Assignment 2: Problem Formulation

CSE 473: Introduction to Artificial Intelligence
The University of Washington, Seattle, Autumn 2020

Due Wednesday, October 28, 2020 at 23:59 PM, via GradeScope.

Overview:
This collaborative assignment consists of two parts. You'll be using Python 3.8 on this assignment in both Part A and Part B. Part A involves formulating a well-structured problem: The Farmer-Fox-Chicken-Grain puzzle. Part B involves formulating an ill-structured problem: a "wicked problem" from a list of acceptable wicked problems. You'll turn in two code files and one report file.

Collaboration:
This is a partnership assignment. Most students should work in partnerships of two. Working alone is an option, but students working alone will not be eligible for the extra-credit "partnership retrospective."

Getting Started:
Either (1) try to find a partner for this assignment, possibly by looking over some of the names of people you met during past breakout sessions, or in virtual sections or in virtual office hours; or (2) take part in automatic partnership, being set up by the staff. To take part, or not, enter your choice in the WebQ survey at https://catalyst.uw.edu/webq/survey/tanimoto/397777.

You can also choose to work alone. However, you are strongly encouraged to work with a partner if you can.

The reading for this assignment is *Applying AI Methodology in Problem Solving*.

Starter code is available here: a2-starter-code.tar.

The file **`a2ff.py`** is a starting template. Edit this file to develop your solution for Part A.

The file **`a2wp.py`** is another starting template. Edit this file to develop your solution for Part B.

The file **`Missionaries.py`** provides an example of a formulated problem, in this case, the Missionaries and Cannibals puzzle.

The file **`TowersOfHanoi.py`** provides another example of a formulated problem, in this case, the Towers of Hanoi puzzle.

The file **`Int_Solving_Client.py`** is a program that helps you test a problem formulation by supporting interactive solving by a user.

The file **`ItrDFS.py`** is a program that automatically solves a well formulated problem by running Depth-First Search on it.

You can see how ItrDFS.py solves the Missionaries and Cannibals puzzle by typing the following

```
python3 ItrDFS.py Missionaries
```

The file **`BFS.py`** is like ItrDFS.py but implements Breadth-First Search rather than Depth-First Search. Thus it will find shortest paths (in graphs with unweighted edges).

Start a report, possibly in Word or other text-processing tool, that you will eventually convert to PDF and name **`a2report.pdf`**. This should have a heading that reads like this

```
CSE 473, Autumn 2020, University of Washington
Assignment 2
Jon Doe and Fanny Fox

Wicked Problem Chosen: Finding a vaccine for COVID-19.

Part A:
```

(note that the partners' names are listed alphabetically by last name.

Part A (40 points).

Create a formulation for the puzzle presented in class that we call the Farmer-Fox-Chicken-Grain problem. The formulation must have the same components and overall structure as the provided example formulations have. For example, the Missionaries and Cannibals formulation has sections for common code (including a State class), operators, goal test, etc.

Test your formulation first with the provided Interactive Solving Client. When that seems to be working, test your formulation with the ItrDFS and BFS automated solving methods.

In the report under Part A, give the sequence of states found by BFS that represents the optimal solution to the Farmer-Fox-Chicken-Grain problem.

Part B (60 points).

Using the same format for problem formulation as in Part A, develop a formulation for one of the following ill-structured ("wicked") problems. No matter which of these you choose, you should follow the steps of wicked problem formulation covered in the reading. Note that since these problems have no definitive formulation, there is no single correct answer for any of these wicked problems. We do not expect you to become experts at your selected wicked problem. Rather, be thoughtful about the problem, spend at least an hour or two (each partner) researching your chosen problem, and then follow the formulation process outlined in the reading to show how you would approach the application of AI state-space search techniques to the real-world problem.

1. Finding a vaccine for Covid-19
2. Avoiding a severe economic depression in the United States (or in a given state)
3. Preventing large-scale wildfires
4. Ensuring large public protests remain peaceful
5. Eliminating systemic inequality in the US
6. Ensuring a safe and fair election in the US
7. Preventing disinformation on social media
8. Resettling climate change refugees
9. Reducing plastic waste
10. Minimizing algorithmic bias in software design

Indicate in your report, under Part B, which wicked problem you are formulating. (This will repeat the info you provided in the heading.)

After you have formulated your wicked problem (or along with it), do the following.

- Create an evaluation function Q(s) that maps each state of your state space to a non-negative real number that represents how good that state is in relation to the goal criteria.

  In your report, with a subheading "Evaluation Function", explain what Q(s) measures or computes and give some examples of states and their Q values.

- Demonstrate that your problem formulation works by showing some alternative candidate solutions and the derivations (operators and state sequences) that produce them. At the very least, each operator should be demonstrated by giving "before-and-after" state pairs with an explanation of what has changed. Give these sequences in your report under a subheading "Demonstration sequences".
- Next, in the report under a subheading "Summary", write a summary that explains

what you attempted to capture in your problem formulation, and the extent to which you believe you were successful.

You do not have to create any admissible heuristic functions for this wicked problem.

References You should cite at least two references that support the modeling and decision-making you did to formulate your problem in Part B.

Partnership Retrospective This is an option available to teams of two. It's worth 5 points of extra credit. If you are in a partnership and wish to get this extra credit, create a subheading "Partnership Retrospective". Then, explain how the work was broken out by partner (or who did most of each part). Next, describe what worked well with the partnership. After that, describe what did not work as well in the partnership. Finally, each team member should write, in her or his own words, what she or he learned during the project, including what, if anything is due to working in a partnership.

(Late addition to this specification...) After submitting your files, each person should answer five questions in the A2 WebQ survey about your own experience in the partnership. Be sure you do this before midnight on Friday, October 30.

Turn-In Instructions Turn in your Python files, named as **`a2ff.py`**, and **`a2wp.py`**. Also, turn in a PDF file **`a2report.pdf`** that represents your report. Turn these in via GradeScope. Only the partner whose last name comes first alphabetically, should turn in the files.

Updates and Corrections

If needed, updates and corrections will be posted here, and/or in ED.