Informatica Magistrale (2023)

90720 - Usability & User Experience Design

PDF, ADOC.

Learning outcomes

At the end of the course the student is able to design, implement and evaluate software systems with respect to the dimensions of practicality, experience, affection, meaning and value that they may have on the target audience. Characteristics such as ease of use, usefulness and efficiency are fundamental for the positive evaluation of the user experience of the system. The student will be able to focus the functional analysis of the software on the characteristics and needs of the target audience, will be able to drive the development process so as to guarantee a constant connection between the technical and implementation features and the expectation of the audience, and will be able to evaluate whether and according to which metrics a software satisfies these expectations.

Course contents

Teaching will be mostly in English (some explanations will be repeated in Italian if needed and when asked). The exam will be allowed in both Italian and English. The course content is divided in distinct parts:

Background The evolution of the discipline from Human Computer Interaction to User Experience Design. A description of its scope: the human, the computer, and their interaction.

Usability analysis and design A systematic discussion of the techniques and standards for the management of the process of user experience design, with particular attention to the phases of usability analysis (with and without the participation of users) and the user- and goal-oriented usability design methodologies.

Guidelines, patterns and methods for usability design A discussion, with historical aspects, of the framework on which the concrete aspects of usability design is based. we will also give strong attention to the problem of usability for web applications and mobile apps.

Applications of User Experience Design to complex systems (e.g. AI Systems): short module reserved to students in Artificial Intelligence, optional for the other students.

72671 - Complements of Programming Languages

PDF, ADOC.

Learning outcomes

Nowadays the software development requires fast and sophisticated code

transformation and analysis tools. For example, Java code is verified by the Virtual Machine before execution to check basic correctness properties about the memory and the locks that are used. Facebook, before releasing its mobile apps, always submits them

to a tool that finds bugs without running the code. The applications of

performance-critical systems and asset-management systems would be impossible to build

and evolve without compilers that derive correct and optimized machine code from high-level source code.

The objective of this course is to discuss modern code transformation and analysis

techniques and illustrate their implementation. For this reason, the course will

refer to a framework, ANTLR, now widely used in academia and industry to build all sorts of languages, tools, and frameworks. For example, Twitter search uses ANTLR for query parsing, with over 2 billion queries a day.

At the end of the course, the student knows the fundamentals of code transformation

and static analysis. He is able to apply the theory by extending a small, yet expressive and powerful language, by means of the ANTLR framework.

Course contents

Introduction to code transformation and analysis. The ANTLR framework and the corresponding Lexical and Syntactical Analysis. Semantic analysis: symbol table, type checking, static analysis of properties about resource and asset consumptions. Intermediate code generation for standard and advanced features of programming languages. Virtual Machines and Interpreters.

87469 - Decision Making with Constraint Programming

PDF, ADOC.

Learning outcomes

This course covers the fundamental methods in artificial intelligence to model and solve combinatorial problems requiring to take (optimal) decisions in the presence of many complex constraints. Such problems appear often in diverse domains of science, business and industry. The inherent difficulty in solving such practically crucial problems has lead to the development of Constraint Programming (CP), an intelligent decision-support technology. A growing number of companies and research institutions worldwide successfully apply CP technology and contribute to

its advancement. Skills in this area are therefore in high demand in the job market. The course combines theoretical foundations with practical modelling and solving of realistic problems. At the end of the course, the student has an understanding of the advanced modelling techniques and efficient solution methods, and possesses the necessary skills for modelling in a CP language and solving with a CP solver.

Course contents

Prerequisites: Basic computer science courses such as discrete mathematics, logic, algorithms and data structures, programming. Prior knowledge on artificial intelligence is not necessary.

Course topics

Overview and successful applications of Constraint Programming (CP)

Modeling techniques

Local consistency notions and constraint propagation

Global constraints

Constructive tree search algorithms, search and propagation, branching heuristics, randomization and search restarts

Optimization problems

Constraint-based scheduling

Heuristic search methods

Hybrid CP and heuristic search methods

Advanced topics and hot research topics in CP

81676 - Digital Forensic

PDF, ADOC.

Learning outcomes

At the end of the course the students will know the main topics of digital forensics. Moreover, they have used several basic tools to manage some common scenarios: single device (computer, tablet, smartphone) and several kinds of file, networking (wireless and wired), e-mail and social media. The students will know the importance of the chain of custody and also the main procedures to acquire, conserve and analyze the data. The students will know both the importance of the final report and the conceptual instruments for its appropriate drafting

Course contents

Understanding the Digital Forensics Profession and Investigations

The Investigator’s Office and Laboratory

Data Acquisition

Processing Crime and Incident Scenes

Working with Windows and CLI Systems

Current Digital Forensics Tools

Linux and Macintosh File Systems

Recovering Graphics Files

Digital Forensics Analysis and Validation

Virtual Machine Forensics, Live Acquisitions, and Network Forensics

E-mail and Social Media Investigations

Mobile Device Forensics

Cloud Forensics

Report Writing for High-Tech Investigations

Expert Testimony in Digital Investigations

Ethics for the Expert Witness

77803 - Service - Oriented Software Engineering

PDF, ADOC.

Learning outcomes

At the end of the course, the student knows the design and implementation of complex software systems using an approach based on

the abstractions of service and process. The student is able to design, model and implement

modern virtualized software architectures based on enterprise SOA and/or microservices; the student is also able to model and support the enactment of business processes integrating the services that compose them.

Course contents

Service-oriented architectures (SOAs) are used to build large software systems that operate across multiple organizations (as is the case in the enterprise environment) but also as a basic structure to

build flexible and extremely scalable applications (as in the case of microservices).

Designing this class of systems and reasoning about their properties requires the use of appropriate abstractions and modeling techniques.

In this course we will see how service and process abstractions can interoperate to describe complex distributed systems and we will see modeling techniques based on these abstractions that help design them.

We will learn how to model processes (using BPMN), services (using UML) and how to model the interaction between processes and services and between different services through choreographies.

We will see how to exploit these techniques to design applications adhering to the service-oriented architectural style both across-enterprises and through microservices (also by adopting specific patterns).

We will also see in practice the protocols and the technologies that can be used for their implementation; in particular for the technologies to support web services we will see SOAP / WSDL and the RESTful style; for microservices we will see the use of containers (eg Docker), microservice orchestrators (eg Kubernetes), service meshes (eg Linkerd) and serialization technologies (eg Protobuf). As far as microservices are concerned, we will try in particular to understand the strengths and weaknesses of this class of applications and we will understand how to build systems that are both scalable and reliable.

To facilitate the prompt realization of examples of some of the technologies related to web services as they are introduced, we will use the Jolie language which allows rapid prototyping of SOA solutions.

Below is a list of the main topics of the course:

Enterprise software systems, architecture and modeling

Business Process Management e BPMN

SOA/Coreografie

Web services (SOAP/WSLD/RESTful/gRPC)

Microservices properties, design, patterns and implementation

Virtualized infrastructures supporting SOAs: containers (Docker), distributed containers (Kubernetes)

Jolie

37760 - Systems Simulation

PDF, ADOC.

Learning outcomes

At the end of the course students will have acquired methods and tools to design, implement and validate simulation models for the performance analysis and assessment of computer and communication systems and for the analysis of social systems. Students will be able to design, implement and validate simulation models for the analysis and assessment of complex systems.

Course contents

1. classification of systems and models;
2. analytical models: queueing systems and queueing networks;
3. design of simulation experiments;
4. random number generation and random variate generation;
5. discrete events simulazione techniques;
6. design and implementation of simulators and simulation tools;
7. input and output data analysis of simulation models;
8. verification, validation and testing of simulation models;
9. introduction to parallel and distributed simulation;
10. agent based simulation;
11. simulation and machine learning;
12. digital twin;
13. analysis and evaluation of complex systems.
14. Introduction to Virtual Reality.

90749 - Computing Education

PDF, ADOC.

Learning outcomes

The course aims to provide theoretical knowledge, techniques and tools helpful in teaching computer science. At the end of the course, the student is familiar with the main pedagogical and didactical approaches for teaching computer science at different school levels. Students can organise and teach computer science courses, compare and choose different methodologies to generate teaching materials, and evaluate learning.

Course contents

Learning objectives of this course include:

- knowledge of some historical, epistemological and ethical aspects of Computer Science as a scientific discipline and of the motivations underlying the necessity of its teaching;

- understanding of pedagogical aspects and learning theories in the context of computer science teaching;
- knowledge of multiple CS-specific teaching approaches;
- knowledge of the main cognitive difficulties in learning CS (with particular focus on programming), and knowledge of possible strategies to adopt to overcome them;
- ability to formulate and manage learning paths consistent with national standards and curricula related to Computer Science in schools of all levels;
- planning ability to organize laboratories, classroom equipment; ability to integrate students' devices as useful tools for learning.

Topics covered in the course include:

The scientific vision of Computer Science.

Computational Thinking.

Constructivism and constructionism applied to CS teaching.

Top-down and bottom-up approaches in CS teaching.

Teaching of Programming, Algorithms and Data Structures.

Pedagogical implications in the choice of programming languages.

Teaching of technological aspects: computer architecture, operating systems, networks.

CS teaching methods:

- unplugged
- dramatization/visualization
- code reading exercises
- debugging of others’s code
- program execution visualization
- making/tinkering
- repositories as software museums

Difficulties and misconceptions in learning to program.

Creation of learning paths

- in primary schools
- in lower secondary schools
- in scientific lyceums - applied sciences
- in specific courses of technical institutes
- in other upper secondary schools

Assessment methodologies of computer programs.

Importance, planning, and management of computer laboratories.

- Use of BYOD in lessons.

Software licenses: libre (free and open source) and proprietary software: implications in education

Affective and motivational aspects in computer science learning.

66870 - Concurrent Models and Systems

PDF, ADOC.

Learning outcomes

At the end of the course, the student will learn the basic ingredients of concurrency theory, their models and verification systems on such models. (S)He will be able to analyze simple concurrent programs with automatic or semi-automatic tools.

Course contents

- Introduction to concurrency and to the problem of the correctness of reactive systems design.
- Labeled transition systems
- Behavioral equivalences: traces, simulation, bisimulation f(strong and weak), properties.
- The language CCS: syntax and SOS semantics.
- Subclasses of CCS: finite processes, finite-state processes, regular processes, BPP, finite-net processes, finitary CCS.
- Turing completeness of finitary CCS; undecidability of behavioral equivalences of finitary CCS.
- Value-passing CCS.
- Algebraic properties, behavioral congruences and axiomatizations.
- Espressivieness of CCS: encodability of additional operators (internal choice, hiding, sequential composition)
- The problem of muti-way synchronization: Multi-CCS; case study: dining philosophers.
- Petri nets: definition, equivalences, decidable properties, expressiveness.
- Languages for representing Petri nets. Distributed computability.
- Fixpoint theory, least and greates fixpoints, strong bisimilarity as a greatest fixpoint.
- Hennessy-Milner Logic (HML), estention with recursively defined formulae, modal mu-calculus.
- Analysis and verification tools for CCS and HML: Concurrency Workbench (CWB).
- Modeling, analysis and verification of some mutual exclusion algorithms with CWB.

84401 - Context-Aware Systems

PDF, ADOC.

Learning outcomes

At the end of the course, the student is able to design, deploy and evaluate ubiquitous systems and mobile applications able to adapt their behaviors to the context characteristics and to the current location/activity of the user. At the end of the course, the student: -knows the fundamental concepts of context-aware computing, and the main techniques for the localization of users/devices and the human activity recognition; -knows the fundamental models of context-data representation and managing; - knows the main middleware and software architectures in order to deploy adaptive and ubiquitous applications and services

Course contents

The course addresses the design and deployment of ubiquitous and context-aware services and applications, made possible by the pervasive diffusion on the market of devices able to sense the environment and to analyze the sensed data. The course program is structured in two main parts. The first part illustrates the definition of "context" and context-aware systems, focusing on the design and implementation of location-aware and activity-aware systems. Special focus will be given to the spatial data management, by illustrating the main technologies for indoor/outdoor positioning, mapping APIs, geo-data storage and location intelligence.The second part will present the lifecycle of distributed context-aware applications by focusing on mobile edge computing systems where the allocation of workloads take into account contextual data, such as the user position. To this aim, we will introduce edge-computing technologies and frameworks for software containerization (e.g., Docker), task orchestration (e.g., Docker Swarm and Kubernates), and workload/service migration based on users' mobility. In the following, we provide a brief summary of the course program:

Introduction and definition of context and context-awareness

Use case of context-aware systems

Location-aware systems

Location-based services (LBS)

Positioning technologies

Mapping APIs

Spatial database

Location intelligence

Privacy in LBS

Activity-aware systems

Human Activity Recognition (HAR): enabling technologies

Supervised AI/ML techniques for HAR systems

Mobile edge computing

Architectures and applications of mobile edge computing

Software containerization (Docker)

Workload/service orchestration (Docker Swarm, Kubernates)

Metrics for context-aware/location-aware workload allocation

Rea

23762 - Physics of Complex Systems

PDF, ADOC.

Learning outcomes

Basic knowledge of physical and mathematical methods to develop dynamic and statistical model for the study of complex systems. Basic knowledge of graphical methods 2D and 3D used to illustrate the results.

Course contents

Introduction to the Complex Systems Physics and definition of the concept of complexity in science. The role of mathematical models in Physics: concept of predictivity. Construction of a model for a complex system and the role of nonlinear interactions. Introduction to the study of dynamical systems with applications to complex systems models. Methods for the study of stochastic dynamical systems. Introduction to statistical mechanics: concept of emergent property, critical state and phase transition. Analysis of models both from a theoretical and numerical point of view for the description of complex systems. Applications to physics, physical chemistry, biology, economics and social systems. Analysis of data distributions, comparison of exponential laws and power laws. Examples of study of a complex system model.

30214 - Logical basis of Computer Science

PDF, ADOC.

Learning outcomes

At the end of this course, students have acquired the logical foundations of areas of theoretical computer science such as ambda-calculus, rewriting systems, computational complexity, database theory

and formal methods. He is able to encode programming constructs in lambda-calculus, to describe simple algorithms in logical form, to

express queries to databases in predicative form and to specify properties of reactive systems as formulas of temporal logic.

Course contents

First module:

1. Propositional and First-Order Logic (recall)

Syntax, Semantics, Soundness and Completeness, Undecidability of First Order Logic

1. Untyped Lambda Calculus

Syntax and operational semantics. The lambda-calculus as a programming language: evaluation strategies and encodings of data types; Turing completeness

1. Meta-theory of untyped lambda calculus

Confluence.

1. Simply typed lambda-calculus and Curry-Howard isomorphism

Curry's style and Church's style syntaxes. Isomorphism with propositional minimal logic. Type checking and type inference algorithms.

1. Meta-theory of simply typed lambda-calculus

Weak and strong normalization theorems

1. Curry-Howard isomorphism and extensions to the typing system

Products and coproduts, empty and singleton types. Parametric polymorphisms (minimal introduction to System-F). Minimal introduction to dependent types and Hindley-Milner polymorphisms.

Second module:

1. Logic and Databases

Relational algebra, FO as Query Language, Cilindrical Algebras, Implementation aspects

1. Logic and Computational Complexity

Finite structures and decision problems. NP and PSPACE characterization via first order logics. SAT Solving.

1. Logic and Formal Methods

LTL and CTL: syntax and semantics. Reactive systems and their verifications. Model Checking.

1. Logic and Artifcial Intelligence

Epistemic logic: syntax, semantics, applications

Readi

70090 - Computer Graphics

PDF, ADOC.

Learning outcomes

At the end of the course, students know fundamentals of 3D computer graphics (polygonal modeling and real-time rendering).

In particular, they are able to model and render scenes making use of suitable open source softwares and libraries.

Course contents

Raster-scan systems, I/O devices, graphics libraries, event-driven programming. 3D mesh and graphics representation. 2D/3D geometric transformations, viewing transformations, perspective and parallel projections, window-viewport transformations. Graphics pipeline. Real-time rendering, algorithms with hidden parts removal (hidden lines and hidden surfaces), illumination models and shading algorithms (Z-buffer), texture mapping. 3D polygonal modeling. The course includes a practical activity in which the HTML5 programming language + JavaScript + WebGL graphics library and GLSL programming will be used (module 1 (6 CFU)).

In module 2 (2 CFU) we will see the Three.js cross-browser JavaScript library, an API used to create interactive and animated 3D Computer Graphics applications on the Web using WebGL. Geometric modeling of NURBS curves and surfaces with examples in Three.js.

12569 - Computational Mathematics

PDF, ADOC.

Learning outcomes

At the end of the course, the student knows varied techniques and tools at the basis of the computational solution of problems from scientific calculus. She/He is able to solve applicative problems, from interdisciplinary study or didactic

areas, in an integrated, symbolic and numeric, software environment.

Course contents

Introduction to the Mathematica environment: Kernel, FrontEnd, notebook.

Introduction to programming within Mathematica.

Graphics and visualization tools.

Employing the system capabilites to analize and solve a particular applied problem, of didactical interest to the student, via a package development.

30216 - Probability Models

PDF, ADOC.

Learning outcomes

At the end of the course the student knows elements of some advanced probability theories with applications to computer science, such as Markov chains with discrete and continuous time. She / he is able to analyze some simple stochastic systems related to applications.

Course contents

Recaps of basic probability topics in discrete spaces.

Probability spaces, random variables, independence, measurability, expected value and conditional expected value. Convergence and law of large numbers.

Stochastic processes in discrete time and discrete spaces.

General notions. Filtrations. Martingales and random walks.

Math finance in discrete time.

One-period market models. Valuation and hedging of derivatives. Fundamental theorems of asset pricing. Multi-periodal models. Binomial model and extension.

Markov chains.

Introduction to Markov chains. Construction off Markov chains. State classifications. Stationary distributions and convergence.

Stochastic control.

Formulation of stochastic control problems in discrete time. Dynamic programming: Bellman's equation and verification theorems. Applications.

Readings/Bibliography

90748 - Blockchain and Cryptocurrencies

PDF, ADOC.

Learning outcomes

At the end of the course, the student knows the relevant themes related to blockchain technologies, cryptocurrencies, smart contracts and novel applications that can be built over the blockchain. The student is able to develop simple smart contracts that can be deployed on a blockchain.

Course contents

Bitcoin and novel cryptocurrencies gathered momentum in the last months. More and more investors look with interest at these technologies, while others label them as a dangerous speculative bubble. The truth is that the blockchain, and the alternative implementations of a distributed ledger, represent very interesting technologies, that can be exploited to build novel distributed applications. The underlying building blocks are related to many concepts and research areas of computer science in general. This course will illustrate the main principles and conceptual foundations of the blockchain and the Bitcoin network.

Program

Introduction to peer-to-peer sysystems

Overlay topologies and decentralization

Introduction to Crypto and Cryptocurrencies

The blockchain: how to achieve decentralization

Transactions and transaction scripting languages

Mining

Attacks to the blockchain

Anonymity

Smart contracts

90733 - Data Analytics

PDF, ADOC.

Learning outcomes

At the end of the course, the student: (i) is aware of different types of data-analytics (diagnostic, predictive, prescription, etc) and of the main enabling techniques; (ii) is able to design and implement a full data-pipeline process, from the data acquisition until the data analysis and valorization; (ii) knows the main applications of data analytics, with a special emphasis on industrial and business applications.

Course contents

The course introduces concepts, techniques and tools for the design and implementation of data valorization and analytics processes. To this purpose, the course aims at providing an exhaustive illustration of all the stages of a digital data pipeline, from the data acquisition, pre-processing and knowledge extraction through statistical and Machine Learning techniques, to the data visualization and performance evaluation. In addition, it discusses state-of-the-art applications of the data-analytics on business use-cases and on technological scenarios characterized by high industrial impact, and enabled by the availability of big-data (e.g. IoT and Industry 4.0). After a brief recap of Python programming concepts (including the libraries for the data processing), the course illustrates -in sequence- each stage of the data-pipeline. More in detail, we review the essential techniques of data acquisition (data querying, APIs, Web scraping, etc) and the main architectures of data streaming and pipelining (e.g. AWS DP). At the next stage, we illustrate the pre-processing techniques for data filtering/cleaning, feature selection/transformation, dimensionality reduction. We then present the most common techniques of data visualization, aimed at showing the outcomes but also at supporting the design choices of the analytics process, as well as their relative implementations through Python library. A key component of the data pipeline, and hence of the course, is constituted by the illustration of techniques for automatic knowledge extraction from the datasets; to this purpose, we present in detail the most used techniques of Data Mining/Machine Learning, based on Supervised/Unsupervised approaches, and their implementations through Python frameworks (e.g. Scikit Learn, PyTorch). Finally, we address metrics and methodologies for the performance assessment of the data analytics process. We conclude the course with seminars on relevant applications of the data-analytics in business/industrial use-cases, by envisaging the participation of external companies working on the field. In the following, we list the course contents discussed so far:

Recap of Python programming and Python libraries for data science (Pandas, Numpy, etc)

Stages of the data-pipeline:

Data acquisition: techniques and architectures

Data pre-processing: cleaning, transformation, dimensionality reduction, feature extraction, etc

Data visualization: markers and channels, separability, graph types

Modeling

Base concepts (classification vs regression, overfitting vs underfitting, generalization, regularization, etc)

Supervised approaches (e.g. KNN, SVM, introduction to neural networks)

Unsupervised approaches (e.g. k-Means, Gaussian Mixture model)

Performance analysis: metrics, evaluation methods, hyperparameters optimization

Data analytics applications in business/industrial use-cases

90730 - Social Network Analysis

PDF, ADOC.

Learning outcomes

The course is meant to examine the essential models, methods and topics of social network analysis, while also comprehending information networks, where more generally nodes and links represent respectively data and relations between data. The first half is devoted to those analytical methods based on the comparison between real-world networks and random graphs, with emphasis on the configuration model and modularity clustering.The second half is devoted to objective function-based methods and to fuzzy models for community/module detection, with focus on the cluster score of vertex subsets quantified by pseudo-Boolean functions.

Course contents

The programme is the same for both attending and non-attending students:

Introduction to Network Analysis, gentle introduction to the field of network analysis and its usages in other fields of research (e.g., computer science, forensics, archeology, literature, history, science of religion, etc.).

Research Design and How to Read a (Network Analysis) Research Paper: introduction to the scientific publication process, elements of research papers (on network analysis), research design, analysis of research papers.

Mathematics of Networks: networks and their representation, types of networks, graph representations, paths and components, adjacency matrices and matrix representations, ways and modes, operations on Matrices.

Data Collection and Data Management: network questions, data collection and reliability, data formats and transformation, algorithms and software for network analysis and visualisation.

Measures and Metrics, Nodes: kinds of measures, multi-category nominal scales, ordinal and scalar measures, centrality, degree and other kinds of centrality (e.g. Google's PageRank), hubs and authorities, closeness and betweenness centrality, groups of nodes (cliques, cores, components and k-components), clustering and clustering coefficients, reciprocity and similarity, structural and other types of equivalence, homophily and types of assortative mixing.

Testing Hypotheses: the role of hypotheses in the scientific method, testing hypotheses in network analysis, permutation tests, dyadic hypotheses.

Measures and Metrics, Networks: small-world effects, degree distribution, power laws and scale-free networks, visualisation and properties of power-law distributions, local-clustering coefficient, cohesion, reciprocity, transitivity and the clustering coefficient, triad census, centralisation and core-periphery indices, centrality, random graphs, means on edges and degree, degree distribution, giant and small component(s), locally tree-like networks.

Network Visualisation: the importance of network visualisation, graph-layout algorithms,

embedding node attributes, node filtering, visualising ego networks, embedding tie characteristics, tie strengths, visualising network change.

Handling Large Networks: reducing the size of the problem, eliminating edges, pruning nodes, divide and conquer, aggregation, sampling, small-world and scale-free networks.

90747 - Corporate Social Responsibility and Business Ethics

PDF, ADOC.

Learning outcomes

The aim of the course is to present the fundamental theories that deal with (i) the role of companies in society and the meaning to be assigned to profits and to share value, (ii) with the relationships with stakeholders and (iii) with the business ethics. Specifically, the objective of the course is to achieve a level of knowledge that allows students to interpret the transformations taking place in the relationship between business and society as the consequence of the evolution of information technology. The course is articulated in two sections. A first section introduces the theoretical background while, in a second section, the economic and socio-political role played by companies such as Google, Facebook e Amazon will be analysed.

Course contents

Introduction to business ethics

Corporate Social Responsibility

Stakeholder theory

Normative ethical theory

Descriptive ethical theory

Tools and techniques of business ethics management

Typical ethical dilemmas in the economic activity of companies

Ethical dilemmas in the relationship between firms and politics

B3567 - INTRODUCTION TO QUANTUM COMPUTING

PDF, ADOC.

Learning outcomes

At the end of the course the student is acquainted with the basic mathematical notions underlying quantum computing seen as an alternative computing paradigm, and knows how to implement quantum algorithms as families of quantum circuits. Moreover, he has an understanding of more advanced topics like quantum cryptography, quantum error correction, and the development of graphical languages for quantum circuits.

Course contents

- overview on quantum computing
- mathematical background: linear algebra and complex Hilbert spaces
- qubits and the physics of quantum computation
- a quantum computational model: quantum circuits
- quantum algorithms (Grover, Shor)
- overview on quantum error correction and quantum cryptography
- overview on quantum programming and quantum graphical languages

81943 - Complex Systems & Network Science

PDF, ADOC.

Learning outcomes

At the end of the course, the student will have acquired the basic notions of complexity as it arises through interactions among simple agents in nature and in modern computing systems. The student will be able to identify, formulate, model and analyze new problems using the mathematical framework of dynamical systems and network science.

Course contents

Description: Modern information systems and services often rely on large numbers of independent interacting components to provide their functions. Under certain conditions, the behavior that results from these interactions can be unexpected and surprising. Complexity Science is an interdisciplinary field for studying global behaviors resulting from many simple local interactions in an effort to characterize and control them. Networks allow us to formalize the structure of interactions. They play a central role in the transmission of information, transportation of goods, spread of diseases, diffusion of innovation, formation of opinions and adoption of new technologies. Network Science is an interdisciplinary field for studying the interconnectedness of modern life by exploring fundamental properties that govern the structure and dynamic evolution of networks.

Contents: Complex systems: definitions, methodologies; Dynamical systems, Nonlinear dynamics; Chaos, Bifurcations and Feigenbaum constant, Predictability, Randomness and Chaos; Models of complex systems, Cellular automata, Wolfram's classification, Game of life; Autonomous agents,

Flocking, Schooling, Synchronization, Formation creation; Cooperation and Competition, Game theory basics, Nash equilibrium; Game theory: Prisoner's Dilemma, Coordination games, Mixed strategy games; Adaptation, Evolution, Genetic algorithms, Evolutionary games; Network Science: Definitions and examples; Graph theory, Basic concepts and definitions; Diameter, Path length, Clustering, Centrality metrics; Structure of real networks, Degree distribution, Power-laws, Popularity; Models of network formation; The Erdos-Renyi random model; Clustered models; Models of network growth, Preferential attachment; Small-world networks, Network navigation; Peer-to-peer systems and overlay networks; Structured overlays, DHTs, Key-based routing, Chord; Distributed network formation: Newscast, Cyclon, T-Man; Processes on networks: Aggregation; Rational dynamics: Cooperation in selfish environments, Homophily, Segregation; Diffusion, Percolation, Tipping points, Peer-effects, Cascades.

Prerequisites: Basic notions of computer system architecture, computer networks, operating systems, and probability theory.

91721 - Laboratory of Virtual Reality and Advanced Reality

PDF, ADOC.

Learning outcomes

At the end of the course, the student knows the fundamental theoretical principles, the hardware and the main software platforms required to design and create virtual and augmented reality environments. Furthermore, the student acquires the skills necessary to design, create, modify and interact with such environments

Course contents

Introduction to Augmented and Virtual Reality (AR/VR)

Principles: Computer Graphics in AR/VR

Principles: Perception and Human Factors (Designing and developing 3D UI: strategies and evaluation)

Principles: Interaction in AR/VR (Selection and manipulation, Travel, Navigation, Way finding, System control, Symbolic input)

Hardware: Output (Visual, Auditory, Haptic, Vestibular, and Olfactory Channels)

Hardware: Input (Tracking Systems, Input Devices)

VR contents (Standards and Designing)

Software: Platforms (Introduction, VR runtime systems, Real Time Physics Engines, Distributed Virtual Environments, Collaborative Virtual Environments, Game Engines)

Mixed Reality (Mixed Reality Continuum - AR to VR, Mixed Reality Technologies)

Applications of AR/VR

Future of AR/VR

Exercises/Demonstrations

91258 - Natural Language Processing

PDF, ADOC.

Learning outcomes

At the end of this course, the student acquires foundational notions about Natural Language Processing with particular attention at the statistical/algorithmic techniques. The methods and instruments from Natural Language Processing will be then applied at each level of linguistic analysis.

Course contents

Part I: Foundations

Introduction

Natural Language Processing - Problems and perspectives

Introduction/Recall to/of probability calculus

N-grams and Language Models

Markov Models

Recurrent Neural Network Language Models

The evaluation of NLP applications

Corpora

Corpora and their construction: representativeness

Concordances, collocations and measures of words association

Methods for Text Retrieval

Part II: Natural Language Processing

Computational Phonetics

Speech samples: properties and acoustic measures

Analysis in the frequency domain, Spectrograms

Applications in the acoustic phonetic field.

Speech recognition with HMM and Deep Neural Networks

Computational Morphology

Morphological operations

Static lexica, Two-level morphology

Computational Syntax

Part-of-speech tagging

Grammars for natural language

Natural language Parsing

Supplementary worksheet: formal grammars for NL

Formal languages and Natural languages. Natural language complexity

Phrase structure grammars, Dependency Grammars

Treebanks

Modern formalisms for parsing natural languages

Computational Semantics

Lexical semantics: WordNet and FrameNet

Word Sense Disambiguation

Word-Space models

Logical approaches to sentence semantics

Part III: Applications and Case studies:

Solving Downstream Tasks

Document classification

Sentiment Analysis

Named Entity Recognition

Semantic Textual Similarity

Prompting Pre-Trained Language Models

B1459 - ARTIFICIAL INTELLIGENCE, BLOCKCHAIN E CRIPTOVALUTE NELLO SVILUPPO SOFTWARE

PDF, ADOC.

Learning outcomes

This course aims at ensuring that at the end of the course the students: • know the main cognitive models that can explain how people develop software • are aware of the opportunities and the limits of applying techniques of artificial intelligence to develop software • are familiar at how the principles of software engineering can guide the development of AI systems • have an understanding of the principles of blockchain and their applications • understand the role and the potentials of cryptocurrencies and the problems associated to them when developing software • are able to build complex models of production processes and of products combining the most modern approaches, with specific attention to AI, blockchain, and cryptocurrencies.

Course contents

In the last years there has been a completely change of paradigm in software production that has lead to rethink at processes and production contexts. In particular, there has been a renewed interest in artificial intelligence, with a strong interest in the application of machine learning in predicting quality and productivity and in the use of cognitive models to orient the production processes.

On the other side, a strong need has emerged to identify software tools suitable to handle platforms for data management, that are always more complex and generate systems that at first appear to have strong effects but then are hard to evolve. Moreover, there are always more distributions of applications and development processes, interconnected with new methods of management of aspects and applications related to the introduction of blockchain systems and cryptocurrencies.

Prerequisites:

Even if there is not any formal prerequisite, the course is characterised as a software engineering course and will not present fundamental aspects of artificial intelligence and machine learning; to this end, it is recommended that the students not having such knowledge take before this course the one of Deep Learning, cod. 91250, of Prof. Asperti.

Knowledge and ability to acquire:

This is a software engineering course that intends to educate the students so that at the end of the course students:

know the main cognitive models that can explain how people develop software

are aware of the opportunities and limits of the applications of artificial intelligence for the development of software

become acquainted on how the principles of software engineering can guide the development of systems based on artificial intelligence

master the principles of blockchain and their applications

understand the role and the potentiality of cryptocurrencies and the problems associated to them in the development of software

manage to build complex production models and complex products combining the most advantages existing technologies, methods, and tools, including AI, blockchain systems, and cryptocurrencies

Topics:

Cognitive models for software development (systemic approaches, impulsive-reflective models, ...)

Application of AI to software development (testing, prediction, evaluation)

Use of software engineering principles in the development of AI systems

Blockchain (principles, distributed ledger technologies, smart contracts, platforms)

Cryptocurrencies (principles, transactions, consensus algorithms, overview of existing currencies

Readings/B

93478 - COMPUTER VISION

PDF, ADOC.

Learning outcomes

At the end of the course the students will be able to implement algorithms addressing relevant computer vision tasks, such as: object detection, semantic segmentation, image and video captioning. During the course they will learn the basics of image, video analysis and computer vision. They will gain knowledge about the design and implementation of convolutional neural networks, recurrent neural networks and how to combine them. During the course they will also acquire familiarity with the relevant frameworks used to design modern deep architecture.

Course contents

The course introduces concepts, and tools for the design and implementation of image acquisition, processing, and analysis techniques. List of the contents:

Image Formation and Acquisition: geometry of image formation; lenses; field of view and depth of field; image sampling and quantization.

Spatial Filtering: Linear shift-invariant operators. Mean and Gaussian filtering. Median Filtering.

Bilateral filtering.

Edge Detection: image gradient; non-maxima suppression; Laplacian of Gaussian; Canny edge detector.

Local Invariant Features: detectors and descriptors; Harris Corners; scale-invariant features; SIFT features;

Camera calibration: Projective coordinates and perspective projection matrix. Intrinsic and extrinsic camera parameters. Zhang's algorithm.

Instance Detection: pattern matching; shape-based matching; Hough transform.

Object Detection: two-stages, one-stage, and anchor-free detectors, RoI pooling operator, feature pyramid networks);

Semantic Segmentation: fully convolutional networks, transposed and dilated convolutions, RoI Align operator, semantic, instance, and panoptic segmentation;

Metric Learning: deep metric learning, contrastive and triplet losses, multi-task learning, application to different recognition/identification tasks;

Attention Mechanism: self-attention in RNN, Transformer architecture, Vision Transformer.

Prerequisites:

- Linear Algebra
- Basic knowledge of Machine Learning and Deep Learning
- Programming and Python

Re

90749 - Computing Education

PDF, ADOC.

Learning outcomes

The course aims to provide theoretical knowledge, techniques and tools helpful in teaching computer science. At the end of the course, the student is familiar with the main pedagogical and didactical approaches for teaching computer science at different school levels. Students can organise and teach computer science courses, compare and choose different methodologies to generate teaching materials, and evaluate learning.

Course contents

Learning objectives of this course include:

- knowledge of some historical, epistemological and ethical aspects of Computer Science as a

scientific discipline and of the motivations underlying the necessity of its teaching;

- understanding of pedagogical aspects and learning theories in the context of computer science teaching;
- knowledge of multiple CS-specific teaching approaches;
- knowledge of the main cognitive difficulties in learning CS (with particular focus on programming), and knowledge of possible strategies to adopt to overcome them;
- ability to formulate and manage learning paths consistent with national standards and curricula related to Computer Science in schools of all levels;
- planning ability to organize laboratories, classroom equipment; ability to integrate students' devices as useful tools for learning.

Topics covered in the course include:

The scientific vision of Computer Science.

Computational Thinking.

Constructivism and constructionism applied to CS teaching.

Top-down and bottom-up approaches in CS teaching.

Teaching of Programming, Algorithms and Data Structures.

Pedagogical implications in the choice of programming languages.

Teaching of technological aspects: computer architecture, operating systems, networks.

CS teaching methods:

- unplugged
- dramatization/visualization
- code reading exercises
- debugging of others’s code
- program execution visualization
- making/tinkering
- repositories as software museums

Difficulties and misconceptions in learning to program.

Creation of learning paths

- in primary schools
- in lower secondary schools
- in scientific lyceums - applied sciences
- in specific courses of technical institutes

- in other upper secondary schools

Assessment methodologies of computer programs.

Importance, planning, and management of computer laboratories.

- Use of BYOD in lessons.

Software licenses: libre (free and open source) and proprietary software: implications in education

Affective and motivational aspects in computer science learning.

66870 - Concurrent Models and Systems

PDF, ADOC.

Learning outcomes

At the end of the course, the student will learn the basic ingredients of concurrency theory, their models and verification systems on such models. (S)He will be able to analyze simple concurrent programs with automatic or semi-automatic tools.

Course contents

- Introduction to concurrency and to the problem of the correctness of reactive systems design.
- Labeled transition systems
- Behavioral equivalences: traces, simulation, bisimulation f(strong and weak), properties.
- The language CCS: syntax and SOS semantics.
- Subclasses of CCS: finite processes, finite-state processes, regular processes, BPP, finite-net processes, finitary CCS.
- Turing completeness of finitary CCS; undecidability of behavioral equivalences of finitary CCS.
- Value-passing CCS.
- Algebraic properties, behavioral congruences and axiomatizations.
- Espressivieness of CCS: encodability of additional operators (internal choice, hiding, sequential composition)
- The problem of muti-way synchronization: Multi-CCS; case study: dining philosophers.
- Petri nets: definition, equivalences, decidable properties, expressiveness.
- Languages for representing Petri nets. Distributed computability.
- Fixpoint theory, least and greates fixpoints, strong bisimilarity as a greatest fixpoint.
- Hennessy-Milner Logic (HML), estention with recursively defined formulae, modal mu-calculus.
- Analysis and verification tools for CCS and HML: Concurrency Workbench (CWB).
- Modeling, analysis and verification of some mutual exclusion algorithms with CWB.

91269 - Multimedia Data Management

PDF, ADOC.

Learning outcomes

The course aims to provide the knowledge and skills necessary for the effective and efficient management of multimedia (MM) data, with particular attention to the problems of MM data representation, MM data retrieval models, and interaction paradigms between the user and the MM system (both for purposes of data presentation and exploration). We first consider architectures of traditional ("standalone") MM systems; then, we concentrate on more complex MM services, by primarily focusing on search engines, social networks and recommendation systems.

Course contents

Basics on Multimedia Data Management

Multimedia data and content representations

MM data and applications

MM data coding

MM data content representation

How to find MM data of interest

Description models for complex MM objects

Similarity measures for MM data content

MM Data Base Management Systems

Efficient algorithms for MM data retrieval

MM query formulation paradigms

Sequential retrieval of MM data

Index-based retrieval of MM data

Automatic techniques for MM data semantic annotations

Browsing MM data collections

MM data presentation

User interfaces

Visualization paradigms

Dimensionality reduction techniques

Result accuracy, use cases and real applications

Quality of the results and relevance feedback techniques

Use cases and demos of some applications

Multimedia Data on the Web

Web search engines

Graph-based data: semantic Web and social networks

Web recommender systems

N.B. For students of the second cycle degree programmes (LM) in Artificial intelligence and Computer Science, the "Efficient algorithms for MM data retrieval" part of the program is not required.

84401 - Context-Aware Systems

PDF, ADOC.

Learning outcomes

At the end of the course, the student is able to design, deploy and evaluate ubiquitous systems and mobile applications able to adapt their behaviors to the context characteristics and to the current location/activity of the user. At the end of the course, the student: -knows the fundamental concepts of context-aware computing, and the main techniques for the localization of users/devices and the human activity recognition; -knows the fundamental models of context-data representation and managing; - knows the main middleware and software architectures in order to deploy adaptive and ubiquitous applications and services

Course contents

The course addresses the design and deployment of ubiquitous and context-aware services and applications, made possible by the pervasive diffusion on the market of devices able to sense the environment and to analyze the sensed data. The course program is structured in two main parts. The first part illustrates the definition of "context" and context-aware systems, focusing on the design and implementation of location-aware and activity-aware systems. Special focus will be given to the spatial data management, by illustrating the main technologies for indoor/outdoor positioning, mapping APIs, geo-data storage and location intelligence.The second part will present the lifecycle of distributed context-aware applications by focusing on mobile edge computing systems where the allocation of workloads take into account contextual data, such as the user position. To this aim, we will introduce edge-computing technologies and frameworks for software containerization (e.g., Docker), task orchestration (e.g., Docker Swarm and Kubernates), and workload/service migration based on users' mobility. In the following, we provide a brief summary of the course program:

Introduction and definition of context and context-awareness

Use case of context-aware systems

Location-aware systems

Location-based services (LBS)

Positioning technologies

Mapping APIs

Spatial database

Location intelligence

Privacy in LBS

Activity-aware systems

Human Activity Recognition (HAR): enabling technologies

Supervised AI/ML techniques for HAR systems

Mobile edge computing

Architectures and applications of mobile edge computing

Software containerization (Docker)

Workload/service orchestration (Docker Swarm, Kubernates)

Metrics for context-aware/location-aware workload allocation

Rea

81613 - Business Intelligence

PDF, ADOC.

Learning outcomes

The students is expert in the design, implementation and information systems management for intelligence of decisions that melt on the management knowledge of analytical nature. The second part (30 hours) of the course is developed in the computing laboratory using the system language of SAS System and SAS Enterprise Miner. At the end of the course, the student knows: - Customer relationship management (CRM) and Customer Intelligence; - Problems of evaluation of the advertising impact; - Oriented information system implementation for the analytical approach to the Business Intelligence; - Software realization for Analytical Business Intelligence projects and statistical data mining.

Course contents

Supervised and unsupervised classification Multivariate analysis: principal component analysis, correspondence analysis, discriminant analysis

73387 - Creativity and Innovation M

PDF, ADOC.

Learning outcomes

At the end of the course the student will gain knowledge of the following topics: The necessity for creativity. Learnings from the science of creativity studies. The DA VINCI Model & Method or the creative thinking process. Strategies and components for specific thinking stages. Innovation: hurdles and strategies for success. Practical applications to study cases.

Course contents

1) The necessity for creativity and its definition

2) Creativity in the history of art and science.

3) Theoretical foundations of creative thinking. Cognitive modelling.

4) The Da Vinci thinking model. Strategies and processes for specific thinking stages.

5) Application of creative thinking to study cases.

81799 - Project Management and Soft Skills M

PDF, ADOC.