates of same arity.

Instance checking

- The constant a is an instance of the unary predicate P
- $\Gamma \models P(a)$

Decidability

Given a logic $\mathcal{L}$, a reasoning problem is said to be **decidable**, if there exists a computational process (an algorithm) that solves the problem in a finite number of steps, i.e., the process terminates always.

- The problem of deciding whether a formula ϕ is logically implied by a theory Γ is undecidable in full FOL
- Logical implication is decidable if we restrict it to propositional logic
- Logical implication is decidable if we restrict FOL to use only at most *two* variable names ($= \mathcal{L}_2$)

The problem of (un)decidability is a general property of the problem and not of a particular algorithm solving it.

Expressive Power

- Some logics can be made decidable by sacrificing some *expressive power*.
- A logical language $\mathcal{L}_a$ has more expressive power than a logical language $\mathcal{L}_b$, if each formula of $\mathcal{L}_b$ denotes the "same" set of models as its corresponding formula in $\mathcal{L}_a$, and if there is a formula of $\mathcal{L}_a$ denoting a set of models which is denoted by no formula in $\mathcal{L}_b$.

Example:

Let $\mathcal{L}_a$ be FOL and $\mathcal{L}_b$ be FOL without negation and adjunction.
Given a common domain, the $\mathcal{L}_a$ formula $\bigvee_x(P(x) \vee Q(x))$ has a set of models which cannot be captured by any formula of $\mathcal{L}_b$.

Extensions to Standard FOL

Why do we consider extensions to the standard logical language(s)?

$\Rightarrow$ Requirements of knowledge representation / domain modelling and processing

- Representation of structured domains / structured objects
  Ontological parsimony of standard logic; domains are "flat"
  $\Rightarrow$ Need for *expressive adequateness* and *notational efficiency*
- Object sorts: Multisorted logics

In later chapters:

- Intensional expressions: Create contexts which violate standard principles of logic, e.g. substitution of identities

- Modalities:
  Necessity and possibility
  Epistemic operators (knowledge and belief)
- Temporal logic
- Intensional logic with types for natural language semantics: Montague
- Non-monotonic reasoning and reason maintenance

Multisorted Logics — A Simple Extension

- First order logics in which more than one sort of individual variables exists;
- are interpreted not over a unique domain, but on a variety of domains;
- can be reduced to one-sorted by suitable one-place predicates and relative quantifiers.

Notation:

$$\bigwedge_{x \in \mathbf{N}} x > 0 \quad \text{instead of} \quad \bigwedge_x \mathbf{N}(x) \rightarrow x > 0$$

Technical reasons for introducing sorts in automatic theorem proving: In a proof, a variable in a wff can be substituted by any term — only unifiability is decisive. Problem: This may cause a *huge* combinatorial overhead.

Solution: Each term in a formula is assigned a class (i.e., a *sort*) and unification is only allowed if both terms are of the same sort.

In programming languages ("types") and KR, sorts are usually related to each other; in the special case of hierarchical taxonomies we have *partial orders*: **"order-sorted logic"**.

Relevance Logic

A subjunction $A \rightarrow B$ shall only exist if antecedent A is relevant for consequent B, i.e. if B is justified relative to the assumption A (A is *required*).

Paradoxes of subjunction are caused by the truth functional definition of subjunction: $A \rightarrow B \quad \rightleftharpoons \quad \neg A \vee B$

Ex-falso-quodlibet of classical and intuitionistic logic trivializes inconsistent theories because it admits the deduction of arbitrary sentences, e.g. Lewis' paradoxes: $A \rightarrow (\neg A \rightarrow B)$ or $A, \neg A \prec B$, resp.

Ex-falso-quodlibet-negatio: $A \rightarrow (\neg A \rightarrow \neg B)$ or $A, \neg A \prec \neg B$
(from false follows the falseness of everyting),
and further paradoxes.