Logic LVA 703600 VU3

http://cl-informatik.uibk.ac.at/teaching/ws05/logic/

Georg Moser (VU)[1] Christian Vogt (VU)[2]

[1]georg.moser@uibk.ac.at
office hours: Thursday 12am–2pm

[2]christian.vogt@uibk.ac.at
office hours: Tuesday 9am–11am

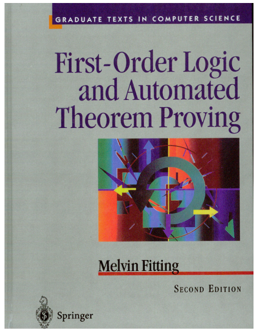
Autumn 2005

Schedule

week 1	October 5	week 7	November 30
week 2	October 12	week 8	December 7
week 3	October 19	week 9	December 14
week 4	November 9	week 10	January 11
week 5	November 16	week 11	January 18
week 6	November 23	1st exam	January 25

Literature & on-line Material

First-Order Logic and Automated Theorem Proving, Series: Texts in Computer Science, Fitting, Melvin, 2nd ed., 1996, XVII, 348 p. 15 illus., Hardcover ISBN: 0-387-94593-8



Transparencies and homework are available from IP starting with 138.232; solutions to selected exercises will be available on-line after they have been discussed.

Overview

Week 1	Introduction & Background
Week 2	Introduction to Formal Proof
Week 3	Propositional Logic: Syntax & Semantics
Week 4	The Replacement Theorem, Uniform Notations
Week 5	Semantic Tableaux (propositional case)
Week 6	Resolution (propositional case)
Week 7	First-Order Logic
Week 8	The Model Existence Theorem (first-order)
Week 9	Applications of the Model Existence Theorem
Week 10	First-Order Proof Procedure & Completeness
Week 11	Implementing Tableaux and Resolution

Organisational Matters

The lecture is a VU, i.e., "Vorlesung" and "Übung" are combined. One grade only will be given.

We offer 3 exercise-groups . . . attendence is not mandatory, but in your own interest.

There will be a mid-term test (45 min) on Novemer 16. For the final grade we take into account:

- The grade of the final exam (2/3)
- The grade of the mid-term test and your cooperation in the exercise classes (1/3)

UE	Group 1	Wednesday 16.00-17.00, HS 10	Georg Moser
	Group 2	Thursday 8.00-9.00, HS 10	Christian Vogt
	Group 3	Thursday 9.00-10.00, HS 10	Christian Vogt

There is not only one logic

There are many logics: temporal, modal, intuitionistic, fuzzy, dynamic, etc.

Especially useful in model checking and program verification: temporal and modal logics.

Scope

We deal with classical, propositional and first-order logic . . . for good reasons this is the basis of everything else.

Definition (informal)

Logic is the study of reasoning and mathematical logic is the study of the mathematical reasoning.

Example

Consider "That is an ugly chair" or "This is a nice lecture room". Are those assertions true?

Classical logic is incapable to deal with these assertions.

Definition (cont'd)

(Classical) Logic deals only with assertions that are either true or false; i.e. there must be no ambiguity (or discussion) about the truth of the assertions.

When dealing with non-mathematical examples, we abstract from the real-word and build a (mathematical) model.

In reasoning we use sentences. A primitive assertion like "The program P terminates" is a sentence.

To build complex sentences we use connectives like "and", "or", "not", "implies", etc. and quantifier: "every", "some", etc.

Definition

We interpret "P implies Q" in the material sense: "P implies Q" will have the same truth value as "either P is false or Q is true".

The language of first-order logic

$\wedge$	and	propositional connectives
$\vee$	or	
$\neg$	not	
$\rightarrow$	implies	
$\forall$	every	quantifiers
$\exists$	some	

Concurrent Processes: Mutual Exclusion

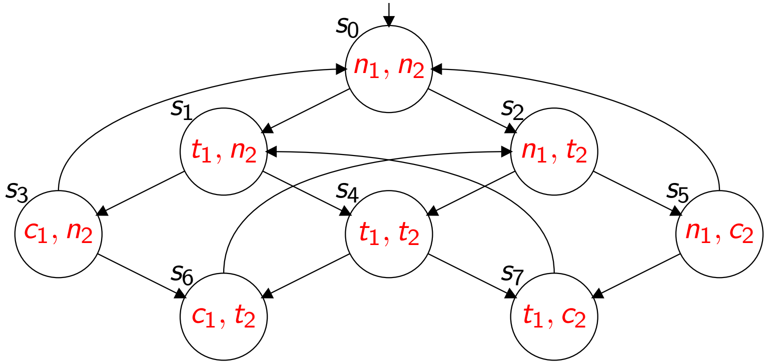
We identify a critical sections in the code of each process and arrange that at most one process has access to its critical section at a time.

Task Define a protocol, to determine which process is allowed to enter the critical section under what circumstances.

Properties

- Safety. Only one process in the critical section at one time.
- Liveness. When a process wants to enter its critical section, it will be allowed to.
- Non-blocking. A process can always request to enter the critical section.

Modelling 'mutual exclusion'



Process i is in non-critical state	n_i
Process i is trying to enter critical state	t_i
Process i is in its critical state	c_i

Our model is abstract; to talk about it, we define a language.

States

To speak about the states, we need names for these states. In this case 8 constant symbols $k_0, k_1, \ldots, k_7$ suffice.

Definition

Names are used to denote abstract objects, like states or numbers. Constants are very simply names; more complicated names make additional use of function symbols.

Transition relation

In the model we can transfere from one state to another ... expressed by the transition relation. To name this relation we introduce a binary relation symbol R.

$R(k_3, k_0)$ expresses that state s_0 is reachable from state s_3 in one step.

Definition

Expressions like $R(k_3, k_0)$ are called atomic formulas.

Propositions

To name the propositions c_i, n_i, t_i, we include unary relation symbols C_i, N_i, T_i.

Path

In the 'mutual exclusion' model we can follow a path. We introduce another binary relation symbol P to express this.

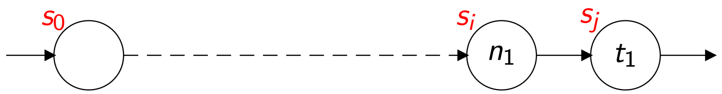
Complex statements

Starting with atomic formulas, using connectives and quantifiers, we build complex expressions, or formulas.

Example

$\forall x \neg(C_1(x) \wedge C_2(x))$ expresses Safety.

Non-Blocking



Example

$$\forall x((P(k_0, x) \wedge N_1(x)) \rightarrow \exists y(R(x, y) \wedge T_1(y))) \ .$$

The formula expresses that

- for all instances k of the variable x, either
  - the assertion $P(k_0, k)$ or the assertion $N_1(k)$ is false, or
  - the assertions $P(k_0, k)$ and $N_1(k)$ are true and
    - it is true that there exists an instance k' of y such that $R(k, k')$ and $T_1(k')$.

If we interpret the atomic formulas for the protocoll this means

for all states s, if s is reachable from s_0 and s is neutral, then there exists some next state s', such that s' is trying.

Summary

- Classical Logic is one logic among many, but provides the basis to the lot.
- Logic is the study of reasoning and mathematical logic is the study of the mathematical reasoning.
- To describe abstract objects, we need names for them. A constant is a name. Names can be generalised to terms.
- We can formalise assertions and express them by building them up (using connectives and quantifiers) from atomic formulas.