CIspace: tools for learning computational intelligence

David Poole and Alan Mackworth*

April 26, 2001

Abstract

This paper describes *CIspace: tools for learning computational intelligence*, a collection of interactive tools for learning AI, and how it fits in with other such tools. *CIspace* can be found at
`http://www.cs.ubc.ca/labs/lci/CIspace/`

CIspace has been developed by a team consisting of Kevin O'Neill, Shinjiro Sueda, Leslie Tung, Audrey Yap, Michael Pavlin, Michael Cline, Cristina Conati, Peter Gorniak, Holger Hoos, Alan Mackworth, and David Poole.

1 Introduction

CIspace is a collection of tools designed to help people learn AI concepts. It is intended that there are exploratory tools as well as interactive online tutorials. It began as an offshoot of our textbook Computational Intelligence: a logical approach [Poole, Mackworth and Goebel, 1998]. We soon discovered that there were concepts that are not easy to describe in a static medium like a book or even in a lecture with traditional audio-visual aids. We needed ways to show the dynamics of reasoning and to make the basic concepts simple and easy to understand. *CIspace* should be seen as an ongoing experiment in pedagogy.

Our main design goals are:

*Department of Computer Science, University of British Columbia, 2366 Main Mall, Vancouver, B.C., Canada V6T 1Z4, `poole@cs.ubc.ca`, `mack@cs.ubc.ca`, `http://www.cs.ubc.ca/spider/poole` `http://www.cs.ubc.ca/spider/mack`

- to have tools that show the dynamics of basic AI concepts, using appropriate graphics and animation
- to keep the simple ideas simple, but not necessarily simplified
- to make algorithms and concepts transparent, so that students can see exactly how they work at various levels of abstraction
- to allow for multi-scale stepping execution so that the tools can be used in a spectrum from fully autonomous to user-driven
- to have load-and-go tools that require a low threshold to be useful
- the graphical user interfaces should have an intuitive, unified look and feel
- the tools should be modular; teaching a unified collection of ideas, rather than a large system trying to fulfill (possibly competing) goals
- the tools should each come with preloaded examples, but also allow the users to modify them or build their own
- the various tools should allow users to experiment with, and evaluate, different techniques for solving the same problems
- the tools should be able to vehicles for studies in intelligent systems, for example in user modelling or adaptive systems [Gorniak and Poole, 2000a; Gorniak and Poole, 2000b]
- to cater to different presentation modes and learning styles, the tools are designed to be used
  - for in-class demonstration
  - as exploratory environments that students can play with
  - within interactive tutorials that provide focussed instruction but still with interaction
  - with examples that work that students can modify and elaborate

2 Current Tools

We currently have the following learning tools:

- The Graph Searching applet lets the user interact with a variety of blind and heuristic search strategies using their own graphs (drawn on a canvas) or using a number of predefined graphs. Currently we have six search strategies ranging from depth-first search to A* search; with each we allow loop detection, multiple-path pruning, or no pruning.

- The Constraint Satisfaction via Arc Consistency applet is for solving constraint satisfaction problems. The user can interactively or automatically prune domains via arc consistency or can split domains.

- The Hill Climbing applet is for solving constraint satisfaction problems using local search techniques. This uses the same graph representation as the arc consistency applet. It lets us have random restarts, choose heuristics to decide which variable to select next and which value to select for it. We can plot how the number of unsatisfied edges changes through a single run of the hill climbing algorithm, as well as plot the run time distribution of a number of runs.

- The Neural Network applet is for training and evaluating neural networks. This lets us design the network and train it on some data using backpropagation. We can plot how the total error evolves for each step of the gradient descent. Once trained we can view the parameters and evaluate the network on new examples.

- The Belief network applet lets the user build and query belief networks (Bayesian network) and decision networks (influence diagrams). The system can explain how each number was computed.

- The Robot Control applet lets the user design and/or modify a hierarchical controller for a simple mobile robot. It uses a logic programming language to specify the controller. You can watch how the robot reaches goal locations while avoiding obstacles in a 2-D environment that can be changed dynamically.

- CILog is a simple logic programming system. Unlike the other learning tools, this is written in Prolog and does not run as an applet. It can be seen as pure Prolog, with declarative debugging and expert-system facilities (why, how, whynot, ask-the-user, etc). It also allows for negation as failure, consistency-based diagnosis, and abduction (although the current form does not work with both negation as failure and assumables). This lets students try to axiomatize a real domain rather than just build tools that lets others represent domains.

We are also expanding these to be part of a collection of interactive tutorials[1]. The aim is that we have interactive tutorials for a number of topics. One of the reasons for the interactive tutorials is that different students have different learning styles; some student like free-form interactive environments, while others get lost in them. Some like to be explained everything from the bottom up, but others are too impatient. We are planning on developing various ways to teach the material so that we can cater to various learning styles and mixed-initiative presentations (and to eventually do research on different learning styles and how to present material to each person in a way that is appropriate for them).

This summer (2001) we are planning on improving all of the applets, improving the functionality and making improvements based on our experience in using them for teaching AI courses. The biggest changes will be to the belief network applet (expanding it to include multi-stage decision problems), and the robot control applet (perhaps expanding it to include multiple agents).

3 Relationship to research and unified resource

We don't believe it is possible or desirable to have a set of official AI teaching resources where we keep the best of the resources on each topic. There are a number of reasons for this.

Different people have different views of what is a better tool. What is a better tool depends on the learning style of the student and what it is you actually want to teach them.

There are reasons why we have chosen not to include some applets. Not all ideas are equally important. Adding more features does not necessarily make a tool better. AI is an ongoing research field. One of the main roles of research is to prune ideas that don't work. Unfortunately this often means claiming that some idea that people have worked on and are still working on will not form the basis

[1]As a first prototype see:
`http://www.cs.ubc.ca/spider/poole/ci/itute/csp/csp.html`

for an intelligent system and so should not be included. We need people to make such judgments; however they could be wrong. It is much better to have a number of such resources, each of which gives a coherent view to the field. The diversity of what they choose to include and omit will make the discipline stronger.

We do believe that cooperation in the form of coordination of effort and comparing/contrasting approaches is useful. We plan on making the underling technologies (e.g., the graph drawing toolkit) open source (e.g., under the GPL) so that others can build upon what we have done.

We view *CIspace* as a vehicle for research as well as for teaching. AI is still young, and one of the main challenges is to have a core of underlying techniques that can be composed to form intelligence. One of our aims is to develop a core of ideas that are modular and fit together conceptually, rather than a complicated collection of ad hoc techniques. Keeping it simple is difficult; but we really only want to develop complicated theories once we have exhausted the simple ones.

We are quite likely wrong in what we think the underlying techniques that are important for intelligence. But it is important to exhaust the simple ideas first. Having a number of competing visions of AI is healthy for the discipline.

References

Gorniak, P. and Poole, D. [2000a]. Building a stochastic dynamic model of application use, *Proc. Sixteenth Conf. on Uncertainty in Artificial Intelligence*, Stanford.

Gorniak, P. and Poole, D. [2000b]. Predicting future user actions by observing unmodified applications, *Proc. 17th National Conference on Artificial Intelligence (AAAI-2000)*, Austin, Texas.

Poole, D., Mackworth, A. and Goebel, R. [1998]. *Computational Intelligence: A Logical Approach*, Oxford University Press, New York.