### Course: CS420 - Artificial Intelligence

01 – Search strategies for single agents

**Question 1.** *Missionaries and Cannibals.* Three missionaries and three cannibals are on one side of a river, along with a boat that can hold one or two people. Find a way to get everyone to the other side without ever leaving a group of missionaries in one place outnumbered by the cannibals in that place.

Consider a state as a tuple of values. Which values should be included in the tuple?

There are many possibilities. One example is: Represent the missionaries by M and the cannibals by C. Let the boat be B. Each state can be represented by the items on each side, e.g., Side1{M, M, C, C}, Side2{M, C, B}.

From this point, answer the following questions according to the definition of a state above.

Define the initial state and the goal state.

Initial state: Side1{M, M, M, C, C, C, B}, Side2{}

Goal state: Side1{}, Side2{M, M, M, C, C, C, B}

Define the successor function in this representation.

A set of missionaries and/or cannibals (call them Move) can be moved from Sidea to Sideb if:

* The boat is on Sidea.
* The set Move consists of 1 or 2 people that are on Sidea.
* The number of missionaries in the set formed by subtracting Move from Sidea is 0 or it is greater than or equal to the number of cannibals.
* The number of missionaries in the set formed by adding Move to Sideb is 0 or it is greater than or equal to the number of cannibals.

What is the path cost in your successor function?

Each move has a unit cost.

What is the maximum number of states in the state space, i.e., including illegal states?

What is the total number of reachable states?

There are 7 factors, and thus the maximum number of states is 2^7 = 128 states.

However, there are only 16 reachable states.

Side1{M, M, M, C, C, C, B}, Side2{}

Side1{}, Side2{M, M, M, C, C, C, B}

Side1{M, M, M, C, C, B}, Side2{C}

Side1{M, M, M, C, C}, Side2{C, B}

Side1{M, M, M, C, B}, Side2{C, C}

Side1{M, M, M, C}, Side2{C, C, B}

Side1{M, M, C, C, B}, Side2{M, C}

Side1{M, M, C, C}, Side2{M, C, B}

Side1{M, C, B}, Side2{M, M, C, C}

Side1{M, C}, Side2{M, M, C, C, B}

Side1{C, C, C, B}, Side2{M, M, M}

Side1{C}, Side2{M, M, M, C, C, B}

Side1{C, C, B}, Side2{M, M, M, C}

Side1{C, C, B}, Side2{M, M, M, C}

Side1{M, M, M}, Side2{C, C, C, B}

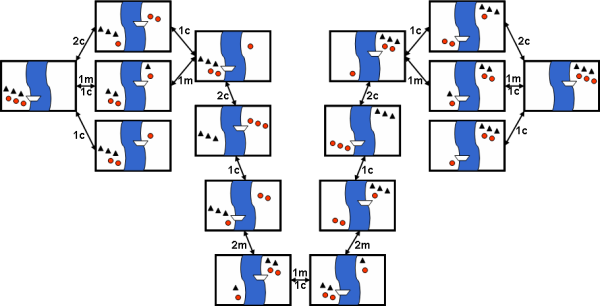
Side1{C, B}, Side2{M, M, M, C, C}

The last one is only reachable through the goal state, but it is still technically reachable (e.g.,if

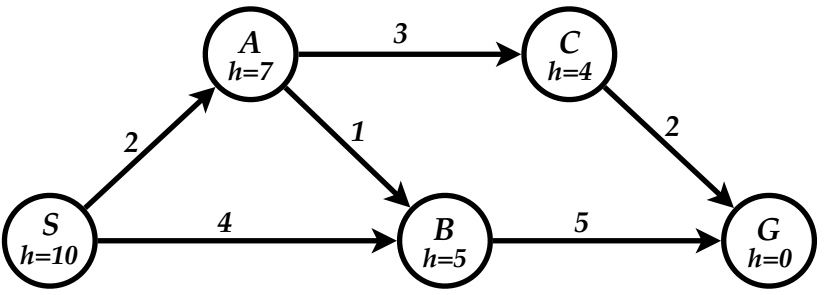
you are just exploring the state space instead of searching for a goal).

Draw a diagram of the complete state space.

A red circle represents a cannibal, a black triangle for a missionary, and the only white trapezoid denotes the boat.



**Question 2.** Consider the following graph, in which S and G are the initial and goal states, respectively. The heuristic values are shown under the vertices’ names, while path costs are shown on every edge.



For each of the search strategies listed below,

(a) list, in order, the states expanded,

(b) list, in order, the states included in the found path, and

(c) show the final content of the frontier (recall that a state is expanded when it is removed from the frontier)

**When all else is equal, nodes should be expanded in alphabetical order.**

1. Breadth-first search (BFS) (shown as an example)

List of expanded nodes: S A B

Path found: S B G

Frontier = { C }

1. Uniform-cost search (UCS)

List of expanded nodes: S A B C **G** (must have G)

Path found: S A C G

Frontier = { }

1. Depth-first search (DFS) (Avoid loops by remembering nodes on the current path).

List of expanded nodes: S A B

Path found: S A B G

1. Iterative deepening search (IDS)

List of expanded nodes for each limit: { S } { S A B } { S A B C B }

Path found: S B G

1. Greedy best first search (GBFS)

List of expanded nodes: S B

Path found: S B G

Frontier = { A }

1. A\* search

List of expanded nodes: S A B **G** (must have G)

Path found: S A B G

Frontier = { C }

1. Is the given heuristic admissible?

No. The heuristic values at nodes S, A, and C are overestimated.

1. Is the given heuristic consistent?

No. The given heuristic is inadmissible and hence inconsistent.

**Question 3.** The 4-queens problem requires you to placethe four queens on a 4×4 chessboard such that no queen attacks another queen. (A queen attacks any piece in the same row or column or diagonal). Here are some important facts:

* The states are any configurations where **all** the queens are on the board, one per column.
* The ***moveset*** includes all possible states generated by moving a single queen to another square in the same column. The function to obtain these states is called the ***successor*** function.
* The heuristic function ***h(state)*** is the number of **attacking** pairs of queens.

1. How many states are there in total? Explain your answer.

44 = 256 states. Each queen can be in any of the four squares in its own column.

1. For each state, how many successor states are there in the moveset? Explain your answer

3 x 4 = 12 states. Each queen can move to any of the rest three squares in its column.

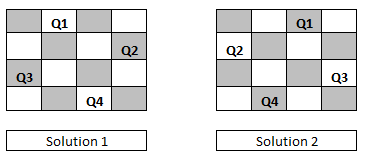
Only one queen is moved in a successor

|  |  |
| --- | --- |
| 1. What value will the heuristic function ***h*(*state*)** return for state *S* shown aside? Explain your answer. | A chess board with black crowns  Description automatically generated |

5. The list of pair of queens attacking each other is {Q1-Q2, Q1-Q3, Q1-Q4, Q2-Q3, Q2-Q4}

1. Apply steepest ascent hill climbing to find a solution for the given problem, given the initial state S shown above. Draw the search tree with root node S, and, for each node on the tree, write down its ***h*( )** value.

There are two solutions for the 4-queen problem.



Students can draw the tree by themselves. The following figures demonstrates successors in the first two levels.

However, the steepest-ascent hill climbing cannot solve this 4-queens instance without using sideway moves. As shown in the figures, the best successor in level 2 has h = 1 and we cannot attain h = 0 within a single move.

