CS 3600 – Introduction to Intelligent Systems

Uninformed Search

1. Missionaries and Cannibals.

Missionaries and Cannibals is a problem in which 3 missionaries and 3 cannibals want to cross from the left bank of a river to the right bank of the river. There is a boat on the left bank, but it only carries at most two people at a time (and can never cross with zero people). If cannibals ever outnumber missionaries on either bank, the cannibals will eat the missionaries.

A state can be represented by a triple, (*m c b*), where *m* is the number of missionaries on the left, *c* is the number of cannibals on the left, and *b* indicates whether the boat is on the left bank or right bank. For example, the initial state is (3 3 L) and the goal state is (0 0 R).

Operators are:

- MM: 2 missionaries cross the river
- CC: 2 cannibals cross the river
- MC: 1 missionary and 1 cannibal cross the river
- M: 1 missionary crosses the river
- C: 1 cannibal crosses the river

Draw a diagram showing all the legal states and transitions from states corresponding to all **legal** operations. See Figure 3.3 in Russell & Norvig (p. 65) for an example of what your diagram should look like.

2. Answer the following question.

Computers are able to solve this problem easily. Why do humans have a hard time with this problem at first? We know that humans and computers think differently than each other. What can we infer from the state space in problem 1 about why this problem seems hard for humans.

3. Breadth-first search

Solve the Missionaries and Cannibals problem by implementing the BFS algorithm given in class. Implement the BFS algorithm to show the open list and closed list at every iteration of the algorithm until the goal is visited. Use the format below. I have given the first two iterations.

Open: (3 3 L)
Closed: nil

Open: (3 1 R), (2 2 R), (3 2 R)
Closed: (3 3 L)

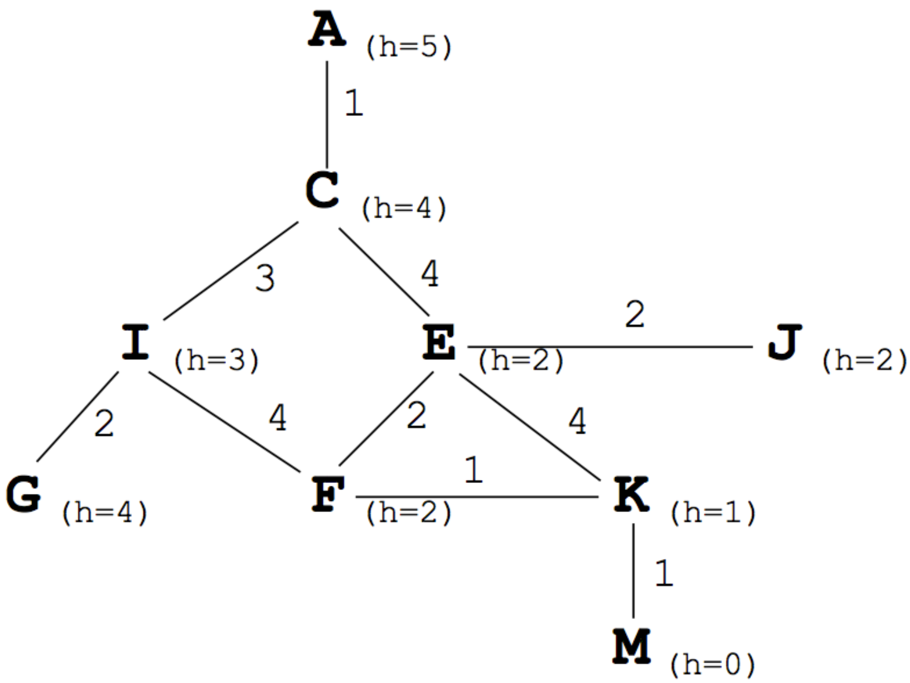
Open:
Closed:

4. Depth-first search

Same as problem 2, but use the **DFS** algorithm given in class.

CS 3600 – Introduction to Artificial Intelligence

Practice Search problem

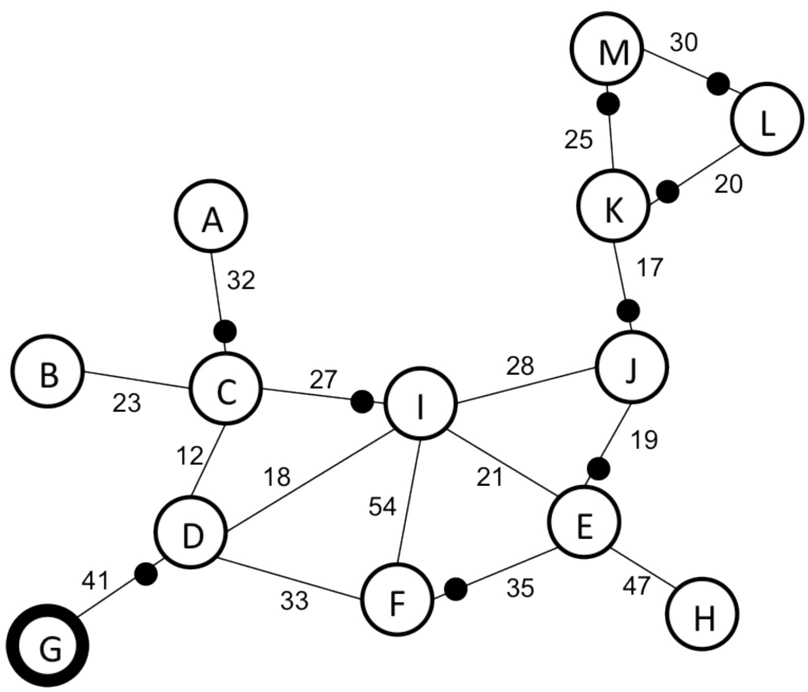


Consider the graph given above. It shows the actual distances between cities on a map. Each city is labeled with an estimate of the distance of the city from city M. Each path from a city to its neighbor is labeled with the length of the path. Consider an informed search for a shortest path from city A to city M using these estimates as heuristic function. Show how a version of A* would find this path. Assume that A* never expands the same state more than once and that it breaks ties toward cities with names earlier in the alphabet (A before B, B before C, and so on). Show the resulting search tree, including the f-value, g-value, and h-value of each node. Also clearly indicate the order in which the nodes are expanded (= label the node expanded first with 1, the node expanded next with 2, and so on - do not forget to label those nodes whose expansion only generates states that have already been expanded).

CS 3600 – Introduction to AI

Practice Search Problem

Consider the search space diagrammed below. Action costs are given next to each arc. Use "E" as the initial state and "G" as the goal state. The successor function for a node starts with the arc with the black dot next to it first, and the generates successors in a counter-clockwise fashion. For example, Successors(E) = {J, I, F, H} and Successors(J) = {K, I, E}.



1. What order will nodes be visited using the **breadth-first** algorithm? Ignore action costs. Give the open list and closed list after each iteration. List the visit order and the final solution.

2. What order will nodes be visited using the **depth-first** algorithm? Ignore action costs. Give the open list and closed list after each iteration. List the visit order and the final solution.

3. What order will nodes be visited using the **uniform cost search** algorithm? Give the open list and closed list after each iteration. List the visit order and the final solution.

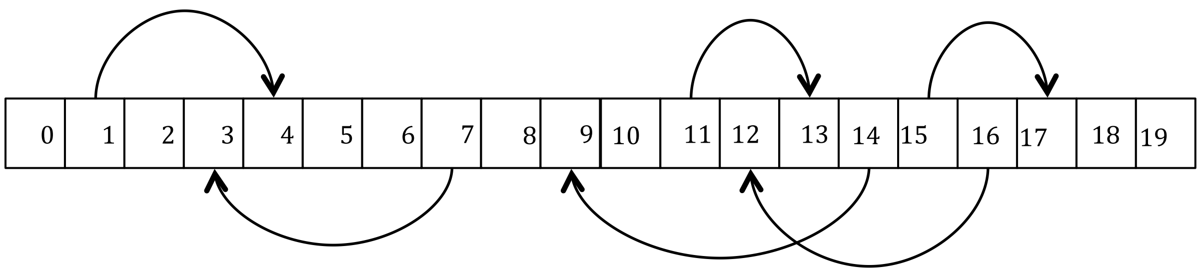
4. What order will nodes be visited using the best-first algorithm? Use the heuristic function given below. Give the open list and closed list after each iteration. List the visit order and the final solution.

h(A) = 65	h(D) = 41	h(H) = 100	h(K) = 50
h(B) = 25	h(E) = 85	h(I) = 55	h(L) = 100
h(C) = 50	h(F) = 30	h(J) = 110	h(M) = 95

CS 3600 – Introduction to Artificial Intelligence

Adversarial Games with Dice

Consider the following board game, based loosely on chutes and ladders. Two players race from position 0 to position 19. The first person to land exactly on position 19 wins. If you land on any position with an arc emanating, you automatically leap to the position at which the arc terminates. For example, if you stop on position 1, you are teleported to position 4.



On each turn, a player may perform one of the following actions:

- Move one position forward or backward.
- Roll one 4-sided die (1D4) and move the resultant number of positions forward or backward.
- Roll two 4-sided dice (2D4) and move the resultant summed number of positions forward or backward.

Problems:

1. Design a state representation. What is the initial state?

2. Write the pseudocode for a successor function. What kinds of states are there? What does each ply in an adversarial search tree represent?

3. Write the pseudocode for a terminal function.

4. Write the pseudocode for a utility function.

5. Draw out a portion of the adversarial search tree. Show enough ply for at least two turns.

6. Suppose the board is very long and it is impractical to reach terminal states each time. Design a cut off function and evaluation function that returns the value of intermediate, non-terminal states.

CS 3600 Introduction to AI

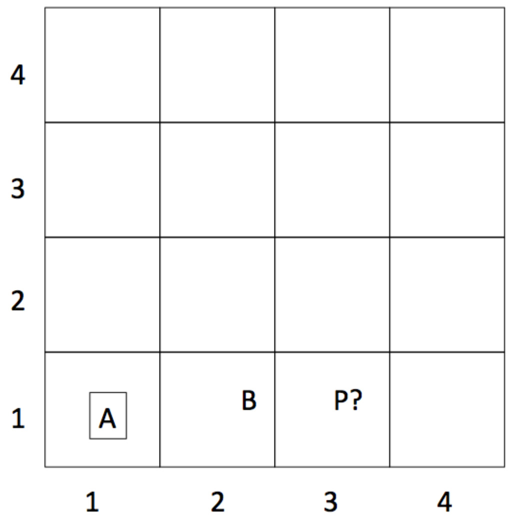
Constraint Satisfaction Problem

Solve the cryptarithmetic problem in Figure 6.2 of the Russell & Norvig textbook by hand, using the strategy of back-tracking with forward checking and the MRV and least-constraining value heuristics.

CS 3600 – Introduction to AI

Resolution

Consider the Wumpus world problem below. The world is a 4 x 4 grid with the agent in (1, 1). The Agent's knowledge base is given, where Bij means breeze in location (i, j), and Pij means pit in location (i, j).



$\neg P11$
$\neg P21$
$\neg P12$
$\neg B11$
$\neg B12$
$B21$
$B11 \leftrightarrow (P12 \vee P21)$
$B21 \leftrightarrow (P11 \vee P22 \vee P31)$
$B12 \leftrightarrow (P11 \vee P22 \vee P13)$

The Agent wants to know if there is a Pit in location (3, 1), i.e., P31.

Use Resolution to determine whether P31 is entailed by the knowledge base.

1. Convert the formulae in the knowledge base to conjunctive normal form.

2. Negate the conclusion and convert the conclusion to conjunctive normal form.

3. Draw out the resolution search space. Use the following heuristic:

Prefer clauses derived from the conclusion from smallest to largest (# of literals).
Next prefer clauses not derived from the conclusion, from smallest to largest.
Break ties in alpha-numeric order, e.g., B11, B12, B13, …, B21, B22, …, P11, P12, …