CIS 730 Artificial Intelligence
CIS 530 Introduction to Artificial Intelligence
Fall 2018

Homework 5 of 10: Machine Problem
Heuristic Search, Part 2 of 2
& Production Systems

Assigned: Mon 17 Sep 2018
Due: Fri 05 Oct 2018 (before midnight)

The purpose of this assignment is to continue to exercise your basic understanding of uninformed and heuristic searches through implementation of **variants of A* and IDA*** and experimentation with these variants. This assignment is also designed to exercise your basic understanding of **first-order logic** and **forward and backward chaining** through simple implementations using the AI programming languages CLIPS (*C Language Integrated Production System*).

This homework assignment is worth a total of 100%.
Each problem is worth 25% for CIS 730 students and 34% for CIS 530 students.

As of Sun 16 Sep 2018, Prolog (SWI-Prolog v7.6.4) is installed on `cislinux.cs.ksu.edu`. Both Prolog and CLIPS (v6.24, 15 Jun 2006) are installed on the class application server, `fingolfin.kdd.cs.ksu.edu`, and will be installed on the class DigitalOcean droplet, `finarfin`. You should be able to request an account on `cislinux` if you do not already have one (see https://support.cis.ksu.edu/CISDocs/wiki/Accounts), and you will receive e-mail with information on your `fingolfin` account. For connection instructions for both systems, see: http://bit.ly/kstate-cs-linux.

- CLIPS is available for Windows, MacOS, and Linux from: http://clipsrules.sourceforge.net/
- SWI-Prolog is available for Windows, MacOS, and Linux from: http://www.swi-prolog.org/download/stable (follow the APT instructions to install builds for Ubuntu and Debian distributions: http://www.swi-prolog.org/build/Debian.html)

References

For Problems 1-3, refer to:

1. The Wikipedia article on 8 queens: https://en.wikipedia.org/wiki/Eight_queens_puzzle
2. The PEAS description of an autonomous vehicle given by Rob St. Amant of North Carolina State University: http://www4.ncsu.edu/~stamant/411/lectures/02-agents/peas.pdf.
3. §7.2 (p. 236 – 239) of *Artificial Intelligence: A Modern Approach, 3rd edition* (2010) by Stuart Russell and Peter Norvig
4. The SWI Prolog tutorial: http://lpn.swi-prolog.org/lpnpage.php?pageid=online
5. The CLIPS textbook, *Expert Systems: Principles and Programming, 3rd edition* (1998) by Joseph Giarratano and Gary Riley.
6. (For MP4-5) TDT4136 *Introduction to Artificial Intelligence* by Pinar Øzturk and Vegard Edvardsen of the Norwegian University of Science and Technology: http://www.idi.ntnu.no/emner/tdt4136/ (see especially the Situation Calculus in Prolog lecture, http://www.idi.ntnu.no/emner/tdt4136/TRANSP/SitCalcProlog.ppt).

Problems

1. **(530 / 730). Greedy Best-First search.** Implement Greedy search using Best-First-Search applied with functions for *h* only (these need not be downward function arguments, but you should pass in some selector constant for the specified heuristic).

   Your program must print out the actual path and total cost in the following format, when run with command line "`mp2_3 [heuristic-number]`":

   ```
   A/A*: Best path found: 0 2 3; cost: 2 + 1 = 3
   ```

   (Your program may ignore the heuristic number if you are not completing the extra credit exercise, but it should accept the input.)

   Show the evolution of the OPEN and CLOSED lists. Sample output will be given in which your program's output will be compared automatically by the grader.

   Turn in your source code (e.g., `mp5_1.py`) and a problem solving trace (`mp5_1-out<n>.txt`) for test file `mp5_1-<n>.txt`.

2. **(730) Iterative Deepening Depth-First Search.** Extend your solution to MP2 to implement ID-DFS as described in Chapter 3 of Russell and Norvig *3e*.

   Your program must print out the actual path and total cost in the following format, when run with command line "`mp2_4`".

   ```
   ID-DFS: [Show each level being expanded]
   Best path found: 0 1 3; cost: 1 + 5 = 6
   ```

   Turn in your source code (e.g., `mp5_2.py`) and a problem solving trace (`mp5_2-out<n>.txt`) for test file `mp5_2-<n>.txt`.

3. **(530/730) First-Order Predicate Calculus (FOPC) *aka* First-Order Logic (FOL) and Constraint Satisfaction Problems (CSP).** Derive a predicate `N-Queens-Solution (N, QueenList)` such that QueenList contains a list of N queen rank positions on consecutive files that satisfies the constraints of N queens. **Note:** If you look up any Prolog solutions to adapt for this problem, cite your source and use FOL syntax rather than Prolog. How would you incorporate algorithms such as forward checking and heuristics such as MRV or LCV into this solution?

   Turn in your written rules in FOL (`mp5_3.pdf`).

4. **(530/730) Learning CLIPS.** See *Expert Systems: Principles and Programming, 3rd edition* by Joseph C. Giarratano and Gary D. Riley (see the Files/Handouts/Supplements directory). Implement rules in CLIPS and use them with facts describing the Wumpus World example from §7.5.4, p. 257-259 R&N 3e, to prove the sentence $\boldsymbol{P_{1,2}}$ (see Figure 7.13, p. 255) using CLIPS inference rather than `PL-FC-Entails` (Figure 7.15, p. 258),

   Turn in your source (`mp5_4.clp`) and a problem solving trace (`mp5_4-out.txt`) for test file `mp5_4-<n>.txt`.

Class Participation (required)

Read And Explain Pairs

After going over your **Read And Explain Pairs** exercise with your assigned partner (to be assigned and described in Canvas), post a short paragraph summarizing *unification* and a second containing any questions you may have on search to the class mailing list (**CIS530-L@listserv.ksu.edu** or **CIS730-L@listserv.ksu.edu**). That is, ask questions about any knowledge representation topic for which your understanding is unclear. This includes propositional logic, first-order logic, description logic, or theorem proving (forward and backward chaining, resolution strategies)

Extra Credit (10%)

Solve MP5-4 using Prolog.

Turn in your source (`mp5_EC.pl`) and a problem solving trace (`mp5_EC-out.txt`) for test file `mp5_EC-<n>.txt`.

Coming Up Next

Problem Set 6 (due Mon 22 Oct 2018) – Knowledge Representation and Reasoning, Part I: Clausal Form (CNF) Conversion, Elicitation, Ontologies, Analogy. You will start to work with eliciting ontological and fact-based knowledge about a simple domain and encoding it in Protégé-OWL. You will also start getting some experience working with analogical reasoning.

Machine Problem 7 (due Mon 29 Oct 2018) – Knowledge Representation and Reasoning, Part II: Ontology Reasoning, Classical and Robust Planning. You will apply the Pellet ontology reasoner on the Protégé-OWL ontology you developed in MP5. You will also solve some planning problems to simulate the behavior of algorithms for classical and modern (reactive) planning. The Waikato Environment for Knowledge Analysis (WEKA). Finally, you will start to work with Hugin, TETRAD, and WEKA.

Machine Problem 8 (due Mon 05 Nov 2017) – Reasoning and Learning, Part I: FOL Converter; Intro to WEKA and `scikitlearn`. You will start to write a program to convert partially parsed first order predicate calculus (FOPC), *aka* first-order logic (FOL) sentences, into CNF, i.e., clausal form. You will practice converting partially parsed first order predicate calculus (FOPC), *aka* first-order logic (FOL) sentences, into CNF, i.e., clausal form, and run these through a Prolog interpreter. You will also get your first hands-on experience by running two machine learning software packages (the Waikato Environment for Knowledge Analysis (WEKA) and `scikitlearn`) to train Logistic Regression, Decision Tree, and Perceptron models.

Problem Set 9 (due Fri 09 Nov 2017) – Reasoning and Learning, Part II: Probabilistic Reasoning (Inference and Causality), Version Spaces, and Decision Trees. You will solve some Bayesian network reasoning and learning tasks and a few classification problems to simulate the behavior of supervised inductive learning algorithms to prepare for the final machine problem.

Machine Problem 10 (due Fri 16 Nov 2017) – Perception and Understanding: Using WEKA; Artificial Neural Networks (ANNs), Genetic and Evolutionary Computation (GEC), Natural Language Processing (NLP), and Vision. You will apply learning algorithms in WEKA and `scikitlearn`, plus an ANN or GEC, to a classification task and to a pattern recognition or obstacle avoidance task for a simple mobile robot. You will solve problems and answer some discussion questions about perception and understanding. (This will include some practice final exam questions.)