Ingegneria Informatica Magistrale (2021)

72935 - Operations Research M

PDF, ADOC.

Learning outcomes

Mathematical models, linear programming, simplex algorithm, duality. Integer linear programming, cutting planes, branch-and-bound. Complexity. Dynamic programming. Discrete simulation.

Course contents

Course contents

Prerequisites:

It is required that the student understands spoken Italian in order to follow the lectures.

MODULE 1 (Mathematical Optimization): Silvano Martello

1. The scientific method: systems, models, methodologies
2. Mathematical Programming

2.1 Optimization problems

2.2 Convex sets and functions

2.3 Convex programming

1. Linear programming

3.1 General, canonical and standard forms

3.2 Bases e basic solutions

3.3 Convex polytopes

1. Simplex algorithm

4.1 Moving among basic solutions

4.2 Tableau and pivoting

4.3 Optimality criterion

4.4 Simplex algorithm

4.5 Two-phase method

4.6 Geometrical aspects

1. Duality

5.1 Dual of a linear programming problem

5.2 Duality properties

5.3 Farkas' lemma

5.4 Complementary slackness

5.5 Dual simplex algorithm

5.6 Sensitivity analysis

1. Integer linear programming

6.1 Unimodularity

6.2 Cutting-plane algorithms and Gomory cuts

6.3 Branch-and-bound algorithm

6.4 Exploration strategies

6.5 0-1 knapsack problem

6.6 Branch-and-cut algorithms

6.7 Software and freeware

1. Complexity theory

7.1 Recognition version

7.2 P and NP classes

7.3 NP-complete problems

7.4 Dynamic programming

7.5 Strong NP-completeness

MODULE 2 (Discrete Simulation): Andrea Lodi

Introduction to discrete simulation

Static and dynamic description of a system

Temporary and permanent entities

Events and event notices.

Exercises on modeling realistic systems.

72937 - Computer Architecture M

PDF, ADOC.

Learning outcomes

Focus is on quantitative aspects of computer architecture.

Expected outcome is understanding of:

instruction level parallelism and latency tolerance

memory hierarchy

Uniform Memory Access (UMA) architectures

protection and task management

architecture impact on power and performance at processor and system level

Within the course the student will practice abstraction as the most effective method to handle the complexity of modern computer architectures. The final test verifies the ability to apply methods and concepts offered by the course.

Course contents

Processor Architecture:

Instruction set architecture for multitasking protected systems (Intel IA32 architecture)

Instruction level parallelism; dependencies; hazards; hazards avoidance

superscalar architectures; superpipelining and underpipelining; non blocking architectures; out of order execution (Tomasulo approach); speculative execution;

Introduction to simultaneous multithreading, logical processors, multicore architectures and virtualisation

Memory Hierarchy

Goals, reference model and performance parameters

memory hierarchy in the Intel IA32 architecture: segmentation, virtual memory, main memory and caches

address mapping, write policies, replacement policies; implementation techniques and performance analysis

the MESI protocol for memory coherence in shared memory multiprocessor architecture (UMA architectures)

System Architecture

Interleaved memory access in multibyte data bus systems

Uniform Memory Access (UMA) architectures, and introduction to NUMA multicore based architectures

Bus hierarchy and bus protocols

Multimaster systems with DMA based I/O: hardware/software power/performance perspective

72940 - Computational Models And Languages M

PDF, ADOC.

Learning outcomes

At the end of this course unit, conceived according to a multi-paradigm and multi-language constructive approach, the student has a deep knowledge on the fundamental concepts of programming languages and related computational models, knows the foundations of computability, knows and is able to apply the basic interpreter and compiler techniques, and possesses the basic concepts of functional programming and is able to apply such concepts in typical practical situations.

More precisely, the student will know the main formal methods for language definition in terms of syntax and semantics, for both programming languages and specification languages, and will be able to apply the main LL and LR techniques for language evaluation and recognition for interpreters and compilers, including the use of the most common tools. Students will also be able to define reasonably simple languages understanding their properties, implementing the corresponding interpreters, and evaluating the pros and cons of different choices/computational paradigms in application design.

Course contents

The course aims to provide a rational view over the fundamental concepts of programming languages, relating them to the different computational models and to the problem of language translation and recognition: solid foundations are coupled to a strong experimental approach.

Contents:

Computability and Turing machine (5hrs)

Formal description of programming languages: grammars and Chomsky classification. Relationship between grammars and language interpreters/translators: lexical analysis, top-down and bottom-up techniques for the syntactical analysis of regular languages and context-free languages. Overview on methods for the formal description of the semantic aspects of a language. (18hrs)

Structure and organisation of interpreters/compilers, and their run-time support: examples in Java. Parser generator tools. (14hrs)

Introduction to Model-Driven approaches: the Xtext case. (3 hrs)

Iterative vs. recursive computational models, tail recursion optimisation, basic concepts of functional programming, closures, models for function evaluation, intro to the basics of Lambda calculus (9hrs)

Javascript as an example of (dynamic) functional language with a prototype-based object model (6hrs)

Scala and Kotlin as notable examples of blended programming languages on the Java platform. (9hrs)

72942 - Information Security M

PDF, ADOC.

Learning outcomes

Knowledge and engineering skills related to the design, development and deployment of algorithms and protocols for securing systems and networks.

Course contents

The aim of the course is to provide an in-depth study of the models, systems and mechanisms for securing processing systems with both a theoretical and a practical focus.

Suggested background: to gain more from the course it is important to have clear the concepts and tools provided by the computer networks, operating systems and computer security laboratory courses.

The course contents are divided into five macro-areas:

1. Modern cryptography applied

Insights and pitfalls in using PRNG, stream ciphers and block ciphers, cryptographically secure hash functions, asymmetric ciphers

Examples of attacks based on the incorrect use of ciphers and correct methods of use

Examples of cryptographic applications in some scenarios (wireless networks, cloud, IoT, ..)

Symmetric and asymmetric cryptographic key management models and systems (Key distribution

center, PKI, PGP)

1. Authentication Models and Systems

Recalls on authentication systems and principles of designing secure authentication protocols

Single Sign-on authentication models with related examples of protocols / systems (Kerberos, ...)

Federated authentication models with relative examples of protocols / systems (Oauth, OpenID, SAML, ..)

1. Access control models and systems

Identity-based models with related examples of protocols / systems

Role-based models with related examples of protocols / systems

Attribute- and context-based models with related examples of protocols / systems

1. Automated security management models and systems: paradigm based on policy-based management
2. Blockchain technologies

Principles of operation

Hints of operation of the Bitcoin and Ethereum platforms

72943 - Digital Systems M

PDF, ADOC.

Learning outcomes

The corse provides an introduction to modern software methodologies for the design of embedded systems. Application contexts and projects are mostly concerned with image processing and deep-learning networks applied to computer vision problems.

Course contents

Design methodologies for embedded systems with particular emphasis on computer vision applications and deep-learning.

72947 - Operating Systems M

PDF, ADOC.

Learning outcomes

Knowledges of main design aspects concerning the organization of concurrent systems. Models for sincronization and communication between processes/threads. Methods for analysis and synthesis of concurrent systems.

Course contents

1.System protection and security

models, policies and mechanisms

multilevel security

Reference Monitor and trusted systems

2.Virtualization

hardware virtualization: goals and solutions

virtual machine monitor implementation

Case study analysis: xen hypervisor

Virtualization and cloud computing

3.Concurrent programming

preliminaries

non sequential processes.

forms of interaction between concurrent processes

architectures and languages for concurrent programming

4.Shared memory model

mutual exclusion

semaphores

monitors

conditions

Use of concurrent languages in the shared memory model. The pthread library for concurrent programming.

5.Message passing model

preliminaries

channels and communication primitives

send and receive

guarded commands

Rendez-vous and RPC

Use of concurrent languages in the message passing model: go, ada.

6.Shared memory kernel

Implementation of thread management/synchronization in a mono-processor kernel

Implementation of thread management/synchronization in a multi-processor kernel: SMP, loosely-coupled kernels.

1. Distributed Systems

distributed programming: centralized and decentralized algorithms.Scalability, fault tolerance.

algorithms for the synchronization of distributed processes: logical clocks, distributed mutual exclusion, election algorithms.

1. Parallel Programming

Parallel architectures, HPC systems.

Architetture per il calcolo parallelo. Sistemi HPC.

Parallel programming models: shared memory and distributed memory.

Parallel software development: MPI and OpenMP libraries.

Outlines on CUDA programming.

72938 - Foundations Of Artificial Intelligence T

PDF, ADOC.

Learning outcomes

Introduction to main principles and methods in Artificial

Intelligence. Artificial Intelligence based methodologies and

techniques for solving problems with particular emphasis on

knowledge-based systems and computational logic techniques. Desing

and implementation of some practical systems based on procedural and declarative programming languages.

Course contents

The course introduces the student to the principles and methods used to solve Artificial Intelligence methods, with a particular attention to knowledge-based systems and computational logic approaches. In particular, the Prolog programming language is introduced as a tool for implementing Artificial Intelligence systems. Moreover, seminars on specific Artificial Intelligence topics are planned. This course is preparatory for the course of Intelligent Systems.

Prerequisites: attending the course requires a medium knowledge of a high level programming language, in order to successfully understand case studies and applications presented during the lessons. Regarding the course contents, no prerequisites are required: the student will be gradually introduced to the fundamental notions of the Artificial Intelligence, and no assumption about previous knowledge is made.

Contents:

Introduction to Artificial Intelligence: brief history of AI, main application fields, introduction to knowledge-based systems and architectural organization.

Problem solving in AI: representation through the notion of state, forward e backward reasoning, solving as a search and search strategies (informed and non). Games, constraint satisfaction problems, and planning problems.

Knowledge Representation: First Order

Predicate Logic, Production Rules Systems, Knowledge-based systems,

Some hints about formal ontologies.

Languages for Artficial Intelligence. Prolog: from logic to

logic programming, Prolog programs as solvers, desing and

development of simple Prolog programs, few notes about

meta-predicates and meta-interpreters.

84531 - Infrastructures For Cloud Computing and Big Data M

PDF, ADOC.

Learning outcomes

The class tends to enhance the capacity of orientating in the process of defining the strategies for a distributed system and applying them in different related applications.

The students face the principles and the main problems of distributed large systems and are exposed to some standard and widely solutions, by following the class and via individual work. At the end, students are expected to be able to know the properties of most diffused middleware and the evolutions one can expect from that technology, by mastering the properties for designing a real application: most well spread strategies are presented and discussed.

Course contents

The course covers several topics central in modern global data and processing infrastructures, such as Data Centers, MultiCloud Systems, Federation of resources, etc. typically supporting Industry 5.0 and Smart city applications:

Advanced models for large distributed & cloud systems, from C/S to message exchange.

Replication, group and many-to-many communication, and systems for QoS

Middleware for development and management of large distributed & cloud systems

Infrastructures for global data storage and processing

The class explores the following topics:

Advanced models for large distributed & cloud systems

Class Starting: general information and presentation of the Class (use cases)

Goals, Basics, and Models: classifications, C/S vs. Message exchange, service and cloud models, parallelization models

Middleware & Cloud Models: definitions, categories, basic organization, and patterns for large distributed and cloud systems, Cloud internals design.

Replication, group and many-to-many communication, and systems for QoS

Different consistency degrees and impact on service properties (BASE and CAP)

Replication: models, strategies and protocols

Communication and groups: models, protocols and algorithms

Systems and protocols for QoS

Multicast and MOM middleware

Middleware for large distributed & cloud systems

CORBA: middleware and operating environment

MOM: examples of very thin environments

OpenStack: an example of a widely-diffused cloud IaaS

Novel infrastructures for global data storage and processing

Global data storage: solutions for non-traditional NoSQL data memorization (Cassandra and MongoDB)

Global data processing: batching and streaming based big data processing (Map-Reduce, Spark, and Storm and S4)

Main properties for effective design projects

72939 - Software Systems Engineering M

PDF, ADOC.

Learning outcomes

Detailed knowledge about languages, models and technologies for the analysis, the development, the documentation, the deployment and the maintenance of (distributed) software systems.

Course contents

At the end of the course, the student:

is able to set-up cooperative software production processes, based on agile (SCRUM) development that exploit also executable models exprressed using custom meta-models;

is able to design and develop software systems with related testing plans, in an incremental and evolutive way, by starting from the problem and from the application domain rather than from the

implementation technology, also by using executable models of requirement and problem analysis;

is able to critically evaluate the continuos evolution of software technologies, both as regars the computational aspects and the software development process, by operationally acquiring knowledge on languages, methodologies and tools such as Kotlin, Gradle, SCRUM, SpringBoot, DevOps, Docker, etc.

is able to understand the role of the different styles of software architectures (layers, client-server, pipeline, microkernel, service-based, event-driven, space-based, microservices) and how to select the most appropriate architectural style for each different sub-system;

is able to face the analysis, the design and development of distributed, heterogeneous proactive-reactive applications (together with related development platforms and run-time supports) with particular reference to computational models based on message-passing and event-driven paradigms;

is able to realize message-based interactions among distributed software components by using high-level logical models and implementations based on different protocols (TCP, UDP, HTTP, CoAP, MQTT);

is able to understand how it is possible to design and build software development environments able to automatic code generation (Sofware Factories in ecosystems like Eclipse and IntelliJ) based on Model Driven Software Development (MDSD) and on Domain Specific Languages (DSL);

is able to develop application able to combine high-level aspects (with particular reference to AI) with low-level aspects related to Internet of Things (IOT) devices, in the context of both virtual and real environments, built by using low-cost computers like RaspberryPi and Arduino;

is able to apply the concepts, the devices and the tools developed in the course for the design and development of a final application that exploits one or more IOT devices - in particular Differental Drive Robots (DDR) with sensors - that can operate in relatively autonomous way in different virtual or real enviroments, without modifying the software that does express the business logic of the problem.

72945 - Real-Time Systems M

PDF, ADOC.

Learning outcomes

The course aims to provide a rational view of the main issues involved in the design of real-time systems, emphasizing peculiar principles, implementation strategies, supporting environments and performance evaluation methods.

Course contents

An introduction to real-time systems: peculiar features and typical application domains. Hard vs. soft real-time tasks. Temporal parameters and reference models for periodic, sporadic and aperiodic tasks. Real-time scheduling approaches: the clock-driven approach and the priority-

driven approach. Algorithms for scheduling hard real-time periodic/sporadic tasks: RM, DM, EDF and LSTF. Server-based strategies for optimizing the response time of soft real-time aperiodic tasks: polling server, deferrable server, priority exchange server, sporadic server, constant utilization server, total bandwidth server. Shared resources access protocols: priority inheritance protocol, priority-ceiling protocol, immediate priority-ceiling protocol, stack resource policy. Pre-run-time schedulability analysis of systems that schedule tasks using static or dynamic priorities: processor-utilization analysis, response-time analysis, processor-demand analysis. Exemplification of theoretical and methodological issues with reference to design patterns typical of the industrial automation application domain.

72957 - Design Activity joined to Computer Architecture M

PDF, ADOC.

Learning outcomes

Apply the knowledge acquired in the Computer Architecture M course to independently carry out an in-depth activity on a topic agreed with the lecturer in charge of the course.

Course contents

Projects of power management of systems and processors for high-performance computing systems.

Projects of RISC-V based computing architectures and low level software layers (Linux, RTOS, NUTTX, FreeRTOS, ROS, Compilers) for Cyber Physical System (CPS) applications.

Neuromorphic / braininspired computing architecture projects and their use in Cyber Physical System (CPS) applications.

72975 - Software Systems Engineering Project Work M

PDF, ADOC.

Learning outcomes

In this module we apply the abilities achieved in the course 34791 - INGEGNERIA DEI SISTEMI SOFTWARE LM for the devolpment of some specific argument or project

Course contents

Development of a software project

72980 - Project Work For Computational Models And Languages M

PDF, ADOC.

Learning outcomes

The aim of this activity is to apply the knowledge, metholodologies and tools learned in the main module to a practical project, either proposed by the student and validated by the teacher, or proposed by the teacher ad accepted by the student.

Course contents

The project can deal with any aspect in the Languages and Computational Models area, including the development of multi-paradigm applications or the contribution to research activity in the area. The activity must state explicitly the initial requirements, provide an in-depth analysys of the problem, discuss the project and the architecture of the proposed solution (including the evaluation of possible alternatives and the motivation(s) for choosing the specific one), up to the implementation choices, testing methodologies and techniques. A final comprehensive demo is expected and adequately evaluated.

72998 - Project Work For Information Security M

PDF, ADOC.

Learning outcomes

Apply the knowledge acquired in the Information Security M course to independently carry out an in-depth activity on a topic agreed with the lecturer in charge of the course.

Course contents

The contents of the project activity concern topics related to the field of IT security including security of mobile systems, IoT systems and cloud / edge

73000 - Project Work For Digital Systems M

PDF, ADOC.

Learning outcomes

Project development with platform and software tools introduced in Digital System M

Course contents

Development of a project with an embedded system

97261 - ATTIVITA' PROGETTUALE DI TECNOLOGIE E SISTEMI PER LA GESTIONE DI DATABASES E BIG DATA M

PDF, ADOC.

Learning outcomes

Autonomous exploitation of knowledge obtained in the course "Technology and systems for the management of Databases and Big Data M" in a technological application on a theme in agreement with the teacher.

Course contents

The project can deal with any practical application about the contents presented in the course "Technology and Systems for the Management of Databases and Big Data M. In general, the project can be also about any topic that can be referred to the field of data management. In general, the subject of the activity must have a practical flavor, i.e. the activity must focus on the practical application of principles, methods and techniques acquired within the course. The subject of the project activity must be discussed and agreed upon with the teacher before the starting of the activity itself.

72968 - Project Work For Foundations Of Artificial Intelligence M

PDF, ADOC.

Learning outcomes

The aim of this activity is to apply the knowledge and tools

acquired in the associated course of "Fondamenti di Intelligenza

Artificiale M"/"Fundamentals of Artificial Intelligence M" to a

practical project. The topic of the project is proposed by the

student and must be agreed with the teacher.

Course contents

The project can deal with any practical application about the contents presented in the course "Fondamenti di Intelligenza Artificiale M"/"Fundamentals of Artificial Intelligence M". In general, the project can be also about any topic that can be referred to the AI. In general, the subject of the activity must have a practical flavour, i.e. the activity must focus on the practical application of principles, methods and techniques acquired within the course. The subject of the project activity must be discussed and agreed upon with the teacher before the starting of the activity itself.

72994 - Project Work For Operations Research

PDF, ADOC.

Learning outcomes

Application of the methodologies acquired in the course "Operations Research M" to the development of an autonomous activity on a theme agreed with the teacher.

Course contents

Implementation and experimental testing of algorithms for combinatorial optimization problems and for real-life applications.

73004 - Real Time Systems Design Activities M

PDF, ADOC.

Learning outcomes

Apply the knowledge acquired in the Real-Time Systems M course for the autonomous development of an in-depth activity on a topic agreed with the lecturer.

Course contents

Typical, although not exclusive, topics related to this activity are:

1) development and implementation of algorithms for scheduling hard/soft real-time tasks in a single- or multi-processor platform;

2) design and implementation of the infrastructure necessary to support the application of sophisticated access protocols to shared resources;

3) development of suitable tools for a priori verification of the schedulability of real-time applications comprising an arbitrary number of tasks and shared resources.

The experimental evaluation of the performance actually achievable with the developed prototype solution represents an essential aspect of the design activity.

84533 - Project Work on Infrastructures for Cloud Computing and Big Data M

PDF, ADOC.

Learning outcomes

The project is assigned on an individual base, with a negotiation with the student to identify an area at the state of the art within the field of large middleware systems. At the end of the project, the students have acquired a deep knowledge of the area and also capacity in understanding the global solution strategies. They have to apply the notions and skills acquired in the related class toward a design of a typical (limited) system with some predefined requirements and specific significant technology. The entire execution life cycle is stressed and some performance tests must be pursued.

Course contents

The choice of the topics to organize the project around is part of the final evaluation, the same as the technical repost given in at the end and the presentation of it (slides for project presentation).

The areas to be chosen can be (other areas can be negotiated):

Middleware and service management

Cloud Computing

Scalable global Computing

Federated Data Centers

IoT Cloud

Sustainable Infrastructures for Smart Environments

Smart city Infrastructures