CS 471/571 - Fall 2023 Problem Set #1
Bret Brouse Jr. (10/11)

Q1. Word Ladder (30 points)

Wikipedia describes the rules for the word game Word Ladder as: The player is given a start word and an end word (of equal length). In order to win the game, the player must change the start word into the end word progressively, creating an existing word at each step. Each step consists of replacing a single letter. For example, the following is a solution to the Word Ladder puzzle between words "Cold" and "Warm":

$$\text{COLD} \rightarrow \text{CORD} \rightarrow \text{CARD} \rightarrow \text{WARD} \rightarrow \text{WARM}$$

1. (10 pts) Formulate the problem of solving word ladders as a search problem.

Solution. Since we're given both the starting word and ending word, this saves us from having to search the entire alphabet at each step. Thus, employing the rules that we choose letters alphabetically unless we have a preference and we check at each step if the change is in fact a word we can employ the following search:

1. Choose a letter to change to the correct letter.
2. If the change results in a word, repeat step 1 for another letter.
3. If the change does not result in a word, we have a dead-end in the decision tree; return to previous node (or start) and choose another letter to change.
4. Repeat this process until we reach the desired end word and at each depth we have a word.

The steps in the fringe would look something like the following:

S	$COLD$
$S \rightarrow C$	
$S \rightarrow O$	
~~$S \rightarrow L$~~	$COLD \rightarrow CORD$
$S \rightarrow D$	
$S \rightarrow L \rightarrow C$	
~~$S \rightarrow L \rightarrow O$~~	$COLD \rightarrow CORD \rightarrow CARD$
$S \rightarrow L \rightarrow D$	
~~$S \rightarrow L \rightarrow O \rightarrow C$~~	$COLD \rightarrow CORD \rightarrow CARD \rightarrow WARD$
$S \rightarrow L \rightarrow O \rightarrow D$	
$S \rightarrow L \rightarrow O \rightarrow C \rightarrow D$	$COLD \rightarrow CORD \rightarrow CARD \rightarrow WARD \rightarrow WARM$

This of course follows the example above where we know a step-by-step solution to get to the goal. However, since this is one of the ideal routes, another fringe could include more steps and more letters. For instance, one could go from COLD to WIND via some number of steps, then eventually via some other number of steps, get to WARM. An example would be going from COLD $\rightarrow$ CORD $\rightarrow$ CORE $\rightarrow$ ... $\rightarrow$ WARM, provided each stell is actually a word.

This problem becomes generalized when we consider the entire alphabet, words of any size, etc. One would need to extend each letter choice to include every letter of the alphabet and check at each step whether or not the letter change results in a word.

2. (10 pts) Consider a restriction where the words can only use the first seven letters (a-g). Draw the state space graph for converting "bee" to "dad".

Solution. I make the assumption that since "bed" has two edges, one could find their way back to bed via the path not taken. Hence in all except the goal "dad," we have double arrows to represent paths they could take. Unfortunately, this creates infinite loops in the search.

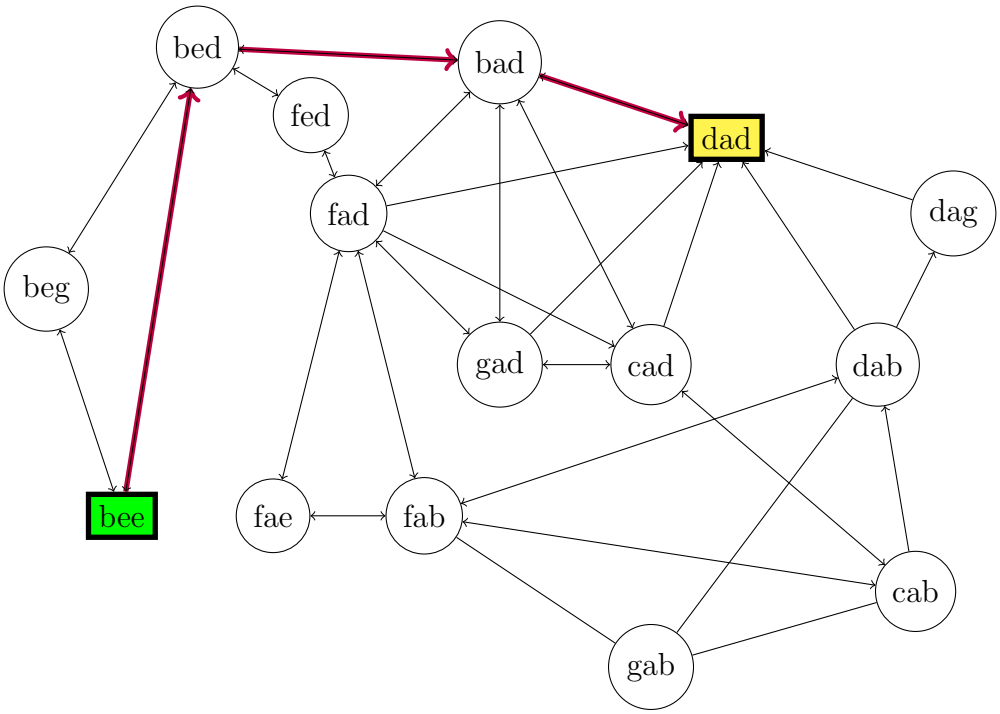


Figure 1: State space diagram for the word ladder bee $\rightarrow$ dad using only the first 7 letters. The ideal path is highlighted in red. Since no rule is explicitly stated that a word can't be used over, endless loops are prevalent in the diagram (a rule that could be fixed easily in application).

3. (10 pts) (Tree Search) Which part of the search tree is expanded by DFS?

Solution. I suppose I'm to assume that we break ties alphabetically. Also, I'm going to try to ignore possible endless loops such as bad → gad → fad → bad. In this case, the tree would expand in the following order first:

$$\text{bee} \to \text{bed} \to \text{bad} \to \text{cad} \to \text{cab} \to \text{dab} \to \text{dad}$$

If we relax the alphabetical rule and instead go by order of letters: change b first if possible, then e, then the second e, followed by alphabetical preference for the whole word, we get:

$$\text{bee} \to \text{bed} \to \text{fed} \to \text{fad} \to \text{gad} \to \text{bad} \to \text{dad}$$

The steps in the former fringe can be seen in Figure 2, where the steps for the latter are left to the reader as an exercise (kidding, I have to make jokes or I'll lose my mind in homework). It's worth noting that with these two rules, some nodes will never be accessed, such as "beg" and "dag." One could argue for different rules for tie-breakers to include these words if necessary.

bee

~~$bee \to bed$~~

$bee \to beg$

~~$bee \to bed \to bad$~~

$bee \to bed \to fed$

~~$bee \to bed \to bad \to cad$~~

$bee \to bed \to bad \to dad$

$bee \to bed \to bad \to fad$

$bee \to bed \to bad \to gad$

~~$bee \to bed \to bad \to cad \to cab$~~

$bee \to bed \to bad \to cad \to dad$

$bee \to bed \to bad \to cad \to fad$

$bee \to bed \to bad \to cad \to gad$

~~$bee \to bed \to bad \to cad \to cab \to dab$~~

$bee \to bed \to bad \to cad \to cab \to fab$

$bee \to bed \to bad \to cad \to cab \to gab$

✓ ~~$bee \to bed \to bad \to cad \to cab \to dab \to dad$~~

$bee \to bed \to bad \to cad \to cab \to dab \to dag$

Figure 2: Fringe steps for the case where the tie-breaker is the edges in alphabetical order. This neglects the ability to reuse a word, avoiding endless loops.

Q2. Graph Search (50 points)

Q2.1. Search Algorithms. (30 points) Consider the following graph. The start node is S and the goal node is G. For the following sub-questions, ties are broken in alphabetical order.

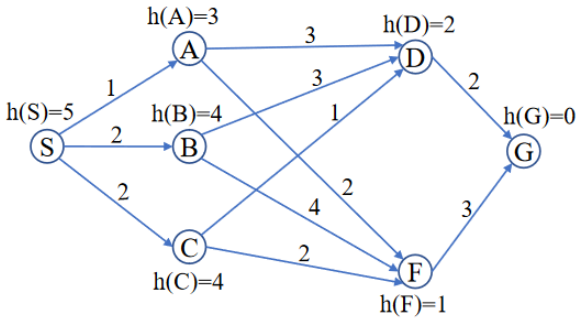


Figure 3: A given state space graph.

1. (10 pts) In what order are states expanded by UCS? What path would UCS return?

Solution. A UCS expands states based on the least expensive *total* path cost to the goal. Here, the "cheapest" path is given by $S \to C \to D \to G$ since the total cost is 5, whereas all other paths have a cost greater than 5. The fringe associated with this expansion is given by Figure 4.

S

$S \to A$

$S \to B$

~~$S \to C$~~

~~$S \to C \to D$~~

$S \to C \to F$

✓ ~~$S \to C \to D \to G$~~

Figure 4: A fringe representing a UCS expansion given Figure 3. UCS takes into consideration the *total* cost of the path to G.

2. (10 pts) In what order are states expanded by Greedy? What path would Greedy return?

Solution. A Greedy search looks for the node that *seems the closest* to the goal based on Euclidean distance. In this case, S first considers A, B, and C. Since the Euclidean distance

is shortest from B to G, the next state expanded is B. Next, D and F are considered. In our diagram above, D is closer to G (since F has a slight $-y$ offset in standard Euclidean geometry) so that state is expanded. Finally, G is selected as the final step. Thus the path returned by Greedy would be $S \to B \to D \to G$. The fringe is represented by Figure 5.

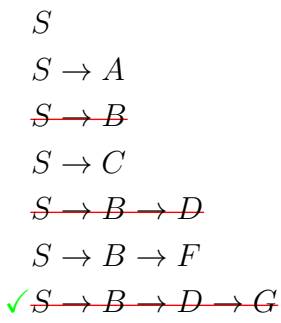


Figure 5: A fringe representing a Greedy expansion of the state space given in Figure 3. Greedy searches choose the edge in the fringe that is *closest to the goal* then chooses that as the next step.

3. (10 pts) In what order are states expanded by A^*? What path would A^* return?

Solution. HERE

Q2.2.Admissibility/Consistency. (20 points)

- (10 pts) Is the above heuristic function h admissible? Explain your answer.

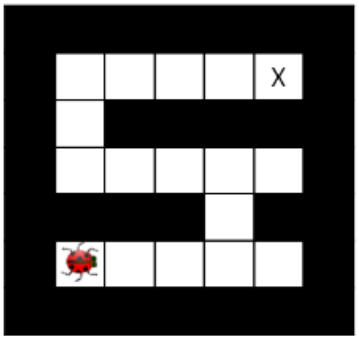
Solution. HERE

- (10 pts) For which TWO states (S, A, B, C, D, F, or G) could you change the heuristic values to make everything consistent? What range of values are possible to make this correction?

Solution. HERE

Q3. Hive Mind: Lonely Bug. (20 points)

You control a single insect in a rectangular maze-like environment with dimensions M × N . Figure below is an example of such environment. The insect must reach a designated target location X, also known as the hive. There are no other insects moving around. At each time step, an insect can either (a) move into an adjacent square if that square is currently free, or (b) stay in its current location. Squares may be blocked by walls, but the map is known. Optimality is always in terms of time steps; all actions have cost 1 regardless of the number of insects moving or where they move.



1. (10 pts) Provide a minimal correct state space representation? What is the state-space size?

Solution. HERE

2. (10 pts) Provide two heuristics which are admissible.

Solution. HERE