

Logik für Informatiker Logic for computer scientists

Till Mossakowski

WiSe 2013/14

Overview

- Motivation
- Overview of the course
- The language of PL1
- Formal Proofs in Fitch
- The Boolean Connectives

Motivation

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 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

- queries for web search; database queries (SQL)
- ontologies and semantic web
- 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

- 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

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

- logical properties of finite state machines can be automatically checked (model checkers)
- more complex systems need semi-automated 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 pervasive errors have been found)

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

Overview of the course

Overview of the course

- propositional consequence
- Hintikka games
- propositional proofs
- resolution
- (semi-)automatic proving: SPASS, Isabelle
- first-order quantifiers
- first-order consequence

Language, proof and logic

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

Boole construct truth tables







Tarski's world evaluate logical formulas within a blocks world

Fitch construct proofs

Grinder gives automatic feedback to your solutions
→ requires purchase of the book (ca. 25 EUR) and
an ID (15 EUR)

We work with **version 1**. Do not buy version 2!

Exercises

- 6.36**  Benutzen Sie Tarski's World, um eine Welt zu konstruieren, in welcher die folgenden Sätze alle wahr sind....
- 6.37**  Reichen Sie einen informellen Beweis dafür ein, dass das folgende Argument logisch gültig ist....
- 6.38**   Ist das folgende Argument gültig? Wenn ja, benutzen Sie Fitch, um einen formalen Beweis von dessen Gültigkeit zu geben. Wenn nicht, erklären Sie, warum es ungültig ist und reichen Sie Ihre Erklärung bei Ihrem Dozierenden ein.
- 6.39**   Entwerfen Sie eine PL1-Sprache, welche es Ihnen erlaubt, die folgenden deutschen Sätze auszudrücken....

Organisation

- Tuesday 11:00 - 13:00 HS 3
- Exercises (bring your Laptops with you!)
- Web:
<http://theo.cs.uni-magdeburg.de/lehre/lehre13w/logik/>

Course Criteria

- Übungen: Übungsblätter mit Übungsaufgaben aus dem Buch.
- Es müssen zu Beginn jeder Übung Aufgaben votiert werden.
⇒ bei Aufforderung vortragen
 - Lösungsvorschläge werden diskutiert, sie müssen nicht gleich perfekt richtig sein
- Alternativ können auch Lösungen bestimmter (nicht aller) Aufgaben über das Programm “Submit” automatisch bewertet werden.
- Für die Zulassung zur Klausur sind folgende Leistungen zu erbringen:
 - Mindestens 2/3 der Übungsaufgaben müssen am Ende des Semesters votiert oder über “Submit” eingereicht worden sein.
 - Mindestens zweimal pro Semester muss man in den Übungen vorgetragen haben.

The language of PL1

Why an artificial language?

- study principles of sound reasoning
 - what is logical consequence?
- need to eliminate all ambiguity
- reveals ambiguity in natural language
- computers can process formal languages

The passage from natural to formal language is known as **formalisation**

The language of PL1: individual constants

- **Individual constants** are symbols that denote a person, thing, object
- Examples:
 - Numbers: 0, 1, 2, 3, ...
 - Names: Max, Claire
 - Formal constants: a, b, c, d, e, f, n1, n2
- Each individual constant must denote an existing object
- No individual constant can denote more than one object
- An object can have 0, 1, 2, 3 ... names

The language of PL1: predicate symbols

- **Predicate symbols** denote a property of objects, or a relation between objects
- Each predicate symbol has an **arity** that tell us how many objects are related
- Examples:
 - Arity 0: Gate0_is_low, A, B, ...
 - Arity 1: Cube, Tet, Dodec, Small, Medium, Large
 - Arity 2: Smaller, Larger, LeftOf, BackOf, SameSize, Adjoins
 - ...
 - Arity 3: Between

The interpretation of predicate symbols

- In **Tarski's world**, predicate symbols have a **fixed interpretation**, that not always completely coincides with the natural language interpretation
- In other PL1 languages, the interpretation of predicate symbols may **vary**. For example, \leq may be an ordering of numbers, strings, trees etc.
- Usually, the binary symbol $=$ has a fixed interpretation: **equality**

Atomic sentences

- in propositional logic (Boole):
 - propositional symbols: a, b, c, \dots
- in PL1 (Tarski's world):
 - application of predicate symbols to constants: $\text{Larger}(a,b)$
 - the **order** of arguments **matters**: $\text{Larger}(a,b)$ vs. $\text{Larger}(b,a)$
 - Atomic sentences denote **truth values** (true, false)

Function symbols

- **Function symbols** lead to more complex **terms** that denote objects. Examples:
 - father, mother
 - $+$, $-$, $*$, $/$
- This leads to new terms denoting objects:
 - father(max) mother(father(max))
 - $3*(4+2)$
- This also leads to new atomic sentences:
 - Larger(father(max),max)
 - $2 < 3*(4+2)$

Logical validity; satisfiability

A sentence A is a **logically valid**, if it is true in all circumstances.
A sentence A is a **satisfiable**, if it is true in at least one circumstance.

A **circumstance** is

- in propositional logic: a valuation of the atomic formulas in the set { true, false }
- in Tarski's world: a block world

Consequences . . .



Logical consequence

A sentence B is a **logical consequence** of A_1, \dots, A_n , if all circumstances that make A_1, \dots, A_n true also make B true.

In symbols: $A_1, \dots, A_n \models B$.

A_1, \dots, A_n are called **premises**, B is called **conclusion**.

In this case, it is a **valid argument** to infer B from A_1, \dots, A_n . If also A_1, \dots, A_n are true, then the valid argument is **sound**.

Logical consequence — examples

- All men are mortal. Socrates is a man. So, Socrates is mortal. (valid, sound)
- All rich actors are good actors. Brad Pitt is a rich actor. So he must be a good actor. (valid, but not sound)
- All rich actors are good actors. Brad Pitt is a good actor. So he must be a rich actor. (not valid)