Overview

Artificial intelligence (AI) is the science and engineering of making intelligent machines. In this age of the computer, artificial intelligence typically involves the fabrication and analysis of intelligent computer programs. It is a field firmly rooted in the discipline of computer science while drawing on insights from philosophy, mathematics, psychology, and neuroscience. As a research endeavor, AI has frequently been mocked for its failure to produce a convincing simulacrum of a human intellect, while many of the successes of AI research efforts have become so well accepted that they are now considered the domain of other areas of computer science.

This course briefly introduces elementary artificial intelligence findings and techniques. While it is impossible to fully cover over 60 years of active research in a single term, the curriculum for this class has been designed to cover
a moderately broad range of foundational topics, including heuristic search, logical reasoning, planning, reasoning under uncertainty, and machine learning. The general educational goal of this course is to make students familiar with the major problems and methods addressed within these foundational areas, allowing successful students to recognize system development situations in which these techniques might be fruitfully applied.

Learning Outcomes

Students who successfully complete this course will have acquired a sufficient understanding of the basic concepts and methods of artificial intelligence to make use of some elementary artificial intelligence techniques in the design of computer systems. They will also come to possess insights into how these methods may be used to model the information processing mechanisms of human cognition. This class will also provide students with preparation for advanced training in artificial intelligence, such as that provided by basic graduate level coursework in the field.

These student learning outcomes are expected to be valuable to students pursuing a variety of undergraduate degree programs at the University of California, Merced. Specifically, with regard to the Cognitive Science B.S. and B.A. degree programs, these student learning outcomes contribute to a number of program learning outcomes, including increasing student knowledge of landmark findings and theories in cognitive science, developing student appreciation for formal and computational approaches in cognitive science, and improving the ability of students to argue for or against the use of computational models to address various scientific questions. With regard to the Computer Science and Engineering B.S. degree program, the student learning outcomes outlined in the previous paragraph contribute to a number of the computer science program learning outcomes, including practice applying knowledge of computing and related mathematics, practice designing, implementing, and evaluating computer-based systems, practice communicating technical information to an interdisciplinary audience, exposure to current computing techniques, and exposure to examples of trade-offs in software system design. In summary, students enrolled in a variety of undergraduate education programs should find this course useful for meeting their program's objectives.

Resources

Meetings

Class meetings will take place every Tuesday and Thursday of this semester according to the schedule outlined at the end of this document. Meetings will take place from 12:00 P.M. to 1:15 P.M. in 170 Student Services Building. Class meetings will consist of lectures reviewing the highlights of course readings, introductions to new material, and discussions of related topics. Attendance at these lectures is required for enrolled students. Class meetings will also include opportunities for discussion, and students are encouraged to actively participate in these conversations. Indeed, student performance will be evaluated, in part, based on class participation.

Three class laboratory sessions have been scheduled for this course. The first is scheduled for Fridays from 8:00 A.M. to 10:50 A.M., and the second is scheduled for Fridays from 2:00 P.M. to 4:50 P.M., with the third scheduled for Fridays from 5:00 P.M. to 7:50 P.M.. All three laboratory sessions will be held in 100 Science & Engineering Building I. Enrolled students are required to attend one of these three laboratory sessions each week. Laboratory sessions will involve the supervised hands-on execution of computer exercises, and they will also provide time to discuss and work on the course’s five computer programming assignments. Student mastery of course material will largely be evaluated based on student performance on computer programming assignments, and these laboratory sessions will provide opportunities for students to receive assistance from the teaching team on these assignments, as well as time to write the required programs.

Web Site

Online materials for this class will be disseminated through the UCMCROPS system. This course management system may be accessed through its central portal:

https://ucmcrops.ucmerced.edu/
Enrolled students should be provided with access to a section of UCMCROPS specifically reserved for this class. In particular, a section shared between CSE 175 and COGS 125, called “F14–INTRO TO AI”, should be available to all enrolled students. The instructor should be promptly informed if such access is not appropriately granted.

This web site will be used to announce updates to the class schedule, as well as to distribute class materials. Students are required to obtain regular access to this resource, and they are strongly advised to consult it frequently (e.g., daily).

Readings

The primary source of expository readings for this course will be:


This book is available in the campus bookstore. In addition to providing explanatory text, this tome will also be used to guide the sequencing of topics discussed in this class, and some exercises and examination questions may be drawn from its pages. Students are required to have regular access to this textbook throughout the semester.

While this book provides rather comprehensive coverage of the topics relevant to this course, it does not always offer particularly deep treatments of those topics. Thus, the instructor will supply students with a few additional readings during the semester. These supplementary readings will typically be made available online, through the course web site. Students are also encouraged to augment their reading with other sources. Recommended readings on specific topics are available, upon request, from the instructor.

Software

Discussing artificial intelligence techniques is rarely sufficient to develop a deep understanding of them. In hopes of fostering a deeper understanding, students enrolled in this class will produce five small computer programs making use of the elementary AI methods covered.

While artificial intelligence methods have been implemented in virtually every widely adopted computer language, the traditional language for AI programming is Lisp. Lisp provides many constructs which have proved particularly useful in AI programming, and early Lisp software development tools have inspired many of the modern software development environments which are popular today.

Unfortunately, few undergraduate students receive formal training in Lisp programming. Instead, other computer programming languages, such as Java™ and C++, form the foundation of early computer science training. Thus, this course assumes that students possess a basic understanding of programming in Java™, and this knowledge will be leveraged during the completion of computer programming assignments. All computer programming assignments in this class must be completed in the Java™ language.

Relatively little class time will be spent on Java™ programming topics. Each student will be expected either to already be competent in Java™ or to be able to quickly acquire or refresh the necessary programming knowledge on their own. Students with strong programming skills in other languages, such as C++, should find Java™ fairly easy to master. There are many self-paced tutorials in Java™ available, both online and in book form. For example, the following book is available in electronic form from the campus library web site:


Students with doubts concerning their Java™ programming skills should consult with the instructor as early in the term as possible. The programming assignments used in this class form a central component of the course curriculum, and the successful completion of these assignments will require basic knowledge of Java™ programming. In order to formalize this requirement, UCM’s introductory programming courses, CSE 20 and CSE 21, which are conducted in Java™, are officially listed as prerequisites for this class.
While this course provides only a little time for refreshing Java™ knowledge, students should feel free to direct Java™ questions to the teaching team via electronic mail throughout the semester.

**Laboratory**

Students are welcome to conduct work on programming assignments in any Java™ development environment on any computer available to them. Laboratory facilities will contain computers equipped with the Eclipse Java™ integrated development environment (IDE). These machines may be used by students to work on computer programming assignments. Indeed, if assignment solutions are developed in any other environment, they should be checked in the laboratory Eclipse environment, as this is the environment in which submitted assignments will be evaluated. The open source Eclipse IDE is available free of charge from “www.eclipse.org”, and students should feel free to install this environment on their own computers so that they may work on programming assignments in a familiar environment on their own machines. Whether obtained in the laboratory or on a personal computer, students are required to gain access to an appropriate Java™ development environment.

**Expectations & Evaluations**

**Background**

Students attending this course are expected to have some background and interest in computer science, cognitive science, or a related field. An introductory familiarity with basic probability theory and differential calculus is also expected. Perhaps most importantly, students must have a basic competence in generating software using the Java™ programming language. UCM’s introductory computer programming course sequence, CSE 20 and CSE 21, are prerequisites for this class, as they provide basic training in Java™ programming.

The University of California, Merced, is committed to ensuring equal educational opportunities for students with disabilities. An integral part of this commitment is the coordination of specialized academic support services through the Disability Services office. Students with a physical or learning disability may ask Disability Services to assist in communicating this fact to the instructor so that appropriate accommodation may be provided. Absent notification, the instructor may assume that no such accommodation is sought.

**Participation**

Studying artificial intelligence is a challenging interdisciplinary endeavor requiring familiarity with notions from algorithm analysis, data structures, formal logic, statistics, psychology, and a number of other disciplines. Thus, students may find this class both rewarding and demanding. Mastery of the course materials will require extensive reading, puzzling through unfamiliar concepts, active participation in class discussions, learning new mathematical formalisms, many hours of hands-on experience constructing small intelligent systems, and a willingness to view computation and cognition in new ways.

Specific readings will be suggested as appropriate for each class meeting, and participants will be expected to have studied those readings prior to gathering, so as to promote thoughtful questions and knowledgeable discussion. Students will be expected to contribute constructively to discussions, bringing to bear both insights into the material at hand and relevant knowledge acquired in other contexts.

One of the best ways to learn artificial intelligence techniques is to actively use them in the fabrication of working computer programs. Thus, five programming assignments will be issued with the goal of helping students acquire a richer understanding of relevant AI methods. These programming assignments will involve the production of working Java™ programs, and they will require substantial time and effort to complete. Student solutions to these assignments will be evaluated by the teaching team, and appropriate feedback will be given.

While the fabrication of intelligent programs is an excellent way to explore AI topics, there is insufficient time in a single academic term to allow for the generation of programs covering the full range of AI methods surveyed in this course. Thus, students will necessarily encounter some concepts and techniques only in readings and lectures.
order to evaluate mastery of these topics, two written examinations will be administered. One exam will be conducted about halfway through the term, and a final comprehensive exam will be administered at the end.

Course participants are expected to embrace the course material with earnest effort, to contribute constructively to the learning of other students, and to always behave ethically and with civic concern. Students should come to every class meeting prepared to discuss relevant topics. Programming assignments are to be completed by their respective due dates. The ideas and contributions of others should be appropriately cited. (This includes ideas and contributions garnered from readings, online resources, presentations, conversations, and any other source.) Students are expected to bring educational obstacles to the instructor’s attention as early as possible, so that such problems may be promptly resolved.

Learning can be greatly facilitated by interactions between course participants, and these interactions are encouraged. Students should feel free to discuss lecture topics, readings, and even programming assignment concepts with each other. The actual completion of programming assignments and other exercises (including examinations), however, should be conducted on an individual basis. All assignments and exercises submitted for evaluation should reflect the understanding and effort of the individual participant. Not a single line of computer code should be shared between course participants. If there is ever any doubt concerning the propriety of a given interaction, it is the student’s responsibility to approach the instructor and clarify the situation prior to the submission of work results. Also, helpful conversations with fellow students, or any other person (including members of the teaching team), should be explicitly mentioned in submitted assignments. Failure to appropriately cite sources is called plagiarism, and it will not be tolerated!

Evaluation

The teaching team will provide comments on all assigned work submitted in a timely manner for evaluation. Those students who are to receive grades for this course will have their work assessed approximately as follows:

<table>
<thead>
<tr>
<th>Activity</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pretest</td>
<td>1%</td>
</tr>
<tr>
<td>Programming Assignment #0</td>
<td>4%</td>
</tr>
<tr>
<td>Programming Assignments #1 – #4</td>
<td>10% each</td>
</tr>
<tr>
<td>Midterm Exam</td>
<td>15%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>20%</td>
</tr>
<tr>
<td>Laboratory Exercises</td>
<td>10%</td>
</tr>
<tr>
<td>Reading Quizzes</td>
<td>5%</td>
</tr>
<tr>
<td>Attendance &amp; Participation</td>
<td>5%</td>
</tr>
</tbody>
</table>

Student performance will be evaluated in comparison to that of other students, both past and present. Classroom participation will be closely monitored throughout the term.

At the beginning of each course lecture meeting, enrolled students will be expected to quickly complete a written quiz on the material covered in the assigned readings for the day. These quizzes are intended to encourage students to keep up with the required readings, as well as to help assess student understanding of those readings. It is worth noting that student performance on these short quizzes, aggregated over the full term, is a substantial evaluation criterion used in the assignment of final course grades.

Programming assignments are to be submitted for evaluation through the “Assignments” mechanism in the course UCMCROPS site. Such submissions must be received by the specified due date and time. Late assignments will not be evaluated, and they will receive no credit, even if they are substantially complete and correct. An “early submission deadline” will be provided for each programming assignment, however, and assignments received prior to that deadline will receive extra credit for their timeliness. Also, submitted programs that do not successfully compile under the laboratory Eclipse Java™ IDE (i.e., they produce “build errors”) will not be evaluated and will receive no credit.

While many laboratory meetings will provide time to receive assistance on programming assignments, some of these sessions will also introduce additional required computer exercises. These exercises are designed to assist with the acquisition of relevant skills, with a focus on programming skills, control system design skills, and knowledge engineering skills. Most of these laboratory exercises demand mastery — they cannot be completed until correct
performance is demonstrated, and, thus, the evaluation of student performance on these exercises will be based on whether or not they have been completed. Note that the collection of laboratory exercises assigned throughout the semester carries about double the weight of reading quizzes when determining final course grades.

Schedule

In the schedule that appears below and on the following pages, a check mark (✓) identifies an assigned reading for the date in question. A heart symbol (♡) identifies a reading that may be considered supplementary and optional.

Introduction

August 28: Introduction
✓ Course Syllabus
✓ AIMA, §1.1.

September 2: History
✓ AIMA, §1.2 – 1.5.

September 4: Philosophy
✓ AIMA, Chapter 26.

September 9: Agents
✓ AIMA, Chapter 2.


Search

September 11: The Concept of Search
✓ AIMA, Appendix A.
✓ AIMA, §3.1 – 3.3.

September 16: Uninformed Search
✓ AIMA, §3.4.

September 18: Heuristic Search
✓ AIMA, §3.5 – 3.7.

September 23: Local Search, Optimization, & Genetic Algorithms
✓ AIMA, §4.1 – 4.2.

September 25: Search in Adversarial Games
✓ AIMA, §5.1 – 5.5, 5.7, 5.9.
Knowledge Representation & Logic

September 30: Propositional Logic
  √ AIMA, §7.1 – 7.4.

October 2: Reasoning With Propositional Logic
  √ AIMA, §7.5.

October 7: Effective Reasoning Algorithms
  √ AIMA, §7.6 – 7.8.

October 9: First-Order Logic
  √ AIMA, §8.1 – 8.2.

October 14: Knowledge Engineering
  √ AIMA, §8.3 – 8.5.

October 16: Unification & Forward Chaining
  √ AIMA, §9.1 – 9.3.

October 21: Backward Chaining
  √ AIMA, §9.4.

October 23: Resolution Theorem Proving
  √ AIMA, §9.5 – 9.6.

October 28: Ontological Engineering
  √ AIMA, §12.1 – 12.6, 12.8.

Planning

October 30: Planning, STRIPS, & PDDL
  √ AIMA, §10.1 – 10.2.

November 4: Planning Algorithms
  √ AIMA, §10.3 – 10.6.

November 6: Midterm Examination
  • No Assigned Readings

November 11: Veterans Day Holiday
  • No Meeting
Uncertainty

November 13: Probability Theory & Decision Theory
\( AIMA, \) Chapter 13.

November 18: Bayesian Belief Networks

Learning

November 20: Learning Decision Trees
\( AIMA, \) §18.1 – 18.3.

November 25: Artificial Neural Networks
\( AIMA, \) §18.4, 18.6 – 18.7, 18.11 – 18.12.

November 27: Thanksgiving Day Holiday
- No Meeting

December 2: Reinforcement Learning
\( AIMA, \) 21.1 – 21.4.

Perception

December 4: Machine Vision
\( AIMA, \) Chapter 24.

Natural Language Processing

December 9: Natural Language Understanding
\( AIMA, \) 23.1 – 23.4, 23.6.

December 11: Course Summary & Final Examination Review
\( AIMA, \) Chapter 27.

December 18: Final Examination
- The final examination will be held 8:00 A.M. to 11:00 A.M. in SSB 170.