Introduction to Artification Intelligence (AI)

Short Summary. The course will deal with classical artificial intelligence methods, starting from intelligent agents, search techniques (uninformed, informed, local search), search with logics, and then continuing with constraint satisfaction problems, adversarial search, planning of complex actions in non-deterministic or uncertain environment, all the way to knowledge representation, planning and expert systems. Throughout the course, students will engage in a variety of activities designed to enhance their learning experience. These include exercise sessions, class presentations, homework and programming practice. Through these activities, students will learn to approach real-world challenges with classical AI techniques.

Instructors

Lecturer: Tatjana Petrov <tatjana.petrov@units.it>, Tutor: Andrea Bertolini <andrea.bertolini@phd.units.it>

Format

The course counts 9 ECTS credits, consisting of 72 teaching hours, consisting of lectures, exercises, programming practice, and exam preparation.

Background

Modern AI applications are underpinned by classical techniques like search algorithms (uninformed, informed, local, adversarial), which drive optimization in robotics, game AI, and navigation systems. Techniques such as constraint satisfaction problems (CSP), logic, and Markov decision processes (MDP) continue to be vital in fields like scheduling, verification, formal reasoning, and reinforcement learning, ensuring that AI systems make efficient and intelligent decisions in uncertain and dynamic environments. Knowing classical AI techniques will equip the students with foundational problem-solving tools that are widely applicable across AI domains.

Objectives.

[Knowledge and understanding]

Students will understand classical artificial intelligence techniques to multiple application domains, from automatic planning, constraints satisfaction, to the control of intelligent agents.

[Applying knowledge and understanding]

Through exercise sessions, homework and projects, students will learn to apply the learnt concepts in practical scenarios: how to identify the problem and which of the techniques shown in class may apply.

[Making Judgement]

The course will equip the student with critical judgment about the difficulty of the algorithmic search problem at hand, as well as possible methods of approaching it.

[Communication skills]
Students will enhance their ability to communicate their progress and final findings through discussion with team-members and presentation in front of the class.

[Learning skills]
Students will enhance both their independent learning skills as well as learning through teamwork.

Requirements. Basics of programming, logic, graph theory, algorithms and complexity.

Content.

- Introduction and history of (classical) artificial intelligence
- Intelligent agents
- Search strategies and heuristic research
- Logic agents, SAT solvers, and logic programming
- Local search techniques
- Constraints Satisfaction Problems
- Adversarial search (games)
- Research in non-deterministic and/or partially observable environments
- Expert systems and knowledge engineering
- Knowledge representation and ontologies
- Automatic scheduling and planning
- Classical planning techniques
- Heuristics for scheduling

Teaching methods.
Each topic of the course will be covered with lectures and hands-on sessions. At the hands-on sessions, the students will be working on a set of tasks which will allow them to practice and deepen their understanding of the concepts seen during the lectures. Implementations will use programming languages Python and Prolog.

Exam and grading.
The verification consists of three parts:
(1) a written exam value 70% of the final grade,
(2) oral exam valued 20% of the final grade,
(3) project work, homework, and activity in class which altogether are valued 10% of the final grade.

To pass the exam, the student must have at least 60% overall performance. The written exam is organized at the end of lecture sessions and it contributes 70% to the final grade. To be admitted to the oral exam, the student must have 60% of the written exam correctly solved. The oral exam then counts towards 20% towards the final grade. It consists of a short examination of the student with questions on two or three theoretical concepts seen in class. The written exam evaluates technical knowledge on specific topics and the ability to focus on detail. On the other

hand, the oral exam serves to evaluate whether the student understands the concepts and is able to transfer knowledge over different applicative scenarios or to connect them into a big picture.

Throughout the year, students will have the chance to submit six (6) homework and eight (8) practical programming assignments. Some homeworks can be done in teams. Students will receive feedback on homework solutions and this can influence the final grade up to 10%. The first homework assignment involves a presentation of studied material in front of the whole class. Presence and activity in class can raise the final grade by 5%.

This course explores topics closely related to one or more goals of the United Nations 2030 Agenda for Sustainable Development (SDGs)

Literature.

RUSSELL, Stuart J.; NORVIG, Peter. *Artificial intelligence: a modern approach*. Pearson, 2016.
The latest is the fourth edition but previous editions are also adequate.