DEPARTMENT OF
COMPUTING SCIENCE
UNIVERSITY OF ALBERTA

CMPUT 366: Assignment #1

Due at 10pm on October 3, 2019

Abstract

For this assignment use the following consultation model:

- you can discuss assignment questions and exchange ideas with other CMPUT 366 students;
- you must list all members of the discussion in your solution;
- you may **not** share/exchange/discuss written material and/or code;
- you must write up your solutions individually;
- you must fully understand and be able to explain your solution in any amount of detail as requested by the instructor and/or the TAs.

Anything that you use in your work and that is not your own creation must be properly cited by listing the original source. Failing to cite others' work is plagiarism and will be dealt with as an academic offence.

First name: Shoyang

Last name: Zhou

CCID: 1410390@ualberta.ca

Collaborators: ____

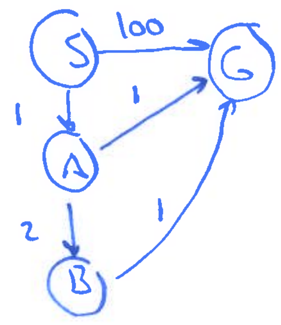
Your mark: ____ out of 140

1

[15 points] Construct a search graph with **no more than 10 nodes** for which all of the following are true:

1. Least-cost first search returns an optimal solution.
2. Breadth-first search returns the highest-cost solution.
3. Depth-first search returns a solution whose cost is strictly less than the highest-cost solution and strictly more than the least-cost solution.

Feel free to include multiple goal nodes in your graph. Be sure to list the start and goal node(s), all edge costs and all edge directions (if your graph is directed). Draw the graph as well.



S = start

G = goal

* Assume neighboring orders nodes favor nodes A, B in DFS, BFS

2

[5 points] List the paths that are removed from the frontier by a depth-first search of the search problem you gave for Question 1, in the order in which they are removed from the frontier. Stop the trace when the path removed ends in a goal state.

Iter

0 Expand S_0

1 Expand ($S_0 \rightarrow A$)

2 Expand ($S_0 \rightarrow A \rightarrow B$)

3 Expand ($S_0 \rightarrow A \rightarrow B \rightarrow G$)

Cost = 4

3

[5 points] List the paths that are removed from the frontier by a breadth-first search of the search problem you gave for Question 1, in the order in which they are removed from the frontier. Stop the trace when the path removed ends in a goal state.

Iter.
0 Expand So
1 Expand So → A
2 Expand So → G

4

Cost = 100

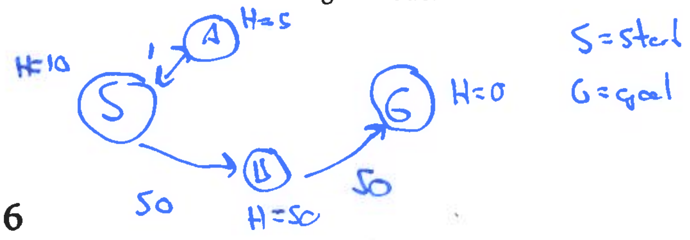
[5 points] List the paths that are removed from the frontier by a least-cost first search of the search problem you gave for Question 1, in the order in which they are removed from the frontier. Stop the trace when the path removed ends in a goal state.

Iter.
0 Expand So
1 Expand So → A
2 Expand So → A → G

5

cost = 3

[2 points] Come up with a *solvable* four-node search graph on which greedy best-first search (PM 3.6) never reaches the goal. The four nodes should include the start node and the goal node. Draw the four-node graph below. Label each edge with its cost and each node with its heuristic value. The heuristic must be admissible. Mark the start node and the goal node.



6

[2 points] Trace the greedy best-first search on the search problem you came up with for Question 5 and list the paths that get removed from the frontier. Stop the trace after showing that the algorithm will never reach the goal node.

Iter
0 Expand S
1 Expand S → A
2 Expand S → A → S
3 Expand S → A → S → A . . . Cycle reached

7

[6 points] A farmer needs to move a hen, a fox, and a bushel of grain from the left side of the river to the right using a raft. The farmer can take one item at a time (hen, fox, or bushel of grain) using the raft. The hen cannot be left alone with the grain, or it will eat the grain. The fox cannot be left alone with the hen, or it will eat the hen. For example, the farmer cannot move from one side x of the river to the other side y if it would mean leaving the fox and hen together on side x. The farmer can load an item onto the raft, move the raft from one side of the river to the other, or unload an item from the raft. The farmer wants to move the items with the fewest number of trips across the river as possible.

Classify this problem using the primary representational dimensions covered in the lectures.

Uncertainty: Deterministic → Initial state and all actions are deterministic hence we are able to fully observe future states.

Interaction: Offline → Agent can plan out entire action in advance.

# of Agents: Single → Only the farmer with the raft can alter the world state.

8

[20 points] Represent the problem in Question 7 as a graph search problem: define the set of states/nodes, the start node, the goal node(s). Define the edges via defining neighbours of each state obtained via the farmer's actions. Define costs of each edge. You do not have to draw the graph or explicitly list all nodes and edges.

- Represent each state as tuple of 2 sets
  - Node.L = left side of river
  - Node.R = right side of river.
- Represent each of the following as a token to be in the sets
  - R = raft / farmer
  - F = Fox
  - H = Hen
  - G = Grain
- Start Node:
  - Node.L = { R, F, H, G }
  - Node.R = { }
- End node:
  - Node.L = { }
  - Node.R = { R, F, H, G }

- Edges
  - Edges cost 1 denoting a trip across the river
  - Edges only connect between two valid nodes (ie, nothing is eaten)
  - Edges move either the ~~raft~~ raft / farmer (R) or the raft / farmer (R) plus an additional item from their (the raft's) current side.
  - A valid node is such that the ~~state~~ side without the raft / farmer cannot contain the Fox-Hen or Hen-Grain pair.

9

[5 points] What is the branching factor for your graph from Question 8? Justify your answer.

- Lower Bound = 1
  - Since actions are reversible, going from valid node to valid node ensures at least one arc.
- Upper Bound = 3, Consider cases of there being 1,2,3 objects being on side of raft.
  - Case 1: raft is on side w/ one item.
    - The raft or the item can be moved,
    - 2 options
  - Case 2: Two items
    - if both items compatible
      - Could take nothing across
      - Could take one obj across
      - ~~[illegible]~~ Two options
    - if ~~one~~ items not compatible:
      - Must take one item across
      - options = 1 or 3
  - Case 3: Three items
    - There must be one item valid check only between the three items given the two relationships stated in the problems, hence there is only 1 option here

10

[10 points] Construct a non-constant admissible heuristic for the problem in Question 7.

- Use the ~~[illegible]~~ length of Node.L ie |Node.L|)
- Rationale
  - In general getting things to the right side of the river indicates progress.
  - To get a ~~[illegible]~~ single item across regardless requires atleast one trip hence we do not over estimate.

11

[5 points] Prove that your heuristic for Question 10 is indeed admissible.

- Consider that to shuttle one item across, regardless takes at least one trip.
- Consider that to shuttle two items across, takes at least 3 trips.
- Taking into the problem's constraint, we note that it is necessary to move some items back and forth more than once.

① • If an item is moved only once ~~[illegible]~~ then its cost is at least 1, hence we do not over estimate

② • If an item is moved more than once then it is similar to moving two items, which we underestimate

- Since moving the three items, the cost is cumulative, we note that decomposing moving the three items into their individual cases notes that they follow ① and ②.

12

[60 points] Implement your representation from Question 8 and heuristic from Question 10 in Python 3 by editing the `River_problem` class in the provided `riverProblem.py`. We will run your code with the command `python3 riverProblem_run.py`. Your code must complete within 2 minutes for full marks.[1]

Submit all of your code (including provided boilerplate files) in a single zip file.

Submission

The assignment you downloaded from eClass is a single ZIP archive which includes this document as a PDF *and* its LaTeX source as well as Python files needed for Question 12. You are to unzip the archive into an empty directory, work on the problems and then zip the directory into a new single ZIP archive for submission.

Each assignment is to be submitted electronically via eClass by the due date. Your submission must be a single ZIP file containing:

1. a single PDF file with your answers;
2. file(s) with your Python code.

To generate the PDF file with your answers you can do any of the following:

- insert your answers into the provided LaTeX source file between `\begin{answer}` and `\end{answer}`. Then run the source through LaTeX to produce a PDF file;
- print out the provided PDF file and legibly write your answers in the blank spaces under each question. Make sure you write as legibly as possible for we cannot give you any points if we cannot read your hand-writing. Then scan the pages and include the scan in your ZIP submission to be uploaded on eClass;
- use your favourite text processor and type up your answers there. Make sure you number your answers in the same way as the questions are numbered in this assignment.

[1]It should really run in far less time than this.