Model AI Assignments 2014

Todd W. Neller
Gettysburg College
tneller@gettysburg.edu

Laura E. Brown
Michigan Technological University
lebrown@mtu.edu

Roger L. West
University of Illinois at Springfield
rwest2@uis.edu

James Heliotis, Sean Strout, Ivona Bezáková
Rochester Institute of Technology
{jeh,sps,ib}@cs.rit.edu

Bikramjit Banerjee, Daniel Thompson
The University of Southern Mississippi
Bikramit.Banerjee@usm.edu
daniel.l.thompson@eagles.usm.edu

Abstract

The Model AI Assignments session seeks to gather and disseminate the best assignment designs of the Artificial Intelligence (AI) Education community. Recognizing that assignments form the core of student learning experience, we here present abstracts of five AI assignments from the 2014 session that are easily adoptable, playfully engaging, and flexible for a variety of instructor needs. Assignment specifications and supporting resources may be found at http://modelai.gettysburg.edu.

An Introduction to Monte Carlo Techniques in AI — Part I

Todd W. Neller

Monte Carlo (MC) techniques have become important and pervasive in the work of AI practitioners. In general, they provide a relatively easy means of providing deep understanding of complex systems as long as important events are not infrequent.

Consider this non-exhaustive list of areas of work with relevant MC techniques:

General: Monte Carlo simulation for probabilistic estimation

Machine Learning: Monte Carlo reinforcement learning

Uncertain Reasoning: Bayesian network reasoning with the Markov Chain Monte Carlo method

Robotics: Monte Carlo localization

Search: Monte Carlo tree search

Game Theory: MC regret-based techniques

In these assignments, we will recommend readings and provide novel exercises to support the experiential learning of the first two of these uses of MC techniques: MC simulation, and MC reinforcement learning. Through these exercises, the student will add such techniques to their problem-solving repertoire, gain understanding of their strengths and weaknesses, and gain discernment of appropriate applications.

Multi-Player Games: Introducing Assignments with Open-Ended Strategies in CS2

James Heliotis, Sean Strout, Ivona Bezáková

We present open-ended freshman projects where students design and implement their own player strategies for a commercial board game. The games chosen in previous terms are Quoridor (Gigamic), San Francisco Cable Cars (Queen Games), Gobblet (Blue Orange Games), and The aMAZE-ing Labyrinth (Ravensburger). Unlike modern computer-based games, most board games are inherently discrete. The board tends to have a fixed number of allowed positions for the game pieces and every player performs a search through a finite number of possible moves to decide which move to take next. As such, designing a player strategy for a board game provides a very natural context for basic data structures and algorithms, including search algorithms and other elementary AI concepts often covered in a freshman-level computer science sequence. The aspect of this type of project that distinguishes it from most others is that it allows for continual improvements to one's strategy, something that is motivating to both beginners as well as more experienced programmers. We provide the students with the software infrastructure including a server-based multi-player game engine and browser-based graphic display, allowing the student to focus on strategy, and permitting the project to be implemented in multiple programming languages.

The presentation will focus on the components of the game engines, the milestones we have established for student deliverables, and demonstrations of a few of the games.

Strimko by Resolution

Bikramjit Banerjee, Daniel Thompson

This assignment is mainly a programming project, where undergraduate students in an introductory AI class will be asked to write a complete solver for the Strimko puzzle. This project will give students hands-on experience with using propositional logic for knowledge representation, logical inference via resolution, and the use of backtracking search (implementing resolution) for completeness of the solver. Additionally, students will be encouraged (via extra credits) to transform this code to use first order logic, and research answer set programming to get an idea of how modern day puzzle solvers are created. This project gives beginning AI students an integrated view of logical inference and backtracking search - traditionally treated as separate topics in an introductory AI class. By avoiding the use of a black-box SAT solver in the base project, the students get a "look under the hood" of logical reasoning. An incomplete inference procedure (RESOLVESTRIMKO) is built, separate from (backtracking) search. Students have fun making inferences where RESOLVESTRIMKO fails.

Basecode is provided to handle the non-AI (GUI) part of the project. A demo version is provided, with (part of) the assigned code included. This can be extended to create an in-class demo of the use of resolution to make useful inferences in a small yet non-trivial domain.

Party Affiliation Classification from State of the Union Addresses

Laura E. Brown

Text classification is a classic problem for the use of Naïve Bayes with many real-world applications related to spam detection, article categorization, and sentiment analysis. The assignment has students extending their understanding of the basic Naïve Bayes classifier, learning how it may be applied to the problem of text classification specifically using a multinomial and Bernoulli model. The students implement a method to train the Naïve Bayes model (estimate the probabilities from a training sample) and apply the model to new samples. The task is to predict party affiliation of US Presidents based on their State of the Union Addresses (SOTU) for the multinomial and Bernoulli models.

Extensions to the assignment can have students learning and applying concepts from text processing including punctuation removal, recognizing compound nouns, word stemming, eliminating stop words, etc. Additional assignment variants include applying other classifiers to the prediction task, e.g., kNN, SVMs. The SOTU assignment was evaluated in a pilot study. Initial student reactions were positive to the assignment, expressing enjoyment of applying a classification method to real data. The assignment required a thorough understanding of the two Naïve Bayes models for project completion.

Comparing Brute-Force Searching versus the MRV Heuristic in Sudoku Puzzles

Roger L. West

In an introductory AI course students learn about many informed search strategies as alternatives to the woefully inefficient brute-force search, which represents the extreme lower bound of efficiency. Informed searches use one or more heuristics to reduce the size of the search space that must be examined, resulting in a faster search, on average. In most cases, the criterion for measuring relative efficiencies of two search algorithms is the elapsed time needed to solve the same problem. Unfortunately, fast computer hardware or a small problem size can mask the increased efficiency of an informed search, which can lead to the misconception that an informed search doesn't represent a significant improvement over an uninformed search. A method of measuring relative efficiency that does not depend on elapsed time would help students to better understand the benefits of informed searching. Here I describe a programming assignment that uses the number of attempted variable assignments as the criterion for comparing a brute-force search versus an informed search employing the Minimum Remaining Values (MRV) heuristic in a popular constraint satisfaction problem: the Sudoku puzzle.