Fall 2005

CS 409/609: Principles of Artificial Intelligence

John C. Gallagher
Wright State University - Main Campus, john.gallagher@wright.edu

Follow this and additional works at: https://corescholar.libraries.wright.edu/cecs_syllabi

Repository Citation
https://corescholar.libraries.wright.edu/cecs_syllabi/142

This Syllabus is brought to you for free and open access by the College of Engineering & Computer Science at CORE Scholar. It has been accepted for inclusion in Computer Science & Engineering Syllabi by an authorized administrator of CORE Scholar. For more information, please contact corescholar@www.libraries.wright.edu, library-corescholar@wright.edu.
Principles of Artificial Intelligence - CS 409/609
Fall, 2005, Wright State University

Catalog Description
Problem solving methods in artificial intelligence (AI) with emphasis on heuristic approaches. Topics include methods of representation, search, intelligent agents, planning, learning, natural language processing, logic, inference, robotics, and case-based reasoning. Three hours lecture, two hours lab.

Objectives and Goals
Artificial Intelligence as a field is FAR too large to cover in one academic quarter. The major goal of this class, therefore, is to give broad coverage of the basic classical methods. Further, we will attempt to discuss both the strengths and weaknesses of these classic methods as a means of understanding the motivations of various AI offshoot movements and as a means of setting the stage for more advanced and more specialized advanced topics courses.

Instructor
John C. Gallagher
259 Russ Engineering Center
(937) 775-3929
gallagh@cs.wright.edu

Meeting Time and Place
8:00 PM to 9:15 PM Tuesday and Thursday, 154 Russ Engineering Center

Textbook

Supplementary materials will be linked from the course WWW site.

Prerequisites
CS400 and experience in either the LISP or Scheme programming languages. CS340 (LISP) meets the language requirement.

Grading
Each student’s grade will be determined by assessing his or her ability to discuss issues, solve problems, and create and debug working AI systems. Each student will be provided with opportunities to demonstrate those abilities through both written examinations and programming assignments. In addition, students taking this course for graduate credit will be required to complete a more challenging final project.
Homework: For nearly every lecture, there will be a small number of homework assignments drawn from the textbook. Although answers will neither be collected nor formally assessed, students are encouraged to maintain a portfolio notebook that contains their written responses to all exercises. In cases of "borderline grades" (i.e., a student is within a percentage point of the boundary between final course letter grades), the student's portfolio will be assessed to determine if the student gets the benefit of the doubt. Complete, neat and essentially correct portfolios will bump the student to the next highest letter grade. Incomplete, sloppy, and/or essentially incorrect portfolios will ensure the student receives the lower grade. Also, since all exams will be largely drawn from and/or very similar to the homework exercises, students may find that the best way to prepare for the exams is to complete the suggested exercises in a timely manner. Refer to the detailed course schedule for lecture content and corresponding homework exercises.

Mini Midterm Examinations: Students will be given four half-hour, closed-book, exams over the course of the quarter. Exams will occur approximately once every two weeks and are designed to help ensure that students are "keeping up" with the material as the quarter progresses. They are also meant to help the instructor assess if basic material has been learned before moving on to more advanced topics. Only the best three of four scores will count toward the final grade. In order to maintain fairness and allow the timely return of mini-exams to students, make-ups for missed exams are NOT PERMITTED. Absences, periodic poor performance, and other difficulties are sometimes unavoidable. This is why only the best three mini-exams are counted. Everyone gets one "miss" or "bad day" for free. Students missing multiple mini-midterms due to a documented emergency are likely missing sufficiently many other course activities to justify petitioning for withdrawal from the course or other appropriate academic remedy. These cases should be discussed with the instructor.

Final Examination: There will be a cumulative final exam scheduled during the normal final exam period for the course. The final exam will be closed-book and will cover all material covered in lecture and in the assigned readings.

Programming Assignments: There will be approximately one programming assignment per week as the quarter progresses. The point values for each will be indicated on the assignment handout(s). Refer to the detailed course schedule for more information.

Graduate Project: Students taking this course for graduate credit will be required to complete a larger-scale programming project. Ideally, these projects should be relevant to the student's ongoing or intended graduate research. The instructor will contact each graduate student and negotiate a project idea individually. The deliverable on the project will be a short paper in the style of a conference proceedings submission. This will be discussed with each student during project negotiation.
### UNIT #1: INTRODUCTION AND PROBLEM SOLVING BY SEARCH

<table>
<thead>
<tr>
<th>DATE</th>
<th>TOPIC</th>
<th>READING</th>
<th>HOMEWORK PORTFOLIO ASSIGNMENTS</th>
<th>PROGRAMMING ASSIGNMENT DUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>9/06</td>
<td>Course Mechanics/Background/History</td>
<td>R&amp;N 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9/08</td>
<td>Scheme Tutorial</td>
<td>Scheme Tutorial</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9/13</td>
<td>Intelligent Agents</td>
<td>R&amp;N 3</td>
<td>2.1-2.6</td>
<td>Assignment #1</td>
</tr>
<tr>
<td>9/15</td>
<td>Problem Solving by Search</td>
<td>R&amp;N 4</td>
<td>3.1-3.7</td>
<td></td>
</tr>
<tr>
<td>9/19</td>
<td>Informed Search</td>
<td>R&amp;N 5</td>
<td>4.1-4.3.4.6</td>
<td></td>
</tr>
<tr>
<td>9/22</td>
<td>Adversarial Search</td>
<td>R&amp;N 6</td>
<td>6.1-6.3</td>
<td></td>
</tr>
</tbody>
</table>

### UNIT #2: LOGIC BASED AGENTS AND INFERENCE

<table>
<thead>
<tr>
<th>DATE</th>
<th>TOPIC</th>
<th>READING</th>
<th>HOMEWORK PORTFOLIO ASSIGNMENTS</th>
<th>PROGRAMMING ASSIGNMENT DUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>9/27</td>
<td>Logical Agents</td>
<td>R&amp;N 7</td>
<td>7.1-7.4</td>
<td></td>
</tr>
<tr>
<td>9/29</td>
<td>First Order Logic</td>
<td>R&amp;N 8</td>
<td>8.1-8.6</td>
<td></td>
</tr>
<tr>
<td>10/04</td>
<td>Inference in First Order Logic</td>
<td>R&amp;N 9</td>
<td>9.1-9.4</td>
<td></td>
</tr>
<tr>
<td>10/06</td>
<td>Knowledge Representation</td>
<td>R&amp;N 10</td>
<td>10.1-10.4</td>
<td>Assignment #3</td>
</tr>
</tbody>
</table>

### UNIT #3: PLANNING AND LEARNING

<table>
<thead>
<tr>
<th>DATE</th>
<th>TOPIC</th>
<th>READING</th>
<th>HOMEWORK PORTFOLIO ASSIGNMENTS</th>
<th>PROGRAMMING ASSIGNMENT DUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/11</td>
<td>Planning</td>
<td>R&amp;N 11</td>
<td>11.1-11.8</td>
<td></td>
</tr>
<tr>
<td>10/13</td>
<td>Planning and Acting</td>
<td>R&amp;N 12</td>
<td>12.1-12.5</td>
<td></td>
</tr>
<tr>
<td>10/18</td>
<td>Learning from Observations</td>
<td>R&amp;N 18</td>
<td>18.1-18.7</td>
<td></td>
</tr>
<tr>
<td>10/20</td>
<td>Statistical Learning</td>
<td>R&amp;N 20</td>
<td>20.1-20.4 20.11</td>
<td>Assignment #4</td>
</tr>
<tr>
<td>10/27</td>
<td>Unit #3 Mini-Midterm</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Unit #4: Fun Stuff and Applications

<table>
<thead>
<tr>
<th>DATE</th>
<th>TOPIC</th>
<th>READING</th>
<th>HOMEWORK PORTFOLIO ASSIGNMENTS</th>
<th>PROGRAMMING ASSIGNMENT DUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/25</td>
<td>Perception</td>
<td>RN 24</td>
<td>24.1– 24.4</td>
<td></td>
</tr>
<tr>
<td>10/27</td>
<td>Mobile Autonomous Robotics</td>
<td>RN 25</td>
<td>25.1– 25.6</td>
<td></td>
</tr>
<tr>
<td>11/1</td>
<td>Special Topics</td>
<td>TBD</td>
<td>TBD</td>
<td></td>
</tr>
<tr>
<td>11/3</td>
<td>Special Topics</td>
<td>TBD</td>
<td>TBD</td>
<td></td>
</tr>
<tr>
<td>11/8</td>
<td>Graduate Student Presentations</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11/18</td>
<td>Final Exam Review</td>
<td>UNIT #4 Mini-Midterm</td>
<td>Assignment #8</td>
<td></td>
</tr>
</tbody>
</table>
Grade Calculation: Grades will be computed on a A/90%, B/80%, C/70%, D/60% scale. Clustering of scores might cause the thresholds to be lowered, but they will never be raised. Curving, if applied at all, will be applied on final composite grades only. Individual exams and/or assignments are never curved or adjusted in any way. The final letter grade percentage will be determined by taking a weighted average of the percentages of each assessment component.

<table>
<thead>
<tr>
<th>Assessment Component</th>
<th>Undergraduate Weight</th>
<th>Graduate Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mini-Midterms</td>
<td>30%</td>
<td>25%</td>
</tr>
<tr>
<td>Programming Assignments</td>
<td>30%</td>
<td>30%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>40%</td>
<td>30%</td>
</tr>
<tr>
<td>Graduate Project</td>
<td>Not applicable</td>
<td>15%</td>
</tr>
</tbody>
</table>

Example: Bob Dobbs, undergraduate, received 83% of possible points on his mini-midterms, 95% of possible points on his programming assignments, and a dismal 65% on his final exam. His final course grade, using undergraduate weightings, would be:

\[
(0.3 \times 83\%) + (0.3 \times 95\%) + (0.4 \times 65\%) = 24.9\% + 28.5\% + 26\% = 79.4\%
\]

Bob is within a percentage point of the cutoff for a "B". If he has a good homework portfolio, he gets the benefit of the doubt and a "B". If not, he gets a "C".

Graduate grading would be similar, except with an additional assessment component and slightly different weights.

About Programming Languages
AI is about techniques – it is not about specific programming languages. Therefore students are free to complete all programming assignments except for assignment #1 in any language that the instructor can compile and execute on his own machine. On the other hand, one cannot afford to ignore the mutual influences between classical AI and the Lisp programming language. Therefore, students WILL be required to complete at least one assignment in the Scheme language. Scheme is a variant of Lisp for which there are many good open source and freeware compilers and tutorials. Students are encouraged to use Scheme and/or Lisp for all their assignments, but are not required to do so. Note that all course examples and exam questions will be written using either pseudo-code in the style of the book or Scheme.

Academic Integrity
Student-teacher relationships are built on trust. For example, students must trust that teachers have made appropriate decisions about the structure and content of the courses that they teach, and teachers must trust that the assignments students turn in are their
## Unit 1: Introduction and Problem Solving by Search

CS 409/609 Principles of Artificial Intelligence  
Fall, 2005

### Course Mechanics

- When in doubt about course policies, refer to the syllabus. If that doesn’t clear it up, ask the instructor -- do not rely on hearsay or rumor.

- Updated course info is ALWAYS available at [http://carl.cs.wright.edu/ceg409.html](http://carl.cs.wright.edu/ceg409.html)

### History/Background

- Scheme tutorial

### Intelligent agents

- Problem solving by search

### Informed search

- Adversarial search

### Unit Outline

<table>
<thead>
<tr>
<th>Course mechanics</th>
</tr>
</thead>
<tbody>
<tr>
<td>History/Background</td>
</tr>
<tr>
<td>Scheme tutorial</td>
</tr>
<tr>
<td>Intelligent agents</td>
</tr>
<tr>
<td>Problem solving by search</td>
</tr>
<tr>
<td>Informed search</td>
</tr>
<tr>
<td>Adversarial search</td>
</tr>
</tbody>
</table>
Prerequisites
CS400 (Data Structures and Software Design)
Familiarity with either LISP or Scheme
CS340 or equivalent
Self-taught in LISP-like functional language

Textbook
ISBN: 0137903952
Other reference material will be provided or is available free of charge on the WWW.

Course is structured as four units of approximately two-weeks in length each.
Syllabus gives a breakdown of lecture topics by date. Breakdown includes readings drawn from textbook.
Syllabus gives a list of suggested homework assignments to be completed after each lecture. These are not graded, however, at least some exam questions will be styled on these problems. Homework problems are drawn from textbook.

Grading
Four closed-book mini-midterms
One per unit, best three count
30 minutes each, given at end of scheduled lecture period
Five programming assignments
First is a language tutorial, remaining four cover topics from each of four units.
Turn-in formats discussed in each homework handout
Comprehensive, closed-book, final exam
Graduate students must complete a short paper on an AI topic of their choosing.
Note: Recommended daily homeworks are not graded. They can kick you over the edge in borderline grade situations. Refer to the syllabus for details.
Course Mechanics

Grade Weights

Undergraduates
  30% mini-midterms, dropping lowest one
  30% programming assignments
  40% final exam

Graduates
  25% mini-midterms, dropping lowest score
  30% programming assignments
  30% final exam
  15% graduate project/paper

Grade Boundaries
90%/A  80%/B  70%/C  60%/D
Boundaries might be moved down if statistics indicate doing so. They will not be moved up.

General Observations

The course will be fast-paced. The lectures will focus on core concepts and may or may not cover every aspect of the reading material. You are still responsible for having done the reading. Some topics not explicitly spoken of in lecture will impact both exams and programming assignments. As with many CS endeavors, AI is at least as much skill-based as it is knowledge-based. You will not develop mastery through readings and lectures alone. Practice on both conceptual and programming problems is critical. If you complete the majority of the recommended homework problems and carefully complete all the programming assignments, there is no reason to think you would not earn an A or B in the course.

Unit Outline

Course mechanics

History/Background

The opening of Russell and Norvig, chapter 1, says:

In which we try to explain why we consider artificial intelligence to be a subject most worthy of study, and in which we try to decide what exactly it is, this being a good thing to decide before embarking

In words not suitable for a textbook, but perhaps more to the point:

Is this all a waste of time? If it is -- bad. If it isn't -- then let's get our heads and our butts wired together before starting so it doesn't turn into a waste of time.