Assignment 1

CMSC 471 (03/01) — Artificial Intelligence

Item	**Summary**
Assigned	Monday February 8th
Due	Sunday February 21st, 11:59 PM Baltimore time
Topic	Search
Points	85

In this assignment you will gain experience with core AI search techniques.

You are to *complete* this assignment on your own: that is, the code and writeup you submit must be entirely your own. However, you may discuss the assignment at a high level with other students or on the discussion board. Note at the top of your assignment who you discussed this with or what resources you used (beyond course staff, any course materials, or public Piazza discussions).

The following table gives the overall point breakdown for this assignment.

Question	1	2	3
Points	25	30	30

Language and External Resources You may use whatever language you prefer, however we must be able to call your program as specified in the assignment. If necessary, we must be able to compile it as well. You may use and reference code from `https://github.com/aimacode`, though acknowledge if you do so. Failure to do so, or the use of other external resources without prior written permission from the instructor, will be considered an academic integrity violation and result, in a minimum, in a 0 on this assignment.

What To Turn In You must turn in two items:

1. A writeup in PDF format that answer the questions.
2. A single `zip` or `tar.gz` file containing all code, environment files (if applicable) and execution instructions necessary to replicate your output.

As part of your submission, be sure to include specific instructions on how to build (compile) your code. Answers to the following questions should be long-form. Provide any necessary analyses and discussion of your results.

How To Submit Submit the assignment on the submission site:

https://www.csee.umbc.edu/courses/undergraduate/471/spring21/01_03/submit.

Be sure to select "`Assignment 1`."

Questions

1. (**25 points**) From Ch 3.11 of Poole and Mackworth, answer the following two questions (reproduced below for ease):

   (a) Question 2: Which of the path-finding search procedures are fair in the sense that any element on the frontier will eventually be chosen? Consider this question for finite graphs without cycles, finite graphs with cycles, and infinite graphs (with finite branching factors).

   (b) Question 5: Draw two different graphs, indicating start and goal nodes, for which forward search is better in one and backward search is better in the other.

2. (**30 points**) This problem is set-up for the next problem, which will involve implementation of search heuristics.

   You may remember *systems of equations* from your math classes. They are M different equations of N variables. For example,

   $$\begin{aligned} 5x_1 + 4x_2 &= 9 \\ 9x_2 &= 9 \end{aligned}$$

   is a system of $M = 2$ equations with $N = 2$ unknown variables (x_1 and x_2). The goal is to find values of the variables x_1 and x_2 that make the equations in the system true. Solving systems of equations when the variables $\mathbf{x}$ are real-valued is relatively straight-forward: solving them when $\mathbf{x}$ are restricted to be binary values (that is, 0 or 1) is *hard*. That is, while it may not seem intuitive, allowing each x_n, for $n = 1 \ldots N$, to be only either 0 or 1 makes the problem difficult to solve efficiently.

   While it's convenient if $M = N$, it is not required. For example,

   $$\begin{aligned} 5x_1 + 4x_2 &= 5 \\ 9x_3 &= 9 \end{aligned}$$

   is a system of $M = 2$ equations with $N = 3$ unknown variables. In general, a system of equations can be written as

   $$\begin{aligned} a_{1,1}x_1 + a_{1,2}x_2 + \ldots + a_{1,N}x_N &= y_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + \ldots + a_{2,N}x_N &= y_2 \\ &\cdots \\ a_{M,1}x_1 + a_{M,2}x_2 + \ldots + a_{M,N}x_N &= y_M. \end{aligned} \tag{1}$$

   The $a_{m,n}$ are the coefficients, and y_m are the required output values. If we use matrix-vector notation, where the matrix (2-d array) A stores the coefficients, the one dimensional vector (array) $\mathbf{x}$ stores the variables and the one dimensional vector (array) $\mathbf{y}$ stores the output values

   $$A = \begin{pmatrix} a_{1,1} & a_{1,2} & \ldots & a_{1,N} \\ a_{2,1} & a_{2,2} & \ldots & a_{2,N} \\ & & \ldots & \\ a_{M,1} & a_{M,2} & \ldots & a_{M,N} \end{pmatrix} \qquad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \ldots \\ x_N \end{pmatrix} \qquad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \ldots \\ y_M \end{pmatrix}$$

then with matrix-vector multiplication we can write Eq. (1) compactly as $A\mathbf{x} = \mathbf{y}$.

(a) For the following system of equations with binary variables, identify A and $\mathbf{y}$. Additionally, provide the solution for $\mathbf{x}$ (as a hint, the variables are $\mathbf{x} = (x_1, x_2, x_3)$).

$$\begin{aligned} 5x_1 + 4x_2 &= 4 \\ 9x_1 + 3x_3 &= 0 \end{aligned} \tag{2}$$

(b) Construct an unsatisfiable system of equations with binary variables. That is, write out A and $\mathbf{y}$ such that there's no possible binary $\mathbf{x}$ that will make $A\mathbf{x} = \mathbf{y}$ true. Briefly explain why this is an unsatisfiable system.

(c) Construct a system of equations with binary variables that has multiple possible solutions. That is, write out A and $\mathbf{y}$ such that there is more than one binary $\mathbf{x}$ that will make $A\mathbf{x} = \mathbf{y}$ true. Provide at least two of these values of $\mathbf{x}$.

(d) If you were to construct an agent for this problem, identify/describe

  (i) the environment

  (ii) the state

  (iii) actions your agent may take, along with any possible constraints on those actions

  (iv) possible costs associated with your actions

  (v) the goal/goal test predicate.

  As a hint, there is not a correct answer for the possible costs. Additionally, the goal/goal test predicate must not assume any "correct" values of $\mathbf{x}$: it must work for *any* binary $\mathbf{x}$.

(e) For the system provided by Eq. (2), use your representation from (d) to provide a trace of breadth-first search. Use $\mathbf{x} = (0, 0, 0)$ as your initial state.

3. (**30 points**) In this problem, you are to complete the implementation of a solver for this problem. There is some starter Python code available at `https://www.csee.umbc.edu/courses/undergraduate/471/spring21/01_03/materials/a1/code` but you are not required to use this code (or Python, even). However, your code **must** be runnable via the bash script

   `https://www.csee.umbc.edu/courses/undergraduate/471/spring21/01_03/materials/a1/code/test_ilp.bash`.

   The specification for the system of equations will be provided by a JSON file. This JSON file is a dictionary with two keys: `A`, which maps to a 2-dimensional array of coefficients, and `y`, a 1-dimensional array of output values. See

   `https://www.csee.umbc.edu/courses/undergraduate/471/spring21/01_03/materials/a1/code/q2a.json`.

   for an example encoding of Eq. (2). (We *will* test your program on other systems with binary variables. For full credit, your program must work appropriately for these other files.)

   Specifically:

(a) Implement a basic agent that is capable of solving a provided system of linear equations with binary variables. For now, use a uniform unit cost function g_1 and a trivial heuristic (i.e., $h = 0$). Demonstrate that your solver can find the correct solution by applying it to the system in Eq. (2); use the search algorithm of your choice.

(b) Implement a second cost function g_2. Let $c_{0\to1}$ = the number of variables that an action changes from 0 to 1, and let $c_{1\to0}$ = the number of variables that an action changes from 1 to 0. Then compute g_2 as

$$g_2 = \exp\left(c_{0\to1} - c_{1\to0}\right). \tag{3}$$

this cost function should compute the total number of additional variables that change from 0 to 1, minus the number of variables that change from

(c) Implement a non-trivial heuristic h of your choice. *Clearly* describe your heuristic in prose in your writeup. Identify whether it is admissible or not, and provide proof of this. For each of the previous cost functions g_1 and g_2, use your heuristic in an A^* search. Provide output from each combination of $g_1 + h$ and $g_2 + h$, and comment on the differences, if any, your heuristic makes on the solutions found (or the time/computation required for those solutions).

(d) Using A^*, test out each of $g_1 + 0$, $g_2 + 0$, $g_1 + h$ and $g_2 + h$ on the file
`https://www.csee.umbc.edu/courses/undergraduate/471/spring21/01_03/materials/a1/code/7x10.json`
Provide the output for each run, and again comment on the similarities or differences in the solutions found, or the required time/computation to find those solutions.

While not required, I **highly** suggest you also test your program on the examples you came up with in 2(b) and 2(c).

For this assignment, you may use code provided by the `aimacode` github organization: `https://github.com/aimacode`. There is a `search` file, which contains the implementation of the different search algorithms. If you are coding in Python, this will be `search.py` in the `aima-python` repo. You are welcome to use these pre-implemented search functions, but you are not required to.