Projects

CSE 473 – Introduction to Artificial Intelligence, University of Washington, Spring 2010.

Demonstrations in class on June 2 and June 4.

A project in CSE 473 should bring together one or more AI techniques and involve the application of these techniques to the solution of some model problem inspired by a real-world problem. The project should be not only a learning experience for the students creating it, but during the demonstrations and in possible informal pre-demonstration meetings the project should teach others about the techniques being used and issues in applying them.

Part I. Topic Selection.

Choose a topic within the categories below. The project topics are varied but not unlimited. In addition each project must follow certain guidelines.

Categories:

1. *Thoughtful reconstructions*. Create a Python program that demonstrates the key technique and application in one of the following classical programs of AI: The Logic Theory Machine (propositional logic theorem-proving using human-oriented rules of inference), The General Problem Solver (forerunner of more general state-space-search approaches), The Geometry Theorem Prover (Gelernter's program that used both a deductive method and geometrical diagrams to help steer the search), STUDENT (Dan Bobrow's program that accepted algebra word problems from the user, interpreted, and solved them), SHRDLU (Winograd's integration of a microworld with a natural-language interface), AM (the automated mathematician of Lenat, based on heuristic search), MYCIN (Shortliffe's expert system for infectious diseases, using certainty values in production systems).
2. *PROLOG implementation of a probabilistic model*. Analyze a real-world problem for which probabilistic reasoning would be helpful. Construct a Bayes net that represents a probabilistic model for the problem. Implement predicates that support querying the model with different states of input evidence. Include predicates that can show the full joint probability distribution for the random variables in your model.
3. *Event-stream analysis with Bayes nets*. Develop a model for a problem requiring the analysis of a stream of events, such as an unfolding emergency, a security attack, or the actions of a group of people. Develop a Bayes net for making probabilistic inferences from the event reports. A Bayes Net Toolkit, written in Python, is available to support projects in this category.
4. *Case-Based Reasoning*. Apply Case-Based Reasoning to some family of problems. For the problems, see the list of major world problems in category 6.
5. *AI in image understanding*. Apply an AI technique to image understanding. For example, treat the image segmentation problem as a constraint satisfaction problem. Alternatively, develop and train a classifier to recognize the pixels in an image that represent human skin. The classifier might be a two-level feedforward (backpropagation) network, or a support vector machine. A special software tool called PixelMath is available for use in this option for those who wish to use it.
6. *Serious games*. Using a major world problem (e.g., global warming, species extinction, overpopulation, world peace, sustainability, etc.) as inspiration, develop a game based on a model of that problem, and create an agent

that can play that game against a human opponent. The agent should use one or more of these AI techniques to make its moves: state-space search, logical reasoning, probabilistic reasoning. Projects in this category may, if desired, use either of two special software tools: (a) CoSolve, a web-based facility to makes it possible for multiple users (who are either solving problems or playing games) to apply the operators (defined in Python) through a browser; (b) TStar, a tree-based interface (written in Python) for manually applying operators to explore a state space. If you are interested in using these tools, let the instructor know in your planning questionnaire answers.

Guidelines:

A. Each project (except those in the PROLOG probabilistic reasoning category) must be in Python 2.6 (or 2.5).
B. Each project should be designed not only to work (that is, not only to compute outputs correctly as a function of inputs), but also to show and teach the mechanisms and model that it uses. This can be done with one or more of the following: (a) graphical displays (for example using Tkinter or the Python Imaging Library), (b) printing of a narrative to the console that explains what the program is doing, (c) interactive controls for setting parameters and inspecting data structures, and (d) an easy way to run a demo session based on a prepared file.
C. Each project should feature both an AI technique (or more than one technique) and an application. The technique might be a heuristically guided search, the use of logical or probabilistic reasoning, a constraint satisfaction procedure, or a machine learning algorithm. The application should involve the thoughtful modeling of some real-world situation or problem. As part of the model, the project should provide a justification for each model variable, state space, rule or constraint, and the way in which it simplifies or abstracts some aspect of the real-world problem.

Part II. Project design

Specification: Provide the following when you submit your project planning questionnaire: (a) name and partnership intentions, (b) category of project topic, (c) particular topic in the category, (d) technique(s) to be featured, (e) application to be featured, (f) at least one reference to material you can use to get started on the project, (g) plan for dividing up the work between partners, (h) possible scenario showing what the user types in or does and showing the system's response – simulated screen shot or sample dialog.

Important Dates:

May 21: Planning questionnaire due.
May 28: Progress questionnaire due.
June 1: Preliminary code due.
June 2: Begin demos; first round of project peer evaluations due.
June 4: Finish demos; second round of project peer evaluations due.
June 7: Reports and final code due.