CMPUT 261, Fall 2023
Assignment #1

Due: Thursday, February 1/2024
Total points: 124

For this assignment use the following consultation model:

1. you can discuss assignment questions and exchange ideas with other *current* CMPUT 261 students;
2. you must list all members of the discussion in your solution;
3. you may **not** share/exchange/discuss written material and/or code;
4. you must write up your solutions individually;
5. 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: Mikael

Last name: Nineza

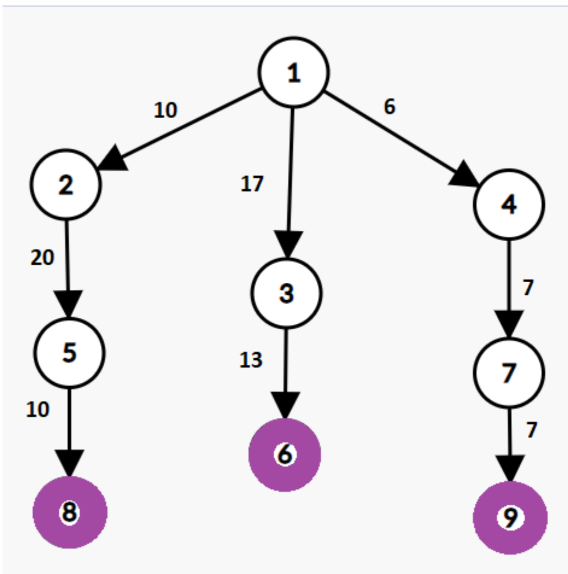
CCID: lnineza`@ualberta.ca`

1. (**Uninformed search**; 24 points)

   (a) [**14 points**] Construct a search graph with **no more than 10 nodes** for which all of the following are true:

   i. Least-cost search returns an optimal solution.
   ii. Depth-first search returns the highest-cost solution.
   iii. Breadth-first search returns a solution whose cost is strictly less than the highest-cost solution and strictly more than the least-cost solution.

   Note that this means your search graph must have at least 3 solution paths of differing costs. (You are allowed to have multiple goal nodes.) Be sure to include and formally describe each component of the search graph. If necessary for concreteness, specify the order in which successor paths are added to the frontier by each algorithm.



   They all start with 1 (the start node) in the frontier, they will remove 1 and test it to see if its the goal node then will add $\{\langle 1,2\rangle, \langle 1,3\rangle, \langle 1,4\rangle\}$ to the front of the frontier.

   LCS will remove the cheapest path, $\langle 1,4\rangle$, check if its the goal node , and put its neighbour, $\langle 4,7\rangle$at the start of the frontier. It will remove $\langle 4,7\rangle$from frontier, check if it's a goal node, and then add $\langle 7,9\rangle$. It will remove $\langle 7,9\rangle$, check if it's a goal node. It is so we end the search. This solution cost 20.

   DFS will remove the newest path, $\langle 1,2\rangle$, check if its the goal node , and put its neighbour, $\langle 2,5\rangle$at the start of the frontier. It will remove $\langle 2,5\rangle$from frontier, check if it's a goal node, and then add $\langle 5,8\rangle$. It will remove $\langle 5,8\rangle$, check if it's a goal node. It is so we end the search. This solution cost 40.

   BFS will remove the oldest path, $\langle 1,2\rangle$, check if its the goal node , and put its neighbour, $\langle 2,5\rangle$at the end of the frontier. It will remove $\langle 1,3\rangle$from frontier, check if it's a goal node, and then add its neighbour, $\langle 3,6\rangle$at the end of the frontier. It will remove $\langle 1,4\rangle$, check if its the goal node , and put its neighbour, $\langle 4,7\rangle$at the end of the frontier. It will remove $\langle 2,5\rangle$from frontier, check if it's a goal node, and then add its neighbour, $\langle 5,8\rangle$at the end of the frontier. It will remove $\langle 3, 6\rangle$, check if its the goal node. It is so we will end the search. This solution cost 30.

   (b) [**5 points**] List the paths that are removed from the frontier by a depth first search of the problem you specified in part (1a), **in the order in which they are removed**. The algorithm should stop as soon as it returns a solution. In the order they are removed from the frontier from first to last: $\langle 1,2\rangle$,$\langle 2,5\rangle$,$\langle 5,8\rangle$.

(c) **[5 points]** List the paths that are removed from the frontier by a least cost first search of the problem you specified in part (1a), **in the order in which they are removed**. The algorithm should stop as soon as it returns a solution.

In the order they are removed from the frontier from first to last: $\langle 1,4 \rangle$, $\langle 4,7 \rangle$, $\langle 7,9 \rangle$.

2. (**Heuristic search**; 100 points)

A group of researchers are being chased by zombies through a dark forest late at night. They come to a rope bridge over a ravine. If they can all get over the bridge before the zombies arrive, they will survive.

At most two people can cross at a time. A person or pair of people can only cross when they have a flashlight with them. The group has only a single flashlight among them, so one person must bring the flashlight back across the bridge to the starting side before anyone else can cross. Each pair moves at the pace of its slowest member; i.e., the Undergrad and the Professor will take 10 minutes to cross if they go together.

Each person moves at a different pace:

- The Undergrad can cross the bridge in 1 minute
- The Grad Student can cross the bridge in 2 minutes
- The Postdoc can cross the bridge in 5 minutes
- The Professor can cross the bridge in 10 minutes

How can the whole group get to the other side of the bridge in the shortest possible time?

(a) [**20 points**] Represent this problem as a search graph. Be sure to include and formally describe each component of the search graph.

This problem can be represented as a search graph problem in the following way:

The nodes represent the current state in the form:

$V = \{(x, y, t, f) | x \in \mathscr{P}(U, G, PD, PF), y \in \mathscr{P}(U, G, PD, PF) \backslash x, t \in \mathbb{Z}, f \in \{0, 1\}\}$

- x and y are the sets that represents who is on the original side of the bridge and who is on the other side of the bridge respectively.

- t represents the cost of the arc that we took to reach this state (e.i. the time it took to cross the bridge)

- f represents the current position of the flashlight. f = 0 if it's at the original side and f = 1 if it's on the other side of the bridge.

A path will be in the form $[V_1, V_2, V_3, V_4, ...]$ where (V_n, V_{n+1}) is an arc

The start state is [{U, G, PD, PF}, {}, 0, 0] and the end state is [{}, {U, G, PD, PF}, t*, 1, where t* is the time for the last arc.

There are two types of actions:

The first one is when the flashlight is on the original side, then two people will cross the bridge. The second type is when the flashlight is on the other side, then one person will have to cross the bridge to bring back the flashlight to the original side so that whoever is left there can cross.

(b) [**5 points**] What is the forward branching factor for your representation from part (2a)? Justify your answer.

We have four people and only two of them can cross the bridge so there are $\binom{4}{2} = 6$ possible combinations. The forward branching factor for my representation from part (2a) is 6.

(c) [**10 points**] Construct an admissible heuristic for this problem that is non-constant (i.e., returns different values for at least two states).

Let $h(n)$ be my heuristic which returns the heuristic value of node n. $h(n) = x - (x//6)$, where x is the time it takes for the slowest person on the original side to cross the bridge. Note that $(x//6)$ is integer division, as in we truncate the floating point number if we get one.

(d) **[5 points]** Argue that the heuristic from part (2c) is admissible.

My heuristic will always underestimate or be equal to the true cost to reach a goal node. Proof:

Let's relax the problem: Now everyone can cross at the same time. Then the cost for this will be x, the time it takes for the slowest person to cross the bridge but my heuristic $= x - (x//6) <= x$ as long as x $= 0$ which is the case here (you cannot cross the bridge in negative amount of time). Since my heuristic is admissible in a relaxed version of the problem, then it is also admissible for this problem.

(e) **[60 points]** Implement your representation from part (2a) and heuristic from part (2c) in Python 3 by editing the `Zombie_problem` class in the provided `zombie.py`. We will run your code with the command `python3 zombie.py`. Your code must complete within 2 minutes for full marks.[1]

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

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 a Python file needed for Question 2e. 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.