

Discrete Mathematics in Computer Science

E6. Advanced Concepts in Predicate Logic and Outlook

Malte Helmert, Gabriele Röger

University of Basel

Discrete Mathematics in Computer Science

— E6. Advanced Concepts in Predicate Logic and Outlook

E6.1 Free and Bound Variables

E6.2 Reasoning in Predicate Logic

E6.3 Summary and Outlook

E6.1 Free and Bound Variables

Free and Bound Variables: Motivation

Question:

- ▶ Consider a signature with variable symbols $\{x_1, x_2, x_3, \dots\}$ and an interpretation \mathcal{I} .
- ▶ **Which parts of the definition of α are relevant** to decide whether $\mathcal{I}, \alpha \models (\forall x_4(R(x_4, x_2) \vee (f(x_3) = x_4)) \vee \exists x_3 S(x_3, x_2))$?
- ▶ $\alpha(x_1), \alpha(x_5), \alpha(x_6), \alpha(x_7), \dots$ **are irrelevant** since those variable symbols occur in no formula.
- ▶ $\alpha(x_4)$ also is **irrelevant**: the variable occurs in the formula, but all occurrences are **bound** by a surrounding quantifier.
- ▶ \rightsquigarrow only assignments for **free variables** x_2 and x_3 relevant

German: gebundene und freie Variablen

Variables of a Term

Definition (Variables of a Term)

Let t be a term. The set of **variables** that occur in t , written as $var(t)$, is defined as follows:

- ▶ $var(x) = \{x\}$
for variable symbols x
- ▶ $var(c) = \emptyset$
for constant symbols c
- ▶ $var(f(t_1, \dots, t_k)) = var(t_1) \cup \dots \cup var(t_k)$
for function terms

terminology: A term t with $var(t) = \emptyset$ is called **ground term**.

German: Grundterm

example: $var(\text{product}(x, \text{sum}(k, y))) =$

Free and Bound Variables of a Formula

Definition (Free Variables)

Let φ be a predicate logic formula. The set of **free variables** of φ , written as $free(\varphi)$, is defined as follows:

- ▶ $free(P(t_1, \dots, t_k)) = var(t_1) \cup \dots \cup var(t_k)$
- ▶ $free((t_1 = t_2)) = var(t_1) \cup var(t_2)$
- ▶ $free(\neg\varphi) = free(\varphi)$
- ▶ $free((\varphi \wedge \psi)) = free((\varphi \vee \psi)) = free(\varphi) \cup free(\psi)$
- ▶ $free(\forall x \varphi) = free(\exists x \varphi) = free(\varphi) \setminus \{x\}$

Example: $free((\forall x_4(R(x_4, x_2) \vee (f(x_3) = x_4)) \vee \exists x_3 S(x_3, x_2)))$
=

Closed Formulas/Sentences

Note: Let φ be a formula and let α and β variable assignments with $\alpha(x) = \beta(x)$ for all free variables x of φ .

Then $\mathcal{I}, \alpha \models \varphi$ iff $\mathcal{I}, \beta \models \varphi$.

In particular, α is **completely irrelevant** if $free(\varphi) = \emptyset$.

Definition (Closed Formulas/Sentences)

A formula φ without free variables (i. e., $free(\varphi) = \emptyset$) is called **closed formula** or **sentence**.

If φ is a sentence, then we often write $\mathcal{I} \models \varphi$ instead of $\mathcal{I}, \alpha \models \varphi$, since the definition of α does not influence whether φ is true under \mathcal{I} and α or not.

Formulas with at least one free variable are called **open**.

Closed formulas with no quantifiers are called **ground formulas**.

German: geschlossene Formel/Satz, offene Formel,
Grundformel/variablenfreie Formel

Closed Formulas/Sentences: Examples

Question: Which of the following formulas are sentences?

- ▶ $(\text{Block}(b) \vee \neg \text{Block}(b))$
- ▶ $(\text{Block}(x) \rightarrow (\text{Block}(x) \vee \neg \text{Block}(y)))$
- ▶ $(\text{Block}(a) \wedge \text{Block}(b))$
- ▶ $\forall x(\text{Block}(x) \rightarrow \text{Red}(x))$

E6.2 Reasoning in Predicate Logic

Terminology for Formulas

The terminology we introduced for propositional logic equally applies to predicate logic:

- ▶ Interpretation \mathcal{I} and variable assignment α form a **model** of the formula φ if $\mathcal{I}, \alpha \models \varphi$.
- ▶ Formula φ is **satisfiable** if $\mathcal{I}, \alpha \models \varphi$ for at least one \mathcal{I}, α .
- ▶ Formula φ is **falsifiable** if $\mathcal{I}, \alpha \not\models \varphi$ for at least one \mathcal{I}, α .
- ▶ Formula φ is **valid** if $\mathcal{I}, \alpha \models \varphi$ for all \mathcal{I}, α .
- ▶ Formula φ is **unsatisfiable** if $\mathcal{I}, \alpha \not\models \varphi$ for all \mathcal{I}, α .

German: Modell, erfüllbar, falsifizierbar, gültig, unerfüllbar

All concepts can be used for the special case of **sentences**.

In this case we usually omit α . **Examples:**

- ▶ Interpretation \mathcal{I} is a **model** of a sentence φ if $\mathcal{I} \models \varphi$.
- ▶ Sentence φ is **unsatisfiable** if $\mathcal{I} \not\models \varphi$ for all \mathcal{I} .

Sets of Formulas: Semantics

Definition (Satisfied/True Sets of Formulas)

Let \mathcal{S} be a signature, Φ a set of formulas over \mathcal{S} , \mathcal{I} an interpretation for \mathcal{S} and α a variable assignment for \mathcal{S} and the universe of \mathcal{I} .

We say that \mathcal{I} and α **satisfy** the formulas Φ (also: Φ is **true** under \mathcal{I} and α), written as: $\mathcal{I}, \alpha \models \Phi$, if $\mathcal{I}, \alpha \models \varphi$ for all $\varphi \in \Phi$.

German: \mathcal{I} und α erfüllen Φ , Φ ist wahr unter \mathcal{I} und α

We may again write $\mathcal{I} \models \Phi$ if all formulas in Φ are sentences.

Logical Equivalence and Logical Consequences

We again we use the same concepts and notations as in propositional logic.

- ▶ A set of formulas Φ logically entails/implies formula ψ , written as $\Phi \models \psi$, if all models of Φ are models of ψ .
- ▶ For a single formula φ , we may write $\varphi \models \psi$ for $\{\varphi\} \models \psi$.
- ▶ Formulas φ and ψ are **logically equivalent**, written as $\varphi \equiv \psi$, if they have the same models.
 - ▶ Note that $\varphi \equiv \psi$ iff $\varphi \models \psi$ and $\psi \models \varphi$.

Important Theorems about Logical Consequences

Theorem (Deduction Theorem)

$KB \cup \{\varphi\} \models \psi$ iff $KB \models (\varphi \rightarrow \psi)$

German: Deduktionssatz

Theorem (Contraposition Theorem)

$KB \cup \{\varphi\} \models \neg\psi$ iff $KB \cup \{\psi\} \models \neg\varphi$

German: Kontrapositionssatz

Theorem (Contradiction Theorem)

$KB \cup \{\varphi\}$ is *unsatisfiable* iff $KB \models \neg\varphi$

German: Widerlegungssatz

These can be proved exactly the same way as in propositional logic.

Logical Equivalences

- ▶ All **logical equivalences of propositional logic** also hold in predicate logic (e. g., $(\varphi \vee \psi) \equiv (\psi \vee \varphi)$). (**Why?**)
- ▶ Additionally the following equivalences and implications hold:

$$(\forall x\varphi \wedge \forall x\psi) \equiv \forall x(\varphi \wedge \psi)$$

$$(\forall x\varphi \vee \forall x\psi) \models \forall x(\varphi \vee \psi)$$

$$(\forall x\varphi \wedge \psi) \equiv \forall x(\varphi \wedge \psi)$$

$$(\forall x\varphi \vee \psi) \equiv \forall x(\varphi \vee \psi)$$

$$\neg\forall x\varphi \equiv \exists x\neg\varphi$$

$$\exists x(\varphi \vee \psi) \equiv (\exists x\varphi \vee \exists x\psi)$$

$$\exists x(\varphi \wedge \psi) \models (\exists x\varphi \wedge \exists x\psi)$$

$$(\exists x\varphi \vee \psi) \equiv \exists x(\varphi \vee \psi)$$

$$(\exists x\varphi \wedge \psi) \equiv \exists x(\varphi \wedge \psi)$$

$$\neg\exists x\varphi \equiv \forall x\neg\varphi$$

but not the converse

if $x \notin \text{free}(\psi)$

if $x \notin \text{free}(\psi)$

but not the converse

if $x \notin \text{free}(\psi)$

if $x \notin \text{free}(\psi)$

Normal Forms (1)

Analogously to DNF and CNF for propositional logic there are several normal forms for predicate logic, such as

- ▶ **negation normal form (NNF)**:
negation symbols (\neg) are only allowed in front of atoms
- ▶ **prenex normal form**:
quantifiers must form the outermost part of the formula
- ▶ **Skolem normal form**:
prenex normal form without existential quantifiers

German: Negationsnormalform, Pränexnormalform, Skolemnormalform

Normal Forms (2)

Efficient methods transform formula φ

- ▶ into an **equivalent** formula in **negation normal form**,
- ▶ into an **equivalent** formula in **prenex normal form**, or
- ▶ into an **equisatisfiable** formula in **Skolem normal form**.

German: erfüllbarkeitsäquivalent

Inference Rules and Calculi

There exist correct and complete **proof systems (calculi)** for predicate logic.

- ▶ An example is the **natural deduction** calculus.
- ▶ This is (essentially) Gödel's Completeness Theorem (1929).
- ▶ However, one can show that correct and complete algorithms that prove that a given formula **does not** follow from a given set of formulas **cannot exist**.
- ▶ How are these statements reconcilable?

First-Order Resolution

- ▶ **Resolution** can be extended to predicate logic with the concept of **unification**.
- ▶ Predicate logic resolution is correct and **refutation-complete** and can therefore be used as a general reasoning algorithm for showing $\Phi \models \varphi$.
- ▶ However, by the discussion on the previous slide, if $\Phi \not\models \varphi$, the algorithm cannot always terminate.

E6.3 Summary and Outlook

Summary

- ▶ **Predicate logic** is more expressive than propositional logic and allows statements over **objects** and their **properties**.
- ▶ Objects are described by **terms** that are built from variable, constant and function symbols.
- ▶ Properties and relations are described by **formulas** that are built from predicates, quantifiers and the usual logical operators.
- ▶ **Bound** vs. **free** variables: to decide if $\mathcal{I}, \alpha \models \varphi$, only free variables in α matter
- ▶ **Sentences** (closed formulas): formulas without free variables

Summary

Once the basic definitions are in place, predicate logic can be developed in the same way as propositional logic:

- ▶ logical consequence
- ▶ deduction theorem etc.
- ▶ logical equivalences
- ▶ normal forms
- ▶ inference rules, proof systems, resolution

Other Logics (1)

- ▶ We considered **first-order** predicate logic.
- ▶ **Second-order** predicate logic allows quantifying over predicate symbols.
- ▶ There are intermediate steps, e. g., monadic second-order logic (all quantified predicates are unary) and **description logics** (foundation of the semantic web).

Second-Order Logic Example

Second-order logic example:

- ▶ “ T is the transitive closure of R ”
- ▶ conjunction of
 - ▶ $\forall x \forall y (R(x, y) \rightarrow T(x, y))$
“ T is a superset of R ”
 - ▶ $\forall x \forall y \forall z ((T(x, y) \wedge T(y, z)) \rightarrow T(x, z))$
“ T is transitive”
 - ▶ $\forall Q ((\forall x \forall y (R(x, y) \rightarrow Q(x, y)) \wedge \forall x \forall y \forall z ((Q(x, y) \wedge Q(y, z)) \rightarrow Q(x, z))) \rightarrow \forall x \forall y (T(x, y) \rightarrow Q(x, y)))$
“All supersets Q of R that are transitive are supersets of T ”
- ▶ impossible to express in first-order logic

Other Logics (2)

- ▶ **Modal logics** have new operators \Box and \Diamond .
 - ▶ classical meaning: $\Box\varphi$ for “ φ is necessary”,
 $\Diamond\varphi$ for “ φ is possible”.
 - ▶ temporal logic: $\Box\varphi$ for “ φ is always true in the future”,
 $\Diamond\varphi$ for “ φ is true at some point in the future”
 - ▶ epistemic logic: $\Box\varphi$ for “ φ is known”,
 $\Diamond\varphi$ for “ φ is possible”
 - ▶ doxastic logic: $\Box\varphi$ for “ φ is believed”,
 $\Diamond\varphi$ for “ φ is considered possible”
 - ▶ deontic logic: $\Box\varphi$ for “ φ is obligatory”,
 $\Diamond\varphi$ for “ φ is permitted”
 - ▶ ...
- ▶ very important in computer-aided verification

Other Logics (3)

- ▶ In **fuzzy logic**, formulas are not true or false but have values between 0 and 1.
- ▶ **Intuitionist logic** is “constructive” and excludes indirect proof methods such as the principle of the excluded third.
- ▶ **Non-monotonic logics** have rules with exceptions (e.g., default logic, cumulative logic).
- ▶ ... and there is a lot more