PRINCIPLES OF COGNITIVE ROBOTICS
HOMEWORK ASSIGNMENT 2

ALEXANDER MORIARTY

Question 1:

Definitions in own words.

a) *State:* One possible situation in which an agent can be.
b) *State Space:* The set of all States in a graph; with possible actions represented by edges connecting one state to the next state.
c) *Search Tree:* A tree where the root node is the starting state and the leaf nodes are reachable states, connected by actions resulting in transition from the parent node to the child node.
d) *Search Node:* A node in the search tree.
e) *Goal:* The state which the agent is trying to arrive.
f) *Action:* Something the agent can do, that may or may not result in a change in state.
g) *Successor Function:* A function which returns a set of resulting state - taken action pairs. Representing the outcome states of available actions.
h) *Branching Factor:* The branching factor of a tree is the average number of children each non-leaf has.

Question 2:

Explain why problem formulation must follow goal formulation:

The goal formulation must occur before problem formulation to allow for efficient abstraction of the problem. If it is known what is required in the goal state, then unessential information need not be carried through the problem states. E.g. In defining a navigation for a road trip, states for driving with or without the radio on are not required; the radio (likely) has no effect the agent reaching its goal state.

Question 3:

Does a finite state space always lead to a finite search tree? How about a finite state space that is a tree? Can you be more precise about what types of state spaces always lead to finite search trees?

A graph with loops, and a search algorithm which does not track if all edges from a node have been explored, could result in an infinite search space. With loop detection, and properly implemented search algorithms and tree data structures can detect and avoid this (pre[], post[]). Trees, or in general, Directed Acyclic Graphs (DAGs), will always lead to finite search trees.

Date: April 16, 2013.

Question 4:

The missionaries and cannibals problem is usually stated as follows. 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 every- one to the other side, without ever leaving a group of missionaries in one place outnumbered by the cannibals in that place.

a) *Formulate the problem precisely, making only those distinctions necessary to ensure a valid solution. Draw a diagram of the complete state space.*

It is necessary to keep for each side of the river the number of monks, cannibals and boats (the boat in this case is a boolean, 0 or 1, but extended versions of this problem exist). Thus six values define a state. There are a number of actions available; however some result in invalid states, where cannibals outnumber monks. The actions can be encompassed in 3 integer values, representing the change in number of monks, cannibals or boats from one site of the river to the other (with the sign indicating direction.)

In the following diagram, the direction of the transition has been ignored for a cleaner graph.

Note: Having error with graphviz, including image and last working version of graph. State spaces accord to those in image

b) *Implement and solve the problem optimally using an appropriate search algorithm. Is it a good idea to check for repeated states?*

Yes. Without checking for repeated states it is very easy to initially or finally get stuck in loops.

c) *Why do you think people have a hard time solving this puzzle, given that the state space is so simple?*

Because most people take a greedy approach and do not think to bring cannibals or monks back across the river. To many this is a step away from the solution; analogous to being stuck in a local minima while performing a gradient decent to find a global minima.