Name: Shan Ying Yang Student No.: 301305759 Date: February 2, 2022

CMPT 310 Assignment 1 Part A

Question 1 (2 points)

Please write about 'The Chinese Argument' briefly.

"The Chinese Argument" refers John R. Searle's thought experiment on experimenting whether the machine, or Strong AI as he referred to, can have a mind or consciousness like human do. Searle claims that a machine should not be considered to have a "mind" if it cannot "understand", therefore claiming that strong AI is false. Strong AI, on the other hand, refers to a programmed computer having a mind that is the same as human minds if they have the right input and output. The experiment was constructed by having one person (an English speaker) imagining himself locked in a room with a batch of Chinese input, a second batch of Chinese scripts, and a set of rules and constraints for Chinese but written in English. Then he asked the person to use the given rules and constraints to construct more Chinese outputs according. For example, given input "How are you" in Chinese, the person supposedly can use the rules to respond in Chinese accordingly. However, Searle claims that the person would not understand the real meanings of the Chinese characters he constructer if he does not understand the meaning of them, since all output were based on the already given rules. Similarly, for a machine that does not understand the output it produces, one may not call it to have a mind like human's mind.

Question 2 (2 points)

Give a complete problem formulation for the below problems. Choose a formulation that is precise enough to be implemented.

(A) A container ship is in port, loaded high with containers. There 13 rows of containers, each 13 containers wide and 5 containers tall. You control a crane that can move to any location above the ship, pick up the container under it, and move it onto the dock. You want the ship unloaded.

Initial state: all the containers are stacked on the ship
Actions: move to a location near the stacked containers, pick up the container, move onto dock, and then unload the container onto the dock.
Transition model: the crane moves to a location, and a container is picked up and unloaded
Goal test: unload all containers from ship
Cost function: number of actions

(B) There are six glass boxes in a row, each with a lock. Each of the first five boxes holds a key unlocking the next box in line; the last box holds a banana. You have the key to the first box, and you want the banana.

Initial state: 5 locked up glass boxes each holds a key, and last glass box holds a banana
Actions: open an unopened box and obtain object from box
Transition model: an unopened box is opened, or the object inside the box is obtained already
Goal test: get the banana from the last box
Cost function: number of actions

Question 3 (2 points)

Describe a state space in which iterative deepening search performs much worse than depth-first search (for example, $O(n^2)$ vs. $O(n)$).

Iterative deepening search visits the child node of the root, and repeats visiting the child node of the previously visited node as depth increments. For example, a state space S = A → B → C → D → E, to find node E with root node A, IDS would explore the nodes as follows: A AB ABC ABCD ABCDE.

So for state spaces with single branching factor (i.e. A → B → C → D → E), with goal node at depth n, DFS finds goal with n steps (time complexity of O(n)), and IDS finds goal node with $\sum_{i=1}^{n} n$ steps ($1 + 2 + \cdots + n$ steps, with time complexity of $O(n2)$).

Question 4 (3 points)

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?

No, a finite state space does not always lead to a finite search tree. A finite state space can lead to an infinite search tree the state space contains any circular paths.
A finite state space that is a tree will lead to a finite search tree because it does not contain circular path, so the search will not go into any loops. Thus, a finite state space that is a tree will always lead to a finite search tree. For instance, a directed acyclic graph has no cycles and will have a finite search tree with the state space is also finite.

Question 5 (3 points)

Suppose two friends live in different cities on a map, such as the Romania map. On every turn, we can simultaneously move each friend to a neighboring city on the map. The amount of time

needed to move from city i to neighbor j is equal to the road distance $d(i,j)$ between the cities, but on each turn the friend that arrives first must wait until the other one arrives (and calls the first on his/her cell phone) before the next turn can begin. We want the two friends to meet as quickly as possible.

(a) Write a detailed formulation for this search problem. (You will find it helpful to define some formal notation here.)

State space: the possible city pairs (i,j) that represent the positions of the two friends.
Actions/transition model: move person 1 to x and person 2 to y, where the successor pairs will be the adjacent pairs of cities and states → $Adj(x,i), Adj(y,j)$
Goal: to have both persons placed in the same city $i, (i,i)$.
Cost function: the arc cost will be the cost of time to move from city i to neighbor j.
$max(d(i,x), d(j,y))$

(b) Let $D(i,j)$ be the straight-line distance between cities i and j. Which of the following heuristic functions are admissible? (i) $D(i,j)$; (ii) $2 * D(i,j)$; (iii) $D(i,j)/2$.

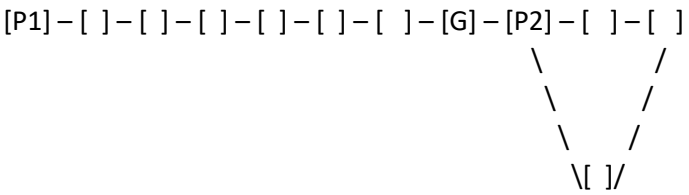
$D(i,j)/2$ is the admissible heuristic function, because the ideal case would be to have both friends move toward each other simultaneously (same pace/equal timestep), which reduces the distance between them at each time step.

(c) Are there completely connected maps for which no solution exists?

Yes, there is completely connected map that no solution exists. If the two friends happen to be on cities that are neighboring cities, they will never be able to meet each other as they continuously swap cities at each time step. So, for a map with two connected nodes the two friends will never be able to meet.

(d) Are there maps in which all solutions require one friend to visit the same city twice?

Since we must move a friend to a neighboring city on the map at each move, the following map requires one friend to visit the same city twice:



For the map above, P2 will have to loop around the triangular path and then reach goal cell G, which visits P2's initial position twice.

Name: Shan Ying Yang Student No.: 301305759 Date: February 2, 2022

Question 6 (3 points)

The heuristic path algorithm (Pohl, 1977) is a best-first search in which the evaluation function is $f(n) = (2 - w)g(n) + wh(n)$. For what values of w is this complete? For what values is it optimal, assuming that h is admissible? What kind of search does this perform for $w = 0, w = 1, and\ w = 2$?

For $0 \leq w < 2$ the evaluation function is complete.

For $w = 0$: $f(n) = 2g(n)$, the function performs uniform-cost search.

For $w = 1$: $f(n = g(n) + h(n)$, the function performs A* search.

For $w = 2$: $f(n) = 2h(n)$, the function performs greedy best-first search.

Since scaling the $g(n)$ has no effect on the ordering of the nodes, for $0 \leq w \leq 1$, the heuristic path algorithm is optimal because it is always less than $h(n)$, making the function heuristic admissible.