
CMPUT 261, Fall 2026
Assignment #1
SOLUTIONS GUIDE

1. (Uninformed search; 22 points)

- (a) [12 points] Construct a search graph with **at least 5 and no more than 10 nodes** for which all of the following are true:

- Least-cost search returns an optimal solution.
- Depth-first search returns a non-optimal solution.
- Breadth-first search returns the same non-optimal solution as depth-first search.

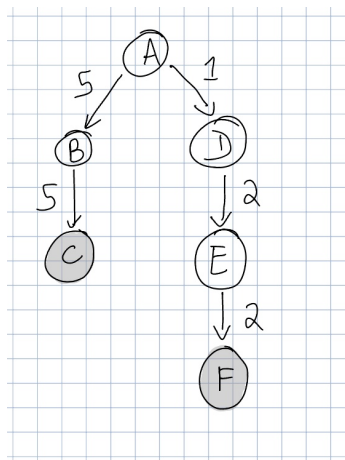
Assume each algorithm stops as soon as it returns a solution. Note that this means your search graph must have multiple solution paths of differing costs. (You are allowed to have multiple goal nodes.) Be sure to include and formally describe each component of the search graph. If necessary for concreteness,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

Consider the graph below.

The start node is A . The goal nodes are $\{C, F\}$. Breadth-first search adds paths to the frontier in alphabetical order (thus, $\langle A, B \rangle$ will be removed from the frontier before $\langle A, D \rangle$). Depth-first search adds paths to the frontier in reverse alphabetical order (thus, $\langle A, B \rangle$ will be removed from the frontier before $\langle A, D \rangle$).

The arcs are:

- A has an arc of cost 5 to B , and an arc of cost 1 to D ;
- B has an arc of cost 5 to C , which is a goal node;
- D has an arc of cost 2 to E ; and
- E has an arc of cost 2 to F , which is a goal node.



- (b) [5 points] List the paths that are **removed** from the frontier by a breadth-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.

The paths removed are:

- $\langle A \rangle$
- $\langle A, B \rangle$
- $\langle A, D \rangle$
- $\langle A, B, C \rangle$

- (c) [3 points] List the paths **added** to the frontier by a least-cost search of the problem you specified in part (1a), **in the order in which they are added**.

- $\langle A \rangle$
 - $\langle A, B \rangle$
 - $\langle A, D \rangle$
 - $\langle A, D, E \rangle$
 - $\langle A, D, E, F \rangle$
- (d) [2 points] Alter the problem you specified in part (1a) such that depth-first search is **not complete**. You are **not** allowed to add or remove any nodes. **Simply adding an arc from B back to A to create a cycle will do.**

2. (Heuristic search; 105 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 20 minutes to cross if they go together.

Each person moves at a different pace:

- The Undergrad can cross the bridge in 2 minutes
- The Grad Student can cross the bridge in 4 minutes
- The Postdoc can cross the bridge in 10 minutes
- The Professor can cross the bridge in 20 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.

First I define the search problem: *States* are tuples (L, R, p) , where L is a set of the crossing times of the people on the left side of the ravine, R is a set of the crossing times of the people on the right side of the ravine, and $p \in \{0, 1\}$ indicates whether the flashlight is on the left or right side of the ravine.

Everyone starts on the left of the ravine with the flashlight, so $(\{2, 4, 10, 20\}, \{\}, 0)$ is the only *start state*.

The *goal* function returns true once everyone is at the right side:

$$goal(L, R, p) = \begin{cases} 1 & \text{if } R = \{2, 4, 10, 20\} \\ 0 & \text{otherwise.} \end{cases}$$

At any given state, any group of size 1 or 2 can travel from the side of the ravine that has the flashlight to the other side, at which point those travellers will join the group on the other side (with the flashlight). So the available actions at state (L, R, p) are

$$A(L, R, p) = \begin{cases} \{T \subseteq L \mid 0 < |T| \leq 2\} & \text{if } p = 0, \\ \{T \subseteq R \mid 0 < |T| \leq 2\} & \text{otherwise.} \end{cases}$$

The *successors* are therefore:

$$succ(L, R, p) = \begin{cases} \{(L \setminus S, R \cup S, 1) \mid S \in A(L, R, p)\} & \text{if } p = 0, \\ \{(L \cup S, R \setminus S, 0) \mid S \in A(L, R, p)\} & \text{otherwise.} \end{cases}$$

The *cost* of a movement is the length of time required to get the slowest member of the group to the other side:

$$\begin{aligned} cost((L, R, 0), (L \setminus S, R \cup S, 1)) &= \max(S) \\ cost((L, R, 1), (L \cup T, R \setminus T, 0)) &= \max(T). \end{aligned}$$

The search graph is a graph $G = (N, A)$, where

$$N = \{(L, R, p) \mid L, R \subseteq \{2, 4, 10, 20\}, L \cap R = \emptyset, p \in \{0, 1\}\}$$

and $A = \{(u, v) \mid u \in N, v \in \text{succ}(u)\}$, and the start node, cost and goal functions are as described above.

Describing the search graph directly instead of first defining the search problem and then the graph in terms of the search problem is acceptable for full marks. Describing the search problem without defining the graph is a **1 point** deduction only.

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

10.

Suppose the set of people on the side of the ravine with the flashlight is S . Then there is one successor for each of the $|S| = n$ singleton subsets of S , and one successor for each of the $\binom{n}{2}$ two-element subsets of S . There are only 4 people in total, so there are at most

$$4 + \binom{4}{2} = 4 + 6 = 10$$

successors from any given state.

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

Suppose the current state is (L, R, p) . Then we need to get all of the people in L to the right of the ravine. If the flashlight is on the right of the ravine, then we must spend $\sum_{i \in L} i$ time (one trip for each person, each trip takes at least as long as the person's required time). If the flashlight is on the left of the ravine, then answer is the same, except that we can send the longest-delay person and the second-longest delay person together, so the total delay is $(\sum_{i \in L} i) - \text{second}(L)$, where $\text{second}(L)$ is the second-longest delay in L if there are at least 2 elements in L , and 0 otherwise. So

$$h(L, R, p) = \begin{cases} \sum_{i \in L} i & \text{if } p = 1, \\ (\sum_{i \in L} i) - \text{second}(L) & \text{otherwise.} \end{cases}$$

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

If the flashlight is on the right, then for each person d on the left, we must spend at least d minutes to get the person from left to right (assuming that the person who brings the flashlight back each time uses 0 minutes on the return trip, which is smaller than any of the actual delays).

If the flashlight is on the left, then if we send the slowest two people first, they will take $\max(L)$ minutes. The remaining time is then at least d minutes for each of the people d who were left behind on the left side (again assuming that it takes 0 minutes to return the flashlight to the left).

- (e) **[5 points]** Is the heuristic from part (2c) consistent? Justify your answer.

Recall that a heuristic is *consistent* if it satisfies the following condition:

$$\forall n, n' \in N, \text{paths } \langle n, \dots, n' \rangle : h(n) \leq \text{cost}(\langle n, \dots, n' \rangle) + h(n'). \quad (1)$$

I claim that h as defined in part (2c) is *not* consistent. To prove this, it is sufficient to show a single violation of (1). Consider the path $\langle n, n' \rangle$ where $n = (\{4, 10, 20\}, \{2\}, 0)$ and $n' = (\{2, 4, 10, 20\}, \{\}, 1)$. Plugging in values, we have

$$\begin{aligned} h(n) &= 4 + 10 + 20 = 34 \\ h(n') &= 2 + 4 + 10 + 20 - 10 = 26 \\ \text{cost}(\langle n, n' \rangle) &= 2. \end{aligned}$$

But $34 = h(n) > \text{cost}(n, n') + h(n') = 28$, violating (1). Thus h is not consistent.

- (f) **[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`. You can run the solution with the command `python3 zombie.py`. Your code should complete in far less than 2 minutes.