Name: Shashank Gowdagiri

USC ID: 1956-0195-02

User login: sgowdagi@usc.edu

Question 3 Answers:

The distance for calculation is the Manhattan distance for each node. It gives the sum of the differences of the coordinates of the current node from the start node, and its distance till the end node.

The Heuristic is in time, which is the sum of the path taken (i.e distance already travelled) divided by the speed (which is calculated as given)

(Current[x]-goal[x]) + (current[y]-goal[y])/current speed

It is consistent because the f(n) value is always consistently increasing, and the order of expansion is directly dependent on the total heuristic value calculated. (lowest first)

The second consistent heuristic would be the straight line distance. In this case as well, there are no path where the f(n) value is decreasing. It is consistent.

Due to the differing run times of the heuristic dependent algorithms, neither heuristic is dominant.

Question 6 Table with values

		Runtime(ms)	**Number of Iterations**	**Path Length**	**Solution cost (time)**
Input1.txt	**BFS**	**180**	**401**	**93**	**46.157**
	DFS	**96**	**179**	**125**	**115.22**
	A*(Heuristic 1)	**189**	**379**	**93**	**46.100**
	A* (Heuristic 2)	**200**	**389**	**93**	**46**
	Beam (k=5)	**148**	**116**	**56**	**34**
	Beam (k=10)	**160**	**283**	**53**	**39**

	Beam(k=50)	**183**	**379**	**93**	**39**
Input 2.txt	**BFS**	**102**	**546**	**49**	**24.078**
	DFS	**104**	**50**	**49**	**25.184**
	A*(Heuristic 1)	**280**	**104**	**47**	**24.0**
	A* (Heuristic 2)	**270**	**230**	**47**	**24.0**
	Beam (k=5)	**135**	**379**	**49**	
	Beam (k=10)	**167**	**458**	**47**	
	Beam(k=50)	**198**	**104**	**47**	
Input3.txt	**BFS**	**231**	**2969**	**326**	**353.12286**
	DFS	**190**	**Path not found**		
	A*(Heuristic 1)	**200**	**2469**	**324**	**162.5**
	A* (Heuristic 2)	**190**	**2450**	**324**	**167**
	Beam (k=5)	**100**	**Path not found**		
	Beam (k=10)	**98**	**Path not found**		
	Beam(k=50)	**311**	**2469**	**324**	

In terms of comparing the algorithms for time and space complexity,

BFS – It is complete when the branching factor is finite. With constant step cost, it is also optimal. Time and space complexity reduces if , instead of just visiting the node, we could check it for the goal node instead of waiting to expand.

DFS – It is not optimal. IT is complete only when the nodes expanded are finite. The time and space complexity of DFS is O(b(power) m).

A* - Depending on the heuristic used for A*, the distance and time towards the goal changes. It is therefore, very important to choose a right heuristic. An overestimated heuristic, (which I encountered while testing and debugging) could lead to disastrous run time complexities.

Beam – The nodes expanded from every node, (in this case the maze cell) , prune the options of finding the goal cell. If only a certain number of states are allowed to be expanded, then there is a very real possibility of losing the goal cell multiple times. The run times increases as the value of the beam decreases, since the odds of finding the goal are reduced.