Satzung vom 11.10.2004 zur Änderung der Studienordnung der Fakultät Informatik der Technischen Universität Dresden für den internationalen Masterstudiengang Computational Logic vom 09.04.2003 (veröffentlicht in den Amtlichen Bekanntmachungen der TUD Nr.: 7/2003)

Auf Grund von § 21 des Gesetzes über die Hochschulen im Freistaat Sachsen (Sächsisches Hochschulgesetz - SächsHG) vom 11. Juni 1999 (SächsGVBl. S. 293), geändert durch Gesetz vom 28. Juni 2001 (SächsGVBl. S. 426, 428), erlässt die Technische Universität Dresden die nachstehende Änderungssatzung.

Artikel 1 Änderung der Studienordnung

Die Studienordnung für den internationalen Masterstudiengang Computational Logic vom 09.04.2003 wird wie folgt geändert:

1. In § 6 Abs. 2 Satz 2 wird "36 cr Pflichtmodule" durch "42 cr Pflichtmodule" sowie "42 cr Wahlpflichtmodule" durch "36 cr Wahlpflichtmodule" ersetzt.

2. Die Bezeichnung des Moduls "Introduction to Computational Logic" wird durchgängig geändert in "Foundations", die des Moduls "Foundations of Logic and Constraint Programming" in "Logic and Constraint Programming".

3. In Anlage 1: Stundentafel in cr wird die den Modulen "Foundations" (vormals "Introduction to Computational Logic") und "Logic and Constraint Programming" (vormals "Foundations of Logic and Constraint Programming") zugeordnete Zahl von jeweils 9 cr geändert in jeweils 12 cr. Gleichzeitig wird in den jeweiligen Modulbeschreibungen die Angabe "The total of 9 credit points" ersetzt durch "The total of 12 credit points".

4. In Anlage 1: Stundentafel in cr wird die den Wahlpflichtmodulen zugeordnete Zahl von 42 cr geändert in 36 cr.

5. In Anlage 1: Stundentafel in cr wird das Modul "Deduction Systems" ersetzt durch das Modul "Integrated Logic Systems". Die Modulbeschreibung für das Modul "Deduction Systems" in Anlage 2: Modulbeschreibungen wird ersetzt durch die Modulbeschreibung für das Modul "Integrated Logic Systems" in der dieser Satzung als Anlage beigefügten Fassung.

6. In den Beschreibungen der Wahlfplichtmodule "Knowledge Representation and Artificial Intelligence", "Specification and Verification", "Theoretical Computer Science and Logic" und "Syntax-Directed Semantics" wird jeweils "14 credit points" durch "13 credit points" ersetzt; in der Beschreibung des Wahlpflichtmoduls "Inference Techniques" wird "14 credit points" durch "10 credit points" ersetzt.

7. Die Beschreibungen der Module "Foundations", "Logic and Constraint Programming" und "Advanced Logics" werden aktualisiert und erhalten die in der Anlage beigefügte Fassung. Gleiches gilt für die Wahlpflichtmodule.

Artikel 2 In-Kraft-Treten und Übergangsbestimmungen

Die Änderungen treten mit Wirkung vom 01.10.2004 in Kraft und werden in den Amtlichen Bekanntmachungen der Technischen Universität Dresden veröffentlicht.

Ausgefertigt auf Grund des Senatsbeschlusses der Technischen Universität Dresden vom 14.04.2004 und der Anzeige beim Sächsischen Staatsministerium für Wissenschaft und Kunst.

Dresden, den 11.10.2004

Der Rektor
der Technischen Universität Dresden

Prof. Hermann Kokenge

Foundation Modules

Module: Foundations

Contact person: Prof. Hölldobler
Keywords: propositional logic; first order logic; deduction; proof theory; abduction and induction; knowledge representation and reasoning; complexity theory; computer algebra.

The module offers a comprehensive introduction to Computational Logic covering the main subareas as well as main methods and techniques. After recalling basic notions from propositional and first order logic, complexity theory and computer algebra, the areas of equational reasoning, deduction, proof theory, abduction and induction, non-monotonic reasoning, logic-based program development, natural language processing and machine learning as well as logic and connectionism are covered.

This module consists of lectures and tutorials. The total of 12 credit points can be scored by passing the final written examination of the module.

The module takes one semester and is offered every winter semester.

Prerequisites: none.

Module: Logic and Constraint Programming

Contact person: Prof. Thielscher
Keywords: unification; declarative, procedural, and operational semantics; logic programming; constraint logic programming; combinatorics and analysis of algorithms.

This module is concerned with the foundations of logic programming and constraint logic programming. The basic computation mechanisms of unification and SLD-resolution are introduced. The declarative and the operational semantics of logic programs are given and related to the procedural semantics. A logic programming language is introduced as an example of a declarative programming language. Logic programs with constraints are introduced and basic computation mechanisms given. The module concludes with examples of constraint logic programming languages. In addition basic knowledge of combinatorics and analysis of algorithms is taught.

After the successful completion of this module, students will have acquired a profound understanding of the mathematical principles of logic programming. Students will also have experience in using logic programming languages and constraint logic programming languages for problem solving.

This module consists of lectures and tutorials. The total of 12 credit points can be scored by passing the final written examination of the module.

The module takes one semester and will be offered every winter semester.

Prerequisites: none.

Module: Advanced Logic

Contact person: Prof. Reichel
Keywords: higher order logics; lambda calculus; lambda prolog; modal logics, epistemic logic; temporal logic; mu-calculus; CTL*; schematic tableaux; model theory.

The aim of this module is to introduce basic concepts beyond first-order predicate logics. In Computer Science many different logics and deductive systems exist. First we introduce higher order logic (HOL) as a framework for specifying syntactic and deductive notions of different logics. HOL is used in several interactive proof tools, like PVS and Isabelle. In addition, specific families of logics aimed at different application areas are introduced: logics of time and computation (modal logics, temporal Logics), logics for reasoning about knowledge (epistemic logic). Finally we introduce the mu-calculus which allows to define recursive temporal properties and we present a tableau based deduction calculus for the mu-calculus. The mu-calculus and its deduction system can be used to define problem oriented systems of modal operators and corresponding deduction systems.

This module consists of lectures and tutorials. The total of 9 credit points can be scored by passing the final written examination of the module.

The module takes one semester and will be offered every summer semester.

Prerequisites: Modules Foundations and Logic and Constraint Programming

Module: Integrated Logic Systems

Contact person: Prof. Schroeder
Keywords: Logic & computers, logic & other systems; logic & interfaces; logic & applications.

The module shall meet the demand for more practice-oriented subjects in the curriculum. By means of selected examples it shall give an insight to various issues of how logic can be used in real world applications. The students shall be introduced to how logic can be linked to computers (e.g. data structures, WAM, distributed computation), to other systems (e.g. ODCB, Oracle, Java, Interplay between Prolog and data bases) and to human interfaces (e.g. ML, Java). The principle techniques shall be supplemented by examples.

This module consists of lectures and tutorials. The total of 9 credit points can be scored by passing the final written examination of the module.

The module takes one semester and will be offered every summer semester.

Prerequisites: Modules Foundations and Logic and Constraint Programming

Advanced Modules

Module: Knowledge Representation and Artificial Intelligence

Contact person: Prof. Thielscher
Keywords: declarative representation of knowledge; automated reasoning with knowledge; knowledge-based systems; artificial intelligence applications.

This module is concerned with techniques for the declarative representation of knowledge and inference methods based on formalized knowledge. Introduced are standard representation formalisms for various kinds of knowledge (like temporal, dynamic, categorical, or grammatical knowledge). The mathematical properties of the formalisms are discussed. Calculi for inferring knowledge are given and analyzed. Principles for designing and building knowledge-based systems are introduced, and applications of knowledge representation and reasoning techniques to artificial intelligences are covered. The successful completion of this module enables students to understand and create knowledge representation formalisms, to analyze, design, and use algorithms for drawing inferences from formal knowledge, and to build and apply knowledge-based systems.

The total number of 13 credit points for this module are attained by lectures with tutorials, and possibly a seminar.

The module can be completed within two successive semesters and it will be offered every year.

Prerequisites: Basic knowledge in logic and reasoning as presented in the module `Foundations'.

Module: Specification and Verification

Contact person: Prof. Reichel
Keywords: constructive and declarative specification techniques, algebraic and coalgebraic specifications, inititial and final semantics, process algebras, Petri nets, induction, coinduction, distributed computing.

The module presents formal specification techniques for both the axiomatic and operational specification of software (and hardware) systems. The students learn to specify generic data types and functional enrichments of generic data types by means of initial semantics, to prove properties by induction, and to reason about the correctness of refinements. They can learn to specify the dynamic behavior of a system by means of Petri nets, to use algorithms on Petri nets to reason about the behavior, and to model concurrent systems by means of process algebras and to express dynamic properties using modal and temporal properties, and to apply coinduction as a fundamental definition and proof technique.

The total number of 13 credit points for this module are attained by lectures with tutorials, and possibly a seminar.

The module can be completed within two successive semesters and will be offered every year.

Prerequisites: basic knowledge in predicate logic and modal logics as presented in the module "Advanced logics".

Module: Theoretical Computer Science and Logic

Contact person: Prof. Baader
Keywords: complexity and computability theory, automata theory, algorithms, algebra, model theory.

This module is concerned with the application of advanced techniques and results from theoretical computer science (like automata on infinite objects, complexity results, term rewriting techniques, etc.) to the analysis of formal properties of different logics (like axiomatizations, proof-theoretic properties, design of algorithms and analysis of the complexity for logical inference problems, etc.). Building on the basic knowledge about automata, formal languages, and computability from the Bachelor studies and the introductory courses in CL, this module will introduce different such advanced techniques and then show how they can be applied in Computational Logic. After a successful completion of the module the students should have both, a working familiarity with different methods of theoretical computer science, and a good knowledge of formal properties of various logics.

The total number of 13 credit points for this module are attained by lectures with tutorials, and possibly a seminar.

The module can be completed within two successive semesters and it will be offered every year.

Prerequisites: basic knowledge in theoretical computer science and logics.

Module: Syntax-Directed Semantics

Contact person: Prof. Vogler
Keywords: denotational semantics, implementation of imperative, functional, and logic-pro-gramming program schemes of functional programming and tree transducers weighted automata.

The content of this module is the investigation of translations of syntactic structures into semantic structures. Here syntactic structures have the form of trees like derivation trees of programs; semantic structures can be complexes like 1. state space, environment, and continuations or 2. dependency graph between semantic values or 3. abstract machines. By abstracting from the operations in the semantic domain, we obtain trees as another, very general type of semantic structure. Whereas the more particular semantic structures lead to the areas of denotational semantics and implementation of imperative, functional, and logic-programming, the more abstract point of view can be called the theory of program schemes of functional programming and the theory of tree transducers (with weights).

The total number of 13 credit points for this module are attained by lectures with tutorials, and possibly a seminar.

The module can be completed within two semesters and it will be offered every year.

Prerequisites: basic knowledge about automata, formal languages, and computability as it is gained on the Bachelor level.

Module: Inference Techniques

Contact person: Prof. Hölldobler
Keywords: resolution; term rewriting; answer set programming; inductive theorem proving.

The module is concerned with the in-depth study of inference techniques. After recalling basic notions and techniques from automated reasoning some of the following methods and techniques will be presented in detail: Resolution, tableaux, connection or related methods for automated theorem proving; Term rewriting, superposition or related methods for equational reasoning; Answer set programming or related methods for non-monotic rasoning; Inductive theorem proving.

The total number of 10 credit points for this module are attained by lectures with tutorials, and possibly a seminar.

The module can be completed within two successive semesters and will be offered every second year.

Prerequisites: Basic knowledge in logic and reasoning as presented in the module `Foundations'.