Module Handbook for the Bachelor’s Degree in Computer Science

For the 2007, 2013, 2016 examination regulations

Institute of Computer Science
of the Faculty of Mathematics and Natural Sciences
at the Heinrich Heine University

Published by the
Committee for the Bachelor Examination
in Computer Science

Foreword

The module handbook is intended to provide orientation for the basic bachelor's degree in computer science. In particular, it should make it easier to choose courses and support the organization of the course.

The module handbook lists the usual courses. However, it is not a complete, exhaustive or definitive listing. Rather, the module handbook is continuously updated and thus reflects the development in research and teaching at the Institute of Computer Science.

Please note, however, that the respective bachelor's or master's examination regulations for the subject of computer science are decisive for all questions relating to studies and examinations.

Düsseldorf, October 10, 2022

The Committee for the Bachelor Examination in Computer Science

Table of Contents

Qualification goals of the B.Sc. in Computer Science

Upon completion of their studies, students will have achieved the following qualification goals.

Scientific qualifications and qualifications for employment

Graduates

- are able to compare and apply methods and procedures of computer science, such as systematic problem definitions, algorithmic problem solving, logical proof procedures or evaluation procedures.
- can develop applications using state-of-the art software technologies. This includes conception, implementation and testing.
- are familiar with a logical, analytical and systemic approach to thinking that enables them to analyse and solve problems in computer science, for example in fields of data science and machine learning.
- are able to apply learned knowledge and research methods for solving research problems and assessing the usefulness of the results, taking into account principles of good scientific practice.

Personal development

Graduates

- can assess their own skills with regard to aspects such as data analytics, software development or communication and already have ideas for their further development. They can independently acquire new specialised knowledge.
- can take responsibility for themselves and their tasks within a group.
- know a range of professional images (e.g. a data scientist or a software engineer) and have adopted their own.
- are able to communicate and discuss with others computer science problems and appropriate solutions, which they practiced in exercises and seminars
- are aware of basic ethical questions and challenges from the perspective of computer science as well as the social, cultural and political significance of their discipline.

Course Plans

You can find the recommended course plans

- for the PO 2016 at https://www.cs.hhu.de/bachelor/pruefungsordnung-2016.
- for the PO 20013 and 2007 at https://www.cs.hhu.de/bachelor/alte-pruefungsordnungen.

Compulsory Modules in Computer Science

Algorithms and Data Structures

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	90 hours	210 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

This module showcases a selection of fundamental algorithms and data structures from theory and practice. By analyzing these basic algorithms and data structures, the lecture explains how resource consumption (running time and memory requirements) may be theoretically analyzed and predicted.

- Foundations of algorithms
- Models of computation and complexity measures
- Search strategies (binary search)
- Algorithms for sorting (quicksort, heapsort, merge sort, …)
- Fundamental data structures (array lists, linked lists, stacks, queues)
- Search trees (binary search trees, balanced trees)
- Dictionaries (open and closed hashing)
- Managing systems of sets (union find)
- Amortized analysis
- Algorithms on graphs (depth first search, breadth first search, spanning trees, shortest paths)
- Design patterns (greedy, divide-and-conquer, dynamic programming)
- Limits of efficient algorithms (outlook)

Learning Outcomes

After completing the course, students are able to

- apply, analyze and explain the algorithms/data structures discussed in this module,
- recognize a suitable algorithm/data structure for a given problem and to select one from a repertoire,
- develop a specification for an algorithm or a data structure based on an informal description, and to explain a given specification and answer questions on a given specification
- use the fundamental techniques from the lecture to analyze, to predict and to compare the resource consumption of algorithms and data structures
- prove and convince others that a given algorithm works correctly and
- modify and combine the algorithms and data structures presented in the module.

Bibliography

- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein: Algorithmen – Eine Einführung. De Gruyter Oldenbourg. 2017. 4. Auflage.
- Robert Sedgewick, Kevin Wayne: Algorithmen. Pearson Studium. 2014. 4. Auflage.
- Thomas Ottmann, Peter Widmayer: Algorithmen und Datenstrukturen. Spektrum. 2012. 5. Auflage.
- Jon Kleinberg, Eva Tardos: Algorithm Design. Addison Wesley. 2006.

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Compulsory Area Bachelor study programme PO 2016, 2013

- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Module CL6 Bachelor study programme Computer Linguistics

Prerequisites

- Contents of module Mathematics for Computer Science 1

Conditions for awarding credit points

- active participation in the tutorials
- handing in the homework
- written exam (regularly 90 minutes) or oral exam at the end of the term

Responsible persons

Dr. Daniel Schmidt, Prof. Dr. Gunnar W. Klau, Prof. Dr. Melanie Schmidt

Professional Software Development

ECTS credits	Total hours	Contact hours	Self-study hours
8 CP (PO 2013: 10 CP)	240 hours	66 hours	174 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW) Lab (1 HPW)	every summer semester	Bachelor Computer Science	German

Remarks

- This module is no longer offered. Instead, take the Programming Project 1 module of PO 21.
- In PO 2013 this module is called Programming Project I and is weighted with 10 CP.
- The specification of the HPW for the contact time is not exact but rounded.

Content

The goal of the programming projects is to enable teams of students to develop a larger, webbased information system of high quality in Java. The first module contains:

- Development tools (IDE, build tools, version control systems)
- Automatic testing and test driven development
- Code quality and code smells
- Principles and practices to modularize larger systems using object-oriented programming (SOLID principles, Information Hiding, Coupling and Cohesion, Law of Demeter, Polymorphism, Dependency Injection)
- Basics of domain-driven design (ubiquotous language and tactical design)
- Systematic debugging
- Advanced Java concepts (Streams, Records, Optional, DateTime, …)

Learning Outcomes

After completing the course, students are able to:

- develop larger, maintainable systems based on a problem statement
- analyze the quality of given code and discuss the effect on its maintainability
- improve the quality and maintainabiliy of given code
- develop code driven by tests
- use Java language features idiomatically
- systematically find and fix bugs
- use common development tools (IDE, build tools, version control systems)

Bibliography

- Eigenes Script

Module compatibility

- Compulsory Area Bachelor study programme PO 2016, 2013
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

- Module CL6 Bachelor study programme Computer Linguistics

Prerequisites

- Formal: Successful completion of the module Programming

Conditions for awarding credit points

- Active and successful participation in exercises
- Written exam (60 min)

Responsible persons

Dr. Jens Bendisposto, Dr. Markus Brenneis

Programming

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	120 hours	180 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW) Exercise Course (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

This module teaches the basics of imperative, object-oriented programming using the Java programming language as an example. Aspects of algorithms and data structures are dealt with in an introductory manner. No previous programming experience is required.

- primitive data types and variables
- control structures
- arrays
- standard input and output, files
- program structures in memory (heap, stack)
- recursion
- concepts of object-oriented programming (classes, objects, polymorphism, interfaces, inheritance)
- error handling
- searching and sorting (linear search, binary search, insertion sort, mergesort)
- dynamic data structures (lists, binary search trees, hashing)

Learning Outcomes

After completing the course, students are able to

- explain given Java program code,
- translate descriptions of iterative/recursive algorithms into structured code and design their own simple algorithms,
- use standard input/output and files to read and output textual data,
- write object-oriented programs that model real-world objects,
- explain and apply advantages of polymorphism,
- expound on the advantages and disadvantages of different data structures (arrays, singly linked lists, binary search trees, hash sets),
- explain algorithms for different data structures (arrays, sinlgy linked lists, binary search trees, hash sets) and implement them, and
- fix compile-time errors and handle run-time errors appropriately.

Bibliography

- R. Schiedermeier: Programmieren mit Java. Pearson Studium. München, 2010. 2. aktualisierte Ausgabe
- C. Ullenboom: Java ist auch eine Insel. Rheinwerk Computing. Bonn, 2021. 16. Auflage
- R. Sedgewick & K. Wayne: Introduction to Programming in Java. Addison-Wesley, United States, 2007. 1st Edition

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Compulsory Area Bachelor study programme PO 2016, 2013
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Module I: Computer Science Bachelor study programme Computer Linguistics

Prerequisites

- None

Conditions for awarding credit points

- Active and successful participation in exercises
- Written exam (120 min)

Responsible persons

Dr. Markus Brenneis, Prof. Dr. Michael Schöttner

Computer Architecture

ECTS credits	Total hours	Contact hours	Self-study hours
9 CP	270 hours	120 hours	150 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture Rechnerachitektur (3 HPW) Tutorial Rechnerachitektur (2 HPW) Lecture Hardwarenahe Programmierung (1 HPW) Praktische Tutorial Hardwarenahe Programmierung (2 HPW)	every winter semester (Rechnerarchitektur) nicht mehr angeboten (Hardwarenahe Programmierung ab WS 2022/23)	Bachelor Computer Science	German

Remarks

This course is no longer offered.

- Instead of the lecture and exercise Rechnerarchitektur, please take the module *Computer Architecture* of PO21.
- Instead of Hardwarenahe Programmierung, please take the module *C-Programming for Algorithms and Data Structures* of PO21.

Content

The courses about computer architecture provide a basic understanding of the structure and functioning of modern computers. The following topics will be addressed:

- Data representation (ASCII, Unicode, b-adic numbers, two's complement, Qx.y fixed-point numbers, floating-point numbers according to IEEE 754),
- Two-element boolean algebra (definition according to Huntington, further rules for calculation, functionally complete sets , truth tables, disjunctive and conjunctive normal form, KV diagrams, disjunctive and conjunctive minimal form, don't care assignments),
- Digital logic (gates, decoders and encoders, multiplexers and demultiplexers, shifters, half and full adders, clock signals, flip-flops and latches (SR-, D- each), circuits made from these elements, hazard errors),
- Error detection and correction (duplication, parity bit, two-dimensional parity, Hamming distance, Hamming code),
- Microarchitecture (implementation of a reduced JVM instruction set for a sample architecture, translations between the sample assembly language and binary code, improvements to this architecture by reducing cycles, instruction prefetching, pipelining, branch prediction),
- Caching (replacement strategies, fully associative caches, direct mapped caches, n-way set-associative caches),
- Basics of x86 assembler programming (arithmetic and logical instructions, jumps and loops, stack management, cdecl calling convention, functions), and
- Virtual memory (page replacement strategies, paging).

Learning Outcomes

After completing the courses about computer architecture, students are able to

- describe the various layers of a computer architecture, taking into account their interconnections,
- indicate and evaluate Boolean functions in various forms (formula, truth table, KV diagram) and show the equivalence of functions,
- design digital circuits with the above elements and minimize them using the KV diagram,
- use known circuits to design larger circuits, especially a simple ALU for integer calculations or memory chips
- draw digital timing diagrams and use them or KV diagrams to find and eliminate possible hazard errors in digital circuits,
- explain how a CPU/ALU is constructed from elementary digital circuits and how it works,
- calculate the Hamming distance of words and (finite) codes and name the Hamming distance of known codes,
- apply known methods for error detection and correction and name which type of errors can be detected or corrected by them,
- discuss advantages and disadvantages of the cache types discussed,
- execute caching procedures on paper and compare different caching types and replacement strategies,
- develop and analyze simple assembler programs in x86 assembler,
- write functions considering the cdecl calling convention and show the stack development, and
- explain what paging is used for and perform conversions between virtual and physical addresses.

After successful participation in the courses on Hardware-related Programming, the students can

- develop programs in the C programming language, taking into account dynamic memory management
- using tools for typical programming tasks (memory management, build processes, tests).

Bibliography

- A. S. Tanenbaum, T. Austin: Structured Computer Organization. Pearson. Boston, 2013. 6th Edition
- P. A. Carter: PC Assembly Language. Online, 2019.
- D. Griffiths and D. Griffiths (dt. Lars Schulten): C von Kopf bis Fuß. O'Reilly Verlag. Beijing, 2012. 1. Auflage
- Ergänzende Empfehlungen werden in der Lehrveranstaltung genannt.

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Compulsory Area Bachelor study programme PO 2016, 2013
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Module CL6 Bachelor study programme Computer Linguistics

Prerequisites

- As regards content (Computer Architecture): It is assumed that the students will attend the programming course at the same time or have basic programming knowledge.
- Contentually (Computer Architecture): It is assumed that the students will attend the programming course at the same time or have basic programming knowledge.
- Contentually (Hardware-related Programming): It is assumed that the participants can use common program elements such as variables, branches, loops and functions with confidence.

Conditions for awarding credit points

- active and successful participation in the tutorials (Computer Architecture)
- written exam (exam Computer Architecture, usually 90 minutes)
- active and successful participation in the lab course (Hardware-related Programming)

Responsible persons

Prof. Dr. Stefan Conrad, Janine Golov, Prof. Dr. Martin Mauve,

Teamwork in Software Development

ECTS credits	Total hours	Contact hours	Self-study hours
8 CP	240 hours	124 hours	116 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW) Lab (1 HPW) Block Lab (4 HPW)	every winter semester	Bachelor Computer Science	German

Remarks

- This module is no longer offered. Instead, take the Programming Project 1 module of PO 21.
- The specification of the HPW for the contact time is not exact but rounded.

Content

The goal of the programming projects is to enable teams of students to develop a larger, web-based information system of high quality in Java. The second module contains:

- Webdevelopment (HTTP, HTML, Accessibility, Authentification and Authorization, Security, REST, Servlets, Spring Web)
- Accessing Databases (JDBC, Spring Data)
- Software Architecture and Architectural pattern
- Integration testing and Archictecture testing (ArchUnit)
- Software development processes
- Documentation of Architecture (arc42, UML)

Learning Outcomes

After completing the course, students are able to:

- implement and secure web applications
- ensure accessibility of web applications
- develop informations systems that use databases
- describe and apply architectural patterns
- write integration test for web-based information systems
- write tests to ensure that architectural decision are respected
- describe and apply common software development processes
- write documentation of a system's architecture

Bibliography

- Eigenes Script

Module compatibility

- Pflichtbereich Bachelor-Studiengang PO 2016

Prerequisites

- Formal: Successful completion of the module Programming Project 1

Conditions for awarding credit points

- Active and successful participation in exercises
- Written exam (60 min)

Responsible persons

Dr. Jens Bendisposto

Theoretical Computer Science

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	90 hours	210 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

This module provides a basic understanding of the foundations, models, methods, and results of theoretical computer science, in particular introducing into the theory of formal languages and automata, the theory of computability, and the theory of NP-completeness

Formal Languages and Automata

- Foundations (Strings, Languages, and Grammars; the Chomsky Hierarchy)
- Regular Languages (Finite Automata; Regular Expressions; Equation Systems; the Pumping Lemma; Theorem of Myhill and Nerode and Minimal Automata; Closure Properties and Charakterizations of Regular Languages)
- Contextfree Languages (Normal Forms; the Pumping Lemma; Closure Properties of Contextfree Languages; the Algorithm of Cocke, Younger, and Kasami; Push-Down Automata)
- Deterministic Contextfree Languages (Deterministic Push-Down Automata; Application: Syntax Analysis by LL(k)-Parsers)
- Contextsensitive and L_0 Languages (Turingmaschinen; Linear Bounded Automata; Overview)

Computability

- Intuitive Notion of Computability and Church's Thesis
- Turing Computability
- LOOP, WHILE, and GOTO Computability
- Primitive Recursive and Partial Recursive Functions (Primitive Recursive Functions; the Ackermann Function; Total and Partial Recursive Functions; the Main Theorem of Computability Theory)
- Decidability and Enumerability
- Undecidability (Rice's Theorem; Reducibility; Post's Correspondence Problem; Undecidability in the Chomsky Hierarchy; Overview)

NP-Completeness

- Problems in P and NP (Deterministic Polynomial Time; the Satisfiability Problem of Propositional Logic; Nondeterministic Polynomial Time)
- NP-Completeness and Cook's Theorem

Learning Outcomes

After completing the course, students are able to

- classify formal languages in the Chomsky hierarchy,
- transform the considered equivalent automata models into each other into the considered grammars of the respective type,
- provide for a given language a grammar generating it or an automaton of a suitable type (e.g., a finite automaton or a push-down automaton or a linear bounded automaton or a Turing machine) accepting it,
- conversely, determine for a given grammar or automaton the corresponding language,
- give arguments for the inequivalence of considered automata models or grammar types,
- describe how a compiler is built,
- describe the tasks and methods of lexical and syntax analysis,

- discuss the algorithmic decidability of problems,
- give arguments for the undecidability of problems,
- argue that there exist functions that are not computable,
- apply the acquired skills dealing with formal concepts and models and with formal argumentations and proof techniques (such as diagonalization) and
- provide reductions between problems to show their undecidability or NP-completeness.

Bibliography

- Uwe Schöning: Theoretische Informatik kurz gefasst. Spektrum Akademischer Verlag. Heidelberg, 2008. 5. Auflage.
- John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman: Einführung in die Automatentheorie, Formale Sprachen und Komplexitätstheorie. Pearson Studium. München, 2002. 2. Auflage.
- Klaus W. Wagner: Theoretische Informatik. Eine kompakte Einführung. Springer-Verlag. Berlin, Heidelberg, 2003. 2. Auflage.
- Ergänzende Empfehlungen werden in der Lehrveranstaltung genannt.

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Compulsory Area Bachelor study programme PO 2016, 2013
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Module CL6 Bachelor study programme Computer Linguistics

Prerequisites

- Mathematics 1 (or, alternatively, Linear Algebra 1)

Conditions for awarding credit points

- active and successful participation in the exercises
- written exam (exam, usually 90 minutes)

Responsible persons

apl. Prof. Dr. Dorothea Baumeister, Prof. Dr. Michael Leuschel, Prof. Dr. Jörg Rothe

Compulsory Modules in Mathematics

The compulsory modules of mathematics are offered by the Mathematics Institute and described in the module handbook of their bachelor's degree program. Please refer to this especially with regard to prerequisites, contents and learning objectives of the modules.

The following modules make up this area:

- Analysis I
- Analysis II
- Lineare Algebra I
- Angewandte Mathematik (either Numerik I or Stochastik)

It is your decision wether to cover the Angewandte Mathematik module via Numerik I or Stochastik.

Minor modules

The modules in the “minor modules” area depend on the minor subject selected (see Computer Science website: https://www.cs.hhu.de/bachelor/nebenfaecher). The following subjects are available: biology, physics, chemistry, mathematics and psychology (limited to 5 places per academic year; always begins in the winter semester). The modules that can be taken in the respective minor subject are announced by the examination board on the subject’s website. Other minor subjects can be approved by the examination board upon written application, provided that there is a sufficient connection to computer science. The minor subject is usually determined by the Student and Examination Administration Department in the third semester, in any case before the first partial examination in the minor subject is taken. A change of minor subject is permitted upon request, as long as no subject examination in the minor subject has been definitively failed. In the selected minor, 30-40 CP (depending on the respective examination regulations) must be earned, which may be spread over more than three modules, depending on the minor.

You can find information on the individual compulsory elective and specialization modules in the bachelor’s and master’s degree programs on the websites of the relevant chairs and working groups.

Professional Issues

Courses on practical and career orientation are available from the Heinrich Heine University in Düsseldorf.

At least 2 CP must be acquired in courses that teach techniques of scientific work or presentation techniques. These include, among other things:

- Rhetoric seminars/courses
- Seminars in computer science or in the chosen minor
- Courses of the KUBUS program on techniques of scientific work or on presentation techniques

The remaining credit points (maximum 3 CP) can be acquired in one or more of the following types of courses:

- Internships with a high relation to computer science inside or outside the university (but not as part of any course). An internship certificate which shows the amount of time and describes the content of the internship in detail is required for recognition.
- rheinjug events
- Application training

Module Description

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	depends on chosen modules	depends on chosen modules
Components	**Cycle**	**Course of Study**	**Language of instruction**
depends on chosen modules	anytime	Bachelor Computer Science	German

Content

Depending on the selected modules.

Learning Outcomes

- In this module, the students should learn the basics of scientific working techniques and the professional presentation of results.
- The students can continue to practically apply and deepen their theoretical knowledge as part of an industrial internship in the field of computer science. They will also get to know the operational processes and can further improve your social skills.

Bibliography

- Abhängig vom jeweiligen Modul

Module compatibility

- Praxis- und Berufsorientierung

Prerequisites

- determined by the respective module coordinators

Conditions for awarding credit points

- successful participation in the selected events
- The acquisition of credit points depends on the respective regulations for the courses attended.
- In the case of internships, the awarding of credit points depends on the duration of the internship: 1 CP usually corresponds to 28 hours of internship plus around 2 hours of follow-up work.

Responsible persons

Lehrende der jeweiligen Fächer

Courses for elective areas

In order to improve the feasibility of studying the course and to increase the choice of options for the students, modules of different sizes are offered in the bachelor's course in computer science. Modules must be combined in such a way that the total number of credits corresponds to the requirements of the examination regulations. Modules can be freely combined for the elective area. The composition of the major subject must be discussed with the mentor and supervisor of the bachelor thesis in the major.

In principle, the language of instruction is German, but knowledge of English is required as a prerequisite for the study program. This is documented in the respective study regulations: „It is pointed out that the computer science study program requires knowledge of the English language.“ Literature for some events is often (forced) in English. Lecture slides and scripts are also sometimes written in English.

The written theses must be written in German or English. (PO BSc § 16 para. 1: “The bachelor thesis can be written in German or English.”)

In the bachelor's program in computer science, 30 CP can be earned as additional work. These can also come from the courses offered in the master's program in computer science. For the modules offered, please refer to the module handbook for the Master's course, which you can find on the computer science website. Participation in these modules is only permitted if the courses Programming, Computer Architecture, Algorithms and Data Structures and Theoretical Computer Science have been passed.

A module for the bachelor's degree can only be used for the individual supplement in the master's degree in computer science if it has not already been used for a previous bachelor's degree and is approved for this (this is usually not the case after September 1st, 2022).

Algorithms in Bioinformatics

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

The module teaches introductory concepts of algorithmic bioinformatics. The students will deepen and apply them in theoretical and practical exercises.

- Biological background
- Exhaustive search: DNA motifs
- Greedy algorithms: genome rearrangements
- Dynamic Programming: sequence alignments
- Graph algorithms: assembly
- Combinatorial pattern matching and suffix trees
- Clustering
- Phylogenetic trees and molecular evolution
- Hidden Markov Models: CpG islands

Learning Outcomes

After completing the course, students are able to

- apply the discussed algorithmic design principles, prove correctness and analyze running times,
- differentiate between tractable and intractable algorithmic problems and explain the consequences,
- distinguish different classes of algorithms,
- explain and apply classic bioinformatics algorithms,
- implement many of these algorithms in the programming language Python, and
- select an appropriate algorithm to solve a given task.

Bibliography

- Neil C. Jones, Pavel A. Pevzner: An Introduction to Bioinformatics Algorithms. The MIT Press, 2004
- Phillip Compeau, Pavel A. Pevzner : Bioinformatics Algorithms, An Active Learning Approach, Vol. I and II, Active Learning Publishers, 2015

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of modules Programmierung and Algorithmen und Datenstrukturen and (Mathematik für Informatik 1 or Lineare Algebra I or Analysis I)

Conditions for awarding credit points

- active participation in the tutorial classes
- successful completion of the exercises (50%)
- final exam (written, usually 90 min)

Responsible persons

Prof. Dr. Gunnar W. Klau

Computational Geometry

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Content

This module imparts basic knowledge from the following areas:

- convex hull
- plane sweep algorithms / segment intersection problems
- distance problems
- geometric divide and conquer / closest point pair
- Voronoi diagrams / Delaunay triangulations / nearest neighbor queries
- triangulation of polygons / monotonic polygons
- area queries / ham-sandwich theorem
- rectangle queries / area trees
- point/line duality/line arrangement
- smallest enclosing circles / randomized algorithms

Learning Outcomes

After completing the course, students are able to

- deal with data structures for computing the convex hull, a voronoi diagram, a Delauney triangulation or a line arrangement
- apply plane-sweep techniques, divide-and-conquer methods and randomized approaches to solve geometric problems
- determine lower bounds for the complexity of geometric problems on sets of points in the Euclidean plane

Bibliography

- de Berg et al: Computational Geometry, Algorithms and Applications. Springer-Verlag. Berlin, 2000. 2. rev. ed.
- Preparata, Shamos: Computational Geometry, an Introduction. Springer-Verlag. New York, 1985.
- Edelsbrunner: Algorithms in Combinatorial Geometry, EATCS Monographs in Computer Science 10. Springer-Verlag, 1987.
- Matousek: Lectures on Discrete Geometry, Graduate Texts in Mathematics, 212. Springer-Verlag, New York, 2002.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of modules Algorithms and Datastructures and Mathematics for Computer Science 1 (or Linear Algebra I and Calculus I)

Conditions for awarding credit points

- active participation in the exercises,
- submission of selected homework,
- written exam (usually 90 minutes) or oral exam at the end of the semester

Responsible persons

Prof Dr. Egon Wanke

Computational Complexity Theory

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Content

This module imparts basic knowledge from the following areas:

- introduction to computationalcomplexity (nondeterminism, NP-completeness, …)
- SAT, 3-SAT, Clique, IS, VC, 3-DM, Dominating Set, 3-Partition
- pseudopolynomial algorithms, knapsack, partition problems
- approximation algorithms
- circuit complexity
- parameterized algorithms, FPT
- Savich's theorem, Immerman's and Szelepcsényi's theorem
- randomized algorithms

Learning Outcomes

After completing the course, students are able to

- apply non-deterministic calculation models,
- estimate the complexity of algorithmic problems,
- apply basic techniques for approximating solutions,
- develop pseudo-polynomial solution and
- analyze parameterized questions.

Bibliography

- Michael R. Garey and David S. Johnson: Computers & Intractability: A Guide to the Theory of NP-Completeness. Freeman. 1979
- Christos H. Papadimitriou: Computational Complexity. Addison-Wesley. 2008

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of modules Algorithms and Datastructures and Mathematics for Computer Science 1 (or Linear Algebra I and Calculus I)

Conditions for awarding credit points

- active participation in the exercises,
- submission of selected homework,
- written exam (usually 90 minutes) or oral exam at the end of the semester

Responsible persons

Prof Dr. Egon Wanke

Applied Algorithmics

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

> "In theory, there is no difference between theory and practice. In practice, there is."

Algorithms are the foundation of every computer program. Traditionally, the focus of algorithm design is on the theory of efficient algorithms and their worst-case analysis. In this module we focus on practically efficient algorithms for provably hard optimization problems. The aim is to not (totally) give up the principles of optimality. Topics are:

- Fundamental aspects of algorithms and complexity
- Complete enumeration and dynamic Programmierung
- Branch and Bound
- Approximation algorithms
- Heuristics and metaheuristics
- (Integer) linear programming
- Fixed-parameter tractable algorithms
- Modelling with satisfiability (SAT)

Learning Outcomes

After completing the course, students are able to

- apply the presented applied algorithmic design techniques to new problems,
- solve problems practically efficient and implement, test and evaluate these solutions.

Bibliography

- Steven Skiena, "The Algorithm Design Manual", Second Edition, Springer, 2008

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of modules Programming, Algorithms and Datastructes, Theoretical Computer Science as well as contents of one of Mathematics for Computer Science 1 or Lineare Algebra I or Calculus I

Conditions for awarding credit points

- active participation in the tutorial classes
- successful completion of the exercises (50%)
- final exam (written, usually 90 minutes)

Responsible persons

Prof. Dr. Gunnar W. Klau

Bachelor's Seminar: Introduction to blockchain technology

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Tutorial (2 HPW) Seminar (2 HPW)	irregular	Bachelor Computer Science	German

Remarks

- Students who do not study according to PO 2021 will be given lower priority in the allocation of places.

Content

- Rules for constructive feedback
- Basics of blockchain technology (hash functions, signatures, proof of work, …)
- Decentralized currencies (e.g. Bitcoin)
- Further content will be determined in consultation with the students based on the seminar presentations (e.g. attacks, defense strategies, various protocols)

Learning Outcomes

After completing the course, students are able to

- Familiarize themself with a given topic area independently,
- Describe this topic in writing and present it in a lecture,
- Use feedback of fellow students about their work to revise it,
- Give constructive feedback on other drafts and presentations,
- Explain the basics of blockchain technology,
- Name and justify different areas of application of the blockchain, and
- Summarize the contents of the individual seminar presentations.

Bibliography

Die Literatur wird zu Beginn des Seminars bekannt gegeben und richtet sich nach den Themen der einzelnen Vorträge.

Module compatibility

- Seminar Bachelor-Studiengang PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016

Prerequisites

- Formal: Successful completion of the scientific work module

Conditions for awarding credit points

- At least sufficient essay(this will be made available to all participants of the course)
- Participation in the peer review process of the elaborations

- At least sufficient seminar presentation
- Pass an oral exam

Responsible persons

Janine Golov

Bachelor's-Seminar: Programming Languages

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Seminar (2 HPW) Tutorial (2 HPW)	every winter semester	Bachelor Computer Science	German

Remarks

- Students who do not study according to PO 2021 will be given lower priority in the allocation of places.

Content

The seminar Programming Languages conveys general knowledge about programming languages as well as their typical properties and concepts. The seminar consists on the one hand of a series of lectures in which participants introduce a language and on the other hand an exercise in which this language is used by programming tasks. Writing an elaboration on lecture language prepares participants for the bachelor thesis.

- General knowledge of programming languages (Properties, differences, applications/areas of application, syntax and semantics)
- Properties and concepts of programming languages
- give a lecture on an independently developed topic, as well as write an elaboration on it
- Writing a written elaboration on a programming language

Learning Outcomes

After completing the course, students are able to

- describe and explain properties of the discussed programming languages,
- independently create a lecture and an elaboration on a given topic, and
- give this lecture.

Bibliography

Bruce A. Tate. 2010: Seven Languages in Seven Weeks: A Pragmatic Guide to Learning Programming Languages. Pragmatic Bookshelf. 1st edition

Module compatibility

- Bereich Bachelor-Studiengang PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016

Prerequisites

- Contentual: Contents of module Programming

Conditions for awarding credit points

- Create an elaboration
- Giving a lecture

- Processing of programming tasks

Responsible persons

Dr. John Witulski

Bachelor's-Seminar: Introduction to Artificial Intelligence

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	90 hours	60 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture/Seminar (2 HPW) Tutorial (2 HPW)	every winter semester	Bachelor Computer Science und Master DSAI	German / English (DSAI)

Remarks

- Students who do not study computer science according to PO 2021 will be given lower priority in the allocation of places.

Content

The lecture covers the following topics:

- Rules for constructive feedback
- Basics of artificial intelligence, from classical symbolic AI to modern techniques like deep learning
- Further content will be determined in consultation with the students based on the seminar presentations

Learning Outcomes

After completing the course, students are able to

- familiarize themselves with a given topic area independently,
- describe this topic in writing and present it in a lecture,
- use feedback of fellow students about their work to revise it,
- give constructive feedback on other drafts and presentations,
- asses the capabilities and problem-specific applicability of different AI approaches (e.g., expert systems, SVM, decision trees, random forests, CNNs, …)
- explain the goals and techniques of various subfields of AI,
- name and explain the limits of current AI approaches (e.g., bias, explainability, safety, …) and judge the relevance for concrete practical questions.

Bibliography

- Bratko: Prolog Programming for Artificial Intelligence. Addison Wesley
- Russel, Norvig: Artificial Intelligence: A Modern Approach. Prentice Hall
- Szeliski: Computer Vision: Algorithms and Applications. Springer
- Goodfellow, Bengio, Courville: Deep Learning. MIT Press

Module compatibility

- Bereich Seminar Bachelor-Studiengang PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Wahlbereich im Master-Studiengang Artificial Intelligence and Data Science

Prerequisites

- Formal: Successful completion of the scientific work module

Conditions for awarding credit points

- seminar presentation
- successful creation of a written paper
- written exam

Responsible persons

Prof. Dr. Michael Leuschel

Operating Systems and System Programming

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	90 hours	210 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

- C programming
- Libraries, binder, and loader
- Processes and threads
- Scheduling: different algorithms (multi level, feedback, realtime); case studies: Linux, Windows, …
- Synchronisation: mutex, sempaphores, deadlocks, lock-free synchronisation
- Main memory: heap, stack, memory management, garbage collection
- Virtual memory: one and several levels, inverted page tables, page reclamation strategies, case study: Linux memory management
- Secondary storage: HDD/SDD characteristics, partitions, memory management
- File systems: FAT, UNIX, ext4, NTFS (including journaling)
- Inter process communication: signal, message queue, pipes, shared memory, sockets
- Input/output: interrupts, I/O software, Linux kernel modules and drivers
- Security: access control, hardware protection mechanisms, buffer exploits, shellcode, meltdown, address space layout randomization, kernel page table isolation
- Architectures: monolith, micro kernel, virtual machines, client/server

Learning Outcomes

After completing the course, students are able to

- describe the interaction of operating system kernel, drivers and hardware
- compare operating system concepts
- develop system programs in C based on the UNIX system interface
- design and implement basic parallel programs using threads and synchronzation primitives
- explain operating system security issues and possible solutions based on hardware protection of the 86 architecture and operating system concepts

Bibliography

- A. Tanenbaum: „Modern Operating Systems“, 4. Aufl., Prentice Hall, 2014.
- W. Stallings, „Operating Systems: Internals and Design Principles”, Prentice Hall, 9. Aufl., 2017.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contents of the modules: Programming and Computer Architecture

Conditions for awarding credit points

- Active and successful participation in the exercises
- Written exam (90 min, 50% programming, 50% paper work)

Responsible persons

Prof. Dr. Michael Schöttner

Competitive Programming

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Lab (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

In Competitive Programming the goal is to solve programming problems quickly under stringent runtime and space constraints for the test data. This module teaches basic techniques to recognise common types of contest problems and solve them in C++. The goal is to prepare students for participation at national and international programming contests (e.g., ICPC). There will be weekly training sessions as well as several live contests.

Topics of the module:

- Basic algorithms (binary search, sorting, dynamic programming, backtracking, prefix sums)
- Greedy algorithms
- Graph problems (DFS, shortest paths, spanning trees, union-find)
- Geometric problems (convex hull, intersections)
- Number theoretical problems (prime number decomposition, divisibility problems)
- Word problems (subwords)
- segment trees

Learning Outcomes

After completing the course, students are able to

- use data structures and algorithms of the C++ Standard Library,
- estimate the running time of computer programs on given input data sets,
- identify common patterns in easy program contest problems and
- design and implement efficient programs for easy programming contest problems.

Bibliography

- Ahmed Shamsul Arefin: Art of Programming Contest. Second Edition, Special Online Edition for UVa Online Judge Users
- Steven Halim, Felix Halim, and Suhendry Effendy: Competitive Programming 4: The Lower Bound of Programming Contest in the 2020s, Books 1+2. Lulu Press. 2020
- Nicolai Josuttis: The C++ Standard Library - A Tutorial and Reference. Addison Wesley Longman. 1999. 2nd edition
- Steven Skiena: The Algorithm Design Manual. Springer. 2008. 2nd edition.
- Steven Skiena and Miguel Revilla: Programming Challenges - The Programming Contest Training Manual. Springer Verlag. 2003.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics

- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Good programming skills in C++, Java, or Python and basic knowledge of efficient data structures and algorithms.

Conditions for awarding credit points

The final grade is computed from

- weekly training sessions (50%)
- performance at live contests (50%)

Responsible persons

apl. Prof. Dr. Rudolf Fleischer.

Compiler Construction

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	90 hours	60 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW) Praktische Tutorial (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

In this lecutre we discuss the basics of compiler building.

- Lexical analysis (regular expressions and finite automata)
- Syntax analysis (context-free grammars and deterministic parsing)
- Semantic Analysis
- Code Generation
- Using tools to automatically generate compilers

Learning Outcomes

After completing the course, students are able to:

- explain how programming languages are translated and implemented
- explain and customize syntax descriptions of a programming language (In particular, the students should be able to determine whether the description is suitable for automated processing in a compiler), and
- develop a parser or compiler for a new programming language theirself.

Bibliography

- Andrew W. Appel: Modern Compiler Implementation in Java. Cambridge University Press. 2nd Edition
- Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman: Compilers: Principles, Techniques, and Tools. Addison Wesley. 2nd Edition

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of module Programming

Conditions for awarding credit points

- Successful processing of compulsory exercises
- Successful development of your own compiler

- Passing the exam

Responsible persons

Prof. Dr. Michael Leuschel, Dr. John Witulski

Data Science

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	90 hours	210 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

Data Science is the application of statistical methods and methods of machine learning to data of any kind, using a computer to model systems and predict behaviour.

- probability theory (discrete and continuous distributions, Bayes' theorem, independence, normal distribution, multivariate distributions, transformations)
- machine learning (data and models, estimator theory, classification, regression, dimensionality reduction, cluster analysis)
- python packages for data science (numpy, matplotlib, pandas, scikit-learn)

Learning Outcomes

After completing the course, students are able to

- perform exploratory data analysis using the programming language Python,
- transform data for further processing,
- ask research questions about a dataset,
- select parts of data for further processing,
- merge suitable datasets,
- describe insights from data using statistical terminology and visualize insights from data using the packages pandas and matplotlib,
- consider data protection problems about processing data,
- consider ethical questions about processing data,
- construct statistical models from data using the package scikit-learn,
- evaluate the performance of models and
- use statistical models for classification and prediction of future data.

Bibliography

- Georgii: Stochastik: Einführung In Die Wahrscheinlichkeitstheorie Und Statistik. De Gruyter. Berlin, 2015. 5. Auflage
- VanderPlas: Python Data Science Handbook. O'Reilly Media, Inc. Sebastopol, 2016. 1st edition
- Grus: Data Science from Scratch: First Principles with Python. O'Reilly UK Ltd. UK, 2019. 2nd edition
- Deisenroth et. al.: Mathematics for Machine Learning. CUP. Cambridge, 2020. 1st edition

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of modules Mathematics for Computer Science 1-3

Conditions for awarding credit points

- active and successful participation in the theoretical and practical exercises
- passing the written exam (usually 90 minutes)

Responsible persons

Dr. Konrad Völkel, Prof. Dr. Milica Gasic

Data Science 2

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every second winter semester	Bachelor Computer Science	German

Content

Data Science is the application of statistical methods and methods of machine learning to data of any kind, using a computer to model systems and predict behaviour. In the module Data Science 2 the basics from Data Science are deepened close to applications, in particular with respect to the topics:

- big data
- data parallel processing (algorithms and software packages)

Learning Outcomes

After completing the course, students are able to

- analyze algorithms for big data and/or data streams with respect to runtime and communication complexity,
- in particular in the topic areas Near Neighbor Search in High Dimensional Data, Locality Sensitive Hashing (LSH), Dimensionality reduction, Recommendation Systems, Clustering (variants of k-means), Link Analysis (PageRank), Web Advertising,
- decide which kinds and which size of data needs parallelization,
- decide which techniques of parallel processing are feasible for given problems,
- in particular the map-reduce technique,
- and to implement these together with using common software packages.

Bibliography

- Leskovec, Rajaraman and Ullman: Mining of Massive Datasets. Cambridge University Press. 2020. 3rd Edition (mmds.org)

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016

Prerequisites

- Contentual: Data Science or Machine Learning

Conditions for awarding credit points

- active and successful participation in the theoretical and practical exercises
- passing the written exam (usually 90 minutes)

Responsible persons

Dr. Konrad Völkel

Databases: An Introduction

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

In this module basic theoretic and practical knowledge about databases is presented. The different phases in database design as well as SQL queries and the required formal background are to the fore.

- architektures of database systems; 3-level schema architecture
- tasks of a database system
- data(base) models; in particular the relational model
- conceptual and logical database design (phases model, ER model, transformation into the relational model); normalisation (functional dependencies, normal forms, synthesis algorithm)
- query languages for relational databases and their foundations; relational algebra, relational calculus, SQL
- concurrency control: problems of multiuser operation, transactions and schedules, serializability

Learning Outcomes

After completing the course, students are able to

- name and explain the tasks of database systems,
- design databases on their own,
- formulate database queries using different query languages (in particular, the relational algebra and SQL) and
- test whether schedules for sets of transactions are serializible.

Bibliography

- G. Saake, K.-U. Sattler, A. Heuer: Datenbanken – Konzepte und Sprachen. mitp Verlag. 2018. 6. Auflage
- Kemper, A. Eickler: Datenbanksysteme – Eine Einführung. Oldenbourg Verlag. 2015. 10. Auflage
- G. Vossen: Datenmodelle, Datenbanksprachen und Datenbankmanagementsysteme. Oldenbourg Verlag. 2008.
- R. Elmasri, S.B. Navath: Fundamentals of Database Systems. Pearson. 2016. 7th edition
- H. Garcia-Molina, J.D. Ullman, J. Widom: Database Systems: The Complete Book. Pearson. 2009. 2nd edition

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Module CL6 Bachelor study programme Computer Linguistics

Prerequisites

- None

Conditions for awarding credit points

- active and successful participation in the exercises (usually with weekly homeworks)
- written exam (usually 60 minutes)

Responsible persons

Prof. Dr. Stefan Conrad, Dr. Leonie Selbach

Databases: Further Concepts

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Lab (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

Building upon the basics of database design and query languages for relational databases, this module covers further concepts of databases and database systems. Beside the fundamentals the practical realization (database design and database application programming) is the main focus of this module. For this, the following topics are covered in the lecture:

- database definition (using SQL)
- database application programming and using databases in the Web
- data protection and data security (SQL injection; views and user privileges in databases)
- trigger
- aspects of im plementiong database systems (physical storage; index structures)
- algorithms for query operators and query optimization

Learning Outcomes

After completing the course, students are able to

- develop database applications (including database design, database definition and application programming),
- respect basic aspects of data protection and data security during the development of database applications, and
- explain and assess essential implementation concepts for storing data as well as elementary data structures and algorithms for query processing.

Bibliography

- G. Saake, K.-U. Sattler, A. Heuer: Datenbanken – Konzepte und Sprachen. mitp Verlag. 2018. TODO Auflage
- G. Saake, K.-U. Sattler, A. Heuer: Datenbanken – Implementierungstechniken. mitp Verlag. 2011. 3. Auflage
- Kemper, A. Eickler: Datenbanksysteme – Eine Einführung. Oldenbourg Verlag. 2015. 10. Auflage
- R. Elmasri, S.B. Navathe: Fundamentals of Database Systems. Pearson. 2016. 7th edition
- H. Garcia-Molina, J.D. Ullman, J. Widom: Database Systems: The Complete Book. Pearson. 2009. 2nd edition

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields

Prerequisites

- Contents of module Databases: An Introduction

Conditions for awarding credit points

- active and successful partizipation in the exercises (pratical tasks based on each other)
- final homework (database design and programming project)

Responsible persons

Prof. Dr. Stefan Conrad

Introduction to Functional Programming

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

This module consists of concepts widespread in the functional programming paradigm. We use Clojure - a modern Lisp running on the JVM - as a programming langauge. The following topics are covered:

- Clojure's syntax and Clojure programming
- immutable data structures and laziness
- the epochal time model
- simplicity and Clojure philosophy
- polymorphismus a la carte
- homoiconicity and macros.

Learning Outcomes

After completing the course, students are able to

- name characteristics of functional programming and to compare them with traditional, imperative programming,
- evaluate for which applications functional programming is beneficial,
- create and test functional programs,
- explain and apply the concepts listed above (Content).

Bibliography

- Moseley, Marks: Out of the tarpit. Online
- Fogus, Houser: The Joy of Clojure. Manning. 2014. 2nd edition.
- Emerick, Carper, Grand: Programming Clojure. O'Reilly. 2012. 1st edition.
- Rathore, Avila: Clojure in Action. Manning. 2016. Second 2nd.
- Higginbotham: Clojure for the Brave and True. No Starch Press. 2015. 1st edition.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of modules Programming, Programming Project 1

Conditions for awarding credit points

- Understanding of the material
- Depending on the number of participants:
- preferably written exam (exam, usually 90 minutes)
- oral exam (usually 30-45 minutes)

Responsible persons

Philipp Körner, Dr. Jens Bendisposto, Prof. Dr. Michael Leuschel

Introduction to Scientific Computer Science

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

With the help of selected examples, the module describes the application of computer science and statistics to solve various problems in biology, physics and chemistry. Essential parts of the underlying algorithms are implemented in the Python programming language:

- Fast Fourier-Transform to reduce runtime in multiple Alignment
- Pebble-Game-Algorithm for rigidity analysis of biomolecules
- Motiv-search in DNA sequences using Gibbs-Sampling
- Dynamic programming in pairwise sequence comparison
- Clustering algorithms for sequence and expression data: Neighbor-joining, Markov-clustering-algorithm, k-means, expectation maximization
- Lateral gene transfer or phylogenetic artifact? Statistical test to test congruence of trees with non-identical leaf sets without a reliable reference tree.
- Rooting phylogenetic trees using mean-ancestor-deviation
- Recursion and the problem of independent-phylogenetic-contrasts

Learning Outcomes

After completing the course, students are able to

- describe the underlying scientific background and the associated problems of the presented methods,
- apply the algorithms presented for problem solving to example data,
- critically compare different possible solutions to a problem, and
- independently implement the learned methods in the programming language Python.

Bibliography

- wird im Rahmen der Vorlesung vorgestellt

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields

Prerequisites

- None

Conditions for awarding credit points

- Minimum of 50 percent points from exercises
- Passing the written examination (usually 90 minutes)

Responsible persons

Dr. Mayo Röttger, Prof. Dr. Martin Lercher

Algorithms for Graphs 1

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

This module covers basic knowledge from the following areas.

- searching in graphs
- topological sorting
- connectivity problems
- shortest path problems
- minimal spanning trees
- network flow problems
- matching problems

Learning Outcomes

After completing the course, students are able to

- describe and explain the discussed graph algorithms,
- allocate the discussed algorithms to different problem settings and apply them adequately,
- analyze the discussed algorithms with respect to their running time and correctness and
- design and analyze easy new graph algorithms.

Bibliography

- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein: Algorithmen – Eine Einführung. De Gruyter Oldenbourg. 2017. 4. Auflage.
- Ravindra Ahuja, Thomas Magnanti, James B. Orlin: Network Flows: Theory, Algorithms and Applications. Prentice Hall. Englewood Cliffs, 1993. 1. Ausgabe

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of modules Programming, Algorithms and Data Structures and Mathematics for Computer Science 1

Conditions for awarding credit points

- actively participate in tutorials

- hand in exercises
- final written exam (usually 90 min.) or oral exam at the end of the semester)

Responsible persons

Prof. Dr. Melanie Schmidt, Dr. Daniel Schmidt

Foundations of Computer Networks

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

The module “Foundations of Computer Networks“ targets students who want to understand the technical design principles of computer networks. Fundamental problems arising in the design of computer networks and how they have been addressed in the context of the internet are being discussed. The module aims to teach a solid foundation of computer network technology as well as practical capabilities.

- Introduction and overview
- Application Layer (World Wide Web/HTTP, File Transfer/FTP, E-Mail/SMTP, Domain Name System/DNS, socket programming with UDP and TCP)
- Transport Layer (addressing, reliable data transfer/RDT, congestion control, UDP, TCP)
- Network Layer (virtual connections and datagram networks, design principles of routers, adressing/DHCP and NAT, internet protocol/IP, ICMP, link state routing and distance vector routing, RIP, OSPF, BGP)
- Link Layer (frames, error detection and correction, media access in local networks, addressing/ARP, Ethernet, hubs, switches, PPP, IP over ATM und MPLS, applications)

Learning Outcomes

After completing the course, students are able to

- explain the layered structure of the internet and apply the important protocols in all layers,
- compute the performance of the protocols in simple networks,
- explain address mechanisms in the different layers and discuss their interactions,
- analyse simple local networks and design and implement suitable network architectures and protocols for given applications,
- apply fundamental methods to optimise the performance of complex networks and
- distinguish various types of transmission errors as well as suggest ways to discover and fix transmission errors in the various network layers.

Bibliography

- James F. Kurose and Keith W. Ross: Computer Networking – A Top-Down Approach Featuring the Internet. Pearson, 2020. 8th Edition

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- None

Conditions for awarding credit points

- Homework exercises (50% needed for exam admission)
- Written final exam (typically 90 minutes)

Responsible persons

apl. Prof. Dr. Rudolf Fleischer, Prof. Dr. Martin Mauve

Foundations of Distributed Systems

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

- Architectures: client server, p2p, cloud, edge, fog
- Communication: sockets, remote procedure call, distributed shared memory
- Naming service: Domain Name System
- Coordination: real und logical clocks, distributed mutal exclusion and transactions
- Replication and consistency models
- Group communication: reliable und atomic multicast
- Fault tolerance: failure models, checkpointing
- Peer-to-peer-systems: gnutella and chord
- Security: basics of encryption and authentication
- Consensus: bully, paxos, blockchain

Learning Outcomes

After completing the course, students are able to

- identify properties and challenges in distributed systems
- explain and use different communication concepts
- compare and apply coordination algorithms in distributed systems
- select appropiate consistency models for given application scenarios
- describe group communication solutions
- compare failure models and checkpointing strategies
- explain algorithms for data search in peer-to-peer systems
- describe basic security aspects
- explain consensus algorithms

Bibliography

- G. Coulouris et.al., „Distributed Systems: Concepts and Design“, Addison-Wesley, 5. Aufl. 2011
- A. Tanenbaum and M. van Steen: „Distributed Systems: Principles and Paradigms”, 3. Auflage, Prentice Hall, 2013.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contents of the module Programming

Conditions for awarding credit points

- Passing the exam

Responsible persons

Prof. Dr. Michael Schöttner

Introduction to Logic Programming

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	90 hours	60 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW) Lab (2 HPW)	every winter semester	Bachelor Computer Science	English German

Content

Logic programming is quite different from the classical imperative approach to programming. Logic programming is a declarative programming language where one declares the properties of a solution rather than providing an algorithm to solve a problem step-by-step. The lecture will give students a new perspective on programming which will also prove useful even if software is still developed in classical programming languages like C or Java.

The lecture covers the following topics:

- Propositional logic, predicate logic
- Resolution
- Programming with Horn clauses
- Practical foundations of Prolog
- Search algorithms and AI with Prolog
- Basics of constraint programming

Learning Outcomes

After completing the course, students are able to

- apply the logical foundations of Prolog to perform derivations in propositional and predicate logic
- use Prolog data structures to encode data
- develop smaller Prolog programs independently, as seen in the exercises
- compare advantages and drawbacks of various search algorithms and implement them in Prolog
- solve smaller symbolic AI tasks in Prolog

Bibliography

- Nilsson, Maluszynski: Logic, Programming and Prolog (2nd edition). (eBook)
- Blackburn, Bos, Striegnitz: Learn Prolog Now!. College Publications.

Die Folien sind auf Englisch und Deutsch verfügbar.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Wahlbereich im Master-Studiengang Artificial Intelligence and Data Science

Prerequisites

- Contentual: Programming skills.

Conditions for awarding credit points

- Active and successful participation in the exercises
- Successful participation at the final exam

Responsible persons

Prof. Dr. Michael Leuschel

Introduction to Modelling metabolic networks

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	English

Content

- Introduction to the statistical programming language R
- Repetition of basic linear algebra
- Basic properties and reconstruction of stoichiometric matrices
- Topology and fundamental subspaces of the stoichiometric matrix
- Elementary flux modes
- Properties of the solution space
- Flux balance analysis
- Flux variability, flux coupling
- Modeling of gene knockouts
- Flux balance analysis with molecular crowding
- Resource balance analysis

Learning Outcomes

After completing the course, students are able to

- summarize important constraint-based modeling techniques and apply them to metabolic networks,
- describe biological systems from possible biochemical reactions,
- formulate and solve linear optimization problems using the R programming language and
- consider metabolic modules as a system and simulate their behavior under different conditions.

Bibliography

- Bernhard Ø. Palsson: Systems Biology: Properties of Reconstructed Networks. Cambridge University Press. 2015.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of modules Programmierung und Mathematik für Informatik (or Lineare Algebra I)

Conditions for awarding credit points

- active participation in the exercises
- successful completion of the exercises (50%)

- passing final exam (usually written)

Responsible persons

Prof. Dr. Martin Lercher

Collective Decisions

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Content

This module deals with different methods of collective decision-making under preferences. Such methods are applied in several areas of artificial intelligence, for example in the interaction of autonomous agents. Central contents of this lecture are different methods with their axiomatic and algorithmic properties from the three areas: voting, participatory budgeting, and resource allocation.

- Voting (voting rules, properties, impossibility results)
- Participatory budgeting (preferences, aggregation rules, properties)
- Resource allocation (preferences, social welfare, allocations)

Learning Outcomes

After completing the course, students are able to

- describe different methods for collective decision making and apply them in concrete situations,
- define new decision-making methods and analyze them with respect to their properties, and
- give justified recommendations for the usage of particular decision-making methods under given conditions.

Bibliography

- Jörg Rothe, Dorothea Baumeister, Claudia Lindner und Irene Rothe: Einführung in Computational Social Choice. Individuelle Strategien und kollektive Entscheidungen beim Spielen, Wählen und Teilen. Spektrum, Akademischer Verlag. 2011.
- Felix Brandt, Vincent Conitzer, Ulle Endriss, Jerome Lang, and Ariel Procaccia (eds.): Handbook of Computational Social Choice. Cambridge University Press. 2015.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of module Theoretical Computer Science

Conditions for awarding credit points

- active and successful participation in the theoretical exercise courses
- writen test (exam, usually 90 minutes)

Responsible persons

apl. Prof. Dr. Dorothea Baumeister

Combinatorial Algorithms for Clustering Problems

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Content

This course is a lecture on advanced algorithms and deals with clustering algorithms. In this lecture, we discuss combinatorial clustering problems and related algorithms. In particular, we discuss:

- hierarchical clustering
- algorithms for the k-center problem
- algorithms for the k-supplier problem
- similarity-based clustering procedures

Learning Outcomes

After completing the course, students are able to

- describe and develop combinatorical arguments,
- apply the discussed clustering algorithms,
- discuss and evaluate hierarchical clustering methods, and
- analyse clustering precedures.

Bibliography

- Originalliteratur und eigenes Skript

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of module Mathematics for Computer Science 1

Conditions for awarding credit points

- actively participate in tutorials
- successfully work on exercises
- final exam (written or oral exam)

Responsible persons

Prof. Dr. Melanie Schmidt

Cryptocomplexity 1

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	90 hours	210 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Content

This module provides a basic understanding of the foundations of cryptology, introduces some important cryptosystems, and discusses their security that often relies on the computational complexity of suitable problems. Therefore, the foundations of complexity theory are introduced as well, in particular with the goal to understand methods for proving and applying lower bounds with respect to the complexity measures time and space. A special focus is given to understanding the close connections between these two fields.

Introduction to Cryptology

- Some Classical Cryptosystems and their Cryptoanalysis (Substitution and Permutation Chiffre, Affin Linear Block Chiffre, Block and Stream Chiffres)
- Perfect Secrecy (Shannon's Theorem and Vernam's One-Time Pad, Entropy and Key Equivocation)
- RSA, Primality Tests and the Factorization Problem (the Public-Key Cryptosystem RSA, Digital Signatures with RSA, Security of RSA)

Introduction to Complexity Theory

- Foundations (Complexity Measures and Classes, Compression and Speed-Up Theorems, Hierarchy Theorems)
- Between L and PSPACE (simplee Inclusions, Complexity-Bounded Many-One-Reductions, Complete Problems in NL, NP-Complete Problems)

Learning Outcomes

After completing the course, students are able to

- classify symmetric cryptosystems with respect to their properties,h
- evaluate the security of block chiffres and other classical cryptosystems,
- reason why certain cryposystems possess which properties,
- explain the idea of public-key cryptography,
- explain the basic goals and definitions of complexity theory,
- describe and evaluate the complexity of natural problems,
- unassistedly design reductions between problems to show their lower bounds and provide a corresponding proof of correctness.

Bibliography

- Jörg Rothe: Komplexitätstheorie und Kryptologie. Eine Einführung in Kryptokomplexität. eXamen.Press. Springer-Verlag, Heidelberg, 2008. 1. Auflage.
- Jörg Rothe: Complexity Theory and Cryptology. An Introduction to Crypto-complexity. EATCS Texts in Theoretical Computer Science, Springer-Verlag. Heidelberg, 2005. 1. Auflage.
- Ergänzende Empfehlungen werden in der Lehrveranstaltung genannt.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- None

Conditions for awarding credit points

- active and successful participation in the exercises
- written exam (exam, usually 90 minutes)

Responsible persons

Prof. Dr. Jörg Rothe

Matching

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Content

This module deals with different methods of matching under preferences. Such methods are applied in different areas, examples include school admissions, allocation of residents to hospitals, or the allocation of houses to students. Contents of this lecture are different methods with their axiomatic and algorithmic properties.

- matching in graphs
- bipartite matching with two-sided preferences (stable marriage problem)
- non-bipartite matching with preferences (stable roommates problem)
- bipartite matching with one-sided preferences (house allocation problem)

Learning Outcomes

After completing the course, students are able to

- conduct matching algorithms in different situations,
- identify challenges in practical matching problems,
- develop matching algorithms for specific areas and investigate their properties,
- transfer known matching algorithms to new areas,
- compare different matching algorithms, and
- recommend methods for specific areas of matching.

Bibliography

- Dan Gusfield and Robert W. Irving: The Stable Marriage Problem, Structure and Algorithms. MIT Press. 1989.
- David F. Manlove: Algorithmics of Matching under Preferences. World Scientific Publishing Company. 2013.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of module Theoretical Computer Science

Conditions for awarding credit points

- active and successful participation in the theoretical exercise courses
- writen test (exam, usually 90 minutes)

Responsible persons

apl. Prof. Dr. Dorothea Baumeister

Preference Aggregation by Voting: Algorithmics and Complexity

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	90 hours	210 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Content

This module provides a basic understanding of the foundations, models, methods, and results of the young and rapidly evolving interdisciplinary field *Computational Social Choice* that has applications in the areas *Artificial Intelligence* and *Multiagent Systems*. In particular, properties of voting systems are investigated and the related decision problems (winner determination, manipulation, electoral control, bribery, etc.) are studied in terms of their algorithmical solvability and complexity.

Foundations of Social Choice Theory

- Elections
- Voting Systems (Scoring Protocols like Plurality, Veto, and Borda; Condorcet; Copeland; Maximin; Dodgson; Young; Bucklin; Fallback; etc.)
- Properties of Voting Systems
- Some Voting Paradoxa and Impossibility Results

Algorithmics and Complexity of Voting Problems

- Winner Determination
- Possible and Necessary Winners
- Manipulation
- Electoral Control
- Bribery

Learning Outcomes

After completing the course, students are able to

- describe the most common voting systems (scoring protocols like Plurality, Veto, and Borda; Condorcet; Copeland; Maximin; Dodgson; Young; Bucklin; Fallback; etc.) and their properties,
- discuss the reasonability of axiomatic properties of voting systems,
- determine the winners of the most common voting systems for any given preference profile,
- give examples of successful manipulation, control, and bribery actions for the most common voting systems and describe the related decision problems,
- argue why which of these problems can either be solved by efficient algorithms or are hard to solve.

Bibliography

- Jörg Rothe (ed.): Economics and Computation: An Introduction to Algorithmic Game Theory, Computational Social Choice, and Fair Division. Springer Texts in Business and Economics, Springer-Verlag. Heidelberg, 2015. 1. Auflage.
- Jörg Rothe, Dorothea Baumeister, Claudia Lindner und Irene Rothe: Einführung in Computational Social Choice. Individuelle Strategien und kollektive Entscheidungen beim Spielen, Wählen und Teilen. Spektrum Akademischer Verlag (Springer). Heidelberg, 2011. 1. Auflage.
- Ergänzende Empfehlungen werden in der Lehrveranstaltung genannt.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- None

Conditions for awarding credit points

- active and successful participation in the exercises
- written exam (exam, usually 90 minutes)

Responsible persons

Prof. Dr. Jörg Rothe

Randomized Algorithms und Analysis

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	90 hours	210 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Content

This course is a lecture on advanced algorithms and deals with randomization as a method of algorithm design and analysis. The goal is to obtain efficient algorithms via randomized decisions that run faster than their deterministic variants and at the same time provide precise results with high probability.

- models of randomized algorithms (Las-Vegas and Monte-Carlo algorithms)
- running time and accuracy analysis
- randomized approximation algorithms (e.g., for SAT and graph problems)
- methods for probability amplification
- randomized design paradigms (e.g., probabilistic method, fingerprints, hashing)
- randomization in data analysis

Learning Outcomes

After completing the course, students are able to

- describe technical terms and basic methods of randomization in algorithms,
- match algorithms to different design paradigms and apply them examplarily,
- analyse, evaluate and compare properties such as running time and accuracy and
- design first randomized algorithms according to adequate design methods.

Bibliography

- Juraj Hromkovi¬: Randomisierte Algorithmen: Methoden zum Entwurf von zufallsgesteuerten Systemen für Einsteiger. Teubner-Verlag. Wiesbaden, 2004. 1. Ausgabe
- Michael Mitzenmacher and Eli Upfal: Probability and Computing. Cambridge University Press. Cambridge, 2017. 2nd Edition.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of modules Algorithms and Data Structures, Theoretical Computer Science and Mathematics for Computer Science 3

Conditions for awarding credit points

- actively participate in tutorials
- successfully work on exercises
- final exam (written or oral exam)

Responsible persons

Prof. Dr. Melanie Schmidt, Dr. Anja Rey

Statistical Data Analysis

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

The module is based on a course and related book by Prof. Gianluca Bon-tempi at the Université Libre de Bruxelles on the statistical foundations of machine learning. Contents are in detail:

- The R programming language for statistical computing
- Descriptive statistics
- Basics of probability calculation
- Classic parametric estimation and testing
- Non-parametric estimation and testing
- Statistical Learning
- Linear approaches
- Non-linear approaches
- Dimensionality reduction

Learning Outcomes

After completing the course, students are able to

- summarize basic concepts of statistical data analysis (e.g. probability distributions, parametric and non-parametric hypothesis tests, parameter estimators, conditional probabilities, permutation tests),
- devise parametric and non-parametric tests for statistical hypotheses,
- formulate statistical models with multi-dimensional predictors,
- plan and carry out statistical data analyses with R, and
- create meaningful graphical representations of data using ggplot2.

Bibliography

- Gianluca Bontempi. Handbook statistical foundations of machine learning. Self-published, Brüssel, 2021. 2nd edition.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- None

Conditions for awarding credit points

- active participation in the exercises
- successful completion of the exercises (50%)
- final exam (usually written)

Responsible persons

Prof. Dr. Martin Lercher

From Circuits to Software

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	90 hours	210 hours
Components	**Häufigkeit des Angebots**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

This course offers a journey through different areas of computer science, in order to convey an overall picture of the basic functioning of computers. Starting from the NAND gate, all important hardware components of a computer are reproduced in a simulation, ALU, CPU, RAM, etc.

For the CPU and the associated machine language developed during the lecture, an assembly language, a virtual machine and a programming language as well as the associated compiler will be developed step by step. With all these tools, a simple operating system and application programs are finally developed.

The lecture is accompanied by exercises. The independent practical application of what has been learned will be the focus of the event. In the exercises, the students develop the various components of modern computers presented in the lecture.

Learning Outcomes

After completing the course, students are able to

- to explain and evaluate the basic principles of all discussed levels and
- independently develop new functionalities on these levels.

Bibliography

- Noam Nisan, Shimon Schocken: The Elements of Computing Systems: Building a Modern Computer from First Principles. MIT Press. Cambridge, 2008. 1st Edition

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016

Prerequisites

- Contentual: Contents of module Programming

Conditions for awarding credit points

- successfully completing the exercises
- passing the final exam

Responsible persons

Dr. John Witulski

Bachelor Thesis

Bachelor Thesis

ECTS credits	Total hours	Contact hours	Self-study hours
15 CP	450 hours	-	-
Components	**Cycle**	**Course of Study**	**Language of instruction**
-	anytime	Bachelor Computer Science	German English

Content

The content of the bachelor thesis lies in the selected major. The bachelor thesis must be written in German or English and presented in an oral presentation.

Learning Outcomes

With the written thesis, the students should prove that they are able to:

- within a specified period (of 3 months)
- work independently on a topic and
- to present it appropriately.

Bibliography

- in Absprache mit den Betreuungspersonen

Module compatibility

- Bachelor-Arbeit

Prerequisites

- In order to register for the bachelor's thesis, at least 120 of the credit points to be acquired as part of the bachelor's degree must be proven. The topic of the bachelor thesis is assigned from the field of the selected major. For this purpose, all modules in the main subject should usually have been successfully completed.

Conditions for awarding credit points

- Successful processing of the topic and presentation in a draft submitted on time (Bachelor thesis)
- Presentation in an oral presentation with discussion

Responsible persons

Dozierende der Informatik sowie der als Schwerpunktfach wählbaren mathematisch-naturwissenschaftlichen Fächer