Assignment 2: Basic Problem-Solving Techniques

CSE 415: Introduction to Artificial Intelligence
The University of Washington, Seattle, Winter 2019

Due Wednesday, January 23 at 23:59 PM.

Introduction

This assignment begins a series of three assignments the cover state-space search in depth. We start by introducing a framework for the classical theory of problem solving. This will prepare the way to study the use of heuristics for state-space search in Assignment 3.

Procedure

1. Begin this assignment by reading the 1st chapter of "Applying AI in Problem Solving" (linked from the Readings page).

2. Next, download the starter code for this assignment. Un-tar it using a standard file archiving tool such as tar, WinZip, Windows Compressed Folders, etc.

3. Examine each of the two problem formulation files provided. One is the Missionaries and Cannibals problem. The other is the Towers of Hanoi problem. Each is in its own folder. An additional folder is provided, FarmerFox, and you will be writing one new problem formulation in that folder. In that folder is a file called `put_your_formulation_here`. You'll remove that file and create a new one with the name FarmerFox.py.

In the given formulation files, pay particular attention to how the State class is defined and how the operators are established. We will be talking about this code in class.

4. An interactive solving client is provided. This is implemented in the file Int_Solv_Client.py. Try running the client with Towers of Hanoi by using the following command in a Linux, Darwin, or Cygwin command shell. To do this, cd into the TowersOfHanoi folder, and then issue the command:

```
python3 ../Int_Solv_Client.py TowersOfHanoi 3
```

5. Create your own problem formulation file for the "Farmer, Fox, Chicken, and Grain" problem covered in class. Using the interactive solving client, demonstrate the use of your formulation to create a solving-session log. You can simply capture your screen to show the log.

6. Starting with the starter-code file ItrDFS.py, which implements a loop-based depth-first search method ("DFS" in the following), implement the following two additional search algorithms: BFS (Breadth-First Search), and IDDFS (Iterative-Deepening Depth-First Search). Make sure that these implementations keep track of PREDECESSOR links, and they can report a shortest path from start to goal, for whatever problem they are applied to.

7. Compare DFS, BFS, and IDDFS on the following problems: (i) Missionaries and Cannibals,

(ii) Farmer, Fox, Chicken, and Grain, and (iii) 4-Disk Towers of Hanoi. For each combination of algorithm and problem, report the following: (a) the path found from start to goal, (b) the length of the path, (c) the number of nodes expanded (i.e., removed from OPEN and had successors generated). Create a file **BlindSearches.pdf** that contains a table showing these results in a clear, easy-to-read manner. Your Towers-of-Hanoi paths can be shown outside the table, since they can be relatively long. This file can be created with Microsoft Word, Google Docs, or similar tool.

Keep in Mind

Here are some ideas to keep in mind for this assignment. The problem format we are using not only permits the "manual solving" that you do with the Interactive Solving Client. It also permits automatic solving, such as with DFS, BFS, and IDDFS. Thus it is important that you follow the problem structure correctly. Certain methods are required in the State class, including __init__, __eq__, __hash__, __str__, and copy. The operators must have the correct structure, as well. The nice thing about the Interactive Solving Client is that it provides a pretty good means to test whether your formulation will be OK.

Even if you are familiar with Python, there may be some constructs in the formulation files that you are not familiar with. You can read up on list comprehensions (p.31 of the first reading) and lambda expressions (pp. 47-52 of that reading). We'll also cover them in class.

What to Turn In

Turn in four files: (a) `<YourUWNetID>_Farmer_Fox.py` (b and c) `BFS.py IDDFS.py` and (d): (b and c) `BlindSearches.pdf`

Each of the Python files should start with a multiline comment in the following format with your own information rather than the sample information given here:

```
'''jnsmith321_Farmer_Fox.py
by John Nathan Smith

Assignment 2, in CSE 415, Winter 2019.

This file contains my problem formulation for the problem of
the Farmer, Fox, Chicken, and Grain.
'''
```

The turn-in method is UW Canvas.

Updates and Corrections

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