

# Logik für Informatiker

## Logic for computer scientists

---

Till Mossakowski, Lutz Schröder

# Overview

- Why is logic needed in computer science?
- Overview of the course
- The LPL book and software
- “Scheinkriterien”

# Motivation

# Why is logic needed in computer science?

# Why is logic needed in computer science?

- formal specification and verification

# Why is logic needed in computer science?

- formal specification and verification
- databases, WWW, artificial intelligence

# Why is logic needed in computer science?

- formal specification and verification
- databases, WWW, artificial intelligence
- algorithms & complexity

# Why is logic needed in computer science?

- formal specification and verification
- databases, WWW, artificial intelligence
- algorithms & complexity
- metatheory



# Why is logic needed in computer science?

- formal specification and verification
- databases, WWW, artificial intelligence
- algorithms & complexity
- metatheory
- (semi-)automated theorem proving

# Why is logic needed in computer science?

- formal specification and verification
- databases, WWW, artificial intelligence
- algorithms & complexity
- metatheory
- (semi-)automated theorem proving
- programming languages

# Formal specification and verification

# Formal specification and verification

- formal software and hardware development

# Formal specification and verification

- formal software and hardware development
- verification of existing software and hardware

# Formal specification and verification

- formal software and hardware development
- verification of existing software and hardware
- generation of test cases

# Formal specification and verification

- formal software and hardware development
- verification of existing software and hardware
- generation of test cases
- protocol verification, security (modal and temporal logics)

# Formal specification and verification

- formal software and hardware development
- verification of existing software and hardware
- generation of test cases
- protocol verification, security (modal and temporal logics)
- properties of telephone systems



# Formal specification and verification

- formal software and hardware development
- verification of existing software and hardware
- generation of test cases
- protocol verification, security (modal and temporal logics)
- properties of telephone systems
- Example: Pentium 4 arithmetic completely specified and verified with higher-order logic!

# Formal specification and verification

- formal software and hardware development
- verification of existing software and hardware
- generation of test cases
- protocol verification, security (modal and temporal logics)
- properties of telephone systems
- Example: Pentium 4 arithmetic completely specified and verified with higher-order logic!
- Example: NASA uses logic for testing software

# databases, WWW, artificial intelligence

# databases, WWW, artificial intelligence

- queries for web search; database queries (SQL)

# databases, WWW, artificial intelligence

- queries for web search; database queries (SQL)
- semantic web (XML-based)

# databases, WWW, artificial intelligence

- queries for web search; database queries (SQL)
- semantic web (XML-based)
- expert systems

# databases, WWW, artificial intelligence

- queries for web search; database queries (SQL)
- semantic web (XML-based)
- expert systems
- linguistics

# databases, WWW, artificial intelligence

- queries for web search; database queries (SQL)
- semantic web (XML-based)
- expert systems
- linguistics
- Example: CYC is a very large knowledge base containing over 1.5 Million “facts, rules-of-thumb and heuristics for reasoning about the objects and events of everyday life”—the CYC inference engine uses first-order logic!



# Algorithms & complexity

# Algorithms & complexity

- if a graph property can be stated in monadic second-order logic, there is an efficient algorithm for it

# Algorithms & complexity

- if a graph property can be stated in monadic second-order logic, there is an efficient algorithm for it
- complexity classes can be characterized by classes of logical formulas

# Algorithms & complexity

- if a graph property can be stated in monadic second-order logic, there is an efficient algorithm for it
- complexity classes can be characterized by classes of logical formulas
- Example: the proof of  $NL = Co-NL$  was based on this — the hope is to push this further towards  $P = ? NP$

# Metatheory

# Metatheory

- set theory has been formalized in first-order logic
  - this serves as a foundations for all of mathematics and theoretical computer science

# Metatheory

- set theory has been formalized in first-order logic
  - this serves as a foundations for all of mathematics and theoretical computer science
- Gödel's completeness theorem for first-order logic: semantics can be captured by formal proofs
  - even by machine-driven proofs!

# Metatheory

- set theory has been formalized in first-order logic
  - this serves as a foundations for all of mathematics and theoretical computer science
- Gödel's completeness theorem for first-order logic: semantics can be captured by formal proofs
  - even by machine-driven proofs!
- Gödel's incompleteness theorem for first-order logic + induction: some essential pieces of mathematics and theoretical computer science cannot be captured by formal systems!



# (Semi-)automated theorem proving

# (Semi-)automated theorem proving

- logical properties of finite state machines can be automatically checked (model checkers)

# (Semi-)automated theorem proving

- logical properties of finite state machines can be automatically checked (model checkers)
- more complex systems require semi-automated theorem proving

# (Semi-)automated theorem proving

- logical properties of finite state machines can be automatically checked (model checkers)
- more complex systems require semi-automated theorem proving
- verification of proofs is easy and fully automatic

# (Semi-)automated theorem proving

- logical properties of finite state machines can be automatically checked (model checkers)
- more complex systems require semi-automated theorem proving
- verification of proofs is easy and fully automatic
- Example: some theorem about Boolean algebras has been found by a computer

# (Semi-)automated theorem proving

- logical properties of finite state machines can be automatically checked (model checkers)
- more complex systems require semi-automated theorem proving
- verification of proofs is easy and fully automatic
- Example: some theorem about Boolean algebras has been found by a computer
- Example: several math text books have been verified with a semi-automatic prover  
(and small but inessential errors have been found)

# Programming languages

# Programming languages

- Many programming languages use logical and, or, not



# Programming languages

- Many programming languages use logical and, or, not
- Prolog = programming in logic

# Programming languages

- Many programming languages use logical and, or, not
- Prolog = programming in logic
- concentrates on **what** instead of **how**

# Programming languages

- Many programming languages use logical and, or, not
- Prolog = programming in logic
- concentrates on **what** instead of **how**
- involves non-deterministic search

# Programming languages

- Many programming languages use logical and, or, not
- Prolog = programming in logic
- concentrates on **what** instead of **how**
- involves non-deterministic search
- used for applications in linguistics and artificial intelligence

# Landscape of logics

Temporal logic

Prop

ModalProp

Logic of programs

Spatial logic

FOL

ModalFOL

HOL

# Overview of the course

- history of logic
- overview of logics for computer science
- propositional consequence
- Hintikka games
- propositional proofs
- resolution
- (semi-)automatic proving: SPASS, Isabelle
- first-order quantifiers
- first-order consequence
- multiple quantifiers
- first-order proofs, resolution
- induction, datatypes
- model theory
- soundness
- completeness
- applications, outlook

# Language, proof and logic



# Language, proof and logic

# Language, proof and logic

**LPL book** detailed introduction into first-order logic  
with many exercises

# Language, proof and logic

**LPL book** detailed introduction into first-order logic  
with many exercises

**Tarski's world** evaluate logical formulas within a blocks world

# Language, proof and logic

**LPL book** detailed introduction into first-order logic  
with many exercises

**Tarski's world** evaluate logical formulas within a blocks world

**Fitch** construct proofs

# Language, proof and logic

**LPL book** detailed introduction into first-order logic  
with many exercises

**Tarski's world** evaluate logical formulas within a blocks world

**Fitch** construct proofs

**Boole** construct truth tables

# Language, proof and logic

**LPL book** detailed introduction into first-order logic  
with many exercises

**Tarski's world** evaluate logical formulas within a blocks world

**Fitch** construct proofs

**Boole** construct truth tables

**Grinder** gives automatic feedback to your solutions  
(requires purchase of the book)

**Scheinkriterien etc.**

# Rooms

- Monday 13:00 - 15:00 GW2 B1410
- Thursday 13:00 - 15:00 MZH Senatssaal (1400)
- Exercises inserted flexibly (bring your Laptops with you)
- Web:  
[http://www.tzi.de/~lschrode/teaching/Logic\\_e.htm](http://www.tzi.de/~lschrode/teaching/Logic_e.htm)



# Scheinkriterien

- successful solution of 10 exercises from 7 different chapters, with deadlines as given in the course
  - to be found in the LPL book
  - **but:** only those listed on the website, marked with grades
  - grade is average of 10 best solutions, but only as good as the best fitch solution
  - groups of two students possible (20 exercises, same grade for both)
  - submitted to the Grinder or to us (depending on the exercise)
- and: in case of doubts, oral exam (“Fachgespräch”)

# Language, proof and logic

- for working with the Grinder, **each** student/group of two needs an **own new** book
- try easy exercises first, to reach the minimum of 10 (later on, you can improve: only the 10 best solutions count)
- **only** exercises with a successful report (by the Grinder or us) count
- the Grinder is always right (but some old versions of Fitch are buggy)

# Language, proof and logic

- bulk ordering: 57,- Euros per book
- down payment: 10,- Euros